Organising a Daily Visual Diary Using Multi-Feature Clustering

Ciarán Ó Conaire[1,2], Noel E. O'Connor[1,2], Alan F. Smeaton[1,2], Gareth J. F. Jones[2]

[1]Adaptive Information Cluster, [2]Centre for Digital Video Processing, Dublin City University, Ireland.

ABSTRACT

The SenseCam is a prototype device from Microsoft that facilitates automatic capture of images of a person's life by integrating a colour camera, storage media and multiple sensors into a small wearable device. However, efficient search methods are required to reduce the user's burden of sifting through the thousands of images that are captured per day. In this paper, we describe experiments using colour spatiogram and block-based cross-correlation image features in conjunction with accelerometer sensor readings to cluster a day's worth of data into meaningful events, allowing the user to quickly browse a day's captured images. Two different low-complexity algorithms are detailed and evaluated for SenseCam image clustering.

Keywords: SenseCam, clustering, spatiogram, image similarity, fusion, accelerometer.

1. INTRODUCTION

The SenseCam is a Microsoft prototype wearable device that integrates a colour camera, storage media and multiple sensors in order to facilitate automatically capturing images of a person's life and activities. On an average day, the device will generate a few thousand images. The very real challenge as a result is *how to deal with the increased volume of media for retrieval, browsing and organising*.[1]

In this paper, we present two different approaches to grouping these captured images into events, thereby allowing a user to perform efficient browsing and searching through the large amount of image records of their own activities. Our first method is a Bayesian classification approach that uses the Viterbi algorithm to smooth the classification results and produce coherent image clusters of a person's activities. Our second approach is to use Statistical Region Merging, an algorithm for image segmentation, originally proposed to group pixels into regions, and to use it instead to group images into clusters. Both algorithms have low time complexity and therefore scale very well to larger data-sets.

This paper is organised as follows: In section 2, we review related research in managing and organising large image collections. In section 3 we describe the features we extract from the SenseCam data. Sections 4 and 5 describe our two approaches to the grouping of images into activity clusters. Section 6 shows our experimental results. We summarise the paper and give our conclusions in section 7.

2. RELATED RESEARCH

2.1. SenseCam

The SenseCam is a wearable image-capture device that automatically records images, roughly every 30 seconds, creating a digital picture diary of the wearer's day. The SenseCam also includes other sensors such as accelerometers, a passive infrared sensor and a light meter that take readings every second. These sensors are used to trigger the image capture when they determine that something *interesting* is happening; when the wearer enters a different room, for example. In a full day, the SenseCam will capture and store approximately $3,000$ images. Therefore, the research challenge is to manage this resource of images and make it searchable.

The SenseCam has been incorporated into the larger MyLifeBits[2] project, whose goal is to manage a lifetime's worth of media, including documents, images, music, emails and webpages. This was inspired by the *Memex* vision of Vannevar Bush[3] in his 1945 article.

Send correspondence to Ciarán Ó Conaire. E-mail: oconaire@eeng.dcu.ie

(a) 

(b) 

(c) 

Figure 1. (a)The SenseCam device and (b)(c) some examples of the images it captures

More details of the SenseCam can be found in.[1] We have a later model of the device, shown in figure 1, which is smaller in size, includes a larger 1GB memory and has a USB interface for charging the built-in battery and downloading images and data. Additionally, the device includes 3 accelerometers, not 2, so it can measure motion in three dimensions, and a button that allows the user to manually trigger image capture. The SenseCam we use does not support audio capture and does not have GPS. Later versions of SenseCam may support these technologies.

2.2. Image Collection Management

O'Hare et al.[4] recognise the *huge increase in the quantity of digital photos being taken* and that with it comes the challenge of organising these large collections in order to allow efficient image retrieval. They describe the MediAssist photo management system and demonstrate the benefits of combining contextual features (GPS location of photo capture) with content-based image features (MPEG-7 visual features[5]) for example-based image retrieval.

In a typical user's photo collection, it is possible to exploit the *bursty* nature of photo capture to detect activities (such as a birthday party) and cluster images appropriately.[6,7] This method is not suitable for SenseCam images since they are captured continuously and not in short bursts.

3. FEATURE EXTRACTION

This section details the three features that we extracted from the SenseCam data for our clustering experiments. The features consist of two image-based features (block-based cross-correlation and colour spatiogram) and one sensor-based feature (accelerometer motion).

3.1. Block-based Cross-Correlation

In order to detect if the SenseCam was in a fixed position (e.g. left on a desk or a shelf), we use a normalised cross-correlation (NCC) feature as a sensitive measure for detecting any slight camera movement. NCC was previously used in stereo-vision applications as an illumination-invariant matching measure for disparity estimation.[8] It can be thought of simply as a measure of edge agreement. We compare consecutive frames by computing the NCC between each non-overlapping 8×8 block of pixels. Since we use colour images, we concatenate the RGB values from each 64 pixel block into a vector of length 192 (64×3). We take the median value of all block comparisons which gives a single comparison value between all pairs of consecutive frames. Generally, the value is high (≥ 0.9) for frames where the camera is completely static, even in the presence of significant lighting variation and object

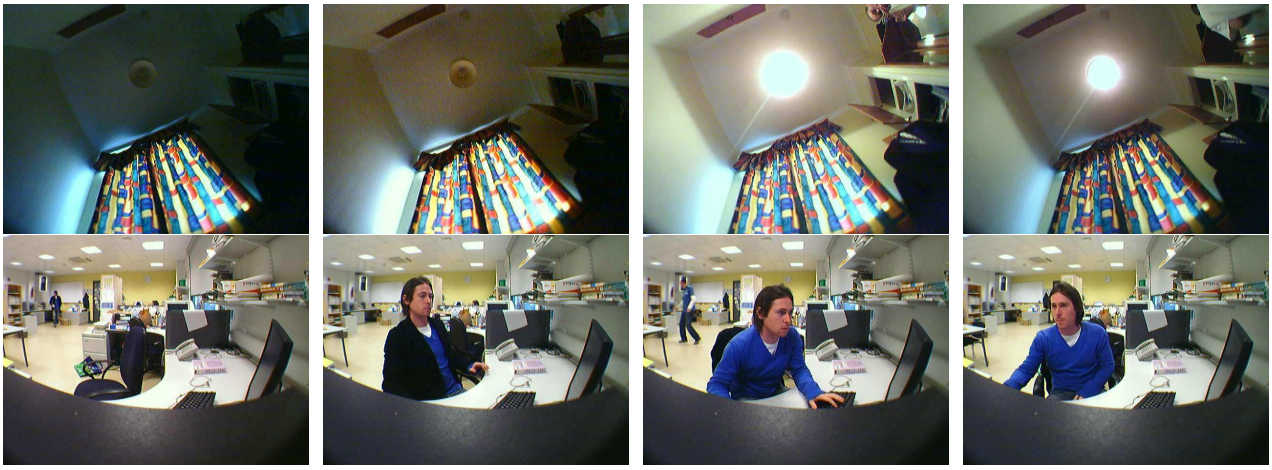


Figure 2. Two examples of typical static-camera situations which are robustly detected using the cross-correlation feature, despite significant lighting changes and object movement.

motion. To compute the NCC score between X and Y, two 8×8 pixel blocks, represented by vectors of length 192, the following equation is used:

$$\frac{\sum_{i=1}^{192}(X-\bar{X})(Y-\bar{Y})}{\sqrt{\left(\sum_{i=1}^{192}(X-\bar{X})^2\right)\left(\sum_{i=1}^{192}(Y-\bar{Y})^2\right)}} \tag{1}$$

The main use of the NCC feature is in distinguishing between image-pairs where the camera is completely static and images-pairs that are taken while the SenseCam is being worn by the user without much movement. Figure 2 shows examples of images from two static camera scenarios.

3.2. Accelerometer Features

In order to determine when to capture images, the SenseCam is equipped with a myriad of sensors, including a temperature sensor, a light-meter and three accelerometers. Unlike images, which are captured approximately every 30 seconds, sensor readings are stored approximately every few seconds, The three accelerometer sensors measure acceleration in three orthogonal directions: X, Y, Z. We combine these three readings to measure how much the SenseCam is moving. Given accelerometer data from one sensor, $\{A_1, A_2, ..., A_N\}$, we first compute its derivative:

$$\hat{A}_i = \quad 0 \quad (when\ i = 1) \tag{2}$$

$$= \quad A_i - A_{i-1} \quad (otherwise) \tag{3}$$

We combine the three sensors using:

$$M_i = \sqrt{\hat{X}_i^2 + \hat{Y}_i^2 + \hat{Z}_i^2} \tag{4}$$

Finally, we smooth the data using a median filter with a window-size of 17 samples. Since sensor data and image data are not always captured simultaneously, we compute the motion values associated with an image using a Gaussian window centred at the image capture time.

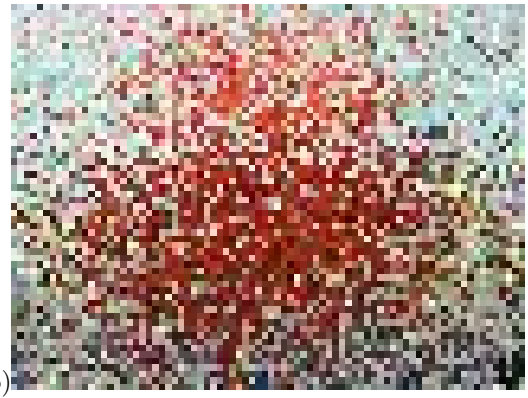
(a)  (b) 

Figure 3. (a)Original image, (b)Approximation of left image generated by selecting pixels from the distribution of its 8×8×8 bin spatiogram

3.3. Spatiograms

Colour histograms are commonly used in image retrieval systems to measure image similarity[9–11] and are a primary MPEG-7 feature for content description.[5] In,[12] the concept of a histogram is generalised into what is termed a *spatiogram*, combining the distribution information of a histogram with spatial moment information. In the paper, 2^{nd}-order spatiograms are used for head-tracking and to our knowledge, spatiograms have not previously been used in image retrieval. Spatiograms are very well suited to SenseCam images, as a person will often remain in a relatively fixed position while performing a task (e.g. sitting at a computer, speaking to someone, sitting in a meeting, eating).

A 2^{nd}-order spatiogram is defined by B bins, each having three parameters: n_b, its normalised histogram count, μ_b, the bin's spatial mean and Σ_b, the bin's diagonal covariance matrix, for $b = 1..B$. Given a region containing N pixels, these parameters are computed as follows:

$$n_b = \frac{1}{N}\sum_{i=1}^{N}\delta_{ib} \tag{5}$$

$$\mu_b = \frac{1}{\sum_{j=1}^{N}\delta_{jb}}\sum_{i=1}^{N}x_i\delta_{ib} \tag{6}$$

$$\Sigma_b = \frac{1}{\sum_{j=1}^{N}\delta_{jb}}\sum_{i=1}^{N}(x_i-\mu_b)(x_i-\mu_b)^T\delta_{ib} \tag{7}$$

where $\delta_{ib} = 1$ if the i^{th} pixel falls in the b^{th} bin and $\delta_{ib} = 0$ otherwise, $x_i = [\mathtt{x},\ \mathtt{y}]^T$ is the spatial position of the pixel, mapped so that it varies between -1 to 1. For all experiments, we used 8 bins per channel, giving a total of 512 colour bins. Images were dithered before quantisation into bins by adding uniformly distributed random noise, which allows finer colour shades to be distinguished, as well as allowing a small degree of lighting-change robustness. In figure 3, the information contained in a spatiogram is illustrated by approximating the original image by sampling from the spatial-colour distribution of its spatiogram. As shown, the spatial information contained in a spatiogram is quite coarse. Calculating a distance measure between two spatiograms is discussed in the next subsection.

3.3.1. Comparing Spatiograms

We found the spatiogram similarity measure used in previous work unsuitable for image matching and therefore introduce a new similarity metric that has a closer relationship to probability-based measures. To compare

two spatiograms, given by (n_b, μ_b, Σ_b) and $(n_b^{'}, \mu_b^{'}, \Sigma_b^{'})$, the following similarity measure was used in the original work:[12]

$$\rho = \sum_{b=1}^{B} N(\mu_b; \mu_b^{'}, \hat{\Sigma}_b)\sqrt{n_b n_b^{'}} \tag{8}$$

where $N(x; \mu, \Sigma)$ is a normalised Gaussian given by:

$$N(x; \mu, \Sigma) = \frac{1}{2\pi|\Sigma|^{1/2}} exp\left\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right\} \tag{9}$$

and $\hat{\Sigma}_b^{-1} = (\Sigma_b^{-1} + (\Sigma_b^{'})^{-1})$, so that the distance between the spatial means is normalised to the average of the two Mahalanobis distances.

We have derived an improved similarity metric, by first converting the 2^{nd} order spatiogram back to a histogram, but adding an extra dimension of space. We then compare spatiograms using the Bhattacharyya coefficient, which is related to the probability of classification error, and therefore is more similar to a probability than equation (8). Given two spatiograms, we propose to measure their similarity using:

$$\rho = \sum_{b=1}^{B} \sqrt{n_b n_b^{'}}\left[8\pi|\Sigma_b\Sigma_b^{'}|^{1/4} N(\mu_b; \mu_b^{'}, 2(\Sigma_b + \Sigma_b^{'}))\right] \tag{10}$$

Comparing equations 8 and 10, we see that our new similarity measure is more tolerant to spatial movement of colours, since it allows greater variance. We also verified this experimentally, as shown in figure 4. We compared a target image region with other overlapping regions by shifting the region left and right by 20%. Note that the previously used similarity measure is normalised so that its maximum value is one. As shown in the graph, the previously used similarity measure is intolerant of small spatial changes, where a shift of only a few pixels causes the measure to report a very significant difference between the target and the shifted region. A histogram-based comparison is completely tolerant of spatial changes, since it stores no spatial information. Our new measure achieves a balance, as it is tolerant of spatial changes but not oblivious to them.

4. CLUSTERING VIA IMAGE CLASSIFICATION

Our first method for grouping the SenseCam images is to use a Bayesian approach to classify each image into one of three classes: Static Person (SP), Moving Person (MP) or Static Camera (SC). For each of the three classes, we use an annotated training set to compute the distribution of each of the three features (Spatiogram similarity, NCC and motion). Assuming equal priors for all classes, and independance of the features given the target class, we can compute the probability of belonging to each class for each test image using Bayes rule. If C is the class of interest and (f_1, f_2, f_3) are the features used, feature independence is expressed as:

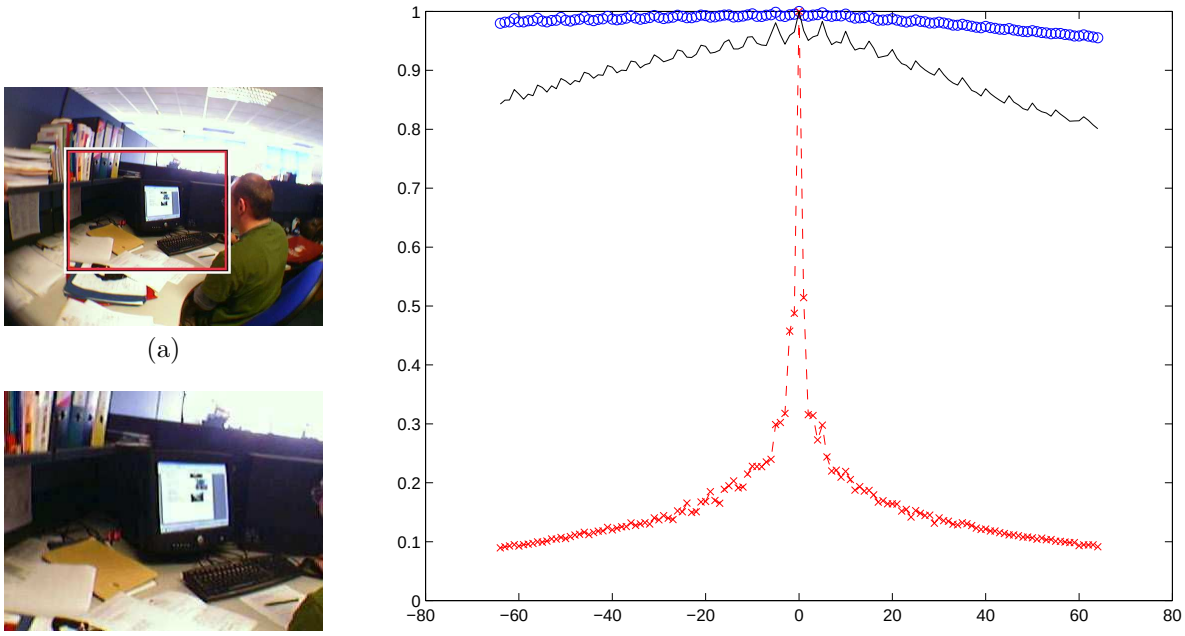
$$P(f_1, f_2, f_3|C) = P(f_1|C)P(f_2|C)P(f_3|C) \tag{11}$$

and Bayes' rule is given by:

$$P(C|f) = \frac{P(f|C)P(C)}{P(f)} \tag{12}$$

Values of $P(f|C)$ and $P(f)$ are computed from histograms of our training data, smoothed with a narrow Gaussian filter, $P(C) = 1/3$ for all classes $C \in \{SP, MP, SC\}$.

A simple approach would be to associate each image with the class which has the highest probability. However, this can produce outliers. For example, during a three hour period, while a person is working at their desk, they may walk to the printer to get some paper. The motion data from this incident could be detected as a MP event, instead of a SP event. For certain applications, such detailed classification could be useful. However, for assisting user-browsing, a smaller number of clusters reduces the ultimate burden on the user. We use the Viterbi algorithm[13] to smooth the transitions between each of the three classes. For all our experiments, the transition probability between classes, $P_{trans} = 10^{-12}$.



(a) (b) (c)

Figure 4. (a)Test image with target section highlighted, (b)Close-up of target section, (c)Similarity scores between target patch and similar patch moved horizontally from target position ±20%: Histogram similarity using Bhattacharya coefficient (blue circles), Previous spatiogram similarity measure (dashed-red), Our new measure (solid black).

5. CLUSTERING VIA STATISTICAL IMAGE GROUPING

The second method we use for grouping SenseCam images is based on the Statistical Region Merging algorithm of Nock and Nielsen.[14] The algorithm is designed to group pixels into regions and was proposed for fast image segmentation. We adapt their algorithm to group images (not pixels) into clusters. In the same way that the original algorithm exploits local spatial connections, we use it to exploit the temporal connection between consecutive images captured by the SenseCam. At present, we only use the spatiogram features in this approach.

5.1. Algorithm

The input to the algorithm is a sorted list $L = \{(a_1, b_1, c_1), (a_2, b_2, c_2), ..., (a_N, b_N, c_N)\}$, which is a list of *links* between neighbouring nodes (images), where c_i is the cost of the link between nodes a_i and b_i. The list is sorted in order of ascending cost. Initially, there are J image clusters, where J is the number of images, each cluster containing exactly one image. The algorithm parses the list only once, one item at a time, and never resorts the list. When parsing item (a_i, b_i, c_i), if image a_i and b_i are already in the same cluster, then it is ignored. Otherwise, the merging cost of the clusters A and B, containing a_i and b_i respectively, is computed. The two clusters are merged if the cost is below a threshold $T(A, B)$, given by:

$$T = \sqrt{R(A)^2 + R(B)^2} \tag{13}$$

with $R(x)$ given by:

$$R(x) = \sqrt{\frac{log(|x|) - log(P)}{2Q|x|}} \tag{14}$$

P and Q are parameters of the algorithm and $|x|$ denotes the number of images in cluster x. We set $P = Z^{-2}$, as was done in,[14] where Z is the total number of images. The Q parameter is used to decide the scale at which clustering is done. A larger value of Q reduces the threshold and therefore makes it more difficult for clusters

to merge, creating a fine-scale clustering with many clusters. Smaller Q values produce large clusters. For our purposes, we compare images only to their two immediate neighbours on either side temporally. We set the cost of the links equal (and the cost of merging clusters) to $\sqrt{1-\rho^2}$, where ρ is the spatiogram similarity score between the images (or clusters).

5.2. Spatiogram model updating

In this subsection, we detail a procedure to determine the spatiogram model that should represent an image cluster after a merging. Given that we have the current spatiogram model of both clusters being merged, we wish to create a new spatiogram model to represent the merged cluster. The update procedure simply involves adding the bin-counts and moments of the two spatiograms. First we convert from spatial means and variances of each spatiogram to moment sums:

$$m_{(x),b} = \mu_{(x),b} n_b \tag{15}$$
$$m_{(y),b} = \mu_{(y),b} n_b \tag{16}$$
$$s_{(x),b} = (\Sigma_{(x),b} + \mu^2_{(x),b}) n_b \tag{17}$$
$$s_{(y),b} = (\Sigma_{(y),b} + \mu^2_{(y),b}) n_b \tag{18}$$

where $m_{(x),b}$ and $m_{(y),b}$ are the first-order moment sums of bin b in the x and y directions, $s_{(x),b}$ and $s_{(y),b}$ are the second-order moment sums, n_b, μ_b and Σ_b are the spatiogram parameters. After both spatiograms have been converted to moments sums, the updated model is computed as a weighted sum of the moments:

$$n_b^{'} = \beta n_b^{(A)} + (1-\beta) n_b^{(B)} \tag{19}$$
$$m_b^{'} = \beta m_b^{(A)} + (1-\beta) m_b^{(B)} \tag{20}$$
$$s_b^{'} = \beta s_b^{(A)} + (1-\beta) s_b^{(B)} \tag{21}$$

where the superscript refers to the two clusters being merged (A and B). To weight the clusters according to their size (number of images each contains) we set $\beta = \frac{|A|}{|A|+|B|}$. The moment sums are then converted back to spatiogram parameters to obtain the updated spatiogram:

$$\mu_{(x),b}^{'} = \frac{m_{(x),b}}{n_b^{'}} \tag{22}$$
$$\mu_{(y),b}^{'} = \frac{m_{(y),b}}{n_b^{'}} \tag{23}$$
$$\Sigma_{(x),b}^{'} = \frac{s_{(x),b}}{n_b^{'}} - \left(\frac{m_{(x),b}}{n_b^{'}}\right)^2 \tag{24}$$
$$\Sigma_{(y),b}^{'} = \frac{s_{(y),b}}{n_b^{'}} - \left(\frac{m_{(y),b}}{n_b^{'}}\right)^2 \tag{25}$$

6. RESULTS

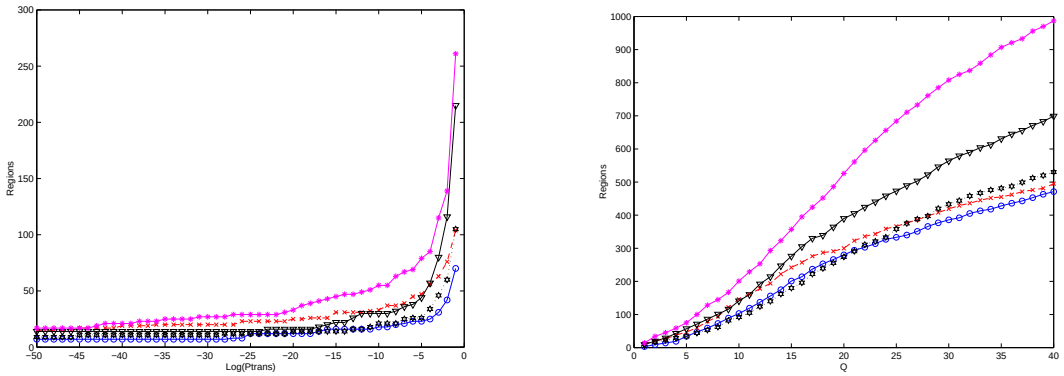
In this section, we first examine the effect of varying the parameter values of each method on the number of image clusters that are generated. Next, using SenseCam data from two different users, we perform clustering experiments using our two methods and demonstrate their effectiveness for grouping images into semantic activities.

6.1. Clustering Scale

Both methods we evaluated include a parameter to control the scale at which image clustering takes place. This is a useful feature as it can provide a user with a multiscale representation of their activities. We examined the effect of varying these parameters on the number of clusters generated by the algorithms. Results are shown

in figure 5. Clearly, by increasing P_{trans} or Q, we obtain a larger number of clusters, resulting in a finer scale clustering.

Since both algorithms have low time complexity, they can be efficiently executed multiple times with different parameters. This can be used to select the most appropriate scale, using a measure of some desired criteria. A more interactive system could first present the user with a small number of image clusters. When a cluster is selected, the algorithm is executed again at a higher resolution, allowing the system to present users with a more detailed break-down of their activities.



(a) (b)

Figure 5. (a)Log(P_{trans}) versus Number of clusters, (b)Q versus Number of clusters. The plots shown are of five separate days from 3 users.

6.2. Classification results

In figure 6, we show the result of clustering 719 SenseCam images from a single day. For the SP and SC clusters, representative images were selected by computing an estimate of the median spatiogram of all images in the cluster and selecting the best match. For the MP clusters, 4 representative images were selected by sampling uniformly from the cluster. No static camera classes were detected during this day (none were present). Figure 7 shows another set of results from clustering 1915 SenseCam images from a different user. Representative images are generated in the same way. Two clusters with less than 5 images are not shown. One SC cluster was detected when the user left the camera down to go to the bathroom.

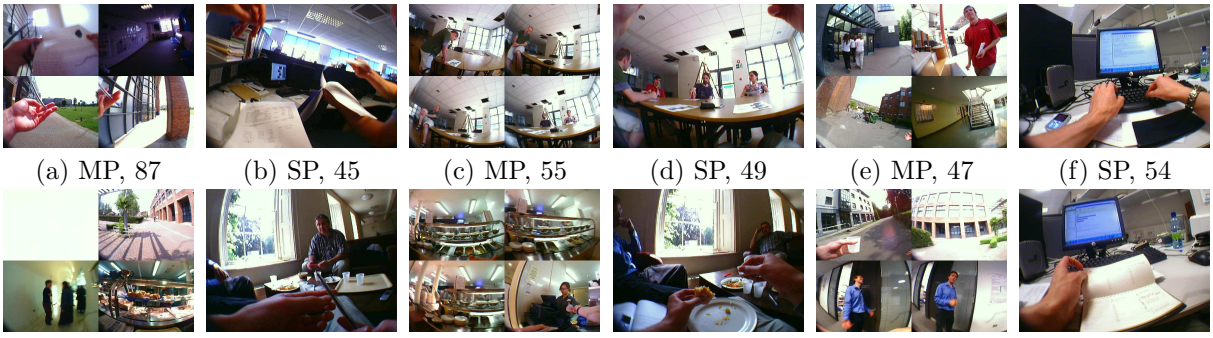
The results corresponded well to the users' actual activities, such as eating lunch/breakfast, working at a computer, watching television and sitting in a meeting. We noticed that SP clusters are almost always separated by MP clusters, corresponding to a user moving between locations for activities. The images selected for 6(c) are not completely appropriate, as the movement from one building to another was quite brief. To improve this, perhaps the motion data could be used to select images where the movement is taking place.

Another interesting possibility for future work would be to use the SenseCam as a smart surveillance camera in the SC clusters, since it captures images when people are present. Background modelling techniques[15] could be used to segment objects in the scene and thus facilitate more advanced content-based retrieval of the SenseCam images.

6.3. Clustering results

Using the same two data-sets as used in the classification experiments, we performed image clustering using the algorithm described in section 5. Results are shown in figures 8 and 9. We used a value of $Q = 2$ for the first dataset and $Q = 1$ for the second, in order to generate roughly the same number of clusters.

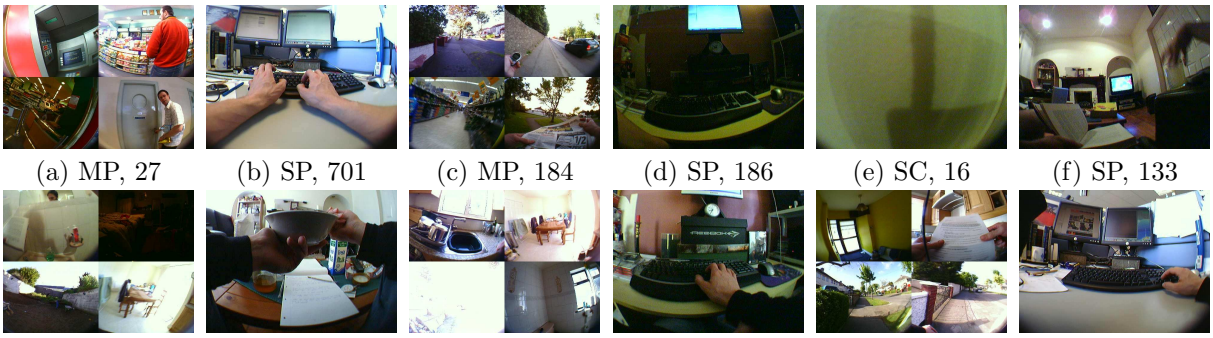
Comparing this approach to the Bayesian classification, the statistical image clustering method groups image based on appearance alone and therefore treats images captured during movement as outliers, since they are not



(a) MP, 87 (b) SP, 45 (c) MP, 55 (d) SP, 49 (e) MP, 47 (f) SP, 54

(g) MP, 36 (h) SP, 185 (i) MP, 15 (j) SP, 67 (k) MP, 45 (l) SP, 34

Figure 6. Results of Bayesian classification clustering, with associated cluster type and number of images



(a) MP, 27 (b) SP, 701 (c) MP, 184 (d) SP, 186 (e) SC, 16 (f) SP, 133

(g) MP, 23 (h) SP, 55 (i) MP, 26 (j) SP, 90 (k) MP, 33 (l) SP, 434

Figure 7. Results of Bayesian classification clustering, with associated cluster type and number of images

similar to each other. These images are grouped last and usually merged to a larger static activity cluster. Figure 9(b) is an exception to this because there was a large number of images where the user was moving. Figure 8(i) shows a small cluster of images from when the SenseCam lens was obscured by clothing. Although it is small, this cluster was not merged with another since it spatiogram was significantly different to all surrounding clusters.

Overall, both methods performed well, grouping images in a way that was very similar to the users own semantic interpretation of their activities.

7. CONCLUSIONS

In this paper, we presented and evaluated two low-complexity algorithms for efficient clustering of SenseCam images: (i) a Bayesian classification approach combined with Viterbi smoothing and (ii) a clustering approached inspired by the Statistical Region Merging image segmentation algorithm. We demonstrated the use of spatiograms as useful features for image clustering, and derived, from the Bhattacharya coefficient, a spatiogram similarity measure that is superior to the originally proposed measure. We showed that the statistical region merging algorithm for single-image segmentation can be successfully used for multi-image clustering. For both approaches, we examined how parameter changes affect the level of detail or *granularity* at which images are clustered and discussed briefly how to incorporate scale selection into a browsing system.

Future work will focus on whether the two approaches described in this paper could be fused or implemented in a hierarchical structure that would allow browsing at various levels of granularity. Additionally, more SenseCam features, both sensor and image-based, may also assist the automatic semantic classification of a user's activities. For example, the light-meter, combined with the time of day may be used to detect whether the wearer is

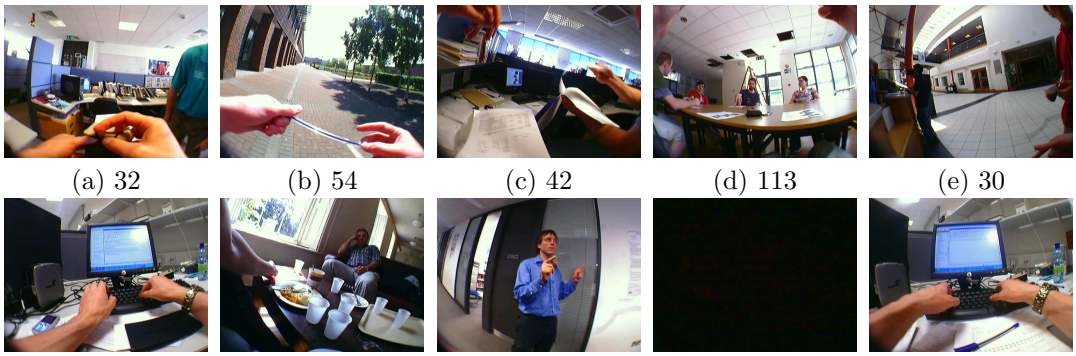
indoors or outdoors. Additionally, we hope to develop methods to make spatiogram comparison more robust to the frequent strong lighting changes present in typical SenseCam images.

Acknowledgments

This material is based on works supported by **Science Foundation Ireland** under Grant No. 03/IN.3/I361 and sponsored by a scholarship from the **Irish Research Council for Science, Engineering and Technology** (IRCSET): Funded by the National Development Plan. The authors would also like to express their gratitude to **Microsoft Research Cambridge** for their contribution to this work.

REFERENCES

1. J. Gemmell, L. Williams, K. Wood, R. Lueder, and G. Bell, "Passive capture and ensuing issues fort a personal lifetime store," in *CARPE 2004, New York, NY, USA*, October 2004.
2. J. Gemmell, G. Bell, R. Lueder, S. Drucker, and C. Wong, "Mylifebits: Fulfilling the memex vision," in *Proceedings of ACM Multimedia*, pp. 235–238, December 2002.
3. V. Bush, "As we may think," *The Atlantic Monthly* , July 1945.
4. N. O'Hare, C. Gurrin, G. J. F. Jones, and A. Smeaton, "Combination of content analysis and context features for digital photograph retrieval," in *2nd IEE European Workshop on the Integration of Knowledge, Semantic and Digital Media Technologies*, November 2005.
5. B. S. Manjunath, J.-R. Ohm, V. V. Vasudevan, and A. Yamada, "Color and texture descriptors," *IEEE Transactions on Circuits and Systems for Video Technology* **11**, pp. 703–714, June 2001.
6. A. Graham, H. Garcia-Molina, A. Paepcke, and T. Winograd, "Time as essence for photo browsing through personal digital libraries," in *ACM Joint Conference on Digital Libraries*, July 2002.
7. M. Cooper, J. Foote, and A. Girgensohn, "Automatically organising digital photographs using time and content," in *IEEE International Conference on Image Processing (ICIP)*, Sept 2003.
8. M. Lhuillier and L. Quan, "Robust dense matching using local and global geometric constraints," in *Proc. 15th Int'l Conf. Pattern Recognition*, **1**, pp. 968–972, 2000.
9. Y. Rui, T.S. Huang, and S. Chang, "Image retrieval: Current techniques, promising directions and open issues," *Journal of Visual Communication and Image Representation* , pp. 39–62, April 1999.
10. A. Smeulders, M. Worring, S. Santini, A. Gupta and R. Jain, "Content based image retrieval at the end of the early years," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**, pp. 1349–1380, December 2000.
11. P.C. Aigrain, H.C. Zhang, D.C. Petkovic, "Content-based representation and retrieval of visual media - a state-of-the-art review," *Multimedia Tools and Applications* **3**, pp. 179–202, March 1996.
12. S. T. Birchfield and S. Rangarajan, "Spatiograms versus histograms for region-based tracking," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1158–1163, June 2005.
13. G. D. Forney, "The viterbi algorithm," *Proceedings of the IEEE* **61**, pp. 268–278, March 1973.
14. R. Nock and F. Nielsen, "Statistical region merging," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **26**, pp. 1452–1458, Nov 2004.
15. A. M. McIvor, "Background subtraction techniques," in *Image and Vision Computing, Hamilton, New Zealand*, Nov 2000.



(a) 32 (b) 54 (c) 42 (d) 113 (e) 30

(f) 75 (g) 300 (h) 38 (i) 15 (j) 20

Figure 8. Results of statistical image clustering with associated number of images

(a) 731 (b) 165 (c) 206 (d) 36 (e) 67

(f) 52 (g) 112 (h) 82 (i) 464

Figure 9. Results of statistical image clustering with associated number of images