Motion Estimation

S. Papadopoulos, Prof. Ioannis Pitas
Aristotle University of Thessaloniki
pitas@csd.auth.gr
www.aiia.csd.auth.gr
Version 3.4.3

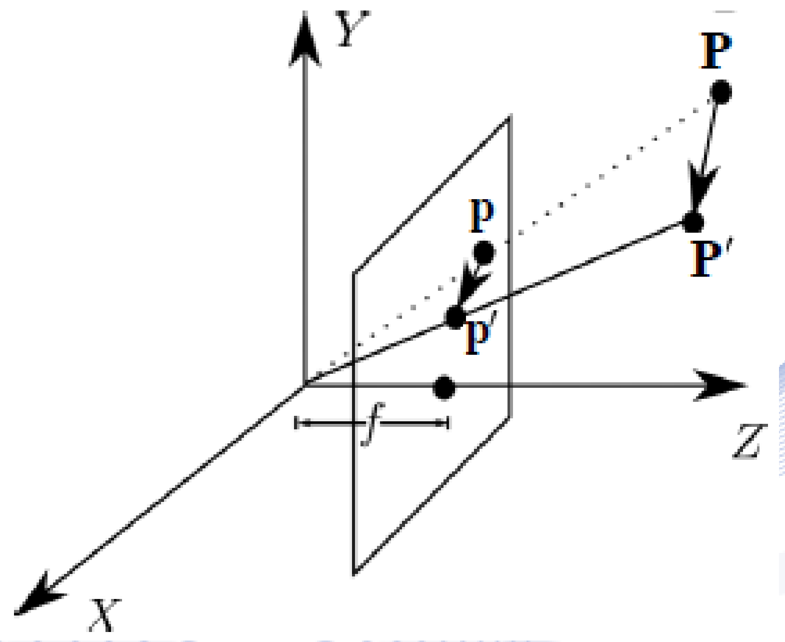
Artificial Intelligence &
Information Analysis Lab

Motion Estimation

- **2D motion**
- 3D motion models
- 2D motion models
- Estimation of 2D correspondence vectors
- Block matching
- Phase correlation
- Optical Flow Equation Methods
- Neural Optical Flow Estimation

2D motion

- Two-dimensional (2D) motion or ***projected motion*** is the perspective projection of the 3D motion on the image plane.
- Object point $\mathbf{P}$ at time t moves to point $\mathbf{P}'$ at t' and its perspective projection in the image plane from $\mathbf{p}$ to $\mathbf{p}'$.



Artificial Intelligence & Information Analysis Lab

2D motion

- The 2D displacement $t' = t + \ell\Delta t$ can be defined for all points $\mathbf{x}_t = [x, y, t]^T \in \mathbf{R}^3$ by the 2D ***displacement vector*** field $\mathbf{d}_c(\mathbf{x}_t; \ell\Delta t)$ as a function of the continuous spatiotemporal variables $[x, y]^T$ and t.
- The sampled 2D displacement field over a sampling is given by:

$$\mathbf{d}(n_1, n_2, n_t; \ell) = \mathbf{d}_p(\mathbf{x}_t; \ell\Delta t)\Big|_{\mathbf{x}_t = \mathbf{V}\mathbf{n}}, \qquad (n_1, n_2, n_t) \in Z^3$$

where $\mathbf{V}$ is a sampling matrix of the grid Λ^3.

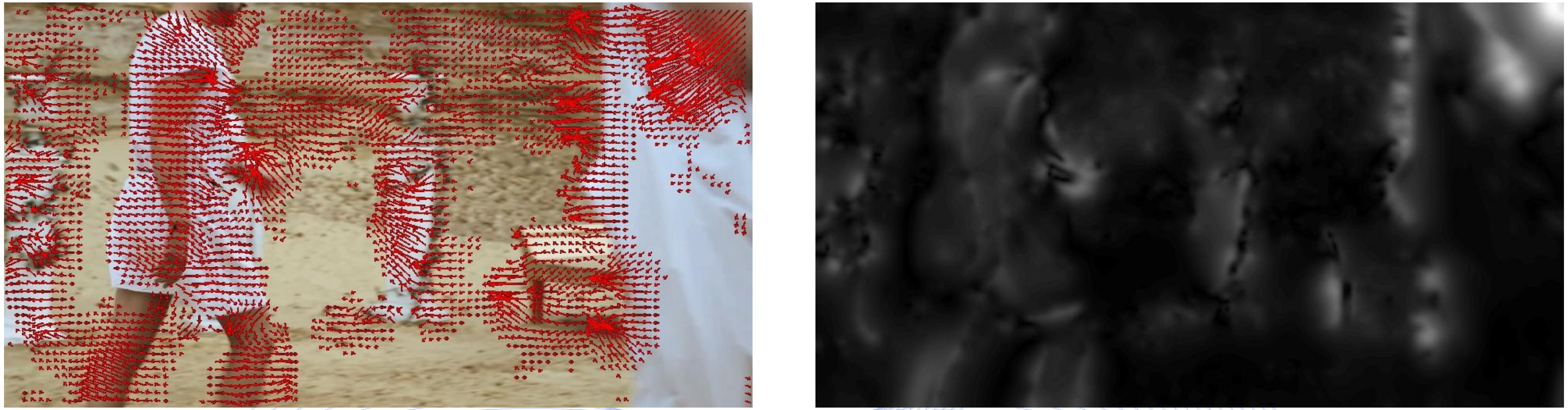
2D motion

- The 3D instantaneous velocity field $[dX/dt, dY/dt, dZ/dt]^T$ produces the projected velocity vector $\mathbf{v_p}(x, y, t)$ at time t.
- Discrete ***2D velocity vector*** field $\mathbf{v}(n_1, n_2, n_t) = \mathbf{v_p}(\boldsymbol{x}_t)$, for $\mathbf{x}_t = \mathbf{Vn} \in \Lambda^3$ and $\mathbf{n} = [n_1, n_2, n_t]^T \in \mathbf{Z}^3$.
- ***Correspondence vector*** denotes the displacement between the corresponding points $\mathbf{x} = [x, y]^T$ on the video frame at time t and $\mathbf{x}' = [x', y']^T$ at time t'.

2D motion

- ***Optical flow*** vector: the derivative of the correspondence vector: $[v_x, v_y]^T = [dx/dt\,, dy/dt]^T$.
- It describes the spatiotemporal changes of luminance $f_a(x, y, t)$.
- ***Motion speed***: magnitude of the motion vector.
- The correspondence or optical flow vectors determine the apparent motion.

2D motion



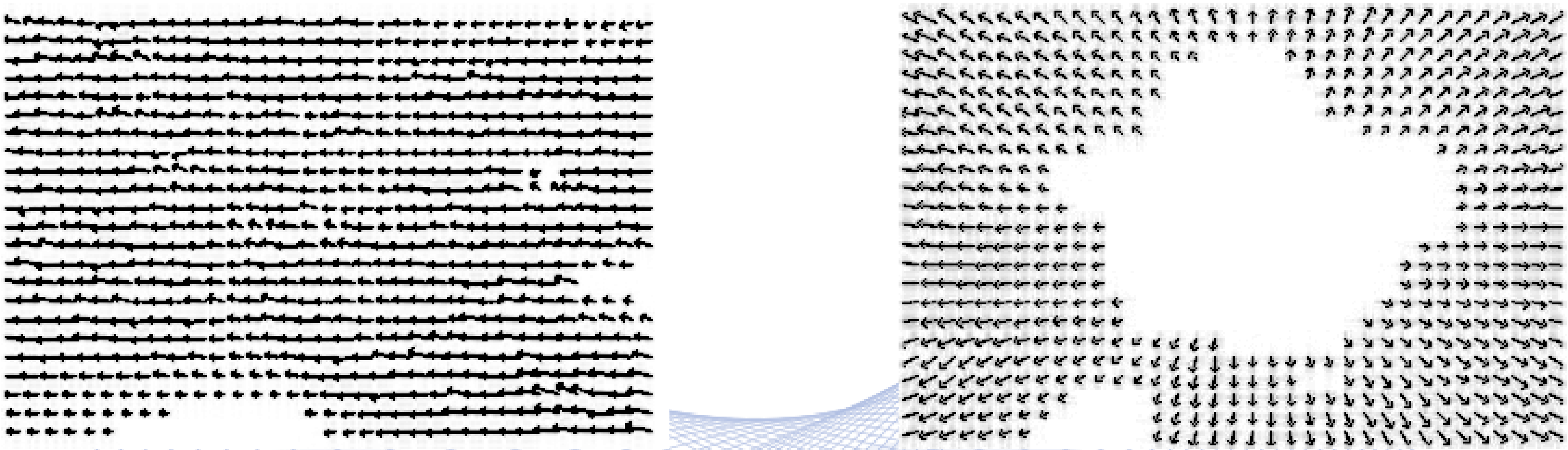
a) Motion field; b) motion speed.

Artificial Intelligence &
Information Analysis Lab

2D motion

- 2D motion can be generated by:
  - Object(s) motion
- Global 2D motion can be generated by:
  - Camera motion (***pan***, ***tilt***)
  - Camera zoom
- 2D apparent motion can be generated by a motion of the illumination source.

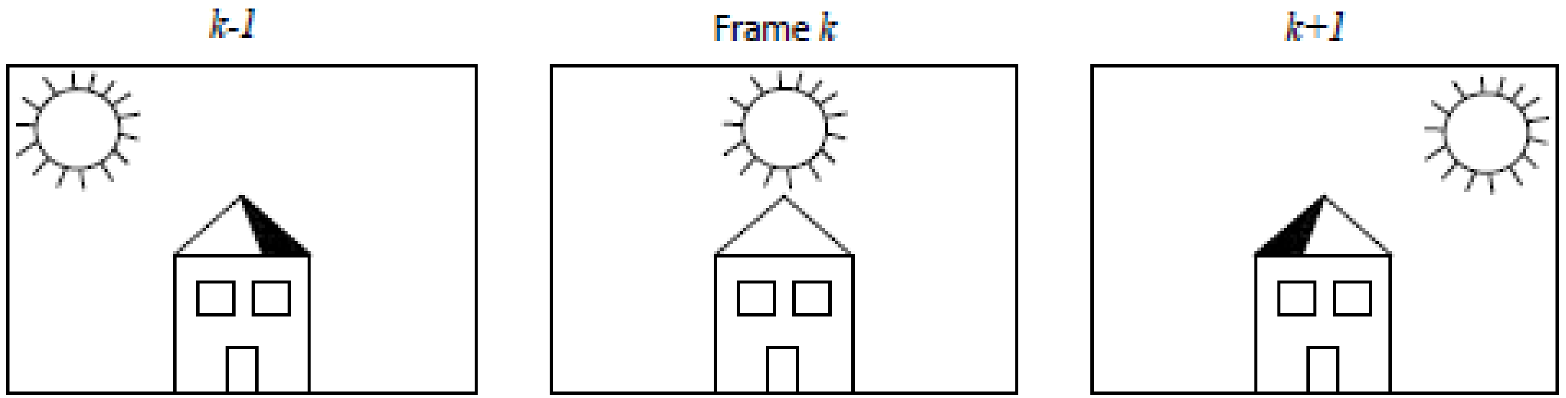
2D motion



Global optical flow generated by: a) camera pan and b) zoom.

2D motion

- The optical flow field may be different from the 2D displacement field:
  - When the image has insufficient spatial information, the actual motion field is not observable.
  - Illumination changes alter luminance value of a static object.



2D motion

- We may have real object motion but not apparent motion (optical flow).
  - If a big white sheet moves in a plane perpendicular to camera axis, there is 3D motion, but we observe no apparent motion.
  - If a while disk rotates around the camera axis, there is 3D motion, but we observe no apparent motion.

Motion Estimation

- 2D motion
- **3D motion models**
- 2D motion models
- Estimation of 2D correspondence vectors
- Block matching
- Phase correlation
- Optical Flow Equation Methods
- Neural Optical Flow Estimation

3D Motion Models

- 3D solid object motion can be described by the affine transformation:

$$\mathbf{X}' = \mathbf{R}\mathbf{X} + \mathbf{T},$$

where $\mathbf{T}$ is a 3×1 translation vector:

$$\mathbf{T} = \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}.$$

and $\mathbf{R}$ is a 3×3 rotation matrix (various forms).

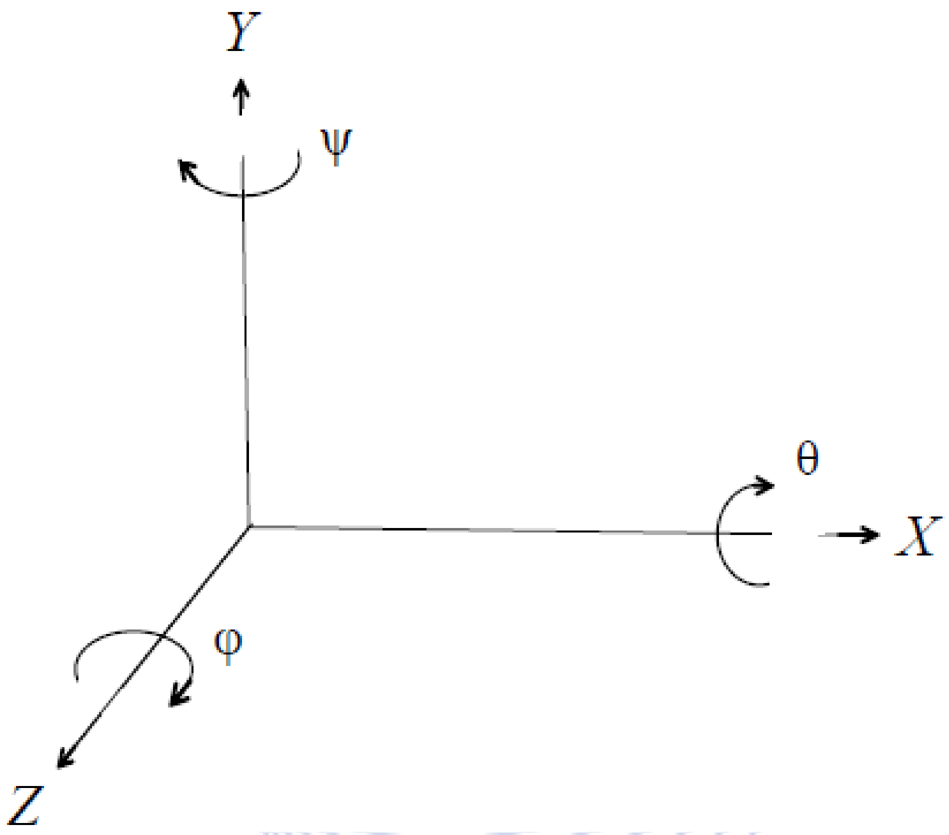
3D Motion Models

- In Cartesian coordinates, $\mathbf{R}$ can be described:
  - either by the ***Euler rotation angles*** about the three coordinate axes X, Y, Z.
  - or by a rotation axis and a rotation angle about this axis.
- The matrices describing the clockwise rotation around each axis in the three dimensional space, are given by:

$$\mathbf{R} = \mathbf{R}_Z \mathbf{R}_Y \mathbf{R}_X.$$

- Their order ***does matter***.
- $\mathbf{R}$ is ***orthonormal***, satisfying $\mathbf{R}^T = \mathbf{R}^{-1}$ and $det(\mathbf{R}) = \pm 1$.

3D Motion Models



Euler rotation angles.

$$\mathbf{R}_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix},$$

$$\mathbf{R}_Y = \begin{bmatrix} \cos\psi & 0 & \sin\psi \\ 0 & 1 & 0 \\ -\sin\psi & 0 & \cos\psi \end{bmatrix},$$

$$\mathbf{R}_Z = \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

3D Motion Models

- Infinitesimal 3D point rotation approximations:

$$\theta \approx \Delta\theta \approx 0, \quad \varphi \approx \Delta\phi \approx 0, \quad \psi \approx \Delta\psi \approx 0,$$

$$\cos\Delta\phi \approx 1, \qquad \sin\Delta\varphi \approx \Delta\phi \text{ (rad)}.$$

- Then, $\mathbf{R}$ takes the following form:

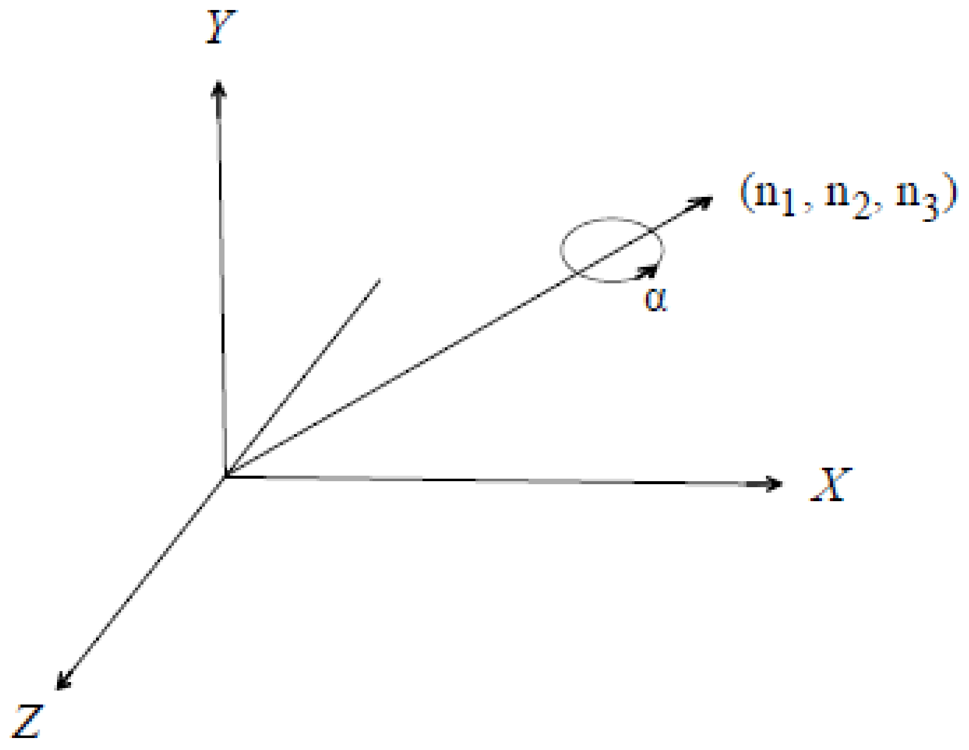
$$\mathbf{R} = \begin{bmatrix} 1 & -\Delta\phi & \Delta\psi \\ \Delta\phi & 1 & -\Delta\theta \\ -\Delta\psi & \Delta\theta & 1 \end{bmatrix}.$$

3D Motion Models

- In this case, the order of matrix multiplications in $\mathbf{R} = \mathbf{R}_Z\mathbf{R}_Y\mathbf{R}_X$ is irrelevant.
- 3D solid object rotation by an angle α about arbitrary axis passing through the origin and determined by the unit vector orientation vector $\mathbf{n} = [n_1, n_2, n_3]^T$:

$$\mathbf{R} = \begin{bmatrix} n_1^2 + (1 - n_1^2)\cos a & n_1 n_2(1 - \cos a) - n_3 \sin a & n_1 n_3(1 - \cos a) + n_2 \sin a \\ n_1 n_2(1 - \cos a) + n_3 \sin a & n_2^2 + (1 - n_2^2)\cos a & n_2 n_3(1 - \cos a) - n_1 \sin a \\ n_1 n_3(1 - \cos a) - n_2 \sin a & n_2 n_3(1 - \cos a) + n_1 \sin a & n_3^2 + (1 - n_3^2)\cos a \end{bmatrix}.$$

3D Motion Models



Object rotation about a rotation axis.

3D Motion Models

- For an infinitesimal rotation angle $\Delta\alpha \approx 0$:

$$\mathbf{R} = \begin{bmatrix} 1 & -n_3\Delta\alpha & n_2\Delta\alpha \\ n_3\Delta\alpha & 1 & -n_1\Delta\alpha \\ -n_2\Delta\alpha & n_1\Delta\alpha & 1 \end{bmatrix}.$$

- The infinitesimal rotation assumption holds when object motion is relatively slow and/or the time interval is small.
  - Valid for a relatively short time interval between video frames and slow moving video content.

Motion Estimation

- 2D motion
- 3D motion models
- **2D motion models**
- Estimation of 2D correspondence vectors
- Block matching
- Phase correlation
- Optical Flow Equation Methods
- Neural Optical Flow Estimation

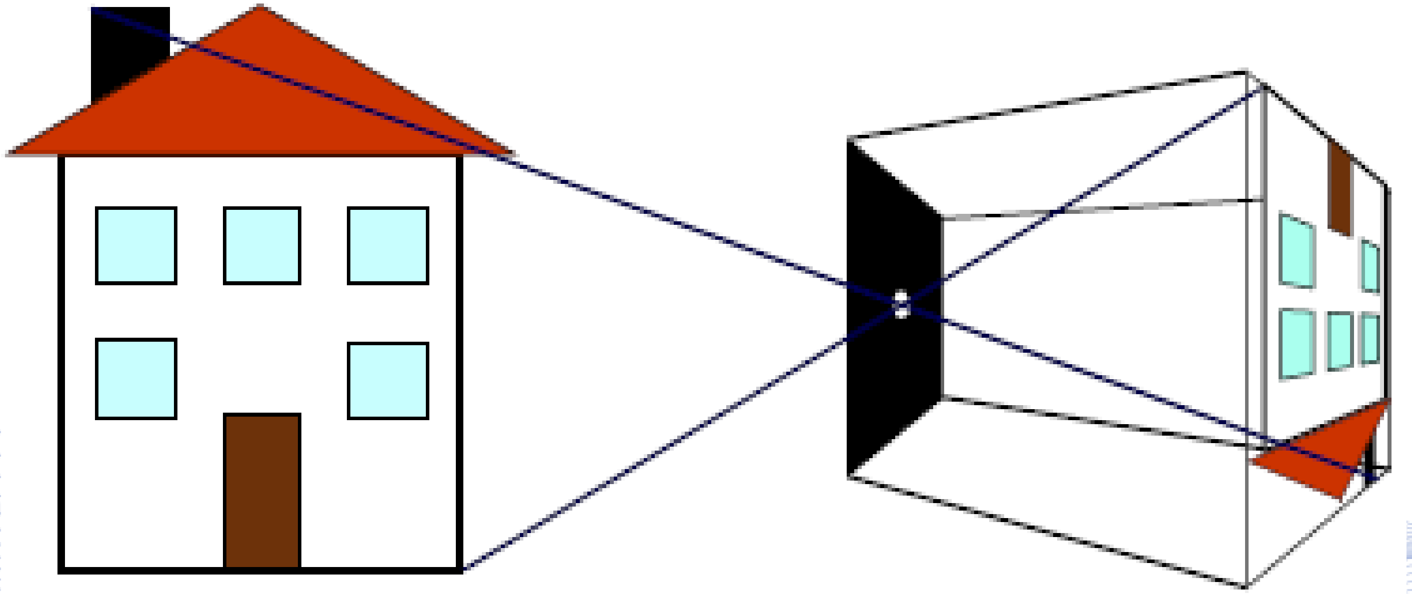
2D Motion Models

- In many occasions, it is difficult to distinguish between camera and visualized object motion.
- We consider that the camera remains static and the scene objects move:

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}.$$

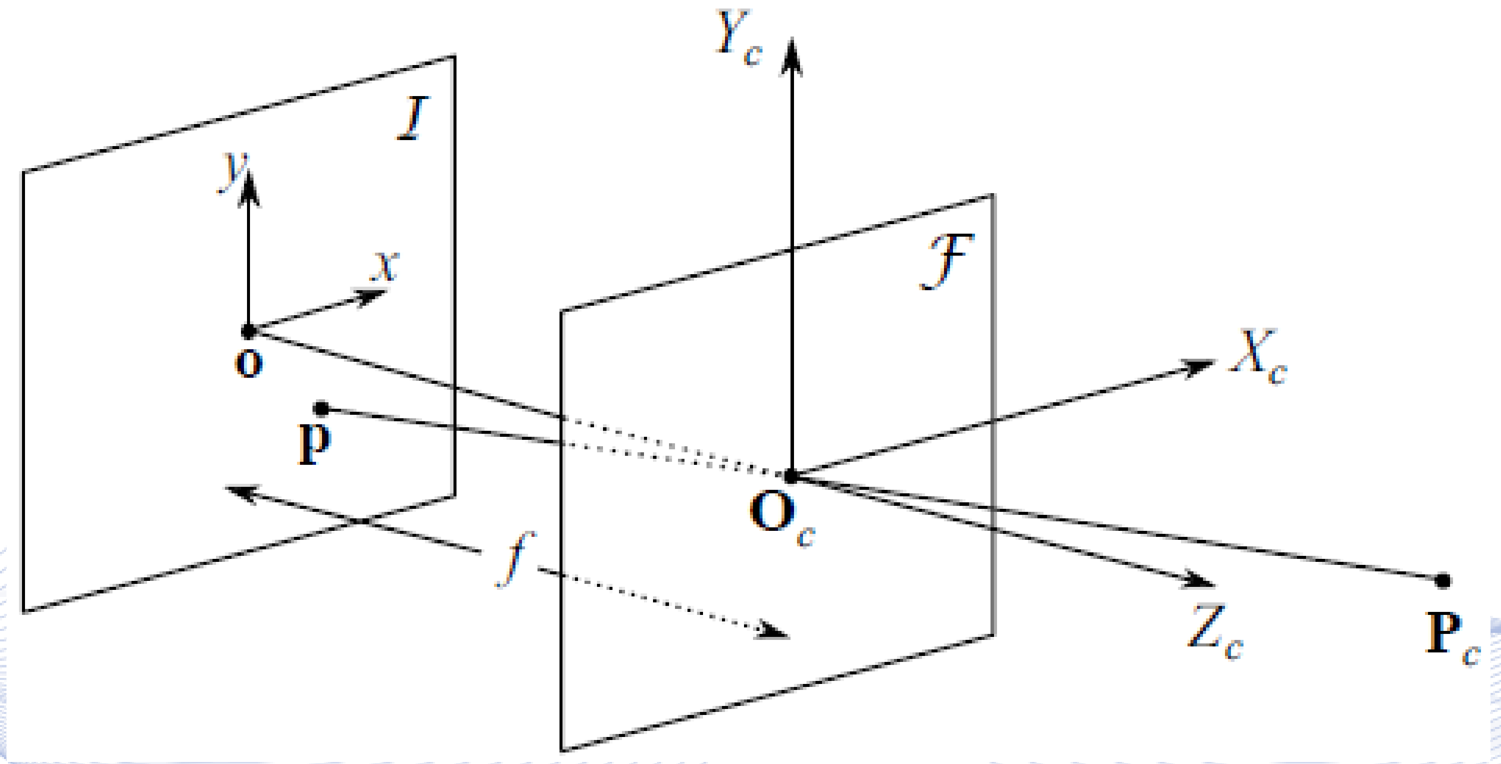
- From the 12 relevant parameters, only 6 are independent (3 rotation parameters and 3 translation vector components).

2D Motion Models

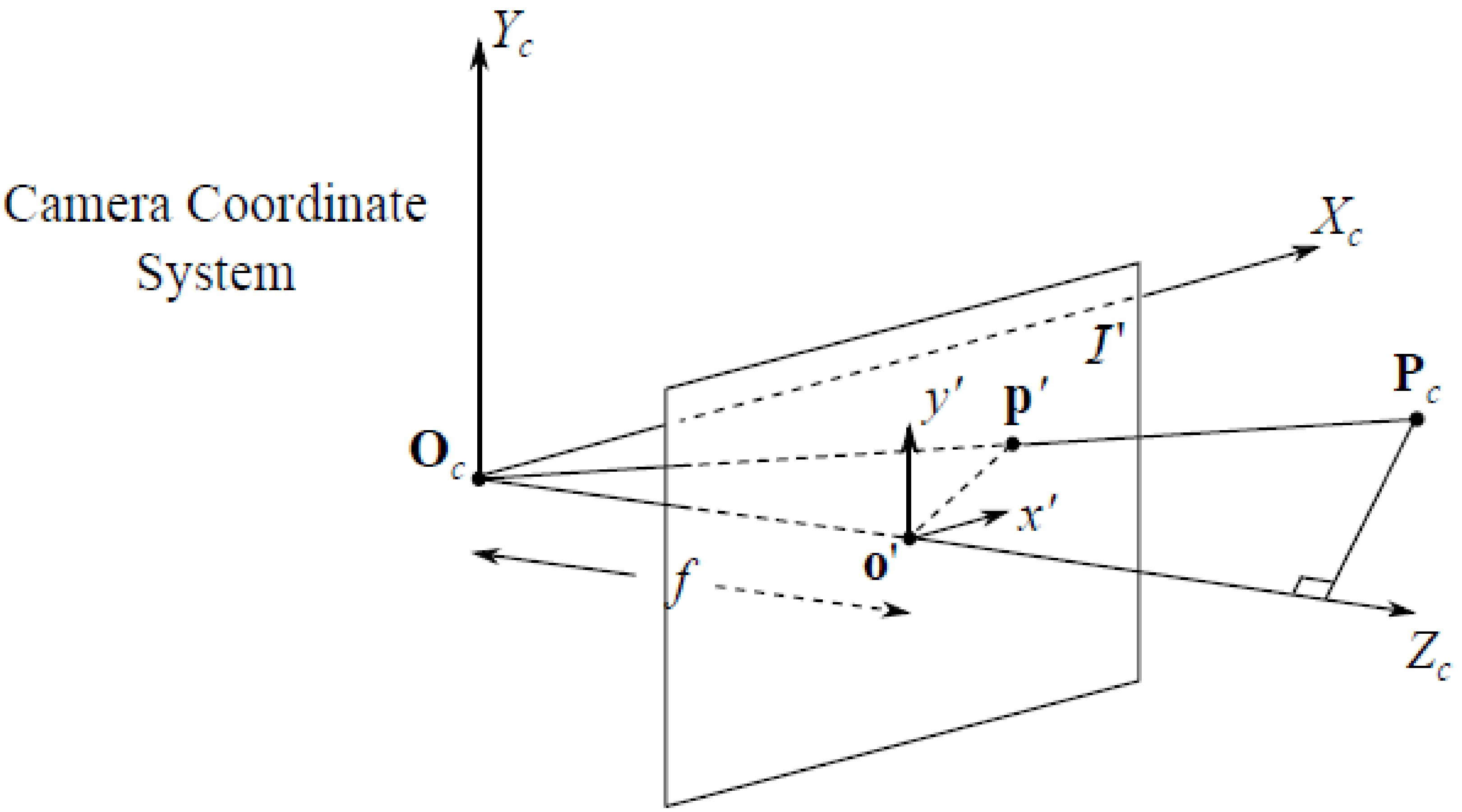


Pinhole camera geometry.

2D Motion Models



2D Motion Models



2D Motion Models

- We want to derive the equations that connect a 3D point (3D vector) $\mathbf{P}_c = [X_c, Y_c, Z_c]^T$ referenced in the camera coordinate system with its projection point (2D vector) $\mathbf{p} = [x, y]^T$ on the virtual image plane.
- By employing the similarity of triangles $\mathbf{O}_c\mathbf{o}'\mathbf{p}'$ and $\mathbf{O}_c\mathbf{Z}_c\mathbf{P}_c$:

$$x = f\frac{X_c}{Z_c}, \quad y = f\frac{Y_c}{Z_c}.$$

- Coordinates on the real image plane are given by the same equations, differing only by a minus sign.

2D Motion Models

- The new image point coordinates $[x', y']^T$ must be calculated as projections of the world coordinates.
- Analytical expression of the new coordinates $[x', y']^T$ on the image plane as a function of the old position $[x, y]^T$ and depth Z:

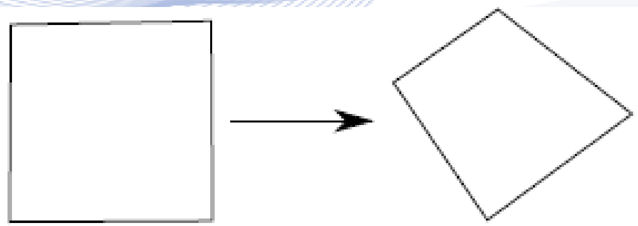
$$x' = \frac{(r_{11}x+r_{12}y+r_{13}f)Z+T_X f}{(r_{31}x+r_{32}y+r_{33}f)Z+T_Z f}, \quad y' = \frac{(r_{21}x+r_{22}y+r_{23}f)Z+T_X f}{(r_{31}x+r_{32}y+r_{33}f)Z+T_Z f}.$$

2D Motion Models

- ***Projective mapping transformation*** for no camera or object translation along the Z axis, or planar object:

$$x' = \frac{a_1+a_2x+a_3y}{1+a_7x+a_8y}, \qquad y' = \frac{a_4+a_5x+a_6y}{1+a_7x+a_8y}.$$

  - Parallel lines in the 3D space are represented by straight lines, converging to a vanishing point, on the image plane
  - Two successive projective mappings can be synthesized in one projective mapping.

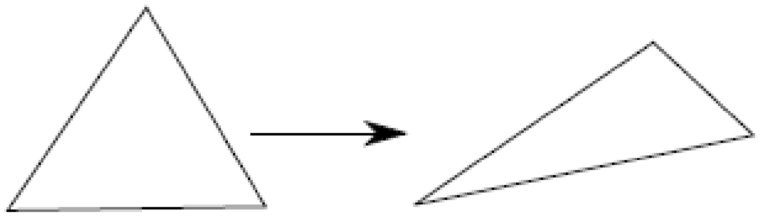


2D Motion Models

- Affine mapping transformation. The projected 2D motion of several camera motions as well as an arbitrary 3D motion of a planar object can be approximated by an affine transformation:

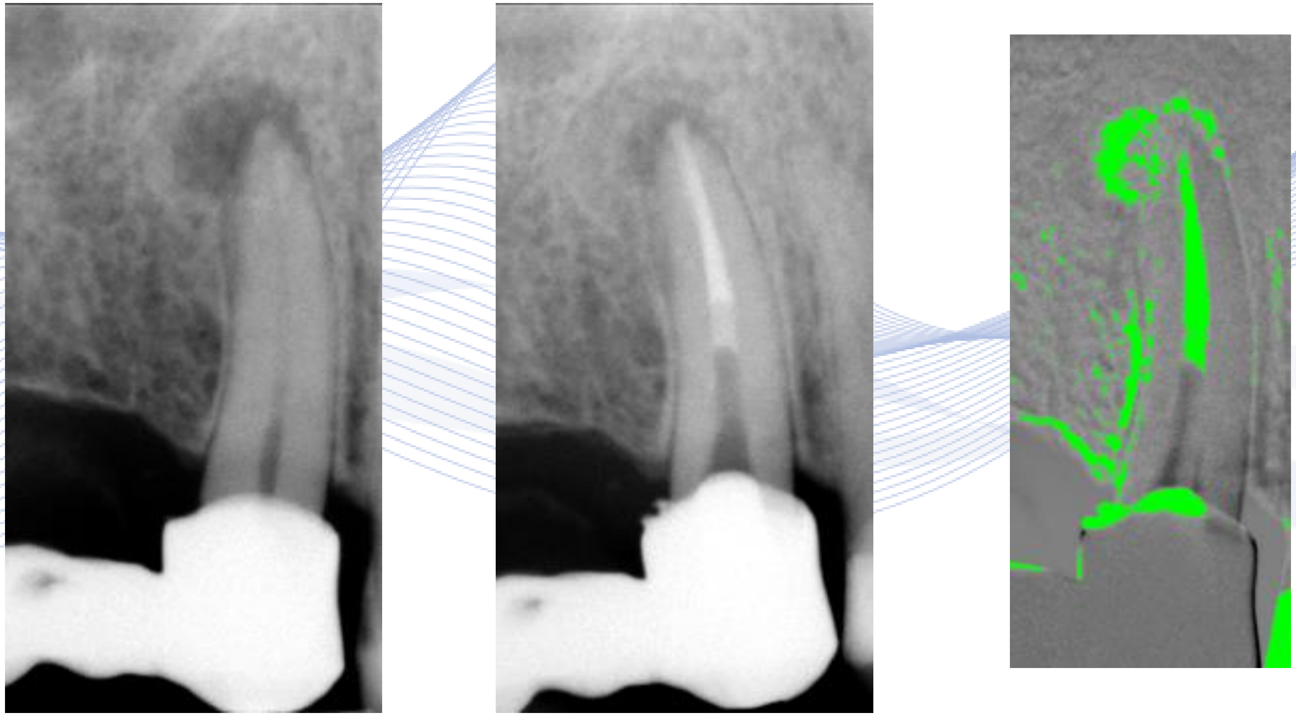
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_1 + a_2 x + a_3 y \\ a_4 + a_5 x + a_6 y \end{bmatrix}$$

- Deforms a triangle to another by shifting the triangle corners.



2D Motion Models

- 2D affine mapping transformation: it describes 2D rotation, translation and scaling.
- It can be used for 2D image registration.



Subtractive radiography.

Artificial Intelligence & Information Analysis Lab

2D Motion Models

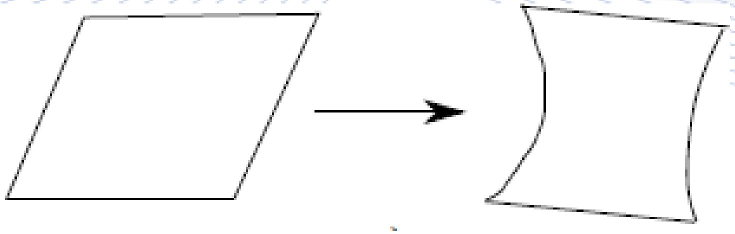
- 2D affine mapping transformation for image mosaicing.

2D Motion Models

- Quadratic (or bilinear) mapping transformation:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_1 + a_2 x + a_3 y + a_4 xy \\ a_5 + a_6 x + a_7 y + a_8 xy \end{bmatrix}.$$

  - It maps a straight line on a curved one, unless the original line is either horizontal or vertical.
  - It distorts a square into a (possibly curvilinear) quadrangle:



Motion Estimation

- 2D motion
- 3D motion models
- 2D motion models
- **Estimation of 2D correspondence vectors**
- Block matching
- Phase correlation
- Optical Flow Equation Methods
- Neural Optical Flow Estimation

Estimation of 2D correspondence vectors

- The correspondence problem can be studied:
  - As ***forward motion estimation***:
    - the motion vector is defined from frame t to $t+1$;
    - displacement vectors $\mathbf{d}(x,y) = [dx(x,y),\ dy(x,y)]^T$ should satisfy:

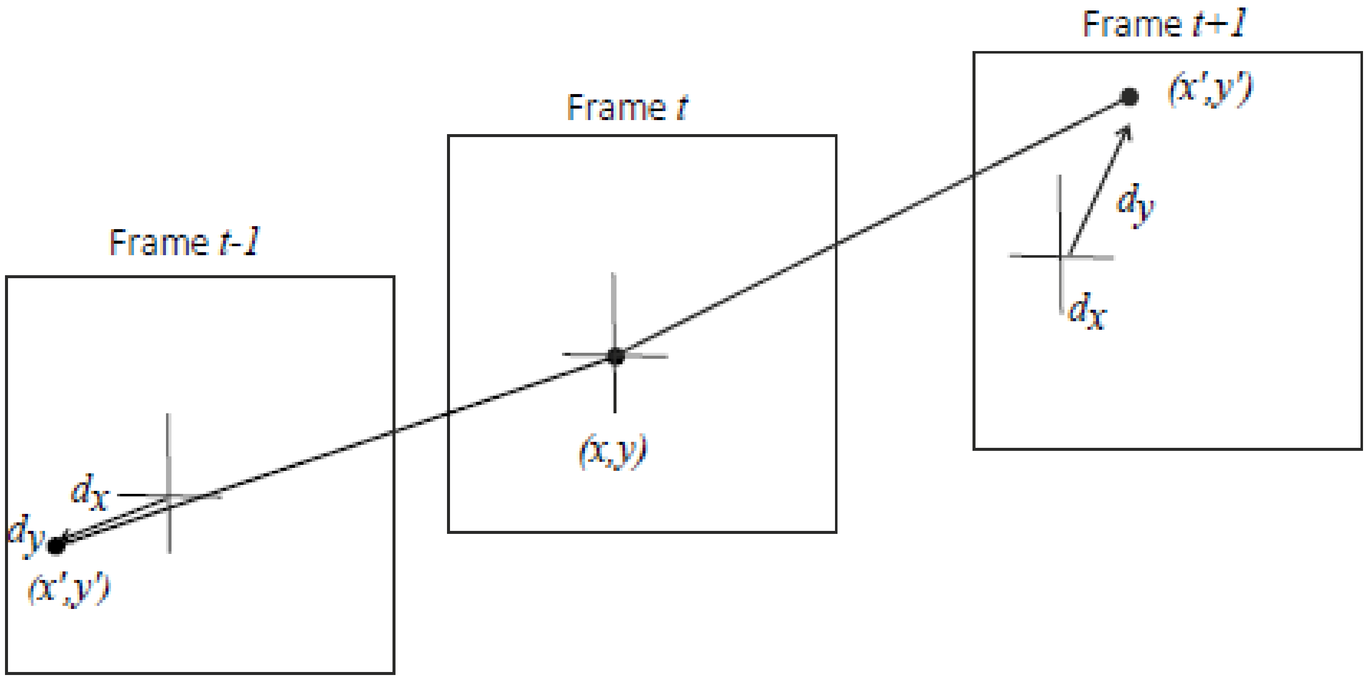
$$f(x,y,t) = f(x + dx(x,y), y + dy(x,y), t+1).$$

Estimation of 2D correspondence vectors

- As ***backward motion estimation***:
  - the motion vector is defined from frame t to $t-1$;
  - displacement vectors should satisfy:

$$f(x, y, t) = f(x + dx(x, y), y + dy(x, y), t - 1).$$

Estimation of 2D correspondence vectors



Forward and backward 2D motion estimation.

Estimation of 2D correspondence vectors

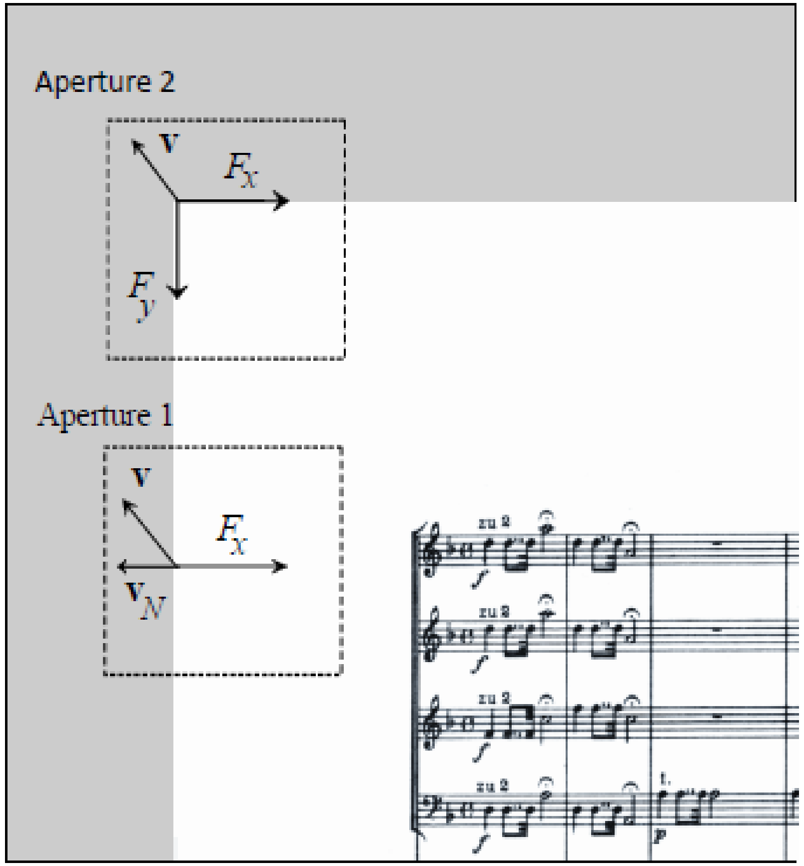
- For video compression, backward motion estimation is preferred.
- Problems associated with the uniqueness of object point matching over successive video frames:
  - ***Occlusion***: no correspondence can be found between occluded and un-occluded object or background region, due to object motion.
  - Partial or total occlusion. ***Self-occlusion***.

Estimation of 2D correspondence vectors

Object occlusion (right) and de-occlusion (left).

Estimation of 2D correspondence vectors

- ***Aperture problem***: only local spatial information (within the camera aperture) is used for motion estimation.



Quality metrics for motion estimation

- ***Peak Signal to Noise Ratio*** (***PSNR***): Metric for testing the quality of motion estimator results, measured in dB:

$$PSNR = 10\log_{10}\frac{N\times M}{\sum[f(x,y,t)-f(x+dx(x,y),\ y+dy(x,y),\ t-1)]^2}.$$

- $N \times M$: video frame size in pixels.
- Video luminance scaled in the range $[0,1]$.
- dx, dy: the displacement components resulting from motion estimation at pixel $\mathbf{p} = [x, y]^T$.

Quality metrics for motion estimation

- PSNR definition employs the ***Displaced Frame Difference (DFD)*** between the target frame t and the reference frame $t-1$.
- ***Motion field entropy***:

$$H = -\sum_{dx} p(dx) \log_2 p(dx) - \sum_{dy} p(dy) \log_2 p(dy).$$

- $p(dx), p(dy)$: the probability density function (relative frequency) of the horizontal and vertical components of the displacement vector $\mathbf{d}(x,y) = [dx(x,y),\ dy(x,y)]^T$.

Quality metrics for motion estimation

- Entropy: a measure of motion field smoothness.
  - Small when motion field estimation is good.
  - Large for poor motion field estimation, due to noise or lack of spatial frequencies:
    - More bits are required for motion field compression.
- Of particular interest in video compression with motion compensation.
  - Minimization of both the entropy of the motion field and of DFD is required in order to obtain higher compression.

Motion Estimation

- 2D motion
- 3D motion models
- 2D motion models
- Estimation of 2D correspondence vectors
- **Block matching**
- Phase correlation
- Optical Flow Equation Methods
- Neural Optical Flow Estimation

Block matching

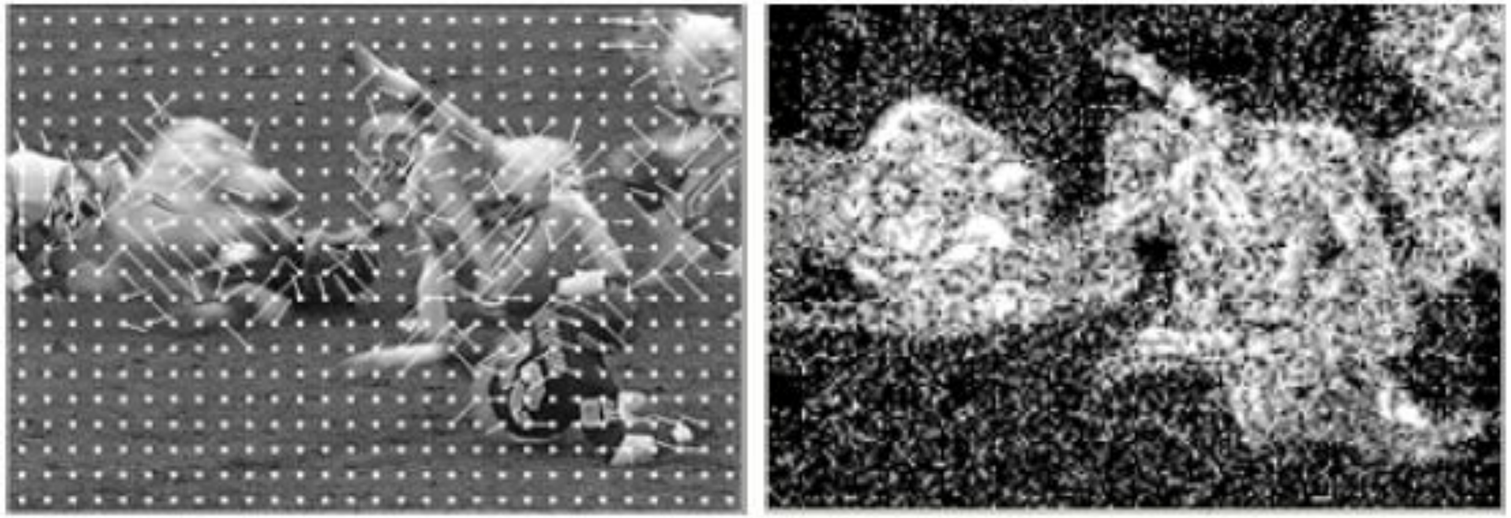
Block matching matches image blocks in consecutive video frames.

Block displacement $\mathbf{d}$ can be estimated by minimizing the displaced section difference for selecting the optimal displacement $\mathbf{d} = [dx, dy]^T$:

$$\min_{dx,dy} E(\mathbf{d}) = \sum_{n_1} \sum_{n_2} \|f(n_1, n_2, t) - f(n_1 + dx, n_2 + dy, t - 1)\|.$$

- n_1, n_2 are pixel coordinates.
- L_1, L_2, L_p norms can be used for displaced frame difference estimation.

Block matching



Sparse and dense motion fields.

Block matching

- Supposing a $N \times N$ video frame and a $m \times m$ ***pixel block*** $\mathcal{B}$ centered at $\mathbf{x}_0$ at frame t:
  - The ***search area*** at frame $t - 1$ for the $E(\mathbf{d})$ minimum is a $(2d_{max} + 1) \times (2d_{max} + 1)$ block.
  - Block $\mathcal{B}$ is moved by $\pm d_{max}$ horizontally and vertically around $\mathbf{x}_0$ and the minimum $E(\mathbf{d})$ in $(2d_{max} + 1)^2$ positions is calculated.

Block matching

- Total computational complexity for $(N/m) \times (N/m)$ ***non-overlapping blocks*** in each frame:

$$m \times m \times (N/m) \times (N/m) \times (2d_{max} + 1)^2 = N^2(2d_{max} + 1)^2.$$

- Total computational complexity for $N \times N$ ***overlapping blocks*** in each frame:

$$m \times m \times N \times N \times (2d_{max} + 1)^2 = N^2 m^2 (2d_{max} + 1)^2.$$

Block matching

- Search window block size: it is very important in a block-based motion estimation algorithm.
- It must be chosen in such a way that the window is:
  - large enough to accommodate large displacement vectors but
  - small enough to facilitate computations.
- $O(N^4)$ computational complexity of block matching by exhaustive search for d_{max} comparable to N.

Block matching

- Block matching may fail:
  - for small d_{max}, in cases of fast object motion with large displacement vectors.
  - in homogeneous image regions with comparable local minima.
- Good motion estimation is achieved, if there are edges within block $\mathcal{B}$.

Block matching

- Faster methods than exhaustive block matching:
  - Two-dimensional logarithmic search.
  - Three step search.
  - One dimensional search.

- There is no guarantee that they will reach the global minimum of the displaced block difference $E(\mathbf{d})$.

Block matching

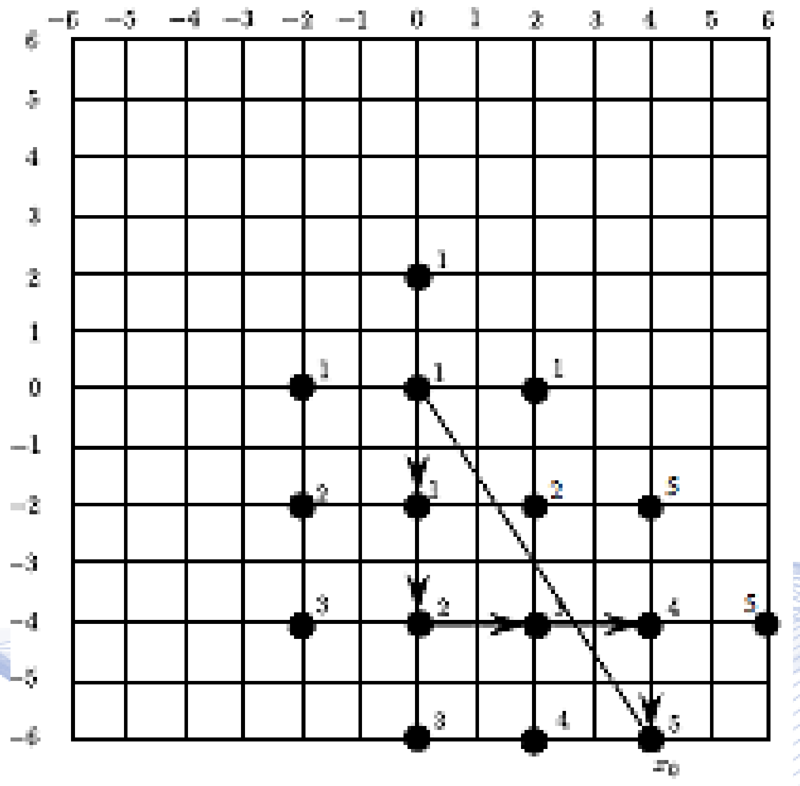
2D logarithmic search.

- The minimum search algorithm follows the direction of minimum difference:

$$\min_{dx,dy} E(\mathbf{d}) = \sum_{n_1} \sum_{n_2} \| f(n_1, n_2, t) - f(n_1 + dx, n_2 + dy, t-1) \|.$$

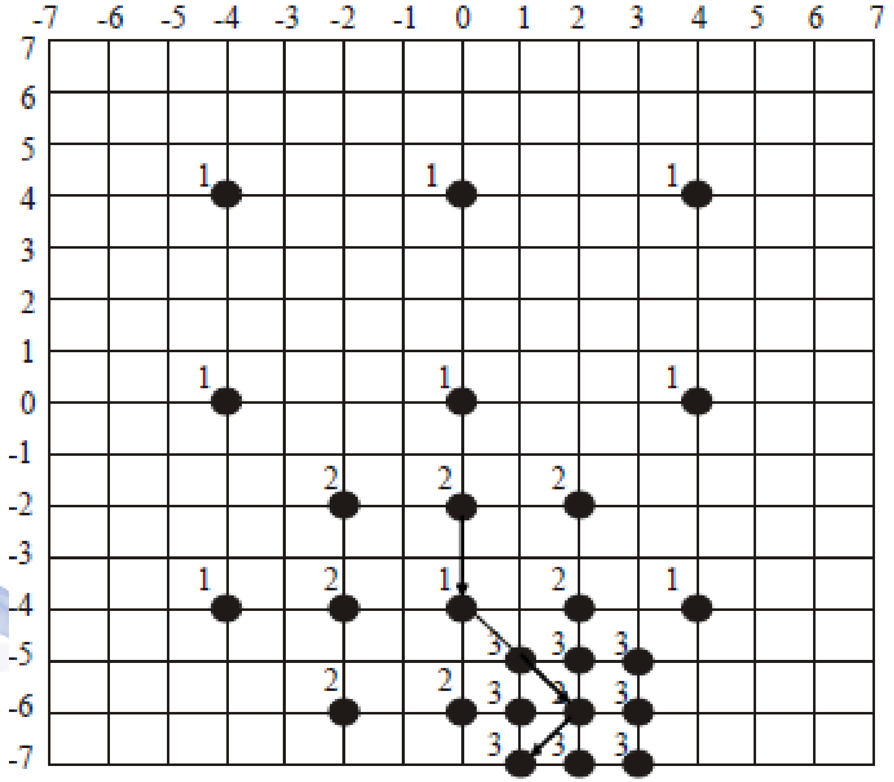
- Distance between the five possible search points becomes smaller, if the minimum is located at the center of the search area.

Block matching



- $d_{max} = 6$ pixels.
- Displacement from $\mathbf{x}_0 = [0,0]^T$ to $\mathbf{x}'_0 = [4,-6]^T$.

Block matching

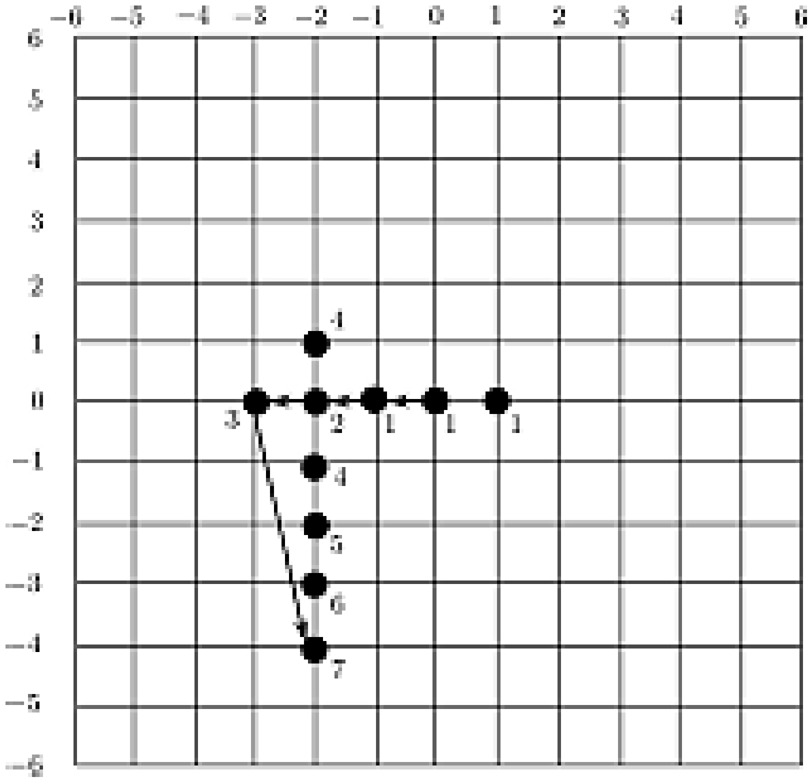


Three step search:

- 1st step: Eight pixels around $\mathbf{x}_0$ are checked.
- 2nd step: Eight pixels around the pixel of minimum $E(\mathbf{d})$ of step 1 are searched.
- …
- Search step size reduces at each step.

Artificial Intelligence & Information Analysis Lab

Block matching



In ***1D search***, $E(\mathbf{d})$ minimum is searched first along the horizontal and then along the vertical direction:

- ***1st step***. Search along the horizontal direction.
- ***2nd step***. Based on the results of step 1, the minimum is searched for along the vertical direction.

Motion Estimation

- 2D motion
- 3D motion models
- 2D motion models
- Estimation of 2D correspondence vectors
- Block matching
- **Phase correlation**
- Optical Flow Equation Methods
- Neural Optical Flow Estimation

Phase correlation

- Relative image blocks displacement is calculated using a normalized cross-correlation function calculated on the 2D spatial or Fourier domain.
- ***Cross-correlation*** between two video frames of size $N_1 \times N_2$ at times t and $t-1$:

$$r_{t,t-1}(n_1, n_2) = \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} f(k_1, k_2, t) f(n_1 + k_1, n_2 + k_2, t-1) = f(n_1, n_2, t) ** f(-n_1, -n_2, t-1).$$

** denotes a 2D convolution.

Phase correlation

- Taking the Fourier on both sides, we get the expression of complex cross-correlation spectrum:

$$R_{t,t-1}(\omega_x, \omega_y) = F_t^*(\omega_x, \omega_y) F_{t-1}(\omega_x, \omega_y).$$

* denotes complex conjugation.

- Phase of the cross-correlation spectrum:

$$\tilde{R}_{t,t-1}(\omega_x, \omega_y) = \frac{F_t^*(\omega_x, \omega_y) F_{t-1}(\omega_x, \omega_y)}{|F_t^*(\omega_x, \omega_y) F_{t-1}(\omega_x, \omega_y)|}.$$

Phase correlation

- Fourier transform of a displaced object, assuming linear translation motion by $[dx, dy]^T$ from frame $t-1$ to t:

$$F_t(\omega_x, \omega_y) = F_{t-1}(\omega_x, \omega_y)\exp\left(-i(\omega_x dx + \omega_y dy)\right).$$

- ***Normalized cross-correlation***:

$$\tilde{R}_{t,t-1}(\omega_x, \omega_y) = \exp\left(i(-\omega_x dx - \omega_y dy)\right),$$

$$\tilde{r}_{t,t-1}(n_1, n_2) = \delta(n_1 - dx, n_2 - dy).$$

Phase correlation

- Desirable properties:
  - Normalized cross-correlation peaks at (dx, dy).
  - Robustness to illumination changes: such changes do not affect the Fourier transform phase.
  - Detection of multiple moving objects in the same window:
    - If several correlation peaks are detected, each one of them indicates the motion of a particular object.

Phase correlation

The correlation of blocks of frames t and $t-1$ can be calculated:

- In the spatial domain:

$$r_{t,t-1}(n_1,n_2)=\sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} f(k_1,k_2,t)f(n_1+k_1,n_2+k_2,t-1),$$

- or by 2D Discrete Fourier Transform (DFT) and inverse DFT:
  - 2D Fast Fourier Transform (2D FFT).

Phase correlation

- Effects of using the 2D DFT:

  - Boundary problems,
  - Spectrum leakage,
  - Support area of displacement estimators.

Motion Estimation

- 2D motion
- 3D motion models
- 2D motion models
- Estimation of 2D correspondence vectors
- Block matching
- Phase correlation
- **Optical Flow Equation Methods**
- Neural Optical Flow Estimation

Optical flow equation methods

- The continuous spatiotemporal video luminance $f_a(x, y, t)$, not $f_a(x, y, t)$ does not change along the object motion trajectory.
- For $\mathbf{x}_t = [x, y, t]^T$ on motion trajectory, the ***total derivative*** $\frac{df_a(\mathbf{x}_t)}{dt} = 0$ leads to ***optical flow equation*** (***OFE***):

$$\frac{\partial f_a(\mathbf{x}_t)}{\partial x} v_x(\mathbf{x}, t) + \frac{\partial f_a(\mathbf{x}_t)}{\partial y} v_y(\mathbf{x}, t) + \frac{\partial f_a(\mathbf{x}_t)}{\partial t} = 0.$$

- $\mathbf{x} = [x, y]^T$, $\mathbf{x}_t = [x, y, t]^T$, $v_x(\mathbf{x}, t) = dx/dt$, $v_y(\mathbf{x}, t) = dy/dt$.

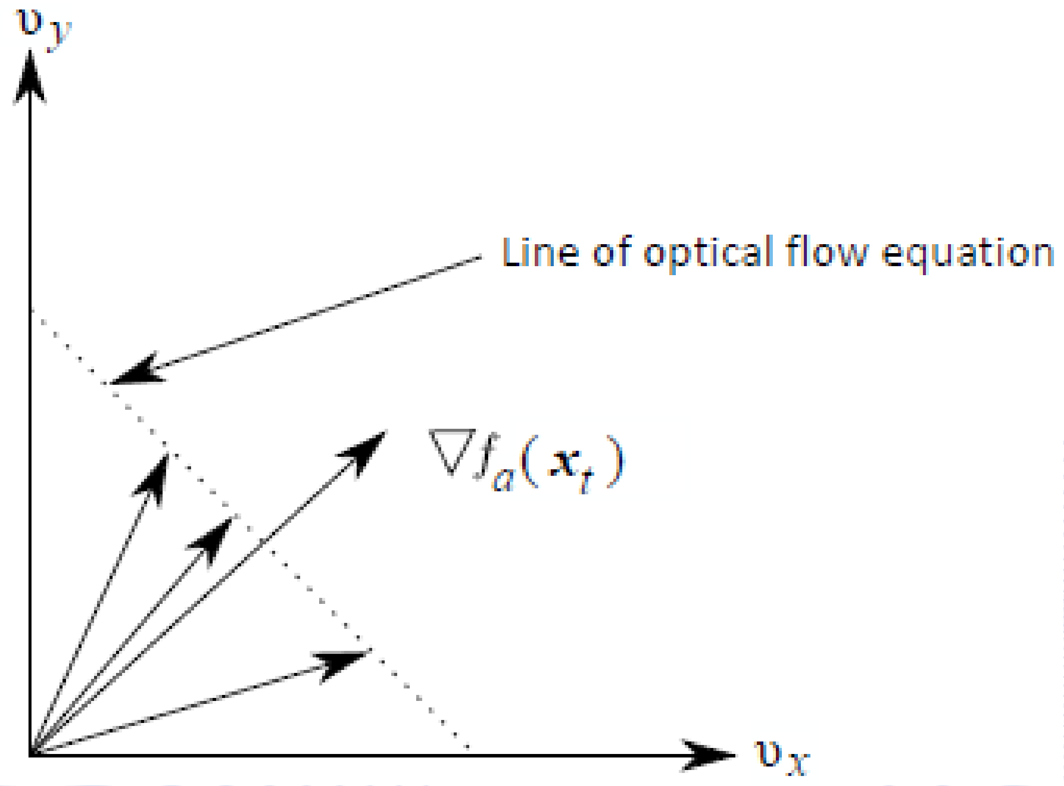
Optical flow equation methods

- OFE has two unknown factors, $v_x(\mathbf{x}, t)$ and $v_y(\mathbf{x}, t)$ for each $(\mathbf{x}, t)$, thus another equation is needed.
- The two velocity vector components are located on a straight line in the space (v_x, v_y).
- OFE can be expressed as:

$$\frac{\partial f_a(\mathbf{x}_t)}{\partial t} + \nabla f_a(\mathbf{x}_t)\mathbf{v}^T(\mathbf{x}_t) = 0,$$

where $\mathbf{v}(\mathbf{x}_t) = \left[v_x(\mathbf{x}_t, t), v_y(\mathbf{x}_t, t)\right]^T$ and $\nabla f_a(\mathbf{x}_t) = \left[\frac{\partial f_a(\mathbf{x}_t)}{\partial x}, \frac{\partial f_a(\mathbf{x}_t)}{\partial y}\right]^T$.

Optical flow equation methods



Line of optical flow equation.

Optical flow equation methods

- The velocity vector $\mathbf{v}(\mathbf{x}_t)$ component, the only one which can be estimated, is parallel to the direction of the spatial image gradient, the normal flow $\upsilon(\mathbf{x}, t)$:

$$\upsilon(\mathbf{x}, t) = \frac{-\frac{\partial f_a(\mathbf{x}_t)}{\partial t}}{\|\nabla f_a(\mathbf{x}_t)\|}.$$

- The object edges are invariant along motion trajectory and spatial image gradient $\nabla f_a(\mathbf{x}_t)$ is constant therein:

$$\frac{d\nabla f_a(\mathbf{x}_t)}{dt} = 0.$$

Optical flow equation methods

- The equation:

$$\frac{d\nabla f_a(\mathbf{x}_t)}{dt} = 0,$$

along with the OFE (two equations), suffice for the estimation of (v_x, v_y).

- Second order derivatives of luminance, enhance the image noise and may result in noisy optical flow estimates.

Optical flow equation methods

- Constant motion vector within an image block $\mathcal{B}$ can also be assumed through:

$$\mathbf{v}(\mathbf{x},t)=\mathbf{v}(t)=\left[v_x(t)\;\; v_y(t)\right]^T, \qquad \text{for } \mathbf{x}\in\mathcal{B}.$$

- The optical flow equation holds only approximately within $\mathcal{B}$.
- Optical flow equation error for the entire $\mathcal{B}$:

$$E(v_x,v_y)=\sum_{\mathbf{x}\in\mathcal{B}}\left(\frac{\partial f_a(\mathbf{x},t)}{\partial x}v_x(t)+\frac{\partial f_a(\mathbf{x},t)}{\partial y}v_y(t)+\frac{\partial f_a(\mathbf{x},t)}{\partial t}\right)^2.$$

Optical flow equation methods

- Equating the partial derivatives of the error function $E(v_x, v_y)$ with respect to $v_x(t)$ and $v_y(t)$ to 0, we get:

$$\begin{bmatrix}\hat{v}_x(t)\\ \hat{v}_y(t)\end{bmatrix} = \begin{bmatrix}\sum_{\mathbf{x}\in\mathcal{B}} \frac{\partial f_a(\mathbf{x}_t)}{\partial x}\frac{\partial f_a(\mathbf{x}_t)}{\partial x} & \sum_{\mathbf{x}\in\mathcal{B}} \frac{\partial f_a(\mathbf{x}_t)}{\partial x}\frac{\partial f_a(\mathbf{x}_t)}{\partial y}\\ \sum_{\mathbf{x}\in\mathcal{B}} \frac{\partial f_a(\mathbf{x}_t)}{\partial x}\frac{\partial f_a(\mathbf{x}_t)}{\partial y} & \sum_{\mathbf{x}\in\mathcal{B}} \frac{\partial f_a(\mathbf{x}_t)}{\partial y}\frac{\partial f_a(\mathbf{x}_t)}{\partial y}\end{bmatrix}^{-1} \begin{bmatrix}-\sum_{\mathbf{x}\in\mathcal{B}} \frac{\partial f_a(\mathbf{x}_t)}{\partial x}\frac{\partial f_a(\mathbf{x}_t)}{\partial t}\\ -\sum_{\mathbf{x}\in\mathcal{B}} \frac{\partial f_a(\mathbf{x}_t)}{\partial y}\frac{\partial f_a(\mathbf{x}_t)}{\partial t}\end{bmatrix}.$$

- Only first order derivatives are employed in this solution.
- Less sensitive to noise.

OFE smoothing methods

- They are based on the assumption that object motion is smooth, so that correspondence motion fields change smoothly in space.
  - Small spatial gradients.

- ***Horn-Schunck*** method: searches for a motion field that both satisfies the OFE and has small spatial optical flow vector changes.

OFE smoothing methods

- Satisfaction of OFE requires minimization of the squared error of:

$$E_1\big(\mathbf{v}(\mathbf{x},t)\big) = \nabla f_\alpha(\mathbf{x}_t)\mathbf{v}^T(\mathbf{x},t) + \frac{\partial f_a(\mathbf{x}_t)}{\partial t}.$$

- Spatial changes in the velocity vector field can be quantified by:

$$E_2^2\big(\mathbf{v}(\mathbf{x},t)\big) = \|\nabla v_x(\mathbf{x},t)\|^2 + \left\|\nabla v_y(\mathbf{x},t)\right\|^2 =$$
$$= \left(\frac{\partial v_x}{\partial x}\right)^2 + \left(\frac{\partial v_x}{\partial y}\right)^2 + \left(\frac{\partial v_y}{\partial x}\right)^2 + \left(\frac{\partial v_y}{\partial y}\right)^2.$$

OFE smoothing methods

- OFE smoothing minimizes $E_1^2(\mathbf{v}), E_2^2(\mathbf{v})$ wrt the velocity vector components (v_x, v_y) at each point $\mathbf{x} = [x, y]^T$:

$$\min_{\mathbf{v}(\mathbf{x},t)} \int_{\mathcal{A}} \left(E_1^2(\mathbf{v}) + \lambda E_2^2(\mathbf{v}) \right) dx.$$

λ: chosen heuristically parameter controling motion field smoothing.

OFE smoothing methods

- Simultaneous solution of two equations is required:

$$\left(\frac{\partial f_a}{\partial x}\right)^2 v_x(\mathbf{x},t) + \frac{\partial f_a}{\partial x}\frac{\partial f_a}{\partial y} v_y(\mathbf{x},t) = \lambda \boldsymbol{\nabla}^2 v_x(\mathbf{x},t) - \frac{\partial f_a}{\partial x}\frac{\partial f_a}{\partial t},$$

$$\frac{\partial f_a}{\partial x}\frac{\partial f_a}{\partial y} v_x(\mathbf{x},t) + \left(\frac{\partial f_a}{\partial y}\right)^2 v_y(\mathbf{x},t) = \lambda \boldsymbol{\nabla}^2 v_y(\mathbf{x},t) - \frac{\partial f_a}{\partial y}\frac{\partial f_a}{\partial t}.$$

$\boldsymbol{\nabla}^2$: Laplacian operator.

OFE smoothing methods

- Horn-Schunck implementation: Laplacian operator is approximated by high-pass FIR filters. Iterative Gauss-Seidel calculation method:

$$v_x^{(n+1)}(\mathbf{x},t) = \bar{v}_x^{(n)}(\mathbf{x},t) - \frac{\partial f_a}{\partial x} \frac{\frac{\partial f_a}{\partial x}\bar{v}_x^{(n)}(\mathbf{x},t) + \frac{\partial f_a}{\partial y}\bar{v}_y^{(n)}(\mathbf{x},t) + \frac{\partial f_a}{\partial t}}{\lambda + \left(\frac{\partial f_a}{\partial x}\right)^2 + \left(\frac{\partial f_a}{\partial y}\right)^2},$$

$$v_y^{(n+1)}(\mathbf{x},t) = \bar{v}_y^{(n)}(\mathbf{x},t) - \frac{\partial f_a}{\partial y} \frac{\frac{\partial f_a}{\partial x}\bar{v}_x^{(n)}(\mathbf{x},t) + \frac{\partial f_a}{\partial y}\bar{v}_y^{(n)}(\mathbf{x},t) + \frac{\partial f_a}{\partial t}}{\lambda + \left(\frac{\partial f_a}{\partial x}\right)^2 + \left(\frac{\partial f_a}{\partial y}\right)^2}.$$

- n: iteration counter;
- $\bar{v}_x, \bar{v}_y$: weighted local averages of v_x, v_y.

OFE smoothing methods

Horn-Schunck method applies optical flow field smoothing over the entire video frame.

- May have negative consequences in the accuracy of motion estimation.
- As the motion field is normally ***discontinuous at moving object boundaries***, universal smoothing constraints blur motion field boundaries.

It enforces optical flow in occluded and uncovered regions. It can be avoided by changing λ, controlling optical flow relative strength and smoothing terms.

Adaptive OFE methods

- Motion field can be maintained at edges by applying motion smoothing only along directions where image luminance does not change significantly.
- Approaches of adapting OFE to image content:
  - Application of smoothing constraints along the object contours, but not perpendicularly to them.
  - In occluded image regions:
    - Motion field smoothing constraint is in full force.
    - Optical flow constraint is not applied at all.

Adaptive OFE methods

- Directional motion field smoothing constraint:

$$E_2^2\big(\mathbf{v}(\mathbf{x},t)\big) = (\nabla v_x)^T \mathbf{W}(\nabla v_x) + \left(\nabla v_y\right)^T \mathbf{W}\left(\nabla v_y\right).$$

- **W**: a weight matrix punishing changes in the motion field, depending on the spatial image luminance changes:

$$\mathbf{W} = \frac{\mathbf{F} + \alpha \mathbf{I}}{trace(\mathbf{F} + \alpha \mathbf{I})}.$$

- **I**: the identity matrix, α: a scale factor.
- **F**: matrix containing spatial derivatives of $f_a(\mathbf{x}_t)$.

Adaptive OFE methods

- On object edges the diagonal elements of matrix $\mathbf{F}$ get large values, $\mathbf{W}$ elements are small and smoothing term vanished.
- In homogeneous image regions, matrix $\mathbf{F}$ is almost zero and the motion field smoothing term is in full force.
- Horn-Schunck method is a special case of adaptive motion field smoothing for $\alpha = 1$ and $\mathbf{F} = 0$.

Partial Differentiation in Motion Estimation

Numerical differentiation for spatiotemporal signals (digital video) $f(n_1, n_2.n_t)$:

$$\widehat{f_x} = \frac{1}{4}\{f(n_1+1, n_2, n_t) - f(n_1, n_2, n_t) + f(n_1+1, n_2+1, n_t) - f(n_1, n_2+1, n_t) + f(n_1+1, n_2, n_t+1) - f(n_1, n_2, n_t+1) + f(n_1+1, n_2+1, n_t+1) - f(n_1, n_2+1, n_t+1)\}.$$

Partial Differentiation in Motion Estimation

$$\widehat{f_y} = \frac{1}{4}\{f(n_1, n_2 + 1, n_t) - f(n_1, n_2, n_t) + f(n_1 + 1, n_2 + 1, n_t) -$$
$$-f(n_1 + 1, n_2, n_t) + f(n_1, n_2 + 1, n_t + 1) - f(n_1, n_2, n_t + 1) +$$
$$+f(n_1 + 1, n_2 + 1, n_t + 1) - f(n_1 + 1, n_2, n_t + 1)\},$$

$$\widehat{f_t} = \frac{1}{4}\{f(n_1, n_2, n_t + 1) - f(n_1, n_2, n_t) + f(n_1 + 1, n_2, n_t + 1) -$$
$$-f(n_1 + 1, n_2, n_t) + f(n_1, n_2 + 1, n_t + 1) - f(n_1, n_2 + 1, n_t) +$$
$$+f(n_1 + 1, n_2 + 1, n_t + 1) - f(n_1 + 1, n_2 + 1, n_t)\}.$$

Motion Estimation

- 2D motion
- 3D motion models
- 2D motion models
- Estimation of 2D correspondence vectors
- Block matching
- Phase correlation
- Optical Flow Equation Methods
- **Neural Optical Flow Estimation**

Neural Optical Flow estimation

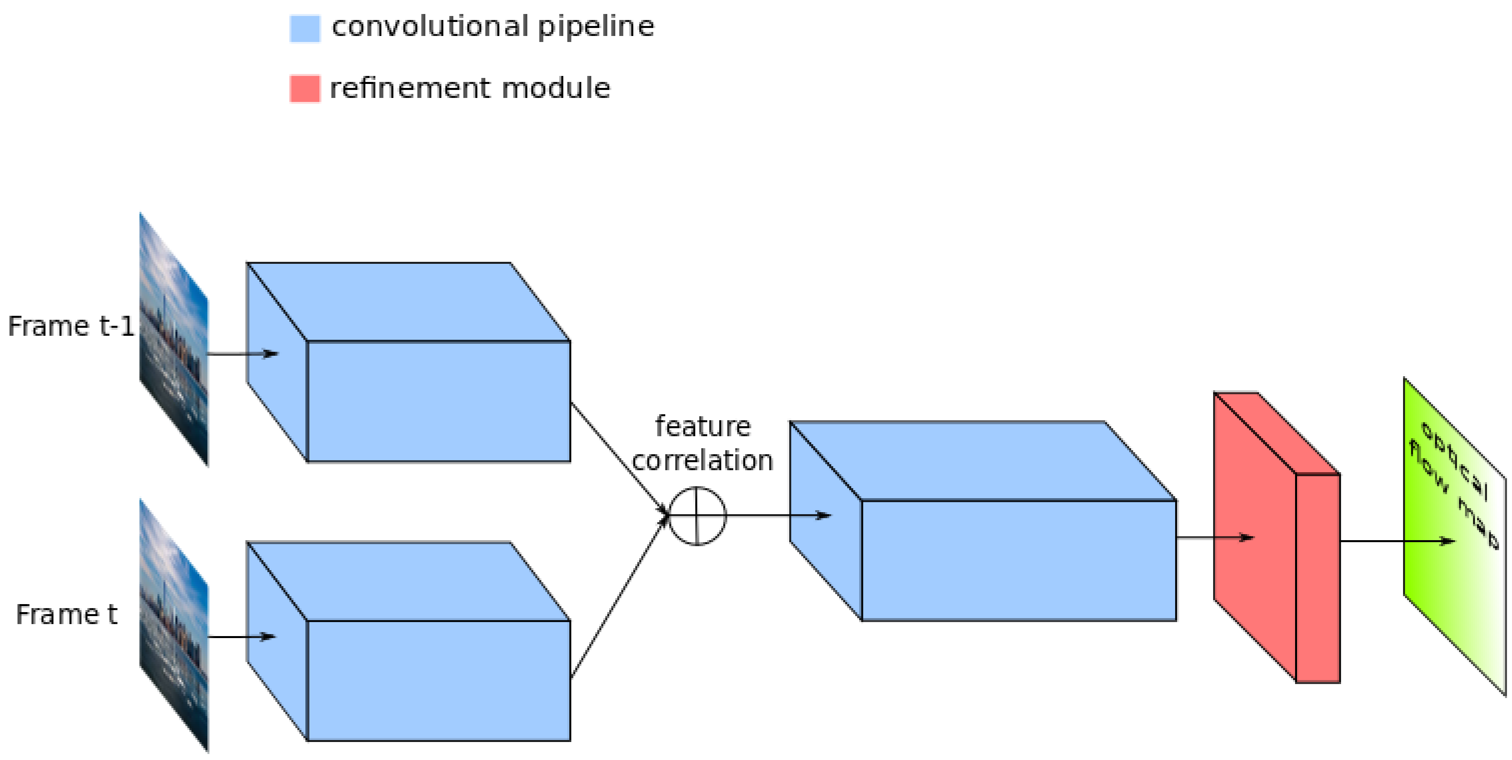
- Optical flow estimation by using ***Convolutional Neural Networks*** (***CNN***).
- High accuracy, dense flow field, fast implementations.
- Supervised methods:
  - Highest accuracy;
  - Ground truth for real world video sequences is required.
- Unsupervised methods:
  - Lower, but comparable accuracy;
  - No need for optical flow ground truth.

Neural Optical Flow estimation

Flownet: Supervised NN optical flow estimation.

- Foundation stone for almost all later supervised networks.
- ***FlowNetS*** (**S**imple):
  - A single network branch.
  - Refinement module upscales conv6 output, using outputs from various intermediate stages.
  - Two consecutive input frames, concatenated in the channel dimension.

Neural Optical Flow estimation



[DOS2015].

Neural Optical Flow estimation

FlowNetC (**C**orrelation):

- two separate branches extracting features for each input image;
- they are later merged into one branch by correlating the extracted feature maps:

$$r_{f_1,f_2}(n_1,n_2) = f_1(n_1,n_2) ** f_2(-n_1,-n_2).$$

- f_1, f_2: $(2K+1) \times (2K+1)$ 2D feature maps.

[8]

Neural Optical Flow estimation

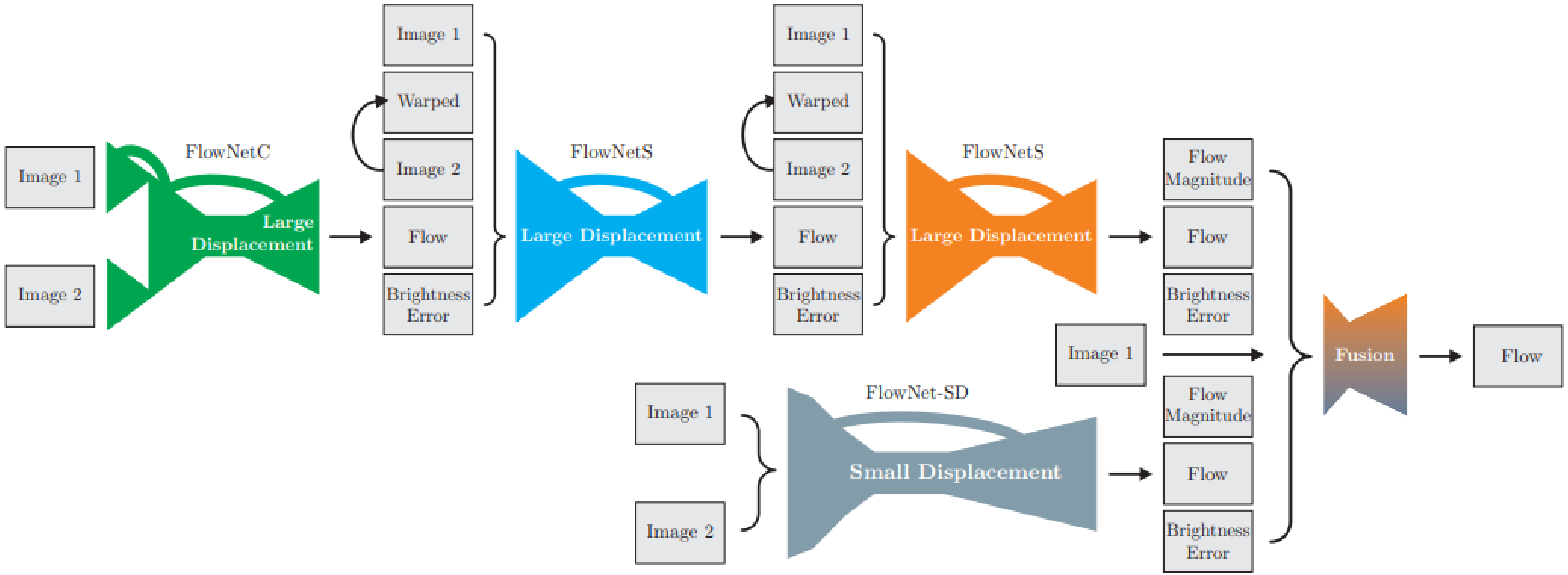
FlowNet 2.0:

- Warping of the 2nd image of the input image pair via the optical flow and bilinear interpolation;
- Substantial accuracy improvement;
- Marginal speed decrease;
- Modified training schedules can greatly improve performance.

Neural Optical Flow estimation

- Multiple FlowNets are combined to compute large displacement optical flow.
- Small diplacements are dealt with small strides and convolutions between upconvolutions in FlowNet.
- The final estimate is provided by a small fusion network.

Neural Optical Flow estimation



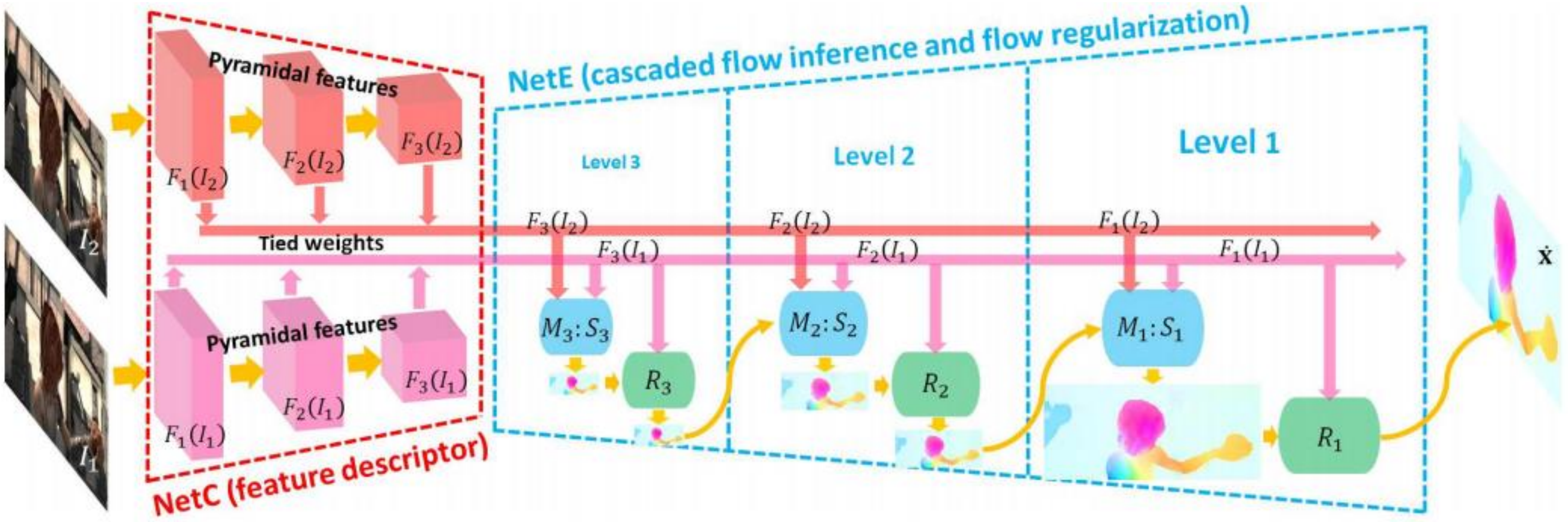
FlowNet 2.0 [ILG2017].

Neural Optical Flow estimation

LightFlowNet: Lightweight NN targeting FlowNet2 accuracy.

- Parameter number reduction from 162.49 to 5.37 million.
- Given an image pair, NetC generates two pyramids of high level features.
- NetE yields multi-scale flow fields each of which is generated by a cascaded flow inference model.
- M: descriptor matching, S: sub-pixel refinement, R: a regularization module.

Neural Optical Flow estimation



LightFlowNet. M: descriptor matching, S: sub-pixel refinement, R: a regularization module [HUI2018].

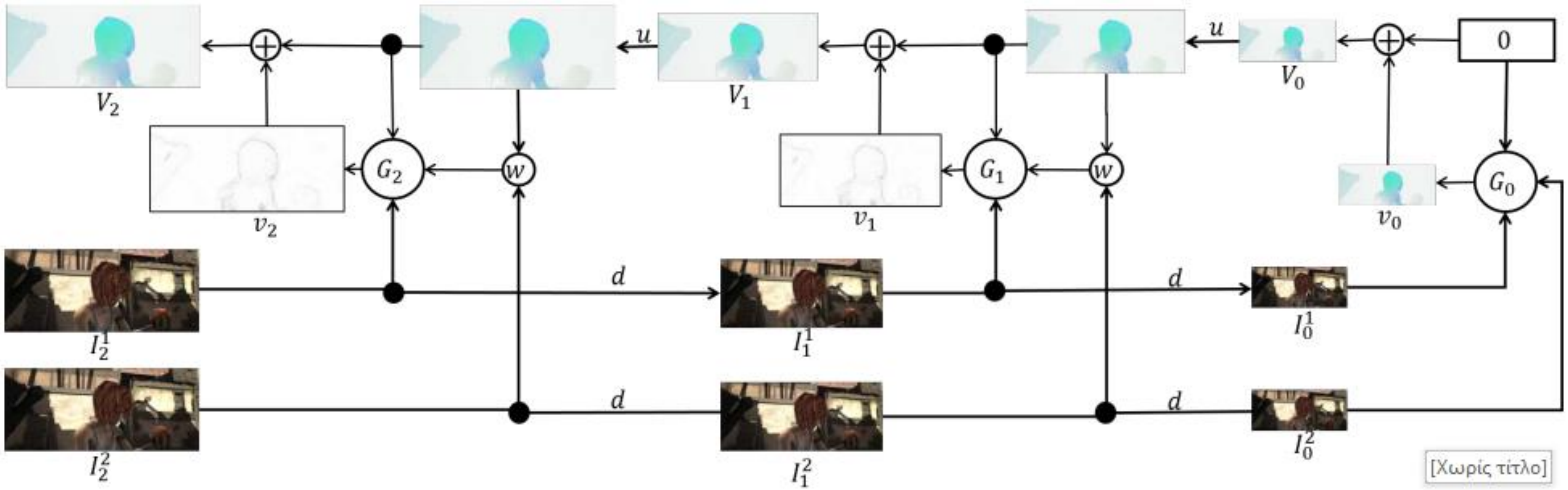
Neural Optical Flow estimation

SPyNet:

- 3-Level Pyramid Network.

- Better performance in many metrics than FlowNetC.
- More than twice as fast as FlowNetC.
- It uses the coarse-to-fine spatial pyramid structure to learn residual flow at each pyramid level.

Neural Optical Flow estimation



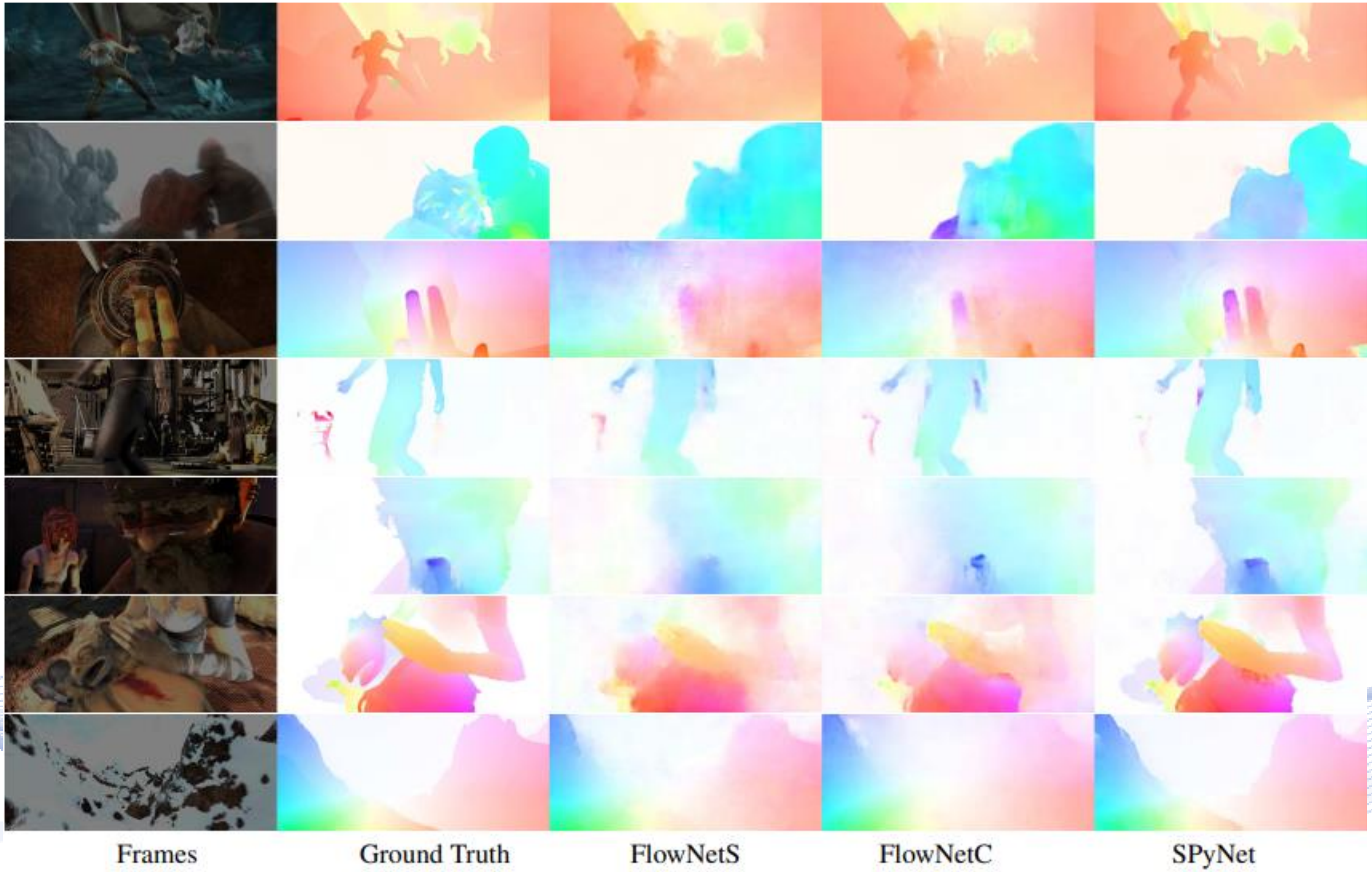
SPyNet 3-Level Pyramid Network [RAN2017].

Neural Optical Flow estimation

- Network G_0 computes residual flow $\mathbf{v}_0$ using the lowest resolution images $\{I_0^1, I_0^2\}$.
- At each pyramid level, G_k computes $\mathbf{v}_k$ using $\{I_k^1, I_k^2\}$ and $\mathbf{v}_{k-1}$.

- Finally, flow $\mathbf{v}_2$ is obtained at the highest resolution.

Neural Optical Flow estimation



Qualitative comparison of neural optical flow estimators [RAN2017].

Neural Optical Flow estimation

Unsupervised neural optical flow estimation methods:

- Ever more popular.
- Same NN configuration for joint training of:
  - optical flow, depth, camera pose, camera motion estimation, motion segmentation.
- Main idea: train a NN to minimize ***photometric loss*** between two consecutive video frames (one of which is warped via the estimated optical flow).

Object detection and Tracking

- Motion estimation estimates motion vectors on entire video frames.
- Object tracking relies on:
  - Object detection on a video frame.
  - Tracking of this object (essentially estimating its motion) over subsequent video frames.

Object Detection and Tracking

1st frame | 6th frame | 11th frame | 16th frame

- Problem statement:
  - To detect an object (e.g. human face) that appear in each video frame and localize its ***Region-Of-Interest*** (***ROI***).
  - To track the detected object over the video frames.

Object detection and Tracking

- Tracking associates each detected object ROI in the current video frame with one in the next video frame.

- Therefore, we can describe the ***object ROI trajectory*** in a video segment in (x, y, t) coordinates.

Object Detection and Tracking

- ***Tracking failure*** may occur, i.e.,
  - after occlusions;
  - when the tracker drifts to the background or to another object.
- In such cases, **object re-detection** is employed.
- However, if any of the detected objects coincides with any of the objects already being tracked, the former ones are retained, while the latter ones are discarded from any further processing.

Object Detection and Tracking

- ***Periodic object re-detection*** can be applied to account for new faces entering the camera's field-of-view.

- ***Forward and backward tracking***, when the entire video is available.

Bibliography

[PIT2017] I. Pitas, “Digital video processing and analysis” , China Machine Press, 2017 (in Chinese).

[PIT2013] I. Pitas, “Digital Video and Television” , Createspace/Amazon, 2013.

[PIT2021] I. Pitas, “Computer vision”, Createspace/Amazon, in press.

[NIK2000] N. Nikolaidis and I. Pitas, “3D Image Processing Algorithms”, J. Wiley, 2000.

[PIT2000] I. Pitas, “Digital Image Processing Algorithms and Applications”, J. Wiley, 2000.

Bibliography

[DOS2015] Dosovitskiy, Alexey, et al. "Flownet: Learning optical flow with convolutional networks." *Proceedings of the IEEE international conference on computer vision*. 2015.

[ILG2017] Ilg, Eddy, et al. "Flownet 2.0: Evolution of optical flow estimation with deep networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

[HUI2018] Hui, Tak-Wai, Xiaoou Tang, and Chen Change Loy. "Liteflownet: A lightweight convolutional neural network for optical flow estimation." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

[RAN2017] Ranjan, Anurag, and Michael J. Black. "Optical flow estimation using a spatial pyramid network." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

[ZHI2018] Yin, Zhichao, and Jianping Shi. "Geonet: Unsupervised learning of dense depth, optical flow and camera pose." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

Bibliography

[RAN2019] Ranjan, Anurag, et al. "Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019.

[ZOU2018] Zou, Yuliang, Zelun Luo, and Jia-Bin Huang. "Df-net: Unsupervised joint learning of depth and flow using cross-task consistency." *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.

[ZHOU2019] Zhu, Alex Zihao, et al. "Unsupervised event-based learning of optical flow, depth, and egomotion." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019.

Q & A

Thank you very much for your attention!

More material in
http://icarus.csd.auth.gr/cvml-web-lecture-series/

Contact: Prof. I. Pitas
pitas@csd.auth.gr