Layered representations for image coding

Edward H. Adelson

Media Laboratory
Massachusetts Institute of Technology, Cambridge, MA 02139
e-mail: adelson@media-lab.media.mit.edu

1.0 Introduction

In order to represent a television image sequence with a minimal number of bits, it is necessary to perform efficient image coding. Transform coding, subband coding, and vector quantization are examples of popular techniques that have been applied to the coding of still and moving images. In the case of moving image sequences, these techniques are often combined with frame-differencing or motion compensation, to take advantage of the temporal structure of image sequences.

Any image coding technique contains an implicit vocabulary and image model; that is, it offers a way to describe images. For example, in a discrete cosine transform technique, an image is described as a sum of cosine basis functions, and this forms the implicit image model. If motion compensation is used for moving images, then the implicit image model includes the idea that regions of the image move coherently across time.

There are three interrelated parts to the design of an image coding system: the encoding process, the representation, and the decoding process. The choice of representation is the central determinant of the system as a whole: the encoding process converts the input images into the chosen representation, and the decoding process converts the representation into an array of pixels for the display. The representation must be chosen before the encoding or decoding procedure can be designed.

I describe a new image representation based on the concept of "layers" that captures several important aspects of images and may substantially improve the efficiency and flexibility of image coding for many image sequences.

The layered representation also allows for high-quality frame-rate conversion, so that, for example, an original image sequence shot at 50 hz, can be converted to a frame rate of 60 hz without introducing artifacts.

When an image sequence is represented as layers, it also becomes possible to perform interesting editing and post-production effects that would be impossible with ordinary representations.

2.0 Limitations of existing motion compensation techniques:

In standard motion compensation methods, the contents of one frame is estimated by applying a spatial transformation to the contents of a previous frame. One popular technique is block-matching, in which a given block of pixels in one frame is estimated as the rigidly translated block from a previous frame. If the translated block correctly estimates the corresponding block in the new frame, then it is only necessary to send the translation vector. If the estimate is imperfect, then one must also send an error signal to correct it. In more sophisticated methods, the mapping from one frame to another may involve smooth distortions, including affine transformations or rubber-sheet warps.

These techniques are founded on the assumption that the optic flow field is smooth and single-valued. This assumption is frequently violated. For example, when a foreground object occludes a background object, the flow field is discontinuous at the occlusion boundary, and information at the occlusion boundary disappears while information at the “disocclusion” boundary appears. The occlusion and disocclusion information cannot be captured with the motion compensation, and must be explicitly coded with error signals; this leads to inefficient coding.

The standard motion assumptions are also violated in the case of transparent motion, where there are two valid motion vectors at the same image location. Transparent motion occurs when, for example, a scene is viewed through a windowpane, and dirt or specular reflections on the pane are combined with the scene viewed through the pane. Another form of transparency occurs when the edge of a foreground object is blurry, due to optical blur or motion blur, and the blurred edge combines with the background in a transparent manner. There can be two different valid motions in the overlapping blurred region.

3.0 Layered representations

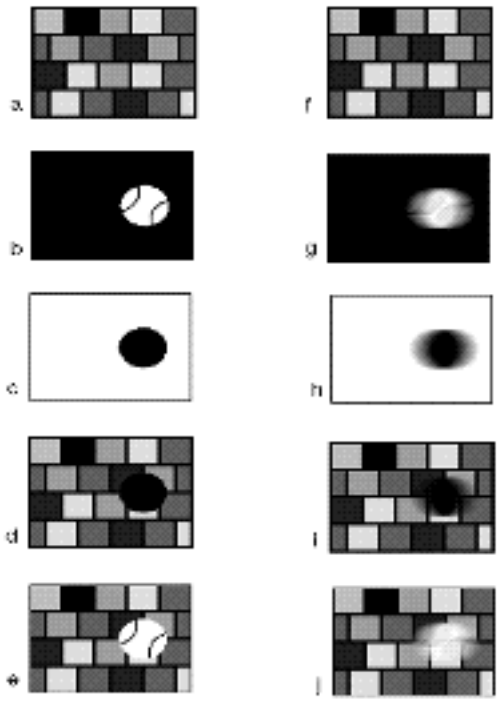
A more sophisticated image model is based on the notion of image layers: the image is considered to consist of a set of overlapping layers, with some ordering in depth. This is the representation used by traditional cel animators, who paint a set of images on transparent plastic sheets and then stack them up to produce the final images. It is, after all, much easier to paint a background once and move the characters in front of it than to repaint the entire scene for every frame. The cels contain areas of varying color and transparency, mimicking the image formation process of objects in the natural world. This is also the model used in many computer animation programs, in which multiple image layers are defined, and each layer has its own motion, its own depth, and its own pattern of color and intensity, and its own pattern of transparency.

The concept of layers is used in computerized compositing, as applied in computer graphics. This method is well-known (see [1][2]). Figure 1(a) shows a background image, $E_0(x,y)$. Figure 1(b) shows in image of a baseball, $E_1(x,y)$, which is to be pasted in front of the background. The pasting is achieved with a compositing mask, $A_1(x,y)$ shown in (c). In the simplest appraoch, this is a binary mask, with values 1 and 0, shown as white and black. The background is multiplied by the mask to remove the region hidden by the baseball, producing the modified background shown in (d). Then the foreground is multiplied by the inverse of the mask, $(1-A_1(x,y))$, to select the baseball itself; this is added to the

background to produce the final image shown in (e), which may be called $I_1(x,y)$. Thus the compositing occurs according to the equation:

$$I_1(x,y) = E_0(x,y)A_1(x,y) + E_1(x,y)(1-A_1(x,y)). \quad (1)$$

Figure 1 . (a): The background image. (b) The foreground image. (c) The compositing mask. (d)The background after removal of the foreground region. (e) The final image. (f) The background image. (g) The foreground image under motion blur. (h) The compositing mask under motion blur. (i) The background attenuated by the compositing mask. (j) The final image.



In a more sophisticated approach to compositing, the mask can take on continuous values between 0 and 1. For example, consider the case where the baseball is blurred due to motion. In the blurred edge region, the image consists of an average of the background and the (blurred) baseball. The proportions of background and foreground depend on the distance from the edge. This situation can be captured in the same computational framework as before. Again, Figure 1(f) shows the background $E_0(x,y)$; (g) shows the foreground $E_1(x,y)$, which contains motion blur; (h) shows the compositing mask, $A_1(x,y)$, which contains the same motion blur; it takes on values between 0 and 1; (i) shows the

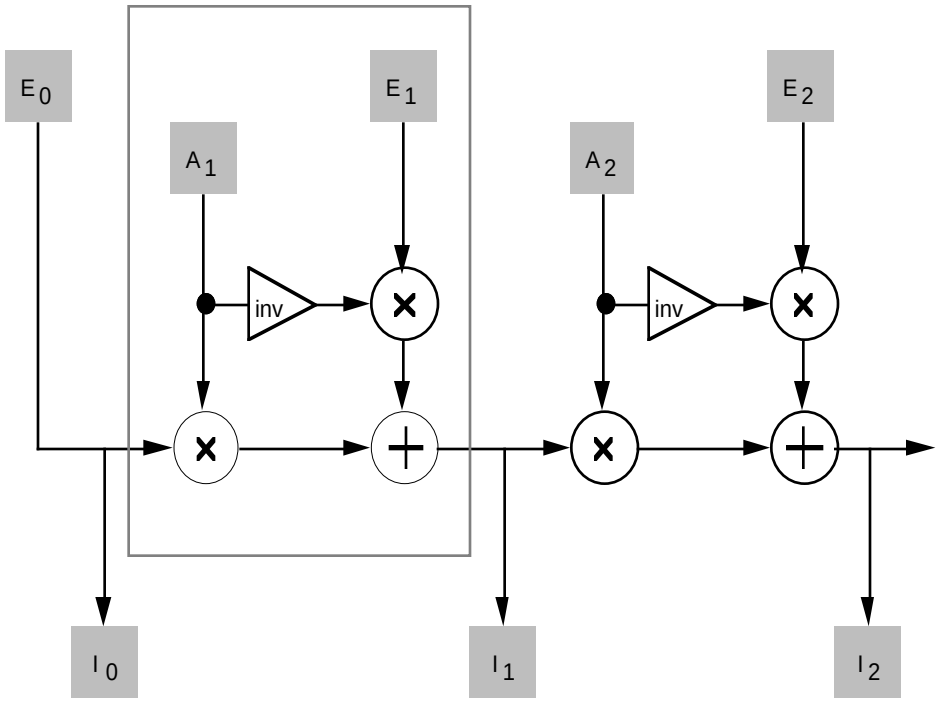
modified background, which is attenuated in varying amounts by the mask; and (j) shows the final composited image.

(Note that in computer graphics, the variable α is usually used, with the inverse of the convention used here for A. That is, $A = (1-\alpha)$.)

The same processing stages may be applied to combine a series of layers, as shown in Figure 2. The background image is E_0. It is combined with the foreground image E_1 under the control of the attenuation map A_1. The box labeled “inv” computes the inverse mask function, $(1-A_1)$. The result is the image I_1. This image may then be combined with the next layer, E_2, under control of the mask A_2, to produce the image I_2.

The basic compositing stage is enclosed in the dashed rectangle. One may repeat as many stages as desired.

Figure 2 A flow chart for compositing a series of layers.



This example illustrates how layers may be used to produce a single static image. The same concepts can be extended to produce a powerful and flexible method for encoding image sequences.

4.0 Layers for image sequences

Suppose that the baseball in the prior example is translating across the scene. Then it will produce the series of images shown in Figure 3 as (a1) to (a3). In order to synthesize this sequence, we can use the set of images shown in the left column, (b1), (c1), (d1), and (e1). (b1) is the stationary background; (c1) is the baseball image; (d1) is the mask; and (e1) is the velocity map that is applied successively to (c1) and (d1). After one application, the baseball and the mask are as shown in (c2) and (d2); after compositing they produce the image (a2). After a second application they are as shown in (c3) and (d3), leading to the image (a3).

In the proposed encoding method, the original image sequence, (a1) to (a3), could be represented with the set of maps (b1), (c1), (d1), (e1). Thus, rather than explicitly representing the sequence, onerepresents a set of component layers along with rules for their transformation and combination. This approach offers several advantages. The encoding can be efficient, since each of the components only has to be encoded once, as a still image, and the change from one frame to the next is captured in the very simple velocity

map. In addition it is possible to adjust the speed of the motion or to adjust the frame rate by changing the amplitude of the velocity map..

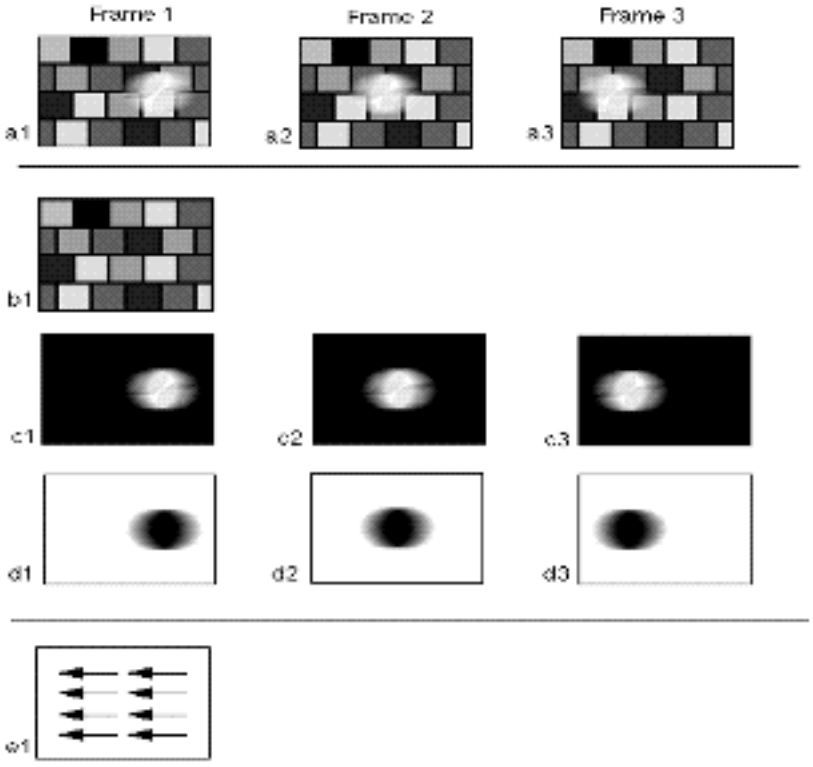


Figure 3 . Frames 1 to 3 form a sequence of images of a moving baseball seen against a stationary background. (a1) to (a3) are the final frames. (b1) is the background. (c1) to (c3) are the set of baseball images, and (d1) to (d3) are the set of masks. (e1) shows the motion map that is applied to the baseball and the corresponding mask in order to generate the image sequence

In the above example, the motion is a simple translation. One could also use an affine transformation in order to achieve rotation, dilation, and shearing, or one could use more complex warp maps to achieve arbitrary distortions.

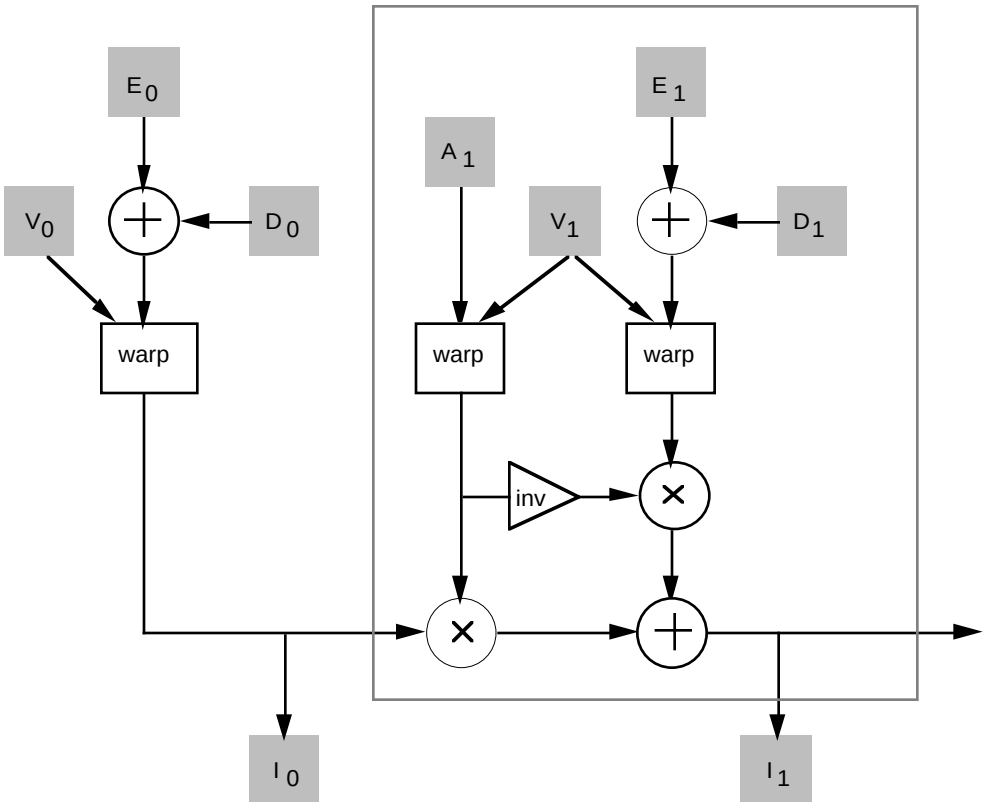
5.0 Delta maps

The velocity map will not always convey all the changes occuring to a given layer. Therefore it is useful to add information from a “delta map” (change map). This is an image that defines the temporal derivative at each point in a layer. Thus, in addition to undergoing a warp from one frame to the next, each layer can be combined with an additive correction.

As before, this change map can be scaled in order to interpolate or extrapolate to different points in time. (A multiplicative change map may also be used to account for temporal variations in contrast.))

A combined system incorporating velocity maps, V(x,y), and delta maps, D(x,y), is shown in Figure 4. From one frame to the next, the background E_0 is modified by the delta map D_0 and then warped by the velocity map V_0. This is then composited with the image E_1, which is itself added to D_1 and warped by V_1. The attenuation mask, A_1, is warped by the same velocity map, V_1.

Figure 4 . A flow chart for compositing that incorporates velocity maps, V, and delta maps, D.



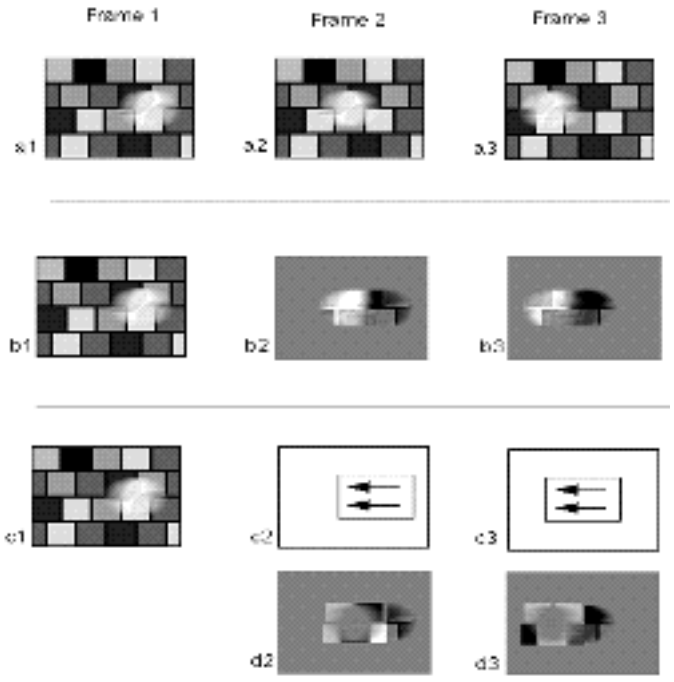
6.0 Comparison with conventional techniques.

It is illuminating to compare the layered approach with the more conventional approaches to coding image sequences. In Figure 5 the three frames are shown as (a1) to (a3). The tra-

ditional frame-difference method is shown in the middle row. First an initial frame is coded, shown in (b1); then the frame differences are coded for each succeeding frame, shown in (b2) and (b3).

The traditional motion-compensated method is shown in the bottom two rows. First the initial frame is coded, as in (c1); then the blocks shown in (c2) and (c3) are used to show regions that are considered to be in rigid translation. Finally the error images, (d2) and (d3), are used to correct for the failures of motion compensation.

Figure 5 (a1) to (a3): original image sequence. (b1): initial frame for frame-differencing. (b2), (b3): difference frames. (c1): initial frame for motion compensation. (c2), (c3): moving image blocks for motion compensation. (d2), (d3): error signals added to correct for motion compensation.



Neither the frame-differencing method nor the motion-compensated method uses a very accurate model of image formation, and for this reason the amount of corrective information that must be sent is quite large. In many cases, the layered format will therefore be more efficient. The layered format will have its greatest advantage in cases where the layered model accurately represents the image information over many frames, so that one can send a minimum of corrective information in the form of change maps.

One should also note that neither the frame-differencing nor the motion-compensated approach allow for good temporal interpolation. It is true that one can add partial amounts of the corrective signals, in an attempt to synthesize an intermediate frame, but in a traditional motion-compensated technique this will lead to poor results due to “ghosting.” The layered format can prevent these artifacts.

7.0 Specifying the layered format:

In a digital system we can attach a rich variety of image information to the layers. In addition to color and transparency, we can attach information about motion, blur, and depth. An example of such a format is as follows:

There are N layers, which are typically ordered in depth. Associated with each layer is a set of maps. These maps are defined at a set of discrete 2-D locations, (x,y), and a set of times, t. The locations would normally be the spatial sampling lattice and the times would normally be the frame times. Note, however, that it is not necessary that all of the maps be sampled on the same lattice in space and time. As long as interpolation rules are given, it is possible to synthesize the requisite values of the maps at any position and time. Thus the different maps may be sampled on different lattices in space and in time. (Indeed there is no need to define the map by a set of samples; it could also be defined by a function or a set of coefficient).

The basic maps are as follows:

E(x,y,t), the intensity map (where E may be a vector such as (R,G,B) in the case of color images).

A(x,y,t), the attenuation map, where A is between 0 and 1. Each layer multiplies the image intensities below it by A, and adds its own intensity E, scaled by (1-A).

D(x,y,t), a “delta map” (change map) that describes the temporal derivative of the points in E(x,y,t). This can also serve as an error signal to correct for failures of the other temporal variation (motion) to adequately describe the signal.

V(x,y,t), a velocity map, which tells how all points in the layer are warped over time. In some cases it will be possible to represent this map with a few parameters, as in the case of translation, or affine distortion.

In addition, the following optional information may be included:

C(x,y,t), a contrast change map, which specifies how the intensity map, E(x,y), should be multiplied to make the transition from one time to the next.

B(x,y,t), a blur map, which can be used to add motion-blur or focus-blur to each point in E(x,y,t). The blur map contains a set of parameters for a space-variant filter that is to be applied at each point in the image. In the simplest case the filter is a fixed kernel for the entire layer.

Z(x,y,t), a depth map.

N(x,y,t), a surface orientation map, coding the surface normals (also known as a “bump map”).

For efficiency one may wish to associate a bounding box with each layer as well; outside of the box all the values revert to the default, e.g. E=0; A=1; D=0; V=0, etc.

8.0 Temporal interpolation:

If a different set of maps are defined for every layer and every frame in time, then there might seem to be no need for the time dependent maps such as V(x,y,t), and D(x,y,t). But these maps have two uses. One is to do temporal interpolation between frames; the other is to allow efficient representation.

The virtue of temporal interpolation is this: if the image sequence is represented at one frame rate, but one wishes to display it an a different frame rate, then it is possible to synthesize intermediate frames. Given a velocity map and a delta map it is easy to smoothly interpolate to any intermediate time.

In addition these maps may be used for image data compression. It might be possible, for example, to send a sequence of 4 images with only a single intensity map for each layer, along with a single velocity map for each layer.

9.0 Special cases:

There are many phenomena in image sequences that lend themselves naturally to the layered representation. A shadow moving over a surface can be represented as a purely multiplicative layer, that is, a layer with values for A(x,y), but zero values for E(x,y). Conversely, a specular reflection such as the reflection in a window pane is a purely additive layer, with non-zero values for E(x,y) and a constant value for A(x,y). A change in contrast (such as occurs in a fade or with changes in lighting) can be expressed in terms of a layer with zero-valued E(x,y), and a time-varying A(x,y). It can also be expressed through a contrast change map, C(x,y).

10.0 Editing and special effects:

There are many interesting special effects that can be performed if an image has been converted into a layered format. Indeed, one of the most important class of special effects in film and video, known as “blue-screening” or “matting,” involved the creation of layers. In blue-screening, an actor is filmed performing in front of a blue screen. Later, all blue pixels are classified as belonging to the background, and all non-blue pixels are classified as belonging to the foreground. It then is possible to paste the image of the actor (the foreground) onto another background. This technique is widely used in the television and film industries.

The disadvantage of blue-screening is that it requires an artificial controlled environment. It would be prefereable to be able to segregate the actor from the background of any scene. Conversion of the image sequence into a layered format would allow one to manipulate existing footage. (Ideas along these lines are being explored by the Movies of the Future Group at the MIT Media Lab).

The layered format would also allow one to manipulate the motions of objects in the scene, so as to speed them up or slow them down, or change their trajectories. It would also allow one to synthetically add or remove objects from a scene, as is done now with blue-screening. It would also be a convenient format for combining natural image sequences with and synthetic image sequences generated by computer graphics.

11.0 The encoding process:

The layered representation and the procedure for decoding it are well-defined, and they constitute the core of the approach described here. But the encoding procedure is a separate problem and it is extremely challenging. The encoding process is non-unique: there are innumerable ways of representing a given image sequence within the constraints of the format described here. Some of the encodings will be useful and efficient; others will not. For example, one could simply use a single layer, in which case the system would revert to standard image representations.

Finding an optimal encoding of a given image sequence requires powerful and sophisticated techniques of image analysis and machine vision. Current techniques will produce sub-optimal encodings. However, current techniques can be considered to use special limited versions of the layered representation, which is to say that the layered approach offers a superset that includes current techniques. Thus the layered format can provide a standard that is bakcward compatible with existing techniques, but which allows for the growth toward more sophisticated techniques, as the technology for doing image analysis improves.

Motion analysis based on the methods described by [4][5][6][7][8] may be useful. Single-frame layering techniques as described by [3][10][11][12] etc. may be useful.

I describe here one example of a procedure for converting an image sequence into a layered format. This technique relies on various assumptions that will not hold in the general case, but it serves as a useful illustration of how one might procede.

Suppose that a foreground object, with emittance *F(x,y,t)* moves in front of a background with emittance *G(x,y,t)*, and that they are combined with an attenuation map *A(x,y,t)*. The observed image is:

$$I(x, y, t) = A(x, y, t)\, G(x, y, t) + (1 - A(x, y, t))\, F(x, y, t)$$

Now suppose that the foreground is transformed, from frame to frame, by a warp operator (velocity map) *P*, and that the foreground is transformed by *Q*, so that

$$F(x, y, t) = P_t(F(x, y, 0))$$
$$G(x, y, t) = Q_t(G(x, y, 0))$$

which may be expressed in the simplified notation,

$$F_t = P_t F_o$$
$$G_t = Q_t G_o$$

The attenuation mask is transformed in parallel with the foreground, so that

$$A(x, y, t) = P_t(A(x, y, 0))$$
$$A_t = P_t A_0$$

The sequence is then:

$$I_t = A_t G_t + (1 - A_t) F_t$$

We can then stabilize the foreground and the attenutation map by applying an inverse warp, P^{-1}, leading to:

$$P_t^{-1} I_t = A_0 P_t^{-1} G_t + (1 - A_0) F_0$$

Now if we define

$$I_t^* = P_t^{-1} I_t$$

$$G_t^* = P_t^{-1} Q_t G_0$$

then we have,

$$I_t^* = A_0 G_t^* + (1 - A_0) F_0$$

Now suppose that we know the warps P and Q, and we know the background G_0. Then we can determine A_0 as follows. For two different frames, taken at times *t1* and *t2*, we have:

$$I_{t1}^* - I_{t2}^* = A_0 (G_{t1}^* - G_{t2}^*)$$

and thus,

$$A_0 = \frac{I_{t1}^* - I_{t2}^*}{G_{t1}^* - G_{t2}^*}$$

Thus, A_0 can be determined from the known quantities on the right hand side. And once A_0 is known, one can determine F_0 by plugging A_0 back into the original equation.

Since the solution will be unstable when the denominator is near zero, it will be advantageous to make multiple observations at multiple times and to take a least squares solution:

$$A_0 = \frac{\sum_{n,m} (I_n^* - I_m^*)(G_n^* - G_m^*)}{\sum_{n,m} (G_n^* - G_m^*)^2}$$

12.0 Image data accumulation.

The information stored in a layer can extend over a larger visual angle than is presented on a given image. Consider the case where a camera pans over a stationary scene. Each image presents a glimpse into this scene through a rectangular window. To represent this image sequence, one would like to accumulate all of the information into a single large image; later, the whole sequence can be replayed by simply specifying the portion of the scene that is to be presented on each frame. The concept is illustrated in Figure 6.

When information is accumulated over frames, one must also consider how the individual samples in the frames are to be placed into the individual sample points in the accumulated layer, as shown in Figure 7. The layer will have a sampling lattice of a certain resolution, and the incoming images will generally present samples that do not align precisely

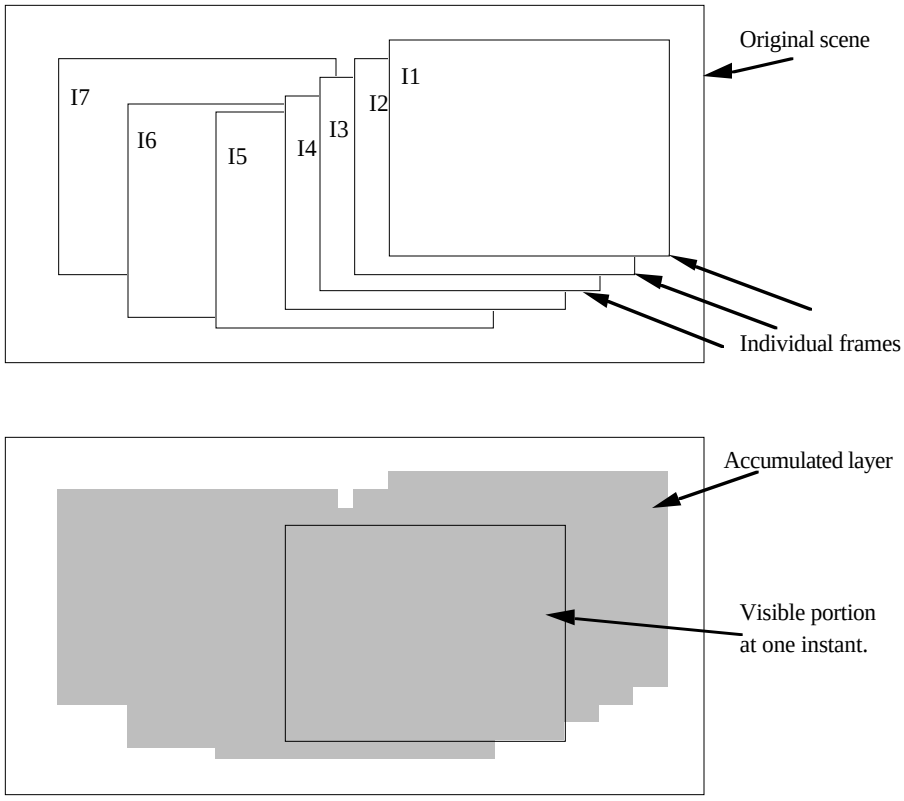


Figure 6 : Top: the frames in the sequence are take from an original scene that is larger than any individual frame. Bottom: the information from all the frames may be accumulated into a single large layer. Each frame is then a glimpse into this layer as viewed through a window.

with those of the layer. In this case it is advantageous to think of the underlying layer as a continuous function that is represented by a set of samples, and to think of the incoming data as having been acquired from a continuous signal by some sampling process (which is dependent on the prefiltering characteristics of the camera). Then one can define a method for estimating the correct layer sample values given the set of observed image samples. It will often be possible to generate a layer with higher spatial resolution than any of the individual images, especially when the camera system is spatially undersampled (as in interlace), or has some degree of aliasing (as in a CCD array). (Work along these lines was done at the Media Lab by Suenaga[15]). For display, one takes samples from the layer, but since the samples my fall on intermediate positions, it will again be necessary to use a form of interpolation. Popular methods such a bicubic interpolation can be used.

In some cases, different images in a sequence will have different degrees of resolution. One would prefer to retain the highest resolution at each point. In this case the information can be combined by the method described by Adelson[9]. A Laplacian pyramid is built from each image, and the pyramid coefficients from the different images are compared. The coefficient with the largest deviation from zero is retained. Adelson described this

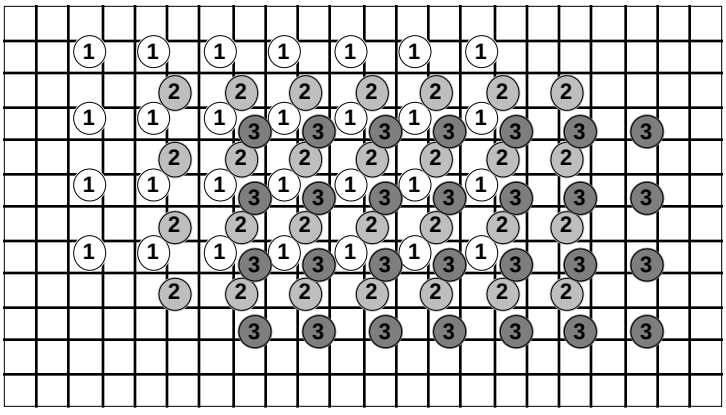


Figure 7 : The layer samples are represented on the rectangular lattice. The incoming samples from frames “1”, “2” and “3” may fall at intermediate positions on the lattice. One may interpolate from these samples in order to generate a best estimate of the underlying layer.

method in the context of Laplacian pyramids; however, it could also be used with subband representations such as pyramids using quadrature mirror filters, wavelets, or steerable filters. It could also be used with non-subsampled representations rather than pyramids. Also, rather than using the coefficient value as the selection criterion, one could use a local energy measure computed for each position in the subband.

13.0 Data compression:

The layered representation, by itself, does not necessarily offer data compression. To achieve data compression the representation should be used in conjunction with other image coding techniques. Each of the maps used in the layers can be compressed using standard image coding techniques such as transform coding or subband coding. Many of the maps will be greatly compressed in this way. For example, the velocity map will probably be of low spatial resolution, and therefore will be greatly compressed. In some cases the velocity will be a simple warp for an entire layer, and thus can be represented by a few parameters. The attenuation map will often be nearly 1 or 0 most places, and will only take on intermediate values in a few places, so it too can be greatly compressed. In addition, it is possible to encode maps in terms of each other. For example, the blur map can often be computed if one knows the velocity map and the effective shutter duration of the camera, as well as information about the camera's focus and the depth of the layer. In these cases some of the maps can be encoded with only the smallest amount of additional information.

14.0 Depth encoding:

Depth is important mainly because it determines which layer is in front of which. In many cases the ordinal relationships are enough, and one can just assign integers to the layers. It may occasionally be necessary to change the numbering. For instance a new object might enter the scene, in which case it might have to squeeze between, say, layers 1 and 2. Thus the new object is given layer 2 and the old layer 2 is renumbered 3. As long as all the layers are renumbered there should be no problem with this abrupt change.

It may be simpler to assign z values, or even z-maps, to the layers, in which case it is more natural to insert one layer between preexisting layers. The z values would not necessarily be accurate representations of object distances. For example, a still scene would initially be treated as a flat plane, with uniform and arbitrary z; only as things moved in the scene would the z-map evolve. Rendering is accomplished by looking at the z values of every layer at each pixel, and doing the right combination based on the ordering of the z's. (In the simplest case the values of the z's don't matter except as they affect the ordering).

A single layer may split into two if a stationary object suddenly moves. There will be a threshold rate of movement: slow movements will be encoded as warps of the single layer plus an error map to account for the occlusions and disocclusions; faster movements will demand a split into layers.

The z-map could also be used to help encode focal blur. A z-map can also be used to generate a bump map (its gradient) which may allow efficient encoding based on shading and light-source direction. A z-map can also account for non-uniform warps due to camera motion.

15.0 References

[1]Porter, T., and Duff, T, “Compositing Digital Images,” *Computer Graphics*, v. 18, #3, 253-259 (1984).

[2]Duff, T., “Compisiting 3-D Rendered Images,” *Siggraph Proceedings*, v. 19, 41-44 (1985)

[3]Adelson, E.,H., and Anandan, P., ``Ordinal characteristics of transparency," *AAAI Workshop on Qualitative Vision*, Boston, MA, pp.77-81 (1990).

[4]Shizawa, M., and Mase, K., “A unified computational theory for motion transparency and motion boundaries based on eigenenergy analysis,” *Proc. CVPR*, Maui, HI, 289-295 (1991).

[5]Girod, B., and Kuo, D., “Direct estimation of displacement histograms,” *Proc. Image Understanding and Machine Vision Workshop*, Cape Cod, MA pp73-76 (1989).

[6]Bergen, J., Burt, P., Hingorani, R., and Peleg, S., “Computing two motions from three frames,” *Proc. 3rd ICCV,* Osaka, Japan, pp. 27-32 (1990).

[7]Darrel, T.and Pentland, A., “Robust estimation of a multi-layered motion representation,” *Proc. IEEE Motion Workshop*, Princeton, NJ., pp. 173-178 (1991);

[8]Black, M., and Anandan, P., “A model for the detection of motion over time,” Proc. 3rd ICCV, Osaka, Japan (1990).

[9]Adelson, E. H., “Depth of field imaging process method,” U. S. Patent no. 4,661,986 (1987).

[10]Kersten, D., “Transparency and the cooperative computation of scene attributes,” in Landy, J., and Movshon, J., eds., *Computational Models of Visual Processing*, MIT Press (1991).

[11]Williams, L. R., “Perceptual organization of occluding contours,” *Proc. 3rd ICCV,* Osaka, Japan, 133-137 (1990).

[12]Nitzberg, M., and Mumford, D., “The 2.1-D sketch” *Proc. 3rd ICCV,* Osaka, Japan, 138-144 (1990)

[13]Lin, H.-D., and Messerschmitt, D., “Video composition methods and their semantics, *IEEE ICASSP* (1991).

[14]Metelli, F., “The perception of transparency,” *Scientific American*, v. 230, (4), 91-98 (1974).

[15]Suenaga’s work at the Media Lab on resolution out of time.