"Free hand Image Control System"

Miss. Shweta Soni

Student, Computer Science, GHRIETW, Nagpur (India)
shwetaghrietw@gmail.com

Abstract- As the personal computing industry pursues more user friendly, inexpensive user interfaces the concept of a touch less interface is worthy of inspection. Touch less is an SDK (software development kit) that allows users to create and experience multi-touch applications. The main idea is 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.

This next generation input technology allows the user to navigate without touching the surface. Computers are no longer relegated to use in the home study or on an office desk. These days people travel everywhere with their smart handsets, personal media players, e-books and tablet PCs. Coffee shops, restaurants, gyms, bus stops, plane terminals and even lavatories are fair usage environments for this new generation of touchless interface. If a customer is reading an e-book at the gym while on a treadmill and wants to turn a page, it would be a much easier to swipe across the device with a touchless gesture. A touch less interface can allow an automobile driver to safely adjust volume with the touch less swipe of hand without having to navigate through a complicated instrument cluster to find control buttons. The day will soon come when even the most commonplace home appliance, handheld devices, computing platform and industrial interface can be activated and controlled with the wave of a hand.

Index Terms- marker, touch less interface, HSV colorspace

I. Introduction

As the personal computing industry pursues more user friendly, inexpensive user interfaces the concept of a touch less interface is worthy of inspection. Touch less is an SDK (software development kit) that allows users to create and experience multi-touch applications. The main idea is 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. This next generation input technology allows the user to navigate without touching the surface.

Human-machine interaction has evolved significantly over the past decade through enhancements in user interfaces and smart design. Many of these changes have focused around touchscreen interfaces with high-precision, low-power capacitive touchscreens at the forefront particularly in the handset market. Now, through advancements in human interface (HI) technology and design, touch less gesturing is poised to usher in the next user interface innovations. Not all devices have or need complex graphical displays with touchscreen either, and for such devices a touch less interface can provide an innovative and differential approach for operation.

Small scale video advertising billboards within public spaces can change the context of their messages based on whether someone is near or far away and then use touch less gesture input to interact with the potential customer. Such "environment aware" electronics can enable smarter end-products that are simultaneously more energy-efficient.

II.ANALYSIS AND DESIGN

We designed our project and made a descriptive class diagram showing all the classes with its properties and methods.

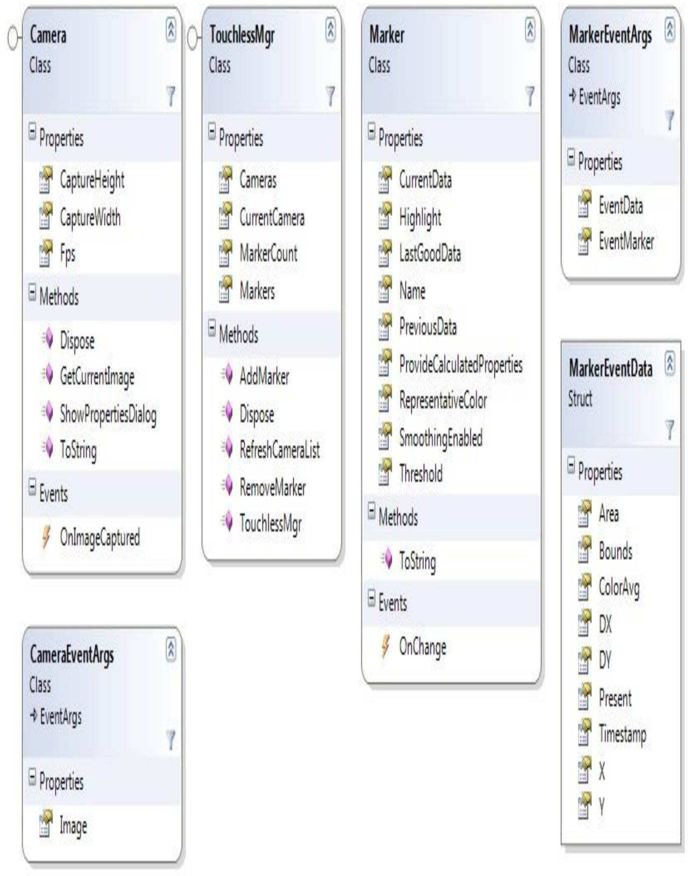

Fig. 1. Class diagram

MODULE 1: CAMERA

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.

MODULE 2: TOUCH LESS MGR

Add functionality to save and load marker configuration files (reduce repeat training of the same marker, possibly provide auto configured 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. Let Touch less Mgr actually expose a way to get a list of the current markers. Make a better exception for camera start failure. Validate the Pixel format of incoming images. Create a utility function to retrieve image data in a consistent manner, we have a bit of code duplication right now. Make a public interface for demo classes to implement and 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.

MODULE 3: 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". Also 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.

SUB-MODULES:

SCROLL

In computer graphics applications such as filmmaking, television production, and other kinetic displays, **scrolling** is sliding text, images or video across a monitor or display. "Scrolling", as such, does not change the layout of the text or pictures, but incrementally moves (pans or tilts) the user's view across what is apparently a larger image that is not wholly seen. A common special effect is to scroll credits, while leaving the background stationary. **Smooth scrolling** is a feature to reduce what the viewer would perceive as "jumps" (discontinuous movement) in the display. The computational effort of moving images and video smoothly is high , therefore successful smooth scrolling in text is most common. Frame rate is speed at which an entire image is redisplayed. It is related to scrolling, in those changes to text and image position can only happen as often as the image can be redisplayed. When frame rate is a limiting factor, one smooth scrolling technique is to blur images during movement that would otherwise appear to "jump".

RESIZE

In computer graphics, **image scaling** is the process of resizing a digital image. Scaling is a non-trivial process that involves a trade-off between efficiency, smoothness and sharpness. As the size of an image is increased, so the pixels which comprise the image become increasingly visible, making the image appears "soft". Conversely, reducing an image will tend to enhance its smoothness and apparent sharpness.

Apart from fitting a smaller display area, image size is most commonly decreased (or sub sampled or down sampled) in order to produce thumbnails. Enlarging an image (up sampling or interpolating) is generally common for making smaller imagery fit a bigger screen in full screen mode, for example. In "zooming" an image, it is not possible to discover any more information in the image than already exists, and image quality inevitably suffers. However, there are several methods of increasing the number of pixels that an image contains, which evens out the appearance of the original pixels. zoom is a method of decreasing (narrowing) the apparent angle of view of a digital photographic or video image. Digital zoom is accomplished by cropping an image down to a centred area with the same aspect ratio as the original, and usually also interpolating the result back up to the pixel dimensions of the original. It is accomplished electronically, with no adjustment of the camera's optics, and no optical resolution is gained in the process.

When comparing the image quality achieved by digital zoom with image quality achieved by resizing the image in post-processing, there's a difference between cameras that perform potentially lossy image compression like JPEG, and those that save images in an always lossless Raw image format. In the former case, digital zoom tends to be superior to enlargement in post-processing, because the camera may apply its interpolation before detail is lost to compression. In the latter case, resizing in post production yields results equal to or superior to digital zoom. Some digital cameras rely entirely on digital zoom, lacking a real zoom lens, as on most camera phones. Other cameras do have a real zoom lens, but apply digital zoom automatically once its longest focal length has been reached. Professional cameras generally do not feature digital zoom.

ROTATE

The rotation operator performs a geometric transform which maps the position (x_1, y_1) of a picture element in an input image onto a position (x_2, y_2) in an output image by rotating it through a user-specified angle θ about an origin O. In most implementations, output locations (x_2, y_2) which are outside the boundary of the image are ignored. Rotation is most commonly used to improve the visual appearance of an image, although it can be useful as a preprocessor in applications where directional operators are involved. Rotation is a special case of affine transformation.

The rotation operator performs a transformation of the form:

$$x_2 = cos(\theta) * (x_1 - x_0) - sin(\theta) * (y_1 - y_0) + x_0$$
$$y_2 = sin(\theta) * (x_1 - x_0) + cos(\theta) * (y_1 - y_0) + y_0$$

where (x_0, y_0) are the coordinates of the center of rotation (in the input image) and θ is the angle of rotation with clockwise rotations having positive angles. (Note here that we are working in image coordinates, so the y axis goes downward. Similar rotation formula can be defined for when the y axis goes upward.) Even more than the translate operator, the rotation operation produces output locations (x_2, y_2) which do not fit within the boundaries of the image (as defined by the dimensions of the original input image). In such cases, destination elements which have been mapped outside the image are ignored by most implementations. Pixel locations out of which an image has been rotated are usually filled in with black pixels.

III.APPLICATIONS

A valid challenge to touch less interfaces is why they should be implemented at all. Why do away with tactile buttons and touchscreens if they work? Infrared systems are not going to replace existing systems, but instead they are going to augment the user experience. Increased integration and miniaturization are changing the way customers use electronics products. No longer are “computers” relegated to use in the home study or on an office desk. These days people travel everywhere with their smart handsets, personal media players, e-books and tablet PCs. Coffee shops, restaurants, gyms, bus stops, plane terminals and even lavatories are fair usage environments for this new generation of embedded electronics. In such diverse operating environments, users’ hands are sometimes occupied, dirty, sweaty or covered in food -- all conditions not conducive to touchscreen operation. If a customer is reading an e-book at the gym while on a treadmill and wants to turn a page, it would be a much easier to swipe across the device with a touch less gesture to turn the page rather than physically contacting a touchscreen or hunting down a small button For example, a touch less interface can allow an automobile driver to safely start/end a call or adjust volume with the touch less swipe of a hand without having to navigate through a complicated instrument cluster to find control buttons. Not all devices have or need complex graphical displays with touchscreens either, and for such devices a touch less interface can provide an innovative and differentiated approach for operation.

IV.FUTURE SCOPE

No longer are “computers” relegated to use in the home study or on an office desk. These days people travel everywhere with their smart handsets, personal media players, e-books and tablet PCs. The Touch less SDK is a set of .Net components that can be used to simulate the gestural interfaces of many devices using nothing fancier than an ordinary USB Webcam. Mouse movements and mouse click movements can be developed and these can be implemented for various gaming application. If the sensor could be made small enough, it may even find its way into cell phones and mobiles. The technology requires no special hardware and uses the standard camera that is already built into the advanced models to control functions and applications such as calls, music and video players, games, web browsing and other usability options.

V.CONCLUSION

Human-machine interaction has evolved significantly over the past decade through enhancements in user interfaces and smart design. Many of these changes have focused around touchscreen interfaces with high-precision, low-power capacitive touchscreens at the forefront particularly in the handset market.

Now, through advancements in human interface (HI) technology and design, touch less gesturing is poised to usher in the next user interface innovations. And the day will soon come when even the most commonplace home appliance, handheld devices, computing platform and industrial interface can be activated and controlled with the movement of a hand.

VI. ACKNOWLEDGEMENT

It brings us a great pleasure in submitting this paper with a deep sense of gratitude & profound respect, We would like to thank our parents and our lecturers for the valuable guidance & constructive suggestions which helped us in preparing this paper. Last, but not least, we will be very thankful to all our friends who co-operated us in one way or the other in making this paper

REFERENCES

[1]. DOT.NET by Evangelos Petroutsos

[2]. Visual Studio .NET - Beginning .NET Database Programming by Wrox

[3]. Mastering Visual Basic .NET Database Programming

[4]. VB .NET Developer's GUIde by john Paul Mueller

[5]. A Programmer’s Introduction to Visual Basic.Net by Ed Robinson

[6]. Developing Applications with Visual Studio.Net by Richard Grimes

[7]. http://www.csharp-station.com

AUTHOR

Author – Shweta Soni, Mtech-3rd sem(CSE), GHRIETW
email-id shwetaghrietw@gmail.com.