Computer Vision, Lectures 17,18

Professor Hager
http://www.cs.jhu.edu/~hager

Outline for the Week

Computing optical flow

From optical flow to tracking

Motion estimation using factorization

Projects and Exam

Computing Motion: Optical Flow

- Optical flow is the *apparent motion* in the image which, in some cases, corresponds to the motion field at that point.

- To compute it, think of image brightness as a function of time:
  - $E(u(t),v(t),t)$

- Let us assume that the image brightness of a point is constant. Then we have

  - $dE/dt = dE(x(t),y(t),t)/dt = E_x\, dx/dt + E_y\, dy/dt + E_t = 0$

  - equivalently, $\nabla E \cdot v + E_t = 0$, where v is the velocity vector of an image pt.

- Note that locally, this implies that we can only compute motion perpendicular to the image gradient. This is commonly referred to as the aperture problem.

When is the ICC Valid?

- ..... the same development as in the book ...

Computing Motion: Optical Flow

- Since we can't solve the IC equation at a point, two general approaches to modifying the problem:
  - regularization
  - finite patch

- Regularization assumes that points move locally consistently
  - $O(v(i)) = (\nabla E(i) \cdot v(i) + E_t(i))^2 + \sum_j \| v(j) - v(i) \|^2$
    - $O'(v(i)) = 2 \nabla E(i) (\nabla E(i) \cdot v(i) + E_t(i)) - 2 \sum_j (v(j) - v(i))$
    - $= (2n I_{2x2} + \nabla E(i) \nabla E(i)^t) v(i) - 2 \sum_j v(j) + 2 \nabla E(i) E_t(i)$
  - solution is given by a linear system over the entire image!
  - local iterations approach solution

Computing Optical Flow

- Finite patch model:
  - $O(v(i)) = \sum_j (\nabla E(j) \cdot v(i) + E_t(j))^2$
  - $O(v(i)) = || A\, v(i) + b ||^2$

    where $A = [\nabla E(1); \nabla E(2); \ldots \nabla E(n)]$ $\quad b = [E_t(1); E_t(2); \ldots E_t(n)]$
    or $A = [vec(E_x), vec(E_y)]$ and $b = vec(E_t)$
  - $O'(v(i)) = A' (A\, v(i) + b) = 0 \rightarrow v(i) = - (A'A)^{-1} A' b$
- Algorithm:
  - Filter in u,v, and t using appropriate smoothing derivative
  - Solve previous equations for each point in image

From Optical Flow to Tracking

- Now, let's introduce a warping operator
  - Simple translation: $w(R;d,t)(u,v) = I(u+d_u,v+d_v,t)$
  - Note that d is not guaranteed to be integral!
    - Typically sample using bilinear interpolation (look at the interp2 function in matlab).
    - Typically we end up warping **backwards** rather than forwards
    - Define a warped region by $R(d,t)(u,v) = I(u+d_x,v+d_y,t)$ $u \in [-w,w]$; $v \in [-h,h]$
  - Now, we can do a simple iteration to define $Oflow(d_0,\delta d_0,t_1,t_2)$
    1. Set i = 0
    2. While ($|| vec(R(d_0+\delta d_i,t_2)) - vec(R(d_0,t_1)) || > \varepsilon$)
       1. Compute E_x, E_y, and E_t from $R(d_i+\delta d_i,t_2)$ and $R(d_0,t_1)$
       2. set $\delta d_{i+1} = \delta d_i - [vec(E_x),vec(E_y)] \setminus vec(E_t)$
       3. set i = i+1
    3. Return δd_i

More Tracking

- Now, we can compute multi-frame flow (tracking) in two ways:

- Incremental integration
  - $d_{t+1} = d_t + \text{Oflow}(d_t,0,t,t+1)$
    - this computes interframe motion for a given patch
    - note that any error in Oflow is propagated to the next iteration
    - we could also predict the offset to the next frame for better convergence

- Reference tracking
  - $d_{t+1} = \text{Oflow}(d_0,d_i-d_0, t_0,t+1)$
    - this always references images back to the same reference image
    - not subject to slippage, but calls the image constancy assumption more into question

- Note that in both cases, it makes sense to first subtract the mean from both images and possibly divide by variance
  - interesting exercise: can you include these photometric components in the calculation?
- The book talks about adding Kalman predictors; we'll skip this....

Remarks

- When is A'A full rank?
  - when spatial gradient spans R^2
  - SVD of A gives a measure of confidence
  - this is a good heuristic for finding "good features to track"

- How fast can we track?
  - derivatives are good for ½ pixel
  - smoothing correlates pixels and increases attraction
  - subsampling decreases # of pixels and increases effective pixel size → usually Oflow is done in a heirarchical computation.

On to the Factorization Method ...

- Assumptions:
  - orthographic camera
  - n non-coplanar points tracked in N >= 3 frames

- Form the registered measurement matrix W* = [X*;Y*]
  - where $x^*_{i,j} = x_{i,j} - mx_i$
  - where $y^*_{i,j} = y_{i,j} - my_i$
  - mx and my are the means of the points in frame i
  - j ranges over points in a frame

- The rank theorem: The registered measurement matrix has rank at most 3

- By showing the rank theorem, we will also develop an algorithm for computing structure and motion.

The Algorithm

0. Compute the registered measurement matrix W* for N frames and n points
1. Compute the SVD of W*; $W^* = U D V^t$
   - U is 2N x 2N
   - V is n x n
   - D is 2N x n
2. Zero all but the first 3 singular values in D
   - D' is upper 3x3 of D
   - U' is corresponding 2Nx3 matrix from U
   - V' is corresponding nx3 from V
3. Define
   - $R = U' D'^{1/2}$; $S = D'^{1/2} V'^t$
4. Solve for Q s.t. Q orthogonalizes R
5. $R^* = R Q$ and $S^* = Q^{-1} S$

The Results

The Exam

- Chapter 2 (excluding 2.5)
- Chapter 3
- Chapter 4 (excluding 4.4)
- Chapter 5.1, 5.2
- Chapter 6
- Chapter 7 (excluding 7.4.3)
- Chapter 8 (excluding 8.5.2)
- Corke article
- extra material on reflection from F&P
- basic Matlab
- extra material on color

Projects

- Ground rules
  - ideally two people/project
  - due at the end of reading period
  - submit a writeup showing results of experiments and code.
  - sanity check one week before the end of the semester
    - show results on simulation and/or simulated data
    - identify data that will be used for real experiments

- Possible projects
  - stereo
  - motion
  - grouping
  - tracking
  - other topics by approval

- Grading
  - simulation, testing at checkpoint (30%)
  - completeness and thoroughness (40%)
  - evidence of understanding topic (20%)
  - style (10%)