

# Computer Vision, Lectures 15,16, 17

## Motion

Professor Hager

<http://www.cs.jhu.edu/~hager>

# Outline for Today

From Stereo to Motion

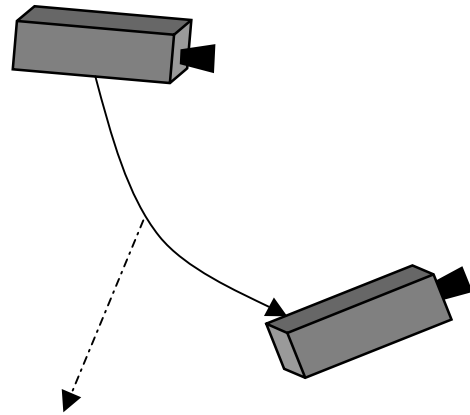
The motion field and optical flow

Your next assignment

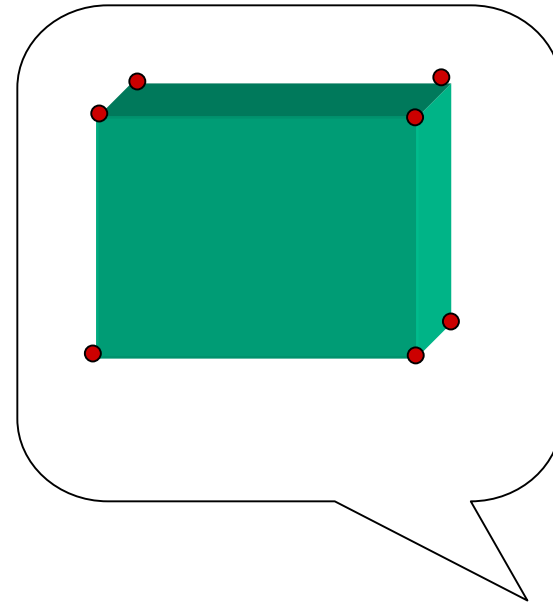
# DYNAMIC VISION: THE PROBLEM



# MOVING CAMERAS ARE LIKE STEREO



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



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

# Result of Tomasi's SFM algorithm

Simplify camera model (linear)

Use large amounts of data (multiple views, many points)

Solve correspondence using tracking methods



# SOME CORE PROBLEMS

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

W/O loss of generality,  
we will assume a unit  
focal length, metric 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 ( $\omega = "dR/dt"$ )
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 ( $T_r$ )
  - rotation ( $R$ )
2. We are interested in time change
  - trans. velocity ( $T = dT_r/dt$ )
  - rot. velocity ( $\omega = "dR/dt"$ )
3. Other rigid bodies appear to move in the opposite direction
4. Points on those bodies project into camera images

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

$$u = x/z$$

$$v = y/z$$

# What is Visual Motion?



1. A body moving in space experiences
  - translation ( $T_r$ )
  - rotation ( $R$ )
2. We are interested in time change
  - trans. velocity ( $T = dT_r/dt$ )
  - rot. velocity ( $\omega = "dR/dt"$ )

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

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

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

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 ( $T_r$ )
  - rotation ( $R$ )
2. We are interested in time change
  - trans. velocity ( $T = dT_r/dt$ )
  - rot. velocity ( $\omega = "dR/dt"$ )

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

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

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

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

4. Points on those bodies project into camera images

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

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

$$u = x/z$$

$$v = y/z$$

# What is Visual Motion?



1. A body moving in space experiences
  - translation ( $T_r$ )
  - rotation ( $R$ )
2. We are interested in time change
  - trans. velocity ( $T = dT_r/dt$ )
  - rot. velocity ( $\omega = "dR/dt"$ )
3. Other rigid bodies appear to move in the opposite direction
4. Points on those bodies project into camera images

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

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

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

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

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

$$dv/dt = -(T_y + v T_z) / z$$

$$u = x/z$$

$$v = y/z$$

# What is Visual Motion?



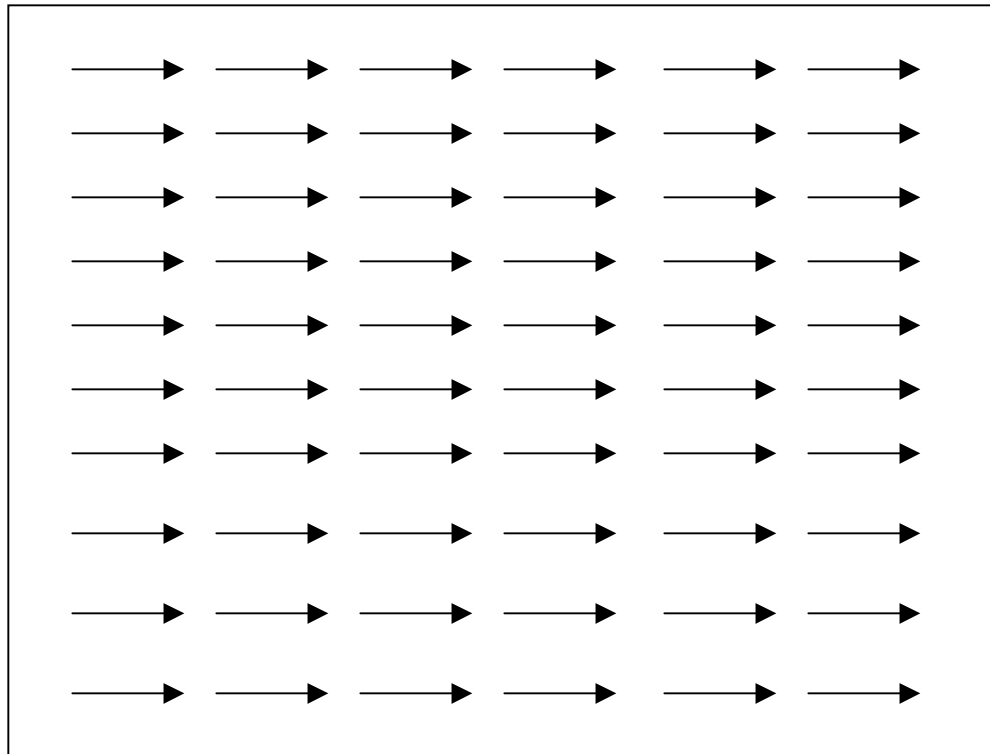
$$\begin{aligned} du/dt &= -(T_x + u T_z)/z \\ dv/dt &= -(T_y + v T_z)/z \end{aligned}$$

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

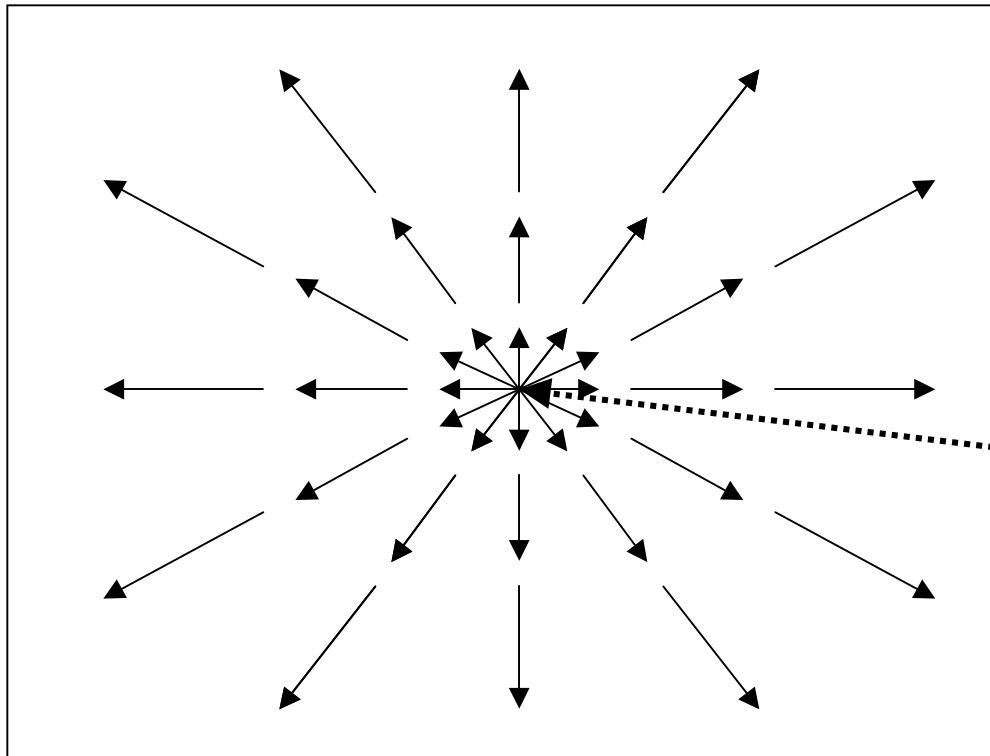
The “instantaneous” velocity of points in an image



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

$$\begin{aligned} du/dt &= -(T_x + u T_z)/z \\ dv/dt &= -(T_y + v T_z)/z \end{aligned}$$

- Define  $u_0 = T_x/T_z$  ( $T_z \neq 0$ );  $v_0 = T_y/T_z$ 
  - note  $(u_0, v_0, 1)$  is direction of translation
- $u' = (u - u_0) T_z/Z$ ;  $v' = (v - v_0) T_z/Z \rightarrow p' = (p - p_0) T_z/Z$ 
  - 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  $R(\theta) = [\cos(\theta), -\sin(\theta); \sin(\theta), \cos(\theta)]$ ;
  - ${}^c p = R(\theta) {}^w p$
  - ${}^c dp/dt = d R(\theta) / dt {}^w p + R(\theta) d {}^w p/dt$
- If  $\theta$  is a time-varying function, then  $dR/dt = dR/d\theta d\theta/dt =$ 

$$\omega [-\sin(\theta), -\cos(\theta); \cos(\theta), -\sin(\theta)] = [0, -\omega; \omega, 0] R(\theta) = sk(\omega) R(\theta)$$
  - Note that the differential rotation creates the instantaneous motion vector of the point
  - We only care about case  $\theta = 0$
- In 3D it's essentially the same story
  - $dp/dt = S p$
  - $S = \begin{bmatrix} 0, & -\omega_z, & \omega_y; \\ \omega_z, & 0, & -\omega_x; \\ -\omega_y, & \omega_x, & 0 \end{bmatrix} = sk(\omega) \rightarrow dp/dt = \omega \times p$

# General Visual Motion

Let us assume there is one rigid object and the camera moves with velocities  $T$  and  $\omega$

For a given point  $P$  on the object, we have

$$p = P/z$$

Therefore, we have

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

The apparent velocity of the point is

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

# To Put it Another Way

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

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



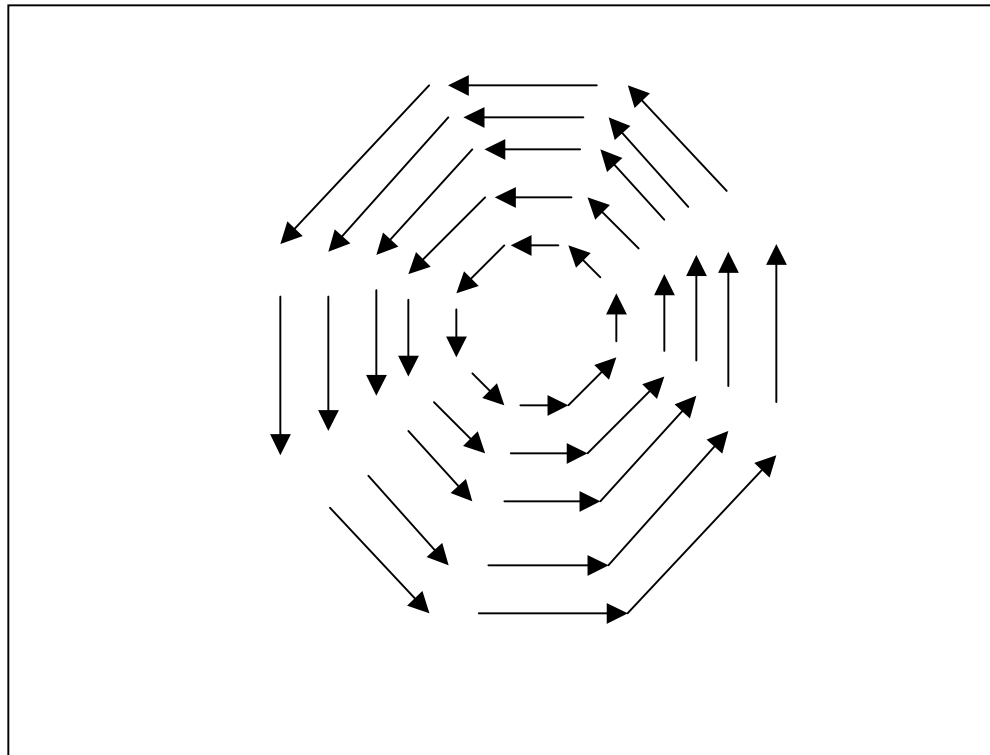
Motion due to translation:  
depends on depth



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 due to rotation in this model)
    - $p = O s (R' P - Tr)$
    - At origin,  $p = O s P$
    - $dp/dt = O s sk(\omega)' P + s (T_x, T_y)' = sk_2(-\omega) p + b z + d$
    - $sk(\omega) = [0 \ -\omega_z \ \omega_y; \ \omega_z \ 0 \ -\omega_x; \ -\omega_y \ \omega_x \ 0]$
    - $O sk(\omega) = [0 \ -\omega_z \ \omega_y; \ \omega_z \ 0 \ -\omega_x]$

# 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

It is possible to show 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

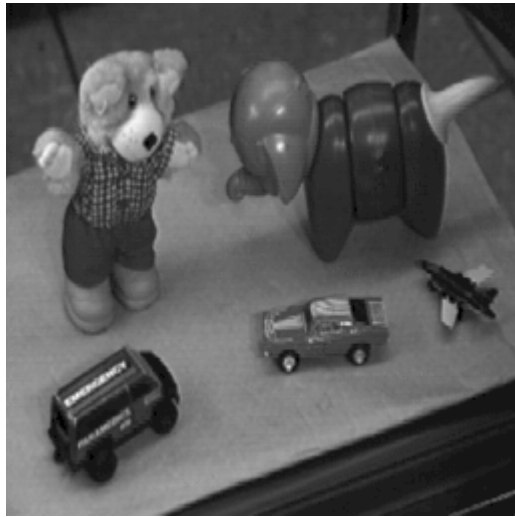
- Consider two points on a rigid object that project to the same point instantaneously
- Note that in this case, the rotation fields are identical, hence taking a difference, we arrive at
  - $\Delta u' = (T_z u - T_x)(1/Z - 1/Z')$
  - $\Delta v' = (T_z v - T_y)(1/Z - 1/Z')$
- These can be thought of as the relative motion field
- For example, we could think of motion as a background planar motion combined with a superimposed parallax ....

$$v = v_p + \Delta v$$

- compute a planar motion
- compute the effective planar motion field
- compute  $\Delta v$  and estimate depth from plane

# MOTION IN REAL IMAGES

Detecting motion:



—

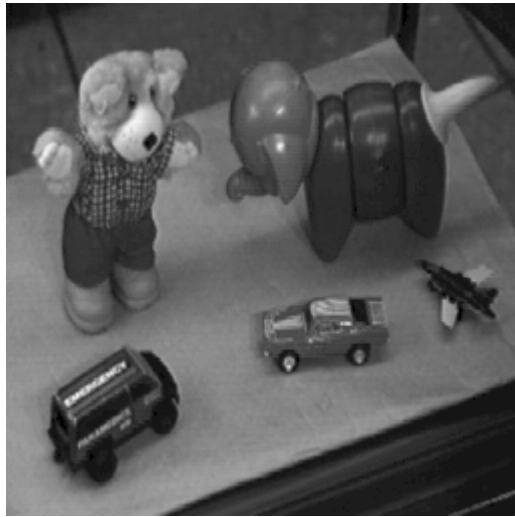


=



# MOTION IN REAL IMAGES

Detecting motion:



—



$> 50$

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

# Computing Derivatives: 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 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
  - $O(v(i)) = (\nabla E(i) \cdot v(i) + E_t(i))^2 + \sum_j \|v(j) - v(i)\|^2$ 
    - $O'(v(i)) = 2 \nabla E(i) (\nabla E(i) \cdot v(i) + E_t(i)) - 2 \sum_j (v(j) - v(i))$
    - $= (2n I_{2 \times 2} + \nabla E(i) \nabla E(i)^t) v(i) - 2 \sum_j v(j) + 2 \nabla E(i) E_t(i)$
  - 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  $A = [\nabla E(1); \nabla E(2); \dots \nabla E(n)]$      $b = [E_t(1); E_t(2); \dots E_t(n)]$

or  $A = [\text{vec}(E_x), \text{vec}(E_y)]$  and  $b = \text{vec}(E_t)$

- $O'(v(i)) = A' (A v(i) + b) = 0 \rightarrow v(i) = - (A'A)^{-1} A' b$

- Algorithm:

- Filter in  $u, v$ , and  $t$  using appropriate smoothing derivative

- Solve previous equations for each point in image

# When is the ICC Valid?

- $E = \rho s^t n$ 
  - image brightness is albedo times direction dotted with surface normal
- $dn/dt = \omega \times n$ 
  - change of surface normal with motion
- $\nabla E^t v + E_t = \rho s^t (\omega \times n)$ 
  - normal flow:  $(-E_t + \rho s^t (\omega \times n)) / \|\nabla E\|$
  - plugging this into the brightness constancy constraint
- $|\Delta v_n| = \rho |s^t \omega \times n| / \|\nabla E\|$ 
  - and 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

# 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  $R(t; d) = I(u+d_x, v+d_y, t)$   $u \in [-w, w]$ ;  $v \in [-h, h]$
  - Now, we can do a simple iteration to define  $O_{\text{flow}}(d_0, \delta d_0, t_1, t_2)$ 
    1. Set  $i = 0$
    2. While  $(\| \text{vec}(R(d_0 + \delta d_i, t_2)) - \text{vec}(R(d_0, t_1)) \| > \epsilon)$ 
      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  $\delta d_{i+1} = \delta d_i - [\text{vec}(E_x), \text{vec}(E_y)] \setminus \text{vec}(E_t)$
      3. set  $i = i+1$
    3. Return  $\delta d_i$

# 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  $\frac{1}{2}$  pixel
  - smoothing correlates pixels and increases attraction
  - subsampling decreases # of pixels and increases effective pixel size  
→ usually Oflow is done in a heirarchical computation.

# On to the Factorization Method ...

- Assumptions:
  - orthographic camera
  - $n$  non-coplanar points tracked in  $N \geq 3$  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  $p_{m,j} = \sum_j p_{i,j}/n$  from  $p_{i,j}$  we have
  - $p'_{i,j} = O R_i (P_j - P_m)$ , but  $P_m$  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  $2N \times 3$  rotation matrix  $R$  from the  $i$ 's and  $j$ 's and a  $3 \times n$  shape matrix  $S$  from the points, and
  - $W = R S$ .
  - It follows that  $W$  has rank 3 provide  $N \geq 3$  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  $2N \times 2N$
  - $V$  is  $n \times n$
  - $D$  is  $2N \times n$
2. Zero all but the first 3 singular values in  $D$ 
  - $D'$  is upper  $3 \times 3$  of  $D$
  - $U'$  is corresponding  $2N \times 3$  matrix from  $U$
  - $V'$  is corresponding  $n \times 3$  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

