**1. INTRODUCTION**

Virtually all home computers today use traditional interfacing i.e., with keyboard and mouse.New home pc’s are also being developed and introduced which include touch screens and voice support which increases the level of interactivity but increases the cost significantly. Wouldn’t it be cool if you could just point at screen, wave your hands, make gestures, give voice commands and watch your actions controlling your pc or workstation and that too which is cost effective? Definitely it would be super cool. Our Project Aims Exactly at this…It Aims at Creating a Hybrid Gesture Induced Computer Interface at Low Cost. Also Our Project can convert any flat surface into low cost touchscreen To form the Gesture Induced Interface…a number for devices, software’s are used which are made from ground up or ready-made. In our Project we will be (tentatively) host of gaming controllers and computer peripherals which includes Wii’s Wiimote, Xbox 360’s Kinect. Microphone, webcam and more Programmer Software for the game controllers: GlovePIE. The front end will be Visual Studio and Swing mix.

**WIIMOTE:**

Our Project Deals with controlling the computer by implementing the concepts of Hand Motion movements, Universal Touch screens, Finger Tracking, Gesture Tracking All the Above Concepts are Inbuilt into the Wiimote. So Using Capabilities of Wiimote and Nunchuk (IR Tracking, Motion Capture, Gesture recognition and Buttons!!) we can control the pc

**XBOX KINECT:**

This game controller has a 2D camera a 3D depth camera and a heat camera and has supported SDKs which provide complete body skeleton tracking. This provides excellent base for implementing different applications and ease of coding.

**2. AIMS AND OBJECTIVE**

* To provide a comprehensive hands free interaction environment.
* To free the user of confined and traditional workspace and work format of interacting with computers.
* To provide the above stated environment at low cost compared to other commercial products.
* To enforce portability, intuitiveness and ease of use.
* Apart from the above mentioned objectives, the popularity and simple object oriented and linear execution flow like programming language, future expansion of this project becomes simple as this project follows modular programming and additional features can be added easily without refactoring the entire code. This gives third party developers to freely incorporate this project and low maintenance and deployment costs.

**3. LITERATURE REVIEW**

With the massive influx of computers in society, human computer interaction, or HCI, has become an increasingly important part of our daily lives. It is widely believed that as the computing, communication, and display technologies progress even further, the existing HCI techniques may become a bottleneck in the effective utilization of the available information flow. For example, the most popular mode of HCI is based on simple mechanical devices— keyboards and mice. These devices have grown to be familiar but inherently limit the speed and naturalness with which we can interact with the computer. This limitation has become even more apparent with the emergence of novel display technology such as virtual reality. Thus in recent years there has been a tremendous push in research toward novel devices and techniques that will address this HCI bottleneck. One long-term attempt in HCI has been to migrate the “natural” means that humans employ to communicate with each other into HCI. With this motivation automatic speech recognition has been a topic of research for decades. Tremendous progress has been made in speech recognition, and several commercially successful speech interfaces have been deployed. However, it has only been in recent years that there has been an increased interest in trying to introduce other human-to-human communication modalities into HCI. This includes a class of techniques based on the movement of the human arm and hand, or hand gestures. Human hand gestures are a means of non-verbal interaction among people. They range from simple actions of using our hand to point at and move objects around to the more complex ones that express our feelings and allow us to communicate with others. To exploit the use of gestures in HCI it is necessary to provide the means by which they can be interpreted by computers. The HCI interpretation of gestures requires that dynamic and/or static configurations of the human hand, arm, and even other parts of the human body, be measurable by the machine. First attempts to solve this problem resulted in mechanical devices that directly measure hand and/or arm joint angles and spatial position. This group is best represented by the so-called glove-based devices. Glove-based gestural interfaces require the user to wear a cumbersome device, and generally carry a load of cables that connect the device to a computer. This hinders the ease and naturalness with which the user can interact with the computer controlled environment. Even though the use of such specific devices may be justified by a highly specialized application domain, for example simulation of surgery in a virtual reality environment, the “everyday” user will certainly be deterred by such cumbersome interface tools. This has spawned active research toward more “natural” HCI techniques.

**4. EXISTING SYSTEM**

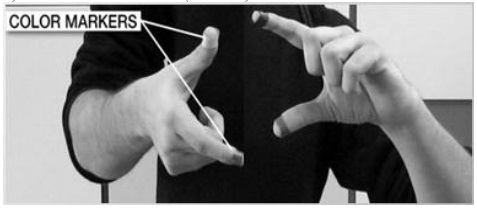
**4.1 CAMERA:**

A webcam captures and recognizes an object in view and tracks the user’s hand gestures using computer-vision based techniques. It sends the data to the computer. The camera, in a sense, acts as a digital eye, seeing what the user sees. It also tracks the movements of the thumbs and index fingers of both of the user's hands. The camera recognizes objects around you instantly.

**4.2 SINGLE CAMERA SYSTEM:**

This system would provide considerably less information about the hand. Some features (such as the finger against a background of skin in the example above) would be very hard to distinguish since no depth information would be recoverable. Essentially only “silhouette” (Detection of all skin within the hand without any feature detection information could be accurately extracted. The silhouette data would be relatively noise free (given a background sufficiently distinguishable from the hand) and would require considerably less processor time to compute than either multiple camera system. It is possible to detect a large subset of gestures using silhouette information alone and the single camera system is less noisy, expensive and processor hungry. Although the system exhibits more ambiguity Stereographic system and multiple two dimensional view system this disadvantage is more than outweighed by the advantages mentioned above. Therefore, it was decided to use the single camera system.

**4.3 COLOUR STICKERS (R G B):**



**Fig 4.1 Colour Stickers**

It is at the tip of the user’s fingers. Marking the user’s fingers with red, green, and blue tape helps the webcam recognize gestures. The movements and arrangements of these Markers are interpreted into gestures that act as interaction instructions for the projected application interfaces.

**5. PROBLEM DEFINITION**

This project will design and build a man-machine interface using a video camera to interpret the hand gestures (plus others for additional keyboard and mouse control).The keyboard and mouse are currently the main interfaces between man and computer. Humans communicate mainly by vision and sound, therefore, a man-machine interface would be more intuitive if it made greater use of vision and audio recognition. Another advantage is that the user not only can communicate from a distance, but need have no physical contact with the computer. However, unlike audio commands, a visual system would be preferable in noisy environments or in situations where sound would cause a disturbance. The visual system chosen was the recognition of hand gestures. The amount of computation required to process hand gestures is much greater than that of the mechanical devices; however standard desktop computers are now quick enough to make this project hand gesture recognition using computer vision a viable proposition. In this project will be using next generation features i.e. hand gestures for controlling computer functions like navigation, zoom in/out, changing of slides.

.

**6. SCOPE OF PROJECT**

This project mainly focuses on the universal scenarios of interacting with a computer. This includes the scenarios of interaction dictated by the operating system design and in this case, windows 7 which largely depend on traditional keyboard mouse combination and, to an extent, touch screens. However, as functional as the keyboard and mouse may be, this day and age demands a much more intuitive and fun way of interaction. This system emulates all the functions of the traditional keyboard mouse combination and adds over it, various gestures and speech recognition using the game controllers and microphone to perform complex and compound set of actions such as taking a screen shot, opening an application, zoom in, play pause media etc which would otherwise require the user, in case of taking a screenshot, to manually take the screen shot, save it and then view it.Furthermore, complex scripts can be written for interacting with a particular app for example an FPS game in which the user can move around by leaning in the respective direction.

**7. SYSTEM REQUIREMENT SPECIFICATION**

**7.1 Purpose:**

Virtually all home computers today use traditional interfacing i.e., with keyboard and mouse. New home pc’s are also being developed and introduced which include touch screens and voice support which increases the level of interactivity but increases the cost significantly. We are designing a system that would do all above by just point at screen, wave your hands, make gestures, give voice commands and watch your actions controlling your pc or workstation and that too which is cost effective. Our project aims exactly at this. It aims at creating a hybrid gesture induced computer interface at low cost.

**7.2 Product prospective:**

The purpose of our system is to provide a comprehensive hands free interaction environment which can be free the user of confined and traditional workspace and help to interacting with computers Apart from the above mentioned objectives, the popularity and simple object oriented and linear execution flow like programming language, future expansion of this project becomes simple as this project follows modular programming and additional features can be added easily without refactoring the entire code. This gives third party developers to freely incorporate this project and low maintenance and deployment costs. This project mainly focuses on the universal scenarios of interacting with a computer. This includes the scenarios of interaction dictated by the operating system design and in this case, windows 7 which largely depend on traditional keyboard mouse combination and, to an extent, touch screens. However, as functional as the keyboard and mouse may be, this day and age demands a much more intuitive and fun way of interaction. This system emulates all the functions of the traditional keyboard mouse combination and adds over it, various gestures and speech recognition using the game controllers and microphone to perform complex and compound set of actions such as taking a screen shot, opening an application, zoom in, play pause media etc. which would otherwise require the user, in case of taking a screenshot, to manually take the screen shot, save it and then view it. Furthermore, complex scripts can be written for interacting with a particular app for example an FPS game in which the user can move around by leaning in the respective direction.

**7.3 Product function:**

This project we will design and build an interactive system using Kinect and Wii mote’s video camera to interpret the hand gestures. We will recognize the hand gesture using camera and then these hand gestures will be used for controlling computer functions like navigation, zoom in/out, changing of slides. We will also use gesture for mouse movements and convert computer screen into a virtual touch screen where users can interact with out computer.

**7.4 User characteristics:**

NA

**7.5 General constraints:**

The range of the system is dependent on Range of the Kinect and Wii Mote. The range is also affected by Bluetooth Dongle that is going to be used. The programing for this system will be done on Beta SDK. The resolution of camera on Kinect and Wii Mote is 1280 x 960 so it will not be that precise. All the operations need to be done by standing in line of sight of the camera. For kinect we need power adapter and Wii mote is battery operated.

**7.6 Assumptions and dependencies:**

In our project will use the Windows’s default speech synthesizer or a third party open source speech synthesizer written in java called sphinx

**7.7 External interface requirement:**

We are using Blue Soleil which is used to connect the wii mote and other Bluetooth peripherals to the computer.

**7.8 Functional requirement:**

Windows 7, Windows 8

32 bit (x86) or 64 bit (x64) processor

Dedicated USB 2.0 bus

2 GB RAM

Bluetooth dongle V2.0

Speakers and Dedicated Microphone

**7.9 Non-functional requirements:**

NA

**7.10 Designed constraints:**

NA

**7.11 Risk analysis:**

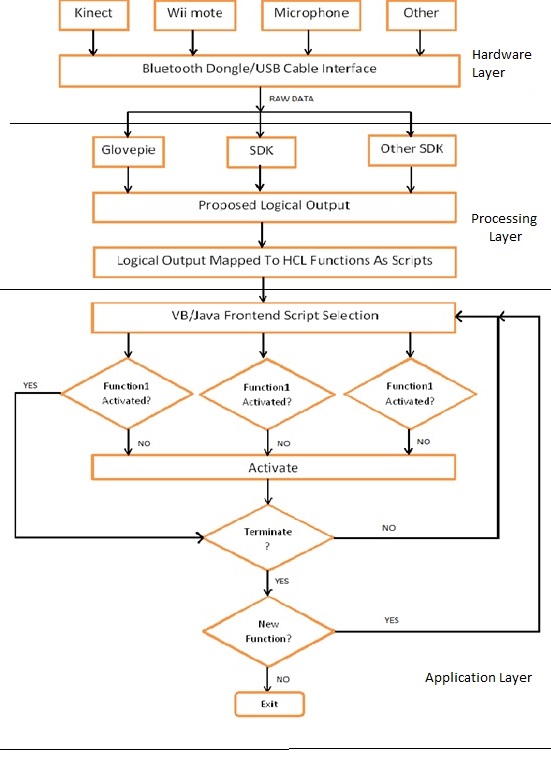
NA

**8. METHOGOLOGY**

* This project strictly follows Object Oriented Programming principles
* This is because of the inherent properties of the GlovePIE programmer which accesses every controller as an object and every programmable feature of it as its attribute
* Example: in order to access the led1 on the Wiimote,
* The code will be Wiimote1.led1 where Wiimote1 is the object of class Wiimote (This object is automatically declared and initialized) and led1 is the physical led light present on the Wiimote
* Another prominent feature of writing scripts in GlovePIE is linear programming.
* This is basically a programming technique, or more precisely, a programming design concept applied in the compiler or the run time environment where the lines of codes are executed from left to right, top to bottom. Once the bottom rightmost position is reached, the program again starts executing from top left most position
* Use of visual basic and java ensure a stricter object oriented design which has its advantages as widely known already.
* We follow strict modular programming principles (with a twist) for all functions and sub functions of our project. This means that the project is not developed as a whole. The entire vision and implementation is broken into atomic functions or modules. Each of these modules is coded, tested separately and then will be combined to form a single solution. The twist is that these modules are not actually combined to form a single solution as it may appear to the end user as all the back end complexity is hidden from the user this is also known as data abstraction. All the modules retain their atomicity as all the complex functions are implemented by calling the sub functions. This ensures that the modules do not get too big and ultimately suffer a performance hit as smaller modules execute faster due to the multi-threading support.

**9. SYSTEM ANALYSIS**

**9.1 FLOWCHART:**



**Fig 9.1 System Analysis (Flowchart)**

**9.2 UML DIAGRAMS:**

**9.2.1 Class Diagram:**



**Fig 9.2 Class Diagram**

**9.2.2 Use case Diagram:**



**Fig 9.3 Use case Diagram**

**9.2.3 Sequence Diagram:**



**Fig 9.4 Sequence Diagram**

**9.2.4 Collaboration Diagram:**



**Fig 9.5 Collaboration Diagram**

**9.2.5 State Diagram:**



**Fig 9.6 State Diagram**

**9.2.6 Activity Diagram:**



**Fig 9.7 Activity Diagram**

**10. SYSTEM ARCHITECTURE**

**10.1 HARDWARE LAYER:**

It includes all the hardware such as the Wiimote, Kinect, microphone, Bluetooth dongle, webcams etc.

These hardware provide raw data to the processing layer. This raw data is in “as is” digital form and indicate no application logic. Three prominent hardware are:

**10.1.1 Kinect:**



**Fig 10.1 Kinect**

Kinect builds on software technology developed internally by Rare, a subsidiary of Microsoft Game Studios owned by Microsoft, and on range camera technology by Israeli developer Prime Sense, which developed a system that can interpret specific gestures, making completely hands-free control of electronic devices possible by using an infrared projector and camera and a special microchip to track the movement of objects and individuals in three dimension. This 3D scanner system called light coding employs a variant of image-based 3D reconstruction.

**10.1.2 Wiimote:**

**Fig 10.2 Wiimote**

A main feature of the Wii Remote is its motion sensing capability, which allows the user to interact with and manipulate items on screen via gesture recognition and pointing  through the use of accelerometer and optical sensor (IR Tracking).

* It also includes 12 digital buttons
* The Wii Remote,sometimes unofficially nicknamed "**Wiimote**", is the primary controller for Nintendo's Wii console.
* A main feature of the Wii Remote is its motion sensing capability, which allows the user to interact with and manipulate items on screen via gesture recognition and pointing through the use of accelerometer and optical sensor technology.
* Another feature is its expandability through the use of attachments. The attachment bundled with the Wii console is the Nunchuk, which complements the Wii Remote by providing functions similar to those in gamepad controllers.
* Some other attachments include the Wii Classic Controller, Wii Zapper, and the Wii Wheel, originally used for Mario Kart.
* The controller was revealed at the Tokyo Game Show on October 14, 2005, with the name "Wii Remote" announced April 27, 2006. It has since received much attention due to its unique features and the contrast between it and typical gaming controllers. It has also gained significant attention from hackers using it to control non Wii-related devices through Wii homebrew.
* It was only a matter of time. Because Nintendo chose to use the Bluetooth standard for its Wii remote, enterprising developers have already discovered ways to use it on PCs. Early Windows, Mac, and Linux drivers are available that let your Wiimote act as a mouse (for gaming, presentations, etc.), or to control a PC game. Other uses are more imaginative, for example here's a video of someone using the Wiimote to compose music or sound effects.
* The Wiimote is quite a sophisticated device internally. According to Wikipedia and manufacturer information it contains an ADI ADXL330 3-axis iMEMS accelerometer:
* The ADXL330 is used to sense motion of the game player in three dimensions of freedom: forward-backward, left-right, and up-down. When the new controller is picked up and manipulated, it provides a quick element of interaction, sensing motion, depth and positioning dictated by the acceleration of the controller itself.
* If that weren't enough, the front of the Wiimote houses a tiny one-megapixel camera that constantly looks for the ten tiny IR lights in the Wii's sensor bar, and uses those to figure out exactly where you're pointing the remote. Unlike traditional light guns (which rely on special patterns on the display), this point could be on the screen or off. Feedback in the form of a rumble motor and a tiny speaker can make the immersion even more complete. More technical info can be found here.
* Imagine the possibilities such a rich controller can offer beyond gaming, such as remote robotic manipulation, 3d sculpting, database exploration, and more. Thanks to a standard wireless protocol, commodity pricing, and open source drivers, the technology in the Wiimote may herald a turning point in human-machine interfaces.



**Fig 10.3:** Johnny Lee's Wii Mote Whiteboard software used for light pen-type computer input

**10.1.2.1 Development:**

Development of a motion enabled controller began in 2001, coinciding with development of the Wii console. In that year, Nintendo licensed a number of motion-sensing patents from Gyration Inc., a company that produces wireless motion-sensing computer mice. Nintendo then commissioned Gyration Inc. to create a one-handed controller for it, which eventually developed the "Gyropod", a more traditional gamepad which allowed its right half to break away for motion-control. At this point, Gyration Inc. brought in a separate design firm Bridge Design to help pitch its concept to Nintendo. Under requirement to "roughly preserve the existing Game Cube button layout", it experimented with different forms "through sketches, models and interviewing various hardcore gamers". By "late 2004, early 2005", however, Nintendo had come up with the Wii Remote's less traditional "wand shape", and the design of the Nunchuk attachment. Nintendo had also decided upon using a motion sensor, infrared pointer, and the layout of the buttons, and by the end of 2005 the controller was ready for mass production.

**10.1.2.2 Third-party use:**

* Since the release of the Wii console, people have been exploring different new ways in which to use the Wii Remote. Many third-party applications are currently in development through Wii homebrew. One popular Windows program called GlovePIE allows the Wii Remote to be used on a personal computer to emulate a keyboard, mouse or joystick.
* Connecting the Wii Remote to a personal computer is done via a Bluetooth connection. The Bluetooth program BlueSoleil has been proven to successfully connect a Wii Remote to a PC. Still another program (like GlovePIE) is needed to utilize the Wii Remote's protocol and to use the data it offers.
* Connecting the Wii Remote to a personal computer is done via a Bluetooth connection. The Bluetooth program BlueSoleil has been proven to successfully connect a Wii Remote to a PC. Still another program (like GlovePIE) is needed to utilize the Wii Remote's protocol and to use the data it offers.
* The Wii Remote Bluetooth protocol can be implemented on other devices including cell phones, which often have poor usability with games. Two students have demonstrated this concept by creating a driver software that has the capability to connect the Wii Remote to a Symbian OS smartphone. The idea behind this driver is that a mobile phone with a TV-out port can replace the game console.
* Programmer Johnny Lee has posted video demos and sample code at his website related to the use of the Wii Remote for finger tracking, low-cost multipoint interactive whiteboards, and head tracking for desktop VR displays. This was the subject for his presentation at the prestigious TED conference, where he demonstrated several such applications.
* The Wii mote Project forum has become the discussion, support and sharing site for Lee's Wii Remote projects and other newer developments.
* Studies have also been conducted to use the Wii Remote as a practice method to fine-tune surgeons' hand motions. Utilizing Darwiin Remote, researchers at the University of Memphis have adapted the Wii Remote for data collection in cognitive psychology experiments.

## 10.1.2.3 Design:

The Wii Remote assumes a one-handed remote control-based design instead of the traditional gamepad controllers of previous gaming consoles. This was done to make motion sensitivity more intuitive, as a remote design is fitted perfectly for pointing, and in part to help the console appeal to a broader audience that includes non-gamers. The body of the Wii Remote measures 148 mm (5.83 in) long, 36.2 mm (1.43 in) wide, and 30.8 mm (1.21 in) thick. The Wii Remote model number is RVL-003, a reference to the project codename "Revolution". The controller communicates wirelessly with the console via short-range Bluetooth radio, with which it is possible to operate up to four controllers as far as 10 meters (approx. 30 ft) away from the console. However, to utilize pointer functionality, the Wii Remote must be used within five meters (approx. 16 ft.) of the Sensor Bar. The controller's symmetrical design allows it to be used in either hand.

At E3 2006, a few minor changes were made to the controller from the design presented at the Game Developer's Conference. The controller was made slightly longer, and a speaker was added to the face beneath the center row of buttons. The "B" button became more curved resembling a trigger. The "Start" and "Select" buttons were changed to plus "+" and minus "–", and the "b" and "a" buttons were changed to 1 and 2 to differentiate them from the "A" and "B" buttons. Also, the symbol on the "Home" button was changed from a blue dot to a shape resembling a home/house, the shape of the power button was made circular rather than rectangular, and the blue LEDs indicating player number are now labeled using small Braille-like raised dots instead of Arabic numerals, with "1" being "•", "2" being "••", "3" being "•••", and "4" being "••••". The Nintendo logo at the bottom of the controller face was replaced with the Wii logo. Also, the expansion port was redesigned, with expansion plugs featuring a smaller snap-on design.

The blue LEDs also show how much battery power remains on the Wii Remote. By pressing any button, besides the power button while the controller is not being used to play games, a certain number of the four blue LEDs will light up, showing the battery life: four of the LEDs flash when it is at, or near, full power. Three lights flash when it is at 75%, two lights when at 50%, and one light flashes when there is 25% or less power remaining. Similarities have been noted between the Wii Remote and an early Dreamcast controller prototype.

[](http://en.wikipedia.org/wiki/File:Wii_Remote.jpg)

**Fig 10.4 Wiimote**

Demo Wii Remote shown at a Nintendo event at the Hotel Puerta America. In the Red Steel trailer shown at E3 2006, the Wii Remote featured a smaller circular shaped image sensor, as opposed to the larger opaque IR filters shown on other versions. In the initial teaser video that revealed the controller at Tokyo Game Show 2005, the 1 and 2 buttons were labelled X and Y, respectively.

**10.1.3 Bluetooth Dongle:**

Bluetooth is a proprietary open wireless technology standard for exchanging data over short distances (using short wavelength radio transmissions) from fixed and mobile devices, creating personal area networks (PANs) with high levels of security. Created by telecoms vendor Ericsson in 1994, it was originally conceived as a wireless alternative to RS-232 data cables. It can connect several devices, overcoming problems of synchronization.

Bluetooth is managed by the Bluetooth Special Interest Group, which has more than 14,000 member companies in the areas of telecommunication, computing, networking, and consumer electronics. The SIG oversees the development of the specification, manages the qualification program, and protects the trademarks. To be marketed as a Bluetooth device, it must be qualified to standards defined by the SIG. A network of patents are required to implement the technology and are only licensed to those qualifying devices; thus the protocol, whilst open, may be regarded as proprietary.



**Fig 10.5 Bluetooth Dongle**

**10.1.3.1 Bluetooth v2.0 + EDR:**

This version of the Bluetooth Core Specification was released in 2004 and is backward compatible with the previous version 1.2. The main difference is the introduction of an Enhanced Data Rate (EDR) for faster data transfer. The nominal rate of EDR is about 3 Mbit/s, although the practical data transfer rate is 2.1 Mbit/s. EDR uses a combination of GFSK and Phase Shift Keying modulation (PSK) with two variants, π/4-DQPSK and 8DPSK. EDR can provide a lower power consumption through a reduced duty cycle.

The specification is published as "Bluetooth v2.0 + EDR" which implies that EDR is an optional feature. Aside from EDR, there are other minor improvements to the 2.0 specification, and products may claim compliance to "Bluetooth v2.0" without supporting the higher data rate. At least one commercial device states "Bluetooth v2.0 without EDR" on its data sheet.

**Bluetooth v2.1 + EDR:**

Bluetooth Core Specification Version 2.1 + EDR is fully backward compatible with 1.2, and were adopted by the Bluetooth SIG on July 26, 2007.

The headline feature of 2.1 is secure simple pairing (SSP): this improves the pairing experience for Bluetooth devices, while increasing the use and strength of security. See the section on Pairing below for more details. It offers APIs for raw sensor stream and human motion tracking. The SDK allows developers using C++, C# and Visual Basic via Visual Studio 2012 to access the capabilities of Kinect. The SDK allows raw sensor streams to be accessed that are generated by the Kinect sensor. The SDK supports skeletal tracking, advanced audio capabilities and it comes with sample code and documents. The SDK installer is under 100 MB and easy to install.

**10.2 PROCESSING LAYER:**

**10.2.1 Kinect sdk:**

It offers APIs for raw sensor stream and human motion tracking. The SDK allows developers using C++, C# and Visual Basic via Visual Studio 2012 to access the capabilities of Kinect. The SDK allows raw sensor streams to be accessed that are generated by the Kinect sensor. The SDK supports skeletal tracking, advanced audio capabilities and it comes with sample code and documents. The SDK installer is under 100 MB and easy to install.

**10.2.2 Glove Pie:**

GlovePIE stands for Glove Programmable Input Emulator. It doesn't have to be used with VR Gloves, but it was originally started as a system for emulating Joystick and Keyboard Input using the Essential Reality P5 Glove. Now it supports emulating all kinds of input, using all kinds of devices, including Polhemus, Intersense, Ascension, WorldViz, 5DT, and eMagin products. It can also control MIDI or OSC output.In the GlovePIE window you type or load a simple script. For example to control the WASD keys with a glove:

W = glove.z > -50 cm  
S = glove.z < -70 cm  
A = glove.x < -10 cm  
D = glove.x > 10 cm

You can also use GlovePIE to play Joystick-only games without a joystick, or keyboard-only games with a joystick. Or you can use it to create macro buttons for complex keystrokes. You can even use it to control multiple mouse pointers with multiple mice.

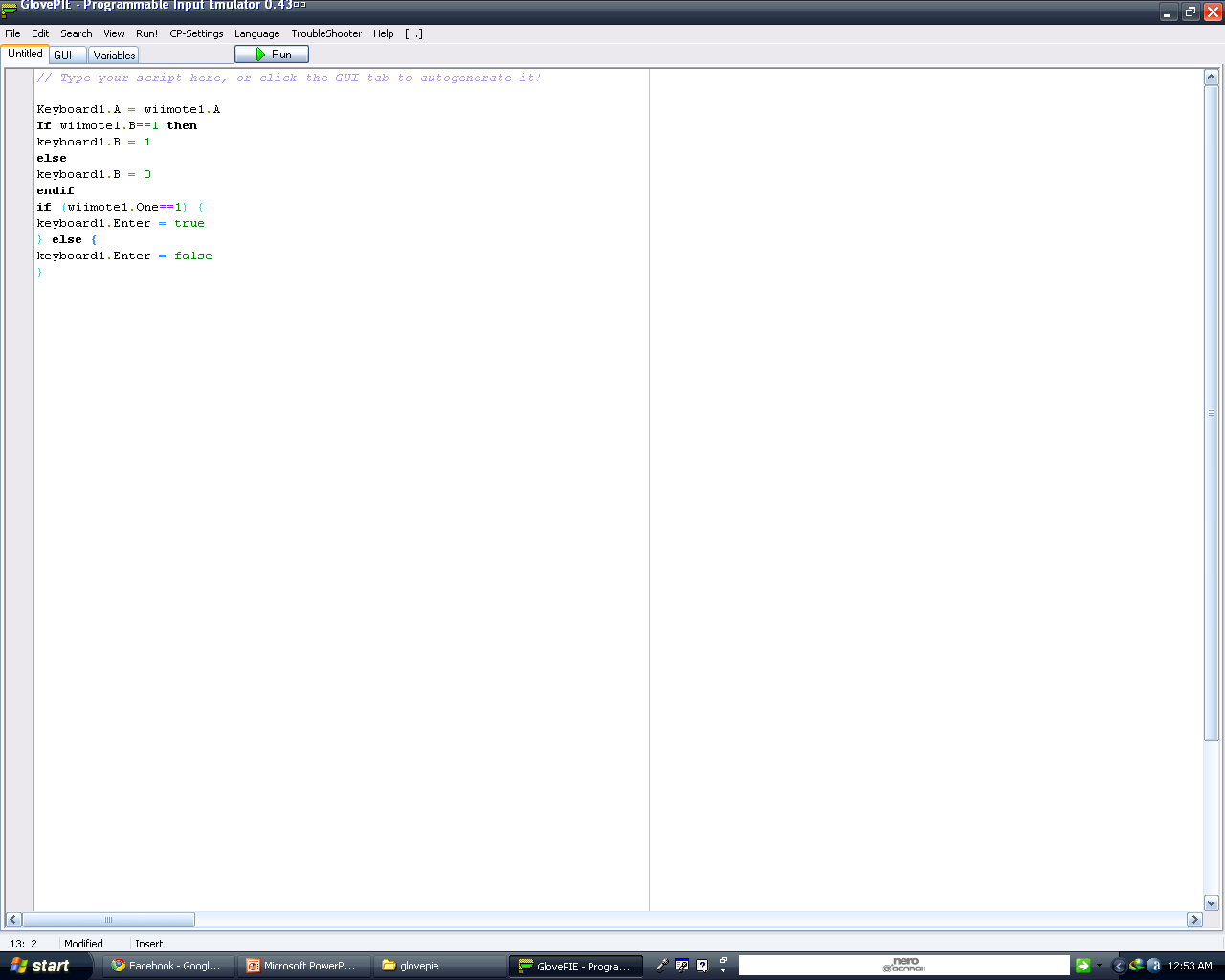
**System Requirements:**

* Windows 98 or above (Windows 2000 or above to emulate keys in DirectInput games or use multiple fake cursors - Windows XP or above to get input from multiple mice or keyboards individually or to read some special keys).
* DirectX 8 or above.
* There is other optional software you might need for certain features. See the download page for links to download them. Joystick emulation requires PPJoy. Speech requires SAPI 5.1 with Microsoft recognizer.
* You don't need any special hardware.

**Hardware Supported:**

* Nintendo Wii Remote (Wiimote)
* Sony Dual Shock 3 (except for Gyro) or Sony SIXAXIS
* NaturalPoint (Or eDimensional) TrackIR, OptiTrack, SmartNav
* FakeSpace Pinch Gloves (9600 baud by default, but can be changed)
* Concept 2 PM3 rowing machines (ergo or erg)
* All joysticks or gamepads recognized by Windows
* Parallel port gamepads (with PPJoy)
* All keyboards
* Mice with up to 5 buttons and 2 scroll wheels
* Most microphones (don't have to be high quality)
* Most MIDI input or output devices
* Essential Reality P5 Glove
* 5DT Data Glove (all versions)
* eMagin Z800 3D Visor HMD
* Polhemus trackers (must be set to 115200 baud): IsoTrak II, FasTrak, Liberty, Patriot, Liberty Latus
* Ascension trackers: Flock of Birds, MotionStar, etc.
* Intersense trackers: InterTrax, InertiaCube, IS-300, IS-600, IS-900, IS-1200, etc.
* WorldViz PPT trackers (all versions)
* GameTrak (only as a joystick, no direct support)

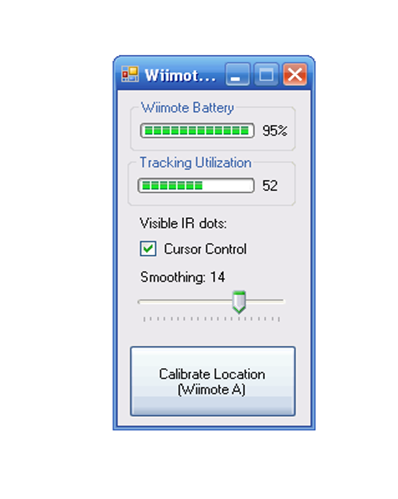
As said previously, GlovePIE can interpret various controllers which includes microphone. It can interpret what is being said into the microphone using Windows’ built in Voice Interpretation tool. Thus we can also write codes which accept voice commands and then assign different functions.Ex: if “minimize all” is said into the microphone, Windows key+M of keyboard would be emulated which would minimize all open windows.



**Fig 10.6 Glovepie**

The picture shows the GlovePIE Programmer and a sample code of emulating keyboard keys via wiimote in different ways.The GlovePIE uses a Linear but iterative execution method which means once the code has been executed from top to bottom, it will again start from top. Green when connected to another Bluetooth enabled device.

**10.2.3 Wiimote Whiteboard:**

****

**Fig 10.7 Wiimote Whiteboard**

Since the Wiimote can track sources of infrared (IR) light, you can track pens that have an IR led in the tip. By pointing a Wiimote at a projection screen or LCD display, you can create very low-cost interactive whiteboards or tablet displays. Since the Wiimote can track upto 4 points, up to 4 pens can be

used. It also works great with rear-projected displays.

**10.2.3.1 Software:**

The calibration and mouse cursor emulation software is available for you to download and try yourself. Note: My mouse emulation code isn't perfect. If any of you are programmers and can get it working with

Alias Sketchbook, drop me a line.  
1. Connect your Wiimote to your PC via Bluetooth. There are a number of tutorials online on how to do this, possibly even for you specific software/hardware configuration. The Wiimote works with many (but not all) Bluetooth drivers. You can report/read about compatibility issues at WiimoteProject.com   
2. Download the Wiimote Whiteboard software to the right. Please read the "READ ME.txt" file first! Make sure your wiimote is connected via Bluetooth, and then run the ".exe" in the main folder. NOTE: Good placement of the Wiimote is key to good tracking. View the README for more info.   
  
**10.2.3.2Multitouch:**

The multitouch demos are custom C# DirectX programs. You may download the sample program to the right, but this is provided for developers without support or documentation. The code is built on top of this Wiimote library. Unfortunately, multi-touch capable applications are currently extremely rare. Hopefully,

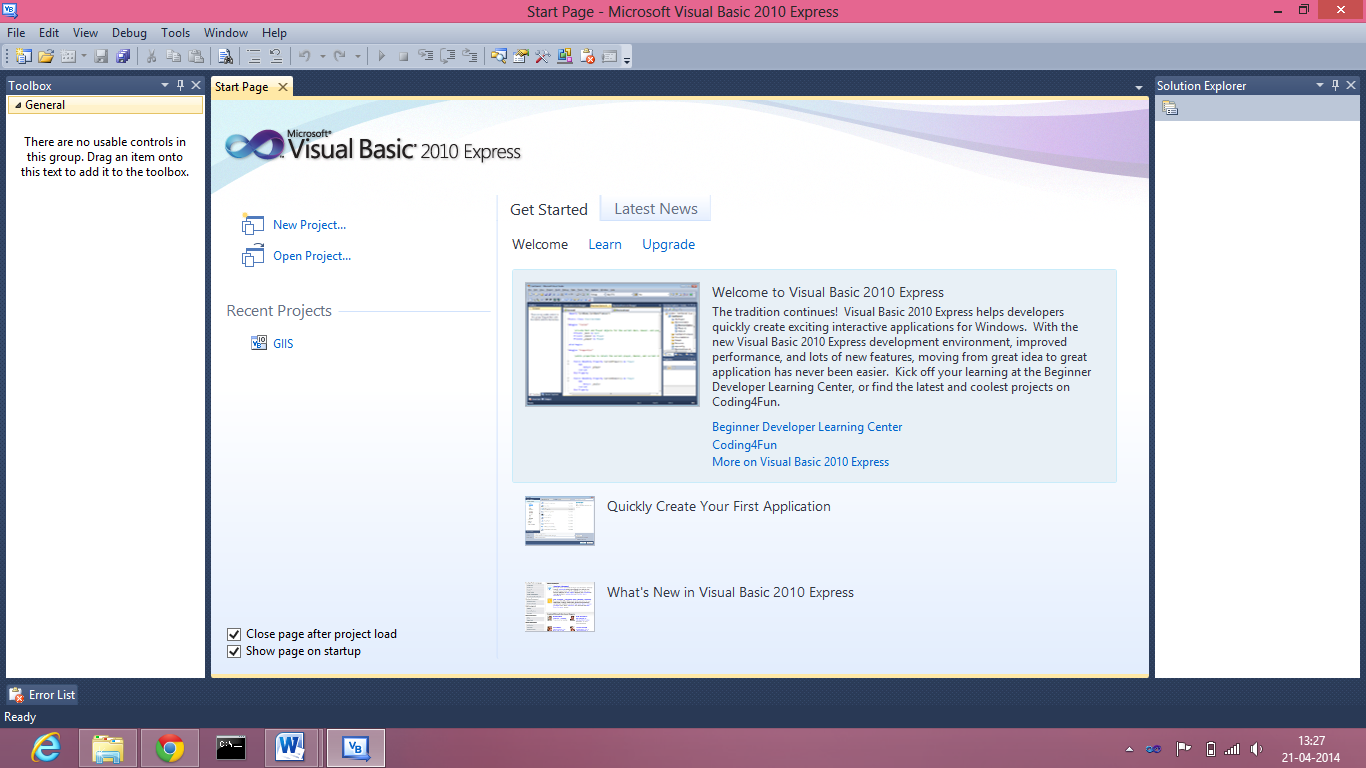
that will change as more developers explore its potential.   
  
**10.3 APPLICATION LAYER:**

This layer combines various logical outputs from the processing layer to combine them into functions.   
As the processing layer itself modularizes each logical output, combination of this output is used to implement the application For example, hand co-ordinates and voice recognition can be combined to emulate mouse control and voice functions such as close all applications. This has the Visual basic coded GUI that controls various working modes. Depending upon the mode of operation as dictated by the user, this switches from “near mode – touch mode” and “far mode – hands free mode”. Additionally this has a sub program coded in C# to capture the user’s voice command which then causes to switch between the working modes

**11. DESIGN MINUTIAE**

**11.1 FRONT END DESIGN:**

**11.1.1 Visual Basic .NET:**



**Fig 11.1 Visual Basic .NET**

* Visual Basic (VB) is the third-generation event-driven programming language and integrated development environment (IDE) from Microsoft for its COM programming model. Visual Basic is relatively easy to learn and use.
* Visual Basic was derived from BASIC and enables the rapid application development (RAD) of graphical user interface (GUI) applications, access to databases using Data Access Objects, Remote Data Objects, or ActiveX Data Objects, and creation of ActiveX controls and objects. Scripting languages such as VBA and VBScript are syntactically similar to Visual Basic, but perform differently.
* A programmer can put together an application using the components provided with Visual Basic itself. Programs written in Visual Basic can also use the Windows API, but doing so requires external function declarations.
* The final release was version 6 in 1998. Microsoft's extended support ended in March 2008 and the designated successor was Visual Basic .NET (now known simply as Visual Basic).

visual Basic .NET (VB.NET) is an object-oriented computer programming language that can be viewed as an evolution of the classic Visual Basic (VB), implemented on the .NET Framework. Microsoft currently supplies two main editions of IDEs for developing in Visual Basic: Microsoft Visual Studio 2012, which is commercial software and Visual Basic Express Edition 2012, which is free of charge. The command-line compiler, VBC.EXE, is installed as part of the freeware .NET Framework SDK. Mono also includes a command-line VB.NET compiler. The most recent version is VB 2013, released November 18, 2013.

**11.1.1.1 Language features:**

Like the BASIC programming language, Visual Basic was designed to be easily learned and used by beginner programmers. The language not only allows programmers to create simple GUI applications, but can also develop complex applications. Programming in VB is a combination of visually arranging components or controls on a form, specifying attributes and actions of those components, and writing additional lines of code for more functionality. Since default attributes and actions are defined for the components, a simple program can be created without the programmer having to write many lines of code. Performance problems were experienced by earlier versions, but with faster computers and native code compilation this has become less of an issue.

Although programs can be compiled into native code executable from version 5 onwards, they still require the presence of runtime libraries of approximately 1 MB in size. This runtime is included by default in Windows 2000 and later, but for earlier versions of Windows like 95/98/NT it must be distributed together with the executable.

Forms are created using drag-and-drop techniques. A tool is used to place controls (e.g., text boxes, buttons, etc.) on the form (window). Controls have attributes and event handlers associated with them. Default values are provided when the control is created, but may be changed by the programmer. Many attribute values can be modified during run time based on user actions or changes in the environment, providing a dynamic application. For example, code can be inserted into the form resize event handler to reposition a control so that it remains centered on the form, expands to fill up the form, etc. By inserting code into the event handler for a keypress in a text box, the program can automatically translate the case of the text being entered, or even prevent certain characters from being inserted.

Visual Basic can create executable (EXE files), ActiveX controls, or DLL files, but is primarily used to develop Windows applications and to interface database systems. Dialog boxes with less functionality can be used to provide pop-up capabilities. Controls provide the basic functionality of the application, while programmers can insert additional logic within the appropriate event handlers. For example, a drop-down combination box will automatically display its list and allow the user to select any element. An event handler is called when an item is selected, which can then execute additional code created by the programmer to perform some action based on which element was selected, such as populating a related list.

Alternatively, a Visual Basic component can have no user interface, and instead provide ActiveX objects to other programs via Component Object Model (COM). This allows for server-side processing or an add-in module.

The language is garbage collected using reference counting, has a large library of utility objects, and has basic object oriented support. Since the more common components are included in the default project template, the programmer seldom needs to specify additional libraries. Unlike many other programming languages, Visual Basic is generally not case sensitive, although it will transform keywords into a standard case configuration and force the case of variable names to conform to the case of the entry within the symbol table. String comparisons are case sensitive by default, but can be made case insensitive if so desired.

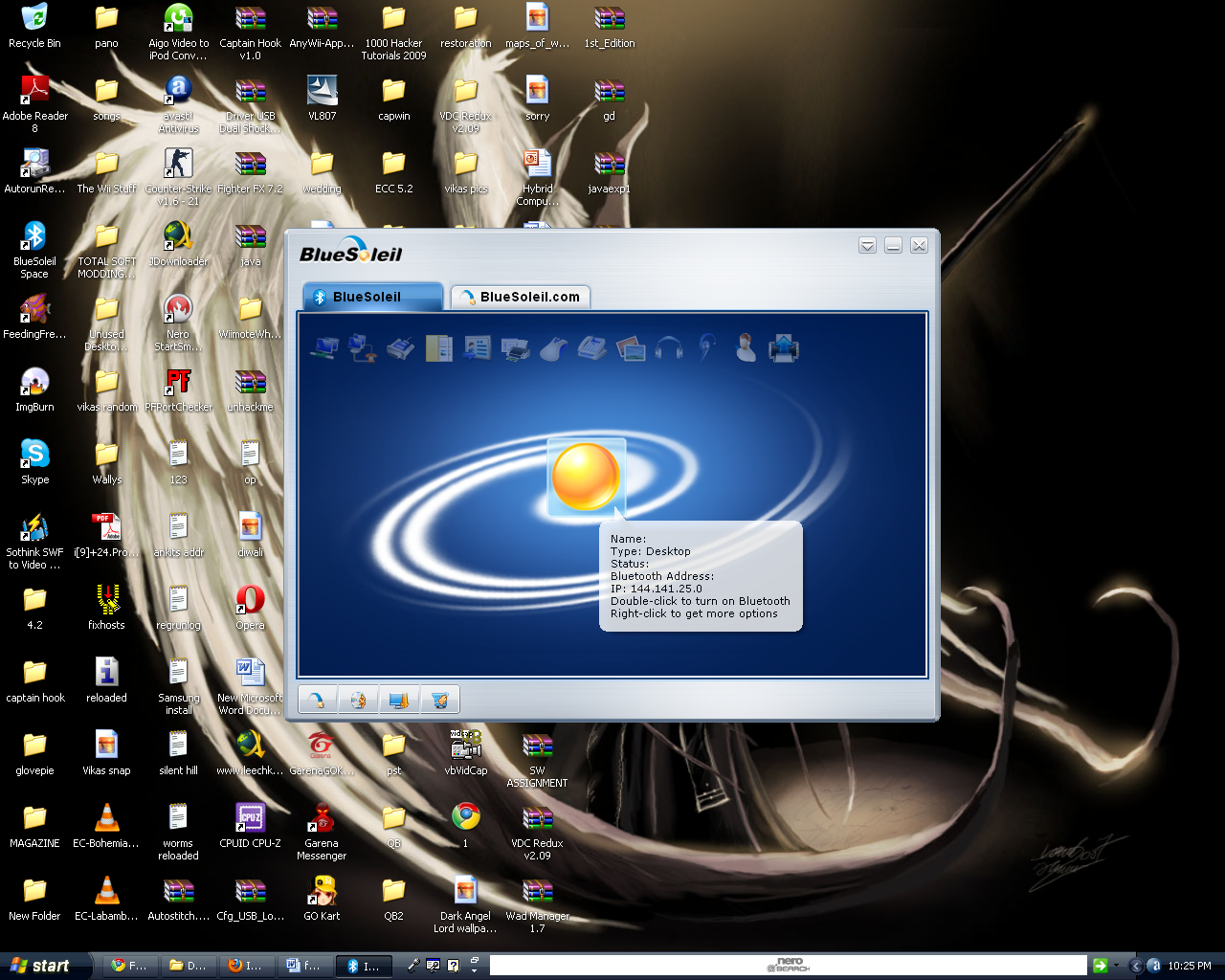
The Visual Basic compiler is shared with other Visual Studio languages (C, C++), but restrictions in the IDE do not allow the creation of some targets (Windows model DLLs) and threading models.

In April 2010, Microsoft released Visual Basic 2010. Microsoft had planned to use Dynamic Language Runtime (DLR) for that releasebut shifted to a co-evolution strategy between Visual Basic and sister language C# to bring both languages into closer parity with one another. Visual Basic's innate ability to interact dynamically with CLR and COM objects has been enhanced to work with dynamic languages built on the DLR such as Iron Python and Iron Ruby. The Visual Basic compiler was improved to infer line continuation in a set of common contexts, in many cases removing the need for the "\_" line continuation character. Also, existing support of inline Functions was complemented with support for inline Subs as well as multi-line versions of both Sub and Function lambdas.

**11.2 MIDDLE WARE:**

**11.2.1 BlueSoleil:**

BlueSoleilis a Bluetooth software/driver for Windows XP, Windows Vista, Windows 7, Linux, WinCE and Windows Mobile. It supports Bluetooth chipsets from CSR, Broadcom, Marvell etc. Bluetooth dongles, PCs, Laptops, PDAs, PNDs and UMPCs are sometimes bundled with a version of this software.BlueSoleil is developed by IVT Corporation and the first version was released in 2002.



**Fig 11.2 Bluesoleil**

BlueSoleil is a Bluetooth software/driver for Windows XP, Windows Vista, Windows 7, Linux,

WinCE and Windows Mobile. It supports Bluetooth chipsets from CSR, Broadcom, Marvell etc. Bluetooth dongles, PCs, Laptops, PDAs, PNDs and UMPCs are sometimes bundled with a version of this software.BlueSoleil is developed by IVT Corporation and the first version was released in 2002.

**11.2.1.1 Demo software:**

A demonstration version of BlueSoleil is available, restricting the device after 2MB data transfer, approximately 1.5 minutes of high-quality audio or 2-4 hours of mouse use. The software must be purchased to enable unlimited use.

**Features:**

BlueSoleil features the following technologies:

* Voice over IP
* Advanced Audio Distribution Profile and Audio/Video Remote Control Profile
* Personal Area Network
* Basic Imaging Profile
* Cordless Telephony Profile
* Instant Messaging
* Integration Phone tools as a profile

**Interoperability:**

* BlueSoleil has been distributed over 30 million copies. IVT has also established an interoperability testing center where it has built up a large library of Bluetooth products which are on the market in order to perform interoperability testing.
* Various Bluetooth dongles are delivered with an obsolete or demonstration version of Bluesoleil. New versions are available as a standalone purchase from the vendor's website. Regardless of whether the bundled or the standalone version is purchased, the software enforces licensing restrictions which tie it to the address of a specific Bluetooth dongle.
* Bluesoleil works with the main Bluetooth Silicon Vendors hardware, such as Accelsemi, Atheros, CSR, Conwise, 3DSP, Broadcom, Intel, Marvell, NSC, RFMD, SiRF as well as baseband IP such as RivieraWaves BT IP.
* If there is no Bluetooth dongle attached to the PC the Bluetooth logo will be grey, blue if a dongle is

attached, and green

**11.3 BACK END DESIGN:**

This would include various programs and scripts written in GlovePIE, java, C++ etc.

These programs (modules) would be called by the front end by clicking on its respective button.

The main backend component is GlovePIE and the Kinect SDK (Apart from the official Kinect sdk from microsoft, various other third party sdk may be used such as openni, NITE etc

Below is a sample code for emulating keyboard buttons from wii mote

mouse.LeftButton = Wiimote1.A

mouse.RightButton = Wiimote1.B

mouse.MiddleButton = Wiimote1.home

mouse.WheelUp = Wiimote1.Plus

mouse.WheelDown = Wiimote1.Minus

**Below is a sample code to access Kinect via Glove PIE:**

Var.rightHand3dDist = sqrt( sqr(player1.rightshoulder.x-player1.righthand.x) + sqr(player1.rightshoulder.y-player1.righthand.y) +  sqr(player1.rightshoulder.z-player1.righthand.z) )

var.leftHand3dDist = sqrt( sqr(player1.leftshoulder.x-player1.lefthand.x) + sqr(player1.leftshoulder.y-player1.lefthand.y) +  sqr(player1.leftshoulder.z-player1.lefthand.z) )

**12. IMPLEMENTATION**

**A code snippet for user auto tracking calibration and manual calibration with sample voice commands:**

if said("close this") then

keyboard1.Alt = true

keyboard.F4 = true

keyboard1.Alt = false

keyboard.F4 = false

end if

(keyboard1.Windows + keyboard.M) = said("minimize all",5)

if said("switch") then

keyboard1.alt + keyboard.Tab = true

end if

if said ("select") then

keyboard1.Alt = false

keyboard1.tab = false

end if

keyboard.Alt + keyboard.Space + keyboard.N = said("minimize this")

////////////////////////mouse//////////////////////////

if (key.Up AND key.PageUp) then

kinect.SmoothPitch = Kinect.SmoothPitch+10

end if

if (key.Down AND key.PageDown) then

kinect.SmoothPitch = Kinect.SmoothPitch-10

end if

debug = 'bottom' +player.ClippedBottom +' top: ' +player.ClippedTop +' pitch: ' +kinect.SmoothPitch +' track: ' +player.tracking

if ((key.R AND key.S AND key.T) OR player.tracking = false) then

kinect.SmoothPitch = 0

wait 2 seconds

end if

if(key.A) then

if (player.ClippedBottom = true) then

kinect.SmoothPitch = kinect.SmoothPitch-2

wait 1 second

end if

if (player.ClippedTop) then

kinect.smoothpitch = kinect.SmoothPitch +2

wait 1 seconds

end if

end if

if (key.PageUp)

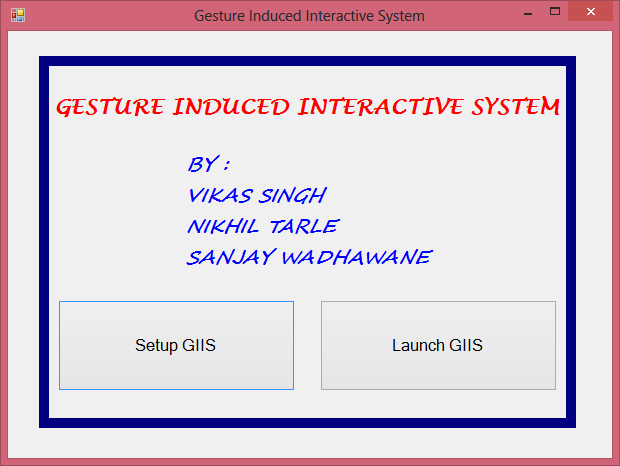
Kinect.SmoothPitch += 5

else if (key.Down)

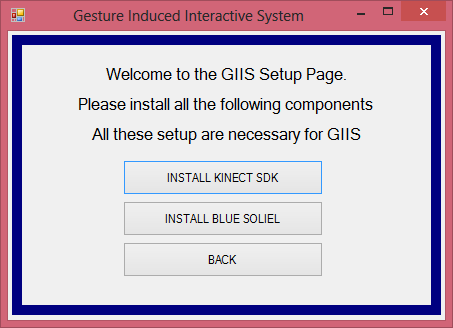
Kinect.SmoothPitch -= 2

end if

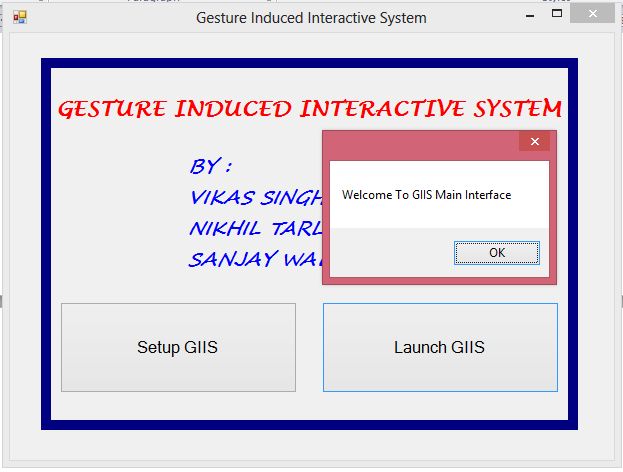
**13. RESULT AND OBSERVATION**



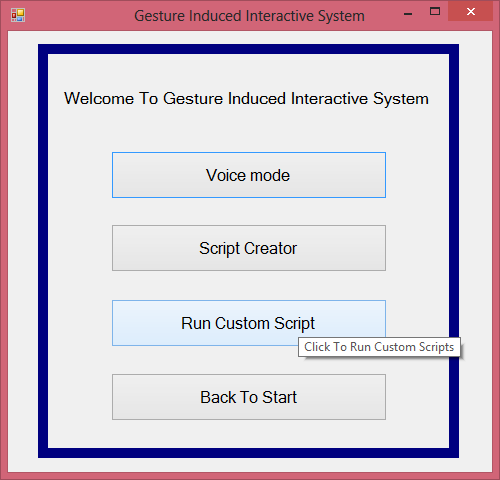
**Fig 13.1 Introduction screen**



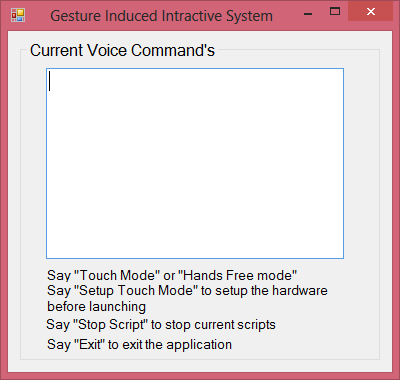
**Fig 13.2 Setup page which installs various required cop,oents for the project**

****

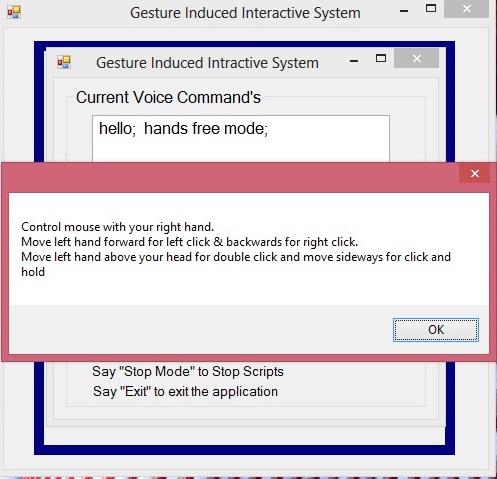
**Fig 13.3 Clicking on the Launch GIIS takes you to the main interface**



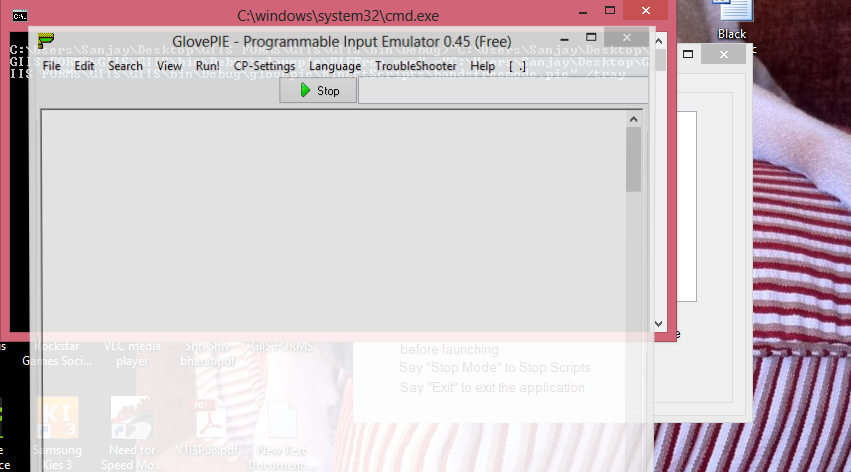
**Fig 13.4 Main Interface**



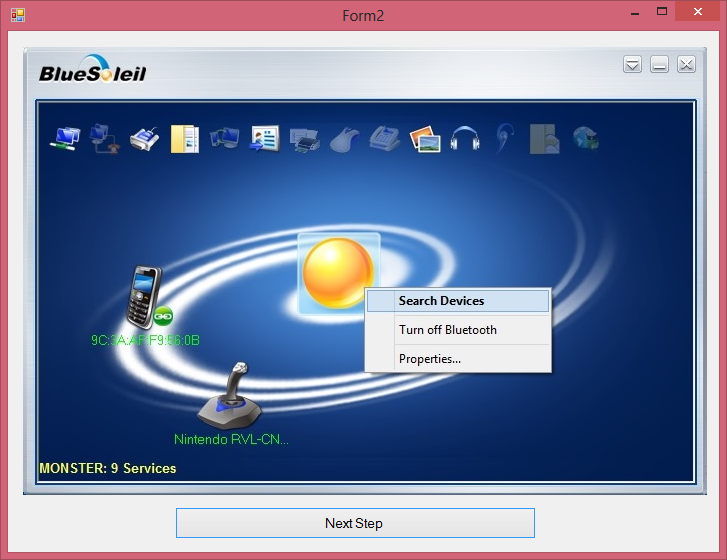
**Fig 13.5 Clicking on the voice mode starts a command Listener which swithes between touch mode or hands free mode**

****

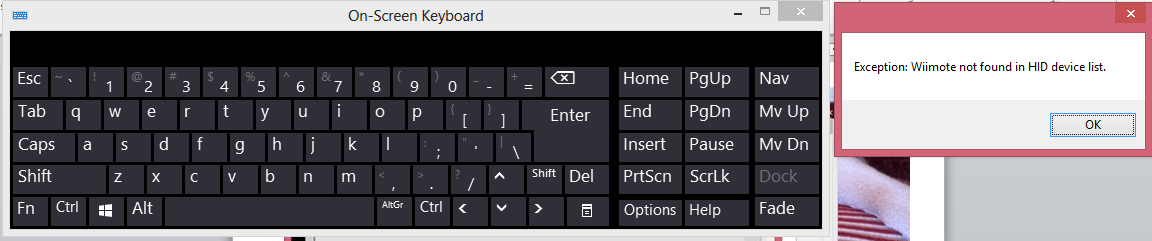
**Fig 13.6 Instruction for hand free mode**

****

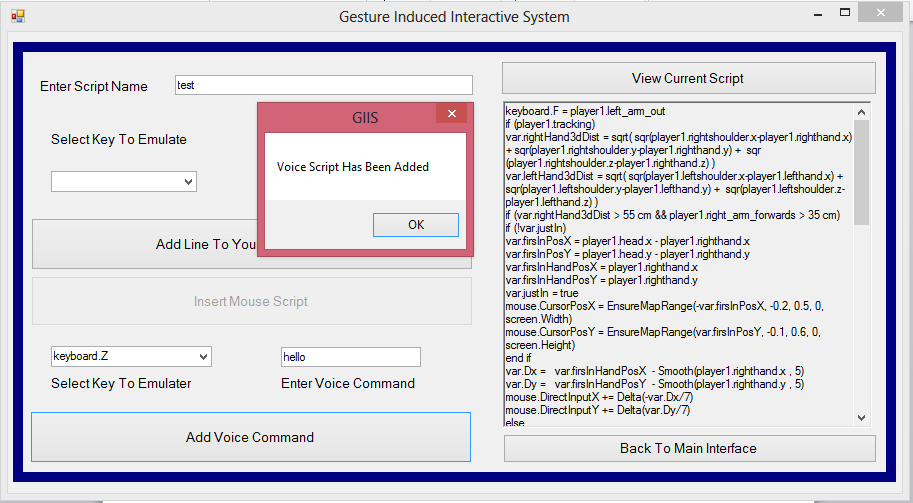
**Fig 13.7 Hands free mode**



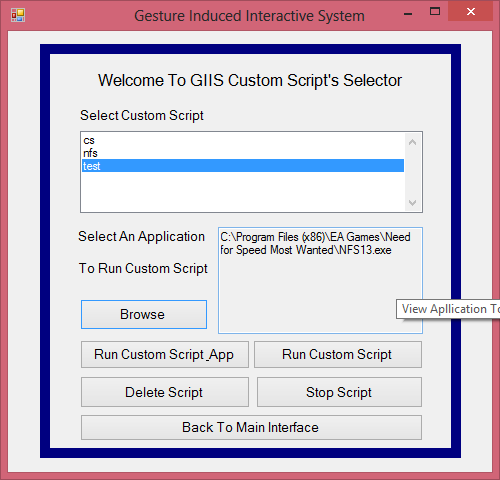
**Fig 13.8 Pre touch mode setup page which guides the user to connect the wiimote via Bluetooth**

****

**Fig 13.9 The onscreen keyboard in conjuction with touch mode for on screen typing. Touch mode sub program throws out an exception if the Wii mote is not connected.**

****

**Fig 13.10 Clicking on the custom Script creator allows user to create custom scripts for various applications/games etc**



**Fig 13.11 Clicking on the “run custom script” allows to launch the user created scripts, or scripts and application both along with other actions as indicated above.**

**14. CONCLUSION**

* With further improvements on the controllers or the hardware of Bluetooth, we can control out pc without any additional gloves or add-ons.
* That is we can simply use our body movements to control the s/w
* This would also include making various facial expressions for deeper and tighter integration for various functions
* Like we can tilt our head to rotate a picture onscreen and blink our eyes to capture the screen.
* Also, currently in beta stages, we can use brain signal mapping for various function like we can think of sentences in our mind, and the brain waves will be mapped to type those sentences