Consumer Media Capture: Time-Based Analysis and Event Clustering

Ullas Gargi
Imaging Systems Laboratory
HP Laboratories Palo Alto
HPL-2003-165
August 7th , 2003*

E-mail: ullas.gargi@hp.com

fractal, clustering, photography, consumer, capture, organization, management

We present an analysis of consumer media capture behavior based on timestamp metadata. We show that it is bursty and is not well described by a Poisson model. We show that it is in fact a fractal process, with fractal dimension characteristic of the capturer. We also present an algorithm for splitting consumer media into event clusters based on capture-time metadata, an algorithm for minimal labeling of such clusters, and a method for visualizing the results of such clustering on highly bursty data. We present results of the analysis on consumer photograph collections and results of the clustering using two software implementations, one for static organization and one for dynamic browsing.

* Internal Accession Date Only
Approved for External Publication

Consumer Media Capture: Time-Based Analysis and Event Clustering

Ullas Gargi
Hewlett-Packard Laboratories, Palo Alto, California
ullas.gargi@hp.com

1 Introduction

Media capture by a person, in particular photographic image capture, is a bursty (i.e. inter–arrival times are very far from the mean) statistical process. Typically, a person will not capture media for long periods of time and then capture a lot of media in a relatively short period of time. These bursts of capture actions typically correspond to interesting events or scenes that he or she wishes to record, such as parties, birthdays, holidays, etc. Given a large collection of digital photographs or media objects captured or recorded by one person, with timestamps but little other metadata, it is natural to seek a method for temporal segmentation that maps well to the events the person is likely to remember. We describe some analysis and modeling of real capture-event streams and describe an algorithm for clustering consumer captured media into events. We generalize the list of timestamps as a capture-event stream to allow other metadata than timestamps to be amenable to the methodology.

1.1 Data sets

We used the personal photograph collections of five people to generate our data. Table 1 lists the photograph collections we used. The codenames will be used to refer to them. The first two characters denote the initials of the owner. Note that as far as possible, we tried to use photos belonging to one person's collection, or their immediate family's, that were captured by them or their immediate family. We avoided photos in their collection that were captured by temporary visitors or shared by others. This was so that we model the true media capture behavior of one person or family. In practice, most photos in a family are probably taken by a single designated photographer.

Table 1: Photograph collections used as datasets

	Codename	Number of Images	Temporal Extent	Comments
1	PO-meercat	814	3 years	
2	PO-SanDiegoMay2003	314	6 days	No overlap with PO-meercat
3	UG-full	1255	3 years 3 months	
4	UG-nik	430	15 months	subset of UG-full
5	TZ-Iris	589	10 months	
6	TZ-ChinaVisit	114	16 days	no overlap with TZ-Iris
7	MP-USVisit	361	36 days	
8	YD	649	2 years 3 months	

2 Modeling

We seek mathematical models for our data: a sequence of events with timestamps but no value attributes. Time-series analysis is inapplicable because of the lack of values at each datapoint, unless one assigns a binary value to each time unit, depending on whether a photograph is taken or not, which seems rather arbitrary. We follow the approach taken in the modeling of network packet traffic data, specifically the arrival times of TCP/IP packets at a network router [CS00].

2.1 Poisson model

The Poisson distribution is often used to model processes such as network packets arriving at a router or customers arriving at a service node. If $A(t)$ is a counting process representing the number of arrivals (events) from time 0 to time t, arrivals in disjoint time intervals are independent, then [BG92] :

$$P\{A(t+\tau) - A(t) \; = \; n\} = e^{-\lambda\tau}\frac{(\lambda\tau)^n}{n!}$$

The mean of the distribution is λ. The variance is also λ.

However this is not a good fit for photographic capture by a single person because the consecutive capture events are probably not independent. Taking a photograph increases the probability that a second one will be taken. This was corroborated by attempting a Poisson fit to two data sets, UG-nik and PO-meercat. For the first, quantizing time into units of weeks, days, hours and minutes and then fitting a Poisson model using the Matlab `poissfit()` function yielded the means and variances listed in table 2.

Table 2: Poisson fit to dataset UG-nik

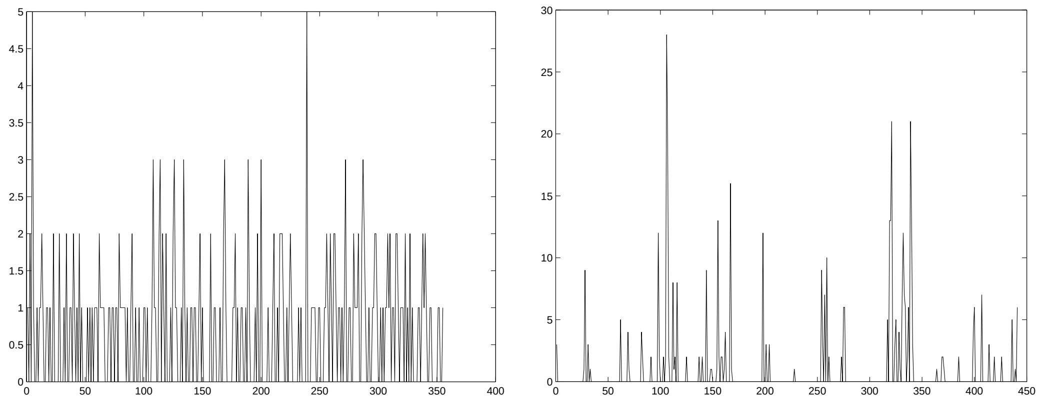
Time Unit	Lambda	Sample Mean	Sample Variance
Week	6.82	6.82	148.24
Day	0.6873	0.6873	9.56
Hour	0.0406	0.0406	0.2018
Minute	0.0006	0.0006	0.0012

Figure 1 shows a random Poisson sequence with a λ of 0.68 parameter (using the Matlab `poissrnd()` function) and the actual event counts for the day resolution. Comparing these two figures we see the difference in variances. Numerically, the data has a variance of 9.56, an order of magnitude larger than the Poisson model. Clearly a Poisson process is not bursty enough to model our data.

2.2 Fractal nature of photographic capture process

Consider Figure 2. This shows the capture-event stream for the PO-meercat dataset with three different time quantizations, day, hour and minute respectively. They are quite similar in appearance at the different scales. This suggests that perhaps the capture process is fractal.

We use the box-counting method[PJS92] to estimate fractal dimension. We change the time unit through weeks, days, hours and minutes and count how many time bins are occupied. We arbitrarily assign a scale value of s=1 to the week time scale. The scales for the others are then 1/7, 1/168 and 1/10080. We plot log (count) vs. log (1/s). Figure 3 shows the plots. They are quite linear. We fit a line to the plots and the slopes of the lines are the respective box-counting fractal dimensions. Table 3 lists the computed box-counting fractal dimensions for all the datasets.



(a) (b)

Figure 1: Comparing (a) a Poisson to (b) a capture-event counting process, both with $\lambda = 0.68$

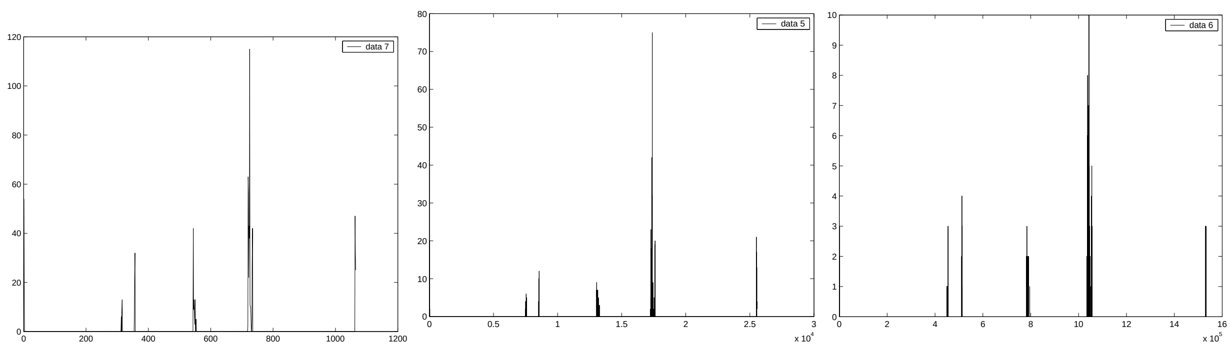


Figure 2: Capture-event arrival plots at different time scales (unit = day, hour, minute) for the PO-meercat dataset

The values of the fractal dimension for the capture-event streams are all well below 1. This makes sense for bursty processes, where increasing the number of bins will not linearly increase the number of occupied bins. It is interesting to note that the fractal dimension of a person's media capture behavior is relatively constant over different collections. For example, the TZ-Iris and TZ-ChinaVisit datasets are completely distinct and have very different time scales, but have very similar fractal dimensions. This leads to a conjecture that the fractal dimension of the media capture process is characteristic of a person. We digress briefly to discuss why this might be so.

In the literature, achievement (or motivation) theory and rational decision theory are used to model human choice of alternative actions. The classic formula here is Atkinson and Feather's qualification of Lewin et al and Herbert Simon's earlier work on rational choice [Str98, AF66]:

$$T_r = M_s P_s I_s + M_f P_f I_f$$

T_r is the resultant tendency, the utility or subjective valence assigned to a particular behavioral alternative. It is split into parts for success and failure. M is the personal motive strength of the person, P the probability of that outcome and I is the incentive strength of the task itself [Str98]. In the consumer media capture context, I corresponds to the personal psychological need to record events by capturing media. One can further break this down into I_c, the incentive to take the camera along, I_t, the incentive to take the camera off the shoulder or out of the case, and I_p the incentive to raise the camera and turn it on in anticipation of capturing media (the failure counterparts correspond to disincentives such as the weight and hassle of carrying the camera, the loss of participation in the event involved in capturing it, etc.). P_s is the probability that something of interest happens resulting in the shutter button being depressed. If the I values are constant for a person, then the fractal dimension of their capture behavior may be derivable from these values. For example, people with very bursty capture behaviors may have low I_c and I_t but high I_p. This equation needs to be modified to include the past history of capture actions to fully model capture behavior.

Table 3: Box-counts and fractal dimensions for datasets

	Boxcount				Fractal
Dataset	Week	Day	Hour	Minute	Dimension
PO-meercat	11	33	135	505	0.4106
PO-SanDiegoMay2003	4	11	54	168	0.4071
UG-full	99	169	260	683	0.2016
UG-nik	37	80	137	308	0.2186
TZ-Iris	16	31	102	397	0.3505
TZ-ChinaVisit	3	7	20	83	0.3549
MP-USVisit	3	8	46	243	0.4806
YD	27	51	169	546	0.3296

2.3 What is the natural scale of a sequence?

Given our desire not just to model the capture process but to write an algorithm to allow automatic clustering and organization (see section 3), we would like to to be able to characterize a particular sequence or sub-sequence as occurring at its natural or characteristic scale. For example, if we know (somehow) that a particular sequence of captures was taken at the "day" scale, then this information could be used to tune the clustering algorithm or choose the scale at which it is applied.

The question of what the natural scale of a sequence is, is related to the question of how many clusters that sequence should be segmented into. In the latter case the Facility Location problem [1] or a statistical method such Tibshirani et al [RTH00] which proposes a method for estimating the number of clusters in a dataset, can be used. The latter method has the advantage of working with any clustering technique and not requiring the definition of cluster centers. The authors remark that their simulation studies suggest that their gap estimate is good at identifying well separated clusters, but not so if the data is not well separated.

This appears to be an open research problem in general, and for our particular domain, leads to the following question.

Is a multi-resolution hierarchy needed?

When people remember events, it is likely that they do use a very short multi-resolution scheme, the year of occurrence perhaps followed by the event. In the absence of any psychological data that we are aware of to support this statement, we merely conjecture that, although attractive to software designers, a multi-level multi-resolution organization scheme is not necessary for personal media organization. i.e. A division into years followed directly by the event-cluster is an organization that most people will find useful. We note here that the algorithm described in later sections does not explicitly cluster at a particular time scale, but allows 3 separate levels of clustering, roughly designed for "many years worth", "a few months worth" and "a few days worth" of photographs. As noted in earlier work [GDT03] personal digital media collections do not yet extend over many decades so event-time makes a good index. Once collections span much larger temporal extents, other indexing schemes may become more important.

[1] The facility location problem in operations research is: given a set of facilities and a set of customers, pick how many and which facilities should be kept open to minimize the cost of serving customers, where total cost is a function of both each customer's distance to the nearest facility and the cost of keeping a facility open. Closed facilities incur no cost.

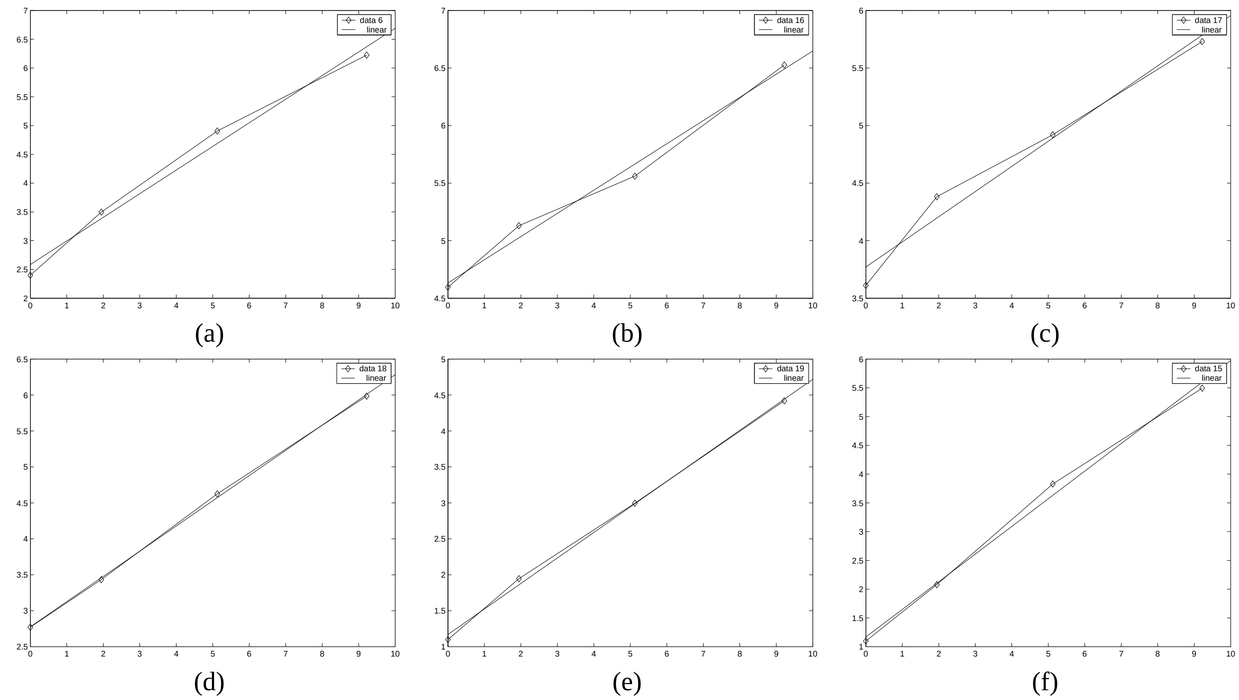


Figure 3: Log-log plots of box counts vs. time scales and fitted lines for (a) PO-meercat, (b) UG-full, (c) UG-nik, (d) TZ-Iris, (e) TZ-ChinaVisit and (f) MP-USvisit datasets.

3 Clustering

There are many algorithms for clustering a data sequence. See for example [GMM$^+$03] for a method for data stream clustering. One difference to our problem is that we are not concerned with finding cluster centers to minimize some cost function (distance of data points to cluster centers) but rather segmentation or splitting of the time sequence, so that intra-cluster variance is reduced. Thus, where a traditional clustering algorithm such as K-means might find clusters and cluster-centers given K, and where the cost function would change if those cluster-centers are moved, we only care for the assignment of points to clusters and as long as the segmentation remains the same, there could be more than one set of cluster centers that would be satisfactory.

3.1 Heuristics

Our algorithm attempts to heuristically split a capture-event stream into clusters that correspond to events. It uses the following two empirically observed properties of capture-event stream events:

- A long interval with no capture usually marks the end of an event
- A sharp upward change in the frequency of capture usually marks the start of a new event

The first heuristic needs to define "long". This is defined as being either large relative to the extent of the cluster currently being grown, or large relative to the average inter-capture interval.

3.2 Splitting algorithm

The algorithm is bottom-up and adaptive. $|C_k|$ is the extent (i.e. range, maximum - minimum) of cluster C_k. $|C_k|_\#$ is the cardinality of cluster C_k.

1. Given the capture-event stream $x(i),\ i\ = 0...N-1$, sort it ascending.
2. The first cluster C_0 is begun with $x(0)$. Examine the next item $x(1)$.
3. At any time if item $x(i)$ is being examined for candidate inclusion in the current cluster C_k, mark $x(i)$ as a split point, i.e., as being the start of a new cluster C_{k+1} if 3a and 3b and either of 3c or 3d is true.
   (a) C_k has a certain minimum number of items $|C_k|_{\#\ min}$
   (b) C_k has a certain minimum extent $|C_k|_{min}$
   (c) $x(i)\ -\ x(i-1) > f(|C_k|_\#) \times |C_k|$
   (d) $x(i)\ -\ x(i-1) > g(|C_k|_\#) \times$ *recentaveragediff*$_w(i)$
4. Else if *recentaverage*$_w(i) < h \times$ *recentaverage*$_w(i\ -\ w)$ mark a split.
5. Else, add $x(i)$ to C_k, update *extent*(C_k), *recentaverage*$_w(C_k)$, f, g, and h and continue with $x(i+1)$.
6. Once $x(N-1)$ has been processed, reverse the stream and repeat steps 2 - 5
7. Merge the two sets of clusters from the forward and backward phases.

recentaverage$_w(C_k)$ returns the average inter-capture interval for the current cluster over a local window w in the past. The functions f, g and h are empirically determined and gradually decrease as the cardinality of C_k increases. Thus as a cluster grows, the relative gap needed to start a new event decreases. The initial values of both functions as well as the values of $|C_k|_{min}$ and $|C_k|_{\#\ min}$ are determined by a user-settable parameter that controls the amount of splitting. Our current implementation provides three levels of splitting roughly designed for “many years worth”, “a few montsh worth” and “a few days worth” of photographs: more, medium and less.

3.3 Automatic minimal-labeling by relative extent

Given a capture sequence and a clustering, each cluster can be automatically minimally-labeled based on its extent relative to its neighbors. For example, a cluster with photographs in [11 June 2002, 18 June 2002] with neighbors containing photos taken solely in May and July, would reasonably be labeled “June 2003”. This is the minimal-labeling, by which we mean the most general label that also completely distinguishes that cluster. The same cluster with neighbors [2 june 2002, 5 june 2002] and [25 june 2002, 30 june 2002] might be labeled “June 11-18, 2002”. The same cluster with neighbors having photos taken solely in 2001 and 2003 might reasonably be labeled “2002”. If new photos taken in 2002 are later introduced into the collection, the cluster would then be re-named appropriately.

The algorithm for automatic labeling is very simple: the labeling level is the smaller of the largest time units at which the left and right ends of the cluster fall into different units than the neighboring clusters’ ends.

1. For each cluster we will generate a label $L_p - L_s$ to denote the extent of the cluster where L_p is the prefix (starting point) and L_s the suffix, or the end.
2. Compute the smallest standard unit enclosing the cluster (e.g. day, month, season, or year; street, zipcode, city, county, metropolitan area, state, or country; etc.). Let this unit be U.

3. For the particular case of a 1-D event-stream, e.g. timestamps, let the cluster have left and right ends C_l and C_r. Let the cluster's left neighbor have right end L and its right neighbor have left end R. Let U_l = largest standard unit possible (e.g. year).
4. Quantize C_l and L by U_l.
5. If quantized C_l and L values are different, return U_l.
6. Else, Decrement U_l and repeat.
7. If U_l reaches smallest standard unit, return it.
8. Repeat this for C_r and R to get U_r.
9. Labeling level U = min $\{U_l, U_r\}$
10. Label prefix $L_p = C_l$ quantized at the U level. Label suffix $L_s = C_r$ quantized at the U level. Optionally the two may have different levels.
11. If L_p and L_s are identical, then they are collapsed into one label.

Note that this is the only place where time boundaries are used. Unlike some other time-based organization methods, we do not use explicit day boundaries to segment capture-event streams.

This algorithm is quite general and can be used for naming of any grouped data with continuous-valued metadata that has multiple resolutions of representation. It can also be generalized to more than one dimension. Particularly, it can use geographical location or any other continuous-valued metadata.

3.4 Software implementation

A software implementation called FotoSplit was prototyped in 2001-2003. It incorporates the algorithm described above. It is written in Perl using the Tk toolkit (through the Perl/Tk interface) for the GUI and also converted to a Windows binary. It has the following functionalities: it allows a directory root with photographs to be specified. It then invokes a JPEG EXIF header reader to obtain the capture timestamps. These are clustered and each cluster is represented using the first and last image in it. Clusters are automatically minimally-labeled. They can also be hand labeled. The resulting clusters can be used to create directories into which the photograph collection is organized.

The clustering algorithm is also implemented within the Meercat multimedia information management system as a browsing tool. It allows a users photograph album to be dynamically grouped into time-based clusters. Figure 4 shows an album of 168 images from a five-day trip with the default organization (a list of the images) and the dynamic organization into time-based clusters. The clustering splits the raw list into one cluster per day, without knowing anything about day boundaries. The time-cluster organization is presented as a choice of virtual view for all user albums and computed on the fly when the user chooses that view of an album.

Even if this kind of dynamic organization for browsing is fast and always available, static organization is probably still valuable because of the psychological need people have to organize their possessions into bins.

4 Results

A note on visualizing bursty univariate data

In many applications of visualizing data one can either assume that the data is not highly bursty and hence can be well represented on a single scale, albeit after using dynamic range compression; or else that a dynamic zooming technique [Spe01] can be used to switch between context (overview) and focus (burst view) modes.

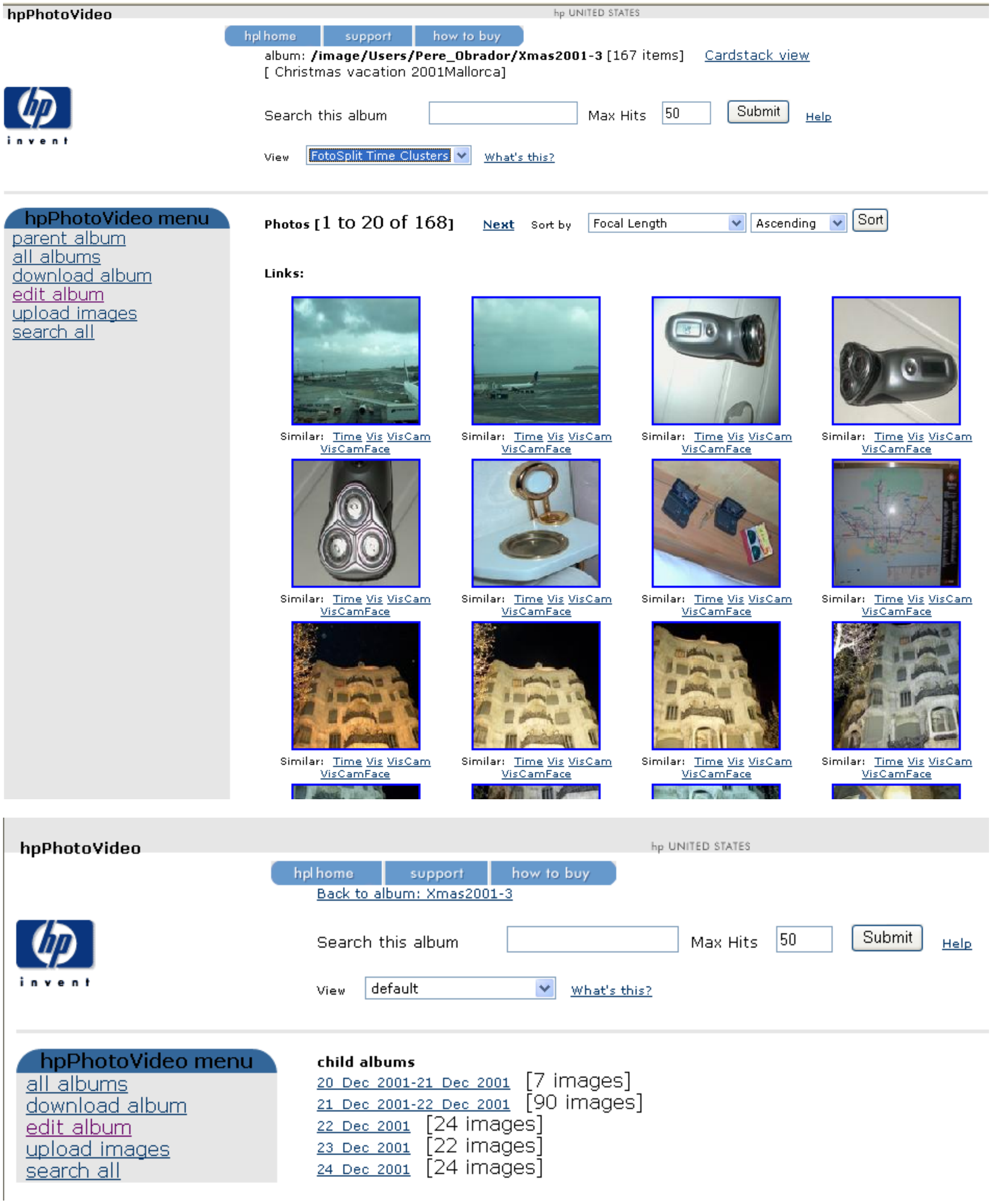


Figure 4: Screenshot of (a) default organization and (b) dynamic time cluster organization of a user album within the Meercat system

Neither of these is true in our case. Depicting the capture-event stream for our datasets on a static page is challenging because of the large dynamic range of inter-event times involved. Consider that a capture-event cluster can have average inter-event times on the order of seconds, while the inter-cluster distance can be in the order of weeks or more, a six orders of magnitude difference. Figure 5 shows the capture-event streams with the capture-events marked with black dots on the main timeline. Each visible dot is in fact composed of many capture-events that are too close together to be distinct. Even with log compression or log(log(log(log))) compression, the points are not separated because of their relatively small timescale. In evaluating the clustering performance visually it is important to see the number and separation of points within a cluster as well as the inter-cluster separation. Only the latter is visible.

We experimented with using the vertical dimension to separate close points by assigning (meaningless) separated vertical axis values to consecutive points. However, this a) did not separate the points enough to make them visible and b) introduced distracting vertical cues.

We then tried the technique shown in Figure 5. The main line consists of the entire data set at the largest scale. Each cluster is zoomed in on in parallel lines on either side of the main line, in alternation. This is also insufficient to expand each cluster.

However, the basic idea of using the free extra dimension available was sound. We finally settled on the technique used to display clustering results in the following sections. While the overview plot is linear, each cluster sub-plot is modulated by a high-frequency sine wave. From 0 to 500π works well. This vertically separates the points that were horizontally indistinct but without introducing visual artifacts of vertical separation.

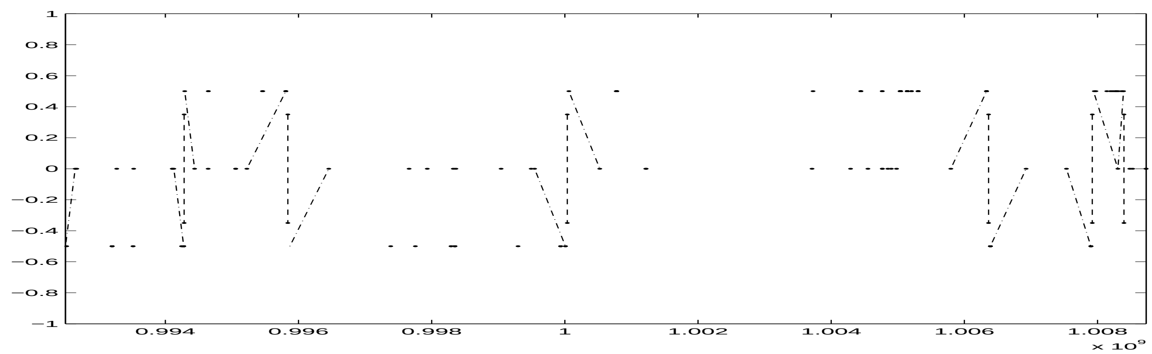


Figure 5: Linearly expanding each cluster with sub-plots

Performance

Running the algorithm on three of the datasets yielded the split points in sequences depicted in Figures 6 through 8. For each figure, the splitting level is listed. The time values are represented in seconds since the start of the epoch, i.e. January 1, 1970. Each capture-event is marked with a black dot and the split points, which are the mid points between clusters, are marked with vertical dashed lines.

Screenshots of the program on datasets MP-USvisit and TZ-ChinaVisit are shown in Figure 9. The clusters are automatically named. For the latter collection, the owner had also created a manual organization that corresponded very well with the automatic groupings.

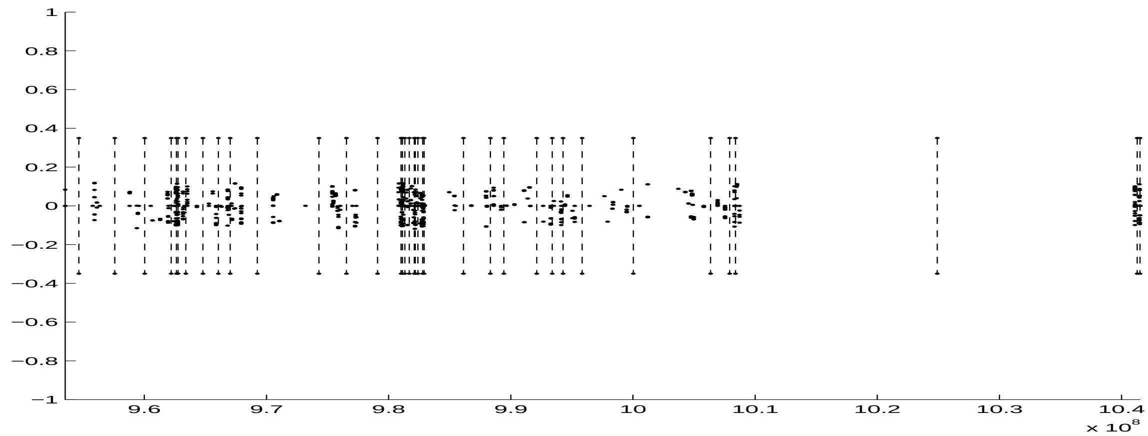


Figure 6: Clusters resulting from running the splitting algorithm on dataset UG-nik with medium splitting (38 clusters)

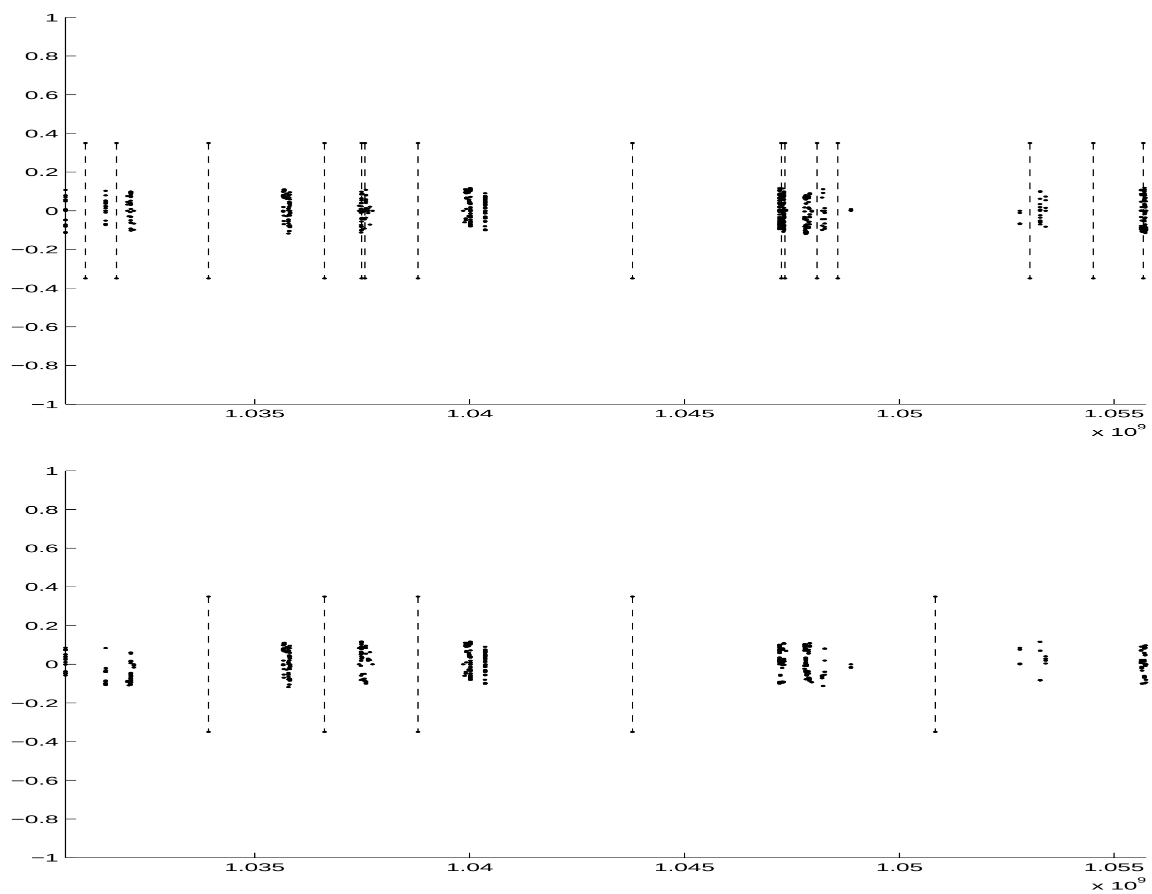


Figure 7: Clusters resulting from running the splitting algorithm on dataset TZ-Iris with (a) medium splitting (16 clusters) and (b) less splitting (6 clusters)

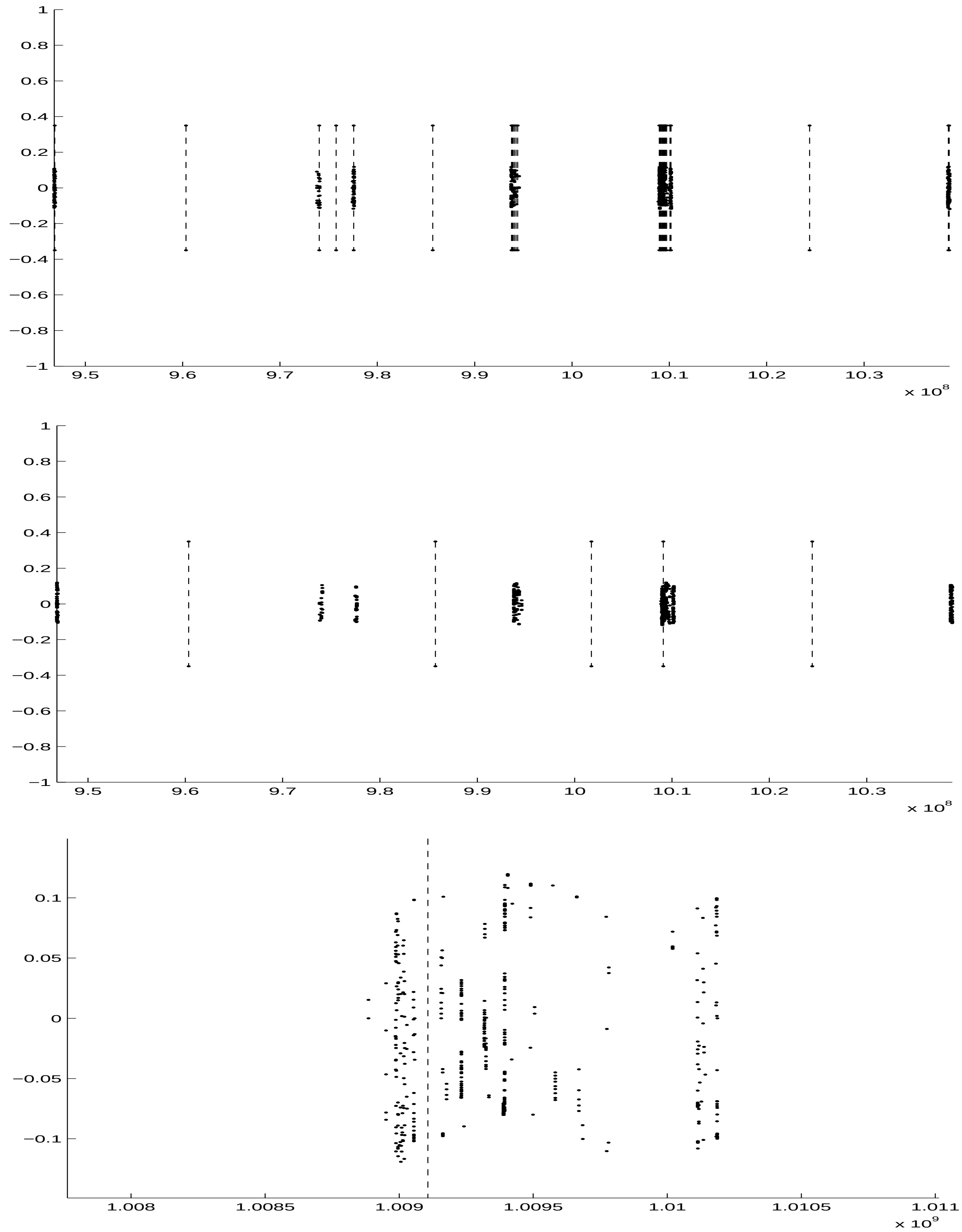


Figure 8: Clusters resulting from running the splitting algorithm on dataset PO-meercat with (a) medium splitting (28 clusters) and (b) less splitting (6 clusters). (c) shows a zoomed portion of the split between 4th and 5th clusters of (b)

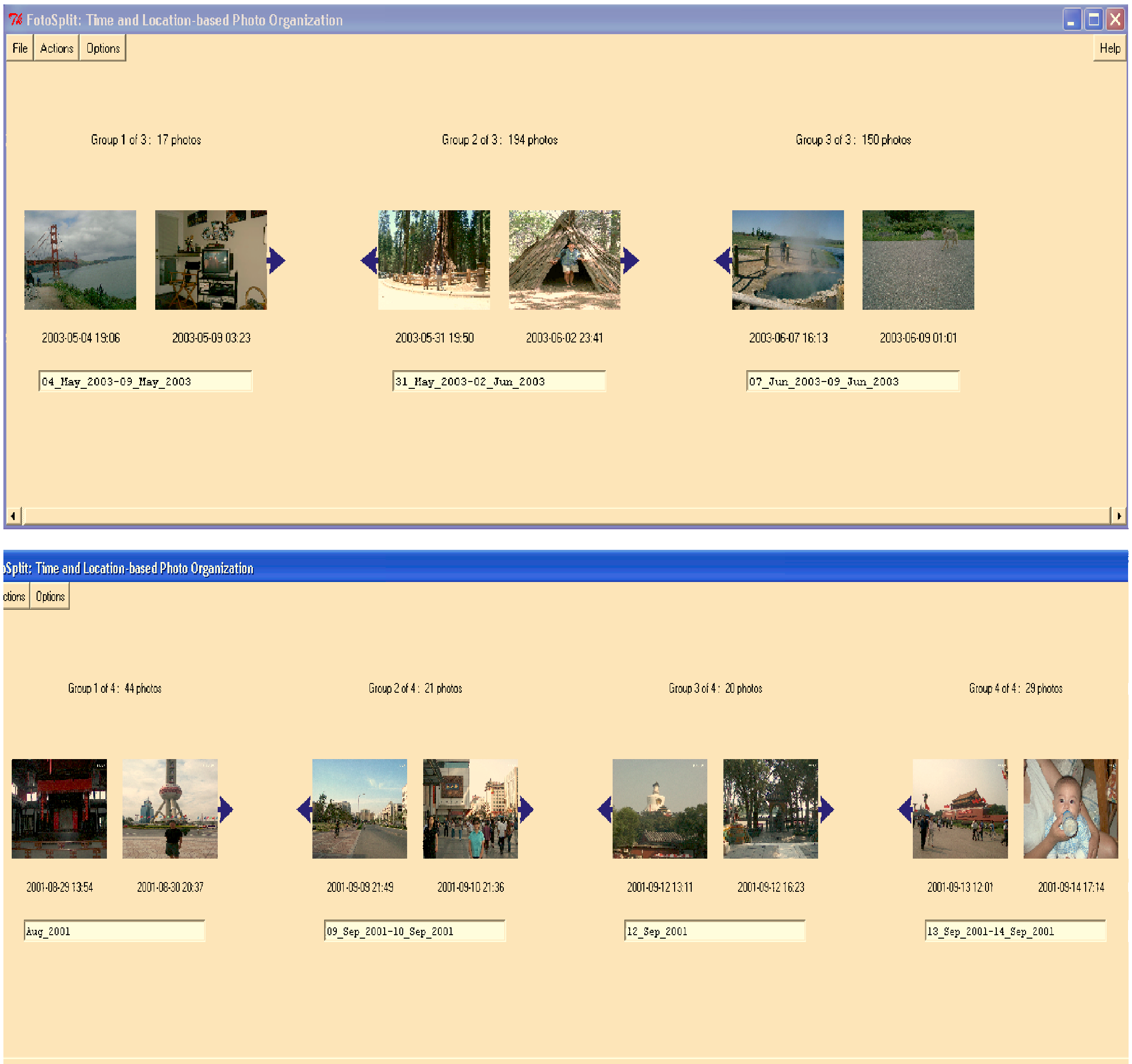


Figure 9: Screenshots of the FotoSplit program with results on dataset MP-USvisit and TZ-ChinaVisit

5 Related Work

Multiscale analysis and modeling of bursty processes such as the use of Multifractals and the multifractal spectrum, fractional brownian motion, etc. is common in fields such as network traffic modeling.

Loui and Savakis [LS00] describe an algorithm for event segmentation using photograph timestamps. They use 2-means clustering of inter-photo time-gaps in minutes to divide into intra-event and inter-event time-gaps. The latter are then used to split the stream into events, i.e. gaps belonging to the latter cluster are marked as split points (thus making the algorithm not locally-adaptive). They also use low-level similarity to break an event into sub-events.

Graham et al. [GGMPW02] describe an algorithm for clustering photos based on timestamps. Their system uses a hierarchical clustering algorithm. At the top level it performs fixed time difference splitting using an empirically determined threshold. Then each such cluster formed is split based on whether the current time difference is an outlier for the normal cluster rate. Outliers are defined as those whose values is more than the 75th percentile value + 2.5 times the (75th-25th) percentile range. Clusters are merged into ones occupying the same day for presentation purposes.

The PhotoTOC [PCF02] application uses a locally-adaptive clustering algorithm: the time difference is compared to a local window average of time differences and thresholded. Time differences are logarithmized to handle their large dynamic range.

Rodden et al [RW03] evaluated which features implemented in the Shoebox photo management system citeshoe-box2000 were found useful by consumer for organizing photogaphs taken over a six-month period. They found that most people liked to organize by events, such as holidays, with photos within an event album sorted chronologically. People tended to label rolls or albums but very rarely re-labeled individual images. Advanced functions such as content-based retrieval or speech recognition were not found useful. These findings support the approach and work described in this report.

Various EXIF-relevant utilities exist. For example, EXIFRenamer is a shareware software application for the Apple platform that renames files based on their EXIF timestamps. However, no clustering is performed.

6 Summary

We have presented an analysis of consumer photographic media capture using real datasets. We have shown that personal media capture is bursty and fractal, with fractal dimension characteristic of each person. We have presented an adaptive algorithm for clustering media based on timestamp metadata that allows the creation of a relatively-flat organization. The algorithm has been implemented and tested on a number of datasets. We have presented an algorithm for minimally labeling groups based on their relative extent. We have presented a method to visualize highly bursty univariate data without interaction.

7 Future Work

More analysis and modeling of the media capture behavior of people is required to well understand the phenomenon. In addition, this work could be extended to aggregated behavior, such as that of many different unrelated people at the same tourist spot, or by extended families over a period of time. It is likely that the Poisson model may fit such aggregated media capture processes better.

Methods to estimate the characteristic scale of a dataset would have applicability in many domains beyond that of media organization.

One could model the photographic capture behavior of a person by a set of states such as “Month mode”, “Week mode", "Day mode" "Minute mode" and "Photo Capture" with probabilistic transitions between them. Representing a capture-event stream by an observation vector such as the numbers of photos in the next month, week, day, hour, one could fit a Markov model to an observed capture-event stream.

Given a model of a particular person's media capture behavior, one could detect when a photograph collection was captured by that person. The fractal dimension or Markov model could be used for this purpose.

Knowledge of the fractal dimension of a person's media capture behavior could be used to tune the clustering algorithm. For example, small dimensions indicate a burstier process so the (relative) time interval needed to mark a split could be reduced. It could conceivably also be used to personalize the capture device, say in terms of setting the auto-power-off time and changing the image-pipeline data flow.

The software implementation could be extended to better visualize the contents of a cluster and to allow splitting and merging for interactive adjustment of automatically generated clusters. These features would make it more useful as a general purpose media organizer. Work to extend the time clustering to include location metadata, such as from a GPS track log, is ongoing.

The methods of analysis presented here could possibly also be extended to consumer photograph printing behavior. It is possible that consumers do not print photos for a long time but when they do print, they print more than one photo. A bursty statistical or fractal model of printing behavior could be coupled with a measure of subjective image quality or semantic importance to make personalized print recommendations.

Event clustering could also be used for episodic segmentation of life stream data obtained from always-on sensors recording a humans status, activities and information flow.

Event clustering is also applicable to other kinds of event data such as customer interaction events, business process events and events in large communication networks.

8 Acknowledgements

We are grateful to Yining Deng, Pere Obrador, Maurizio Pilu and Tong Zhang for providing us access to their personal photograph collections. We are grateful to Mauriziu Pilu for interesting discussions regarding modeling media capture. We are grateful to Dan Tretter for motivating the work and suggesting the GUI for the software implementation.

9 References

[AF66] J.W. Atkinson and N.T. Feather. *A theory of achievement motivation*. Wiley, New York, 1966.

[BG92] Dimitri Bertsekas and Robert Gallager. *Data Networks*. Prentice Hall, 1992.

[CS00] William S. Cleveland and Don X. Sun. Internet traffic data. *Journal of the American Statistical Association*, 2000.

[GDT03] Ullas Gargi, Yining Deng, and Daniel R. Tretter. Managing and searching personal photo collections. In *IS&T/SPIE Conference on Storage and Retrieval for Media Databases*, volume SPIE Vol. 5021, pages 13–21, January 2003.

[GGMPW02] Adrian Graham, Hector Garcia-Molina, Andreas Paepcke, and Terry Winograd. Time as essence for photo browsing through personal digital libraries. In *Proceedings of the Second ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 326–335, July 2002.

[GMM+03] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams: theory and practice. *Knowledge and Data Engineering, IEEE Transactions on*, 2003.

[LS00] Alexander C. Loui and Andreas E. Savakis. Automatic image event segmentation and quality screening for albuming applications. In *IEEE International Conference on Multimedia and Expo, New York City, New York*, July 2000.

[PCF02] John C. Platt, Mary Czerwinski, and Brent A. Field. Phototoc: Automatic clustering for browsing personal photographs. Technical Report 17, Microsoft Research, 1 Microsoft Way, Redmond, Washington, 2002.

[PJS92] Heinz-Otto Peitgen, Hartmut Jürgens, and Dietmar Saupe. *Chaos and Fractals: New Frontiers of Science*. Springer-Verlag, 1992.

[RTH00] G. Walther R. Tibshirani and T. Hastie. Estimating the number of clusters in a dataset via the gap statistic. Technical report, Stanford University, 2000.

[RW03] Kerry Rodden and K. Wood. How do people manage their digital photographs? In *ACM Conference on Human Factors in Computing Systems (ACM CHI)*, 2003.

[Spe01] Robert Spence. *Information Visualization*. ACM Press-Addison-Wesley, 2001.

[Str98] G. Strube. *Mind Modelling*, chapter Modelling Motivation and Action Control in Cognitive Systems, pages 89–108. Pabst, Berlin, 1998.