Description of the Invention
‘GeoTouch’

I. INTRODUCTION AND BACKGROUND

In recent years, the amount of research going on in the field of multi-touch technology has increased manifold. Multi-touch screen technology, both in consumer electronics (for example Smart Phones, MP3 players etc) and in Wall mount displays (for example Kiosks) seem to be omnipresent in the future. These multi-touch screens provide a collaborative workspace for the users so that they can perform various tasks without the use of traditional input devices like keyboard and mouse, and with the use of their bare fingers.

The word Multi-touch or Multitouch represents various human computer interaction techniques which allow users to interact and do work on computers with the use of their bare hand gestures and without the use of the traditional computer IO control devices namely mouse and keyboard [1]. Multi-touch devices normally consist of a touch screen which can capture human or stylus touch with the help of a sensor or a camera and software which decodes and recognizes multiple touch points, thus distinguishing multi-touch from the single touch devices such as ATM kiosks which at one time can only detect a single touch point input from the user.

In the field of human computer interaction, the traditional Graphical User Interface called WIMP (Windows, Icons, Menus, Pointing Device) is the current and the main mode of human-computer interaction. This interface consists of a pointing device called mouse with the help of which the user can select icons and open menus on the windows screen, but the mouse pointer has only 2 degrees of freedom thus not much efficient for multitasking. However human fingers have 23 degrees of freedom and can interact with the computer screen without the use of an intermediate connection (as required in mouse, keyboard) thus increasing the efficiency of human-computer interaction. [2]

Mehta and Nimish at the University of Toronto developed the first multi-touch screen technology in 1982, consisting of a frosted glass panel which could detect human input (finger pressure) and transmit it to a computer [3]. This was the first version of a multi-touch screen. This technique was followed by development of a capacitive array based touch screens at Bell labs in 1984. This technology developed by Bob Boie used capacitive array of sensors which were overlaid on a Cathode Ray Tube [3]. This was the first sensor based multi-touch screen technology.

The next 20 years saw a tremendous amount of growth in the multi-touch technology and many devices such as Flip Keyboard (Bill Buxton of Xerox PARC) [4] or the Active Desk (University of Toronto) [5] were invented and used for providing user a rich and new experience in human computer interaction.

The release of Apple iPhone in 2007 brought a revolution to the smart phone industry. It was the first touch screen smart phone which employed multi-touch capability to a limited degree [6]. Also in the same year Microsoft released Surface 1.0 [7], which was basically an interactive table surface which was capable of sensing more than one hand and finger at the same time. This was used as a tool in collaborative and informative learning environment. Surface 1.0 was followed by Surface 2.0 in 2011.

Nowadays almost all smart phone makers have either started producing multitouch screen smart phones or started doing research in this direction after the trend set by Apple in 2007. The world is looking at an explosion of multi-touch display devices/surfaces in the consumer space in the years to come. This report provides a basic understanding and survey of all the main multi-touch technologies available in the market. These technologies differ in the approach they follow in sensing multiple touch of user and recognizing and interpreting them. Thus they can be categorized into two main categories as shown in Fig 1.1.

In Fig 1.1, it can be seen that the basic hierarchy of multi-touch devices is categorized into hardware or software. In the hardware category, the devices available can be divided into two main categories. Sensor based (present mostly in mobile devices) and Computer Vision based (present in Wall mounts and Table tops).
The software category mostly deals with the protocols, standards and methods used in the multi-touch programming. Multi-touch programming can be divided into two parts [1].

1. Detecting, reading and translating the input from the user into a gestural meaning. The input from user is also known as 'blob' [8].
2. Transmitting the input through a defined protocol to the main framework, where this information can be translated as a command to the high level application.

We will concentrate on the Software technologies only because GeoTouch is a software product.

**II. SOFTWARE TECHNOLOGIES**

Multi-touch Programming can be done in several languages like Python, C, C++, C# and Java [1]; but in Multi-touch Programming there are certain protocols and standards which the programmer has to follow in order to develop software. Developing a Multi-touch application consists of three main steps:

1. Detecting and Reading the ‘blob’ input from camera.
2. Translating the input using the predefined protocols into meaningful gestures.
3. Using these signals as command inputs to a high level application layer.

**1. DETECTING AND READING THE ‘BLOB’ INPUT FROM CAMERA:**

Almost all the known techniques in Optical/Vision based suffer from ambient light effect and occlusion. This makes it quite difficult for the microprocessor to ignore all the background noise from the image captured by the camera. The first
make use of a Gaussian mixture models which are adaptive and thus make sure that all the background objects in the captured image have been erased.

After background subtraction and thresholding, the next step is the tracking of the finger blob. This is done by tracking the blob frame by frame. The best approach to achieve this is to make use of a tracking model which can be built on Particle or Kalman filter. Kalman filter has a state equation and an observation equation in its model. As we do not have direct observation to our object or in other words our state is unobservable, so we use Kalman filter to predict the next state for us.

However modern multi-touch systems have such robust and accurate hardware, that these tracking filters are becoming obsolete in multi-touch programming.

Another method of performing the finger tracking in multitouch systems is to use K-NN (K Nearest Neighbors) technique. As the finger can travel only a small distance between the capture of two frames, thus a finger blob in a frame will be compared by all its closest neighboring blobs by their Euclidian distance and the best match will be returned.

2. TRANSLATING THE INPUT USING THE PREDEFINED PROTOCOLS INTO MEANINGFUL GESTURES:

After accurate detection of the position of finger on the screen, the next step is to recognize the gesture intended by the user. The primary goal of this step is to create an automatic system which can identify gesture input and use the meaning of these gestures to transmit information or to control the display on the touch screen.

The most widely used technique in gesture recognition is the use of Hidden Markov Model and Artificial Neural Networks [1].

Hidden Markov models were originally developed for Speech Recognition but with the passage of time, they have become a well known approach in multitouch gesture recognition as well. This is because these models provide a probabilistic framework which can make the recognition of dynamically changing gestures made by human very easy. In such models each gesture is represented by left-right hidden Markov model[9]. First the training is done in which samples of gestures are collected and fed into the system to establish a trained model. Then after filtering and smoothening the data is fed into the system which recognizes the gesture using the trained model.

3. USING THESE SIGNALS AS COMMAND INPUTS TO A HIGH LEVEL APPLICATION LAYER.

After recognition of a meaningful gesture, the next and final step is to transmit the information gathered from the gesture to the High level application layer which can interpret this information as command and data and change the display on the touch screen accordingly.

Some of the major high level programming languages used in multi-touch programming are discussed below.

1. PYTHON:

Python is a general purpose, high level programming language which runs on Windows as well as Linux/Unix and MAC OS. It is free to use because of its open source license [10]. Python is often said to be the easiest programming language to learn because of its simplicity of syntax. It helps developers to write simpler, easy to understand code in less amount of time as compared to other programming languages.

PyMT is an open source library of Python which can be used for free for developing multitouch applications. Its power lies in the cross platform support and ease to use.
2. .NET/C#:

Wikipedia [11] defines .NET framework as a software framework which runs on windows OS. It includes a huge library and a virtual machine which provides memory management and security.

Microsoft provides various frameworks for multitouch programming, however it is up to the software developer to either use a built in framework or to create his own. Existing frameworks are called WPF multi-touch frameworks (WPF stands for Windows Presentation Foundation). One example of a built in framework is Multitouch Vista which allows multi-touch interactions with Windows Vista and Windows 7.

If someone wants to create his/her own framework, then the first step is to get the data captured by the camera and processed by the microprocessor into .NET, which can be easily achieved using TUIO or CsTI (C# Touchlib Interface).

TUIO stands for Tangible User Interface Objects. It is a multi-touch protocol which is used in detection and identifying blobs. Thiago de F. O. Ara’ujo et al used a modification of this protocol to detect blobs and for sending the blob position to the application layer [12].

CsTI was developed by Donovan Solms in 2007 while working in NUI group [13]. Raw Touch data could be sent to application layer using this interface through a binary connection. The main strength of this interface is that it converts touch events to the standard .NET events which can be easily handled by the application developer.

III. MULTITOUCH GESTURES AND INTERACTION

The earliest multi-touch devices predated the computer graphical user interfaces (GUI) which dominate our daily interactions with non-embedded computer systems today. In these cases, custom human-computer interface methodologies had to be designed, implemented and taught. However today, an entire generation of computer users expect a WIMP-based method of interaction. These users know what happens when a button is pressed (clicked), or a scroll-bar is dragged.

This expectation of users provides an initial benefit to designers but also imposes limits based upon user expectations and comfort level when expectations are not met. The benefit that designers will experience is that as long as their hardware system can map touch gestures in a sensible way to common WIMP mouse events, then these events can be fed into the windowing system, and the software running on the system will respond in a predictable way: little modification will then be required for software to operate on the multitouch system.

Fu et. al. [14] classifies interactions into direct and indirect gestures. The former allowing manipulation of on-screen objects that can have immediate effect (such as rotation, movement, deletion). The latter gestures are used to represent abstract ideas (such as “display menu”, or “exit application”). In fact, this is nothing new, the author himself was using mouse strokes on Mentor Graphics workstations in 1991, shapes that could be drawn with the mouse on the screen to trigger shortcut actions without having to move the mouse cursor to some area of the screen to press an action button – a relatively time consuming proposition given the very large screen resolutions and slow mouse movements of these CAD workstations.

In terms of gestures, one and two-finger gestures are most common. A single finger can emulate a mouse click, or a mouse click-drag. The touch itself may be short or long, with many interaction systems being configured to raise a menu as a response to a long touch. Two-finger gestures may be a precursor to a zoom-in or zoom-out movement (dragging the fingertips closer together or further apart), a rotation (the fingertips rotate about an imaginary central point), or some kind of translation or scroll (both fingertips move together).

Some two-finger gestures are naturally ones that can be made with a single hand, whereas others are more likely to be made by the index finger of both hands. The primary means of deciding whether it is a single handed or two handed two-finger gesture is through the physical distance between touch points. A further complication in a multi-user display is that two concurrent touch points could in fact signal a one-finger gesture by two users, rather than a two-finger gesture by one user. Again, distance between touch points should be the primary means of differentiating between these two possibilities. The basic one-finger and two-finger gestures are shown in Fig. 3.1.
IV. NOVELTY OF THE INVENTION

The GeoTouch Project was launched in 2008 and the main purpose of this project was to build a multitouch system for education, collaboration, public dissemination and discussion within Earth Observatory of Singapore (EOS) for Geologists and Earth Scientists. As the scientists required a means of accessing accurate data quickly in a showcase location for dignitaries, press and other visitors, GeoTouch emerged as a useful tool to visualize GIS content on a large multi-touch screen.

The software uses Microsoft Surface APIs and combined with ESRI WPF SDK provides an awesome learning and interactive environment for GIS related work. There are some companies that are making multitouch GIS related apps for iPads, For Example GIS Pro By Garafa, LLC [15] but specifically for large scale touch screens, currently no other commercial software exists which combines all the useful features from Surface profiling to importing layers into one single user interface design. It works well as an education tool, extremely well as a tool to interact with visiting media representatives, and works well as a 'high-technology' draw to attract attention to the research centre. It also provides good access to the static data sets/map layers uploaded on to the device.

V. TECHNICAL DESCRIPTION

This chapter will deal with the technical details of the GeoTouch Software. We will start by describing the software framework and the design pattern used and then go on to talk about the programming languages involved and the design/development tools. The chapter will also cover in detail all the Assemblies and individual code files and their interaction with each other.

Client-Server Model:
GeoTouch runs on a Client Server Model for getting and displaying data. The map and layers are stored in a local Server and the PC running the software queries the server for data and thus after receiving it displays on the touch screen. The Server used is ArcGIS Server which is specifically made by ESRI for hosting GIS datasets. The software at the client side uses C# libraries and ArcGIS SDK functions to query maps and layers constantly.

Operating System:
GeoTouch currently can only run on PCs running 64bit Windows 7, and it requires a touch screen monitor with 1920 x 1280 resolution for best display. However work is under progress to develop GeoTouch for iPads as well.

Operating Environment:
Currently GeoTouch can be used only in premises of Nanyang Technological University, because the Server is a local machine. However we are trying to upgrade to Web Server so that GeoTouch can be mounted anywhere outside and still be able to access EOS data.

GeoTouch Equipment:
GeoTouch currently runs on a Windows 7 64 bit PC with 8 GB RAM and requires less than 2 GB of storage space. The display unit is a large touch screen monitor (55 inch for GeoTouch A and 60 inch for GeoTouch B). This is basically a Samsung LED TV screen on which MICR3 touch screen overlay has been installed. The touch screen monitor can connected to the PC via DVI cable. Any computer running 64bit Windows 7 with significant RAM can run GeoTouch.
**.NET Framework:**
GeoTouch is built around .NET framework provided by Microsoft Corporation. The .NET Framework (pronounced *dot net*) is a software framework that runs primarily on Microsoft Windows. It includes a large library and supports several programming languages.

The .NET Framework's Base Class Library provides user interface, data access, database connectivity, cryptography, web application development, numeric algorithms, and network communications. Programmers produce software by combining their own source code with the .NET Framework and other libraries. The .NET Framework is intended to be used by most new applications created for the Windows platform. Microsoft also produces an integrated development environment largely for .NET software called Visual Studio.

GeoTouch uses C-sharp (C#) as its programming language and thus the code makes use of most of the libraries provided by .NET framework.

**MVVM Architecture:**
GeoTouch uses the Model-View-View-Model (MVVM) architecture which is specially designed for WPF applications. There are a number of free open source MVVM frameworks and libraries available for download on the web but GeoTouch uses MVVMLight framework which is being developed by a company called IdentityMine. Figure 5.1 below explains the basic flow of MVVM pattern.

![Fig. 5.1: Basic Concept of Model View View Model System.](image)

The basic idea in GeoTouch software architecture is that a new feature when being introduced in the code (for example a “button press feature”) will be divided into two parts, View and ViewModel. The View will be responsible for the styling, display and color of the button itself and the ViewModel will be responsible for the background code which will run after the button is pressed.

ViewModels and Views must not share any variables and the ViewModel must know anything about its View. That is the beauty of the MVVM pattern that it provides the developer an ability to easily change the user interface of the software without changing the code behind which is running in the ViewModel.

Using Data Binding, RelayCommands and Events to communicate between View and ViewModels, we keep a distinct separation between the 2 entities. When designing ViewModel, one must consider if that ViewModel is able to cater to any kind of UI design. As much as possible, we should try **not** to put UI-specific logic or properties in the ViewModel.

**ViewModelLocator:**
GeoTouch uses a ViewModelLocator which helps in binding the View files to their respective ViewModels.

**Data Bindings** are used for automatic update of properties in the View. Data Binding is a WPF feature. We use it by using the Binding extension in the Xaml.
**RelayCommands** are used to call a method from the View to the ViewModel. It is provided by the MVVM Light Framework.

**Events** are used to inform the View that something has happened in the ViewModel. We can achieve this functionality using the System.Windows.Interactivity.EventTrigger class.

**C#/XAML combination:**
GeoTouch uses Extensible Application Markup Language (XAML) for the View files and C# for the ViewModel files, the ViewModelLocator and for custom methods. XAML is also created by Microsoft and the key aspect of this technology is the reduced complexity needed for tools to process XAML than other programming languages. Thus it is one of the ideal choices for designers/developers when they design user interfaces.

**Tools used in Developing GeoTouch:**
The following tools and IDEs were used in development of GeoTouch.

1. Microsoft Visual Studio: for development of code behind
2. Microsoft Expression Blend: for designing of user interface
3. Adobe illustrator: for making icons for buttons

**GeoTouch Assemblies:**
GeoTouch is a combination of 5 interdependent projects listed below:

- **GeoTouch.View:**
  This contains the user interface part of GeoTouch, buttons, tiles, images and panels. It is responsible for styling of the user interface items. It is also the entry point to the application.

- **GeoTouch.ViewModel:**
  This assembly contains the code behind logic behind every user interface element

- **GeoTouch.CustomControls:**
  This assembly contains some custom made controls (for example Zap Scroller and infoWindow controls)

- **GeoTouch.Common:**
  This Assembly contains common classes, enums and properties used by other assemblies during the execution of the code.

- **GeoTouch.ArcGISServerExplorer.ParallelAsync:**
  This assembly handles the downloading of an ArcGIS Server from a URL in a tree structure.
The relationship between these assemblies can be easily seen from Figure 5.2 below

![Figure 5.2 Relationship between GeoTouch Assemblies](image)

1. **GeoTouch.View:**

   **File Structure**
   - GeoTouch.View
     - Properties
     - References
     - AttachedProperties
     - bin
     - Controls
     - Converters
     - FloatObjectViews
     - Fonts
     - Images
     - InfoWindowViews
     - Interactivity
     - LayerFunctions
     - Modules
     - obj
     - Resources
     - Singletons
     - TemplateSelectors
     - Themes
       - app.config
     - App.xaml
     - ClassDiagram1.cd
     - ClassDiagram2.cd
     - ClassDiagram3.cd
     - GeoTouch.View.cproj, vs10x
     - GeoTouch.View.xaml
     - GTViewTokens.cs
     - MainSurfaceWindow.xaml
     - MainWindow.xaml
     - NLog.config

   **Namespace structure**
   - GeoTouch.View
     - Project References
     - GeoTouch.View
       - GeoTouch.View.AttachedProperties
     - GeoTouch.View.Controls
     - GeoTouch.View.Converters
     - GeoTouch.View.FloatObjectViews
     - GeoTouch.View.InfoWindowViews
     - GeoTouch.View.Interactivity
     - GeoTouch.View.LayerFunctions
     - GeoTouch.View.Properties
     - GeoTouch.View.TemplateSelectors
     - XamlGeneratedNamespace
**GeoTouch.View.AttachedProperties:**
Contains UI specific properties. For Example icon display on buttons.

**GeoTouch.View.Controls:**
Contains UI specific controls. For Example Color Picker and Resizeable Frame.

**GeoTouch.View.Converters:**
Contains various custom convertors for use in various View files. For Example Mercator to Geographic Convertor is responsible for conversion between various coordinate systems.

**GeoTouch.View.FloatObjectViews:**
Contains the views responsible to show float objects (Float Objects are draggable dialogue windows).

**GeoTouch.View.InfoWindowViews:**
Contains the user interface for infowindows (pop up windows UI when a user taps on the map).

**GeoTouch.View.Interactivity:**
Contains the Behaviors, Triggers and Action classes for custom UI logic.

**GeoTouch.View.LayerFunctions:**
Contains the UI logic of the functions which the user can perform on layers-on-map. For Example Save, load, changing opacity etc.

**GeoTouch.View.Modules.ImportLayer:**
Contains the Views for importing layers onto map from Server.

**GeoTouch.View.Modules.Measure:**
Contains the helper classes for the Measure distance/area feature.

**Singletons Folder (GeoTouch.View):**
This folder contains Views that are created at the start of the application and exists throughout the lifetime of the application. This includes the panels on left and right and the features present in them.

The most important xaml files in the Singleton folder are

**AvailableLayersView:**
The available layer view contains the View inside the right panel.

![Available Layers](image-url)
It has the features of adding layers to the map. Layers are categorized in geographical terms and preplaced in different folders as shown above. The four buttons at the middle launch gallery window (containing images and videos), basemaps, landsat imagery and import layer feature.

**MapView:**
This UserControl represents the View for the MapViewModel.

**SurfaceProfileAndMeasureInkCanvas:**
This UserControl contains a SurfaceInkCanvas and all the logic (InkCanvasBehavior.cs) to insert graphics for Surface Profile Layer, Measure layer and InkCanvas layer.

**SelectionsView:**
This UserControl holds the pop up windows whenever a Graphic is selected.

**MeasureScatterView:**
This UserControl handles the logic for the Measure feature.

**ResizableFrames:**
This UserControl holds the Touch Pad and the Multi Layering feature.

**Left & Right:**
The Left & Right UserControls hold the LayersManagerView and the AvailableLayersView respectively.

**FloatObjects:**
This UserControl facilitates the draggable dialogue windows. The FloatObjectViews folder contains the UserControls that shows on FloatObjects.

There are 2 kinds of Float Objects:

1) Movable, not scalable, not rotatable objects (Layer properties, Email screenshot, etc)

2) Movable, scalable, rotatable objects (Gallery items)

![Fig 5.3 Float Objects](image)

**Bottom:**
This UserControl shows the map load status indicator

**DialogServiceView:**
This UserControl represents the DialogServiceViewModel which handles the modal dialog windows
**LayerManagersView:**

The layerManagerView contains the view inside the left panel.

It has ink canvas, search location, surface profile and measure added as preloaded layers, which means that these layers are initialized at the start of the application and remain active throughout the life of it.

![Tools](image)

**Fig 5.4 Tools**

**Resources Folder:**

This folder contains the ResourceDictionaries that GeoTouch uses. Basically contains reusable xaml-defined objects that are referenced by GeoTouch.View assembly.

**Images Folder:**

Contains the images for this application (icons, thumbnail images, etc.).
The MainWindow.xaml is the entry point of the whole code. It initializes different View files as shown in the figure

2. GeoTouch.ViewModel:

**GeoTouch.ViewModel.CustomLayers:**
The folder contains classes which provide custom layer functions which are not provided by ESRI.

**GeoTouch.ViewModel.Extras:**
Contains some extra customs functions for example Querying of Feature layers.

**GeoTouch.ViewModel.Factory:**
This namespace contains static classes that “produce” default instances. For Example basemaps default instances.

**GeoTouch.ViewModel.LayerFunctions:**
This namespace has those features which are used in manipulation of layers and layer properties. For Example DisplayFuncation ViewModel is used in changing the opacity of a layer.

**GeoTouch.ViewModel.Modules:**
This namespace contains supporting classes of a specific feature. For Example Importing Layers.

**GeoTouch.ViewModel.Serialization:**
This name space contains data contracts used in serialization/deserialization of XML files.
**Singletons Folder:**
This folder contains ViewModels that are created at the start of the application and exists throughout the lifetime of the application. This folder contains the following ViewModel classes.

**AvailableLayersViewModel:** This class is the code behind for AvailableLayersView. It is also responsible for reading the AvailableLayers.xml file to find out what layers are available for display.

**DialogServiceViewModel:** Contains the code behind for the GeoTouch Prompt or Acknowlegde states. There are five kinds of states as shown below

![Image of Dialog Service View Model States]

Fig 5.5 Dialog Service View Model States
MapViewModel:
This class manages the adding and removing of layers. Before adding a layer, we invoke the Initialize() method. If there is an error, we do not add it. Otherwise, we add the new layer at the bottom of the InkCanvasLayer. This ViewModel also has the MapMode property which tells the Map which mode it is currently in. The modes basically tell the map to not pan and zoom. The MapViewModel also contains logic for Map Coordinates display.

PreloadedLayers:
This class is not a ViewModel, but a class that stores the references to the PreloadedLayers: InkCanvas, SearchLocation, SurfaceProfile and Measure.

LayersManagerViewModel:
This class is the logic for the left side UI, where it shows layers that are currently on the map. This class splits the layers into to collection using the CollectionViewSource concept. The ViewModel also handles the reset functionality

EmailScreenshotViewModel:
This class handles the sending of the screenshot via email. It has the capability of becoming fully transparent when the email is being sent so that it doesn’t get captured in the screenshot.

ExtentViewModel:
This class handles the saving bookmark feature of GeoTouch. It contains a collection of bookmarks which gets updated everytime user adds a new bookmark.

GalleryViewModel:
This ViewModel class is the codebehind of Gallery View. Handles the media path and is responsible for the thumbnails too.

HelpViewModel/HelpViewSPModel:
These ViewModel classes contain only the properties required by the HelpView in displaying the Help content of Surface Profile and Measure tool.

ProgramsViewModel:
This class handles the switching to external programs. For Example Google Earth, Microsoft Surface Globe, Internet Explorer and Microsoft Power Point.

BaseMapsViewModel:
This class contains the available BaseMaps. And is responsible for loading the basemap when user clicks on a basemap in the View.

Settings:
This namespace defines the settings used for GeoTouch. For Example Email settings, and the Password.

ViewModelLocator: This class registers all the ViewModels and provides reference to them. Thus a View can talk to any ViewModel through ViewModelLocator.
3. GeoTouch.CustomControls:

GeoTouch custom controls as already mentioned contain some custom classes which are used by different Views repeatedly throughout the life of the application. These custom controls include:

**Zap Scroller:** This is a iphone/ipad navigation style control which is heavily used in FloatObjects and in ImportLayerView.

**Themes:** Contains XAML files for custom themes that can be adopted by any View as per requirement.

**InfoWindow/InfoWindowItemControl:** Contains the style of the popup information window and its content. InfoWindows are popped up when user clicks on the selectable item in a feature layer.

4. GeoTouch.Common:

This assembly contains all the common enums, keys, interfaces and other properties required by the ViewModels and Views. This also contains the Video Controller which implements functions of video playback (play, stop, pause and repeat).

**Layer & GTLayerBase:**
The Layer class, provided by the ESRI API, does not have properties that are relevant to GeoTouch, for example, LayerName and its available functions, therefore a custom wrapper class, named GTLayerBase is being designed, to store these relevant properties. Layer and GTLayerBase reference one another (2-way referencing). Layer references GTLayerBase by using WPF’s attached property concept via GTLayerBase.GetLayerAP. GTLayerBase references Layer by the EsriLayer property.
GTLayer & GTBaseMapLayer:
GTLayer and GTBaseMapLayer inherit from GTLayerBase. These strongly-typed classes are used to differentiate whether a layer is a base map layer and a regular layer.

![UML Diagram]

Fig 5.6 GTLayer and GTLayerBase Concept

5. GeoTouch.ArcGISServerExplorer.ParallelAsync:

This assembly contains methods and classes to download the data from ArcGIS server in the form of a URL tree and then display it to the user. The user can add the layers to map or to any other folder by his choice.

The tree structure in the import layer looks like this:
GeoTouch settings and related XML files:

There are 6 (xml + text) files in the build folders(debug/release) used by GeoTouch:

- Settings.xml
- AvailableLayers.xml
- EmailSettings.xml
- NLog.xml
- Log.txt
- CustomBaseMapLayers.xml

Settings.xml:
This file will set the various constant properties like where is the gallery located, what external programs are switchable to, the ArcGIS Servers Url for the Import Layer feature, etc, the Bing key used for Bing maps and the search feature.

AvailableLayers.xml:
This file defines the tree of layers in the Available Layers feature.

EmailSettings.xml:
This file defines the email parameters to send the screenshot for the Email Screenshot feature.

NLog.xml:
This file is created by NLog. NLog is responsible for sending an email to the specified email account whenever an error occurs while the code is still running.

Log.txt:
This is the log file used by NLog.

CustomBaseMapLayers.xml:
This file will add any additional Base Map layers to the BaseMapLayers feature.
VI. GEOTOUCH FEATURES
The list below provides a total set of GIS /Non GIS related functions GeoTouch can perform. The features and feature changes in italic have been added to the GeoTouch after July 2012.

![Image of GeoTouch interface]

Fig 6.1 The new User Interface of GeoTouch

1. **Left Panel:**

   **Draw:**
   This feature enables the user to draw freehand on the map, with one or more than one fingers at the same time. *Draw button has been removed from the bottom menu and is placed in Tools panel. The color selected is shown in the form of a small rectangle above the draw button.*

   **InkCanvas:**
   InkCanvas allows the user to change the colors for drawing and also gives user the functionality of changing the brush thickness, stroke opacity and allows storing and loading of strokes. *InkCanvas has been made smaller in size and the color options are reduced from 28 to 7.*

   **Search Location:**
   Search feature allows the user to search for any geographical place in the map. It shows the result in the form of a list inside the Search Location toolbox and by clicking on any name on the list the application zooms in to the region of interest. *Some predefined bookmarks of continents have been added to the Search Location toolbox and SaveExtent function results are also being displayed in the Search Location toolbox*

   **Surface Profile:**
   After activating surface profile, user can draw a line (preferably straight) on the map and get in response a popup window with an image showing the surface profile of the region selected by the user. The image can be dragged out of the window and user can rotate translate or scale it. *Surface profile toolbox is now removed and user can directly click on the button in left panel and start drawing on the map. Help for surface profile is displayed as little help icon in the left panel when surface profiling is active. Surface profile now also contains data for ocean bathymetry.*

   **Measure:**
   Measure tool allows the user to measure point to point, measure distance and measure area just by drawing strokes on the map. *Help for measure tool is also displayed in left panel when measure tool is active.*
SaveExtent:
This feature allows the user to save and load bookmarks. The saved bookmark is displayed in the Search Location window.

Email Screenshot:
This allows the user to send screenshot of the map to his/her email address. More explanation is now being added in the Email Screenshot feature in order to facilitate user.

MultiLayering:
Provides the functionality to overlay multiple basemaps on one another and frame of the multilayering window is rescalable.

Lock:
The lock feature allows the screen to be locked for panning and zooming of the map. However user can still draw on the map. Lock functionality has been included in the left panel of GeoTouch.

SwitchPrograms:
User can switch to external programs like Microsoft Surface Globe, Internet Explorer and Microsoft PowerPoint.

Reset:
Allows user to reset GeoTouch to its default state.

About:
Provides the developer’s email in case user wants to report any issue or any suggestion for the future.

Map scalebar can also be translated to any corner of the map by dragging.

2. Right Panel:
This panel contains Available layers, Gallery items, landsat imagery and import layer feature. The layers active on map are also displayed in this panel.

Map Coordinates are now available on the Top Right corner of the application window.

Available Layers:

Animation layers:
Following georeferenced animations are present in this folder:
1. SouthWest Monsoon
2. Katrina
3. Manila
4. Mauritius
5. Sandy
6. NorthEast Monsoon
7. Vamei

Bedrock:
Following Layers are present in this folder:
1. IrianJaya
2. Sumatra

Neotectonic:
Following Layers are present in this folder:
1. Myanmar
2. Taiwan Faults
3. Myanmar EQ
4. Sugar

Geodesy:
Following layers are present in this folder:
1. Siburet

Seismic:
Following layers are present in this folder:
1. Past 7 days Earthquakes RSS feed
2. Indian Ocean Bulletins RSS feed
3. Smithsonian weekly Volcano Report RSS feed
4. Pacific Ocean Bulletin RSS feed
5. ChristChurch
6. GPS Sites
7. USGS earthquakes

Volcanic:
Following layers are present in this folder:
1. SE-Asia volcanos
2. World Volcanos

Demo:
Following layers are present in this folder:
1. USA 1990-2000 Population Change
2. USA Retail Spending Potential
3. World Light Gray Base
4. ESRI StateCityHighway USA
5. ESRI Population World

• If provided with the legend for the layers in the form of an image, GeoTouch can display the legend in the form of a draggable item when the layer is loaded on map.
• USGS earthquake allows the user to click on any point in the map and then search for earthquakes that happened in that region since 1970.
• User can do query on feature layer items (points) to retrieve information from the Server.
• RSS feeds can be added just by emailing developer the URL of the feed.

Gallery:
Gallery contains the non georeferenced images and videos. It is now present in the right panel and is in the form of a draggable floating object.

Basemaps:
User can choose between a variety of basemaps for GeoTouch using the basemap feature.

Landsat imagery:
This feature allows the user to view satellite landsat imagery provided by ESRI. The user can tap and hold on any point in the map to bring up the menu which allows him/her to choose from various kinds of landsat imagery.

Import Layer:
This feature is used for importing layers which are being hosted on ESRI sample online servers and local GeoTouch Server. User can save the imported layers using ‘Save available layers’ button. User can also import landsat imagery (vegetation, population or natural).

VI. COMMERCIAL APPLICATIONS

The use of GeoTouch A and GeoTouch B by earth scientists has revolved around the presentation of GIS data, linked to several ancillary data sets. The multi-touch aspect is widely used, but the features exploited from the available set of features, are limited. Digital urban modelling, in the sense of ValueLab [16], also revolves around the presentation of GIS data, however it
has the very strong information architecture component. In fact, GeoTouch allied with city model scale data (such as CityEngine) could serve well as an urban planning tool.

GeoTouch is relatively an early example of the application-driven use of multi-touch interfaces in a non-research environment (i.e. not computer engineering/science research). It has the potential to act as a strong tool in the future for geology education in schools and universities and as well as in urban planning and modelling for future cities. Even now GeoTouch is being used by Professors at EOS to teach their students about geological concepts. The team at EOS is working on developing GeoTouch for iPad which will surely bring GeoTouch as a commercial application in the consumer market.
REFERENCES