7º CAPÍTULO

Análise de Movimento

Prof. Arnaldo Abrantes

Movimento 2D

- Análise de movimento em vídeo é importante – permite revelar características (*features*) úteis, como a velocidade dos objectos ou do observador, a forma dos objectos, as suas trajectórias
  - Ao observar as alterações na intensidade (ou cor) dos pixels é possível detectar a presença de objectos e fazer o seu reconhecimento
- A detecção de alguns eventos (exemplo, transições de cena) em sequências longas de vídeo permite o seu fraccionamento (em diversos *clips*) tornando mais fácil operações como o acesso, a análise e a edição
- Quatro situações onde ocorre movimento (alterações nos pixels):
  - Câmara fixa, um só objecto em movimento, fundo constante
    - Situação simples de vigilância
  - Câmara fixa, vários objectos em movimento, fundo constante
    - Situação realista num contexto de vigilância
  - Câmara em movimento, cena relativamente imutável
    - Permite obter maior número de observações da cena, construir mosaicos, determinar profundidade relativa dos objectos, detectar eventos como *pan* ou *zoom* de câmara
  - Câmara em movimento, vários objectos em movimento
    - Veículo robótico a circular num ambiente de tráfego intenso

Métodos de subtracção de imagem

- Subtracção de imagens consecutivas

$$\left|I_n(r,c) - I_{n-1}(r,c)\right| > T_n(r,c)$$

- Subtração de fundo

$$\left|I_n(r,c) - B_n(r,c)\right| > T_n(r,c)$$

$$B_{n+1}(r,c) = \begin{cases} B_n(r,c) \quad se \quad (r,c) \text{ é pixel activo} \\ \alpha B_n(r,c) + (1-\alpha) I_n(r,c) \quad \text{caso contrário} \end{cases}$$

Algoritmo simples

- **Entrada**: duas imagens monocromáticas $I_n(r,c)$ e $I_{n-k}(r,c)$ e o limiar τ
- **Saída**: imagem binária, I_{out} e conjunto de caixas, ***B*,** com a localização dos objectos detectados

- Algoritmo com 5 passos:

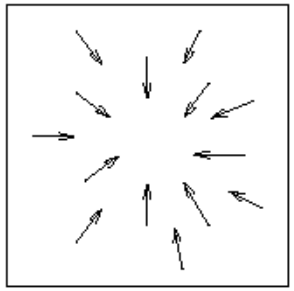
  1. Calcular imagem binária (pixels activos)

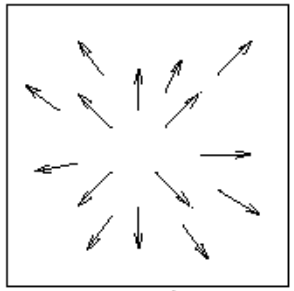
$$I_{out}(r,c) = \begin{cases} 1 & \text{se } \left|I_n(r,c) - I_{n-k}(r,c)\right| > \tau \\ 0 & \text{caso contrário} \end{cases}$$

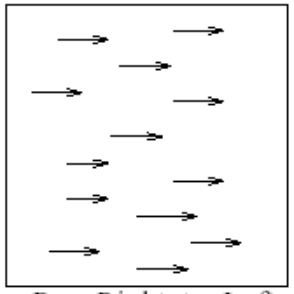
  2. Realizar extracção de componentes conexos sobre I_{out}
  3. Remover as regiões com pequena área (ruído)
  4. Realizar operação morfológica de fecho usando um pequeno disco
  5. Para cada região, determinar a caixa rectangular que a contém (*bounding box*)

Cálculo de vectores de movimento

- Um conjunto de vectores 2D representando o movimento aparente (na imagem) de objectos 3D é chamado um **campo de movimento**


Zoom out


Zoom in

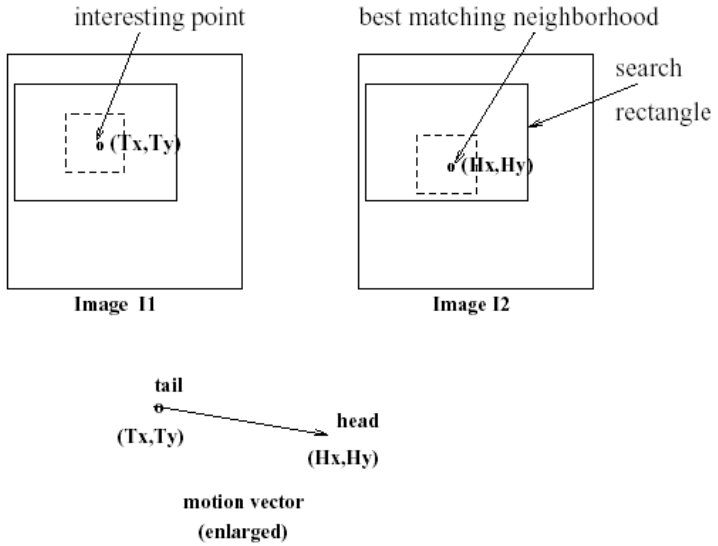

Pan Right to Left

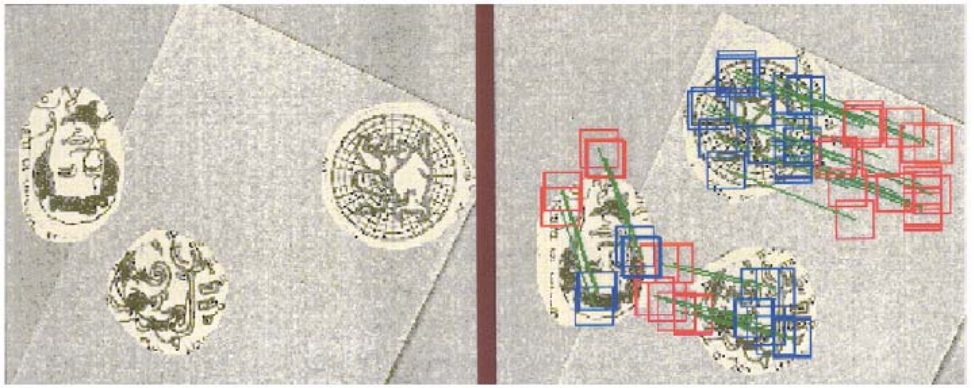
- O foco de expansão (**FOE**) é o ponto da imagem de onde todos os vectores divergem. O foco de contracção (**FOC**) é para onde todos os vectores convergem

- **Fluxo óptico** é o campo de movimento calculado sob a hipótese de que as intensidades em redor de pontos em correspondência nas duas imagens se mantêm constante

Método esparso – correspondências de pontos

- Detectar pontos de interesse na primeira imagem (por exemplo, algoritmo 9.2 na Shapiro, página 258, para detecção de cantos)
- Por cada ponto de interesse detectado na primeira imagem (Tx, Ty) procurar na segunda imagem o ponto correspondente (por exemplo, método de correlação)
- Caso o *match* seja bom, as coordenadas do ponto definem a cabeça do vector movimento (Hx,Hy)





Algoritmo análogo – compressão MPEG

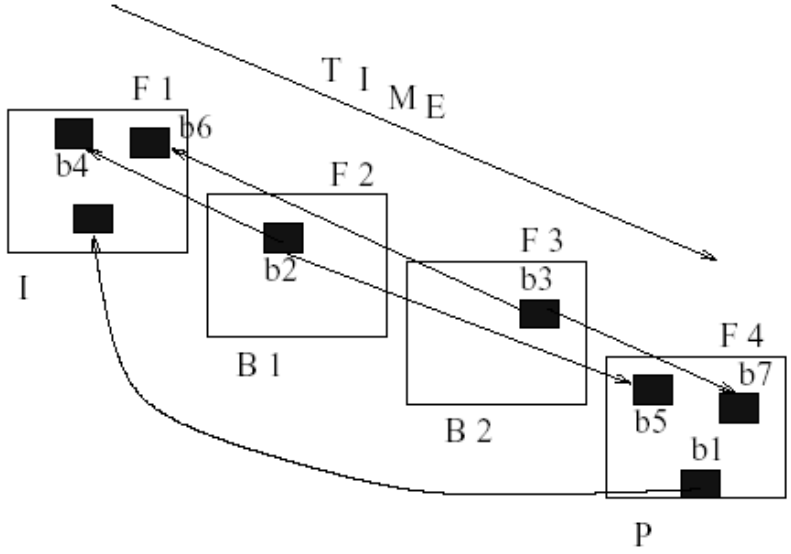


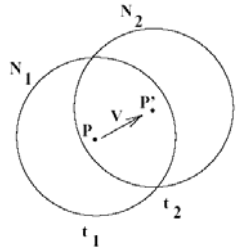
Figure 9.8: **A coarse view of the MPEG use of motion vectors to compress the video sequence of four frames F1, F2, F3, F4.** F1 is coded as an *independent* (**I**) frame using the JPEG scheme for single still images. F4 is a P frame *predicted* from F1 using motion vectors together with block differences: 16 x 16 pixel blocks (**b1**) are located in frame F1 using a motion vector and a block of differences to be added. *Between frames* B1 and B2 are determined entirely by interpolation using motion vectors: 16 x 16 blocks (**b2**) are reconstructed as an average of blocks (**b4**) in frame F1 and (**b5**) in frame F4. Between frames **F2** and F3 can only be decoded after predicted frame **F4** has been decoded even though these images were originally created before **F4**. Between frames yield the most compression since each 16 x 16 pixel block is represented by only two motion vectors. **I** frames yield the least compression.

- Representação da imagem (modelo contínuo) usando a série de Taylor (1ª ordem)

$$f(x+\Delta x, y+\Delta y, t+\Delta t) = f(x,y,t) + \frac{\partial f}{\partial x}\Delta x + \frac{\partial f}{\partial y}\Delta y + \frac{\partial f}{\partial t}\Delta t + h.o.t.$$

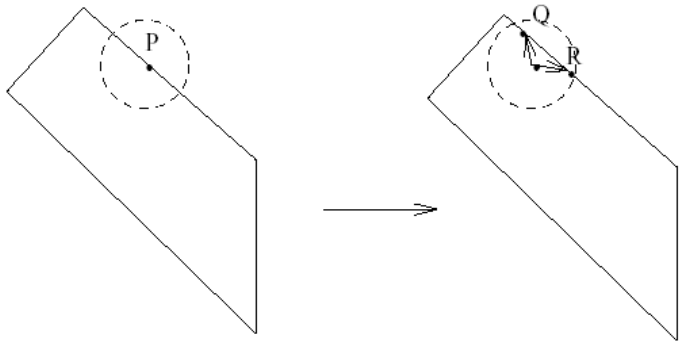
- Hipótese de intensidade constante

$$f(x+\Delta x, y+\Delta y, t+\Delta t) = f(x,y,t) \longrightarrow$$

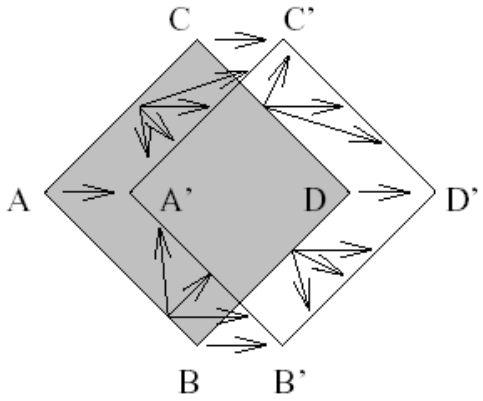


- Resultado:

$$-\frac{\partial f}{\partial t}\Delta t = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right) \circ (\Delta x, \Delta y)$$



Problema de abertura



Importância dos cantos

Algoritmo rápido (Freeman *et. al.*)

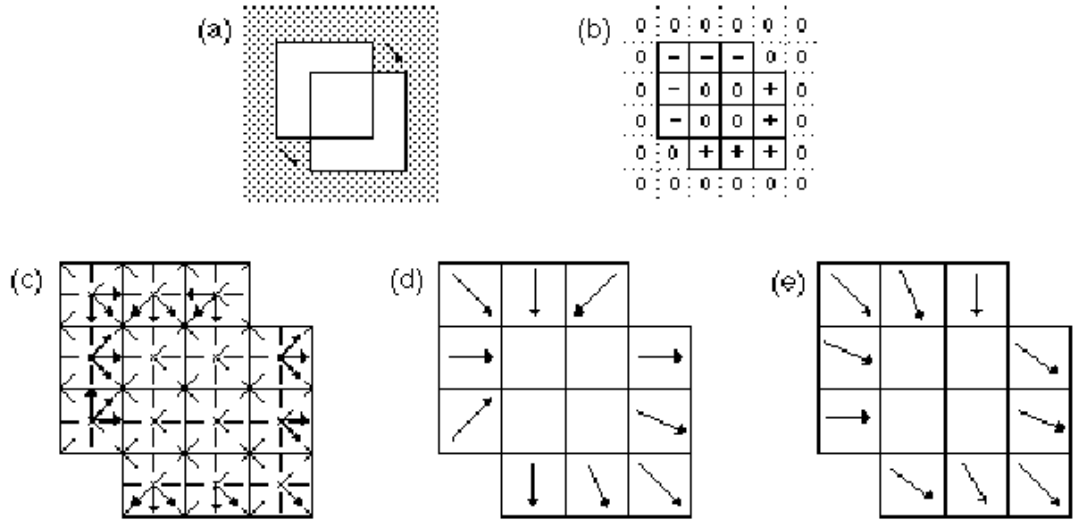
- Calcular a diferença, entre a imagem actual e a anterior

$$d_n(r,c) = I_n(r,c) - I_{n-1}(r,c)$$

- Para todos os pixels onde a diferença seja não nula (ou não muita pequena), enumerar todos os possíveis movimentos consistentes com as observações:
  1. Se $d_n(r,c) < 0$ então é porque o movimento se faz em direcção ao vizinho com maior intensidade
  2. Se $d_n(r,c) > 0$ então é porque o movimento se faz em direcção ao vizinho com menor intensidade
- Considerar, no passo anterior, quatro direcções possíveis (quatro tipos de vizinhança): horizontal, vertical e as duas diagonais
- Tratar cada uma das 4 estimativas como vectores e realizar a sua soma, para cada pixel
- Finalmente, estimar o fluxo óptico em cada pixel como a média espacial, calculada numa vizinhança 3x3

Ilustração do algoritmo rápido

	(1)	(2)	(3)	(4)
edge contrast and its motion direction	light, dark, →	light, dark, ←	light, dark, →	light, dark, ←
pixel value subtraction (positive or negative)	0, ⊖	⊕	⊕	⊖



- Objectivo: Detecção de alterações significativas em vídeo
  - Mudança de cena
  - Mudança de shot
  - Pan
  - Zoom in e zoom out
  - Efeitos especiais
    - fade in, fade out $C = A\left(1-\frac{t}{T}\right)u(T-t) + B\frac{t-T}{T}u(t-T)$
    - Dissolve $C = A\left(1-\frac{t}{T}\right) + B\frac{t}{T}$
    - wipe

- Solução simples (e estúpida)

$$d(I_1, I_2) = \frac{\sum_{r=0}^{N-1}\sum_{c=0}^{M-1}\left|I_1(r,c) - I_2(r,c)\right|}{NM}$$

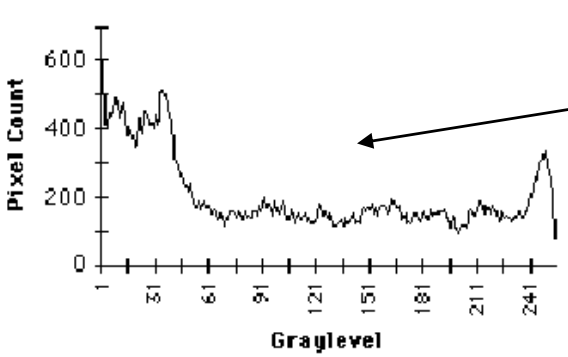
- Melhor solução: partir a imagem em blocos rectangulares e calcular médias e variâncias em cada bloco

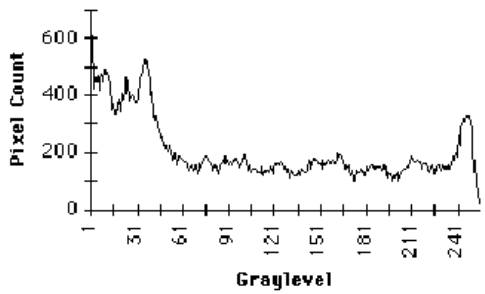
$$d_{block}\left(B_i, B_j\right) = \begin{cases} 1 & se \quad r > \tau \\ 0 & se \quad r \le \tau \end{cases} \qquad r = \frac{\left[\frac{\sigma_i^2 + \sigma_j^2}{2} + \left(\frac{\mu_i - \mu_j}{2}\right)^2\right]^2}{\sigma_i^2 \sigma_j^2}$$

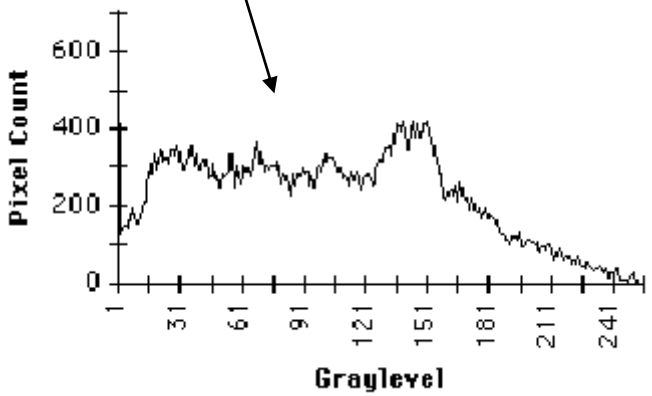
$$d(I_1, I_2) = \sum_{B_{1i} \in I_1; B_{2i} \in I_2} d_{block}(B_{1i}, B_{2i})$$

Alternativa – distâncias entre histogramas

Distâncias entre histogramas







- Fade out seguido de fade in (duração 2T)

$$C = A\left(1 - \frac{t}{T}\right)u(T - t) + B\frac{t - T}{T}u(t - T)$$

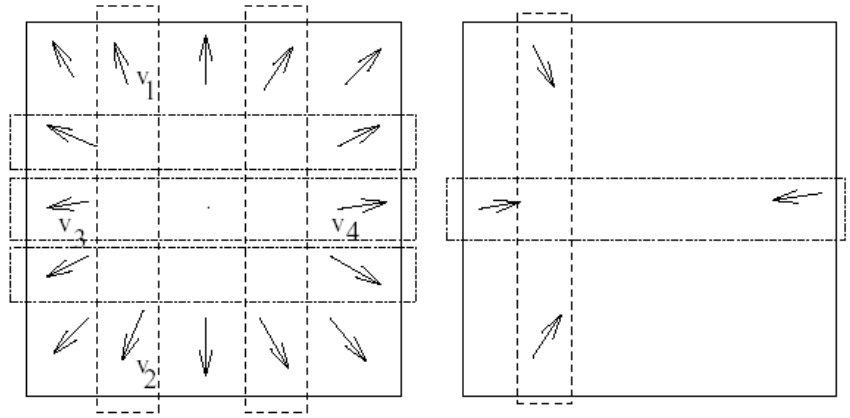
- Dissolve

$$C = A\left(1 - \frac{t}{T}\right) + B\frac{t}{T}$$

- Wipe

Como detectar certos efeitos de câmara

- Detecção de zoom



Heuristica horizontal/vertical

$$\left|v_{1r} - v_{2r}\right| > \max\left\{\left|v_{1r}\right|, \left|v_{2r}\right|\right\}$$

$$\left|v_{3c} - v_{4c}\right| > \max\left\{\left|v_{3c}\right|, \left|v_{4c}\right|\right\}$$

Armazenar vídeo

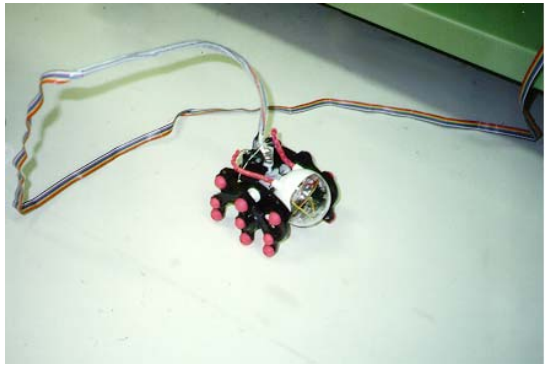
- Detecção de shots
- Keyframes
- Clips de vídeo; acesso aleatório
- Bases de dados de imagens

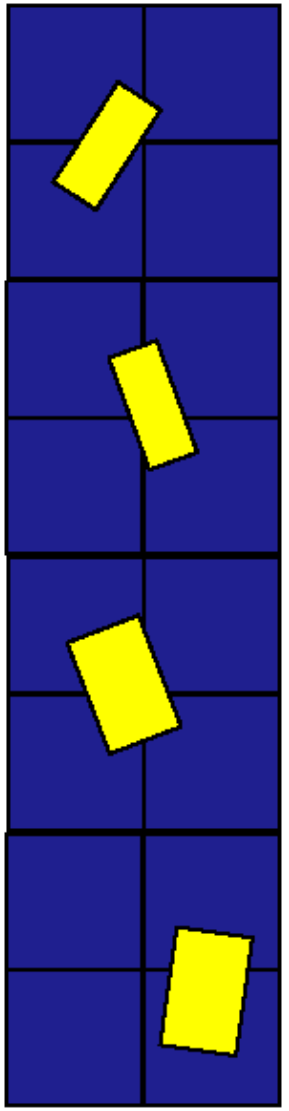
Computer Vision for Interactive Computer Graphics

- Interacção Pessoa – Máquina
- Os computadores são capazes de interpretar movimentos, gestos e olhares do utilizador
- Algoritmos visuais de suporte:
  - Seguimento de objectos;
  - Reconhecimento de formas;
  - Análise de Movimento.
- Em aplicações gráficas interactivas, estes algoritmos necessitam de ser: robustos, rápidos e correr em hardware genérico.
- Felizmente, as aplicações interactivas também facilitam o problema da visão:
  - Restringe o número de interpretações possíveis.
  - Fornece retroacção visual útil

- Aplicações interactivas colocam desafios particulares:
  - O tempo de resposta deverá ser muito rápido
  - Os algoritmos de visão por computador devem ser robustos, funcionarem para diferentes tipos de pessoas e fundos diferenciados.
  - Devem ser baratos. Um joystick actual ou comando de televisão custa 40€. Mesmo com aumento de funcionalidades os consumidores não quererão pagar muito mais.
- As boas notícias:
  - O contexto da aplicação restringe muito as interpretações visuais possíveis (pelo contexto do jogo sabe-se que a pessoa está a correr; apenas importa definir a velocidade com que a corrida é realizada)
  - Existe um ser inteligente (humano) no ciclo. O utilizador pode explorar a rápida retroacção visual que lhe é fornecida via display gráfico para alterar o seu gesto, se necessário, para alcançar o efeito desejado. Se o jogador se inclina para fazer uma curva e observa que não está a curvar o suficiente, então dever-se-á inclinar mais.

- Existe um nicho para aplicação de algoritmos de visão rápidos e pouco sofisticados que tiram vantagem das simplificações que as aplicações gráficas interactivas introduzem.
- Semelhante em espírito a uma corrente na visão artificial, chamada de visão activa, que enfatiza (tira partido) da resposta em tempo-real dos sistemas reais de visão.





- Visão (uma câmara) pode ser um potente dispositivo de interface em computadores
  - Posição do corpo
  - Orientação da cabeça
  - Direcção do olhar
  - Gestos
- Aplicações podem incluir:
  - Controlo (via computador) de jogos ou máquinas
  - Interface natural com o computador
- Em vez de premir botões, o jogador pode realizar mímica de acções e gestos, que o computador reconhece
- Um utilizador de um máquina pode dar comandos através de gestos manuais (útil em cirurgiões, soldadores, deficientes).