Touchless SDK Documentation

Homepage: http://www.codeplex.com/touchless

Project Author: Michael Wasserman

Michael.Wasserman@microsoft.com

Thanks due to: Gary Caldwell, Isabel Mattos, Nicole Steinbok, Quinn Hawkins, MS Office Labs Community Projects, John Kender, and you!

0. Table of contents

1. What's Touchless?

Touchless is an SDK that allows users to create and experience multi-touch applications. Touchless started as Mike Wasserman's college project at Columbia University. The main idea: to offer users a new and cheap way of experiencing multi-touch capabilities, without the need of expensive hardware or software. All the user needs is a camera, which will track colored markers defined by the user.

Mike presented the project at the Microsoft Office Labs Productivity Science Fair, Office Labs fell in love with it, and Touchless was chosen as a Community Project. Our deliverables include an extensible demo application to showcase a limited set of multi-touch capabilities, but mainly we are delivering an SDK to allow users to build their own multi-touch applications.

Now, Touchless is released free and open-source to the world under the Microsoft Public License (Ms-PL) on CodePlex. Our goals are to drive community involvement and use of the SDK as it continues to develop.

Remember that this is just the beginning; and you're invited to join our journey. Send us

your questions and feedback, use Touchless SDK in your .NET applications and XNA games, and support the community by contributing to the source code.

Unleash the power of your webcam!

2. What's in this release?

This release includes a short list of binary files to demonstrate Touchless:

- WebCamLib.dll - Interfaces with DirectShow to grab webcam frames
- TouchlessLib.dll - Contains the functionality of Touchless SDK
- TouchlessDemo.exe - A small demonstration of what Touchless can be used for
- Image.gif - Used in TouchlessDemo's image demo
- Touchless.chm - A documentation file of the Touchless API for developers
- Touchless.jpg - A class diagram of the Touchless API for developers
- Touchless.rtf - This file; you're reading it right now!

If you just want to try out the simple demos offered with the SDK, extract the release to any location on your computer, and run "TouchlessDemo.exe".

3. What's in the source repository?

The source repository is hosted at the homepage of Touchless:

http://www.codeplex.com/touchless

The directories you'll find there are:

- bin - Binary versions of the files, updated more frequently than our releases
- Documentation - Browse media documenting the project
- Samples - Apps/Games that demonstrate or use Touchless (submit yours!)
- TouchlessLib - Touchless functionality, written in C#, uses .NET 3.0+, VS2005+

- TouchlessUnitTests - Unit tests to prevent regression
- WebCamLib - Webcam functionality, written in C++, uses DX SDK Aug. 2007

Become a contributor by adding your Touchless apps or improving the SDK code!

4. How to add Touchless to your project

By adding Touchless to your project, you can give your users a truly fun, novel, and functional multi-touch experience, and all they need is a webcam!

The prerequisites you'll need to develop using Touchless are:

- Visual Studio 2005 or 2008
- .NET 3.0 or higher
- "TouchlessLib.dll" and "WebcamLib.dll"

To add Touchless to an existing visual Studio project, simply right click “References” and select “Add Reference…” go to the browse tab, and select "TouchlessLib.dll". Ensure that both "TouchlessLib.dll" and "WebcamLib.dll" are copied to the same output directory to be used with your builds.

5. Reference Information

You can find extensive information on all properties and methods of each class in our XML-generated documentation file “TouchessLib.chm”. The Touchless class diagram reflects public classes, methods, and properties of the API; it is avaiable as "Touchless.jpg". These files are distributed with each release, and also available in the source code repository, located in the documentation folder.

6. Backlog of Ideas

This is a very terse list of ideas for improving Touchless SDK.

- **ColorLib**
    - Improve HSV colorspace partitioning model. We could group perceived similar colors better. Potentially replace with a group clustering algorithm. Perhaps just

refine the per-dimension bin counts, or replace the hash function.
    - Use a lookup table instead of transforming RGB to HSV. We can just terminate early if it's not in the lookup table.
    - Reduce loop overhead of converting ARGB values into RGB values, then into HSV values, then into Binned HSV values, then finally into a hash # for color lookup during marker update. Potentially use a lookup table for a subset of colors to avoid the math altogether.
    - Improve HSV color grouping, consider refining the per-dimension bin counts or using a different HSV color-space partitioning model that better suits human perception of similar colors.

- **Marker**
    - Implement a way of getting higher degree moments of inertia. Mostly, we are interested in the axis of least rotational momentum and the roundness factor.
    - Allow the user to send a mask image with the add marker bitmap for arbitrary marker region selections."
    - Extend or replace alpha smoothing with exponential decay to provide smoothed marker data and reduce the marker jumpiness.
    - Optimize threshold, or replace threshold concept with a partial matching. Also, step threshold by numbers that actually make a difference, or just have sensitivity +/- buttons and increment functions
    - Expose smoothing factor as a public marker property.
    - Fix and improve the automated marker tests
    - Standardize some marker colors, create an "auto-find makers"
    - Improve the search bounds of a previously absent marker.
    - Improve the meta-tracking (cases where small numbers of pixels are missing from the middle of a marker, or are outliers of the concentration of pixels)
    - Periodically/continuously adopt surrounding pixels of confirmed marker pixels
    - Coloravg is currently just marker representative color. Implement a way of actually getting a color average from the set of colors found
    - Improve Marker highlighting
    - Improve upon the raster scan algorithm used for marker updating.
    - Optimize the method for getting the marker appearance from a circular area of a bitmap; we could use hierarchical bounds intersection or something smarter than the current scan algorithm.
    - Optimize the values used to increment/decrement color frequencies for marker appearance detection. This should be somehow based on signal/noise ratios.
    - Improve the expected marker regions used for scanning on update. We could consider the marker's acceleration, rather than just the velocity. Perhaps try using regions that aren't axis-aligned rectangles.

- **TouchlessMgr**
    - Add functionality to save and load marker configuration files (reduce repeat

training of the same marker, possibly provide autoconfig files for standard markers... will variant lighting allow for this?)

- Implement additional marker data such as ColorAverage, ColorSpace, Axis and Roundness.
- Add flood fill algorithm so we can add a marker with a few points in the Bitmap.
- Refine the marker tracked colors as we find colors around the marker.
- The representative color doesn't always match the perceived color of the marker.
- Provide subsequent examples of a marker appearance
- Have TouchlessMgr actually expose a way to get a list of the current markers
- Make a better exception for camera start failure
- Validate the PixelFormat of incoming images.
- Create a utility function to retrieve ImageData in a consistent manner; we have a bit of code duplication right now.
- Make a public interface for demo classes to implement, then allow the user to just invoke start and stop of a demo class on the library
- Standardize error handling and exception generation across the project.