Motion Estimation (I)

Ce Liu
celiu@microsoft.com
Microsoft Research New England
We live in a moving world

- Perceiving, understanding and predicting motion is an important part of our daily lives
Motion estimation: a core problem of computer vision

• Related topics:
  – Image correspondence, image registration, image matching, image alignment, ...

• Applications
  – Video enhancement: stabilization, denoising, super resolution
  – 3D reconstruction: structure from motion (SFM)
  – Video segmentation
  – Tracking/recognition
  – Advanced video editing
Contents (today)

• Motion perception
• Motion representation
• Parametric motion: Lucas-Kanade
• Dense optical flow: Horn-Schunck
• Robust estimation
• Applications (1)
Readings

- Rick’s book: Chapter 8
- Ce Liu’s PhD thesis (Appendix A & B)
- Horn-Schunck (wikipedia)
Contents

• Motion perception

• Motion representation

• Parametric motion: Lucas-Kanade

• Dense optical flow: Horn-Schunck

• Robust estimation

• Applications (1)
Seeing motion from a static picture?

http://www.ritsumei.ac.jp/~akitaoka/index-e.html
More examples
How is this possible?

- The true mechanism is to be revealed
- FMRI data suggest that illusion is related to some component of eye movements
- We don’t expect computer vision to “see” motion from these stimuli, yet
What do you see?
In fact, ...
The cause of motion

• Three factors in imaging process
  – Light
  – Scene
  – Camera

• Varying either of them causes motion
  – Static camera, moving scene (surveillance)
  – Moving camera, static scene (3D capture)
  – Moving camera, moving scene (sports, movie)
  – Static camera, moving scene, moving light (time lapse)
Motion scenarios (priors)

- Static camera, moving scene
- Moving camera, static scene
- Moving camera, moving scene
- Static camera, moving scene, moving light
We still don’t touch these areas
Not challenging enough?
Motion analysis: human vs. computer

• Challenges of motion estimation
  – *Geometry*: shapeless objects
  – *Reflectance*: transparency, shadow, reflection
  – *Lighting*: fast moving light sources
  – *Sensor*: motion blur, noise

• Key: motion *representation*
  – Ideally, solve the inverse rendering problem for a video sequence
    • *Intractable!*
  – Practically, we make strong assumptions
    • *Geometry*: rigid or slow deforming objects
    • *Reflectance*: opaque, Lambertian surface
    • *Lighting*: fixed or slow changing
    • *Sensor*: no motion blur, low-noise
Contents

• Motion perception

• **Motion representation**

• Parametric motion: Lucas-Kanade

• Dense optical flow: Horn-Schunck

• Robust estimation

• Applications (1)
Parametric motion

- **Mapping:** \((x_1, y_1) \rightarrow (x_2, y_2)\)
  - \((x_1, y_1)\): point in frame 1
  - \((x_2, y_2)\): corresponding point in frame 2

- **Global parametric motion:** \((x_2, y_2) = f(x_1, y_1; \theta)\)

- **Forms of parametric motion**
  - **Translation:**
    \[
    \begin{bmatrix}
    x_2 \\
    y_2
    \end{bmatrix} = \begin{bmatrix}
    x_1 + a \\
    y_1 + b
    \end{bmatrix}
    \]
  - **Similarity:**
    \[
    \begin{bmatrix}
    x_2 \\
    y_2
    \end{bmatrix} = s \begin{bmatrix}
    \cos(\alpha) & \sin(\alpha) \\
    -\sin(\alpha) & \cos(\alpha)
    \end{bmatrix} \begin{bmatrix}
    x_1 + a \\
    y_1 + b
    \end{bmatrix}
    \]
  - **Affine:**
    \[
    \begin{bmatrix}
    x_2 \\
    y_2
    \end{bmatrix} = \begin{bmatrix}
    ax_1 + by_1 + c \\
    dx_1 + ey_1 + f
    \end{bmatrix}
    \]
  - **Homography:**
    \[
    \begin{bmatrix}
    x_2 \\
    y_2
    \end{bmatrix} = \frac{1}{z} \begin{bmatrix}
    ax_1 + by_1 + c \\
    dx_1 + ey_1 + f
    \end{bmatrix}, \; z = gx_1 + hy_1 + i
    \]
Parametric motion forms

- Translation
- Similarity
- Affine
- Homography
Optical flow field

• Parametric motion is limited and cannot describe the motion of arbitrary videos
• Optical flow field: assign a flow vector \((u(x, y), v(x, y))\) to each pixel \((x, y)\)
• Projection from 3D world to 2D
Optical flow field visualization

- Too messy to plot flow vector for every pixel
- Map flow vectors to color
  - Magnitude: saturation
  - Orientation: hue

Input two frames

Ground-truth flow field

Visualization code [Baker et al. 2007]
Matching criterion

- Brightness constancy assumption
  \[ I_1(x, y) = I_2(x + u, y + v) + n \]
  \[ n \sim N(0, \sigma^2) \]
- Noise \( n \) (will revisit later)
- Matching criteria
  - What’s invariant between two images?
    - Brightness, gradients, phase, other features...
  - Distance metric (L2, robust functions)
    \[ E(u, v) = \sum_{x,y} (I_1(x, y) - I_2(x + u, y + v))^2 \]
  - Correlation, normalized cross correlation (NCC)
Contents

• Motion perception
• Motion representation
• Parametric motion: Lucas-Kanade
• Dense optical flow: Horn-Schunck
• Robust estimation
• Applications (1)
Lucas-Kanade: problem setup

- Given two images $I_1(x, y)$ and $I_2(x, y)$, estimate a parametric motion that transforms $I_1$ to $I_2$
- Let $x = (x, y)^T$ be a column vector indexing pixel coordinate
- Two typical transforms
  - Translation: $W(x; p) = \begin{bmatrix} x + p_1 \\ y + p_2 \end{bmatrix}$
  - Affine: $W(x; p) = \begin{bmatrix} p_1x + p_3y + p_5 \\ p_2x + p_4y + p_6 \end{bmatrix} = \begin{bmatrix} p_1 & p_3 & p_5 \\ p_2 & p_4 & p_6 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
- Goal of the Lucas-Kanade algorithm
  $$p^* = \arg \min_p \sum_x [I_2(W(x; p)) - I_1(x)]^2$$
An incremental algorithm

• Difficult to directly optimize the objective function
  \[ p^* = \arg \min_p \sum_x [I_2(W(x; p)) - I_1(x)]^2 \]

• Instead, we try to optimize each step
  \[ \Delta p^* = \arg \min_{\Delta p} \sum_x [I_2(W(x; p + \Delta p)) - I_1(x)]^2 \]

• The transform parameter is updated:
  \[ p \leftarrow p + \Delta p^* \]
Taylor expansion

- The term $I_2(W(x; p + \Delta p))$ is highly nonlinear
- Taylor expansion:
  \[ I_2(W(x; p + \Delta p)) \approx I_2(W(x; p)) + \nabla I_2 \frac{\partial W}{\partial p} \Delta p \]
- $\frac{\partial W}{\partial p}$: Jacobian of the warp
- If $W(x; p) = (W_x(x; p), W_y(x; p))^T$, then
  \[
  \frac{\partial W}{\partial p} = \begin{bmatrix}
  \frac{\partial W_x}{\partial p_1} & \cdots & \frac{\partial W_x}{\partial p_n} \\
  \frac{\partial W_y}{\partial p_1} & \cdots & \frac{\partial W_y}{\partial p_n}
  \end{bmatrix}
  \]
Jacobian matrix

• For affine transform: \( W(x; p) = \begin{bmatrix} p_1 & p_3 & p_5 \\ p_2 & p_4 & p_6 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \)

The Jacobian is \( \frac{\partial W}{\partial p} = \begin{bmatrix} x & 0 & y & 0 & 1 & 0 \\ 0 & x & 0 & y & 0 & 1 \end{bmatrix} \)

• For translation: \( W(x; p) = \begin{bmatrix} x + p_1 \\ y + p_2 \end{bmatrix} \)

The Jacobian is \( \frac{\partial W}{\partial p} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \)
Taylor expansion

• \( \nabla I_2 = [I_x \ I_y] \) is the gradient of image \( I_2 \) evaluated at \( W(x; p) \): compute the gradients in the coordinate of \( I_2 \) and warp back to the coordinate of \( I_1 \)

• For affine transform \( \frac{\partial W}{\partial p} = \begin{bmatrix} x & 0 & y & 0 & 1 & 0 \\ 0 & x & 0 & y & 0 & 1 \end{bmatrix} \)

\[
\nabla I_2 \frac{\partial W}{\partial p} = [I_xx \ I_yx \ I_xy \ I_yy \ I_x \ I_y]
\]

• Let matrix \( B = [I_x X \ I_y X \ I_x Y \ I_y Y \ I_x \ I_y] \in \mathbb{R}^{n \times 6}, I_x \) and \( X \) are both column vectors. \( I_x X \) is element-wise vector multiplication.
Gauss-Newton

• With Taylor expansion, the objective function becomes

\[
\Delta p^* = \arg \min_{\Delta p} \sum_x \left[ I_2(W(x; p)) + \nabla I_2 \frac{\partial W}{\partial p} \Delta p - I_1(x) \right]^2
\]

Or in a vector form:

\[
\Delta p^* = \arg \min_{\Delta p} (I_t + B\Delta p)^T(I_t + B\Delta p)
\]

Where \( B = \begin{bmatrix} I_x X & I_y X & I_x Y & I_y Y & I_x & I_y \end{bmatrix} \in \mathbb{R}^{n \times 6} \)

\( I_t = I_2(W(p)) - I_1 \)

• Solution:

\[
\Delta p^* = -(B^T B)^{-1}B^T I_t
\]

Hessian matrix
How it works
How it works
How it works
How it works

Step 1: Image

Step 2: Image Gradient X

Step 3: Image Gradient Y

Step 4: Warp Parameters

Error:

$T(x) - J(W(x; p))$
How it works

Compute matrix

\[ B = \begin{bmatrix} \nabla I_2 \frac{\partial W}{\partial p} \end{bmatrix} \]
How it works

Compute inverse Hessian: \((\mathbf{B}^T \mathbf{B})^{-1}\)

\[
\mathbf{B} = \begin{bmatrix}
\nabla I_2 \frac{\partial W}{\partial p}
\end{bmatrix}
\]
How it works

Compute: $\mathbf{B}^T \mathbf{I}_t$

$$\mathbf{B} = \begin{bmatrix} \nabla I_x \frac{\partial W}{\partial p} \
\nabla I_y \frac{\partial W}{\partial p} \end{bmatrix}$$
How it works

Solve linear system:
\[
\Delta p^* = - (B^T B)^{-1} B^T I_t
\]

\[
B = \begin{bmatrix}
\nabla I_2 \frac{\partial W}{\partial p}
\end{bmatrix}
\]
How it works

\[ p \leftarrow p + \Delta p^* \]
Translation

- Jacobian: \( \frac{\delta W}{\delta p} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \)
- \( \nabla I_2 \frac{\delta W}{\delta p} = [I_x \ I_y] \)
- \( \mathbf{B} = [I_x \ I_y] \in \mathbb{R}^{n \times 2} \)
- Solution:

\[
\Delta p^* = -(\mathbf{B}^T \mathbf{B})^{-1} \mathbf{B}^T \mathbf{I}_t \\
= -\begin{bmatrix} I_x^T I_x & I_x^T I_y \\ I_y^T I_x & I_y^T I_y \end{bmatrix}^{-1} \begin{bmatrix} I_x^T \mathbf{I}_t \\ I_y^T \mathbf{I}_t \end{bmatrix}
\]
Coarse-to-fine refinement

• Lucas-Kanade is a greedy algorithm that converges to local minimum
• Initialization is crucial: if initialized with zero, then the underlying motion must be small
• If underlying transform is significant, then coarse-to-fine is a must

Smooth & down-sampling

\[(u_2, v_2) \times 2\]

\[(u_1, v_1) \times 2\]

\[(u, v)\]
Variations

• Variations of Lucas Kanade:
  – Additive algorithm [Lucas-Kanade, 81]
  – Compositional algorithm [Shum & Szeliski, 98]
  – Inverse compositional algorithm [Baker & Matthews, 01]
  – Inverse additive algorithm [Hager & Belhumeur, 98]

• Although inverse algorithms run faster (avoiding re-computing Hessian), they have the same complexity for robust error functions!
From parametric motion to flow field

• Incremental flow update \((du, dv)\) for pixel \((x, y)\)

\[
I_2(x + u + du, y + v + dv) - I_1(x, y) = I_2(x + u, y + v) + I_x(x + u, y + v)du + I_y(x + u, y + v)dv - I_1(x, y)
\]

\[
I_x du + I_y dv + I_t = 0
\]

• We obtain the following function within a patch

\[
\begin{bmatrix} du \\ dv \end{bmatrix} = - \begin{bmatrix} I_x^T I_x & I_x^T I_y \\ I_x^T I_y & I_y^T I_y \end{bmatrix}^{-1} \begin{bmatrix} I_x^T I_t \\ I_y^T I_t \end{bmatrix}
\]

• The flow vector of each pixel is updated independently

• Median filtering can be applied for spatial smoothness
Example

Input two frames

Coarse-to-fine LK

Flow visualization

Coarse-to-fine LK with median filtering
Contents

• Motion perception
• Motion representation
• Parametric motion: Lucas-Kanade
• Dense optical flow: Horn-Schunck
• Robust estimation
• Applications (1)
Motion ambiguities

• When will the Lucas-Kanade algorithm fail?

\[
\begin{bmatrix}
    du \\
    dv
\end{bmatrix} = - \begin{bmatrix}
    I_x^T I_x & I_x^T I_y \\
    I_y^T I_x & I_y^T I_y
\end{bmatrix}^{-1} \begin{bmatrix}
    I_x^T I_t \\
    I_y^T I_t
\end{bmatrix}
\]

• The inverse may not exist!!!

• How?
  – All the derivatives are zero: flat regions
  – X- and y-derivatives are linearly correlated: lines
Aperture problem
Dense optical flow with spatial regularity

- **Local motion is inherently ambiguous**
  - *Corners*: definite, no ambiguity (but can be misleading)
  - *Lines*: definite along the normal, ambiguous along the tangent
  - *Flat regions*: totally ambiguous

- **Solution**: imposing spatial smoothness to the flow field
  - Adjacent pixels should move together as much as possible

- **Horn & Schunck equation**

\[
(u, v) = \arg \min \int \int (I_x u + I_y v + I_t)^2 + \alpha(|\nabla u|^2 + |\nabla v|^2) \, dx \, dy
\]

\[
|\nabla u|^2 = \left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 = u_x^2 + u_y^2
\]

- \(\alpha\): smoothness coefficient
2D Euler Lagrange

- 2D Euler Lagrange: the functional

\[ S = \int_{\Omega} L(x, y, f, f_x, f_y) \, dx \, dy \]

is minimized only if \( f \) satisfies the partial differential equation (PDE)

\[ \frac{\partial L}{\partial f} - \frac{\partial}{\partial x} \frac{\partial L}{\partial f_x} - \frac{\partial}{\partial y} \frac{\partial L}{\partial f_y} = 0 \]

- In Horn-Schunck

\[
\begin{align*}
- L(u, v, u_x, u_y, v_x, v_y) &= (I_x u + I_y v + I_t)^2 + \alpha (u_x^2 + u_y^2 + v_x^2 + v_y^2) \\
- \frac{\partial L}{\partial u} &= 2(I_x u + I_y v + I_t)I_x \\
- \frac{\partial L}{\partial u_x} &= 2\alpha u_x, \frac{\partial}{\partial x} \frac{\partial L}{\partial u_x} = 2\alpha u_{xx}, \frac{\partial L}{\partial u_y} = 2\alpha u_y, \frac{\partial}{\partial y} \frac{\partial L}{\partial u_y} = 2\alpha u_{yy}
\end{align*}
\]
Linear PDE

• The Euler-Lagrange PDE for Horn-Schunck is

\[
\begin{align*}
(I_x u + I_y v + I_t)I_x - \alpha(u_{xx} + u_{yy}) &= 0 \\
(I_x u + I_y v + I_t)I_y - \alpha(v_{xx} + v_{yy}) &= 0
\end{align*}
\]

• \(u_{xx} + u_{yy}\) can be obtained by a Laplacian operator:

\[
\begin{bmatrix}
0 & -1 & 0 \\
-1 & 4 & -1 \\
0 & -1 & 0
\end{bmatrix}
\]

• In the end, we solve the large linear system

\[
\begin{bmatrix}
I_x^2 + \alpha L & I_x I_y \\
I_x I_y & I_y^2 + \alpha L
\end{bmatrix}
\begin{bmatrix}
U \\
V
\end{bmatrix}
= -
\begin{bmatrix}
I_x I_t \\
I_y I_t
\end{bmatrix}
\]
How to solve a large linear system $Ax=b$?

$\begin{bmatrix}
I_x^2 + \alpha L & I_x I_y \\
I_x I_y & I_y^2 + \alpha L
\end{bmatrix} \begin{bmatrix} U \\ V \end{bmatrix} = - \begin{bmatrix} I_x I_t \\ I_y I_t \end{bmatrix}$

- With $\alpha > 0$, this system is positive definite!
- You can use your favorite iterative solver
  - Gauss-Seidel, successive over-relaxation (SOR)
  - (Pre-conditioned) conjugate gradient
- No need to wait for the solver to converge completely
Incremental Solution

• In the objective function

\[(u, v) = \arg\min \iint (I_x u + I_y v + I_t)^2 + \alpha(|\nabla u|^2 + |\nabla v|^2) dx dy\]

The displacement \((u, v)\) has to be small for the Taylor expansion to be valid

• More practically, we can estimate the optimal incremental change

\[\iint (I_x du + I_y dv + I_t)^2 + \alpha(|\nabla (u + du)|^2 + |\nabla (v + dv)|^2) dx dy\]

• The solution becomes

\[
\begin{bmatrix}
I_x^2 + \alpha L & I_x I_y \\
I_x I_y & I_y^2 + \alpha L
\end{bmatrix}
\begin{bmatrix}
dU \\
dV
\end{bmatrix} = - \begin{bmatrix}
I_x I_t + \alpha LU \\
I_y I_t + \alpha LV
\end{bmatrix}
\]
Example

Input two frames

Flow visualization

Horn-Schunck

Coarse-to-fine LK

Coarse-to-fine LK with median filtering
Continuous Markov Random Fields

• Horn-Schunck started 30 years of research on continuous Markov random fields
  – Optical flow estimation
  – Image reconstruction, e.g. denoising, super resolution
  – Shape from shading, inverse rendering problems
  – Natural image priors

• Why continuous?
  – Image signals are differentiable
  – More complicated spatial relationships

• Fast solvers
  – Multi-grid
  – Preconditioned conjugate gradient
  – FFT + annealing
Contents

• Motion perception
• Motion representation
• Parametric motion: Lucas-Kanade
• Dense optical flow: Horn-Schunck
• Robust estimation
• Applications (1)
Spatial regularity

• Horn-Schunck is a Gaussian Markov random field (GMRF)

\[
\iint (I_x u + I_y v + I_t)^2 + \alpha (|\nabla u|^2 + |\nabla v|^2) \, dx \, dy
\]

• Spatial over-smoothness is caused by the quadratic smoothness term

• Nevertheless, real optical flow fields are sparse!
Data term

- Horn-Schunck is a Gaussian Markov random field (GMRF)
  \[ \iiint (I_x u + I_y v + I_t)^2 + \alpha (|\nabla u|^2 + |\nabla v|^2) dx dy \]

- Quadratic data term implies Gaussian white noise
- Nevertheless, the difference between two corresponded pixels is caused by
  - Noise (majority)
  - Occlusion
  - Compression error
  - Lighting change
  - ...

- The error function needs to account for these factors
Noise model

• Explicitly model the noise $n$

$$I_2(x + u, y + v) = I_1(x, y) + n$$

• It can be a mixture of two Gaussians, inlier and outlier

$$n \sim \lambda N(0, \sigma_i^2) + (1 - \lambda) N(0, \sigma_o^2)$$
More components in the mixture

- Consider a Gaussian mixture model
  \[ n \sim \frac{1}{Z} \sum_{k=1}^{K} \xi^k N(0, (ks)^2) \]

- Varying the decaying rate \( \xi \), we obtain a variety of potential functions
Typical error functions

L2 norm
\[ \rho(z) = z^2 \]

L1 norm
\[ \rho(z) = |z| \]

Truncated L1 norm
\[ \rho(z) = \min(|z|, \eta) \]

Lorentzian
\[ \rho(z) = \log(1 + \gamma z^2) \]
Robust statistics

- Traditional L2 norm: only noise, no outlier
- Example: estimate the average of $0.95, 1.04, 0.91, 1.02, 1.10, 20.01$
- Estimate with minimum error
  \[ z^* = \arg \min_z \sum_i \rho(z - z_i) \]
  - L2 norm: $z^* = 4.172$
  - L1 norm: $z^* = 1.038$
  - Truncated L1: $z^* = 1.0296$
  - Lorentzian: $z^* = 1.0147$
The family of robust power functions

- Can we directly use L1 norm $\psi(z) = |z|$?
  - Derivative is not continuous
- Alternative forms
  - L1 norm: $\psi(z^2) = \sqrt{z^2 + \varepsilon^2}$
  - Sub L1: $\psi(z^2; \eta) = (z^2 + \varepsilon^2)^\eta$, $\eta < 0.5$
Modification to Horn-Schunck

- Let $x = (x, y, t)$, and $w(x) = (u(x), v(x), 1)$ be the flow vector
- Horn-Schunck (recall)
  \[ \int \int (I_x u + I_y v + I_t)^2 + \alpha (|\nabla u|^2 + |\nabla v|^2) \, dx \, dy \]
- Robust estimation
  \[ \int \int \psi(|I(x + w) - I(x)|^2) + \alpha \phi(|\nabla u|^2 + |\nabla v|^2) \, dx \, dy \]
- Robust estimation with Lucas-Kanade
  \[ \int \int g * \psi(|I(x + w) - I(x)|^2) + \alpha \phi(|\nabla u|^2 + |\nabla v|^2) \, dx \, dy \]
A unifying framework

- The robust object function

\[ \iint g \ast \psi(|I(x + w) - I(x)|^2) + \alpha \phi(|\nabla u|^2 + |\nabla v|^2) dx dy \]

  - Lucas-Kanade: \( \alpha = 0, \psi(z^2) = z^2 \)
  - Robust Lucas-Kanade: \( \alpha = 0, \psi(z^2) = \sqrt{z^2 + \varepsilon^2} \)
  - Horn-Schunck: \( g = 1, \psi(z^2) = z^2, \phi(z^2) = z^2 \)

- One can also learn the filters (other than gradients), and robust function \( \psi(\cdot), \phi(\cdot) \) [Roth & Black 2005]
Derivation strategies

- **Euler-Lagrange**
  - Derive in continuous domain, discretize in the end
  - Nonlinear PDE’s
  - Outer and inner fixed point iterations
  - Limited to derivative filters; cannot generalize to arbitrary filters

- **Energy minimization**
  - Discretize first and derive in matrix form
  - Easy to understand and derive
  - Iteratively reweighted least square (IRLS)

- **Variational optimization**

- **Euler-Lagrange = Variational optimization = IRLS**
Iteratively reweighted least square (IRLS)

- Let $\phi(z) = (z^2 + \varepsilon^2)^\eta$ be a robust function

- We want to minimize the objective function

$$\Phi(Ax + b) = \sum_{i=1}^{n} \phi \left( (a_i^T x + b_i)^2 \right)$$

where $x \in \mathbb{R}^d$, $A = [a_1 \ a_2 \ \cdots \ a_n]^T \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$

- By setting $\frac{\partial \Phi}{\partial x} = 0$, we can derive

$$\frac{\partial \Phi}{\partial x} \propto \sum_{i=1}^{n} \phi' \left( (a_i^T x + b_i)^2 \right) (a_i^T x + b_i) a_i$$

$$= \sum_{i=1}^{n} w_{ii} a_i^T x a_i + w_{ii} b_i a_i$$

$$= \sum_{i=1}^{n} a_i^T w_{ii} x a_i + b_i w_{ii} a_i$$

$$= A^T W A x + A^T W b$$

$$w_{ii} = \phi' \left( (a_i^T x + b_i)^2 \right)$$

$$W = \text{diag}(\Phi'(Ax + b))$$
Iteratively reweighted least square (IRLS)

- Derivative: \( \frac{\partial \Phi}{\partial x} = A^T W A x + A^T W b = 0 \)
- Iterate between *reweighting* and *least square*

1. Initialize \( x = x_0 \)
2. Compute weight matrix \( W = \text{diag}(\Phi'(Ax + b)) \)
3. Solve the linear system \( A^T W A x = -A^T W b \)
4. If \( x \) converges, return; otherwise, go to 2

- Convergence is guaranteed (local minima)
IRLS for robust optical flow

- **Objective function**
  \[
  \int \int g \ast \psi(|I(x + w) - I(x)|^2) + \alpha \phi(|\nabla u|^2 + |\nabla v|^2) dx dy
  \]

- **Discretize, linearize and increment**
  \[
  \sum_{x,y} g \ast \psi \left(|I_t + I_x du + I_y dv|^2\right) + \alpha \phi(|\nabla (u + du)|^2 + |\nabla (v + dv)|^2)
  \]

- **IRLS (initialize \( du = dv = 0 \))**
  - Reweight: 
    \[
    \Psi'_{xx} = \text{diag}(g \ast \psi'I_x I_x), \Psi'_{xy} = \text{diag}(g \ast \psi'I_x I_y), \\
    \Psi'_{yy} = \text{diag}(g \ast \psi'I_y I_y), \Psi'_{xt} = \text{diag}(g \ast \psi'I_x I_t), \\
    \Psi'_{yt} = \text{diag}(g \ast \psi'I_y I_t), L = D_x^T \Phi' D_x + D_y^T \Phi' D_y
    \]
  - Least square:
    \[
    \begin{bmatrix}
    \Psi'_{xx} + \alpha L & \Psi'_{xy} \\
    \Psi'_{xy} & \Psi'_{yy} + \alpha L
    \end{bmatrix}
    \begin{bmatrix}
    du \\
    dv
    \end{bmatrix}
    =
    - \begin{bmatrix}
    \Psi'_{xt} + \alpha LU \\
    \Psi'_{yt} + \alpha LV
    \end{bmatrix}
    \]
Example

Input two frames

Robust optical flow

Horn-Schunck

Flow visualization

Coarse-to-fine LK with median filtering
Contents

• Motion perception
• Motion representation
• Parametric motion: Lucas-Kanade
• Dense optical flow: Horn-Schunck
• Robust estimation

• Applications (1)
Video stabilization
Video denoising
Video super resolution

Low-Res
Summary

• Lucas-Kanade
  – Parametric motion
  – Dense flow field (with median filtering)
• Horn-Schunck
  – Gaussian Markov random field
  – Euler-Lagrange
• Robust flow estimation
  – Robust function
    • Account for outliers in the data term
    • Encourage piecewise smoothness
  – IRLS (= nonlinear PDE = variational optimization)
Contents (next time)

• Feature matching
• Discrete optical flow
• Layer motion analysis
• Contour motion analysis
• Obtaining motion ground truth
• Applications (2)