# Bringing together EagleEyes users: Creating TCP Peer-to-Peer programs for two person interaction over the internet

Sharif Tai
Computer Science Thesis
April 25, 2003

Advisor: James Gips, Professor, John R. and Pamela C. Egan Chair,
Computer Science Department

# *Table of Contents*

## *Introduction*

How could I get EagleEyes users to connect to each other and be able to do activities together? Previously when using EagleEyes, the user was constrained to games or programs that had to be run locally, with no interaction with other users. They could use internet games, but these were not designed with the limitations of the EagleEyes system in mind. Often, the action was confined in a very small area, or there were actions needed like click-and-drag, or the user could click on too many things that were not related to the program (such as ads). To improve on this process, I designed and created a series of games specifically for EagleEyes, called EagleEyes Connect. This would allow two EagleEyes users to

## *How did I create EagleEyes Connect?*

To make this program, the users first create a TCP connection between the two computers. I chose TCP instead of UDP because I needed to ensure that the two computers were in sync – if one user launched a game and the other user did not get the message, then the system would not work. As for the network architecture, I chose to use a peer-to-peer setup in that the users connect directly to each other. One user does play the host in the sense tha

Another initial problem was connecting through the firewall, but I was able to solve this (after consultation with the IT staff) by ensuri

Othello, and Aliens. The Aliens program was one of the first games created for EagleEyes, and it consists of the user trying to get the cursor over a randomly placed alien on the screen. In the two-player version, each user has a cursor on the screen (they can see the other player moving) and they are competing to see who can hit the alien first.

To simplify the process of distribution, I have also created an automatic update application. This allows the users to upgrade to the latest version without doing any work (it is totally transparent to the user). This upgrade system downloads a text file from the internet containing the latest versions numbers available. Using the contents of that file, it compares the latest version numbers with the ones currently on their system. If there is a later version available, the update program automatically downloads the new file and replaces the old file. Then, a short text file pops up explaining the new changes. This system allows the user to be kept constantly up to date without having to constantly download files and replace old ones themselves, as it is run automatically every two weeks without any intervention.

The system is also highly expandable. Additional programs can be written in any language, as long as it caT BT 12 0 0 12 104i6393.3594 322.76Tm /Tc2 89.99937 295.1 1 Tf (l) Tj ET BT

***Us***

school is also looking to play against another local sch

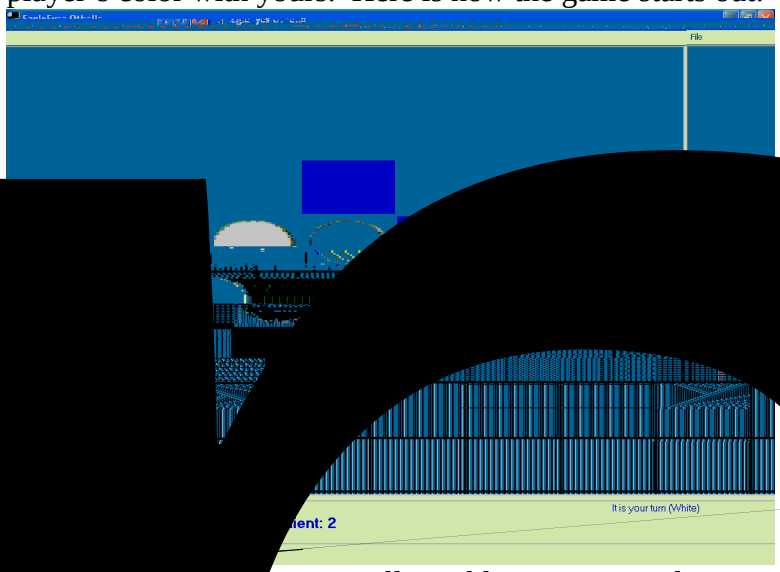
## Pong

Pong is the classic game where

## Con

## Othello

This is a game where you try to take over the board with your color (you can be white or black). You do this by clicking on a square that sandwich any number of the other player's color with yours. Here is how the game starts out:




n it is will see blue squares indicating valid
any blue squares until it is their turn. Because t
available. By clicking o

The program must also save its current version so the update program can easily find it. To do so, in the Form_Load sub, insert the following line:

This adds a line to the registry with the current Major, Minor, and Revision version numbers.

## Where to put the applications

All of the applications must be in the same directory – the ChatClient program will attempt to launch the programs by simply taking its path, and appending the name of the program. Also, this structure is important for uploading the files to the

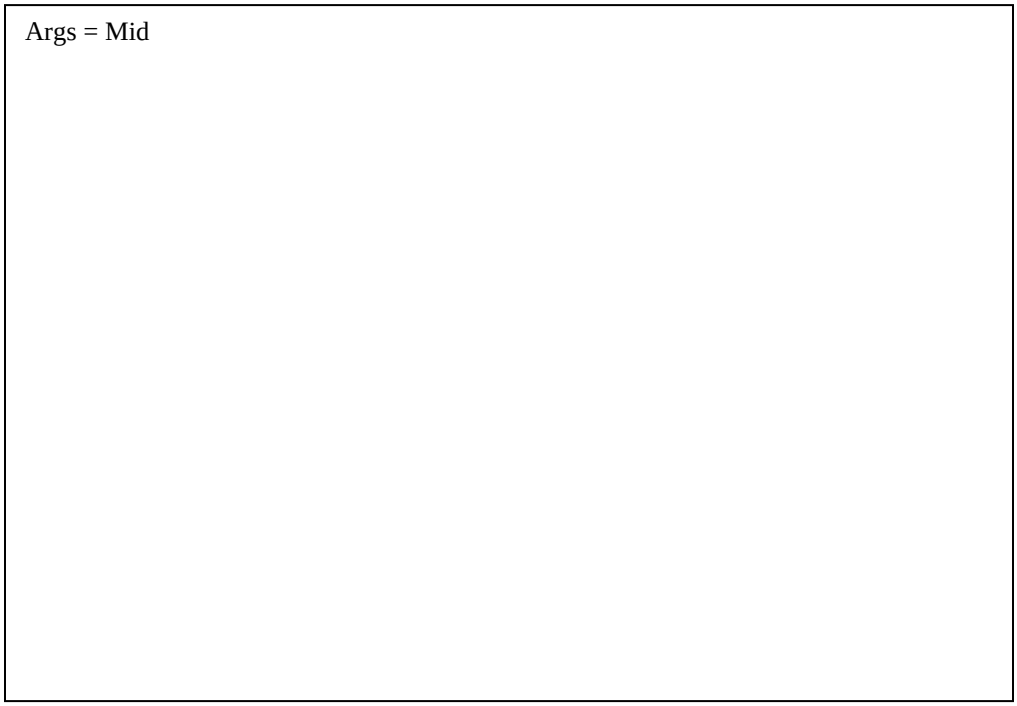
h

# Adding on to EagleEyes Connect – new program steps

Once you have put in the functionality into the ChatClient, you can now begin to add the network ability to your program. There are several steps to this. First, you must parse the command line arguments that were passed to the program via the ChatClient. You can simply copy the code below, which will handle splitting up the arguments into their proper elements. In the Form_Load sub, add the line: Call ProcessCommandLine(Command). This passes the command line arguments to the ProcessCommandLine sub, which is shown here:

```
Sub ProcessCommandLine(Args As String)
Dim CurArg As String, NextComma As Long, Port As Integer

NextComma = InStr(1, Args, ",") 'finds arguments seperated by spaces
If (NextComma = 0)áThen
   MultiGame = False
```




The majority of this code will remain the same for each program that is added, as all it is doing is creating a connection between the host and client programs and setting variables letting each program know what they are.

## Processing the Winsock Data

Once the data is split into t

# EagleEyes Connect Update pro