WASHINGTON UNIVERSITY IN ST. LOUIS

School of Engineering and Applied Science

Department of Computer Science and Engineering

Masters Project Examination Committee:
Tao Ju
Robert Pless
Bill Smart

VISUALIZATION TOOLS FOR WEBCAM SCENES

by

David Ross

A masters project presented to the School of Engineering
of Washington University in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

May 2009
Saint Louis, Missouri

ABSTRACT OF THE THESIS

Visualization Tools for Webcam Scenes

by

David Ross

Master of Science in Computer Science

Washington University in St. Louis, 2009

Research Advisor: Professor Robert Pless

Webcams are cheap and abundant sensors that are easily deployed, maintained, and archived. The AMOS Dataset, which has captured over 1,000 webcams for three years, provides a multitude of diverse scenes, many of which contain interesting variation. The great size of this data makes keeping track of each scene's content impractical. This project explores techniques for automatically visualizing and characterizing these variations over static scenes. The mathematical tool Principal Component Analysis (PCA) is useful for learning common image variations, and its error can be leveraged to show less common and more interesting variation. With this technique, it is more reasonable to maintain and use such large sets of scenes.

Acknowledgments

First, I want to thank my advisor, Professor Robert Pless, for four years of teaching, guidance, and support. Without his direction, this project would not have been completed and my four years here would not have been so rewarding.

I want to thank the graduate students in the Media and Machines Lab, especially Nathan Jacobs and Michael Dixon, for helping me along the way. Very rarely did a question go unanswered with them around. Also, thanks to Professors Tao Ju and Bill Smart for agreeing to be on my presentation committee. I hope you learned something from me for once.

Finally, my utmost gratitude to Washington University, specifically the Computer Science Department. My four years here have prepared me so well for what's ahead.

David Ross

Washington University in Saint Louis
May 2009

Contents

List of Figures

Chapter 1

Introduction

Static cameras that observe the same scene for a long period of time can give us valuable information, such as clues for geo-location[2]. Given such a sensor, how can we help users to understand the variation in the scene? This project explores automatic visualization tools to quickly show interesting variation. As opposed to natural variation, such as day vs. night or sunny days vs. cloudy days, unnatural variation is less predictable and more interesting. People, cars, and other objects in the scene give more high level semantic information, and to understand a scene is to understand this variation.

1.1 AMOS Dataset

The Archive of Many Outdoor Scenes (AMOS) consists of over 3,000 webcams, 1,020 of which we have been actively capturing every half hour for the last three years. In this time, we have amassed 35,376,886 images, totaling roughly 868 gigabytes of data. Most of the cameras are from the USA, many of which are plotted in Figure 1.1. Clearly, it is impractical to maintain and understand such a large dataset without help. We need to automatically present users with visualizations of this variation.

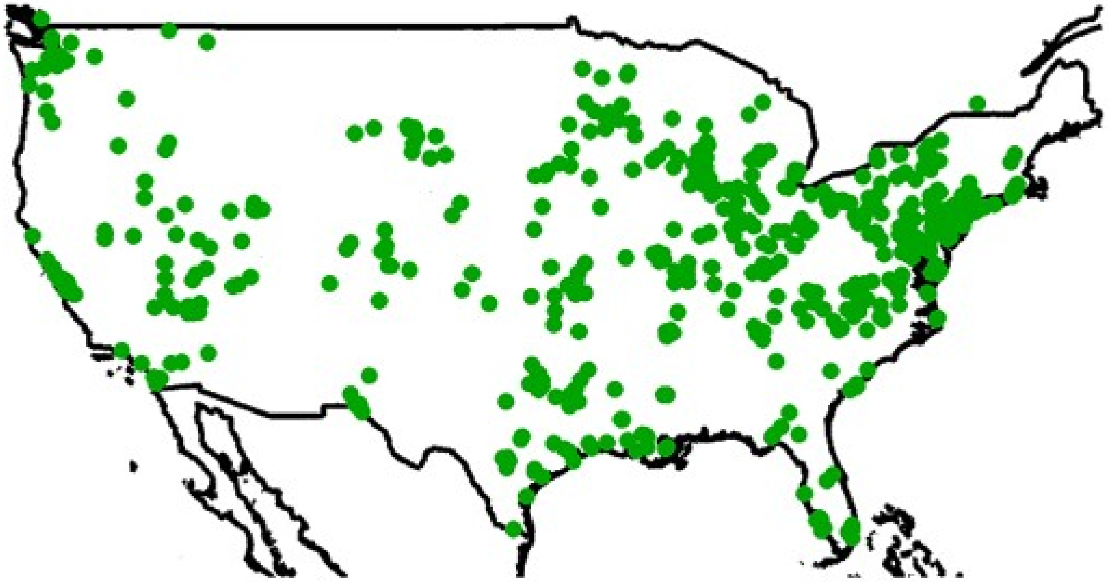


Figure 1.1: Many of the AMOS webcams are located in the United States, as shown in this map.

1.2 Outline of the Paper

We have many images from which we need to find interesting variants. A lot of the variance is natural and controlled - day to night transitions, seasonal changes, and weather are all predictable and easy to lean. First, we will present the mathematical tool Principal Component Analysis (PCA) as an algorithm to learn this natural variation. Next, we will discuss different inputs to the PCA algorithm to best capture our goal. Finally, we will show several visualization tools and evaluation criteria, and discuss their success in capturing PCA error, which is the interesting variation for which we search.

Chapter 2

Principal Component Analysis

Principal Component Analysis (PCA) is a powerful tool that extracts the most significant features from a given dataset. Given a set of images I and an integer $k > 0$, PCA projects I onto the k orthogonal basis vectors that best approximate I. Each image can be approximated by a linear combination of the basis vectors.

2.1 PCA Math

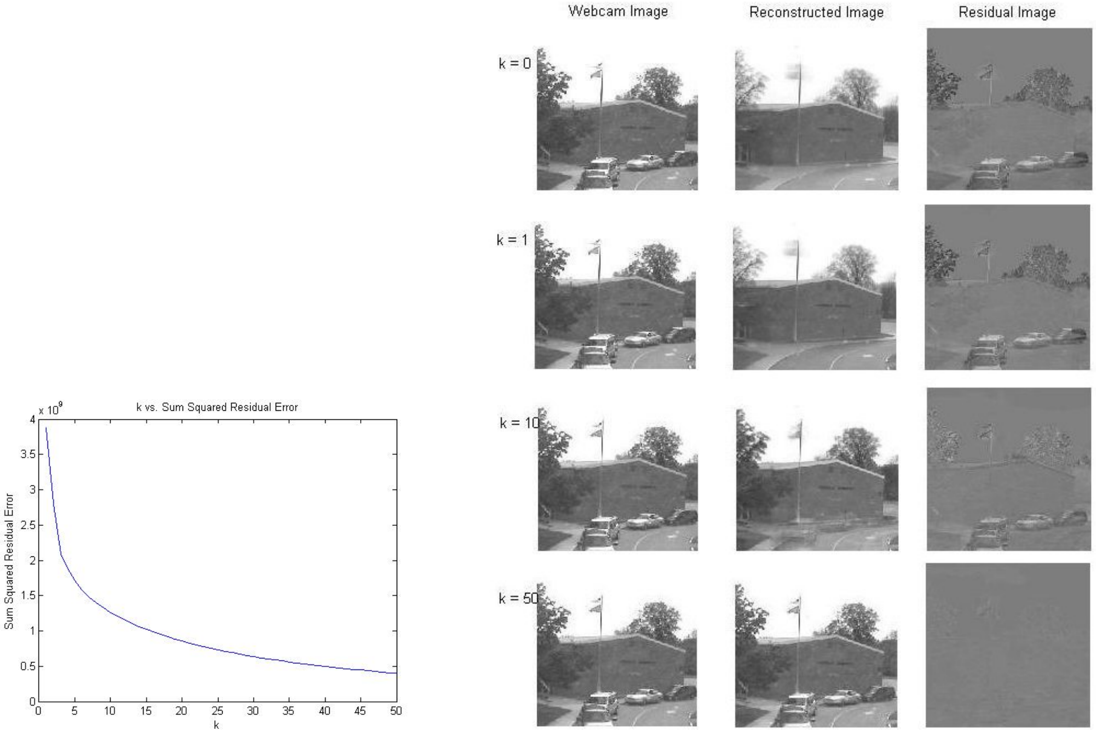
Given a set of n w-by-h images I, we create an n-by-$(w * h)$ matrix D, where each column is an image flattened into a vector with the mean image subtracted from it. We can now use a linear algebra technique called singular value decomposition (SVD) to solve for matrices U, S, V such that

$$M \approx USV.$$

where U contains the k orthogonal basis images as its columns, S is a diagonal matrix which contains the weights, or singular values, of each feature vector, and V contains the coefficients of the linear combination of basis vectors that best approximates each vector in D.

To reconstruct an image vector I_x, we convert it to a vector $I_{x-mean} = I_x - I_{mean}$, and we can say

$$I_{x-mean} \approx I_{reconstructed} = USV_x$$



(a)

(b)

Figure 2.1: Figure 2.1(a) plots the sum squared residual error for reconstruction of a webcam image for values of k from 0 to 50, and $n = 189$. When $k = 0$, the reconstruction is simply the mean image. Figure 2.1(b) shows the image, the reconstruction, and residual for $k = \{0, 1, 10, 50\}$. Notice that we start to model the cars at $k = 10$ and get a great approximation at $k = 50$.

where V_x is the vector of coefficients corresponding to I_x. If $k < n$, our reconstruction is not guaranteed to be perfect, but as k approaches n, the reconstruction gets better, as shown in Figure 2.1. The reconstruction error, or residual, can be calculated as

$$I_{residual} = I_{x-mean} - I_{reconstructed}$$

In situations where dataset sizes conflict with memory constraints, we can use a modified algorithm called Incremental PCA, which allows allows us to iteratively update our matrices U, S, V for new images[1].

Chapter 3

Visualization Tools

The main contribution of this project is automatic methods for visualizing how a webcam scene varies and for understanding the most important variations. In most natural scenes, we notice changes in lighting, weather, and camera conditions which fail to describe the typical behavior of a scene. The goal of these tools is to learn these variations and to point out changes independent of them. PCA is a commonly used tool, and captures a linear model of consistent image variations. By analyzing the results of certain PCA trials, we can obtain important and interesting information.

3.1 PCA Input

A naive algorithm to learn about scenes is to take the PCA of the entire webcam scene. Modifying our input, however, can produce better results.

3.1.1 Temporal Narrowing

The AMOS Dataset consists mostly of outdoor scenes. These outdoor scenes vary significantly over the course of a day, going from night to day and back. The change in lighting dominates all other changes across the scene, and causes many images to be dark and noisy. Instead of wasting PCA basis vectors on learning this change, we can limit our input to daytime images which we expect to be well-lit. This strategy is used for all scenes.

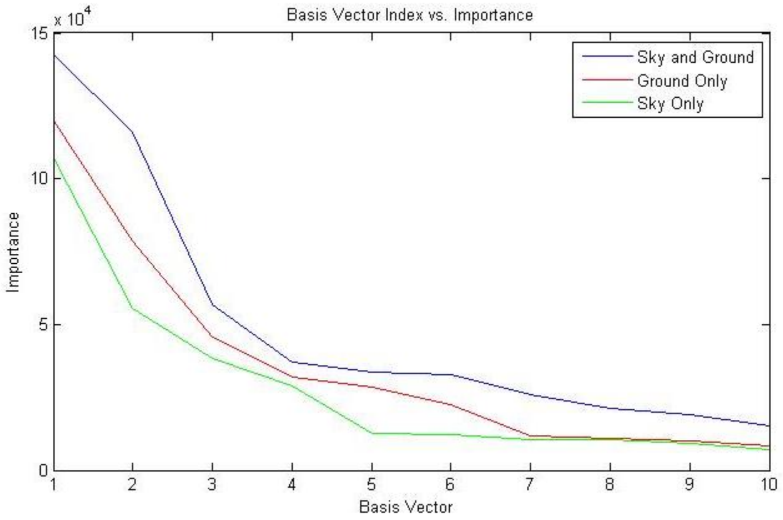


Figure 3.1: The plot shows the singular value, or importance, or each of the ten basis vectors for PCA on a scene with both the sky and ground, with the ground only, and with the sky only. Since the latter two's singular values do not at up to the former's, we can conclude that looking at the ground only captures more information about the ground than looking at both concurrently.

3.1.2 Sky Mask

In many outdoor scenes, even when narrowed to a particular time of day, the most difficult image variation to characterize is the sky. PCA has a difficult time learning changes in sunlight, clouds, and other sky-based variation. The variance in the sky dominates the PCA reconstruction, causing the basis vectors to focus less on more important variation, as shown in Figure 3.1.

Fortunately, it is fairly easy to learn which regions of an image should be labeled as sky. A side effect of the rising and setting of the sun is that the first principal component of most scenes - looking at all images, not just daytime images - is the sky, as shown in 3.2(a). By thresholding the values of this vector, we can effectively segment the sky from the rest of the image, as shown in 3.2(b).

This simple mask allows PCA to focus on more interesting changes in the image, and most of the results that follow use a simple mask to ignore the sky regions of the images. By setting all pixels values in the sky to 0, we can effectively remove this difficulty.

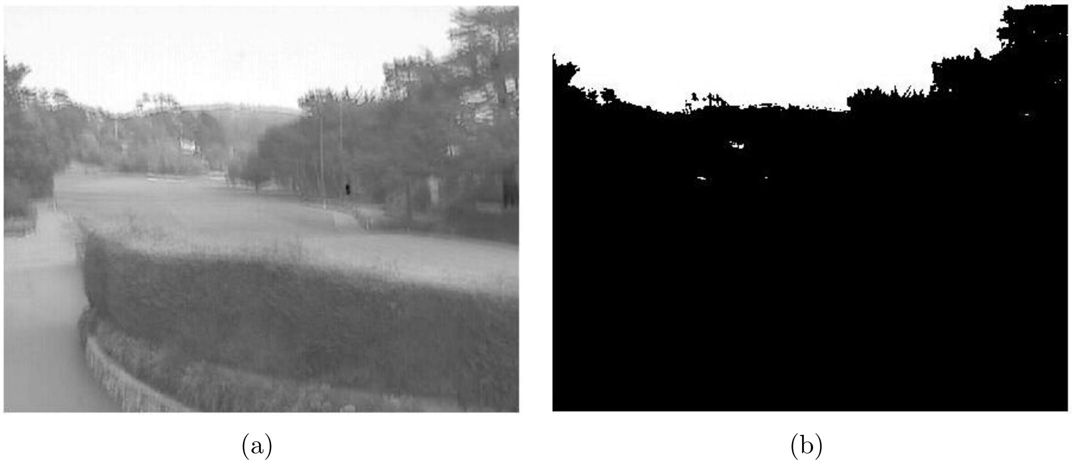


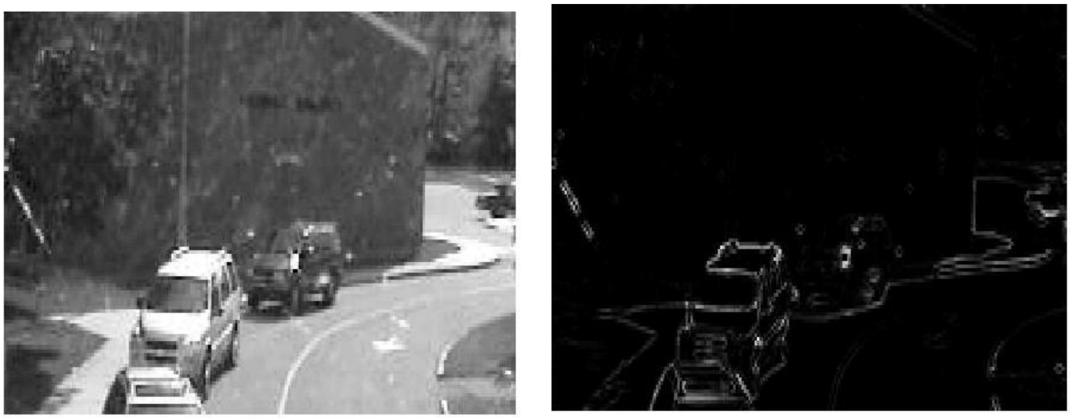
Figure 3.2: Figure 3.2(a) shows the first PCA component of a webcam scene. By adding or subtracting this component, we control how dark or how light the sky is. Figure 3.2(b) shows that a simple thresholding of this image effectively segments the sky from the rest of the image.

3.1.3 Gradient Image

The gradient magnitude image highlights edges in an image, as they are image locations where pixel values differ greatly from their neighbors. By performing PCA on the gradient magnitude images, we tend to ignore the potentially noisy surfaces of objects, and instead focus on the locations of the objects. Figure 3.3(a) and Figure3.3(b) show the edges of an interesting webcam image highlighted in a gradient magnitude image.

The x and y derivatives of an image are approximated as $I_x(x, y) = I(x, y) - I(x-1, y)$ and $I_y(x, y) = I(x, y) - I(x, y-1)$ where the function $I(x, y)$ describes the intensities of an image's pixels. Once we have calculated the derivative images, we define the gradient magnitude of an image to be $G(x, y) = \sqrt{I_x(x, y)^2 + I_y(x, y)^2}$.

In practice, taking PCA of the gradient magnitude image does not add much to our results, so we bypassed this step in our work.



(a) (b)

Figure 3.3: Figure 3.3(a) shows a grayscale webcam image. Figure 3.3(b) shows its gradient magnitude image. Notice the noise is mostly removed and the edges of the vehicles and building are highlighted.

3.2 Visualizations

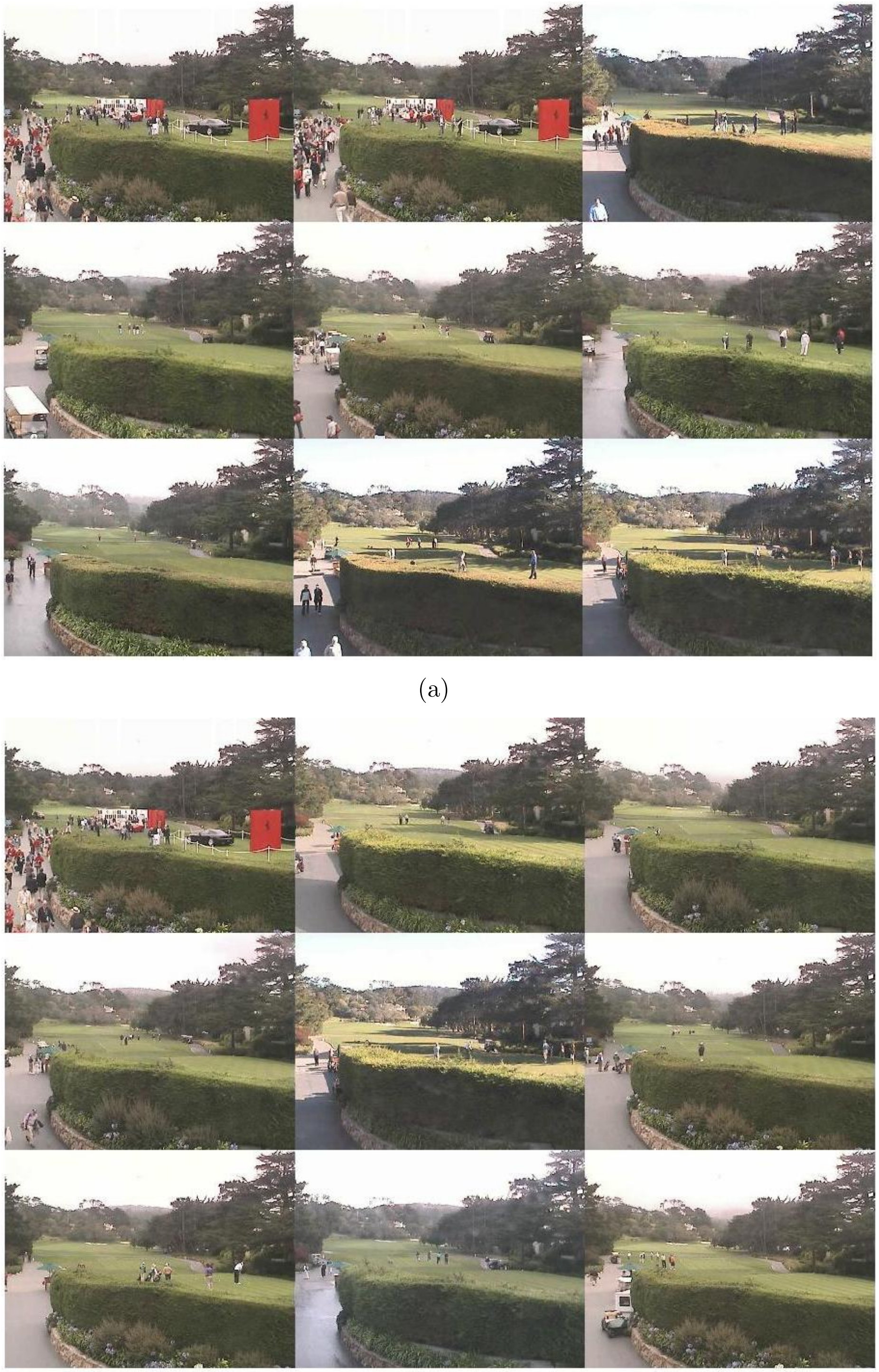
In this section, we present several tools to visualize the webcam scene once we have evaluated the PCA error.

3.2.1 Image Montage

A simple way to visualize a webcam scene is to view a montage of several images from that scene. We can easily sort the images along some dimension, and then show the n images with the highest values in that dimension. Figure 3.4(a) Is an example montage of several interesting images from a golf course scene.

3.2.2 Well-Separated Set Montage

In several of these montages, we notice that many of the most unusual images are similar to each other. This presents the problem of finding unusual images that are different from the images we have already chosen.



(a)

(b)

Figure 3.4: Figure 3.4(a) shows a montage of interesting images from a golf course scene. Figure 3.4(b) shows a montage of interesting images of the same scene using the Well-Separated Set technique.

A simple but effective algorithm for this problem uses the L2 norm in image space. If we consider an image as a vector of pixel intensities, such as $v = \{v_1, ..., v_m\}$, the L2 norm of two images v and u is

$$L2\_norm(v, u) = \sqrt{\sum_{i=1}^{m} (v_i - u_i)^2}$$

Given a large set of unusual images $\{x_1, ..., x_n\}$, we compute a distance matrix D where $D_{i,j} = L2\_norm(x_i, x_j)$. Now we iteratively find unusual images by choosing the image that has the largest distance from all of the images we have chosen so far. Specifically, for each remaining image calculate its distance to our set of exemplars, and choose the image whose distance is the smallest. We define the distance of an image x_0 to a set of exemplars $\{x_1, ..., x_n\}$ as

$$d = \min_i(D(x_0, x_i))$$

Then we can choose the image that has the highest value of this distance. In this way, on each iteration, we pick the image that is least likely to be similar to an image we have already selected. Figure 3.4(b) shows how this technique omits similar images.

3.2.3 Two-dimensional Explorer

Using a simple GUI, we can explore a webcam scene in two dimensions. The GUI, shown in 3.5, displays a a plot of image scores for two different criteria, and displays the image corresponding to the highlighted data point. Using this GUI, it is possible to see which evaluations are related and which are not, and makes it easier to learn how the evaluations appear in image space.

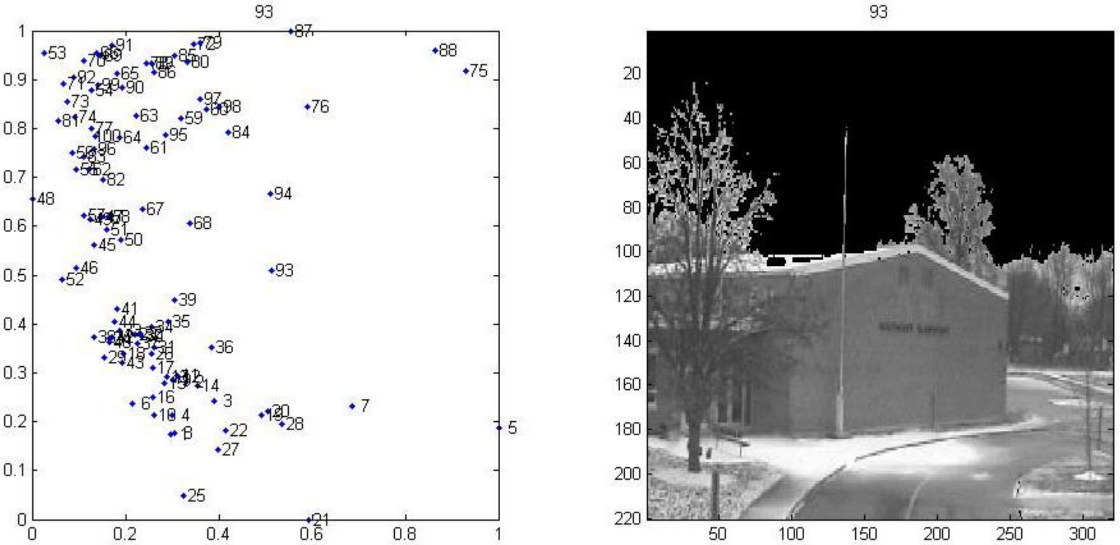


Figure 3.5: This figure shows a webcam scene displayed using the 2D GUI.

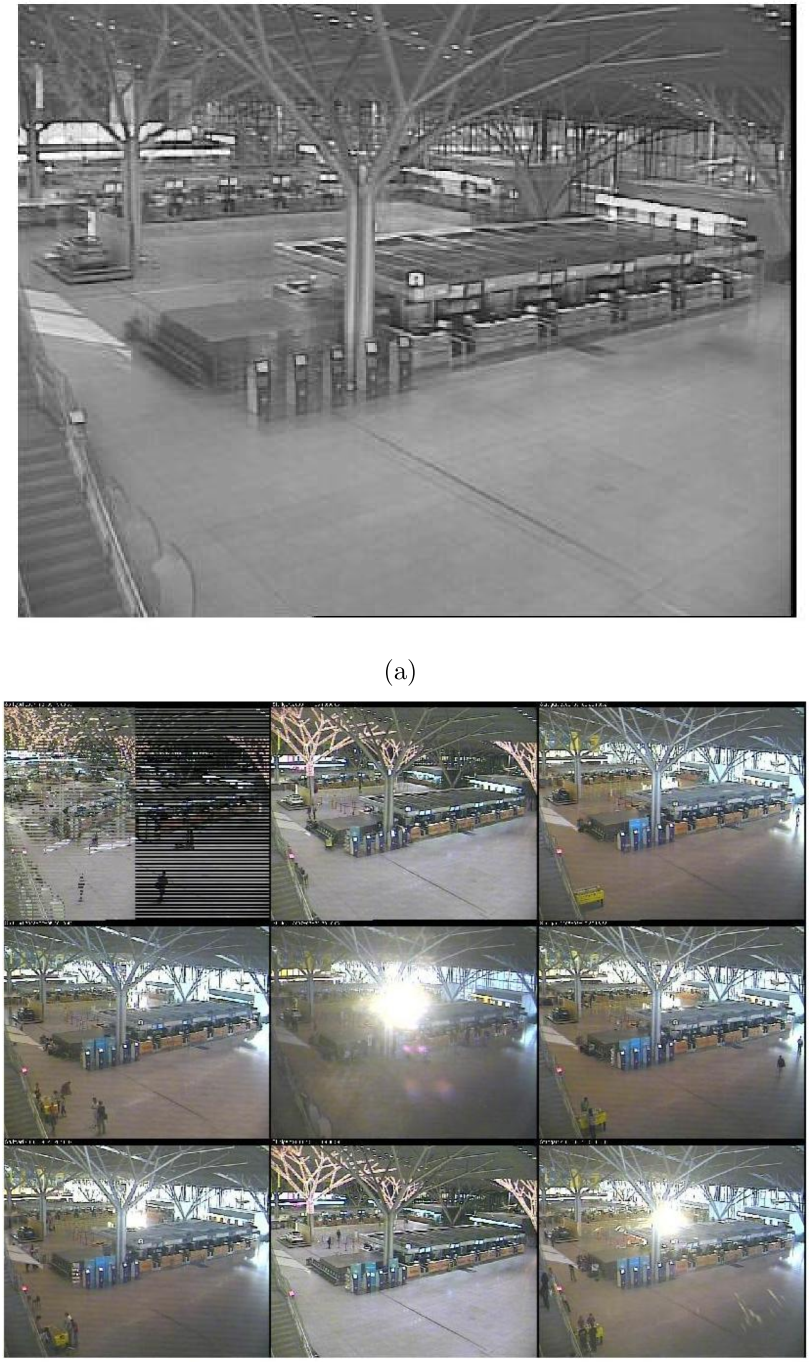
3.3 Characteristics of Image Reconstructions and Residuals

Once webcam scenes are projected onto a PCA basis, there are many different ways to analyze the results. In this section, we will present several different methods for evaluating this appearance model of a scene.

3.3.1 PCA Basis Coefficient Vector Magnitude

For each image in a scene, PCA gives a vector of coefficients that correspond to the best linear combination of basis images to reconstruct that image. From this vector, we can assign each image a score equal to the magnitude of this vector. For a vector $v = (v_0, v_1, ..., v_n)$, the vector magnitude is

$$||v|| = \sqrt{\sum_{i=0}^{n} v_i^2}$$



(a)

(b)

Figure 3.6: Figure 3.6(a) shows the mean image of an airport scene. Figure 3.6(b) shows several images found using this technique that are far from the mean.

This effectively gives us a measure for how far from the mean image in our basis space each image is. Figure 3.6(a) shows the mean image of a webcam scene and Figure 3.6(b) shows several images from that webcam scene that are especially far from the mean. Evaluating and montaging images a scene in this way is an effective strategy for finding interesting images, but gives no semantics as to how the images are interesting.

3.3.2 Residual Error

Given an image and a PCA basis, we can project the image onto the basis and see how well we this reconstructs the image by analyzing the residual image, as shown in Figure 3.7.

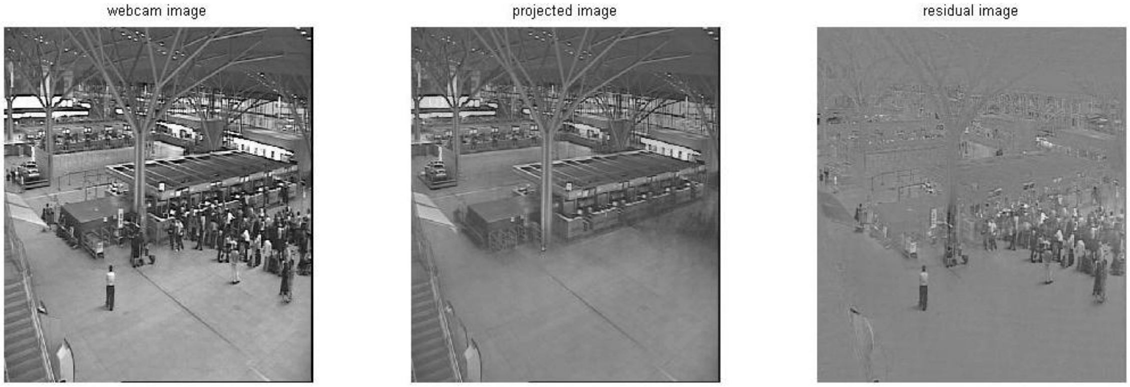


Figure 3.7: This figure shows a webcam scene image, the reconstruction of that image from a PCA basis, and the residual image. Notice how the residual image contains most of the foreground objects.

If we sort evaluate our images based on the sum of the squared residual values of each pixel, we get a good measure for how much variation is not captured by PCA, as shown in Figure 3.8. Whereas the PCA coefficient vector magnitude evaluation gives images that are different from the mean image, this evaluation captures more of the interesting variation that we are looking for.

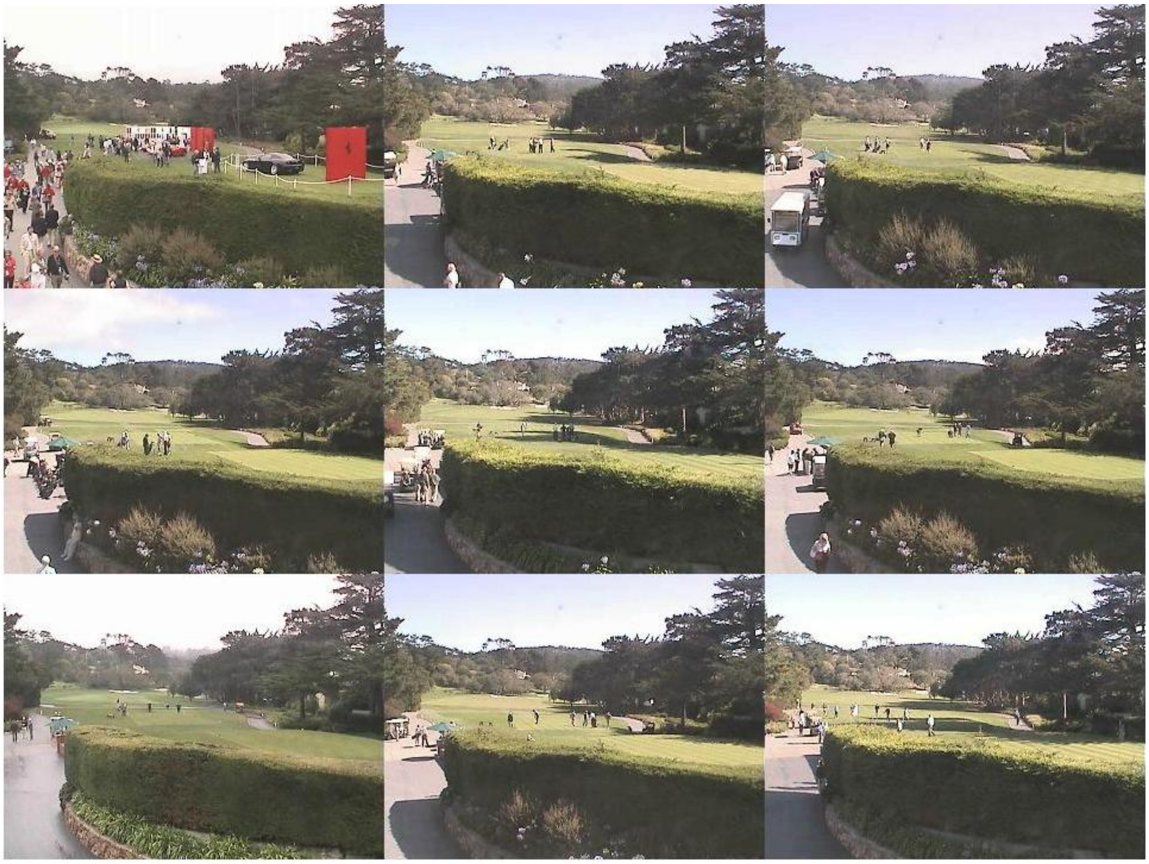


Figure 3.8: This figure shows a the most interesting images from a golf course scene sorted by the sum of the squared residual values.

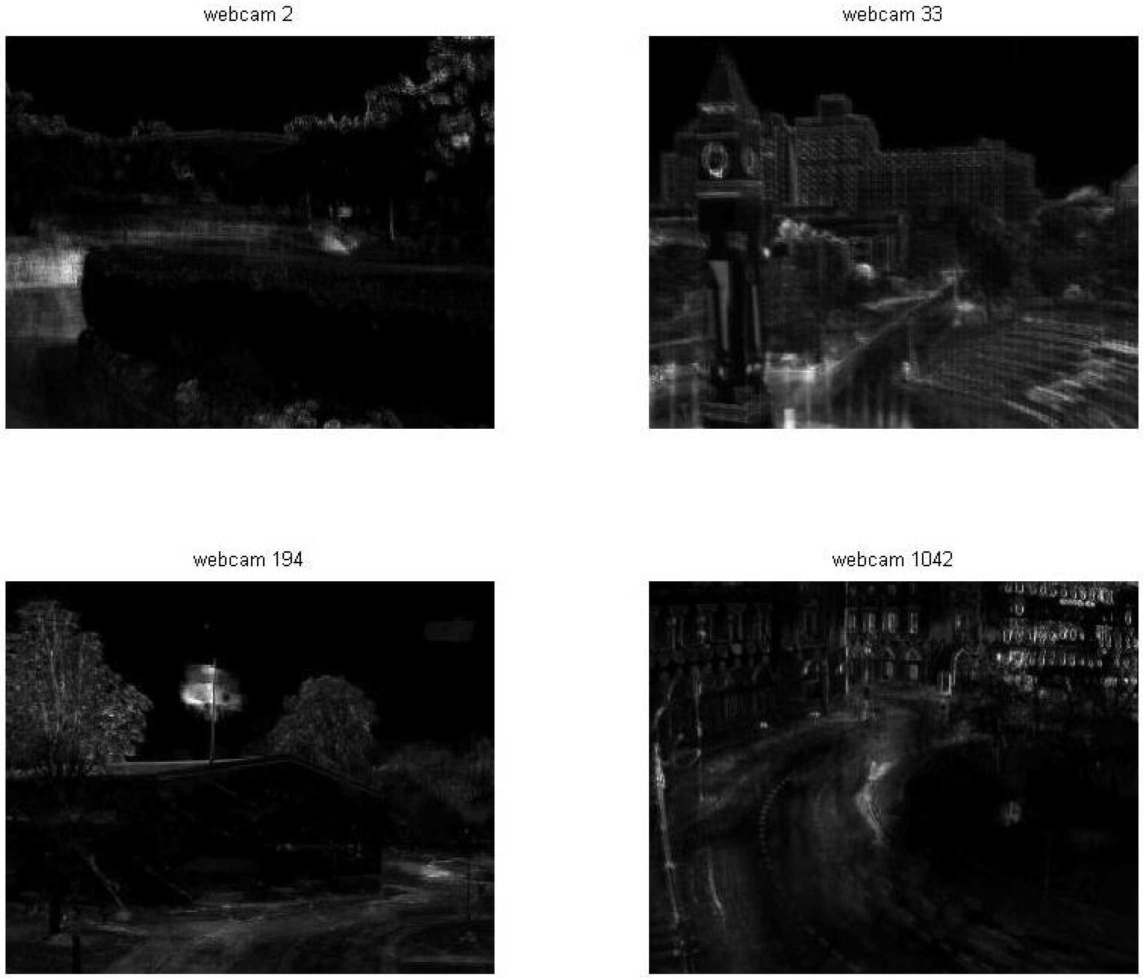


Figure 3.9: This figure shows estimated variance images from several webcam scenes.

3.3.3 Variance Model

We can estimate the variance image of a webcam scene as the average of the square of each residual image. This image gives a good summary as to where most activity occurs in a webcam scene. Several examples are shown in Figure 3.9.

One we have this variance image, we can attempt to isolate independent pixels in a z-score image. We define our z-score image as

$$Z(x,y) = \frac{R(x,y)}{V(x,y)}$$

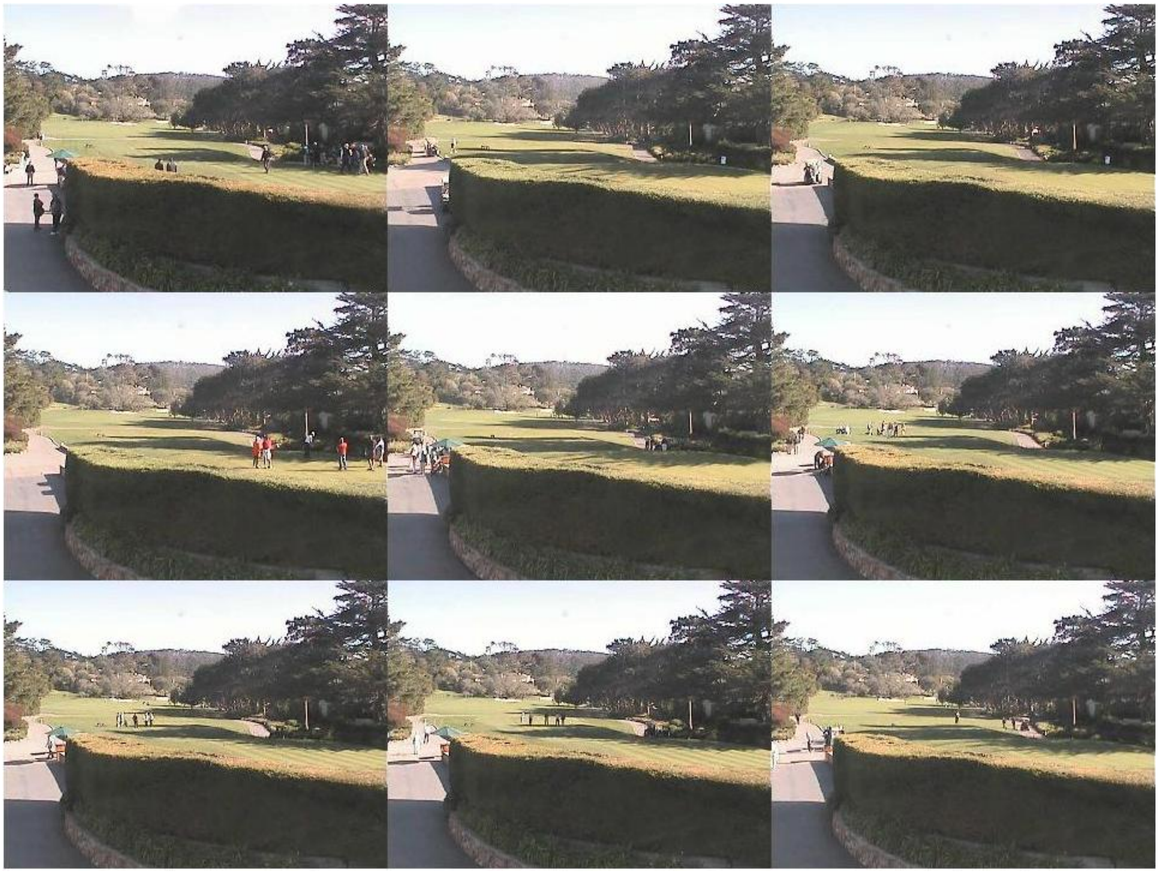
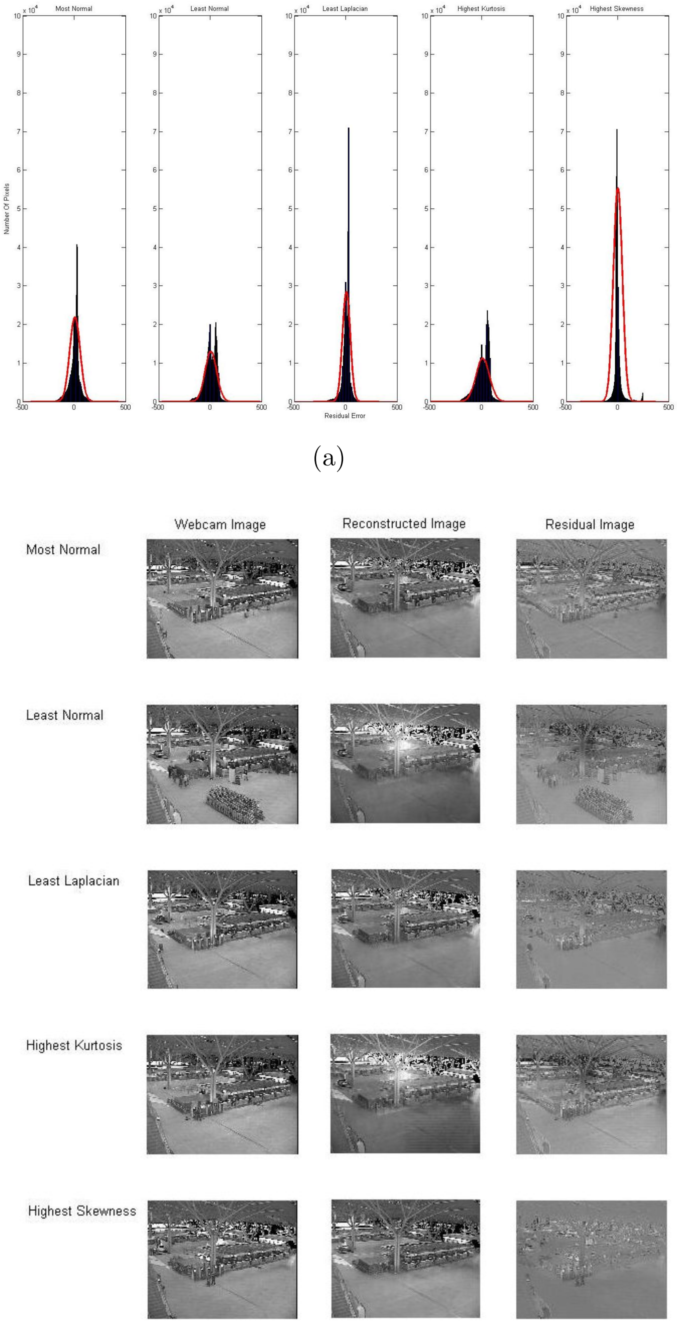


Figure 3.10: Figure 3.10 shows several images from the same scene as 3.8, but sorted by the sum of the Z-Score values. The objects in these images are in more unlikely positions as those in the residual sum squared error montage.

where $R(x, y)$ is the residual value of the pixel and $V(x, y)$ is the variance value of the pixel. By summing the per pixel z-score for an particular image, we can infer how atypical the residual of that image is. Figure 3.10 shows several images that contain atypical objects. Such a tool can be used to quickly highlight unusual behavior in a scene, such as unusual shots on a golf course or security breaches in a bank scene.

3.3.4 Statistical Distribution of Residual Images

We can also try to learn about an image reconstruction by treating its residual image as samples from an underlying probability density function. If image deviations are due mostly to noise, we predict that these deviations will be normally distributed,



(a)

(b)

Figure 3.11: Figure 3.11 shows several residual image histograms and the the best approximation of a normal distribution fitting the data. Figure 3.11(b) shows the image corresponding to each histogram and the corresponding reconstructed images and residual images.

but as shown in Figure 3.11, this is not always the case. We have found that these distributions vary in the same way as webcam scenes, and analyzing them in the right way tells an interesting story.

1. **Gaussian Likelihood**

   A simple strategy is to treat each residual image pixel as a sample from a Gaussian distribution. We can easily estimate the mean and variance of this PDF and then, for each residual value, use the Gaussian distribution equation

   $$f(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}}e^{\frac{(x-\mu)^2}{2\sigma^2}}$$

   to evaluate the overall likeliness that the residual image was sampled from the estimated Gaussian distribution.

   Figure 3.12(a) shows a Well-Separated Set Montage of images that were not likely Gaussian. These images capture some interesting variation, but also some variation in lighting and camera artifacts.

2. **Laplacian Likelihood**

   Many of the residual images have a majority of pixels that are very close zero. This causes the histogram to look very similar to the Laplacian distribution. We can estimate the Laplacian likeliness in the same way as for Gaussian distribution, but using the Laplacian distribution definition

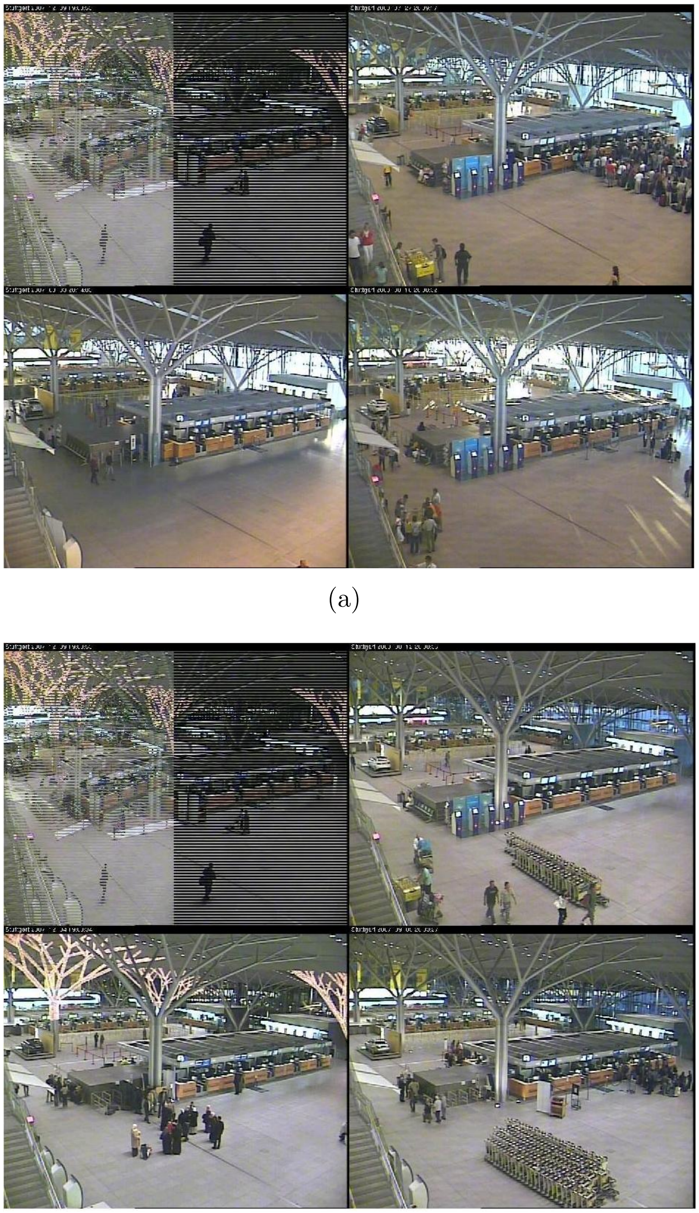
   $$f(x|\mu,\beta) = \frac{1}{2\beta}e^{\frac{-|x-\mu|}{\beta}}$$

   Figure 3.12(b) shows a Well-Separated Set Montage of images that were not likely Laplacian. These images tend to capture more interesting variation than those in Figure 3.12(a).

3. **Kurtosis**

   The kurtosis of a real-valued random variable is a measure of its peakedness. Larger values of kurtosis means more variance is due to less frequent extreme deviations rather than frequent less extreme deviations. It is defined as

   $$\gamma_2 = \frac{\mu_4}{\sigma^4}$$



(a)

(b)

Figure 3.12: Figure 3.12(a) shows a montage of interesting images from an airport scene that were labeled very non-Guassian. Figure 3.12(b) shows a montage of interesting images from an airport scene that were labeled very non-Laplacian. Notice that the least Laplacian images contain foreground objects than do the least Guassian images.

where μ_4 is the fourth moment about the mean and σ^4 is the estimated standard deviation to the fourth power. For a function $f(x)$, the k^{th} moment about the mean is defined as

$$\mu_k = \int_{-\infty}^{\infty} (x-\mu)^k f(x)dx$$

Evaluating images based on the kurtosis of their residual images tend to give results that mirror the images sorted by sum of the squared residual values.

4. **Skewness**

The skewness of a random variable is a measure of asymmetry. Higher skewness values mean deviations on one side of the mean do not have corresponding deviations on the other side. It is defined as

$$\gamma_1 = \frac{\mu_3}{\sigma^3}$$

where μ_3 is the third moment about the mean and σ^3 is the cube of the standard deviation.

Evaluating images based on skewness does not give positive results. The is possibly the result of the fact that small objects on either side of the mean that would add to the asymmetry of the distribution are dominated by the noise over the rest of the image.

Chapter 4

Conclusion

The original goal of this project was to find clean mathematical tools to catalog the objects found in a particular webcam scene. Cameras vary in sufficiently difficult ways to make this hard, so we investigated general tools for visualizing these variations. These automatic visualizations use PCA to learn less interesting variation, and then extract the interesting variation from the PCA error. By passing in the right input to PCA and analyzing the output correctly, we can get good results.

4.1 Future Work

The tools presented in this project are especially useful in situations where large amounts of data necessitate efficient tools for understanding it. AMOS is an example of such a large dataset, and some future work could make understanding AMOS even easier.

First, the AMOS website has tools to allow users to understand each scene. These tools, however, are neither automatic nor efficient, so finding scenes with certain characteristics can be difficult. To remedy this, the tools from this project can be set to automatically run on each webcam, and display results on each camera's page.

Additionally, although this goal was not met, these visualization tools give information that would be helpful for the original problem of object extraction. The tools are effective in separating interesting variation from natural variation, so work could be done to connect the interesting pixels into objects.

Finally, another area that this work could be expanded in is user customizability. A tool with which a user could specify what type of information he or she is interested in would be powerful. Much as in the same way the variance image can affect which variation is labeled more interesting, a user-defined importance map can add a layer of customizability.

References

[1] Matthew Brand. Incremental singular value decomposition of uncertain data with missing values. In *In ECCV*, pages 707–720, 2002.

[2] Nathan Jacobs, Nathaniel Roman, and Robert Pless. Toward fully automatic geo-location and geo-orientation of static outdoor cameras. In *IEEE Workshop on Applications of Computer Vision (WACV 2008)*, Copper Mountain, CO, January 2008.

Visualization Tools for Webcam Scenes, Ross, M.S. 2009