CUSTOM MADE PYRAMIDS

Shmuel Peleg
Orna Federbusch

Dept. of Computer Science
The Hebrew University of Jerusalem
91904 Jerusalem, Israel

ABSTRACT

Pyramids are data structures used to store and process images in multiple resolutions. The bottom level of the pyramid usually stores the original image, while higher levels of the pyramid represent the image in reduced resolutions. The reduction of image size between levels is usually by a factor of two, and identical factors are used to reduce each level to the higher level.

This paper describes a scheme that enables the pyramid to be adapted to different applications. The pyramid can be reduced arbitrarily between levels from any given rectangular size to any desired rectangular size. Also, the reduction factor can be made adaptive to region properties, enabling smooth regions to be reduced more than "busy" regions.

1. Introduction

In the most common pyramid scheme each pixel in a given level represents a block of pixels from the level below. A survey on pyramids can be found in [1].

In this paper a method is described to construct pyramids of arbitrary size at each level, so that resolution can be reduced as needed and not only by fixed powers of two. Further, the reduction from level to level does not have to be uniform over the entire image, and can vary based on local image properties.

The basic step in the proposed pyramid construction is a spatial resampling technique used in graphics [2,3]. This transformation is carried out in two passes, where the first pass affects only rows, and the second only columns. As each pass processes a one dimensional entity, the method will be described in one dimension.

The spatial resampling has the desirable property that all input pixels fully contribute to the output pixels with no gaps or overlaps, therefore minimizing sampling artifacts.

1.1. One Dimensional Contraction-Expansion

Given is a vector $V = (v_1, \ldots, v_N)$ of N pixels, to be changed to a vector $W = (w_1, \ldots, w_K)$ of length K. Each pixel in V contributes $f = K/N$ to pixels in W, such that the total contribution to each $w_i \in W$ is one. Also, if v_i contributes to w_j, and $k > i, l < j$, then v_k does not contribute to w_l (no cross-overs). For example, to reduce (v_1, v_2, v_3, v_4) to (w_1, w_2, w_3), $f = 3/4$, and the reduction is as shown in Figure 1.

This method has the following features:

This paper has been supported by a grant from the Israel Academy of Sciences.
Author's electronic address: peleg@hujics.bitnet

NATO ASI Series, Vol. F 25
Pyramidal Systems for Computer Vision
Edited by V. Cantoni and S. Levialdi

$$w_1 = \frac{3}{4}v_1 + \frac{1}{4}v_2$$

$$w_2 = \frac{1}{2}v_2 + \frac{1}{2}v_3$$

$$w_3 = \frac{1}{4}v_3 + \frac{3}{4}v_4$$

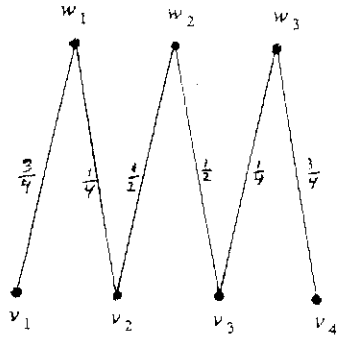


Figure 1: Reduction from a level of size 4 to a level of size 3.

- Each pixel of the input vector has equal contribution to the output vector.
- There are no border effects, as there is no need to use values of pixels outside the image boundary.

1.2. One Dimensional Weighted Resampling

In the resampling of the previous section all input pixels had the same contribution f to the output pixels. The algorithm can be adapted to the case where contribution can vary, and each pixel v_i is assigned a "forward weight" $f(v_i) > 0$, such that $\sum_{i=1}^{N} f(v_i) = K$, where K is the number of the pixels in the output vector W.

We construct W using the principle of the previous section, except that each pixel v_i has his own weight $f(v_i)$. For example, when $V = (v_1, v_2, v_3, v_4)$, with weights $f(v_1) = 1$, $f(v_2) = 1$, $f(v_3) = 0.5$, $f(v_4) = 0.5$, is to be reduced to $W = (w_1, w_2, w_3)$, the reduction is as shown in Figure 2.

$$w_1 = v_1$$

$$w_2 = v_2$$

$$w_3 = \frac{1}{2}v_3 + \frac{1}{2}v_4$$

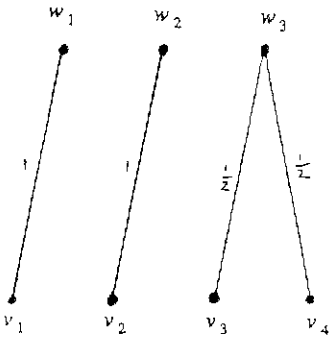


Figure 2: Weighted reduction, with weights as marked, from a level of size 4 to a level of size 3.

2. Pyramid Construction

By use of the contraction method in Section 1.1, an input image of any size $M \times N$ can be reduced to an output image of any desired size $K \times L$. Through repetition of this process (each level being of any desired size) a complete pyramid can be built. In Figure 3 a one-dimensional example of the construction of a three level pyramid is presented. Note that the contribution of bottom level to top level varies with the properties of intermediate levels. Higher levels are in general weighted averages of bottom level pixels, where the overlap between regions included in the averages increases with the number of intermediate levels.

This pyramid scheme has the following properties:

- Each pixel has equal contribution to the level above.
- Reduction can be made from a level of any given rectangular size to any desired rectangular size.
- Pixels in higher levels of the pyramid are weighted averages of pixels in the bottom level, with the possibility of overlap: a pixel in the bottom level can contribute to several pixels in any given higher level. This property is similar to that of Burt's overlapping pyramids [4].
- The degree of overlap depends mostly on the number of intermediate layers.

An example of an image embedded in such a pyramid is shown in Figure 4.

These pyramids can be used for same applications as Burt's overlapping pyramids [4]. We also follow Burt by construction of "Laplacian Pyramid", where each level L_i in the Laplacian Pyramid is the difference between levels G_{i+1} and G_i of the original pyramid. Level G_{i+1} is enlarged thru the same resampling technique to the size of level G_i before performing the subtraction. As in Burt's Laplacian Pyramid, each level in L_i is a band-pass filter as a difference of two low pass filters.

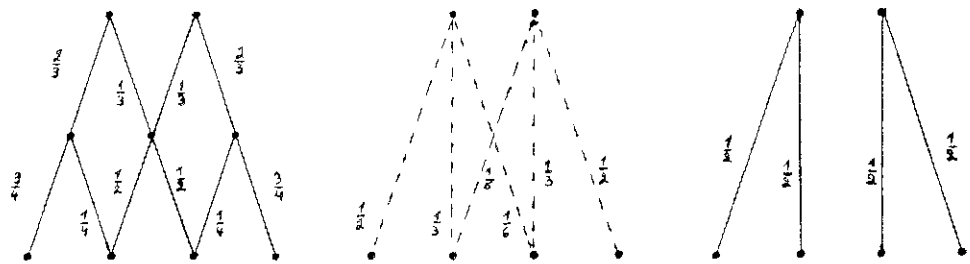


Figure 3: One dimensional pyramid.

a) Three levels, 1-D pyramid of size 4-3-2. A number on an arc is the contribution of the lower pixel to the upper pixel.

b) The overall contribution of the lower level to the top level for the pyramid (a).

c) A pyramid that reduces from base of size 4 to size 2 directly.

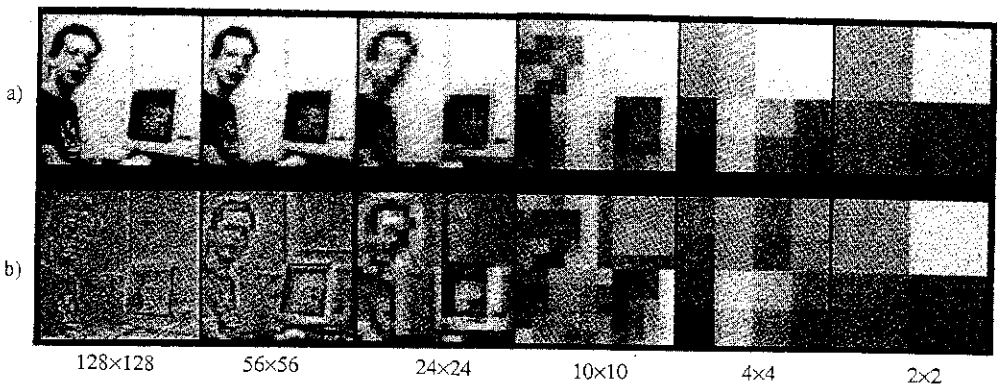


Figure 4:

An image embedded in a 6-level pyramid with level sizes as marked. Levels were enlarged to original size for visibility.

a) Levels of the pyramid. The leftmost image is just the original; successive pictures are the reduced images at higher levels.

b) Levels of the Laplacian Pyramid (see text), where each level is a band-pass filter. Each image is the difference between the pyramid image of same size, and the image at the level above (expanded). Zero is represented as gray, negative parts are lighter and positive parts are darker.

3. Adaptive Resampling

The spatial resampling scheme employed in the pyramid construction allows us to vary the resampling at each region of the image, rather than the uniform resampling common in existing pyramid schemes. Non-uniform sampling has the advantage that regions with "interesting" activity can be resampled more densely than "uninteresting" regions. This effect can also be observed in human perception where a segment with higher "busyness" seems longer than a straight line segment [5], as in Figure 5. This might indicate that more storage is used in the brain for the divided segment.

To achieve the effect of adaptive sampling, we need to use a "busyness" measure. Measures could be selected to best fit an application, and in this paper we use the smoothed absolute value of the Laplacian.

3.1. One Dimensional Adaptive Pyramid

In the one dimensional case, we first apply a Laplacian to the vector V of length N, take its absolute value, and smooth the result. The weights are then normalized to get the "forward weights" f_i $(1 \le i \le N)$ such that $\sum_{i=1}^{N} f_i = K$, where K is the number of pixels in the output vector W. The weighted resampling of Section 1.2 is then applied. While resampling, each pixel at the output vector W obtains a "backward weight" b_i $(1 \le i \le K)$, which indicates how many pixels of the lower levels contribute to it. $\sum_{i=1}^{K} b_i = N$,

Figure 5:
An illusory distortion in which the divided space appears longer than the undivided space (Oppel-Kundt illusion).

and the b_i weights will be used as weights when reconstructing a level from the higher levels.

Example 1: In Section 1.1 (Figure 1) we used equal contributions of first level pixels to the second level:

$$f(v_1) = f(v_2) = f(v_3) = f(v_4) = \frac{3}{4}.$$

This resulted in the following construction for the second level: $w_1 = \frac{3}{4}v_1 + \frac{1}{4}v_2$, $w_2 = \frac{1}{2}v_2 + \frac{1}{2}v_3$, $w_3 = \frac{1}{4}v_3 + \frac{3}{4}v_4$. As w_1 was built from all of v_1 and $\frac{1}{3}$ of v_2, its "backward weight" $b(w_1) = 1\frac{1}{3}$; w_2 was built from $\frac{2}{3}$ of v_2 and $\frac{2}{3}$ of v_3, yielding $b(w_2) = 1\frac{1}{3}$; similarly, $b(w_3) = 1\frac{1}{3}$. Therefore, when building V back from W, equal weights will be used.

Example 2: In Section 1.2 (Figure 2) we had a weighted construction with the first level weights:

$$f(v_1) = 1,\ f(v_2) = 1,\ f(v_3) = 0.5,\ f(v_4) = 0.5.$$

The resulting construction has been $w_1 = v_1$, $w_2 = v_2$, $w_3 = 0.5\,v_3 + 0.5\,v_4$. In this case we have $b(w_1) = b(w_2) = 1$ as each is being supported by one element in V, while $b(w_3) = 2$ as w_3 is being supported by both v_3 and v_4. Using the backward weights, when the V level will be expanded from the W level we will use $v_1 \leftarrow w_1$, $v_2 \leftarrow w_2$, $v_3 = v_4 \leftarrow w_3$.

3.2. Two Dimensional Adaptive Pyramids

We build a two dimensional adaptive pyramid using the same principle as the one dimensional case of the previous section. However, if each row or column will be processed independently, the image will loose its structure in the sense that connectivity will not be preserved. This can be caused when the weights of each row or column are computed independently of the neighboring rows/columns. As the loss of connectivity structure is unacceptable, we select a scheme where all rows and columns use the same weights. In this scheme, row weights are computed based on average busyness for all rows, and column weights are computed based on average busyness for all columns. The following examples will show the

reduction from 4×4 to 2×2, given the computed busyness measures.

Example 3: Given the following busyness matrix

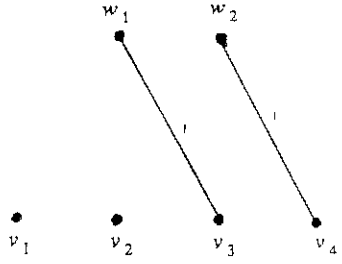
$$\begin{bmatrix} 0 & 0 & 2 & 2 \\ 0 & 0 & 2 & 2 \\ 2 & 2 & 0 & 0 \\ 2 & 2 & 0 & 0 \end{bmatrix}$$

The average row busyness is (2 , 2 , 2 , 2), yielding row weights of (0.5 , 0.5 , 0.5 , 0.5). Same is true for the column weights. This will result in a reduction of the 4×4 image to a 2×2 image, where each output pixel is the average of a 2×2 block in the input image.

Example 4: Given the busyness measure

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 2 & 2 \end{bmatrix}$$

the average row busyness is (0 , 0 , 2 , 2) yielding row weights of (0 , 0 , 1 , 1), and the same happens for the columns. The row and column reduction will both be:



and given the image

$$\begin{bmatrix} v_{11} & \cdots & v_{14} \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ v_{41} & \cdots & v_{44} \end{bmatrix}$$

the reduction will be:

$$\begin{bmatrix} v_{11} & \cdots & v_{14} \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ v_{44} & \cdots & v_{44} \end{bmatrix} \xrightarrow{\text{row reduction}} \begin{bmatrix} v_{13} & v_{14} \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ v_{43} & v_{44} \end{bmatrix} \xrightarrow{\text{column reduction}} \begin{bmatrix} v_{33} & v_{34} \\ v_{43} & v_{44} \end{bmatrix}$$

The effects of embedding an image in an adaptive pyramid can be seen in Figure 6.

4. Concluding Remarks

A pyramid scheme which is adaptive to the reduction of resolution both between levels and also spatially within levels has been presented. Such pyramids can be of use when many levels of resolution are

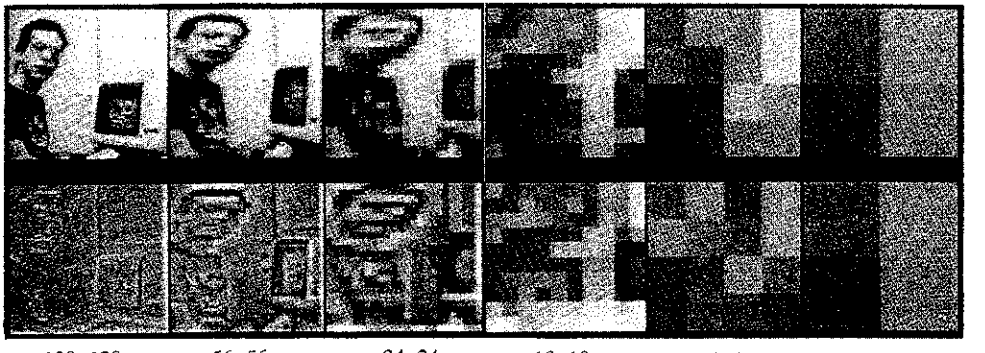


Figure 6:
6-level adaptive pyramid, with level sizes as marked. Levels were enlarged to original size for visibility.

a) Levels of the adaptive pyramid.

b) Levels of the Laplacian Pyramid.

needed, and may help in compression when using denser resampling for busy regions.

5. References

1. A. Rosenfeld (Ed.), *Multiresolution Image Processing and Applications*, Springer, 1984.

2. K. Fant, A nonaliasing, real-time spatial transform technique, *IEEE Computer Graphics and Applications*, Vol 16, January 1986, 71-80

3. E. Catmul and A. R. Smith, 3D transformations of images in scanline order, *Computer Graphics (Proc. SIGGRAPH 80)*, Vol 14, July 1980, 279-289.

4. P. J. Burt and E. H. Adelson, The Laplacian pyramid as a compact image code, *IEEE Trans. Communications*, Vol COM-31, 1983, 532-540.

5. S. Coren and J. S. Girgus, *Seeing is Deceiving*, Lawrence Erlbaum Associates, 1978.