"Detecting Unusual Activity in Video", by H. Zhong, J. Shi, and M. Visontai.

This paper describes an interesting application of normalized cuts to document clustering. Unlike latent semantic indexing, this algorithm learns which words are relevant for clustering and exploits previously-learned similarity between related words. Then this algorithm is applied to detect unusual events in video sequences, where the "documents" are short video segments and the "words" are single-frame motion-based features.

The co-occurrance trick used to clustering both features and segments is very nice and seems like it should have broad applications; any algorithm currently using latent semantic indexing might benefit from this approach. The authors point out briefly that it is actually normalized cuts being applied in a new way, which makes intuitive sense since they create a graph where weights correspond to some measure of similarity and then proceed to cluster it. Glancing over the "Normalized Cuts for Image Segmentation" paper, I see that: 1. that paper hinted at this application of Normalized Cuts, and 2. Jianbo Shi is an author of both papers. So it's not surprising that "Detecting Unusual Activity in Video" used normalized cuts, but it is surprising that they derived it from a completely different direction, which is a valuable contribution.

The unusual event detection isn't that interesting once the video segments are clustered; it simply involves finding clusters which are dissimilar from most of the others. Some of the weaknesses of the unusual-event-detection algorithm:

- won't work if the camera is in motion, although it should be possible to work around this using a better motion detector
- insensitive to the direction of motion; again that might be possible to remedy using a better motion detector and richer feature descriptor
- vulnerable to motion noise -- for example any action taking place in front of a fast-moving train or waving trees would be obscured and ignored
- insensitive to small-scale features, so would fail to recognize:
  - a ninja moving very slowly
  - someone wearing good camoflouge
  - a man holding a gun while walking normally
  - two men exchanging a briefcase
  - in short, any event designed to look normal to a very casual observer
- possible to "game" by staging unusual events until they are considered usual
- requires a very large body of training data to be useful (enough so that every "usual" event (at a coarse scale) is represented many times)
- not clear if it can be run online for real-time event detection

These weaknesses would seem to make this algorithm ineffective for high-security scenarios where attackers are expected to actively attempt to undermine surveillance (despite the poker cheating example, this would include casinos, where cheaters probably try to be less obvious), but fine for low-security public places where you mainly want to be

notified if there is an accident or emergency.

I'm curious how well this would work for real-time detection since it doesn't seem too hard to extend it for that case. The detector would have to be trained on a large body of data including the entire range of "usual" scenarios, and then in near-real-time new segments could be created and features extracted based on the previously-learned feature set. If any frame doesn't match any existing feature prototype within a certain distance, then the segment is unusual (there's a chance that their offline algorithm would identify this as a low-frequency random event and ignore it, but that would be hard to do online). Otherwise, the segment is compared to learned clusters and recognized as unusual as in the offline algorithm.

Vision-Based Global Localization and Mapping for Mobile Robots, by S. Se, D. Lowe, and J. Little.

This paper describes a purely vision-based system for SLAM (simultaneous localization and mapping). Their approach uses a combination of appearance and 3D structure to efficiently and robustly localize the robot.

Their algorithm begins by extracting SIFT features. This seems to be a popular approach for place recognition because SIFT features are fairly invariant to viewpoint and illumination changes but also distinctive. The robot has a trinocular vision system so using epipolar constraints together with SIFT scale and orientation, features are matched between images taken at the same time, giving a 3D ego-centric location for each feature. This is easier than general Structure From Motion because the relationship between the cameras is known. The authors don't explicitly account for any uncertainty in this process, which I guess is because it's fairly reliable and the uncertainty is considered later when matching features between different frames.

As the robot moves, odometry is used to predict correspondences between features across frames, and then these correspondences are used to refine the robot's position using least-squares optimization. It's unclear whether SIFT also imposes constraints on feature correspondences at this stage, but I guess it must.

If the robot can't find good correspondences, it assumes that it has been kidnapped and tries to localize itself globally using RANSAC to find spatially-consistent matches with known points in the map. A Hough transform approach was also evaluated, but performs worse than RANSAC if the features are relatively distinctive. This makes sense, as the Hough transform wastes time voting on every correspondence no matter how spatially improbable, while RANSAC eliminates most spatially-inconsistent correspondences fairly quickly.

The most interesting aspect of the paper is the use of submaps. To avoid allowing small errors in estimation to accumulate over time, the robot represents the map as a collection of submaps, each of which has very good spatial consistency. To build the global map, all the submaps are aligned using a weighted least-squares approach which gives higher weight to the more spatially-consistent features.

It seems like this approach will not scale well to larger-scale maps for several reasons. First, SIFT-based stereo depth perception works much better indoors than out, where the camera baseline is much smaller relative to the distances to typical objects and there is likely to be more movement (cars) and repetitive features (bricks, trees, windows). So for outdoors it seems better not to rely on 3D information too much. Second, as the number of submaps grows, storing, matching, and correcting them becomes much harder. They have not extended global localization and map alignment to 3D yet, but that seems like it would make the scaling problem even more difficult. To solve this it makes sense to use a purely topological approach like Kuipers', or maybe there is some way to hierarchically cluster and merge submaps after some point to avoid unnecessary redundant information.

The paper, "Vision-Based Global Localization and Mapping for Mobile Robots", presents a set of methods to allow a robot to map an area and localize itself within that map. Using a 'trinocular' camera system and range finders, the system obtains spatial information on the environment and then picks out salient features using SIFT. The robot builds a local map and tracks local features using a Kalman filter as is navigates locally (according to "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks"). Eventually, error accumulates and the system starts a new local map. Several local maps are aligned using RANSAC or the Hough transform.

According to the paper, its main contributions are (1) using SIFT features, (2) using the Hough transform and RANSAC for aligning submaps, and (3) using backward correction with uncertainty for loop closure. In my opinion, the comparison of the two alignment methods is interesting, and the concept of building up chunks that can then behave as "super-features" is very useful.

This paper has a good literature review that acknowledges a large body of related work and several different approaches. The ideas themselves are also very interesting. I think that, perhaps, the paper goes a bit overboard in emphasizing the usefulness of their "highly distinctive visual landmarks". Section 3a, ends with the statement, "Recently a performance evaluation of various local descriptors showed that SIFT feature descriptors perform best among them." Such a claim, even with the accompanying reference to another paper, definitely needs to explain a little more about what it means

to "perform best". What is the optimality metric? Using the SIFT detector and features myself, I have found that the features are not always as robust as advertised.

The only quantitative comparison in the paper relates the computational complexity of Hough and RANSAC for mapping. This makes a nice graph, but the result is rather intuitive. The Hough transform, as typically implemented, takes about the same length of time to execute on clean or noisy data. RANSAC, on the other hand, only needs one iteration if there are no outliers. The Hough transform could be adapted to start looking at the bins with the most votes and if a single bin accumulates a disproportionately large number of votes early on, the algorithm could terminate early or simply verify with the remaining points, rather than cast additional votes. The interesting experimental data, however, is in the accuracy of the map. The results section provides a few tables that give a few example estimates, but there is nothing to compare them too. I want to know the accuracy of the mapping when compared to the other mapping approaches describe in the related work. Without that comparison, it is difficult to evaluate the value of the approach compared to prior art.

As mentioned in the paper, this mapping method (and most others) assumes a two-dimensional world and could benefit from a three dimensional extension. In rescue robotics, the robotic may have to navigate a hazardous and three dimensional course. Another extension would be to further discretize map construction so that one routine gathers the immediately local map, another routine pieces together several local maps into a second stage, still another routine groups second stage maps into a third stage, and so on. I felt like the dichotomy of local and global localization was forced. It ought to be more of a spectrum. Yet another extension is to use probabilistic borders instead of thresholds so that every match can be only a partial match. For example, once features are matched and given to RANSAC, the RANSAC algorithm has no way of accounting for the quality of those matches. The matches that were really close should carry more weight than those matches that just barely squeaked in under the threshold.

I had a couple questions about the paper. Section 4b mentions that the Hough voting scheme uses co-variance of features, but I didn't understand where that co-variance comes from. I didn't quite understand how the algorithm determines exactly when it's time to start building a new submap. I also wish that the paper would explain its weaknesses. The approach will certainly do better under some circumstances. When?

This paper describes a SLAM algorithm using visual features for alignment. In particular, they use SIFT features to find corresponding landmarks in

the current view and landmarks previously stored. They compare the performance of using Hough transform and RANSAC in global localization. Finally, they use an algorithm that builds submaps and then aligns them globally to deal the loop closing problem.

The main advantage of their approach is using a robust, distinctive feature for calculating correspondence. SIFT features allow them to fairly accurately localize both locally and globally. Using submaps to align further strengthen the matching especially in the presence of a non-descript frame. This however, may require more investigation. It is not uncommon to encounter a blank wall in an indoor environment, or bushes and trees in the outdoor environment. In such circumstances, it may be hard to find suitable SIFT features to match.

The experiment consist of a typical SLAM scenario where the robot goes around a corridor. The the section on the speed performances, it would be preferable to also have a measure of the actual time required for the various algorithms instead of just a theoretical calculation and measurement for each step. Also, as the authors noted, it would be good to test the robot in a more extensive environment. Aligning four submaps may be fairly easy, but when the number of submaps grow to hundreds or thousands, performance may be poor. Also, more complicated loop closure scenarios need to be investigated.

Using distinctive visual features such as SIFT provides a good way to do global localization as landmarks can be detected fairly well. One extension would be to construct a full 3D map instead of just a 2D map. This would require estimating the full pose of the robot. This is important for example when the robot is going up a spiral ramp. The idea of using submap can also be extended to do periodic merging of submaps to keep the memory requirement low. If the robot goes outside into a city, it may be quite a while before the robot closes a loop. In the meantime, it may have accumulated a large number of submaps that need to be merged. Incrementally merging submaps as the number of submaps grow can help bound the cost of doing loop closure.

Another interesting problem to explore is the problem of dynamics in the environment. If the robot detects a person and records their SIFT features as landmarks, and later sees the same person in a different location, it may have a hard time localizing. One way to deal with this problem is to separate the static and dynamic parts of the frames.

Review of "A Binning Scheme for Fast Hard Drive Based Image Search",
Fraundorfer, Stewenius, Nister
Class date: February 22, 2008

This paper describes a technique for fast image search over a very large number of images – in particular, when 1,000,000 or more images need to be searched and the inverted structure that indexes these

images is too large to be stored in RAM. Current state-of-the-art techniques specific to image retrieval are not designed for such large numbers of images, and standard large-database techniques scale linearly in the number of images, leading to excessively long hard-disk retrieval times. The paper describes a scheme that can efficiently handle such large numbers of images and allows time vs. accuracy trade-offs to be made to achieve needed performance.

The main contribution of the paper is to describe a scheme for fast hard-drive search that uses multiple independent bins in a hashing-type scheme so as to reduce the number of bytes that must be read from a hard disk to find the correct image. The overall approach is based on the bag-of-words scheme described in Sivic and Zisserman's Video Google paper. Images are converted to vectors of counts over a vocabulary of "visual words" computed from quantized SIFT feature vectors.

The specific contribution of the paper is to create a scheme where a number of binnings are created for the image database, each of which classifies the total image set into one of a specific number of bins. The classification is based on associating a "prototype" image with each bin and finding the closest prototype for each image in the database. The prototypes are created by randomly choosing a set of 20 to 50 visual words and assembling them, along with the four closest neighbors of each chosen word, into a prototype vector. A set of such prototypes makes up a particular binning. Locating an image from the entire database proceeds by searching each binning in turn, at each step finding the closest bin and then searching the images in that bin to find the closest one. Then, of the resulting images found, the single closest one to the search image is the one returned.

The strength of this scheme is that, since the components of a prototype are chosen randomly, different sets of prototypes will be statistically independent of each other, and hence strong statistical guarantees can be made concerning the overall performance of the search. Unlike a traditional inverted-file scheme, this method is not deterministically guaranteed to find the single best image. However, because of the statistical bounds that can be made, the performance of the method can be made effectively just as good as a deterministic scheme by using a sufficient number of binnings. In particular, the independence of the binnings means that the probability of an error decreases exponentially with the number of binnings, and hence even a moderate number of binnings will yield very good performance. Meanwhile, the total amount of data needed to be processed is much less than that required for a deterministic scheme. Furthermore, by varying the number of binnings a trade-off can be made between speed and accuracy, increasing the versatility of the method in adapting to different needs – something that may be critical in many sorts of applications, such as time-limited web searching.

Overall, the paper is well written, and the theoretical performance is

clearly explained.
The experiments that are done show clearly that above a level of about 15 or 20 binnings, the performance of their method becomes indistinguishable from a deterministic inverted-file method. My main criticism is that the experiments are inadequate in showing the time performance of their algorithm, which is essentially the single critical strength that this method has over conventional database methods. For one thing, the description of their experiments gives no indication whatsoever of the time taken to perform them. Worse is that they show no experiments that quantify the accuracy-vs.-space tradeoff that is a fundamental part of their algorithm: Without such information, it is difficult to impossible for an application designer to judge whether to choose their method over others. Finally, the experiments were performed over much too small of an image set. The authors indicate that their method is particularly useful for image sets measured in the millions, and Table 1 of the paper, showing the disk access time of an inverted file scheme as a function of database size, shows clearly that disk overhead coming from reading additional data is insignificant until the database contains well over 10,000,000 images. Yet their experiments are done over databases of no more than about 100,000 images, making it impossible to say with certainty how their method performs in the database size range it is intended for.

"Learning Embeddings for Fast Approximate Nearest Neighbor Retrieval" by V Athitsos, J Alon, S Sclaroff, G Kollios.

This paper describes an improvement on previous embedding methods by applying the AdaBoost algorithm to find a good embedding. Briefly, each object in a high-dimensional space is represented by a vector of distances to a weighted set of prototype objects. The prototype objects are chosen and weighted by AdaBoost so that nearby objects in the high-dimensional space will map to nearby vectors.

Unlike binning or hashing, this method does not reduce the number of comparisons necessary to find nearby neighbors. So it's more directly comparable to the vector quantization method described in Video Google; it's still necessary to compare the query vector to every vector in the database. The hope is that computing vector distances is much cheaper than computing distances in object space. However there's no reason a hashing or tree scheme couldn't be applied on top of this method to reduce the number of vector comparisons needed. Also as the authors pointed out, this method can be easily extended to train in parameter space instead of object space, so it could be applied to tasks like pose estimation in place of LSH. And of course once this is done a technique like LWR can be used to approximate the query parameters.

One nice thing about this approach relative to other techniques for feature extraction is that it does not assume that features have been quantized (for binning) or that there is a family of hash functions

operating on features (for LSH); instead the embedding infers significant features implicitly from the training set. However this also means that the vector for each query object must be calculated by calculating the distance to each prototype object, which is potentially very expensive.

The biggest thing I felt was missing from the paper was a comparison of this method to feature quantization methods specialized for a domain, like vocabulary trees or LSH. BoostMap is clearly better than related mapping techniques like FastMap, which is expected because it learns a set of dimensions for a reasonable quality measure rather than choosing them randomly. However it's not clear how it compares to methods based on feature extractors like SIFT. The impression I get is that this method is likely to be both a great deal slower and less effective than methods specialized to a particular domain like SIFT for images. On the other hand, it is easier to apply it to more complex problems like searching for similar motions (as in the sign language example).

I'd also really like to understand better how this method is theoretically related to LSH. It seems like since they're both "learning" a set of dimensions there is some underlying principle they share and it should be possible to compare them directly.

"Shape Matching and Object Recognition Using Shape Contexts" presents a method for recognizing objects by matching shape descriptors. Shape descriptors, calculated by sampling from edge points of an object and then forming a histogram of the angles and log of the distance to other points, can be matched to similar descriptors in another image. The set of matched points provide for the calculation of a distance function between the two images by accounting for bending energy and feature matching. With a way to measure distance, objects (shapes) belonging to a specific class can be clustered into a small set of prototypes that can classify a new image with KNN approach.

The main contribution of this paper is the introduction of the shape context recognition that serves as a good descriptor for shapes. The general approach to shape matching and recognition is also neat and well-described. Their discussion of properties and invariances of the detector/descriptor is also useful.

The paper has several strengths: it provides compelling results, informative figures, and a fairly clear explanation of how the algorithm works. A few details are swept under the rug though. The paper lacks much insight into the selection of bin-size for their histograms. Figure 3 mentions some examples, but there doesn't seem to be much in the actual body of the paper. Section 3.3 discusses outlier detection and a process for eliminating the effect of outliers. However, I couldn't figure out how that occurs. I really like the use of figures and graphs in the paper. The datasets used

for producing results, however, seem to be a bit simple. Perhaps that is just a function of the age of the paper.

I can't really take issue with the experiments. They compare over datasets that have been used by many other algorithms and compare quite favorably, assuming the authors have done adequate diligence in finding results on those datasets. Again, the datasets themselves seem to be a bit cooked up. I wonder how the algorithm performs when faced with the high levels of noise present in natural images.

I would like to see this paper extended to allow hierarchy within an object. Many objects are made up of smaller components. I want to see a way to find small components and then use those to find bigger components in a multi-step hierarchy. Very similar technology is used for morphing images and producing movie special effects. I wonder if there might be an interesting use of morphing for training a vision system. The paper points out that it intentionally avoids picking salient points. However, this doesn't seem completely logical. I agree that many other points are interesting, but salient points can often correspond to important points on an object. If they're easy to compute, it seems silly not to use them. As with almost all vision papers, this one also ignores color. I want to see more papers analyze color and how much more it adds or takes away, both in accuracy and complexity.

One additional question I had relates to equation (13) in the paper. This equation describes the computation of image similarity in a Gaussian region around a pair of matching points. I wondered if this measure is normalized to account for variation in lighting or if pixels are taken directly at face value.

The goal of this paper is to extract shape descriptors from visual objects and use them for object matching and recognition. Shape information can be combined with appearance for better accuracy. This is a highly influential paper with 836 citations according to Google Scholar.

Points are sampled uniformly on the edge-detected image of an object. A shape context is computed at each point which essentially describes how the rest of the points are positioned with respect to the current one. In order to match two shapes, corresponding points are identified according to their shape contexts, and other appearance information if desired, using bipartite graph matching. A thin plates splines transformation is computed that can best align the two shapes. The distance between the two shapes is based on the errors between corresponding points and the degree of transformation required.

The main contribution of this paper is the development of shape context that is robust to noise and illumination changes. It is translation and scale invariant. Rotational invariance can be added if desired.

The paper is very well written. In addition to pointing to the sources, it explains all relevant details so that the reader does not have to refer a lot of other papers (for example, for TPS, k-medoids etc). The reasoning behind most of the choices is explained.
In the context of object recognition, the method presented is prototype-based. Therefore, it suffers from the limitation of all methods in this category, that is, we need to have good representative prototypes of each class. This might be difficult for real world object classes like, say, a tiger. It can appear in many different postures and having a prototype for each of them is difficult. The method does not learn incrementally from the new examples as they become available. One could use each image in the 'training set' as a prototype image but that could hurt because it won't be able to focus on the common repeating aspects of the class and bad samples will badly affect accuracy.

Experiments have been designed meticulously. They were carried out on publically available datasets. That allowed the technique to be evaluated in comparison with other methods. All error cases in the MNIST test case are shown in the paper itself, and some of them look like easy mistakes to make, even by humans.

It is mentioned in the caption for Figure 6 that the value of epsilonD does not affect the solution. However, I'm curious what range of values is permissible for epsilonD.
By the description of the technique, it seems that it has not been developed for the cases where the object is not separated clearly from the background. But, if we assume that the variation in the background is higher than that within the object and try to incorporate that into the shape matching algorithm, perhaps we can find a way to do that reliably.
The method is shown to work for matching 3D objects by using it on images taken from different viewpoints of the objects. It seems that we can also extend the algorithm to work with real 3D representations of objects. These objects could be the 3D meshes used in Graphics, or space-time volumes of objects extracted from a video. In the latter case, it could serve as a motion descriptor.