Multimedia - Department of Computing, Imperial College
Professor GZ Yang ● **http://www.doc.ic.ac.uk/~gzy**

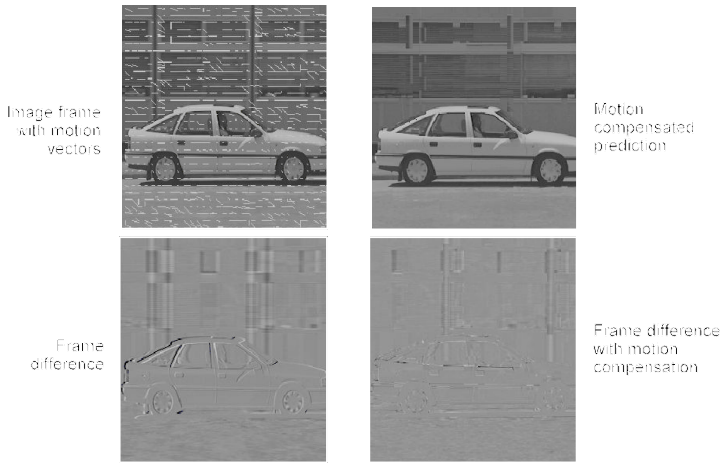
Sources and Types of Motion

In recent years, motion picture processing is becoming one of the most important topics in multimedia development, both in research and industrial applications. This is not only due to the challenging nature of motion processing, but also due to its potential commercial impact in effective multimedia delivery. Most concepts currently adopted for motion picture processing in multimedia have their roots in Computer Vision research. In this case, the input to the system is a sequence of image frames taken from a changing world. The camera used to acquire the image sequence may also be in motion. Each frame represents an image of the scene at a particular instant in time. The changes in a scene may be due to the motion of the camera, the motion of the objects, illumination changes, or changes in the structure, size, or shape of the object. It is usually assumed that the changes in a scene are due to camera and /or object motion, and that objects are either rigid or quasi-rigid. The system must detect changes, determine the motion characteristics of the camera and the objects, characterise the motion using high-level abstraction, recover the structure of the object, and recognise moving objects.

In multimedia, the main purposes of motion processing are to perform

- motion compensated prediction for image compression
- motion compensated interpolation for generating views between frames
- image sequence analysis for video segmentation

Since in most video sequences, the moving parts within the image are usually limited to a small region of interest, the use of temporal changes between frames permits an effective way of data compression. The following figure illustrates one image frame extracted from a video sequence with motion vectors superimposed. The inter-frame difference is also shown. It is evident that the difference image contains less details and therefore can be compressed more rigorously. This can be regarded as an extension to our previously described DPCM concept. However, if we know exactly how the motion vectors are distributed, we can warp the original image by using the motion vectors to derive a motion compensated prediction image. Since the correspondence between the warped image and the next image frame is high, the difference between the two is expected to be rather small, as illustrated by the following figure. In motion picture compression, one only needs to transmit the first image frame followed by motion vectors. Since the motion vectors themselves are sparse and in most cases are smoothly varying, a very high compression ratio can be achieved. This is the basic idea behind multimedia motion picture processing. Of course, there are many technical details need to be addressed before it can be practically useful.



Source: Inald Lagendijk of Delft Univers

Block Matching and Motion Estimation

One of the most common techniques used for motion estimation is through block matching. The technique assumes a 2D translational model for each block and a single vector is found by minimising the total pixel intensity difference. In practice, a distortion function is used to quantify the similarity between the source and target blocks. Due to the amount of data that needs to be processed, the selected distortion function should be easy to compute and result in good matches. Two most commonly used matching criteria are mean absolute difference (MAD) and mean square difference (MSD). Mathematically, they can be expressed by the following two equations

$$MSD(dx,dy) = \frac{1}{mn}\sum_{p=1}^{m}\sum_{q=1}^{n}\left[A(p,q) - B(p+dx,q+dy)\right]^2$$

and

$$MAD(dx,dy) = \frac{1}{mn}\sum_{p=1}^{m}\sum_{q=1}^{n}\left|A(p,q) - B(p+dx,q+dy)\right|$$

where A and B are the source and target blocks and (dx, dy) is the motion vector. The advantage of a cost function such as the MAD is its simplicity, which has a very direct implementation in hardware. Of course, as a result of its simplicity, it sacrifices some of the attraction of the MSD. The MAD tends to overemphasize small differences over large differences, and may therefore cause fast search algorithms to converge less quickly, or even to give different results than would the MSD.

Other matching criteria have also be used in practice, they include Pel Difference Classification (PDC) and Integral Projection (IP). PDC calculates block matching by comparing the source and target blocks pixel by pixel, and a pixel pair is a match if

the difference is smaller than a predefined threshold. Therefore, the greater the number of matching pixels, the better the match. IP, on the other hand, performs matching by summing the values of pixels from each column and row of a given block. This is essentially a low pass filtering process and therefore the technique is less sensitive to noise. Mathematically, they can be represented as

$$PDC(dx,dy) = \sum_{p=1}^{m}\sum_{q=1}^{n}\left[\left|A(p,q) - B(p+dx,q+dy)\right| \le t \;?\; 1 : 0\right]$$

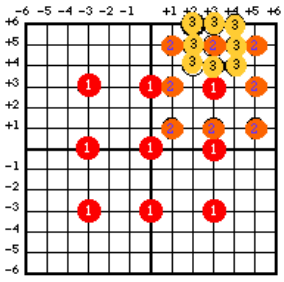
and

$$IP(dx,dy) = \sum_{p=1}^{m}\left|\sum_{q=1}^{n} A(p,q) - \sum_{q=1}^{n} B(p+dx,q+dy)\right| + \sum_{q=1}^{n}\left|\sum_{p=1}^{m} A(p,q) - \sum_{p=1}^{m} B(p+dx,q+dy)\right|$$

Finding the minimum distance can be guaranteed only by an exhaustive search of candidate points within the search range. However, this full-search scheme requires a significant amount of computational power. Hence in the 1980's, several fast search algorithms were exploited as a substitute of the full-search. Most sub-optimal search strategies rely on the principle of locality, which suggests that very good matches, if they exist, are likely to be found in the neighbourhood of other good matches. For example, when using a two level hierarchical search, one can first examine a number of sparsely spaced candidate blocks from the search area and choose the best match as the centre of a second, and a finer, search. The following are popular sub-optimal search strategies that have been used in practice.

Three Step Search (TSS)

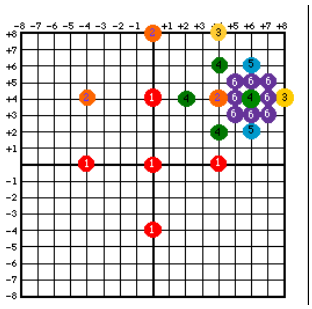
```
Given a maximum displacement d, set step size s = d/2.
Given a center [cx,cy], test nine points: [cx +/- 0 or s, cy
+/- 0 or s]. Take best match as new center, s = s/2 ,
repeat.
The first description of TSS uses a maximum displacement of
+/- 6, hence the name.
```



Two Dimensional Logarithmic Search (TDL)

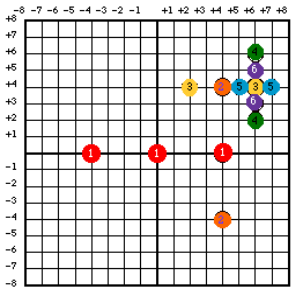
1st block matching algorithm to exploit the quadrant monotonic model.

```
Given a center [cx, cy], test 5 points: [cx, cy] & [cx +/- s, cy +/- s].
If [cx, cy] is the best match, s = s/2, repeat;
else if [a, b] is the best match, take it as the new center and repeat.
If s = 1, then all nine points around the center are tested. Take best
match.
All points outside the search area are ignored.
```



Orthogonal Search Algorithm (OSA)

```
Given a center [cx, cy], test 3 points: [cx, cy], [cx-s, cy], [cx+s, cy].
Let [a, b] be the best match, test 3 points: [a, b], [a, b+s], [a, b-s].
Take best match as new center, set s = s/2, repeat.
```



These search strategies assume that the matching is quadrant monotonic, i.e., the value of the distortion function increases as the distance from the point of minimum distortion increases. In other words, if there is a local minimum, these techniques can be trapped and never be able to find the global minimum. This is why these algorithms are called sub-optimal search strategies. The main drawbacks of block matching based techniques are due to their computational complexity and noise sensitivity. Furthermore, the estimated vector fields are sometimes not smooth, which can cause problems to the final compression ratio. To circumvent these problems, a hierarchical matching strategy has been used. The method starts with estimation on low resolution image frames and propagate vectors to higher resolution levels. During the process, it allows refinement of propagated motion vectors.

Another refinement to the aforementioned techniques is to relax the constraint from 2D rigid transformation to 2D local affine transformation. A rigid-body transformation is composed of a combination of a rotation, translation, and scale change. In 2D case, this can be represented as

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} t_x \\ t_y \end{pmatrix} + s \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Since the rotation matrix is orthogonal (the rows or columns are perpendicular to each other), the angles and lengths in the original image are preserved after registration. The scalar scale factor s, on the other hand, ensures that the changes in length relative to the original image are uniform for x and y. It is apparent that the transformation represented above can not account for more general spatial distortions

such as shear (skew). In this case, a more general transformation called affine transformation has to be used, and

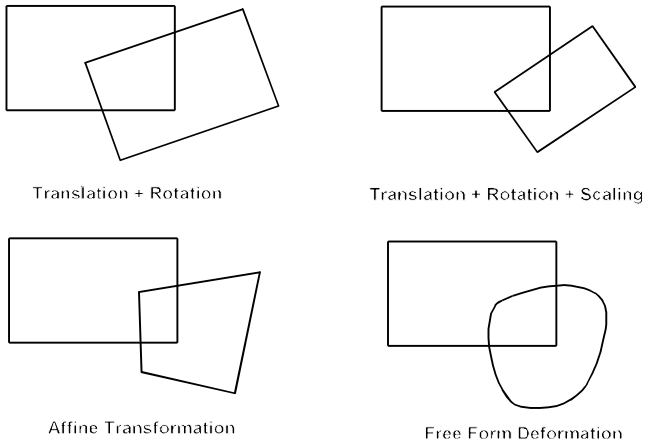
$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a_{13} \\ a_{23} \end{pmatrix} + \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Affine transformation does not have the properties associated with the orthogonal rotation matrix, angles and lengths are no longer preserved. However, parallel lines in the original image do remain parallel after transformation. Shear along x or y axis can thus be represented as

$$Shear_x = \begin{pmatrix} 1 & \alpha \\ 0 & 1 \end{pmatrix}, \quad and \quad Shear_y = \begin{pmatrix} 1 & 0 \\ \alpha & 1 \end{pmatrix}$$

respectively.

To account for more general deformation, higher order transformation has to be used. This can be achieved either by using bi-variate polynomial transformations, or general



spline approach to piecewise interpolation. In the case polynomial transformation, the following mapping functions are used:

$$x' = \sum_{i=0}^{m} \sum_{j=0}^{n} a_{ij} x^i y^j$$

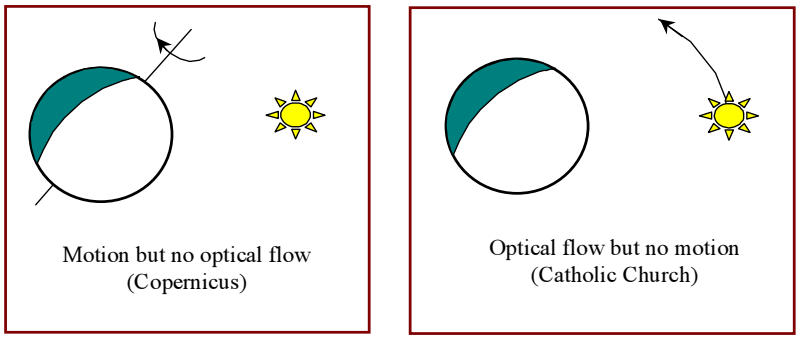
$$y' = \sum_{i=0}^{m} \sum_{j=0}^{n} b_{ij} x^i y^j$$

The order of the polynomial depends on the trade-off between accuracy and computational cost. In many cases, as the order of the polynomial transformation increases, the dependencies between the parameters multiplies, using the normal equations to solve the least squares approximation can become computationally expensive and inaccurate. This can be alleviated by using orthogonal polynomials which basically involves representing the original polynomial mapping as a combination of orthogonal polynomials that constructed from linearly independent functions. Because the polynomials are orthogonal, the dependencies between parameters are simplified, and the parameters of the new representation can be found analytically, i.e., there is no need to solve a system of linear equations.

Optical Flow

The difficulties of first of all matching points on moving objects, followed by measuring positions and velocities with sufficient accuracy, has prompted a local approach to the problem where the intensity change at one pixel is considered. Some assumptions need to be made. The following diagram shows that a rotating sphere, with fixed light and camera positions will not show any changes of pixel intensities, whereas a sphere that is stationary relative to a moving light source will display intensity changes. Similarly translation relative to a stationary light source will produce intensity changes. We assume that the lighting is stationary, and consider the movement of rigid objects relative to a fixed camera position.

Let us express the intensity at a pixel as f(x,y,t), and consider the intensity after a



small change in time and position: f(x+dx,y+dy,t+dt). The Taylor series expansion for this is:

$$f(x,y,t)+\left(\frac{\partial f}{\partial x}\right)dx+\left(\frac{\partial f}{\partial y}\right)dy+\left(\frac{\partial f}{\partial t}\right)dt+higher\ order\ terms$$

Now we come to the tricky bit. Remember that we are translating an object by a small amount, [dx,dy] in the image frame. If this is sufficiently small relative to the camera and illumination, the illumination function of that point should not change, so:

$$f(x+dx,y+dy,t+dt)=f(x,y,t)$$

so, ignoring the higher order terms:

$$\left(\frac{\partial f}{\partial x}\right)dx+\left(\frac{\partial f}{\partial y}\right)dy+\left(\frac{\partial f}{\partial t}\right)dt=0$$

and using the previous notation for velocity we have that:

$$u=\frac{dx}{dt}\quad and\quad v=\frac{dy}{dt}$$

so

$$\left(\frac{\partial f}{\partial x}\right)u+\left(\frac{\partial f}{\partial y}\right)v+\left(\frac{\partial f}{\partial t}\right)=0$$

it is possible to measure the three partial derivatives directly from the image changes at each pixel in the image. This equation is a constraint equation, which constrains the velocities to lie on a line in the *u-v* space. The absolute values must be found by a search (relaxation) technique. For example, if we assume that we have a measure of the velocity at each pixel (u_i, v_i), then we can estimate the square of the error at each pixel using:

$$R(p_i) = \left\{ \frac{\partial f}{\partial x} u_i + \frac{\partial f}{\partial y} v_i + \frac{\partial f}{\partial t} \right\}^2$$

We can also include a term related to smoothness at the pixel Pi using:

$$S(p_i) = \left\{ \left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 \right\}$$

These two terms combine into one error equation with the usual fiddle factor λ:

$$E^2(p_i) = R(p_i) + \lambda S(p_i)$$

We can get a global error by integrating (summing) over every pixel in the image. To find the *u-v* values that minimise E, we can differentiate the error equation with respect to u and v, and set the result to zero. A technique called the calculus of variations can be employed. I don't expect you to follow the results shown below, as they go beyond the scope of this course. For those who are interested, the results are:

$$u = \bar{u} - \frac{\partial f}{\partial x} Q, \quad and \quad v = \bar{v} - \frac{\partial f}{\partial y} Q,$$

where

$$Q = \frac{\left(\frac{\partial f}{\partial x}\right)\bar{u} + \left(\frac{\partial f}{\partial y}\right)\bar{v}}{\lambda^2 + \left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

and $\bar{u}$ and $\bar{v}$ are the average velocities of the pixels that neighbour the one being calculated. The relaxation iteration takes place over every pixel in the image until a stable solution is found. An alternative to iterating over the estimates based on two frames is to use a sequence of frames:

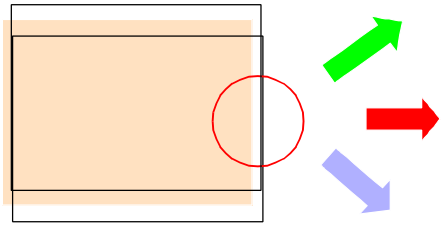
$$u(x, y, t) = \bar{u}(x, y, t-1) - \left(\frac{\partial f}{\partial x}\right) Q$$

$$v(x, y, t) = \bar{v}(x, y, t-1) - \left(\frac{\partial f}{\partial y}\right) Q$$

Challenges in Motion Estimation

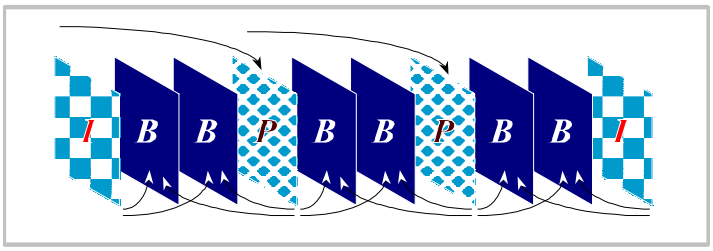
It must be noted that motion estimation is a challenging problem. Motion effects are local and vary throughout the frame. The use of 2D projection of the 3D world to calculate motion is incomplete and imperfect. Problems such as occlusion are difficult to avoid. Temporal variations in frame contents may be due to changes in illumination and shadows rather than object motion. None of the algorithms described above can

handle these problems. The use of optical flow allows dense motion distribution to be derived, however, the technique is associated with one major problem called the aperture problem. Because the algorithm is based on local intensity variations, it causes local ambiguity in the derived motion fields as indicated in the following figure. Different directional motion of the object can result in the same local motion estimation within the highlighted area. The problem can only be resolved by using higher levels of processing through object recognition.



MPEG Video Coding - an overview

Motion estimation plays a key part in MPEG image coding. MPEG-1 is the standard for storage and retrieval of moving pictures and audio on storage media. MPEG-1 provides a nominal data stream compression rate of about 1.2 Mbits per second -the typical CD-ROM data transfer rate -but can deliver data at a rate of up to 1,856,000 bps. MPEG-1 distinguishes four types of image coding for processing: I (intra-coded pictures), P (predictive coded pictures), B (bidirectionally predictive pictures), and D-Frame (coding based on discrete cosine only parameter) images.



I-frames (Intra-coded frames)
These are self-contained image frames, coded without the need for reference to other frames. Use JPEG for I-frame encoding. Lowest compression as no temporal redundancy exploited. Provides points for random access in an MPEG stream.

P-frames (Predictive-coded frames)
They require previous I, P frames for encoding and decoding. Temporal redundancy is exploited through block matching. Only motion vectors and small difference of the macroblocks between the reference and target are encoded. For motion estimation, the algorithm aims to find a nearby macro-block (MB) such that the sum of the differences of the luminance component of MB and the target macro-block is minimum. Since motion vectors of adjacent macro blocks often differ only slightly, only the differences are encoded.

B-frames (Bi-directional frames)
They are bi-directional frames for searching in past and future frames for macro block. It also facilitates backward playback.

To allow audio compression in acceptable quality, MPEG-1 enables audio data rates between 32 and 448 Kbps. MPEG-1 explicitly considers other standards and functionalities, such as JPEG and H.261, suitable for symmetric and asymmetric compression. It also provides a system definition to specify the combination of several individual data streams. Note that MPEG-1 doesn't prescribe compression in real time. Furthermore, though MPEG-1 defines the process of decoding, it doesn't define the decoder itself. The quality of an MPEG-1 video without sound at roughly 1.2 Mbps (the single-speed CD-ROM transfer rate) is equivalent to a VHS recording.

MPEG-2, the digital television standard, strives for a higher resolution—up to 100 Mbps—that resembles the digital video studio standard CCIR 601 and the video quality needed in HDTV. As a compatible extension to MPEG-1, MPEG-2 supports interlaced video formats and a number of other advanced features, such as those to support HDTV. As a generic standard, MPEG-2 was defined in terms of extensible profiles, each supporting the feature required by an important application class. The Main Profile, for example, supports digital video transmission at a range of 2 to 80 Mbps over cable, satellite, and other broadcast channels. Furthermore, it supports digital storage and other communications applications. An essential extension from MPEG-1 to MPEG-2 is the ability to scale the compressed video, which allows the encoding of video at different qualities (spatial-, rate-, and amplitude-based scaling2). The MPEG-2 audio coding was developed for low bit-rate coding of multichannel audio. MPEG-2 extends the MPEG-1 standard by providing five full bandwidth channels, two surround channels, one channel to improve low frequencies, and/or seven multilingual channels, and the coding of mono and stereo (at 16 kHz, 22.05 kHz, and 24 kHz). Nevertheless, MPEG-2 is still backward compatible with MPEG-1.

Though the results of MPEG-1 and MPEG-2 served well for wide-ranging developments in such fields as interactive video, CD-ROM, and digital TV, it soon became apparent that multimedia applications required more than the established achievements. Thus, in 1993 MPEG started working to provide the standardized technological elements enabling the integration of the production, distribution, and content access paradigms of digital TV, interactive graphics applications (synthetic content), and interactive multimedia (distribution of and access to enhanced content on the Web). MPEG-4 version 1, formally called ISO/IEC 14496, has been available as an international standard since December 1998. MPEG-4 aims to provide a set of technologies to satisfy the needs of authors, service providers, and end users, by avoiding the emergence of a multitude of proprietary, incompatible formats and players. The standard should allow the development of systems that can be configured for a vast number of applications (among others, real-time communications, surveillance, and mobile multimedia).

The preceding MPEG standards have mainly addressed coded representation of audio-visual information. MPEG-7, on the other hand, focuses on the standardization of a common interface for describing multimedia materials (representing information about the content, but not the content itself-the bits about the bits). In this context, MPEG-7 addresses aspects such as facilitating interoperability and globalization of data resources and flexibility of data management. Thus, the commonalties between previous MPEG standards and MPEG-7 rely on the fact that previous standards can use MPEG-7 descriptions to improve their facilities of content description. This means that we can use MPEG-7 independently of the other MPEG standards - for example, conceptually a description might even be attached to a celluloid film.