Intelligent Control Systems

Visual Tracking (1)

— Tracking of Feature Points and Planar Rigid Objects —

Shingo Kagami
Graduate School of Information Sciences,
Tohoku University
swk(at)ic.is.tohoku.ac.jp

http://www.ic.is.tohoku.ac.jp/ja/swk/

Outline

- Tracking of a Point
- Block Matching
- Gradient Methods
- Feature Point Detector

Tracking of “a Point”

To track a point
= To determine the motion vector of a point from a frame to its next frame (discrete time)
$\simeq$ To determine the velocity vector of a point (continuous time)

- Distribution of the motion vectors over the image is called optical flow
  - dense optical flow
  - sparse optical flow

- Sometimes the terms “motion vector” and “optical flow” are used interchangeably (depending on the context)

Optical Flow Constraint

Assuming that the intensity of the tracked point is constant,

$$I(x, y, t) = I(x + dx, y + dy, t + dt)$$
$$= I(x, y, t) + \frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy + \frac{\partial I}{\partial t}dt + \epsilon$$

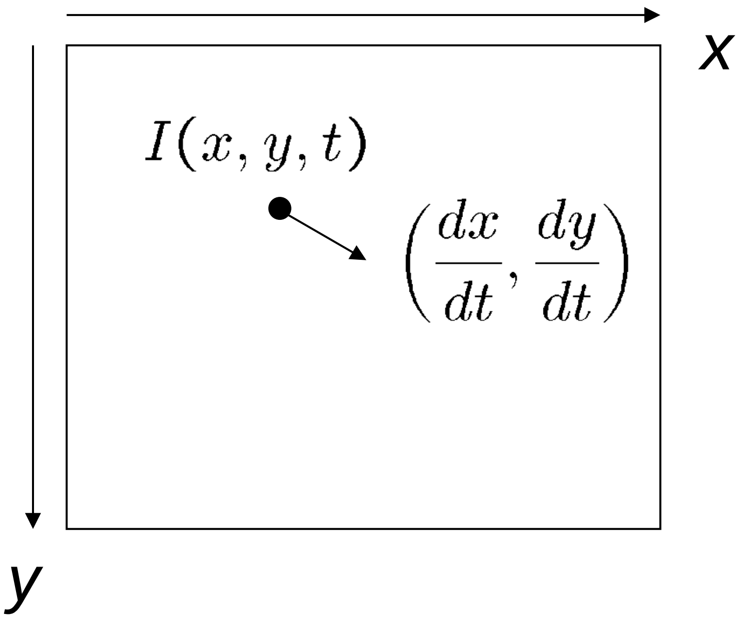
Note: we don't distinguish infinitesimal dx and finite Δx in today's lecture note

Ignoring the 2nd order or higher terms ϵ yields

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

This single equation is not enough to determine the two components $\left(\frac{dx}{dt}, \frac{dy}{dt}\right)$

[Horn and Schunck 1981]



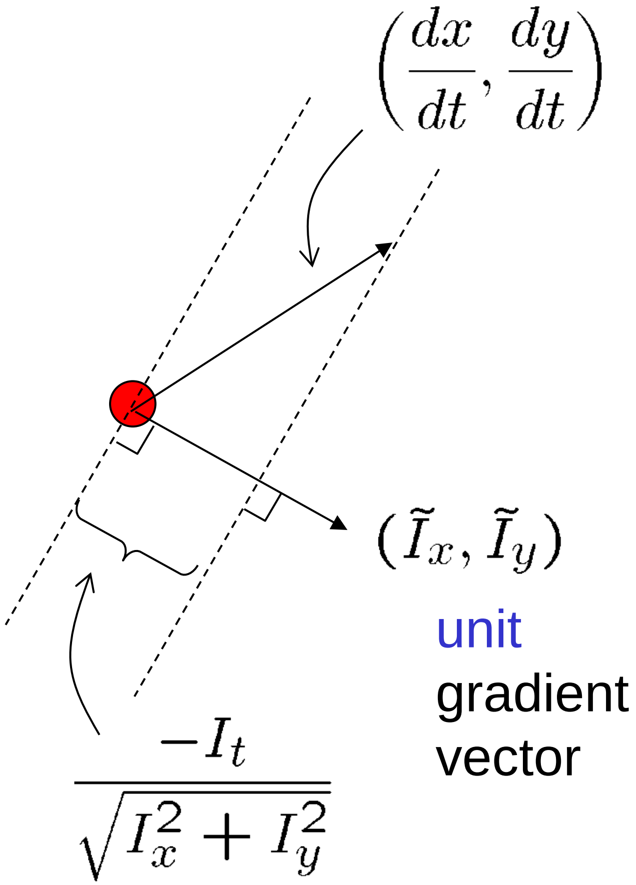
Interpretation of the Constraint

With $\dfrac{\partial I}{\partial x} = I_x,\ \dfrac{\partial I}{\partial y} = I_y,\ \dfrac{\partial I}{\partial t} = I_t$

$$(I_x, I_y)\left(\frac{dx}{dt}, \frac{dy}{dt}\right)^T = -I_t$$

$$(\tilde{I}_x, \tilde{I}_y)\left(\frac{dx}{dt}, \frac{dy}{dt}\right)^T = \frac{-I_t}{\sqrt{I_x^2 + I_y^2}}$$

where $(\tilde{I}_x, \tilde{I}_y)$ is a unit vector parallel to (I_x, I_y)



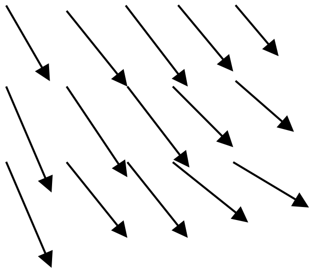
Only the component in the direction of the gradient vector is determined (aperture problem)

Additional Assumptions

Thus we cannot determine the optical flow from I_x, I_y and I_t. Additional assumptions are needed.

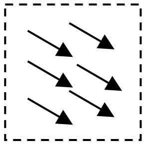
ex1) Optical flow changes smoothly in space

- [Horn and Schunck 1981]



ex2) Optical flow is constant within a small neighborhood of a point

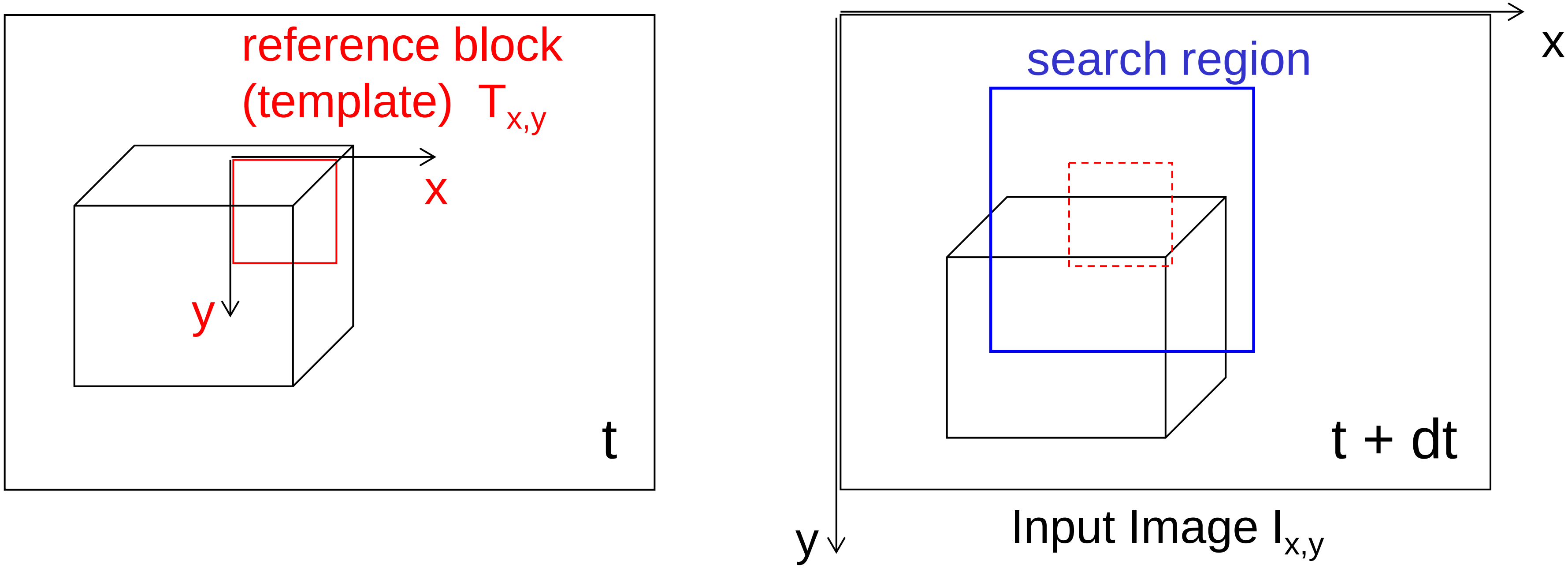
- We will investigate this in the followings
- So, what we call “point tracking” is actually “patch (block) tracking”



Outline

- Tracking of a Point
- Block Matching
- Gradient Methods
- Feature Point Detector

Block Matching



(When you are sure the incoming images only translate, you can use a fixed T. Otherwise T is updated every frame)

Slides reference block through search region and compare

- How to compare?: by computing evaluation functions

Evaluation Functions

$$d_{\mathbf{SSD}}(x,y) = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(T_{i,j} - I_{x+i,y+j})^2$$

: sum of squared differences $\rightarrow$ min

$$d_{\mathbf{SAD}}(x,y) = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}|T_{i,j} - I_{x+i,y+j}|$$

: sum of absolute differences $\rightarrow$ min

$$C(x,y) = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}T_{i,j}I_{x+i,y+j}$$

: cross correlation $\rightarrow$ max

$$C_{\mathbf{n}}(x,y) = \frac{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(T_{i,j} - \bar{T})(I_{x+i,y+j} - \bar{I}_{x,y})}{\sqrt{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(T_{i,j} - \bar{T})^2 \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(I_{x+i,y+j} - \bar{I}_{x,y})^2}}$$

average ($\bar{I}_{x,y}$)

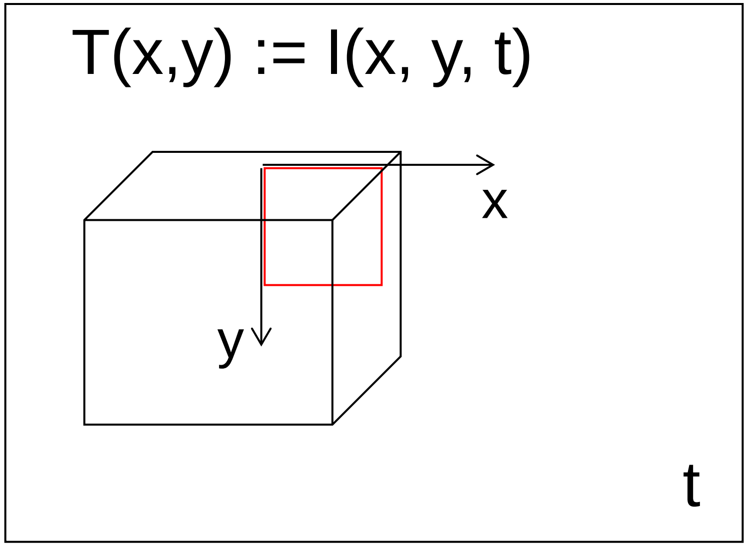
: normalized cross correlation $\rightarrow$ max

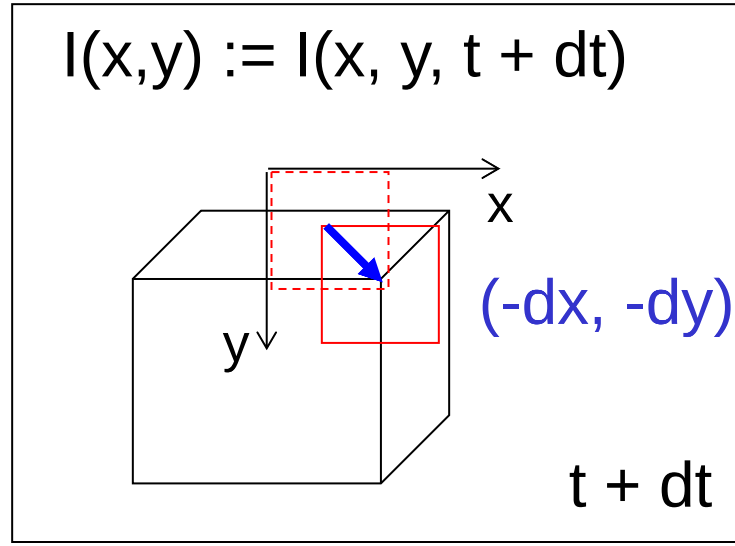
Note: $T_{x,y}$ and $I_{x,y}$ are short for T(x,y) and I(x,y) here (Not derivative!)

Outline

- Tracking of a Point
- Block Matching
- Gradient Methods
- Feature Point Detector

Utilize Gradients to Explore the Solution





From the intensity constancy assumption,

$$T(x, y) = I(x - dx, y - dy)$$

Achieve this by minimizing SSD:

$$E(dx, dy) = \sum_{u,v} \{T(x + dx + u, y + dy + v) - I(x + u, y + v)\}^2$$

Using 1st order Taylor expansion of T(x, y),

$$E(dx, dy) = \sum_{u,v} \{T(x+u, y+v) + T_x(x+u, y+v)dx$$

$$+ T_y(x+u, y+v)dy - I(x+u, y+v)\}^2$$

$$= \sum \{e + T_x dx + T_y dy\}^2 \quad (e := T - I)$$

$$= \sum \left\{ e + (T_x, T_y) \begin{pmatrix} dx \\ dy \end{pmatrix} \right\}^2$$

To minimize E, derivative of E w.r.t. (dx, dy) is equated to 0

$$\sum \left\{ e + (T_x, T_y) \begin{pmatrix} dx \\ dy \end{pmatrix} \right\} \begin{pmatrix} T_x \\ T_y \end{pmatrix} = 0$$

$$\sum \begin{pmatrix} T_x^2 & T_x T_y \\ T_x T_y & T_y^2 \end{pmatrix} \begin{pmatrix} dx \\ dy \end{pmatrix} = -\sum e \begin{pmatrix} T_x \\ T_y \end{pmatrix}$$

$$\begin{pmatrix} \sum T_x^2 & \sum T_x T_y \\ \sum T_x T_y & \sum T_y^2 \end{pmatrix} \begin{pmatrix} dx \\ dy \end{pmatrix} = - \begin{pmatrix} \sum (T - I) T_x \\ \sum (T - I) T_y \end{pmatrix}$$

$$H \begin{pmatrix} dx \\ dy \end{pmatrix} = -\boldsymbol{g}$$

Then, (dx, dy) is obtained by solving this linear equation (Lucas-Kanade method [Lucas and Kanade 1981])

- (dx, dy) is only approximately obtained because of the 1^{st} order Taylor approximation. We often need to iteratively run the above process by setting I(x, y) := I(x - dx, y - dy) to obtain a good result

Gauss-Newton Method

Lucas-Kanade method can be viewed as an application of Gauss-Newton method (an iterative non-linear optimization method for least square problems).

$$\min_{\boldsymbol{x}} ||\boldsymbol{e}(\boldsymbol{x})||^2 = \min_{\boldsymbol{x}} ||\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{s}||^2$$

$$\begin{aligned} ||\boldsymbol{e}(\boldsymbol{x}_0 + d\boldsymbol{x})||^2 &\simeq ||\boldsymbol{f}(\boldsymbol{x}_0) + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} d\boldsymbol{x} - \boldsymbol{s}||^2 \\ &= ||\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} d\boldsymbol{x} + \boldsymbol{e}(\boldsymbol{x}_0)||^2 = ||J d\boldsymbol{x} + \boldsymbol{e}_0||^2 \end{aligned}$$

$$\frac{\partial}{\partial \boldsymbol{x}} ||\boldsymbol{e}(\boldsymbol{x})||^2|_{\boldsymbol{x}=\boldsymbol{x}_0} = (J d\boldsymbol{x} + \boldsymbol{e}_0)^T J = 0$$

$$(J^T J) d\boldsymbol{x} = -J^T \boldsymbol{e}_0$$

$$H d\boldsymbol{x} = -\boldsymbol{g}$$

Generalization

Formulations for more general image transformations

$$J = \frac{\partial f}{\partial x} = \frac{\partial f}{\partial w} \cdot \frac{\partial w}{\partial x}$$

x: transform parameters

w(**x**): warp function
(i.e. how x and y coordinates change w.r.t. params)

e.g.

- translation + rotation (3 dof)
- translation + rotation + magnification (4 dof)
- affine transformation (6 dof)
- perspective transformation (8 dof)

Generalization

Other optimization methods

- Levenberg-Marquardt method

$$(J^T J + \lambda I)d\boldsymbol{x} = -J^T \boldsymbol{e}_0$$

I: identity matrix
λ: scalar coefficient
(small λ: Gauss-Newton, large λ: steepest descent)

- Efficient Second-order Minimization method [Banhimane and Malis 2007]

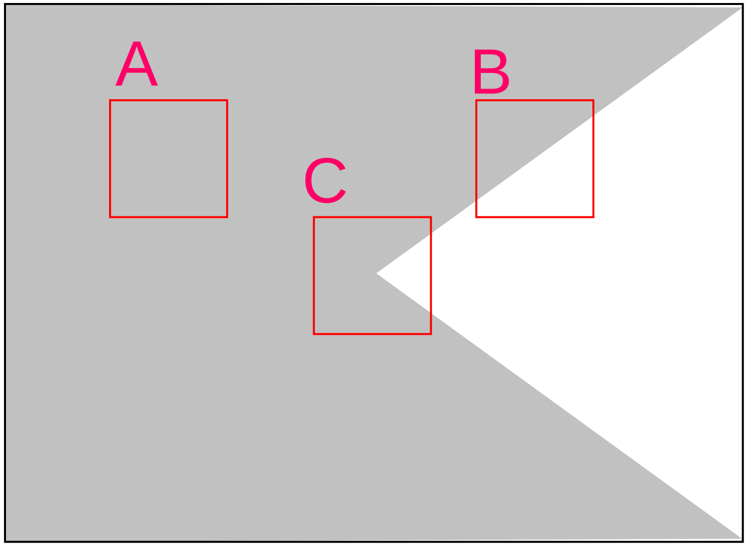
$$(J^T J)d\boldsymbol{x} = -J^T \boldsymbol{e}_0,\ J = (J_1 + J_2)/2$$

J_1: derivative of template image w.r.t. param.
J_2: derivative of current warped image w.r.t. param.
(Possible when parametrized with special care)

Outline

- Tracking of a Point
- Block Matching
- Gradient Methods
- Feature Point Detector

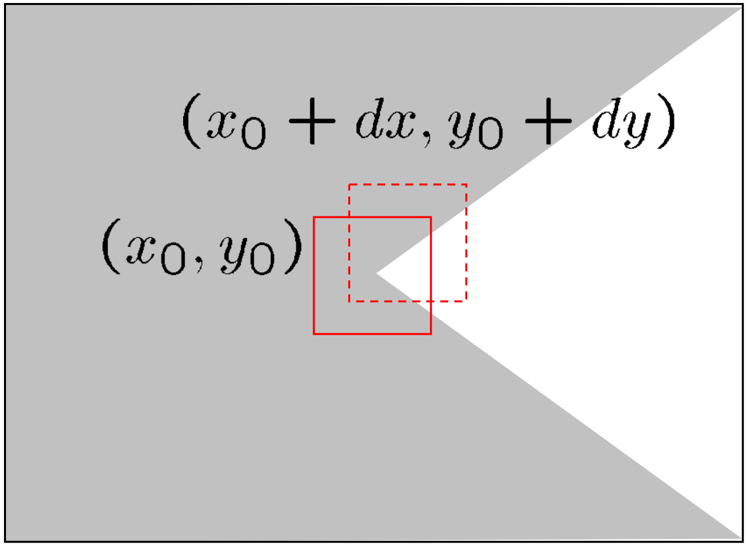
What is Good Point to Track



Recall that we aggregate many flows within a small block to obtain enough constraints

- A: Block with constant intensity is not suitable (0 constraint)
- B: Block including only edges with the same direction is also not suitable (essentially 1 constraint)

How to find a block like C?



Consider two blocks

- around a point of interest (x_0, y_0)
- around the point $(x_0 + dx, y_0 + dy)$

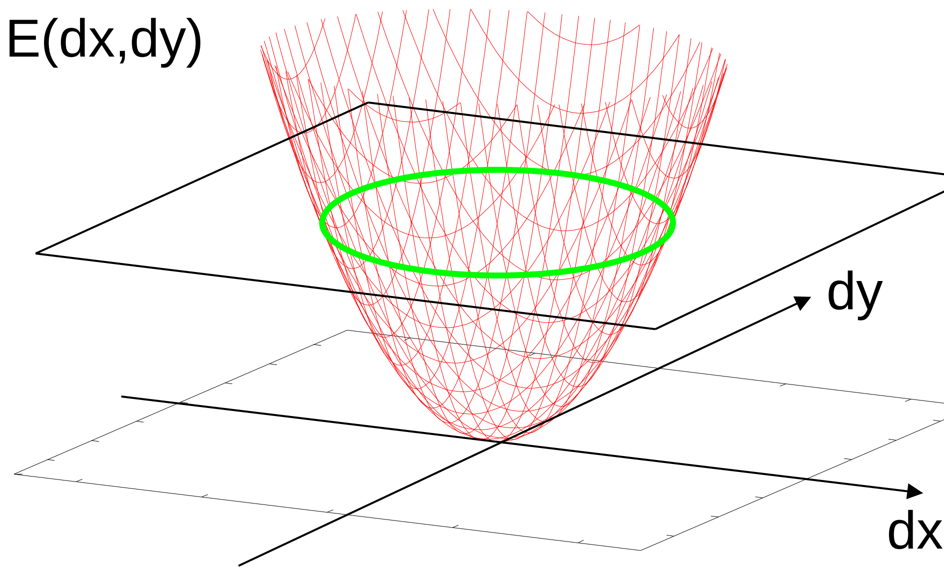
These two blocks should not resemble each other for any choice of (dx, dy)

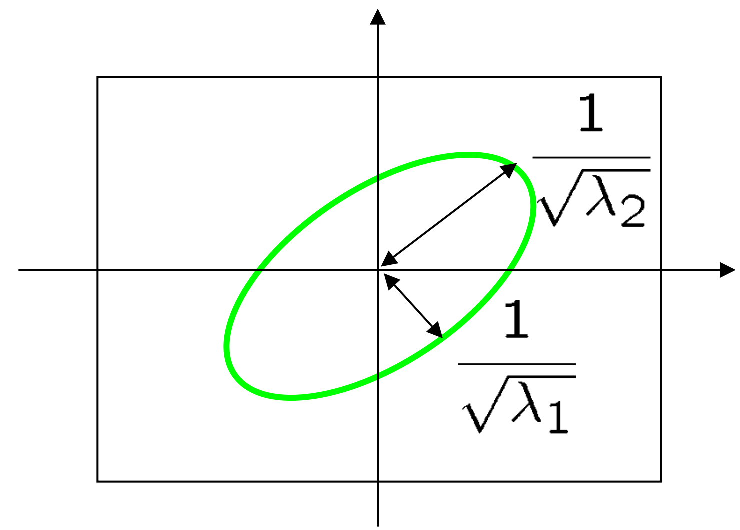
Let's measure how they do not resemble by SSD

$$E(dx, dy) \equiv \sum_{u,v} \{I(x_0 + dx + u, y_0 + dy + v) - I(x_0 + u, y_0 + v)\}^2$$

With 1st order Taylor expansion,

$$\begin{aligned} E(dx, dy) &= \sum_{u,v} \{I_x(x_0 + u, y_0 + v)dx + I_y(x_0 + u, y_0 + v)dy\}^2 \\ &= \sum I_x^2 dx^2 + 2\sum I_x I_y dx dy + \sum I_y^2 dy^2 \\ &= (dx, dy) \begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix} \begin{pmatrix} dx \\ dy \end{pmatrix} \\ &= (dx, dy) H \begin{pmatrix} dx \\ dy \end{pmatrix} \end{aligned}$$





E(dx,dy) = 1

$$E(dx, dy) = (dx, dy) H \begin{pmatrix} dx \\ dy \end{pmatrix} = 1$$

is an ellipse in (dx, dy) plane. This ellipse should be as small as possible and should be close to true circle.

i.e.: Eigenvalues λ_1, λ_2 of H should be large enough and close to each other.

Compatible with numerical stability in solving $H \begin{pmatrix} dx \\ dy \end{pmatrix} = -\boldsymbol{g}$

(Just in case you forget linear algebra)

$$(dx, dy)H \begin{pmatrix} dx \\ dy \end{pmatrix} = 1$$

Noting that H is symmetric, H can be diagonalized by an orthonormal matrix P (i.e. $P^{-1} = P^T$) so that $P^T H P = \mathrm{diag}(\lambda_1, \lambda_2)$

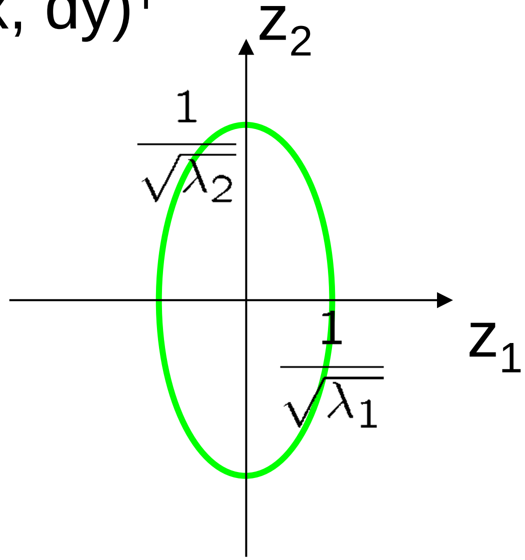
$$(dx, dy)P \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} P^T \begin{pmatrix} dx \\ dy \end{pmatrix} = 1$$

Viewed in a new coordinate system $\mathbf{z} = P^T (dx, dy)^T$

$$z^T \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} z = 1$$

Or, equivalently

$$\lambda_1 z_1^2 + \lambda_2 z_2^2 = 1$$



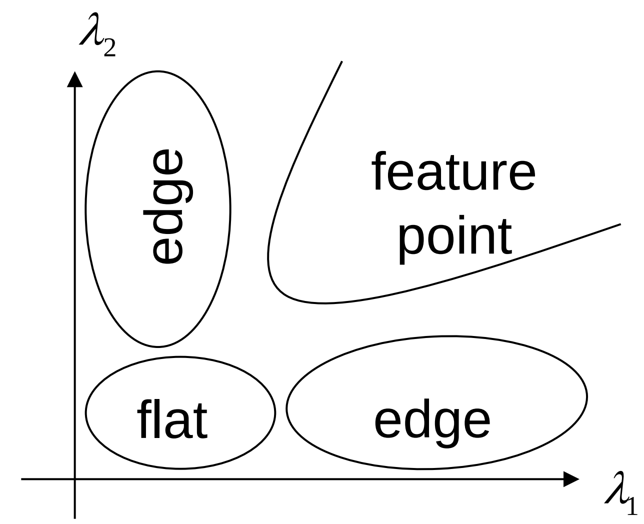
Feature Point Detector

Harris operator

[Harris and Stephens 1988]

$$\det H - k(\operatorname{tr} H)^2$$

$$= \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2$$

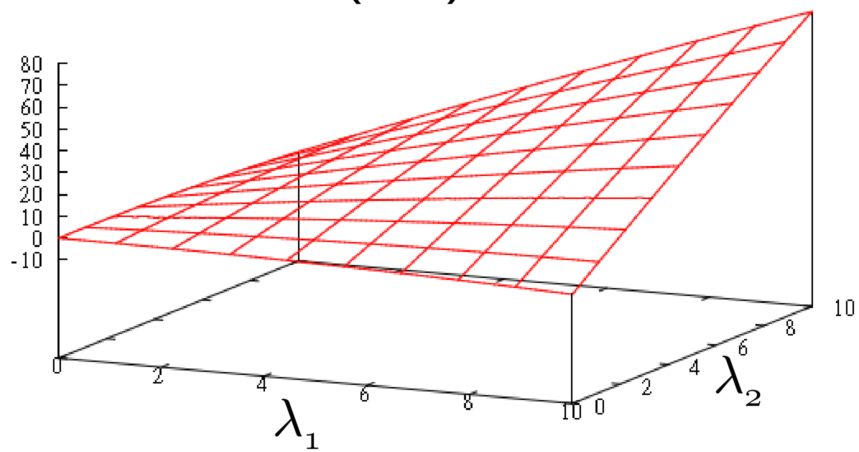


Good Features to Track

[Tomasi and Kanade 1991]

$$\min(\lambda_1, \lambda_2)$$

det H – k(tr H)^2



These “good” points for tracking and/or matching are called feature point, interest point, keypoint and so on.

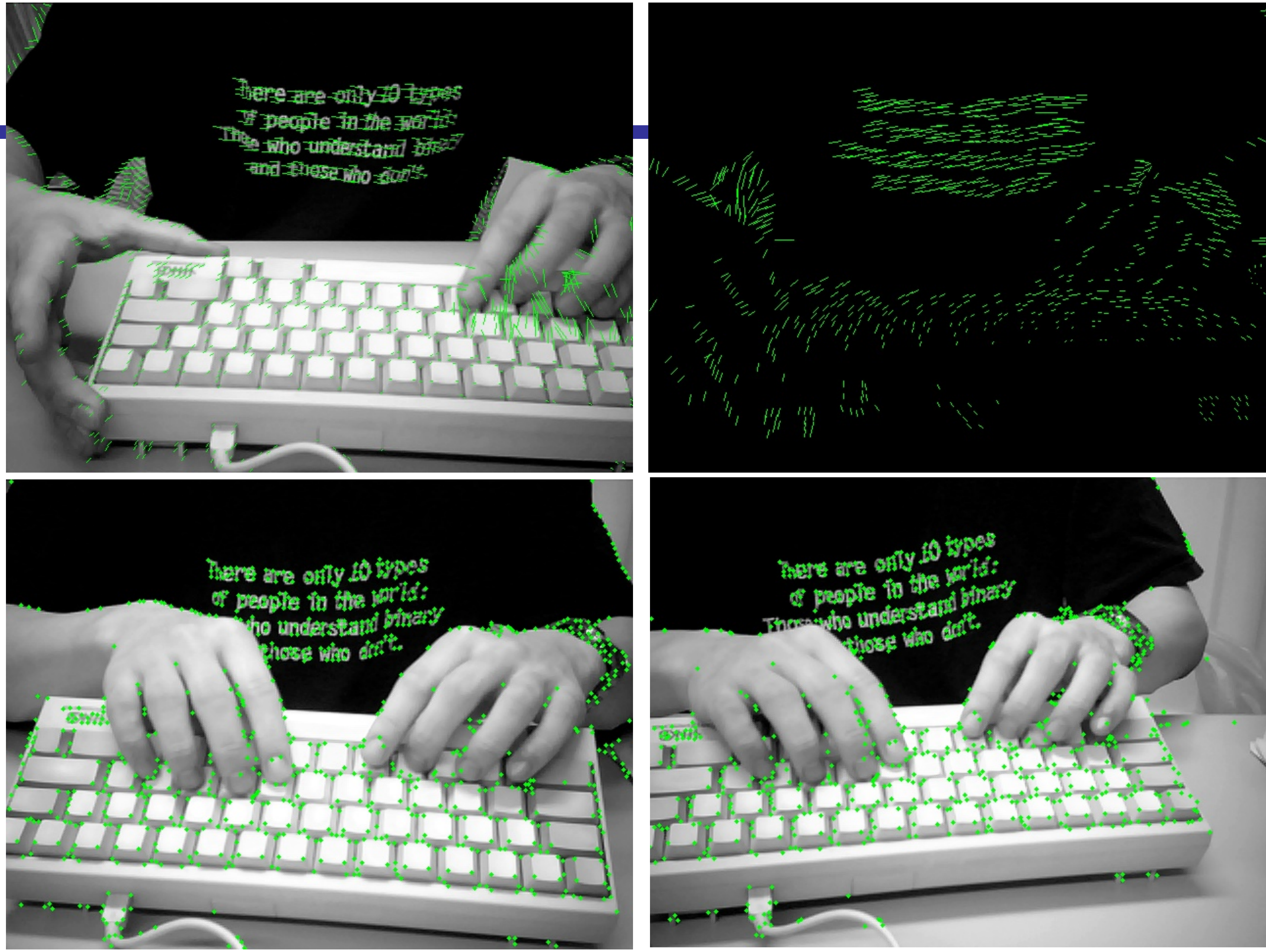
Other Feature Point Detectors

SIFT detector [Lowe 2004]

- Build a Gaussian scale space and apply (an approximate) Laplacian operator in each scale
- Detect extrema of the results (i.e. strongest responses among their neighbor in space as well as in scale)
- Eliminate edge responses
- (Often followed by encoding of edge orientation histogram in the neighborhood into a fixed-size vector, called a feature point descriptor, which can be compared with each other by Euclidean distance)

FAST detector [Rosten et al. 2010]

- Heuristics based on pixel values along a surrounding circle
- Optimized for speed and quality by machine learning approach



Lucas-Kanade method applied to “Good-features-to-track” points is often called KLT (Kanade-Lucas-Tomasi) tracker

Summary

- Tracking of a Point
  - is an ill-posed problem
  - a block is often considered instead of a point

- Block Matching
  - full-search optimization of an evaluation function
    - SSD, SAD, (normalized) cross correlation

- Lucas-Kanade method
  - a Gradient method for optimization of SSD

- Feature Point Detector
  - Harris operator, Good feature to track
  - KLT tracker

References

- B. K. P. Horn and B. G. Schunck: Determining Optical Flow, Artificial Intelligence, vol.17, pp.185-203, 1981.
- C. Harris and M. Stephens: A Combined Corner and Edge Detector, Proc. 14th Alvey Vision Conference, pp.147-151, 1988.
- B. D. Lucas and T. Kanade: An Iterative Image Registration Technique with an Application to Stereo Vision, Proc. 7th International Conference on Artificial Intelligence, pp.674-679, 1981.
- S. Benhimane and E. Malis: Homography-based 2D Visual Tracking and Servoing, International Journal of Robotics Research, vol. 26, no. 7, pp.661-676, 2007.
- D. Lowe, Distinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision, vol. 60, no. 2, pp. 91-110, 2004.
- E. Rosten, R. Porter and T. Drummond: Faster and Better: A Machine Learning Approach to Corner Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 1, pp. 105-119, 2010.
- C. Tomasi and T. Kanade: Detection and Tracking of Point Features, Shape and Motion from Image Streams: a Factorization Method –Part 3, Technical Report CMU-CS-91-132, Carnegie Mellon University, 1991.

Sample codes are available at http://www.ic.is.tohoku.ac.jp/~swk/lecture/