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)

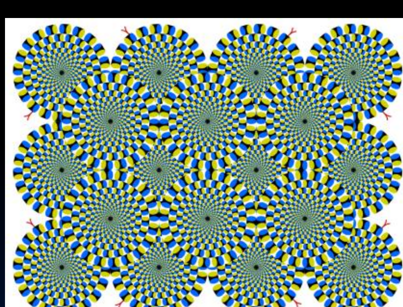
Readings

- Rick's book: Chapter 8
- Ce Liu's PhD thesis (Appendix A & B)
- 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

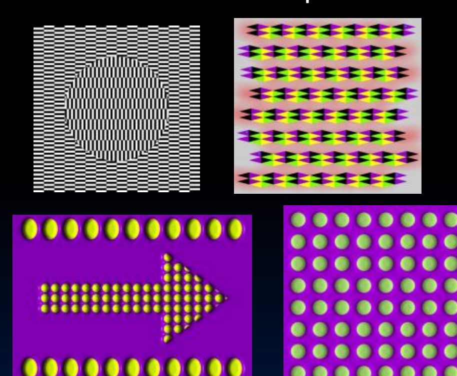
- Motion perception
- Motion representation
- Parametric motion: Lucas-Kanade
- Dense optical flow: Horn-Schunck
- Robust estimation
- Applications (1)

Seeing motion from a static picture?



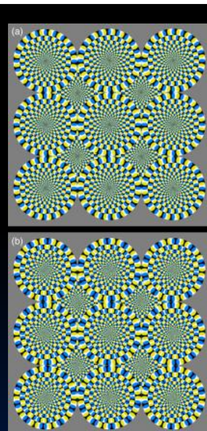
http://www.ritsumei.ac.jp/~akitaoka/index-e.html

More examples

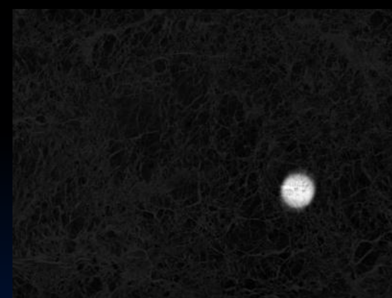


How is this possible?

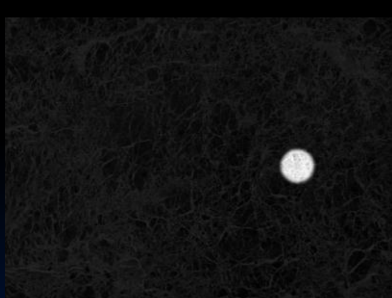
- The true mechanism is to be revealed
- FMRI data suggest that illusion is related to some component of eye movements
- We don't expect computer vision to "see" motion from these stimuli, yet



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

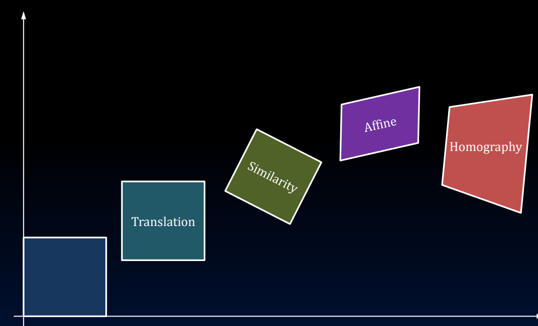
Contents

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

Parametric motion

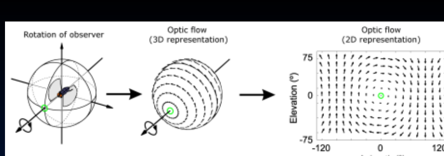
- Mapping: $(x_1, y_1) \rightarrow (x_2, y_2)$
  - (x_1, y_1): point in frame 1
  - (x_2, y_2): corresponding point in frame 2
- Global parametric motion: $(x_2, y_2) = f(x_1, y_1; \theta)$
- Forms of parametric motion
  - Translation: $\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 +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



Optical flow field

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

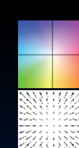


Optical flow field visualization

- 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]

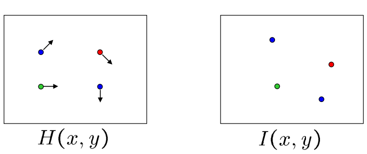
Matching criterion

- 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)
    $E(u,v) = \sum_{x,y} (I_1(x,y) - I_2(x+u, y+v))^2$
  - Correlation, normalized cross correlation (NCC)

Contents

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

Problem definition: optical flow



How to estimate pixel motion from image H to image I?

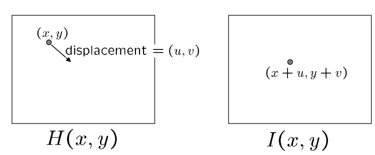
- Solve pixel correspondence problem
  - given a pixel in H, look for nearby pixels of the same color in I

Key assumptions

- **color constancy:** a point in H looks the same in I
  - For grayscale images, this is **brightness constancy**
- **small motion**: points do not move very far

This is called the **optical flow** problem

Optical flow constraints (grayscale images)



Let's look at these constraints more closely

- brightness constancy: Q: what's the equation?
- small motion: (u and v are less than 1 pixel)
  - suppose we take the Taylor series expansion of I:

$$I(x+u, y+v) = I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$
$$\approx I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

Optical flow equation

Combining these two equations

shorthand: $I_x = \frac{\partial I}{\partial x}$

Optical flow equation

Combining these two equations

shorthand: $I_x = \frac{\partial I}{\partial x}$

$$\begin{aligned} 0 &= I(x+u, y+v) - H(x,y) \\ &\approx I(x,y) + I_x u + I_y v - H(x,y) \\ &\approx (I(x,y) - H(x,y)) + I_x u + I_y v \\ &\approx I_t + I_x u + I_y v \\ &\approx I_t + \nabla I \cdot [u\ v] \end{aligned}$$

In the limit as u and v go to zero, this becomes exact

$$0 = I_t + \nabla I \cdot [\tfrac{\partial x}{\partial t}\ \tfrac{\partial y}{\partial t}]$$

Lucas-Kanade flow

How to get more equations for a pixel?

- Basic idea: impose additional constraints
  - most common is to assume that the flow field is smooth locally
  - one method: pretend the pixel's neighbors have the same (u,v)
    - » If we use a 5x5 window, that gives us 25 equations per pixel!

$$0 = I_t(\mathbf{p_i}) + \nabla I(\mathbf{p_i}) \cdot [u\ v]$$

$$\underset{25\times2}{\underbrace{\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix}}_{A}} \underset{2\times1}{\underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{d}} = - \underset{25\times1}{\underbrace{\begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}}_{b}}$$

RGB version

How to get more equations for a pixel?

- Basic idea: impose additional constraints
  - most common is to assume that the flow field is smooth locally
  - one method: pretend the pixel's neighbors have the same (u,v)
    - » If we use a 5x5 window, that gives us 25*3 equations per pixel!

$$0 = I_t(\mathbf{p_i})[0,1,2] + \nabla I(\mathbf{p_i})[0,1,2] \cdot [u\ v]$$

$$\underset{75\times2}{\underbrace{\begin{bmatrix} I_x(\mathbf{p_1})[0] & I_y(\mathbf{p_1})[0] \\ I_x(\mathbf{p_1})[1] & I_y(\mathbf{p_1})[1] \\ I_x(\mathbf{p_1})[2] & I_y(\mathbf{p_1})[2] \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}})[0] & I_y(\mathbf{p_{25}})[0] \\ I_x(\mathbf{p_{25}})[1] & I_y(\mathbf{p_{25}})[1] \\ I_x(\mathbf{p_{25}})[2] & I_y(\mathbf{p_{25}})[2] \end{bmatrix}}_{A}} \underset{2\times1}{\underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{d}} = - \underset{75\times1}{\underbrace{\begin{bmatrix} I_t(\mathbf{p_1})[0] \\ I_t(\mathbf{p_1})[1] \\ I_t(\mathbf{p_1})[2] \\ \vdots \\ I_t(\mathbf{p_{25}})[0] \\ I_t(\mathbf{p_{25}})[1] \\ I_t(\mathbf{p_{25}})[2] \end{bmatrix}}_{b}}$$

Lucas-Kanade flow

Prob: we have more equations than unknowns

$$\underset{25\times2}{A}\ \underset{2\times1}{d} = \underset{25\times1}{b} \longrightarrow \text{minimize } \|Ad - b\|^2$$

Solution: solve least squares problem

- minimum least squares solution given by solution (in d) of:

$$\underset{2\times2}{(A^T A)}\ \underset{2\times1}{d} = \underset{2\times1}{A^T b}$$

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

- The summations are over all pixels in the K x K window
- This technique was first proposed by Lucas & Kanade (1981)

Conditions for solvability

- Optimal (u, v) satisfies Lucas-Kanade equation

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

When is This Solvable?

- $\mathbf{A^T A}$ should be invertible
- $\mathbf{A^T A}$ should not be too small due to noise
  – eigenvalues λ_1 and λ_2 of $\mathbf{A^T A}$ should not be too small
- $\mathbf{A^T A}$ should be well-conditioned
  – λ_1 / λ_2 should not be too large (λ_1 = larger eigenvalue)

Does this look familiar?

- $\mathbf{A^T A}$ is the Harris matrix

Observation

This is a two image problem BUT

- Can measure sensitivity by just looking at one of the images!
- This tells us which pixels are easy to track, which are hard
  – very useful for feature tracking...

Errors in Lucas-Kanade

What are the potential causes of errors in this procedure?

- Suppose $A^T A$ is easily invertible
- Suppose there is not much noise in the image

When our assumptions are violated

- Brightness constancy is **not** satisfied
- The motion is **not** small
- A point does **not** move like its neighbors
  – window size is too large
  – what is the ideal window size?

Improving accuracy

Recall our small motion assumption

$$0 = I(x+u, y+v) - H(x,y)$$
$$\approx I(x,y) + I_x u + I_y v - H(x,y)$$

This is not exact

- To do better, we need to add higher order terms back in:

$$= I(x,y) + I_x u + I_y v + \text{higher order terms} - H(x,y)$$

This is a polynomial root finding problem

- Can solve using **Newton's method**
  - Also known as **Newton-Raphson** method
  - For more on Newton-Raphson, see (first four pages)
    - » http://www.ulib.org/webRoot/Books/Numerical_Recipes/bookcpdf/c9-4.pdf
- Lucas-Kanade method does one iteration of Newton's method
  - Better results are obtained via more iterations

1D case on board

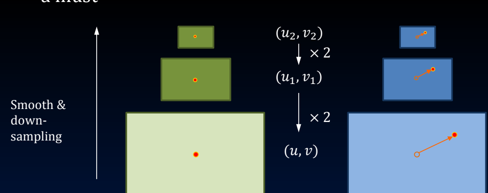
Iterative Refinement

Iterative Lucas-Kanade Algorithm

1. Estimate velocity at each pixel by solving Lucas-Kanade equations
2. Warp H towards I using the estimated flow field
   - use image warping techniques
3. Repeat until convergence

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



Example

Contents

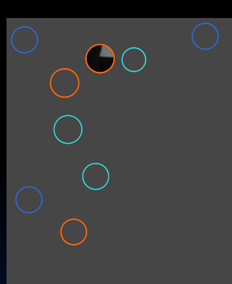
Motion ambiguities

- 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

Aperture problem

http://www.123opticalillusions.com/pages/barber_pole.php

Dense optical flow with spatial regularity

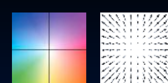
- Local motion is inherently ambiguous
  - *Corners*: definite, no ambiguity (but can be misleading)
  - *Lines*: definite along the normal, ambiguous along the tangent
  - *Flat regions*: totally ambiguous
- Solution: imposing spatial smoothness to the flow field
  - Adjacent pixels should move together as much as possible
- Horn & Schunck equation

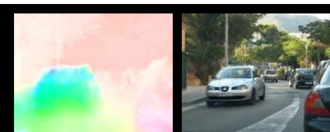
$$(u,v) = \arg\min \iint \left(I_x u + I_y v + I_t\right)^2 + \alpha\left(|\nabla u|^2 + |\nabla v|^2\right)dxdy$$

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

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

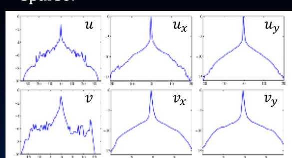
- Motion perception
- Motion representation
- Parametric motion: Lucas-Kanade
- Dense optical flow: Horn-Schunck
- Robust estimation
- Applications (1)

Spatial regularity

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

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

- Spatial over-smoothness is caused by the quadratic smoothness term
- Nevertheless, real optical flow fields are sparse!



Data term

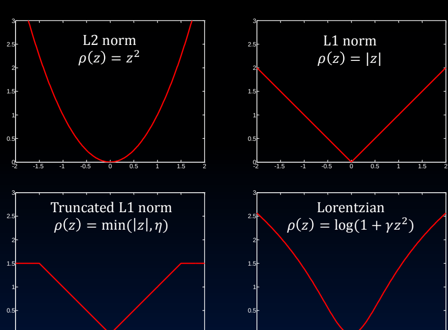
- Horn-Schunck is a Gaussian Markov random field (GMRF)

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

- 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

Typical error functions

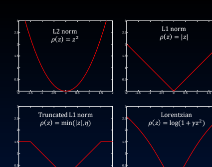


Robust statistics

- 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

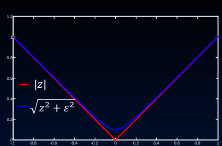
$$z^* = \arg\min_z \sum_i \rho(z - z_i)$$

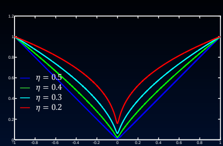
  - L2 norm: $z^* = 4.172$
  - L1 norm: $z^* = 1.038$
  - Truncated L1: $z^* = 1.0296$
  - Lorentzian: $z^* = 1.0147$



The family of robust power functions

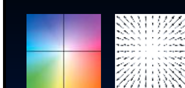
- Can we directly use L1 norm $\psi(z) = |z|$?
  - Derivative is not continuous
- Alternative forms
  - L1 norm: $\psi(z^2) = \sqrt{z^2 + \varepsilon^2}$
  - Sub L1: $\psi(z^2; \eta) = (z^2 + \varepsilon^2)^\eta, \eta < 0.5$



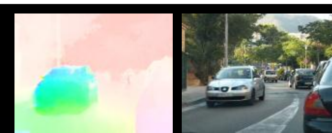


Example

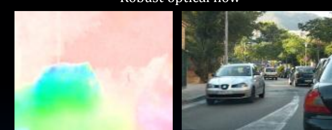
Input two frames



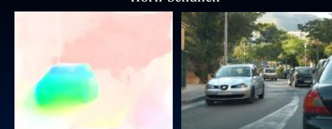
Flow visualization



Robust optical flow



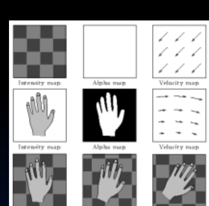
Horn-Schunck



Coarse-to-fine LK with median filtering

Layer representation

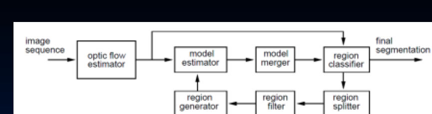
- Optical flow field is able to model complicated motion
- Different angle: a video sequence can be a composite of several moving layers
- Layers have been widely used
  - Adobe Photoshop
  - Adobe After Effect
- Compositing is straightforward, but inference is hard



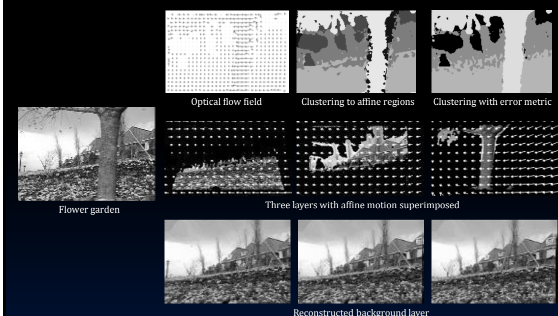
Wang & Adelson, 1994

Wang & Adelson, 1994

- Strategy
  - Obtaining dense optical flow field
  - Divide a frame into non-overlapping regions and fit affine motion for each region
  - Cluster affine motions by k-means clustering
  - Region assignment by hypothesis testing
  - Region splitter: disconnected regions are separated



Results

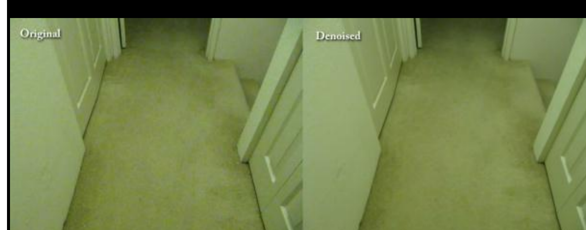


57

Contents

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