Computing Motion from Images

Chapter 9

General topics

- Low level change detection
- Region tracking or matching over time
- Interpretation of motion
- MPEG compression
- Interpretation of scene changes in video
- Understanding human activities
Motion important to human vision

1. definitely used in human vision
2. object detection and tracking
3. navigation and obstacle avoidance
4. analysis of actions in a scene
5. segmentation and understanding of video

What’s moving: different cases

- still camera, single moving object, constant background
- still camera, several moving objects, constant background
- moving camera, relatively constant scene
- moving camera, several moving objects
Image subtraction

Simple method to remove unchanging background from moving regions.

Change detection for surveillance

- (Left) person appears in an unoccupied workspace
- (Center) Image subtraction reveals changed regions where person occludes background and at door and a CRT.
- (Right) change due to person is deemed significant while the other two are expected and hence ignored.
Change detection by image subtraction

Input $I[r,c]$ and ${I}_{t-d}[r,c]$: two monochrome input images taken $d$ seconds apart.
Input $r$ is an intensity threshold.
$I_{out}[r,c]$ is the binary output image; $B$ is a set of bounding boxes.

1. For all pixels $[r,c]$ in the input images,
   set $I_{out}[r,c] = 1$ if $(|I[r,c] - {I}_{t-d}[r,c]| > r)$
   set $I_{out}[r,c] = 0$ otherwise.
2. Perform connected components extraction on $I_{out}$.
3. Remove small regions assuming they are noise.
4. Perform a closing of $I_{out}$ using a small disk to fuse neighboring regions.
5. Compute the bounding boxes of all remaining regions of changed pixels.
6. Return $I_{out}[r,c]$ and the bounding boxes $B$ of regions of changed pixels.

What to do with regions of change?

- Discard small regions
- Discard regions of non interesting features
- Keep track of regions with interesting features
- Track in future frames from motion plus component features
Some effects of camera motion that can cause problems

- Effects of zooming and panning on imaged features.
- The effect of zoom in is similar to that observed when we move forward in a scene.
- The effect of panning is similar to that observed when we turn.

Motion field

1 Definition A 2D array of 2D vectors representing the motion of 3D scene points is called the motion field. The motion vectors in the image represent the displacements of the images of moving 3D points. Each motion vector might be formed with its tail at an imaged 3D point at time $t$ and its head at the image of that same 3D point imaged at time $t + \delta$. Alternatively, each motion vector might correspond to an instantaneous velocity estimate at time $t$. 
FOE and FOC

2 Definition The focus of expansion (FOE) is that image point from which all motion field vectors diverge. The FOE is typically the image of a 3D scene point toward which the sensor is moving. The focus of contraction (FOC) is that image point toward which all motion vectors converge, and is typically the image of a 3D scene point from which the sensor is receding.

Will return to use the FOE or FOC or detection of panning to determine what the camera is doing in video tapes.

Gaming using a camera to recognize the player’s motion

Decathlete game
Decathlete game

Cheap camera replaces usual mouse for input

- The man at the left is making running motions with his arms and hands to control the game of running the hurdles.

- The game display is shown at the right.

Running speed and jumping of the avatar is controlled by detected motion of the player’s hands.

Motion detection input device

- Running (hands)
- Jumping (hands)

from IEEE Computer Graphics, Vol 18, No. 3 (May-June 1998)
Motion analysis controls hurdling event (console)

- Top left shows video frame of player
- Middle left shows motion vectors from multiple frames
- Center shows jumping patterns

Related work

- Motion sensed by crude cameras
- Person dances/gestures in space
- System maps movement into music
- Creative environment?
- Good exercise room?
Computing motion vectors from corresponding “points”

High energy neighborhoods are used to define points for matching

Match points between frames

- Find interest points $P_{t,j}$ in frame $t$
- Search for matching points $Q_{t+\delta t,j}$ in frame $t + \delta t$
- Form motion vectors $V_j = [P_{t,j}, Q_{t+\delta t,j}]$
Requirements for interest points

- have unique multidirectional energy / texture
- detected and located with high confidence
- edge detector is not good – constraint in only 1 direction
- corner detector is better – constraint in 2 directions
- autocorrelation can be used for matching in second image

Match small neighborhood to small neighborhood. The previous “scene” contains several highly textured neighborhoods.

Interest = minimum directional variance

real procedure interest-operator (I, r, c, w )
{
   "w is the halfwidth of operator window"
   "See alternate texture-based interest operator in the exercises."
   v1 := variance of intensity of horizontal pixels I[r - w, c] . . . I[r + w, c];
   v2 := variance of intensity of vertical pixels I[r - w, c] . . . I[r - w, c];
   v3 := variance of intensity of diagonal pixels I[r - w, c - w] . . . I[r + w, c + w];
   v4 := variance of intensity of diagonal pixels I[r - w, c + w] . . . I[r + w, c - w];
   return minimum {v1, v2, v3, v4};
}

Used by Hans Moravec in his robot stereo vision system.
Interest points were used for stereo matching.
Detecting interest points in I1

\[\text{procedure } \text{detect\_corner\_points}(I, V);\]
\[\{\]
  \[\text{"I[r,c] is an input image of MaxRow rows and MaxCol columns"}\]
  \[\text{"V is an output set of interesting points from I"}\]
  \[\text{"r is a threshold on the interest operator output"}\]
  \[\text{"w is the halfwidth of the neighborhood for the interest operator"}\]
  \[\text{for } r := 0 \text{ to MaxRow - 1}\]
  \[\text{for } c := 0 \text{ to MaxCol - 1}\]
  \[\{\]
    \[\text{if } I[r,c] \text{ is a border pixel then break;}\]
    \[\text{else if } (\text{interest\_operator}(I, r, c, w) \geq r) \text{ then add } [(r,c)] \text{ to set V};\]
    \[\text{"The second (r,c) is a placeholder in case vector tip found later."}\]
  \[\}\]
\[\}\]

Match points from I1 in I2

\[I_1[r,c] \text{ and } I_2[r,c] \text{ are input images of MaxRow rows and MaxCol columns.}\]
\[V \text{ is the output set of motion vectors } \{(T_x, T_y), (H_x, H_y)\}\]
\[\text{where } (T_x, T_y) \text{ is the tail of a motion vector and } (H_x, H_y) \text{ is its head.}\]

\[\text{procedure } \text{extract\_motion\_field}(I_1, I_2, V);\]
\[\{\]
  \[\text{"Detect matching corner points and returning motion vectors V"}\]
  \[\text{"T2 is a threshold on neighborhood cross-correlation"}\]
  \[\text{detect\_corner\_points}(I_1, V);\]
  \[\text{for all vectors } [(T_x, T_y), (U_x, U_y)] \text{ in } V\]
  \[\text{match := best\_match( } I_1, I_2, T_x, T_y, H_x, H_y);\]
  \[\text{if } (\text{match} < T_2) \text{ then delete } [(T_x, T_y), (U_x, U_y)] \text{ from V};\]
  \[\text{else replace } [(T_x, T_y), (U_x, U_y)] \text{ with } [(T_x, T_y), (H_x, H_y)] \text{ in V};\]
\[\}\]
Search for best match of point P1 in nearby window of I2

```
real procedure best_match(I1, I2, Tx, Ty, Hx, Hy);

("(Hx, Hy) is returned as the center of the neighborhood in I2 that matches best"
"to the neighborhood centered at (Tx, Ty) in I1.")
{
    "first indicate that a good match has not been found"
    Hx := -1; Hy := -1; best := 0.0;
    for i := Ty - sh to Ty + sh
    for c := Tx - sw to Tx + sw
    {  
    "cross correlate N in I1 with N in I2 as in Chapter 5"
    match := cross_correlate(I1, I2, Tx, Ty, r, c, h, w);
    if (match > best) then
    {  
    Hx := i; Hy := c; best := match;
    }
    }
}
```

For both motion and stereo, we have some constraints on where to search for a matching interest point.

---

Motion vectors clustered to show 3 coherent regions

Motion coherence: points of same object tend to move in the same way

All motion vectors are clustered into 3 groups of similar vectors showing motion of 3 independent objects. (Dina Eldin)
Two frames of aerial imagery

Video frame N and N+1 shows slight movement: most pixels are same, just in different locations.

Can code frame N+d with displacements relative to frame N

- for each 16 x 16 block in the 2nd image
- find a closely matching block in the 1st image
- replace the 16x16 intensities by the location in the 1st image (dX, dY)
- 256 bytes replaced by 2 bytes!
- (If blocks differ too much, encode the differences to be added.)
Frame approximation

Left is original video frame N+1. Right is set of best image blocks taken from frame N. (Work of Dina Eldin)

Best matching blocks between video frames N+1 to N (motion vectors)

The bulk of the vectors show the true motion of the airplane taking the pictures. The long vectors are incorrect motion vectors, but they do work well for compression of image 12!

Best matches from 2nd to first image shown as vectors overlaid on the 2nd image. (Work by Dina Eldin.)
Motion coherence provides redundancy for compression

MPEG “motion compensation” represents motion of 16x16 pixels blocks, NOT objects

MPEG represents blocks that move by the motion vector
MPEG has ‘I’, ‘P’, and ‘B’ frames

- Example video sequence has four frames F1, F2, F3, F4
- F1 is coded as an *independent* (I) frame using JPEG
- 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
- 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

Computing Image Flow

- Optical Flow
  - The apparent flow of intensities across the retina due to motion in the scene or
    motion of the observer
  - Compute at each image point I[x,y,t] the spatio-temporal gradient, which represent
    a flow
Computing Image Flow

- The object reflectivity and the illumination of the object does not change during the interval [t1, t2].
- The distance of the object from the camera or light sources does not vary significantly over this interval.
- Each small intensity neighborhood $N_{x,y}$ at time $t1$ is observed in some shifted position $N_{x+\delta x, y+\delta y}$.
- We assume a continuous intensity function $f(x,y)$ of continuous spatial parameters.

Assumptions for Computing Image Flow

An example of image flow. A brighter triangle moves one pixel upward from time $t_1$ to time $t_2$. Background intensity is 3 while object intensity is 9.
Image Flow Equation

\[ f(x + \delta x, y + \delta y, t + \delta t) = f(x, y, t) + \frac{\partial f}{\partial x} \delta x + \frac{\partial f}{\partial y} \delta y + \frac{\partial f}{\partial t} \delta t + \text{h.o.t.} \]  

The image flow vector \( V = [\delta x, \delta y] \) carries the intensity neighborhood \( N_1 \) of \((x, y)\) at \( t_1 \) to an identical intensity neighborhood \( N_2 \) of \((x + \delta x, y + \delta y)\) at \( t_2 \). This assumption means that

\[ f(x + \delta x, y + \delta y, t + \delta t) = f(x, y, t) \]

\[ -\frac{\partial f}{\partial t} \delta t = \frac{\partial f}{\partial x} \delta x + \frac{\partial f}{\partial y} \delta y = [\delta x, \delta y] \circ \nabla f \circ [\delta x, \delta y] \]

Solving Image Flow by Propagating Constraints

A square object is moving toward the right. Motion vectors with their tails on the edge at time \( t_1 \) are constrained by a linear relationship that puts their heads on an edge at time \( t_2 \). Common constraints at the corners A,B,C,D force the move right interpretation, which can then be propagated to all edge points by enforcing consistency along the entire boundary.
Tracking object to object trajectories

Trajectories of three objects, ○, △, □ are shown: the location of each object is shown for six instants of time. ○ and △ are generally moving from left to right while □ is moving right to left.

- At each instant of time, which object is which?
- What features determine the objects?
- What constraints can we use from physics?
- What constraints can we use from media domain?