**OPPA European Social Fund**
**Prague & EU: We invest in your future.**

# Patch tracking based on comparing its pixels[1]

**Tomáš Svoboda**, svoboda@cmp.felk.cvut.cz
Czech Technical University in Prague, Center for Machine Perception
http://cmp.felk.cvut.cz

**Last update:** March 22, 2010

### Talk Outline

- comparing patch pixels
- normalized cross-correlation, ssd . . .
- KLT - gradient based optimization
- good features to track

[1] Please note that the lecture will be accompanied be several sketches and derivations on the blackboard and few live-interactive demos in Matlab

## What is the problem?



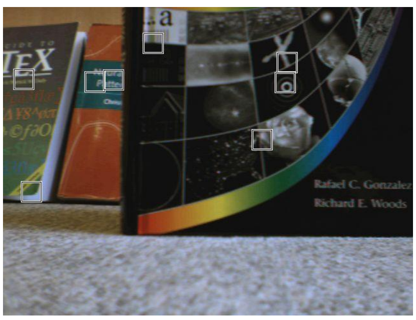
video: CTU campus, door of G building

## Tracking of dense sequences — camera motion

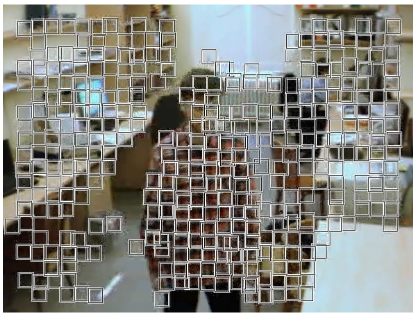


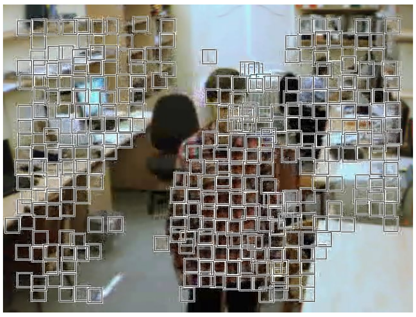
**I**

**J**



## Tracking of dense sequences — object motion






## Alignment of an image (patch)

Goal is to align a template image $T(\mathbf{x})$ to an input image $I(\mathbf{x})$. $\mathbf{x}$ column vector containing image coordinates $[x, y]^\top$. The $I(\mathbf{x})$ could be also a small subwindow withing an image.



## How to measure the aligment?

- What is the best criterial function?
- How to find the best match, in other words how to find extremum of the criterial function?

### Criterial function

What are the desired properties (on a certain domain)?

- convex (remember the optimization course?)
- discriminative
- . . .

## Normalized cross-correlation

You may know it as correlation coefficients

$$r_{ij} = \frac{c_{ij}}{\sqrt{c_{ii}c_{jj}}}$$

where $c_{i,j}$ are elements of the covariance matrix

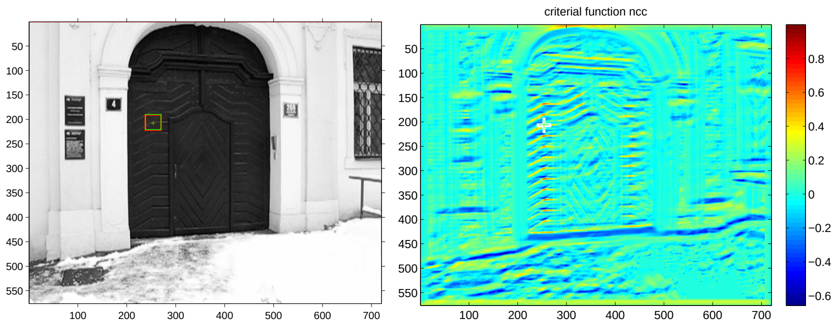
Having template $T(x,y)$ and image $I(x,y)$,

$$r(x,y) = \frac{\sum_k \sum_l \left(T(k,l) - \overline{T}\right)\left(I(x+k,y+l) - \overline{I(x,y)}\right)}{\sqrt{\sum_k \sum_l \left(T(k,l) - \overline{T}\right)^2 \sum_k \sum_l \left(I(x+k,y+l) - \overline{I(x,y)}\right)^2}}$$

Be careful about coordinate systems (sketch on blackboard)



## Normalized cross-correlation – in picture

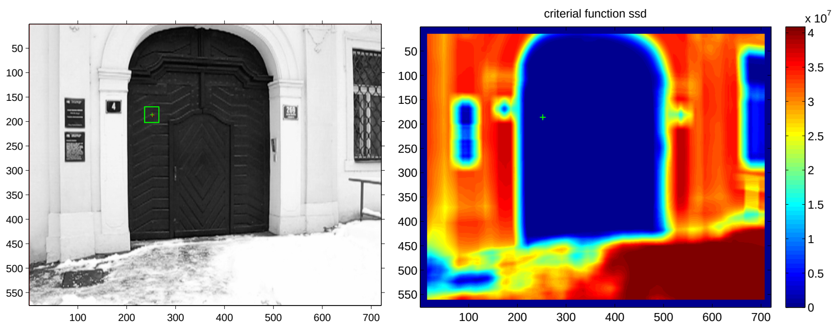



- well, definitely not convex
- but the discriminability looks promissing
- very efficient in computation, see [3].



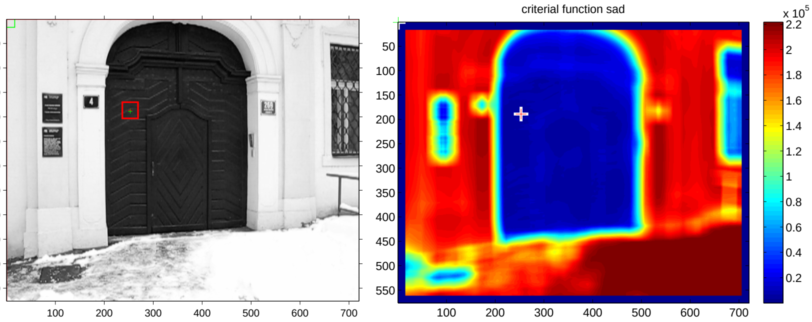
## Sum of squared differences

$$ssd(x,y) = \sum_k \sum_l \left(T(k,l) - I(x+k,y+l)\right)^2$$



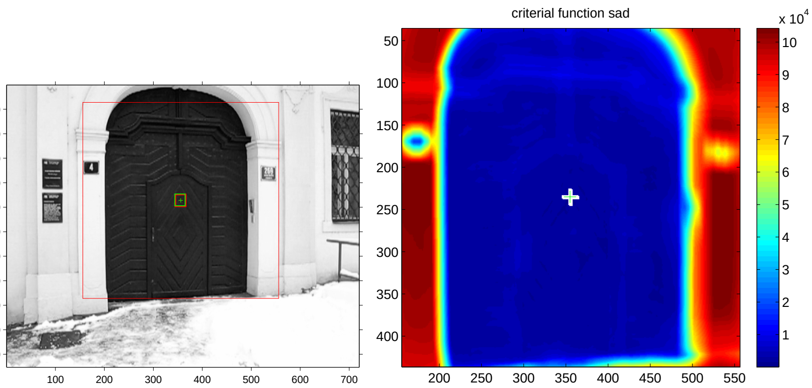
## Sum of absolute differences

$$sad(x, y) = \sum_k \sum_l |T(k, l) - I(x + k, y + l)|$$



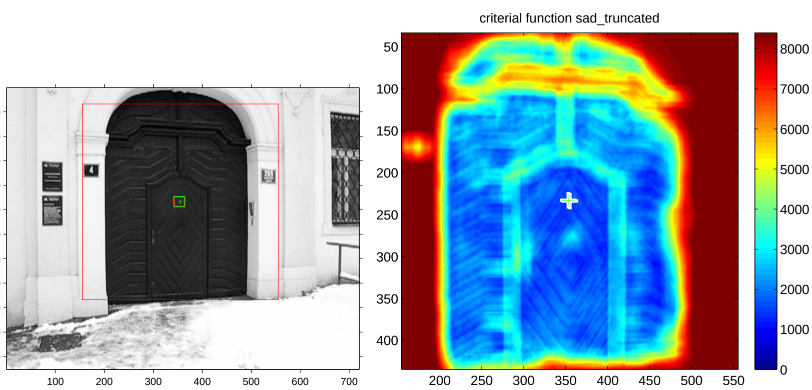



## SAD for the door part






## SAD for the door part – truncated




Differences greater than 20 intensity levels are counted as 20.

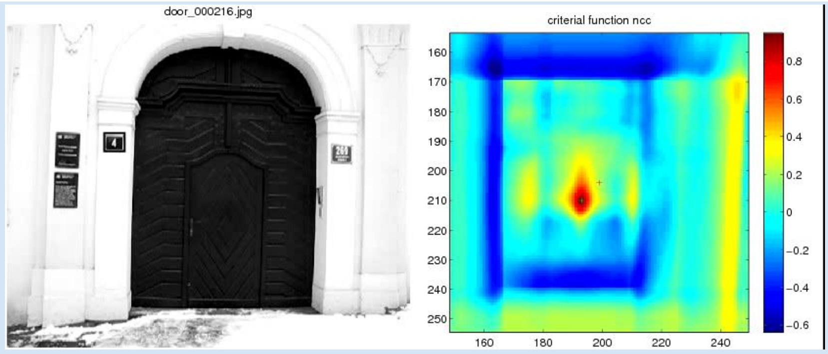
## Normalized cross-correlation: how it works



live demo for various patches

## Normalized cross-correlation: tracking






video

- What went wrong?
- Why did it failed?

**Suggestions for improvement?**

## Iterations? – Tracking as an optimization problem



- finding extrema of a criterial function . . .
- . . . sounds like an optimization problem

**Kanade–Lucas–Tomasi (KLT) tracker**

- Iteratively minimizes sum of square differences.
- It is a Gauss-Newton gradient descent non-linear optimization algorithm.

## Importance in Computer Vision



- Firstly published in 1981 as an image registration method [4].
- Improved many times, most importantly by Carlo Tomasi [5, 6]
- Free implementation(s) available[2]. Also part of the OpenCV library[3].
- After more than two decades, a project[4] at CMU dedicated to this single algorithm and results published in a premium journal [1].
- Part of plethora computer vision algorithms.

Our explanation follows mainly the paper [1]. It is a good reading for those who are also interested in alternative solutions.

[2] http://www.ces.clemson.edu/~stb/klt/
[3] http://opencv.willowgarage.com/wiki/
[4] http://www.ri.cmu.edu/projects/project_515.html

## Original Lucas-Kanade algorithm I



Goal is to align a template image $T(\mathbf{x})$ to an input image $I(\mathbf{x})$. $\mathbf{x}$ column vector containing image coordinates $[x, y]^\top$. The $I(\mathbf{x})$ could be also a small subwindow withing an image.

Set of allowable warps $\mathbf{W}(\mathbf{x};\mathbf{p})$, where $\mathbf{p}$ is a vector of parameters. For translations

$$\mathbf{W}(\mathbf{x};\mathbf{p}) = \begin{bmatrix} x + p_1 \\ y + p_2 \end{bmatrix}$$

$\mathbf{W}(\mathbf{x};\mathbf{p})$ can be arbitrarily complex

The best alignment, $\mathbf{p}^*$, minimizes image dissimilarity

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

## Original Lucas-Kanade algorithm II



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

is a nonlinear optimization! The warp $\mathbf{W}(\mathbf{x};\mathbf{p})$ may be linear but the pixels value are, in general, non-linear. In fact, they are essentially unrelated to $\mathbf{x}$.

It is assumed that some $\mathbf{p}$ is known and best increment $\Delta\mathbf{p}$ is sought. The the modified problem

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

is solved with respect to $\Delta\mathbf{p}$. When found then $\mathbf{p}$ gets updated

$$\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$$

## Original Lucas-Kanade algorithm III



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

linearized by performing first order Taylor expansion[5]

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

$\nabla I = [\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}]$ is the gradient image[6] computed at $\mathbf{W}(\mathbf{x};\mathbf{p})$. The term $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ is the Jacobian of the warp.

[5] Detailed explanation on the blackboard.

[6] As a vector it should have been a column wise oriented. However, for sake of clarity of equations row vector is exceptionally considered here.

## Original Lucas-Kanade algorithm IV



Derive

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

with respect to $\Delta\mathbf{p}$

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

setting equality to zero yields

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

where $\mathtt{H}$ is (Gauss-Newton) approximation of Hessian matrix.

$$\mathtt{H} = \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]^\top \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]$$

## The Lucas-Kanade algorithm—Summary

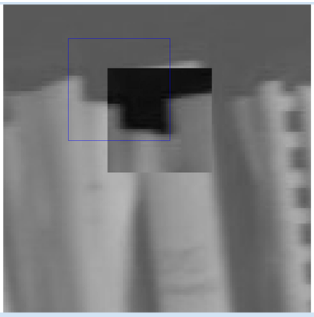


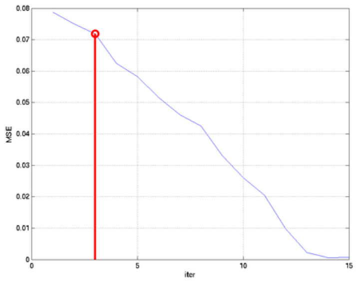
Iterate:

1. Warp $I$ with $\mathbf{W}(\mathbf{x};\mathbf{p})$
2. Warp the gradient $\nabla I$ with $\mathbf{W}(\mathbf{x};\mathbf{p})$
3. Evaluate the Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ at $(\mathbf{x};\mathbf{p})$ and compute the steepest descent image $\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}$
4. Compute the $\mathtt{H} = \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]^\top \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]$
5. Compute $\Delta\mathbf{p} = \mathtt{H}^{-1} \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]^\top [T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))]$
6. Update the parameters $\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$

until $\|\Delta\mathbf{p}\| \leq \epsilon$

## Example of convergence

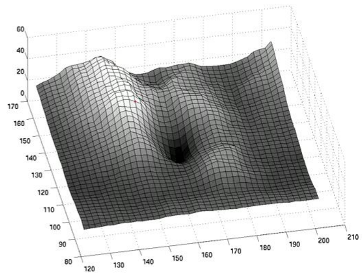





video



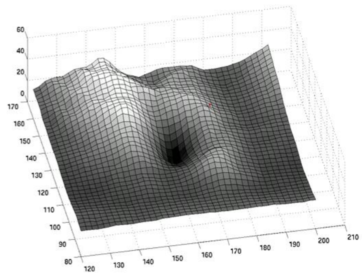
## Example of convergence




Convergence video: Initial state is within the basin of attraction



## Example of divergence




Divergence video: Initial state is outside the basin of attraction

## Example – on-line demo



Let play and see . . .

## What are good features (windows) to track?



How to select good templates $T(\mathbf{x})$ for image registration, object tracking.

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

where $\mathtt{H}$ is the matrix

$$\mathtt{H} = \sum_{\mathbf{x}} \left[ \nabla I \, \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]^{\top} \left[ \nabla I \, \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]$$

The stability of the iteration is mainly influenced by the inverse of Hessian. We can study its eigenvalues. Consequently, the criterion of a good feature window is $\min(\lambda_1, \lambda_2) > \lambda_{min}$ (texturedness).

## What are good features (windows) to track?



Consider translation $\mathbf{W}(\mathbf{x};\mathbf{p}) = \begin{bmatrix} x + p_1 \\ y + p_2 \end{bmatrix}$. The Jacobian is then $\frac{\partial \mathbf{W}}{\partial \mathbf{p}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

$$\begin{aligned} \mathtt{H} &= \sum_{\mathbf{x}} \left[ \nabla I \, \tfrac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]^{\top} \left[ \nabla I \, \tfrac{\partial \mathbf{W}}{\partial \mathbf{p}} \right] \\ &= \sum_{\mathbf{x}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix} \left[ \tfrac{\partial I}{\partial x}, \tfrac{\partial I}{\partial x} \right] \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\ &= \sum_{\mathbf{x}} \begin{bmatrix} \left( \frac{\partial I}{\partial x} \right)^2 & \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \\ \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} & \left( \frac{\partial I}{\partial y} \right)^2 \end{bmatrix} \end{aligned}$$

The image windows with varying derivatives in both directions. Homeogeneous areas are clearly not suitable. Texture oriented mostly in one direction only would cause instability for this translation.

## What are the good points for translations?



The matrix

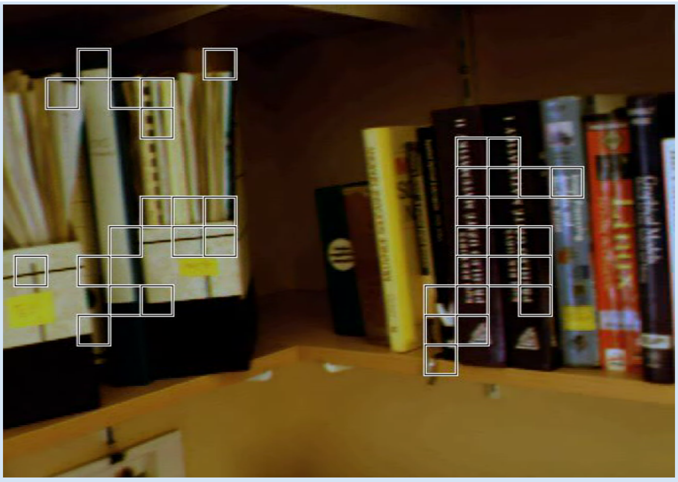
$$\mathtt{H} = \sum_{\mathbf{x}} \begin{bmatrix} \left(\frac{\partial I}{\partial x}\right)^2 & \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} \\ \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} & \left(\frac{\partial I}{\partial y}\right)^2 \end{bmatrix}$$

Should have large eigenvalues. We have seen the matrix already, where?

Harris corner detector [2]! The matrix is sometimes called Harris matrix.

## Experiments - no occlusions

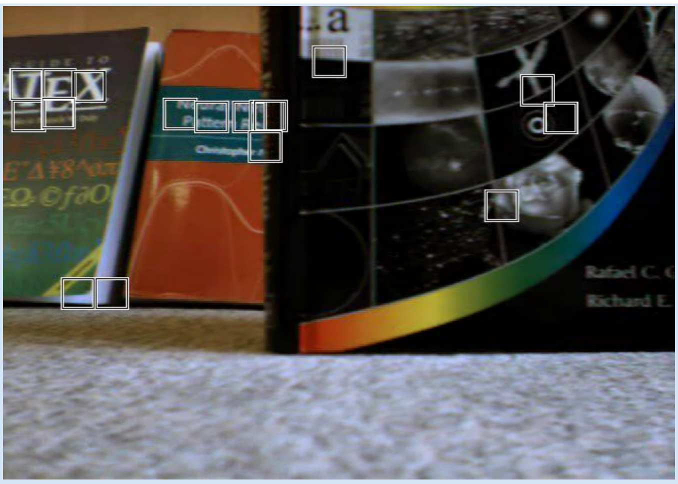




video

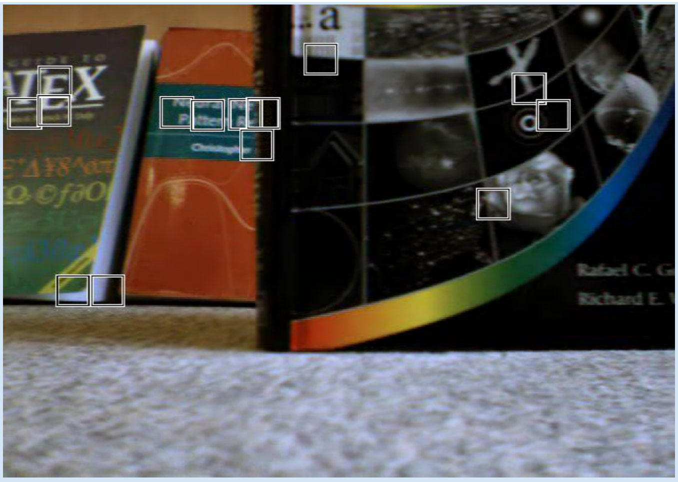
## Experiments - occlusions





video

## Experiments - occlusions with dissimilarity



video

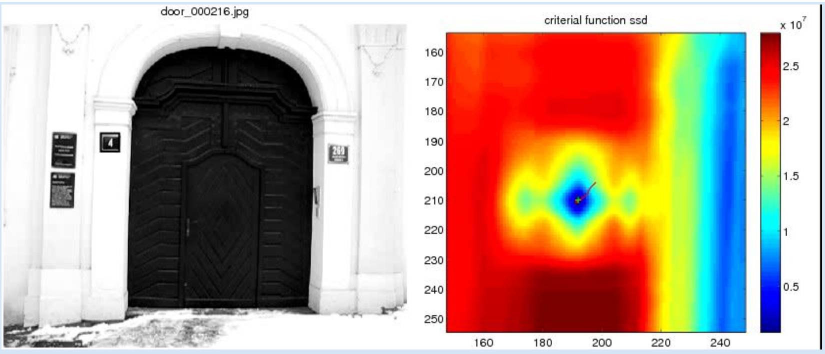


## Experiments - object motion

video



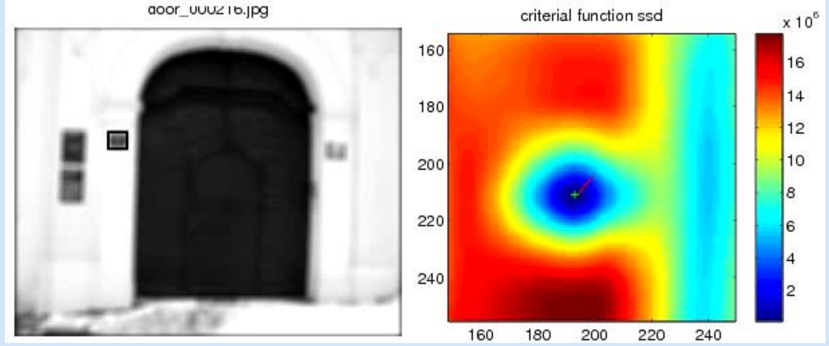
## Experiments – door tracking




video

# Experiments – door tracking – smoothed

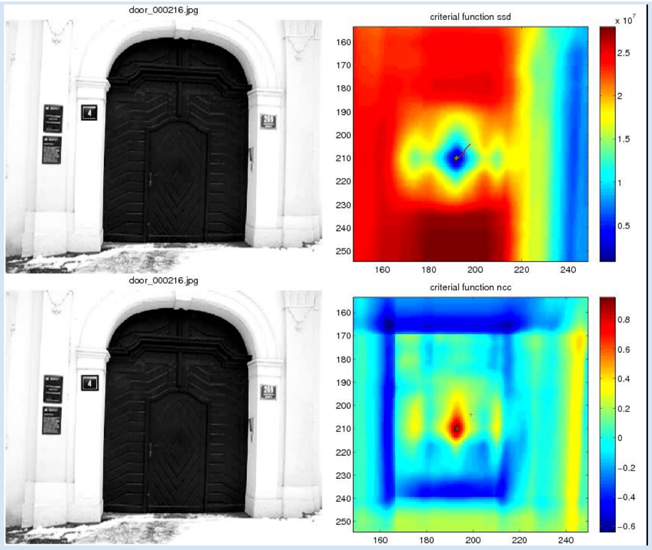





video

# Comparison of ncc vs KLT tracking






video

# References




[1] Simon Baker and Iain Matthews. Lucas-Kanade 20 years on: A unifying framework. International Journal of Computer Vision, 56(3):221–255, 2004.

[2] C. Harris and M. Stephen. A combined corner and edge detection. In M. M. Matthews, editor, Proceedings of the 4th ALVEY vision conference, pages 147–151, University of Manchaster, England, September 1988. on-line copies available on the web.

[3] J.P. Lewis. Fast template matching. In Vision Interfaces, pages 120–123, 1995. Extended version published on-line as "Fast Normalized Cross-Correlation" at http://www.idiom.com/~zilla/Work/nvisionInterface/nip.pdf.

[4] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In Proceedings of the 7th International Conference on Artificial Intelligence, pages 674–679, August 1981.

[5] Jianbo Shi and Carlo Tomasi. Good features to track. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 593–600, 1994.

[6] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, April 1991.

# End

**OPPA European Social Fund**
**Prague & EU: We invest in your future.**