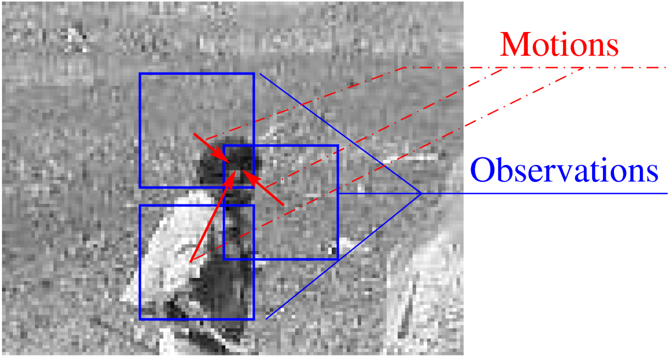
# Tracking by Regression

Tomáš Svoboda

Department of Cybernetics, Center for Machine Perception

March 31, 2014

# Linear mapping for tracking




$\Phi($$) = (0, 0)^T$ $\Phi($$) = (12, 7)^T$

$\Phi($$) = (-14, 2)^T$ $\Phi($$) = (-9, 18)^T$

$\Phi($$) = (14, -14)^T$ $\Phi($$) = (-16, -12)^T$

# Optimization vs. regression based tracking

KLT:

$$\Delta \mathbf{p}^* = \operatorname*{argmin}_{\Delta \mathbf{p} \in S} \sum_{\mathbf{x}} [I(\mathbf{W}(\mathbf{x}; \mathbf{p} + \Delta \mathbf{p})) - T(\mathbf{x})]^2$$

Optimization in general:

$$\mathbf{p}^* = \operatorname*{argmin}_{\mathbf{p} \in S} f(\mathbf{p}; I, \mathbf{p}_0),$$

Regresssion based tracking:

$$\mathbf{p}^* = \varphi(I, \mathbf{p}_0)$$

# Optimization vs. regression based tracking

KLT:

$$\Delta \mathbf{p}^* = \operatorname*{argmin}_{\Delta \mathbf{p} \in S} \sum_{\mathbf{x}} \left[ I(\mathbf{W}(\mathbf{x}; \mathbf{p} + \Delta \mathbf{p})) - T(\mathbf{x}) \right]^2$$

Optimization in general:

$$\mathbf{p}^* = \operatorname*{argmin}_{\mathbf{p} \in S} f(\mathbf{p}; I, \mathbf{p}_0),$$

Regresssion based tracking:

$$\mathbf{p}^* = \varphi(I, \mathbf{p}_0)$$

# Optimization vs. regression based tracking

KLT:

$$\Delta \mathbf{p}^* = \operatorname*{argmin}_{\Delta \mathbf{p} \in S} \sum_{\mathbf{x}} [I(\mathbf{W}(\mathbf{x}; \mathbf{p} + \Delta \mathbf{p})) - T(\mathbf{x})]^2$$

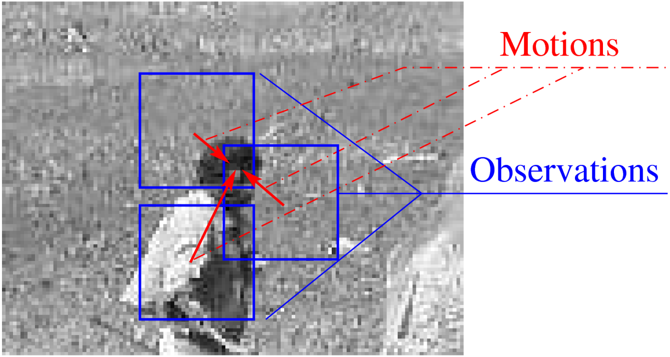
Optimization in general:

$$\mathbf{p}^* = \operatorname*{argmin}_{\mathbf{p} \in S} f(\mathbf{p}; I, \mathbf{p}_0),$$

Regresssion based tracking:

$$\mathbf{p}^* = \varphi(I, \mathbf{p}_0)$$

# Regression learned from data - sample image




$\Phi($$) = (0,0)^T$ $\Phi($$) = (12,7)^T$

$\Phi($$) = (-14,2)^T$ $\Phi($$) = (-9,18)^T$

$\Phi($$) = (14,-14)^T$ $\Phi($$) = (-16,-12)^T$

# Regression learning

$$\varphi^* = \operatorname*{argmin}_{\varphi} \sum_{\mathbf{p}} \|\varphi\Big(I(\mathbf{p} \circ \mathbf{x})\Big) - \mathbf{p}\|.$$

# Connection to KLT

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

where:

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

Reformulating *regression*:

$$\mathbf{p} = \varphi\Big(I(\mathbf{x})\Big) = \mathtt{H}\Big(I(\mathbf{x}) - T(\mathbf{x})\Big)$$

How to get $\mathtt{H}$?

# Connection to KLT

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

where:

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

Reformulating *regression*:

$$\mathbf{p} = \varphi\Big(I(\mathbf{x})\Big) = \mathtt{H}\Big(I(\mathbf{x}) - T(\mathbf{x})\Big)$$

How to get $\mathtt{H}$?

# Connection to KLT

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

where:

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

Reformulating *regression*:

$$\mathbf{p} = \varphi\Big( I(\mathbf{x}) \Big) = \mathtt{H}\Big( I(\mathbf{x}) - T(\mathbf{x}) \Big)$$

How to get $\mathtt{H}$?

# Collecting training data

Image template: $T = T(\mathbf{x})$
Collected training pairs $(I^i,\ \mathbf{p}^i)$ $(i = 1 \ldots d)$ of observed intensities $I^i$ and corresponding motion parameters $\mathbf{p}^i$, which align the object with current frame.

*Training set* is an ordered pair $(\mathtt{I}, \mathtt{P})$, such that $\mathtt{I} = [\mathbf{I}^1 - \mathbf{T}, \mathbf{I}^2 - \mathbf{T}, \ldots \mathbf{I}^d - \mathbf{T}]$ and $\mathtt{P} = [\mathbf{p}^1, \mathbf{p}^2, \ldots \mathbf{p}^d]$.

The bolds $\mathbf{I}, \mathbf{T}$ are vectorized $I$, $T$. Think about: $\mathbf{I} = I(:)$

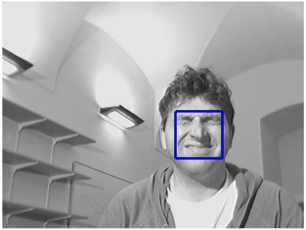
# Collecting training data

Image template: $T = T(\mathbf{x})$
Collected training pairs $(I^i,\ \mathbf{p}^i)$ $(i = 1 \dots d)$ of observed intensities $I^i$ and corresponding motion parameters $\mathbf{p}^i$, which align the object with current frame.

*Training set* is an ordered pair $(\mathtt{I}, \mathtt{P})$, such that
$\mathtt{I} = [\mathbf{I}^1 - \mathbf{T}, \mathbf{I}^2 - \mathbf{T}, \dots \mathbf{I}^d - \mathbf{T}]$ and $\mathtt{P} = [\mathbf{p}^1, \mathbf{p}^2, \dots \mathbf{p}^d]$.

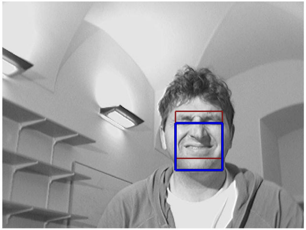
The bolds $\mathbf{I}, \mathbf{T}$ are vectorized $I$, $T$. Think about: $\mathbf{I} = I\mathtt{(:)}$

# Learning alignment for one predictor



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

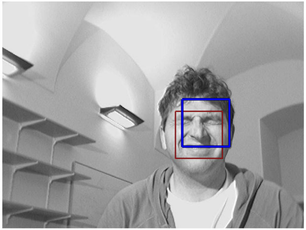
# Learning alignment for one predictor



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

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

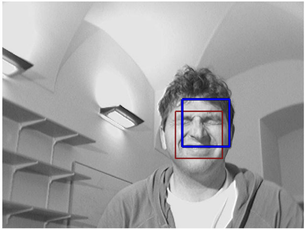
# Learning alignment for one predictor



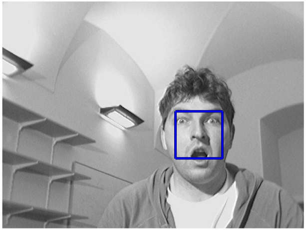
- $\varphi($  $) = (0, 0)^\top$
- $\varphi($  $) = (-25, 0)^\top$
- $\varphi($  $) = (25, -15)^\top$


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

# Learning alignment for one predictor

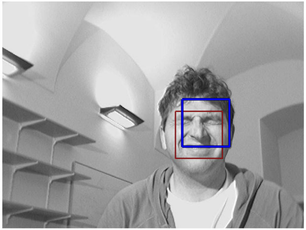


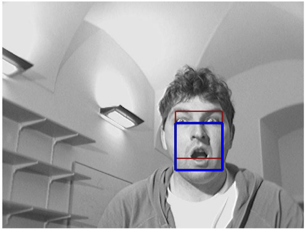
- $\varphi($  $) = (0,0)^\top$
- $\varphi($  $) = (-25,0)^\top$
- $\varphi($  $) = (25,-15)^\top$



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

# Learning alignment for one predictor

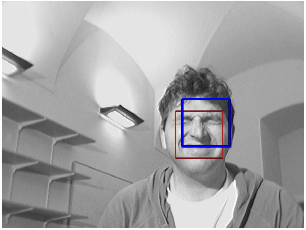




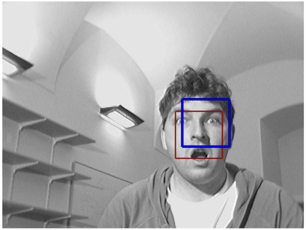
- $\varphi($  $) = (0, 0)^\top$
- $\varphi($  $) = (-25, 0)^\top$
- $\varphi($  $) = (25, -15)^\top$

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

# Learning alignment for one predictor

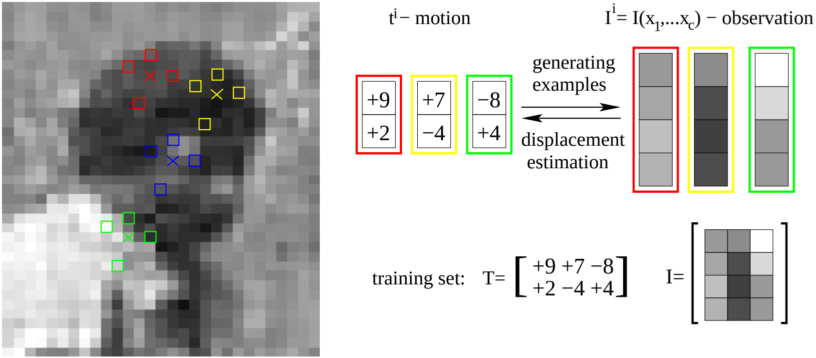


- $\varphi($  $)= (0,0)^\top$
- $\varphi($  $)= (-25,0)^\top$
- $\varphi($  $)= (25,-15)^\top$



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

# Generating training examples




Training set: $(\mathtt{I}, \mathtt{P})$
$\mathtt{I} = [\mathbf{I}^1 - \mathbf{T}, \mathbf{I}^2 - \mathbf{T}, \ldots \mathbf{I}^d - \mathbf{T}]$ and $\mathtt{P} = [\mathbf{p}^1, \mathbf{p}^2, \ldots \mathbf{p}^d]$.

# Learning $\mathtt{H}$ from data

$$
\begin{aligned}
\mathtt{H}^* &= \operatorname*{argmin}_{\mathtt{H}} \sum_{i=1}^{d} \|\mathtt{H}(\mathbf{I}^i - \mathbf{T}) - \mathbf{p}^i\|_2^2 = \operatorname*{argmin}_{\mathtt{H}} \|\mathtt{HI} - \mathtt{P}\|_F^2 = \\
&= \operatorname*{argmin}_{\mathtt{H}} \operatorname{trace}(\mathtt{HI} - \mathtt{P})(\mathtt{HI} - \mathtt{P})^\top = \\
&= \operatorname*{argmin}_{\mathtt{H}} \operatorname{trace}(\mathtt{HII}^\top\mathtt{H}^\top - 2\mathtt{HIP}^\top + \mathtt{PP}^\top).
\end{aligned}
$$

Next its derivative is set equal to zero:

$$
\begin{aligned}
2\mathtt{H}^*\mathtt{II}^\top - 2\mathtt{PI}^\top &= 0 \\
\mathtt{H}^*\mathtt{II}^\top &= \mathtt{PI}^\top \\
\mathtt{H}^* &= \mathtt{P}\underbrace{\mathtt{I}^\top(\mathtt{II}^\top)^{-1}}_{\mathtt{I}^+} = \mathtt{PI}^+.
\end{aligned}
$$

# Learning $\mathtt{H}$ from data

$$
\begin{aligned}
\mathtt{H}^* &= \operatorname*{argmin}_{\mathtt{H}} \sum_{i=1}^{d} \|\mathtt{H}(\mathbf{I}^i - \mathbf{T}) - \mathbf{p}^i\|_2^2 = \operatorname*{argmin}_{\mathtt{H}} \|\mathtt{HI} - \mathtt{P}\|_F^2 = \\
&= \operatorname*{argmin}_{\mathtt{H}} \operatorname{trace}(\mathtt{HI} - \mathtt{P})(\mathtt{HI} - \mathtt{P})^\top = \\
&= \operatorname*{argmin}_{\mathtt{H}} \operatorname{trace}(\mathtt{HII}^\top\mathtt{H}^\top - 2\mathtt{HIP}^\top + \mathtt{PP}^\top).
\end{aligned}
$$

Next its derivative is set equal to zero:

$$
\begin{aligned}
2\mathtt{H}^*\mathtt{II}^\top - 2\mathtt{PI}^\top &= 0 \\
\mathtt{H}^*\mathtt{II}^\top &= \mathtt{PI}^\top \\
\mathtt{H}^* &= \mathtt{P}\underbrace{\mathtt{I}^\top(\mathtt{II}^\top)^{-1}}_{\mathtt{I}^+} = \mathtt{PI}^+.
\end{aligned}
$$

# LS learning - summary

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

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

Example for *linear mapping*:

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

after some derivation

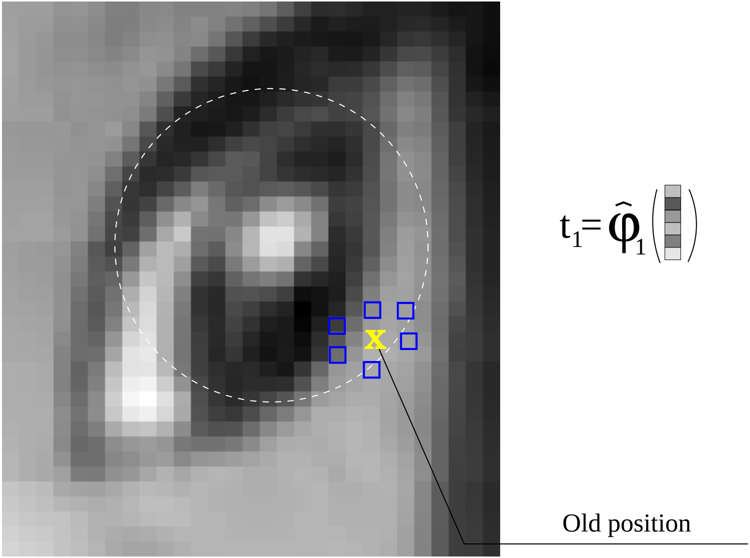
$$\mathtt{H}^* = \mathtt{P}\underbrace{\mathtt{I}^\top(\mathtt{II}^\top)^{-1}}_{\mathtt{I}^+} = \mathtt{PI}^+.$$

# Min-max learning

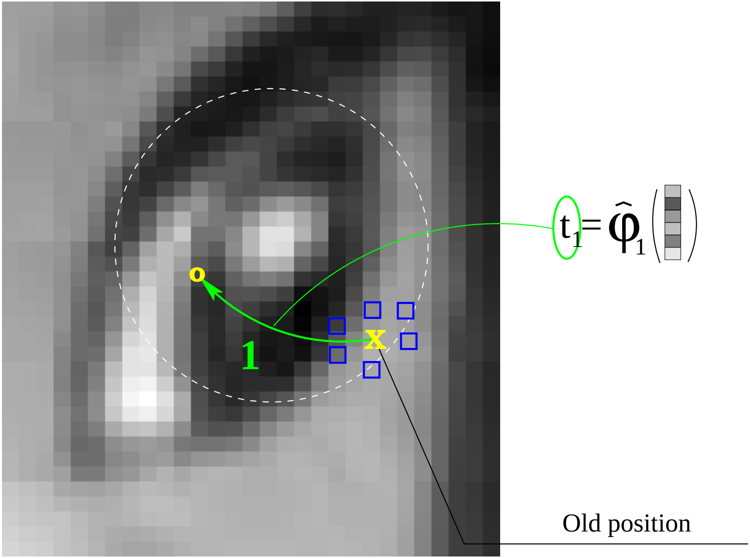
$$\varphi^* = \mathrm{argmin}_{\varphi} \max_{\mathbf{p}} \left\|\varphi\Big(I(\mathbf{p} \circ \mathbf{x})\Big) - \mathbf{p}\right\|_\infty .$$

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

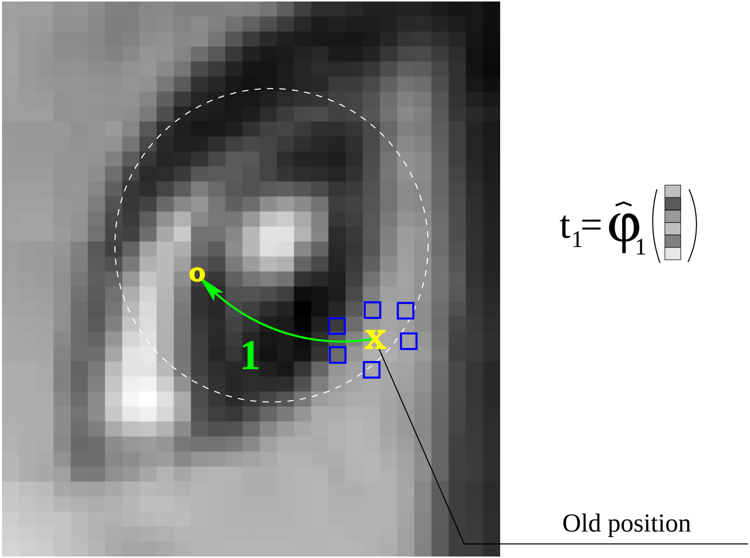
# Tracking of a single point by a *sequence* of predictors



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

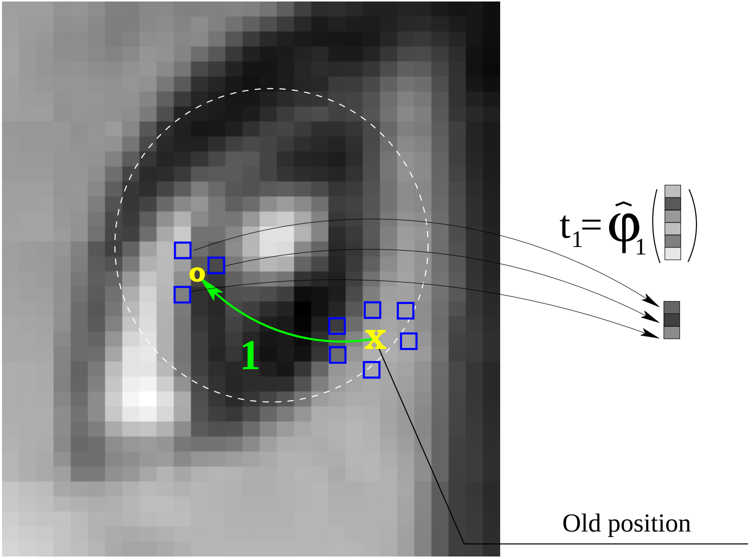


# Tracking of a single point by a *sequence* of predictors

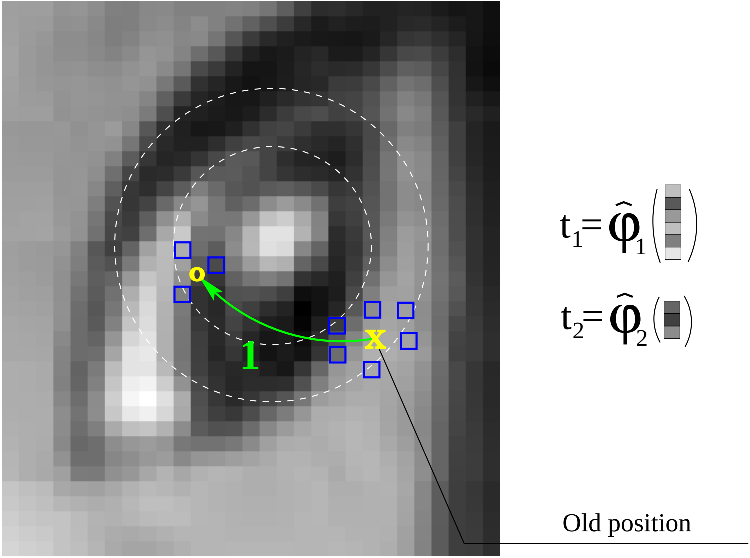



$$t_1 = \hat{\varphi}_1\left(\,\cdot\,\right)$$

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



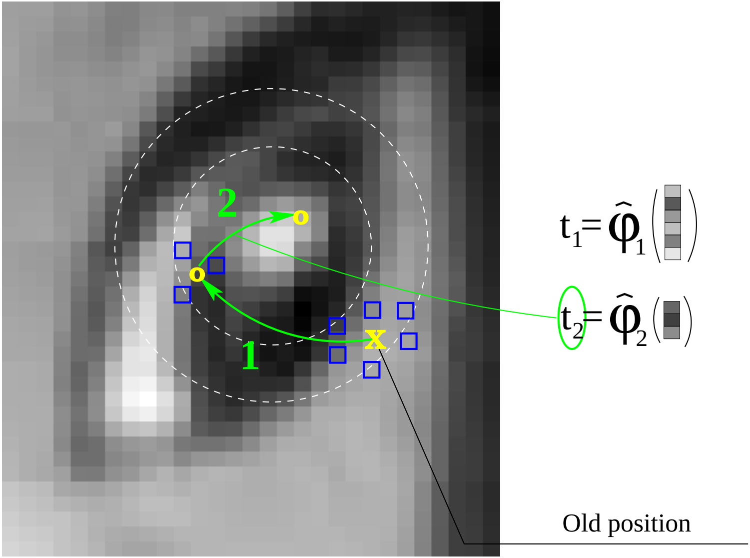
# Tracking of a single point by a *sequence* of predictors



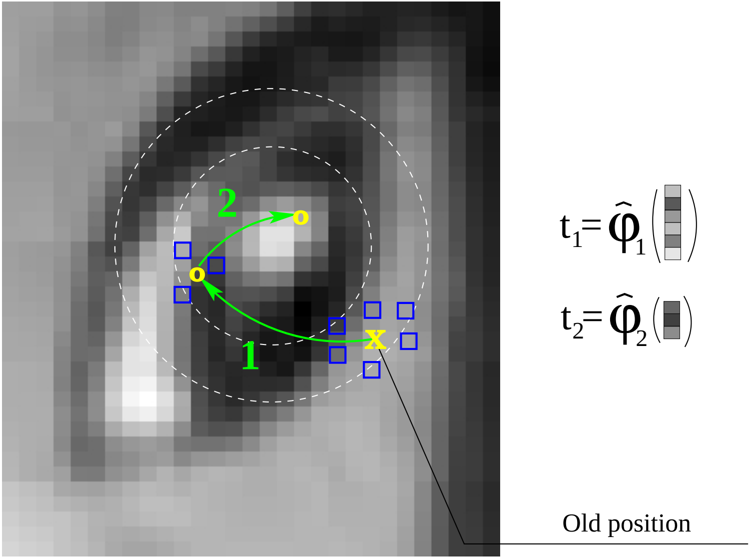

$t_1 = \hat{\varphi}_1(\cdot)$

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

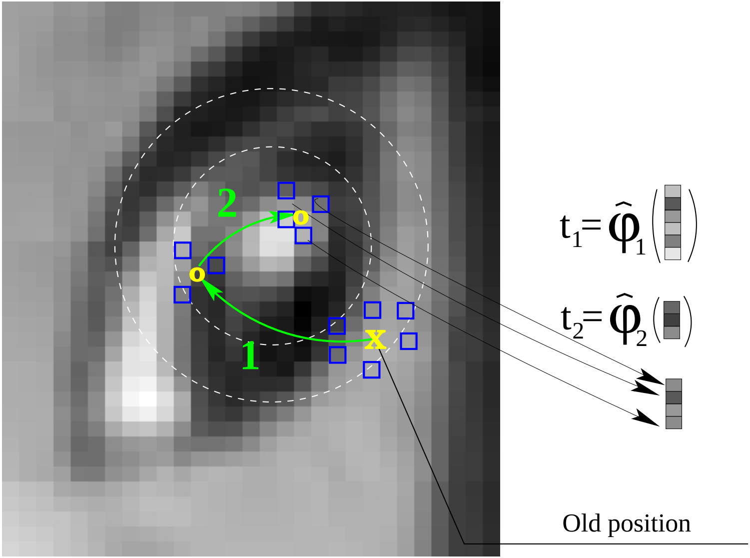
# Tracking of a single point by a *sequence* of predictors



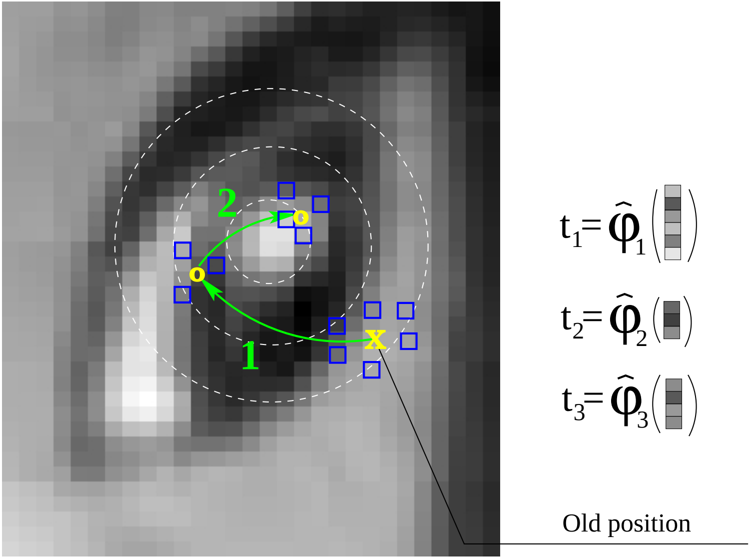
# Tracking of a single point by a *sequence* of predictors



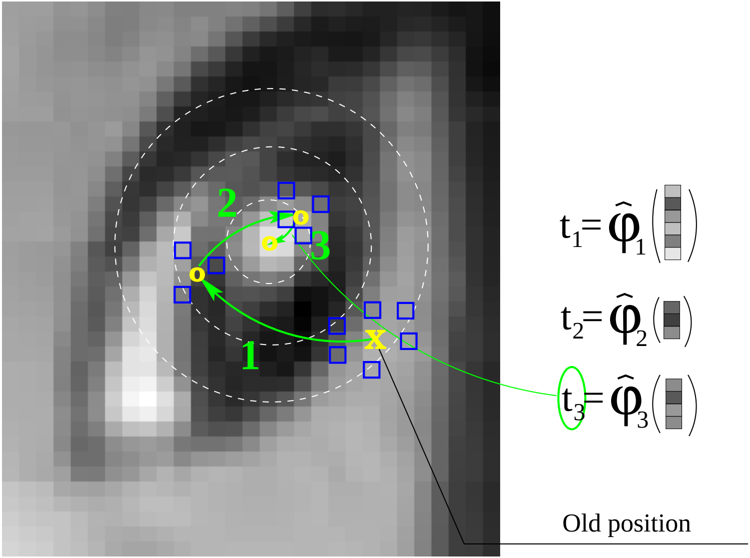
# Tracking of a single point by a *sequence* of predictors



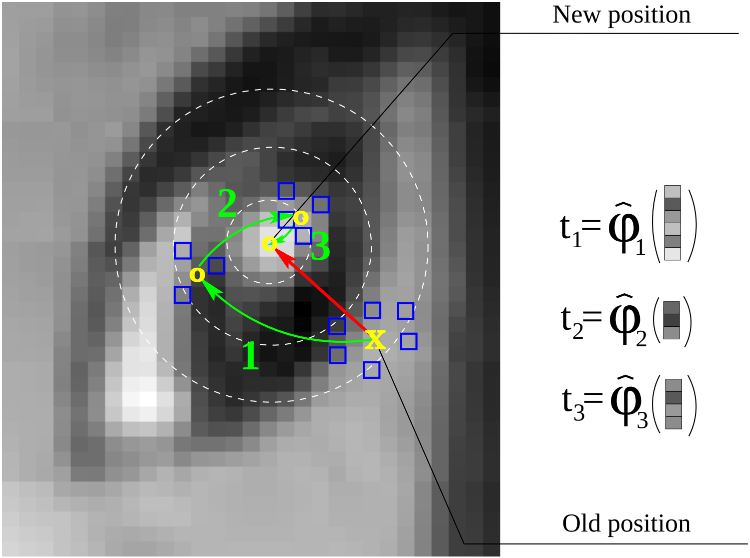
# Tracking of a single point by a *sequence* of predictors



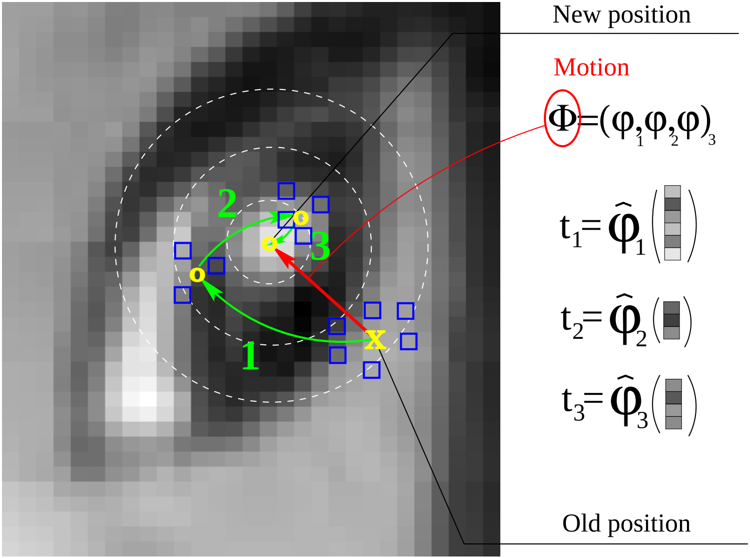
# Tracking of a single point by a *sequence* of predictors



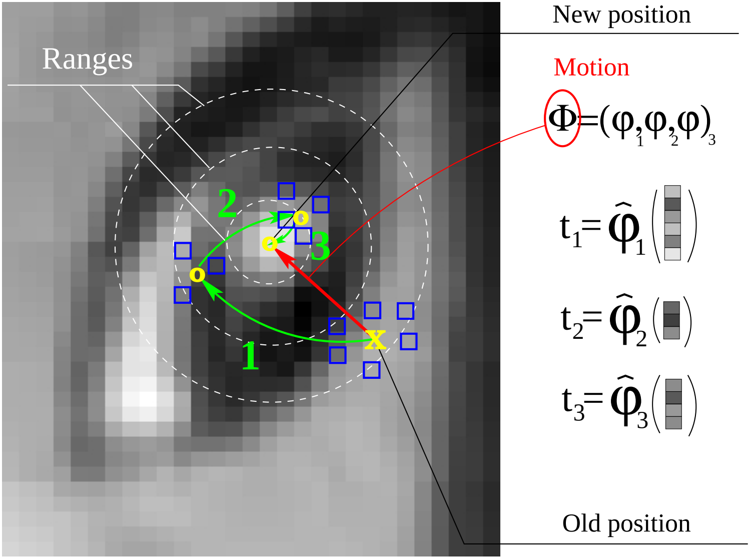
# Tracking of a single point by a *sequence* of predictors



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



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



# 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*, vol. 27, No 11

# 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*, vol. 27, No 11

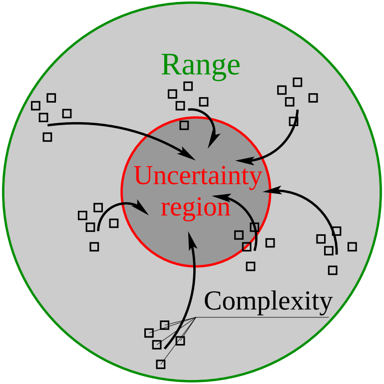
# Learning of sequential predictor

- **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).

---

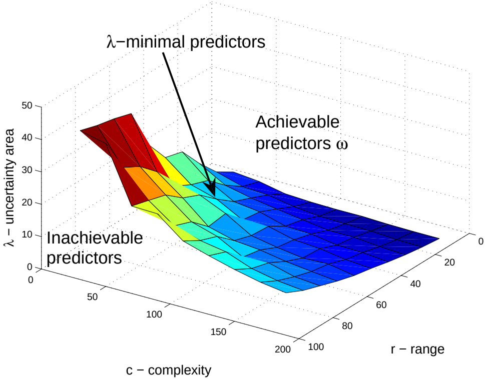
[Zimmermann-PAMI-2009] K.Zimmermann, J.Matas, T.Svoboda. Tracking by an Optimal Sequence of Linear Predictors, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE computer society, 2009, vol. 31, No 4, pp 677–692.
[Zimmermann-IVC-2009] K.Zimmermann, T.Svoboda, J.Matas. Anytime learning for the NoSLLiP tracker. *Image and Vision Computing*, vol. 27, No 11

# Learning of the optimal sequence of linear predictors



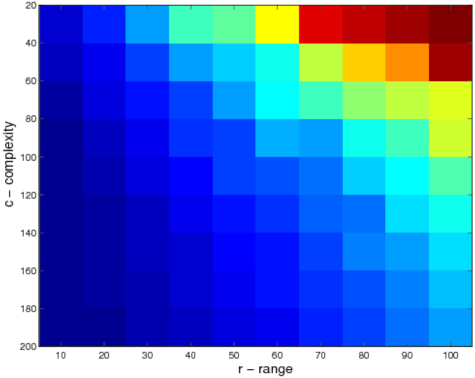

- **Range**: the set of admissible motions, $r$.
- **Complexity**: cardinality of support set, $c$.
- **Uncertainty region**: the region within which all predictions lie, $\lambda$. Small red circles show acceptable uncertainty.

# Learning of the optimal sequence of linear predictors



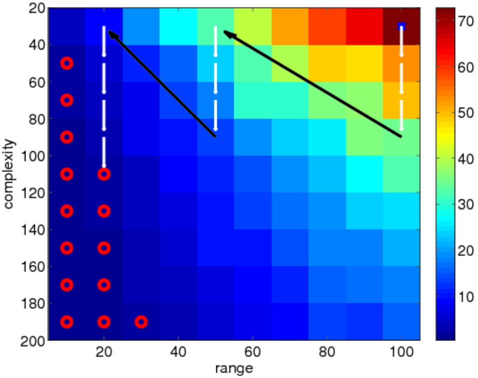

- **Range**: the set of admissible motions, $r$.
- **Complexity**: cardinality of support set, $c$.
- **Uncertainty region**: the region within which all predictions lie, $\lambda$. Small red circles show acceptable uncertainty.

# Learning of the optimal sequence of linear predictors



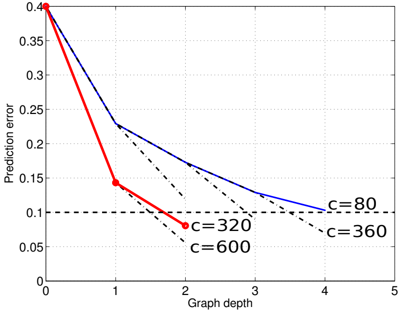

- **Range**: the set of admissible motions, $r$.
- **Complexity**: cardinality of support set, $c$.
- **Uncertainty region**: the region within which all predictions lie, $\lambda$. Small red circles show acceptable uncertainty.

# Learning of the optimal sequence of linear predictors



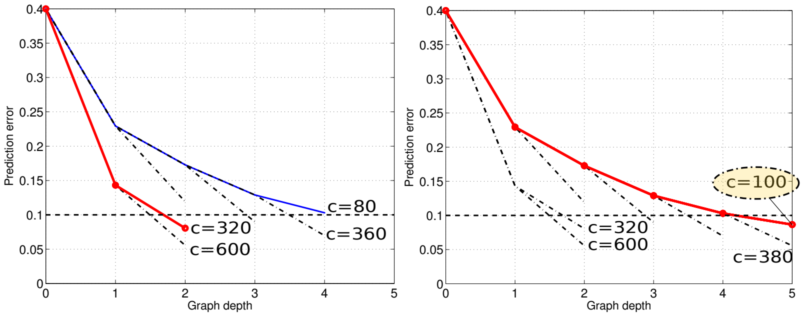

- **Range**: the set of admissible motions, $r$.
- **Complexity**: cardinality of support set, $c$.
- **Uncertainty region**: the region within which all predictions lie, $\lambda$. Small red circles show acceptable uncertainty.

# Branch and Bound



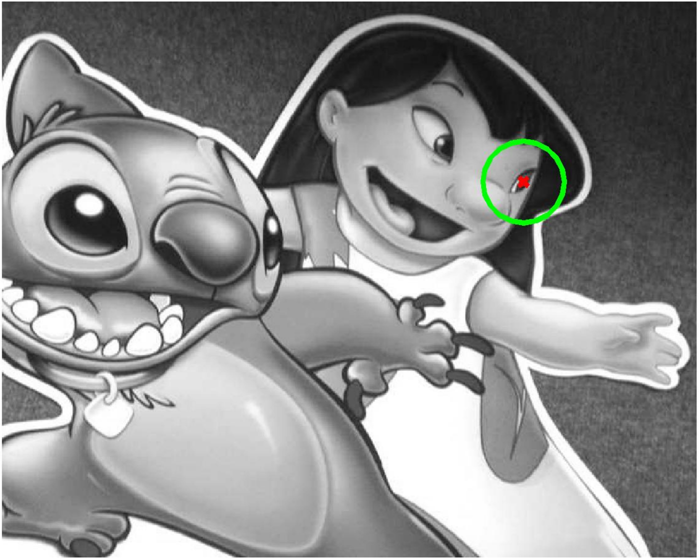

Don't forget to show the live demo!

# Branch and Bound

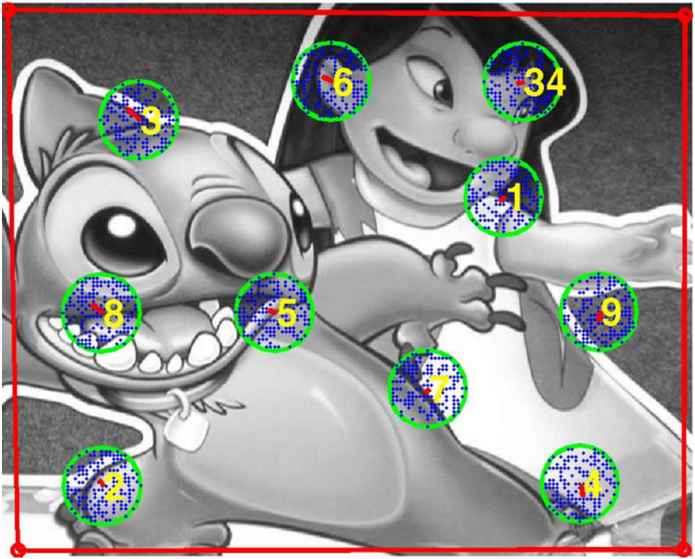



Don't forget to show the live demo!

# Tracking with one linear predictor.



# Modeling motion by number of linear predictors.



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