Hidden Markov Model for Event Photo Stream Segmentation

Jesse Prabawa Gozali, Min-Yen Kan
Department of Computer Science
National University of Singapore, Singapore
Email: {jprabawa, kanmy}@comp.nus.edu.sg

Hari Sundaram
Arts Media & Engineering
Arizona State University, USA
Email: hari.sundaram@asu.edu

***Abstract*—A photo stream is a chronological sequence of photos. Most existing photo stream segmentation methods assume that a photo stream comprises of photos from multiple events and their goal is to produce groups of photos, *each corresponding to an event*, *i.e.* they perform automatic albuming. Even if these photos are grouped by event, sifting through the abundance of photos in each event is cumbersome. To help make photos of each event more manageable, we propose a photo stream segmentation method for an event photo stream — the chronological sequence of photos of a single event — to produce groups of photos, *each corresponding to a photo-worthy moment in the event*.**

Our method is based on a hidden Markov model with parameters learned from time, EXIF metadata, and visual information from 1) training data of unlabelled, unsegmented event photo streams and 2) the event photo stream we want to segment. In an experiment with over 5000 photos from 28 personal photo sets, our method outperformed all six baselines with statistical significance ($p < 0.10$ with the best baseline and $p < 0.005$ with the others).

***Keywords*-Event photo stream segmentation; hidden Markov model; digital photo library**

I. Introduction

The advent of inexpensive, easy-to-use and portable photo capture devices with large memory stores have changed people's photo taking habits – people now are more liberal with their photo taking, as compared to the previous era of film rolls and analog cameras [1]. Most personal photos are commonly associated with an event: a holiday trip, picnic, dinner or walk in the park. Many academic and commercial photo browsers, like iPhoto and Picasa, advocate event-based photo organization. Even so, sifting through the hundreds of photos associated with an event is still cumbersome.

To complement event-based photo organization and help make photos of each event more manageable, we propose a method to segment an **event photo stream** — the chronological sequence of photos of *a single event* — to produce groups of photos, each of which corresponds to a photo-worthy moment in the event. For example, Figure 1 shows how an event photo stream segmentation can reveal different photo-worthy moments in the event.

Event photo stream segmentation *is the process of finding contiguous groups of photos from an event photo stream, each corresponding to a photo-worthy moment in the event.*

We distinguish between an event photo stream and a photo stream, which is a more general term that refers to a chronological sequence of photos that may consist of many days or even months of photos. Many segmentation methods have been proposed for such photo streams to produce groups of photos where each group corresponds to an event. To distinguish between their task and ours, we shall refer to their task as **automatic albuming**.

While both tasks segment photo streams, automatic albuming methods may not be suitable for event photo stream segmentation due to issues of data sparsity, indistinct time gaps, and visual similarities:

Data sparsity – Each group of photos produced through event photo stream segmentation has only a handful of photos as each corresponds to a photo-worthy moment in the event. In contrast, each group produced through automatic albuming corresponds to an event and has many more photos. A photo stream of multiple events also has many more photos than an event photo stream, which is of just one event. The increased sparsity associated with event photo stream segmentation makes it harder to develop computational models.

Indistinct time gaps – In a photo stream, **time gap** is the time difference between the capture times of two consecutive photos. While the time gap between two photos of different events is in hours or even days, the time gap between photos of the same event is typically in seconds or minutes. This time scale difference is useful to identify event boundaries for automatic albuming. In contrast for event photo stream segmentation, the time gap between two consecutive photos belonging to different photo-worthy moments in the event is also in seconds or minutes. Indistinct time gaps at segment boundaries in an event photo stream makes the segment boundaries difficult to identify using simple heuristics.

Visual similarities – Photos in an event are often visually similar because they share aspects such as participants, location, and scene. With photos of other events, however, they are often visually distinct because these aspects are different. The visual difference between photos of different

We would like to acknowledge the support of the NExT Research Center, funded by MDA, Singapore, under the research grant WBS: R-252-300-001-490.

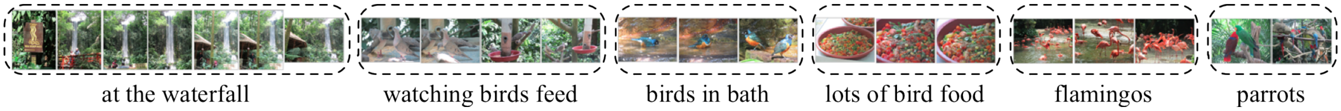

Figure 1. A part of an event photo stream shown with segmentation and semantic labels.

events is useful for automatic albuming, but the visual similarities among photos of an event make event photo stream segmentation more difficult.

To address these challenges, we propose a hidden Markov model (HMM) -based approach that uses a combination of time, EXIF[1] metadata, and visual information to determine the segment boundaries in an event photo stream. Parameters of the HMM are learned from 1) a set of unlabelled, unsegmented event photo streams and 2) the event photo stream we want to segment. Our model supposes that an event photo stream is the result of a stochastic process that generates feature vectors from a set of foreground and background models. The foreground models generate feature vectors corresponding to segment boundaries while the background models generate feature vectors that do not. This generative model follows from our observation that photos taken in events are often the result of several **photo taking sessions** — each session corresponds to a photo-worthy moment. At such a moment, we take several photos. Then, our camera idles until the next moment arises and invites us for another photo taking session. In each session, photos would likely be similar in terms of visual appearance, photo metadata and timing. The photographer, for example, could choose to adjust the focal length and aperture settings to suit the scene of the moment. These camera parameter values would be similar for photos within the same session. If we look at photo timestamps, each session would appear to be a burst of photo activity [2].

II. Related Work

To our knowledge, the closest work to ours is by Graham *et al.* [2]. They posit that people tend to take photos in bursts and these bursts can be identified by looking at time gaps that are statistical outliers and not part of any burst. Their event photo stream segmentation method finds segments corresponding to bursts of photo taking activity.

Other photo stream segmentation methods were devised for automatic albuming. Most of these methods rely on time information. The simplest method to find segment boundaries is to check for time gaps that are greater than a fixed threshold (*e.g.* average time gap). Loui *et al.* [3] used a time scaling function and K-means clustering with K=2 to determine this fixed threshold. Platt *et al.* [4] proposed a method where the threshold becomes adaptive, computed over a sliding window. Some methods are similarly adaptive, although based on keen observations instead of thresholding; Zhao *et al.* [5] observed that the probability of an event ending increases as more photos are taken and as the time span increases; Gargi [6] observed that a long interval with no photo taking usually marks the end of an event and that a sharp upward change in the frequency of capture usually marks the start of a new event. Pigeau and Gelgon [7] proposed a model-based incremental unsupervised classification where distinct classifications are built from both temporal and location information.

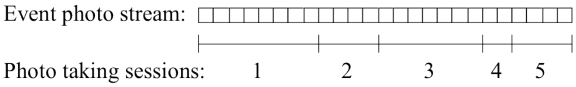

Figure 2. An event photo stream with 26 photos, captured over 5 photo taking sessions.

Few methods have utilized EXIF metadata. Gong and Jain [8] proposed a segmentation method based on changes in scene brightness. Mei *et al.* [9] proposed a clustering approach using EXIF metadata like aperture diameter, exposure time, and focal length. Their method also used time, location and visual features such as color histogram, and Tamura descriptor (texture). There are only few others that have utilized visual information. Platt *et al.* [4] proposed a best-first model merging method based on color histograms. Cooper *et al.* [10] proposed an approach based on scale-space analysis of both color and time information.

Most automatic albuming methods utilize time gap information. Because the time gaps at event boundaries are typically much larger than the time gaps between photos in an event, these methods work effectively to segment a photo stream by event. For event photo stream segmentation however, the segment boundaries may not be distinguishable from time information alone and other information based on EXIF metadata and visual information should be utilized. This is the path we took for our approach.

III. Event Photo Stream Segmentation

When we view photos from an event (*e.g.* in Figure 1), we often make inferences about how each photo relates to its surrounding photos and how different groups of photos in the stream fit together to capture different moments in the event. Without semantic knowledge of the event, we make such inferences based on the visual appearance and timestamp of the photos. Our algorithm embodies this inference process.

Given an event photo stream, we want to find groups of photos in the stream such that each group corresponds to a photo taking session. The groups should also form a partition

[1] JEITA Exchangeable image file format for digital still cameras

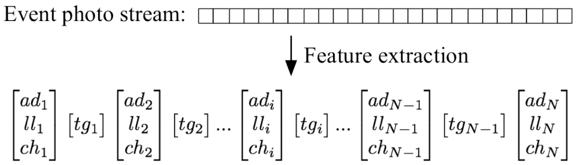
Event photo stream:
Feature extraction
ad_1 ll_1 ch_1 $[tg_1]$ ad_2 ll_2 ch_2 $[tg_2]$ ... ad_i ll_i ch_i $[tg_i]$ ... ad_{N-1} ll_{N-1} ch_{N-1} $[tg_{N-1}]$ ad_N ll_N ch_N

Figure 3. Extracted photo and photo gap features from the event photo stream form a sequence of alternating feature types. Symbols ad_i, ll_i, and ch_i denote aperture diameter, LogLight, and color histogram feature values of photo i; tg_i denotes the time gap between photos i and $i+1$.

over all the event photos (see Figure 2). By photo taking session, we mean a period of time devoted to photo taking, producing photos with similarities in visual appearance, EXIF metadata, and timing. We assume that photo taking sessions are correlated with photo-worthy moments in the event.

A. Features

Similar to the manual inference process we described above, our algorithm also relies on visual and time information from the photos. Additionally, it also uses information from camera parameters, *i.e.* how the photos were captured using the camera, as encoded in the photos' EXIF metadata. We extract features from the event photo stream and distinguish between two feature types: 1) **photo feature**, *i.e.* feature about the photo, and 2) **photo gap feature**, *i.e.* feature about the gap between consecutive photos.

We experimented with various photo features, *e.g.* focal length, gradient direction autocorrelogram and features based on SIFT. We also experimented with various photo gap features by taking the difference between consecutive photo feature values. The best feature combination we found for our approach consists of simple features that work best under our task constraint of data sparsity: 1) *Aperture Diameter* – a photo feature measuring the size of the opening through which light enters the camera, 2) *LogLight* [11] – a photo feature measuring the ambient light in an image, 3) *Color Histogram* – a photo feature measuring the color distribution in an image, and 4) *Time gap* – a photo gap feature measuring the time difference between capture times of consecutive photos.

Most photo stream segmentation methods rely on just time gaps. Some incorporate visual features and very few use features derived from EXIF metadata. In our work, we use all these features in a generative model. Additionally, our observation that these features belong to two types — photo feature and photo gap feature — is novel and forms the basis of how we formally define the problem and our model structure.

B. Problem Definition

With the features we extract from the event photo stream, we end up with a sequence of vectors with alternating types (see Figure 3). From an event photo stream of N photos, we get a sequence of $2N-1$ vectors, of which $N-1$ are photo gap features whose locations correspond to *potential segment boundaries* in the photo stream segmentation.

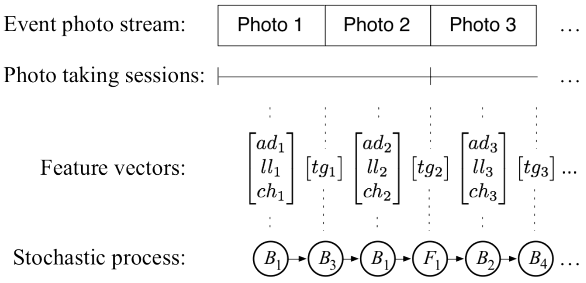
Event photo stream:
Photo 1
Photo 2
Photo 3
...
Photo taking sessions:
...
Feature vectors:
ad_1 ll_1 ch_1 $[tg_1]$ ad_2 ll_2 ch_2 $[tg_2]$ ad_3 ll_3 ch_3 $[tg_3]$ ...
Stochastic process:
B_1 B_3 B_1 F_1 B_2 B_4 ...

Figure 4. Our model views an event photo stream as the result of a stochastic process consisting of a set of foreground and background models. In the above, the first photo taking session consists of two photos. The time gap, tg_2, corresponding to the segment boundary between photo 2 and photo 3, is generated by the foreground model, F_1, of the stochastic process. The remaining models shown are the background models, B_i.

We define an event photo stream segmentation X as a sequence of Boolean variables $\langle X_1, X_2, ..., X_{N-1}\rangle$ corresponding to these potential segment boundaries, such that $X_k = 1$ if there is a segment boundary between photos k and $k+1$, and 0 otherwise. Given a sequence of feature vectors S, our task is to find which gaps between consecutive photos correspond to segment boundaries and which do not:

$$f(X_k|S) = \begin{cases} 1 & \text{if the gap between photos } k \text{ and} \\ & k+1 \text{ is a segment boundary,} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

C. Hidden Markov Model with Alternating Observation Types

To find which gaps between consecutive photos correspond to segment boundaries, our approach takes the view that an event photo stream is the result of a stochastic process that generates feature vectors. A key aspect of our modeling is that we view the generation process as consisting of a set of foreground and background models. The foreground models generate the feature vectors that we want to find, *i.e.* the photo gap feature vectors corresponding to segment boundaries. The remaining models are background models that generate the surrounding feature vectors, *i.e.* photo feature vectors or photo gap feature vectors that do not correspond to segment boundaries.

To generate the event photo stream, the process emits alternating photo feature and photo gap feature vectors from the background models. At some point, the process switches to a foreground model at a segment boundary before switching back to a background model. This process continues until the end of the event photo stream (see Figure 4).

In this process, feature vectors in each photo taking session is generated by a pair of background models: one background model for photo features and another for photo gap features. For example in Figure 4, feature vectors in the photo taking session consisting of photos 1–2 are generated by the pair B_1 and B_3. This pair could generate feature vectors for other photo taking sessions in the stream. Suppose feature vectors in the photo taking session consisting of photos 6–10 are also generated by B_1 and B_3. The feature vectors in the two photo taking sessions, *i.e.* photos 1–2 and photos 6–10, would then follow the generated feature distributions of B_1 and B_3. For example, the feature distributions can be indicative of photos that are taken a few seconds apart, under good lighting conditions, at a medium distance from the subjects, with a similar background view, etc. Similarly, other photo taking sessions are generated by other pairs of background models with their own feature distributions.

The stochastic process of the foreground and background models can be described by a hidden Markov model (HMM). An HMM is a finite state automaton with stochastic state transitions and observation emissions [12]. An HMM assumes the process to be Markovian and as such, computations with HMMs are very efficient. Even though a simple probabilistic model, the HMM is a well-developed tool for modeling observation sequences and have been successfully applied to tasks in domains such as speech recognition [12], text segmentation and topic detection [13], and information extraction [14].

An HMM generates a sequence of observations, *e.g.* vectors of feature values, by starting at one of its states according to its prior probability. In this state, an observation is generated according to the emission probabilities of the state. The HMM then transitions to one of its states according to its state transition probabilities, which depends only on the current state. After the transition, another observation is generated according to the emission probabilities of the new state. The process continues until all observations have been generated.

With our concept of foreground and background models, the simplest HMM structure consists of three states: two states for the pair of background models and one state for the foreground model. This 3-state HMM is shown in Figure 5a. For two or more pairs of background models, we can use the 3-state HMM as a basic building block to form larger HMMs. Figure 5b shows an HMM with two pairs of background models: (B_1, B_3) and (B_2, B_4).

Since the event photo stream consists of alternating feature types, our HMM has two types of states to generate each of the feature types. In Figures 5a and b, only states B_1 and B_2 generate photo features. The remaining four states generate photo gap features. Of these four states, F_1 and F_2 are the foreground models that generate photo gap features corresponding to segment boundaries. All states model their emissions with a single Gaussian distribution per dimension to simplify parameter estimation. With the state transitions in this structure, the HMM will alternatingly transition from a photo feature state to a photo gap feature state, thus generating alternating photo and photo gap feature vectors.

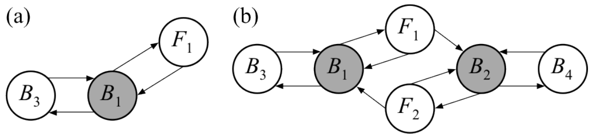

Figure 5. Grey HMM states generate photo features, while white HMM states generate photo gap features. States F_1 and F_2 represent foreground models that generate feature vectors corresponding to segment boundaries. States B_i represent background models that generate the surrounding feature vectors. The HMM in (a) has one pair of background models while the HMM in (b) has two pairs.

In our early experiments, we evaluated many HMM structures, from standard left-right and ergodic HMMs to a more sophisticated structure involving two HMMs, one for each feature type. We also explored HMMs that use a single observation type by concatenating adjacent photo feature and photo gap feature vectors into a single vector. Of them all, we found the HMM in Figure 5b with two pairs of background models to work best for our approach.

Note that more than two pairs of background models is possible with our approach. Ideally, the pairs of background models need to be trained with many labelled photo taking sessions from segmented event photo streams. This is similar to how phone models in automatic speech recognition is trained from thousands of labelled audio recordings [15]. Unfortunately, our task is not as mature as automatic speech recognition and so, there is no such segmented or labelled training data. As such, instead of training each 3-state *separately* — *i.e.* training $\langle B_1, B_3, F1\rangle$ separately from $\langle B_2, B_4, F_2\rangle$ — we train the *entire HMM* with unlabelled, unsegmented event photo streams and rely on smoothing with deleted interpolation [16] to alleviate issues with data sparsity and parameter initialization. We describe this training process as follows.

D. HMM Training and Application for Task

To facilitate further discussion, let us refer to the given event photo stream we want to segment as the `TARGET` photo stream. This photo stream is unlabelled and unsegmented. Let us then refer to the training data of unlabelled, unsegmented event photo streams as the `DATASET` photo streams.

First (see Figure 6), an HMM is trained using the `DATASET` photo streams. We call this the `DATASET` HMM. Parameters from this HMM is then used to initialize the parameters of a second HMM, the `TARGET` HMM, which is trained with the `TARGET` photo stream. In its training, the `TARGET` HMM parameters converge when they maximize the `TARGET` HMM's probability of generating the

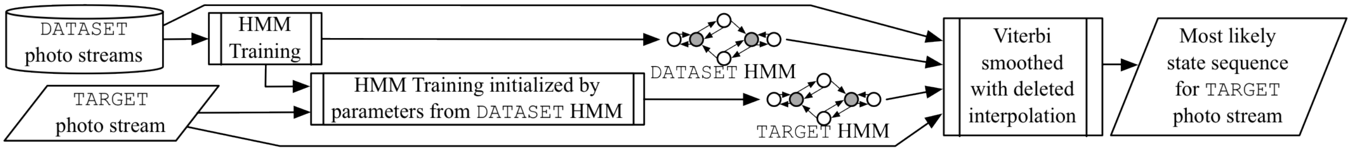

Figure 6. We use a separate set of event photo streams (DATASET) to alleviate data sparsity in the event photo stream we want to segment (TARGET). All photo streams are unlabelled and unsegmented. The four inputs are needed to perform the Viterbi algorithm with deleted interpolation [15], [16].

TARGET photo stream feature vectors. To determine the TARGET HMM's state sequence in generating the given feature vectors with maximum probability, we use the Viterbi algorithm [12] with deleted interpolation, a smoothing technique that finds the smoothing parameters between two distributions depending on how well-trained each distribution is. We use deleted interpolation, as is typical in speech recognition [15], to alleviate data sparsity by smoothing the parameters of the TARGET HMM with parameters from the DATASET HMM, which was trained with much more data. Finally, with the state sequence we can determine which photo gap feature vectors were generated by the foreground models, and hence correspond to segment boundaries.

In the next section, we will show that our method, while trained suboptimally *without* labelled, segmented event photo streams, outperforms existing methods including the state-of-the-art cluster tree algorithm by Graham *et al.* [2].

IV. Evaluation

We collected 28 event photo streams of various event types, *e.g.* wedding, travel, cruise, concert, etc. Four event photo streams are from publicly available Flickr photo sets[2]. The remaining 24 were obtained from seven volunteers. In total, our evaluation data set consists of 5188 photos, with an average and median of 185 and 168 photos respectively.

For the four streams from Flickr, the photo owners were not available to annotate the sets. As such, the first author manually segmented the photos to provide ground truth. For the remaining 24, we asked the contributors — as photo owners — to provide the ground truth. This practice is in line with many photo stream segmentation works we reviewed in Section II, which also require ground truth for their evaluation.

As baselines, we have implemented the cluster tree event photo stream segmentation algorithm [2] and five automatic albuming algorithms from Section II: fixed threshold, best-first model merging [4], adaptive threshold [4], K-means [3], and event ending probability [5].

To evaluate the segmentation results of our method and the baselines against the ground truth segmentations, we used the error rate metric, Pr_{error}, proposed by Georgescul *et al.* [17]. This metric improves on *WindowDiff*, previously used by Naaman *et al.* [18] to evaluate their automatic albuming method. A lower Pr_{error} indicates better agreement with the manually segmented ground truth; a score of 0 indicates perfect agreement. Pr_{error} is an average of the miss and false alarm rates. As such, a method that proposes no segment boundaries or proposes segment boundaries everywhere will have an error rate of about 0.5.

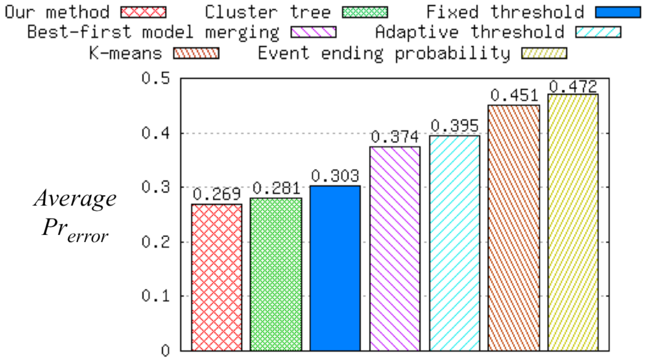

Figure 7. Comparison between our method and the baselines, averaged over all event photo streams, in terms of error rate against ground truth segmentations (smaller number is better).

Results are shown in Figure 7. We can derive several salient points from the results. The best-first model merging method which utilizes visual information alone did not perform well and ranked fourth place. This was caused by a relatively high miss rate, suggesting that visual similarities amongst the photos hinder the method from finding any segment boundaries. The adaptive threshold method which is a simple and well-known automatic albuming method, performed worse than the simplest baseline — the fixed threshold method — when used to segment event photo streams. Methods that rely on heuristics such as the K-means and the event ending probability methods performed the worst, finding very few segment boundaries, resulting in very high miss rates and correspondingly high error rates.

The best baseline is the state-of-the-art cluster tree event photo stream segmentation algorithm. Our method however, had the lowest error rate overall. Our method is statistically significantly better than the cluster tree method ($p < 0.10$) and even more statistically significant compared to the other baselines ($p < 0.005$).

Our method has the lowest miss rate among all methods we studied, but the highest rate of false alarms. We believe

[2] Flickr photo set ID: 847825, 1068265, 72157601961445922, and 72157603826353321.

that for end users, having a low miss rate is more valuable than having a low false alarm rate. To correct a false alarm is a one-step process of removing the incorrect segment boundary. But to correct a miss, the user must first realize that there is a miss, then figure out the position of the segment boundary.

Why does our method produce more false alarms? We believe it is produced during the Viterbi algorithm when the HMM — with its trained parameters — incorrectly finds that transitioning to a foreground model (*e.g.* transitioning from B_1 to F_1 in Figure 5b) has a higher probability than transitioning to a background model (*e.g.* B_3). One possible reason for the lower probability is the lack of training data for the feature vectors corresponding to the false alarms. A more likely reason is however, the lower accuracy associated with training the HMM without labelled data.

Nonetheless, the error rate was computed by penalizing misses and false alarms equally. Overall, our method outperformed all the baselines despite training our HMM suboptimally without labelled, segmented training data. We expect to achieve even better results with access to such data.

V. Conclusion

To help make large event photo streams more manageable, we proposed a method for event photo stream segmentation, *i.e.* the process of finding contiguous groups of photos from an event photo stream, each of which corresponds to a photo-worthy moment in the event. Our model leverages our observation that photo streams exhibit alternating photo and photo gap feature types. We use it to formulate the problem and the structure of our proposed HMM. We then described how the HMM can be trained without labelled data and how we addressed the issue of data sparsity and parameter initialization with deleted interpolation smoothing. In the evaluation, we showed that many existing photo stream segmentation methods are unsuitable for our task. Overall, our method performed better than all baselines, including the state-of-the-art cluster tree algorithm.

Our next step is to collect labelled training data. This will allow us to train each 3-state as a separate HMM and obtain more accurate HMM parameters. Obtaining significantly more training data would also allow us to re-evaluate other photo and photo gap features that may not have worked well due to data sparsity issues.

References

[1] D. Kirk, A. Sellen, C. Rother, and K. Wood, "Understanding photowork," in *Proc. CHI*, 2006, pp. 761–770.

[2] A. Graham, H. Garcia-Molina, A. Paepcke, and T. Winograd, "Time as essence for photo browsing through personal digital libraries," in *Proc. JCDL*, 2002, pp. 326–335.

[3] A. Loui and A. Savakis, "Automated event clustering and quality screening of consumer pictures for digital albuming," *IEEE Transactions on Multimedia*, vol. 5, no. 3, pp. 390–402, September 2003.

[4] J. Platt, M. Czerwinski, and B. Field, "PhotoTOC: Automatic clustering for browsing personal photographs," in *Proc. ICICS*, 2003, pp. 6–10.

[5] M. Zhao, Y. Teo, S. Liu, T.-S. Chua, and R. Jain, "Automatic person annotation of family photo album," in *Proc. ACM CIVR*, 2006, pp. 163–172.

[6] U. Gargi, "Modeling and clustering of photo capture streams," in *Proc. MIR*, 2003, pp. 47–54.

[7] A. Pigeau and M. Gelgon, "Spatial-temporal organization of one's personal image collection with model-based ICL clustering," in *Proc. CBMI*, 2003.

[8] B. Gong and R. Jain, "Segmenting photo streams in events based on optical metadata," in *Proc. IEEE-ICSC*, 2007.

[9] T. Mei, B. Wang, X.-S. Hua, H.-Q. Zhou, and S. Li, "Probabilistic multimodality fusion for event based home photo clustering," in *Proc. ICME*, 2006, pp. 1757–1760.

[10] M. Cooper, J. Foote, A. Girgensohn, and L. Wilcox, "Temporal event clustering for digital photo collections," in *Proc. ACM Multimedia*, 2003, pp. 364–373.

[11] P. Sinha and R. Jain, "Classification and annotation of digital photos using optical context data," in *Proc. ACM CIVR*, 2008, pp. 309–317.

[12] L. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proc. the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[13] P. V. Mulbregt, I. Carp, L. Gillick, S. Lowe, and J. Yamron, "Text segmentation and topic tracking on broadcast news via a hidden markov model approach," in *Proc. ICSLP-98*, 1998, pp. 2519–2522.

[14] D. Freitag and A. K. Mccallum, "Information extraction with hmms and shrinkage," in *Proc. AAAI Workshop on Machine Learning for Information Extraction*, 1999, pp. 31–36.

[15] K.-F. Lee, *Automatic Speech Recognition: The Development of the Sphinx System.* AH Dordrecht: Kluwer Academic Publishers, 1989.

[16] F. Jelinek and R. Mercer, "Interpolated estimation of markov source parameters from sparse data," in *Proc. Workshop on Pattern Recognition in Practice*, 1980.

[17] M. Georgescul, A. Clark, and S. Armstrong, "An analysis of quantitative aspects in the evaluation of thematic segmentation algorithms," in *Proc. SIGdial Workshop on Discourse and Dialogue*, 2006, pp. 144–151.

[18] M. Naaman, Y. Song, A. Paepcke, and H. Garcia-Molina, "Automatic organization for digital photographs with geographic coordinates," in *Proc. JCDL*, 2004, pp. 53–62.