Building Book Inventories using Smartphones

David Chen[1], Sam Tsai[1], Cheng-Hsin Hsu[2], Kyu-Han Kim[2], Jatinder Pal Singh[2], Bernd Girod[1]

[1]Department of Electrical Engineering, Stanford University, Stanford, CA 94305, USA
{dmchen, sstsai, bgirod}@stanford.edu

[2]Deutsche Telekom R&D Laboratories USA, 5050 El Camino Real 221, Los Altos, CA 94022, USA
{cheng-hsin.hsu, kyu-han.kim, j.singh}@telekom.com

ABSTRACT

Manual generation of a book inventory is time-consuming and tedious, while deployment of barcode and radio-frequency identification (RFID) management systems is costly and affordable only to large institutions. In this paper, we design and implement a mobile book recognition system for conveniently generating an inventory of books by snapping photos of a bookshelf with a smartphone. Since smartphones are becoming ubiquitous and affordable, our inventory management solution is cost-effective and very easy to deploy. Automatic and robust book recognition is achieved in our system using a combination of spine segmentation and bag-of-features image matching. At the same time, the location of each book is inferred from the smartphone's sensor readings, including accelerometer traces, digital compass measurements, and WiFi signatures. This location information is combined with the image recognition results to construct a location-aware book inventory. We demonstrate the effectiveness of our book spine recognition and location estimation techniques in recognition experiments and in an actual mobile book recognition system.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval —*Search Process*

General Terms

Design, Performance

1. INTRODUCTION

Maintaining an up-to-date inventory of a large set of books is a difficult task for both individuals and institutions. How can a person automatically generate an inventory of all the books he/she has in the home or office, without going through the tedious process of typing in the title, authors, publisher, and edition of each book into a computer? Similarly, except on a much larger scale, how can a library generate an inventory of all the books currently on the bookshelves in an efficient manner? Manual creation of the inventory is very costly in terms of human labor and error-prone. Alternatively, a barcode or radio-frequency identification (RFID) book management system can be deployed, but this approach is very expensive and requires physically attaching a marker to each book. A more cost-effective and convenient method for building and updating book inventories is desired.

Modern smartphones are becoming ubiquitous, and these phones are equipped with high-resolution cameras, various location sensors, and access to high-speed networks like WiFi. Our proposed system utilizes a smartphone to capture (i) images of bookshelves with the phone's camera and (ii) corresponding location readings, like accelerometer traces, digital compass measurements, and WiFi signatures, with the phone's location sensors. The images and location data are transmitted to a server over a high-speed network with very low transmission delay. On the server, first, a *book recognition* algorithm is used to reliably segment and identify each book in a photo. Recognizing book spines in a photo of a bookshelf is very challenging because each spine has a small surface area containing few image features and the other spines' image features act as clutter in feature-based image matching. Second, a *location tracking* algorithm infers each book's location, at the time each photo is taken, from the gathered location data. Despite the availability of the smartphone's sensor measurements, determining the precise location of each book is still challenging because these measurements are often noisy.

Our main contributions in this paper are summarized as follows:

- We design and implement a cost-effective book inventory management system using smartphones. To the best of our knowledge, building scalable book inventories using smartphones with visual search and location inference has never been considered before in the literature.
- We demonstrate that feature-based book spine recognition is more robust against photometric and geometric distortions in photos than traditional techniques like optical character recognition (OCR).
- We show that book spine segmentation is critical for accurate feature-based image matching, as the segmentation greatly reduces clutter from the perspective of any single spine.
- We develop a location tracker that utilizes WiFi fingerprints and other location measurements collected by the smartphone to determine where each photo is taken.

2. RELATED WORK

Recognizing books on bookshelves has been previously considered in [6, 9, 13]. Lee et al. [9] propose to apply color quantization on each book spine and use color indices for book spine image matching. They assume the books are perfectly aligned with the vertical axis and each image is taken right in front of a bookshelf.

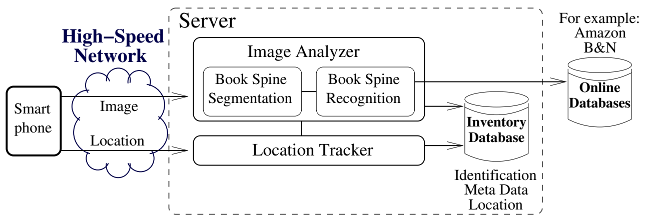

Figure 1: Book inventory management system.

Quoc and Choi [13] propose a system for book recognition, which consists of three components: book region extraction, book segmentation, and title extraction. For title extraction, they use OCR to read the book title. OCR may not be robust, however, against photometric and geometric distortions often encountered in photos taken by smartphones. Crasto et al. [6] develop a projector-camera system to replay a movie of the recent bookshelf appearance, highlighting removals and additions of individual books. Their system uses a fixed camera arrangement to monitor a single bookshelf and would require recalibration if moved to a different location.

Our system differs from previous work [6, 9, 13] in several important aspects. First, we employ scale- and rotation-invariant local features for robust book recognition. Second, our system leverages smartphones and requires no system calibration. Third, our system seamlessly combines image-based book recognition with location tracking to build location-aware inventory databases.

3. AUTOMATED BOOK MANAGEMENT

Fig. 1 illustrates the architecture of our system. The smartphone runs a light-weight mobile agent that captures images and location data and sends the information over a high-speed network to a server. On the server, an image analyzer and a location tracker run in parallel. The image analyzer performs book spine recognition by first segmenting each spine in the query photo and then matching the segmented spine against an online database of book spines. Meta-data like titles and authors in the online database can also be retrieved and included into the book inventory. The location tracker analyzes the raw readings from various phone sensors to determine the locations of individual books, specifying which room, which bookshelf, and where in the bookshelf each recognized book can be found.

3.1 Image Analyzer

Fig. 2 shows two typical query images of bookshelves taken with a smartphone. Recognition of spines in these photos is more challenging than recognition of the front covers of books and CD/DVD cases [5, 15]. First, a book's spine has much smaller surface area than its cover, yielding fewer visual features that can be utilized for robust recognition. Second, each query photo of a bookshelf contains many book spines and, from the perspective of a single spine, all the other spines' visual features represent background clutter. Trying to directly match a photo containing many spines against a database of individual spines would result in low recognition accuracy. To solve these problems and achieve high recognition performance, our image analyzer processes each query photo in two steps: spine segmentation and spine recognition.

3.1.1 Book Spine Segmentation

To segment out individual book spines in the query image, we exploit the fact that often books are densely packed together on a

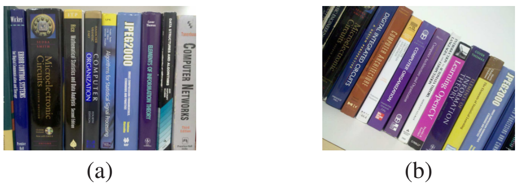

Figure 2: Query image showing book spines in (a) Nearly vertical and (b) Slanted orientation.

bookshelf and have roughly the same orientation. Fig. 3 shows a block diagram of the operations performed for segmentation. First, a Canny edge map is extracted from the query image [4]. In the Canny edge map, the crevices between book spines appear as long and straight edges, while the text and graphics on the spines appear as short and curved edges. Connected components are detected in the edge map, and all components containing fewer pixels than a lower threshold are removed.

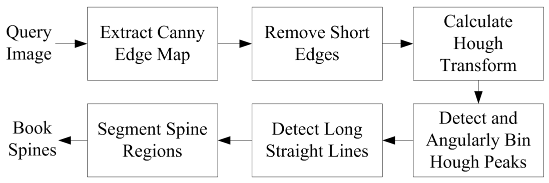
Query Image
Extract Canny Edge Map
Remove Short Edges
Calculate Hough Transform
Detect and Angularly Bin Hough Peaks
Detect Long Straight Lines
Segment Spine Regions
Book Spines

Figure 3: Operations for book spine segmentation.

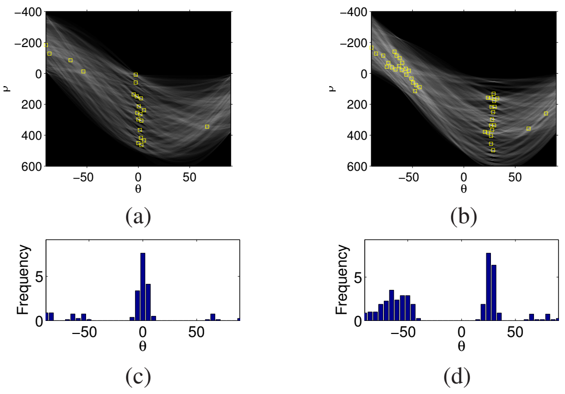

Figure 4: Hough transform for bookshelf image with (a) Nearly vertical and (b) Slanted orientation. Yellow squares denote peaks in the Hough transform. Angular histogram for Hough peaks with (c) Nearly vertical and (d) Slanted orientation.

Next, a Hough transform is calculated from the edge map with short edges removed. It is found that removal of short edges improves the resolution of peaks in the Hough transform. Fig. 4(a)-(b) displays the Hough transform calculated for the two query images from Fig. 2. In these plots, yellow squares denote the locations in the (ρ, θ) space where peaks in the Hough transform occur. A large group of peaks around a common angle θ corresponds to the nearly parallel lines between book spines. To estimate the dominant angle θ_{spines} of the spines, the Hough peak occurrences are binned in an angular histogram as shown in Fig. 4(c-d). The bin attaining the highest count in the angular histogram yields the spines' dominant angle θ_{spines}.

Given the dominant angle θ_{spines} of the book spines, we search for long straight lines with angle near θ_{spines}. Long straight lines can be detected from the Canny edge map using the method described in [8]. Each such line with angle in $(\theta_{\text{spines}} - \Delta\theta, \theta_{\text{spines}} + \Delta\theta)$ is considered as the boundary between two books. In our experiments, $\Delta\theta = 15°$ was found to be a good parameter. Fig. 5 shows that our methods successfully find line boundaries between books. Regions between the line boundaries are segmented out as the individual book spines.

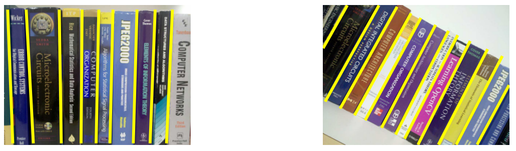
(a) (b)

Figure 5: Segmentation boundaries for bookshelf image with (a) Nearly vertical and (b) Slanted orientation.

3.1.2 Book Spine Recognition

Each of the segmented spines is individually matched against a database of book spines. Fig. 6 illustrates the steps used in the recognition process. First, a set of scale- and rotation-invariant features, or a bag-of-visual-features (BoVF), are extracted from each query spine. Since achieving low query latency is very important for interactive book searches, we employ SURF features [3], which are much faster to compute than and offer comparable performance to SIFT features [10]. Recognition with BoVF is also resilient against imperfect segmentation.

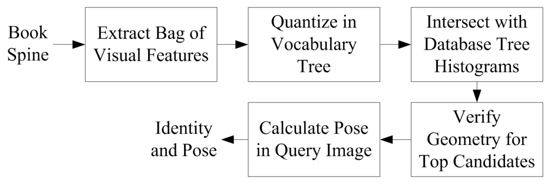
Book Spine
Extract Bag of Visual Features
Quantize in Vocabulary Tree
Intersect with Database Tree Histograms
Verify Geometry for Top Candidates
Calculate Pose in Query Image
Identity and Pose

Figure 6: Operations for book spine recognition.

For fast search through a large database of book spines, each query spine's BoVF is quantized through a vocabulary tree [11]. Soft binning is used to mitigate quantization errors [12]. The quantized BoVF of the query spine forms a tree histogram, counting how often each node in the vocabulary tree has been visited. A similarity score between the query tree histogram and each database tree histogram is computed by histogram intersection, which can be performed efficiently using an inverted index [11]. Thereafter, a shortlist of the 50 top-ranked database spines are further subjected to rigorous pairwise geometric verification using the ratio test [10] and affine model RANSAC [7] to find spatial consistency between the feature matches. The best matching database spine is then selected. During geometric verification, feature matches enable us to generate a transformation model from the query spine's coordinate frame to the matching database spine's coordinate frame. If we now map the corners of the database spine into the query spine's coordinate frame using the transformation model, we can accurately calculate the pose of each spine in the query image.

3.2 Location Tracker

Localization systems based on Global Positioning System (GPS) coordinates and cell-tower identification [16] are accurate outdoors. Indoors, WiFi-based localization [2] has shown more promising results. Two drawbacks of WiFi-based localization are the extensive training required beforehand and the sensitivity to individual WiFi access points being switched on/off. Furthermore, WiFi-based localization only provides 2D location coordinates excluding height, whereas for accurate book localization we require 3D coordinates. Alternatively, Azizyan et al. [1] propose to use multiple ambient sound and light sources to infer the user's current location, and this system works well in shopping malls where sound and light can be highly discriminative. In a library or bookstore, however, sound and light are unlikely to help us discern the current location of a user and his/her smartphone.

In contrast to the prior approaches, our location tracker utilizes both WiFi signatures and the smartphone's digital compass and accelerometer readings. First, we sample the signal strengths of nearby WiFi access points at each location of interest, so that later on, when we encounter a similar combination of signal strengths from the access points, we can infer that we are in roughly the same location. WiFi-based localization provides a coarse-grain estimate of the current location. Then, more specific localization can be achieved using the digital compass and accelerometer on the smartphone. The digital compass tells us which direction the phone is facing when a book is photographed, while a temporal trace of the accelerometer informs us how far vertically and horizontally the phone has moved from the last anchor point. Digital compass and accelerometer readings enable us to refine our location estimate from the coarse WiFi-based estimate.

4. EXPERIMENTAL RESULTS

We have implemented a mobile book recognition system, employing the techniques discussed in this paper. A demo video is available online.[1] The demo shows how a user's inventory database can be gradually constructed. Each time the user snaps a photo of part of a bookshelf, our system automatically recognizes and localizes each spine in the photo, and the recognized books are added to the inventory database. When the recognition result is sent to the phone, the user can also select each spine that appears in the query photo. A selected spine has its boundary highlighted and its title displayed in the phone's viewfinder, enabling easy visualization of the new books added to the inventory. Additionally, we have prototyped the location tracker on a smartphone. Our system uses sensor readings from the WiFi sensor, accelerometer, and digital compass to determine the location where each photo is taken, and the location is appended to the spine identification in the inventory.

To test recognition accuracy, we constructed a database of 2148 book spine images, simulating an online database of books covering, say, a particular subject. The book spine images are taken at the Stanford Engineering Library. SURF features are extracted from each database spine image. The set of all database features is used to train a vocabulary tree with 6 levels and 10^6 leaf nodes. For query images, we took 50 photos of bookshelves with a camera phone, showing books in different orientations and illuminations.[2]

We compare two different schemes: *SURF* and *SegSURF*. The *SURF* scheme extracts SURF features from the entire query image and quantizes these features in the vocabulary tree. Following vocabulary tree scoring, a shortlist of the 50 top-ranked database spines are verified geometrically. This scheme suffers from the fact that from the perspective of one spine, the features for all the other spines in the query image act as clutter and reduce the ratio of inlier to outlier features. The recognition accuracy, defined as the

[1] http://msw3.stanford.edu/~dchen/CIBR/Book-Spines.wmv
[2] http://msw3.stanford.edu/~dchen/CIBR/BookSpineQuery

Table 1: Comparison of two recognition schemes.

Scheme	SURF	SegSURF
Total Spines	407	407
Recognized Spines	32	303
Accuracy	7.9 %	74.5 %
Processing Time	2.72 sec	1.76 sec

number of correctly recognized spines divided by the total number of spines in the query images, for the *SURF* scheme is shown in Table 1. Out of 407 spines in the 50 query images, only 32 are recognized, giving a very low accuracy of 7.9 percent.

In contrast, the *SegSURF* scheme is the process we described in Sec. 3.1. Each spine in the query image is segmented out before feature extraction and quantization in the vocabulary tree. Since the amount of clutter is significantly reduced from the perspective of any single spine, this scheme dramatically boosts the ratio of inlier to outlier features and therefore greatly improves recognition accuracy. Out of 407 total spines, *SegSURF* correctly recognizes 303 spines, yielding an accuracy of 74.5 percent. Our experimental results show that segmentation is a crucial step required for accurate spine recognition.

Another advantage of the *SegSURF* scheme is that it reduces the server processing time to 1.76 sec per query image, compared to the *SURF* scheme which takes 2.72 sec, where the timing was measured on a Linux 2.0 GHz server. Server processing time is the main part of the overall system latency, as data transmission over WiFi only takes about 100 msec. The savings in processing time are due to the extraction of fewer total features after segmentation and hence fewer features being subjected to time-consuming geometric verifications. Following segmentation, SURF features are detected only in the interior of the segmented spines. Without segmentation, SURF features would also be detected along the spine edges, in the crevices between books, and in the surrounding bookshelf frames.

For comparison, we performed a preliminary experiment where we tried to recognize the segmented spines by OCR. We employed the open-source Tesseract OCR engine [14]. Because the spines often contain non-text graphics, uneven illumination, and varying fonts, the OCR engine almost always gave unrecognizable text outputs. Fig. 7 shows some segmented spines and their OCR readings from Tesseract. It can be seen that the OCR readings are very noisy, and even post-processing operations like spelling correction or nearest Levenshtein distance matching against a dictionary are unlikely to yield the correct results. A more thorough evaluation of OCR-based spine recognition is planned for our future work.

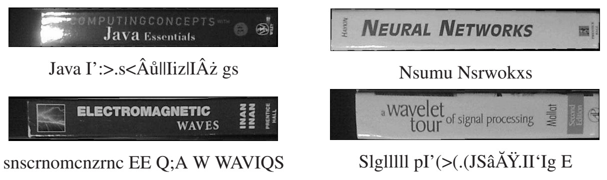
COMPUTINGCONCEPTS WITH Java Essentials
Java I’:>.s<Âů||Iiz|IÂż gs
NEURAL NETWORKS
Nsumu Nsrwokxs
ELECTROMAGNETIC WAVES
snscrnomcnzrnc EE Q;A W WAVIQS
a wavelet tour of signal processing
Slgllll pI’(>(.(JSâĂŸ.II‘Ig E

Figure 7: Segmented spines and their OCR readings.

5. CONCLUSIONS

We have created a mobile book recognition system, in which a user takes a photo of a bookshelf with a camera phone and the books in the photo are automatically identified. Leveraging camera phones which are affordable and ubiquitous enables us to build a cost-effective book management system. After a photo of a bookshelf is transmitted from the phone to a server, individual book spines are segmented out by analyzing the line structures, and each segmented spine is represented as a bag of rotation- and scale-invariant features. The query features are matched against a vocabulary tree-structured database of spines. Recognized spines are also tagged with location information provided by the phone. After spine recognition, the user can select individual spines in the phone’s viewfinder to obtain more information about a particular book, without ever having to take that book off the bookshelf.

6. REFERENCES

[1] M. Azizyan, I. Constandache, and R. Choudhury. SurroundSense: Mobile phone localization via ambience fingerprinting. In *Proc. ACM International Conference on Mobile Computing and Networking (MobiCom’09)*, pages 261–272, Beijing, China, September 2009.

[2] P. Bahl and V. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *Proc. IEEE International Conference on Computer Communications (INFOCOM’00)*, pages 775–784, Tel Aviv, Israel, March 2000.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[4] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[5] D. Chen, S. Tsai, V. Chandrasekhar, G. Takacs, J. Singh, and B. Girod. Robust image retrieval using multiview scalable vocabulary trees. In *Proc. Visual Communications and Image Processing (VCIP’09)*, page 72570V, San Jose, CA, January 2009.

[6] D. Crasto, A. Kale, and C. Jaynes. The smart bookshelf: A study of camera projector scene augmentation of an everyday environment. In *Proc. IEEE Workshop on Applications of Computer Vision (WACV’05)*, pages 218–225, Breckenridge, CO, January 2005.

[7] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[8] J. Kosecka and W. Zhang. Video compass. In *Proc. European Conference on Computer Vision (ECCV’02)*, pages 476–490, Copenhagen, Denmark, May 2002.

[9] D. Lee, Y. Chang, J. Archibald, and C. Pitzak. Matching book-spine images for library shelf-reading process automation. In *Proc. IEEE International Conference on Automation Science and Engineering (CASE’08)*, pages 738–743, Arlington, VA, September 2008.

[10] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[11] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR’06)*, pages 2161–2168, New York, NY, June 2006.

[12] J. Philbin, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR’08)*, pages 1–8, Anchorage, AL, June 2008.

[13] N. Quoc and W. Choi. A framework for recognition books on bookshelves. In *Proc. International Conference on Intelligent Computing (ICIC’09)*, pages 386–395, Ulsan, Korea, September 2009.

[14] R. Smith. An overview of the Tesseract OCR engine. In *Proc. Conference on Document Analysis and Recognition (ICDAR’07)*, pages 629–633, Parana, Brazil, September 2007.

[15] S. Tsai, D. Chen, J. Singh, and B. Girod. Rate-efficient, real-time CD cover recognition on a camera-phone. In *Proc. ACM Multimedia (MM’08’)*, pages 1023–1024, Vancouver, BC, Canada, October 2008. Technical Demo.

[16] Y. Zhao. Standardization of mobile phone positioning for 3G systems. *IEEE Communications Magazine*, 40(7):108–116, 2002.