International Scholarly Research Network
ISRN Machine Vision
Volume 2012, Article ID 945973, 15 pages
doi:10.5402/2012/945973

Research Article

An Effective Color Addition to Feature Detection and Description for Book Spine Image Matching

Spencer G. Fowers and Dah-Jye Lee

Electrical and Computer Engineering Department, Brigham Young University, Provo, UT 84602, USA

Correspondence should be addressed to Dah-Jye Lee, djlee@byu.edu

Received 18 August 2011; Accepted 18 September 2011

Academic Editor: W. L. Woo

The important task of library book inventory, or shelf-reading, requires humans to remove each book from a library shelf, open the front cover, scan a barcode, and reshelve the book. It is a labor-intensive and often error-prone process. Technologies such as 2D barcode scanning or radio frequency identification (RFID) tags have recently been proposed to improve this process. They both incur significant upfront costs and require a large investment of time to fit books with special tags before the system can be productive. A vision-based automation system is proposed to improve this process without those prohibitively high upfront costs. This low-cost shelf-reading system uses a hand-held imaging device such as a smartphone to capture book spine images and a server that processes feature descriptors in these images for book identification. Existing color feature descriptors for feature matching typically use grayscale feature detectors, which omit important color edges. Also, photometric-invariant color feature descriptors require unnecessary computations to provide color descriptor information. This paper presents the development of a simple color enhancement feature descriptor called Color Difference-of-Gaussians SIFT (CDSIFT). CDSIFT is well suited for library inventory process automation, and this paper introduces such a system for this unique application.

1. Introduction

Taking inventory is a daunting task in any industry, especially when the number of items reaches into the multimillions, as is the case with most major libraries. It turns into a very challenging and costly task because each item has to be accounted for without the benefit of automation. A comparison done by the On-line Computer Library Center shows that libraries in the United States alone circulate more books every day than the shipping giant FedEx delivers packages. Approximately 5.4 million books are checked out daily from libraries across the US. Furthermore, libraries worldwide hold an estimated 16 billion volumes, and this number continues to grow. Even allocating just one second per book, a full inventory would require over 507 man-years. When equipment such as a barcode scanner is used, each book must be taken off the shelf, its cover opened, the barcode scanned, and then reshelved. Even with such improved technology, the amount of time and labor required is still substantial. Another promising alternative is to use radiofrequency identification (RFID) chips. This approach, however, requires replacing existing call numbers, special labels, or barcodes, constituting a substantial initial cost for a large library. Also, the location resolution of RFID is typically not accurate enough to determine proper book order in a row, only whether or not the book is in the general area.

A low-cost shelf-reading system to improve the library inventory process was proposed in our previous work [1]. Figure 1 shows the design of the system. A hand-held device such as a smart phone or an imaging device equipped with wireless communication capability is used to capture high-resolution digital images of the spines of books on the shelf. The captured images are transmitted wirelessly to a server system for processing. The server system extracts features from the images and calculates descriptors to match features in the captured images to features stored in a central database in order to identify misplaced or missing books. It then generates a report and a list of actions or graphical instructions for the user to either remove or reshelve the misplaced items. This solution eliminates the need of manually removing, scanning, and reshelving the books, reducing the time and cost of a full inventory, as well as increasing its accuracy.

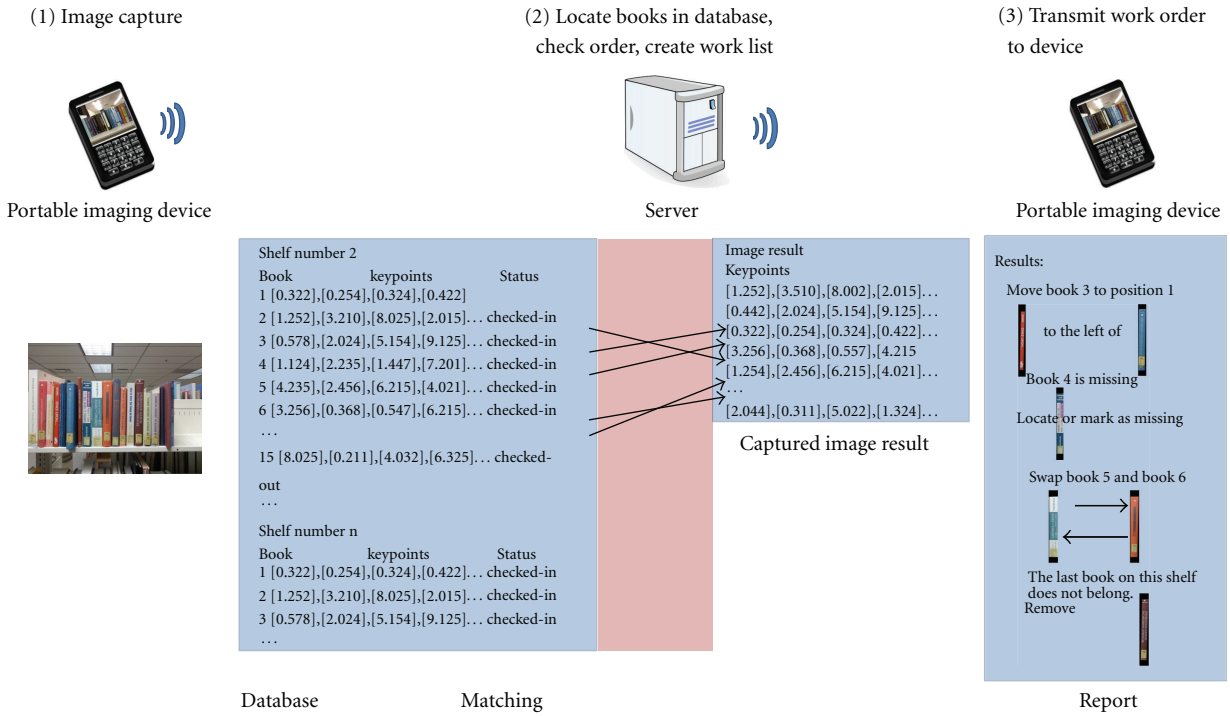
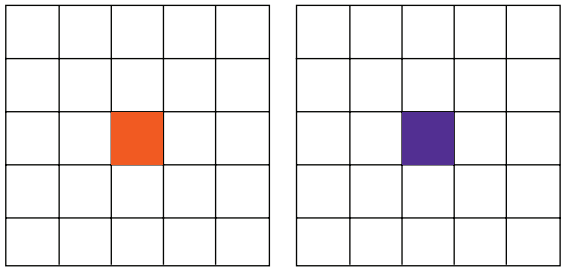


Figure 1: Library book inventory system.



(a) RGB (255, 90, 0) (b) RGB (0, 160, 255)

Figure 2: Two example feature keypoints. The center pixel of both keypoints has the same grayscale value of 129.6 using (1).

No changes to existing books are required, so the solution is more cost effective than the alternative approaches and technologies mentioned previously.

All functions shown in Figure 1 can be implemented easily using existing technologies except feature detection and feature description, which are the main challenges and the focus of this work. In recent years, machine vision technology has been used to automate numerous applications. As examples of the importance of using image features, machine vision systems have been designed for factory automation tasks such as versatile online visual inspections [2, 3], lumber production [4], microscopic imaging for biology [5], closed-loop online process control [6], computer-aided medical diagnosis of antibodies [7], and breast cancer detection [8]. Two vital parts of these systems are how to detect image features (feature detector) and how to describe them (feature descriptor) for pattern matching.

A large percentage of book spines contain color information. However, the large majority of feature detectors and feature descriptors in use in today's applications rely upon grayscale features alone. Grayscale images do not contain as much information as color images [9]. However, color images are not often used for feature detection because of the difficulty in representing 3D color information or processing an image with three data channels [10]. Three channels of data implies three times as much computation, and it can be difficult to interpret the results from a color feature detector since the output from all channels must be recombined into a single disparity value. For these reasons, some of the most recent feature detectors and descriptors (including most color feature descriptors) still use grayscale images for feature detection [11].

It has been shown that color feature detectors find more features and provide more unique information about those features than grayscale detectors [12, 13]. Grayscale detectors and descriptors cannot take advantage of important color edges found in many computer vision tasks because they do not process color information. More importantly, using grayscale feature detectors for the book inventory process will not yield accurate results, especially for those book spine images that have high color contrast but fairly uniform grayscale values.

While some attempts have been made to generate color feature descriptors, these attempts still utilize only a grayscale feature detector, providing little added benefit and once again omitting color edges from descriptor computation. We introduced a color Difference-of-Gaussians (DoG) feature detector to address this feature detection issue in [13, 14]. It is a simple enhancement to the grayscale DoG algorithm

Figure 3: (a) A photometric invariance test image and (b) its Y channel, (c) C_b channel, and (d) C_r channel.

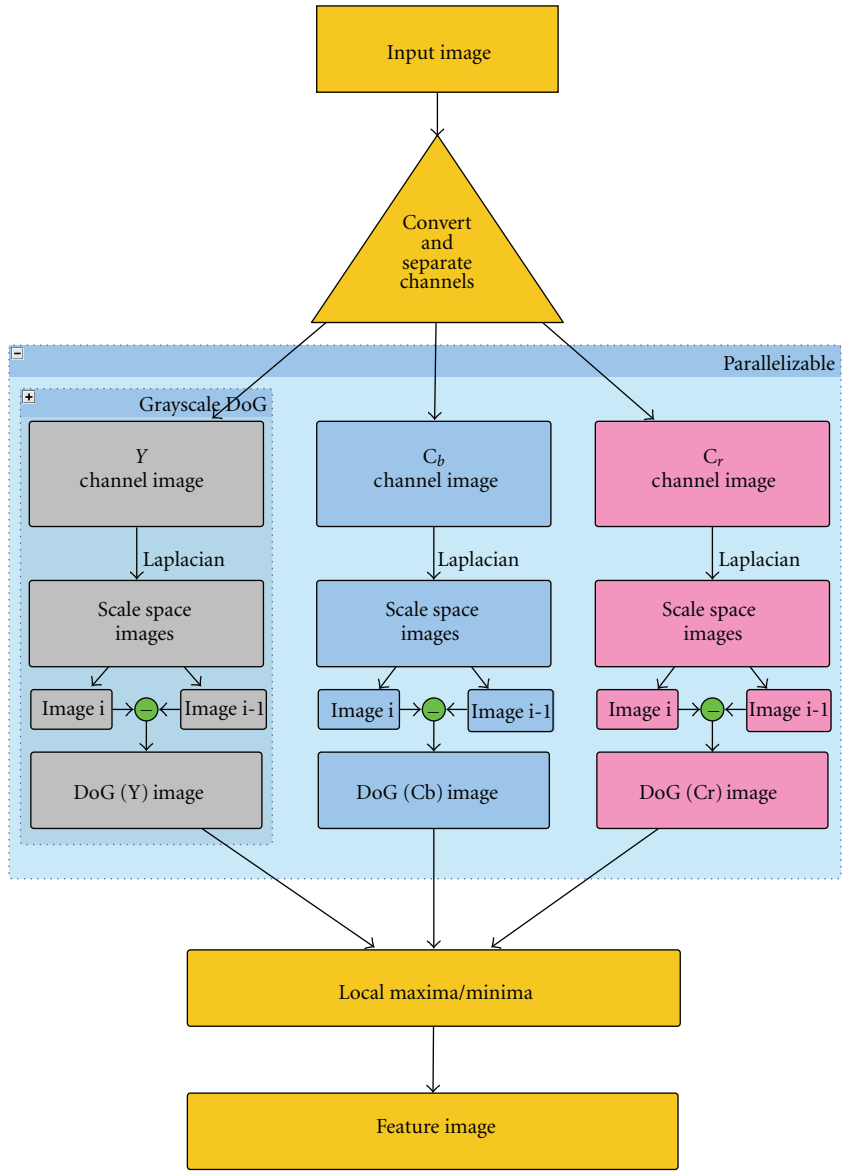
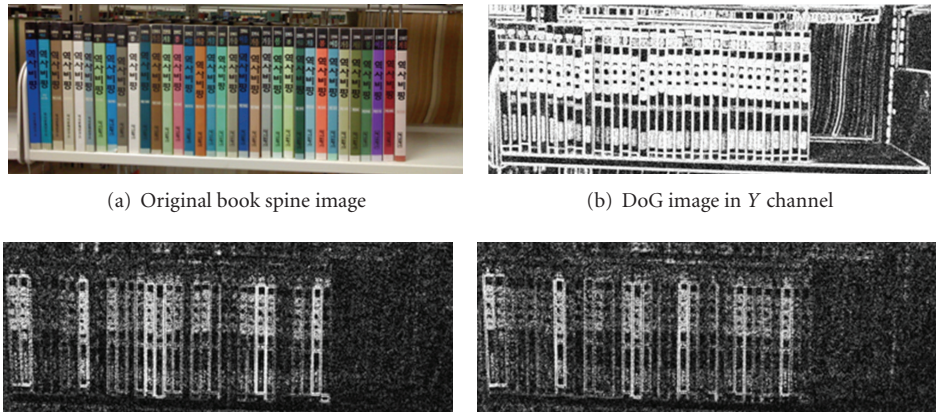


Figure 4: Flow of the color DoG algorithm. The equivalent steps of the grayscale DoG algorithm are labeled. Color DoG has the advantage that the entire center square is parallelizable into three separate threads so processing takes no longer than the single-channel DoG.



(a) Original book spine image

(b) DoG image in Y channel

(c) DoG image in C_r channel

(d) DoG image in C_b channel

Figure 5: The source image in each channel is independently convolved with a Gaussian kernel at differing values of σ, and then adjacent σ images are differenced to produce a Difference of Gaussians. The Y channel DoG is shown in (b), the C_r in (c), and the C_b in (d). Different strong edges are visible in each channel. A grayscale feature detector would only find those visible in the Y channel and miss the very obvious features in the chrominance channels.

that operates on color images and provides results that are guaranteed to improve upon the feature detection results of the grayscale DoG.

This paper presents the extension of our color DoG feature detector to feature descriptors for the task of pattern matching in an automated library bookshelf inventory system. Our new color feature descriptor, Color Difference-of-Gaussians Scale-Invariant Feature Transform (CDSIFT), computes SIFT descriptors for color features found by the color DoG. The CDSIFT algorithm then creates a 384-element descriptor similar to other color feature descriptors [9–12, 15–17], but without the additional computations required to compute photometric color invariants.

Section 2 presents previous work in feature detection and photometric invariants and discusses the advantages of the proposed CDSIFT algorithm. Section 3 describes the color DoG algorithm, our CDSIFT implementation, and its mathematical justification. Using CDSIFT, we have developed a low-cost automated shelf-reading system for library inventory. The demonstration system was built using a standard desktop computer, with bookshelf images taken using a 3-megapixel digital camera found in a typical cellular phone. The images were taken while holding the camera without the assistance of a tripod, and with only the standard lighting provided by the library overhead lights. Our initial tests on a small real-world dataset taken from actual library shelves show excellent pattern matching results for book identification. The system is described in Section 4. In Section 5, experimental results are presented and compared to the output of the current best performing photometric-invariant color descriptor, c-color-SIFT. Finally, Section 6 concludes the research and outlines future work.

2. Background

2.1. Feature Detection. In 1987, Biedermann published his work on the human ability of object recognition. His study found that the corners where lines meet on an object are paramount to our ability to recognize that object. When corners were occluded in an image, the time it took for a human to recognize the object went up drastically [18]. Accepting this fact, research into computer vision commonly begins with corner detection. Harris and Stephens in 1988 clearly explained the problem of tracking edges and developed the Harris corner detection algorithm [19]. From this, the majority of corner detectors followed the same method. An image is represented as a 2-dimensional matrix, and a kernel is convolved with the matrix resulting in an increased definition of edges. These edges are then refined and thresholded, and then endpoints and corners are found and returned as feature points [19–25].

Grayscale corner detectors (such as [19–22, 26–29]) return corners as a location (x, y) and a magnitude denoting the "strength" of the corner. A basic corner algorithm calculates this strength as the intensity-normalized magnitude of the gradient along both the x- and y-axes of a grayscale image. Rosin proposed that augmenting this strength value with more information about the feature would improve correlation of features (feature matching) [30]. This added feature information creates a unique descriptor for each feature point, and correlation of features between two images can then be accomplished by choosing as a match the two features with the smallest disparity between descriptors.

2.2. Color Feature Invariants. Color has been found to be very important for distinction between objects [31] and forms a large area of interest in feature descriptor development (see [9, 12, 32–34]). Laptev compared the performance of color and grayscale histogram descriptors and found that on natural scenes, color outperformed grayscale, and on all other scenes, a combination of color and grayscale descriptors performed the best on all test classes [35]. Almeida et al. conducted an evaluation of color feature descriptors to find the best descriptor for content-based image retrieval

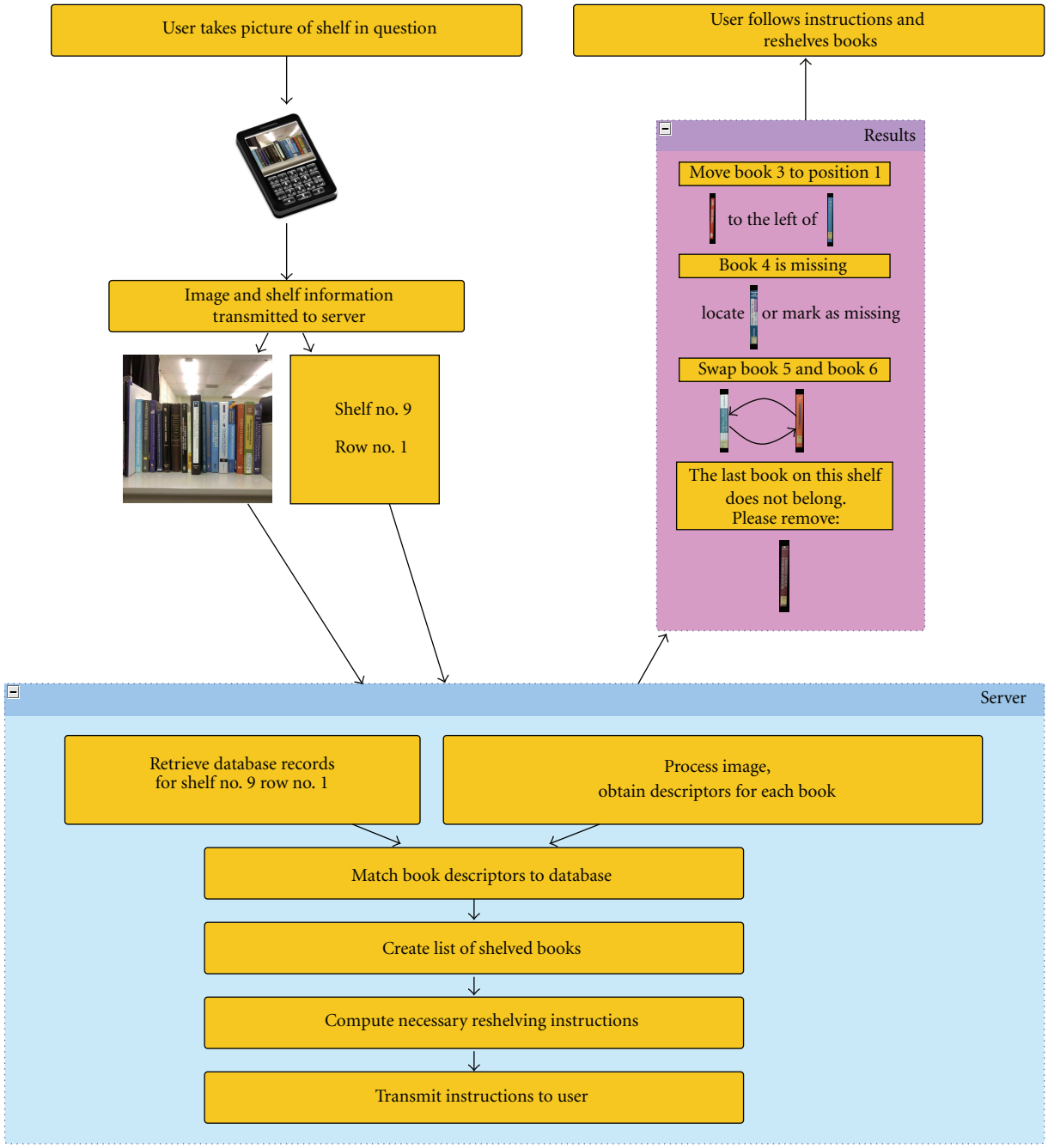
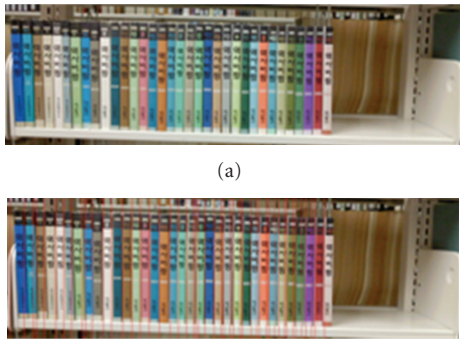


Figure 6: Example process flow of the implemented book inventory system. The user is only required to submit a picture of a book shelf with information about which shelf is being queried, and the system will return simple instructions to follow.

[36]. Although the evaluation was of color descriptors, a grayscale Difference-of-Gaussian operator was used to detect features. Burghouts and Geusebroek evaluated color invariants and found that performance of color invariants far exceeds performance of intensity-only invariants [11]. Their research also used intensity-based feature detectors to detect features and then computed color descriptors about those feature points. Color invariance models such as that developed by Geusebroek et al. make it possible to combine color information with grayscale information [37]. In fact, Abdel-Hakim and Farag [38] used these invariants to develop a color version of SIFT. Their results show an increase in performance compared to the original SIFT.

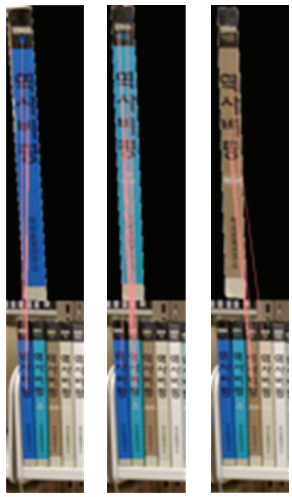
In a later paper, van de Sande et al. also compared the performance of color invariants for the task of object recognition given varying lighting and viewpoint changes [12], but they used only grayscale feature detectors. Their results showed grayscale SIFT performed about as well as the color-invariant methods. This is expected due to the fact that features were found using a grayscale feature detector. No



(a)

(b)

Figure 7: (a) Example image submitted to the server from a cell phone camera of shelf 1, row 1. (b) A line finding algorithm detects vertical lines as divisions between books in order to subsample the image.



(a) (b) (c)

Figure 8: The input image is divided into subimages based on book divisions found by the line finding algorithm and then searched for feature descriptors. These descriptors are then matched with descriptors for books in the database to find book coordinates. (a) Shelf 1, row 1, book 1 subimage and search results. (b) Shelf 1, row 1, book 2. (c) Shelf 1, row 1, book 3.

color edges were used for features, and; hence, features that have strong color edges but weak intensity edges were not detected or used.

2.3. Photometric Invariants. Photometric invariants have been pursued as a way to incorporate color information while removing the effects of shadows, highlights, and specularities. However, because photometric invariants are based on derivatives of the color channels, shadow and highlight areas pose a problem because the derivatives become unstable in such situations. Van de Weijer et al. developed photometric quasi-invariants to circumvent this problem [39, 40]. Photometric invariance, namely, that obtained by van de Weijer et al. [39], requires computation of a color derivative using the application of Gaussian kernels.

Upon closer inspection of the photometric invariants' algorithm, two important observations are made. First, the invariant calculation separates hue from intensity. Second, both derivatives in the invariant calculation are normalized by the square root of sum of squares of the R, G, and B channels. In essence, the majority of the reflectance information caused by shadows, shading, and specularities is located in the intensity information and can be removed from an RGB image by calculating hue and removing intensity. In RGB space, removal of intensity information requires normalization by the square root of sum of squares of the R, G, and B channels. While color-invariant performance does present a large improvement over intensity-only invariant performance, color invariants require a large number of computations to calculate the invariants at each pixel, and even the more stable quasi-invariants are unreliable or unstable in low-light (low-intensity) or low-color (low-saturation) situations [39].

Burghouts et al. compared a number of color descriptors and color invariants and found c-color-SIFT to perform the best [11] under varying illumination color and direction, and various viewpoint changes. The c-color-SIFT approach returns two color-invariant values per pixel, one for the yellow-blue channel and another for the red-green channel. c-color-SIFT includes the orientation and magnitude returned from the grayscale invariant and these two color invariants, producing a 384-element descriptor. Using their results as a baseline, we then compare our algorithm against c-color-SIFT in order to avoid unnecessarily re-evaluating other algorithms that did not perform as well as the top-ranked algorithm (c-color-SIFT) in [11] and focus this paper on our library inventory application.

2.4. The Advantages of CDSIFT. Two keypoints in an image may have drastic chrominance differences but almost equivalent grayscale values. Figure 2(a) shows an example feature point of an image. The RGB value of the keypoint is (255, 90, 0), and the surrounding points are white (255, 255, 255). Using the standard equation

$$Y = 0.3 \times \mathrm{R} + 0.59 \times \mathrm{G} + 0.11 \times \mathrm{B} \tag{1}$$

to convert RGB images to grayscale, the resulting intensity of the center pixel is 129.6, and the surrounding white pixels are now 255. Figure 2(b) shows a similar feature point, with a very distinct color. The RGB value of the keypoint is (0, 160, 255), but when it is converted to grayscale using (1), it also has an intensity value of 129.6. Although these two keypoints are very distinct in color, their grayscale SIFT descriptors would be identical, and matching would fail. By providing color information, the difference between distinct keypoints whose intensity descriptors are similar can still be retained in the chrominance descriptors.

We developed a new color descriptor called CDSIFT for this pattern-matching application. It processes images in an intensity-separated color space such as YC_bC_r that does not require additional computation due to the fact that

(a) (b) (c)

Figure 9: Bookshelf images from the dataset. (a) An image of the shelf was taken and saved as the original, (b) another one was then taken with book(s) removed, and (c) another one taken with book(s) misplaced.

Figure 10: Images of each individual book were created from the original image (Figure 9(a)) and the resulting CDSIFT keypoints and descriptors stored in the database.

most cameras already output color data in this format. The use of intensity-independent YC_bC_r color space provides shadow, highlight, and specularity invariance similar to that offered by photometric invariants and photometric quasi-invariants but without the excessive computations required by photometric color invariants.

Although based on a simple concept, the CDSIFT algorithm works well on color images and grayscale or low-saturation images, and CDSIFT descriptors do not become unstable in low-light or low-color situations because the movement to intensity-independent color space does not require derivatives of the chrominance channels. Since CDSIFT uses both color and intensity edges for feature descriptors, it still uses important grayscale features if color features are unusable.

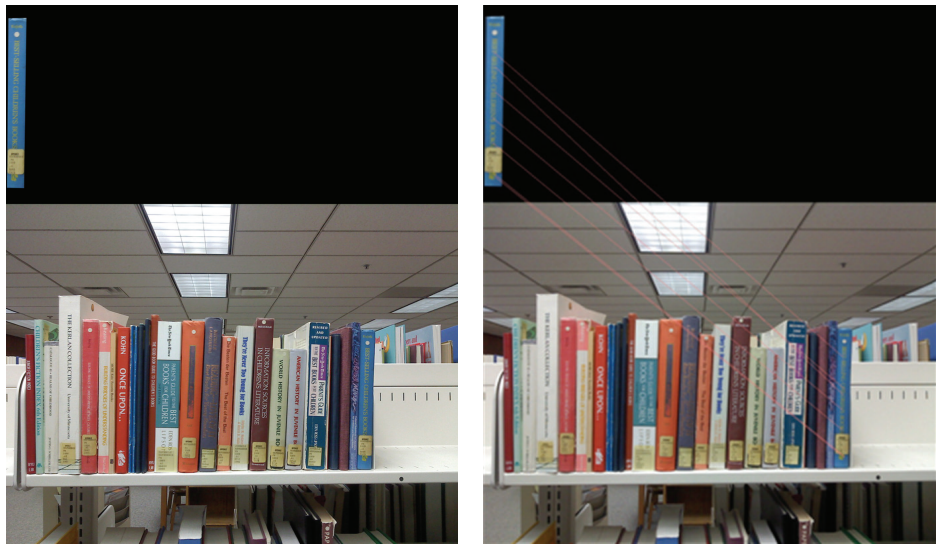
3. Algorithms

The proposed CDSIFT algorithm uses our color Difference of Gaussians as feature detector [13] and calculates scale-invariant feature transform feature descriptors using both color and grayscale edges. This section discusses each of these steps.

3.1. Color Difference of Gaussians. The proposed color DoG algorithm takes as input a YC_bC_r, intensity-separated image. For each pixel,

$$Y = K_r \times \mathrm{R} + (1 - K_r - K_b) \times \mathrm{G} + K_b \times \mathrm{B},$$
$$P_b = \frac{1}{2}\frac{\mathrm{B} - Y}{1 - K_b}, \qquad P_r = \frac{1}{2}\frac{\mathrm{R} - Y}{1 - K_r}, \tag{2}$$

where K_r and K_b are constants derived from the definition of the RGB color space. Since processing for our book inventory



(a) (b)

Figure 11: Matching results of CDSIFT and c-color-SIFT for book 9 of the shelf 2 test set. CDSIFT found 13 matches. Upon visual inspection, 12 are correct. c-color-SIFT did not find any viable matches.

task is performed on images coming from digital cameras, the ITU-R BT.601 standard is used which defines $K_b = 0.114$ and $K_r = 0.299$. P_b and P_r are the analog equivalents of C_b and C_r, respectively.

Because the color DoG also uses the same intensity information as the grayscale DoG, it guarantees to find at least the same features as the grayscale DoG algorithm that operates on an intensity-only image, along with additional strong features from color edges. Depending on the image, it often finds some features that are visible in the grayscale image but are more apparent in the color channels, so it is able to report back more useful information than the original grayscale DoG.

One of the main motivations for photometric invariants is resistance to shadows and highlights. Figure 3 shows an example to demonstrate the effect of shadows and highlights in color-edge detection. The gradient blue edge causes a serious problem with edge detection in the RGB space because the highlights cause values in all three channels (R, G, and B) in the gradient to differ, reducing the benefit of color information. The photometric-invariant method takes care of these issues by removing shadows before computing color information. In a similar, but more computationally efficient manner, the color DoG transforms the image to an intensity-independent color space to successfully reduce this effect. Figures 3(b), 3(c), and 3(d) show the separated Y, C_b, and C_r channels of 3(a). The shadow simulated by the gradient is significantly reduced in the C_b channel and completely removed from the C_r channel, leaving a very obvious color edge. Converting to an intensity-separated space provides invariance to shadows and highlights without the intense computations required to compute photometric invariants or quasi-invariants.

3.2. Color DoG Creation. Figure 4 shows the steps in the color DoG algorithm. The grayscale (Y) DoG is still a part of the color DoG, but two additional channels (C_b and C_r) are computed independently to provide color information. Once the image has been converted to YC_bC_r color space (Most digital cameras output this format and computation for this conversion is not needed.), it is separated into three distinct channel images (Y, C_b, and C_r). Different calculations in the two chrominance channels are based on color, not intensity. The channel images are then converted to scale-space by convolution with a Gaussian. The scale-space image is defined by Lowe [41], for a grayscale image, as

$$L(x, y, \sigma) = \mathrm{G}(x, y, \sigma) * I(x, y), \tag{3}$$

where

$$\mathrm{G}(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}. \tag{4}$$

For the color DoG, (3) is expanded to

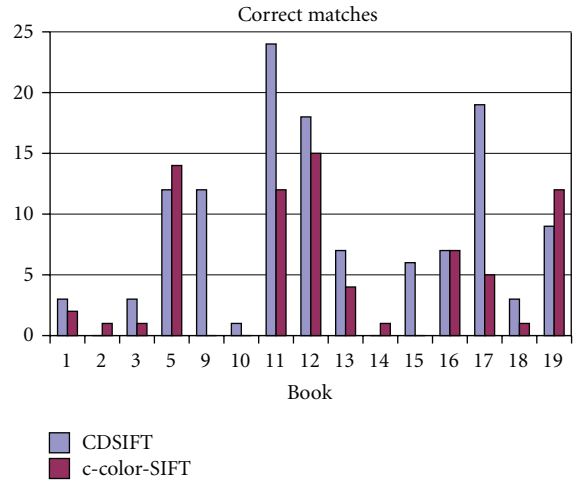
$$L_\lambda(x, y, \sigma) = \mathrm{G}(x, y, \sigma) * I_\lambda(x, y), \tag{5}$$

where I_λ is the current channel image, $\lambda \in \{Y, C_b, C_r\}$. Two of these scale-space images (in the same channel but with differing values of σ) are then differenced to produce a scale-space extrema image:

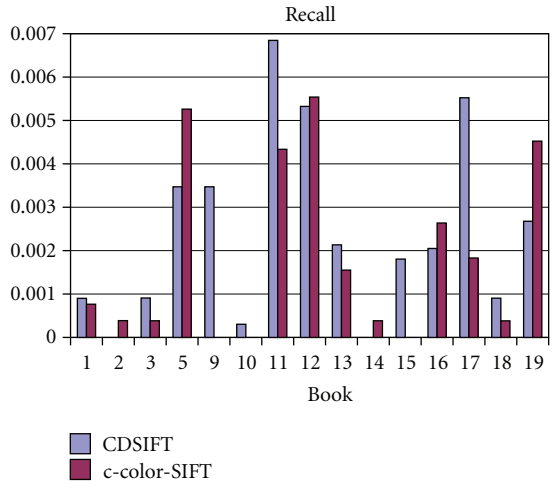
$$\begin{aligned} D_\lambda(x, y, k\sigma) &= (\mathrm{G}(x, y, \sigma) - \mathrm{G}(x, y\ (k-1)\sigma)) * I_\lambda(x, y) \\ &= L_\lambda(x, y, k\sigma) - L_\lambda(x, y, (k-1)\sigma). \end{aligned} \tag{6}$$

These extrema images $D_\lambda(x, y, k\sigma)$ are then searched for maxima/minima peaks and compared with the extrema images at scale-spaces above and below ($D_\lambda(x, y, (k-1)\sigma)$, $D_\lambda(x,\ y,\ (k+1)\sigma)$) to find global maxima/minima peaks. These maxima/minima are then saved as the extrema points at which feature descriptors for CDSIFT will be calculated.

(a) Color image



(b) Correct matches



(c) Recall

Figure 12: Performance comparison on Shelf 2.

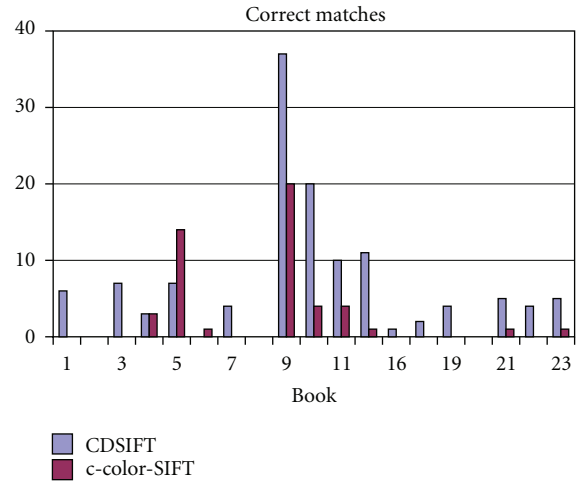
Figure 5 shows an original color image taken from a library bookshelf and the features found using the color Difference of Gaussians. While the Y channel DoG image 5(b) easily detects edges between books and letters, many distinctly colored books appear identical. However, the color differences are apparent in the C_b 5(d) and C_r channels 5(c). In essence, the intensity image does an excellent job of picking up intensity edges in the image, while the C_b and C_r channels detect valuable color edges in the image.

3.3. Feature Point Selection. Histograms of the three DoG images are computed to find peaks. The histogram of each channel image is independently searched for local peaks or valleys. If a minima or maxima is found at a given (x, y) pixel, the other two channel images are checked at the same (x, y) location, and the largest peak or valley of the three channels at the given location is stored. The algorithm also saves the information of which channel (Y, C_b, or C_r) contains this peak or valley.

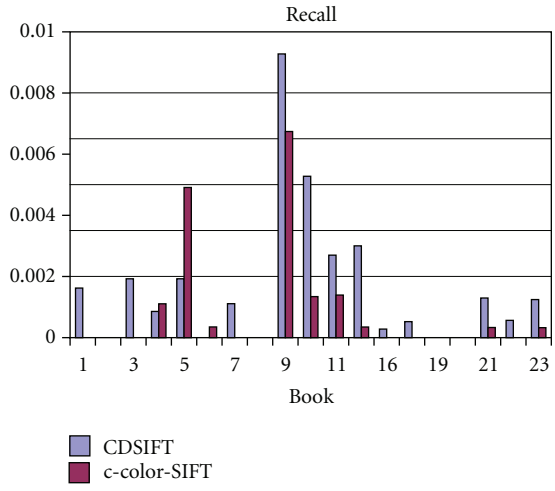
The resulting feature points (or keypoints) are returned as (x, y) pairs with an associated *(strength, channel)* pair. In this way, color-distinct features are detected without calculating a color invariant and without losing color information.

3.4. Color Feature Descriptor. Once the feature space maxima have been detected, descriptors for each of the maxima are computed. Numerous transformations have been used to develop descriptors for unique feature identification. Because the entire benefit of a descriptor is its uniqueness when compared to nonmatching feature points, any transformation that in some way provides a repeatable measure of individuality to a feature point can be useful for feature matching. SIFT uses the histogram of gradient orientations and magnitudes of a small window around each keypoint as a descriptor [42]. SURF utilizes sums of 2-dimensional Haar's wavelet transforms of regions around a keypoint [41]. CDIKP [43] uses a projection kernel to develop a unique descriptor. The proposed CDSIFT algorithm computes SIFT

(a) Color image



(b) Correct matches



(c) Recall

Figure 13: Performance comparison on Shelf 5.

descriptors based on orientation and gradient magnitude from the three Difference-of-Gaussians channels (Y, C_b, C_r). This approach yields equivalent features to those found in the grayscale image and additional descriptors for the C_b and C_r channels. CDSIFT calculates descriptors in the same manner as SIFT, which we will review here. Gradient magnitude and orientation are calculated from the scale-space image corresponding to the scale at which the feature was originally found. The magnitude and orientation are calculated as

$$m(x,y) = \sqrt{(L_\lambda(x+1,y) - (L_\lambda(x-1,y))^2 + (L_\lambda(x,y+1) - (L_\lambda(x,y-1))^2},$$
$$\theta(x,y) = \tan^{-1}\frac{(L_\lambda(x,y+1) - (L_\lambda(x,y-1))}{(L_\lambda(x+1,y) - (L_\lambda(x-1,y))}. \quad (7)$$

Using this method, CDSIFT creates an orientation histogram based on the orientation and magnitude of the pixels in a small 16×16 window surrounding the feature point detected by the color DoG feature detector. The peaks or valleys in the histogram represent dominant local gradients, and the highest peak is used as the dominant orientation for the entire descriptor. The orientations of the surrounding points are normalized by this dominant orientation to provide rotational invariance. Next, the region around the keypoint is divided into 4×4 blocks. In each of these 16 quadrants, orientation histograms with 8 orientation bins are calculated for the pixels in the quadrant. The resulting 8 bins and values for each of the 16 quadrants are stored as a 128-element feature descriptor vector. This process is then repeated in the C_r and C_b channels to form two additional 128-element descriptor vectors.

These three 128-element descriptor vectors are concatenated to form a 384-element descriptor. In addition, the channel in which the keypoint was detected is saved along with the descriptor. By saving the keypoint's channel

identifier, matching can be performed more quickly on only the 128-element vector corresponding to the keypoint's channel, rather than the entire 384-element vector if comparison speed is an issue. In the results shown below, the entire 384-element vector is used.

4. Implementation

4.1. Feature Matching. The goal of color feature matching is to obtain important color information to aid feature discrimination. Matching is performed using best-bin-first search. For each keypoint in the first image, CDSIFT computes the Euclidean distance between the keypoint's descriptor and every descriptor in the second image. The keypoints are then sorted by distance. If the closest match (smallest distance) is an outlier (the distance between the keypoint and the closest match is less than k times the distance between the keypoint and the second closest match), the match is recorded. For this paper, $k = 0.6$. If the nearest keypoint match is not an outlier, the match is considered uncertain, and the algorithm moves on to the next keypoint in the first image. Although more sophisticated methods of matching could be employed to provide even better results, we found for our system that this best-bin-first matching method provided more than adequate results for our proof of concept pattern analysis system.

4.2. Library Inventory System. In order to test proof of concept, we designed and built a prototype automated library inventory system. A standard desktop computer (AMD Athlon II X4) was used as the server running a MySQL database that is accessible over the Internet. To populate the database, a ground-truth image of each library shelf was taken when all of the books in the shelf were known to be correctly shelved. This image was then sliced into individual book images, and each book image was searched for feature descriptors. The database stores a record of each shelf, row, and book. Each book record in the database contains a list of feature descriptors associated with that book.

When inventory is taken, an employee takes an image of a book shelf with a digital camera and uploads the image, along with the shelf number and row number, to the server (a simple smartphone application will be developed to automate this task). The database system pulls up a record of all books belonging in the specified shelf and row number, along with the book order and individual book feature descriptors. The entire inventory process is illustrated in Figure 6.

A shelf used in the testing of this system is shown in Figure 7(a). The inventory system uses a line finding algorithm to approximately determine book edges. Because the algorithm knows the expected position of the first book, it only needs to search in the near vicinity of the expected location to determine if the book is in place or missing. The result of the line finding algorithm is shown in Figure 7(b). The algorithm computes feature descriptors for the region of the image consisting of the first six books (only in the vicinity of the targeted book) and matches these descriptors to the descriptors stored in the database for the first book.

Figure 8(a) shows the results of searching for the first book in the subimage computed from Figure 7. Fourteen features were matched, and all fourteen features are located on the correct book. The inventory algorithm searches through the matches, and, if the majority of matches are located (x coordinate) within three vertical lines detected by the line finding algorithm (the average width of a book), it considers the book correctly identified. The average location of these inliers (matched feature points) in the input image then determines the coordinates of the first book, and the subimage is moved to the right to the next book to be identified. Figure 8(b) shows the results of searching for the second book in the shelf (according to the database ordering). Again, a majority of inliers are found between three detected lines, and the coordinates of book 2 are computed.

Table 1: Image coordinates for each book are sorted and compared to the order specified in the database to determine if books are misshelved.

Book	Image coordinates	Database order	Status
1	(15, 45)	1	Correct
2	(30, 45)	2	Correct
3	(50, 40)	3	Correct
5	(63, 52)	5	Misshelved
4	(72, 45)	4	Misshelved

Once all books in a shelf row have been found (or marked as missing), their computed coordinates are compared to compute the current ordering in the input image, as shown in Table 1. The x coordinate of book 5 was computed as less than the x coordinate of book 4, which results in the system identifying that books 4 and 5 are not in the proper order, and their status is marked as misshelved. At this point, simple instructions can be returned to the user on how to properly reshelve the books or remove books that do not belong.

5. Results

5.1. Feature Matching. To test the performance of our algorithm, a dataset was created from existing bookshelves in the university library. Images were taken using a standard cellular phone camera (2048×1536 resolution). Figure 9 shows a sample bookshelf dataset from our test set. First, an image of the complete bookshelf was saved (Figure 9(a)). Next, certain books were removed (Figure 9(b)) or rearranged, and the picture was taken again (Figure 9(c)).

The system created individual book-spine images using the original shelf dataset image (Figure 9(a)) exactly as it would be in a real inventory application. Figure 10 shows seven individual books obtained from the original image Figure 9(a). The CDSIFT algorithm then computes feature and descriptor lists for each of these individual book images, which are then saved in the database. This dataset is available from the author upon request.

5.2. Book Matching. Figure 11(a) shows the resulting matches found by the c-color-SIFT algorithm on the test set Shelf

Table 2: Book matching results for c-color-SIFT and CDSIFT on a number of shelf datasets.

Dataset		Shelf 1	Shelf 2	Shelf 3	Shelf 5	Shelf 8	Shelf 9	Shelf 11
No. of Books		34	15	14	19	28	16	17
No. Matched	CDSIFT	**7**	**13**	**11**	**13**	**17**	**14**	**16**
	c-color-SIFT	0	12	5	8	9	8	9
Average Recall	CDSIFT	**0.00010**	**0.00242**	**0.00243**	**0.00166**	**0.00075**	**0.00688**	**0.00707**
	c-color-SIFT	0.00001	0.00187	0.00087	0.00089	0.00030	0.00185	0.00135
Average Accuracy	CDSIFT	**29.76%**	**85.82%**	**83.15%**	**71.05%**	**59.01%**	**87.12%**	**91.32%**
	c-color-SIFT	2.94%	80.00%	33.00%	45.00%	32.14%	49.65%	52.94%

2 (shown in Figure 9(a)) when attempting to locate book 9. The book being searched for in the image is displayed on top of the input shelf image. Matches are drawn as lines from the first image to the second, indicating where the feature point was matched. Book 9 is an especially important case because the book has been misshelved. Book 9 should be the 9th book from the left edge of the image shown in Figure 9(a). Instead, it has been placed at the end of the shelf (Figure 9(c)). In this situation, if book 9 is properly matched, it can be correctly identified as misshelved and commands given to the user to properly shelve the book.

c-color-SIFT does not match any keypoints between these two images. Figure 11(b) shows the resulting matches found by our CDSIFT algorithm on the same image/shelf pair. CDSIFT matched 13 keypoints for this book. Upon visual inspection, 12 of these 13 matches are correct, resulting in a correct book identification. Figure 12 shows the results of both algorithms over the entire Shelf 2 dataset. CDSIFT is able to identify a majority of the books in this set. Recall in Figure 12(c) is defined as

$$\text{recall} = \frac{\text{no. of correct matches}}{\text{no. of features}}. \tag{8}$$

Figure 13 shows the performance of CDSIFT and c-color-SIFT on the Shelf 5 dataset. CDSIFT outperforms c-color-SIFT on all books except books 5 and 6. CDSIFT found 7 matches on book 5, all 7 of which were correct, resulting in a correct book identification, but c-color-SIFT found 14 features, 14 of which were correct. CDSIFT does not find any matches on book 6, while c-color-SIFT finds one. However, c-color-SIFT cannot identify books 1, 3, 7, 16, and 17, whereas CDSIFT is able to identify each of them.

Figure 14 shows the performance of CDSIFT and c-color-SIFT on the Shelf 9 dataset. Shelf 9 contains books with glossy covers and artwork. This provides a much denser feature space and resulted in a large increase in overall features and matches found by both algorithms.

However, c-color-SIFT has difficulty in matching many books in this dataset. Not surprisingly, it is the books with little color variation that cause problems for c-color-SIFT. This is due to the normalization required for photometric invariants. In situations where color and intensity are low, such as in the black book with gold text (book 1), the normalization procedure removes too much information, making the descriptors useless and matching impossible. The same happens for the next book, with white letters. Because the color channels do not provide any information, normalization affects the entire feature vector for c-color-SIFT. In these situations, CDSIFT is still able to perform optimally because the intensity channel is processed independently from the chrominance channels. Looking at the CDSIFT descriptors for features in these two books, the chrominance descriptors do not contain much information, but there is enough information in the grayscale descriptor to still make correct matches.

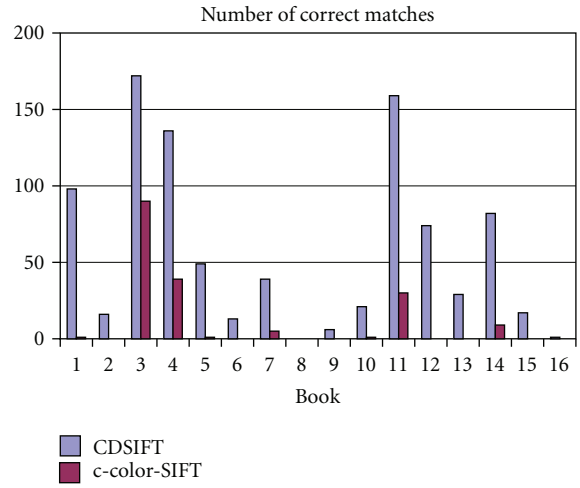
For each test shelf, the average accuracy and average recall was computed and can be found in Table 2. For accuracy, each book in a shelf's test set is checked with a test image (such as the one shown in Figures 9(b) and 9(c)) with missing and/or misplaced books. The Shelf 1 dataset (Figure 7) is an exceptionally difficult dataset. It features books with identical titles, and the only varying characteristic is a solid block of color along the book spine; this color is repeated on multiple books. In this situation, an intensity-only descriptor (such as standard SIFT) is useless. c-color-SIFT was unable to match any books in the set because of normalization. CDSIFT's use of both intensity and color information, however, allows it to identify a few books in this set. With additional tuning of the CDSIFT algorithm parameters, even more matches could be obtained.

It is important to note that these results only report direct matching results comparing the entire input image, rather than a subimage as explained in Section 4.2, and so these results do not take advantage of the ordering and status information of books in the database to assist book identification. In the final book inventory system, the matching performance is improved using this additional information from the database.

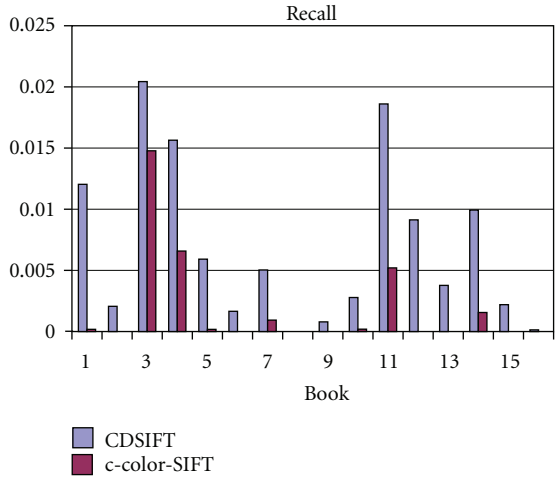
6. Conclusion

This paper has presented a new color feature descriptor and an automated computer system for book spine image matching. CDSIFT utilizes a color Difference-of-Gaussians feature detector to find color edges in an intensity-independent color space. This choice of color space provides similar shadow, highlight, and specularity invariance as photometric invariants and photometric quasi-invariants without the excessive computation required to compute these values. CDSIFT also does not require additional intensity or color space normalization, which makes photometric invariants become unstable and fail in low-color or low-light situations.

(a) Color image



(b) Correct matches



(c) Recall

Figure 14: Performance comparison on Shelf 9.

The automated system was built using standard off-the-shelf components and obtained accurate matching results using 3-megapixel camera images from a hand-held cellular phone without the assistance of lighting kits or stabilizers.

The CDSIFT algorithm used in our system was compared against c-color-SIFT, the current best-performing color feature descriptor for the specific task of book spine identification. CDSIFT provides greater accuracy in our application than the leading photometric invariant color descriptors or the SIFT grayscale descriptor. CDSIFT outperformed c-color-SIFT in every book spine image dataset.

CDSIFT is a parallelizable algorithm that is best suited for a multicore platform such as today's multicore desktops or GPU-based systems. It is especially an efficient algorithm for FPGA implementation in embedded applications such as robotics and automation. The Robotic Vision Lab at the Brigham Young University has developed a low-power, light-weight FPGA board called Helios for embedded vision sensor applications. Our future work will be to implement the CDSIFT algorithm in FPGA fabric on Helios. This small embedded system will then be used as a portable book inventory system and also used to provide real-time, color-aware feature detection for drift stabilization, object identification, pose awareness, localization, and mapping on the quad-rotor helicopter platform being developed in the lab.

References

[1] D. J. Lee, Y. Chang, J. K. Archibald, and C. Pitzak, "Matching book-spine images for library shelf-reading process automation," in *Proceedings of the 4th IEEE Conference on Automation Science and Engineering (CASE '08)*, pp. 738–743, August 2008.

[2] F. Pernkopf, "Detection of surface defects on raw steel blocks using Bayesian network classifiers," *Pattern Analysis and Applications*, vol. 7, no. 3, pp. 333–342, 2004.

[3] T. Möenpöö, J. Viertola, and M. Pietiköinen, "Optimising colour and texture features for real-time visual inspection," *Pattern Analysis and Applications*, vol. 6, no. 3, pp. 169–175, 2003.

[4] S. M. Bhandarkar, X. Luo, R. F. Daniels, and E. W. Tollner, "Automated planning and optimization of lumber production

using machine vision and computed tomography," *IEEE Transactions on Automation Science and Engineering*, vol. 5, no. 4, pp. 677–695, 2008.

[5] P. J. Elbischger, H. Bischof, P. Regitnig, and G. A. Holzapfel, "Automatic analysis of collagen fiber orientation in the outermost layer of human arteries," *Pattern Analysis and Applications*, vol. 7, no. 3, pp. 269–284, 2004.

[6] Y. Cheng and M. A. Jafari, "Vision-based online process control in manufacturing applications," *IEEE Transactions on Automation Science and Engineering*, vol. 5, no. 1, pp. 140–153, 2008.

[7] P. Soda, G. Iannello, and M. Vento, "A multiple expert system for classifying fluorescent intensity in antinuclear autoantibodies analysis," *Pattern Analysis and Applications*, vol. 12, no. 3, pp. 215–226, 2009.

[8] G. A. Khuwaja and A. N. Abu-Rezq, "Bi-modal breast cancer classification system," *Pattern Analysis and Applications*, vol. 7, no. 3, pp. 235–242, 2004.

[9] M. J. Swain and D. H. Ballard, "Color indexing," *International Journal of Computer Vision*, vol. 7, no. 1, pp. 11–32, 1991.

[10] A. Mojsilović and E. Soljanin, "Color quantization and processing by fibonacci lattices," *IEEE Transactions on Image Processing*, vol. 10, no. 11, pp. 1712–1725, 2001.

[11] G. J. Burghouts and J. M. Geusebroek, "Performance evaluation of local colour invariants," *Computer Vision and Image Understanding*, vol. 113, no. 1, pp. 48–62, 2009.

[12] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek, "Evaluating color descriptors for object and scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1582–1596, 2010.

[13] S. G. Fowers, D. J. Lee, and D. K. Wilde, "Color DoG: a three-channel color feature detector for embedded systems," in *Proceedings of the 27th Intelligent Robots and Computer Vision: Algorithms and Techniques*, vol. 7539, Jan 2010.

[14] S. G. Fowers, D.-J. Lee, and G. Xiong, "Improved library shelf reading using color feature matching of book-spine images," in *Proceedings of the International Conference on Control, Automation, Robotics and Vision*, December 2010.

[15] D. Slater and G. Healey, "The illumination-invariant recognition of 3D objects using local color invariants," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 2, pp. 206–210, 1996.

[16] G. D. Finlayson, "Color in perspective," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 10, pp. 1034–1038, 1996.

[17] T. Gevers and A. W. M. Smeulders, "Color based object recognition," *Lecture Notes in Computer Science*, vol. 1310, pp. 319–326, 1997.

[18] I. Biederman, "Recognition-by-Components: a theory of human image understanding," *Psychological Review*, vol. 94, no. 2, pp. 115–147, 1987.

[19] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the Alvey Vision Conference*, vol. 15, 1988.

[20] P. Beaudet, "Rotationally invariant image operators," in *Proceedings of the International Conference of Pattern Recognition*, pp. 579–583, 1978.

[21] K. Paler, J. Föglein, J. Illingworth, and J. Kittler, "Local ordered grey levels as an aid to corner detection," *Pattern Recognition*, vol. 17, no. 5, pp. 535–543, 1984.

[22] O. A. Zuniga and R. Haralick, "Corner detection using the facet model," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 30–37, 1983.

[23] K. Mikolajczyk and C. Schmid, "Scale & affine invariant interest point detectors," *International Journal of Computer Vision*, vol. 60, no. 1, pp. 63–86, 2004.

[24] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, 2005.

[25] F. Godtliebsen, J. S. Marron, and P. Chaudhuri, "Statistical significance of features in digital images," *Image and Vision Computing*, vol. 22, no. 13, pp. 1093–1104, 2004.

[26] R. Deriche and G. Giraudon, "A computational approach for corner and vertex detection," *International Journal of Computer Vision*, vol. 10, no. 2, pp. 101–124, 1993.

[27] L. Kitchen and A. Rosenfeld, "Gray-level corner detection," *Pattern Recognition Letters*, vol. 1, no. 2, pp. 95–102, 1993.

[28] K. Rangarajan, M. Shah, and D. van Brackle, "Optimal corner detector," *Computer Vision Graph Image Processing*, vol. 48, no. 2, pp. 230–245, 1989.

[29] H. Wang and M. Brady, "A practical solution to corner detection," in *Proceedings of the IEEE International Conference on Image Processing*, vol. 1, pp. 919–923, November1994.

[30] P. L. Rosin, "Augmenting corner descriptors," *Graphical Models and Image Processing*, vol. 58, no. 3, pp. 286–294, 1996.

[31] J. van de Weijer and C. Schmid, "Coloring local feature extraction," in *Proceedings of the Computer Vision European Conference on Computer Vision (ECCV '06)*, pp. 334–348, 2006.

[32] A. Bosch, A. Zisserman, and X. Muñoz, "Scene classification using a hybrid generative/discriminative approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 4, pp. 712–727, 2008.

[33] R. Datta, D. Joshi, J. Li, and J. Z. Wang, "Image retrieval: ideas, influences, and trends of the new age," *ACM Computing Surveys*, vol. 40, no. 2, pp. 1–60, 2008.

[34] R. O. Stehling, M. A. Nascimento, and A. X. Falcão, "A compact and efficient image retrieval approach based on border/interior pixel classification," in *Proceedings of the 11th International Conference on Information and Knowledge Management (CIKM '02)*, pp. 102–109, November 2002.

[35] I. Laptev, "Improving object detection with boosted histograms," *Image and Vision Computing*, vol. 27, no. 5, pp. 535–544, 2009.

[36] J. Almeida, A. Rocha, R. Torres, and S. Goldenstein, "Making colors worth more than a thousand words," in *Proceedings of the 23rd Annual ACM Symposium on Applied Computing (SAC '08)*, pp. 1180–1186, November 2008.

[37] J. M. Geusebroek, R. van den Boomgaard, A. W. M. Smeulders, and H. Geerts, "Color invariance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 12, pp. 1338–1350, 2001.

[38] A. Abdel-Hakim and A. Farag, "CSIFT: a SIFT descriptor with color invariant characteristics," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '06)*, vol. 2, pp. 1978–1983, October 2006.

[39] J. van de Weijer, T. Gevers, and J. Geusebroek, "Color edge detection by photometric quasi-invariants," in *Proceedings of the 9th IEEE International Conference On Computer Vision*, pp. 1520–1525, October 2003.

[40] J. van de Weijer, T. Gevers, and J.-M. Geusebroek, "Edge and corner detection by photometric quasi-invariants," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 4, pp. 625–630, 2005.

[41] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[42] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[43] Y. T. Tsai, Q. Wang, and S. You, "CDIKP: a highly-compact local feature descriptor," in *Proceedings of the 19th International Conference on Pattern Recognition (ICPR '08)*, p. 1, December 2008.