# Image and Video Analysis

Dmitrij Csetverikov

Eötvös Loránd University, Budapest, Hungary
csetverikov@inf.elte.hu
csetverikov@sztaki.hu

Faculty of Informatics

# Motion analysis

# Outline

1. Optical flow
   - Basics of optical flow and motion tracking
   - Examples of optical flow
2. Motion tracking
   - Examples of Kanade-Lucas-Tomasi (KLT) tracking
3. Refinement of basic methods
   - Problems to solve
   - Subpixel and multiresolution methods
   - Handling affine distortion and illumination variations

# Optical flow and motion tracking

- **Optical flow** (OF)
  - perceived displacements of pixels between two frames
  - if possible, for all pixels $\rightarrow$ **(dense) optical flow**
  - small time interval $\longrightarrow$ small displacements
- **Motion tracking**
  - tracking feature points in two or more frames
  - for feature points $\rightarrow$ **sparse flow**
  - in principle, displacements can be large
  - we consider *small* displacements only
- Motion models
  - shift without distortion
  - shift with affine distortion

# Notions

- $I(\mathbf{x}(t), t)$: intensity (image value) in point $\mathbf{x}$ at time $t$
  - for simplicity: $I(\mathbf{x}, t)$ or $I$
- $\Delta\mathbf{x} = \mathbf{u}dt$: displacement with velocity vector $\mathbf{u}$ during $dt$
- $I(\mathbf{x}(t) + \mathbf{u}dt, t + dt)$: intensity in shifted point at time $t + dt$
- $\boldsymbol{\nabla} = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right)$: gradient operator (vector)
  - e.g., image gradient: $\boldsymbol{\nabla} I(\mathbf{x}, t) = \left(\frac{\partial I(\mathbf{x},t)}{\partial x}, \frac{\partial I(\mathbf{x},t)}{\partial y}\right) = (I_x, I_y)$
- Local structure matrix (tensor):

$$\mathbf{M} = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

# Optical flow equation

- **Intensity constancy**: basic assumption

$$I(\mathbf{x}(t), t) = I(\mathbf{x}(t) + \mathbf{u}dt, t + dt) \longrightarrow \frac{dI(x(t), y(t), t)}{dt} = 0$$

- This leads to **optical flow equation** (constraint):

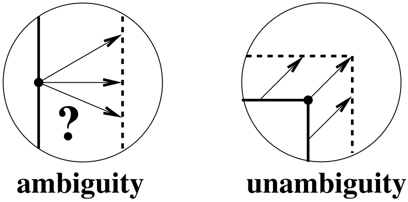
$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0$$

- In other form:

$$\mathbf{u}\nabla I + I_t = 0$$

- Two unknowns: two components of velocity vector **u**
  $\rightarrow$ underdetermined system: needs more constraints

# Aperture problem




- Motion vectors are **ambiguous** at edge
  - locally, normal component can only be determined
  - tangential component cannot be determined
- **Normal flow** in direction of gradient:

$$\mathbf{u}_n \doteq \frac{\mathbf{u}\nabla I}{\|\nabla I\|}\frac{\nabla I}{\|\nabla I\|} = -\frac{I_t}{\|\nabla I\|}\frac{\nabla I}{\|\nabla I\|}$$

- Motion vectors are **unambiguous** at corner

# Computing optical flow 1/2

- Consider window $W(\mathbf{x})$ around point $\mathbf{x}$
  - $\mathbf{x}' \in W(\mathbf{x})$: local coordinates in window
- Integrate constraints in $W(\mathbf{x})$
  - assume uniform motion of points in $W(\mathbf{x})$

  $\rightarrow$ search for **u** that best fits constraints
- Error function in window (IC = Intensity Constancy):

$$E_{IC}(\mathbf{u}) = \sum_{\mathbf{x}' \in W(\mathbf{x})} \left[\mathbf{u} \nabla I(\mathbf{x}', t) + I_t(\mathbf{x}', t)\right]^2$$

- Linear estimation for least squares
  - partial derivative $\nabla E_{IC}(\mathbf{u}) = 0$
  - **linear least squares**, LLS

# Computing optical flow 2/2

- Result of derivation:

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \mathbf{u} + \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix} = 0$$

- In matrix form $\hat{\mathbf{M}}$ with integrated structure matrix

$$\hat{\mathbf{M}}\mathbf{u} + \mathbf{b} = 0$$

- Estimation of velocity (displacement):

$$\mathbf{u} = -\hat{\mathbf{M}}^{-1}\mathbf{b}$$

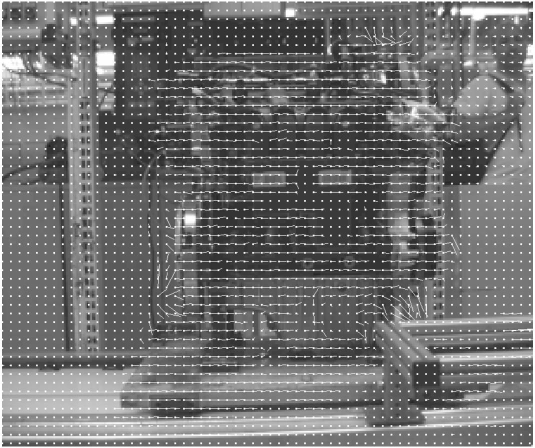
- Does not work if $\hat{\mathbf{M}}$ is not invertible
  - if window is **not textured** enough
  - e.g., if $I_x = 0$ or $I_y = 0 \longrightarrow \det \hat{\mathbf{M}} = 0$

  $\rightarrow$ aperture problem

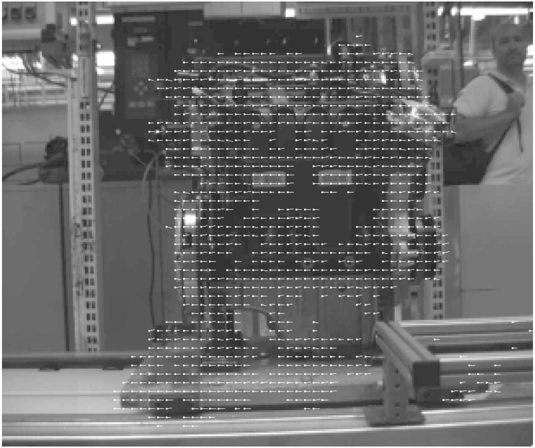
# Outline

1. Optical flow
   - Basics of optical flow and motion tracking
   - Examples of optical flow
2. Motion tracking
   - Examples of Kanade-Lucas-Tomasi (KLT) tracking
3. Refinement of basic methods
   - Problems to solve
   - Subpixel and multiresolution methods
   - Handling affine distortion and illumination variations

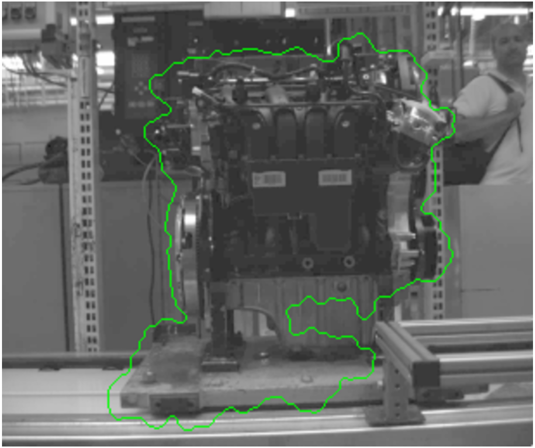
# Example of optical flow: original flow



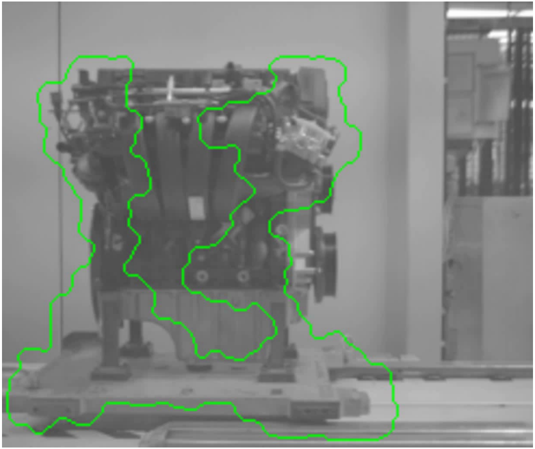
# Static and incoherently moving points removed



# Motor segmentation by optical flow

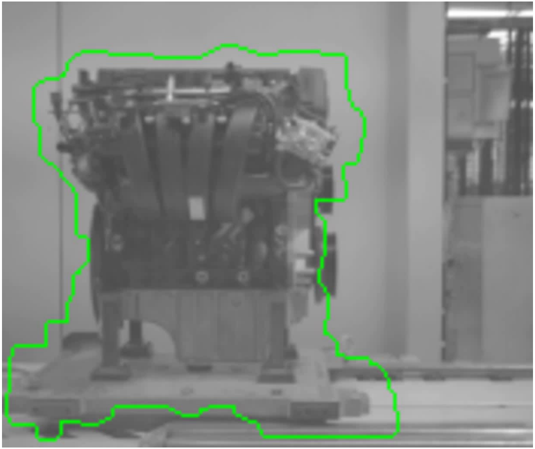


# Motor segmentation by optical flow: video 1



Too large velocity and error in beginning of sequence

# Motor segmentation by optical flow: video 2



Correct segmentation at decreased resolution

# Motion tracking by window matching

- Motion tracking in distinct feature points
  - well-detectable, stable feature points
  - displacement is unambiguous: no aperture problem
  - $\rightarrow$ characteristic neighborhood
- Difference between flow and tracking
  - flow: intensity constancy
  - tracking: window matching
- Tracking window $W$
  - $t, \mathbf{x}$: $W$
  - $t + dt, \mathbf{x} + \Delta\mathbf{x}$: window most similar to $W$ in vicinity of $\mathbf{x}$
- Error function of window matching (SSD):

$$E_{SSD}(\Delta\mathbf{x}) = \sum_{W(\mathbf{x})} \left[ I(\mathbf{x}' + \Delta\mathbf{x}, t + dt) - I(\mathbf{x}', t) \right]^2$$

# Comparison of error functions

- Optical flow: intensity constancy error

$$E_{IC}(\mathbf{u}) = \sum_{\mathbf{x}' \in W(\mathbf{x})} \left[\mathbf{u}\nabla I(\mathbf{x}', t) + I_t(\mathbf{x}', t)\right]^2$$

- Motion tracking: matching error

$$E_{SSD}(\Delta\mathbf{x}) = \sum_{W(\mathbf{x})} \left[I(\mathbf{x}' + \Delta\mathbf{x}, t + dt) - I(\mathbf{x}', t)\right]^2$$

  - no derivation
  - search for minimum in displacement range

- Easy to see that
  - $\mathbf{u}dt = (-\hat{\mathbf{M}}^{-1}\mathbf{b})dt \approx \Delta\mathbf{x}$

  $\rightarrow$ first order approximation of tracking displacement

# Simplified solution for flow and tracking

- Velocity/displacement estimation:

$$\mathbf{u} = \Delta\mathbf{x} = -\hat{\mathbf{M}}^{-1}\mathbf{b}$$

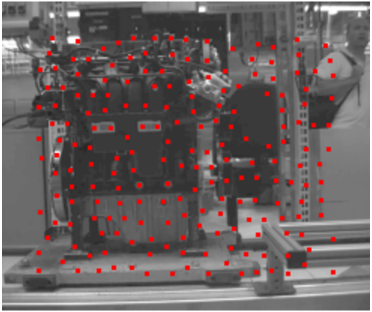
$$\hat{\mathbf{M}} = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

$$\mathbf{b} = \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}, \quad I_t \approx I_1 - I_0$$

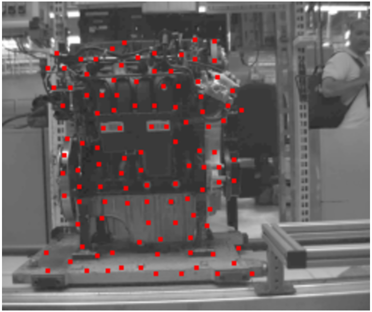
- **Motion tracking**: where $\hat{\mathbf{M}}$ is invertible ($\det \hat{\mathbf{M}} \neq 0$)
  - displacement in point $\mathbf{x}$ at time (frame) $t$
  - repeat **in point** $\mathbf{x} + \Delta\mathbf{x}$ at time $t + 1$
- **Optical flow**: in all image points
  - $\hat{\mathbf{M}}$ invertible $\longrightarrow \mathbf{u}$, $\quad \hat{\mathbf{M}}$ not invertible $\longrightarrow \mathbf{u} = 0$
  - displacement in point $\mathbf{x}$ at time (frame) $t$
  - repeat **in point** $\mathbf{x}$ at time $t + 1$

# Outline

# Tracking motors: no replacement of lost points



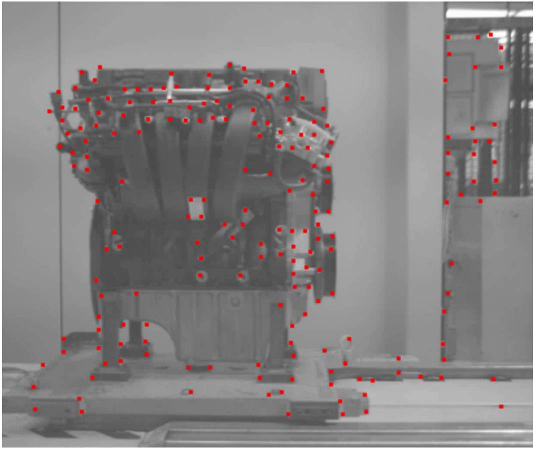
initial frame 1



frame 3

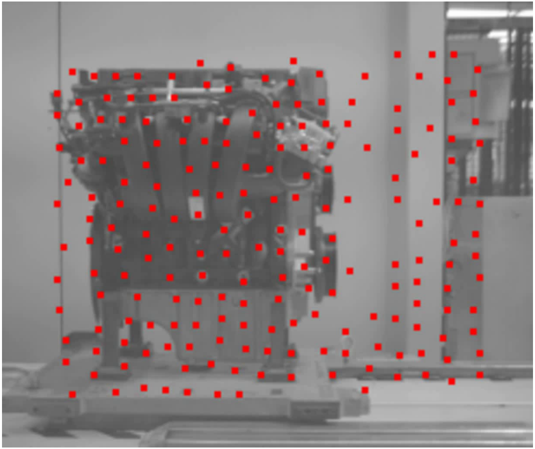
- frame 1 before filtering
  - static points
  - points with erroneous motion (e.g., not motor)
- frame 3 after filtering

# Motion tracking of motors: video 1



Too large velocity at beginning of sequence: points lost

# Motion tracking of motors: video 2



Good tracking at decreased resolution

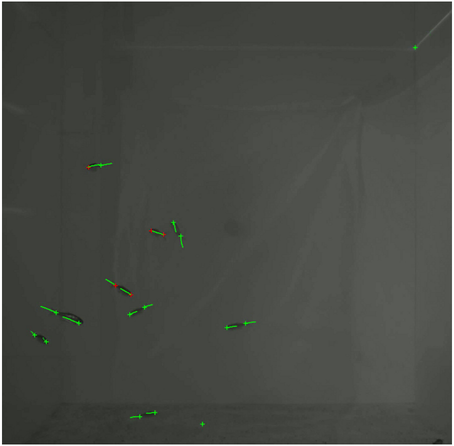
# Fishes in aquarium 1: original video

Pre-processing needed because of bad image quality

# Fishes in aquarium 1: pre-processed video

Person passing aquarium frightens fishes

# Motion tracking with replacement of lost points 1

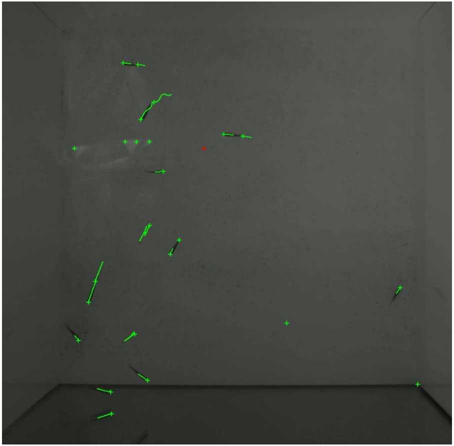


Both ends of fish tracked. Red cross: lost and replaced point

# Fishes in aquarium 2: original video

Better image quality, no pre-processing needed

# Motion tracking with replacement of lost points 2



Both ends of fish tracked. Red cross: lost and replaced point

# Drone motion tracking: input video

Strongly moving camera, poor image quality and visibility

# Drone motion tracking: tracked points

5 strongest points tracked, lost points replaced

# Drone motion tracking: trajectories

Camera moves, trajectories show relative motion

# Outline

1. Optical flow
   - Basics of optical flow and motion tracking
   - Examples of optical flow

2. Motion tracking
   - Examples of Kanade-Lucas-Tomasi (KLT) tracking

3. Refinement of basic methods
   - Problems to solve
   - Subpixel and multiresolution methods
   - Handling affine distortion and illumination variations

# Open questions 1/2

- In tracking, $\mathbf{x} + \Delta\mathbf{x}$ does not point at pixel $\longrightarrow$ rounding
  - small displacements $\longrightarrow$ **large relative error**
  - $\rightarrow$ subpixel solution needed
- Handling large displacements/velocities
  - linearisation of OF equations assumes small $\Delta\mathbf{x}$
  - $\rightarrow$ basic methods handle small displacements (max $2-3$ pix.)
  - $\rightarrow$ **multiresolution** solutions, image pyramids
  - $\rightarrow$ iterative solutions

# Open questions 2/2

- **Gradual distortion** of pattern in tracking by matching
  - typically, **affine distortion**
  - → affine matching
  - → motion tracking under affine distortion
- Optical flow under **varying lighting conditions**
  - explicitly: linear intensity variation
  - implicitly: e.g., by normalized cross-correlation
- Optical flow of **non-textured regions**
  - extension from textured regions
  - regularisation by **smoothness term**
  - → variational methods

# Outline

# Subpixel iterations

- Start from equation $\Delta \mathbf{x} = -\hat{\mathbf{M}}^{-1}\mathbf{b}$
- Refine by subpixel iteration:

$$\begin{aligned}
\delta_0 &= -\hat{\mathbf{M}}^{-1}\mathbf{e}^0 \\
\delta_{i+1} &= -\hat{\mathbf{M}}^{-1}\mathbf{e}^i \\
\Delta \mathbf{x}^{i+1} &\leftarrow \Delta \mathbf{x}^i + \delta_{i+1},
\end{aligned}$$

where:

$$\begin{aligned}
\mathbf{e}_0 &\doteq \mathbf{b} = \sum \begin{bmatrix} I_x I_t & I_y I_t \end{bmatrix}^T = \sum \nabla I(\mathbf{x}) I_t \\
\mathbf{e}^{i+1} &\doteq \sum \nabla I(\mathbf{x}', t) \left[ I(\mathbf{x}' + \Delta \mathbf{x}^i, t + dt) - I(\mathbf{x}', t) \right]
\end{aligned}$$

- $\mathbf{x}' + \Delta \mathbf{x}^i$ is not pixel position $\longrightarrow$ intensity interpolation

# Handling large displacements by image pyramids

- Build **Gaussian pyramid** for both frames
  - large displacements at initial, highest resolution
  - smaller displacements at decreased resolution
- Number of pyramid levels

  $\rightarrow$ max. displacement at lowest resolution: $2-3$ pixels
- **Top-down approach**
  - from top of pyramid to its bottom $\longrightarrow$ growing resolution

# Sketch of multiresolution methods

- Calculate displacement/velocity at **lowest resolution**
  - motion tracking: in stable, distinct feature points
  - optical flow: where only possible
- Proceed to **next level** (double resolution)
  - extend displacements to this level
  - compensate motion, work with small differences
  - $\rightarrow$ initial values for iterations
  - $\rightarrow$ iterative refinement
- Repeat until **bottom of pyramid**

# Outline

# Handling affine distortion

- Error function **A affine distortion**

$$E_{af}(\mathbf{A}, \Delta\mathbf{x}) = \sum_{W(\mathbf{x})} \left[ I(\mathbf{A}\mathbf{x}' + \Delta\mathbf{x}, t + dt) - I(\mathbf{x}', t) \right]^2$$

- Solution: $\mathbf{y} = -\mathbf{M}^{-1}\mathbf{b}$
  - $\mathbf{y} \doteq [d_{11}, d_{12}, d_{21}, d_{22}, d_1, d_2]^T$ (6D vector)
  - $[d_1, d_2]^T = \Delta\mathbf{x}$: 2D shift vector
  - elements of matrix $d_{ij}$, $\mathbf{D} \doteq \mathbf{A} - \mathbf{I}$: distortion (**I**: unit matrix)
  - **M** is now $6 \times 6$ matrix with elements

$$I_p I_q, \quad p I_p I_q, \quad pq I_p I_q, \qquad p, q = x, y$$

  - **b** is now 6D vector with elements

$$p I_t I_q$$

# Handling illumination variations

- **Linear intensity variation**: $\alpha I + \beta$
- Error function for **affine distortion A**:

$$E_{afli}(\mathbf{A}, \Delta\mathbf{x}, \alpha, \beta) = \sum_{W(\mathbf{x})} \left[ (\alpha I(\mathbf{A}\mathbf{x}' + \Delta\mathbf{x}, t + dt) + \beta) - I(\mathbf{x}', t) \right]^2$$

  - two more variables: direct light $\alpha$ and ambient light $\beta$

  $\rightarrow$ $8 \times 8$-os matrix, 8D vectors

- More variables $\longrightarrow$ more interpretatons of changes

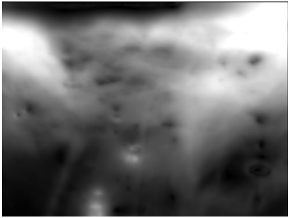
  $\rightarrow$ displacement can be imprecise

- Use simpler model when only possible
  - e.g., shift and cross-correlation

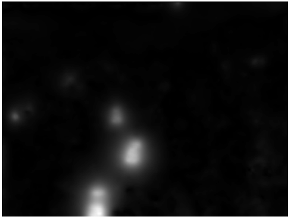
# Magnitude of optical flow: misty street

1.frame

2.frame

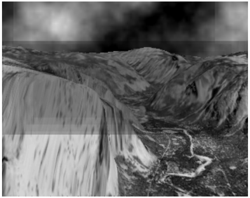


Horn-Schunk

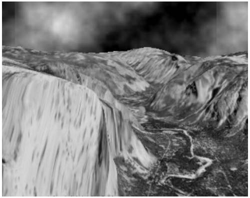


cross-correlation

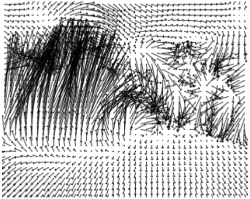
# Optical flow vectors: synthetic images with shadow



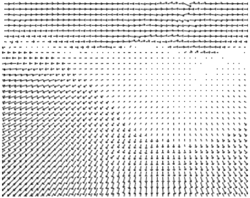
1.frame



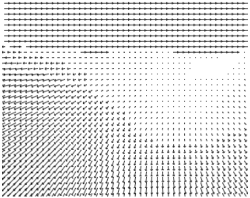
2.frame



Horn-Schunk



cross-correlation



ground truth

# Regularization of optical flow

- Horn-Schunk error function with smoothness term:

$$E_{HS}(\mathbf{u}) = \left(\mathbf{u}\nabla I(\mathbf{x}', t) + I_t(\mathbf{x}', t)\right)^2 + \lambda\left(\|\mathbf{u}_x\|^2 + \|\mathbf{u}_y\|^2\right)$$

  - $\lambda$: Lagrange multiplier (or parameter)

- Smoothness term:

$$\|\mathbf{u}_x\|^2 + \|\mathbf{u}_y\|^2 = u_x^2 + u_y^2 + v_x^2 + v_y^2, \quad \mathbf{u} \doteq [u, v]^T$$

  - $\mathbf{u}_x, \mathbf{u}_y$ : derivatives by $x, y$

→ Penalizes **drastic velocity variations** in image plane

- Iterative solution

# Variational version of Horn-Schunk

- Variational form:

$$\min_{\mathbf{u}(\mathbf{x})} \int_{\Omega} E_{HS}(\mathbf{u})\, dxdy$$

  - $\Omega$: complete image domain

  $\rightarrow$ flow field $\mathbf{u}(\mathbf{x})$ that minimizes **global error**

- Extending optical flow to non-textured regions

  $\rightarrow$ by smoothness constraint

- Optical flow is **ill-posed problem**:
  - infinite number of solutions *or*
  - unstable solution: drastic change for small input variation
  - e.g., **medial axis transform**

$\rightarrow$ **Regularization** by smoothness constraint

# Variational methods for optical flow

- Various flow error functions used

$$E(u,v) = E_D(u,v,I_0,I_1,I_{0p},I_{1q}) + \lambda E_S(u_p,v_q), \quad p,q = x,y,t$$
$$I_0 \doteq I(\mathbf{x},t), \quad I_1 \doteq I(\mathbf{x}+\mathbf{u},t+1)$$

  - various **data terms** (optical constraints) $E_D$
  - various **smoothness terms** $E_S$

- Varying **metrics**, e.g.,
  - $L_1$: $\|\mathbf{a} - \mathbf{b}\|$
  - $L_2$: $\|\mathbf{a} - \mathbf{b}\|^2$

- $L_1$: more robust, but not always derivable
  - $|x|$ derivable version: $\sqrt{x^2+\varepsilon^2}$, $\varepsilon \ll 1$

- $L_2$: derivable, but outlier-sensitive

# Further examples of error functions: Brox et al. (2004)

- Data term:

$$E_{DB}(u,v) = \Psi\left(I_1 - I_0)^2 + \gamma(\nabla I_1 - \nabla I_0)^2\right)$$

  - $\Psi(x^2) = \sqrt{x^2 + \varepsilon^2}$: modified $L_1$
  - $\gamma$: parameter
- Role of image gradients $\nabla I_0, \nabla I_1$:
  - robustness to illumination variations
  - $\rightarrow$ less sensitive, than intensity itself
- Smoothness term:

$$E_{SB}(u,v) = \Psi\left(\|\nabla_3 u\|^2 + \|\nabla_3 v\|^2\right)$$

  - $\nabla_3 u = [u_x, u_y, u_t]^T$: 3D gradient
  - $\rightarrow$ smoothness in image domain, coherence in time
  - $\rightarrow$ no drastic change in time in same point

# Major parameters of KLT motion tracker 1/2

- Translational mode: no affine matching (default)
- Maximum velocity (displacement)
  - defines number of pyramid levels and border margin
  - $\rightarrow$ cannot handle larger displacements
  - $\rightarrow$ no result in border margins
- Number of feature points to track $N$
  - $N$ most distinct KLT corner-like points
  - $\rightarrow$ set by $\lambda_2$, smaller eigenvalue of structure matrix $\mathbf{M}$
- Lower limit for $\lambda_2$
  - controls sensitivity of corner detector
  - $\rightarrow$ more points for smaller value of limit
- Minimum distance between feature points
  - discard point if stronger point exists within distance
  - $\rightarrow$ stronger: with larger $\lambda_2$

# Major parameters of KLT motion tracker 2/2

- Window size
  - default: $7 \times 7$ pixels
  - usual range: $5 - 15$ pixels
  - smaller size $\longrightarrow$ finer corners, greater noise sensitivity
- Lower limit for det **M**
  - controls invertibility of **M**
  - $\rightarrow$ trackability of points
- Maximum number of iterations
  - $\rightarrow$ to reach desired matching accuracy
- Residue
  - upper limit of matching error $\longrightarrow$ when point gets lost
- Replacement of lost points: yes/no
  - either starts new point, or $N$ decreases