MOTION

1. sequence of stills or dynamic approach

2, detection - security

3. traffic flow - mobile robots

4. navigation

types

1. stationary camera, stationary objects (SCSO)

2. stationary camera, moving objects (SCMO)

3. moving camera, stationary objects (MCSO)

4. moving camera, moving objects

To detect motion

1. difference of pictures

local only, accounts for small changes in camera, illumination, noise etc. does not account for velocity of different components

$$DP_{jk}(x,y) = \begin{cases} 1 & |F_j(x,y) - F_k(x,y)| > T \\ 0 & \text{otherwise} \end{cases}$$

Likelihood test: Given 2 pictures (frames) we compute

$$\lambda = \frac{\left[\frac{\sigma_1^2 + \sigma_2^2}{2} + \left(\frac{\mu_1 - \mu_2}{2}\right)^2\right]^2}{\sigma_1^2 \sigma_2^2}$$

$$\text{check } \mathrm{DP}_{jk} = \begin{cases} 1 & \lambda > T \\ 0 & \text{otherwise} \end{cases}$$

2. matching - requires accurate segmentation

Main problem:

1. Noise

2. slow moving and small objects

3. changes in illumination and camera position can introduce false changes

4. texture - difference of frames only measures intensity differences

techniques

1. static segmentation and pairing of features - very expensive

2. segmentation based on motion - especially for SCMO

Time varying edge detection

$$E_t = \frac{dF}{ds} \bullet \frac{dF}{dt} = \text{spatial edges} \times \text{movement}$$

Motion analysis or estimation is dealt with in some detail in Chapters 18 and 19 and there we point out that it is a difficult and unsolved problem. Because of this and also the encoding time constraint, most motion analysis currently used in image compression takes the form of simple inter-frame block matching algorithms which can only approximately represent or model most motion. In the special case of translation of a single object in a plane parallel top the image plane then block matching will suffice. In any other motion, block matching will not work theoretically. That it does work is due to the fact that in practice motion is usually slow with respect to the frame rate or temporal sampling frequency. If we had detailed motion estimation, that is, if we had the motion associated with each pixel, we would only have to initialize a sequence with intensity values and then propagate pixels at the receiver along their motion trajectory in the image plane. We would only ever have to transmit the information required to specify the pixel trajectories. In practice the image is divided up into blocks (usually 16×16) pixels and the pixel motion is encoded using a motion vector for the entire block together with the difference between the blocks.

In moving image compression we can define two processes: motion estimation, or the motion analysis problem enumerated in Chapter 18, followed by the exploitation of this information in the encoding process. This is called motion compensation. It is fairly obvious that, for a given image resolution, the greater the degree of exploitation of motion estimation to enable higher compression, the more accuracy is required from the motion analysis. So the main trade-off in motion compensation is the degree of compression vs. the need for accurate motion estimation with its consequent difficulty and time penalty.

Once motion has been estimated it can be used in various ways. Compression can be achieved by skipping frames and using the motion information in the decoder to reconstruct the missing frames by interpolation along the motion trajectory. Alternatively the information in the current frame can be encoded by references to the previous frame.

The most common form of motion analysis used in image compression is simple block matching. In block matching algorithms, given a block in the frame currently being encoded, we need to find a block that matches it in a reference frame under a transformation. Using the same terms as in fractal encoding of images above, we need to find the domain block (in the reference frame) that under a transformation $\boldsymbol{T}$ matches a range block in the frame to be encoded. The idea of motion estimation and compensation in an image sequence is shown in Figure 26.12. An area in the current image – the range block – is constructed by referring to a matching area – the domain block – in the previous or reference image. The information required to do this is the domain block label and its motion between frames.

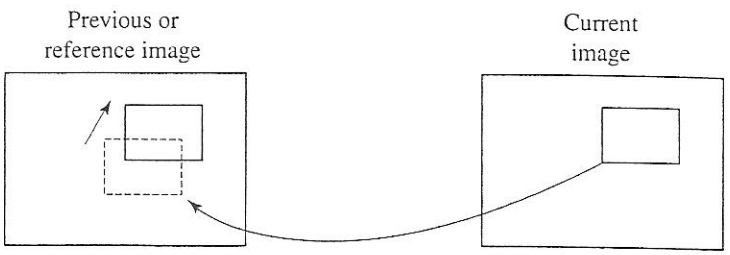

Figure 26.12
Motion compensation: a region in the current image is constructed from a (translated) region in the previous or reference image.

The term 'motion vector' is often used to specify this transformation. This should not be confused with the same term that is used for the two-dimensional motion of individual pixels due to the projection of real motion into the image plane (see Chapter 18). In block matching algorithms we are saying that it is likely that we can find a match between a range block and a domain block under

IMAGE FLOW

velocity field due to motion

E(x,y,t) = image intensity

$$\text{small motion} \Rightarrow \frac{dE}{dt} = 0$$

by the chain rule:

$$0 = \frac{dE}{dt} = \frac{\partial E}{\partial x}\frac{dx}{dt} + \frac{\partial E}{\partial y}\frac{dy}{dt} + \frac{\partial E}{\partial t}$$
$$= \frac{\partial E}{\partial x}u + \frac{\partial E}{\partial y}v + \frac{\partial E}{\partial t}$$

So

$$E_x u + E_y v + E_t = 0$$

This is one equation for two unknowns (aperture problem). One point does not give the image flow velocity. We add a smoothness constraint

$$\min \int\int \left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 \right] dxdy$$

Using Lagrange multipliers we minimize

$$\min \int\int \left(E_x u + E_y v + E_t\right)^2 + v^2 \left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 \right] dxdy$$

by the calculus of variations this leads to

$$v^2 \Delta u = E_x{}^2 u + E_x E_y v + E_x E_t = E_x \left(E_x u + E_y v + E_t\right)$$
$$v^2 \Delta v = E_x E_y u + E_y{}^2 v + E_y E_t = E_y \left(E_x u + E_y v + E_t\right)$$

Replace by finite differences and iterate - smooths discontinuities

We can replace this by

$$u = u_{\text{average}} - E_x \frac{P}{D}$$

$$v = v_{\text{average}} - E_y \frac{P}{D}$$

$$P = E_x u_{\text{average}} + E_y v_{\text{average}} + E_t$$

$$D = \lambda^2 + E_x{}^2 + E_y{}^2$$

We again iterate.