Sequential Learned Linear Predictors for Object Tracking

Tomáš Svoboda

joint work with Karel Zimmermann, Jiří Matas, and David Hurych

Czech Technical University, Faculty of Electrical Engineering
Center for Machine Perception, Prague, Czech Republic

svoboda@cmp.felk.cvut.cz

http://cmp.felk.cvut.cz/~svoboda

June 23, 2009

Object tracking

- **Tracking** - iterative estimation of an object pose in a sequence (e.g., 2D position of Basil's head).
- Trade-off among *accuracy, speed, robustness* is explicitly taken into account.

video: Basil head

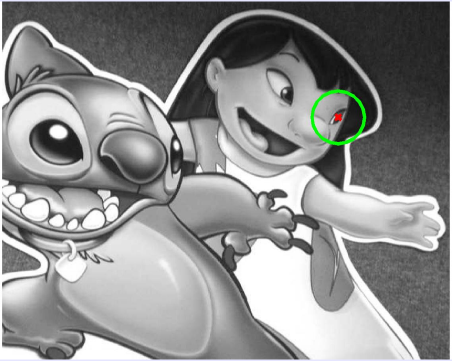
Object tracking

- **Tracking** - iterative estimation of an object pose in a sequence (e.g., 2D position of Basil's head).
- Trade-off among *accuracy, speed, robustness* is explicitly taken into account.

video: Basil head

Object tracking

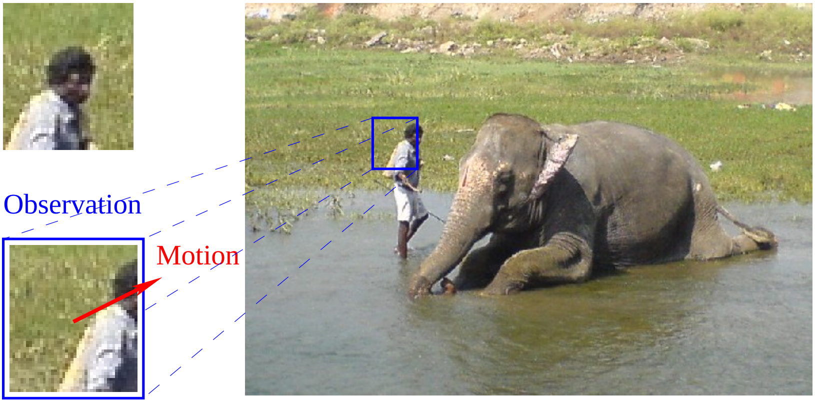
- **Tracking** - iterative estimation of an object pose in a sequence (e.g., 2D position of Basil's head).
- Trade-off among *accuracy, speed, robustness* is explicitly taken into account.



video: Cartoon eye

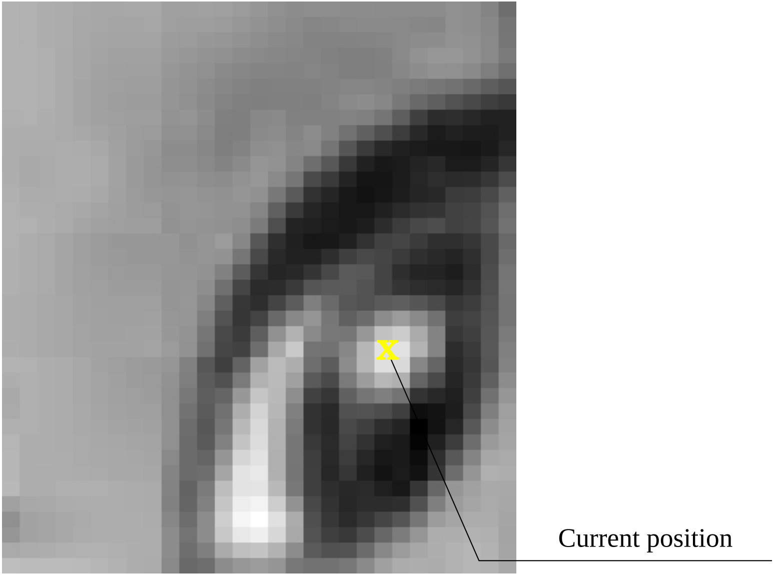
Tracking

Template Current image

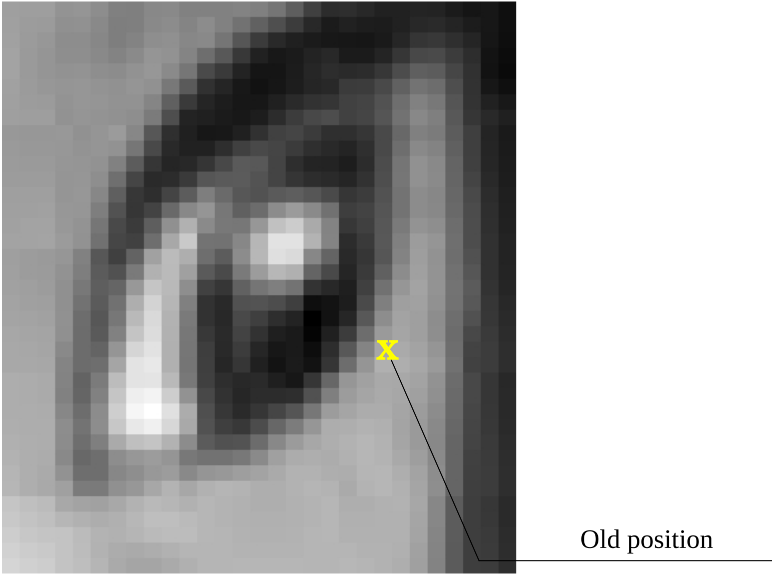


Observation (image) **I** and template **J**

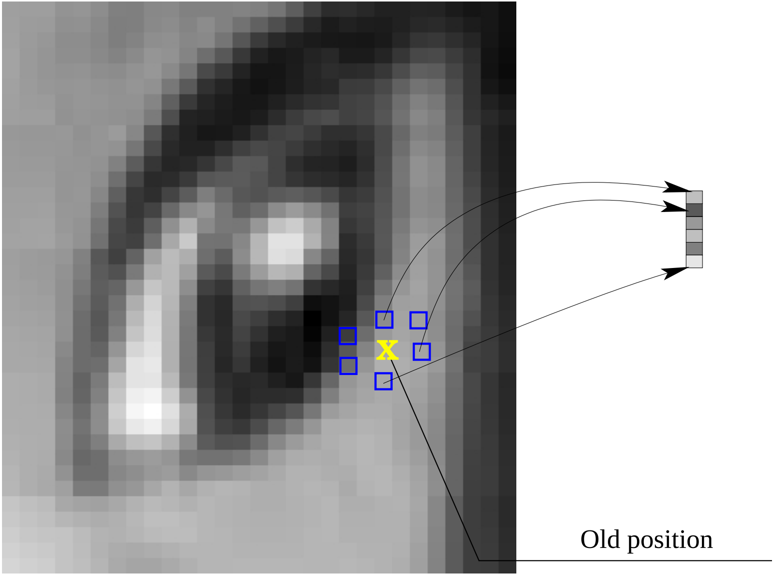
Tracking of a single point



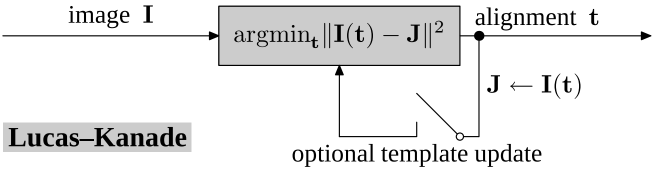
Tracking of a single point

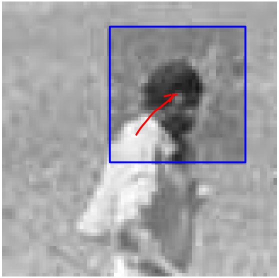


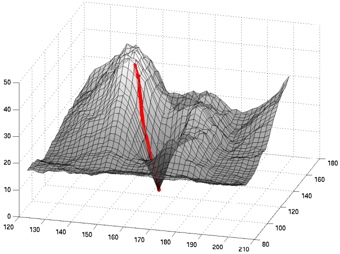
Tracking of a single point



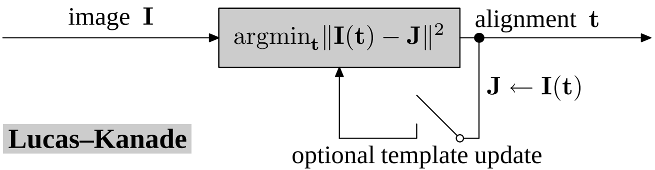
[Lucas-Kanade1981] - Iterative minimization

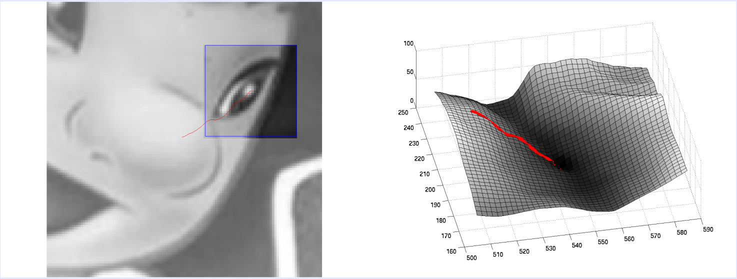






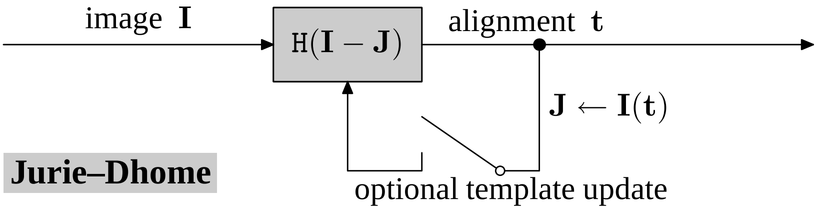
[Lucas-Kanade1981] - Iterative minimization





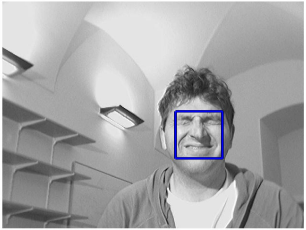
video: KLT iterations

Regression - learn the mapping in advanceo



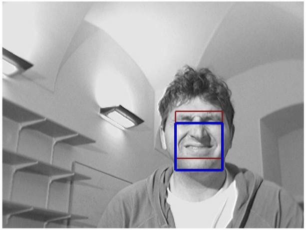
- [Jurie-BMVC-2002] - learned motion and optional hard template update
- [Cootes-PAMI-2001] - learned regression during AAM iterations
- . . .

Learning alignment for one predictor



- $\varphi($  $) = (0,0)^\top$
- $\varphi(\quad) = (0,0)^\top$
- $\varphi(\quad) = (-25,0)^\top$
- $\varphi(\quad) = (-25,0)^\top$
- $\varphi(\quad) = (25,-15)^\top$
- $\varphi(\quad) = (25,-15)^\top$

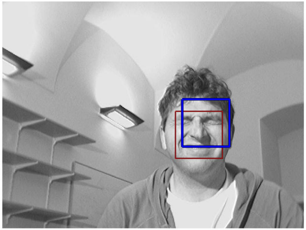
Learning alignment for one predictor



- $\varphi($  $) = (0, 0)^\top$
- $\varphi($  $) = (-25, 0)^\top$
- $\varphi(\quad) = (25, -15)^\top$

- $\varphi(\quad) = (0, 0)^\top$
- $\varphi(\quad) = (-25, 0)^\top$
- $\varphi(\quad) = (25, -15)^\top$

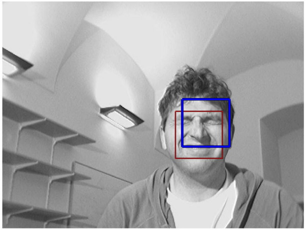
Learning alignment for one predictor

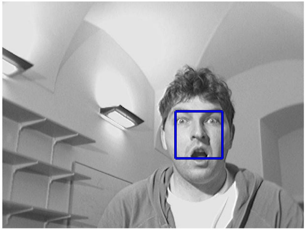


- $\varphi($  $) = (0, 0)^\top$
- $\varphi($  $) = (-25, 0)^\top$
- $\varphi($  $) = (25, -15)^\top$

- $\varphi(\quad) = (0, 0)^\top$
- $\varphi(\quad) = (-25, 0)^\top$
- $\varphi(\quad) = (25, -15)^\top$

Learning alignment for one predictor

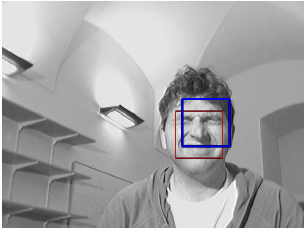


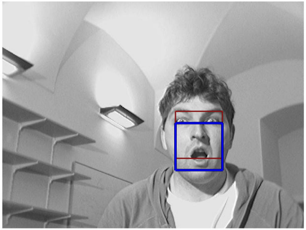


- $\varphi($  $) = (0, 0)^\top$
- $\varphi($  $) = (-25, 0)^\top$
- $\varphi($  $) = (25, -15)^\top$

- $\varphi($  $) = (0, 0)^\top$
- $\varphi(\quad) = (-25, 0)^\top$
- $\varphi(\quad) = (25, -15)^\top$

Learning alignment for one predictor

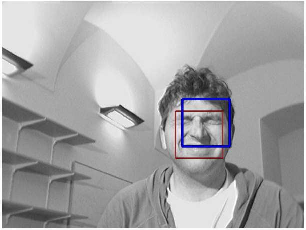


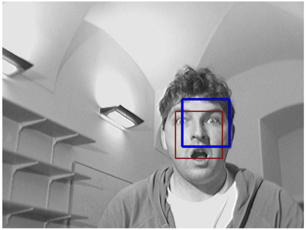


- $\varphi($  $) = (0, 0)^\top$
- $\varphi($  $) = (-25, 0)^\top$
- $\varphi($  $) = (25, -15)^\top$

- $\varphi($  $) = (0, 0)^\top$
- $\varphi($  $) = (-25, 0)^\top$
- $\varphi(\quad) = (25, -15)^\top$

Learning alignment for one predictor

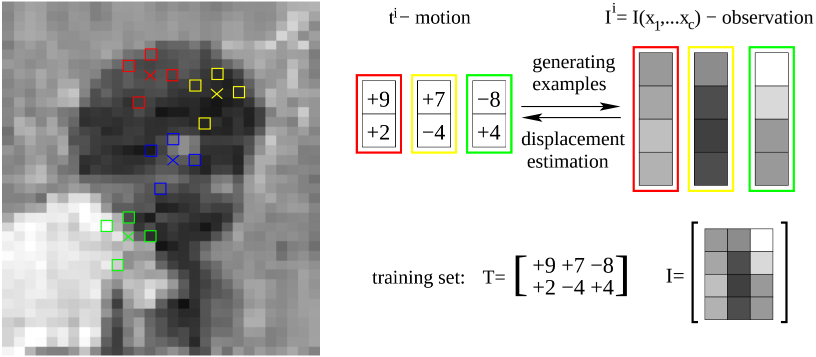




- $\varphi($  $) = (0, 0)^\top$
- $\varphi($  $) = (-25, 0)^\top$
- $\varphi($  $) = (25, -15)^\top$

- $\varphi($  $) = (0, 0)^\top$
- $\varphi($  $) = (-25, 0)^\top$
- $\varphi($  $) = (25, -15)^\top$

Generating training examples



Training set: $(\mathtt{I}, \mathtt{T})$

$\mathtt{I} = [\mathbf{I}^1 - \mathbf{J}, \mathbf{I}^2 - \mathbf{J}, \ldots \mathbf{I}^d - \mathbf{J}]$ and $\mathtt{T} = [\mathbf{t}^1, \mathbf{t}^2, \ldots \mathbf{t}^d]$.

LS learning

$$\varphi^* = \text{argmin}_\varphi \sum_{\mathbf{t}} \|\varphi\Big(\mathbf{I}(\mathbf{t} \circ X)\Big) - \mathbf{t}\|^2.$$

Minimizes sum of square errors over all training set. Leads to matrix pseudoinverse computation.

Example for *linear mapping*:

$$\mathtt{H}^* = \underset{\mathtt{H}}{\text{argmin}} \sum_{i=1}^{d} \|\mathtt{H}(\mathbf{I}^i - \mathbf{J}) - \mathbf{t}^i\|_2^2 = \underset{\mathtt{H}}{\text{argmin}} \|\mathtt{H}\mathtt{I} - \mathtt{T}\|_F^2$$

after some derivation

$$\mathtt{H}^* = \mathtt{T}\underbrace{\mathtt{I}^\top(\mathtt{I}\mathtt{I}^\top)^{-1}}_{\mathtt{I}^+} = \mathtt{T}\mathtt{I}^+.$$

LS learning

$$\varphi^* = \mathrm{argmin}_{\varphi} \sum_{\mathbf{t}} \|\varphi\Big(\mathbf{I}(\mathbf{t} \circ X)\Big) - \mathbf{t}\|^2.$$

Minimizes sum of square errors over all training set. Leads to matrix pseudoinverse computation.

Example for *linear mapping*:

$$\mathtt{H}^* = \underset{\mathtt{H}}{\mathrm{argmin}} \sum_{i=1}^{d} \|\mathtt{H}(\mathbf{I}^i - \mathbf{J}) - \mathbf{t}^i\|_2^2 = \underset{\mathtt{H}}{\mathrm{argmin}} \|\mathtt{HI} - \mathtt{T}\|_F^2$$

after some derivation

$$\mathtt{H}^* = \mathtt{T}\underbrace{\mathtt{I}^\top(\mathtt{II}^\top)^{-1}}_{\mathtt{I}^+} = \mathtt{TI}^+.$$

LS learning

$$\varphi^* = \text{argmin}_\varphi \sum_{\mathbf{t}} \|\varphi\Big(\mathbf{I}(\mathbf{t} \circ X)\Big) - \mathbf{t}\|^2.$$

Minimizes sum of square errors over all training set. Leads to matrix pseudoinverse computation.

Example for *linear mapping*:

$$\mathtt{H}^* = \underset{\mathtt{H}}{\text{argmin}} \sum_{i=1}^{d} \|\mathtt{H}(\mathbf{I}^i - \mathbf{J}) - \mathbf{t}^i\|_2^2 = \underset{\mathtt{H}}{\text{argmin}} \|\mathtt{HI} - \mathtt{T}\|_F^2$$

after some derivation

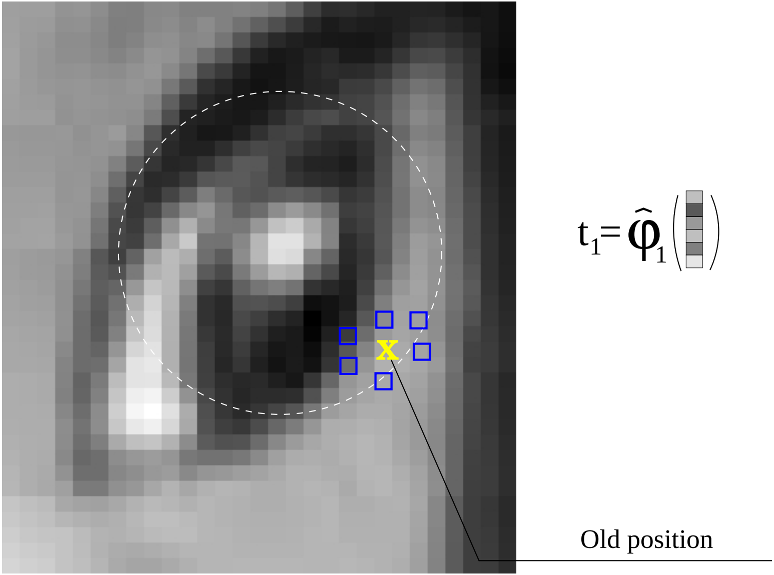
$$\mathtt{H}^* = \mathtt{T}\underbrace{\mathtt{I}^\top(\mathtt{II}^\top)^{-1}}_{\mathtt{I}^+} = \mathtt{TI}^+.$$

Min-max learning

$$\varphi^* = \operatorname{argmin}_\varphi \max_{\mathbf{t}} \|\varphi\Big(\mathbf{I}(\mathbf{t} \circ X)\Big) - \mathbf{t}\|_\infty .$$

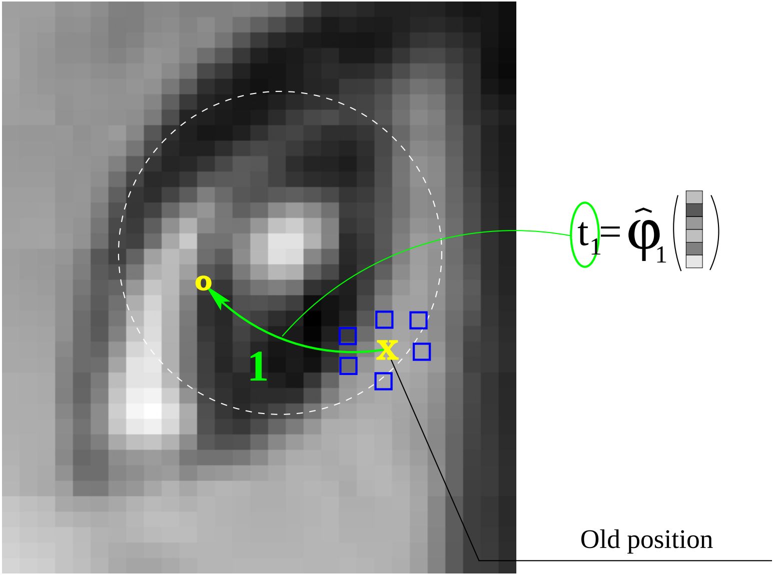
Minimizes the *worst case* (the biggest estimation error) in the training set. Leads to linear programming.

Tracking of a single point by a *sequence* of predictors

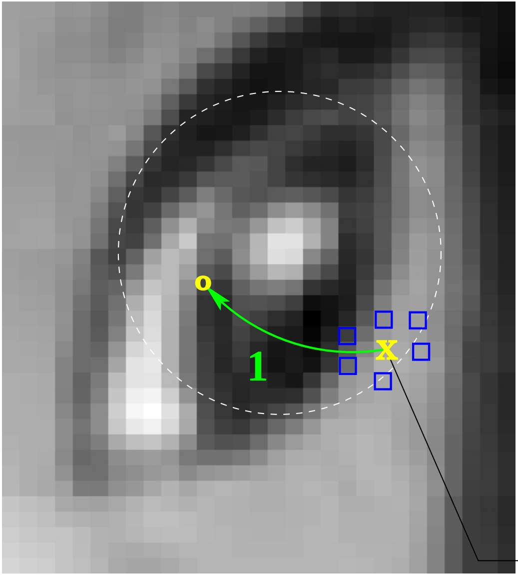


$$t_1 = \hat{\varphi}_1\left(\;\right)$$

Tracking of a single point by a *sequence* of predictors



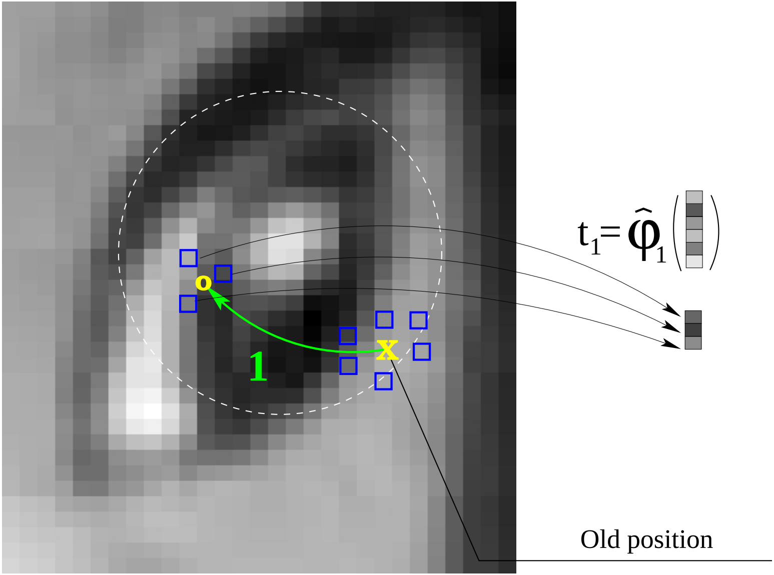
Tracking of a single point by a *sequence* of predictors



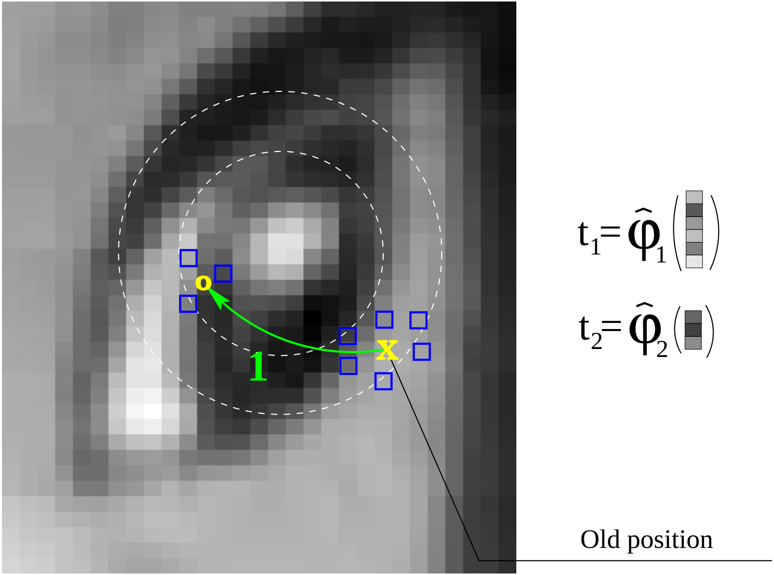
$$t_1 = \hat{\varphi}_1\left(\;\right)$$

Old position

Tracking of a single point by a *sequence* of predictors



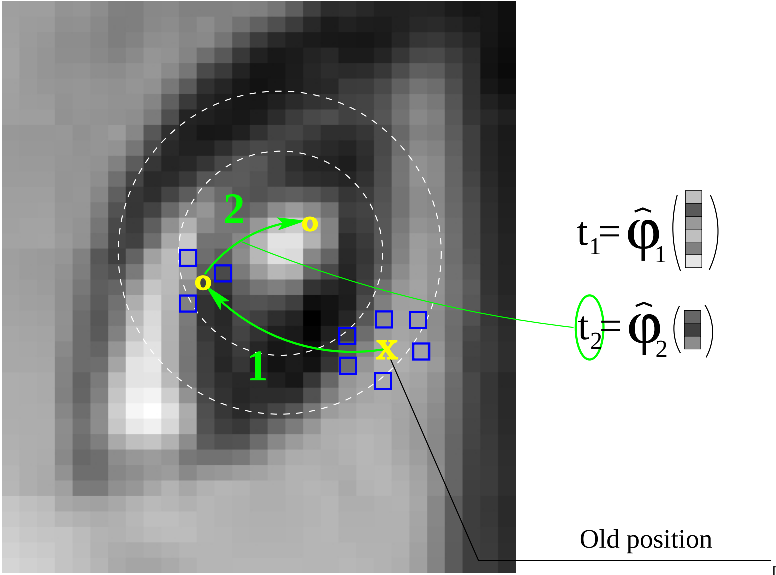
Tracking of a single point by a *sequence* of predictors



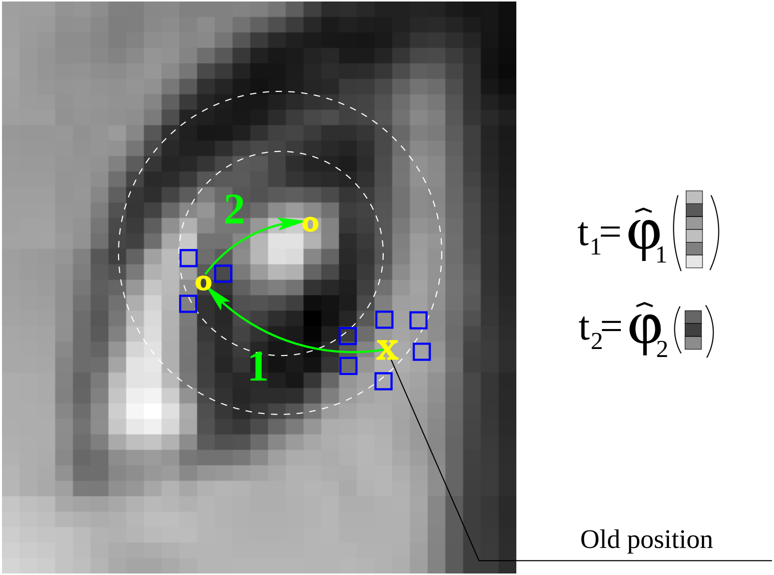
$t_1 = \hat{\varphi}_1(\cdot)$

$t_2 = \hat{\varphi}_2(\cdot)$

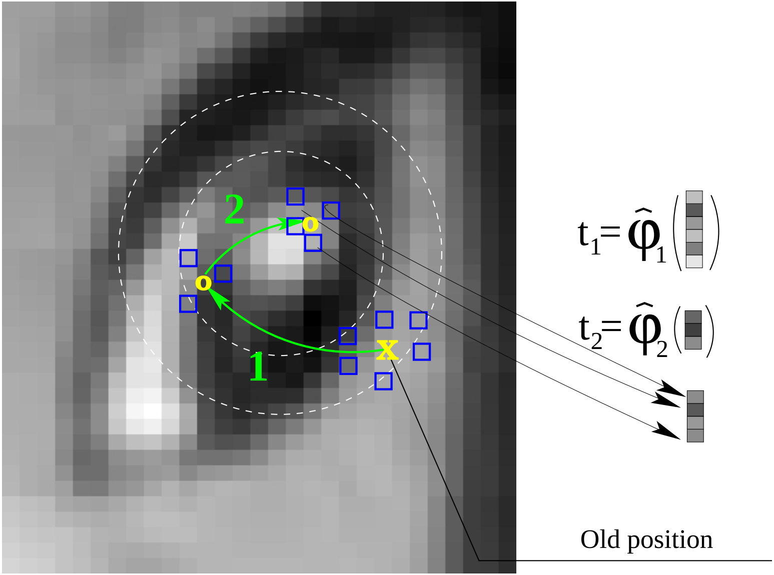
Tracking of a single point by a *sequence* of predictors



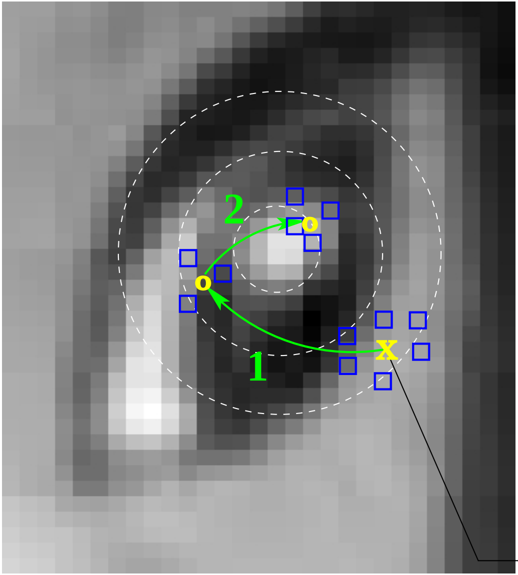
Tracking of a single point by a *sequence* of predictors



Tracking of a single point by a *sequence* of predictors



Tracking of a single point by a *sequence* of predictors



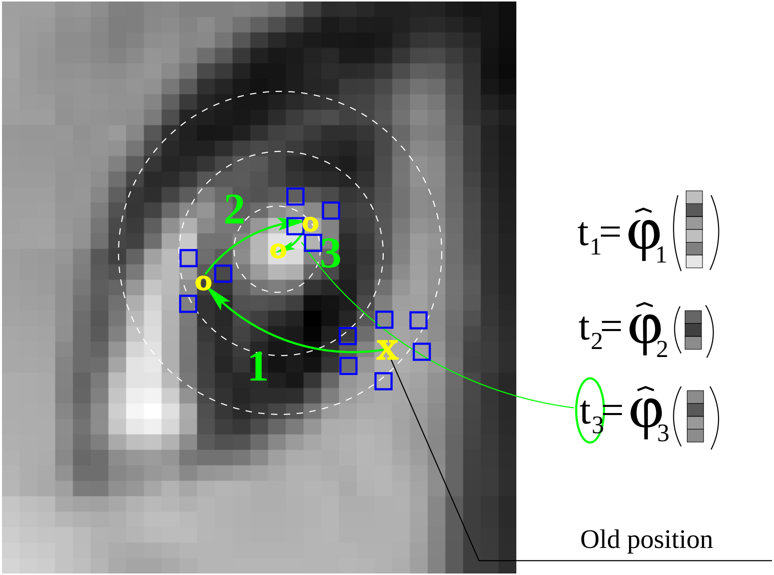
$t_1 = \hat{\varphi}_1(\,\cdot\,)$

$t_2 = \hat{\varphi}_2(\,\cdot\,)$

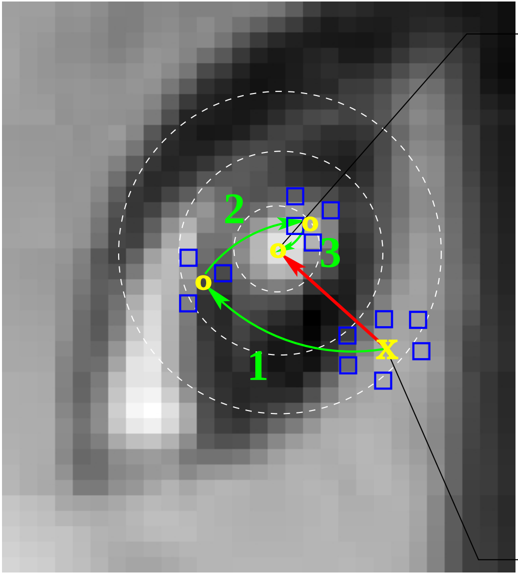
$t_3 = \hat{\varphi}_3(\,\cdot\,)$

Old position

Tracking of a single point by a *sequence* of predictors



Tracking of a single point by a *sequence* of predictors



New position

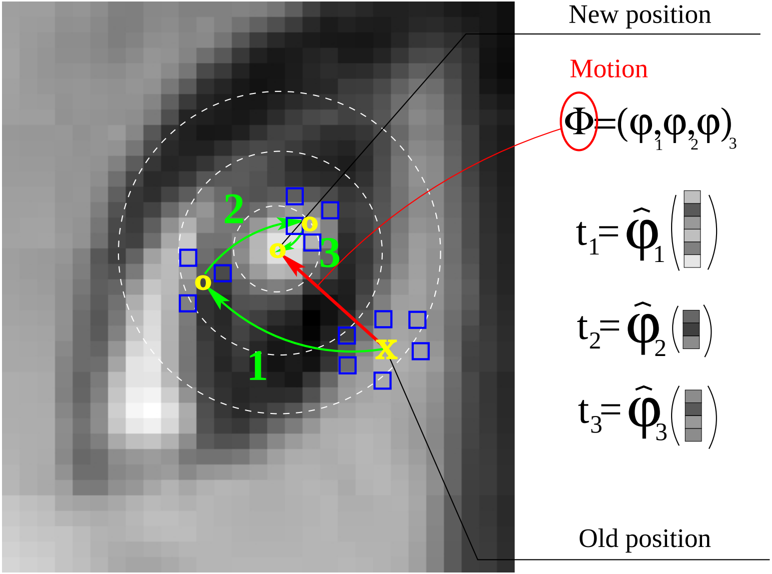
$t_1 = \hat{\varphi}_1 \left(\; \right)$

$t_2 = \hat{\varphi}_2 \left(\; \right)$

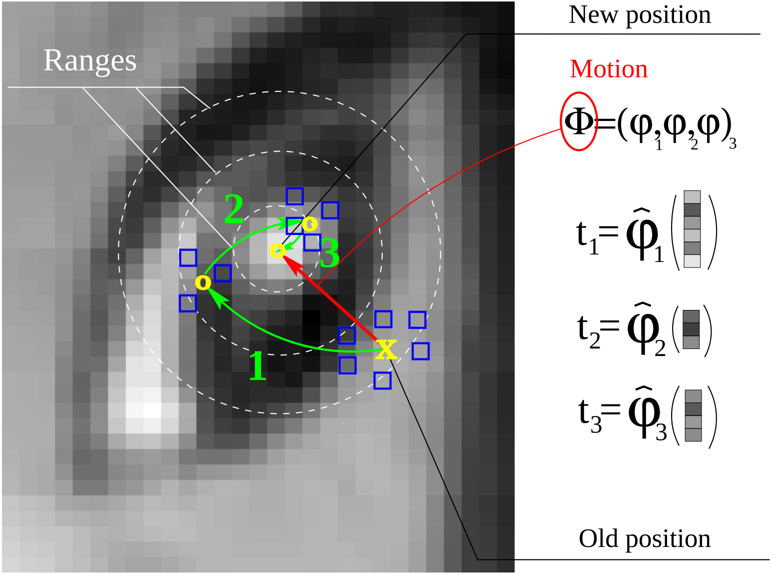
$t_3 = \hat{\varphi}_3 \left(\; \right)$

Old position

Tracking of a single point by a *sequence* of predictors



Tracking of a single point by a *sequence* of predictors



Learning of sequential predictor

- **Learning** - searching for the sequence with predefined *range*, *accuracy* and minimal *computational cost*.
  - [Zimmermann-PAMI-2009] - Dynamic programming estimates the optimal sequence of linear predictors.
  - [Zimmermann-IVC-2009] - Branch & bound search allows for time constrained learning (demo in MATLAB).

[Zimmermann-PAMI-2009] K.Zimmermann, J.Matas, T.Svoboda. Tracking by an Optimal Sequence of Linear Predictors, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE computer society, 2009, vol. 31, No 4, pp 677–692.

[Zimmermann-IVC-2009] K.Zimmermann, T.Svoboda, J.Matas. Anytime learning for the NoSLLiP tracker. *Image and Vision Computing*, Elsevier, accepted, available on-line.

Learning of sequential predictor

- **Learning** - searching for the sequence with predefined *range*, *accuracy* and minimal *computational cost*.
  - [Zimmermann-PAMI-2009] - Dynamic programming estimates the optimal sequence of linear predictors.
  - [Zimmermann-IVC-2009] - Branch & bound search allows for time constrained learning (demo in MATLAB).

[Zimmermann-PAMI-2009] K.Zimmermann, J.Matas, T.Svoboda. Tracking by an Optimal Sequence of Linear Predictors, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE computer society, 2009, vol. 31, No 4, pp 677–692.

[Zimmermann-IVC-2009] K.Zimmermann, T.Svoboda, J.Matas. Anytime learning for the NoSLLiP tracker. *Image and Vision Computing*, Elsevier, accepted, available on-line.

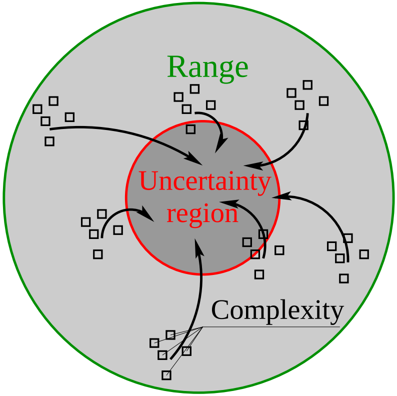
Learning of sequential predictor

- **Learning** - searching for the sequence with predefined *range*, *accuracy* and minimal *computational cost*.
  - [Zimmermann-PAMI-2009] - Dynamic programming estimates the optimal sequence of linear predictors.
  - [Zimmermann-IVC-2009] - Branch & bound search allows for time constrained learning (demo in MATLAB).

[Zimmermann-PAMI-2009] K.Zimmermann, J.Matas, T.Svoboda. Tracking by an Optimal Sequence of Linear Predictors, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE computer society, 2009, vol. 31, No 4, pp 677–692.

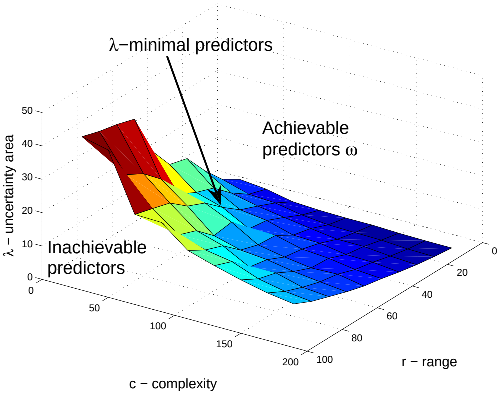
[Zimmermann-IVC-2009] K.Zimmermann, T.Svoboda, J.Matas. Anytime learning for the NoSLLiP tracker. *Image and Vision Computing*, Elsevier, accepted, available on-line.

Learning of the optimal sequence of linear predictors



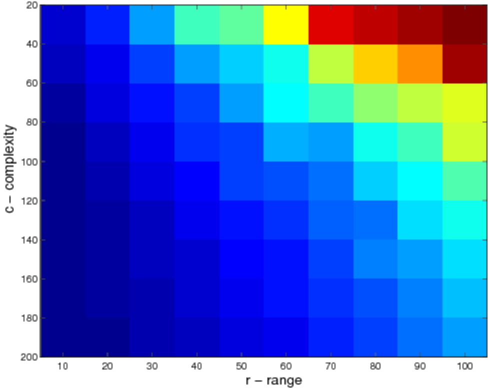
- **Range**: the set of admissible motions, r.
- **Complexity**: cardinality of support set, c.
- **Uncertainty region**: the region within which all predictions lie, λ. Small red circles show acceptable uncertainty.

Learning of the optimal sequence of linear predictors



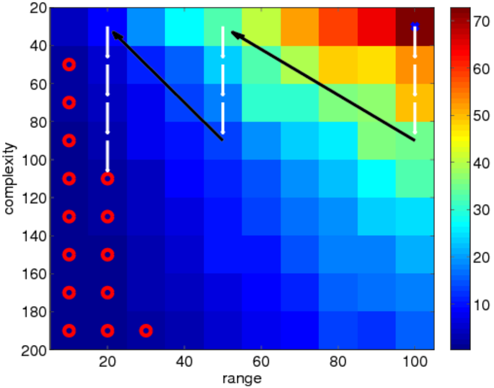
- **Range**: the set of admissible motions, r.
- **Complexity**: cardinality of support set, c.
- **Uncertainty region**: the region within which all predictions lie, λ. Small red circles show acceptable uncertainty.

Learning of the optimal sequence of linear predictors



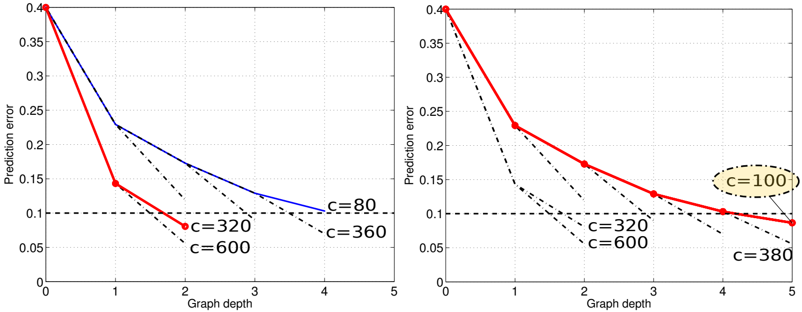
- **Range**: the set of admissible motions, r.
- **Complexity**: cardinality of support set, c.
- **Uncertainty region**: the region within which all predictions lie, λ. Small red circles show acceptable uncertainty.

Learning of the optimal sequence of linear predictors



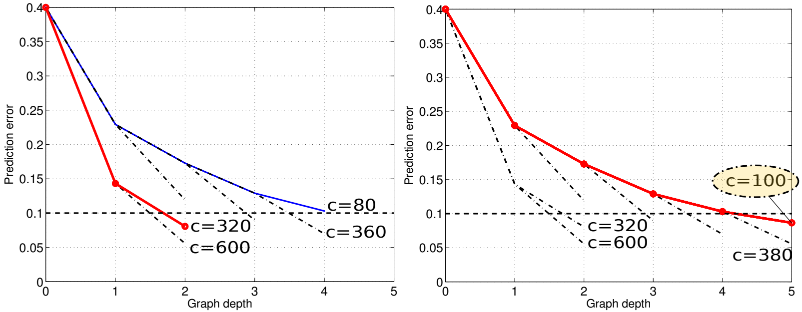
- **Range**: the set of admissible motions, r.
- **Complexity**: cardinality of support set, c.
- **Uncertainty region**: the region within which all predictions lie, λ. Small red circles show acceptable uncertainty.

Branch and Bound



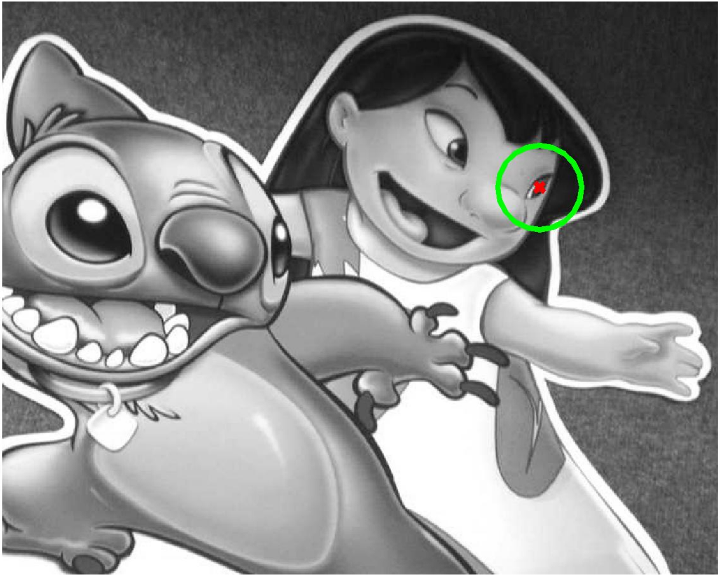
Don't forget to show the live demo!

Branch and Bound

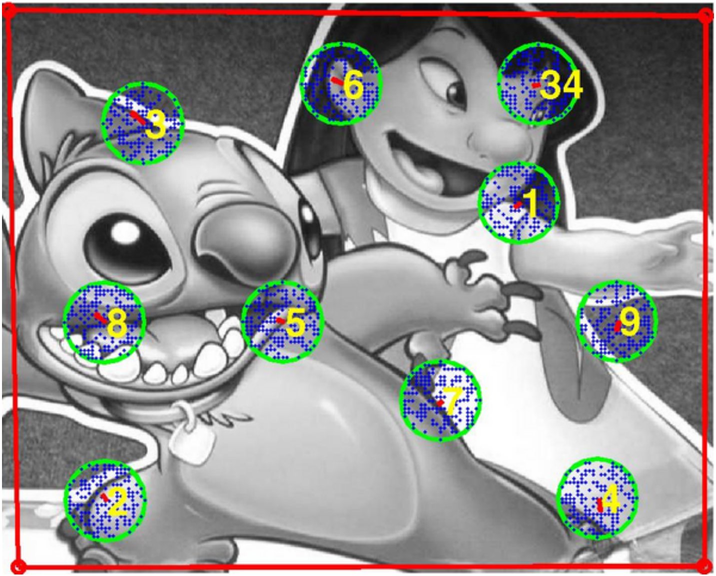


Don't forget to show the live demo!

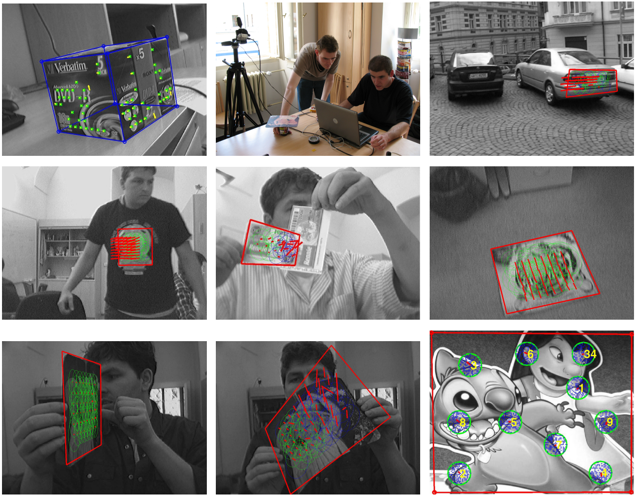
Tracking with one linear predictor.



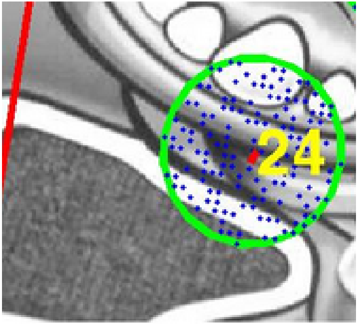
Modeling motion by number of linear predictors.

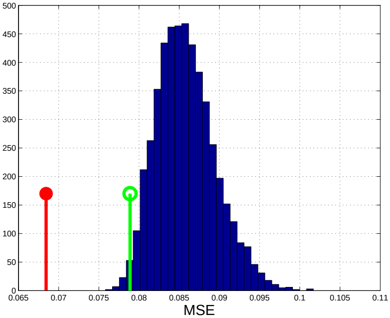


Motion blur, fast motion, views from acute angles and other image distortions.



Support set selection





- Greedy LS selection (red) of an efficient support set.
- Much better than 1%-quantile (green) achieavable by randomized sampling

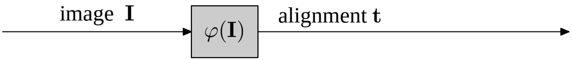
Tracking of objects with variable appearance

- **Variable appearance** - the way how the object looks like in the camera changes due to illumination, non-rigid deformation, out-of-plane rotation, ...

Tracking of objects with variable appearance

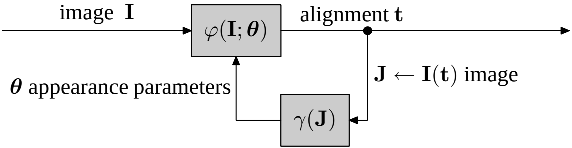
- **Variable appearance** - the way how the object looks like in the camera changes due to illumination, non-rigid deformation, out-of-plane rotation, ...

Simultaneous learning of motion and appearance



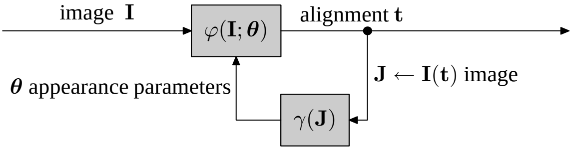
- Introduce feedback which encodes appearance in a low dimensional space and adjust the predictor.
- Appearance parameters learned in unsupervised way.
- Simultaneous learning of φ and $\gamma \Rightarrow$ appearance encoded in the low dimensional space, which is the most suitable for the motion estimation.

Simultaneous learning of motion and appearance



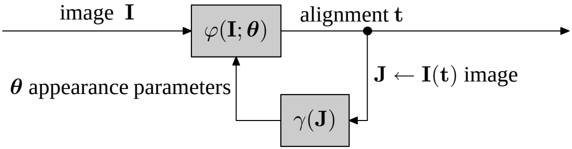
- Introduce feedback which encodes appearance in a low dimensional space and adjust the predictor.
- Appearance parameters learned in unsupervised way.
- Simultaneous learning of φ and $\gamma \Rightarrow$ appearance encoded in the low dimensional space, which is the most suitable for the motion estimation.

Simultaneous learning of motion and appearance



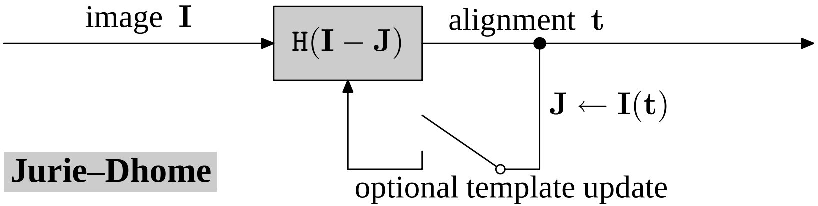
- Introduce feedback which encodes appearance in a low dimensional space and adjust the predictor.
- Appearance parameters learned in unsupervised way.
- Simultaneous learning of φ and γ $\Rightarrow$ appearance encoded in the low dimensional space, which is the most suitable for the motion estimation.

Simultaneous learning of motion and appearance



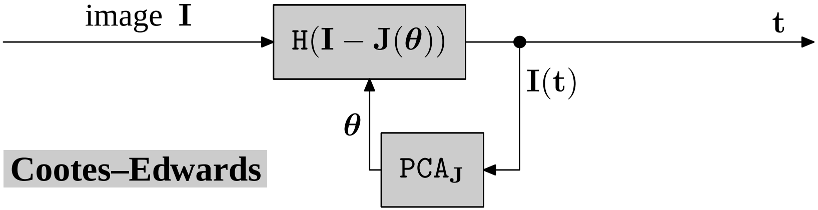
- Introduce feedback which encodes appearance in a low dimensional space and adjust the predictor.
- Appearance parameters learned in unsupervised way.
- Simultaneous learning of φ and $\gamma \Rightarrow$ appearance encoded in the low dimensional space, which is the most suitable for the motion estimation.

Learning appearance



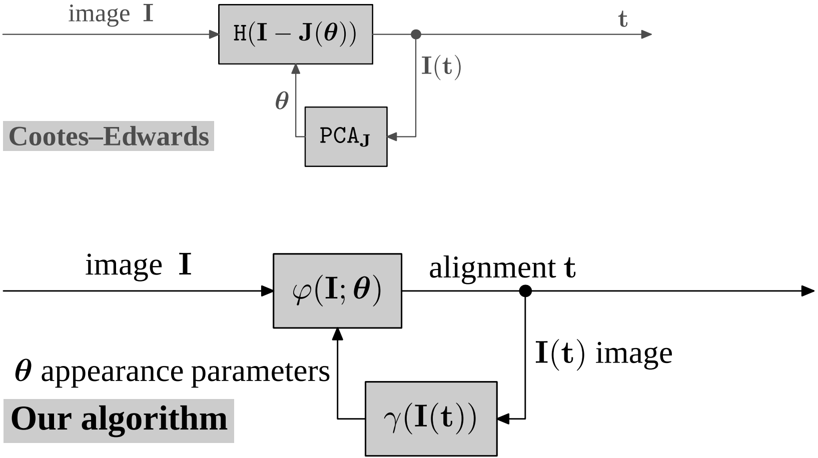
Jurie–Dhome

Learning appearance

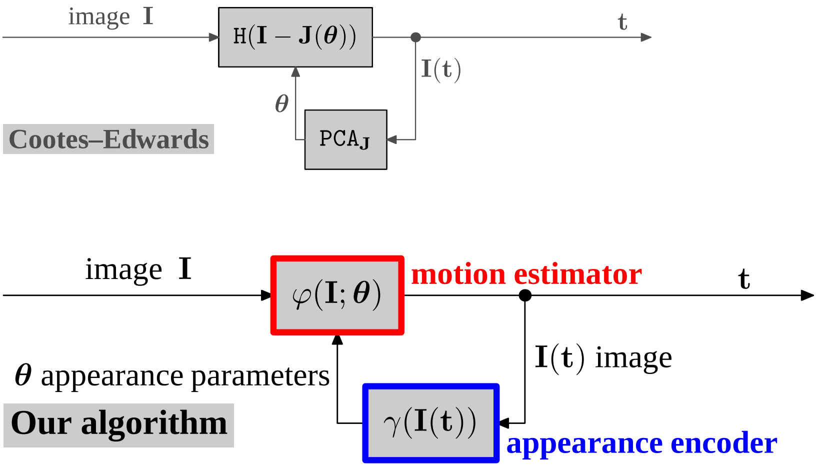


Cootes–Edwards

Learning appearance – our approach

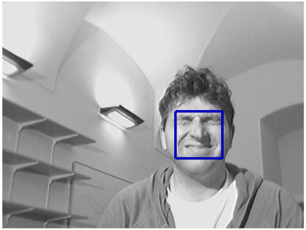


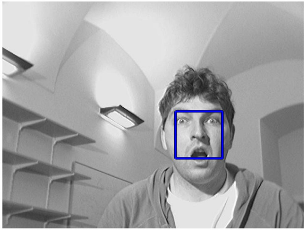
Learning appearance – our approach



Learning the appearance encoder γ

- Current appearance encoded in low-dim parameters.

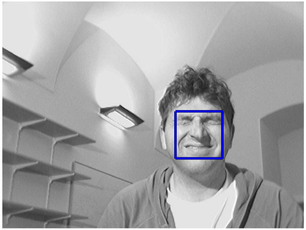


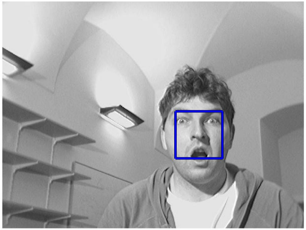


- $\gamma(\quad) = \theta_1$
- $\gamma(\quad) = \theta_2$

Learning the appearance encoder γ

- Current appearance encoded in low-dim parameters.

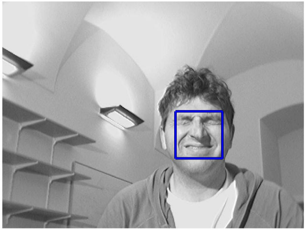


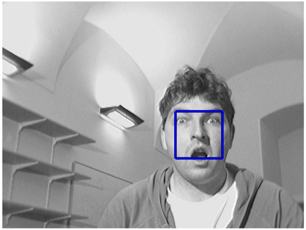


- $\gamma($  $) = \boldsymbol{\theta}_1$
- $\gamma(\quad) = \boldsymbol{\theta}_2$

Learning the appearance encoder γ

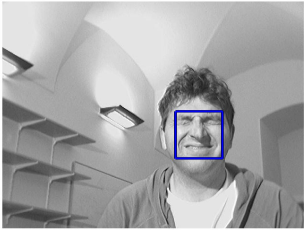
- Current appearance encoded in low-dim parameters.





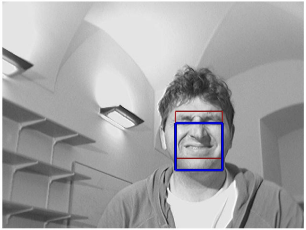
- $\gamma($  $) = \boldsymbol{\theta}_1$
- $\gamma($  $) = \boldsymbol{\theta}_2$

Learning the tracker $\varphi(\mathbf{I};\boldsymbol{\theta})$



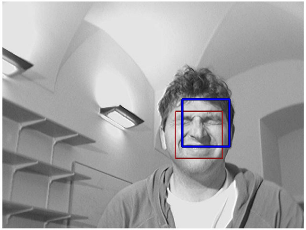
- $\varphi($  $;\boldsymbol{\theta}_1) = (0,0)^\top$
- $\varphi(\quad;\boldsymbol{\theta}_2) = (0,0)^\top$
- $\varphi(\quad;\boldsymbol{\theta}_1) = (-25,0)^\top$
- $\varphi(\quad;\boldsymbol{\theta}_2) = (-25,0)^\top$
- $\varphi(\quad;\boldsymbol{\theta}_1) = (25,-15)^\top$
- $\varphi(\quad;\boldsymbol{\theta}_2) = (25,-15)^\top$

Learning the tracker $\varphi(\mathbf{I};\boldsymbol{\theta})$



- $\varphi($  $;\boldsymbol{\theta}_1) = (0,0)^\top$
- $\varphi($  $;\boldsymbol{\theta}_1) = (-25,0)^\top$
- $\varphi(\quad;\boldsymbol{\theta}_1) = (25,-15)^\top$
- $\varphi(\quad;\boldsymbol{\theta}_2) = (0,0)^\top$
- $\varphi(\quad;\boldsymbol{\theta}_2) = (-25,0)^\top$
- $\varphi(\quad;\boldsymbol{\theta}_2) = (25,-15)^\top$

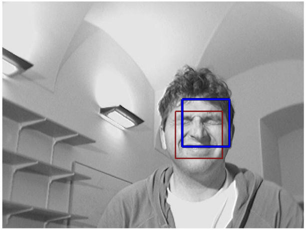
Learning the tracker $\varphi(\mathbf{I};\boldsymbol{\theta})$

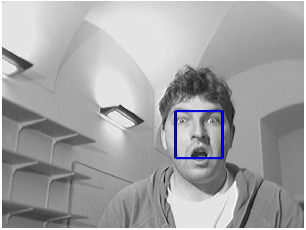


- $\varphi($  $;\boldsymbol{\theta}_1) = (0,0)^\top$
- $\varphi($  $;\boldsymbol{\theta}_1) = (-25,0)^\top$
- $\varphi($  $;\boldsymbol{\theta}_1) = (25,-15)^\top$

- $\varphi(\quad;\boldsymbol{\theta}_2) = (0,0)^\top$
- $\varphi(\quad;\boldsymbol{\theta}_2) = (-25,0)^\top$
- $\varphi(\quad;\boldsymbol{\theta}_2) = (25,-15)^\top$

Learning the tracker $\varphi(\mathbf{I}; \boldsymbol{\theta})$

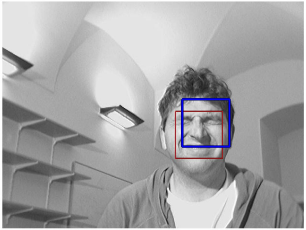


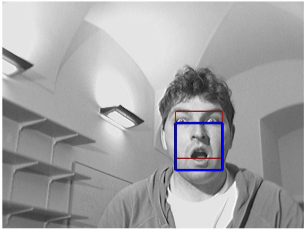


- $\varphi($  $; \boldsymbol{\theta}_1) = (0, 0)^\top$
- $\varphi($  $; \boldsymbol{\theta}_1) = (-25, 0)^\top$
- $\varphi($  $; \boldsymbol{\theta}_1) = (25, -15)^\top$

- $\varphi($  $; \boldsymbol{\theta}_2) = (0, 0)^\top$
- $\varphi(\quad ; \boldsymbol{\theta}_2) = (-25, 0)^\top$
- $\varphi(\quad ; \boldsymbol{\theta}_2) = (25, -15)^\top$

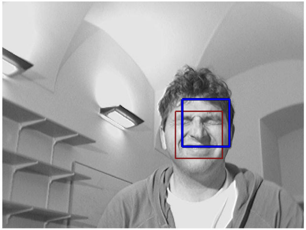
Learning the tracker $\varphi(\mathbf{I}; \boldsymbol{\theta})$

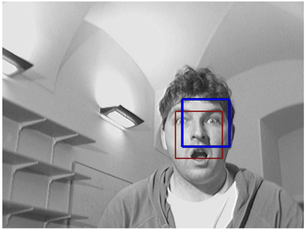




- $\varphi($  $; \boldsymbol{\theta}_1) = (0, 0)^\top$
- $\varphi($  $; \boldsymbol{\theta}_1) = (-25, 0)^\top$
- $\varphi($  $; \boldsymbol{\theta}_1) = (25, -15)^\top$
- $\varphi($  $; \boldsymbol{\theta}_2) = (0, 0)^\top$
- $\varphi($  $; \boldsymbol{\theta}_2) = (-25, 0)^\top$
- $\varphi(\quad ; \boldsymbol{\theta}_2) = (25, -15)^\top$

Learning the tracker $\varphi(\mathbf{I}; \boldsymbol{\theta})$





- $\varphi($  $; \boldsymbol{\theta}_1) = (0, 0)^\top$
- $\varphi($  $; \boldsymbol{\theta}_1) = (-25, 0)^\top$
- $\varphi($  $; \boldsymbol{\theta}_1) = (25, -15)^\top$

- $\varphi($  $; \boldsymbol{\theta}_2) = (0, 0)^\top$
- $\varphi($  $; \boldsymbol{\theta}_2) = (-25, 0)^\top$
- $\varphi($  $; \boldsymbol{\theta}_2) = (25, -15)^\top$

Simultaneous learning of φ and γ

- **Learning = minimization of the least-squares error**

$(\varphi^*, \gamma^*) = \arg\min_{\varphi,\gamma}$ $\Big[\varphi\big($  $;\gamma($  $)\big) - (0,0)^\top\Big]^2 +$

$$\Big[\varphi\big(\quad;\gamma(\quad)\big) - (-25,0)^\top\Big]^2 +$$

$$\Big[\varphi\big(\quad;\gamma(\quad)\big) - (25,-15)^\top\Big]^2 +$$

$$\Big[\varphi\big(\quad;\gamma(\quad)\big) - (0,0)^\top\Big]^2 +$$

$$\Big[\varphi\big(\quad;\gamma(\quad)\big) - (-25,0)^\top\Big]^2 +$$

$$\Big[\varphi\big(\quad;\gamma(\quad)\big) - (25,-15)^\top\Big]^2$$

Simultaneous learning of φ and γ

- **Learning = minimization of the least-squares error**

$(\varphi^*, \gamma^*) = \arg\min_{\varphi,\gamma}$ $\Big[\varphi($  $;\gamma($  $)) - (0,0)^\top\Big]^2 +$

$\Big[\varphi($  $;\gamma($  $)) - (-25,0)^\top\Big]^2 +$

$\Big[\varphi($  $;\gamma($  $)) - (25,-15)^\top\Big]^2 +$

$\Big[\varphi($  $;\gamma($  $)) - (0,0)^\top\Big]^2 +$

$\Big[\varphi($  $;\gamma($  $)) - (-25,0)^\top\Big]^2 +$

$\Big[\varphi($  $;\gamma($  $)) - (25,-15)^\top\Big]^2$

Linear mapping

- $\gamma(\mathbf{J}) : \boldsymbol{\theta} = \mathtt{G}\mathbf{J}$
- $\varphi(\mathbf{I}, \boldsymbol{\theta}) : \mathbf{t} = (\mathrm{H}_0 + \theta_1 \mathrm{H}_1 + \cdots + \theta_n \mathrm{H}_n)\mathbf{I}$
- Criterion is sum of squares of bilinear functions.

Linear mapping

- $\gamma(\mathbf{J}) : \boldsymbol{\theta} = \mathtt{G}\mathbf{J}$
- $\varphi(\mathbf{I}, \boldsymbol{\theta}) : \mathbf{t} = (\mathtt{H}_0 + \theta_1 \mathtt{H}_1 + \cdots + \theta_n \mathtt{H}_n)\mathbf{I}$
- Criterion is sum of squares of bilinear functions.

Linear mapping

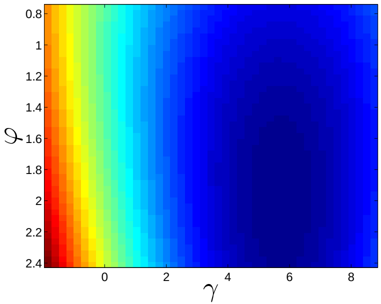
- $\gamma(\mathbf{J}) : \boldsymbol{\theta} = \mathtt{G}\mathbf{J}$
- $\varphi(\mathbf{I}, \boldsymbol{\theta}) : \mathbf{t} = (\mathtt{H}_0 + \theta_1 \mathtt{H}_1 + \cdots + \theta_n \mathtt{H}_n)\mathbf{I}$
- Criterion is sum of squares of bilinear functions.

Algorithm: iterative minimization of criterion $e(\varphi, \gamma)$

φ – motion (geometry mapping), γ – appearance mapping

color encodes criterion value $e(\varphi, \gamma)$

- Iterative minimization:
  - initialization $\gamma^0 = \text{rand}$
  - $\varphi^1 = \arg\min_\varphi e(\varphi, \gamma^0)$
  - $\gamma^1 = \arg\min_\gamma e(\varphi^1, \gamma)$
  - $\varphi^2 = \arg\min_\varphi e(\varphi, \gamma^1)$
  - $\gamma^2 = \arg\min_\gamma e(\varphi^2, \gamma)$
  - $\varphi^3 = \arg\min_\varphi e(\varphi, \gamma^2)$
  - $\gamma^3 = \arg\min_\gamma e(\varphi^3, \gamma)$
  - until convergence reached



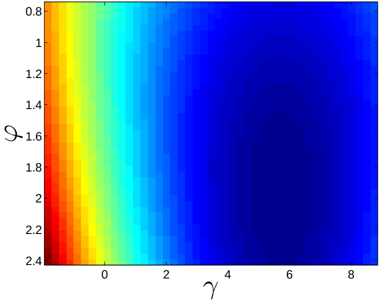
- Global optimality for linear φ, γ experimentally shown.

Algorithm: iterative minimization of criterion $e(\varphi, \gamma)$

φ – motion (geometry mapping), γ – appearance mapping

color encodes criterion value $e(\varphi, \gamma)$

- Iterative minimization:
  - initialization $\gamma^0 = \text{rand}$
  - $\varphi^1 = \arg\min_\varphi e(\varphi, \gamma^0)$
  - $\gamma^1 = \arg\min_\gamma e(\varphi^1, \gamma)$
  - $\varphi^2 = \arg\min_\varphi e(\varphi, \gamma^1)$
  - $\gamma^2 = \arg\min_\gamma e(\varphi^2, \gamma)$
  - $\varphi^3 = \arg\min_\varphi e(\varphi, \gamma^2)$
  - $\gamma^3 = \arg\min_\gamma e(\varphi^3, \gamma)$
  - until convergence reached



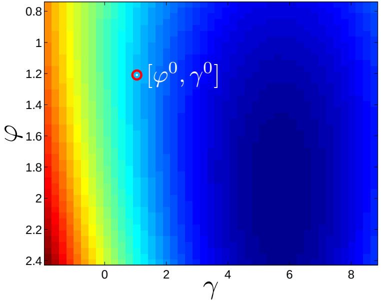
- Global optimality for linear φ, γ experimentally shown.

Algorithm: iterative minimization of criterion $e(\varphi, \gamma)$

φ – motion (geometry mapping), γ – appearance mapping

color encodes criterion value $e(\varphi, \gamma)$

- Iterative minimization:
  - initialization $\gamma^0 = \mathrm{rand}$
  - $\varphi^1 = \arg\min_\varphi e(\varphi, \gamma^0)$
  - $\gamma^1 = \arg\min_\gamma e(\varphi^1, \gamma)$
  - $\varphi^2 = \arg\min_\varphi e(\varphi, \gamma^1)$
  - $\gamma^2 = \arg\min_\gamma e(\varphi^2, \gamma)$
  - $\varphi^3 = \arg\min_\varphi e(\varphi, \gamma^2)$
  - $\gamma^3 = \arg\min_\gamma e(\varphi^3, \gamma)$
  - until convergence reached



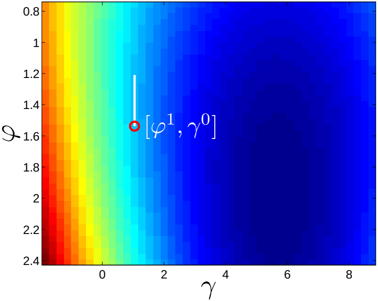
- Global optimality for linear φ, γ experimentally shown.

Algorithm: iterative minimization of criterion $e(\varphi, \gamma)$

φ – motion (geometry mapping), γ – appearance mapping

color encodes criterion value $e(\varphi, \gamma)$

- Iterative minimization:
  - initialization $\gamma^0 = \mathrm{rand}$
  - $\varphi^1 = \arg\min_\varphi e(\varphi, \gamma^0)$
  - $\gamma^1 = \arg\min_\gamma e(\varphi^1, \gamma)$
  - $\varphi^2 = \arg\min_\varphi e(\varphi, \gamma^1)$
  - $\gamma^2 = \arg\min_\gamma e(\varphi^2, \gamma)$
  - $\varphi^3 = \arg\min_\varphi e(\varphi, \gamma^2)$
  - $\gamma^3 = \arg\min_\gamma e(\varphi^3, \gamma)$
  - until convergence reached



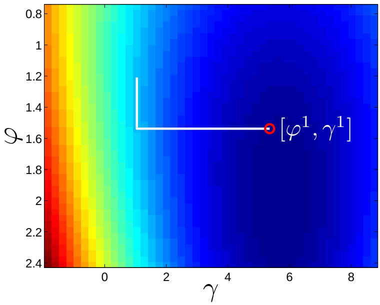
- Global optimality for linear φ, γ experimentally shown.

Algorithm: iterative minimization of criterion $e(\varphi, \gamma)$

φ – motion (geometry mapping), γ – appearance mapping

color encodes criterion value $e(\varphi, \gamma)$

- Iterative minimization:
  - initialization $\gamma^0 = \mathrm{rand}$
  - $\varphi^1 = \arg\min_\varphi e(\varphi, \gamma^0)$
  - $\gamma^1 = \arg\min_\gamma e(\varphi^1, \gamma)$
  - $\varphi^2 = \arg\min_\varphi e(\varphi, \gamma^1)$
  - $\gamma^2 = \arg\min_\gamma e(\varphi^2, \gamma)$
  - $\varphi^3 = \arg\min_\varphi e(\varphi, \gamma^2)$
  - $\gamma^3 = \arg\min_\gamma e(\varphi^3, \gamma)$
  - until convergence reached



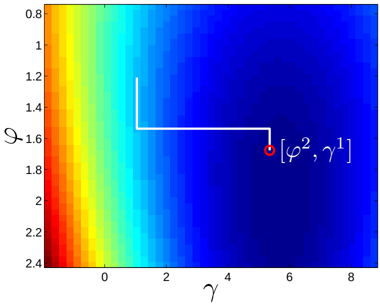
- Global optimality for linear φ, γ experimentally shown.

Algorithm: iterative minimization of criterion $e(\varphi, \gamma)$

φ – motion (geometry mapping), γ – appearance mapping

color encodes criterion value $e(\varphi, \gamma)$

- Iterative minimization:
  - initialization $\gamma^0 = \mathrm{rand}$
  - $\varphi^1 = \arg\min_{\varphi} e(\varphi, \gamma^0)$
  - $\gamma^1 = \arg\min_{\gamma} e(\varphi^1, \gamma)$
  - $\varphi^2 = \arg\min_{\varphi} e(\varphi, \gamma^1)$
  - $\gamma^2 = \arg\min_{\gamma} e(\varphi^2, \gamma)$
  - $\varphi^3 = \arg\min_{\varphi} e(\varphi, \gamma^2)$
  - $\gamma^3 = \arg\min_{\gamma} e(\varphi^3, \gamma)$
  - until convergence reached



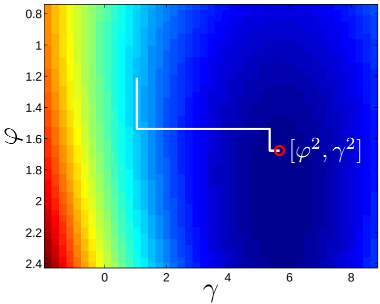
- Global optimality for linear φ, γ experimentally shown.

Algorithm: iterative minimization of criterion $e(\varphi, \gamma)$

φ – motion (geometry mapping), γ – appearance mapping

color encodes criterion value $e(\varphi, \gamma)$

- Iterative minimization:
  - initialization $\gamma^0 = \mathrm{rand}$
  - $\varphi^1 = \arg\min_{\varphi} e(\varphi, \gamma^0)$
  - $\gamma^1 = \arg\min_{\gamma} e(\varphi^1, \gamma)$
  - $\varphi^2 = \arg\min_{\varphi} e(\varphi, \gamma^1)$
  - $\gamma^2 = \arg\min_{\gamma} e(\varphi^2, \gamma)$
  - $\varphi^3 = \arg\min_{\varphi} e(\varphi, \gamma^2)$
  - $\gamma^3 = \arg\min_{\gamma} e(\varphi^3, \gamma)$
  - until convergence reached



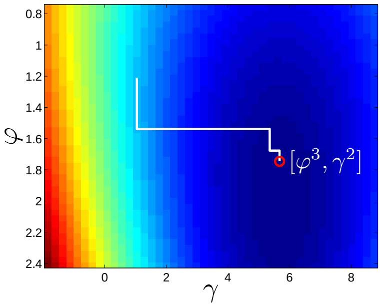
- Global optimality for linear φ, γ experimentally shown.

Algorithm: iterative minimization of criterion $e(\varphi, \gamma)$

φ – motion (geometry mapping), γ – appearance mapping

color encodes criterion value $e(\varphi, \gamma)$

- Iterative minimization:
  - initialization $\gamma^0 = \text{rand}$
  - $\varphi^1 = \arg\min_\varphi e(\varphi, \gamma^0)$
  - $\gamma^1 = \arg\min_\gamma e(\varphi^1, \gamma)$
  - $\varphi^2 = \arg\min_\varphi e(\varphi, \gamma^1)$
  - $\gamma^2 = \arg\min_\gamma e(\varphi^2, \gamma)$
  - $\varphi^3 = \arg\min_\varphi e(\varphi, \gamma^2)$
  - $\gamma^3 = \arg\min_\gamma e(\varphi^3, \gamma)$
  - until convergence reached



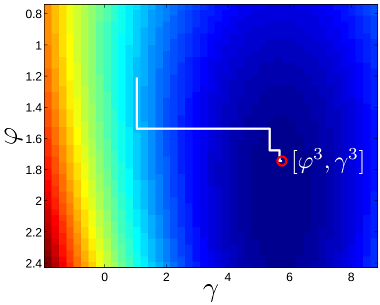
- Global optimality for linear φ, γ experimentally shown.

Algorithm: iterative minimization of criterion $e(\varphi, \gamma)$

φ – motion (geometry mapping), γ – appearance mapping

color encodes criterion value $e(\varphi, \gamma)$

- Iterative minimization:
  - initialization $\gamma^0 = \mathrm{rand}$
  - $\varphi^1 = \arg\min_\varphi e(\varphi, \gamma^0)$
  - $\gamma^1 = \arg\min_\gamma e(\varphi^1, \gamma)$
  - $\varphi^2 = \arg\min_\varphi e(\varphi, \gamma^1)$
  - $\gamma^2 = \arg\min_\gamma e(\varphi^2, \gamma)$
  - $\varphi^3 = \arg\min_\varphi e(\varphi, \gamma^2)$
  - $\gamma^3 = \arg\min_\gamma e(\varphi^3, \gamma)$
  - until convergence reached



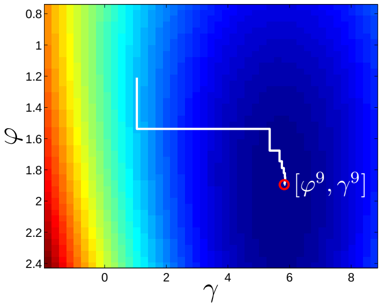
- Global optimality for linear φ, γ experimentally shown.

Algorithm: iterative minimization of criterion $e(\varphi, \gamma)$

φ – motion (geometry mapping), γ – appearance mapping

color encodes criterion value $e(\varphi, \gamma)$

- Iterative minimization:
  - initialization $\gamma^0 = \mathrm{rand}$
  - $\varphi^1 = \arg\min_\varphi e(\varphi, \gamma^0)$
  - $\gamma^1 = \arg\min_\gamma e(\varphi^1, \gamma)$
  - $\varphi^2 = \arg\min_\varphi e(\varphi, \gamma^1)$
  - $\gamma^2 = \arg\min_\gamma e(\varphi^2, \gamma)$
  - $\varphi^3 = \arg\min_\varphi e(\varphi, \gamma^2)$
  - $\gamma^3 = \arg\min_\gamma e(\varphi^3, \gamma)$
  - until convergence reached



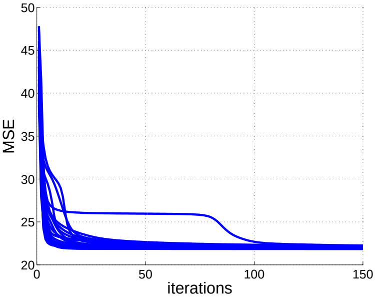
- Global optimality for linear φ, γ experimentally shown.

Algorithm: iterative minimization of criterion $e(\varphi, \gamma)$

φ – motion (geometry mapping), γ – appearance mapping

color encodes criterion value $e(\varphi, \gamma)$

- Iterative minimization:
  - initialization $\gamma^0 = \mathrm{rand}$
  - $\varphi^1 = \arg\min_{\varphi} e(\varphi, \gamma^0)$
  - $\gamma^1 = \arg\min_{\gamma} e(\varphi^1, \gamma)$
  - $\varphi^2 = \arg\min_{\varphi} e(\varphi, \gamma^1)$
  - $\gamma^2 = \arg\min_{\gamma} e(\varphi^2, \gamma)$
  - $\varphi^3 = \arg\min_{\varphi} e(\varphi, \gamma^2)$
  - $\gamma^3 = \arg\min_{\gamma} e(\varphi^3, \gamma)$
  - until convergence reached



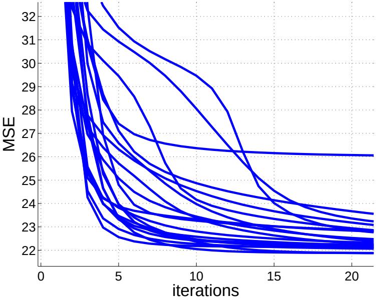
- Global optimality for linear φ, γ experimentally shown.

Algorithm: iterative minimization of criterion $e(\varphi, \gamma)$

φ – motion (geometry mapping), γ – appearance mapping

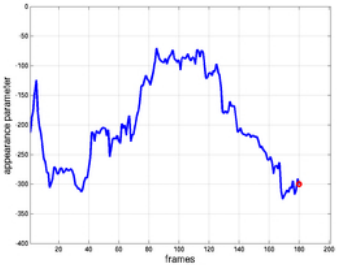
color encodes criterion value $e(\varphi, \gamma)$

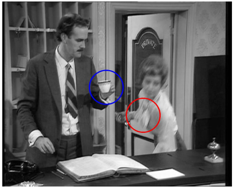
- Iterative minimization:
  - initialization $\gamma^0 = \mathrm{rand}$
  - $\varphi^1 = \arg\min_{\varphi} e(\varphi, \gamma^0)$
  - $\gamma^1 = \arg\min_{\gamma} e(\varphi^1, \gamma)$
  - $\varphi^2 = \arg\min_{\varphi} e(\varphi, \gamma^1)$
  - $\gamma^2 = \arg\min_{\gamma} e(\varphi^2, \gamma)$
  - $\varphi^3 = \arg\min_{\varphi} e(\varphi, \gamma^2)$
  - $\gamma^3 = \arg\min_{\gamma} e(\varphi^3, \gamma)$
  - until convergence reached



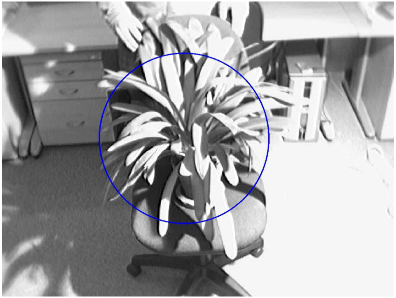
- Global optimality for linear φ, γ experimentally shown.

Simultaneous learning of motion and appearance





Experiments - videos II



Conclusions

- Learnable and very efficient tracking of objects with variable appearance.
- Accuracy, speed, robustness explicitly taken into account.
- Simultaneous learning motion and appearance.

Limitations

- Small, thin objects intractable.

Data, papers, various implemenations freely available at
http://cmp.felk.cvut.cz/demos/Tracking/linTrack/

Conclusions

- Learnable and very efficient tracking of objects with variable appearance.
- Accuracy, speed, robustness explicitly taken into account.
- Simultaneous learning motion and appearance.

Limitations

- Small, thin objects intractable.

Data, papers, various implemenations freely available at
http://cmp.felk.cvut.cz/demos/Tracking/linTrack/