Retrieval of Lecture Slides by Automatic Slide Matching on an Android Device

Kyle Campiotti
Department of Electrical Engineering
Stanford University
Stanford, CA, USA
kylecamp@stanford.edu

***Abstract*—This paper describes the development of an application that allows presentation attendees to take a photo of the projection screen with their Android device and automatically retrieve the correct presentation from a database, along with the corresponding page number. In order to accomplish this, the query image is uploaded to a server where it is processed using image segmentation, SIFT features, REVV ranking, and RANSAC matching. In testing, this image-processing algorithm retrieved the correct slide 91.1% of the time (on average) for reasonable images.**

Keywords—Android; MATLAB; slide matching; image segmentation; SIFT; REVV; RANSAC

I. INTRODUCTION

The motivation for this project comes from the popularity of using slides (generally originating from presentation software, such as Microsoft PowerPoint) in public presentations and the increasing proliferation of smartphones and tablets at these events. These types of presentations occur in many different scenarios, including the classroom setting and technical conferences. In these settings, there will often be an accessible database of presentations in electronic form. Typically, if an attendee would like to follow along with the presentation on their smartphone or tablet, he/she must manually search through the database of presentations, find the correct slide deck, and then search for the slide that is currently being presented. As the size of the presentation database increases, this task rapidly becomes prohibitively time-consuming. This project aims to allow presentation attendees to take a photo of the projection screen with their Android device and automatically retrieve the correct presentation from the database, along with the corresponding page number.

II. METHODS

The main image-processing portion of this project involves matching the slide in an image to a database of known slides. The paper [1] proposes performing this task using scale-invariant feature transform (SIFT) and random sample consensus (RANSAC). This method is further elaborated on in [2]. The original proposal for this project suggested using speeded up robust features (SURF) instead of SIFT, as suggested in [3]. However, also based on the results in [3], it was determined that MATLAB and the processing power of a server would be leveraged for the image processing algorithms in this project. With this in mind, the availability and MATLAB interface of the VLFeat SIFT library outweighed the potential speed benefits of using SURF (i.e. a similar library utilizing SURF was not readily available).

The reference database used to test this project consisted of the slides for the first nine lectures of the spring quarter 2012 EE 368: Digital Image Processing class at Stanford University. This created a database of 387 slides against which to compare query images. Throughout the quarter, 116 photographs were taken during EE 368 lectures using the 5-megapixel (MP) camera on an Apple iPhone 4. This camera is very similar to the 5 MP camera found on the Motorola DROID on which the final application was implemented. These images were used as sample query images during testing of the image-processing algorithm for this project. The following discussions of algorithm development and testing assume the use of the database described above along with these sample query images, tested on a desktop computer with a 3.1 GHz Intel Core i5 processor and 4 GB of RAM.

A. Slide Matching Algorithm

In early algorithm development for this project, an algorithm utilizing only SIFT features and RANSAC matching (using an affine model) was implemented and informally tested using the sample query images. This algorithm searched the entire slide database for the slide with the largest number of RANSAC matches. Although no formal testing of accuracy occurred, this algorithm qualitatively demonstrated that the correct slide could be matched to a wide variety of query images. The main drawback of this naïve implementation was the significant amount of time (several minutes) it took to execute a match. It was determined that the RANSAC algorithm was the main driver of this extended processing time. In order to shorten this time, a coarse ranking algorithm utilizing residual enhanced visual vectors (REVV) [4] was implemented in order to find the most likely matching slides and rank them above less likely matches. REVV was originally designed for use in mobile devices and thus is inherently efficient. In this version of the algorithm, RANSAC matching was only performed on the top matches from the REVV rankings.

During informal testing of the REVV-based algorithm, processing times were significantly decreased for a certain subset of the sample query images. However, even for well-taken, reasonable query images the correct slides were sometimes being ranked extremely low (sometimes lower than 100) in the REVV rankings. It was determined that the cause for this was excessive amounts of clutter in the sample query images (e.g. the presenter, audience members, empty chairs, objects near the projection screen). Since the REVV algorithm does not take into account geometric transformations, the SIFT features from this clutter could sometimes be incorrectly matched to features in slides from the database. This significantly limited the effectiveness of the REVV rankings.

The study in [2] discusses a method of eliminating clutter in slide matching algorithms for presentation videos by learning the scene background over several frames of video. In order to eliminate the clutter in query images for this project, a different image segmentation process for individual images needed to be implemented. This algorithm was added into the process flow prior to SIFT feature extraction. It was noted that the projection screen in the query images generally appeared as a large, bright object compared to the rest of the scene. Using this information, the following image segmentation was implemented. First, using Otsu's method, the query image is converted to a grayscale image and segmented into foreground and background regions. Fig. 1(a) shows a typical query image and Fig. 1(b) shows the results from Otsu's method. Using image-labeling algorithms from MATLAB's image processing toolbox, the foreground regions are then ranked in order of decreasing area. The region with the largest area is next analyzed to determine if it positively identifies the projection screen region of the image. To do this, the region is checked for its ratio of "filled area" to "convex area". The filled area is defined as the area of the region in question with all of the "holes" (i.e. background regions completely contained within the region in question) filled. The convex area is defined by MATLAB as the area of the smallest convex polygon that can contain the region. The ratio of these two quantities will be large for regions such as projection screens, but small for other types of regions (such as "L"-shaped regions). Fig. 1(c) shows the final slide fully segmented from the query image.

With this information, the following process flow was implemented as the final slide matching algorithm for this project:

1) Database establishment: Prior to application use, the database slides are converted from their current format into individual JPEG images. After being resized to a maximum width of 640 pixels (to improve processing time), SIFT features and REVV signatures are extracted from these images and stored.

2) Image segmentation: After receiving the query image, the image is segmented into foreground and background regions using Otsu's method. The foreground region with the largest area that also has a ratio of filled area to convex area greater than 0.75 is assumed to be the projection screen and is segmented from the image.

3) Feature extraction: After resizing the segmented projection screen image to a maximum width of 640 pixels, SIFT features are extracted.

4) Coarse ranking: The REVV signature is generated for the projection screen image and compared to the REVV signatures of each database slide. The database slides are then ranked from the most probable match to the least probable match.

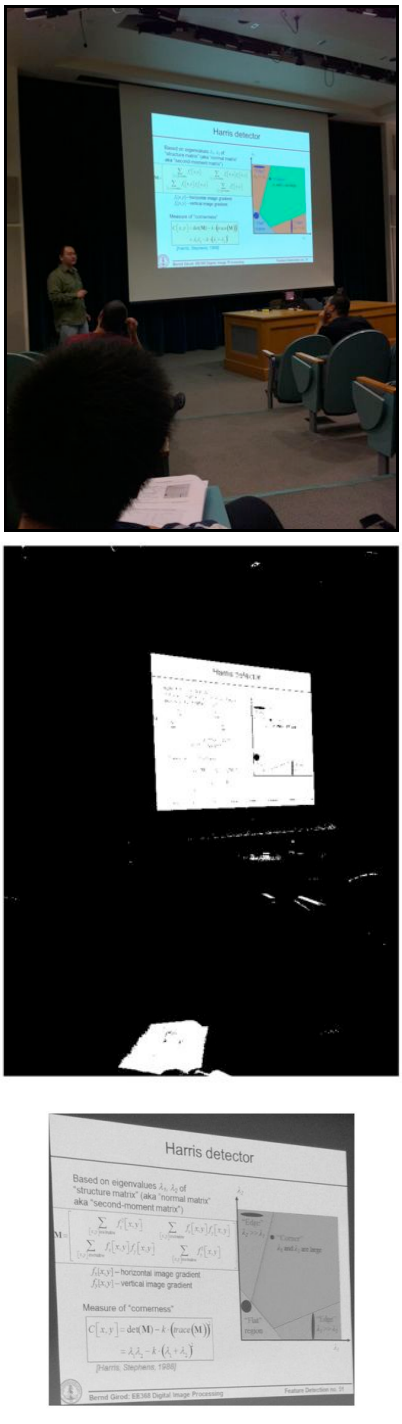


Figure 1. (a) Typical query image; (b) binary version of the query image segmented using Otsu's method; (c) grayscale image of the slide region after being fully segmented from the query image.

5) Final matching: Using RANSAC, the top ten matches from the REVV rankings are analyzed. If none of these database images have at least 20 matches, additional slides are analyzed, up to 50 total slides. The slide with the highest number of matches after RANSAC is returned as the matching slide.

B. Test Methods

Some of the photographs that were taken as sample query images produced images of low quality. This was done both intentionally, in order to order to test the robustness of the algorithm, and accidentally. In order to first test the accuracy of the algorithm without these factors, the sample query images were divided into two groups based on their quality, "reasonable" and "unreasonable". Reasonable images were those that portrayed the slide in a clear, well-lit manner. These images also represent images taken of "complete" slides (i.e. images that depicted slides that had completed all animations and were thus included in the database). Unreasonable images were those query images that did not meet one or more of these criteria. Reasonable image were further divided into three subgroups: "normal", "dark" (i.e. background), and "rotated". Unreasonable images were further divided into four subgroups: "blurry", "overexposed", "incomplete", and "multiple issues" (i.e. two of either dark, rotated, blurry, overexposed, or incomplete; the dark and rotated combination excluded).

The reasonable images were used to fine tune one of the parameters in the RANSAC algorithm, namely the error threshold for a positive match. This threshold determines the number of pixels by which a given feature can miss the affine model determined by RANSAC and still be considered a match. If this value is too large, false-positive results will occur. If this value is too small, correct matches can potentially be excluded. This parameter was swept over several values. For each value, the accuracy rate was determined by finding the average accuracy rate over ten iterations of testing on the reasonable images. Once this parameter had been definitively determined, the unreasonable query images were tested in the same way in order to test the robustness of the algorithm.

III. Results and Discussion

A. RANSAC Error Threshold Testing

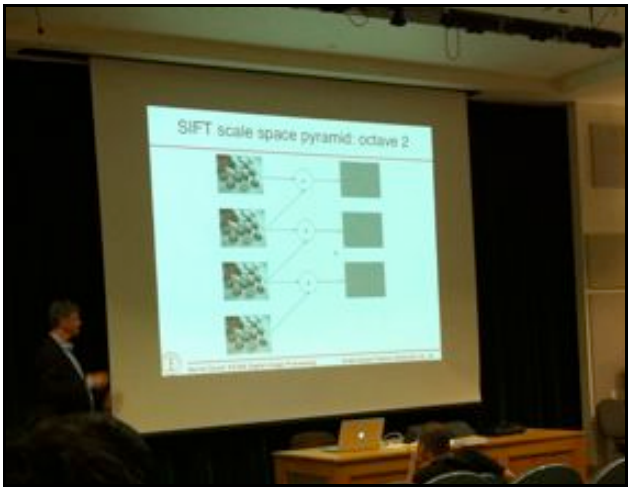
Of the 116 sample query images, 44 were categorized as reasonable. Table I provides accuracy data on four different values of the RANSAC error threshold for this dataset. Due to the random nature of RANSAC, at each error threshold value the query images were tested a total of ten times to ensure that a given reported accuracy rate is representative. In addition to the average accuracy rate, the minimum and maximum accuracy rates over these iterations are also included. As shown in Table I, the best accuracy rate occurred for a RANSAC error threshold value of 3 pixels. This was the value chosen for the threshold in the final application.

In theory, reasonable images should be matched to their corresponding slide every time. In this experiment, this was not the case due to several slides in the database being extremely similar in content. For example, during testing with a RANSAC error threshold value of 3 pixels, there were seven images that were incorrectly matched in at least one of the ten trials. All other slides were matched correctly in every trial. These seven images included slides that are very similar in content to other slides in the database. This implies that the error rate given is highly dependent on the database of slides being queried. Fig. 2(a) shows a typical query image for one of these cases and Fig. 2(b) shows the incorrectly matched slide.

It should be noted that these issues were not related to the REVV rankings (i.e. if the complete database of SIFT features were searched, the correct match would still not have been found every time). This indicates that the error rate is due to the random aspect of RANSAC.

TABLE I. Slide Matching Algorithm Accuracy Rates for Various Values of the RANSAC Error Threshold

RANSAC Error Threshold (pixels)	Average	Minimum	Maximum
10	88.9%	86.4% (38/44)	90.9% (40/44)
5	89.5%	86.4% (38/44)	93.2% (41/44)
3	91.1%	88.6% (39/44)	95.5% (42/44)
2	90.0%	84.1% (37/44)	95.5% (42/44)



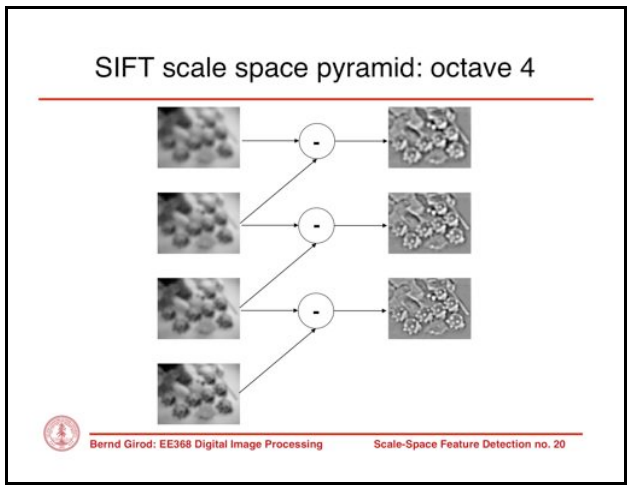


Figure 2. (a) Typical incorrectly matched query image (with some clutter cropped for clarity); (b) the incorrectly matched slide.

B. Slide Matching Algorithm Testing

Table II provides additional statistics for the slide matching algorithm implemented on both the reasonable and unreasonable image datasets with a RANSAC error threshold of 3 pixels. Of the 44 reasonable images, 40 were categorized as normal, two as dark, and two as rotated. Of the 72 unreasonable images, two were categorized as blurry, 42 as overexposed, 15 as incomplete, and 13 as having multiple issues. The following is a brief analysis of this data.

1) Reasonable Images vs. Unreasonable Images: As expected, the reasonable images were matched significantly more accurately than the unreasonable images (91.1% vs. 71.3%). Since the slide matching algorithm was not designed for the unreasonable image dataset, a 71.3% accuracy rate is still a very positive outcome. The data also show that the matches found for reasonable images tended to be higher ranked by the REVV algorithm (3.2 vs. 9.2) and had more than twice as many matching features (86.3 vs. 42.4). The time taken to match the reasonable images was also much shorter than the time taken to match unreasonable images (7.2 sec vs. 10.1 sec). This is likely due to the low number of matches in the unreasonable images requiring the matching algorithm to perform additional RANSAC matching.

2) Correct Matches vs. Incorrect Matches: The data for this comparison are very similar to that of the previous comparison. Correctly matched images tended to be ranked higher, have more features, and be calculated faster than their incorrect counterparts across all datasets. Once again, this is expected.

3) Robustness of the Slide Matching Algorithm

a) Dark, rotated, and, blurry: The algorithm had no incorrect matches in these three categories. These datasets were also the smallest (two images each), meaning this data may not be fully representative. In the dark case, the algorithm should work as well as it does for reasonable-images-at-large, as long as the image segmentation portion of the algorithm works well enough to segment out the projector screen. This will occur in many use cases, as the projector still illuminates the screen even when the background of the slide is black. It should be noted, however, that the dark images had significantly fewer RANSAC matches that the reasonable-images-dataset-at-large (36.2 vs. 86.3). The rotated case should also perform as well as reasonable-images-at-large because RANSAC is rotation invariant. The accuracy rate for blurry images depends entirely on the amount of blur. In the images in this dataset, the level of blur was not extremely significant, although it was noticeable enough to make some text unreadable. In theory, enough blur can significantly change which SIFT features are extracted, negatively affecting accuracy.

b) Overexposed, incomplete, and having multiple issues: These categories showed a significant drop in accuracy compared to the others. Overexposed images tend to "wash out" many of the features that the slide matching algorithm uses to find the best match; incomplete images do not have all of the features on the projector screen by which to find the correct slide; images with multiple issues compound all of these problems. This is reflected in the low REVV rankings in these categories, the small number of RANSAC features matched, and the significantly longer image processing time.

C. Implementation on an Android Device

The purpose of this project was to provide a slide matching application for an Android device. As mentioned previously, in order to provide results most efficiently, MATLAB and the processing power of a server were leveraged for the image-processing portion of the implementation. An Android application was written in which a user takes a photograph of a scene containing a projection screen, the image is uploaded to the server and analyzed using the slide matching algorithm described previously, and the server returns a text file to the Android device signaling it to automatically open the correct PDF.

TABLE II. SLIDE MATCHING ALGORITHM SUMMARIZED TEST RESULTS

Dataset	Percentage of Correct Matches	REVV Rank of Match			Number of Matching Features After RANSAC			Image Processing Time (sec)		
		All Images	*Correct Matches*	*Incorrect Matches*	*All Images*	*Correct Matches*	*Incorrect Matches*	*All Images*	*Correct Matches*	*Incorrect Matches*
Reasonable Images	91.1%	3.2	2.2	14.9	86.3	90.8	38.5	7.2	7.0	9.5
Normal	*90.3%*	*3.5*	*2.4*	*14.9*	*88.0*	*93.2*	*38.5*	*7.3*	*7.1*	*9.5*
Dark	*100.0%*	*1.0*	*1.0*	*N/A*	*36.2*	*36.2*	*N/A*	*5.6*	*5.6*	*N/A*
Rotated	*100.0%*	*1.0*	*1.0*	*N/A*	*101.2*	*101.2*	*N/A*	*6.3*	*6.3*	*N/A*
Unreasonable Images	71.3%	9.2	7.7	12.8	42.4	53.8	14.0	10.1	8.4	14.1
Blurry	*100.0%*	*1.0*	*1.0*	*N/A*	*80.0*	*80.0*	*N/A*	*5.8*	*5.8*	*N/A*
Overexposed	*69.0%*	*7.2*	*6.9*	*7.9*	*39.8*	*48.8*	*19.6*	*10.2*	*8.6*	*13.8*
Incomplete	*73.3%*	*11.7*	*10.0*	*16.2*	*55.6*	*70.2*	*15.6*	*9.0*	*8.0*	*11.8*
Multiple Issues	*63.1%*	*13.9*	*10.0*	*20.1*	*29.5*	*43.2*	*6.2*	*11.4*	*9.5*	*14.8*

In the original proposal for this project, it was desired for the PDF to automatically open to the correct page. However, there are currently no known Android PDF applications that allow this functionality. Instead, the user is provided with a popup message telling him/her what page to turn to.

This implementation assumes that the slides are available prior to the lecture and that they have been downloaded to both the database and the Android device. It also assumes a stable network connection for the server as well as the Android device. Fig. 3 depicts the application being used.

In terms of performance, the application running on a Motorola DROID device using Android 2.2.3 has shown considerable success in matching the correct slide to the query image. Because the Android application was implemented after the lectures in the database had already been given, desktop and laptop computer screens were used as reasonable analogs to projection screens. In testing the application, care was taken to take photographs that had the lecture slides displayed at a reasonable size compared to clutter in the image (i.e. at a similar size in the viewfinder as if a lecture was being given on a projection screen). Although no formal accuracy testing was performed on the Android application, during a poster session in June 2012, live demonstrations were given to approximately 30 attendees. A random slide was chosen from the database and a photograph was taken of a laptop screen displaying the chosen slide. In all but approximately four of the demonstrations the correct slide was retrieved on the first attempt. In the other four demonstrations, the correct slide was retrieved on the second attempt. Only one of the failed attempts was related to an incorrect slide that matched too closely to the query image. In the other failures, there may have been issues with motion blur and/or interference from the laptop keyboard (i.e. the keyboard was brushed aluminum, which may have interfered with the image segmentation portion of the algorithm). In any case, the failure rate seen in live testing was very similar to the rate seen in the set of sample query images.

In terms of time, with the addition of uploading the query image and retrieval of the information about the matching slide, the overall time for slide retrieval on the device is around 15 sec. This is a very reasonable timescale given that most presenters spend 30 sec to 1 min on each slide.

IV. Conclusions

The goal of this project was to develop an Android application that would allow presentation attendees to take a photo of the projection screen with their Android device and automatically retrieve the correct presentation from the database, along with the corresponding page number. The project has succeeded in this endeavor in a way that is not only accurate, but also reasonably fast.

Future work on this project may include implementing the slide matching algorithm entirely on the device in order to remove the dependence on the network connection, making the algorithm faster through novel approaches, implementing a custom PDF application that allows automatic opening to a specific page, or porting the application to iOS devices.

Figure 3. (a) Motorola DROID taking a photograph of a presentation displayed on a desktop computer; (b) status update from the Android application; (c) result returned; (d) correct PDF displayed (and manually turned to the correct page).

ACKNOWLEDGMENT

The author would like to acknowledge Dr. Bernd Girod for his dedicated teaching of the EE 368 class at Stanford University. He would also like to acknowledge and personally thank the course assistant for the class (and mentor for this project), David Chen, for his considerable assistance in the completion of this project, without which this project would not have been successful.

REFERENCES

[1] N. Cheung, D. Chen, V. Chandrasekhar, S. Tsai, G. Takacs, S. Halawa, B. Girod, "Restoration of out-of-focus lecture video by automatic slide matching," in ACM International Conference on Multimedia 2010, Firenze, Italy, 2010.

[2] Q. Fan, K. Barnard, A. Amir, A. Efrat, M. Lin, "Matching slides to presentation videos using SIFT and scene background matching," in ACM International Workshop on Multimedia Information Retrieval, New York, NY, 2006, pp. 239 - 248.

[3] S. Tsai, D. Chen, J. Singh, B. Girod, "Rate-efficient, real-time CD cover recognition on a camera-phone," in ACM International Conference on Multimedia 2008, Vancouver, BC, 2008.

[4] D. Chen, S. Tsai, V. Chandrasekhar, G. Takacs, H. Chen, R. Vedantham, R. Grzeszczuk, B. Girod, "Residual enhanced visual vectors for on-device image matching," in Asilomar Conference on Signals, Systems, and Computers, 2011.