Computer Vision
Motion

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

Outline

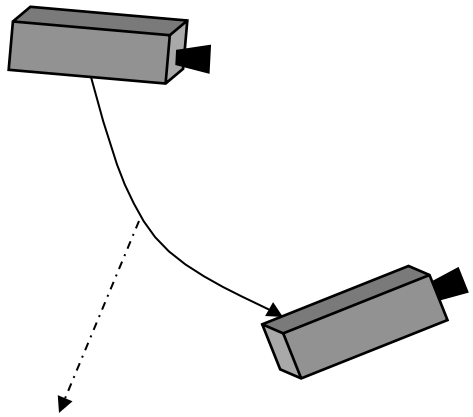
From Stereo to Motion

The motion field and optical flow (2D motion)

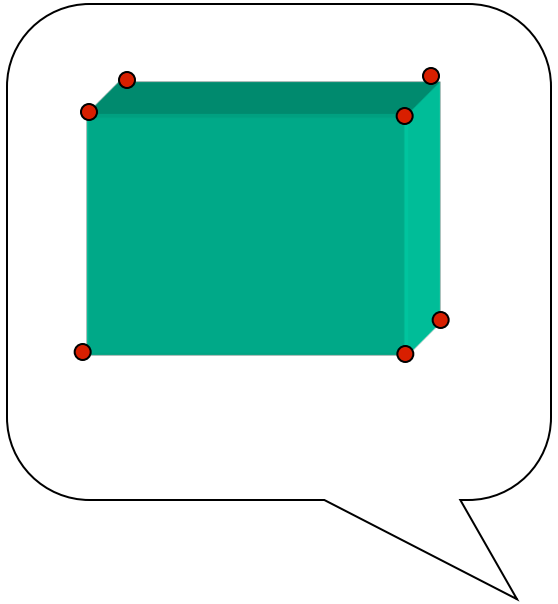
Factorization methods for structure and motion (3D motion)

DYNAMIC VISION: THE PROBLEM

MOVING CAMERAS ARE LIKE STEREO



The change in spatial location
between the two cameras (the “motion”)



Locations of
points on the object
(the “structure”)

SOME CORE PROBLEMS

- Image (2D) motion estimation
  - motion detection
  - The motion field
  - optical flow calculation
- 3D motion estimation
  - ego (self) motion
  - target motion
  - structure from motion differential
  - matching methods
- Structured motion recovery
  - visual tracking

For simplicity
we will assume a unit
focal length, metric camera
(e.g. a calibrated camera)

What is Visual Motion?

1. A body moving in space experiences
   - translation (Tr)
   - rotation (R)

2. We are interested in time change
   - trans. velocity (T = dTr/dt)
   - rot. velocity (w = "dR/dt")

3. Other rigid bodies appear to move in the opposite direction

$$^{c}p = {}^{w}p - Tr$$

What is Visual Motion?

1. A body moving in space experiences
   - translation (Tr)
   - rotation (R)

2. We are interested in time change
   - trans. velocity (T = dTr/dt)
   - rot. velocity (w = “dR/dt”)

3. Other rigid bodies appear to move in the opposite direction

4. Points on those bodies project into camera images

$$u = x/z$$
$$v = y/z$$

$$^{c}p = {}^{w}p - Tr$$

What is Visual Motion?

1. A body moving in space experiences
   - translation (Tr)
   - rotation (R)

2. We are interested in time change
   - trans. velocity (T = dTr/dt)
   - rot. velocity (w = "dR/dt")

$^{c}x = {}^{w}x - Tr_x \rightarrow dx/dt = -T_x$

$^{c}y = {}^{w}y - Tr_y \rightarrow dy/dt = -T_y$

$^{c}z = {}^{w}z - Tr_z \rightarrow dz/dt = -T_z$

3. Other rigid bodies appear to move in the opposite direction

4. Points on those bodies project into camera images

$du/dt = (dx/dt\ {}^{c}z - dz/dt\ {}^{c}x)/{}^{c}z^2$

$u = x/z$

$v = y/z$

What is Visual Motion?

1. A body moving in space experiences
   - translation (Tr)
   - rotation (R)

2. We are interested in time change
   - trans. velocity (T = dTr/dt)
   - rot. velocity (w = "dR/dt")

$^{c}x = {}^{w}x - Tr_x \rightarrow dx/dt = -T_x$

$^{c}y = {}^{w}y - Tr_y \rightarrow dy/dt = -T_y$

$^{c}z = {}^{w}z - Tr_z \rightarrow dz/dt = -T_z$

3. Other rigid bodies appear to move in the opposite direction

4. Points on those bodies project into camera images

$du/dt = (dx/dt - dz/dt\ {}^{c}x/{}^{c}z)/{}^{c}z^2$

$du/dt = -(T_x - u\ T_z)/z$

$u = x/z$

$v = y/z$

What is Visual Motion?

1. A body moving in space experiences
   - translation (Tr)
   - rotation (R)

2. We are interested in time change
   - trans. velocity (T = dTr/dt)
   - rot. velocity (w = "dR/dt")

3. Other rigid bodies appear to move in the opposite direction

4. Points on those bodies project into camera images

$$u = x/z$$
$$v = y/z$$

$$^{c}x = {}^{w}x - Tr_x \rightarrow dx/dt = -T_x$$
$$^{c}y = {}^{w}y - Tr_y \rightarrow dy/dt = -T_y$$
$$^{c}z = {}^{w}z - Tr_z \rightarrow dz/dt = -T_z$$

$$du/dt = (dx/dt - dz/dt\ {}^{c}x/{}^{c}z)/{}^{c}z^2$$
$$du/dt = -(T_x - u\,T_z)/z$$
$$dv/dt = -(T_y - v\,T_z)/z$$

What is Visual Motion?

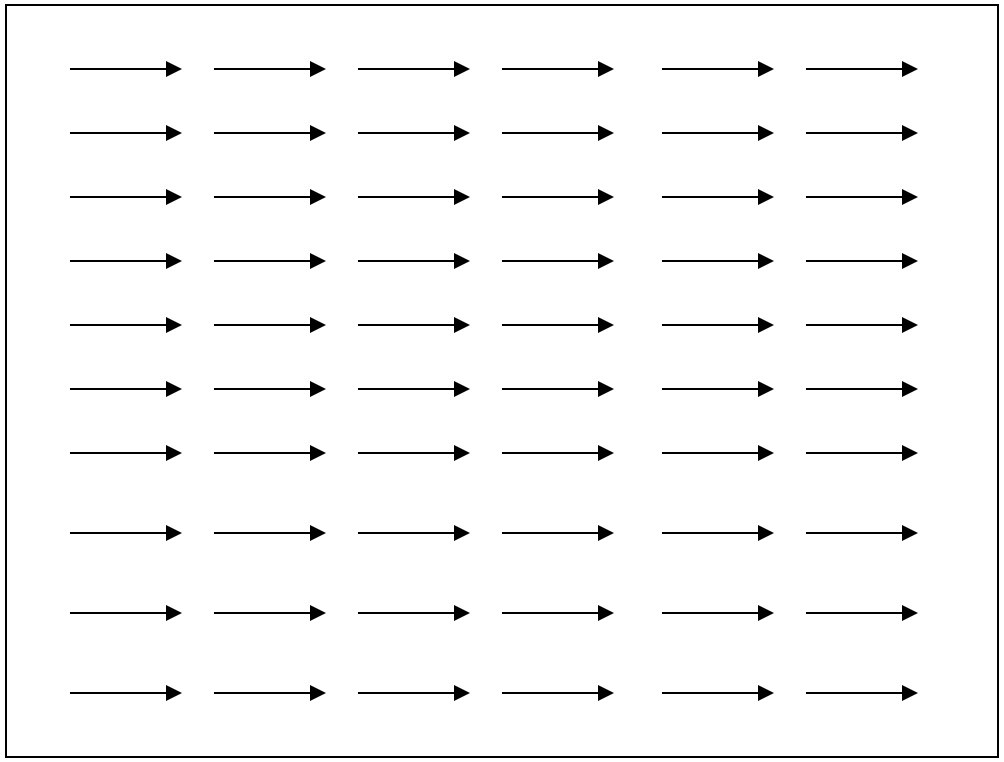
$$du/dt = -(T_x - u\,T_z)/z$$
$$dv/dt = -(T_y - v\,T_z)/z$$

This expression defines the motion field for the image under pure translation

Note that, given a velocity vector, we can determine precisely what the motion field is *up to a scale factor related to the depths of the points that are observed*

THE MOTION FIELD

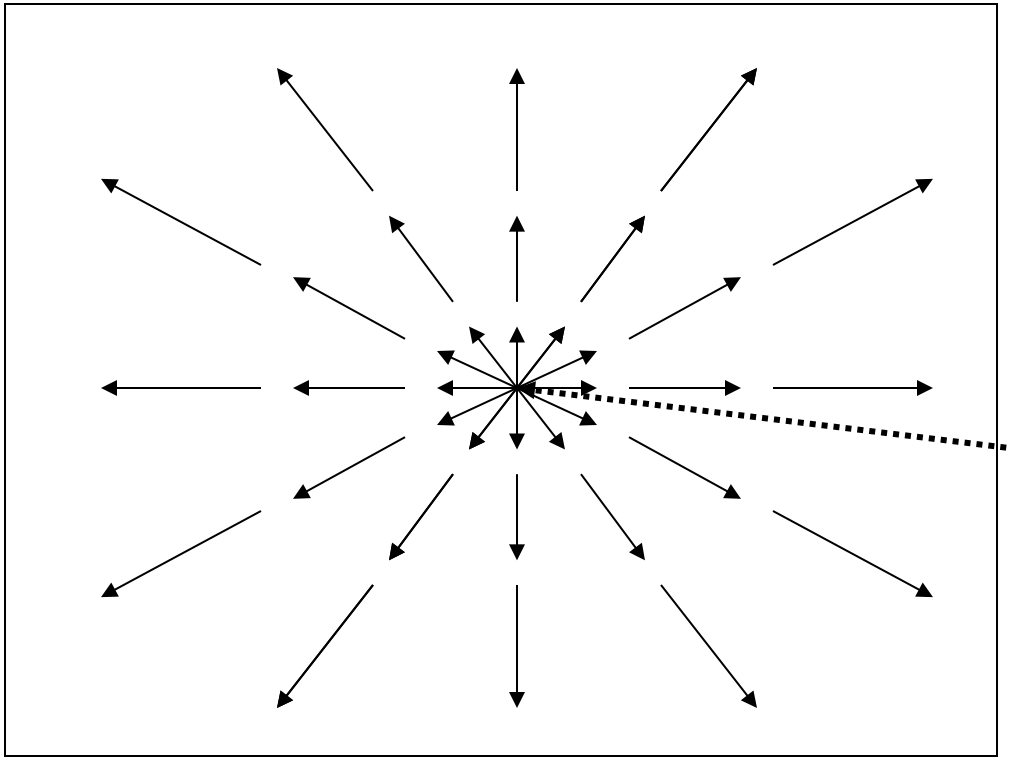
The “instantaneous” velocity of points in an image



TRANSLATION
(OR ROTATION)
TO THE LEFT

THE MOTION FIELD

The “instantaneous” velocity of points in an image



LOOMING

The *focus of expansion*

With just this information it is possible to calculate:

1. Direction of motion
2. Time to collision

Calculations

$du/dt = -(T_x - u\,T_z)/z$
$dv/dt = -(T_y - v\,T_z)/z$

- Solve for du/dt = 0 and dv/dt = 0 to find the focus of expansion

  Define $u_0 = T_x/T_z$ $(T_z \neg= 0)$; $v_0 = T_y/T_z$
  - note $(u_0, v_0, 1)$ is direction of translation

- Pulling out a T_z yields
  - $u' = -(u - u_0)\,T_z/Z$; $v' = -(v - v_0)\,T_z/Z$ -> $p' = (p - p_0)\,T_z/Z$
  - therefore, the field is radial and rooted at $(T_x/T_z, T_y/T_z) = p_0$

- Z/T_z is the number of seconds until the point crosses the image plane ("time to collision") *for this point*
  - $Z/T_z = (u-u_0)/u'$; $Z/T_z = (v - v_0)/v'$

Rotational Motion

- First, consider a 2D rotation R(q) =[cos(q),-sin(q);sin(q),cos(q)];
  - $^{c}p = R(q)\ ^{w}p$
  - $^{c}\ dp/dt = d\ R(q)\ dt\ ^{w}p + R(q)\ d\ ^{w}p/dt$

- If q is a time-varying function, then
  dR/dt = dR/d q d q/dt=
  - w [-sin(q),-cos(q);cos(q),-sin(q)] = [0,-w;w,0] R(q) = sk(w) R(q)
  - Note that the differential rotation creates the instantaneous motion vector of the point
  - We only care about case q = 0

- In 3D it's essentially the same story
  - dp/dt = S p
  - S = [0, $-w_z$, w_y; = sk(w) -> dp/dt = w x p = sk(w) p
    w_z, 0, $-w_x$;
    $-w_y$, w_x, 0]

General Visual Motion

Let us assume there is one rigid object and the camera moves with velocities T and w

For a given point P on the object, we have

$$p = P/z$$

Therefore, assuming P moves with velocity v we have

$$dp/dt = (z\, v - v_z\, P)/z^2 = 1/z\, (v - v_z\, p)$$

But v is

$$v = -T - w \times P$$

The Result:

$$\dot{u} = \frac{T_z u - T_x}{z} - w_y + w_z v + w_x uv - w_y u^2$$

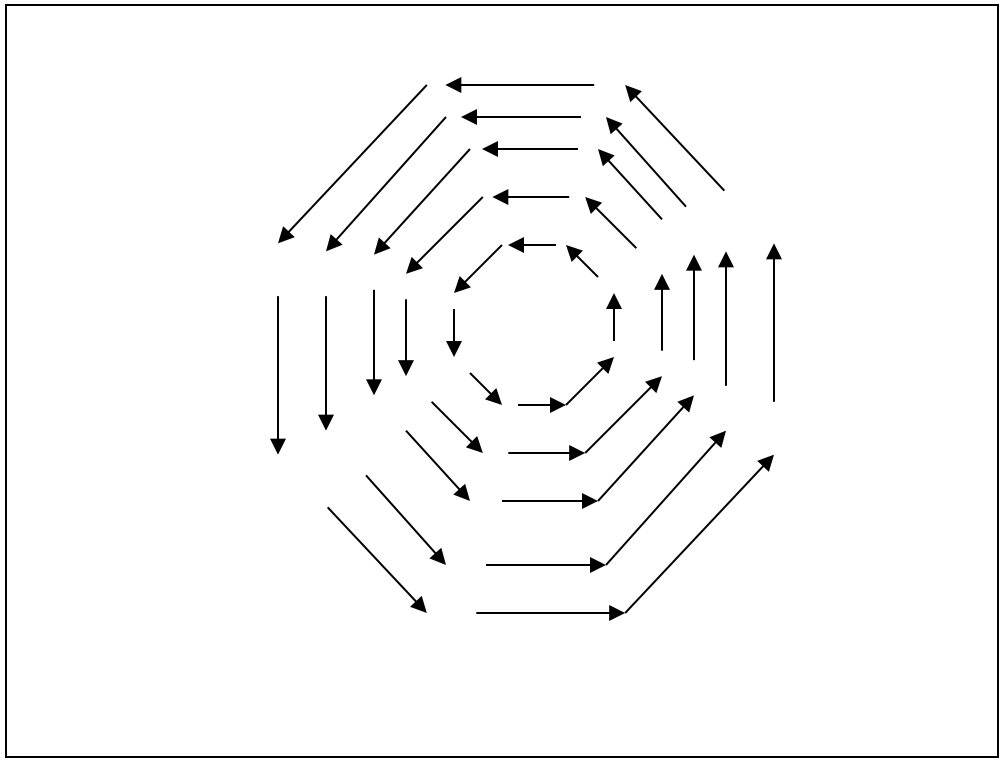
$$\dot{v} = \frac{T_z v - T_y}{z} + w_x - w_z u - w_y uv + w_x v^2$$

Motion due to translation: depends on depth

Motion due to rotation: independent of depth

THE MOTION FIELD

The “instantaneous” velocity of points in an image



PURE ROTATION

The Motion of an Object: Orthography

- Recall the orthographic case:
  - image transformations of planar objects are affine
  - therefore, the motion field itself is linear for planar objects
  - motion of non-planar objects can be viewed as *parallax* on top of this planar motion (note parallax only occurs to due to rotation in this model)
    - $p = O\ s\ (R'\ P - Tr)$
    - At origin, $p = O\ s\ P$
    - $dp/dt = O\ s\ sk(w)'\ P + s\ (T_x, T_y)' = sk_2(-w)\ p + b\ z + d$
    - $sk(w) = [0\ -w_z\ w_y;\ w_z\ 0\ -w_x;\ -w_y\ w_x\ 0]$
    - $O\ sk(w) = [0\ -w_z\ w_y;\ w_z\ 0\ -w_x]$

The Motion of an Object: Perspective

- $n^t P = d$ --- constraint on coordinates
  - $n_x X + n_y Y + n_z Z = d$
  - $n_x u + n_y v + n_z = d/Z$
  - $1/Z = (n_x u + n_y v + n_z)/d$

- Substitute back into optical flow equations

  $u' = a_1 u^2 + a_2 uv + a_3 u + a_4 v + a_5$

  $v' = a_1 uv + a_2 v^2 + a_7 u + a_6 v + a_8$

  where the a's depend on the motion and the plane parameters (see book)

It is possible to show by a counting argument that the motion field of a plane is not unique --- there is more than one situation that gives rise to the same apparent motion

Motion Parallax: Perspective

- Consider two points on a rigid object that project to the same point instantaneously

- Note that in this case, the rotation fields are identical, hence taking a difference, we arrive at
  - $du' = (T_z\, u - T_x)(1/Z - 1/Z')$
  - $dv' = (T_z\, v - T_y)(1/Z - 1/Z')$

- These can be thought of as the relative motion field

- For example, we could think of motion as a background planar motion combined with a superimposed parallax ....

  $v = v_p + dv$

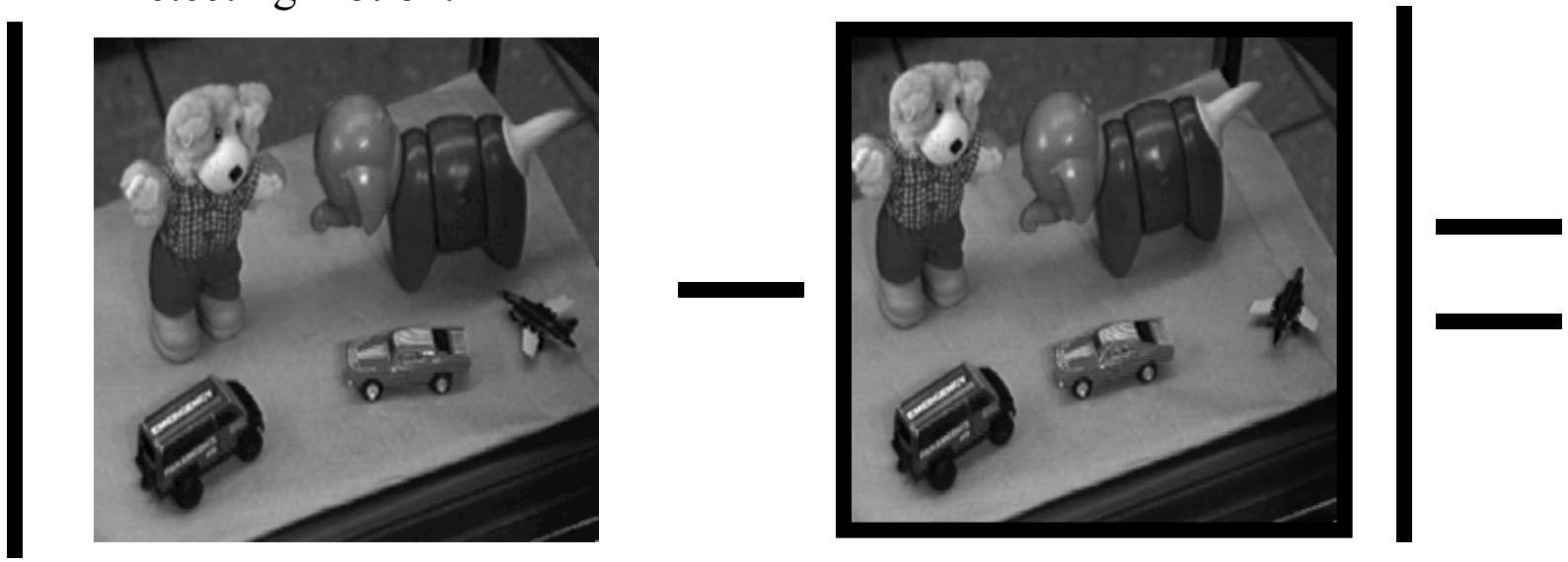
  -- compute a planar motion

  -- compute the effective planar motion field

  -- compute dv and estimate depth from plane

MOTION IN REAL IMAGES

Detecting motion:



MOTION IN REAL IMAGES

Detecting motion:

$$\left| \text{  } - \text{  } \right| > 50$$

Candidate areas for motion

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, $\nabla E \;\; v + E_t = 0$, 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.
  - normal flow: $v_n = \nabla E \,.v \,/\, ||\, \nabla E|| = - E_t/||\nabla E\, ||$

Computing Derivaties: An Aside

- The simplest way to compute the derivatives needed for optical flow is to think of a 2x2x2 cube (space x time) then we have

$$I_x(u,v,t) = [(I(u+1,v,t) - I(u,v,t)) + (I(u+1,v+1,t) - I(u,v+1,t))+ (I(u+1,v,t+1) - I(u,v,t+1)) + (I(u+1,v+1,t+1) - I(u,v+1,t+1))]/4$$

$$I_y(u,v,t) = [(I(u,v+1,t) - I(u,v,t)) + (I(u+1,v+1,t) - I(u+1,v,t))+ (I(u,v+1,t+1) - I(u,v,t+1)) + (I(u+1,v+1,t+1) - I(u+1,v,t+1))]/4$$

$$I_t(u,v,t) = [(I(u,v,t+1) - I(u,v,t)) + (I(u+1,v,t+1) - I(u+1,v,t))+ (I(u,v+1,t+1) - I(u,v+1,t)) + (I(u+1,v+1,t+1) - I(u+1,v+1,t))]/4$$

We can generalize to doing derivatives of Gaussians in space x time in the obvious way

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

When is the ICC Valid?

- $E = r\, s^t\, n$
  - image brightness is albedo times direction dotted with surface normal
- $dn/dt = w \times n$
  - change of surface normal with motion
- $\nabla E^t\, v + E_t = r\, s^t (w \times n)$
    - normal flow: $(- E_t + r\, s^t\, (w \times n))/|| \nabla E||$
  - plug this into the brightness constancy constraint
- $|\Delta v_n| = r\, |s^t\, (w \times n)|/|| \nabla E\, ||$
  - by solving for the difference between ideal solution and the "real" solution
- Result: highest accuracy when image gradient is large
- Result: optical flow is almost never equal to the motion field

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

From Optical Flow to Tracking

- Now, let's introduce a warping operator
  - Simple translation: $R(t; d)(u,v) = I(u+d_u,v+d_v,t)$
  - 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 $R(t; d) = I(u+d_x,v+d_y,t)$ u in [-w,w]; v $\rangle\backslash$ [-h,h]
  - Now, we can do a simple iteration to define $Oflow(d,\Delta d,t_1,t_2)$
    1. While ($|| vec(R(d+\Delta d,t_2)) - vec(R(d,t_1)) || > e$)
       1. Compute E_x, E_y, and E_t from $R(d+\Delta d,t_2)$ and $R(d,t_1)$
       2. set $\Delta d = \Delta d - [vec(E_x),vec(E_y)] \backslash vec(E_t)$
    2. Return Δd

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
  - $d_{t+1} = d_0 + \text{Oflow}(d_0, d_t - d_0, t_0, t+1)$
    - 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 1/2 pixel
  - smoothing correlates pixels and increases attraction
  - subsampling decreases # of pixels and increases effective pixel size
    → usually Oflow is done in a heirarchical computation.

A more elaborate example

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 Rank Theorem

- First, formulate orthographic projection for points
  - $p_{ij} = O\ R_i\ (P_j - T_i)$ [[[we'll omit scaling which doesn't matter]]
  - Choose origin at centroid of points
  - Think of first two rows of R as x and y axis of camera frame

- Note by subtracting $pm_j = \sum_j p_{i,j}/n$ from $p_{i,j}$ we have
  - $p'_{i,j} = O\ R_i(P_j - Pm)$, but Pm is at the origin so we have
  - $x'_{i,j} = \mathbf{i}_i\ P_j$
  - $y'_{i,j} = \mathbf{j}_i\ P_j$

- In this case, we can now create a 2N x 3 rotation matrix R from the i's and j's and a 3 x n shape matrix S from the points, and
  - W = R S.
  - It follows that W has rank 3 provide N >= 3 and the N points are not coplanar

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
   - $A = U' D'^{1/2}$; $S = D'^{1/2} V'^t$
4. Solve for Q s.t. Q orthogonalizes A
5. $R^* = A Q$ and $S^* = Q^{-1} S$

Orthogonalization

- A is of course not orthogonal;however for each position we can pull two rows (the i and j). Let $E_i = [i_i;j_i]$; and let A_i be the two corresponding rows of A; then
  - $E_i = A_i Q$
- This implies that
  - $E_i E_i' = A_i Q Q' A_i' = A_i M A_i' = I_{2x2}$
- Note that M appears linearly, so we can also write
  - $F(A_i)\ vec(M) = vec(I)$
  - We can stack this into a linear system of the form F vec(M) = b and solve for M (note M must be symmetric so we really only have 6 free values)
  - By eigenvalue decomposition $M = R D R' \dashrightarrow Q = R D^{1/2}$

The Results

General Perspective and Motion

- There are iterative methods for differential motion (see book); we will not cover these.
  - In general, any motion and structure method is extremely sensitive for small motion (i.e. in the optical flow case).

- There are extensions of factorization to the perspective case; the method (see Ponce and Forsyth)

- For large motions, E-matrix computation and stereo-like methods are reasonable solutions to get dense estimates of depth

- Motion segmentation (multiple motions) is an important problem. GPCA-like methods have recently been developed (Vidal, Ma) as a way of describing the generalized epipolar constraints that arise in this case.

Another Example

(Courtesy Marc Pollefeys)

Summary

- Motion field and its structure
  - translational motion (depends on depth)
  - rotational motion (quadratic on u/v)
- Optical flow vs. motion field
- Recovering optical flow and tracking
- Structure and motion from multiple views