IJCAT International Journal of Computing and Technology, Volume 1, Issue 3, April 2014
ISSN : 2348 - 6090
www.IJCAT.org

Laser Guided Tic Tac Toe

[1] Chetan Bondre, [2] Deepak Kapgate , [3] Rohan Ponkshe

[1] PG Scholar, CSE GHRAET, Nagpur, Maharashtra, India

[2] Professor, CSE GHRAET, Nagpur, Maharashtra, India

[3] PG Scholar, CSE GHRAET, Nagpur, Maharashtra, India

Abstract - Image processing and games programming may be the most technically challenging job that a programmer can have. The top level programs may require the best from both programmer and computer. Tic-tac-toe is a simple game that makes it easy to build algorithms for the computer to play its own move. In this report, a interactive webcam based Tic Tac Toe game is developed. Program in coded in C# .NET using A Forge .NET Framework for certain image processing tasks and image acquisition. The Laser Guided Tic Tac Toe consists of an automatic segmentation system that is based on the Hough transform, and is able to localise the circular laser and positions. The extracted laser region was then normalised into a rectangular block with constant dimensions to account for imaging inconsistencies. The program searches for the brightest red blob in webcam's field of views. We can do so by using two of the filters defined in A Forge .NET Framework. Color filter to filter out everything except red laser light. Blob Counter to retrieve the position of laser dot or red blob within image. After laser has been detected, the program keeps track on movement of laser dot and once laser is turned off, it places the user's move within that grid.

Keywords **- C# .NET, hough Transform, Color filter, blob counter, automatic segmentation system.**

1. Introduction

Image processing and games programming may be the most technically challenging job that a programmer can have. The top level programs may require the best from both programmer and computer. Tic-tac-toe is a simple game that makes it easy to build algorithms for the computer to play its own move. In this project, an interactive webcam based Tic Tac Toe game is developed. Program in coded in C# .NET using A Forge .NET Framework for certain image processing tasks and image acquisition. The Laser Guided Tic Tac Toe consists of an automatic segmentation system that is based on the Hough transform, and is able to localise the circular laser and positions. The extracted laser region was then normalised into a rectangular block with constant dimensions to account for imaging inconsistencies.

The program searches for the brightest red blob in webcam's field of views. We can do so by using two of the filters defined in A Forge .NET Framework. Color filter to filter out everything except red laser light. Blob Counter to retrieve the position of laser dot or red blob within image[6][7].

After laser has been detected, the program keeps track on movement of laser dot and once laser is turned off, it places the user's move within that grid. The algorithm is designed so that it splits every frame from webcam into nine equal rectangles depicting nine different grids of tic tac toe game. User move is placed within grid in which laser light is turned off.

Tic-tac-toe is a simple game that makes it easy to build algorithms for the computer to play its own move. In this project, an interactive webcam based Tic Tac Toe game is developed. Since the user is going to play against the computer, the algorithm followed by the computer should be as human as possible. The project is an implementation of the various image processing techniques and algorithms. The crucial part of the project is the successful detection or tracking of the user input under different conditions of illumination. Also, it is expected that the software runs successfully with different fps and resolutions of the web cameras.

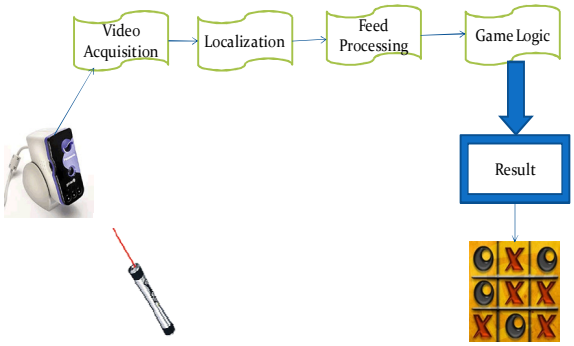

Fig 1: Steps in system for laser guided tic tac toe

2. Laser Detection & Gameplay

The program searches for the brightest red blob in webcam's field of views. We can do so by using two of the filters defined in AForge .NET Framework.

1. Color filter to filter out everything except red laser light.
2. Blob Counter to retrieve the position of laser dot or red blob within image.

2.1 Color Filter

Color filters have many uses in gemology. The Chelsea Colour Filter™ (CF) is the most prominent of them. While the CF is the most used, other filters can serve for a variety of applications. Among those are diffused colored plates used in conjunction with a microscope to inspect sapphires, narrow bandwidth filters to determine dispersion, and blue, red or yellow filters to examine fluorescence in gemstones. Some people regard the CF as a primary tool[1][4][5], yet all modern gemology writers disagree with that statement. It can, however, give clues to the identity of a gemstone when used as an additional (secondary) tool. One can never rely on observations with any color filter alone.

Most color filters show their real power when inspecting parcels of gemstones. When one examines, for instance, a parcel of blue sapphires with a CF, some stones would stand out bright red from the rest of the lot. One should immediately be very suspicious of those appearing red when viewed through a CF, as this is not a typical observation for sapphire. As these filters are highly portable, they are used on locations where one could not take larger gemological equipment. Gem/mineral shows and flea markets are some of those locations. At best, the filters are there to alert suspicion. They tell you little about the identity of the gemstone. All color filters work on the same principle: they absorb certain wavelengths of light, letting only a portion of the visible part of the electromagnetic spectrum pass through. In the image on the right, all colors of the visible range except red are absorbed (blocked) by a color filter, thus letting only red light pass through. This lets us control light, which can be very useful when inspecting gemstones.

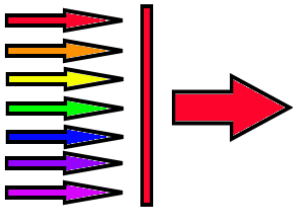

Fig 2: Beams of laser

2.2 Blob Counter

Blob counter is a very useful feature and can be applied in many different applications. What does it do? It can count objects on a binary image and extract them. The idea comes from "Connected components labeling," a filter that colors each separate object with a different color[8][10].

The Blob Counter filter counts all blobs (objects or contours) within an image. The filter contains two filters for blobs. One filter excludes all blobs outside a minimum and maximum size. The second filter excludes all blobs whose seven "hu invariants" are outside a minimum and maximum value. Hu invariants are proved to be invariants to the image scale, rotation, and reflection except the seventh one, whose sign is changed by reflection.

Here are two images: initial image and colored image. So, it looks like the filter is really able to count objects.

$$\partial_t L = \frac{1}{2}\nabla^2 L$$

$$\nabla^2_{norm} L(x,y;t) \approx \frac{t}{\Delta t}\left(L(x,y;t+\Delta t) - L(x,y;t-\Delta t)\right)$$

Figure 3: Blob Counter effect

It follows that the Laplacian of the Gaussian operator $\nabla^2 L(x,y,t)$ can also be computed as the limit case of the difference between two Gaussian smoothed images[12].

3. Strategy

A player can play perfect tic-tac-toe (win or draw) given they move according to the highest possible move from the following table.

1. Win: If the player has two in a row, play the third to get three in a row.
2. Block: If the [opponent] has two in a row, play the third to block them.
3. Fork: Create an opportunity where you can win in two ways.
4. Block opponent's Fork:
    - Option 1: Create two in a row to force the opponent into defending, as long as it doesn't result in them creating a fork or winning. For example, if "X" has a corner, "O" has the center, and "X" has the opposite corner as well, "O" must not play a corner in order to win. (Playing a corner in this scenario creates a fork for "X" to win.)
    - Option 2: If there is a configuration where the opponent can fork, block that fork.
5. Center: Play the center.
6. Opposite corner: If the opponent is in the corner, play the opposite corner.
7. Empty corner: Play in a corner square.
8. Empty side: Play in a middle square on any of the 4 sides.

4. Development Tools

Image processing and games programming may be the most technically challenging job that a programmer can have. The top level programs may require the best from both programmer and computer. Tic-tac-toe is a simple game that makes it easy to build algorithms for the computer to play its own move. In this article, a interactive webcam based Tic Tac Toe game is developed. Program in coded in C# .NET using AForge .NET Framework for certain image processing tasks and image acquisition.

4.1 C# .NET

C# is a multi-paradigm programming language encompassing strong typing, imperative, declarative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines. It was developed by Microsoft within its .NET initiative and later approved as a standard by Ecma (ECMA-334) and ISO (ISO/IEC 23270). C# is one of the programming languages designed for the Common Language Infrastructure.

C# is intended to be a simple, modern, general-purpose, object-oriented programming language. Its development team is led by Anders Hejlsberg. The most recent version is C# 4.0, which was released on April 12, 2010.

4.2 .NET Framework

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 which allows language interoperability (each language can use code written in other languages). Programs written for the .NET Framework execute in a software environment (as contrasted to hardware environment), known as the Common Language Runtime (CLR), an application virtual machine that provides important services such as security, memory management, and exception handling. The class library and the CLR together constitute the .NET Framework[2][9][11].

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 a popular integrated development environment largely for .NET software called Visual Studio.

4.3 AForge .NET

AForge.NET is a computer vision and artificial intelligence library originally developed by Andrew Kirillov for the .NET Framework. The source code and binaries of the project are available under the terms of the Lesser GPL license.

Features

- Computer vision, Image processing and Video processing,
- Neural networks.
- Genetic programming.
- Fuzzy logic.

4.4 Windows Presentation Foundation

Developed by Microsoft, the Windows Presentation Foundation(or WPF) is a computer-software graphical subsystem for rendering user interfaces in Windows-based applications. WPF, previously known as "Avalon", was initially released as part of .NET Framework 3.0. Rather than relying on the older GDI subsystem, WPF utilizes DirectX. WPF attempts to provide a consistent programming model for building applications and provides a separation between the user interface and the business logic. It resembles similar XML-oriented object models, such as those implemented in XUL SVG.

WPF employs XAML, a derivative of XML, to define and link various UI elements. WPF applications can also be deployed as standalone desktop programs, or hosted as an embedded object in a website. WPF aims to unify a number of common user interface elements, such as 2D/3D rendering, fixed and adaptive documents, typography, vector graphics, runtime animation, and pre-rendered media. These elements can then be linked and manipulated based on various events, user interactions, and data bindings.

WPF runtime libraries are included with all versions of Microsoft Windows since Windows Vista and Windows Server 2008[7]. Users of Windows XP SP2/SP3 and Windows Server 2003 can optionally install the necessary libraries.

As of 2011 Microsoft has released four major WPF versions: WPF 3.0 (Nov 2006), WPF 3.5 (Nov 2007), WPF 3.5sp1 (Aug 2008), and WPF 4 (April 2010).
Microsoft Silverlight utilizes WPF to provide embedded web controls comparable to Adobe Flash, but with more focus on a UI object model and less on animation. 3D runtime rendering is supported in Silverlight since Silverlight 5[7][10[14].

5. Combinatories

Despite its apparent simplicity, Tic-tac-toe requires detailed analysis to determine even some elementary combinatory facts, the most interesting of which are the number of possible games and the number of possible positions. A position is merely a state of the board, while a game usually refers to the way a terminal position is obtained[1][2]. A naive count of the number of positions leads to 19,683 possible board layouts (3^9 since each of the nine spaces can be X, O or blank), and a similar count of the number of games leads to 362,880 (i.e. 9!) different sequences for placing the Xs and Os on the board. However, this doesn't take into account the fact that the game ends when three-in-a-row is obtained. Many of the 19,683 positions are unreachable in an actual game. The complete analysis is further complicated by the definitions used when setting the conditions, like board symmetries.

Number of terminal positions
When considering only the state of the board, and after taking into account board symmetries (i.e. rotations and reflections), there are only 138 terminal board positions. Assuming that X makes the first move every time:

- 91 unique positions are won by (X)
- 44 unique positions are won by (O)
- 3 unique positions are drawn.

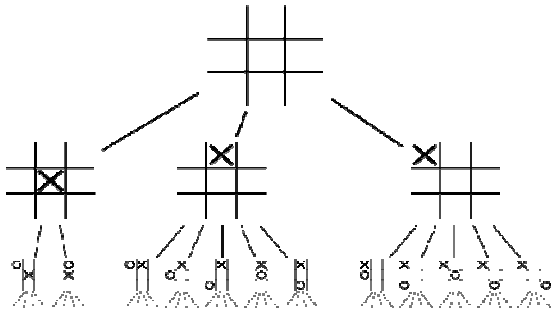

Figure 4: Tic tac toe logic

The first two plies of the game tree for tic-tac-toe. Once rotations and reflections are eliminated, there are only three opening moves - a corner, a side or the middle.

6. Future Work

Controlling Computer with LASER

This can be drastically and easily improved with the use of a diffraction grating (or you could use a convex lens) placed in front of the laser pointer. You can buy laser pointers with a "star lens" (diffraction grating) that work very well with this. The star lens will spread the laser out into many (hundreds of) points with each able to interact with the mouse. This way you can move the laser without really worrying if your dot is close enough.[6] If you use a convex lens (magnifying glass) it will spread the laser beam out to make a wider beam (but less bright); this method works too, but is not as good as with a diffraction grating. If you want the ability to click you will have to put the photo resistor far enough away from the mouse that it will not move the mouse and click at the same time.

Applications:

Controlling Computer with LASER

This can be drastically and easily improved with the use of a diffraction grating (or you could use a convex lens) placed in front of the laser pointer. You can buy laser pointers with a "star lens" (diffraction grating) that work very well with this. The star lens will spread the laser out into many (hundreds of) points with each able to interact with the mouse. This way you can move the laser without really worrying if your dot is close enough. If you use a convex lens (magnifying glass) it will spread the laser beam out to make a wider beam (but less bright)[5][7][14]; this method works too, but is not as good as with a diffraction grating. If you want the ability to click you will have to put the photo resistor far enough away from the

mouse that it will not move the mouse and click at the same time[3][4].

Implementation of a Laser-based Interaction Technique for Projection Screens

The Interactive Wall forms a 6.5 x 1.6m continuous workspace (see Figure 1). Its dimensions offer users a generous presentation space, as well as various forms of interaction in support of group project applications. Contemporary interaction devices, including pointing devices like the mouse, trackball, trackpoint and touchpad, which operate on the principle of relative motion, deliver less than satisfactory results[3][13].

The user cannot directly address co-ordinates, but must instead move the pointer from its current position to the target position through relative motion of the device. On large projection surfaces such as the Interactive Wall described here, the disadvantages of relative-motion pointing devices become instantly visible.
It takes too long to select widely separated objects, and relative-motion pointing is contradictory to the intuitive use of a projection screen. Since the Interactive Wall works much like a blackboard, the pen metaphor is more appropriate.

7. Conclusion

Laser guided Tic Tac Toe is an innovation in getting input from the users and this input can be successfully used for performing different computing tasks. One of the project goals was simplicity – making it easy for handling and understanding. The project can be extended to further applications such as performing OS tasks through laser and so on. The success of the application fairly depends on how accurately the red laser is tracked which in turn depends on the illumination of the room.

We believe an optimal solution for laser control is a combination of direct mode and background light. The direct mode can navigate the laser beam to the area of interest on screen, then background light can be triggered to fine tune the beam location according to user's desire. The two modes can be switched on/off by the detection of some non-rigid deformation.

References

[1] T. Lindeberg (2008/2009). *Scale-Space*. "Scale-space". *Encyclopedia of Computer Science and Engineering (Benjamin Wah, ed), John Wiley and Sons* **IV**: 2495-2504. doi: 10.1002/9780470050118. ecse609. ISBN 0-470-05011-X.

[2] T. Lindeberg (2012). "Scale invariant feature transform". *Scholarpedia*: 7(5):10491. doi:10.4249/scholarpedia.10491.

[3] D. G. Lowe (2004). "Distinctive Image Features from Scale-Invariant Keypoints". *International Journal of Computer Vision* **60** (2): pp 91–110. doi:10.1023/B:VISI.0000029664.99615.94.

[4] J. Matas, O. Chum, M. Urban and T. Pajdla (2002). "Robust wide baseline stereo from maximally stable extremum regions". *British Machine Vision Conference*. pp. 384–393.

[5] US patent 3971065, Bryce E. Bayer, "Color imaging array", issued 1976-07-20

[6] R. Lukac and K. N. Plataniotis, "Data-adaptive filters for demosaicking: A framework," IEEE Transactions on Consumer Electronics, vol. 51, no. 2, pp.560-570, May 2005.

[7] R. Lukac and K. N. Plataniotis, " Universal demosaicking for imaging pipelines with an RGB color filter array," Pattern Recognition, vol. 38, no. 11, pp. 2208-2212, November 2005.

[8] W. Lu and Y. P. Tang, "Color filter array demosaicking: new method and performance measures," IEEE Transactions on Image Processing,
vol. 12, no. 10, pp. 1194-1210, October 2003.

[9] J. Adams, K. Parulski, and K. Spaulding, "Color processing in digital cameras," IEEE Micro, vol. 18, no. 6, pp. 20-30, Nov./Dec. 1998.

[10] Vosselman, G., Dijkman, S: "3D Building Model Reconstruction from Point Clouds and Ground Plans", International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol 34, part 3/W4, October 22–24, 2001, Annapolis, MA, USA, pp.37- 44.

[11] H.C. Andrews (Ed), Digital Image Processing - IEEE Press, New York 1978

[12] Computer Vision: Algorithms and Applications: Richard Szeliski Springer, 2010.

[13] D.H. Ballard, C.M. Brown; Computer Vision, Prentice-Hall Inc New Jersey, 1982

[14] J. L. Starck, F. Murtagh, A. Bijaoui; Image Processinf and Data Analysis: The Multiscale approach, Cambridge University Press, 1998.