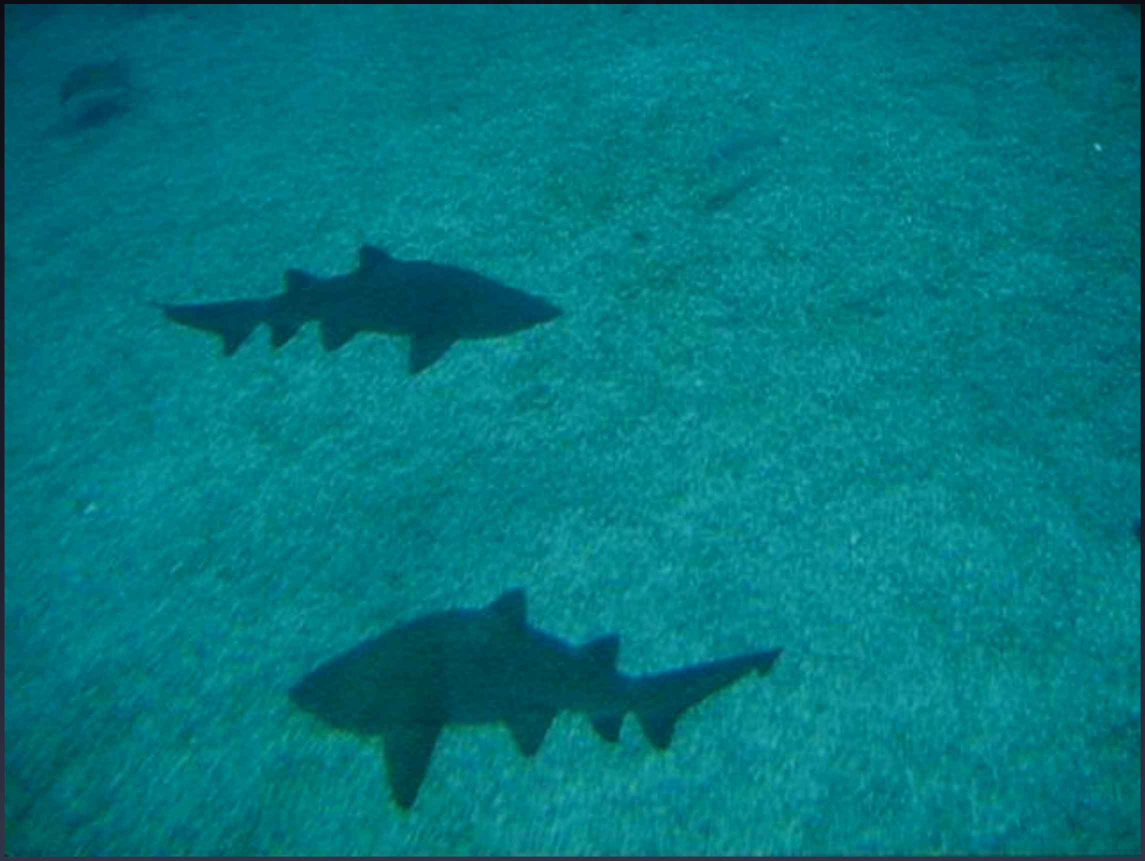
Motion Estimation

Ce Liu
celiu@google.com
Google Research

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)

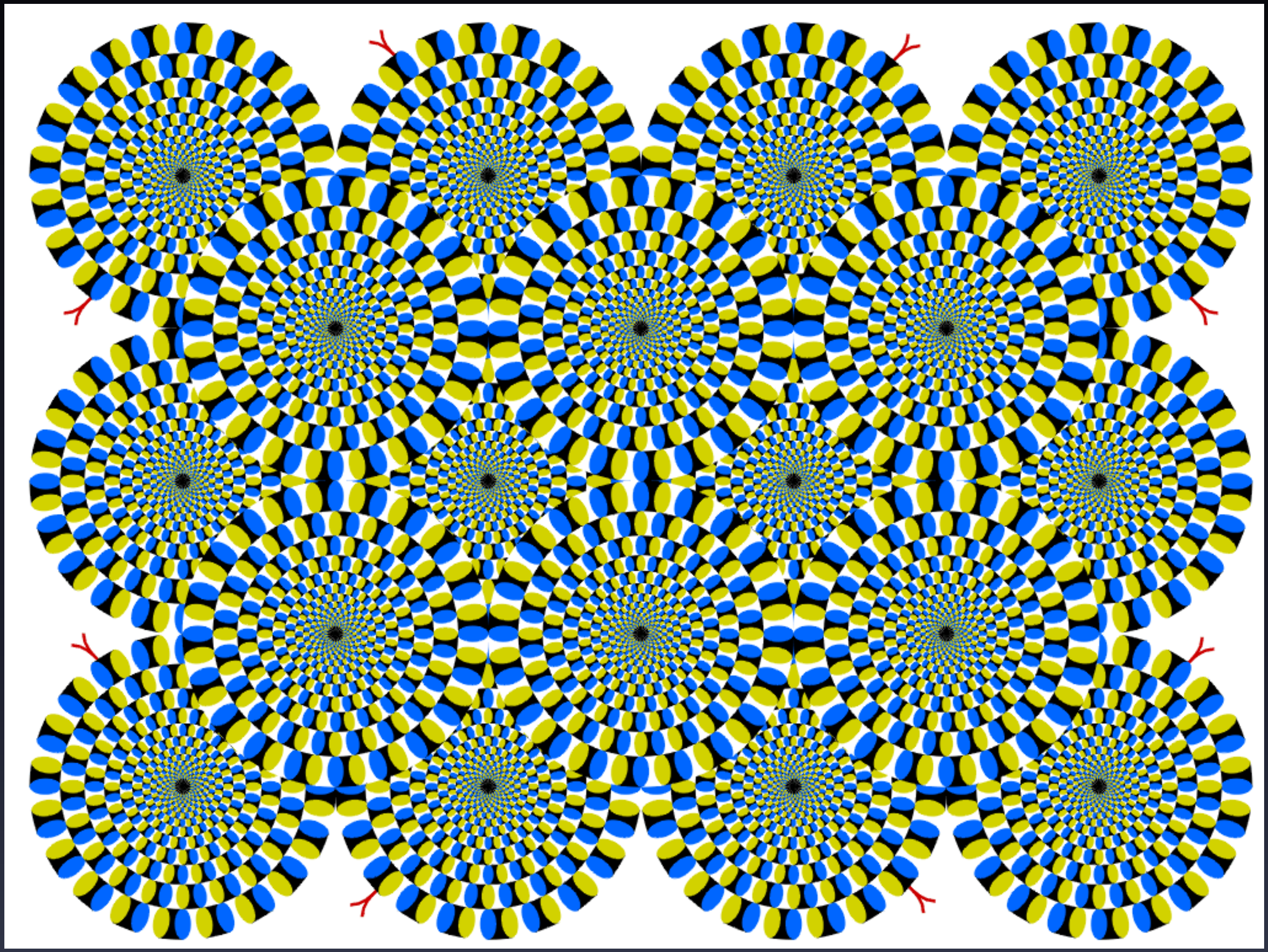
Readings

- Rick’s book: Chapter 8
- Ce Liu’s PhD thesis (Appendix A & B)
- S. Baker and I. Matthews. Lucas-Kanade 20 years on: a unifying framework. IJCV 2004
- Horn-Schunck (wikipedia)
- A. Bruhn, J. Weickert, C. Schnorr. Lucas/Kanade meets Horn/Schunk: combining local and global optical flow methods. IJCV 2005

Contents

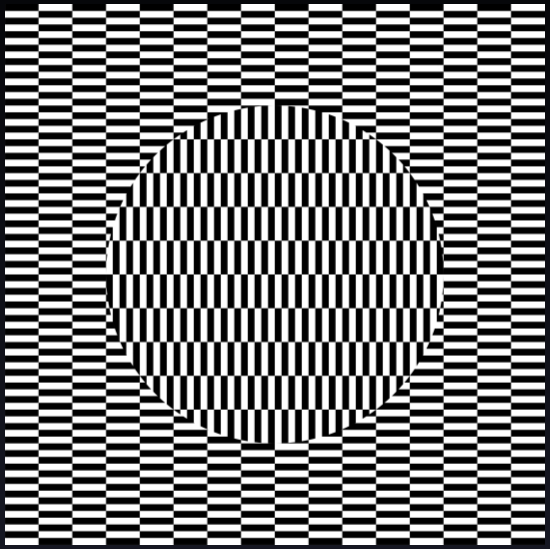
- Motion perception
- Motion representation
- Parametric motion: Lucas-Kanade
- Dense optical flow: Horn-Schunck
- Robust estimation
- Applications (1)

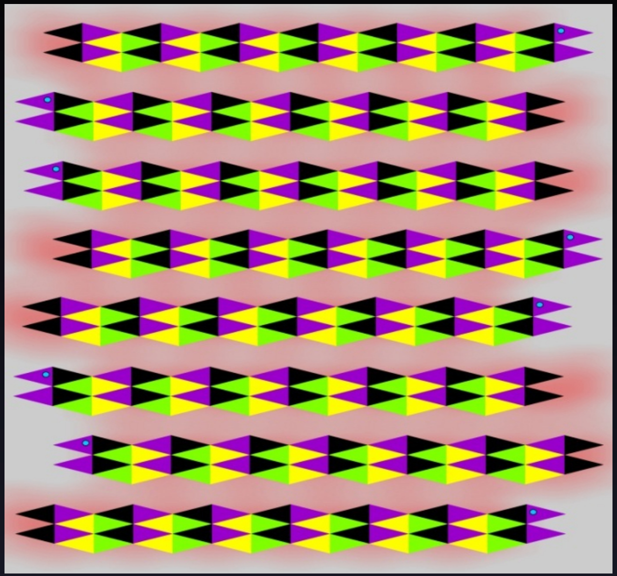
Seeing motion from a static picture?

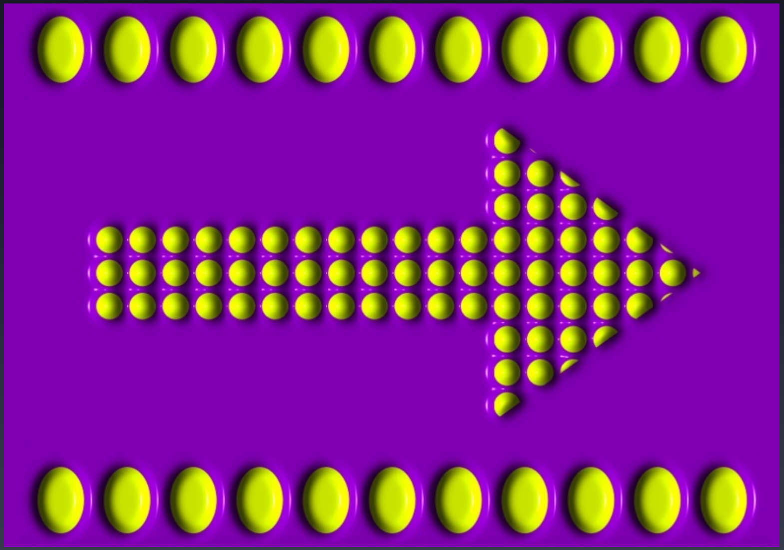


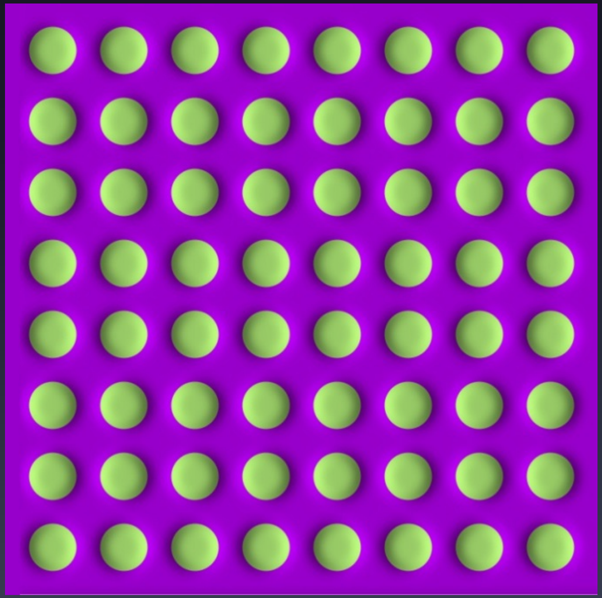
http://www.ritsumei.ac.jp/~akitaoka/index-e.html

More examples



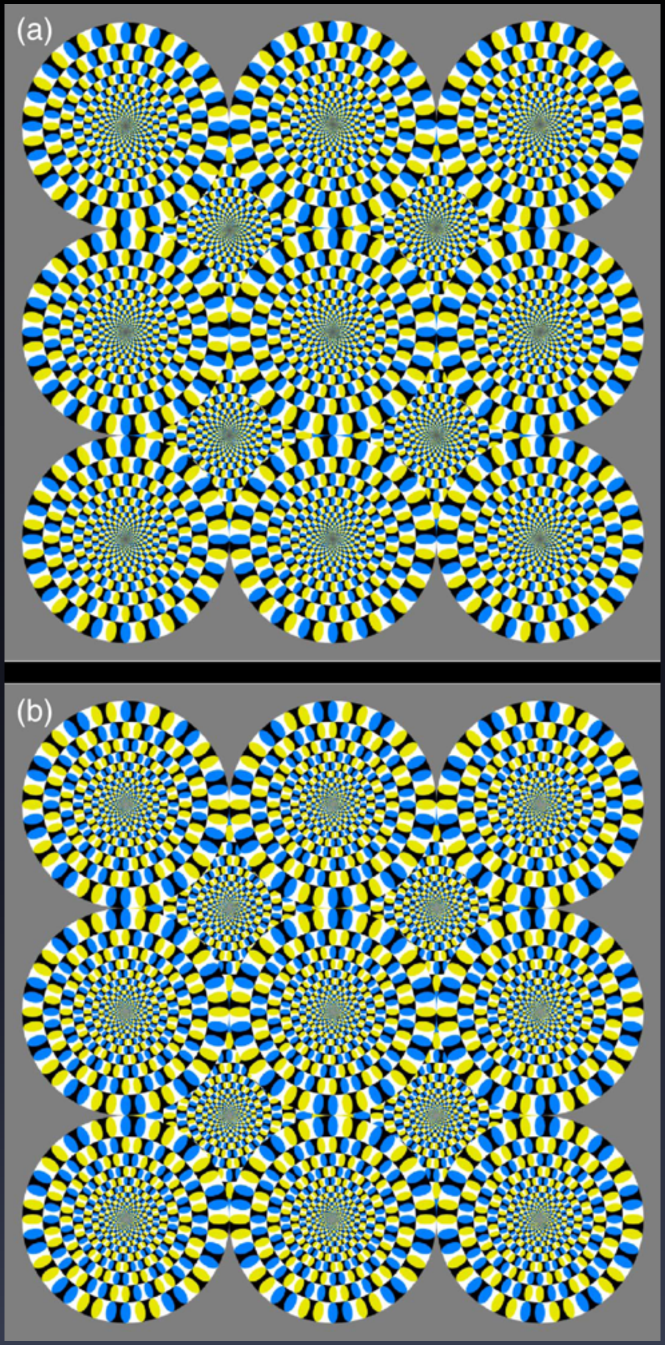




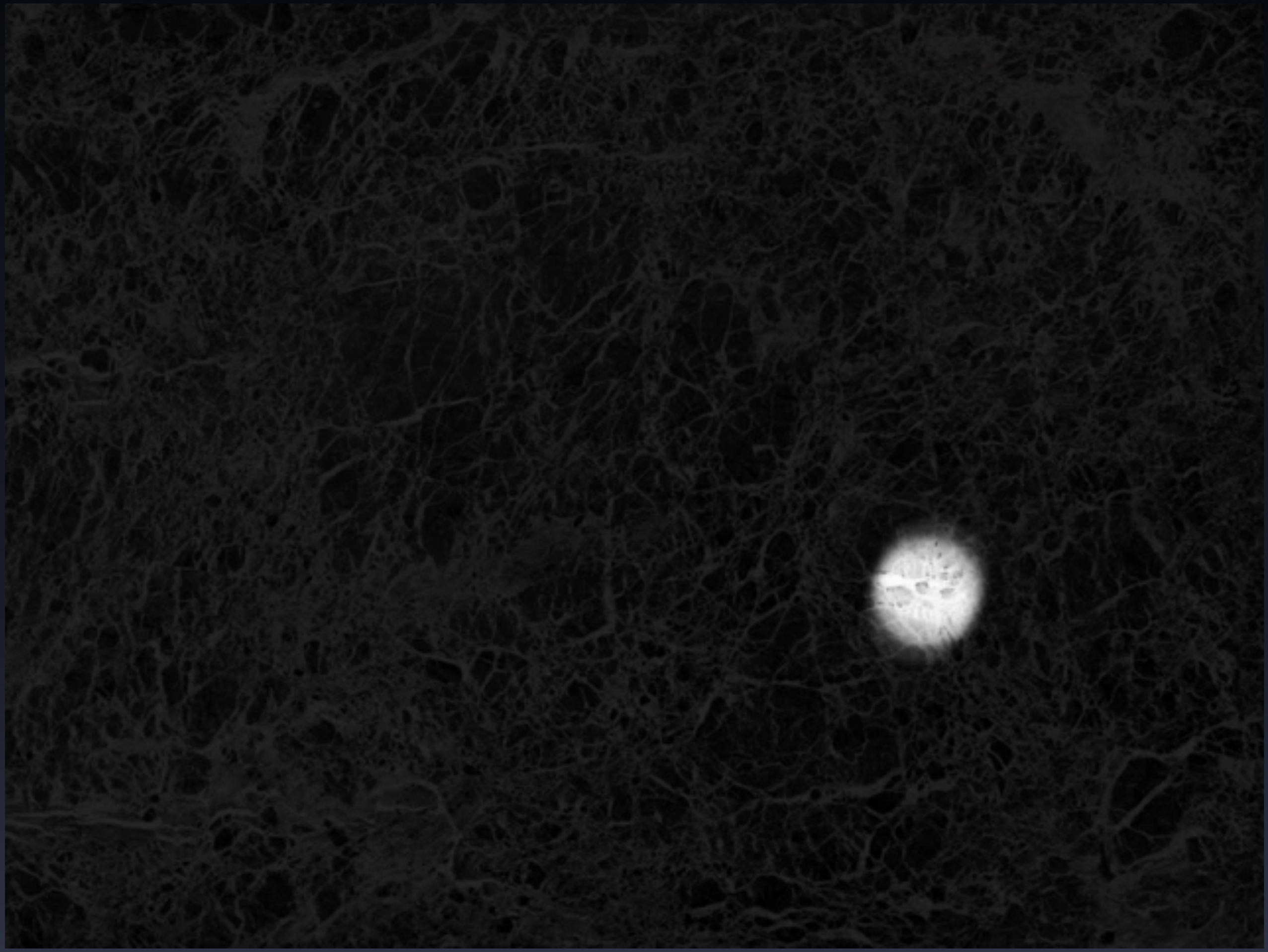


How is this possible?

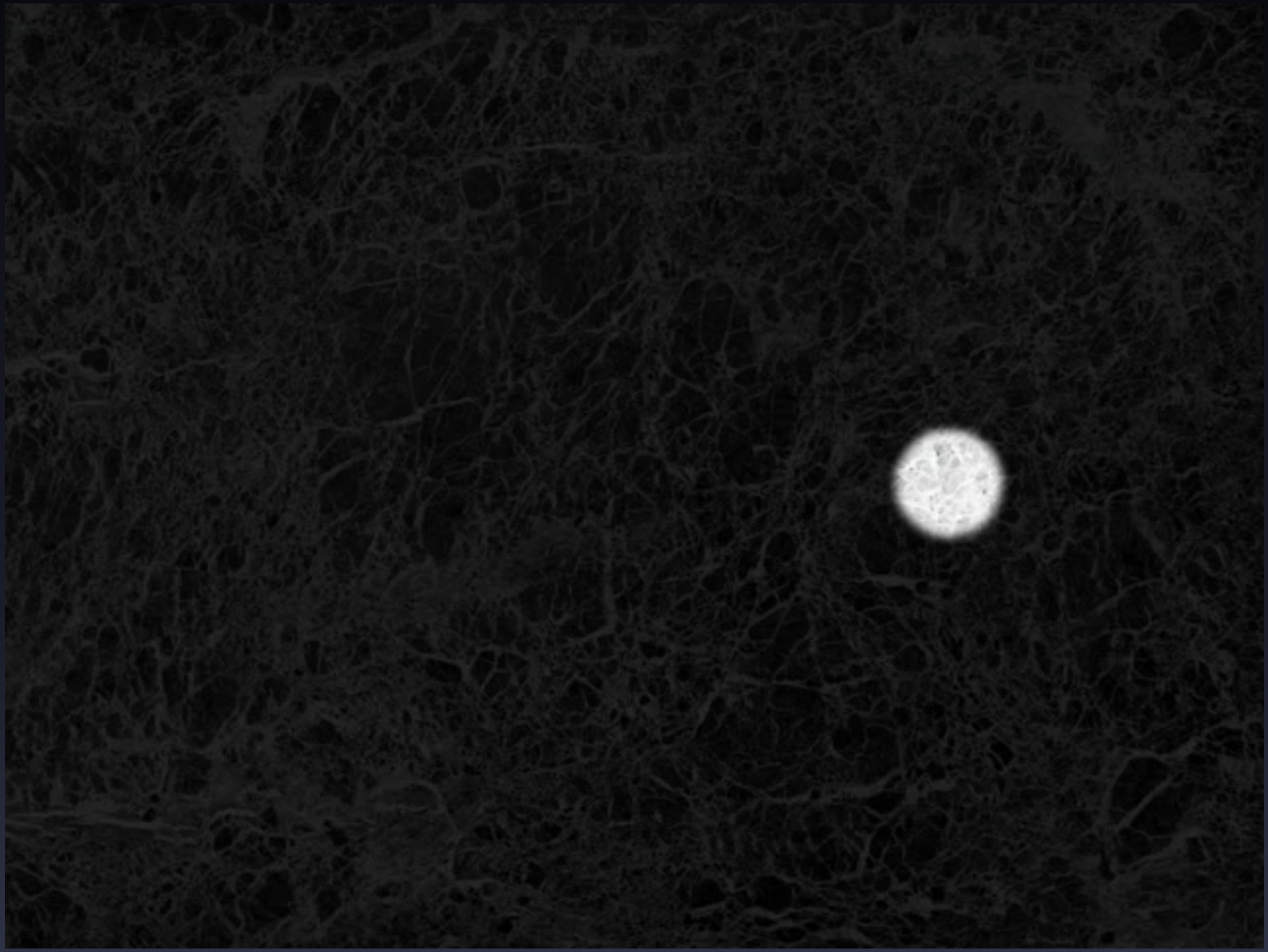
- The true mechanism is to be revealed
- FMRI data suggest that illusion is related to some component of eye movements
- We don't expect computer vision to "see" motion from these stimuli, yet



What do you see?



In fact, …



The cause of motion

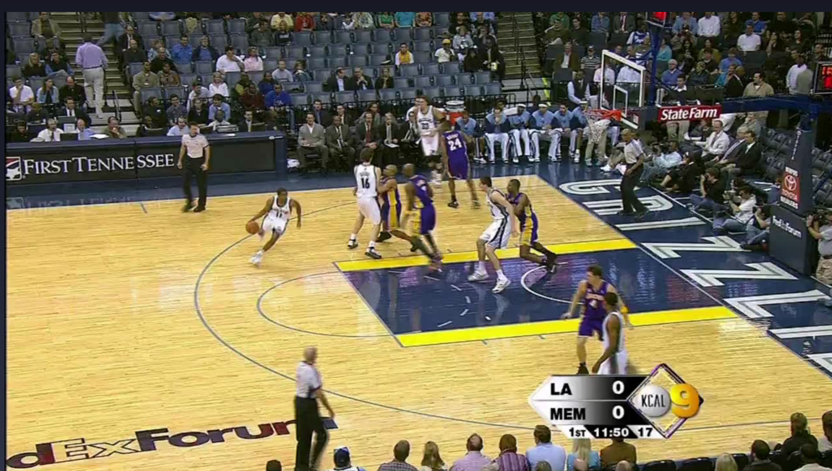
- Three factors in imaging process
  - Light
  - Object
  - Camera

- Varying either of them causes motion
  - Static camera, moving objects (surveillance)
  - Moving camera, static scene (3D capture)
  - Moving camera, moving scene (sports, movie)
  - Static camera, moving objects, 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

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

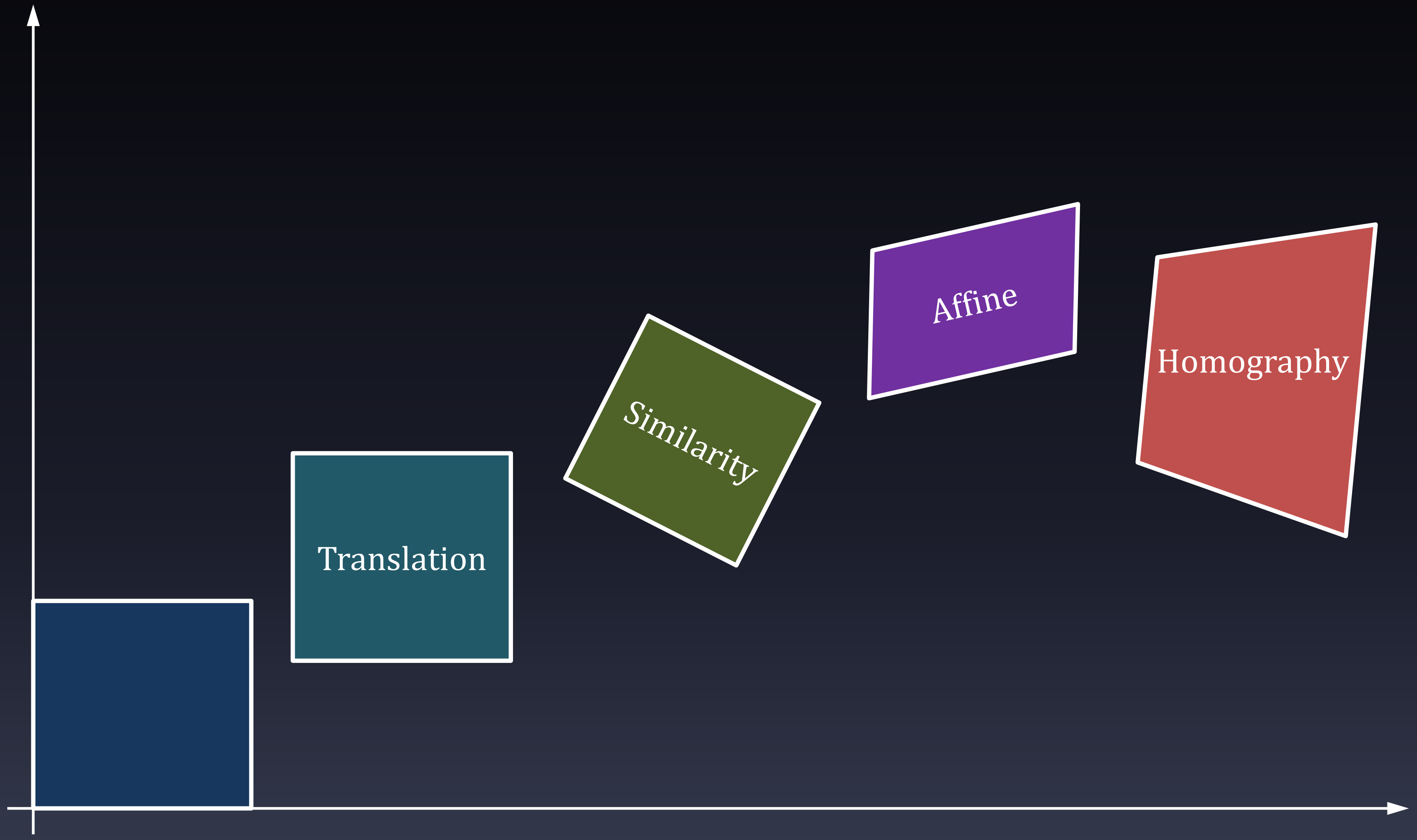
Contents

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

Parametric motion

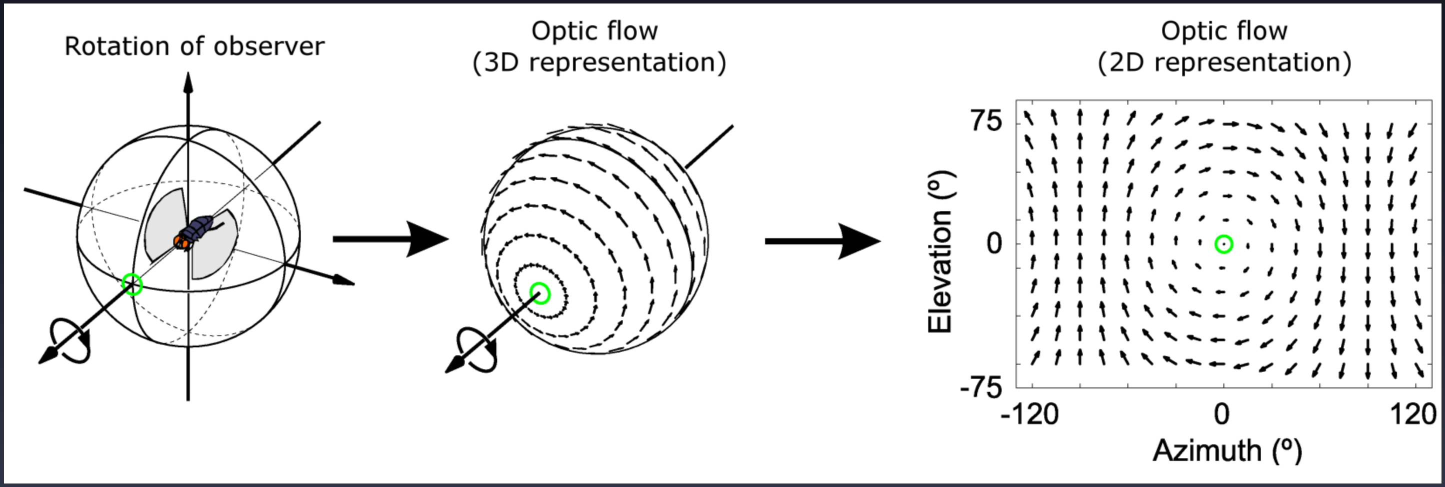
- Mapping: $(x_1, y_1) \to (x_2, y_2)$
  - (x_1, y_1): point in frame 1
  - (x_2, y_2): corresponding point in frame 2
- Global parametric motion: $(x_2, y_2) = f(x_1, y_1; \theta)$
- Forms of parametric motion
  - Translation: $\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 + a \\ y_1 + b \end{bmatrix}$
  - 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



Optical flow field

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

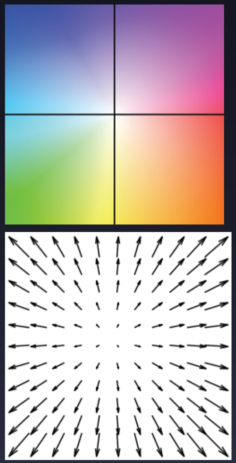


Optical flow field visualization

- 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
- Matching criteria
  - What's invariant between two images?
    - Brightness, gradients, phase, other features…
  - Distance metric (L2, robust functions)

$$E(u, v) = \sum_{x,y} \left(I_1(x, y) - I_2(x + u, y + v) \right)^2$$

  - 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 $\mathbf{x} = (x, y)^T$ be a column vector indexing pixel coordinate
- Two typical transforms
  - Translation: $W(\mathrm{x}; \mathrm{p}) = \begin{bmatrix} x + p_1 \\ y + p_2 \end{bmatrix}$
  - Affine: $W(\mathrm{x}; \mathrm{p}) = \begin{bmatrix} p_1 x + p_3 y + p_5 \\ p_2 x + p_4 y + p_6 \end{bmatrix} = \begin{bmatrix} p_1 & p_3 & p_5 \\ p_2 & p_4 & p_6 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
- Goal of the Lucas-Kanade algorithm

$$\mathrm{p}^* = \arg\min_{\mathrm{p}} \sum_{\mathrm{x}} \left[I_2\big(W(\mathrm{x}; \mathrm{p})\big) - I_1(\mathrm{x}) \right]^2$$

An incremental algorithm

- Difficult to directly optimize the objective function

$$\mathrm{p}^* = \arg\min_{\mathrm{p}} \sum_{\mathrm{x}} \left[I_2\left(W(\mathrm{x};\mathrm{p})\right) - I_1(\mathrm{x}) \right]^2$$

- Instead, we try to optimize each step

$$\Delta\mathrm{p}^* = \arg\min_{\Delta\mathrm{p}} \sum_{\mathrm{x}} \left[I_2\left(W(\mathrm{x};\mathrm{p}+\Delta\mathrm{p})\right) - I_1(\mathrm{x}) \right]^2$$

- The transform parameter is updated:

$$\mathrm{p} \leftarrow \mathrm{p} + \Delta\mathrm{p}^*$$

Taylor expansion

- The term $I_2\big(W(\mathrm{x};\mathrm{p}+\Delta\mathrm{p})\big)$ is highly nonlinear
- Taylor expansion:

$$I_2\big(W(\mathrm{x};\mathrm{p}+\Delta\mathrm{p})\big) \approx I_2\big(W(x;p)\big) + \nabla I_2 \frac{\partial W}{\partial \mathrm{p}} \Delta\mathrm{p}$$

- $\frac{\partial W}{\partial \mathrm{p}}$: *Jacobian* of the warp
- If $W(\mathrm{x};\mathrm{p}) = \big(W_x(\mathrm{x};\mathrm{p}), W_y(\mathrm{x};\mathrm{p})\big)^T$, then

$$\frac{\partial W}{\partial \mathrm{p}} = \begin{bmatrix} \frac{\partial W_x}{\partial p_1} & \cdots & \frac{\partial W_x}{\partial p_n} \\ \frac{\partial W_y}{\partial p_1} & \cdots & \frac{\partial W_y}{\partial p_n} \end{bmatrix}$$

Jacobian matrix

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

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

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

  The Jacobian is $\frac{\partial W}{\partial \mathrm{p}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Taylor expansion

- $\nabla I_2 = [I_x \; I_y]$ is the gradient of image I_2 evaluated at $W(\mathrm{x};\mathrm{p})$: compute the gradients in the coordinate of I_2 and warp back to the coordinate of I_1
- For affine transform $\frac{\partial W}{\partial \mathrm{p}} = \begin{bmatrix} x & 0 & y & 0 & 1 & 0 \\ 0 & x & 0 & y & 0 & 1 \end{bmatrix}$

$$\nabla I_2 \frac{\partial W}{\partial \mathrm{p}} = [I_x x \quad I_y x \quad I_x y \quad I_y y \quad I_x \quad I_y]$$

- Let matrix $\mathbf{B} = \left[\mathbf{I}_x\mathbf{X} \;\; \mathbf{I}_y\mathbf{X} \;\; \mathbf{I}_x\mathbf{Y} \;\; \mathbf{I}_y\mathbf{Y} \;\; \mathbf{I}_x \;\; \mathbf{I}_y\right] \in \mathbb{R}^{n\times 6}$, $\mathbf{I}_x$ and $\mathbf{X}$ are both column vectors. $\mathbf{I}_x\mathbf{X}$ is element-wise vector multiplication.

Gauss-Newton

- With Taylor expansion, the objective function becomes

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

Or in a vector form:

$$\Delta \mathrm{p}^* = \arg\min_{\Delta \mathrm{p}} (\mathbf{I}_t + \mathbf{B}\Delta \mathrm{p})^T (\mathbf{I}_t + \mathbf{B}\Delta \mathrm{p})$$

Where $\mathbf{B} = \left[\mathbf{I}_x\mathbf{X} \;\; \mathbf{I}_y\mathbf{X} \;\; \mathbf{I}_x\mathbf{Y} \;\; \mathbf{I}_y\mathbf{Y} \;\; \mathbf{I}_x \;\; \mathbf{I}_y \right] \in \mathbb{R}^{n\times 6}$

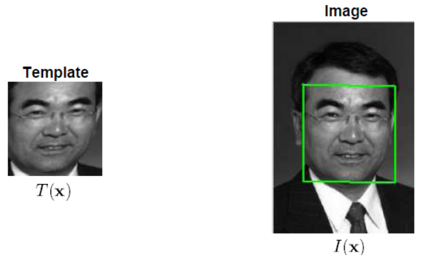
$$\mathbf{I}_t = \mathbf{I}_2\big(\mathbf{W}(\mathrm{p})\big) - \mathbf{I}_1$$

- Solution:

$$\Delta \mathrm{p}^* = -(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{I}_t$$

Hessian matrix

How it works



How it works

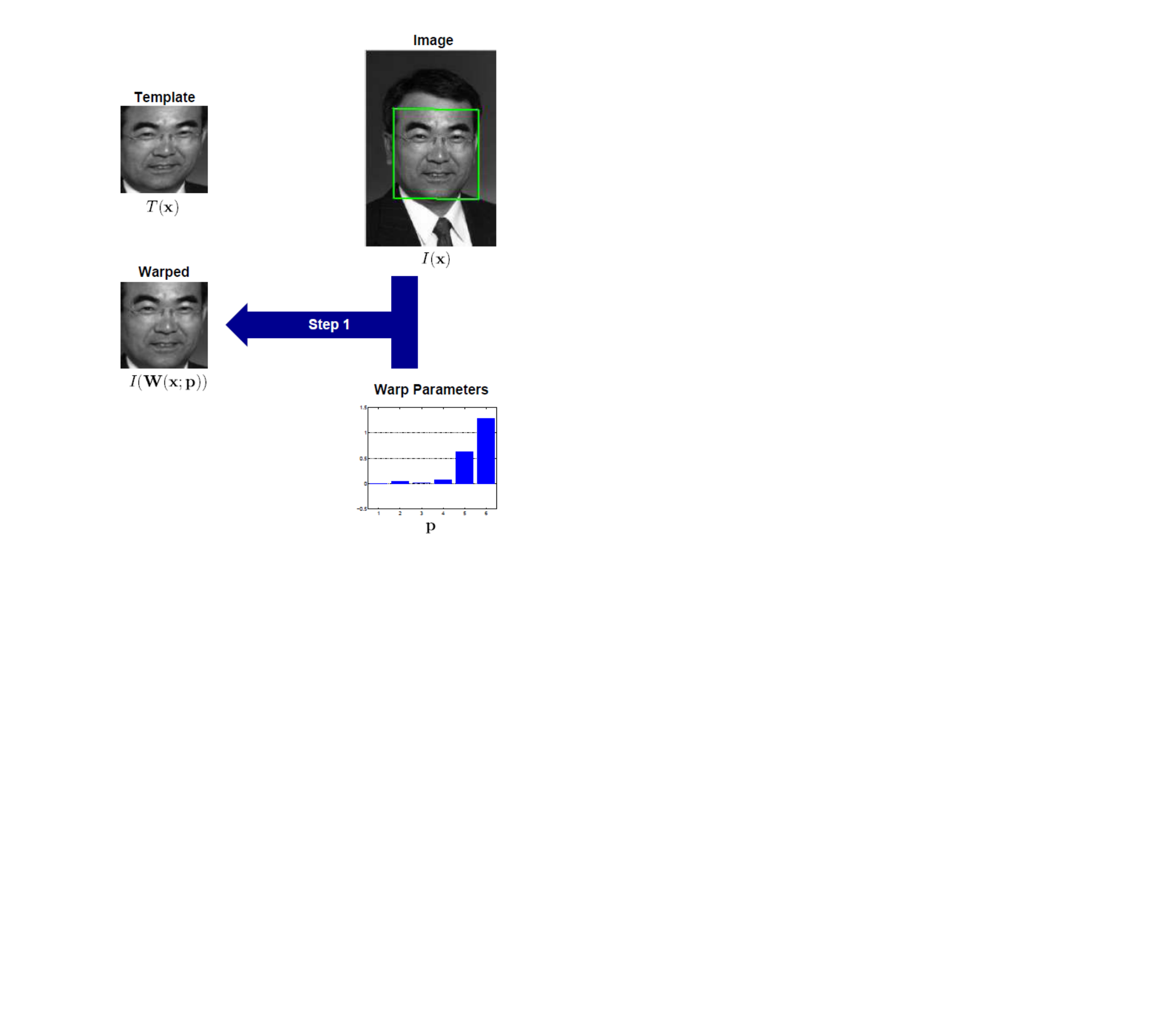


Image
Template
$T(\mathbf{x})$
$I(\mathbf{x})$
Warped
Step 1
$I(\mathbf{W}(\mathbf{x};\mathbf{p}))$
Warp Parameters
$\mathbf{p}$

How it works

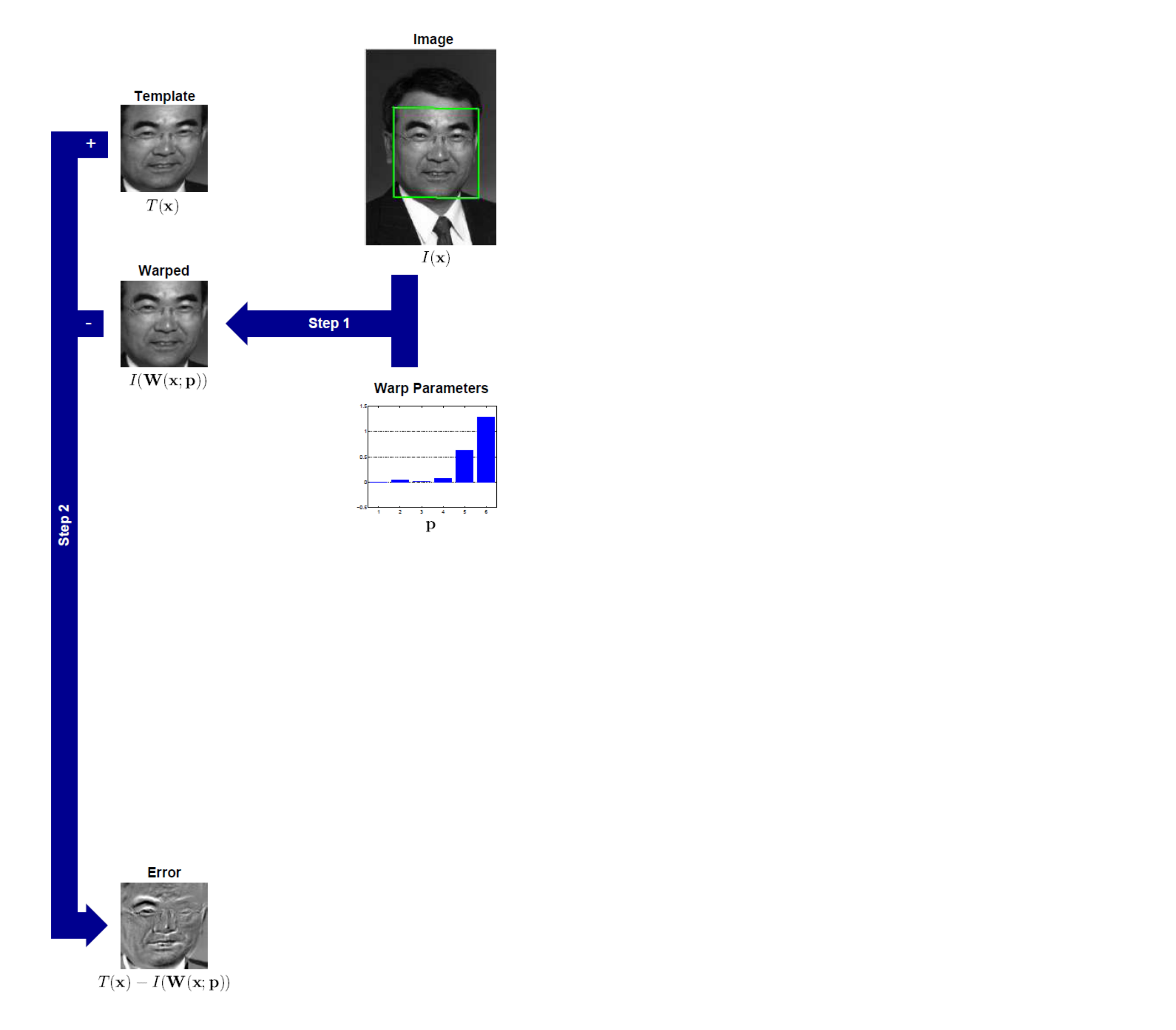


Image
Template
$T(\mathbf{x})$
$I(\mathbf{x})$
Warped
Step 1
$I(\mathbf{W}(\mathbf{x};\mathbf{p}))$
Warp Parameters
$\mathbf{p}$
Step 2
+
-
Error
$T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))$

How it works

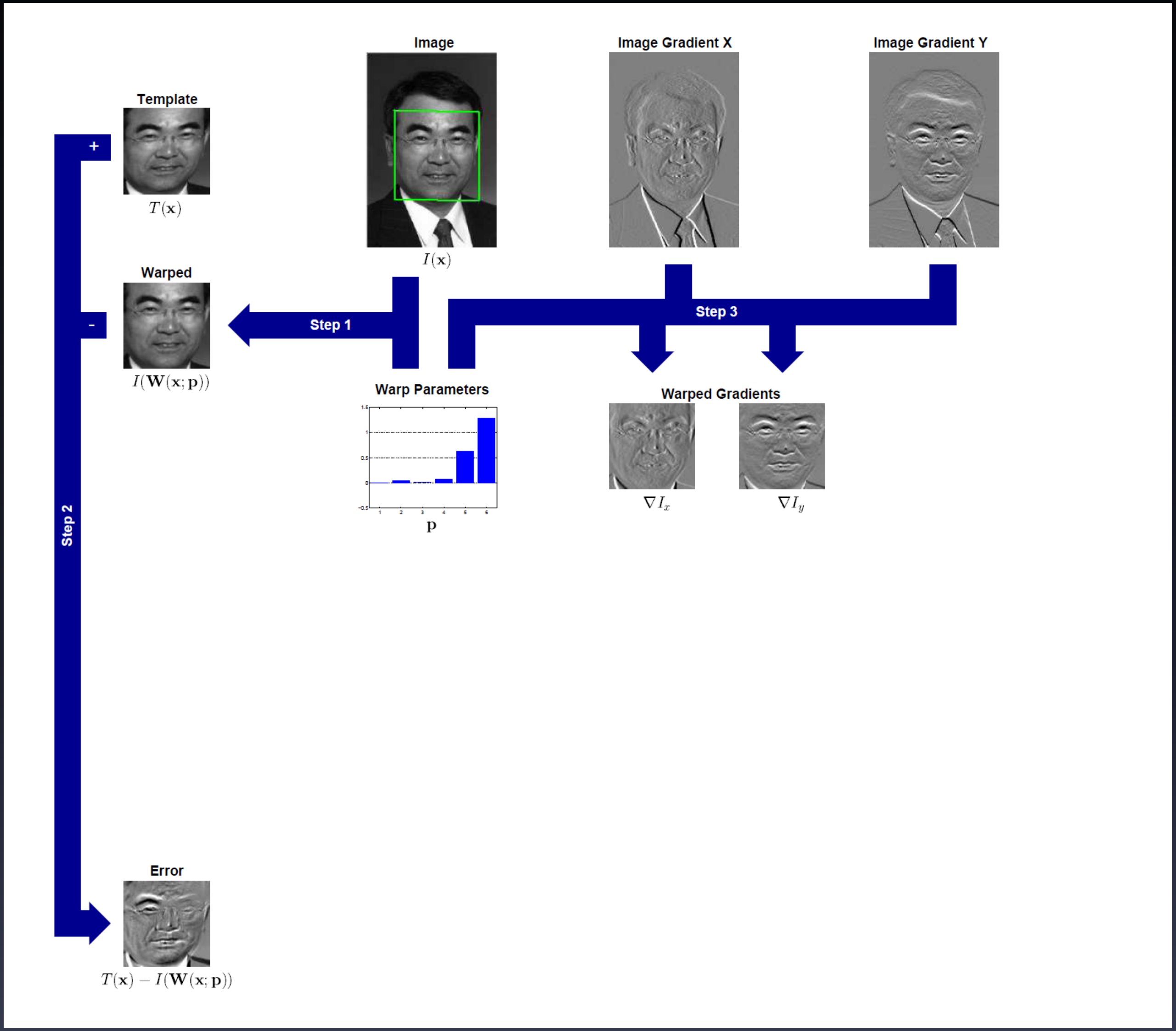


Image
Image Gradient X
Image Gradient Y
Template
$T(\mathbf{x})$
$I(\mathbf{x})$
Warped
Step 1
Step 3
+
-
$I(\mathbf{W}(\mathbf{x};\mathbf{p}))$
Warp Parameters
Warped Gradients
$\mathbf{p}$
∇I_x
∇I_y
Step 2
Error
$T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))$

How it works

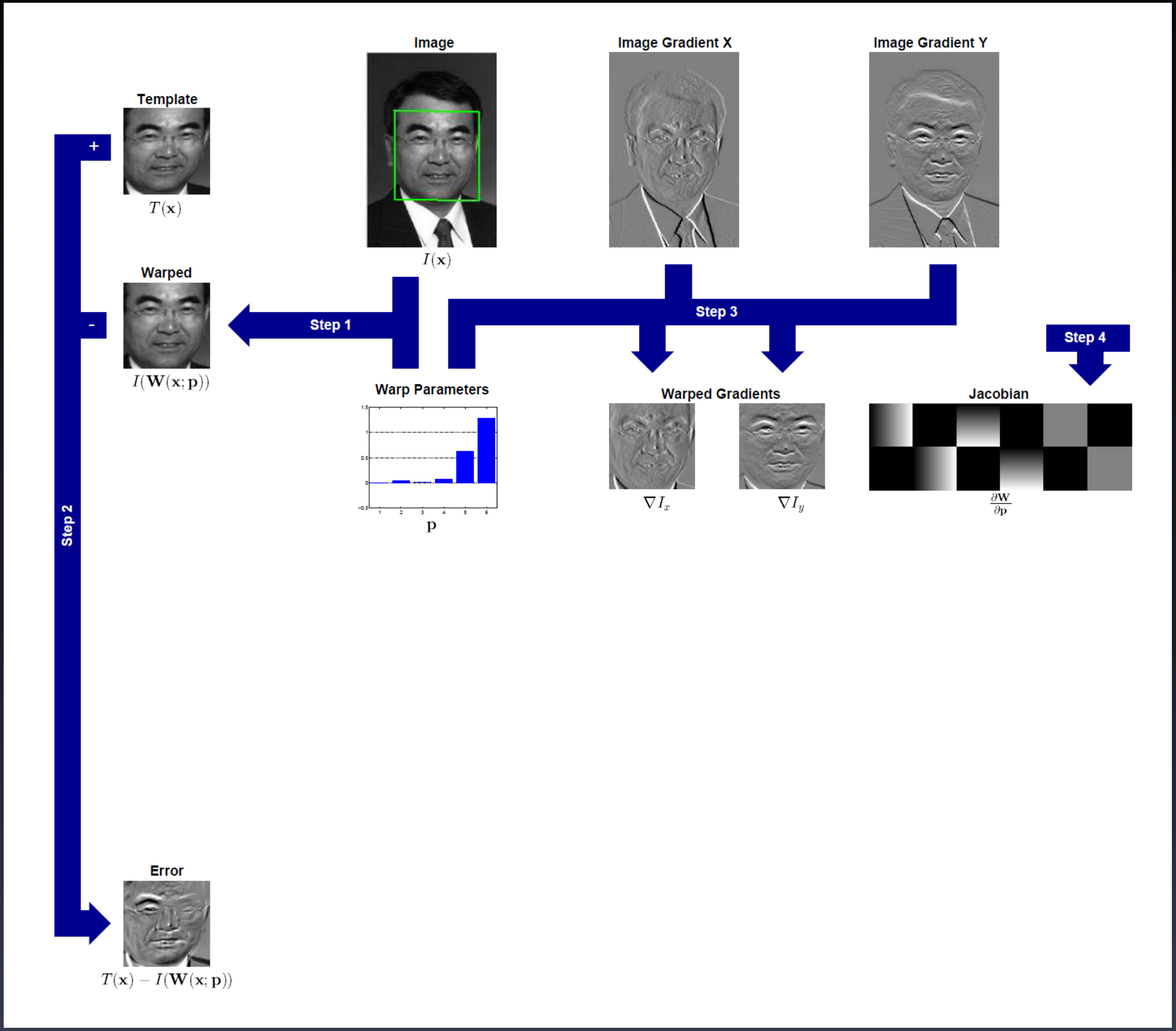
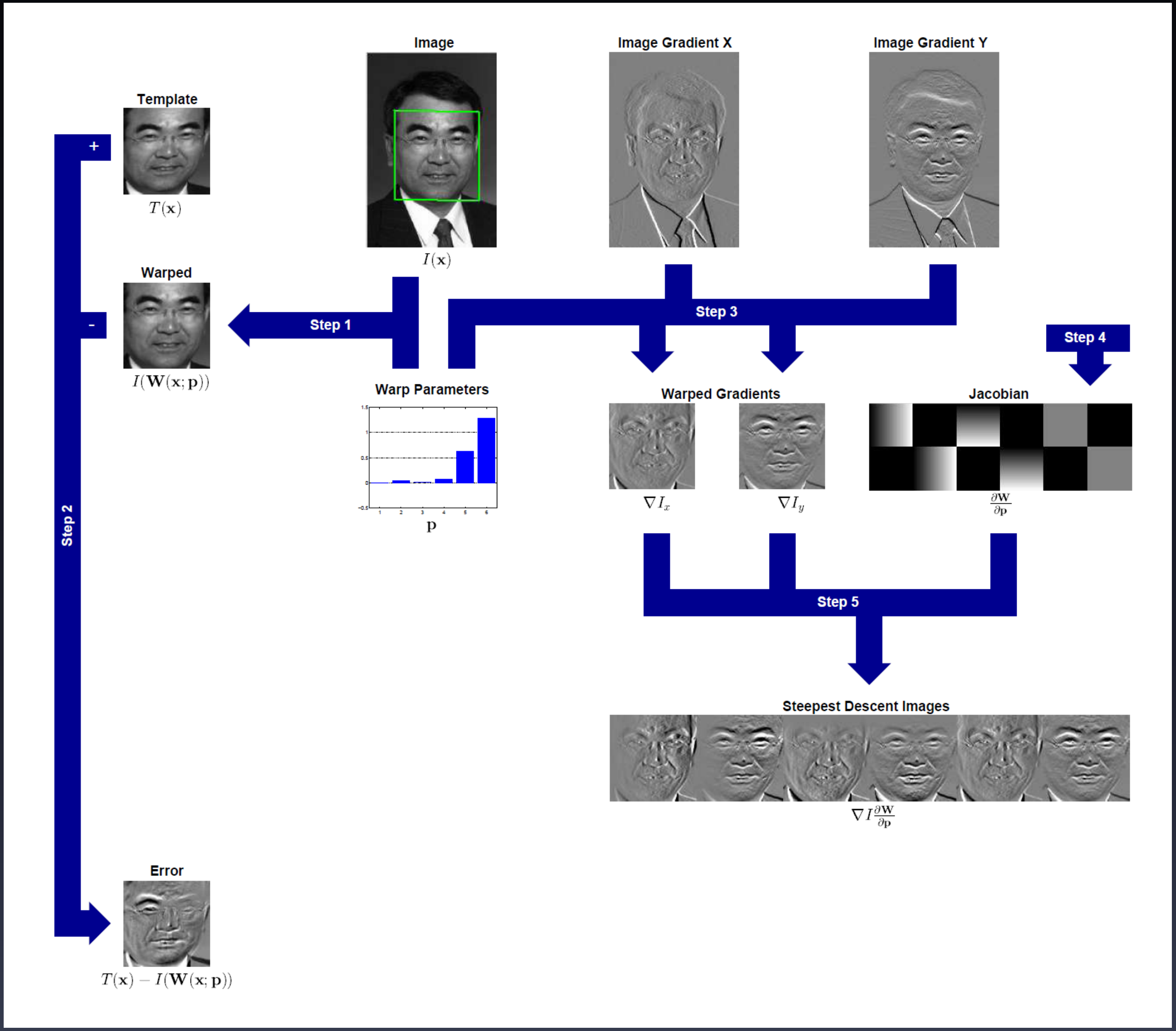


Image
Image Gradient X
Image Gradient Y
Template
$T(\mathbf{x})$
$I(\mathbf{x})$
+
Warped
-
Step 1
Step 3
Step 4
$I(\mathbf{W}(\mathbf{x};\mathbf{p}))$
Warp Parameters
Warped Gradients
Jacobian
$\mathbf{p}$
∇I_x
∇I_y
$\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$
Step 2
Error
$T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))$

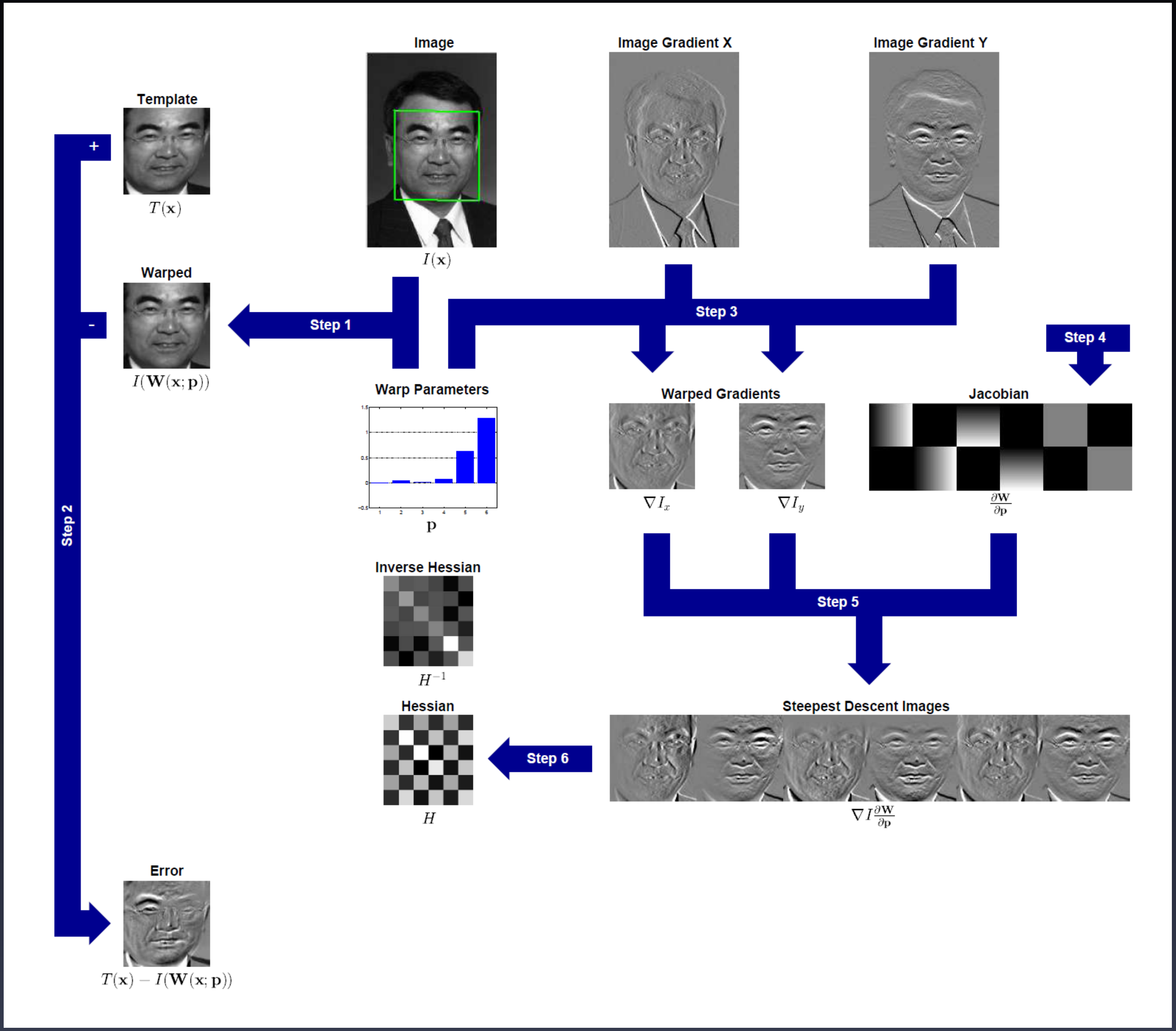
How it works



Compute matrix

$$\mathbf{B} = \left[\nabla I_2 \frac{\partial W}{\partial \mathrm{p}}\right]$$

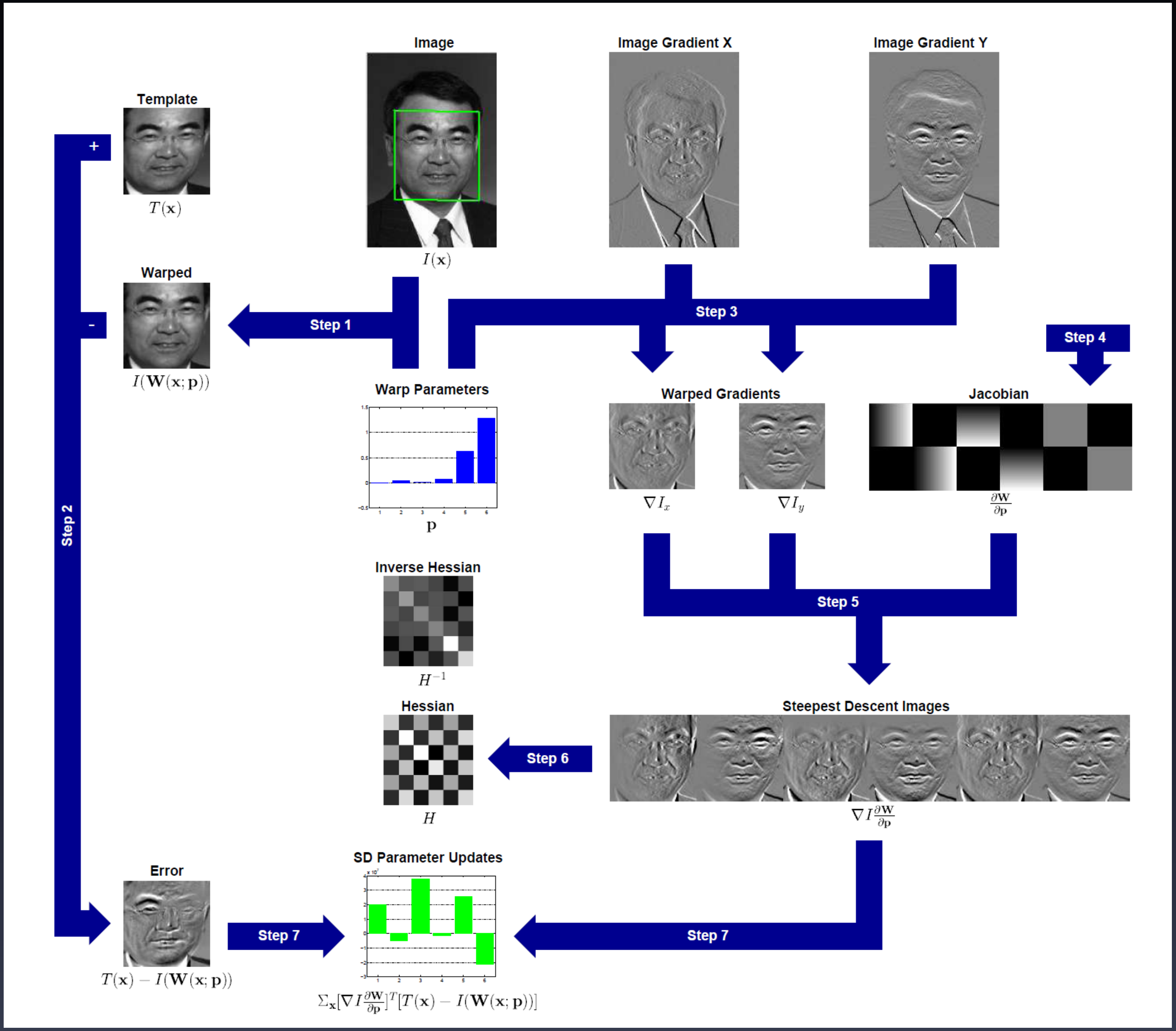
How it works



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

$$\mathbf{B} = \left[\nabla I_2 \frac{\partial W}{\partial \mathrm{p}}\right]$$

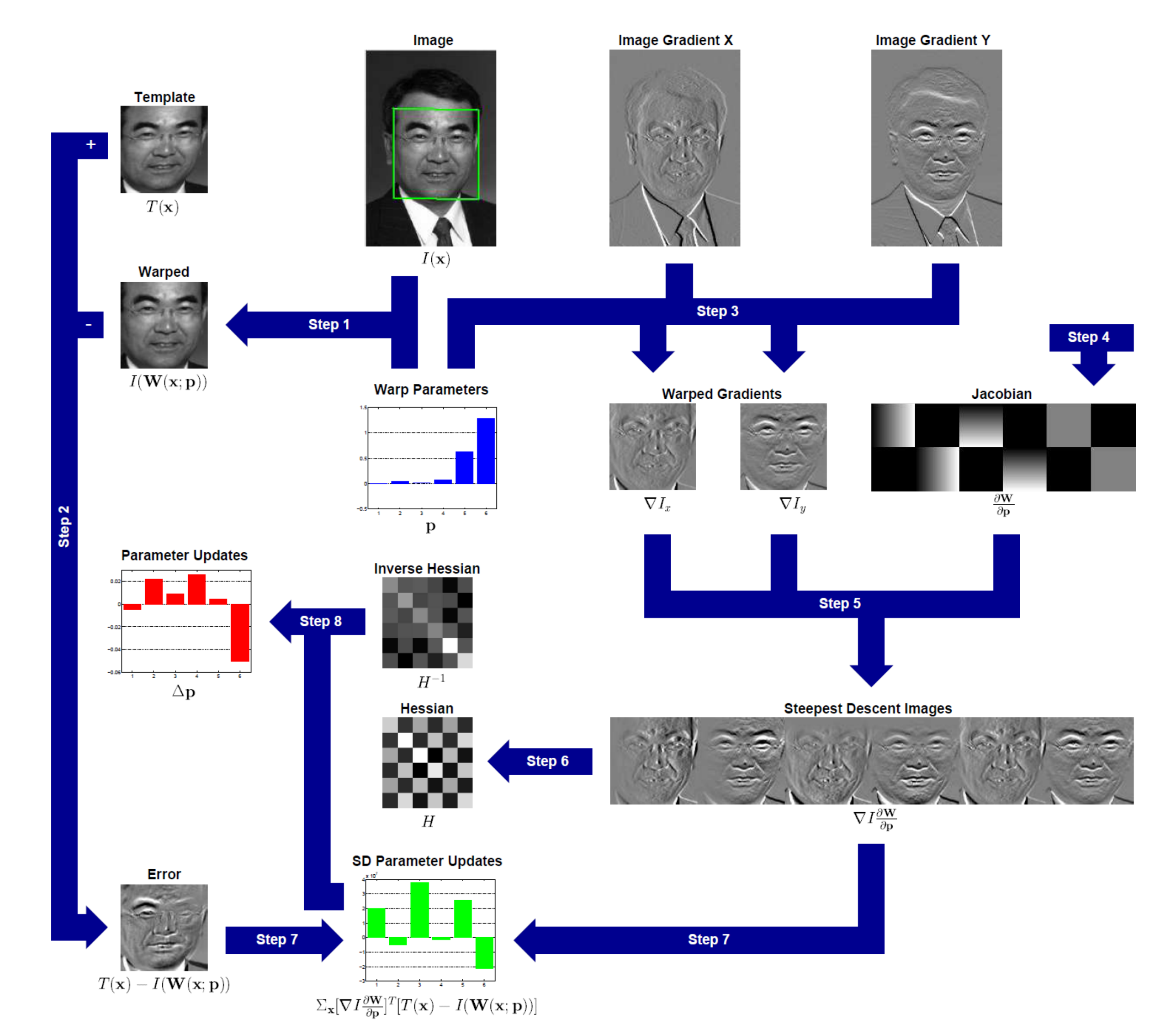
How it works



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

$$\mathbf{B} = \left[\nabla I_2 \frac{\partial W}{\partial \mathrm{p}}\right]$$

How it works

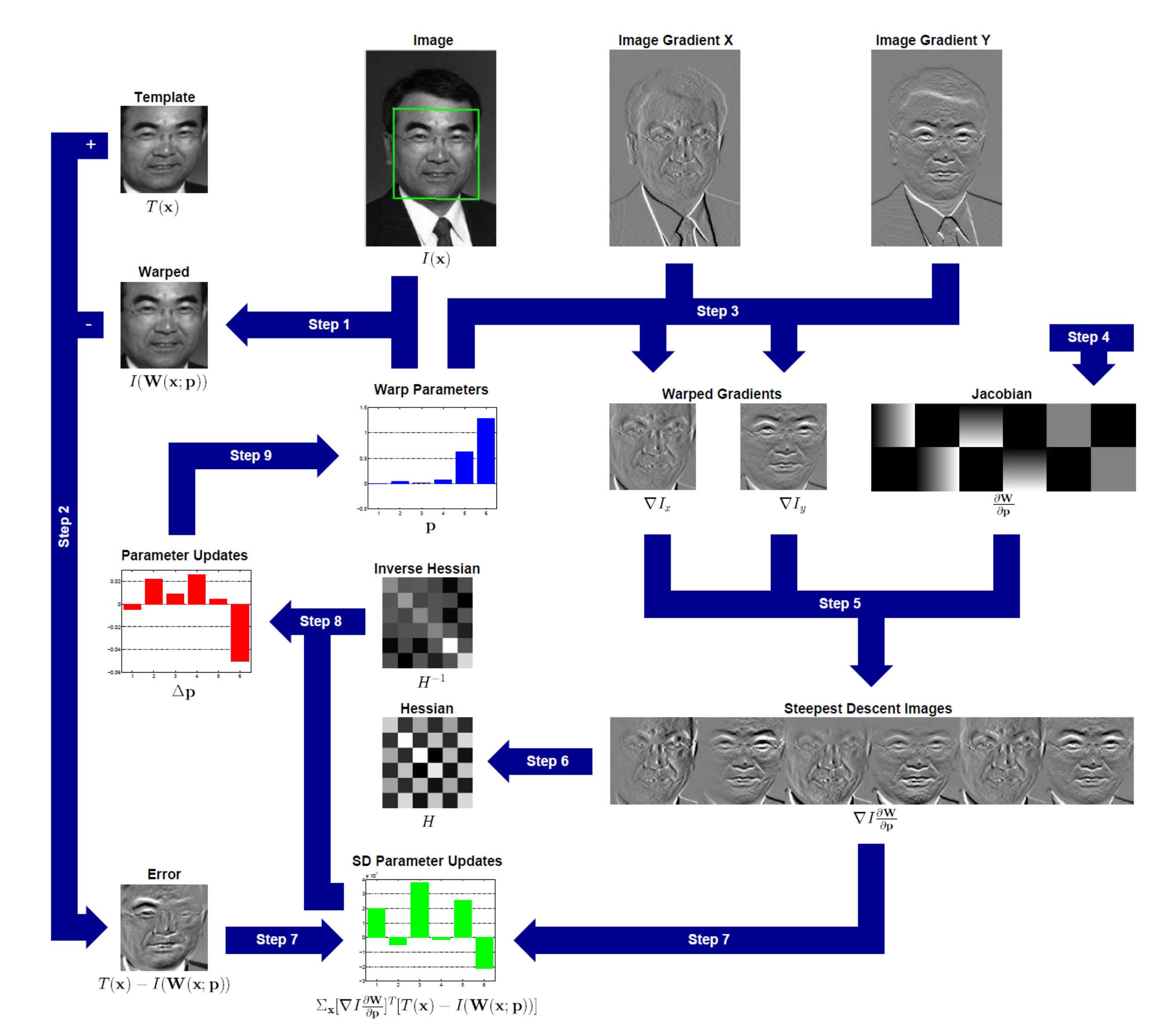


Solve linear system:

$$\Delta \mathrm{p}^* = -(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{I}_t$$

$$\mathbf{B} = \left[\nabla I_2 \frac{\partial W}{\partial \mathrm{p}}\right]$$

How it works



$$\mathrm{p} \leftarrow \mathrm{p} + \Delta \mathrm{p}^*$$

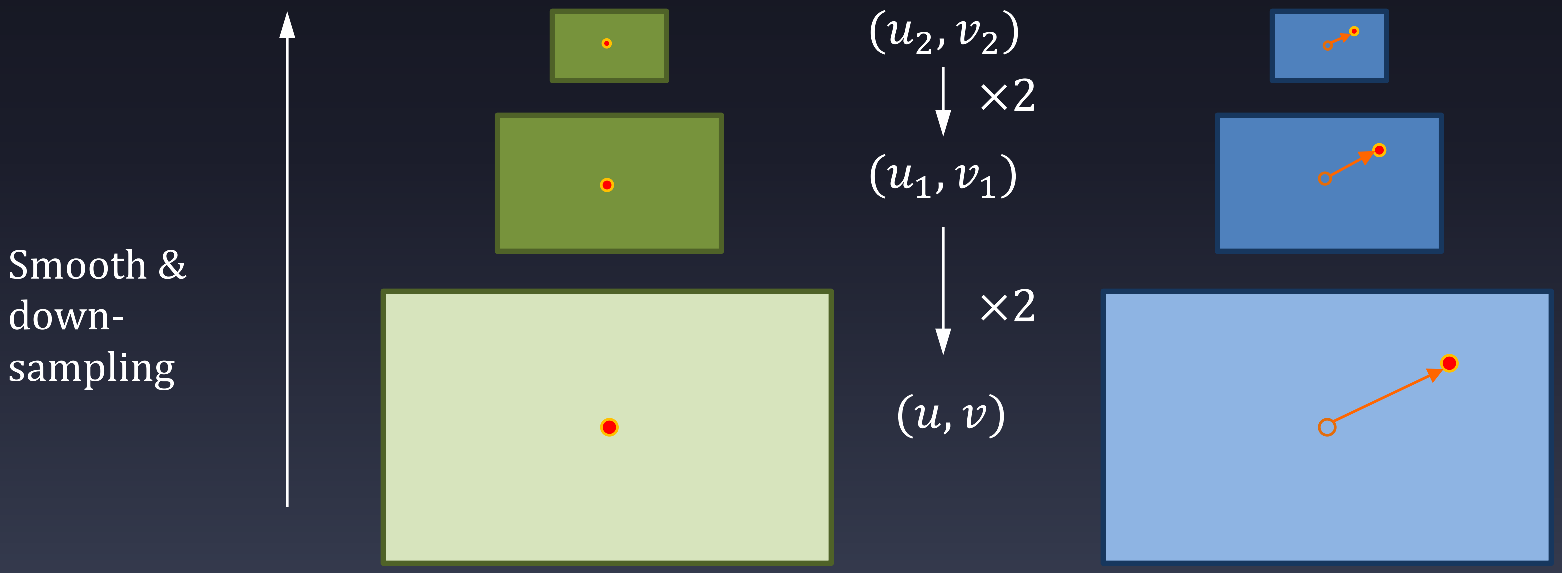
Translation

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

$$\begin{aligned} \Delta\text{p}^* &= -(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{I}_t \\ &= -\begin{bmatrix} \mathbf{I}_x^T\mathbf{I}_x & \mathbf{I}_x^T\mathbf{I}_y \\ \mathbf{I}_x^T\mathbf{I}_y & \mathbf{I}_y^T\mathbf{I}_y \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{I}_x^T\mathbf{I}_t \\ \mathbf{I}_y^T\mathbf{I}_t \end{bmatrix} \end{aligned}$$

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



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)

$$
\begin{aligned}
&I_2(x+u+du, y+v+dv) - I_1(x,y) \\
&= I_2(x+u, y+v) + I_x(x+u, y+v)du + I_y(x+u, y+v)dv - I_1(x,y)
\end{aligned}
$$

$$I_x du + I_y dv + I_t = 0$$

- We obtain the following function within a patch

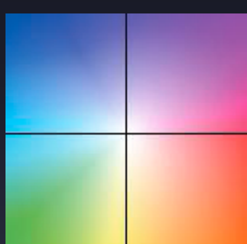
$$\begin{bmatrix} du \\ dv \end{bmatrix} = -\begin{bmatrix} \mathbf{I}_x^T\mathbf{I}_x & \mathbf{I}_x^T\mathbf{I}_y \\ \mathbf{I}_x^T\mathbf{I}_y & \mathbf{I}_y^T\mathbf{I}_y \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{I}_x^T\mathbf{I}_t \\ \mathbf{I}_y^T\mathbf{I}_t \end{bmatrix}$$

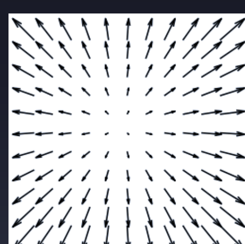
- The flow vector of each pixel is updated independently
- Median filtering can be applied for spatial smoothness

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)

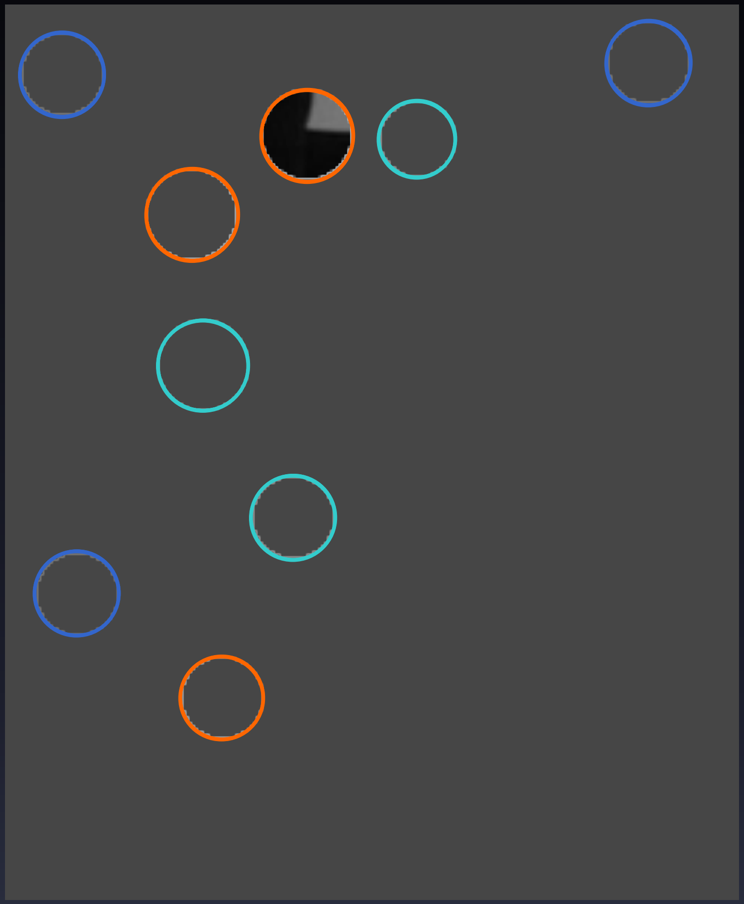
Motion ambiguities

- When will the Lucas-Kanade algorithm fail?

$$\begin{bmatrix} du \\ dv \end{bmatrix} = -\begin{bmatrix} \mathbf{I}_x^T\mathbf{I}_x & \mathbf{I}_x^T\mathbf{I}_y \\ \mathbf{I}_x^T\mathbf{I}_y & \mathbf{I}_y^T\mathbf{I}_y \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{I}_x^T\mathbf{I}_t \\ \mathbf{I}_y^T\mathbf{I}_x \end{bmatrix}$$

- The inverse may not exist!!!
- How?
  - All the derivatives are zero: *flat regions*
  - X- and y-derivatives are linearly correlated: *lines*

Aperture problem



Corners

Lines

Flat regions

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

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

  - $|\nabla u|^2 = \left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 = u_x^2 + u_y^2$
  - α: smoothness coefficient

2D Euler Lagrange

- 2D Euler Lagrange: the functional

$$S = \iint_{\Omega} L(x, y, f, f_x, f_y)\, dx dy$$

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
  - $L(u, v, u_x, u_y, v_x, v_y) = (I_x u + I_y v + I_t)^2 + \alpha(u_x^2 + u_y^2 + v_x^2 + v_y^2)$
  - $\frac{\partial L}{\partial u} = 2(I_x u + I_y v + I_t) I_x$
  - $\frac{\partial L}{\partial u_x} = 2\alpha u_x, \frac{\partial}{\partial x}\frac{\partial L}{\partial u_x} = 2\alpha u_{xx}, \frac{\partial L}{\partial u_y} = 2\alpha u_y, \frac{\partial}{\partial y}\frac{\partial L}{\partial u_y} = 2\alpha u_{yy}$

Linear PDE

- The Euler-Lagrange PDE for Horn-Schunck is

$$\begin{cases} (I_x u + I_y v + I_t) I_x - \alpha(u_{xx} + u_{yy}) = 0 \\ (I_x u + I_y v + I_t) I_y - \alpha(v_{xx} + v_{yy}) = 0 \end{cases}$$

- $u_{xx} + u_{yy}$ 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

$$\begin{bmatrix} \mathbf{I}_x^2 + \alpha \mathbf{L} & \mathbf{I}_x \mathbf{I}_y \\ \mathbf{I}_x \mathbf{I}_y & \mathbf{I}_y^2 + \alpha \mathbf{L} \end{bmatrix} \begin{bmatrix} U \\ V \end{bmatrix} = - \begin{bmatrix} \mathbf{I}_x \mathbf{I_t} \\ \mathbf{I}_y \mathbf{I_t} \end{bmatrix}$$

How to solve a large linear system Ax=b?

$$\begin{bmatrix} \mathbf{I}_x^2 + \alpha \mathbf{L} & \mathbf{I}_x \mathbf{I}_y \\ \mathbf{I}_x \mathbf{I}_y & \mathbf{I}_y^2 + \alpha \mathbf{L} \end{bmatrix} \begin{bmatrix} U \\ V \end{bmatrix} = - \begin{bmatrix} \mathbf{I}_x \mathbf{I_t} \\ \mathbf{I}_y \mathbf{I_t} \end{bmatrix}$$

- 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

$$(u, v) = \arg\min \iint \left(I_x u + I_y v + I_t\right)^2 + \alpha(|\nabla u|^2 + |\nabla v|^2) dxdy$$

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

- More practically, we can estimate the optimal incremental change

$$\iint \left(I_x du + I_y dv + I_t\right)^2 + \alpha(|\nabla(u + du)|^2 + |\nabla(v + dv)|^2) dxdy$$

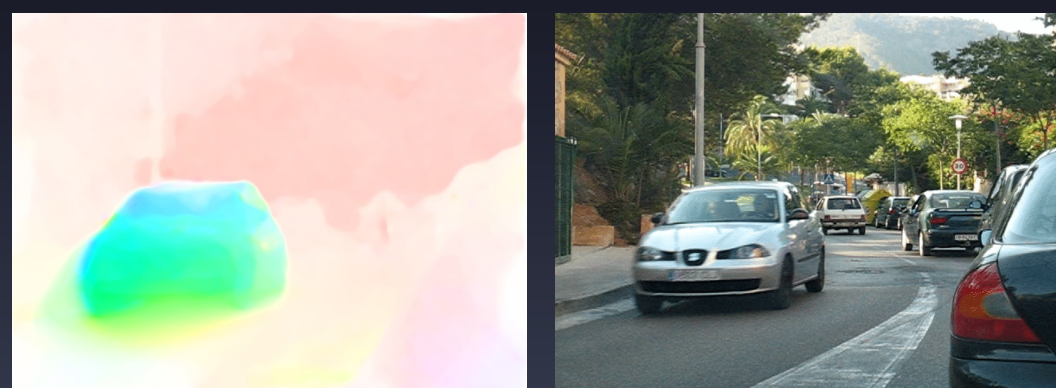
- The solution becomes

$$\begin{bmatrix} \mathbf{I}_x^2 + \alpha \mathbf{L} & \mathbf{I}_x \mathbf{I}_y \\ \mathbf{I}_x \mathbf{I}_y & \mathbf{I}_y^2 + \alpha \mathbf{L} \end{bmatrix} \begin{bmatrix} dU \\ dV \end{bmatrix} = - \begin{bmatrix} \mathbf{I}_x \mathbf{I}_t + \alpha \mathbf{L} U \\ \mathbf{I}_y \mathbf{I}_t + \alpha \mathbf{L} V \end{bmatrix}$$

Example

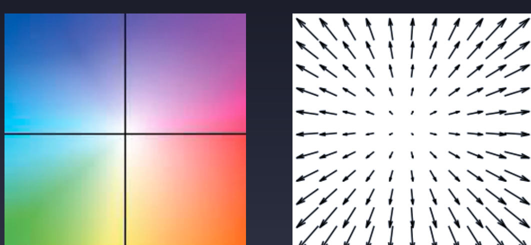
Horn-Schunck

Coarse-to-fine LK



Coarse-to-fine LK with median filtering

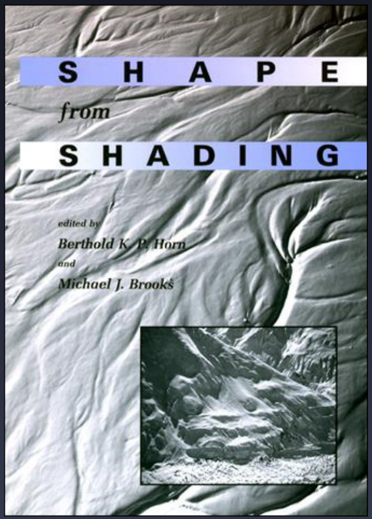
Input two frames



Flow visualization

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

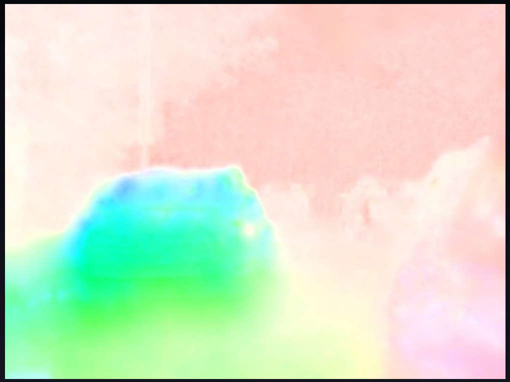
- Motion perception
- Motion representation
- Parametric motion: Lucas-Kanade
- Dense optical flow: Horn-Schunck
- Robust estimation
- Applications (1)

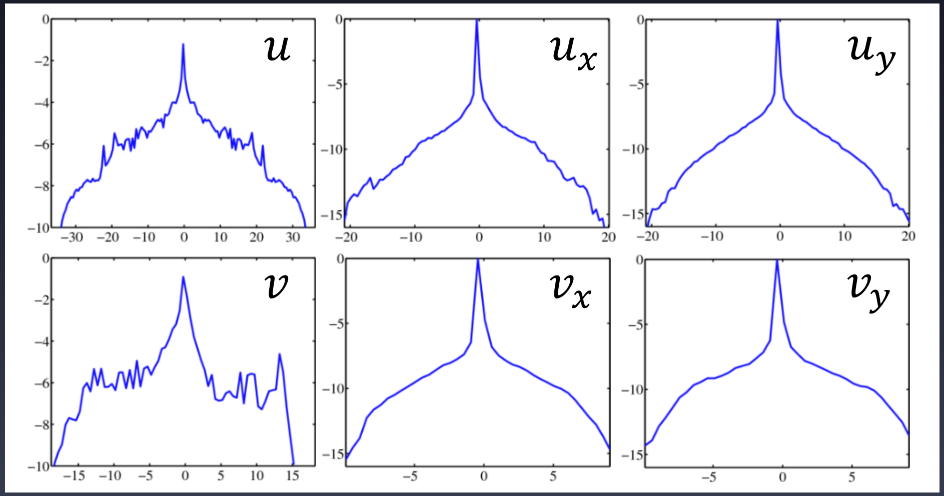
Spatial regularity

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

$$\iint (I_x u + I_y v + I_t)^2 + \alpha(|\nabla u|^2 + |\nabla v|^2)dxdy$$

- 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)

$$\iint \left(I_x u + I_y v + I_t\right)^2 + \alpha(|\nabla u|^2 + |\nabla v|^2)dxdy$$

- 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

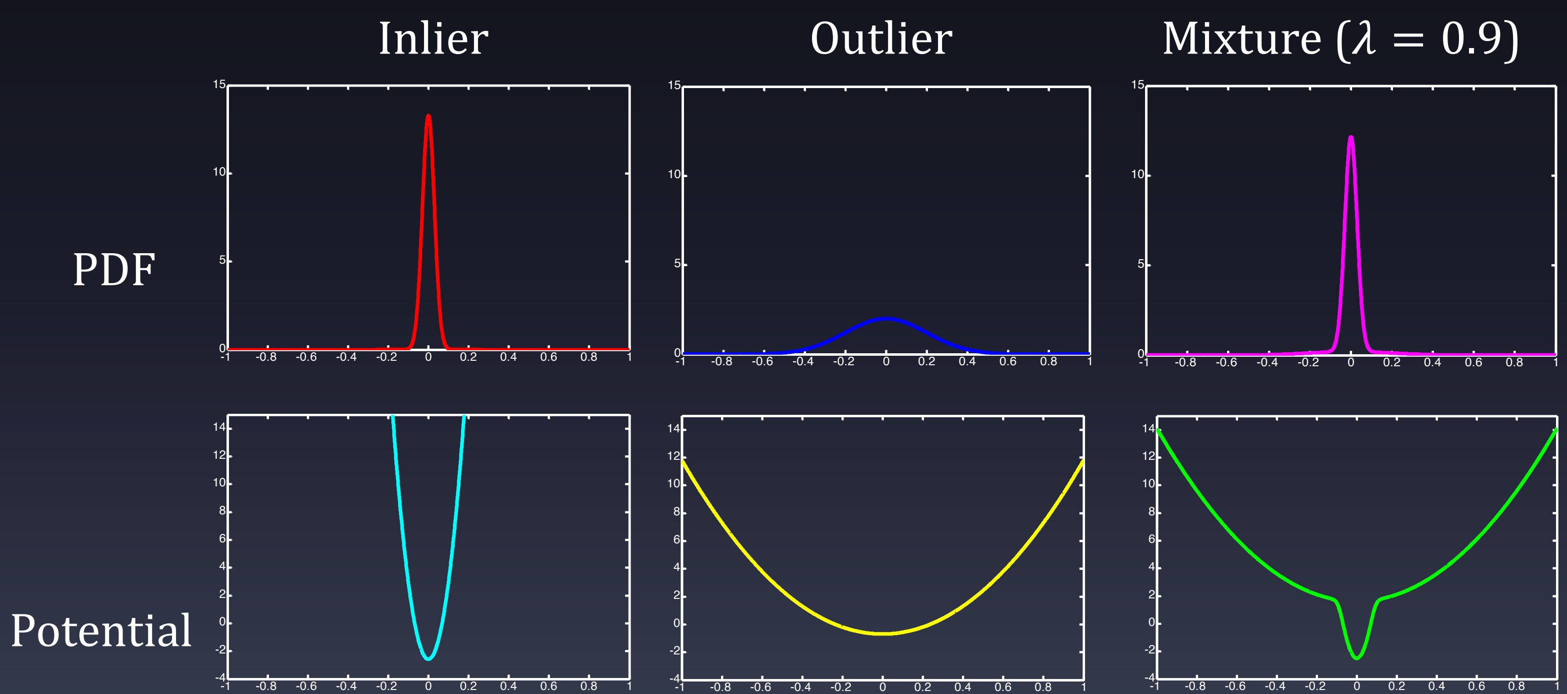
Noise model

- 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*

$$n \sim \lambda N(0, \sigma_i^2) + (1-\lambda) N(0, \sigma_o^2)$$

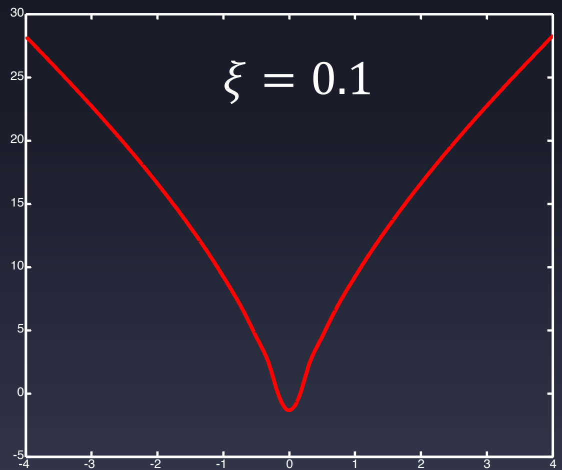


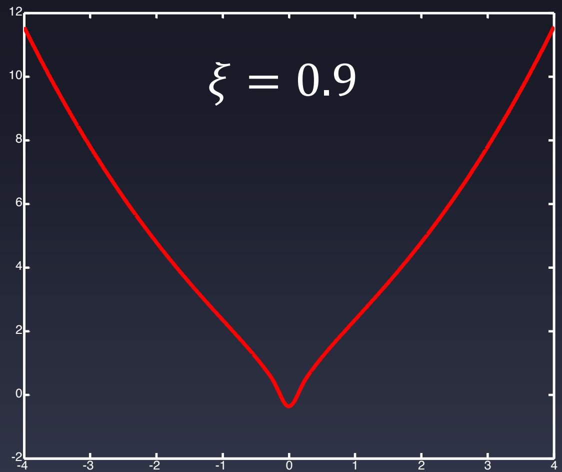
More components in the mixture

- Consider a Gaussian mixture model

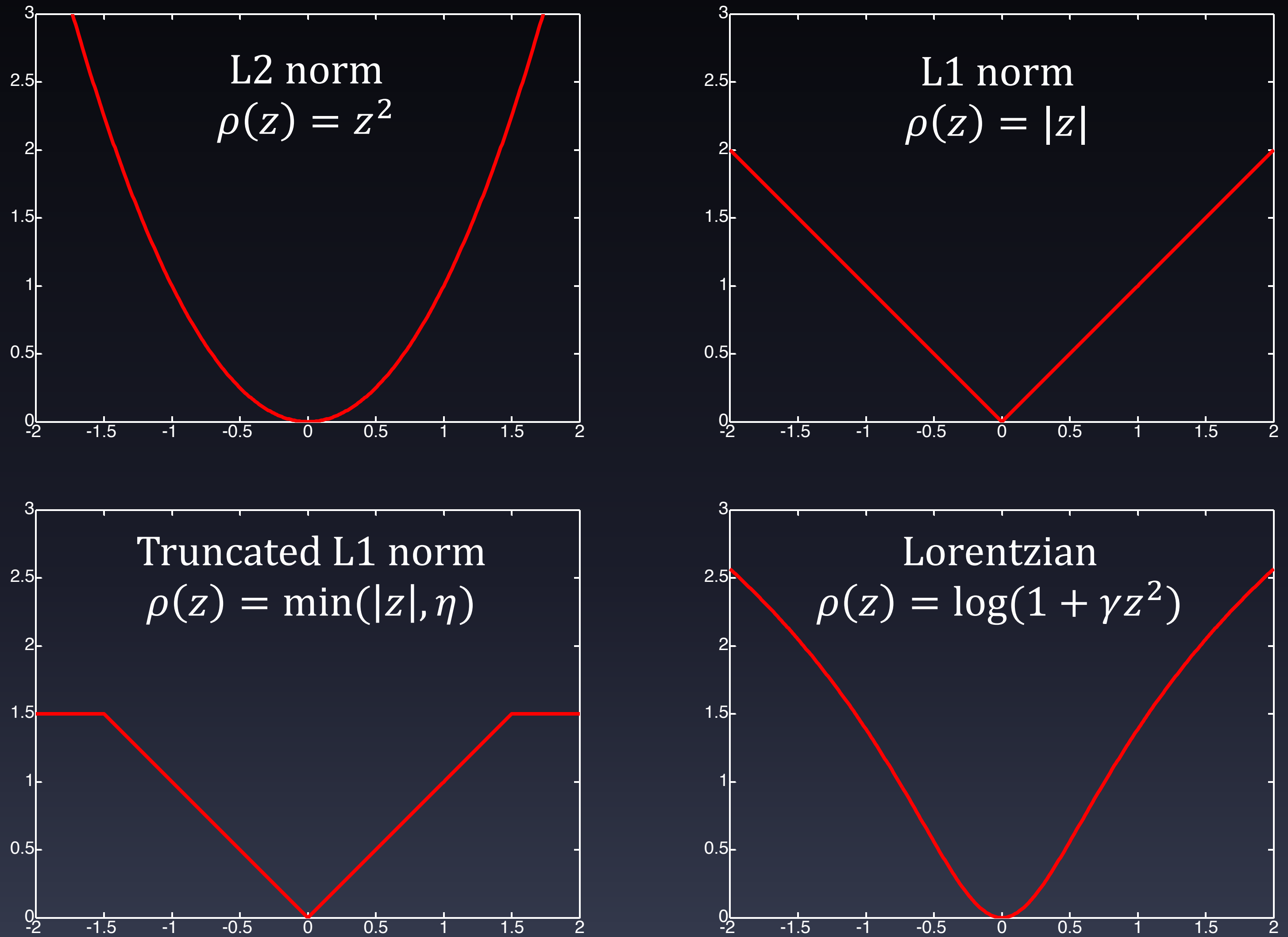
$$n \sim \frac{1}{Z} \sum_{k=1}^{K} \xi^k N(0, (ks)^2)$$

- Varying the decaying rate ξ, we obtain a variety of potential functions





Typical error functions



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

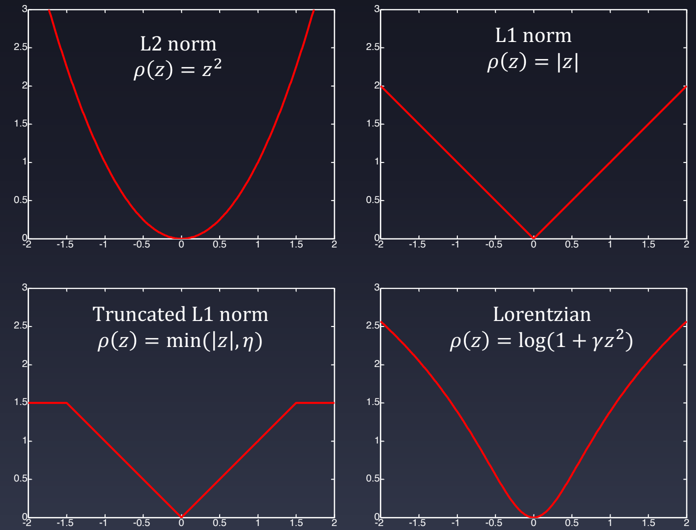
  $$z^* = \arg\min_z \sum_i \rho(z - z_i)$$

  – L2 norm: $z^* = 4.172$

  – L1 norm: $z^* = 1.038$

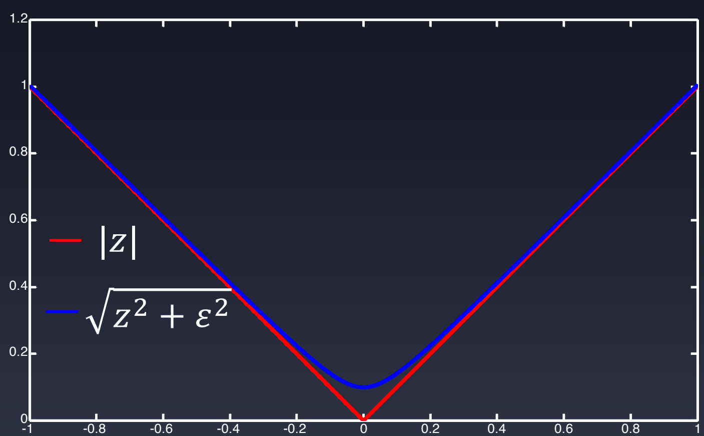
  – Truncated L1: $z^* = 1.0296$

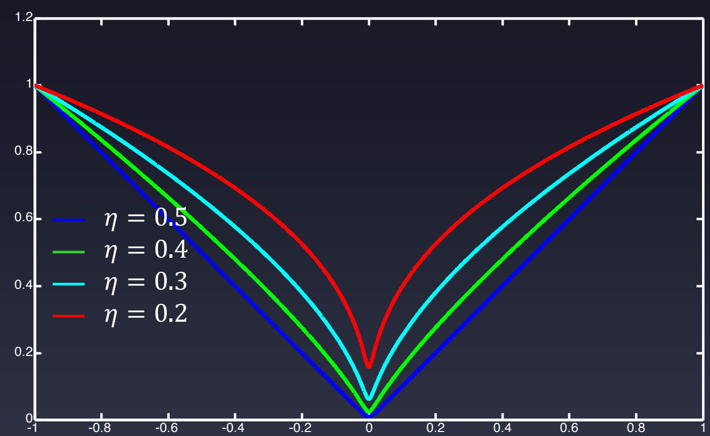
  – Lorentzian: $z^* = 1.0147$



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 $\mathrm{x} = (x, y, t)$, and $\mathrm{w}(\mathrm{x}) = (u(\mathrm{x}), v(\mathrm{x}), 1)$ be the flow vector
- Horn-Schunck (recall)

$$\iint \left(I_x u + I_y v + I_t\right)^2 + \alpha(|\nabla u|^2 + |\nabla v|^2) dxdy$$

- Robust estimation

$$\iint \psi(|I(\mathrm{x}+\mathrm{w}) - I(\mathrm{x})|^2) + \alpha\phi(|\nabla u|^2 + |\nabla v|^2) dxdy$$

- Robust estimation with Lucas-Kanade

$$\iint g * \psi(|I(\mathrm{x}+\mathrm{w}) - I(\mathrm{x})|^2) + \alpha\phi(|\nabla u|^2 + |\nabla v|^2) dxdy$$

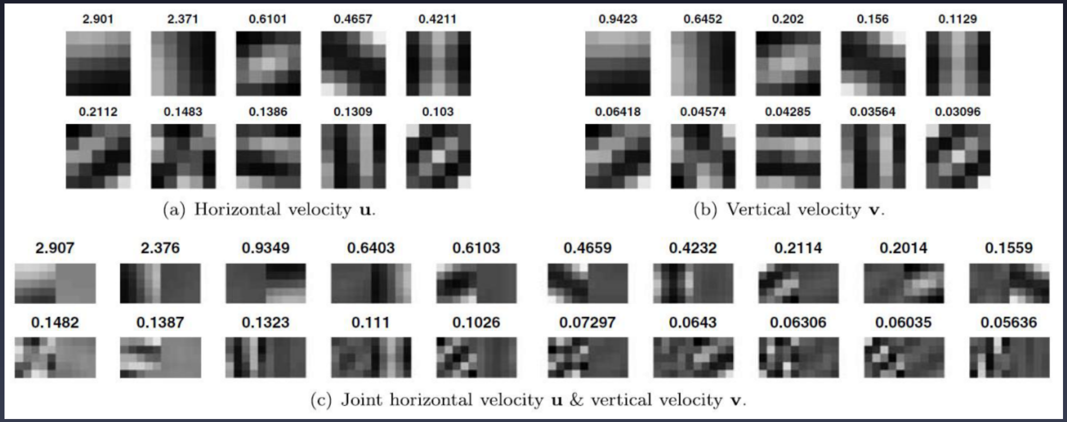
A unifying framework

- The robust object function

$$\iint g * \psi(|I(\mathrm{x}+\mathrm{w}) - I(\mathrm{x})|^2) + \alpha\phi(|\nabla u|^2 + |\nabla v|^2)dxdy$$

  - 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]



(a) Horizontal velocity **u**. (b) Vertical velocity **v**.

(c) Joint horizontal velocity **u** & vertical velocity **v**.

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

Iteratively reweighted least square (IRLS)

- Let $\phi(z^2) = (z^2 + \varepsilon^2)^\eta$ be a robust function
- We want to minimize the objective function

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

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

$$\begin{aligned}\frac{\partial \Phi}{\partial x} &\propto \sum\nolimits_{i=1}^{n} \phi'\left(\left(a_i^T x + b_i\right)^2\right)\left(a_i^T x + b_i\right)a_i \\ &= \sum\nolimits_{i=1}^{n} w_{ii} a_i^T x a_i + w_{ii} b_i a_i \\ &= \sum\nolimits_{i=1}^{n} a_i^T w_{ii} x a_i + b_i w_{ii} a_i \\ &= \mathbf{A}^T \mathbf{W} \mathbf{A} x + \mathbf{A}^T \mathbf{W} b\end{aligned}$$

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

$$\mathbf{W} = \mathrm{diag}(\Phi'(\mathbf{A}x + b))$$

Iteratively reweighted least square (IRLS)

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

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

- Convergence is guaranteed (local minima)

IRLS for robust optical flow

- Objective function

$$\iint g * \psi(|I(\mathrm{x}+\mathrm{w}) - I(\mathrm{x})|^2) + \alpha\phi(|\nabla u|^2 + |\nabla v|^2)dxdy$$

- Discretize, linearize and increment

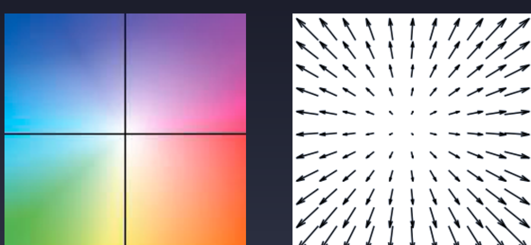
$$\sum_{x,y} g * \psi\left(\left|I_t + I_x du + I_y dv\right|^2\right) + \alpha\phi(|\nabla(u+du)|^2 + |\nabla(v+dv)|^2)$$

- IRLS (initialize $du = dv = 0$)
  - Reweight: $\mathbf{\Psi}'_{xx} = \mathrm{diag}(g * \psi' \mathbf{I}_x \mathbf{I}_x), \mathbf{\Psi}'_{xy} = \mathrm{diag}\big(g * \psi' \mathbf{I}_x \mathbf{I}_y\big),$
    $\mathbf{\Psi}'_{yy} = \mathrm{diag}\big(g * \psi' \mathbf{I}_y \mathbf{I}_y\big), \mathbf{\Psi}'_{xt} = \mathrm{diag}(g * \psi' \mathbf{I}_x \mathbf{I}_t),$
    $\mathbf{\Psi}'_{yt} = \mathrm{diag}\big(g * \psi' \mathbf{I}_y \mathbf{I}_t\big), \mathbf{L} = \mathbf{D}_x^T \mathbf{\Phi}' \mathbf{D}_x + \mathbf{D}_y^T \mathbf{\Phi}' \mathbf{D}_y$
  - Least square:

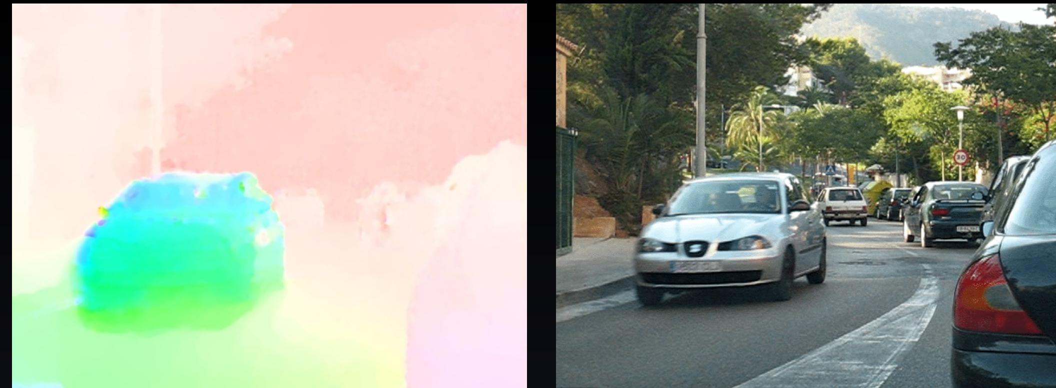
$$\begin{bmatrix} \mathbf{\Psi}'_{xx} + \alpha\mathbf{L} & \mathbf{\Psi}'_{xy} \\ \mathbf{\Psi}'_{xy} & \mathbf{\Psi}'_{yy} + \alpha\mathbf{L} \end{bmatrix} \begin{bmatrix} dU \\ dV \end{bmatrix} = -\begin{bmatrix} \mathbf{\Psi}'_{xt} + \alpha\mathbf{L}U \\ \mathbf{\Psi}'_{yt} + \alpha\mathbf{L}V \end{bmatrix}$$

Example

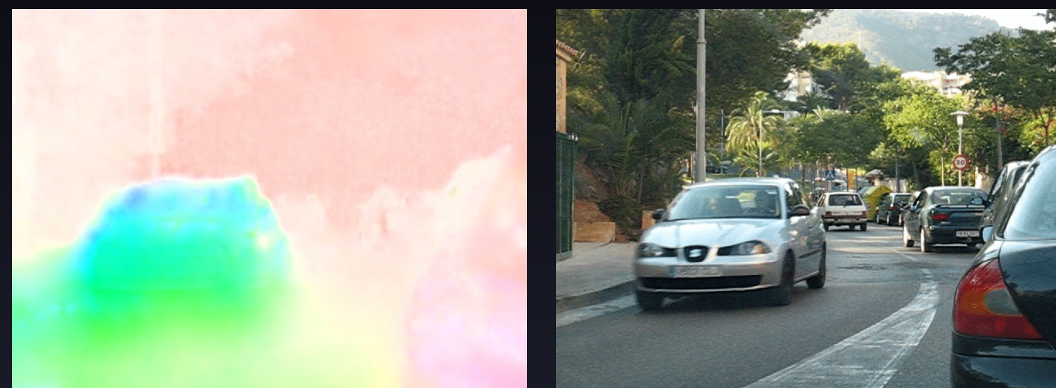
Input two frames



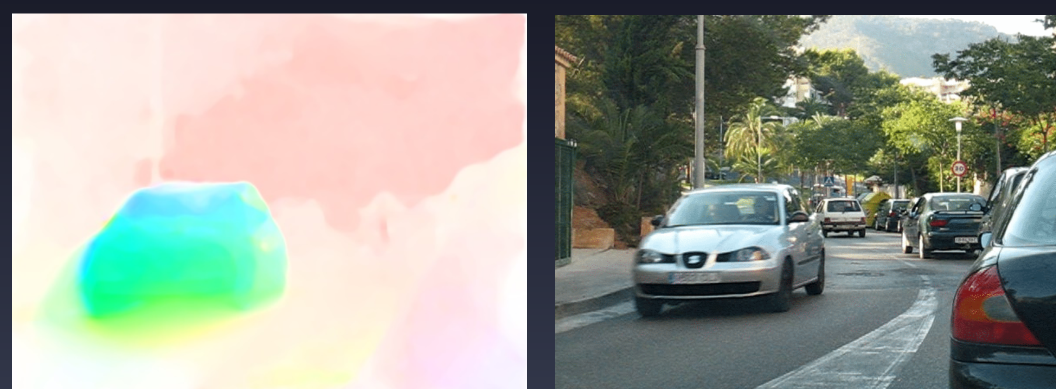
Flow visualization



Robust optical flow



Horn-Schunck



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

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
- Large motion
- Convolutional Neural Networks for flow estimation
- Applications (2)