Chapter 4

Applications of 1:4 Tree Sensor in Image Processing

Our eyesight has the functions of not only just "looking at" the objects, but also "watching" them, for example, making attentions to the restricted objects, or having the intensional adaptivity in dark field. Some approaches for implementing these functions in the solid-state image sensors are studied, and most of them take approaches of implementing the structure of the retina using electronic circuits[3].

In this chapter, the previous studies for implementing the functions of the eyesight in electronic circuits are summerized at first. The efficiency and the way to apply the 1:4 tree scan for implementing some functions of the eyesight, mainly in terms of various adaptivities are discussed in section 4.2 and section 4.3.

It is also discussed the applications of 1:4 tree scan for the movie compression implemented on image sensors, mainly in terms of inter-frame differences and motion compensation, which are often used in conventional movie compression, such as MPEG, section 4.4.

4.1 Previous Approaches for the Functions of Eyesight

Figure 4.1 shows the structure of the typical retina in the creatures' eyes. The retina mainly consists of the following elements.

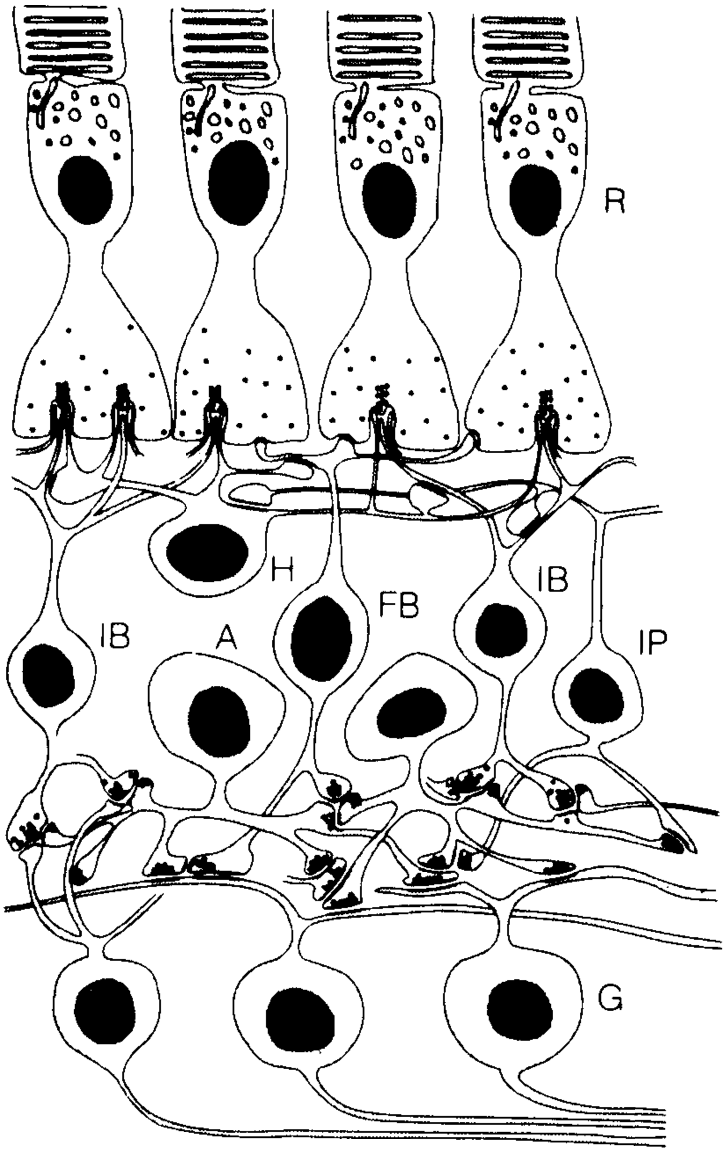


Figure 4.1: Cross-sectional view of the retina's structure.(From [3])

- Photo receptors(R), which convert the light signal to the electrical signal, and are often weakly connected each other.

- Horizontal cells(H), which receive the signal from the photo receptors, and connect them each other. This function implements the smoothing of the images.

- Bi-polar cells(FB and IB), which take the differences of the signal from photo receptors and that from horizontal cells. This function implements a kind of edge detection.

The functions implemented in the retina, the smoothing and the edge detection, are called "early vision problem," and they have been studied for a long time as the pre-processing of image signals.

One of the most previous works in computational sensors is "silicon retina", as introduced in section 1.2. It implements the smoothing of received images by the resistor networks, as the horizontal cells in the retina.

One of the other studies for "silicon retina" is "foveated" image sensors[19], as shown in Figure 4.2. In this sensor, the photo receptors and transfer CCDs are placed like the concentric circles, which is very similar to the placement in the retina, which has the characteristics of higher spatial resolution at the center part. It is also notable that the obtained data from this image sensor will be invariant against the rotation of the objects.

As seen above, most of approaches for implementing the functions of eyesight are made by implementing the structure of the retina in electronic circuits[20, 21].

4.2 Adaptivity for Spatial Resolution

4.2.1 Spatial adaptivity of eyesight

We will usually see the objects, with making attentions just to the area we have interests. In this case, we will obtain the detail information for the areas of interests,

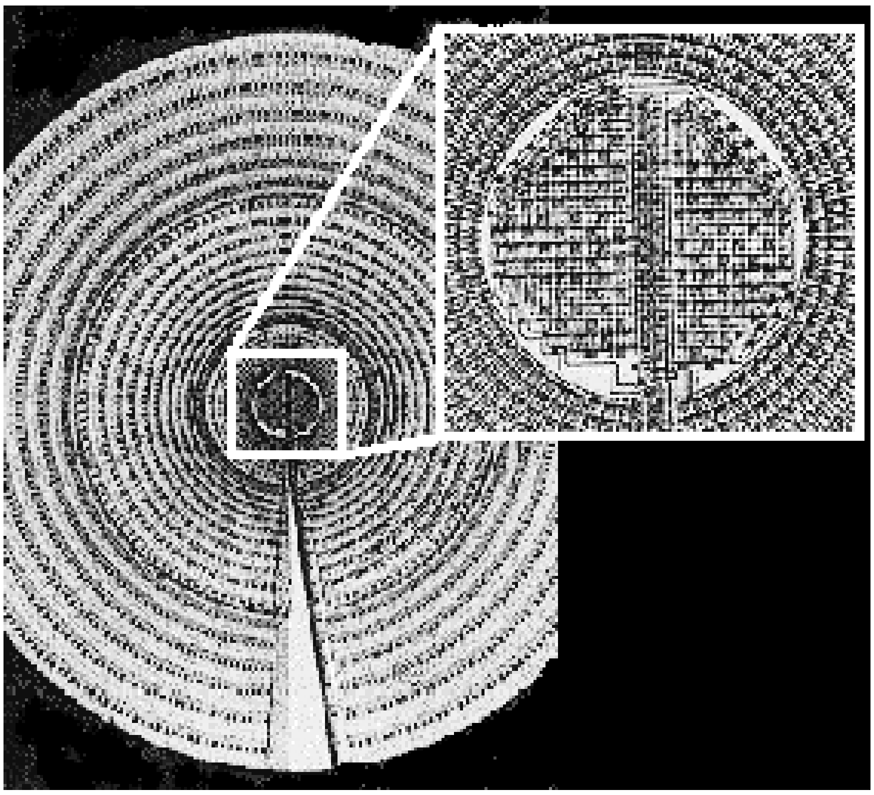


Figure 4.2: Photograph of the foveated CCD retina[19].

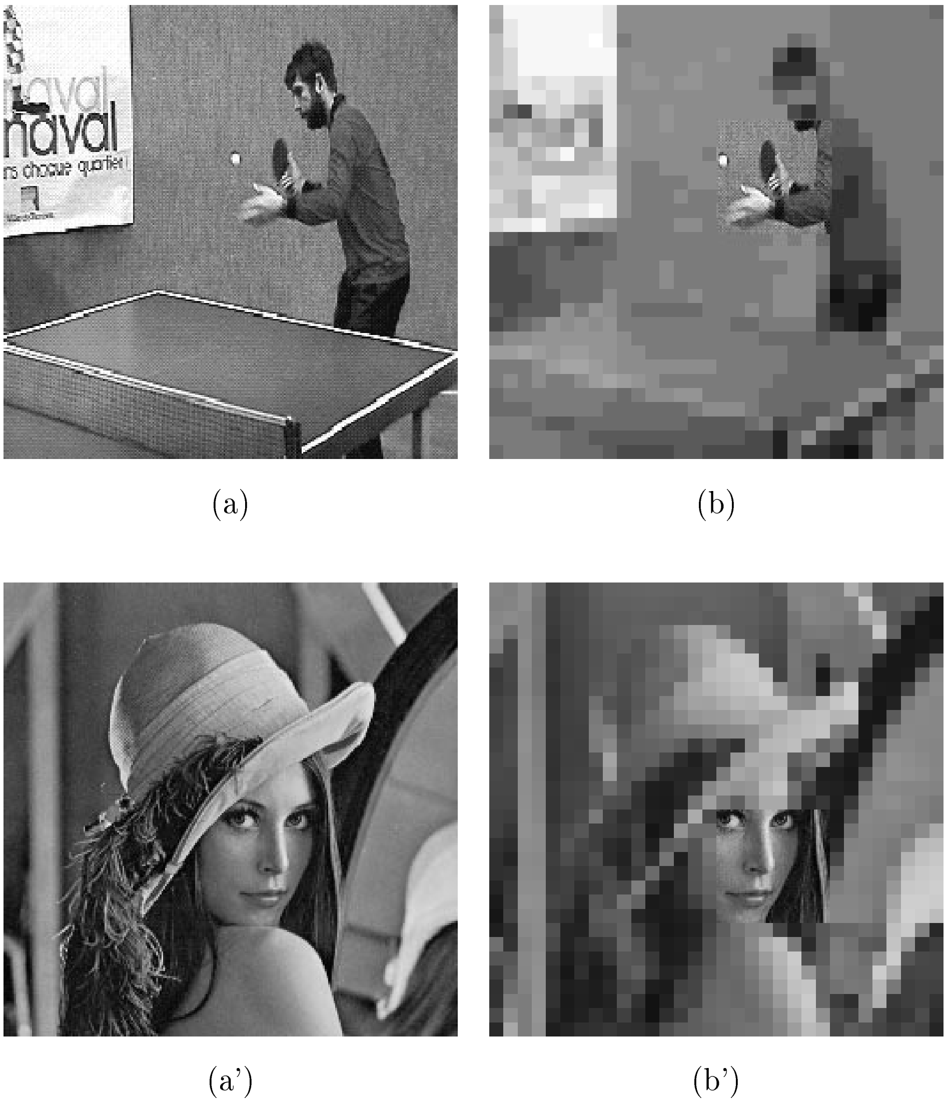


Figure 4.3: Samples of the original images(a)(a'), and the images with making attentions to the restricted area(b)(b').

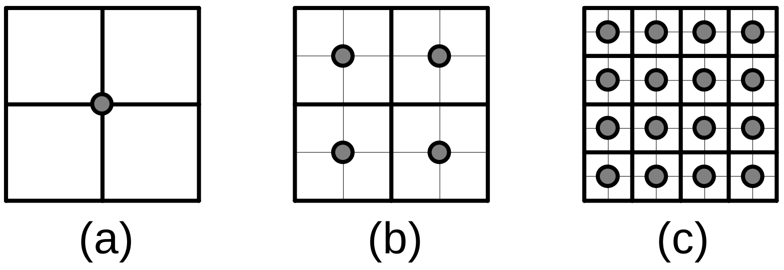


Figure 4.4: Area division by the step of 1:4 tree scan. (a)unit of 4×4 pixels , (b)unit of 2×2 pixels, and (c)unit of 1×1 pixel. (The gray circles in this figure represents the higher nodes for each sub-area.)

and less information from the other areas, such as just the intensity or the information of objects' existence, which will be mosaicked images as shown in Figure 4.3(b)(b').

This fact is expressed as follows; the higher spatial resolution for the areas of interests, and the lower for the other areas, for example by the unit of 8×8 pixels.

4.2.2 Implementing spatial adaptivity using tree sensor

It is notable that the 1:4 tree scan proceeds from the nodes in the higher level to the nodes in the lower level, and to the lower the scan proceeds, the higher spatial resolutions will be obtained, since the nodes have the logical-OR, or *summerized information* of their lower nodes, as shown in Figure 4.4.

This procedure can be implemented by adding the functions for each node, to select whether proceeding the scan normally, or finishing the scan regardless of its value. The nodes for the areas of interests should continue its step of scan normally, and the nodes for the other areas should finish its scan to return the summerized information of its sub-area.

The procedure of "spatial adaptive scan" by 1:4 tree structure is expected to be the efficiency of a kind of data compression, by skipping the redundant scan for the areas of no interests. For example, the conventional 1:4 tree scan of the digitized image in Figure 4.3(a) and (a') give the 1:4 tree code of 59,061 and 42,953 bits, respectively, where the images have $256 \times 256 = 65,536$ pixels. (Note that the

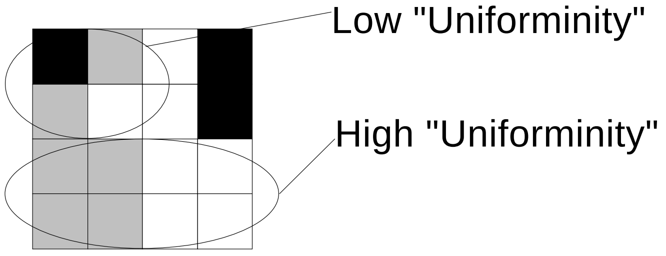


Figure 4.5: Sample of gray scale image with partial uniformity.

pictures in Figure 4.3 are illustrated as gray scale images for readers' convenience. These images are processed after digitizing to binary images in this study.) On the other hand, the extended 1:4 tree scan to obtain the mosaicked images in Figure 4.3(b) and (b') give the 1:4 tree code of 1,581 and 8,177 bits, respectively, which is about 1/27 and 1/8 of the conventional 1:4 tree code, respectively. Note that the scan for the mosaicked areas is proceeded to the resolution up to 8×8 and 4×4 pixels for Figure 4.3(b) and (b'), respectively. These results indicate that 1:4 tree scan can implement a kind of data compression according to the attentions for the images of interests.

4.2.3 A concept of using spatial adaptivity for still image scan

In the still images, whether they are gray scale or binary, there are two parts of areas; the areas whose pixels have almost the same intensity, and the areas whose pixels have much difference intensities. Here we express the "uniform" areas for the former, and the "ununiform" areas for the later. Since the pixels in the "uniform" areas will have almost the same intensity, the scan for the all pixels are not needed; just the scan of the summerized (mean) intensity of the areas is needed, while the complete scan will be needed for the "ununiform" areas to obtain the informations for the details.

This procedure can be implemented by the 1:4 tree scan, by the partially stopped scan for the "uniform" areas, and the complete scan for the "ununiform" areas. This methodology can be expressed as that it uses "uniformity" as the criteria of interest.

Assuming that the intensity of each pixel should has the analog value, the "uni-

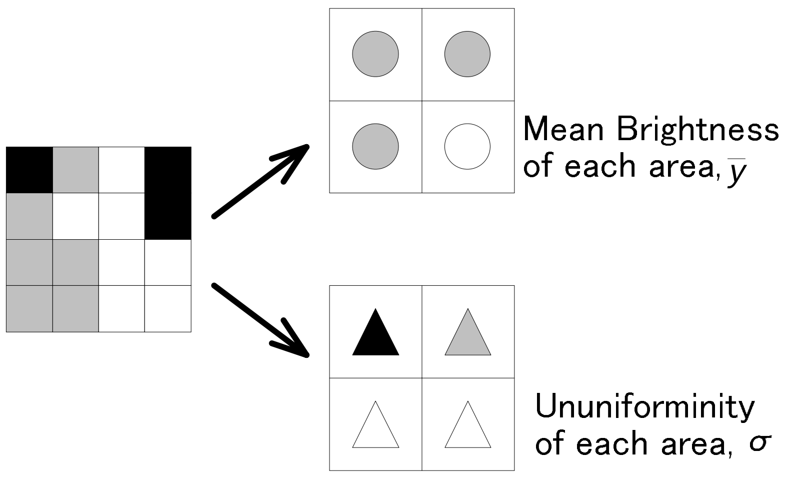


Figure 4.6: Models of the nodes which have two values; mean intensity, $\overline{y}$ and the standard deviation of the intensity, σ.

formity" of the area can be defined by the standard deviation of intensity in the areas, σ as follows.

$$\sigma^2 = \frac{1}{4}\sum_{i}^{4}(y_i - \overline{y})^2 \tag{4.1}$$

$$\overline{y} = \frac{1}{4}\sum_{i}^{4} y_i \tag{4.2}$$

Here the y_i is the mean intensity of each four sub-areas, and $\overline{y}$ is the mean intensity of all four sub-areas, and the models of this equation are shown in Figure 4.6.

The procedure considering the uniformity of the images using the hierarchical structure of this y_i and σ for each node, can be implemented by the following procedure.

1. Scan the uniformity value, σ of the node, as shown in Figure 4.7(a).
2. If σ is smaller than the threshold value, σ_θ, it implies that the area of the node will be uniform enough, and it finishes the scan just by returning the mean intensity, $\overline{y}$, as shown in Figure 4.7(b).
3. If σ is larger than σ_θ, it implies that the area of the node will contain variations of pixels' intensity, and it continues the following scan for the lower nodes, as shown in Figure 4.7(c).

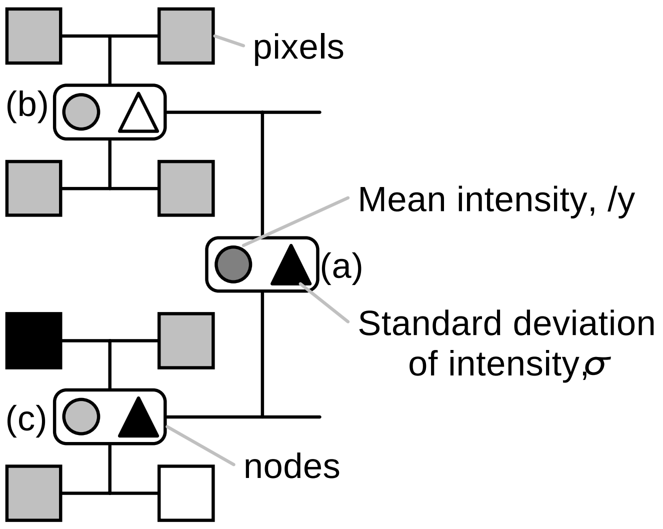


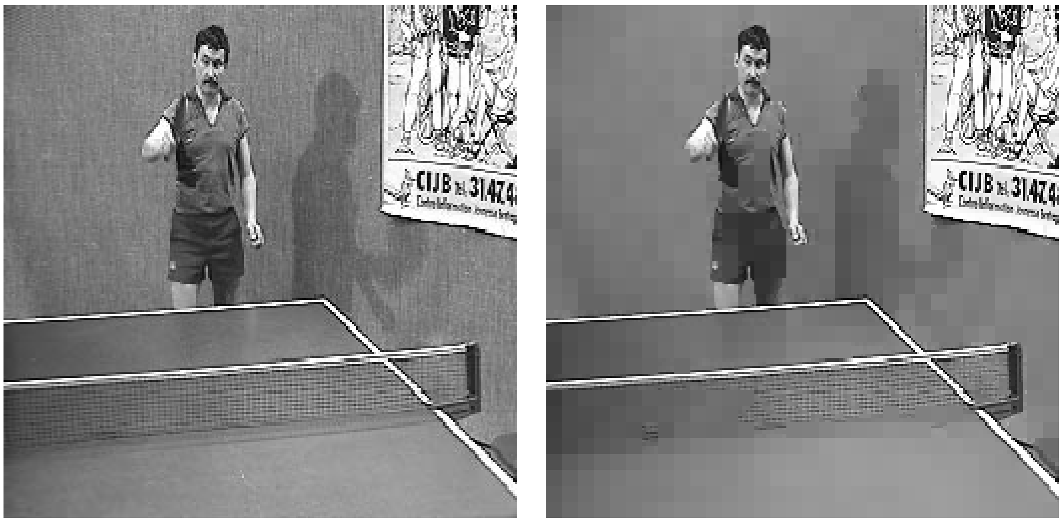
Figure 4.7: The procedure of scan considering the uniformity of each area.

Figure 4.8 shows the example of the above procedure. The original gray scale image with 256×256 pixels is shown in Figure 4.8(a), and the reconstructed image using the scan considering the uniformity of each area discussed above is shown in Figure 4.8(b). In Figure 4.8, the just nodes whose σ is larger than $\sigma_\theta = 10$ are scanned normally, for the smaller sub-area of 16×16 pixels. (Note that the intensity depth of the image is 256) Seen in Figure 4.8(b), the objects with low uniformity, such as the man or the poster on the wall, are scanned to the detail, while the areas with high uniformity, such as the gray background or the table tennis court, are scanned with lower spatial resolution.

The number of scan steps is 19,660, which is about 1/3 of the conventional raster scan, and the signal-to-noise ratio (SNR) of the reconstructed image, defined by the following equations, is 31.4dB,

$$\mathrm{SNR[dB]} = 10\log_{10}\frac{255^2}{\frac{1}{XY}\sum_{x=1}^{X}\sum_{y=1}^{Y}\{p(x,y)-p'(x,y)\}^2}, \tag{4.3}$$

where M and N are the number of pixels in horizontal and vertical direction, respectively, and $p(x,y)$ and $p'(x,y)$ are intensity at (x,y) in the original and reference image, respectively.



(a) (b)

Figure 4.8: Sample of gray scale image(a), and the reconstructed image using the scan considering the uniformity of each area(b).

This result indicates one possibility of the spatial adaptive scan with a kind of data compression maintaining the quality of the images, using 1:4 tree structure.

4.3 Adaptivity for Intensional Resolution

4.3.1 Intensional adaptivity of eyesight

Our eyesight has the function to see the objects both in the bright condition and the dark condition, or have the adaptivity for the intensity of the field. This function is implemented by having two photo receptors in the retina; *the rod* for the dark-field without the sense of color, and *the cone* for the bright-field with the sense of color[3].

One difficulty of receiving the photo signals by the electronic device is that the range of intensity of the field is broad up to 5 decades, and it is difficult to keep the high dynamic range for such a broad range of intensity by using the simple photo-electronic converter. It is known that the sensitivity of the photo receptor in the retina is proportional to the logarithm of the input intensity, which can keep the

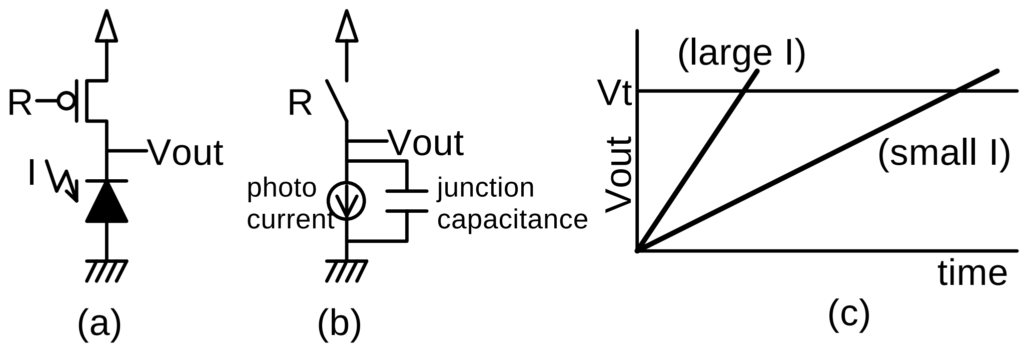


Figure 4.9: A simple circuit of photo-electronic converter(a), its equivalence circuit(b), and its output with the light of small and large intensity(c).

high dynamic ranges[3]. Some studies for the photo-electronic converting circuits with the logarithmic sensitivity[22, 23, 24], and some of them intend to obtain the more wide dynamic range by selecting the gain of photo-electronic converting circuits corresponding to the light intensity[24].

4.3.2 A concept of gray-scale scan using 1:4 tree with intentional adaptivity

Figure 4.9 shows a simple circuit of photo-electronic converter. The charge in the junction capacitance of the photo diode is cleared by the reset signal at first, and then the charge is integrated in the capacitor by the photo current, $I_{\rm photo}$. Since the photo current is proportional to the intensity of light, the output voltage $V_{\rm out}$ will expressed as follows;

$$V_{\rm out} = C \cdot I_{\rm photo} \cdot t \propto I \cdot t, \tag{4.4}$$

where t is the time of integration, and I is the intensity of light. Using the Equation (4.4), the time until $V_{\rm out}$ reaches the threshold voltage, V_t is estimated to be proportional to I^{-1}, and we can obtain the information of the light intensity by this time.

The flag indicating that $V_{\rm out}$ has reached V_t will be activated at the local areas in the focal plain, since the distribution of the intensity is expected to be local in two dimensional focal plain, as shown in Figure 4.10. It is the case which can

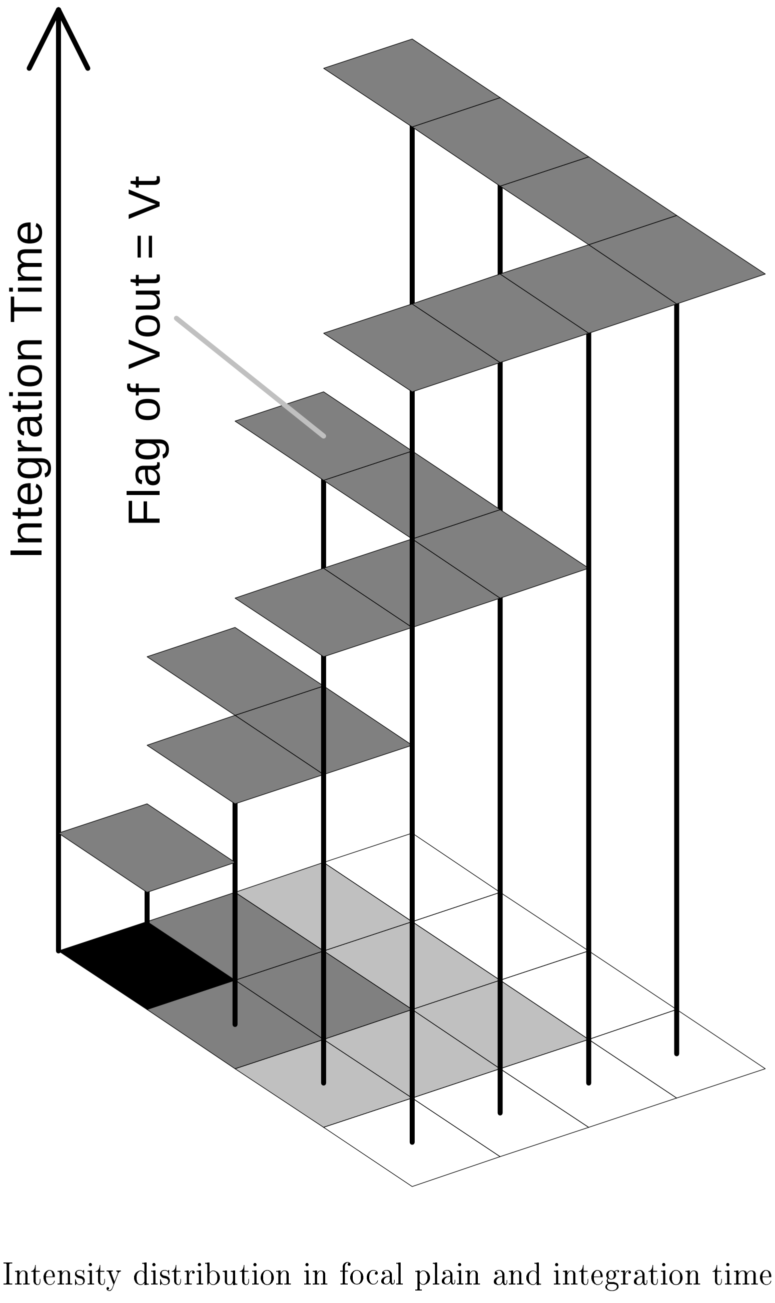


Figure 4.10: Intensity distribution in focal plain and integration time to $V_{\text{out}} = V_t$

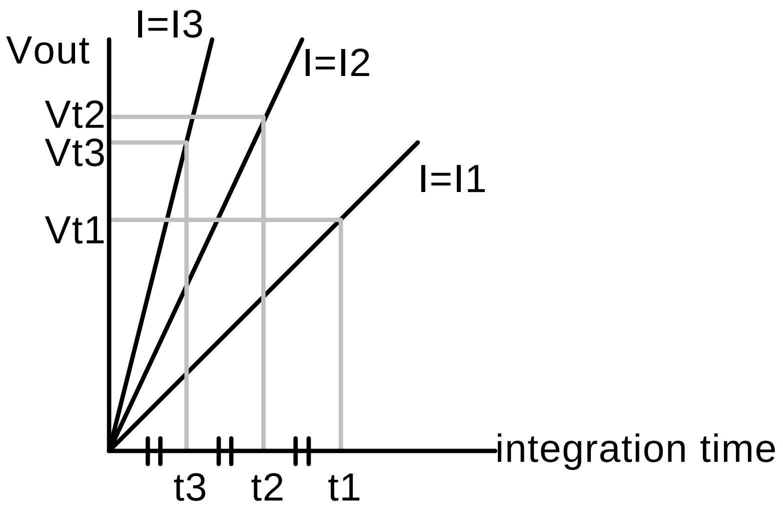


Figure 4.11: Controlled V_t to keep the interval time of V_{out} to reach V_t.

be efficiently scanned by the 1:4 tree scan, where the pixels of "1" are distributed locally, as shown in section 3.2.2. The flag once scanned should be cleared, since the pixel whose V_{out} has reached V_t is not need to be scanned any more.

One problem of the above technique is that V_{out} for the higher intensity will reach V_t rapidly, while that for the lower intensity will rise very slow, and this fact will result in the rush of flags at the early terms in integration time.

One concept to solve this problem is to control V_t according to the terms in integration time, as shown in Figure 4.11. The V_t will be set small at the early terms in integration time to keep the interval time long, while V_t will be set larger at the middle term in the integration time, and V_t will be set small again at the later term for the lower intensity.

Assuming that the slope of V_{out} is proportional to the light intensity I, as $V_{\text{out}} = kIt$. The integration time until V_{out} reaches to V_t is expected to have the equal interval, for the intensity of the equal interval, and the intensity and the integration time of i-th brightness, I_i and t_i, respectively, will expressed as follows.

$$I_i = I_0 + i\Delta I \tag{4.5}$$

$$t_i = t_0 - i\Delta t \tag{4.6}$$

V_t for the intensity of I_i, V_t^i is expressed as the V_{out} at the time of t_i, as follows.

$$V_t^i = kI_i t_i = k(I_0 + i\Delta I)(t_0 - i\Delta t) \tag{4.7}$$

Using this equation, V_t^i is expressed as the function of integration time, by eliminating i from the all equations, as follow.

$$V_t(t) = -k\frac{\Delta I}{\Delta t}\left(t - \frac{\Delta I - I_0\Delta t}{2\Delta I}\right)^2 + \frac{(\Delta I + I_0\Delta t)^2}{\Delta I\Delta t} \tag{4.8}$$

The interval of the integration of time to reach a threshold voltage will be equal by changing $V_t(t)$ according to the integration time, as derived in the Equation (4.8). This control will make the almost constant number of pixels at each scanning term.

4.4 Applications for Movie Compression

4.4.1 Inter-frame difference of movies and movie compression

The differences of pixels between successive two frames in moving pictures, which is called *inter-frame difference*, have the values when there were some motions of objects in the scene, and most of the pixels will have no difference except the restricted area where motion occurred. One of the simplest, and most important technique to compress the movie data is to take the inter-frame difference, and to read out just the value of pixel whose value has changed between two frames, which is called "effective pixel."

4.4.2 Implementing scan of inter-frame difference using 1:4 tree

The ratio of the effective pixels in inter-frame difference is expected to be small enough, except the special case, such as when the scene has changed. The scan for the effective pixels using 1:4 tree structure will make a high efficiency, since 1:4 tree scan is suitable for the images containing a few effective pixels, whose value is "1", as shown in section 3.3.2.

This function can be implemented by modifying the pixels in 1:4 tree structure to have the value as the inter-frame difference, by adding the memory to keep the value in the previous frame, with modifying the nodes for binary images.

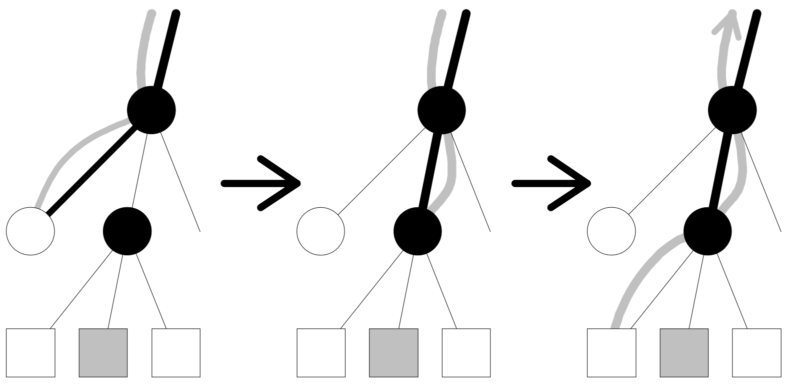


Figure 4.12: 1:4 tree structure for scanning inter-frame difference.

Here we consider the method to extend the above compression methodology for gray-scale movies. One of the simplest algorithms for gray-scale movie compression using inter-frame difference is "Conditional Replenishment Method" as the following procedure.

1. take the inter-frame difference; the arithmetical subtraction of the values of pixels in the successive two frames. (The value of pixel in the previous frame is stored in the memory.)

2. if the difference is smaller than the threshold, the scan for this pixel will be skipped.

3. if the difference is larger than the threshold, the value of this difference is read out outside the sensor.

4. the stored value in the memory is replaced by the value of the pixel in the current frame when scanned.

We can apply the 1:4 tree scan and conditional replenishment method by adding the analog signal pathes as follows.

- each pixel has the flag which indicates that it has the larger inter-frame difference than threshold; "1" for effective pixels.

- the flags are scanned by the conventional 1:4 tree structure.

Table 4.1: 1:4 tree scan code length per pixel, $\overline{L}$ and mean SNR of reproduced images, $\overline{\text{SNR}}$ using conditional replenishment method for various threshold of intensity, θ.

	$\theta = 5$		$\theta = 20$	
name	$\overline{L}$	$\overline{\text{SNR}}$	$\overline{L}$	$\overline{\text{SNR}}$
MissAmerica	0.474	41.1	0.115	32.5
TableTennis	0.774	41.7	0.345	30.7
FlowerGarden	1.137	44.8	0.984	30.8
Rail	0.113	44.4	0.030	38.3
Neck	0.313	42.5	0.126	35.5

- when the scan step has proceeded to each pixel, the analog difference is read out, through the analog signal path along to the scan path from the top to the pixel.

An example of the above procedure is shown in Figure 4.12, where the squares and the circles represent the pixels and nodes, respectively, and black circle is the node whose value is “1”. The gray curve represents the analog signal path for the analog inter-frame difference, from the pixel to the outside of the sensor, and the thick line between nodes represents the current 1:4 tree scan path. The analog signal path will be easily implemented by the transfer gates by the pair of n-channel and p-channel MOSFETs.

Table 4.1 shows the results of the above procedure for some sample movie sequences, where $\overline{L}$ is the mean 1:4 tree code length per pixel for the scan of effective pixels whose difference between two frames is larger than θ, and mean signal to noise ratio (SNR) over all sequences of the reproduced image by conditional replenishment method over the original images.

In most cases, the larger θ gives the shorter 1:4 code length and worse SNR, and we can select θ for the purposes. In the sequence of “FlowerGarden,” the whole area of images moves slowly, and the ratio of effective pixels is constantly very large, which results in the larger $\overline{L}$ than the raster scan.

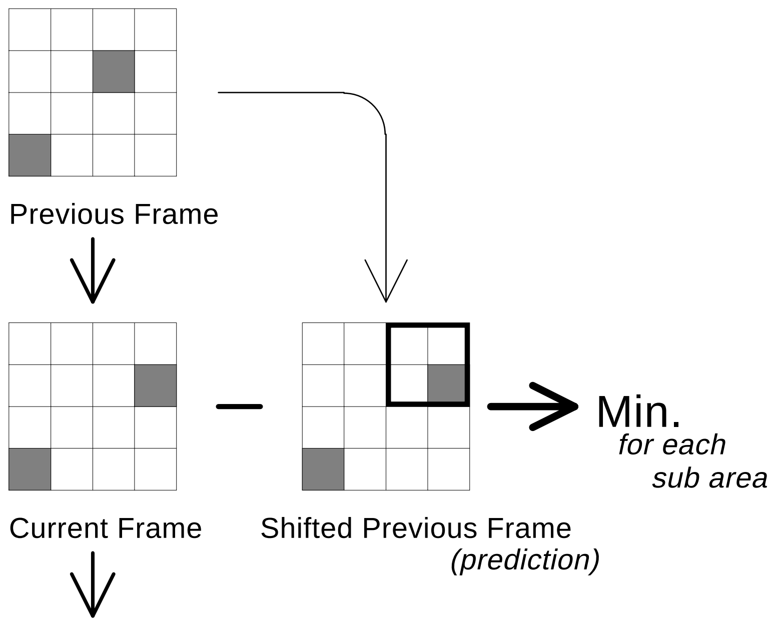


Figure 4.13: The simplest methodology for motion compensation.

4.4.3 Motion compensation and movie compression

Another of the most important techniques for movie compression is "motion compensation," which is the prediction of the motion. The error between the predicted image by the motion compensation and the practical frame will be read out, using the conditional replenishment method, if needed. The standard movie compressions, such as MPEG, employ both the inter-frame difference and motion compensation[25].

The simplest methodology to calculate the motion compensation, which is often done in the conventional movie compression, is as follows, as shown in Figure 4.13.

- scan the images and store the buffer memory.
- create the moved image to various directions with the various distances for divided sub-areas, from the stored previous frame, and calculate the difference between this moved image and the practical image.
- choice the direction and the distance to minimize the difference for each sub-area, and they are compensated motion.

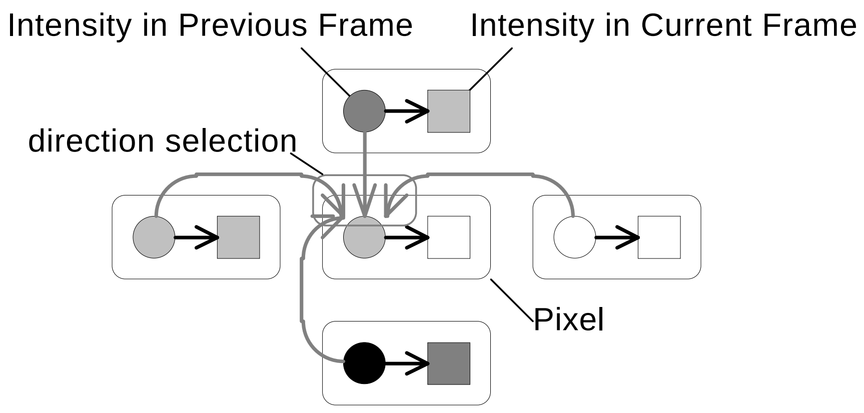


Figure 4.14: Pixels connection for motion compensation in 1:4 tree

This procedure is very complex, and the motion compensation is one of the most complex operations for movie compression.

4.4.4 Implementing motion compensation using tree sensor

The motion compensation is implemented using 1:4 tree structure by adding the following functions as shown in Figure 4.14.

- The pixel has the value of difference between the value in the current frame and that in the previous frame, but it is used the value in the previous frame of the neighbor pixel from the selected distance and direction, instead of the current pixel.

- The node in the 1:4 tree structure has the value as the mean of the connected four nodes' value. The value of one node represents the mean of values in the all sub-area under it.

- The nodes according to the size of area needed for motion compensation are scanned from the outside, and this scan is iterated by changing the distance and direction for the motion to be compensated.

- Select the direction and the distance to minize the difference, for each sub-areas, and they are compensated motion for each sub-area.

Assuming that the pixel should has the mean value of the lower four nodes, the scanned value at any step gives the mean difference at the sub-area lower of the node, and we can select the block size of motion compensation by selecting the scan step.

Figure 4.15 shows the samples of simulation of motion compensation, using the standard movie, "MissAmerica".

The SNR between the practical image and the predicted image by compensated motion using the above algorithm is shown.

In case of Figure 4.15(a), SNR for the two size of sub-areas for independent motion compensation is shown, whole area of "1 of [256x256]" and 16×16 sub-areas of 16×16 size as "256 of [16x16]".

In Figure 4.15(b), the experience results by changing the searching distance are shown, searching just the neighbor pixel, $d = 1$, and the pixels within 15 pixels, $d < 15$.

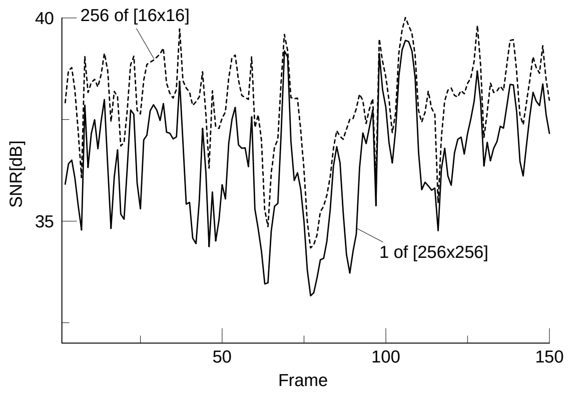
The comparison of the changing the searching directions is shown in Figure 4.15(c), with 8 directions of "8Dir" and 32 directions of "32Dir".

The simulation results shows that the more the range of searching directions and distances or the smaller the size of sub-areas, the prediction by compensated motion will be more precise.

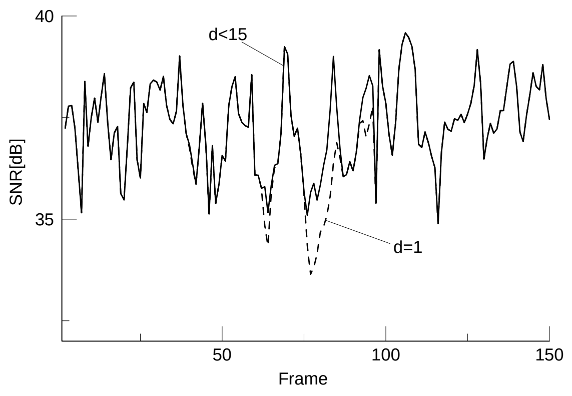
It is indicated that in case of very high frame rate, for example 1ms per frame, the motion will occurred within at least one pixel, since the object may not move with the faster velocity[26]. In such case with scanning with very high frame rate, the motion will be restricted to within one pixel, and then it will be enough for precise motion compensation the search range of just one neighbor pixels and 8 directions.

It will be can easily implemented by adding the signal paths between one pixel and the 8 neighbor pixels and the memory to store the pixel value in the previous frame, and the global signal to select the direction to be moved in Figure 4.14.

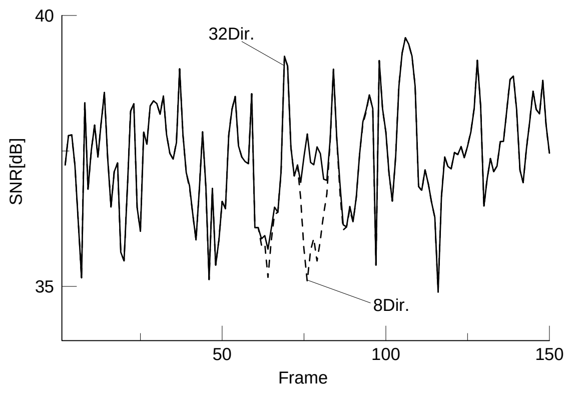
Table 4.2 shows the mean 1:4 tree code length per pixel for the effective pixels, $\overline{L}$ and the mean SNR of reproduced images, $\overline{\mathrm{SNR}}$ using conditional replenishment method. Here the conditional replenishment method is carried out for the difference



(a)



(b)



(c)

Figure 4.15: Simulation results of SNR for the predicted image by compensated motion and the practical image.

Table 4.2: 1:4 tree scan code length, $\overline{L}$ per pixel and mean SNR of reproduced images, $\overline{\text{SNR}}$ using conditional replenishment method after motion compensation.

	$\theta = 5$		$\theta = 20$	
name	$\overline{L}$	$\overline{\text{SNR}}$	$\overline{L}$	$\overline{\text{SNR}}$
MissAmerica	0.364	41.3	0.046	34.8
TableTennis	0.753	41.4	0.296	30.7
FlowerGarden	1.097	42.9	0.809	29.9
Rail	0.116	44.3	0.030	37.3
Neck	0.321	42.0	0.093	35.5

between the original image and the predicted image, which is generated from the previous image using the compensated motion within 5 pixels and 8 directions.

In most cases, the 1:4 tree code length and SNR are better than those without motion compensation, but SNR is worse than without motion compensation in some cases, since the precision of motion compensation is not enough for these movies captured in slow frame rate of about 30ms/frame. Assuming high frame rate, the precision of motion compensation will be enough to keep enough SNR.

4.5 Application for Visual Tracking

In this section, the simple algorithm of visual tracking using 1:4 tree structure is described. The methodology of masking target image used for selecting targets in visual tracking is described at first, and the visual tracking algorithm using image masking is described next.

4.5.1 Image masking using 1:4 tree structure

Here we consider to scan just for the restricted area in images, and the other area should be white, as shown in Figure 4.16. The original image is Figure 4.16(a), and here the target area drawn gray in Figure 4.16(b) in the original image is to be scanned as shown Figure 4.16(c). We assume the bitmap image, we call "window"

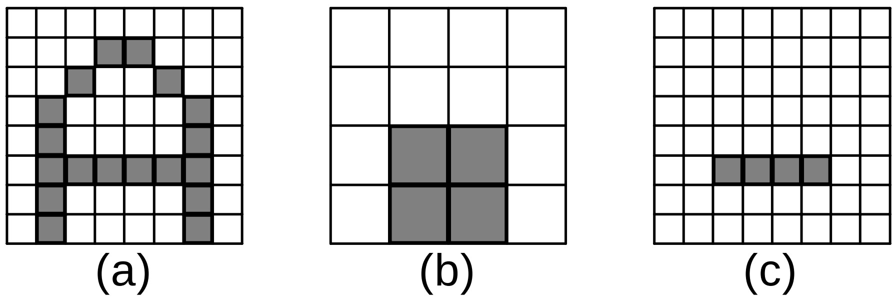


Figure 4.16: Samples of original image(a), target area to be masked(b), and the masked image(c).

here, as the raster image of “1” for the target area and “0” for for the other area, and the resolution of the “window” can be assumed independent on the original image, for example 4×4 in Figure 4.16(b). The 1:4 tree code for this “window” in Figure 4.16(b), here we call “window code” is obtained as follows, with the order of level in 1:4 tree structure l.

Level(l)	1	2	2	2	3	2	3
Window code	1	0	0	1	0101	1	1010

Here we extend the functions of the nodes in 1:4 tree structure as follows.

- Refer the bit of window code corresponding to the level l of the conventional 1:4 tree scan for the original image.
- If the corresponding bit of window code is “0”, it implies that the lower level than the current node is the area out of the target, and it returns 1:4 tree code of “0” since no more scan steps are needed.
- If the corresponding bit of window code is “1”, it implies that the area is in the target area. The following 1:4 tree scan steps for the lower level are carried out, and the corresponding level of window code is also descended.
- If the 1:4 tree scan step has reached the lowest level, the scan is executed for each pixel in target area, and scan steps are carried out regardless of the bit of window code.

For example, in case of Figure 4.16(a), the extended 1:4 tree scan steps proceeds as follows. (C is the current bit of 1:4 tree code)

1. First, since the bit of window code for $l = 1$ is “1”, the usual 1:4 tree scan is executed, and we obtain $C = 1$.

2. The scan proceeds to $l = 2$, and the corresponding bit of window code is “0”. No more scan steps for the lower level is needed, since this area is out of target, and we obtain $C = 0$.

3. The next step for $l = 2$ is also out of target, since the corresponding bit of window code is “0”, and $C = 0$ is obtained.

4. The next bit of window code is “1”, and it implies that the following scan steps are in the target area, and we obtain $C = 1$ at first. Because the following four bits of window code are “0101”, the second and the fourth 2×2 areas (upper-right and lower-right, respectively) are the target area, while the first and the third (upper-left and lower-left, respectively) are out of target.

5. The first 2×2 area is out of target, since the corresponding bit of window code is “0”, and we obtain $C = 0$.

6. The second 2×2 area is in the target area, since the corresponding bit of window code is “1”, the usual 1:4 tree scan steps are executed in order, and we obtain C=1-0011.

7. The third 2×2 area is also out of target, and we obtain $C = 0$. The last 2×2 area is in the target area, and we obtain C=0, since the all pixels in this area is “0”, and here the scan steps for $l = 3$ have finished.

8. The scan step should back to $l = 2$ again, and since the corresponding bit of window code is “1”, we obtain $C = 1$, and the following procedures for the lower level are executed similarly, and we obtain C=1-0011-0-0-0.

Here the following 1:4 tree code is obtained.

Level(l)	1	2	2	2	3	3	4	3	3	2	3	4	3	3	3
1:4 Tree Code	1	0	0	1	0	1	0011	0	0	1	1	0011	0	0	0

The image masking procedure described above can be executed by referring only the corresponding bit of window code in order to decide whether usual 1:4 tree scan steps should be carried out or not.

4.5.2 Visual tracking algorithm using 1:4 tree structure

The motion in the movies is detected simply by taking inter-frame difference, if no motion predictions or motion compensations are executed. The simple visual tracking algorithm is to trace only the moving objects. The moving objects include moving pixels, which are "effective pixels" in the successive two frames. Here we consider the visual tracking algorithm as follows, by determining the target area based on the motion.

- If there are effective pixels in the area corresponding to the window code of "0", it implies that the new motion has occurred in the non-target area, and this area should be added to the target area. The window code of current "0" should be replaced as "1-0000", if the corresponding level is not the lowest, while the current 0 in window code should be changed to "1" if the corresponding level is the lowest.

- If there are no effective pixels in the area corresponding to the window code of "0", no motions have occurred in the non-target area, and the window code should no be modified.

- If there are no effective pixels in the area corresponding to the window code of "1", the motion in the target area has stopped, and this area should be set non-target area. The corresponding bit of window code should be replaced as "0", and the bits for the lower level also should be replaced as "0", if the corresponding level is not the lowest.

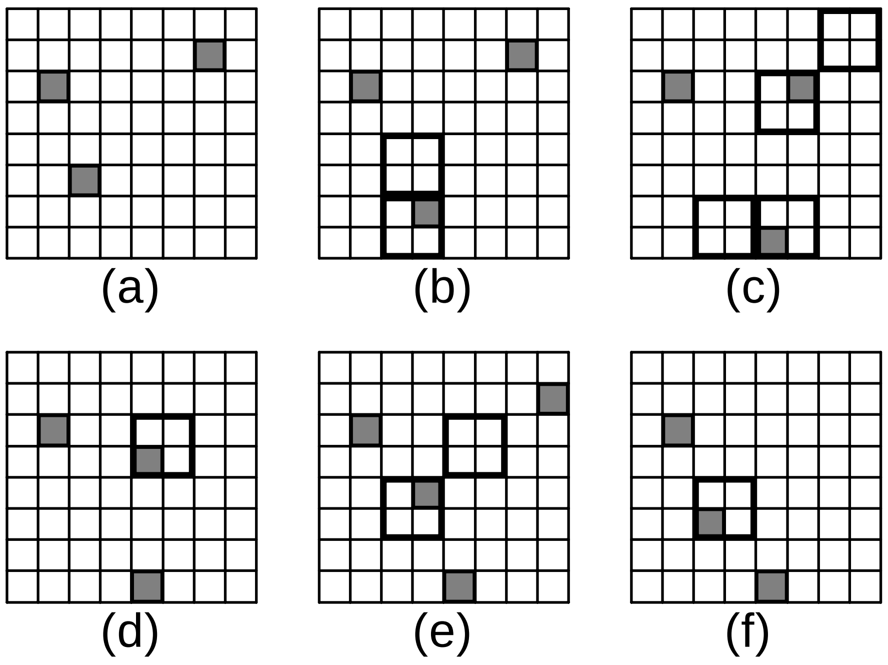


Figure 4.17: Steps of the visual tracking using 1:4 tree scan and window code for masking. (The area surrounded by bold line represents the target area.)

- If there are effective pixels in the area corresponding to the window code of "1", some motion have occurred in the target area, and window code should not be modified.

Figure 4.17 shows the steps of visual tracking using the algorithm discussed above. The visual tracking steps are follows.

1. Figure 4.17(a) indicates the start frame. There are no target areas, since no motions have occurred.

2. Figure 4.17(b) indicates the following frame. The point at the lower-left area has begun to move, and the areas of the motion are set as the target area, which are represented by surrounded area by bold lines.

3. In the following frame in Figure 4.17(c), the point at the upper-right area has also begun to move, and both of the motions are traced by the target areas.

4. In Figure 4.17(d), the point at the lower-left area has stopped moving, and the corresponding area is set out of target.

5. One-shot noise has appeared at the upper-right area in Figure 4.17(e).

6. In Figure 4.17(f), temporary appeared noise has not be traced, since it has disappeared before the window code was modified.

The visual tracking algorithm discussed above has the following characteristics.

- The number of target area is free from restriction.
- Only the successive two frames are used, but motion compensation has not been implemented.
- The targets are disappeared if they have stopped.
- The targets can be traced regardless to their direction or velocity.
- One-shot noise which will disappear within one frame is neglected.
- There are only two modifications of window code; "0" to "1-0000", and "1-xxxx" to "0". These modification will be implemented without the huge global pixel memory.

4.6 Summary and Conclusion

In this chapter, some applications of 1:4 tree structure for image processing are discussed.

The spatial adaptivity is easily implemented by the 1:4 tree scan, since the lower the 1:4 tree scan proceeds, the image with the higher spatial resolution is obtained. It is also discussed the methodology to scan the gray-scale still image considering the local uniformity, and the possibility is shown that this methodology can compress the image about 1/3, with maintaining the quality of the image.

It is also discussed the applications of 1:4 tree structure for movie compression. It is indicated that the two most important techniques in movie compression, inter-frame difference and motion compensation will be implemented effectively by modifying the conventional 1:4 tree structure.