Advanced Applications of the LK Algorithm

Instructor - Simon Lucey

16-623 - Designing Computer Vision Apps

Carnegie Mellon
THE ROBOTICS INSTITUTE

Today

- Theoretical Limits on Search
- Inverse Composition & SDM
- Conditional LK

3 Biggest Problems in Computer Vision?

3 Biggest Problems in Computer Vision?

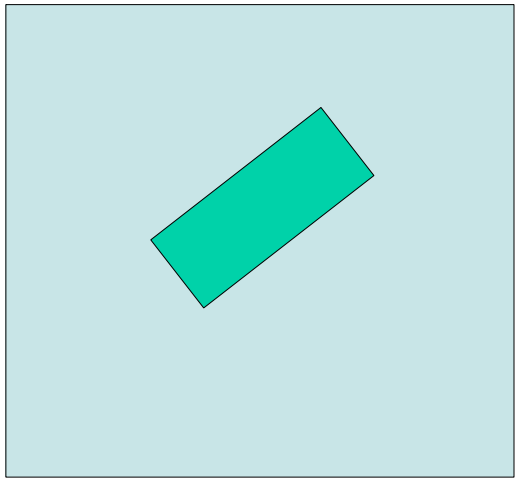
REGISTRATION

REGISTRATION

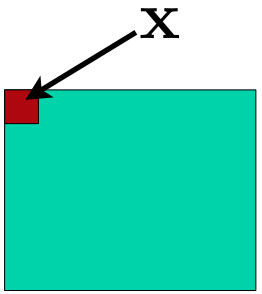
REGISTRATION

(Prof. Takeo Kanade - Robotics Institute - CMU.)

What is Registration?

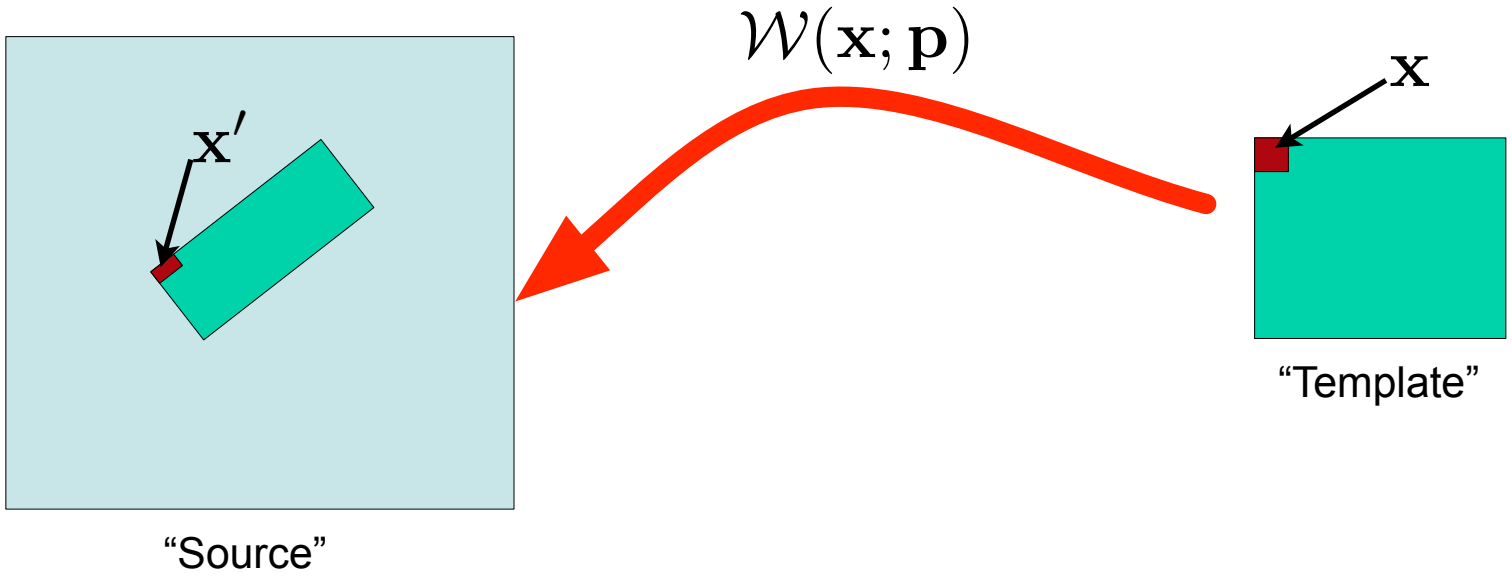


“Source”



“Template”

What is Registration?



Our goal is to find the warp parameter vector $\mathbf{p}$!

$\mathbf{x}$ = coordinate in template $[x, y]^T$

$\mathbf{x}'$ = corresponding coordinate in source $[x', y']^T$

$\mathcal{W}(\mathbf{x};\mathbf{p})$ = warping function such that $\mathbf{x}' = \mathcal{W}(\mathbf{x};\mathbf{p})$

$\mathbf{p}$ = parameter vector describing warp

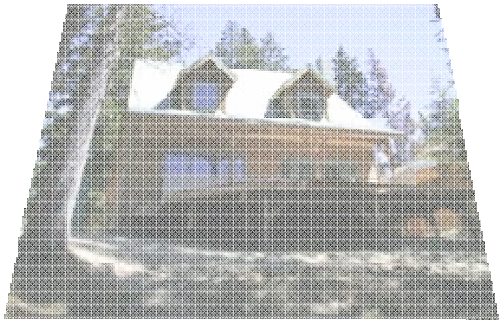
Different Warp Functions

translation

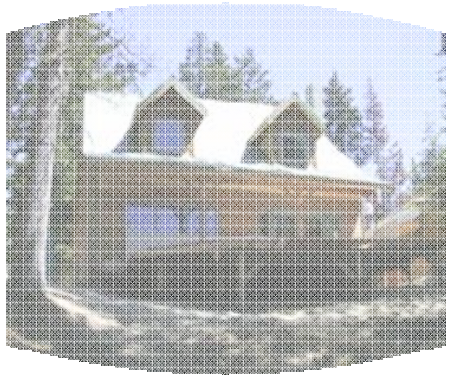
rotation

aspect

affine



perspective



cylindrical

(Szeliski and Fleet)

Defining Warp Functions

$$\mathcal{W}(\mathbf{x};\mathbf{p}) = \mathbf{x} + \mathbf{p}$$

$$\mathbf{p} = [p_1 \; p_2]^T$$

translation

$$\mathcal{W}(\mathbf{x};\mathbf{p}) = \mathbf{M}\begin{bmatrix}\mathbf{x}\\1\end{bmatrix}$$

$$\mathbf{M} = \begin{bmatrix}1-p_1 & p_2 & p_3\\ p_4 & 1-p_5 & p_6\end{bmatrix}$$

$$\mathbf{p} = [p_1 \ldots p_6]^T$$

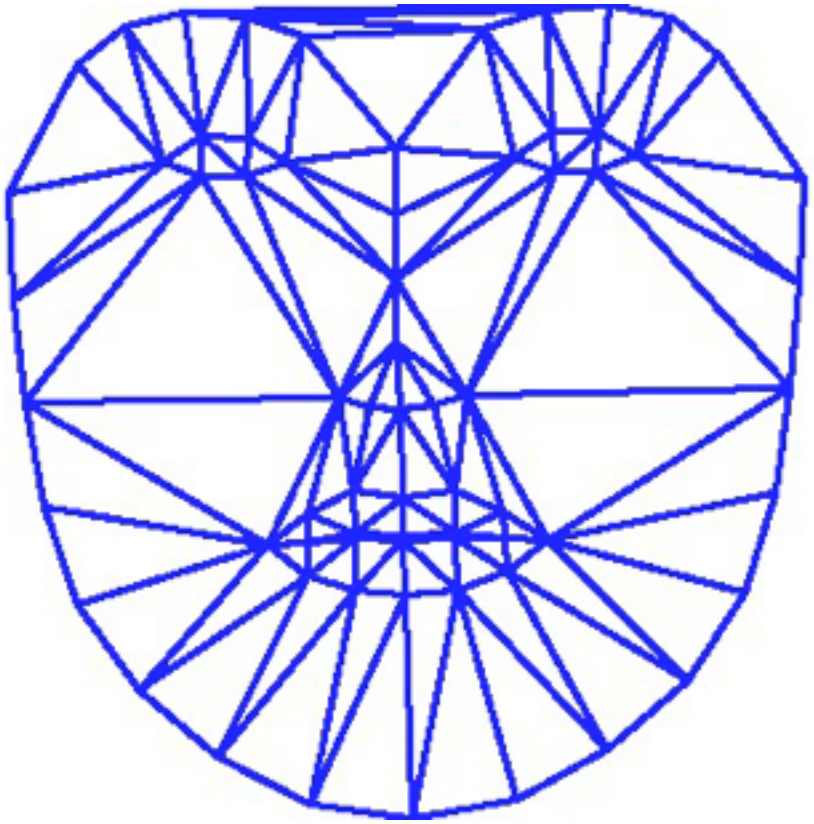
affine

rotation

aspect

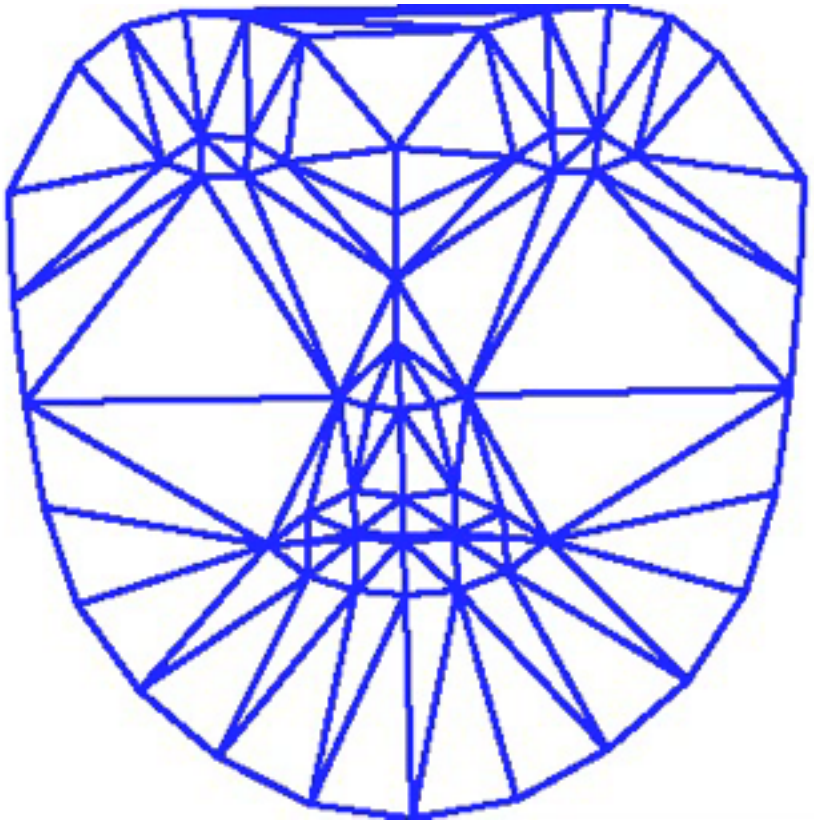
translation

Defining Warp Functions



$$\mathcal{W}(\mathbf{x};\mathbf{p})$$

Defining Warp Functions



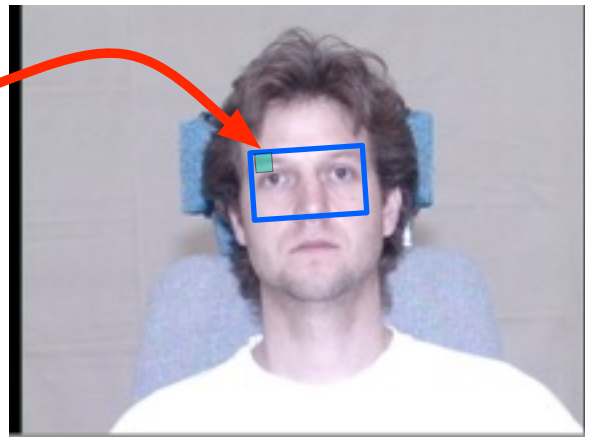
$$\mathcal{W}(\mathbf{x};\mathbf{p})$$

Measures of Image Similarity

- Although not always perfect, a common measure of image similarity is:
  - Sum of squared differences (SSD)

$$\mathrm{SSD}(\mathbf{p}) = \sum_{i=1}^{N} ||\mathcal{I}\{\mathcal{W}(\mathbf{x}_i;\mathbf{p})\} - \mathcal{T}(\mathbf{x}_i)||_2^2$$

$\mathcal{W}(\mathbf{x}_1;\mathbf{p})$



$\mathcal{I}$

"Source Image"

$\mathcal{W}(\mathbf{x}_1;\mathbf{0})$

$\mathcal{T}$

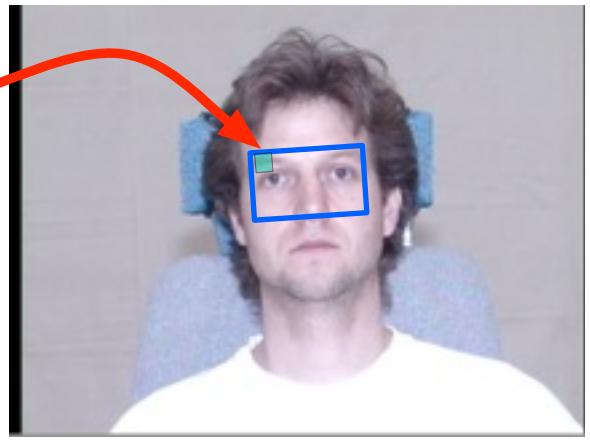
"Template"

Measures of Image Similarity

- Although not always perfect, a common measure of image similarity is:
  - Sum of squared differences (SSD)

$$\mathrm{SSD}(\mathbf{p}) = \sum_{i=1}^{N} ||\mathcal{I}\{\mathcal{W}(\mathbf{x}_i;\mathbf{p})\} - \mathcal{T}(\mathbf{x}_i)||_2^2$$

$\mathcal{W}(\mathbf{x}_1;\mathbf{p})$



$\mathcal{I}$

"Source Image"

$\mathbf{x}_1$

$\mathcal{T}$

"Template"

Measures of Image Similarity

- Although not always perfect, a common measure of image similarity is:
  - Sum of squared differences (SSD)

$$\mathrm{SSD}(\mathbf{p}) = ||\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})||_2^2$$

"Vector Form"

$$\begin{bmatrix} \mathcal{W}(\mathbf{x}_1;\mathbf{p}) \\ \vdots \\ \mathcal{W}(\mathbf{x}_N;\mathbf{p}) \end{bmatrix}$$

$\mathcal{I}$

"Source Image"

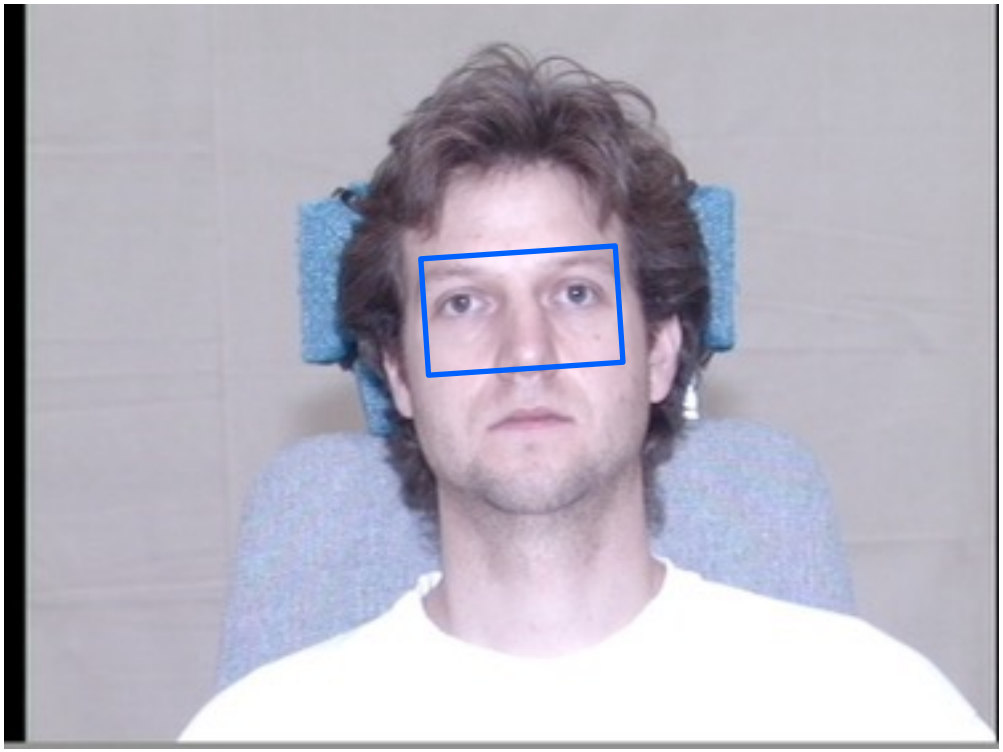
$$\begin{bmatrix} \mathcal{W}(\mathbf{x}_1;\mathbf{0}) \\ \vdots \\ \mathcal{W}(\mathbf{x}_N;\mathbf{0}) \end{bmatrix}$$

$\mathcal{T}$

"Template"

Naive Approach to Registration

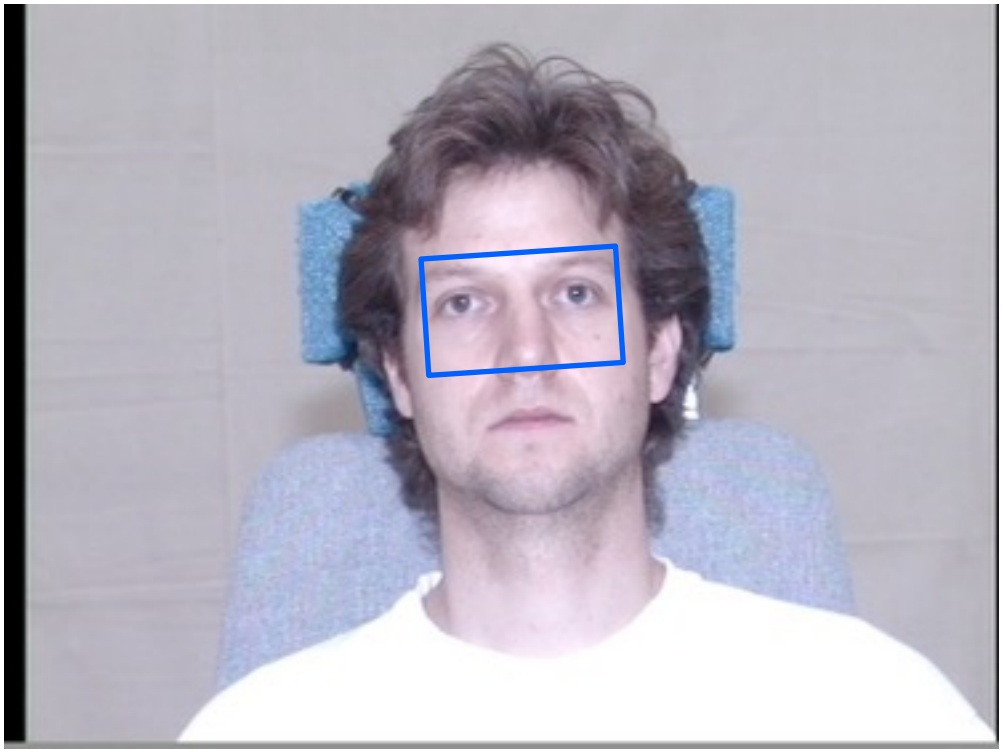
- If you were a person coming straight from machine learning you might suggest,



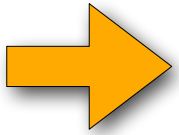
“Images of Object at various warps”

Naive Approach to Registration

- If you were a person coming straight from machine learning you might suggest,



"Images of Object at various warps"



[255,134,45,.....,34,12,124,67]

[123,244,12,.....,134,122,24,02]

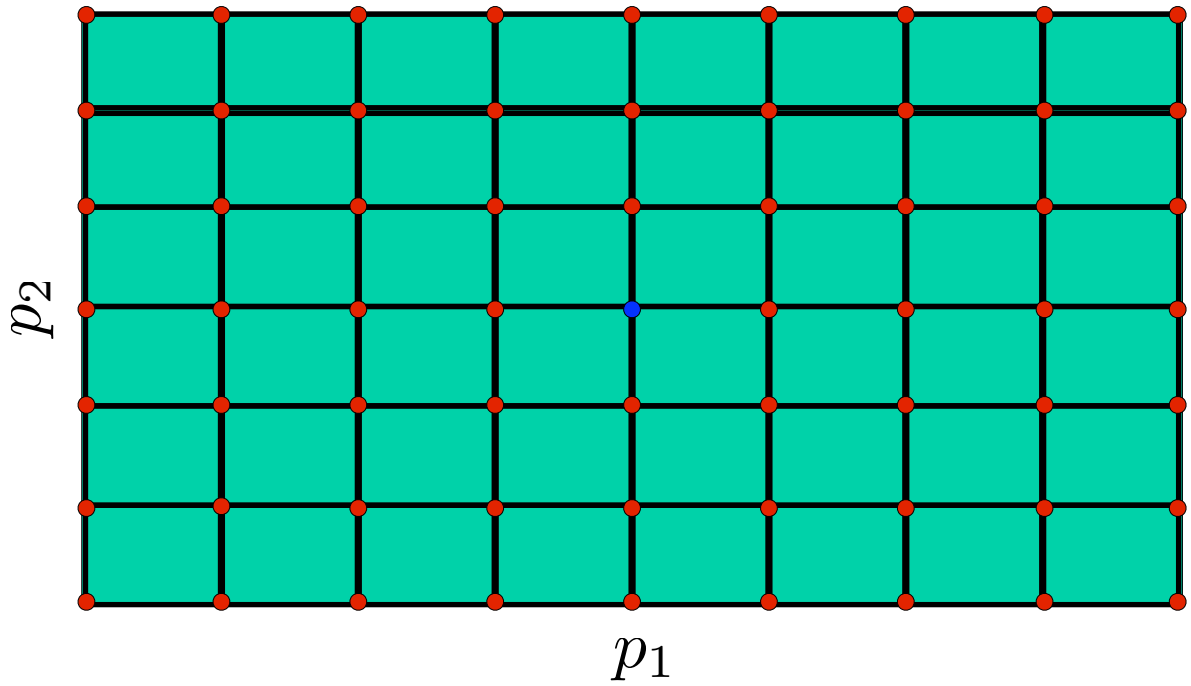
[67,13,245,.....,112,51,92,181]

[65,09,67,.....,78,66,76,215]

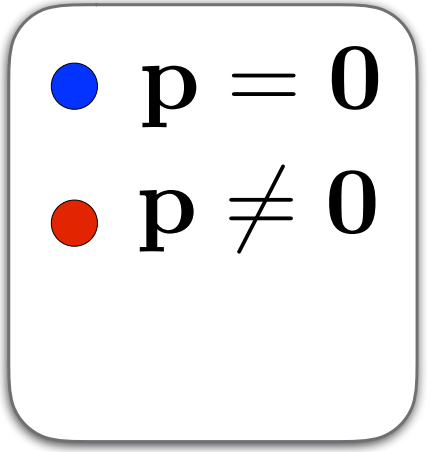
"Vectors of pixel values at each warp position"

Exhaustive Search

$$\arg\min_{\mathbf{p}} ||\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})||_2^2$$



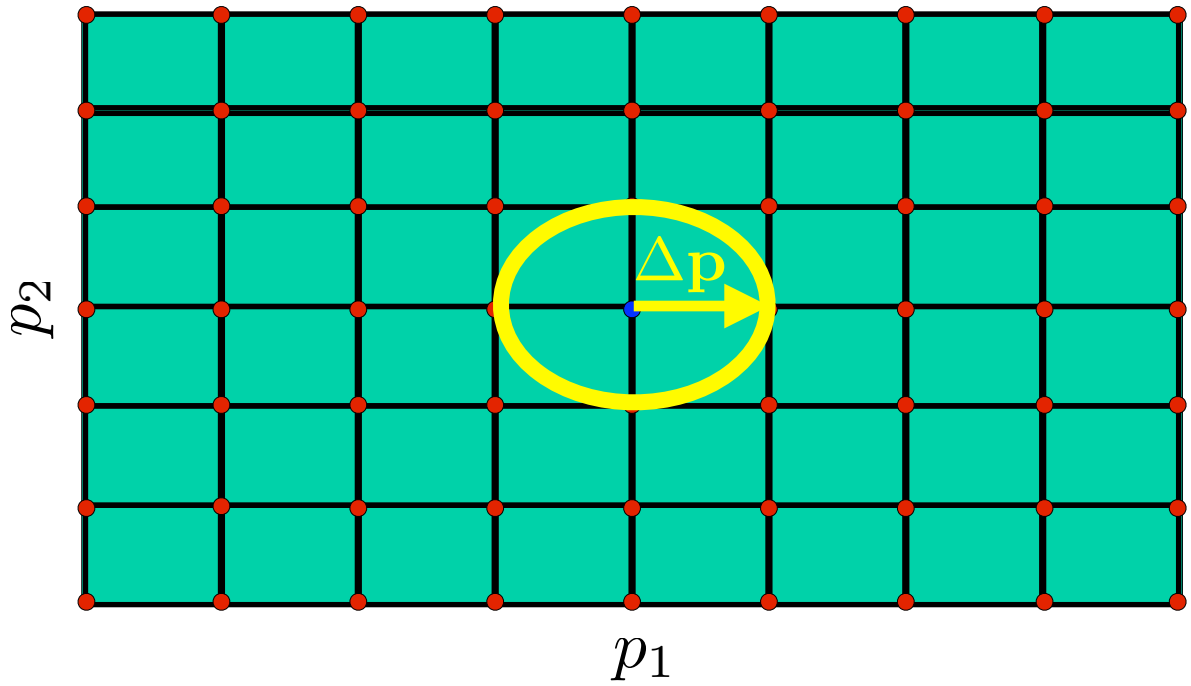
"Possible Source Warps"



$$\mathbf{p} = \{p_1, p_2\}$$

Exhaustive Search

$$\arg\min_{\mathbf{p}} ||\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})||_2^2$$



"Possible Source Warps"

$\mathbf{p} = 0$

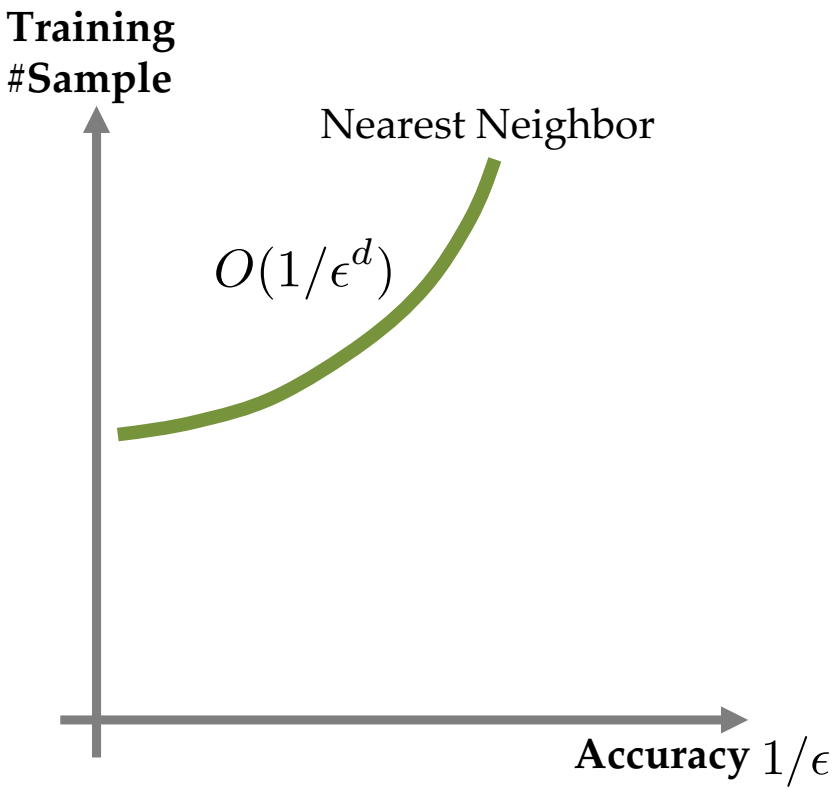
$\mathbf{p} \neq 0$

$||\Delta\mathbf{p}||_2 = \epsilon$

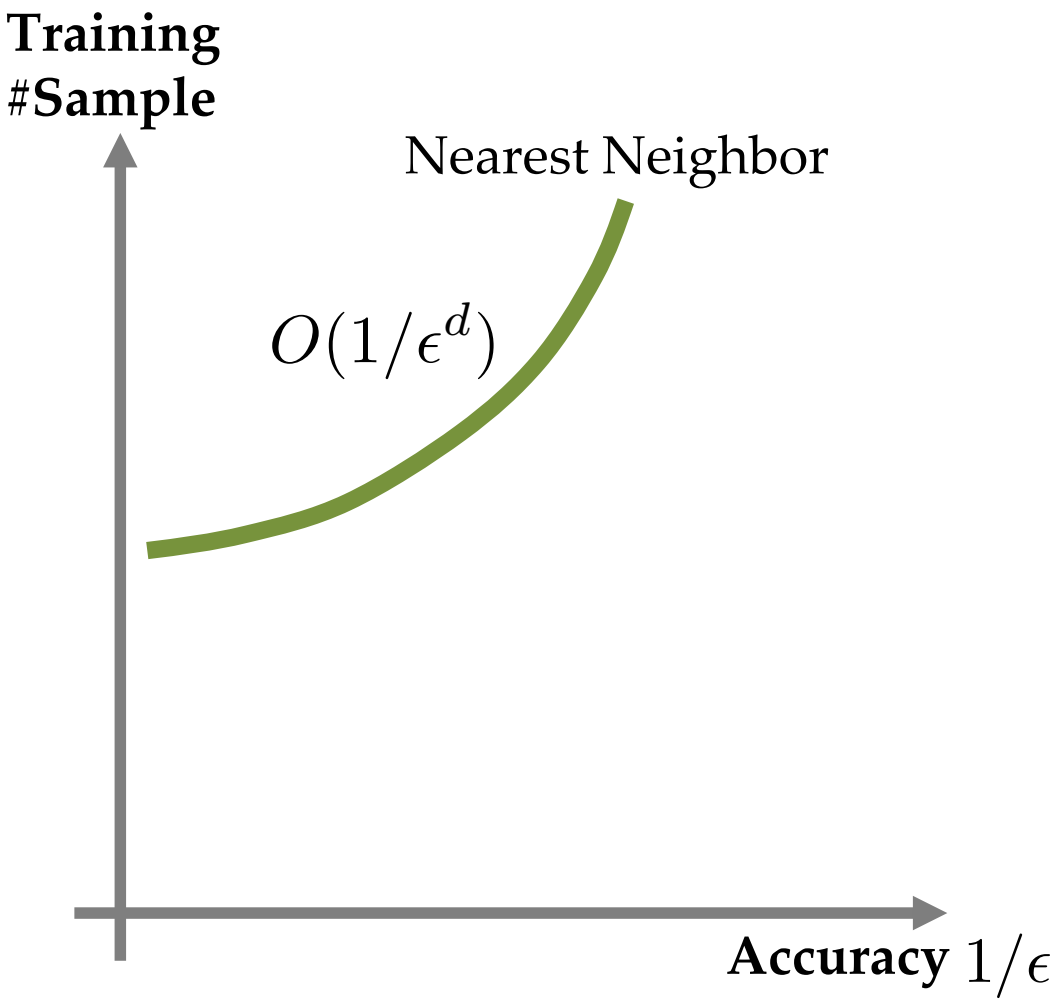
$$\mathbf{p} = \{p_1, p_2\}$$

Exhaustive Search

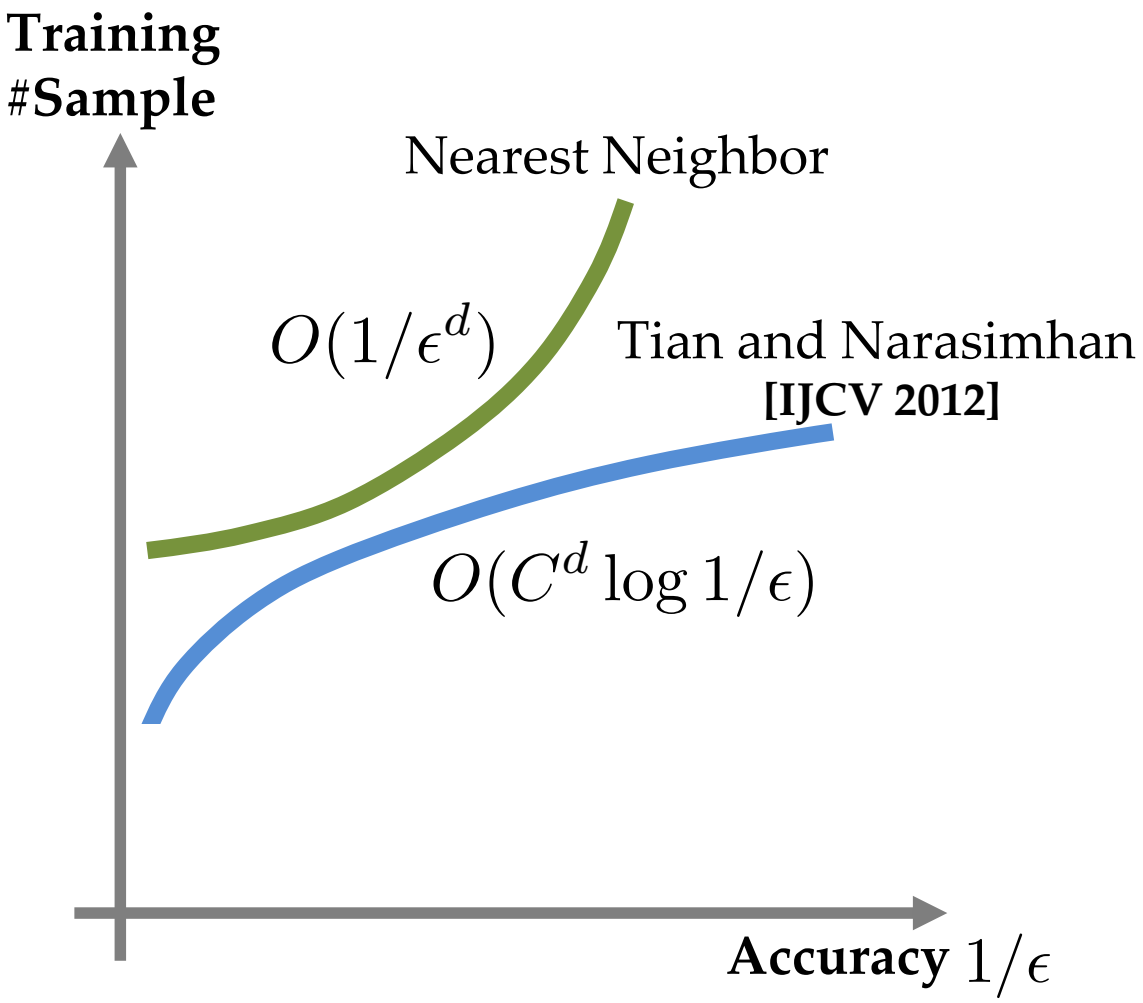
- One can see that as accuracy $1/\epsilon$ linearly increases the number of required training samples increases exponentially as a function of d.



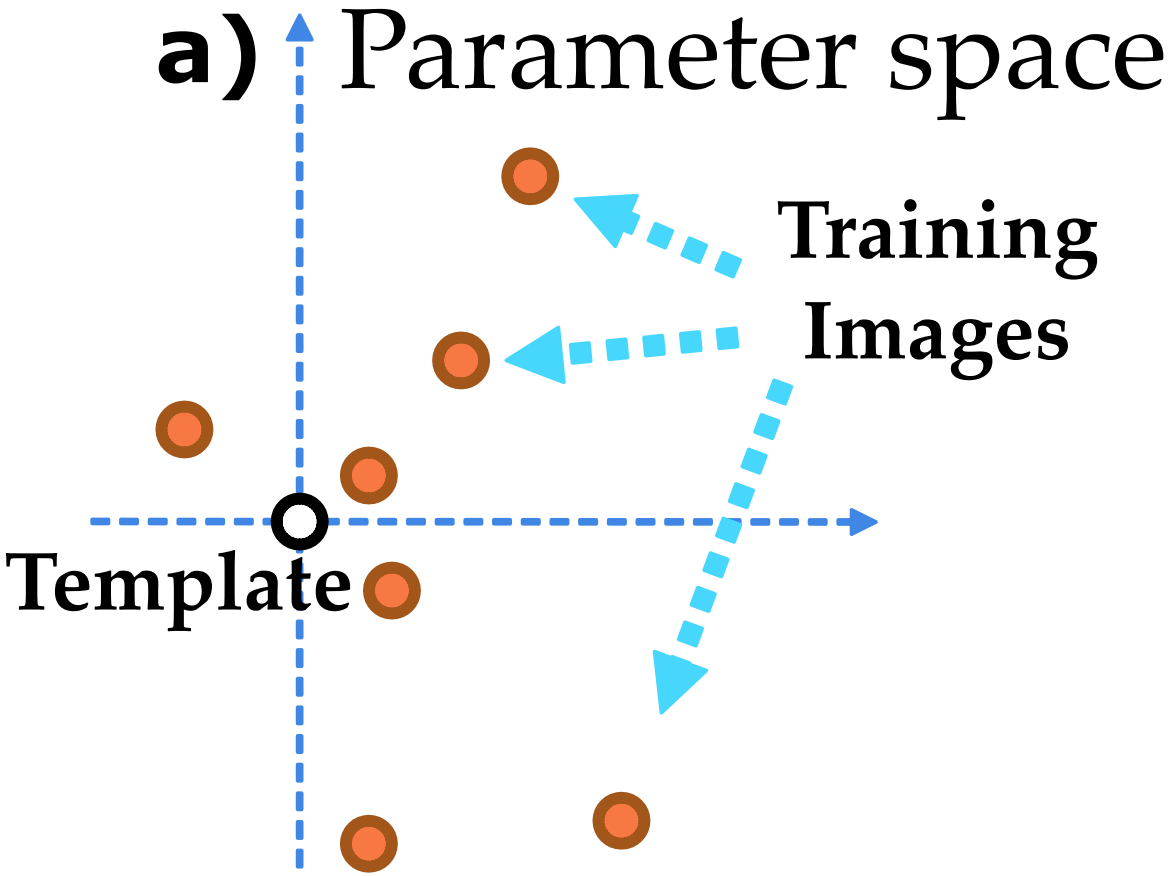
Can we do better?



Can we do better?

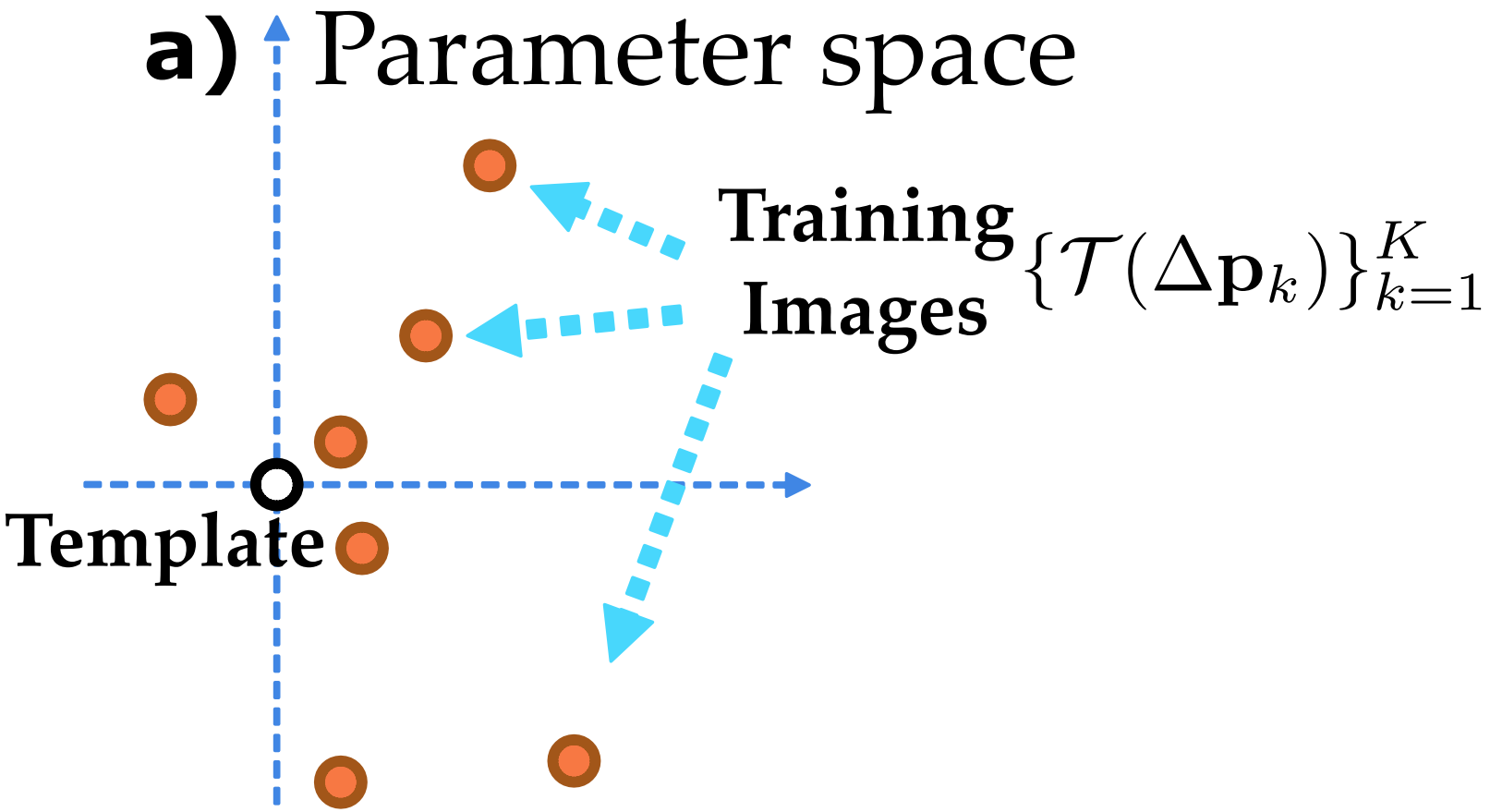


Data Driven Descent



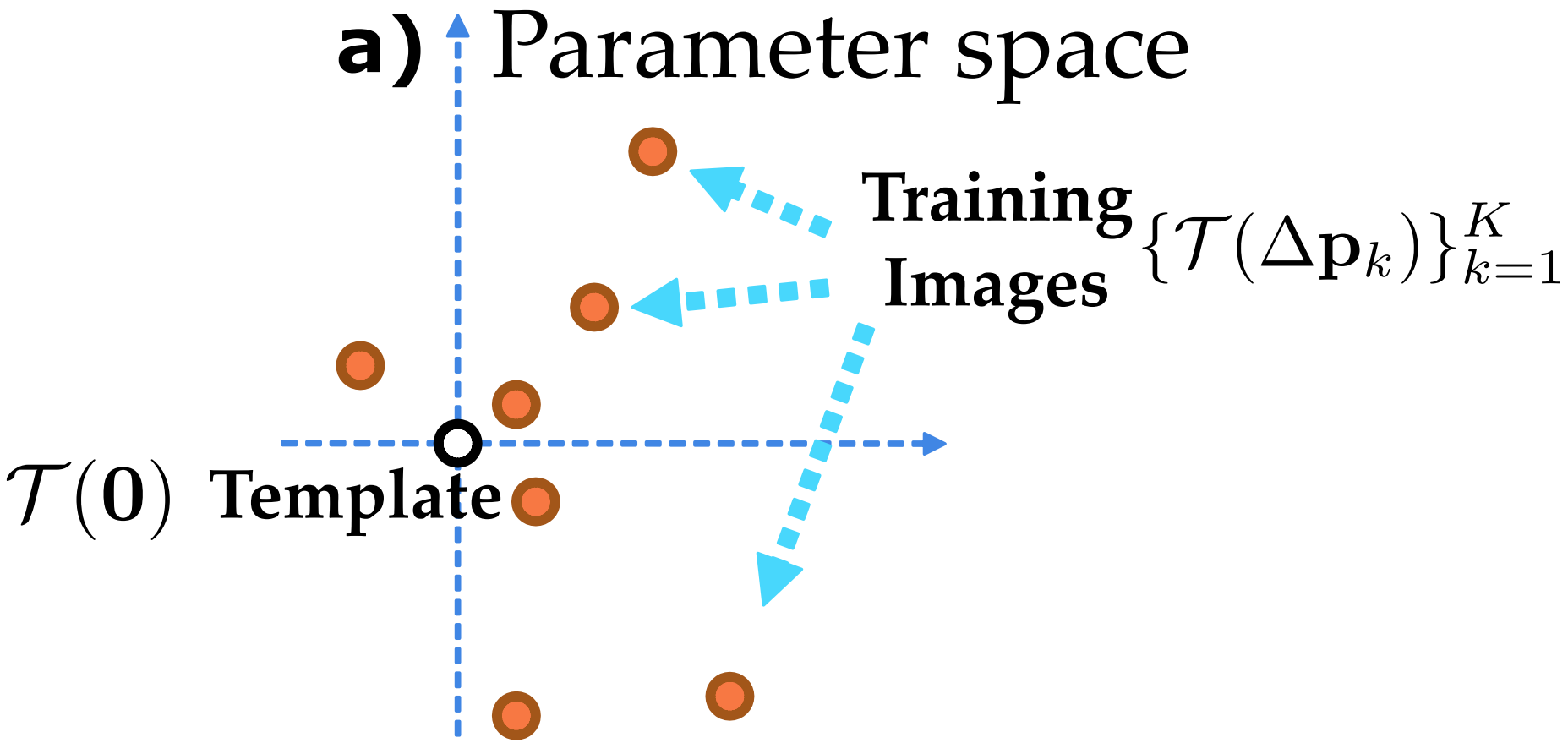
Tian & Narasimhan 2012

Data Driven Descent



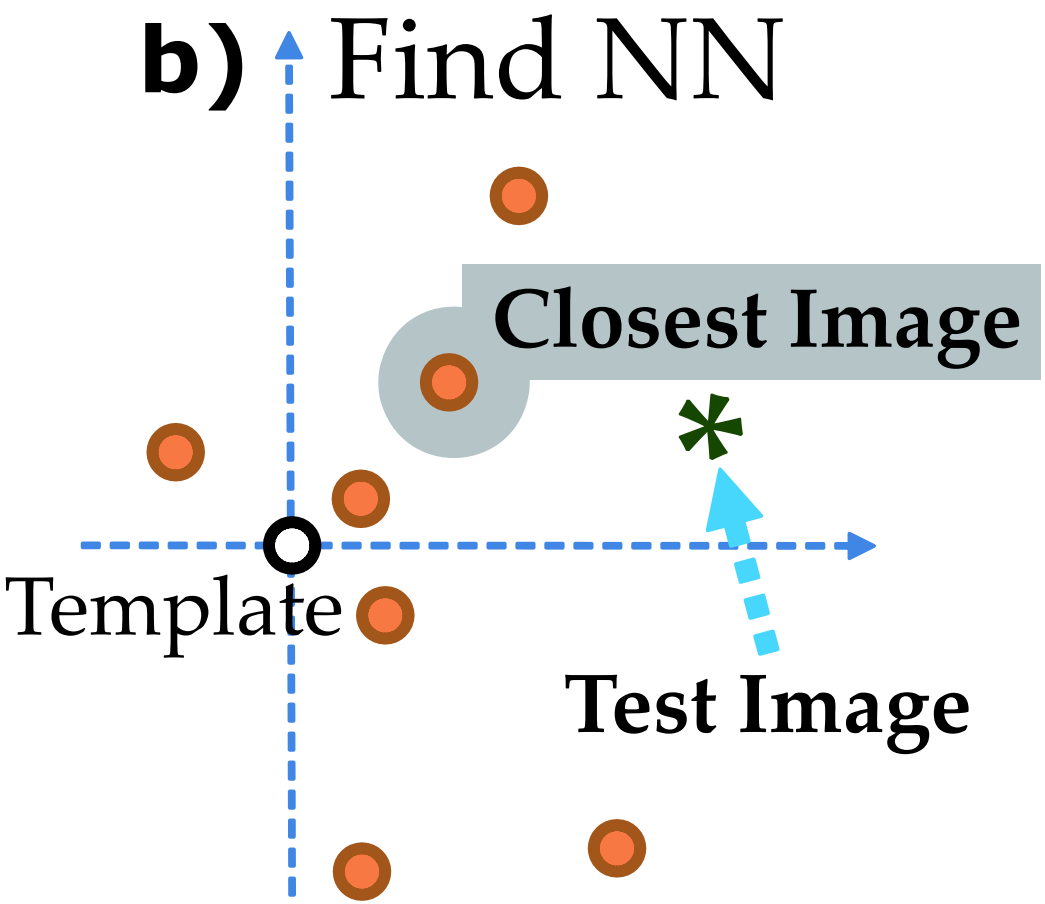
Tian & Narasimhan 2012

Data Driven Descent



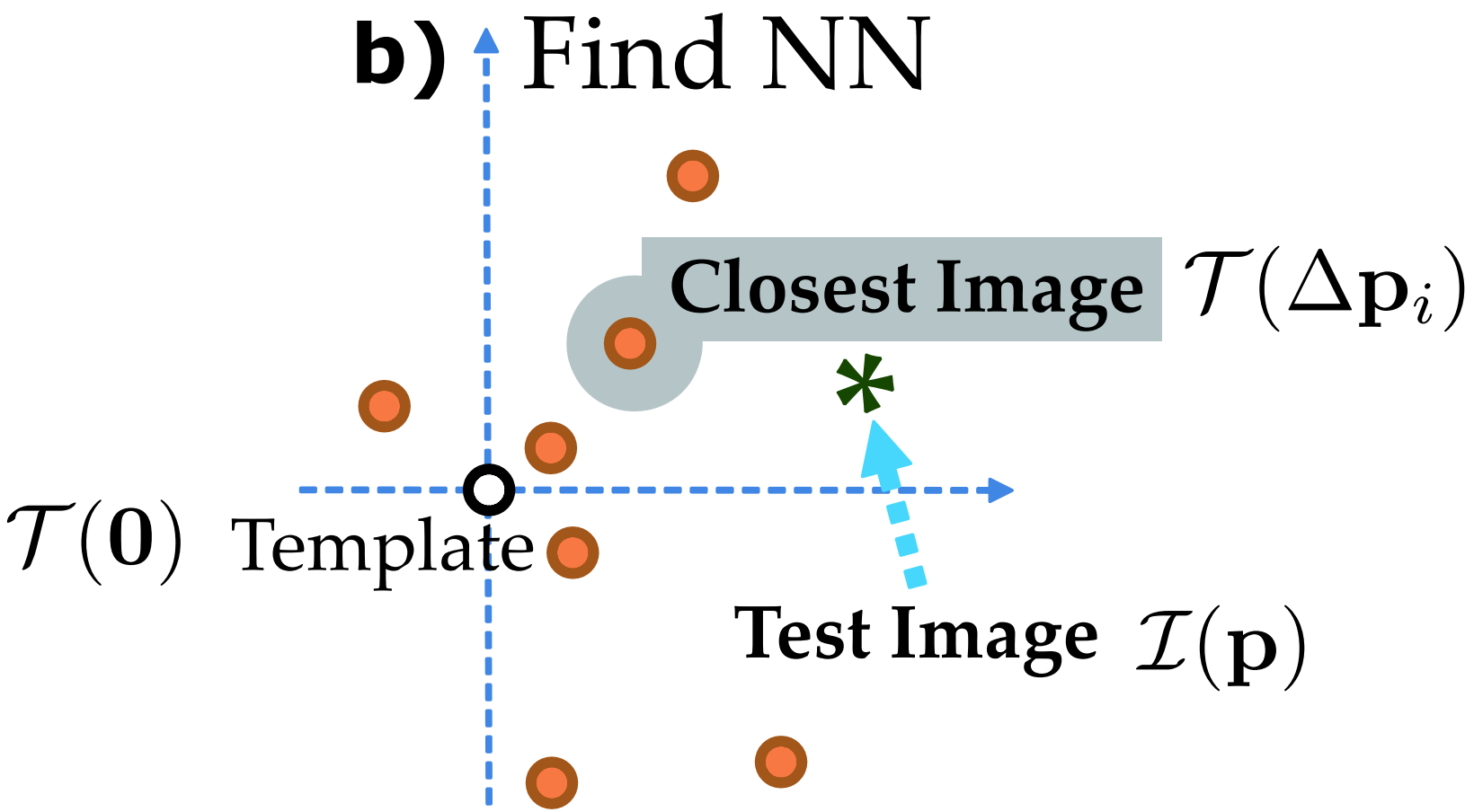
Tian & Narasimhan 2012

Data Driven Descent



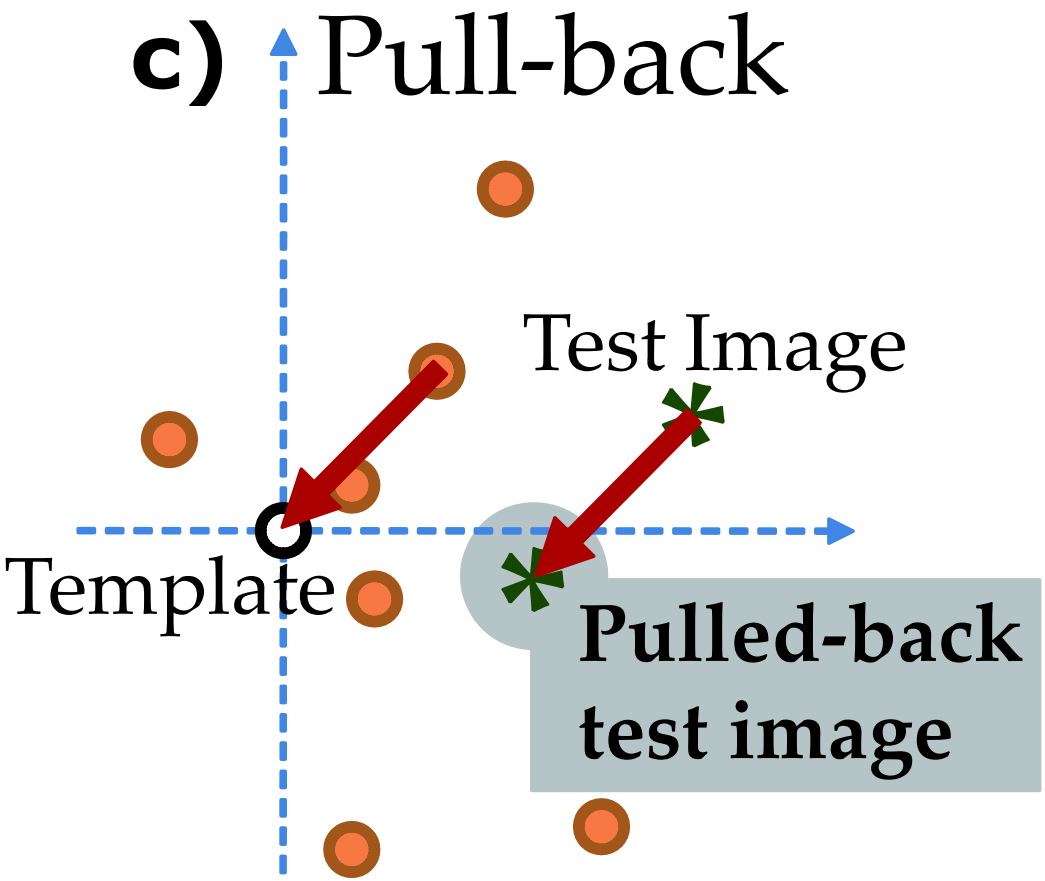
Tian & Narasimhan 2012

Data Driven Descent



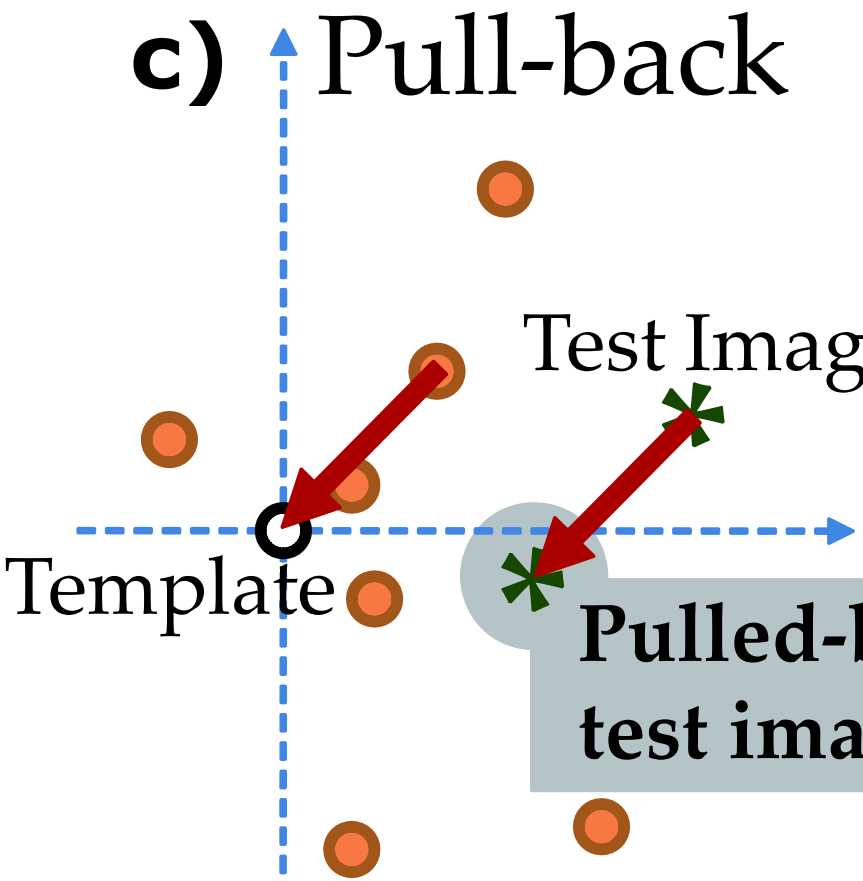
Tian & Narasimhan 2012

Data Driven Descent



Tian & Narasimhan 2012

Data Driven Descent

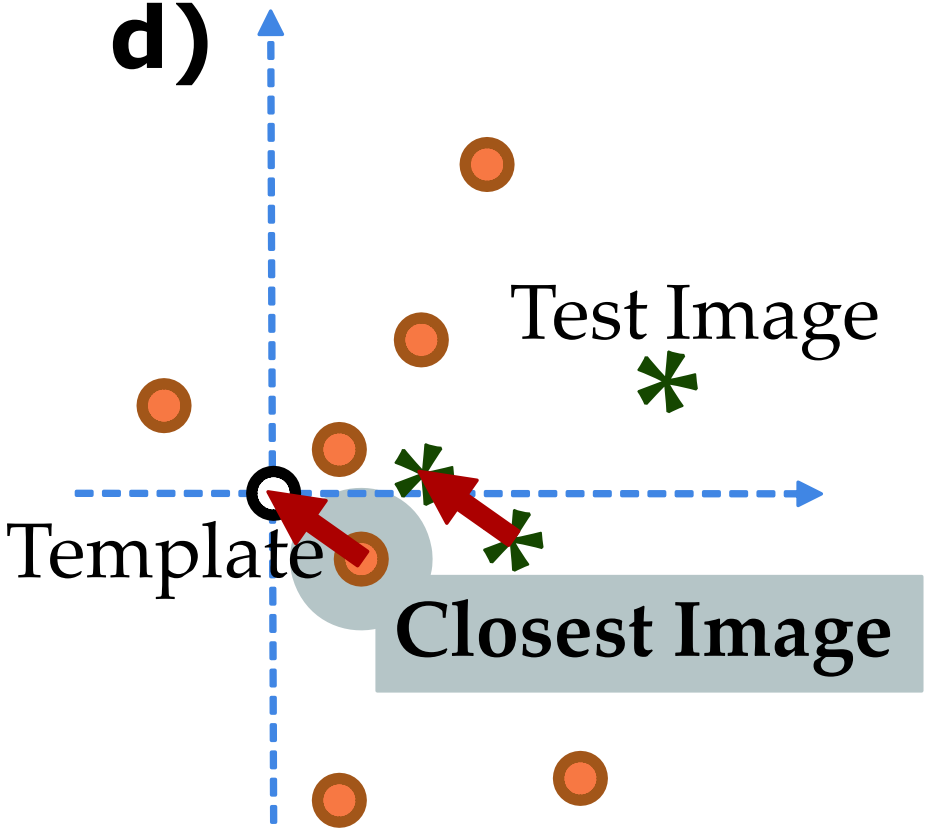


$$\mathbf{p} \rightarrow \mathbf{p} \circ^{-1} \Delta\mathbf{p}_i$$

"inverse composition"

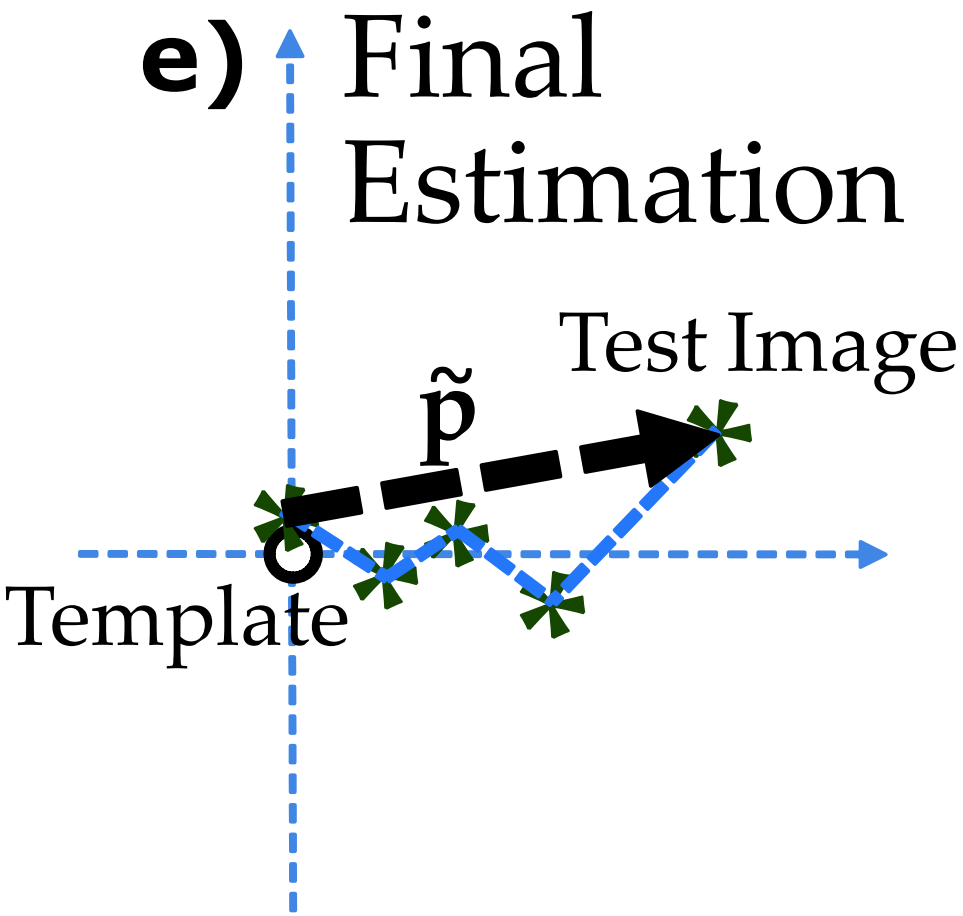
Tian & Narasimhan 2012

Data Driven Descent



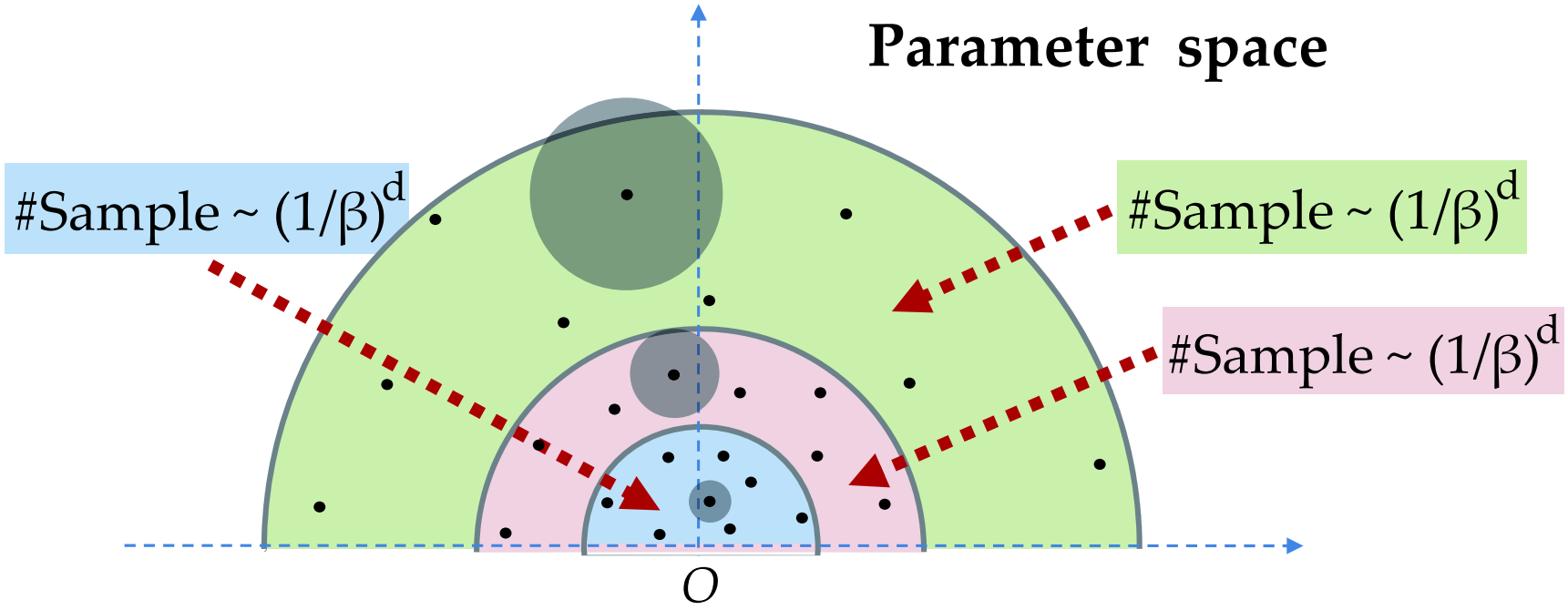
Tian & Narasimhan 2012

Data Driven Descent

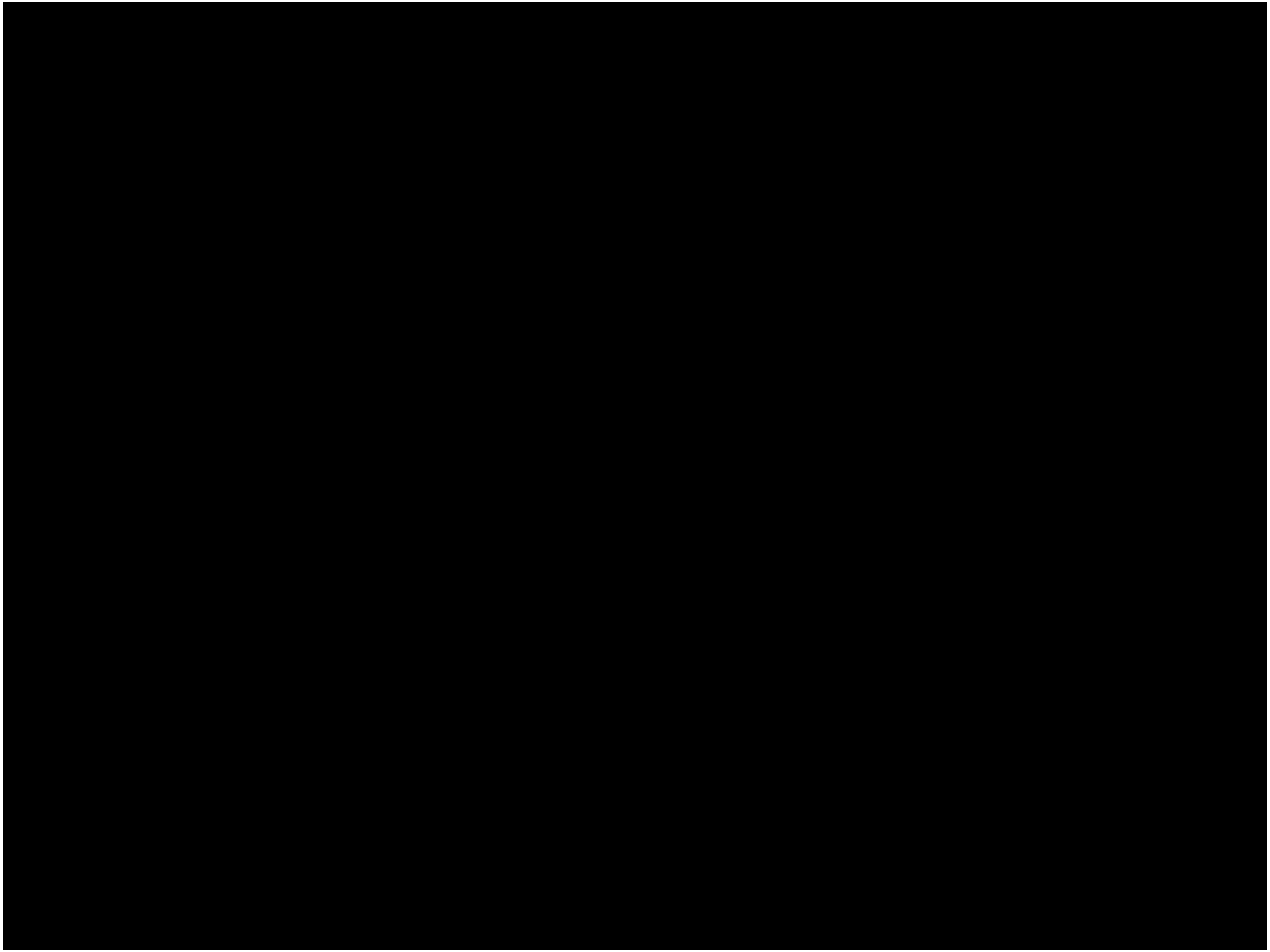


Tian & Narasimhan 2012

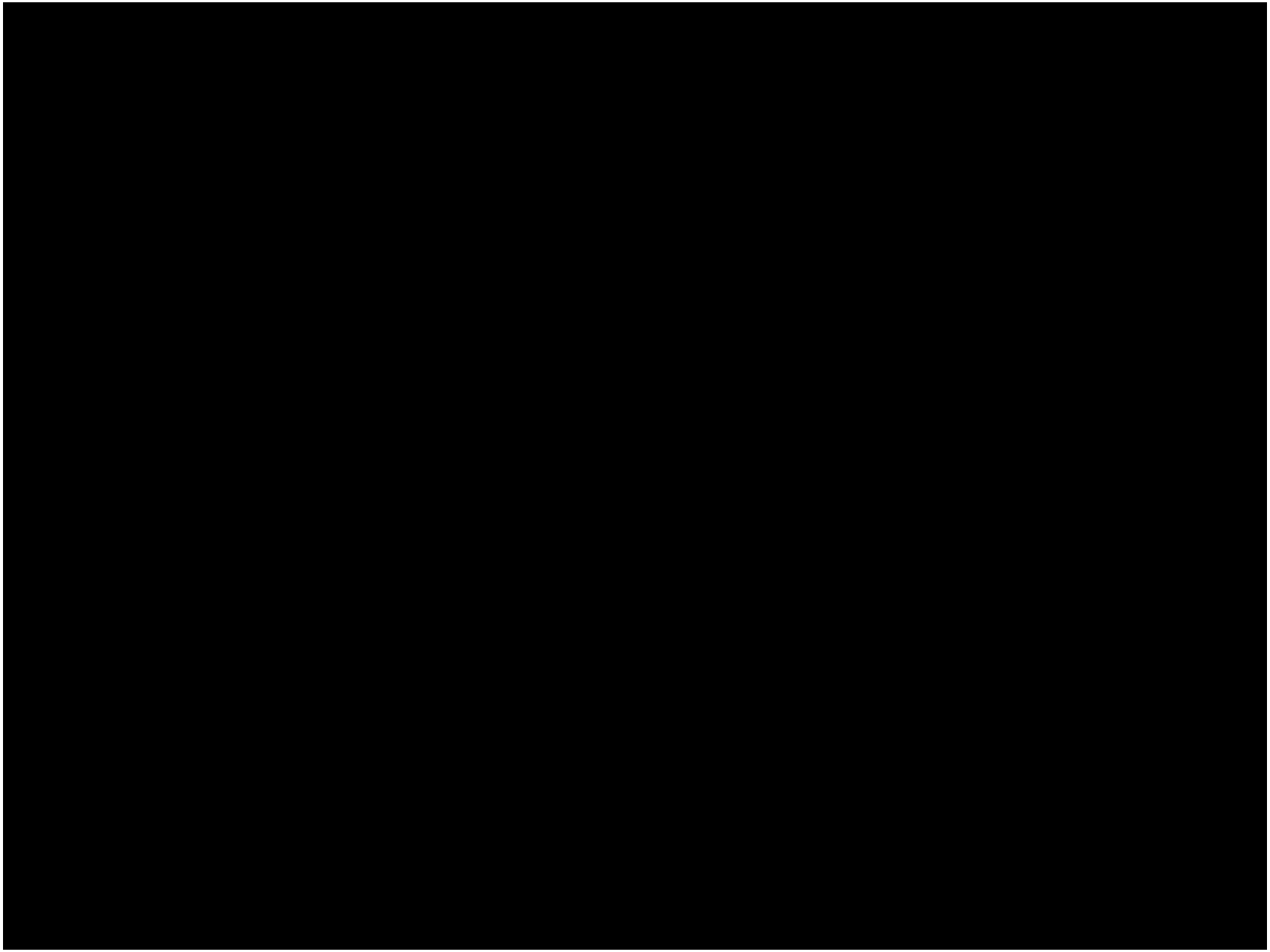
Data Driven Descent



Tian & Narasimhan 2012

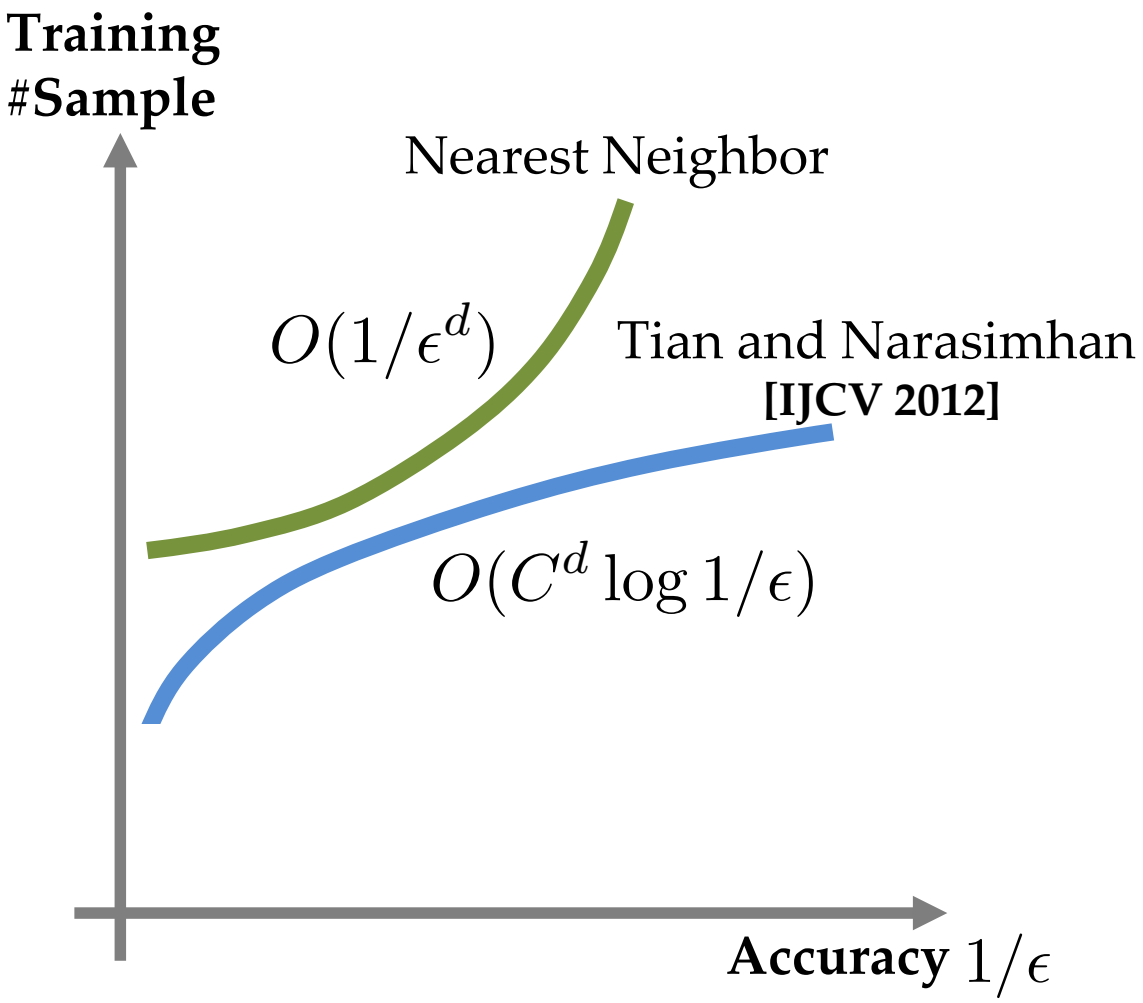


Tian & Narasimhan 2012

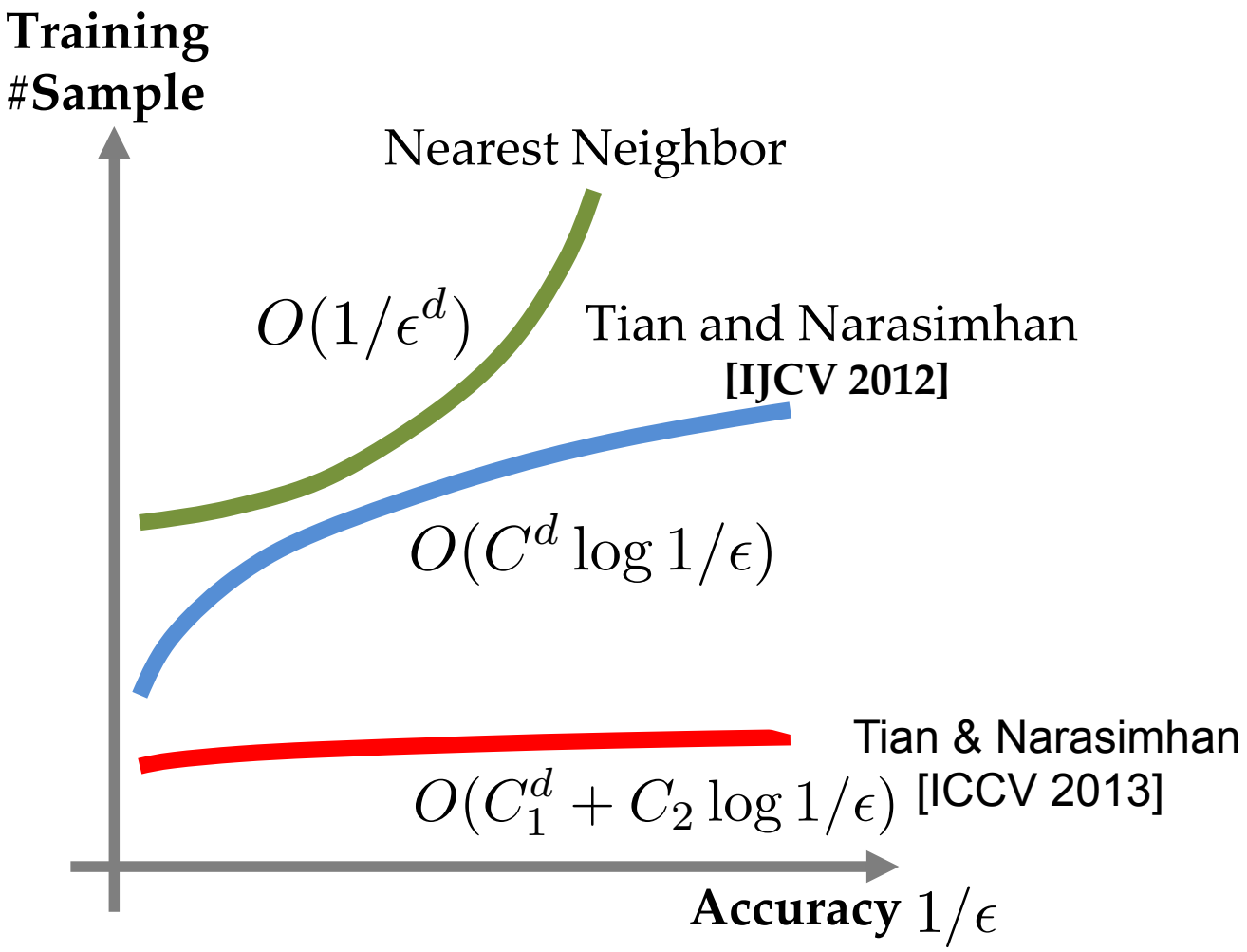


Tian & Narasimhan 2012

Can we do better?



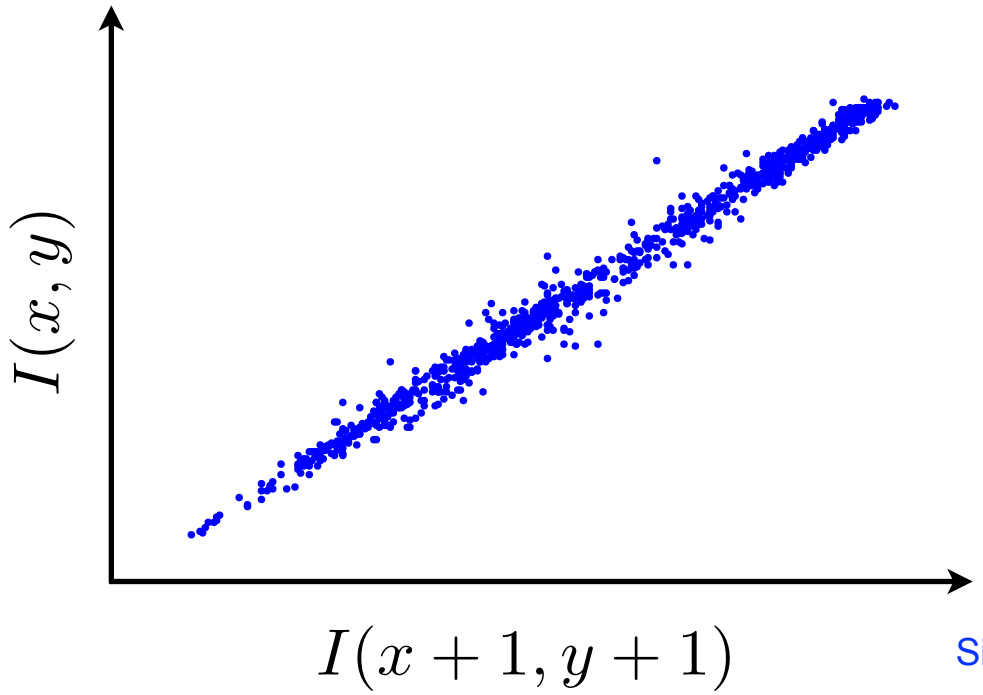
Can we do better?



Today

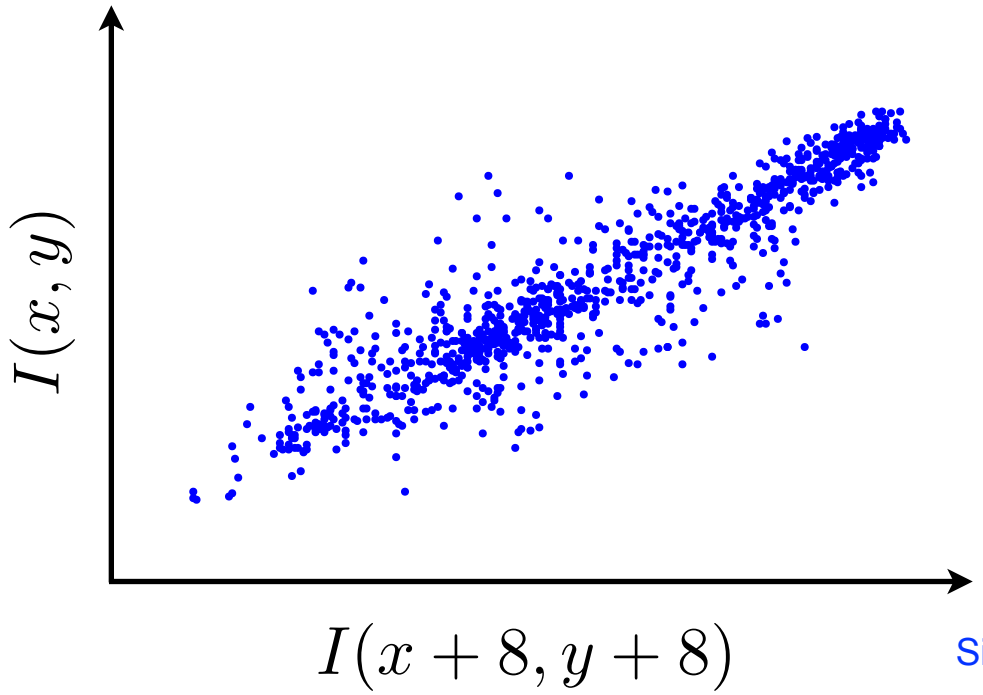
- Theoretical Limits on Search
- Inverse Composition & SDM
- Conditional LK

I



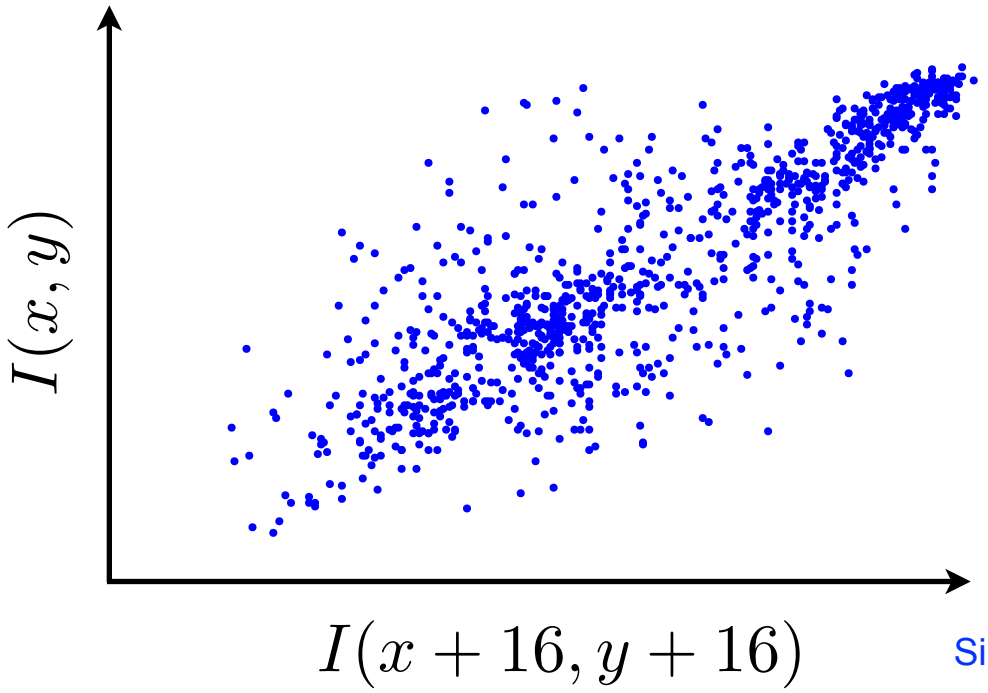
Simoncelli & Olshausen 2001

I



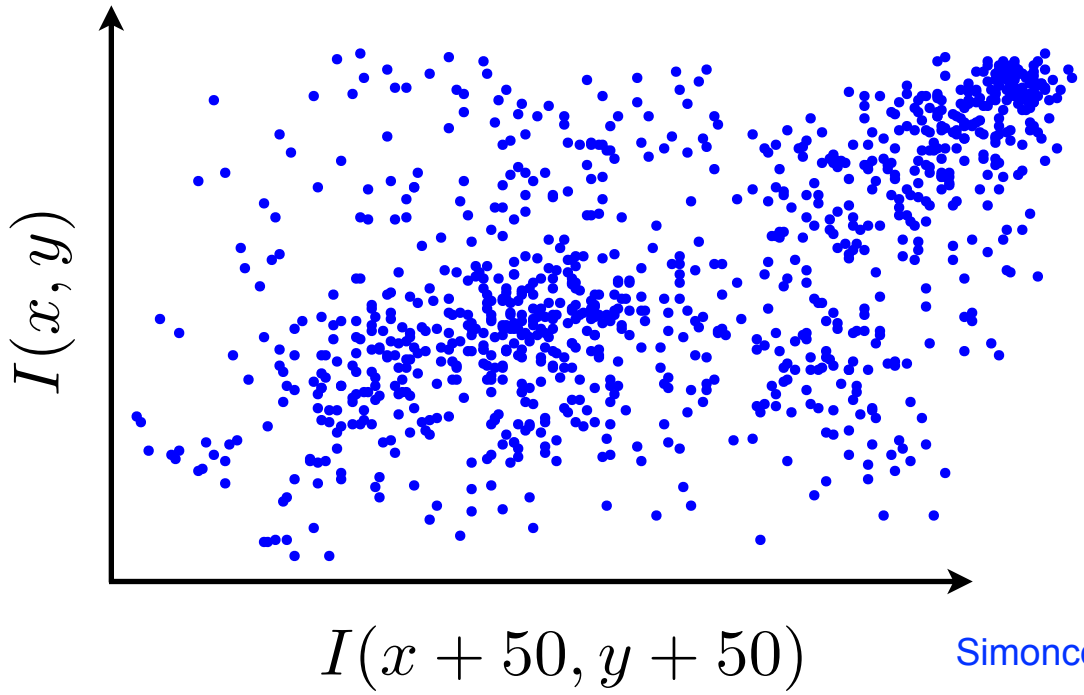
Simoncelli & Olshausen 2001

I



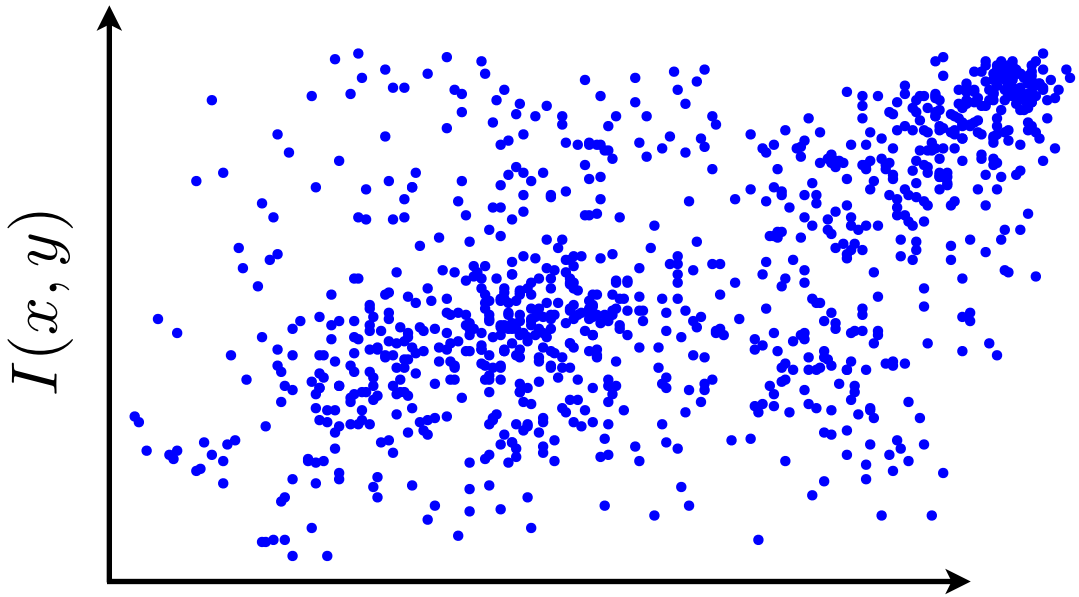
Simoncelli & Olshausen 2001

I



Simoncelli & Olshausen 2001

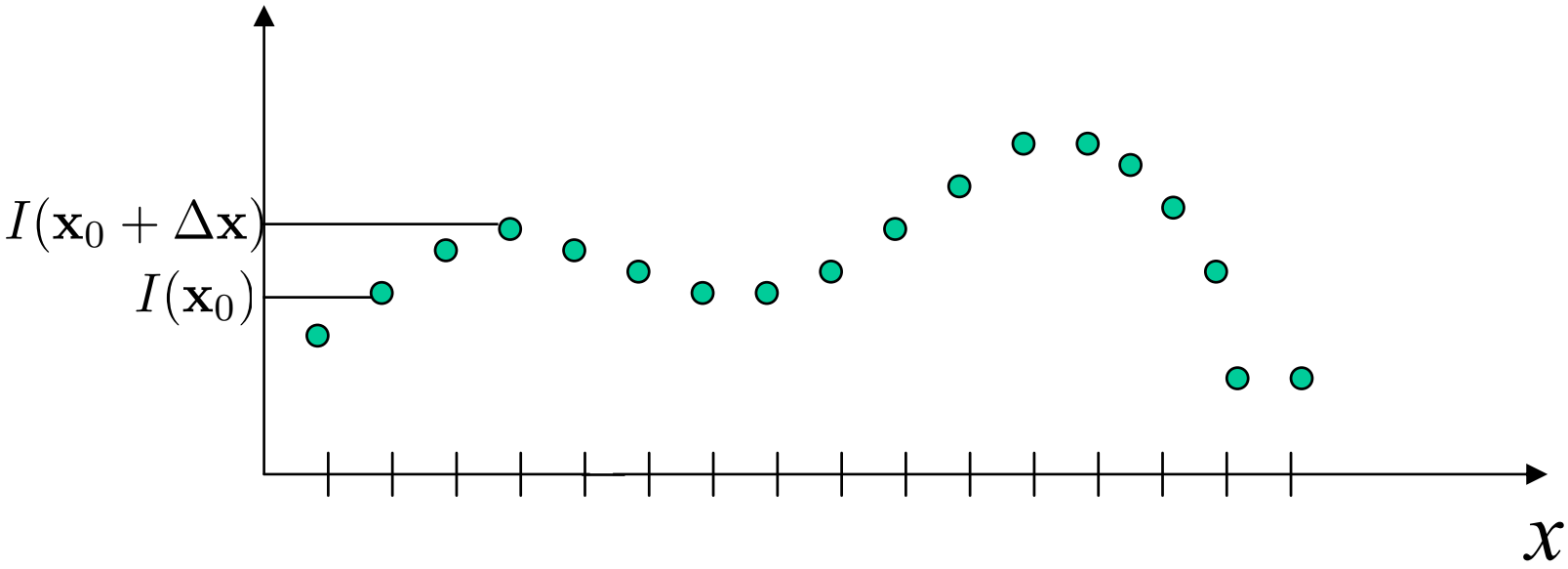
I



Simoncelli & Olshausen 2001

Reminder: Linearizing Registration

- What if I want to know $\Delta \mathbf{x}$ given that I have only the appearance at $I(\mathbf{x}_0)$and $I(\mathbf{x}_0 + \Delta \mathbf{x})$?

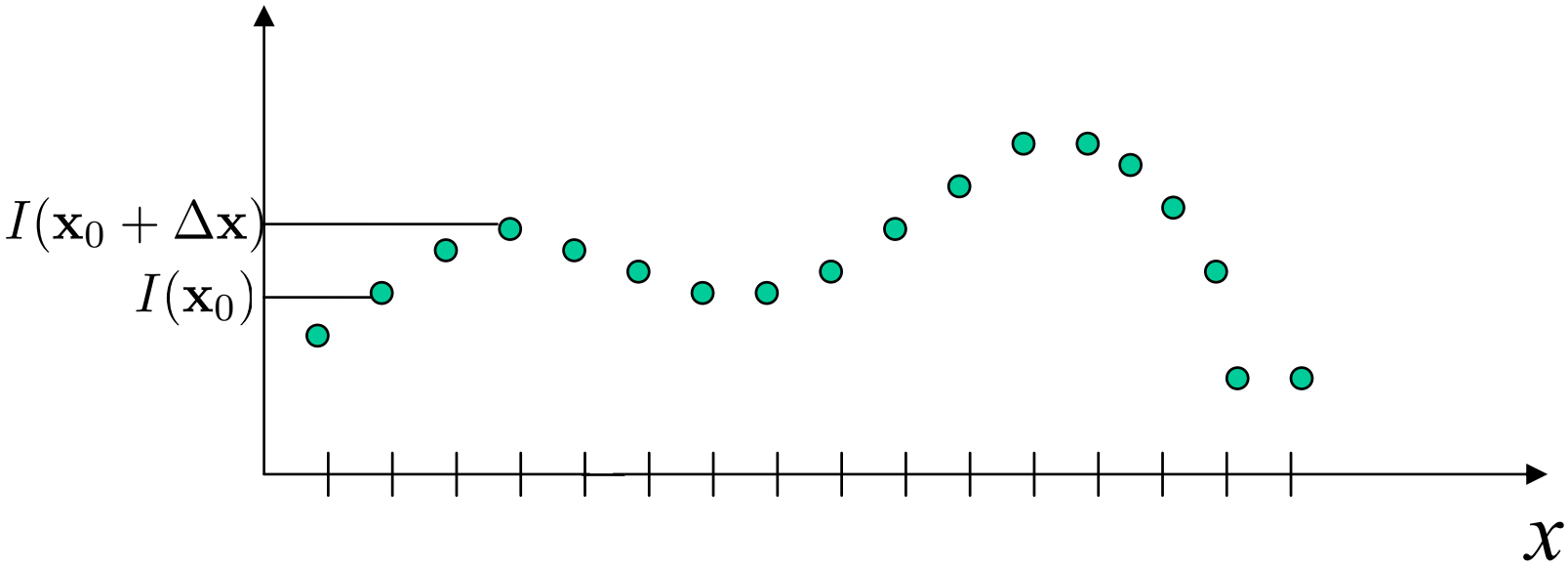


(Black)

Reminder: Linearizing Registration

- Again, simply take the Taylor series approximation?

$$\mathcal{I}(\mathbf{x}_0 + \Delta\mathbf{x}) \approx \mathcal{I}(\mathbf{x}_0) + \frac{\partial \mathcal{I}(\mathbf{x}_0)}{\partial \mathbf{x}^T}\Delta\mathbf{x}$$

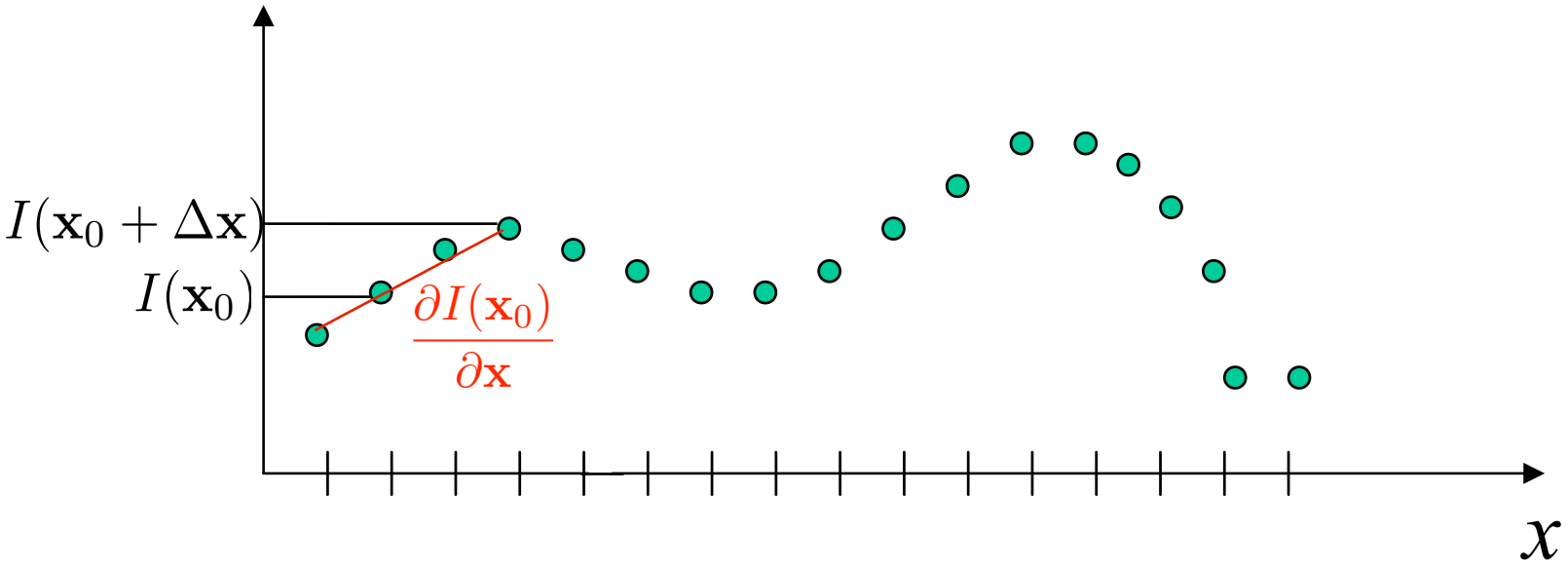


(Black)

Reminder: Linearizing Registration

- Again, simply take the Taylor series approximation?

$$\mathcal{I}(\mathbf{x}_0 + \Delta\mathbf{x}) \approx \mathcal{I}(\mathbf{x}_0) + \frac{\partial \mathcal{I}(\mathbf{x}_0)}{\partial \mathbf{x}^T}\Delta\mathbf{x}$$



(Black)

Reminder: LK Algorithm

- Lucas & Kanade (1981) realized this and proposed a method for estimating warp displacement using the principles of ***gradients*** and ***spatial coherence***.
- Technique applies Taylor series approximation to any spatially coherent area governed by the warp $\mathcal{W}(\mathbf{x};\mathbf{p})$.

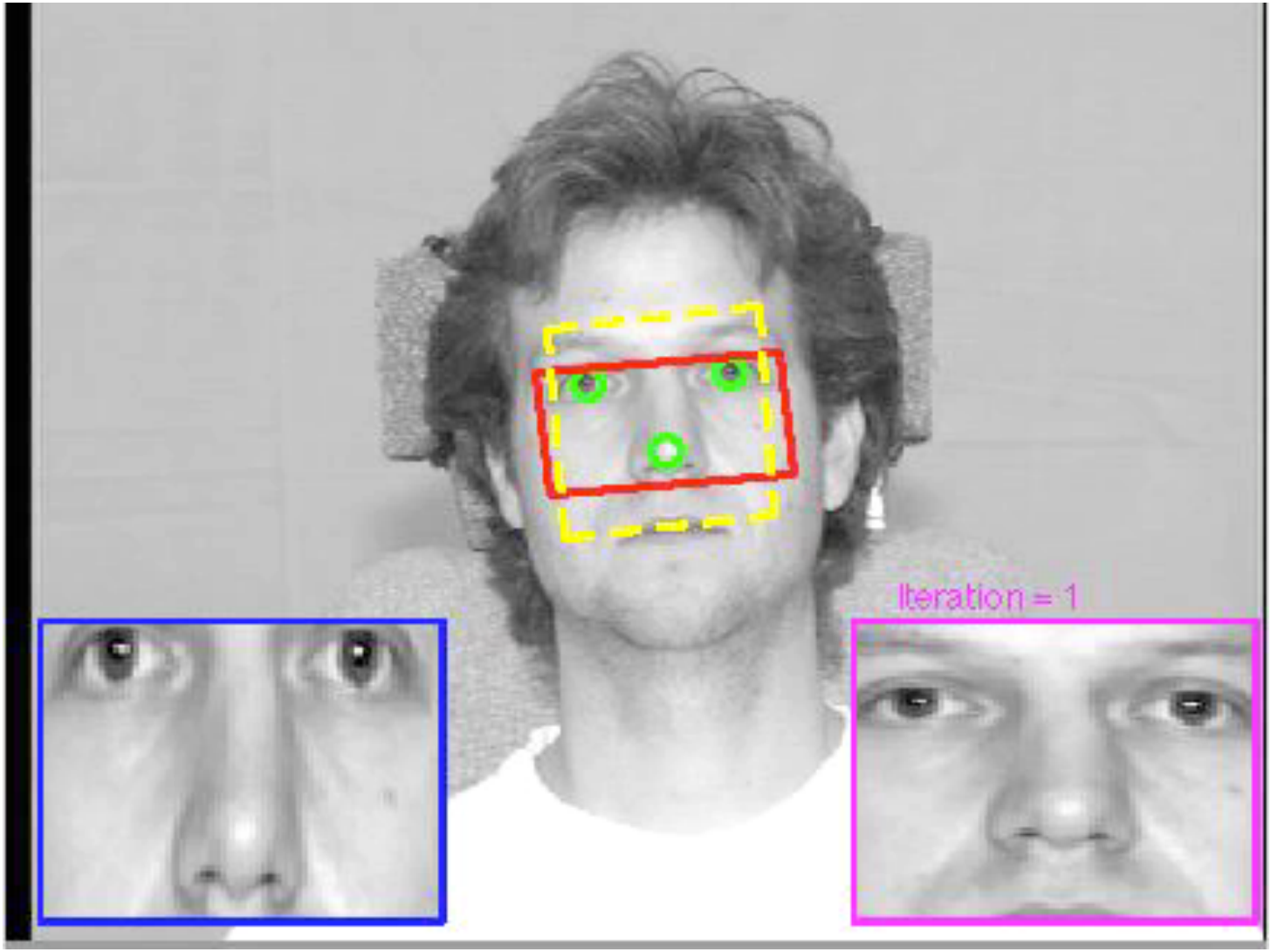
$$\mathcal{T}(\mathbf{0}) \approx \mathcal{I}(\mathbf{p}) + \frac{\partial \mathcal{I}(\mathbf{p})}{\partial \mathbf{p}^T}\Delta\mathbf{p}$$

$N \times 1$ $N \times 1$ $N \times P$

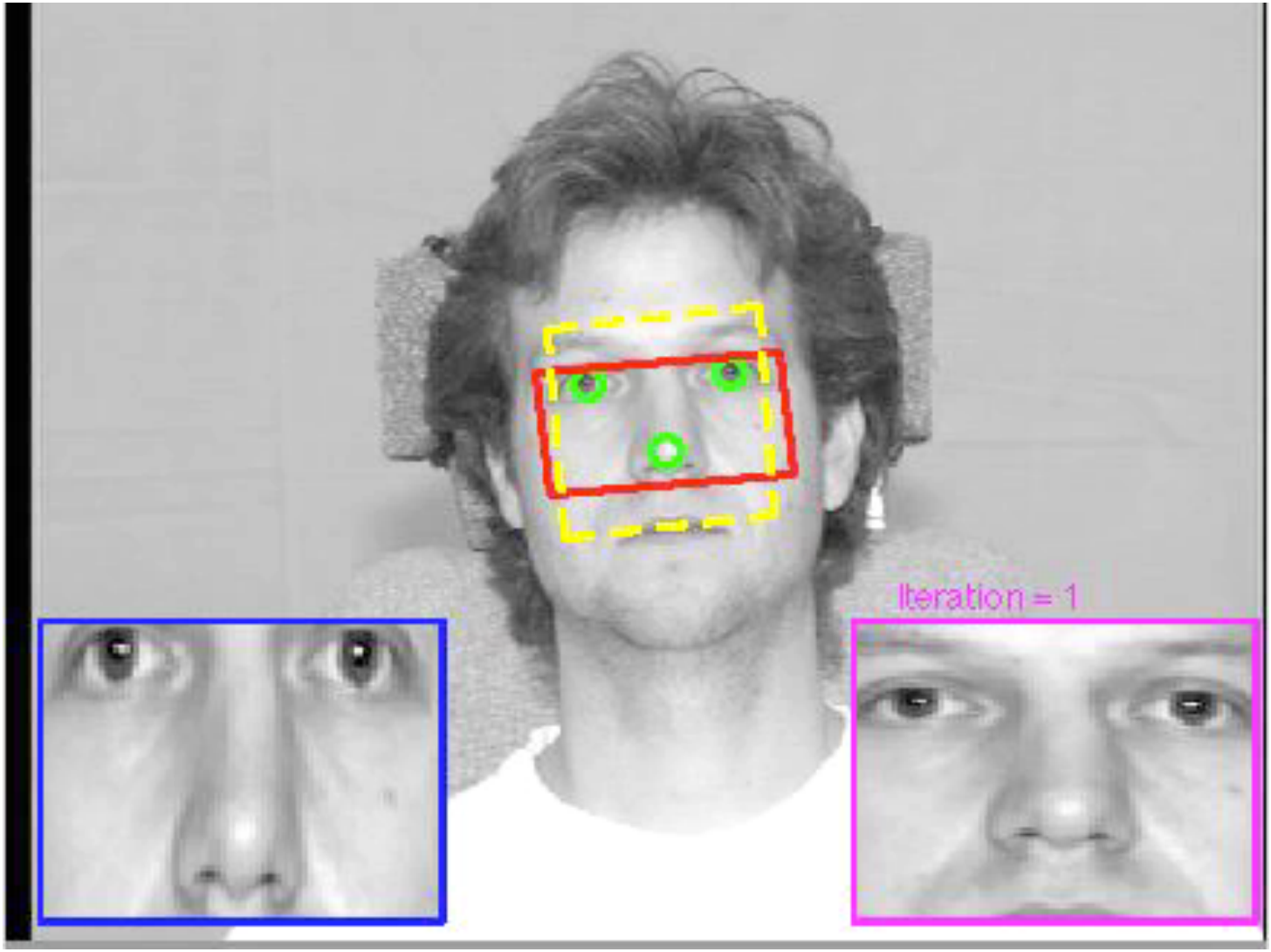
$N =$ number of pixels

$P =$ number of warp parameters

Reminder: Examples of LK Alignment



Reminder: Examples of LK Alignment



Reminder: Linearizing the Template?

$$\mathcal{T}(\mathbf{0}) \approx \mathcal{I}(\mathbf{p}) + \frac{\partial \mathcal{I}(\mathbf{p})}{\partial \mathbf{p}^T} \Delta \mathbf{p}$$

"*template*"

Reminder: Linearizing the Template?

$$\mathcal{T}(\mathbf{0}) \approx \mathcal{I}(\mathbf{p}) + \frac{\partial \mathcal{I}(\mathbf{p})}{\partial \mathbf{p}^T} \Delta \mathbf{p}$$

"*template*"

$$\mathcal{T}(\mathbf{0}) + \frac{\partial \mathcal{T}(\mathbf{0})}{\partial \mathbf{p}^T} \Delta \mathbf{p} \approx \mathcal{I}(\mathbf{p})$$

"Why is this useful if the template must be static?"

Inverse Composition Algorithm

- Actual algorithm is just the application of the following steps,

Step 1:

$$\arg\min_{\Delta\mathbf{p}} ||\mathcal{T}(\mathbf{0}) + \frac{\partial \mathcal{T}(\mathbf{0})}{\partial \mathbf{p}^T}\Delta\mathbf{p} - \mathcal{I}(\mathbf{p})||_2^2$$

Step 2:

$$\mathbf{p} \rightarrow \mathbf{p} \circ^{-1} \Delta\mathbf{p}$$

keep applying steps until $\Delta\mathbf{p}$ converges.

Inverse Composition Algorithm

- Actual algorithm is just the application of the following steps,

Step 1:

$$\Delta \mathbf{p} = \left[\frac{\partial \mathcal{T}(\mathbf{0})}{\partial \mathbf{p}^T} \right]^{\dagger} [\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$$

Step 2:

$$\mathbf{p} \rightarrow \mathbf{p} \circ^{-1} \Delta \mathbf{p}$$

keep applying steps until $\Delta \mathbf{p}$ converges.

$$\mathbf{A}^{\dagger} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T$$

Inverse Composition Algorithm

- Actual algorithm is just the application of the following steps,

Step 1:

$$\Delta \mathbf{p} = \left[\frac{\partial \mathcal{T}(\mathbf{0})}{\partial \mathbf{p}^T}\right]^{\dagger} [\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$$

"Static"

Step 2:

$$\mathbf{p} \rightarrow \mathbf{p} \circ^{-1} \Delta \mathbf{p}$$ *"Inverse Composition"*

keep applying steps until $\Delta \mathbf{p}$ converges.

$$\mathbf{A}^{\dagger} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T$$ *"Pseudo-Inverse"*

Inverse Composition Algorithm

- Actual algorithm is just the application of the following steps,

Step 1:

$$\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$$

Step 2: *"Static"*

$$\mathbf{p} \rightarrow \mathbf{p} \circ^{-1} \Delta \mathbf{p}$$

keep applying steps until $\Delta \mathbf{p}$ converges.

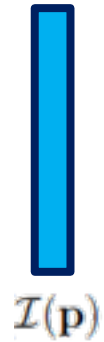
$$\mathbf{R} = \left[\frac{\partial \mathcal{T}(\mathbf{0})}{\partial \mathbf{p}^T} \right]^{\dagger}$$

LK- IC Algorithm

- Both assume a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$

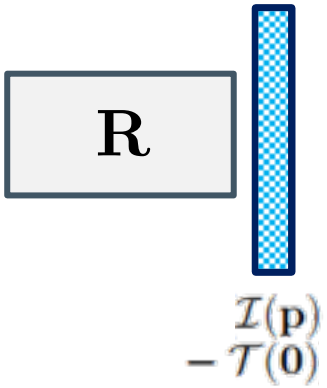
LK- IC Algorithm

- Both assume a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence



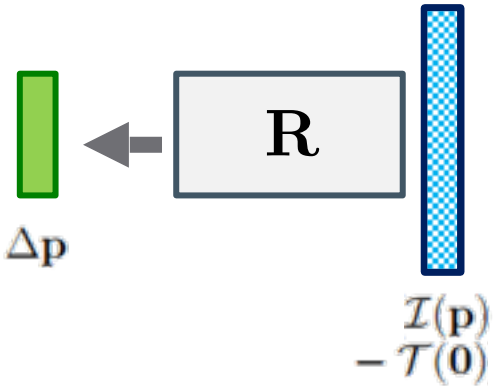
LK- IC Algorithm

- Both assume a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence



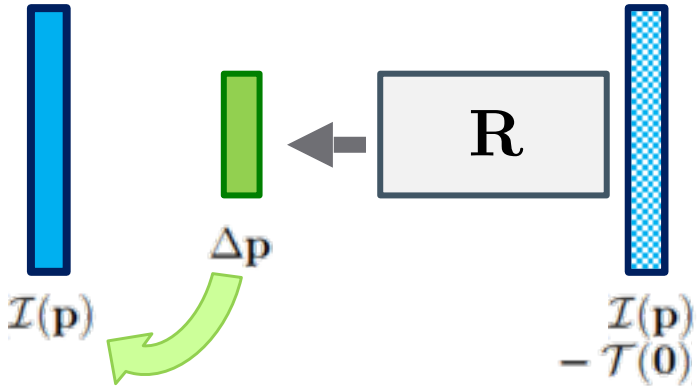
LK- IC Algorithm

- Both assume a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence



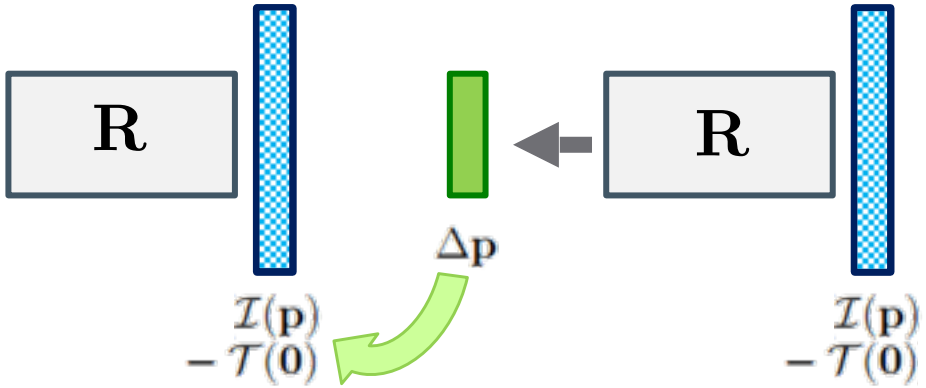
LK- IC Algorithm

- Both assume a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence



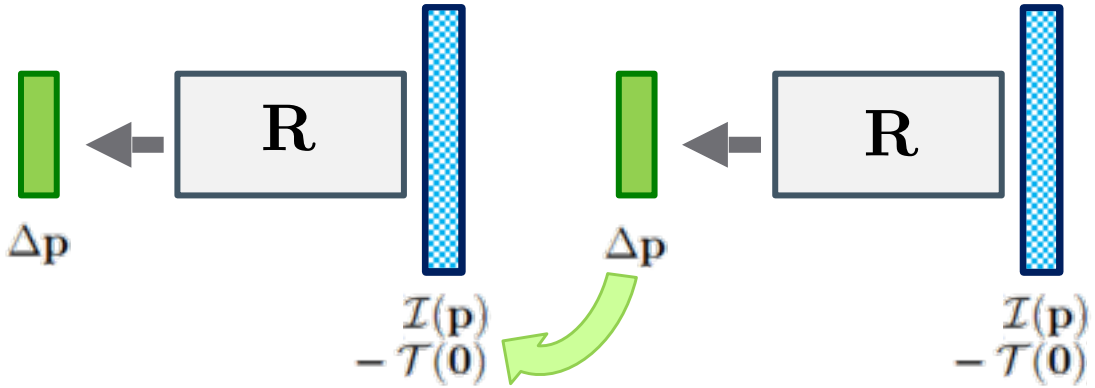
LK- IC Algorithm

- Both assume a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence



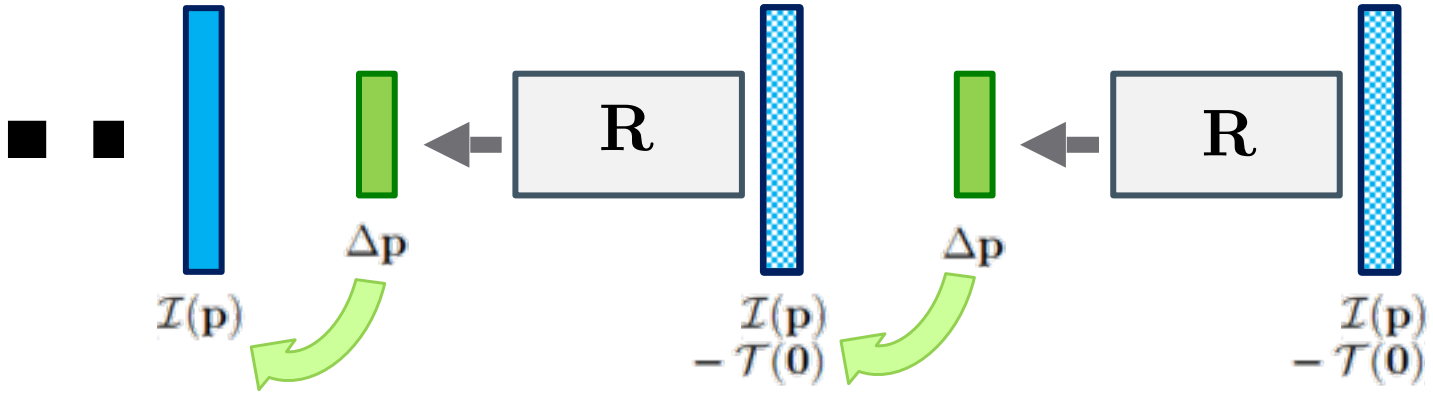
LK- IC Algorithm

- Both assume a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence



LK- IC Algorithm

- Both assume a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence



  - Single linear model $\mathbf{R}$ applied at all iterations.

Supervised Descent Method (SDM)

- Xiong & De La Torre proposed the Supervised Descent Method (SDM) to explicitly learn $\mathbf{R}$.

Supervised Descent Method (SDM)

- Xiong & De La Torre proposed the Supervised Descent Method (SDM) to explicitly learn $\mathbf{R}$.
- $\mathbf{R}$ can be learned from data

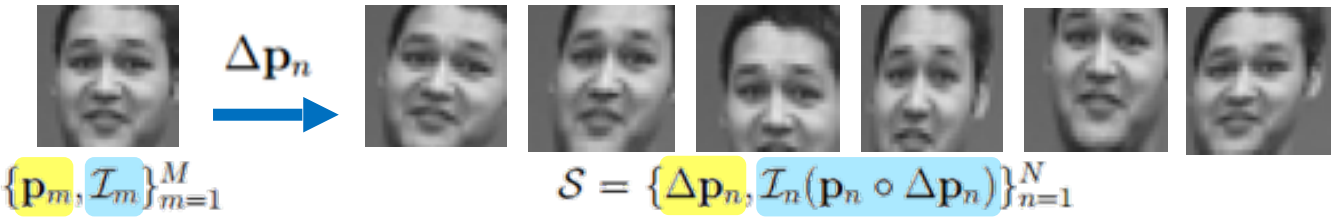
Supervised Descent Method (SDM)

- Xiong & De La Torre proposed the Supervised Descent Method (SDM) to explicitly learn $\mathbf{R}$.
- $\mathbf{R}$ can be learned from data $\mathcal{S} = \{\Delta\mathbf{p}_n, \mathcal{I}_n(\mathbf{p}_n \circ \Delta\mathbf{p}_n)\}_{n=1}^{N}$
  - One can generate a synthetic training set from given data $\{\mathbf{p}_m, \mathcal{I}_m\}_{m=1}^{M}$ by sampling $\Delta\mathbf{p}_n$

$\{\mathbf{p}_m, \mathcal{I}_m\}_{m=1}^{M}$

Supervised Descent Method (SDM)

- Xiong & De La Torre proposed the Supervised Descent Method (SDM) to explicitly learn $\mathbf{R}$.
- $\mathbf{R}$ can be learned from data $\mathcal{S} = \{\Delta\mathbf{p}_n, \mathcal{I}_n(\mathbf{p}_n \circ \Delta\mathbf{p}_n)\}_{n=1}^{N}$
  - One can generate a synthetic training set from given data $\{\mathbf{p}_m, \mathcal{I}_m\}_{m=1}^{M}$ by sampling $\Delta\mathbf{p}_n$



Supervised Descent Method (SDM)

- Xiong & De La Torre proposed the Supervised Descent Method (SDM) to explicitly learn $\mathbf{R}$.
- $\mathbf{R}$ can be learned from data $\mathcal{S} = \{\Delta\mathbf{p}_n, \mathcal{I}_n(\mathbf{p}_n \circ \Delta\mathbf{p}_n)\}_{n=1}^{N}$
  - One can generate a synthetic training set from given data $\{\mathbf{p}_m, \mathcal{I}_m\}_{m=1}^{M}$ by sampling $\Delta\mathbf{p}_n$

  - Training objective: $\min_{\mathbf{R}} \sum_{n \in \mathcal{S}} \|\Delta\mathbf{p}_n - \mathbf{R}[\mathcal{I}_n(\mathbf{p}_n \circ \Delta\mathbf{p}_n) - \mathcal{T}(\mathbf{0})]\|_2^2$
    - Learns to directly predict the geometric displacement conditioned on appearance.

Supervised Descent Method (SDM)

- Like IC-LK, SDM assumes a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$

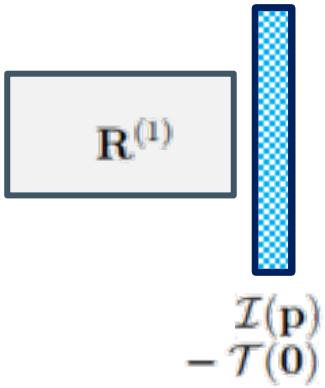
Supervised Descent Method (SDM)

- Like IC-LK, SDM assumes a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence

$\mathcal{I}(\mathbf{p})$

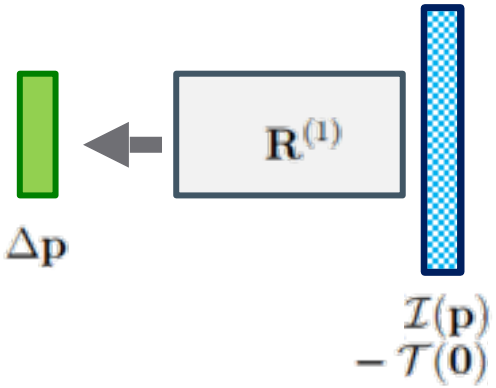
Supervised Descent Method (SDM)

- Like IC-LK, SDM assumes a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence



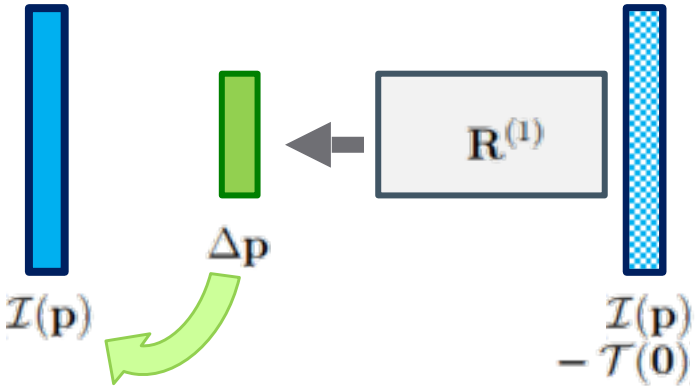
Supervised Descent Method (SDM)

- Like IC-LK, SDM assumes a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence



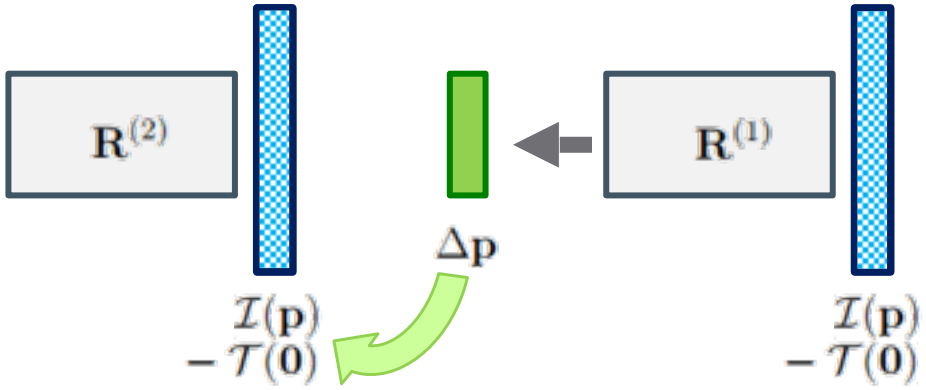
Supervised Descent Method (SDM)

- Like IC-LK, SDM assumes a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence



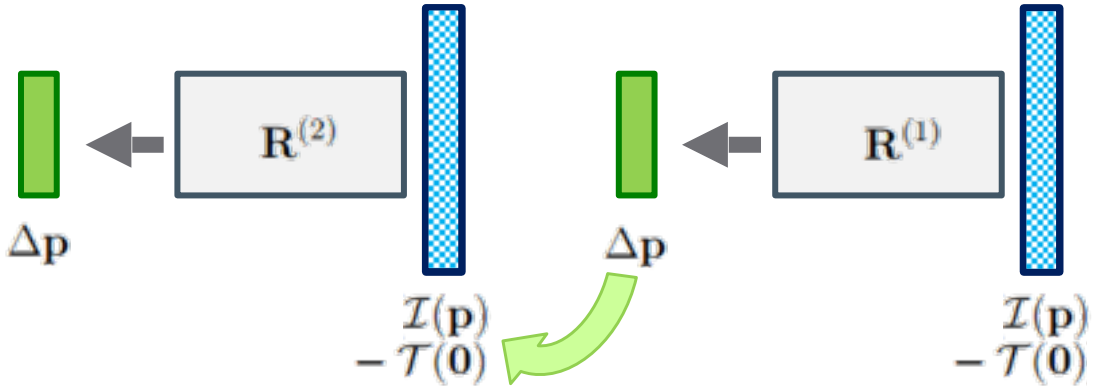
Supervised Descent Method (SDM)

- Like IC-LK, SDM assumes a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence



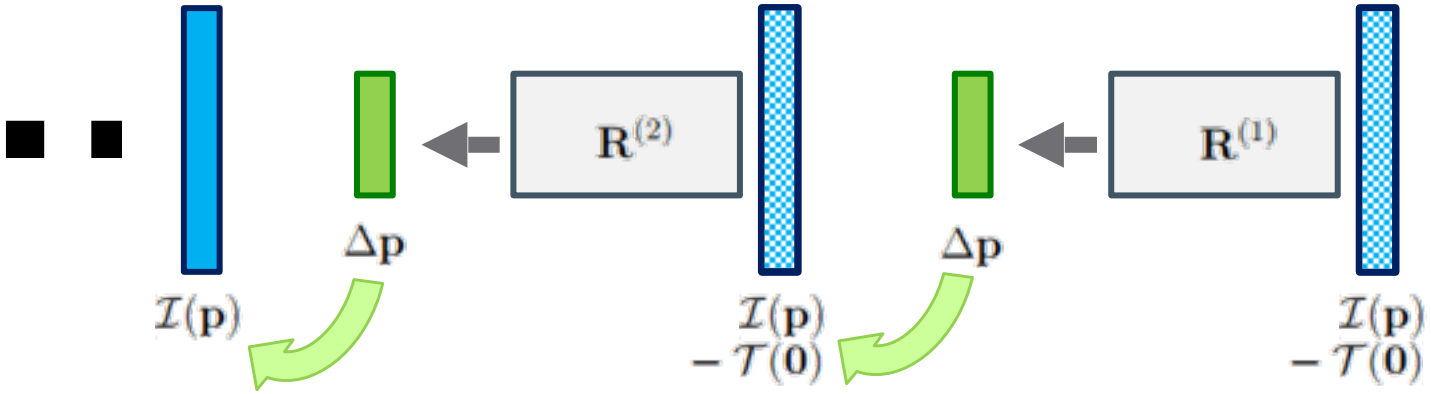
Supervised Descent Method (SDM)

- Like IC-LK, SDM assumes a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence



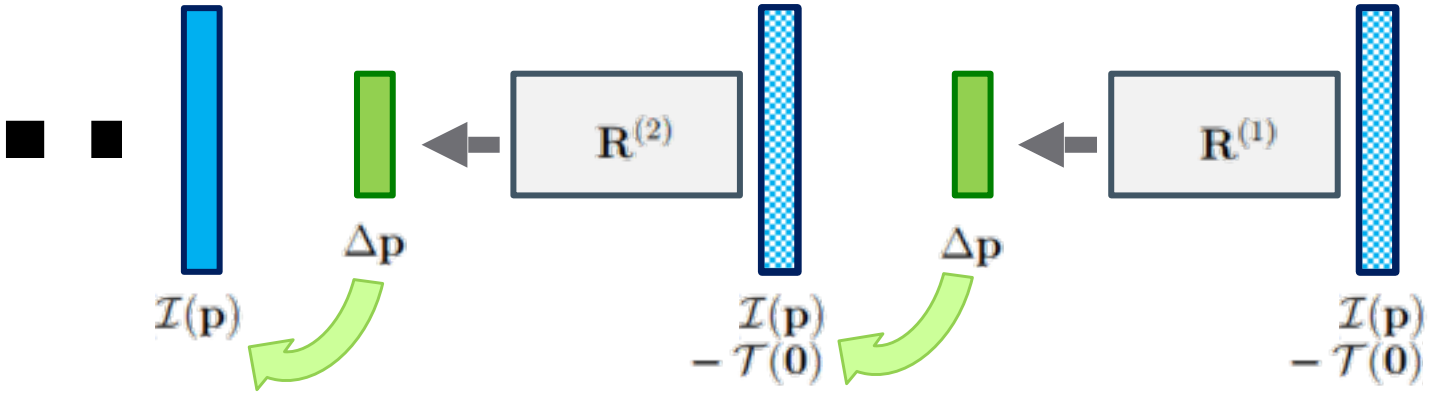
Supervised Descent Method (SDM)

- Like IC-LK, SDM assumes a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence

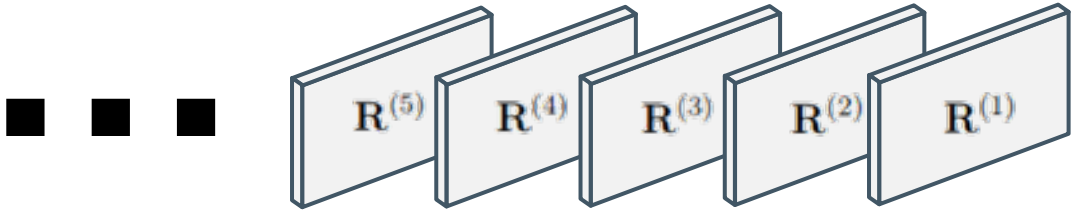


Supervised Descent Method (SDM)

- Like IC-LK, SDM assumes a linear relationship between appearance and geometry: $\Delta \mathbf{p} = \mathbf{R}[\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})]$
  - Iteratively updates until convergence

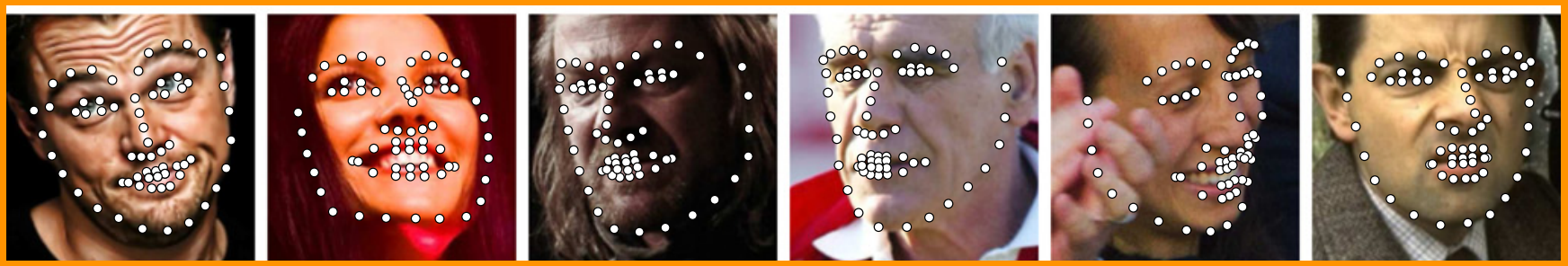


- However, $\mathbf{R}^{(k)}$ is iteration specific.



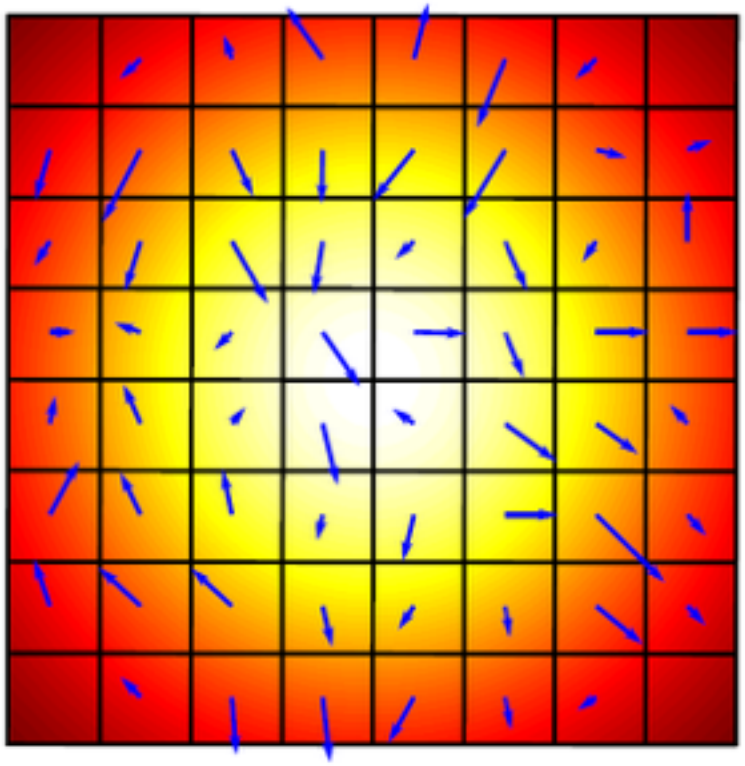
SDM - Object Alignment

- SDM is not only applicable to image registration, but can be applied to the problem of general object alignment (e.g. facial landmark alignment).

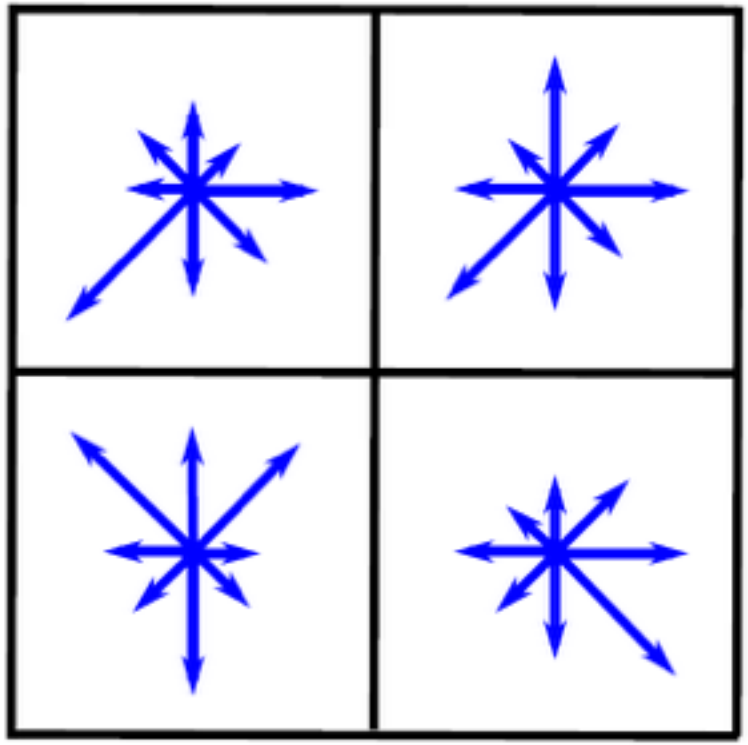


$$\mathcal{T}(\mathbf{0})$$

Robust Representations



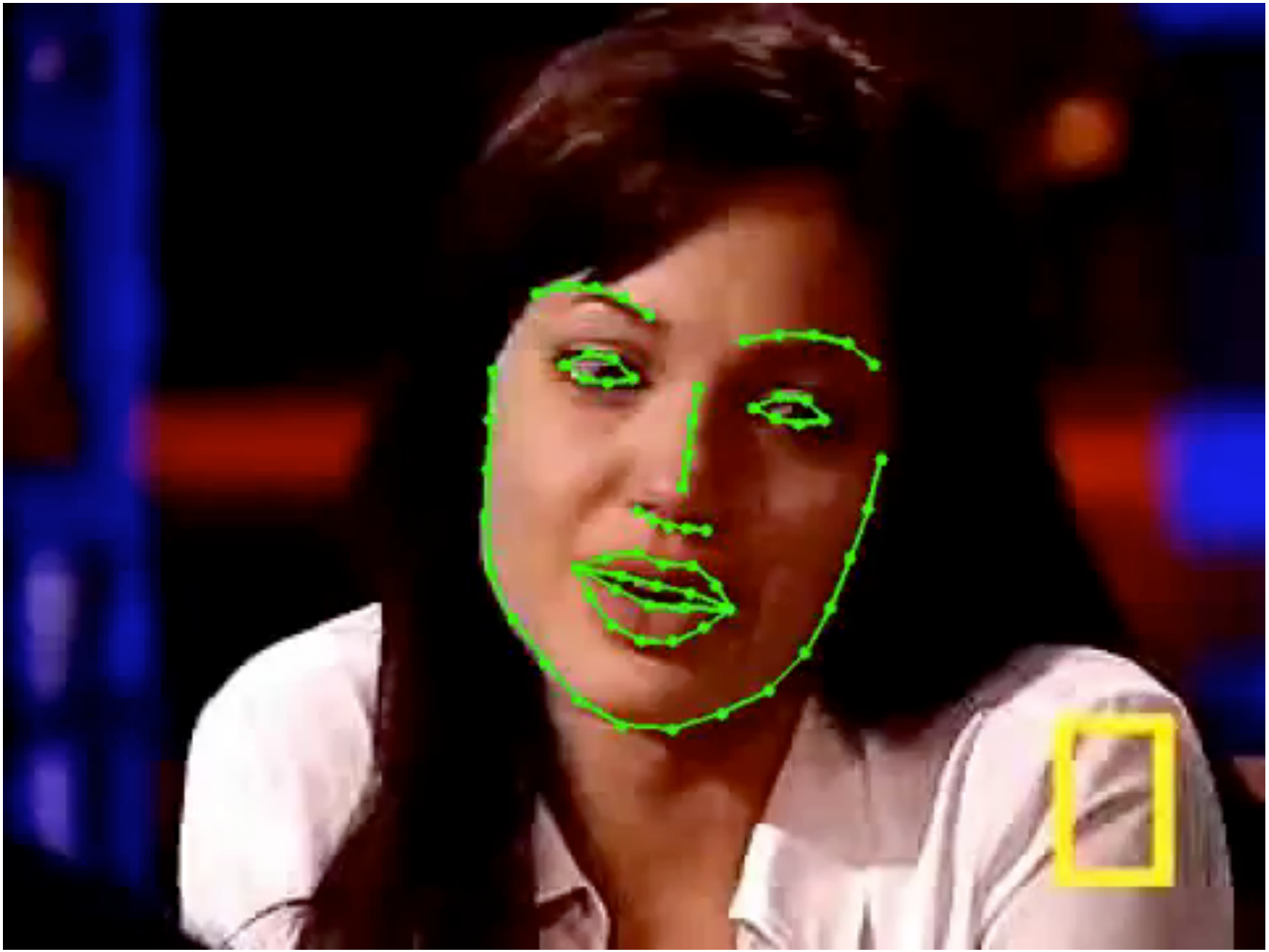
b)



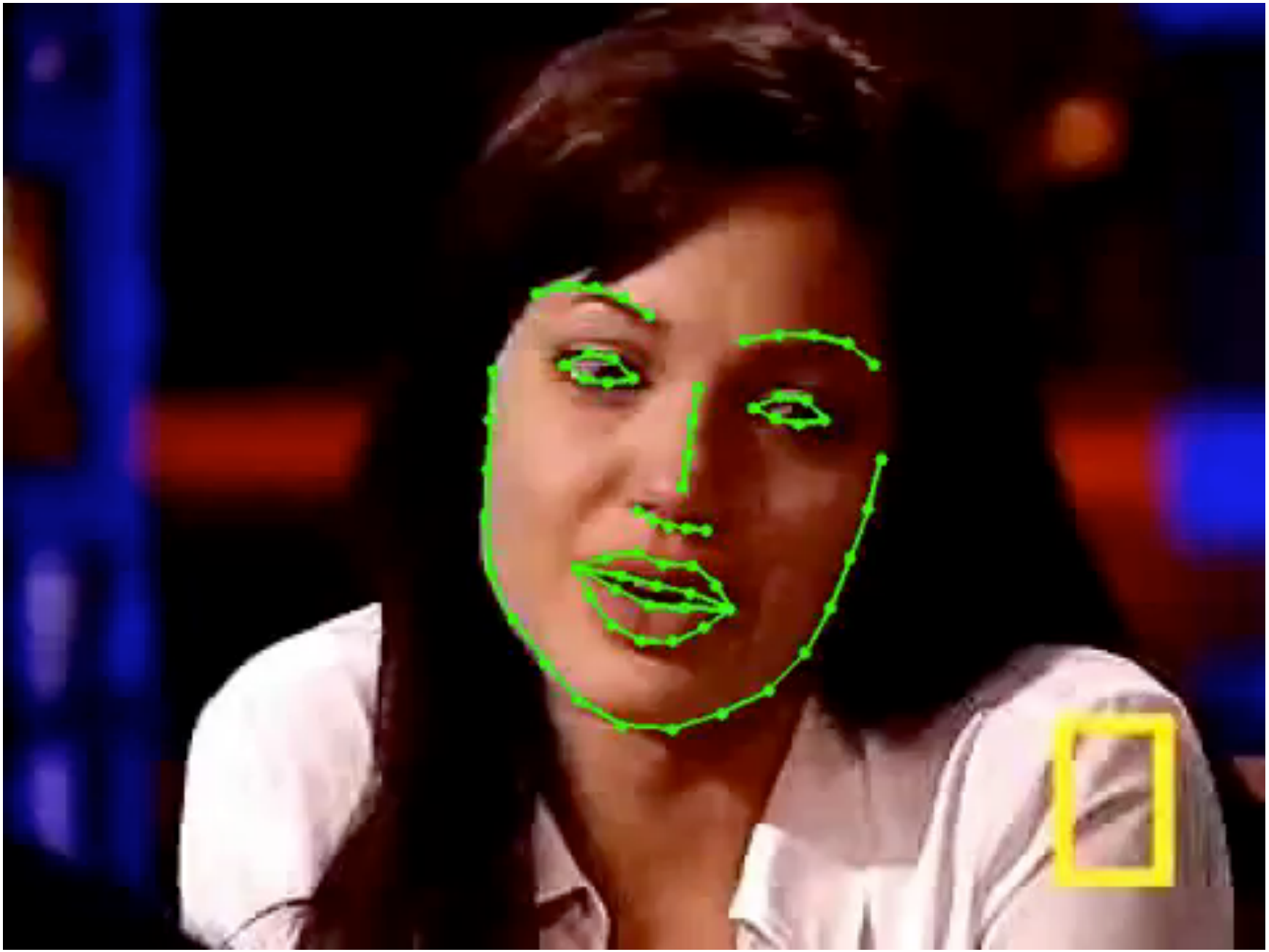
1. Compute image gradients

2. Pool into local histograms
3. Concatenate histograms
4. Normalize histograms

$$\psi \rightarrow \mathbb{R}^N : \mathbb{R}^{N \times K}$$



Xiong & De la Torre 2013



Xiong & De la Torre 2013

Jeni, Cohn, & Kanade 2015

Jeni, Cohn, & Kanade 2015

Today

- Theoretical Limits on Search
- Inverse Composition & SDM
- Conditional LK

LK-IC vs. SDM: Difference

LK-IC vs. SDM: Difference

LK

SDM

LK-IC vs. SDM: Difference

LK

- Pixel independence assumption

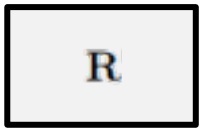
SDM

- Models full dependency across pixels

LK-IC vs. SDM: Difference

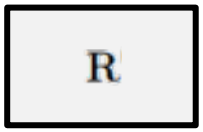
LK

- Pixel independence assumption



SDM

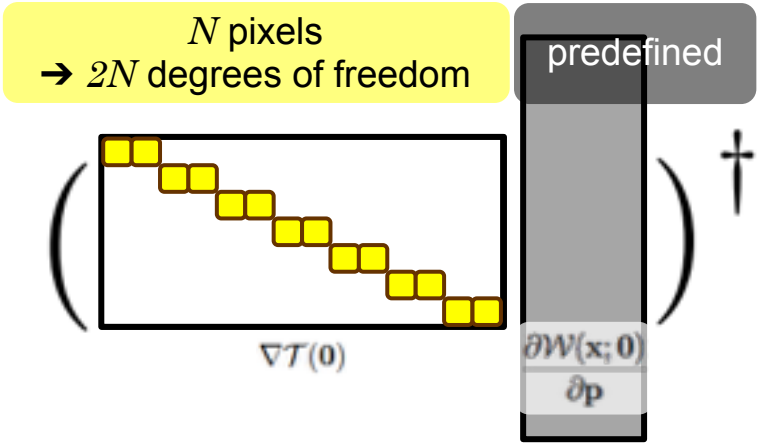
- Models full dependency across pixels



LK-IC vs. SDM: Difference

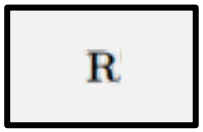
LK

- Pixel independence assumption



SDM

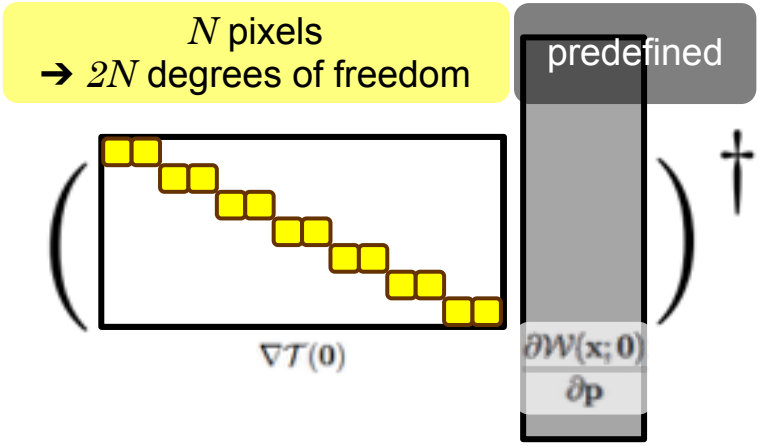
- Models full dependency across pixels



LK-IC vs. SDM: Difference

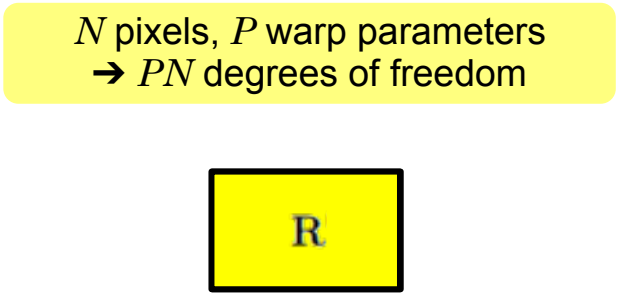
LK

- Pixel independence assumption



SDM

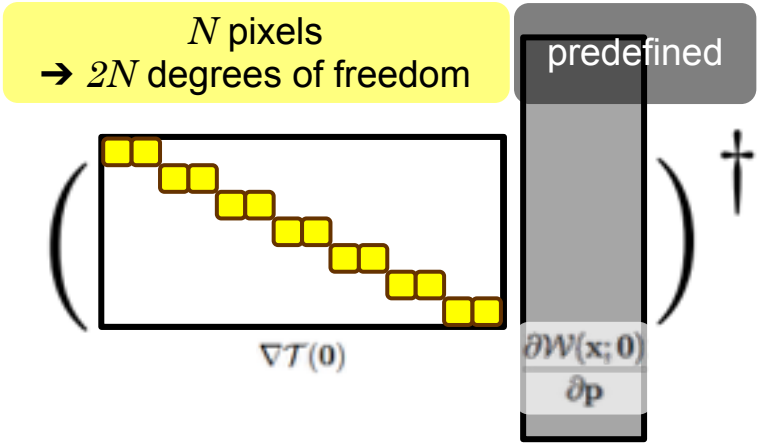
- Models full dependency across pixels



LK-IC vs. SDM: Difference

LK

- Pixel independence assumption
- Generative appearance synthesis



SDM

- Models full dependency across pixels
- Conditional learning objective

N pixels, P warp parameters
➔ PN degrees of freedom

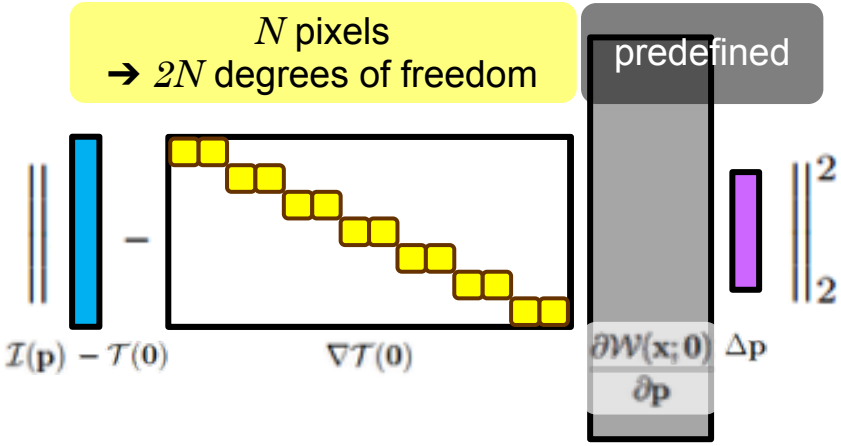
LK-IC vs. SDM: Difference

LK

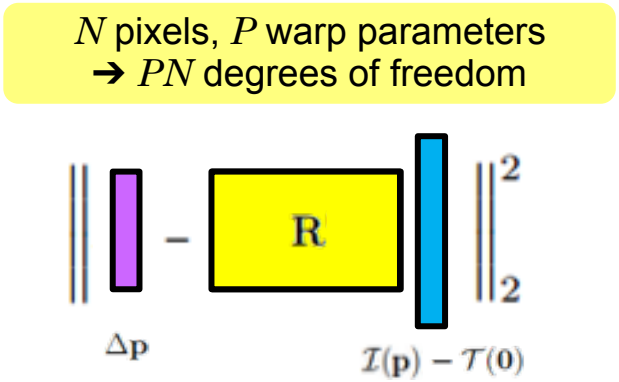
- Pixel independence assumption
- Generative appearance synthesis

SDM

- Models full dependency across pixels
- Conditional learning objective



fits $\Delta\mathbf{p}$ by trying to synthesize $\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})$



predicts $\Delta\mathbf{p}$ conditioned on $\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})$

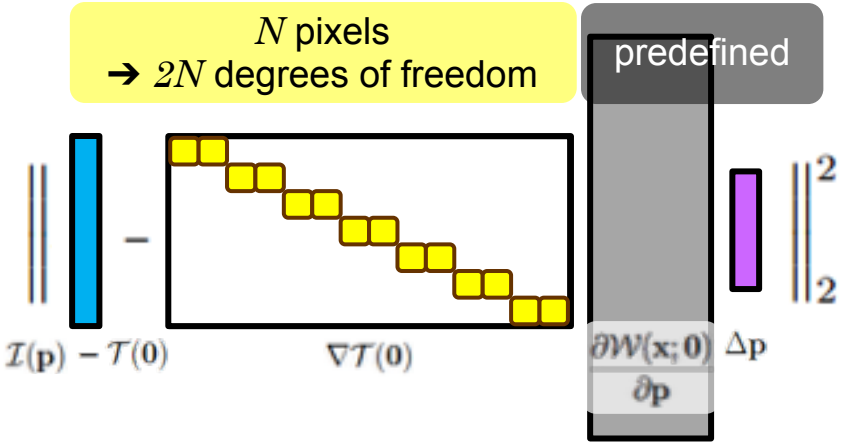
LK-IC vs. SDM: Difference

LK

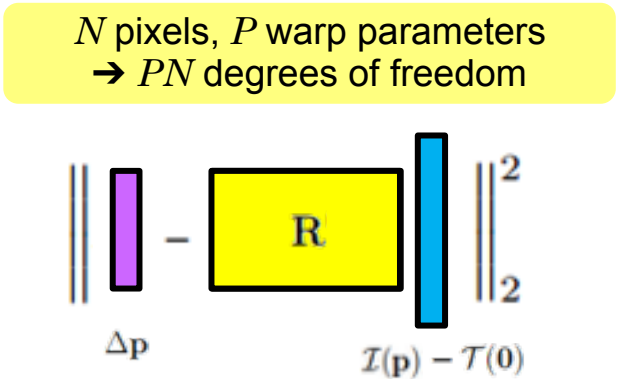
- Pixel independence assumption
- Generative appearance synthesis

SDM

- Models full dependency across pixels
- Conditional learning objective



fits $\Delta\mathbf{p}$ by trying to synthesize $\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})$

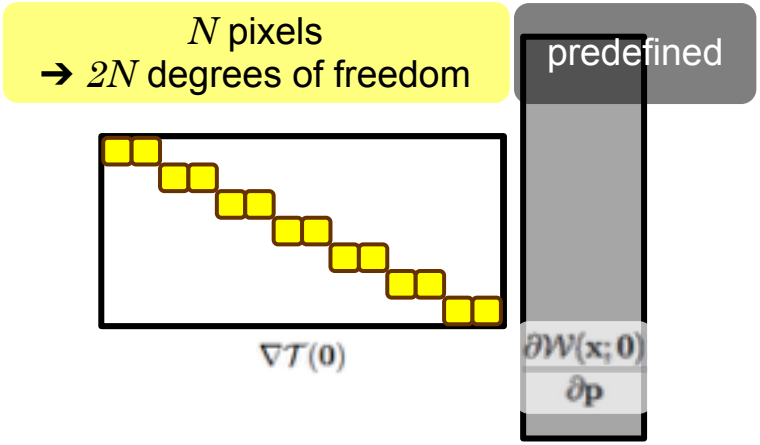


predicts $\Delta\mathbf{p}$ conditioned on $\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})$

LK-IC vs. SDM: Difference

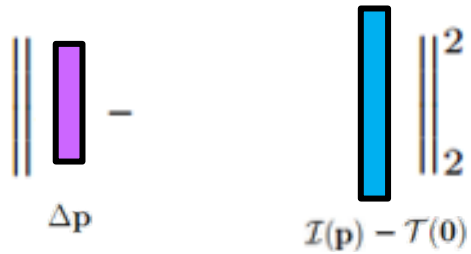
LK

- Pixel independence assumption



SDM

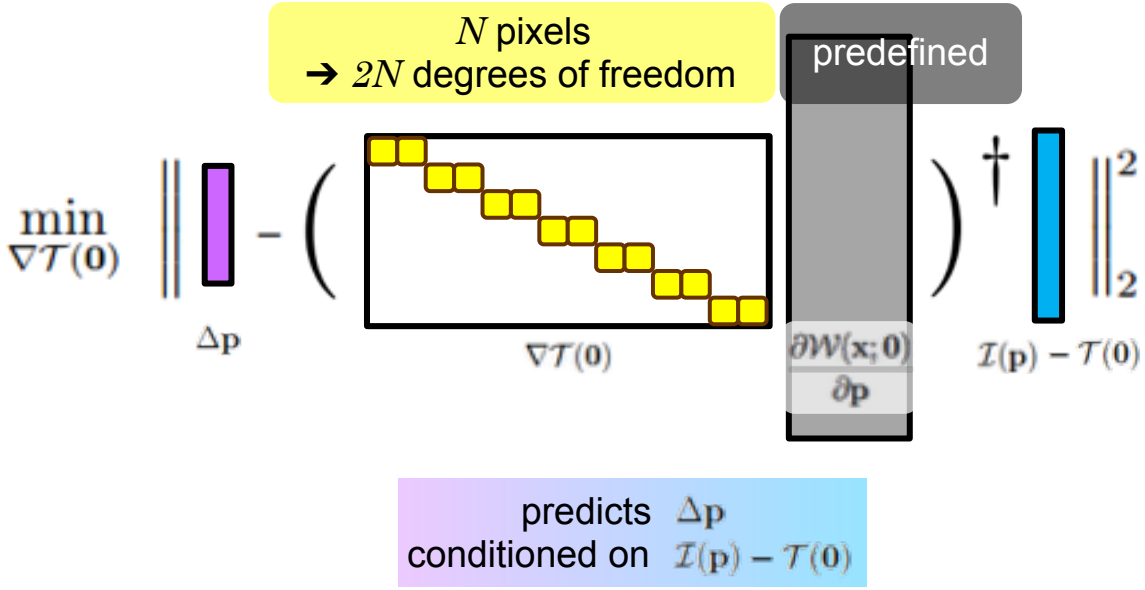
- Conditional learning objective



predicts $\Delta \mathbf{p}$
conditioned on $\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})$

- Learns the LK image gradients through a conditional objective

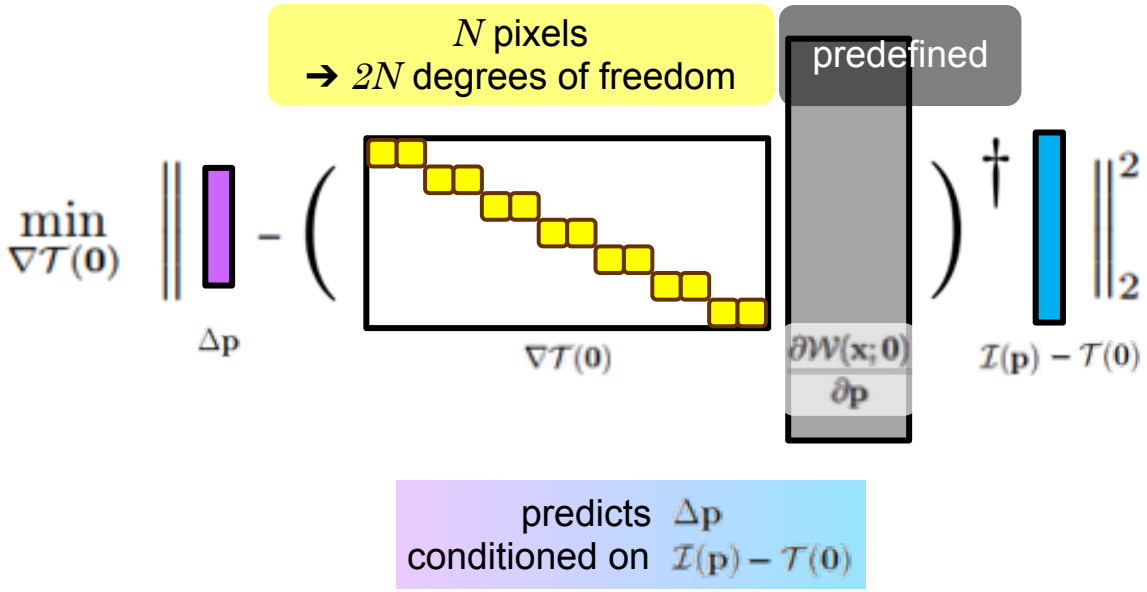
  - Pixel independence assumption
  - Conditional learning objective



The Conditional LK Algorithm

- Learns the LK image gradients through a conditional objective

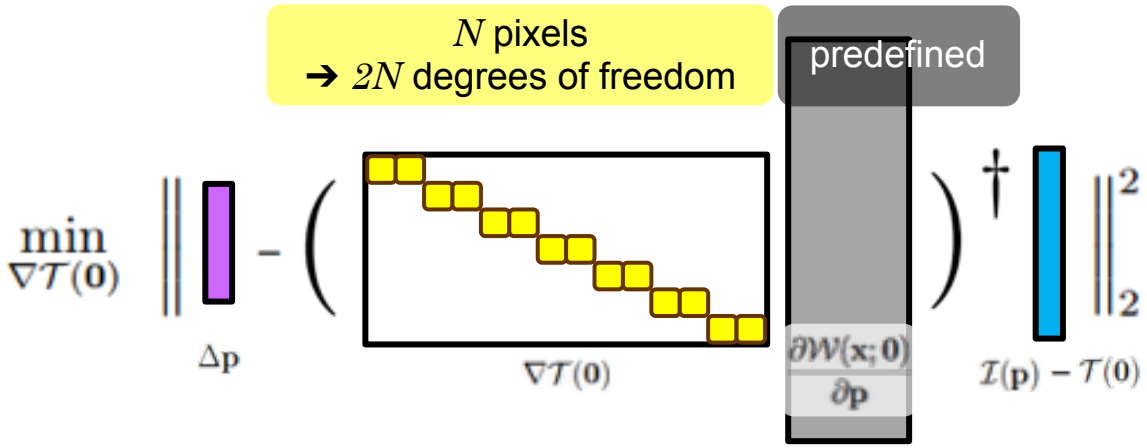
  - Pixel independence assumption
  - Conditional learning objective



Lin & Lucey 2016

The Conditional LK Algorithm

- Learns the LK image gradients through a conditional objective
- Advantages
  - Conditional learning objective
  - Pixel independence assumption
    - **Warp swapping property**: learns the conditional image gradients with one warp function and evaluate on another

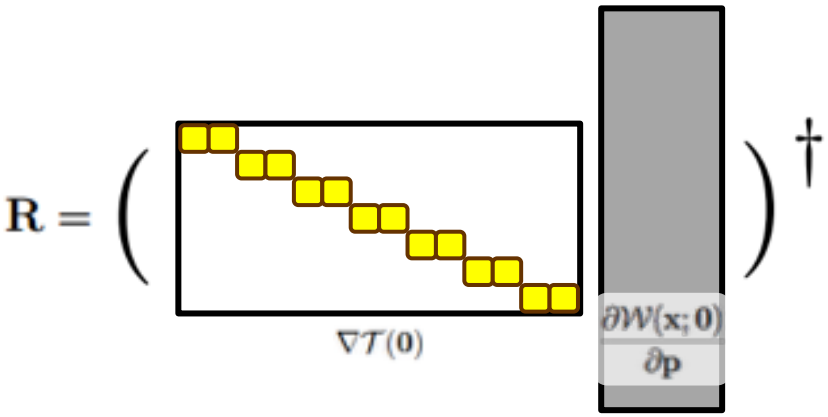


predicts $\Delta\mathbf{p}$
conditioned on $\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})$

Lin & Lucey 2016

The Conditional LK Algorithm

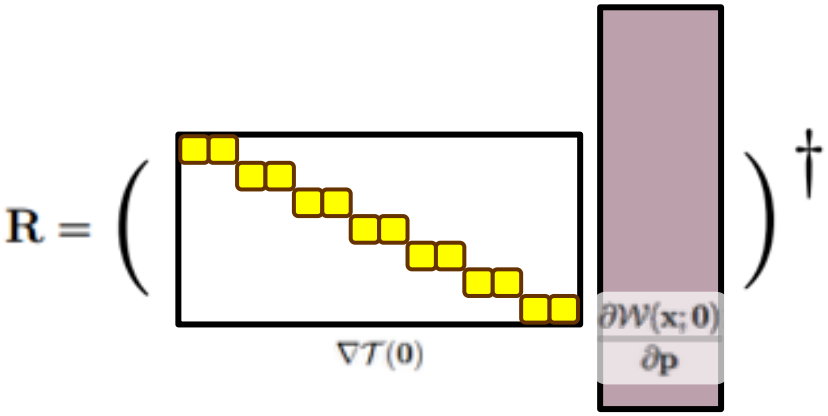
- Learns the LK image gradients through a conditional objective
- Advantages
  - Conditional learning objective
  - Pixel independence assumption
    - **Warp swapping property**: learns the conditional image gradients with one warp function and evaluate on another



Lin & Lucey 2016

The Conditional LK Algorithm

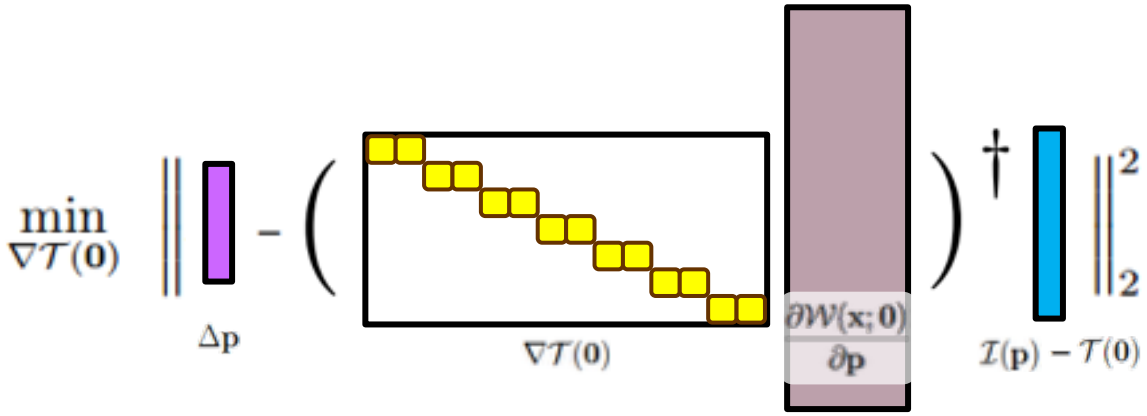
- Learns the LK image gradients through a conditional objective
- Advantages
  - Conditional learning objective
  - Pixel independence assumption
    - **Warp swapping property**: learns the conditional image gradients with one warp function and evaluate on another



Lin & Lucey 2016

The Conditional LK Algorithm

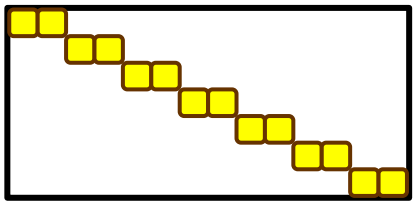
- Learns the LK image gradients through a conditional objective
- Advantages
  - Conditional learning objective
  - Pixel independence assumption
    - **Warp swapping property**: learns the conditional image gradients with one warp function and evaluate on another
- Limitations
  - Nonlinear least-squares problem



predicts $\Delta\mathbf{p}$
conditioned on $\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})$

Lin & Lucey 2016

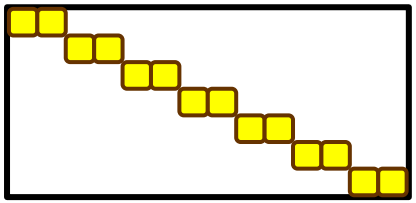
Visualization of Gradients



$\nabla T(\mathbf{0})$

Lin & Lucey 2016

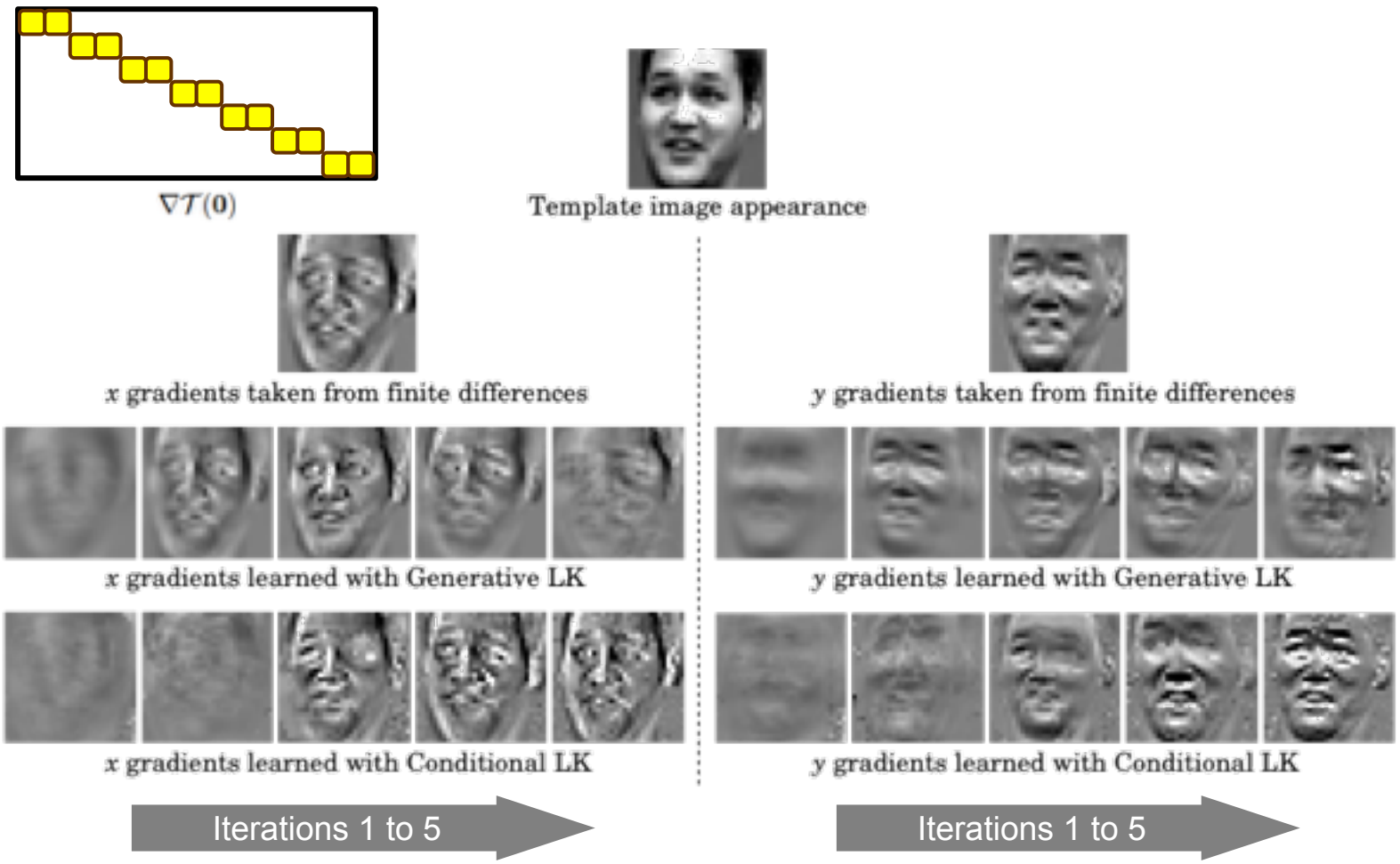
Visualization of Gradients



$\nabla T(\mathbf{0})$

Lin & Lucey 2016

Visualization of Gradients

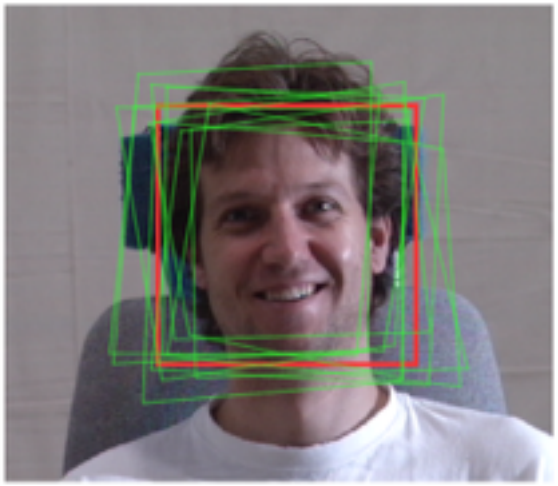


- Generative LK: gradients are learned from data in the generative form

Lin & Lucey 2016

Planar Image Alignment

- Dataset: MultiPIE
- Training data generation



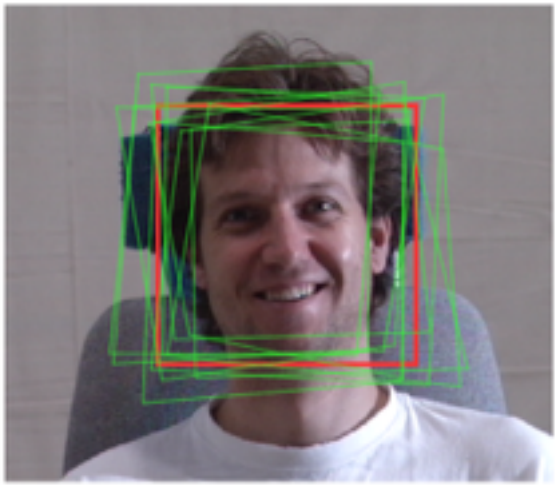
original image

red: ground truth $\{\mathbf{p}_m, \mathcal{I}_m\}_{m=1}^{M}$

green: generated examples $\mathcal{S} = \{\Delta\mathbf{p}_n, \mathcal{I}_n(\mathbf{p}_n \circ \Delta\mathbf{p}_n)\}_{n=1}^{N}$

Planar Image Alignment

- Dataset: MultiPIE
- Training data generation



$\Delta \mathbf{p}_n$

original image

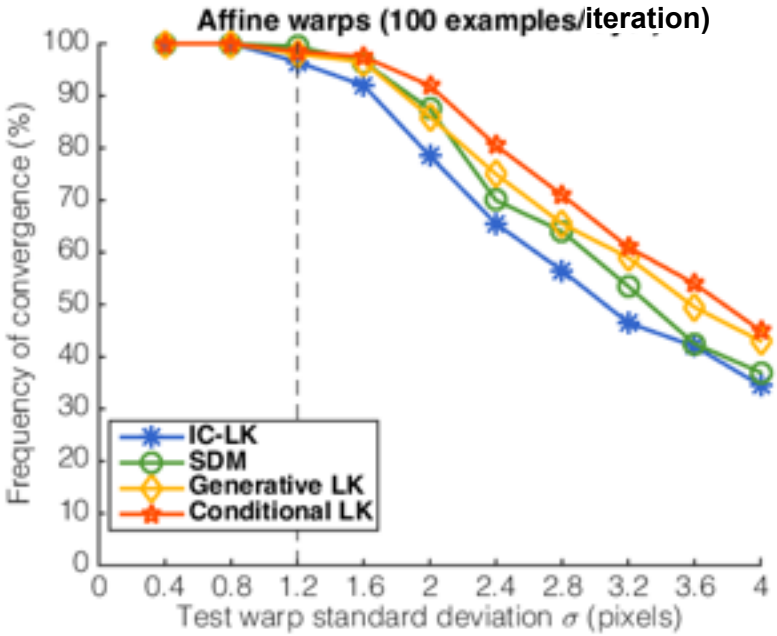
perturbed appearances

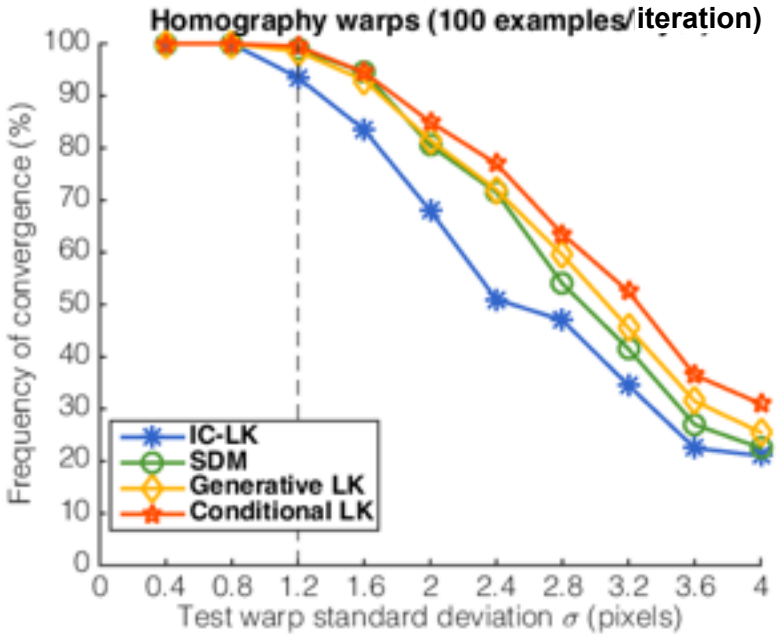
red: ground truth $\{\mathbf{p}_m, \mathcal{I}_m\}_{m=1}^{M}$

green: generated examples $\mathcal{S} = \{\Delta \mathbf{p}_n, \mathcal{I}_n(\mathbf{p}_n \circ \Delta \mathbf{p}_n)\}_{n=1}^{N}$

Planar Image Alignment

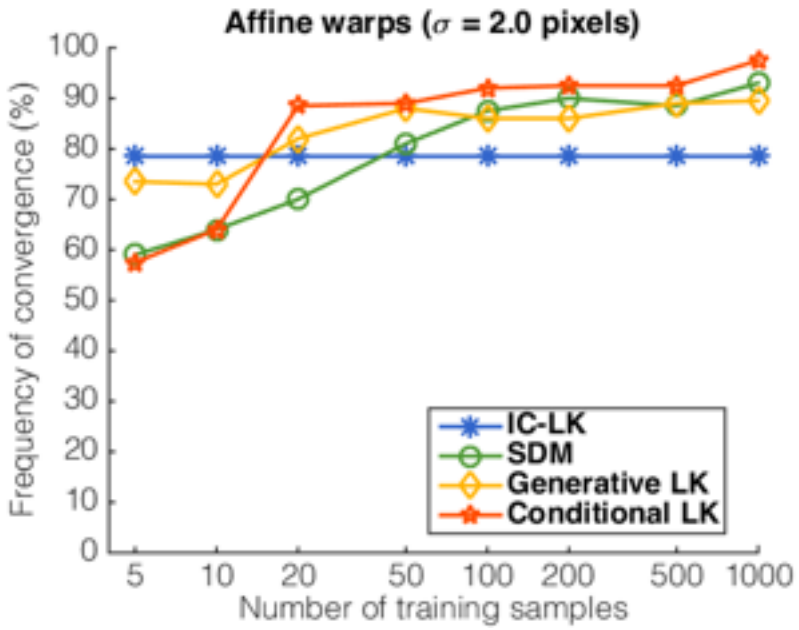
- Frequency of convergence
  - Compare the number of converged runs over total test runs
  - Evaluation: perturbations of different variances

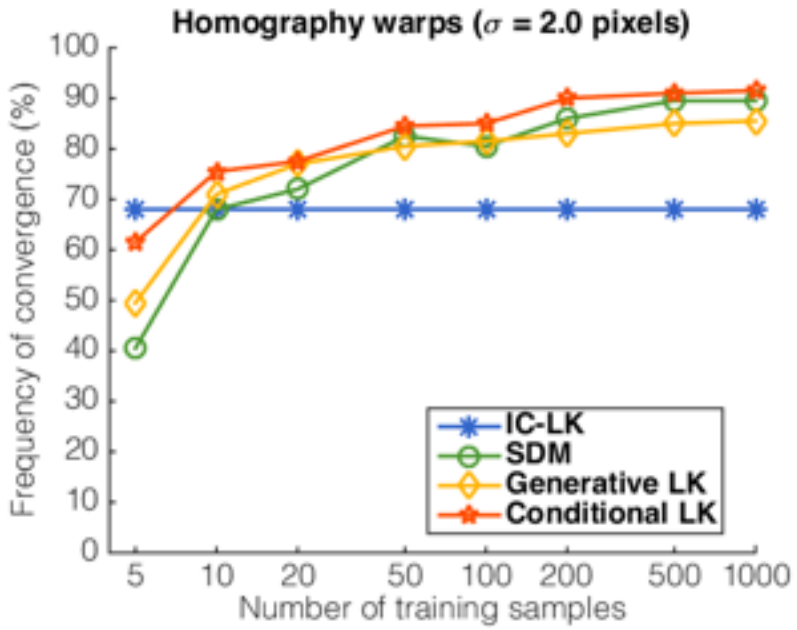




Planar Image Alignment

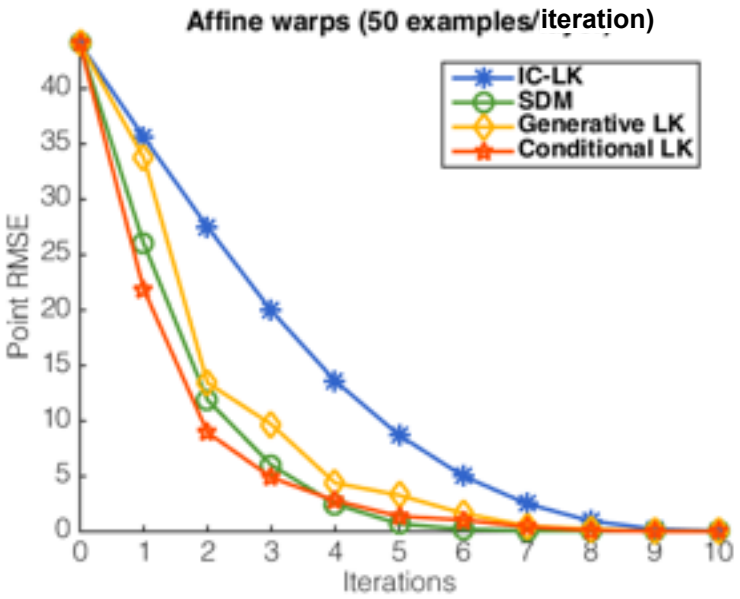
- Frequency of convergence
  - Evaluation: models trained with different amounts of data

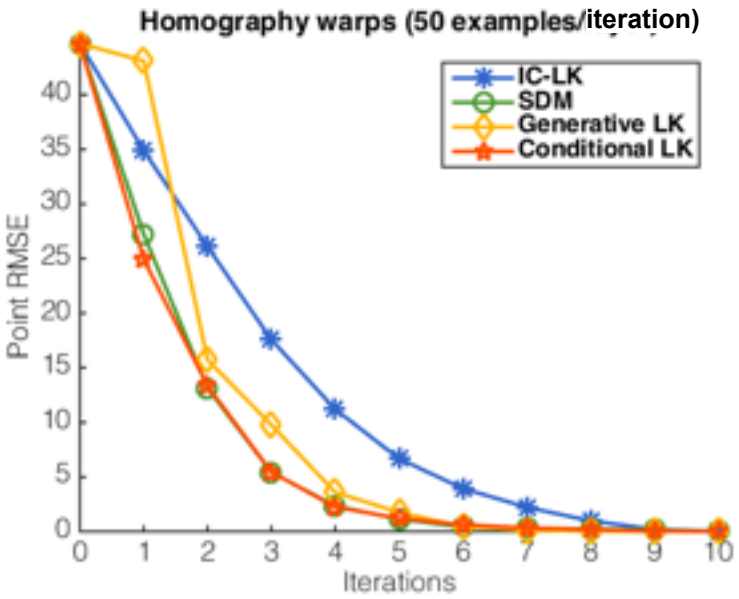




Planar Image Alignment

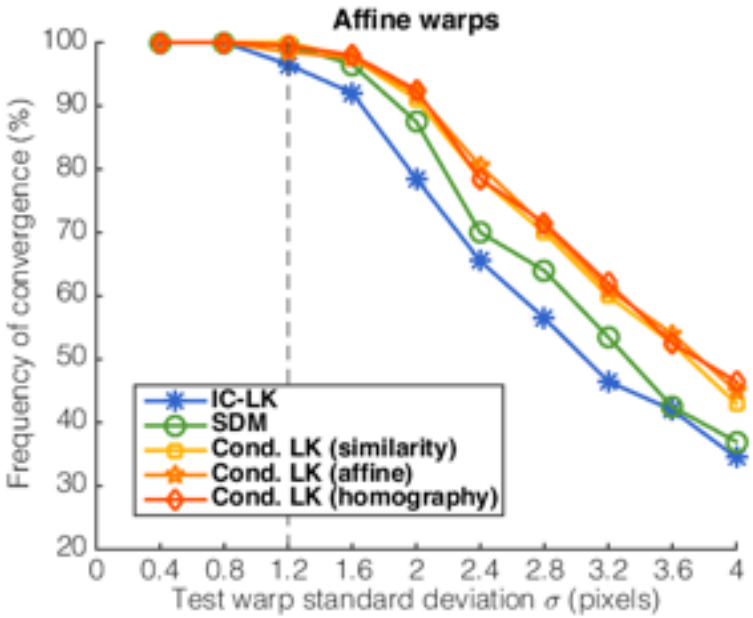
- Convergence rate
  - Evaluation: the test runs which all methods converged

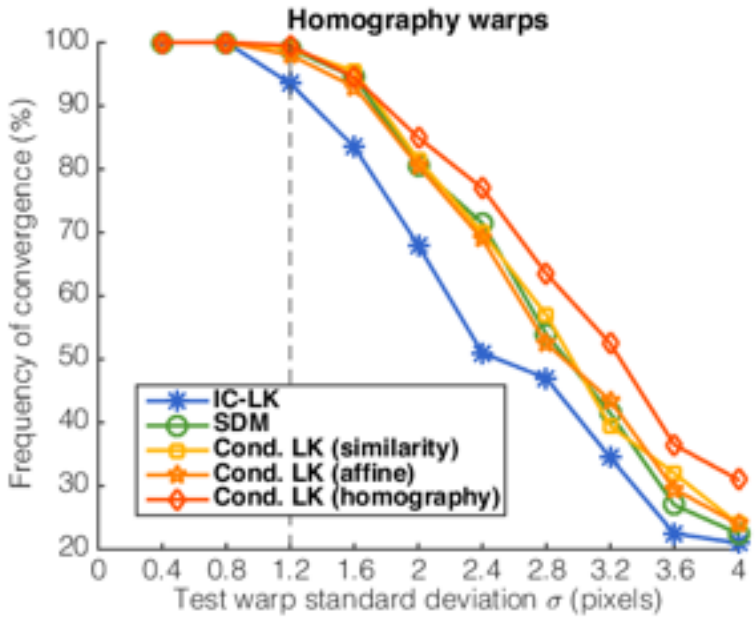




Warp Swapping Property

- Learns the conditional image gradients with different warp functions and evaluate on a specific one
- Frequency of convergence
  - Evaluation: perturbations of different variances





Tracking on Feature Images

- Low frame-rate template tracking
  - Feature: dense local binary pattern (LBP)
  - Constant lighting

BLUE: IC-LK **YELLOW: Conditional LK**

original video (120fps)

low frame-rate video (30fps)

Tracking on Feature Images

- Low frame-rate template tracking
  - Feature: dense local binary pattern (LBP)
  - Constant lighting

BLUE: IC-LK **YELLOW: Conditional LK**

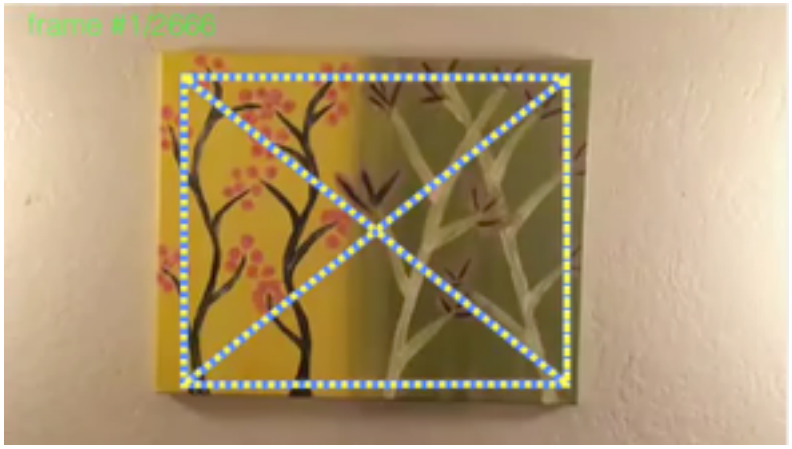
original video (120fps)

low frame-rate video (30fps)

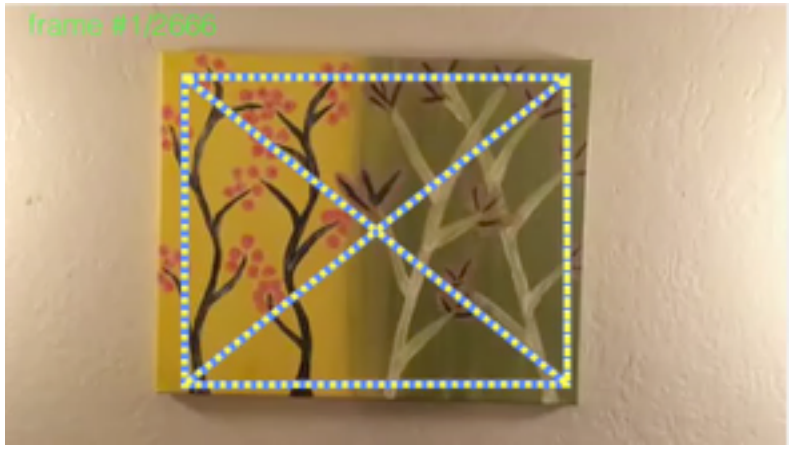
Tracking on Feature Images

- Low frame-rate template tracking
  - Feature: dense local binary pattern (LBP)
  - With lighting variations

BLUE: IC-LK **YELLOW: Conditional LK**



original video (120fps)

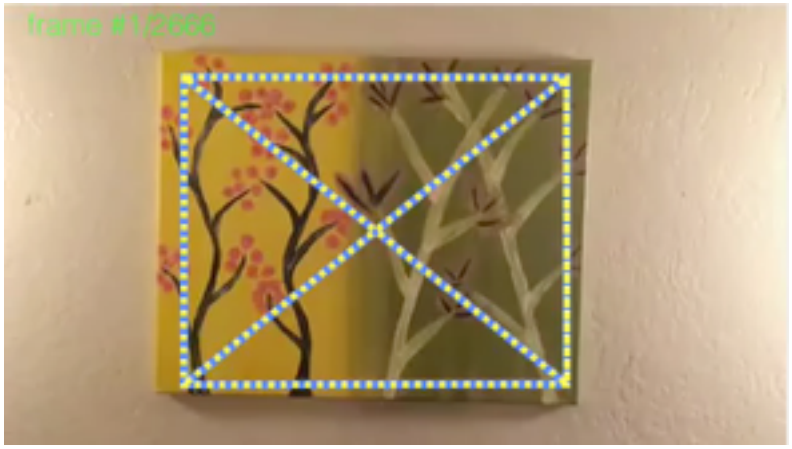


low frame-rate video (15fps)

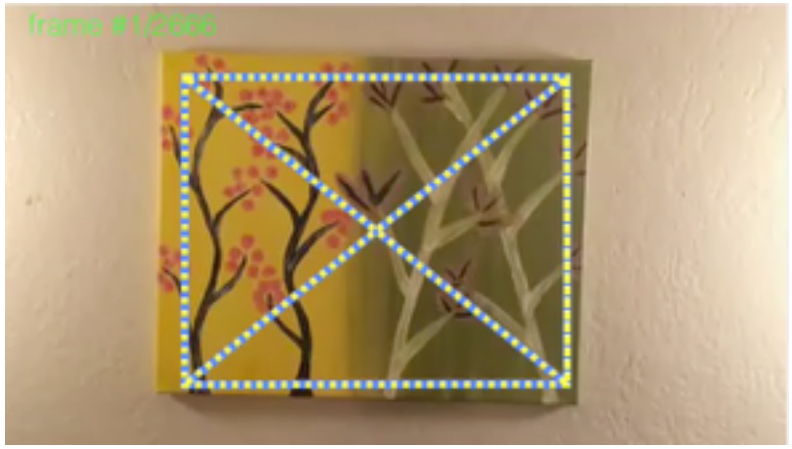
Tracking on Feature Images

- Low frame-rate template tracking
  - Feature: dense local binary pattern (LBP)
  - With lighting variations

BLUE: IC-LK **YELLOW: Conditional LK**



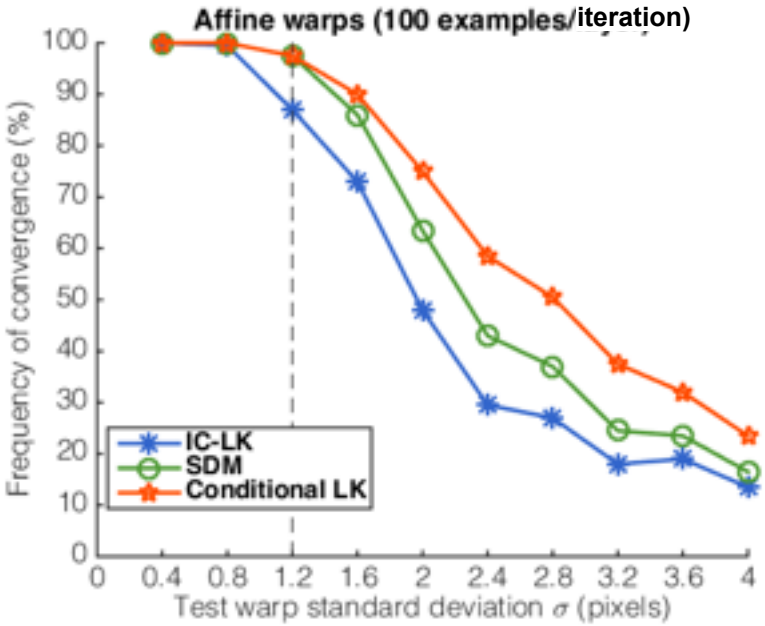
original video (120fps)

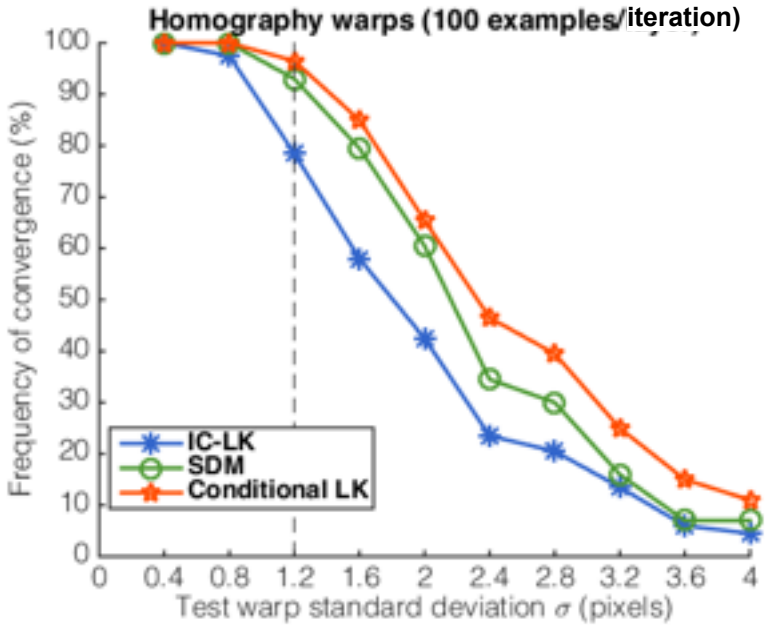


low frame-rate video (15fps)

Tracking on Feature Images

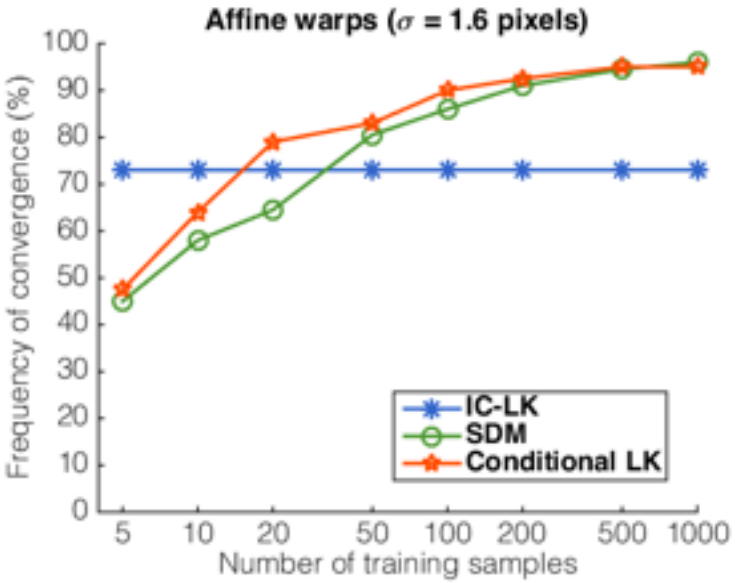
- Frequency of convergence
  - Feature: dense local binary pattern (LBP)
  - Evaluation: perturbations of different variances

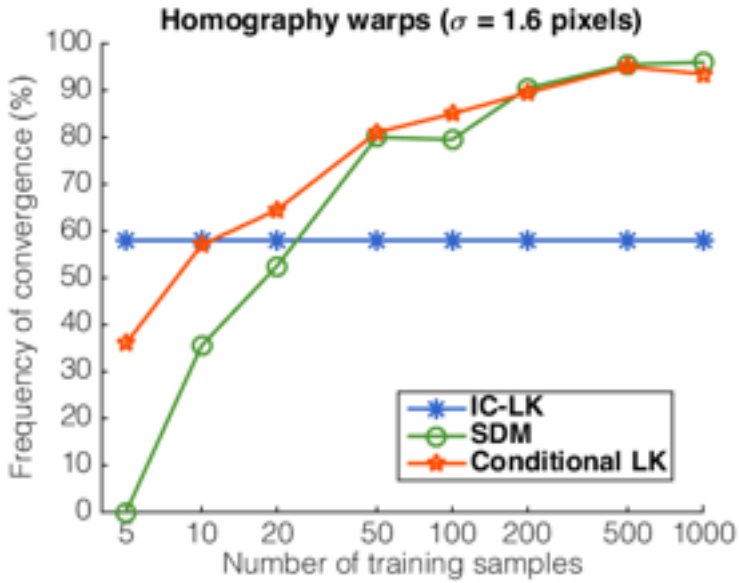




Tracking on Feature Images

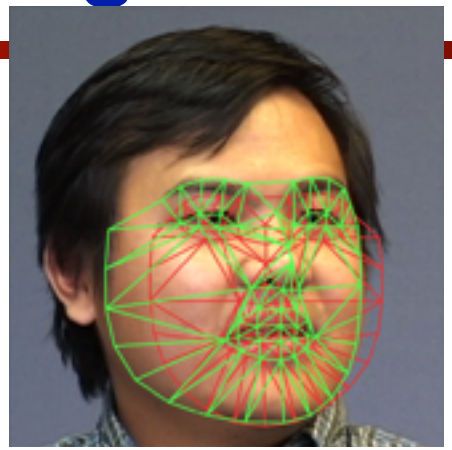
- Frequency of convergence
  - Feature: dense local binary pattern (LBP)
  - Evaluation: models trained with different amounts of data





Facial Model Fitting

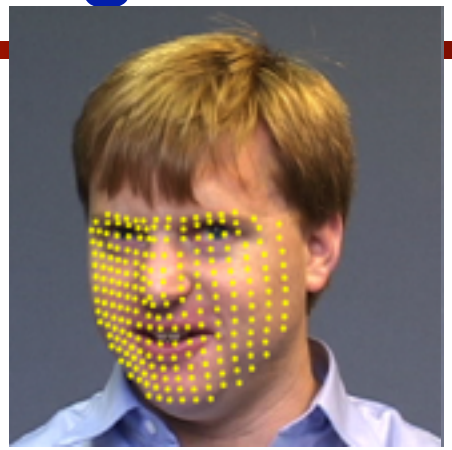
- Dataset: IJAGS
- Training: few examples per subject
  Evaluation: entire dataset



red: initialization
green: fitting result

Facial Model Fitting

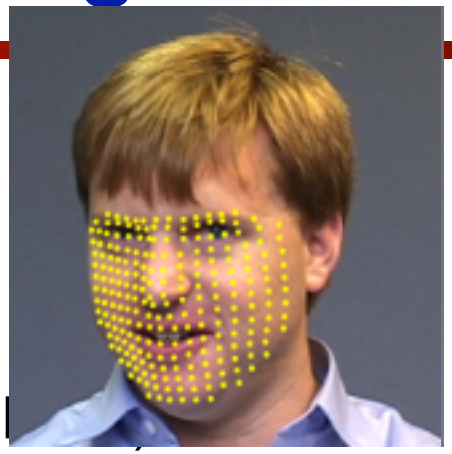
- Dataset: IJAGS
- Training: few examples per subject
  Evaluation: entire dataset



yellow: dense SIFT sampling locations

Facial Model Fitting

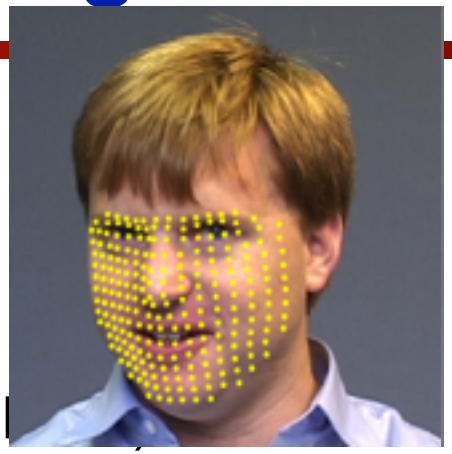
- Dataset: IJAGS
- Training: few examples per subject
  Evaluation: entire dataset
- Warp function: 2D point distribution model (
  - Can be potentially swapped with a 3D PDM



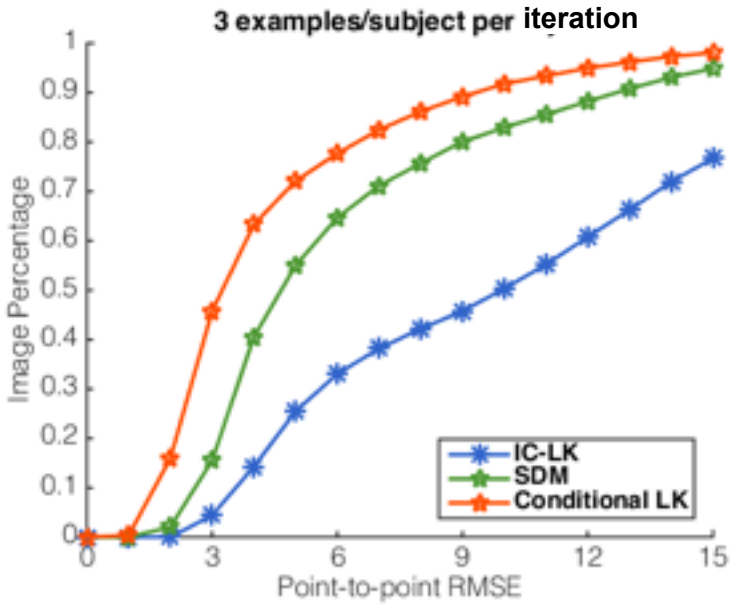
yellow: dense SIFT sampling locations

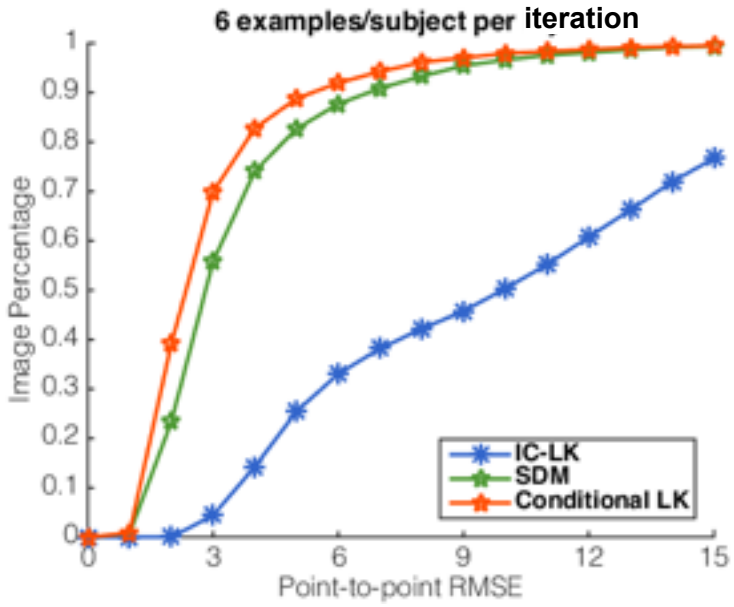
Facial Model Fitting

- Dataset: IJAGS
- Training: few examples per subject
  Evaluation: entire dataset
- Warp function: 2D point distribution model (
  - Can be potentially swapped with a 3D PDM
- Fitting accuracy (facial landmarks)



yellow: dense SIFT sampling locations





More to read…

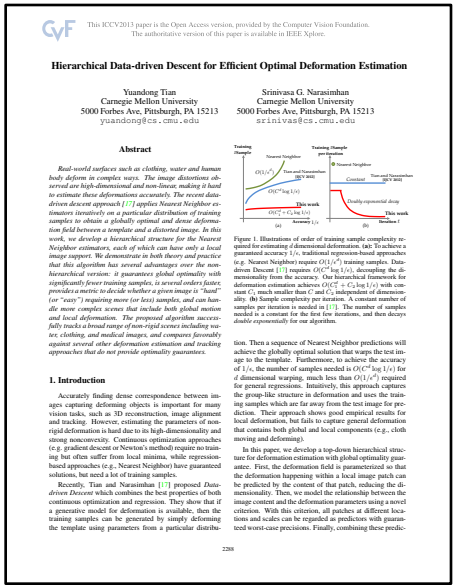
Parameterizing Homographies

CMU-RI-TR-06-11

Simon Baker, Ankur Datta, and Takeo Kanade

The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213

The motion of a plane can be described by a homography. We study how to parameterize homographies to maximize plane estimation performance. We compare the usual 3×3 matrix parameterization with a parameterization that combines 4 fixed points in one of the images with 4 variable points in the other image. We empirically show that this 4pt parameterization is far superior. We also compare both parameterizations with a variety of direct parameterizations. In the case of unknown relative orientation, we compare with a direct parameterization of the plane equation, and the rotation and translation of the camera(s). We show that the direct parameterization is both less accurate and far less robust than the 4-point parameterization. We explain the poor performance using a measure of independence of the Jacobian images. In the fully calibrated setting, the direct parameterization just consists of 3 parameters of the plane equation. We show that this parameterization is far more robust than the 4-point parameterization, but only approximately as accurate. In the case of a moving stereo rig we find that the direct parameterization of plane equation, camera rotation and translation performs very well, both in terms of accuracy and robustness. This is in contrast to the corresponding direct parameterization in the case of unknown relative orientation. Finally, we illustrate the use of plane estimation in 2 automotive applications.



Hierarchical Data-driven Descent for Efficient Optimal Deformation Estimation

Yuandong Tian
Carnegie Mellon University
5000 Forbes Ave, Pittsburgh, PA 15213
yuandong@cs.cmu.edu

Srinivasa G. Narasimhan
Carnegie Mellon University
5000 Forbes Ave, Pittsburgh, PA 15213
srinivas@cs.cmu.edu

Abstract

Real-world surfaces such as clothing, water and human body deform in complex ways. The image distortions observed are high-dimensional and non-linear, making it hard to estimate these deformations accurately. The recent data-driven descent approach [17] applies Nearest Neighbor estimators iteratively on a particular distribution of training samples to obtain a globally optimal and dense deformation field between a template and a distorted image. In this work, we develop a hierarchical structure for the Nearest Neighbor estimators, each of which can have only a local image support. We demonstrate in both theory and practice that this algorithm has several advantages over the non-hierarchical version: it guarantees global optimality with significantly fewer training samples, is several orders faster, provides a metric to decide whether a given image is "hard" (or "easy") requiring more (or less) samples, and can handle more complex scenes that include both global motion and local deformation. The proposed algorithm successfully tracks a broad range of non-rigid scenes including water, clothing, and medical images, and compares favorably against several other deformation estimation and tracking approaches that do not provide optimality guarantees.

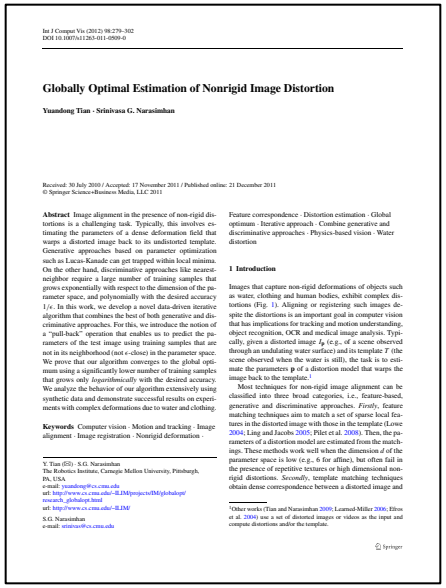
1. Introduction

Accurately finding dense correspondence between images capturing deforming objects is important for many vision tasks, such as 3D reconstruction, image alignment and tracking. However, estimating the parameters of non-rigid deformation is hard due to its high-dimensionality and strong non-convexity. Continuous optimization approaches (e.g. gradient descent or Newton's method) require no training but often suffer from local minima, while regression-based approaches (e.g., Nearest Neighbor) have guaranteed solutions, but need a lot of training samples.

Recently, Tian and Narasimhan [17] proposed *Data-driven Descent* which combines the best properties of both continuous optimization and regression. They show that if a generative model for deformation is available, then the training samples can be generated by simply deforming the template using parameters from a particular distribution. Then a sequence of Nearest Neighbor predictions will achieve the globally optimal solution that warps the test image to the template. Furthermore, to achieve the accuracy of $1/\epsilon$, the number of samples needed is $O(C^d \log 1/\epsilon)$ for d dimensional warping, much less than $O(1/\epsilon^d)$ required for general regressions. Intuitively, this approach captures the group-like structure in deformation and uses the training samples which are far away from the test image for prediction. These approach shows good empirical results for local deformation, but fails to capture general deformation that contains both global and local components (e.g., cloth moving and deforming).

Figure 1. Illustrations of order of training sample complexity required for estimating d dimensional deformation. (a): To achieve a guaranteed accuracy $1/\epsilon$, traditional regression-based approaches (e.g. Nearest Neighbor) require $O(1/\epsilon^d)$ training samples. Data-driven Descent [17] requires $O(C^d \log 1/\epsilon)$, decoupling the dimensionality from the accuracy. Our hierarchical framework for deformation estimation achieves $O(C^d + C_2 \log 1/\epsilon)$ with constant C, much smaller than C and C_2 independent of dimensionality. (b) Sample complexity per iteration. A constant number of samples per iteration is needed in [17]. The number of samples needed is a constant for the first few iterations, and then decays *double exponentially* for our algorithm.

In this paper, we develop a top-down hierarchical structure for deformation estimation with global optimality guarantee. First, the deformation field is parameterized so that the deformation happening within a local image patch can be predicted by the content of that patch, reducing the dimensionality. Then, we model the relationship between the image content and the deformation parameters using a novel criterion. With this criterion, all patches at different locations and scales can be regarded as predictors with guaranteed worst-case precisions. Finally, combining these predic-



Int J Comput Vis (2012) 98:279–302
DOI 10.1007/s11263-011-0509-0

Globally Optimal Estimation of Nonrigid Image Distortion

Yuandong Tian · Srinivasa G. Narasimhan

Received: 30 July 2010 / Accepted: 17 November 2011 / Published online: 21 December 2011

Abstract Image alignment in the presence of non-rigid distortions is a challenging task. Typically, this involves estimating the parameters of a dense deformation field that warps a distorted image back to its undistorted template. Generative approaches based on parameter optimization such as Lucas-Kanade can get trapped within local minima. On the other hand, discriminative approaches like nearest-neighbor require a large number of training samples that grows exponentially with respect to the dimension of the parameter space, and polynomially with the desired accuracy $1/\epsilon$. In this work, we develop a novel data-driven iterative algorithm that combines the best of both generative and discriminative approaches. For this, we introduce the notion of a "pull-back" operation that enables us to predict the parameters of the test image using training samples that are not in its neighborhood (not ϵ-close) in the parameter space. We prove that our algorithm converges to the global optimum using a significantly lower number of training samples that grows only *logarithmically* with the desired accuracy. We analyze the behavior of our algorithm extensively using synthetic data and demonstrate successful results on experiments with complex deformations due to water and clothing.

Keywords Computer vision · Motion and tracking · Image alignment · Image registration · Nonrigid deformation · Feature correspondence · Distortion estimation · Global optimum · Iterative approach · Combine generative and discriminative approaches · Physics-based vision · Water distortion

1 Introduction

Images that capture non-rigid deformations of objects such as water, clothing and human bodies, exhibit complex distortions (Fig. 1). Aligning or registering such images despite the distortions is an important goal in computer vision that has implications for tracking and motion understanding, object recognition, OCR and medical image analysis. Typically, given a distorted image $\mathbf{I_d}$ (e.g., of a scene observed through an undulating water surface) and its template $\mathcal{T}$ (the scene observed when the water is still), the task is to estimate the parameters $\mathbf{p}$ of a distortion model that warps the image back to the template.[1]

Most techniques for non-rigid image alignment can be classified into three broad categories, i.e., feature-based, generative and discriminative approaches. *Firstly*, feature matching techniques aim to match a set of sparse local features in the distorted image with those in the template (Lowe 2004; Ling and Jacobs 2005; Pilet et al. 2008). Then, the parameters of a distortion model are estimated from the matchings. These methods work well when the dimension d of the parameter space is low (e.g., 6 for affine), but often fail in the presence of repetitive textures or high dimensional non-rigid distortions. *Secondly*, template matching techniques obtain dense correspondence between a distorted image and

Y. Tian (✉) · S.G. Narasimhan
The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: yuandong@cs.cmu.edu
url: http://www.cs.cmu.edu/~ILIM/projects/IM/globalopt/research_globalopt.html
url: http://www.cs.cmu.edu/~ILIM/

S.G. Narasimhan
e-mail: srinivas@cs.cmu.edu

[1] Other works (Tian and Narasimhan 2009; Learned-Miller 2006; Efros et al. 2004) use a set of distorted images or videos as the input and compute distortions and/or the template.

- Baker & Matthews, "Lucas-Kanade 20 Years On: A Unifying Framework", IJCV 2004.
- Tian & Narasimhan, "Globally Optimal Estimation of Nonrigid Image Distortion", IJCV 2012.
- Tian & Narasimhan, "Hierarchical Data-driven Descent for Efficient Optimal Deformation Estimation", ICCV 2013.