BU ProPane Project Background

BU ProPane Team:
Griffin Dunn
Colin Madigan
Phillip Stahlfeld

October 12, 2012

The purpose of this document is to introduce the BU ProPANE project to a third party with no prior knowledge concerning the project. The document begins with a high-level discussion of the problems that this project is aiming to solve. Next, the document provides relevant background information regarding current practices and relevant terminology. The document concludes with a technical research section that contains ideas that will prove useful during the implementation phase of the project.

Contents

1 Problem Statement

The goal of this project is to create a system that captures all of the information written on a board during a class in a readily accessible manner. The two driving forces behind solving this problem are: autonomous collection of notes for students with disabilities and providing a means for professors to compare their notes with the actual information presented during a lecture.

Current practices at Bucknell rely on the aid of a student volunteer to provide photocopies of his or her notes to students who are incapable of taking notes themselves during the class. This is not an optimal solution because: the student volunteer must be in class every day, turnaround time for the notes is too long, and the quality of the student volunteer notes cannot be guaranteed. In addition, several professors use different colored writing to represent different forms of information. If the student volunteer did not copy the notes in color, then any students using the notes would not have access to the same information.

Typically, professors have a lecture planned out before class starts and have notes prepared to follow their plan. However, when students ask unexpected questions or do not fully grasp the material, it is easy to deviate from this plan. For example, civil engineers at Bucknell University plan out their lectures by dividing a piece of paper into sixths and filling in each block with what they want to present (See Figure 1). Problems arise when information not included on these pieces of paper are written on the board (due to student questions etc.). When the professor is preparing to give the lecture the following year, he or she will have to remember what the previous year's students struggled with. Using a board capture system, professors could compare previous lectures with previous lecture notes to generate a more effective lecture plan.

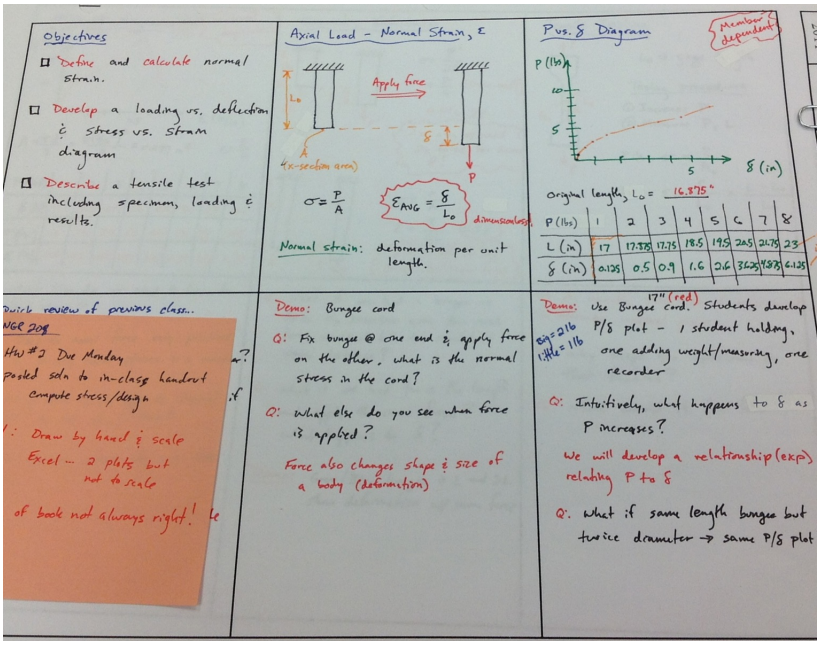

Figure 1: Image of civil engineering lecture notes. Orange note is an example of how lecture notes are recorded and altered. Image courtesy of Doug Gabauer 2012.

2 Background Information

Signal Processing

Source: http://www.signalprocessingsociety.org/about-sps/scope-mission/

According to the IEEE Signal Processing Society,

Signal processing is the enabling technology for the generation, transformation, and interpretation of information. It comprises the theory, algorithms, architecture, implementation, and applications related to processing information contained in many different formats broadly designated as signals. Signal refers to any abstract, symbolic, or physical manifestation of information with examples that include: audio, music, speech, language, text, image, graphics...

Signal processing uses mathematical, statistical, computational, heuristic, and/or linguistic representations, formalisms, modeling techniques and algorithms for generating, transforming, transmitting, and learning from analog or digital signals, which may be performed in hardware or software. Signal generation includes sensing, acquisition, extraction, synthesis, rendering, reproduction and display. Signal transformations may involve filtering, recovery, enhancement, translation, detection, and decomposition. The transmission or transfer of information includes coding, compression, securing, detection, and authentication. Learning can involve analysis, estimation, recognition, inference, discovery and/or interpretation.

Signal processing applies to both analog and digital signals. Analog signals are sets of continuous values, and can be directly processed using both linear and non-linear electronic circuits. These signals can be sampled to obtain a signal with the same values, but only at discrete points over an interval in time. These signals can also be directly processed with electronic circuitry, but they can also be digitized to create digital signals. Digital signal processing can be done conveniently on computers, or on dedicated digital circuits such as application-specific integrated circuits (ASICs) and field-programmable gate arrays (FPGAs). For this project, we are interested in digital signal processing, and specifically the field of image processing.

Image Processing

Source: http://www.library.cornell.edu/preservation/tutorial/contents.html

Digital images are electronic representations of physical media, such as photographs, texts, artwork and more. The digital image is sampled and mapped as a grid of pixels. Each pixel is assigned a tonal value (black, white, shades of gray or color) which is represented in binary. The bits representing each pixel are stored by a computer and can be compressed via mathematical reduction of some kind. To view an image, the bits are interpreted and read by the computer to produce an analog version for display.

Digital images can be displayed with different resolutions. Resolution is the ability to display fine spatial detail. In general, resolution increases with sampling frequency. Common units of resolution are dots-per-inch (dpi) and pixels-per-inch (ppi).

The number of bits used to define each pixel is defined as the bit depth. The greater the bit depth, the greater the number of tones can be represented. Normally, a color image is represented by a bit depth of 8 to 24. In a 24-bit image, the bits are often divided into red, green and blue, with 8 bits for each color. Other colors are obtained from combinations of those bits. In a 24-bit image, there are 16.7 million color values (2^{24}).

To reduce image file size for storage, processing and transmission, compression is used. Compression techniques use algorithms to abbreviate the binary code in an uncompressed image. Both lossless and lossy forms of compression exist. Lossless schemes abbreviate the binary code without discarding any information so that the decompressed image is bit-for-bit identical to the original. Lossy schemes average and discard the least significant information based on an understanding of visual perception. JPEG is a common lossy compression scheme. Some lossy-compressed images may appear identical to the raw image, and are considered "visually lossless".

Image Processing Tools

Image processing can be implemented using a variety of different tools. There are programs, such as Adobe Photoshop, which provide users with a graphical user interface and on-screen tools to use to edit their images. While programs like this can achieve desirable high-quality results, the process can be time-consuming for an individual image, and must be repeated for each subsequent image. A far better option for similar processing of a large quantity of images is to use a programming language with an image processing library. The user can write a program which performs desired effecs on any number of images. A variety of libraries exist to add image processing features to many programming languages. The libraries listed in this section are for common programming languages and seem to have some or all of the functionality required for the ProPANE project.

Eutecus Signal and Image Processing Library
Source: http://www.eutecus.com/ProdServ/InstantVision/SILib.html

A collection of general-purpose, optimized C++ routines and classes, along with utility classes to aid image and video file manipulation. These routines are usually used in computationally intensive real-time apptications, where optimal execution speed is critical. For use, licensing rights must be purchased.

OpenCV
Source: http://opencv.willowgarage.com/wiki/

A free library of programming functions for real time computer vision, originally developed by Intel. It has interfaces for C, C++, Python, and soon Java, running on Windows, Linux, Android and Mac. There are over 2500 algorithms optimized for image processing.

Aforge.NET Framework
Source: http://code.google.com/p/aforge/

A free, open-source C# framework, containing libraries for image processing, robotics, and more. The image processing library, AForge.Imaging, contains different image processing routines, aimed to help with image enhancement and processing.

Python Imaging Library
Source: http://www.pythonware.com/products/pil/

A free library which adds image processing capabilities to the Python interpreter. It supports many file formats and provides powerful image processing and graphics capabilities.

VIPS
Source: http://www.vips.ecs.soton.ac.uk/index.php?title=VIPS

Image Processing Techniques

Source: http://www.ndt-ed.org/EducationResources/CommunityCollege/Radiography/AdvancedTechniques/Real_Time_Radiography/ImageProcessingTechniques.htm

Many techniques currently exist for processing images in numerous different ways. The techniques in this section seem relevant to the image processing necessary for the ProPANE system.

Image Analysis-
Extract information from the image

- **Image statistics:** Calculates the maximum, minimum, average, standard deviation, variance, median, and mean-square intensities of the image data
- **Gray-scale mapping:** Alters the mapping of the intensity of pixels in a file to match the intensity displayed on a computer screen
- **Image extraction:** Extracts a portion or all of an image and creates a new image with the selected area
- **Slice:** Plots intensity versus position for any direction. Lists intensity versus pixel location from any point along the slice

Convolution and Noise Filtering-
Decrease noise by diminishing statistical deviations

- **High-pass filter:** Emphasizes regions with rapid intensity changes
- **Low-pass filter:** Smoothes images, blurs regions with rapid change
- **Adaptive smoothing filter:** Sets pixel intensity to a value between original and mean values, corrected by degree of noisiness. Good for decreasing statistical, especiall signal-dependent noise
- **Median filter:** Sets pixel intensity to the median intensity of pixels in a neighborhood. Excellent for eliminating intensity spikes
- **Sigma filter:** Sets pixel intensity equal to the mean of the intensities in a neighborhood within two of the mean. Good for signal-independent noise

Edge Detection-
Sharpen intensity-transition regions

- **First-difference:** Subtracts intensities of adjacent pixels. Emphasizes noise as well as desired changes.
- **Sobel operator:** Weighs inner pixels twice as heavily as corner values. Calculates intensity differences

Enhancement-

- **Histogram equalization:** Redistributes the intensities of the image

Useful Terminology

WE WILL GET THIS WITH MORE RESEARCH

3 Research

Competing Technologies

There are three general categories: phone apps, scanners, and electronic whiteboards.

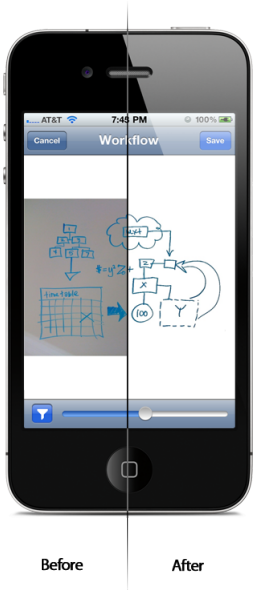

Figure 2: Advertisement demonstrating the image filtering capabilities of the WhiteBoard Capture Pro iPhone app. App eliminates discoloration of board due to glare. Taken from `http://www.beetlebugsoftware.com`.

Phone Apps-
Phone apps scan an image of a whiteboard and filter out the unnecessary information (See Figure 2). These apps compete directly with our product because they serve many of the same functions and are also highly portable. Portability aside, they also capture informaiton in the same ways that our product will, and can thus provide many valuble insitghts into the possabilities open to us as well. Several examples of phone apps are listed below.

- WhiteBoard Capture Pro
- WBConference
- CamScanner
- Whiteboard Snap

Scanners-
Scanners are pieces of hardware that attach to white boards that will tracks movements on the board (some with the usage of an electronic pen). These products will not be fully suited to this project because they require installation and calibration, which violates the idea of portability. Though they compete with our product because they perform the same function of recording everything that is written on a whiteboard, they don't capture the information in the same ways. While their innovations prove interesting from an academic perspective, they don't provide many insights towards the further development of our own products. Several examples of scanners include:

- MimioCapture
- eBeam System 3
- Interlink
- FreeBeam

Electronic Whiteboards-
Smartboards are full sized, touch sensitive whiteboards. You can project images onto them with a projector and then make edits/additions to them with any pointed object. The touch sensitive board senses where you are writing on the board and adds electronic corrections to the image/document.

These devices compete even less directly with our project than do scanners because they're less portable and capture information in even less applicable ways. They are the size of a standard whiteboard and thus cannot be easily moved between classrooms. Several examples of these touch sensitive whiteboards include:

- SMART Board
- Panasonic's Panaboard
- Hitachi's Starboard
- The Promethean Board

Research Data

Smartphone Applications

We chose to seek further information about smartphone apps because they are the technology that most directly competes with our product. While scanners and electronic whilteboards meet the same needs as our product, they do so in ways that are completely different from our own approach, and thus won't provide the numer of insights found in smartphone applications.

Introduction

The following paragraphs are descriptions of several smartphone apps that contain features wish to emulate in our own project. They in many ways solve the needs of our current clients. We therefore hope to take some of these phone-app features and modify them to better meet the needs of our own clients.

Whiteboard Capture Pro

Source: Beetlebug Software's website
http://www.beetlebugsoftware.com/

This is an iPhone app that takes a picture of a white board and then analyzes it for key content. The user selects objects to remove from the photograph. This leaves only the writing on the board behind. The App then analyzes the writing and removes the background image of the whiteboard itself. This sometimes leaves fuzz or imperfections in the white background, so there is then a slider available to filter out this extra noise/fuzz that shows up in the end product. The resulting image is a pure white background with handwriting on it.These photos can then be saved, cropped, shared, and organized within-app tools.

WBConference

Source: Elecom'e website
http://app.elecom.co.jp/en/wbcap/ios/manual.html

This is an Android app that competes with our product because it is another whiteboard capturing device. WBConference differs from Whiteboard Capture Pro in that it is able to automatically recognize which sections of the board are whiteboard. This then allows it to apply its magnified keystone correction to remove the excess background imagery. In cases that it cannot recognize the board you can zoom in on just the board boundaries manually before capturing the image. The app has contrast adjustment and image rotation as well so you can take images from any orientation without problems.This app has editing features as well so you can add postscripts or speech bubbles to the images. The files can then be saved as PDFs along with any notes you want to add to them.This app has a widget for the home screen for quick image capturing, and you can set up an

email address for quick delivery of the images to an external source.

CamScanner -Phone PDF Creator

Source: Intsig's Website http://www.intsig.com/en/camscanner.html

This is the most downloaded scanner app on the market. With it you can take photos of any document, whiteboard, etc that you want. You then go through an editing process in which you can select the important portion of the photograph, change the detail level, contrast, light/darkness etc. It will then save your new document in any number of saved folders. You can make notes about each image and these notes will be saved with the image. You can email, print, fax, or transfer via Bluetooth any of the photos. You can also upload your images to google docs, evernote, skydrive, dropbox, or box.net. These documents get saved as PDFs. There are different enhancement modes: No enhance, low and high enhance, gray mode, and B&W Document modes. These different modes will be better depending on the environment or object that youre trying to scan. The B&W mode is particularly helpful when scanning books/papers because it does a better job of removing the background noise. This app allows for batch photo taking and batch photo scanning, so you can take multiple pictures and it will scan them all at the same time.

Whiteboard Snap

Source: Google Play Store
https://play.google.com/store/apps/details?id=com.magnicode.whiteboard&hl=en

This app is a dumbed down version of the previous three. It does the job, it scans and enhances images, it just isnt as well known as the others and thus doesnt have the money/time to invest in extra features. This aside, it does work, it is free, and it does save images as PDFs for later use. You can email these photos to yourself and store them in different photos. You can attach notes to your images and you can enhance the quality of the whiteboard picture with their auto-enhance tool. On the upside, it IS a much smaller program than your avg whiteboard capture app. Over all a smaller lighter free alternative. I installed this app on my phone and I had trouble using it because it kept crashing.

Conclusion

These smart phone apps will be some of the greatest competition to our project because they meet many of our client's needs already. Not only that, they're free applications so our clients wouldnt need to spend money either. The following is a list of features that we should attempt to emulate when designing our own product.

All of these applications contain ways to filter out background noise so that whiteboards appear pure white and text looks crisp and clean. This will be an important aspect of our own product because our image capture device will be subject to a wide variety of lighting situations and will need to be able to adapt to any of them. Another key feature is the ability of these apps to save and send the images via email, google docs, and other online mediums. This will be an important point in our project as well. Another good feature was the ability for some of the apps (CamScanner for example) to correct for image angles. If the board is photographed from an angle, the smartphone apps will compensate so that the final scanned image looks like it was taken straight on.

Desirable features not found in smartphone apps:

- Photo splicing: These apps do not combine images to add in details covered by professor's body.

- Automatic: These apps do not automatically capture images, process them, and then send them away. They instead require user feedback every step of the way.

By adding these features to the functionality currently found in modern apps we hope to create value for our customer.

Microsoft Whiteboard Capture System

Source: Microsoft Research Technical Report MSR-TR-2002-89
ftp://ftp.research.microsoft.com/pub/tr/tr-2002-89.pdf
All images used in this section are from the source document.

This system accomplishes many of the goals we hope to achieve with the ProPANE system, using methods that seem feasible for our project. Therefore, we present an extensive look into this project, with analysis of methods used and relevance to our project.

Summary

The Whiteboard Capture System came out of Microsoft Research in 2002 as research for a bigger Distributed Meetings project. It can record all information placed on a whiteboard during a session, as well as the audio in the room, and it provides a platform for users to review key information from the session later from their own computers.

Key Features

The Whiteboard Capture System (WCS) requires only a camera, microphone, and computer to operate with any existing whiteboard. It records all information written on the board during a meeting, and all audio in the room during the meeting. The system then analyzes the data and detects "key frames", or pictures of the board which the system determines to contain the most important information. The key frames are time-stamped so they can be associated with the audio that was recorded at the same point in time. Users can access the key frames securely via a web-based browsing system. Audio can be played by clicking on a key frame, and the option exists for the presentation to unfold for the viewer like a movie, where information will appear on their web-based whiteboard as the audio recording talks about it, as shown below in Figure 3.

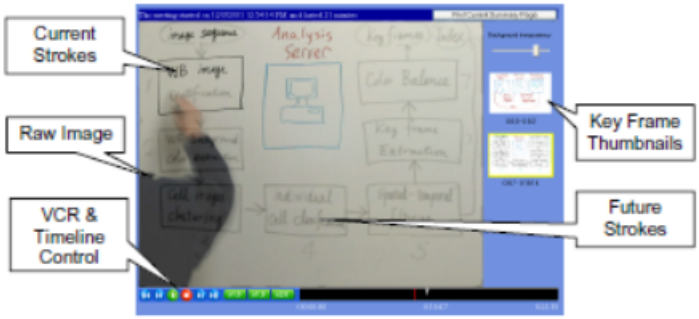

Figure 3: Browsing system representation of meeting playback

Technical information

The WCS uses a Canon PowerShot G2, a 4 megapixel digital still camera. The camera provides images with 2272 x 1704 pixel resolution, equivalent to 31.6 dpi for a 6' x 4' whiteboard. The camera is controlled via USB by a computer, which utilizes the available SDK for the camera to communicate. It takes a picture roughly every 5 seconds, which is as fast as possible for the PowerShot G2. Software specifies camera parameters on a per-shot basis. Pictures are saved as JPEGs and

sent to the computer via USB. The computer is a Pentium III 800 MHz dual CPU PC. It takes approximately 20 minutes of processing for every hour of session time. The input image sequence requires 34 MB/hour using motion JPEG compression, while the audio requires 15 MB/hour to store 16-bit 11 KHz mono mp3 audio.

To analyze the captured images, the WCS first divides the frame into a grid of rectangular cells of approximately 1.5" x 1.5" inches. The frame is represented by a 3D matrix of cell images. Analysis using cells requires much less computational power compared to a per-pixel analysis. Once the frame is divided into cells, the system uses an algorithm to prepare key frames in a readable fashion. The algorithm is as follows:

1. **Rectify** the whiteboard region of every image in the sequence
2. **Extract** the whiteboard background color
3. **Cluster** the cell images throughout the sequence for the same cell. If two images are considered to be the same, they are clustered in the same group.
4. **Classify** each cell image as a stroke, a foreground object, or the whiteboard
5. **Filter** teh cell images both spatially and temporally to refine the classification results.
6. **Extract** the key frame images using the classification results
7. **Color-balance** the key frame images

1. To rectify the whiteboard region, they use a one-time calibration step where the corners of the board are selected to manually define the frame.

2. The whiteboard color is computed by finding cells with the brightest luminance over time and small variance. During this step, cells which are impeded during the capture process (for example, by a person standing in front of the board) act as outliers. These are detected and accounted for using a least-median-squares technique.

3. Cell images are clustered together if they are determined to be identical. This is determined by a modified Normalized Cross-Correlation algorithm and the use of the Mahalanobis distance calculation.

4. Cells are classified as whiteboard, stroke, or foreground cells. Heuristically, whiteboard cells are uniform in color, either white or grey. Stroke cells are mostly white or grey with one or two primary colors mixed in. Foreground cells do not fit either of these descriptions. Essentially, it determines if a cell image is the same as a whiteboard cell image, very similar to a whiteboard cell image (stroke), or completely different (foreground object).

5. Filtering is used to better define cells. The system's spatial filter changes totally isolated foreground cells to stroke cells, since typically foreground cells are clustered (i.e. a person). Stroke cells touching foreground cells are also changed to foreground cells, unless said cells are also filtered temporally. The temporal filter looks at all cell images for a certain cell over time. If two cell images match, all cell images in between are deemed to be the same image as well. This filters out foreground objects.

6. Key frames are considered a summary of the entire presentation. The WCS defines key frames as frames that capture all information on the board just before an erase. They should only contain whiteboard and stroke cells, the whiteboard color should be uniform and the pen colors should be distinct. Image reconstruction is used to ensure the quality of the key frames. If the cell image is a whiteboard or stroke, its own image is used. If it's a foreground object within the span of a stroke, it is replaced with the stroke image from the neighboring frame. Otherwise, it is replaced by a whiteboard cell image. Key frames are only present before large erases, so they are only identified if at least 20% of the total stroke count is erased. The WCS divides the presentation into chapters, with each chapter being concluded by a key frame.

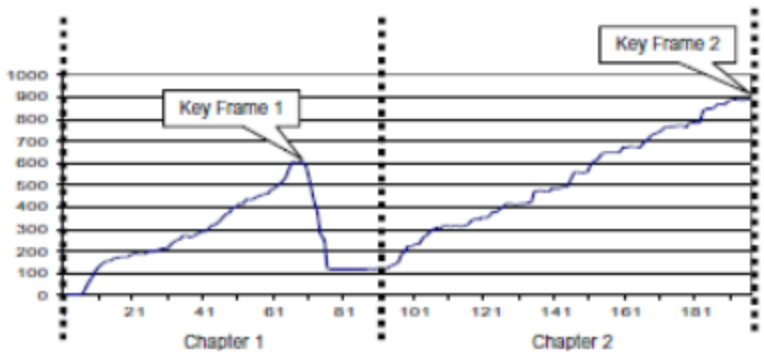

Figure 4: Plot of number of stroke cells vs. time. Key frames are shown at peaks just before stroke count decreases dramatically.

7. Color balance is done to make the background uniformly white and increase the color saturation of the pen strokes. It also reduces image noise. A final image is presented in Figure 5, below.

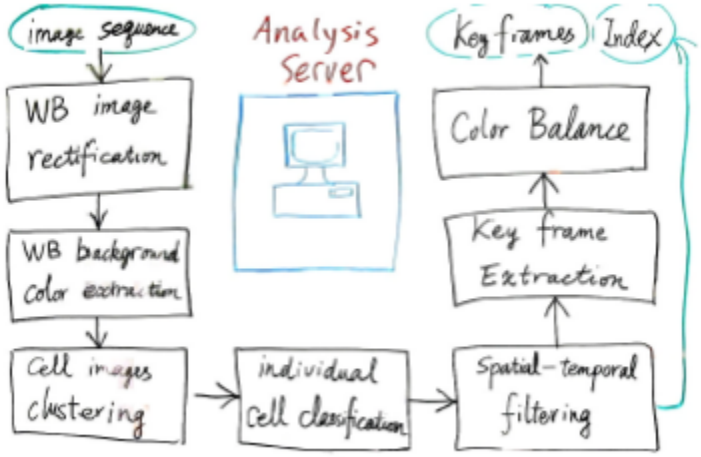

Figure 5: Example of a key frame as processed by the WCS

Browsing software allows viewing of key frames and audio playback. Users can zoom in/out and cut and paste the image to other documents. Double clicking on pen stroke cells plays audio at the time of the time stamp of the cell. Stroke cells will then be displayed as time progresses in the audio to their respective time stamps.

Access to the browsing software is via a token-based access model, where an access URL is emailed to all meeting participants.

Limitations

For proper functionality, the WCS requires a constant use environment. The whiteboard must have a constant color, constant lighting, and the camera must have constant exposure settings for the system to work properly. Most of these factors are standard for a normal whiteboard environment. However, the research team suggests that if a constant environment could not be guaranteed, the system might still work if a patch of known color were placed in the camera's field of vision, so the system could adjust settings on the fly based on the changes to the constant color patch.

At the time of publication, the team noted that the frame rate of cameras on the market was a

limiting factor for the system. Since this research is 10 years old, this is no longer as limiting - a quick search of competitively priced cameras returned cameras with over twice the frame rate as the PowerShot G2 used here. Other problems with the system are less likely. If a person in a white shirt stands perfectly still in front of the board for a long period of time, the system may not deduce that the person is in the foreground. This is interesting to note, but not a likely occurrence in a lecture environment.

Relevance to ProPANE

This system, as a whole, is very similar to our final goals for the BU ProPANE system. Especially useful to us is the key frame idea, since professors using the system will not want to sort through hundreds of images in order to find just a few important ones. The idea of breaking the board up into cells for analysis is also very interesting to us, as it greatly reduces the processing requirements. However, it is not perfect for our project. The WCS is purely a *whiteboard* capture system, whereas our system should work for black boards as well. Additionally, we are not focusing on any audio capture or playback for our system.

Mathematical Processes

Presented here are the general methods used by Microsoft Research to implement various features of their design.

1. Clustering Cell Images Over Time

As the meeting progresses, the WCS groups cell images from the same cell together if it determines that they don't change over a period of time. This is done using a modified Normalized Cross-Correlation algorithm to determine if two cells are the same or different. It is demonstrated here for one color, but applies to all RGB components.
Consider two cell images I and I'. Let $\bar{I}$ and $\bar{I}'$ be their mean colors and σ and σ' be their standard deviations. The normalized cross-correlation score is given by

$$c = \frac{1}{N\sigma\sigma'} \sum (I_i - \bar{I})(I'_i - \bar{I}')$$

where the summation is over every pixel i and N is the total number of pixels. The score ranges from -1, for two images not similar at all, to 1, for two identical images. Since the score is computed after the subtraction of the mean color, it may still give a high value even if two images have very different mean colors. So a different test is performed on the mean color difference, based on the Mahalanobis distance (info). The distance is given by

$$d = \frac{|\bar{I} - \bar{I}'|}{(\sigma + \sigma')}$$

Two cells are considered to be identical and are grouped together if and only if $d < T_d$ and $c > T_c$. In the WCS implementation, $T_d = 2$ and $T_c = 0.707$.

2. Classifying Cells

It must be determined whether a cell image is a whiteboard, stroke, or foreground object. The determination is based on whether or not a cell's color distribution is the same, similar, or very different from that of the whiteboard. As above, the Mahalanobis distance is used, and calculations are for one component of RGB.

Let $\bar{I}_w$ be the whiteboard color and σ_w be the standard deviation (small since a whiteboard cell is basically uniform). Then let $\bar{I}$ and σ be the mean and standard deviation of the current cell image. The cell image is classified as a whiteboard cell if and only if

$$\frac{|\bar{I} - \bar{I}_w|}{(\sigma + \sigma_w)} < T_w \quad \text{and} \quad \frac{\sigma}{\sigma_w} < T_\sigma$$

and as a stroke cell if and only if

$$\frac{|\bar{I} - \bar{I}_w|}{(\sigma + \sigma_w)} < T_w \quad \text{and} \quad \frac{\sigma}{\sigma_w} \geq T_\sigma$$

Otherwise, it is classified as a foreground object cell. In the WCS implementation, $T_w = 2$ and $T_\sigma = 2$.

3. Key Frame Color Balance

The background must be uniformly color-balanced and the color saturation of the pen strokes must be increased. The previously-calculated whiteboard color, $\bar{I}_w$ is used to scale the color of each pixel in the cell.

$$I_{out} = \min(255, \frac{I_{in}}{I_w} \cdot 255)$$

Image noise is reduced by remapping the value of each color channel of each pixel in the key frames according to an S-shaped curve.

ReBoard

Source: http://arxiv.org/ftp/arxiv/papers/0911/0911.0039.pdf
All images used in this section are from the source document.

Summary

ReBoard is a system developed by FX Palo Alto Laboratory, Inc. It captures whiteboard content both actively and passively, while providing a robust platform for data retrieval and sharing. Users can browse captured data by aspects such as time or location of capture, or who was present for data capture.

Key Features

ReBoard's focus is on the ease of sharing captured data between users. The User Interface (UI) provides multiple options for data retrieval and sharing.

- *Calendar View:* Sorts images by date of capture, color-coded by type of content (blue for personal, violet for collaborative, etc...). See Figure 6 below.
- *Timeline View:* Arranges captured images chronologically, with a histogram where higher bars mean more content was created or erased for that capture. Dates at the bottom of the display indicate approximate date of capture. See Figure 7 below.
- *Heatmap View:* Aggregates changes to the board by location over a given time frame. The more a region of the board has changed, the warmer the color used to fill the region. See Figure 8 below.
- *Summary Pane:* Users can click the "detail" button for a particular content record to open in a new browser tab. This view provides a detailed interface for viewing high-resolution images of the captured content and controls to manipulate the associated metadata.

Thanks to these views, users can retrieve data based on who was present during its creation, by a specific region on the whiteboard where it was created, by the time it was created, or by user-assigned labels and descriptions.

ReBoard's unique feature is its ability to detect collaboration. The system will detect when multiple people are working at the board, and the content created at that time will be tagged as collaborative, and users can sort images in the UI by this characteristic.

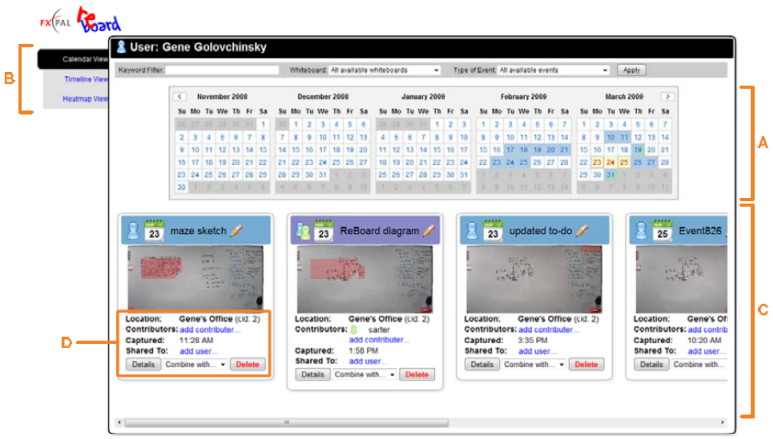

Figure 6: ReBoard's Calendar View

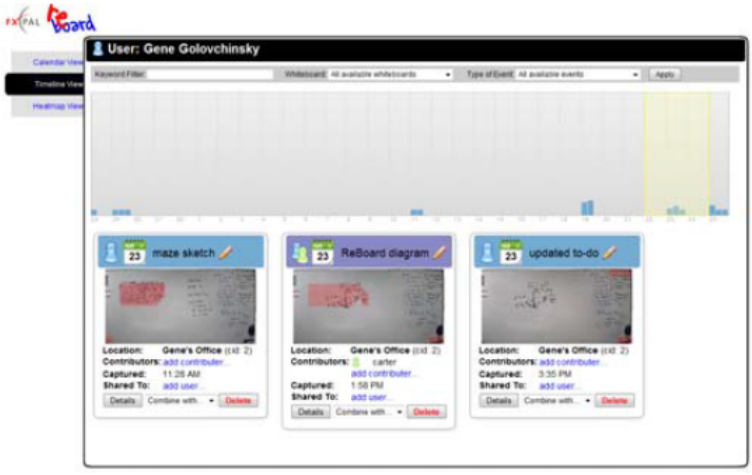

Figure 7: ReBoard's Timeline View

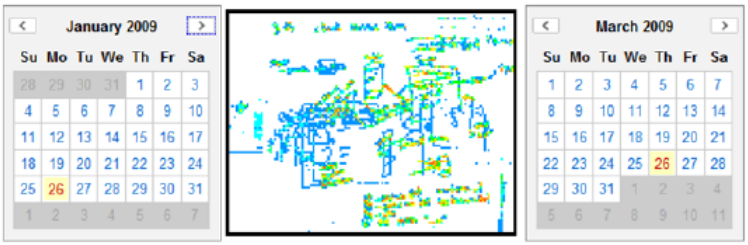

Figure 8: Close-up of ReBoard's Heatmap View

Technical information

The ReBoard system consists of one video camera per whiteboard, an application server, a database to store images and metadata, detectors that sense collaborative activity, detectors that record content changes, and portable user interfaces to manage content and metadata (these are optional). The video cameras record at 10 fps, with important images captured at times indicated by the detectors

and user interfaces. Cameras are calibrated at installation to correct for geometric distortion due to off-axis camera positioning. The content change detector is a Java-based service that leverages Java's Media framework and OpenCV's image processing libraries. The collaboration detector is a C# Windows Service using the AForge computer vision framework for basic video processing and "blob" detection (more information on AForge here). Event detectors have IDs that link them to a specific camera so they can work independent of each other. The application server is written in Grails, with the metadata stored in a MySQL database. The server manages user accounts, sessions and access control, and exposes a set of REST-like web services for AJAX UI clients and for communicating with the event detectors. The persistent UI runs on a portable Chumby device (See Figure 9 below, more information here). It allows for manual image capture, capure disable, and content bookmark. The persistent UI is not necessary for the system to function.

Figure 9: Persistent UI running on a Chumby device

Relevance to ProPANE

While ReBoard is designed for continuous use in the workplace, as opposed to ProPANE's classroom setting, there are still some features of the ReBoard system that could contribute to the ProPANE system. The portable user interface is an interesting concept, because while the ProPANE system should be fully automatic, the ability to manually capture board images would be useful as a kind of fail-safe if problems arose with automatic detection. Another strength of ReBoard is its web UI. The option of multiple views to recover data seems very useful, and some would be particularly useful to ProPANE. The calendar view would make it easy to sort lectures by date, and a modified timeline view with exact dates would help professors identify lectures with more content than others. Finally, the heatmap view could be utilized on a smaller scale to track changes during one particular lecture.

Normally, lectures are led by the professor and do not feature collaboration, sometimes students will contribute to the board by answering questions, or in other ways. It may be useful for the professor to track when board content is authored by a student, and the identity of that student. ReBoard's collaboration detection could be useful in implementing a feature like this, although probably not in the initial build of the ProPANE system.

Cameras we could use in our project

Here are the current top three cameras that we think could help us the most when building our image capturing system.

- Nikon COOLPIX S800c 16 MP Digital Camera
  - `http://www.amazon.com/Nikon-COOLPIX-Digital-Optical-3-5-inch/dp/B0090SLKUM`
- Samsung Camera EK-GC100 Galaxy Camera

  - http://pdadb.net/index.php?m=specs&id=3813&c=samsung_ek-gc100_galaxy_camera
- Polaroid SC1630 Smart Camera
  - http://www.upi.com/Science_News/2012/01/16/Polaroid-joins-digital-camera-arena/UPI-61851326750025/

We are interested in them because they are cameras running the Android operating system. This means that we could create our own custom application for these devices, greatly simplifying our design process. These cameras would also be useful because they can connect to Bucknell's wifi network. This would allow us to wirelessly transferr information from the cameras to whatever image processing hardware we decide to connect it to. The main drawbacks at the moment with these Android cameras is that two of them haven't been released yet (Samsung and Polaroid). IF they are released in time, they would be the most desireable of the various camera options available.

Details on Learning Disabilaties

Introduction

One of the major motivating factors behind designing our image capturing system is to help meet the needs of students with disabilities. The term "students with disabilities" is a very broad term, however, so we would like to use the following section to help discribe some of the things that mildly disabled students have trouble with at Bucknell, and would therefore need our system to capture information presented on the board for them.
Source:
http://www.sfasu.edu/disabilityservices/facultyandstaff/for_service_providers/note_q_a.asp

Disabilities that students might have that impair their ability to take notes:

- Visual Impairments
  - May be fully blind and need notes translated into Braille
  - May not see well and need large print letters
  - May have trouble copying information from whiteboards, projectors, etc.
  - May have trouble seeing certain colors when framed by a white or black background
- Specific Learning Disabilities
  - Reading Disability
  - Writing Disability
  - Spelling Disability
  - Inability to copy what they see
  - Inability to write what they hear
  - Inability to write legibly
  - Number Reversal problems
- Mobility Impairments
  - Physically unable to write
  - Physically unable to write quickly
  - May be unable to effectively handle a writing impliment
- Partial or Full loss of hearing

This is just a small portion of the many disabilities faced by students in universities around the world. We hope to help them by giving them full access all information presented on boards during lectures. By providing easily accessible, easily modifiable images, we hope to help even the playing field for students with disabilites. Secondary goals of our project will help to make the learning process even easier. Some students get distracted if they see more than one line of text at a time. If

we have enough time we will help these students by providing slide bars that will cover portions of the images that students are not currently viewing. This and many other minor features are things that we will accomplish if we have free time after completing our primary objectives.