Gestural Interaction for an Automatic Document Capture System

Christian Kofler, Daniel Keysers
German Research Center for Artificial Intelligence (DFKI),
Kaiserslautern, Germany
{christian.kofler, daniel.keysers}@dfki.de

Andres Koetsier, Jasper Laagland
University of Twente, Enschede, The Netherlands
{a.koetsier-1, j.laagland}@student.utwente.nl

Thomas M. Breuel
Technical University of Kaiserslautern, Germany
tmb@informatik.uni-kl.de

Abstract

The amount of printed documents used today is still very large despite increased use of digital formats. To bridge the gap between analog paper and digital media, paper documents need to be captured. We present a prototype that allows for cost-effective, fast, and robust document capture using a standard consumer camera. The user's physical desktop is continuously monitored. Whenever a document is detected, the system acquires its content in one of two ways. Either the entire document is captured or a region of interest is extracted, which the user can specify easily by pointing at it. In both modes a high resolution image is taken and the contained information is digitized. The main challenges in designing and implementing such a capturing system are real-time performance, accurate detection of documents, reliable detection of the user's hand and robustness against perturbations such as lighting changes and shadows. This paper presents approaches that address these challenges and discusses the integration into a robust document capture system with gestural interaction.

1. Introduction

In 1975, the *Business Week* confidently foresaw the paperless office to be close [1] but still this vision has not become reality. Instead, the use of paper in a typical office doubled since then from 100 to 200 pounds of paper per head [2]. Although the digital alternatives for mail, news and other forms of information are mature, the analog versions are still widely used and will continue to play an important role not only in office life [10]. Instead of ignoring the information available in paper form, easy transformation into the digital world is required to manage, archive and share it using the advantages of modern electronic communication.

Today, there are several ways of performing such a transformation of information from a document which is available on paper into a digital version of it. The user can e.g. use a scanner or a digital camera or even her mobile phone to take a picture of that document. Depending on the intended use of the information to be digitized there are subsequent steps to be taken, such as cropping the image to the exact boundaries of the entire document or a subregion of interest and extracting textual information by employing Optical Character Recognition (OCR). Many users regard these steps as obstacles and still prefer to transcribe the parts of the document they want to have available in digital form.

The system we present here removes these obstacles and supports the user digitizing documents either in oblivious or in interactive mode. A first version of the oblivious mode of the system was presented in [7]. Since then, we have improved the system in terms of performance, accuracy and robustness but the concept is still the same: The user works on his desk and is oblivious of the document capture system which continuously captures all new documents which appear on the users workspace. The textual content of these documents is then made accessible via a full-text search engine. This mode works completely without physical user interaction, and therefore without interrupting the user's everyday work-flow.

In discussions with users of the system it was frequently suggested that a possibility to select a region of interest

within a document would be beneficial. Hence, we started to integrate gesture recognition into the document capture system as a comfortable way to interact with it. In the interactive mode of our current prototype the user can point at the document to define a region he is interested in. The text line which is closest to this point will be detected and the textual content extracted.

In both modes, oblivious and interactive, we employ our open-source OCR system OCRopus[1] to extract text from the document image.

The central component of the presented system is the detection of a document when it is placed in the viewfinder zone of the input device, i.e. a standard consumer camera. After the exact detection of the boundaries of the document a high-resolution image is taken and the document image is extracted. The perspective distortion is then removed from the captured document image to gain a rectified version of it. If the interactive mode is enabled the high-resolution of the placed document is only taken if the user pointed at a region of interest within it. Then the captured document is annotated with the pointing location which can be reused later-on in further processing steps.

2. System Design and Implementation

The current prototype of the presented system is intended to consistently visualize bridging the gap between physical and digital desktop. Hence, the demonstrator is a special wooden desk on top of which a standard consumer camera can be mounted. The only requirement for the employed camera is support for the standardized Picture Transfer Protocol (PTP) developed by the International Imaging Industry Association to allow the transfer of images from digital cameras to computers without the need of additional device drivers. A number of tests with commonly used cameras showed that only few manufacturers incorporate reasonable PTP support in their cameras. Currently we are using a 7 mega-pixel Canon A 620 camera.

The viewfinder image of the camera is requested continuously and handled in a sequence of processing steps. First, the parts of the image which constitute background are detected and masked to reduce data for all subsequent steps. The remaining pixels are considered foreground. An important step to achieve higher accuracy and robustness is the detection of shadows which are cast by foreground objects, such as the hand of the user. The main component of the system is the document detection. If fingertip detection is enabled, the respective processing steps are performed to determine the point of interest in the document at which the user pointed with his index finger.

Figure 1 shows the demonstrator hardware and Figure 2 gives an overview of the software components involved in the presented system. Each of the processing steps is discussed in more detail in the following sections.

[1] http://www.ocropus.org

Figure 1. The demonstrator of the presented document capture system.

2.1. Background Detection

In order to reduce the amount of data for the subsequent processing steps a distinction between background and foreground of the current input image is necessary. Therefore, a dynamic background model is initialized as soon as the system starts to operate and updated continuously to be robust against local changes, such as moving background objects, and global changes, such as changing lighting conditions. Additional sources of global change are the automatic white balancing and color adaptation of the used consumer camera.

Based on the dynamic model, background subtraction is performed on each input image to determine foreground regions. An overview of dynamic background subtraction strategies can be found in [9]. As the requirements for our interactive system include real-time tracking with reasonable accuracy we decided to use a mixture of Gaussians for robust background subtraction. This method is described in [11] where it is used for tracking traffic.

A pixel X is defined in the RGB color space as $X = \{R, G, B\}$. The history of a pixel $\{X_1, ..., X_t\}$ is modeled

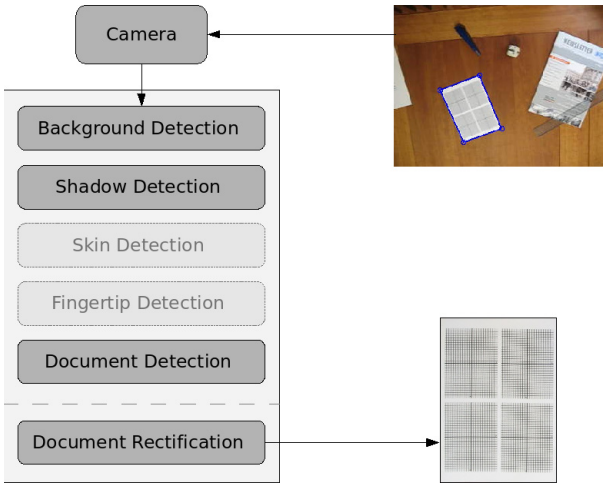

Figure 2. The components of the presented system: The viewfinder image of the camera is processed with background, shadow and document detection, optionally skin and fingertip detection is performed. The detected document is then extracted by taking a high resolution image and rectifying the respective image part.

by K Gaussian densities. The probability of observing pixel X_t is:

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t}\, \mathcal{N}(X_t, \mu_{i,t}, \Sigma_{i,t})$$

where K is the number of Gaussian densities, $\omega_{i,t}$ is the weight of the i-th density at time t, $\mu_{i,t}$ is the mean of the i-th density at time t, $\sum_{i,t}$ is the covariance matrix of the i-th density at time t and $\mathcal{N}$ is the Gaussian probability density function:

$$\mathcal{N}(X, \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}}\, e^{-\frac{1}{2}(X-\mu)^T \Sigma^{-1} (X-\mu)}$$

Here, we use diagonal covariance matrices of the form $\Sigma_{i,t} = \sigma_{i,t}^2 I$

The K Gaussian densities are sorted according to their weights in descending order. We used the first N of the K Gaussians to define background, following a suggestion made in [12]. This means that the N Gaussian with the highest weights are considered background. Whenever a new pixel is presented to the model, the model is updated by first iterating through the K Gaussians and determining the density that best explains the pixel value and then updating the weights of all densities.

If none of the K densities explains the pixel value sufficiently well, the density with the lowest weight (i.e. the K-th density) is replaced by a new density with $\mu_{K,t} = X_t$ and low $\sigma_{K,t}$ and $\omega_{K,t}$.

The weights are updated according to

$$\omega_{i,t} \leftarrow (1-\alpha)\omega_{i,t-1} + \alpha(M_{i,t})$$

where α is the learning rate and $M_{i,t}$ is 1 for the density i that matched X_t and 0 for the other densities.

The remaining parameters are updated according to

$$\begin{aligned} \mu_{i,t} &\leftarrow (1-\rho)\mu_{i,t-1} + \rho X_t \\ \sigma_{i,t}^2 &\leftarrow (1-\rho){\sigma^2}_{i,t-1} + \rho(X_t - \mu_{i,t})^T (X_t - \mu_{i,t}) \\ \rho &= \alpha\, \mathcal{N}(X_t, \mu_{i,t}, \sigma_{i,t}) \end{aligned}$$

The advantage of using a mixture of Gaussians is that objects that are new in the scene can quickly be merged with the background and changes in light intensity can quickly be resolved. One disadvantage is that whenever a foreground object (e.g. a hand or a document) remains at the same position long enough it will dissolve into the background. To avoid this problem we add masks to the background model. In the mask, a value of 1 indicates that the corresponding pixel is to be processed as foreground and not updated, while a 0 indicates that the pixel is background and should be updated. This requires fast detection of hands and documents because otherwise they will be dissolved before they are detected.

2.2. Shadow Detection

Whenever a user points at a document, the arm and hand create a drop shadow that the chosen background model does not take into account. Hence, drop shadows will be considered foreground, complicating the exact detection of the user's fingertip. To eliminate this problem we model drop shadows and exclude them from the foreground. Detected shadows are not included in the background mask as experiments showed that separate masking of shadow results in better performance.

We use a method similar to the approach presented in [5]. Each new pixel that is presented to the background model is first checked with the current background Gaussians. If the pixel is considered to be a shadowed value of one of the foreground Gaussians, the pixel is labeled accordingly. The method used for detecting shadows is based on [3].

The goal of the shadow detection in our system is to reduce the foreground to the actual objects that should be detected in a scene. Figure 3 illustrates how the employed algorithm can accomplish this task.

2.3. Skin Detection

Whenever a new object appears in the viewfinder of the camera the respective region will be considered background

Figure 3. Detection of foreground in an input image (top) using only background detection (center) and background detection in combination with shadow detection (bottom). Only non-black pixels are considered foreground.

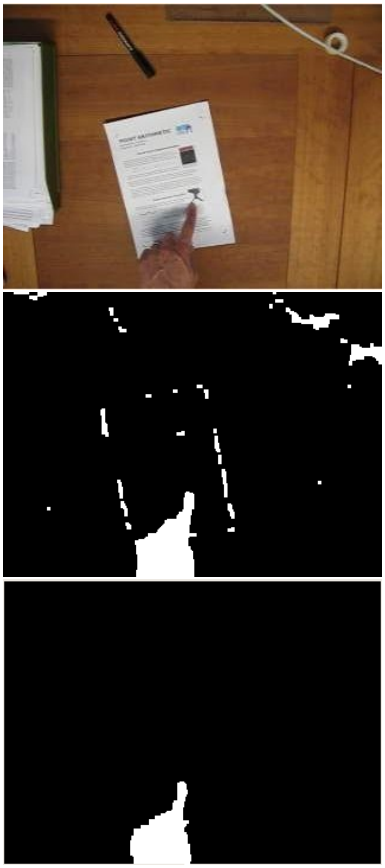

Figure 4. Segmentation of the hand in an input image (top) using only skin-color (center) and skin-color in combination with background detection (bottom).

after a number of frames. In this restricted amount of time, the system has to distinguish hand from non-hand candidate regions. This is done by detecting skin-colored regions. There exist several alternative approaches to detect skin; an overview can be found in [13].

We are using a Bayes classifier with skin probability map as proposed by Jones et al. [4]. The classification of the two sets of pixels involves a histogram which is based on the Compaq Cambridge Research Lab image-database.

An rgb pixel value is then classified according to the ratio $P(\text{rgb}|\text{skin})/P(\text{rgb}|\neg\text{skin})$ obtained from the model, which is compared to a threshold value.

Figure 4 shows the detection of skin in an input image and how the combination of skin and background detection improves accuracy.

2.4. Fingertip Detection

After segmenting foreground from background and detecting skin-colored regions, a matching algorithm verifies the presence of a hand in the segmented foreground. The fast normalized cross correlation [8] is used here.

Once a hand is detected, the exact location of the pointing finger needs to be identified. As the user is usually interacting with the system in front of the desk, the assumption is made that he will always point 'upwards', i.e. away from his body. Hence, the tip of the pointing finger will always be at the highest y-value of the detected hand region. While the image is flood-filled to create a mask for the background subtraction locating the finger tip is done in the same iteration. Based on informal experiments, we examine the eight top-most rows of pixels of the hand region. If their widths are within suitable thresholds the middle of the detected fingertip is considered the region of interest of the user as shown in Figure 5.

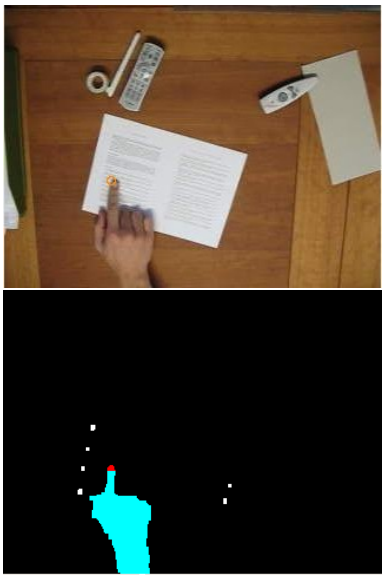

Figure 5. Example of detecting the hand and the pointing finger tip.

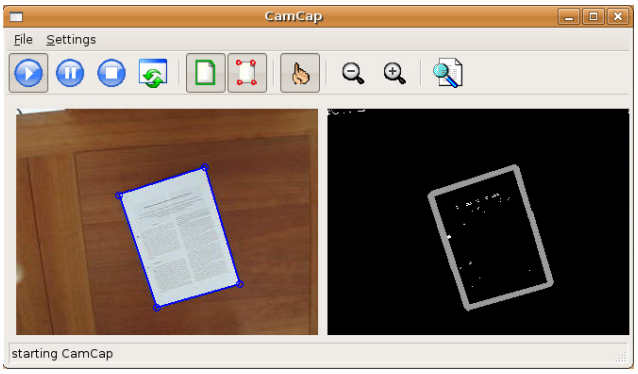

Figure 6. Example of the demonstrator GUI with debug-output to visualize the detection a document by finding gradients in the foreground.

2.5. Document Detection

The first step in capturing a document is to know that a document actually is present in the viewfinder image. To detect a document a number of methods are available including background color differencing and background gradient differencing. Background color differencing compares each pixel color in the viewfinder image to the same pixel in the background image in RGB space. The color difference is the Euclidean distance between the two pixel colors. When the color difference exceeds a certain threshold the pixel is considered to be foreground. Although this approach is intuitive and easy to implement it has a few drawbacks. The first drawback is the sensitivity to noise. If a shadow or a highlight appears in the viewfinder image the pixels inside this noisy region are falsely classified as foreground pixels. Another problem is the fact that, when a document is placed in the viewfinder image of the camera, the lighting will change, causing the camera to adjust its white balance and exposure settings. This will cause a global color change in the viewfinder image which could result in the entire image appearing as foreground.

Because of these problems, another approach was used for detecting possible foreground objects in the viewfinder image. This second approach also uses differencing of the current image and the background image. However, it does not use color information directly to classify a pixel as foreground or background. Instead, it first creates a gradient image of both the background and the viewfinder image using a simple Sobel kernel (-1,0,1) in both horizontal and vertical direction. The next step is subtracting each gradient pixel of the background from the gradient pixels in the viewfinder image. The result is an image with gradients which only appear in the viewfinder image and not in the background image as shown in Figure 6. Gradients that only exist in the background image may be caused by objects being removed from the desk and become negative gradients in the difference image. The difference image is then thresholded to only keep strong positive gradients. The advantage over color differencing is that global color changes do not influence the gradient images and thus will not cause the entire image to be classified as foreground. Also shadows and highlights are removed because they rarely contain any sharp edges. The gradient image now contains all the lines of the foreground objects in the viewfinder image. These objects do not always have to be documents but can also be other items placed on the desk. To reduce the candidate lines to straight lines which could constitute the edges of a document a Hough transform is performed. The result is a set of straight lines, each represented by a start and end point. These lines are then clustered so double lines and close lines are removed. If a document exists in the set of lines it can be assumed that a document is surrounded by four lines. In order to find enclosed areas, we iterate through the set of lines matching end points of one line with the start point of another line. When an end point of a certain line reaches the start point of the first line the algorithm returns the area enclosed by these lines as a document.

2.6. Document Rectification

For each document detected in the viewfinder image a set of corner-points is saved according to [14]. Next, the document detector acquires a high resolution image from the

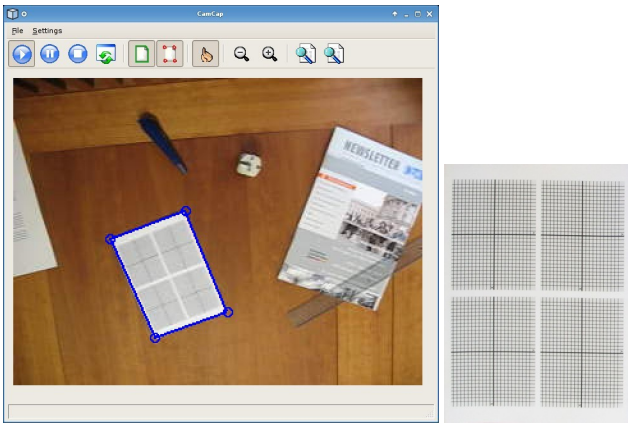

Figure 7. Detected document in the GUI of the system and the high-resolution document image after extraction and rectification.

camera on which the actual document rectification is performed. An example document rectification can be seen in Figure 7. The coordinates of the corner-points of a document are scaled to match the same points in the high resolution image. By scaling these points precision can be lost due to rounding errors or viewfinder deviation with respect to the high resolution image. In order to (re)gain accuracy the document detector will try to detect the exact position of the document corners in the high resolution image. This is done by extracting a patch from the high resolution image for each corner-point. The center of each patch is the scaled location of the corner-point from the original viewfinder image. In this patch edges and lines are detected by using the same methods as for document detection. The crossings of all the lines are calculated and the point with most crossings is considered to be an exact corner-point of the document. This processing step results in less noise and parts of the background at the borders of the captured document. After the detection of the corner-points the algorithm rectifies the image to eliminate distortion caused by projection. For rectification of the document the method in [14] is used which is based on the standard pinhole model to calculate the projection from a space point M to an image point m as shown in Figure 8.

3. Application

The system we present in this paper can serve as a basis for various applications based on the rectified image of a document along with the coordinates of a certain region of interest. The most obvious next step is to find the exact text line at which the user pointed and perform OCR on it. Another use case of the interactive mode of the presented

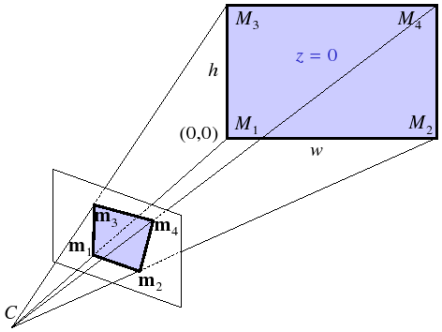

Figure 8. Conversion from image to space points based on the pinhole model.

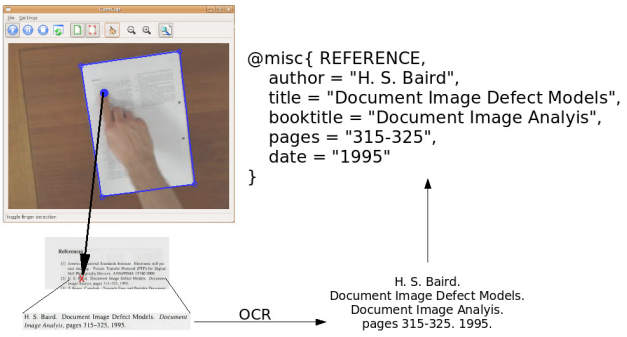

Figure 9. Example for the extraction of bibliographic meta-data: The user points at a reference, a high resolution image is taken and the correpsponding image part is extracted based on the location of the fingertip of the user. Then OCR is performed on the sub-image and the recognized text is processed to obtain a BIBTEX representation of the reference.

system is illustrated in Figure 9: The user points at a reference in a bibliography. First, the paragraph constituting the reference of interest is identified with a layout analysis algorithm. The contained text is then obtained from the respective part of the input image by performing OCR on it. After that, the extracted text of the reference is forwarded to a system that extracts bibliographic meta-data from scientific references[2] [6]. With this information in BIBTEX format, the respective scientific paper can be easily found in digital libraries, such as CiteSeer[3] or CiteULike[4].

[2] http://demo.iupr.org/refrec/
[3] http://citeseer.ist.psu.edu/
[4] http://www.citeulike.org/

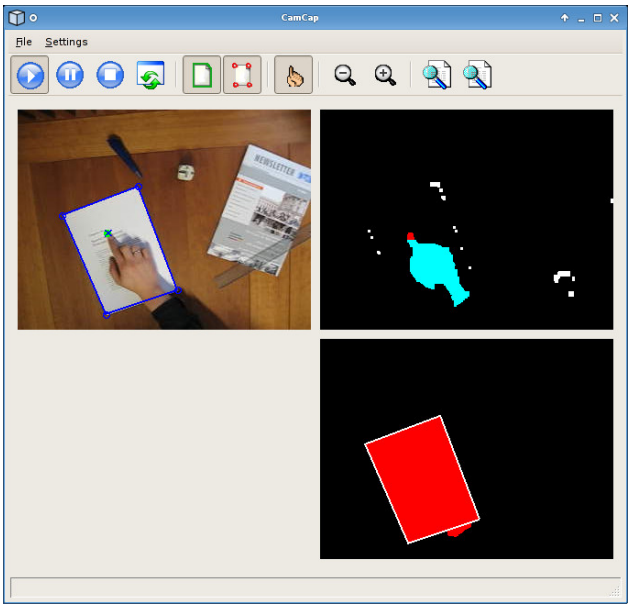

Figure 10. Example of the demonstrator GUI with debug-outputs to visualize the detection of hand/fingertip and document.

4. Summary

We have presented the status of our work on document capture systems. The current prototype supports two modes, i.e. oblivious and interactive capture. In the oblivious mode the user works at his desk as usual and all paper documents that entered the field of view of the mounted camera are archived digitally. In the interactive mode the user can specify a region of interest within a document by pointing at it. The whole document is captured and the region of interest is made accessible immediately.

The system offers a fast, robust and cost-effective way of capturing documents on the physical desktop. The accurate detection allows for real-world applications that bridge the gap to the virtual desktop. Figure 10 shows another example of the demonstrator application in a debug mode that shows the detection of a document and the fingertip of the user.

Acknowledgments

This work was partially funded by the BMBF (German Federal Ministry of Education and Research), project IPeT (01 IW D03).

References

[1] The office of the future. *Business Week*, 2387:48 – 70, 30 June 1975.

[2] J. S. Brown and P. Duguid. *The Social Life of Information*. Harvard Business School Press, Boston, MA, USA, 2002.

[3] T. Horprasert, D. Harwood, and L. Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *Proceedings IEEE ICCV '99 FRAME-RATE Workshop*, 1999.

[4] M. Jones and J. Rehg. Statistical color models with application to skin detection. In *International Journal of Computer Vision*, volume 46, pages 81 – 96, 2002.

[5] P. KaewTrakulPong and R. Bowden. An improved adaptive background mixture model for real-time tracking with shadow detection. In *Proceedings 2nd European Workshop on Advanced Video-based Surveillance Systems (AVBS01)*, Kingston upon Thames, September 2001.

[6] M. Kraemer, H. Kaprykowsky, D. Keysers, and T. Breuel. Bibliographic meta-data extraction using probabilistic finite state transducers. In *Proceedings of ICDAR 2007*, in press.

[7] C. H. Lampert, T. Braun, A. Ulges, D. Keysers, and T. M. Breuel. Oblivious document capture and real-time retrieval. In *International Workshop on Camera Based Document Analysis and Recognition (CBDAR)*, pages 79–86, Seoul, South Korea, aug 2005.

[8] J. Lewis. Fast normalized cross-correlation. In *Vision Interface*, pages 120 – 123, 1995.

[9] M. Piccardi. Background subtraction techniques: a review. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC 2004)*, pages 3099 – 3104, The Hague, Netherlands, October 2004.

[10] A. J. Sellen and R. H. Harper. *The Myth of the Paperless Office*. MIT Press, Cambridge, MA, USA, 2003.

[11] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 1999)*, volume 2, pages 244 – 252, 1999.

[12] C. Stauffer and W. Grimson. Learning patterns of activity using real-time trackingg. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:747 – 757, 2000.

[13] V. Vezhnevets, V. Sazonov, and V. Andreeva. A survey on pixel-based skin color detection techniques. In *Proceedings of the Graphicon*, pages 85 – 92, 2003.

[14] Z. Zhang and L. He. Whiteboard scanning and image enhancement. Technical report, MSR-TR-2003-39, June 2003.