Proc. Third IEEE International Symposium on Intelligent Control, Arlington, VA., August 1988.

Low Data Rate Remote Vehicle Driving

Martin Herman, Karen Chaconas, Marilyn Nashman, Tsai-Hong Hong

Sensory Intelligence Group
Robot Systems Division
National Bureau of Standards
Gaithersburg, MD 20899

ABSTRACT

In order to accurately maneuver a remotely-driven ground vehicle, visual data must be supplied to the operator as quickly and as accurately as possible. Unfortunately, high data rate communication channels are often not feasible for this task. To accomplish remote driving using a low data rate channel, video compression techniques must be incorporated. This paper discusses the remote vehicle driving problem and describes several video compression algorithms that have been implemented on PIPE, a real-time pipelined image processing machine.

1. Introduction

The remote vehicle driving scenario involves an operator who sits at a remote control center and views video images that are transmitted from one or more cameras mounted on the vehicle. The operator drives the vehicle by means of driving controls such as a steering wheel, brake pedal, accelerator pedal, etc. Full-rate video transmission from the vehicle to the operator requires about 60 megabits/second for 512 x 512 images with 8 bits/pixel at 30 frames/second. However there are several problems with using the wide communication bandwidth required for such transmission. First, wide bandwidth radio communication requires direct line of sight between the transmitter and receiver. This is not feasible in realistic outdoor scenarios where vehicles are likely to be driven behind hills and mountains and therefore hidden from direct view by the operator station. Wide bandwidth links are also relatively expensive. Further, even if such a link were available, full-rate video would use up a large part of the bandwidth allocations. This would be particularly true if there were many vehicles being operated simultaneously, where full-rate video might fill up the entire communication spectrum. Wide-bandwidth fiber optic links also have several problems, including limited ruggedness, difficulties in deployment and retrieval, and the problem of repairs.

Many of these difficulties can be overcome by utilizing narrow-band radio links (perhaps 100 kilobits/second). Since the full video required for teleoperation cannot be provided using the narrow band links, efficient and effective techniques of video compression offering compression ratios of 500:1 to 1000:1 must be developed.

2. Requirements on Vision for Remote Driving

A vision system that is used for remote driving can be evaluated in terms of how well it permits the remote operator to perform vehicle mobility and, in threatening situations, to provide for vehicle survival. There are various factors that affect vehicle mobility. The first is the ability of the operator to make trafficability and movement decisions. These include:

1. Local obstacle detection. The vision system should allow the operator to detect local obstacles.
2. Local surface classification. The vision system should allow the operator to determine the kinds of surfaces surrounding the vehicle.
3. Local surface orientation. The vision system should allow the slope of the local terrain to be determined visually.
4. Local path planning. The vision system should provide enough information for the operator to perform local path planning. This involves not only the ability to detect local obstacles or classify the local surface, but also the ability to obtain spatial relations among the obstacles and the surface.
5. Local path following. The vision system should provide enough information so as to be able to follow terrain features such as roads or rivers.

A second factor that affects vehicle mobility is the ability for the operator to maintain a sense of the global vehicle location relative to the background and landmarks. These include the ability to identify landmarks and the background, as well as the ability to track landmarks over time. A third factor that affects vehicle mobility is the ability of the operator to maintain a sense of speed of the vehicle.

In order to satisfy the requirements set forth above, two additional requirements should be met. First, the video images should be displayed in real time. This means that the operator should have the appropriate visual information quickly enough so that he can drive the vehicle interactively. Driving is typically a servoing operation which requires a fast loop involving transmitting driving commands to the vehicle, getting instantaneous visual feedback, and using this feedback to transmit additional driving commands. A second requirement is that the video sequence that appears on the operator's monitor should seem natural (e.g., smooth and continuous).

3. Image Compression Background

This section provides a brief survey of image compression techniques. Most of these techniques have not been developed in the context of remote vehicle driving, and therefore many of them are not applicable to this problem. One of the goals of our study is to determine which ones might be applicable. The effectiveness of a transmitted image for remote vehicle driving is directly related to how well it approximates the original scene, how fast it can be generated, and the compression ratio.

A class of methods known as first-generation data compression techniques achieve compression ratios of about 10:1. One such method is the coder method, a simple error-free data compression method which has a one-to-one input-output relationship. In order to achieve the compression, the coder must use as few bits as possible. In theory, a lower bound on the average number of bits required to code a set of input data cannot be less than the first order entropy of input data if successive input data is coded independently [2]. (Entropy will be described in more detail in a later section.) In general, the entropy ranges from 0 to $\log_2 L$ where L is the number of input data values. As a result, the maximum bound on the compression ratio is $\log_2 L : 1$.

The compression ratio for a coder is a function of N, where N is the number of grey levels in an image. One way to achieve higher compression is to decrease N by quantizing the grey levels. This is called a grey-level quantizer. Another method is to spatially quantize data by sampling. This is called a spatial quantizer. Since quantization is irreversible, the distortion due to quantization must be minimized. The Lloyd-Max quantizer, the compressor-expander, the optimum uniform quantizer, and the Shannon quantizer offer various tradeoffs between simplicity and performance. However, the maximum bound on the compression ratios for the grey-level quantizer is N:M, where M is the number of grey levels in the quantized image. The compression ratio for a spatial quantizer is a function of the sampling interval -- the larger the interval, the worse the distortion. In the case of remote driving, choosing the interval is an important issue.

Another class of methods, predictive compression methods, makes use of the property that values of adjacent grey levels are highly correlated for most images. The major predictive method is called Differential Pulse Code Modulation (DPCM). The maximum compression ratio achieved by this method is about $\log_2 N : 1$, where N is the number of grey levels in the input images. Another predictive method is the interpolative method. Most commonly used interpolators are zero-order and first-order interpolators. However, higher order polynomials or splines can also be used. The maximum compression ratio for the interpolative methods is the same as those of the DPCM methods.

Transformation methods reduce the correlation between pixels in the image by transforming the image into another domain. The performance of a transform method depends on the type of transformations used, the dimension of the transformation, the quantization strategies and the subwindow size. Many transformation methods have been developed. The Hotelling transformation performs best from both a mean-square error and a subjective quality viewpoint. However, a fast discrete cosine transform is most commonly used. The compression ratio for the methods are approximately 10:1.

Recent progress in the study of the properties of the human visual system has led to a new class of image compression methods capable of achieving compression ratios as high as 70:1. This class of methods is known as second generation data compression. We describe these techniques below.

Burt and Adelson [1] have proposed a fast algorithm called the pyramid coding method. The method is a hybrid method which combines features of predictive and transform methods. The algorithm first obtains the predicted value for each pixel in the image by convolving a 5 x 5 Gaussian-like kernel with the image. The result is a low-pass filtered image which can be represented by fewer samples than the original. The first predicted error image is then the difference between the filtered image and the original image. The predicted error image can be quantized and coded with many fewer bits than the original. Recursively obtaining the next predicted error image by using the same algorithm applied to the sampled low-pass filtered image achieves further data compression. The original image can be reconstructed by adding all of the expanded predicted error images. This algorithm can achieve a compression ratio of 10:1. The algorithm has been implemented on the PIPE and will be described in a section below.

The anisotropic non-stationary predictive coding method [5, 10] can be classified as a hybrid method which combines the predictive and transform methods. Lines and edges in an image provide key information. A grey scale image is first transformed into two bias images by measuring these nonstationary linear features. One bias image is a measure of edge magnitude and direction. The other bias image is binary and indicates whether an anisotropic prediction should be used because edge directionality varies rapidly from point to point or whether an isotropic prediction is sufficient. These two bias images control the anisotropic filter, which is an estimator defined in terms of pixel position and its edge angular frequency. The typical compression ratio for the method is 35:1.

Contour and texture coding methods first segment the image into textured regions surrounded by contours such that each contour represents one object in the image. Kunt et al. [6] overviews several approaches for contour and texture segmentation algorithms. After the segmentation, the contour and texture in an image are coded separately. Using this method, the compression ratio can be as high as 50:1.

Image compression techniques for real time applications in remotely piloted vehicles, video conference images, and video telephone images, have been developed. In order to achieve a relatively high compression ratio and meet the real time requirement, the techniques are combined with other previously discussed methods. For example, the remotely piloted vehicle application combines the transformation method and frame-rate-deduction (quantization method) to achieve a compression ratio of 60:1. Lippmann [7] proposes a motion-adaptive frame interpolation method which combines frame-rate-deduction (quantization method) and the linear interpolative method (the predictive method) for video transmission of airborn television images. The most commonly used method for video conference and video telephone images combines motion-compensated prediction, discrete cosine transformation, and quantization.

In order to meet the requirements of remote vehicle driving, a hybrid method will be used, as described in the next section.

4. Approach

Our approach to performing video compression for remote driving is to use a hybrid method which combines image processing techniques (i.e., techniques whose input is an image and whose output is a compressed image), transform techniques (such as the discrete cosine transform), and temporal frame rate reduction (i.e., transmitting fewer than 30 images per second). The sequence of events as they would occur in the system is as follows. Images are obtained from one or more cameras mounted on-board the vehicle. These images then undergo compression using the hybrid technique. After the compressed code is transmitted over a communication link to the operator station, it is decompressed so as to result in a sequence of full size images. The compressed images will be transmitted over the communication link at a rate of at most a few per second. However, we want the images to be displayed on the operator's monitor at 30Hz. This requires an extrapolation procedure, which will be provided by a real-time image warping processor.

Our plan for this project has been to first implement video compression algorithms on the PIPE real-time image processing machine (to be described in more detail below). PIPE is excellent for quickly developing, testing and modifying algorithms. PIPE was then integrated with a remote control vehicle system and the algorithms were evaluated on real-world remote driving tests. Initial results of these tests are described in a companion paper [3]. After a subset of algorithms have been chosen based on such tests, they will be reimplemented on special-purpose image processing boards which will reside on-board the remote vehicle.

The remainder of this paper will describe the PIPE machine and will discuss compression algorithms implemented thus far on PIPE.

5. PIPE

PIPE (Pipelined Image Processing Engine) is a multi-stage, multi-pipelined image processing device that was designed for real-time robot vision applications. PIPE was conceived and designed at the National Bureau of Standards [4]. It is currently commercially available through Aspex, Incorporated. PIPE will accept images from a video camera at field rates – 60 times per second. PIPE's basic cycle rate is 1/60 second.

The PIPE system is composed of up to eight identical modular processing stages, each of which contains two image buffers, five look-up tables, three arithmetic logic units, and two neighborhood operators. Images are transferred from stage to stage at field rate (60 images per second) by three concurrent pathways. The forward path allows traditional pipelined and sequential processing. The recursive path from a stage output back to its input allows feedback and relaxation processing. The backward path from one stage to the previous stage allows for temporal operations. It also allows a hypothesis image to be compared with an input image. The images in the three paths can be combined in arbitrary ways on each cycle of a PIPE program, and the chosen configuration can change on different cycles. In addition, six video buses allow images to be sent from any stage to any one or more stages.

Images can be processed in any combination of four ways on PIPE: point processing, spatial neighborhood processing, sequence processing or Boolean processing. Different processing can occur at individual pixels in the image by using a region-of-interest operator. This operator uses one of the image buffers in the stage as a map for selecting the algorithm to be applied to the contents of the other image buffer.

Point processing can be either a function of one or two input images and includes simple arithmetic and logical operations such as scaling, thresholding, converting number systems, etc. Look-up tables resident on each PIPE stage allow the user to perform more complex arithmetic operations, trigonometric operations, comparisons, rotations, etc.

PIPE can perform up to two 3 x 3 neighborhood convolutions on each stage in parallel. Both neighborhood operators operate on the same image input, but can perform different neighborhood operations. Larger neighborhood convolutions can be achieved by decomposing an odd-sized neighborhood mask into a sequence of 3 x 3 convolutions. The neighborhood operators can be either arithmetic or Boolean and are performed identically on all locations in the image unless a region-of-interest is specified. Special features are provided to prevent inaccurate computations on the image borders.

Multi-resolution pyramids can be constructed by selecting the "squeeze" or "expand" options as an image is stored or written from a buffer. In the former case, each 2 x 2 neighborhood of the input image is sampled and written to the output image resulting in an image half the resolution of the original. This process can be repeated to generate successively smaller resolution images. Expanding an image involves the opposite operation by pixel replication and generates successively larger resolution images.

Sequential processing works on a set of multiple images, e.g. sequences of images over time, a stereo pair of right and left images, or multi-resolution images.

When performing Boolean processing, each pixel of information is considered to be composed of eight independent bit planes, which are operated upon simultaneously. The neighborhood operators can be applied in a Boolean mode, where the output is the combination of the 3 x 3 neighborhood using local operations on each of the eight bit planes.

PIPE programs are written on a host computer using a software package which is an iconic representation of the hardware to generate microcode. Programs are executed by downloading the microcode instructions to PIPE. PIPE interfaces to a host computer through a board called ISMAP. Following feature detection performed on PIPE, ISMAP performs feature extraction, i.e., the iconic image of features, in which symbol values are indexed by image location, is mapped into a list of feature values, where associated with each feature is the set of image locations containing that feature. This is called a histogram. Therefore, image location is indexed by feature value.

6. Compression Techniques Implemented on PIPE

A number of data compression algorithms have been developed and demonstrated on PIPE. These include grey-level quantization, non-maxima suppression, foveal-peripheral simulation, image differencing, histogram slicing, binning, Laplacian pyramids, Poisson interpolation, and linear predictive coding. The following briefly describes each of these methods.

One way of measuring the effectiveness of image compression algorithms is by examining the entropy. Entropy is a measure of the randomness of grey values in an image. For image coding applications, entropy gives a lower bound on the average number of bits per pixel required to code an image using a compact code [2]. Therefore the lower the entropy, the greater the compression. In several of the compression algorithms to be described below, the entropy of the compressed image is given. The entropy value for each algorithm is obtained by applying the algorithm to a test image, and measuring the entropy of the resultant image. The average number of bits per pixel suggested by the entropy measure would be nearly obtained if a compact coding scheme, such as Huffman coding, were applied to the resultant image. To obtain the compression ratio, we multiply the entropy measure by the number of pixels in the image, and compare that with the number of bits required to represent the original image.

Grey scale quantization involves reducing the resolution of each pixel in the image. As represented on PIPE, an image pixel contains 8 bits. However, image resolution and contrast remain acceptable when three or even four low-order bits are dropped. Thus the number of bits required to transmit an image can be reduced by 37.5% or 50% respectively. The concept of "dropping n bits" is easily implemented on PIPE; the image is passed through a look-up table which shifts each grey-scale value right by n positions and then enhances the output by shifting the resultant image left by n positions. For example, dropping 3 bits of a pixel having grey value 52 would result in a quantized grey level of 48. The update rate of this method is one cycle or 1/60th of a second. The entropy measurement is dependent on the number of low-order bits being dropped. It ranges between a value of 6.74 for a full resolution image to 3.10 for an image in which four bits have been dropped.

Non-maxima suppression is an image processing method which results in a binary edge image in which all edges are one pixel wide. On PIPE, a Sobel edge operator is applied to the input image using the neighborhood operators. Edge magnitude and direction are extracted by using a two-valued function look-up table. The edge directions are quantized into eight discrete bins, and by using PIPE's region-of-interest operator, all edge points that are not locally maximal in the given gradient direction are eliminated. The update rate for this algorithm is two cycles (1/30th of a second), and the entropy measure is 0.37. The compression ratio is very high since only one bit of information per pixel is required.

The foveal-peripheral simulation is based on the biology of human vision. In humans, there is a very small area of acute vision, surrounded by areas of degraded vision. Two methods were written based on this idea. In the first (Figure 1a), a square window of the image is displayed with full 8 bit resolution. This square is surrounded by concentric window bands containing 6 bits, 4 bits, 2 bits, and finally 1 bit of resolution. The full resolution window can be repositioned and resized to meet the user's requirements.

The second foveal-peripheral algorithm utilizes the concept of multi-resolution image processing. In multi-resolution processing, a full sized image is successively sampled and reduced in resolution by a factor of two. Thus a 256 x 256 image is reduced to a 128 x 128 image which is reduced to a 64 x 64 image, etc. Using this technique, an arbitrarily sized and positioned square window is displayed at its full resolution (Figure 1b). It is surrounded by a window band obtained from its next level of resolution which in turn is surrounded by a band at the next level in the pyramid. Each level of the pyramid is successively more blurred, but requires fewer bytes of information.

In both methods, image masks representing the size and position of the window of interest and its surrounding bands are generated on either the host computer or the high level processor and are downloaded to PIPE. In the implementation of the first method, the input image is passed through a number of grey scale quantization tables, and the full resolution image and the lower resolution bands are logically combined to produce the reconstructed output image. The implementation of the second method involves generating two successively lower resolution images from the input. During reconstruction, the 128 x 128 resolution image is expanded and blurred using a Gaussian filter; the 64 x 64 resolution image is expanded twice and combined according to the regions defined by the downloaded image mask.

The entropy and the compression ratio in both instances are dependent on the size of the full resolution window of interest. For a square window measuring 60 x 60 pixels, the entropy measure is 2.02 for the first implementation and 4.05 for the second. The update rates are two and four cycles respectively.

Image differencing is an effective compression technique when there is relatively little motion between successive scenes in a sequence of images. For a stabilized camera mounted on a vehicle, the distant background will appear not to move between successive images, while the foreground will appear to move. The difference image is generated by subtracting an image at time t_{n-1} from the image at time t_n. All stationary regions of the image are eliminated and only areas of motion are visible. Thus much less information than the full image sequence need be transmitted. The full image sequence can be reconstructed from the difference images. In practice, good results have been achieved by transmitting a full image every 8 seconds, and difference images every 1/30th of a second. At reconstruction, the difference images are summed with the originally transmitted image to form a sequence of full images as follows:

$$t_0+(t_1-t_0)+(t_2-t_1)+\cdots+(t_n-t_{n-1})=t_n.$$

The update rate for this method is one cycle and a typical entropy measure for a difference image is 2.01.

Image compression and reconstruction using decimation and Poisson interpolation [8] is achieved on PIPE by passing the input image through a Laplacian neighborhood operator and thresholding the resultant image such that strong edges retain their grey level value while homogeneous regions are mapped to zero. This decimated image is transmitted to the remote operator station. Reconstruction involves multiple iterations of interpolation of the decimated image in regions which had been homogeneous. This selective interpolation is achieved using PIPE's region-of-interest operator. The update rate for the decimation is two cycles, and the update rate for the reconstruction is N cycles where N represents the number of iterations performed. Good results have been achieved by using six iterations.

Linear predictive coding is the process by which the greyscale values at each pixel in two sequential images are used to predict the greyscale value at a future point in time. Theoretically, this prediction is based on the assumption that the intensity values at any given pixel location vary only slightly over time, when the length of time is relatively short. Since the time between sequential images is small, the extrapolation of the greyscale values using a linear function is assumed to provide an accurate prediction.

Implementing this concept on PIPE requires storing an image frame in a buffer and then holding it until the next sequential image is stored in another buffer. The older image is multiplied by two and the most recent image is subtracted from it. The images are scaled down before calculations to prevent overflow and the results are scaled back up. The delay time for the algorithm is 1/10 second (6 cycles).

Binning is a term used to describe a general class of methods also known as histogram transformation. Once the histogram of an image is obtained, ranges of greyscale values can be grouped together to a single value based on a number of different criteria. Three methods of performing this grouping were investigated. One method was to construct each bin from an equal range of greyscale values. Another method was to have each bin contain an equal integral portion of the histogram curve. Histogram slicing, the third method, is accomplished by dividing the histogram of an image according to the most populous greyscale values.

The histogram of an image can be produced on PIPE. The buffered image is passed to the ISMAP board which sums the values of each occurrence of greyscale in the image. A symbolic list is produced which can be passed to the host computer for subsequent processing. This symbolic histogram list is used to generate a lookup table to map greyscale values into any arbitrary number of bins based on one of the methods mentioned above. The lookup table is sent back to PIPE so that subsequent images are binned. The frequency of the calculation of this lookup table is dependent on how rapidly the distribution of the greyscale values in an image change from frame to frame. For example, given that the scene rarely changes from being an outdoor open-terrain image with only small variations in lighting, the calculation may only have to occur once. The amount of time to transform an image on the PIPE is 1/60 second. The entropy of the test image after it has been transformed is dependent on the number of bins that are specified; for histogram slicing using seven bins, the entropy of the test image is 2.31. Binning of an image can also be used to obtain additional compression subsequent to performing some other types of compression techniques.

The Laplacian pyramid method of encoding an image [1] is based on both predictive and transform methods of image compression. The prediction of a pixel's value is based on a local Gaussian-weighted average of surrounding pixels. These predicted values are subtracted from the original values so that a measure of the predicted error is produced. The result is that only the predicted error, which requires fewer bits to be represented, and the low-pass filtered image, which can be subsampled, need to be encoded. This concept is repeated to produce a pyramid-like data structure.

The encoding technique is implemented on the PIPE. The first level of the Laplacian pyramid is constructed using the following method (Figure 2). The full resolution image is convolved with a Gaussian filter to produce the image G00, and the resulting image is subsampled. The subsampled image, G10, is then expanded back to full resolution. In order to compensate for the pixel replication that occurs on PIPE during expansion, the expanded image is masked with a template pattern of zeros and ones so that three out of every four pixels are set to zero [9]. The masked image is then convolved with a Gaussian filter and then multiplied by four to produce image G11. G00 is then subtracted from G11 to produce the Laplacian for the full-resolution image, L00. The entire process is repeated using G10 to produce L11 and so on until L44 is produced.

The five levels of the pyramid plus G50 are used to reconstruct the original image, also on the PIPE. Starting with G50 (Figure 3), the lowest resolution image is expanded, masked with the template, convolved with a Gaussian filter, and then added to the Laplacian image on the level below it. This process is repeated until all images have been added together to produce a reconstructed original image. The entire encoding and decoding of a single frame requires 51 cycles. The total entropy of all 6 transmitted images is 4.5.

7. Conclusion

This paper has described several algorithms that have been implemented as possible candidates for a hybrid video compression system to be used for remote driving of a ground vehicle. The algorithms have been implemented on the PIPE real-time image processing machine, and the implementations have also been described. The PIPE has been integrated with a remote control vehicle system and these algorithms were evaluated by means of real-world remote driving experiments. The results of these experiments are presented in a companion paper [3]. Briefly, these experiments have shown that remote vehicle driving is difficult enough without degrading the imagery through compression algorithms. The degraded imagery makes driving even more difficult.

Some difficulties we found in driving in cross country terrain using either the full video or the compressed video were (1) global relative vehicle location is very difficult for the driver to obtain, (2) the orientation of the local ground surface is very difficult to obtain, (3) ditches, gullies, and other obstacles are difficult to distinguish, (4) range of objects from the vehicle are difficult to determine.

It appears that performing compression by transmitting images at a rate of at most a few per second, and then providing a realistic video simulation to the operator, may be one of the most effective ways of performing video compression.

References

1. Burt, P. J. and Adelson, E.H. "The Laplacian pyramid as a compact image code." *IEEE Transactions on Communications*, vol. COM-31, no. 4, 1983, 532-540.

2. Gonzalez, R.C. and Wintz, P. *Digital Image Processing*. Addison-Wesley Publishing Co., Reading, MA, 1977.

3. Herman, M., Chaconas, K., Nashman, M., and Hong, T.-H. "Video Compression for Remote Vehicle Driving." *Proc. SPIE Advances in Intelligent Robotics Systems: Mobile Robots III* Cambridge, MA, November 1988.

4. Kent, E.W., Shneier, M.O. and Lumia, R. "PIPE (Pipelined Image Processing Engine)." *J. Parallel and Distributed Computing*, 2, 50-78, 1985.

5. H. E. Knutsson, R. Wilson, and G. H. Granlund, "Anisotropic nonstationary image estimate and its applications : Part I - Restoration of noisy images," *IEEE Transactions on Communications*, vol. COMM-31, 388-397, 1983.

6. M. Kunt, A. Ikonomopoulos, and M. Kocher, "Second generation image coding techniques," *Proceedings of the IEEE*, vol. 4, 549-574, 1985.

7. R. Lippmann, "Continuous movement regeneration in low-frame-rate aerial images," *Proc. IEEE International Conference on Electronic Image Processing*, 194-198, 1980.

8. McMillen, R. Personal communication. Hughes Aircraft Co., 1988.

9. Singh, Ajit. "Image processing on PIPE." Philips Laboratories - Briarcliff, North American Philips Corporation, TN-87-093, July, 1987.

10. R. Wilson, H. E. Knutsson, and G. H. Granlund. "Anisotropic nonstatationary image estimation and its applications : Part II - Predictive image coding," *IEEE Transactions on Communications*. vol. COMM-31, 398-404, 1983.

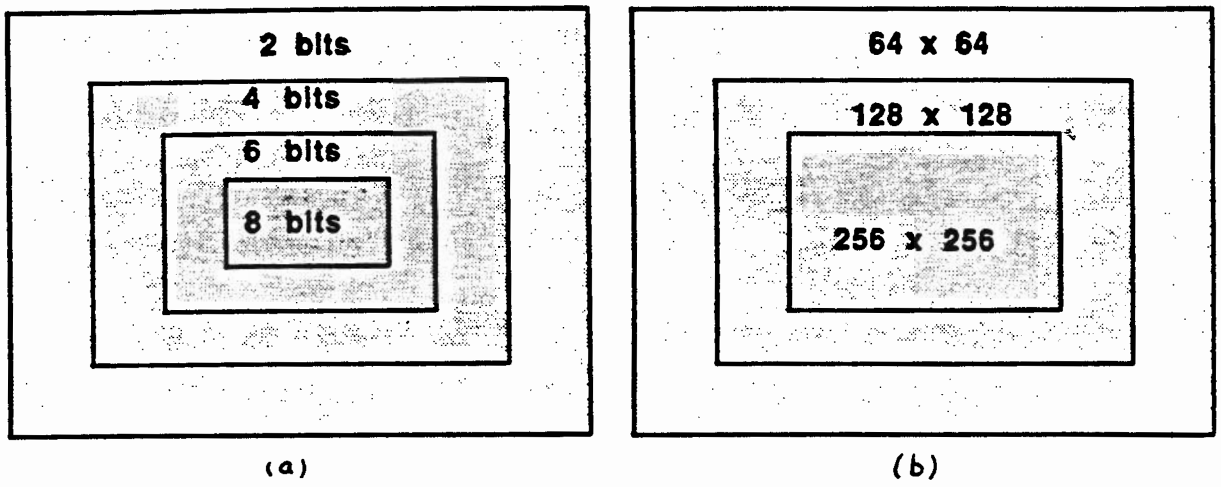
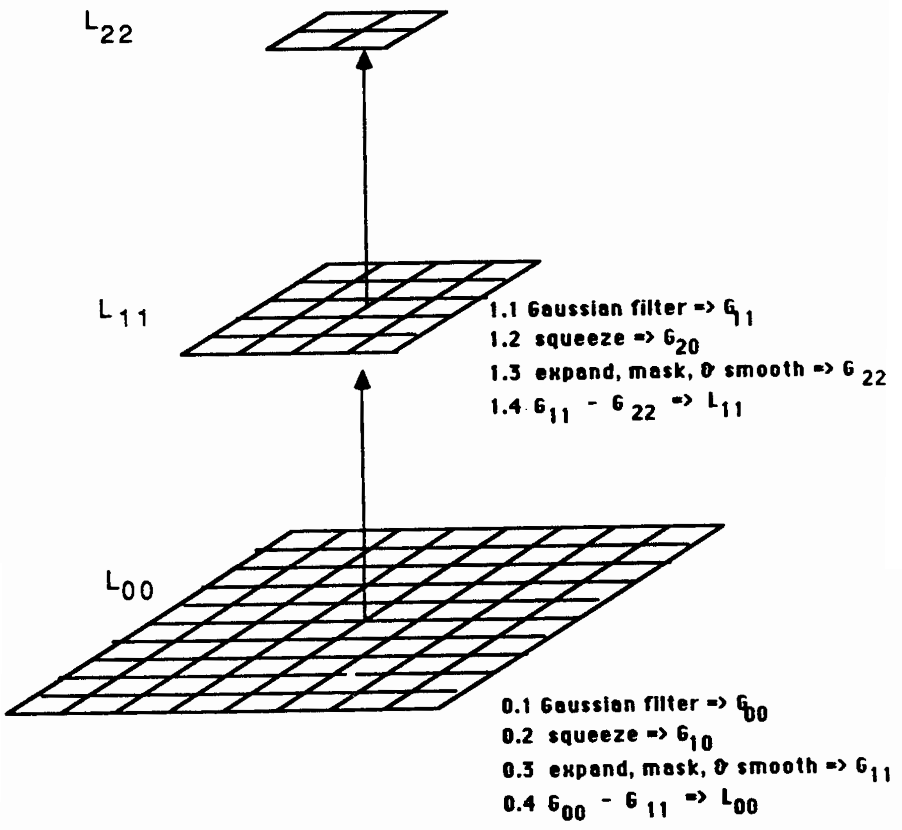


Figure 1. Two compression techniques based on foveal-peripheral vision.



L_{22}
L_{11}
1.1 Gaussian filter => G_{11}
1.2 squeeze => G_{20}
1.3 expand, mask, & smooth => G_{22}
1.4 G_{11} - G_{22} => L_{11}
L_{00}
0.1 Gaussian filter => G_{00}
0.2 squeeze => G_{10}
0.3 expand, mask, & smooth => G_{11}
0.4 G_{00} - G_{11} => L_{00}

Figure 2. Construction of Laplacian pyramid.

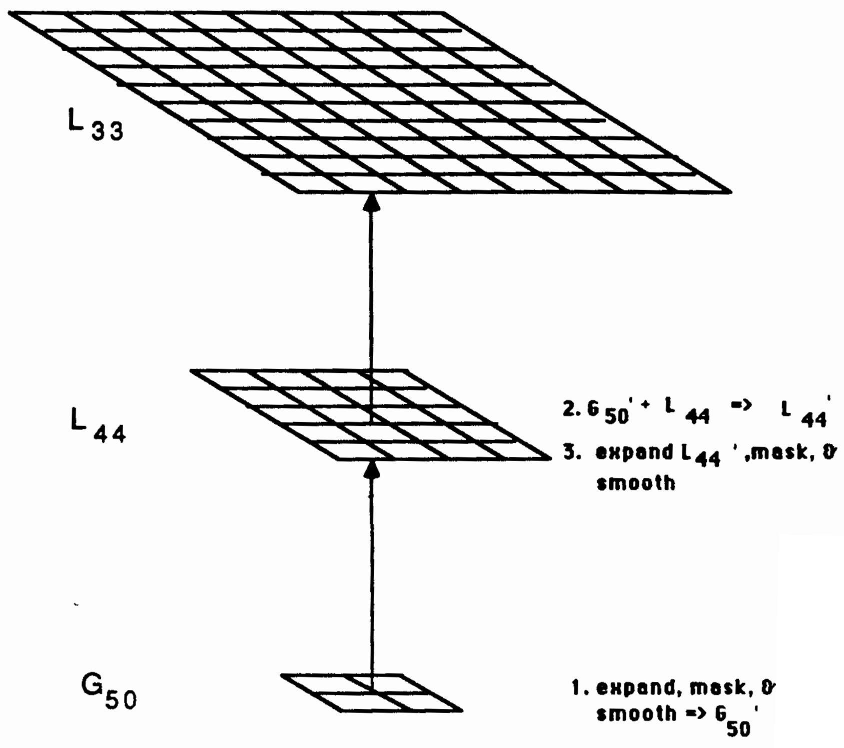

L_{33}
L_{44}
2. $G_{50}' + L_{44} \Rightarrow L_{44}'$
3. expand L_{44}', mask, & smooth
G_{50}
1. expand, mask, & smooth $\Rightarrow G_{50}'$

Figure 3. Reconstruction of original image from Laplacian pyramid.