Computer Vision 2
WS 2018/19

Part 4 – Template-based Tracking II
23.10.2018

Prof. Dr. Bastian Leibe

RWTH Aachen University, Computer Vision Group
http://www.vision.rwth-aachen.de

Course Outline

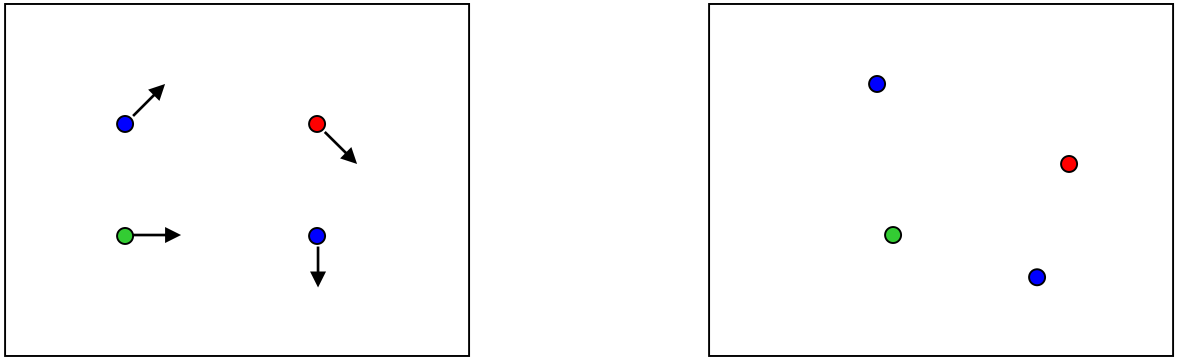
- Single-Object Tracking
  - Background modeling
  - Template based tracking
  - Tracking by online classification
  - Tracking-by-detection
- Bayesian Filtering
- Multi-Object Tracking
- Visual Odometry
- Visual SLAM & 3D Reconstruction
- Deep Learning for Video Analysis

Image source: Robert Collins

Topics of This Lecture

- Recap: Lucas-Kanade Optical Flow
  - Brightness Constancy constraint
  - LK flow estimation
  - Coarse-to-fine estimation
  - KLT feature tracking
- Template Tracking
  - LK derivation for templates
  - Warping functions
  - General LK image registration
- Applications

Recap: Estimating Optical Flow



$I(x,y,t-1)$ $I(x,y,t)$

- Optical Flow
  - Given two subsequent frames, estimate the apparent motion field $u(x,y)$ and $v(x,y)$ between them.

- Key assumptions
  - Brightness constancy: projection of the same point looks the same in every frame.
  - Small motion: points do not move very far.
  - Spatial coherence: points move like their neighbors.

Slide credit: Svetlana Lazebnik

Recap: Lucas-Kanade Optical Flow

- Use all pixels in a $K \times K$ window to get more equations.
- Least squares problem:

$$\begin{bmatrix} I_x(\mathrm{p}_1) & I_y(\mathrm{p}_1) \\ I_x(\mathrm{p}_2) & I_y(\mathrm{p}_2) \\ \vdots & \vdots \\ I_x(\mathrm{p}_{25}) & I_y(\mathrm{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathrm{p}_1) \\ I_t(\mathrm{p}_2) \\ \vdots \\ I_t(\mathrm{p}_{25}) \end{bmatrix} \qquad \underset{25\text{x}2}{A} \; \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

- Minimum least squares solution given by solution of

$$\underset{2\text{x}2}{(A^T A)} \; \underset{2\text{x}1}{d} = \underset{2\text{x}1}{A^T b}$$

Recall the Harris detector!

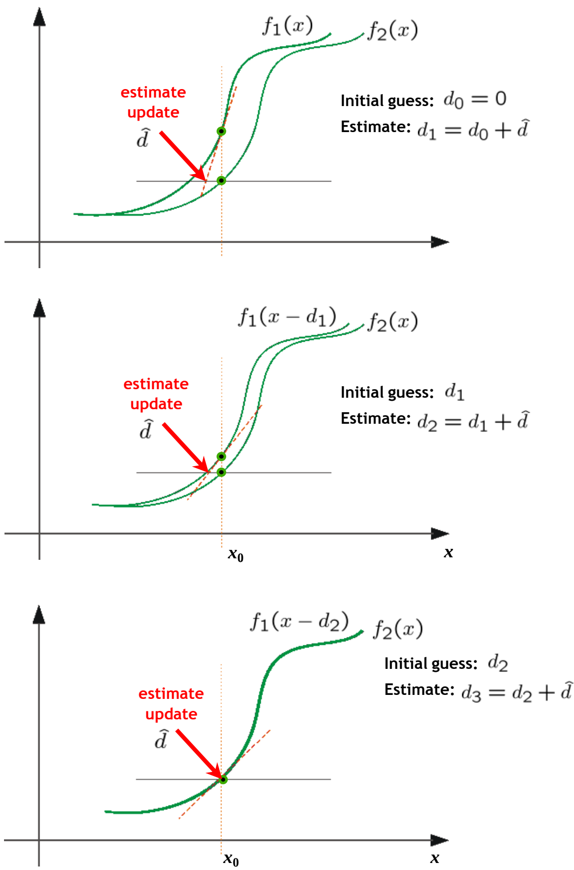
$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

Slide credit: Svetlana Lazebnik

Recap: Iterative LK Refinement

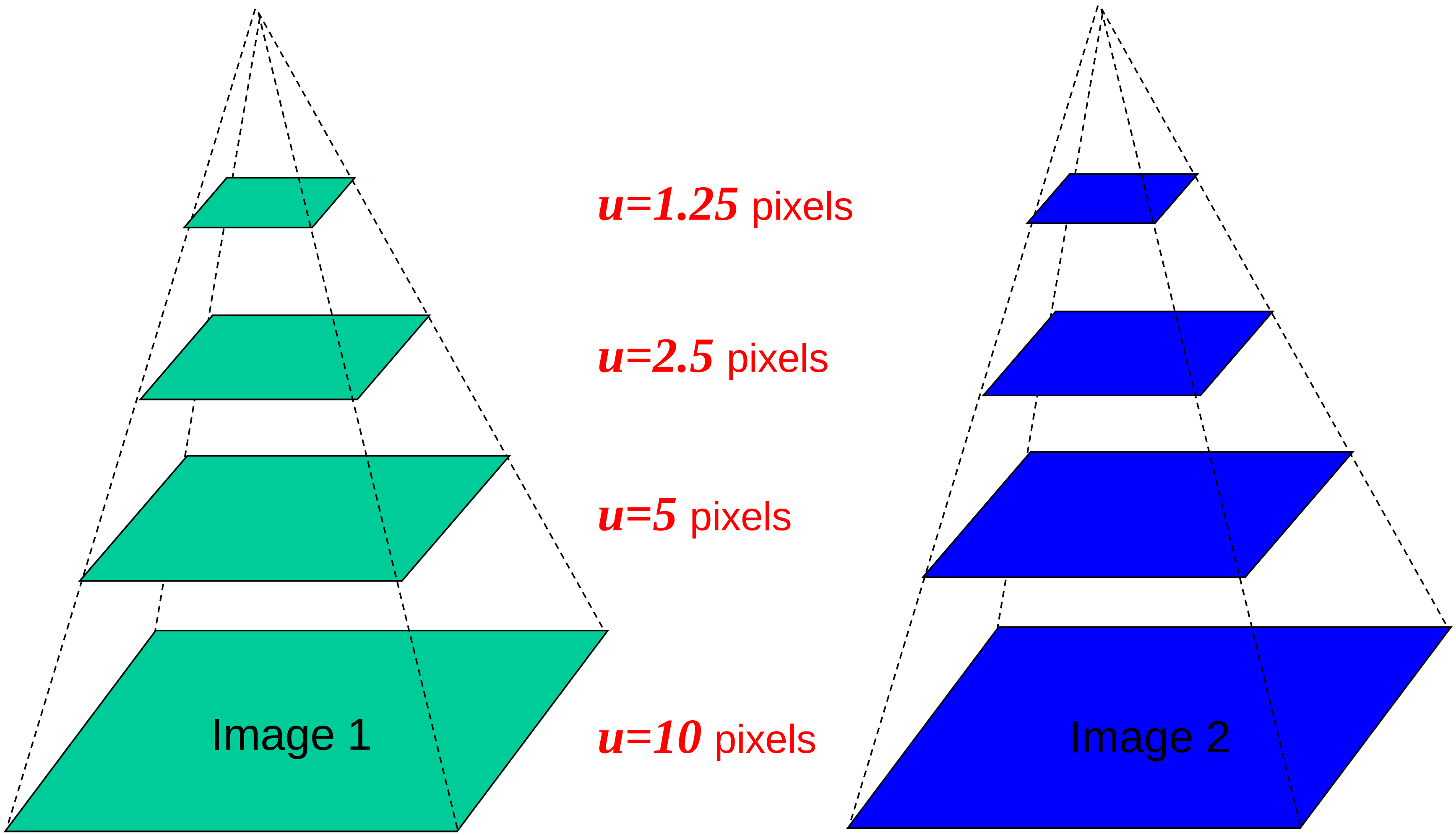
- Estimate velocity at each pixel using one iteration of LK estimation.
- Warp one image toward the other using the estimated flow field.
- Refine estimate by repeating the process.

- Iterative procedure
  - Results in subpixel accurate localization.
  - Converges for small displacements.



Slide adapted from Steve Seitz

Recap: Coarse-to-fine Optical Flow Estimation

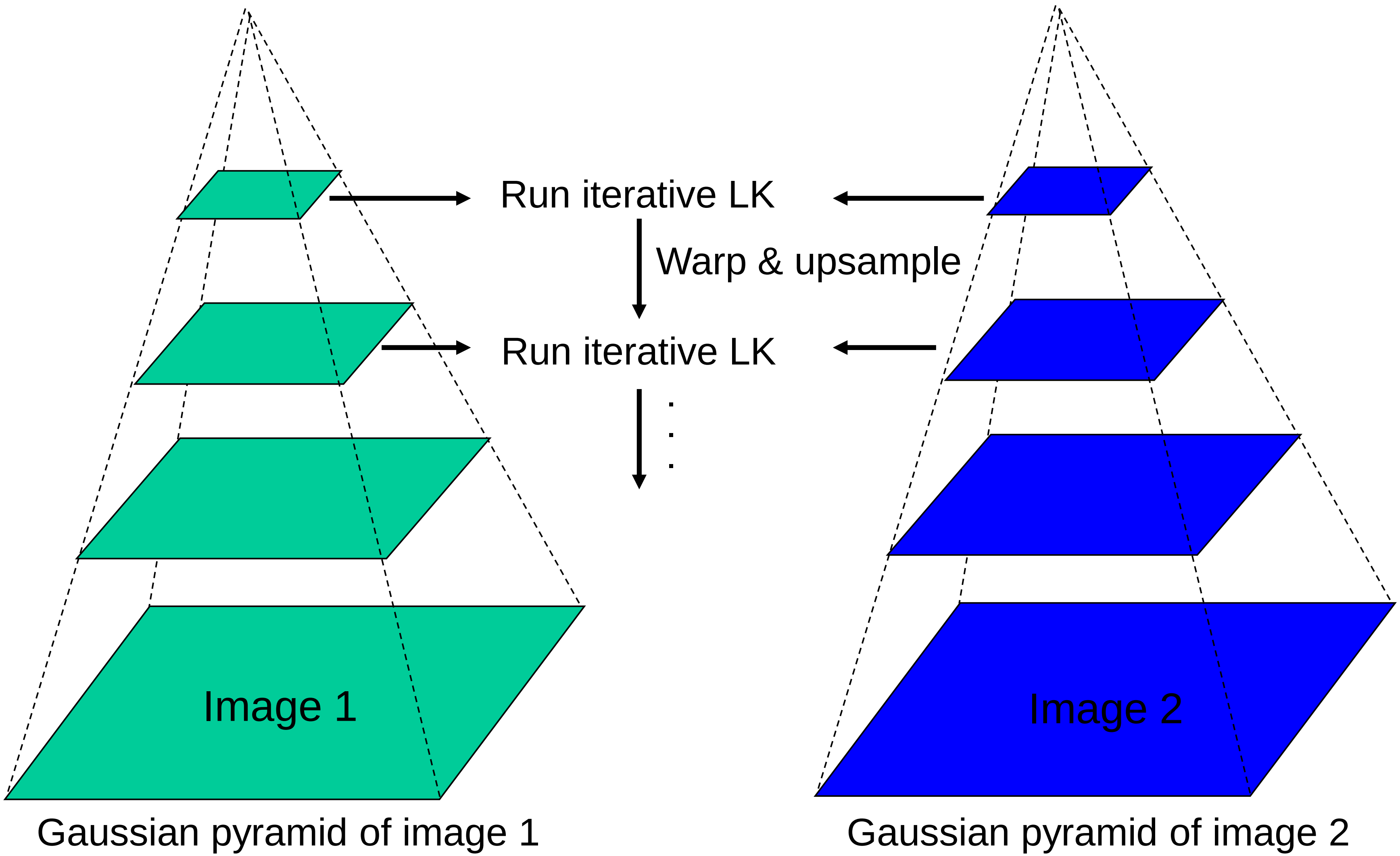


Gaussian pyramid of image 1

Gaussian pyramid of image 2

Slide credit: Steve Seitz

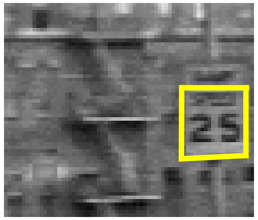
Recap: Coarse-to-fine Optical Flow Estimation

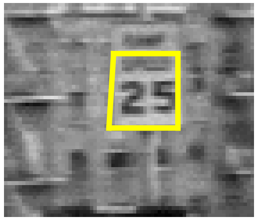


Slide credit: Steve Seitz

Recap: Shi-Tomasi Feature Tracker (→KLT)

- Idea
  - Find good features using eigenvalues of second-moment matrix
  - Key idea: “good” features to track are the ones that can be tracked reliably.





- Frame-to-frame tracking
  - Track with LK and a pure *translation* motion model.
  - More robust for small displacements, can be estimated from smaller neighborhoods (e.g., 5×5 pixels).

- Checking consistency of tracks
  - *Affine* registration to the first observed feature instance.
  - Affine model is more accurate for larger displacements.
  - Comparing to the first frame helps to minimize drift.

J. Shi and C. Tomasi. Good Features to Track. CVPR 1994.

Topics of This Lecture

- Lucas-Kanade Optical Flow
  - Brightness Constancy constraint
  - LK flow estimation
  - Coarse-to-fine estimation
- Feature Tracking
  - KLT feature tracking
- Template Tracking
  - LK derivation for templates
  - Warping functions
  - General LK image registration
- Applications

Lucas-Kanade Template Tracking

- Traditional LK
  - Typically run on small, corner-like features (e.g., $5{\times}5$ patches) to compute optical flow ($\rightarrow$ KLT).
  - However, there is no reason why we can't use the same approach on a larger window around the tracked object.

Slide credit: Robert Collins

Basic LK Derivation for Templates

$$E(u,v) = \sum_{\mathbf{x}} \left[I(x+u, y+v) - T(x,y) \right]^2$$

Current frame

Template model

(u,v) = hypothesized location of template in current frame

Slide credit: Robert Collins

Basic LK Derivation for Templates

- Taylor expansion

$$E(u,v) = \sum_{\mathbf{x}} \left[I(x+u, y+v) - T(x,y) \right]^2$$

$$\approx \sum_{\mathbf{x}} \left[I(x,y) + uI_x(x,y) + vI_y(x,y) - T(x,y) \right]^2$$

$$= \sum_{\mathbf{x}} \left[uI_x(x,y) + vI_y(x,y) + D(x,y) \right]^2 \quad \text{with} \quad D = I - T$$

- Taking partial derivatives

$$\frac{\partial E}{\partial u} = 2 \sum_{\mathbf{x}} \left[uI_x(x,y) + vI_y(x,y) + D(x,y) \right] I_x(x,y) \stackrel{!}{=} 0$$

$$\frac{\partial E}{\partial v} = 2 \sum_{\mathbf{x}} \left[uI_x(x,y) + vI_y(x,y) + D(x,y) \right] I_y(x,y) \stackrel{!}{=} 0$$

- Equation in matrix form

$$\sum_{\mathbf{x}} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \sum_{\mathbf{x}} \begin{bmatrix} I_x D \\ I_y D \end{bmatrix} \quad \Rightarrow$$

Solve via least-squares

Slide credit: Robert Collins

One Problem With This...

- Problematic Assumption
  - Assumption of constant flow (pure translation) for all pixels in a larger window is unreasonable for long periods of time.

- However...
  - We can easily generalize the LK approach to other 2D parametric motion models (like affine or projective) by introducing a “warp” function $\mathbf{W}$ with parameters $\mathbf{p}$.

$$E(u,v) = \sum_{\mathbf{x}} \left[I(x+u, y+v) - T(x,y) \right]^2$$

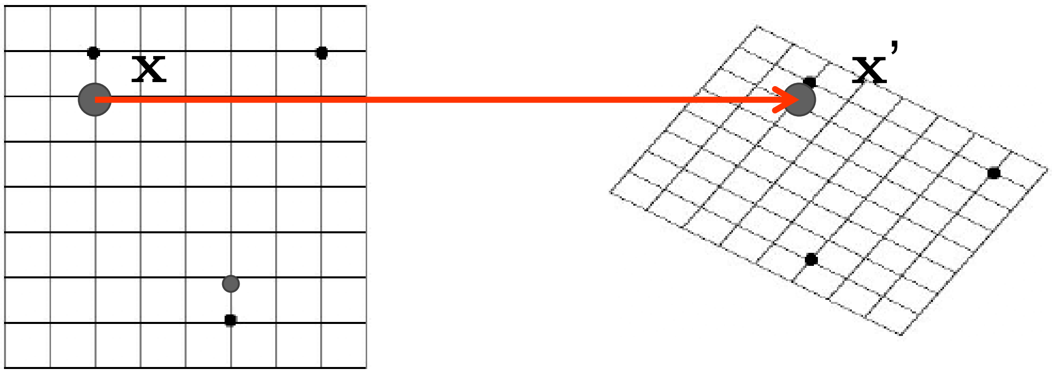
$$\downarrow$$

$$E(\mathbf{p}) = \sum_{\mathbf{x}} \left[I(\mathbf{W}([x,y];\mathbf{p})) - T([x,y]) \right]^2$$

Slide credit: Robert Collins

Geometric Image Warping

- The warp $\mathbf{W}(\mathbf{x};\ \mathbf{p})$ describes the geometric relationship between two images



Input Image

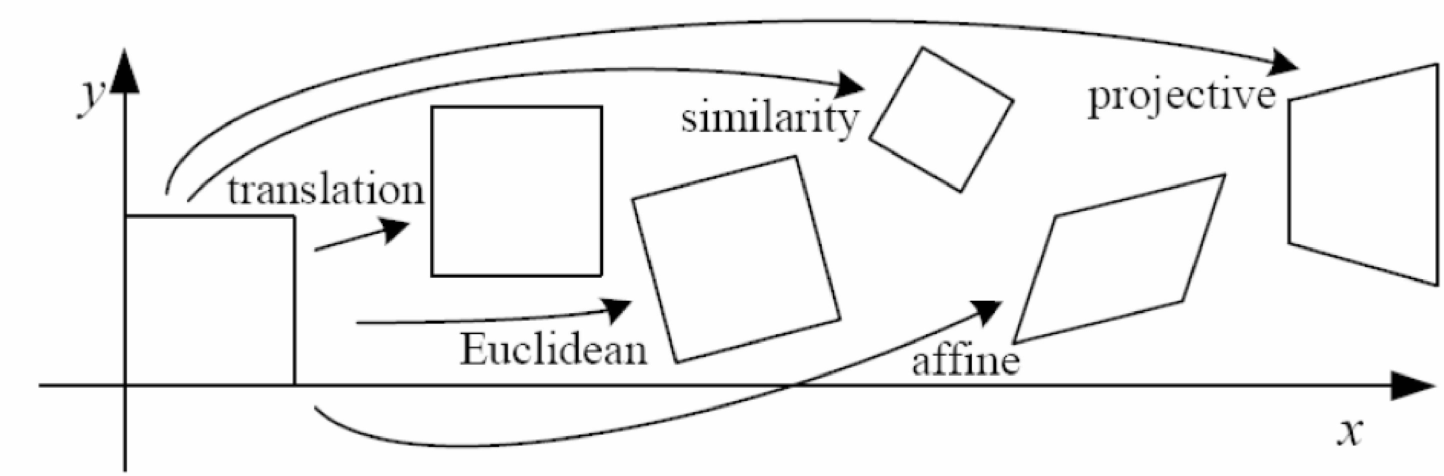
Transformed Image

$$\mathbf{x}' = \mathbf{W}(\mathbf{x}; \boxed{\mathbf{p}}) = \begin{bmatrix} W_x(\mathbf{x}; \mathbf{p}) \\ W_y(\mathbf{x}; \mathbf{p}) \end{bmatrix}$$

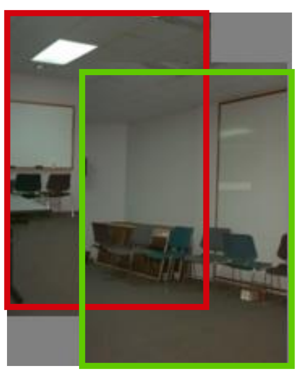
Parameters of the warp

Slide credit: Jinxiang Chai

Example Warping Functions

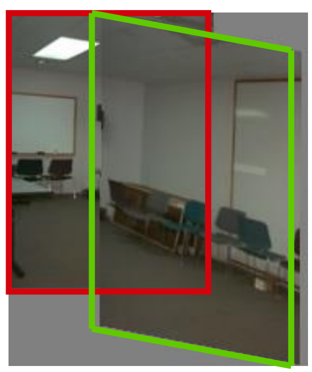


Translation



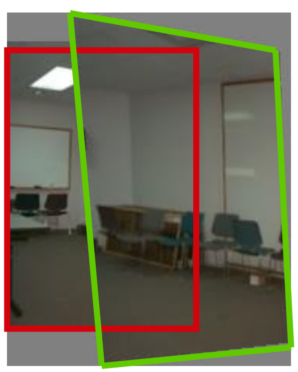
2 unknowns

Affine



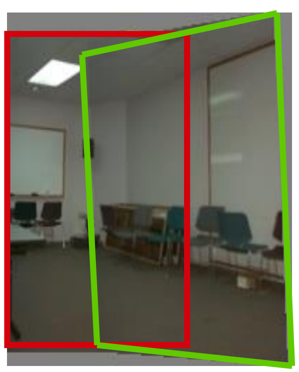
6 unknowns

Perspective



8 unknowns

3D rotation



3 unknowns

Slide credit: Steve Seitz

Example Warping Functions

- Translation

$$\mathbf{W}([x,y];\mathbf{p}) = \begin{bmatrix} x+p_1 \\ y+p_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & p_1 \\ 0 & 1 & p_2 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Affine

$$\mathbf{W}([x,y];\mathbf{p}) = \begin{bmatrix} x+p_1x+p_3y+p_5 \\ y+p_2x+p_4y+p_6 \end{bmatrix} = \begin{bmatrix} 1+p_1 & p_3 & p_5 \\ p_2 & 1+p_4 & p_6 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Perspective

$$\mathbf{W}([x,y];\mathbf{p}) = \frac{1}{p_7x+p_8y+1} \begin{bmatrix} x+p_1x+p_3y+p_5 \\ y+p_2x+p_4y+p_6 \end{bmatrix}$$

  – Note: Other parametrizations are possible; the above ones are just particularly convenient here.

Slide credit: Jinxiang Chai

General LK Image Registration

- Goal
  - Find the warping parameters $\mathbf{p}$ that minimize the sum-of-squares intensity difference between the template image and the warped input image.

- LK formulation
  - Formulate this as an optimization problem

$$\arg\min_{\mathbf{p}} \sum_{\mathbf{x}} \left[I(\mathbf{W}(\mathbf{x};\mathbf{p})) - T(\mathbf{x}) \right]^2$$

  - We assume that an initial estimate of $\mathbf{p}$ is known and iteratively solve for increments to the parameters $\Delta\mathbf{p}$:

$$\arg\min_{\Delta\mathbf{p}} \sum_{\mathbf{x}} \left[I(\mathbf{W}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p})) - T(\mathbf{x}) \right]^2$$

Step-by-Step Derivation

- Key to the derivation
  - Taylor expansion around $\Delta\mathbf{p}$

$$I(\mathbf{W}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p})) \approx I(\mathbf{W}(\mathbf{x};\mathbf{p})) + \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta\mathbf{p} + \mathcal{O}(\Delta\mathbf{p}^2)$$

  - Using pixel coordinates $\mathbf{x} = [x,y]$

$$\begin{aligned} I(\mathbf{W}([x,y];\mathbf{p}+\Delta\mathbf{p})) \approx\ & I(\mathbf{W}([x,y];p_1,\ldots,p_n)) \\ & + \left[\frac{\partial I}{\partial x}\frac{\partial W_x}{\partial p_1} + \frac{\partial I}{\partial y}\frac{\partial W_y}{\partial p_1}\right]_{p_1} \Delta p_1 \\ & + \left[\frac{\partial I}{\partial x}\frac{\partial W_x}{\partial p_2} + \frac{\partial I}{\partial y}\frac{\partial W_y}{\partial p_2}\right]_{p_1} \Delta p_2 \\ & + \ \ldots \\ & + \left[\frac{\partial I}{\partial x}\frac{\partial W_x}{\partial p_n} + \frac{\partial I}{\partial y}\frac{\partial W_y}{\partial p_n}\right]_{p_n} \Delta p_n \end{aligned}$$

Slide credit: Robert Collins

Step-by-Step Derivation

- Rewriting this in matrix notation

$$
\begin{aligned}
I(\mathbf{W}([x,y];\mathbf{p}+\Delta\mathbf{p})) \approx\ & I(\mathbf{W}([x,y];p_1,\ldots,p_n)) \\
& + \begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial W_x}{\partial p_1} \\ \frac{\partial W_y}{\partial p_1} \end{bmatrix}_{p_1} \Delta p_1 \\
& + \begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial W_x}{\partial p_2} \\ \frac{\partial W_y}{\partial p_2} \end{bmatrix}_{p_2} \Delta p_2 \\
& + \ \ldots \\
& + \begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial W_x}{\partial p_n} \\ \frac{\partial W_y}{\partial p_n} \end{bmatrix}_{p_n} \Delta p_n
\end{aligned}
$$

Slide credit: Robert Collins

Step-by-Step Derivation

- And further collecting the derivative terms

$$I(\mathbf{W}([x,y];\mathbf{p}+\Delta\mathbf{p})) \approx I(\mathbf{W}([x,y];p_1,\ldots,p_n))$$

$$+\begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial W_x}{\partial p_1} & \frac{\partial W_x}{\partial p_2} & \cdots & \frac{\partial W_x}{\partial p_n} \\ \frac{\partial W_y}{\partial p_1} & \frac{\partial W_y}{\partial p_2} & \cdots & \frac{\partial W_y}{\partial p_n} \end{bmatrix} \begin{bmatrix} \Delta p_1 \\ \Delta p_2 \\ \vdots \\ \Delta p_n \end{bmatrix}$$

Gradient	Jacobian	Increment parameters to solve for
∇I	$\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$	$\Delta \mathbf{p}$

- Written in matrix form

$$I(\mathbf{W}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p})) \approx I(\mathbf{W}(\mathbf{x};\mathbf{p})) + \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta\mathbf{p}$$

Slide credit: Robert Collins

Example: Jacobian of Affine Warp

- General equation of Jacobian

$$\frac{\partial \mathbf{W}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial W_x}{\partial p_1} & \frac{\partial W_x}{\partial p_2} & \cdots & \frac{\partial W_x}{\partial p_n} \\ \frac{\partial W_y}{\partial p_1} & \frac{\partial W_y}{\partial p_2} & \cdots & \frac{\partial W_y}{\partial p_n} \end{bmatrix}$$

- Affine warp function (6 parameters)

$$\mathbf{W}([x, y]; \mathbf{p}) = \begin{bmatrix} 1 + p_1 & p_3 & p_5 \\ p_2 & 1 + p_4 & p_6 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Result

$$\frac{\partial \mathbf{W}}{\partial \mathbf{p}} = \frac{\partial \begin{bmatrix} x + p_1 x + p_3 y + p_5 \\ p_2 x + y + p_4 y + p_6 \end{bmatrix}}{\partial \mathbf{p}}$$

$$= \begin{bmatrix} x & 0 & y & 0 & 1 & 0 \\ 0 & x & 0 & y & 0 & 1 \end{bmatrix}$$

Slide credit: Robert Collins

Minimizing the Registration Error

- Optimization function after Taylor expansion

$$\arg\min_{\Delta\mathbf{p}} \sum_{\mathbf{x}} \left[I(\mathbf{W}(\mathbf{x};\mathbf{p})) + \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta\mathbf{p} - T(\mathbf{x}) \right]^2$$

- Minimizing this function
  - *How?*

Minimizing the Registration Error

- Optimization function after Taylor expansion

$$\arg\min_{\Delta\mathbf{p}} \sum_{\mathbf{x}} \left[I(\mathbf{W}(\mathbf{x};\mathbf{p})) + \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta\mathbf{p} - T(\mathbf{x}) \right]^2$$

- Minimizing this function
  - Taking the partial derivative and setting it to zero

$$\frac{\partial}{\partial \Delta\mathbf{p}} \stackrel{!}{=} 0 \rightarrow 2 \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]^T \left[I(\mathbf{W}(\mathbf{x};\mathbf{p})) + \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta\mathbf{p} - T(\mathbf{x}) \right] \stackrel{!}{=} 0$$

  - Closed-form solution for $\Delta\mathbf{p}$ (Gauss-Newton):

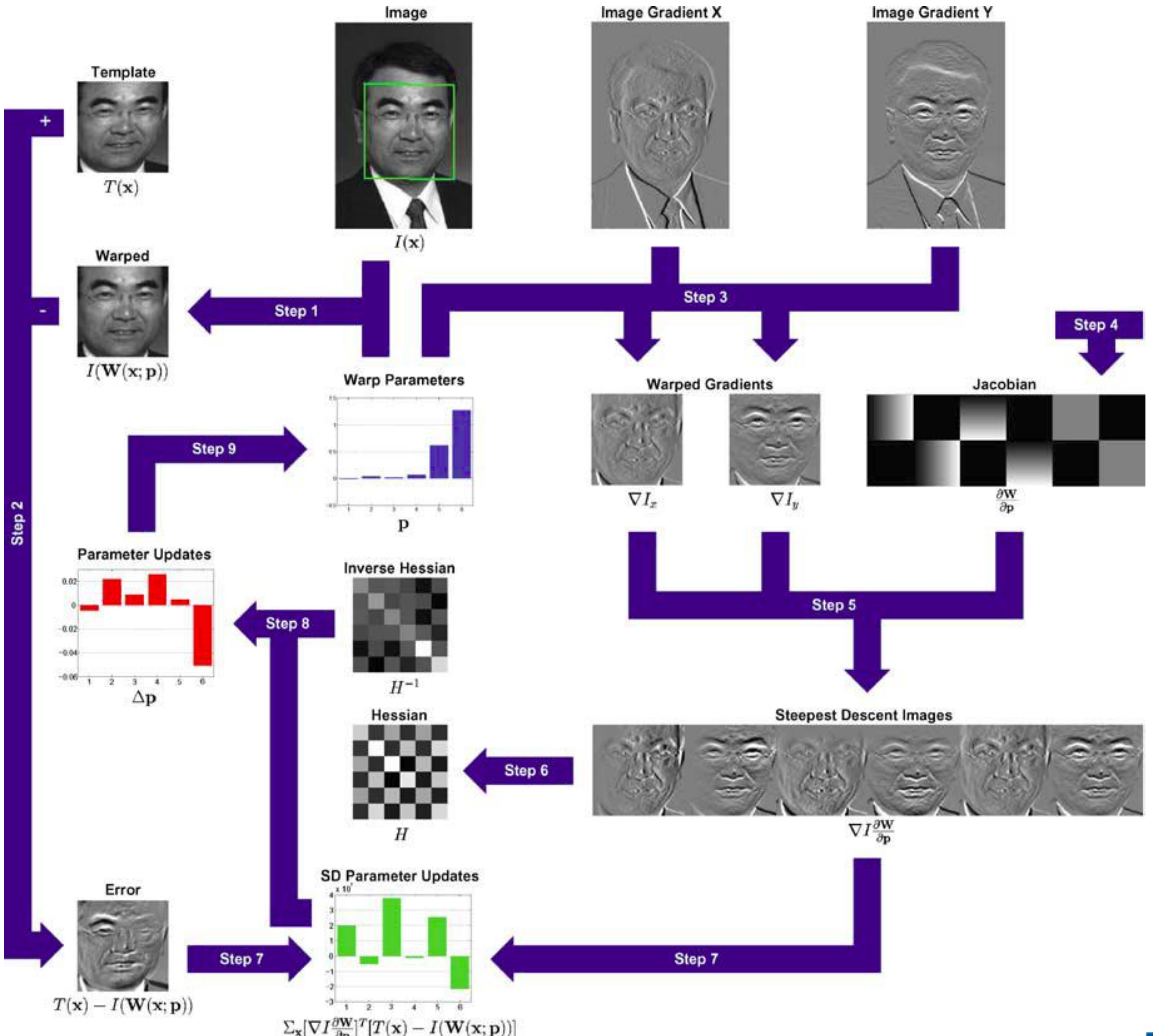
$$\Delta\mathbf{p} = \mathbf{H}^{-1} \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]^T \left[T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p})) \right]$$

  - where $\mathbf{H}$ is the Hessian $\mathbf{H} = \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]^T \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]$

Inverse Compositional LK Algorithm

- Iterate
  - Warp I to obtain $I(\mathbf{W}([x,\ y];\ \mathbf{p}))$
  - Compute the error image $T([x,\ y])$ – $I(\mathbf{W}([x,\ y];\ \mathbf{p}))$
  - Warp the gradient ∇I with $\mathbf{W}([x,\ y];\ \mathbf{p})$
  - Evaluate $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ at $([x,\ y];\ \mathbf{p})$ (Jacobian)
  - Compute steepest descent images $\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}$
  - Compute Hessian matrix $\mathbf{H} = \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]^T \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]$
  - Compute $\sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]^T \left[T([x,y]) - I(\mathbf{W}([x,y];\mathbf{p}))\right]$
  - Compute $\Delta \mathbf{p} = \mathbf{H}^{-1} \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]^T \left[T([x,y]) - I(\mathbf{W}([x,y];\mathbf{p}))\right]$
  - Update the parameters $\mathbf{p} \leftarrow \mathbf{p} + \Delta \mathbf{p}$
- Until $\Delta \mathbf{p}$ magnitude is negligible

Inverse Compositional LK Algorithm Visualization



[S. Baker, I. Matthews, IJCV’04]

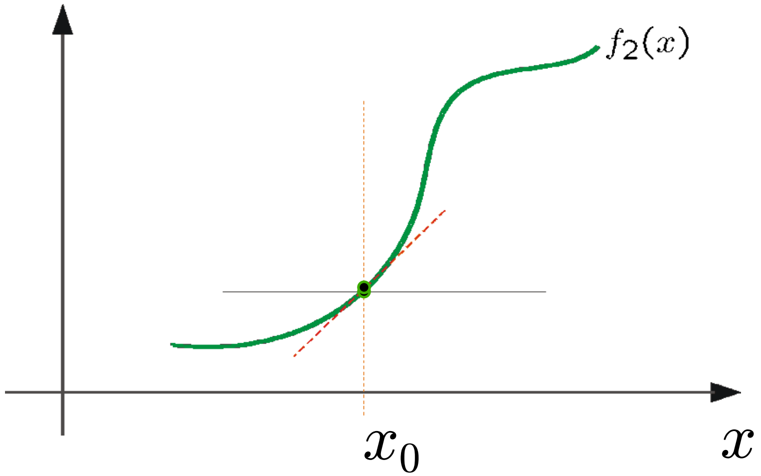
Discussion LK Alignment

- Pros
  - All pixels get used in matching
  - Can get sub-pixel accuracy (important for good mosaicking)
  - Fast and simple algorithm
  - Applicable to Optical Flow estimation, stereo disparity estimation, parametric motion tracking, etc.
- Cons
  - Prone to local minima.
  - Relatively small movement.

  $\Rightarrow$ Good initialization necessary

Side Note

- LK Registration needs a good initialization
  - Taylor expansion corresponds to a linearization around the initial position $\mathbf{p}$.
  - This linearization is only valid in a small neighborhood around $\mathbf{p}$.



- When tracking templates...
  - We typically use the previous frame's result as initialization.

  $\Rightarrow$ The higher the frame rate, the smaller the warp will be.

  $\Rightarrow$ This means we get better results and need fewer LK iterations.

  $\Rightarrow$ *Tracking becomes easier (and faster!) with higher frame rates.*

Discussion

- Beyond 2D Tracking/Registration
  - So far, we focused on registration between 2D images.
  - The same ideas can be used when performing registration between a 3D model and the 2D image (model-based tracking).
  - The approach can also be extended for dealing with articulated objects and for tracking in subspaces.

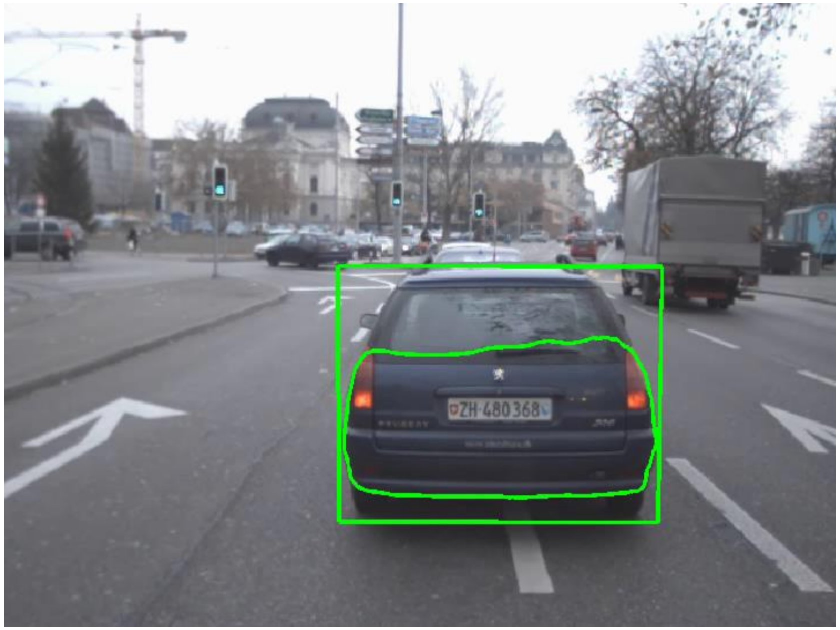
$\Rightarrow$ We will come back to this in later lectures when we talk about model-based 3D tracking...

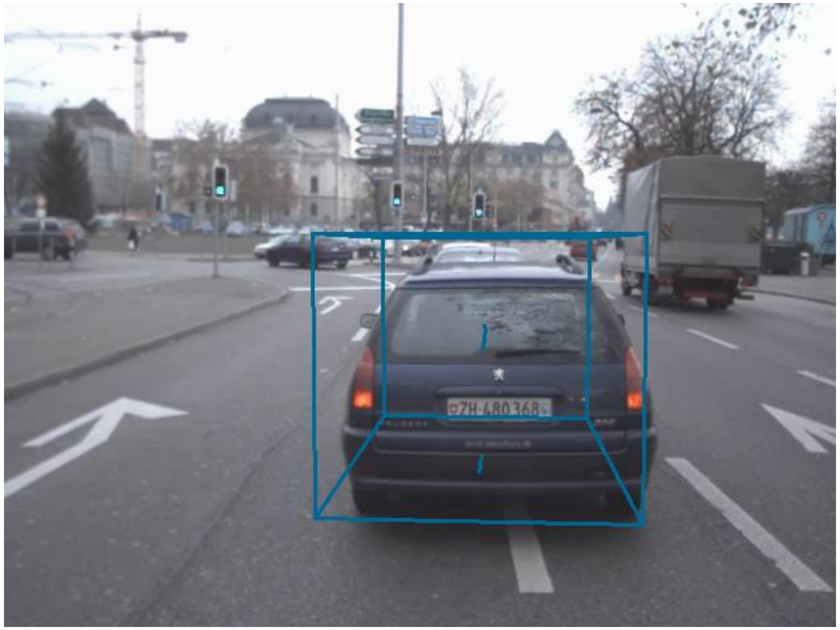
Slide credit: Jinxiang Chai

Topics of This Lecture

- Lucas-Kanade Optical Flow
  - Brightness Constancy constraint
  - LK flow estimation
  - Coarse-to-fine estimation
- Feature Tracking
  - KLT feature tracking
- Template Tracking
  - LK derivation for templates
  - Warping functions
  - General LK image registration
- Applications

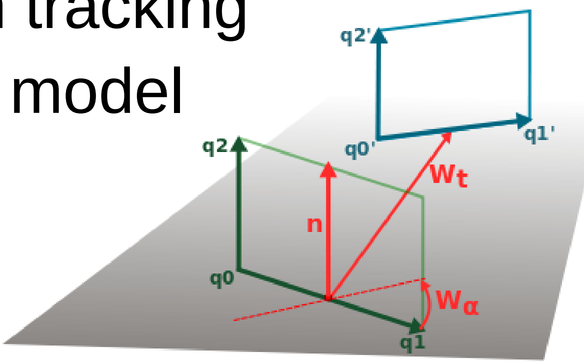
Example of a More Complex Warping Function





- Encode geometric constraints into region tracking
- Constrained homography transformation model
  - Translation parallel to the ground plane
  - Rotation around the ground plane normal
  - $\mathbf{W}(\mathbf{x}) = \mathbf{W}_{obj}\ \mathbf{P}\ \mathbf{W}_t\ \mathbf{W}_\alpha\ \mathbf{Q}\ \mathbf{x}$

$\Rightarrow$ Input for high-level tracker with car steering model.



[E. Horbert, D. Mitzel, B. Leibe, DAGM'10]

References and Further Reading

- The original paper by Lucas & Kanade
  - B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. IJCAI*, pp. 674–679, 1981.
- A more recent paper giving a better explanation
  - S. Baker, I. Matthews. Lucas-Kanade 20 Years On: A Unifying Framework. In IJCV, Vol. 56(3), pp. 221-255, 2004.
- The original KLT paper by Shi & Tomasi
  - J. Shi and C. Tomasi. Good Features to Track. CVPR 1994.