Localization of Humans in Images Using Convolutional Networks

Jonathan Tompson

Advisor: Christoph Bregler

OVERVIEW

Thesis goal:

State-of-the-art human pose recognition

from depth
(hand tracking)

from RGB
(body tracking)

Domain Specific Optimizations:

Kinematic structure in ConvNet (hybrid Graphical Model + ConvNet)

Motion feature inputs

Novel architecture to recover lost spatial precision due to pooling

TALK OUTLINE

1. Hand Tracking
    a) J. Tompson, M. Stein, Y. LeCun, K. Perlin, “Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks”, ACM TOG/SIGGRAPH 2014
2. Body Tracking
    a) J. Tompson, A. Jain, Y. LeCun, C. Bregler, “Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation”, NIPS 2014
    b) A. Jain, J. Tompson, Y. LeCun, C. Bregler, “MoDeep: A Deep Learning Framework Using Motion Features for Human Pose Estimation”, ACCV 2014
    c) J. Tompson, R. Goroshin, A. Jain, Y. LeCun, C. Bregler, “Efficient Object Localization Using Convolution Networks”, CVPR 2015

TALK OUTLINE

1. Hand Tracking
    a) J. Tompson, M. Stein, Y. LeCun, K. Perlin, “Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks”, ACM TOG/SIGGRAPH 2014
2. Body Tracking
    a) J. Tompson, A. Jain, Y. LeCun, C. Bregler, “Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation”, NIPS 2014
    b) A. Jain, J. Tompson, Y. LeCun, C. Bregler, “MoDeep: A Deep Learning Framework Using Motion Features for Human Pose Estimation”, ACCV 2014
    c) J. Tompson, R. Goroshin, A. Jain, Y. LeCun, C. Bregler, “Efficient Object Localization Using Convolution Networks”, CVPR 2015

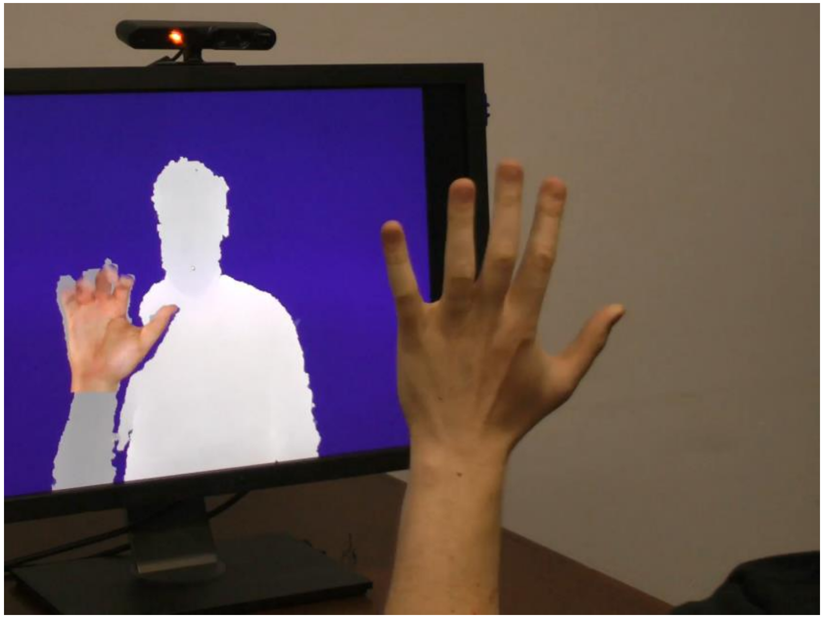
Hand Pose Inference

Target: low-cost markerless mocap

- Full articulated pose with high DoF
- Real-time with low latency

Challenges

- Many DoF contribute to model deformation
- Constrained unknown parameter space
- Self-similar parts
- Self occlusion
- Device noise



PIPELINE

Supervised learning based approach

- Needs labeled dataset + machine learning
- Existing datasets had limited pose information for hands

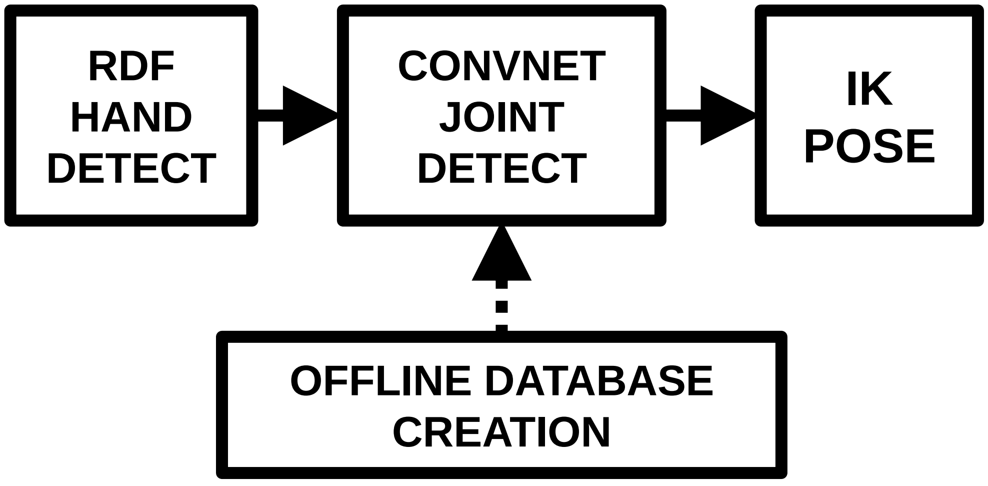
Architecture

PIPELINE

Supervised learning based approach

- Needs labeled dataset + machine learning
- Existing datasets had limited pose information for hands

Architecture

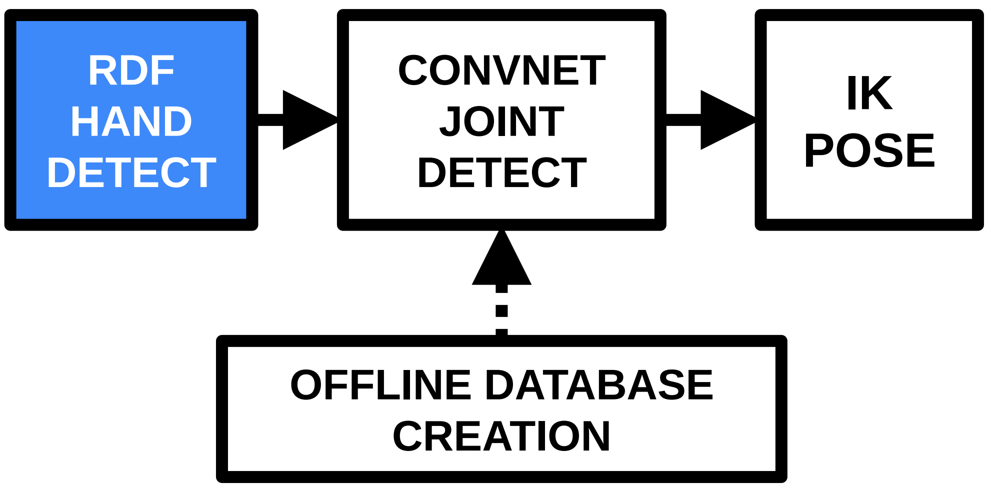


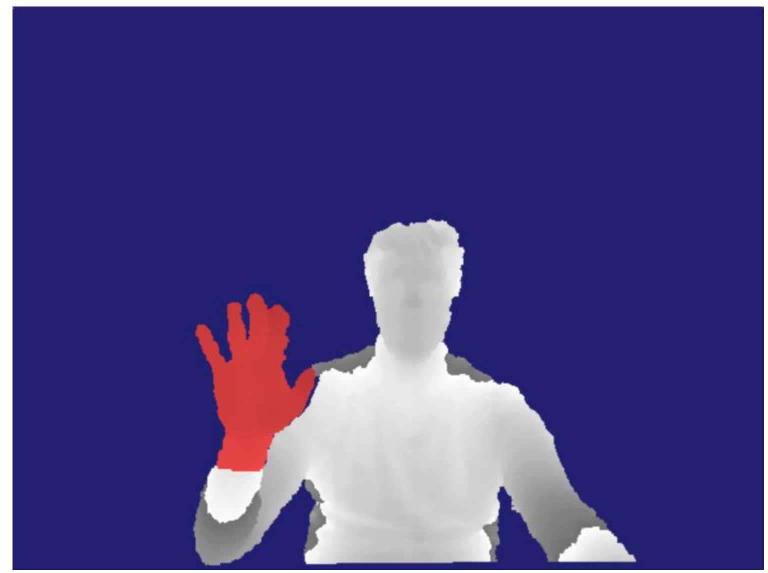
PIPELINE

Supervised learning based approach

- Needs labeled dataset + machine learning
- Existing datasets had limited pose information for hands

Architecture



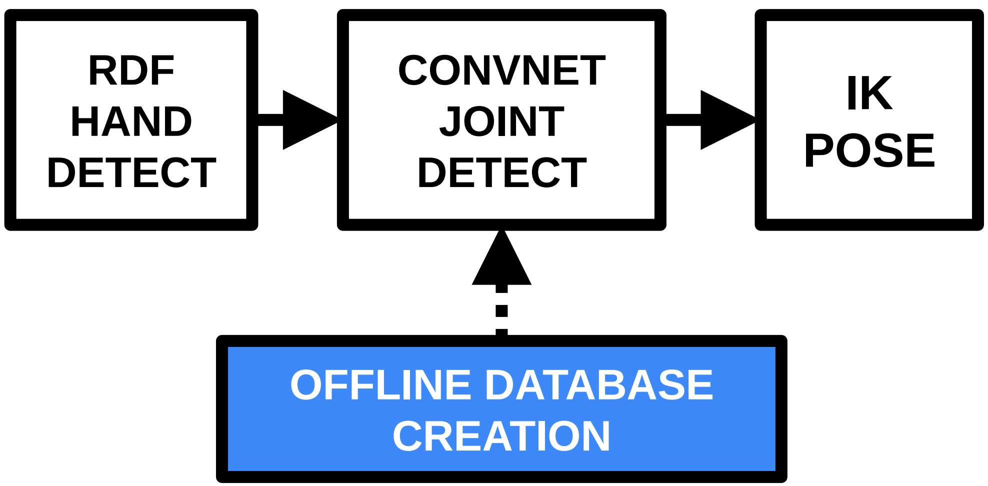


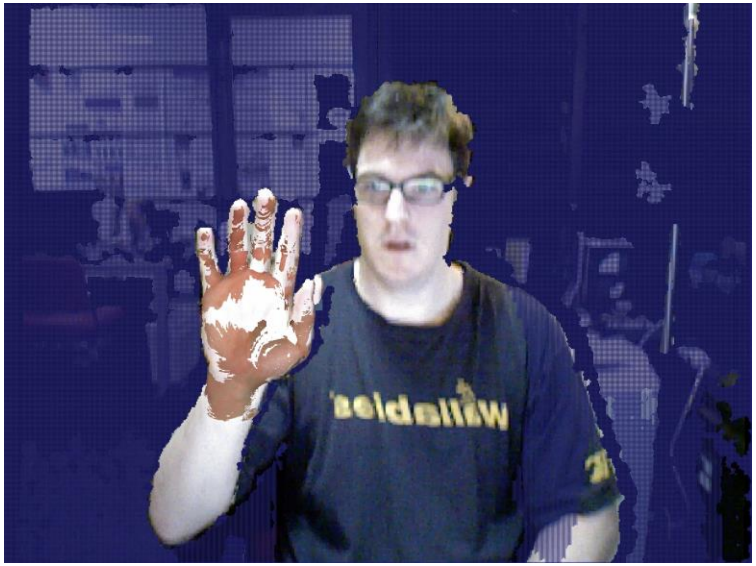
Pipeline

Supervised learning based approach

- Needs labeled dataset + machine learning
- Existing datasets had limited pose information for hands

Architecture



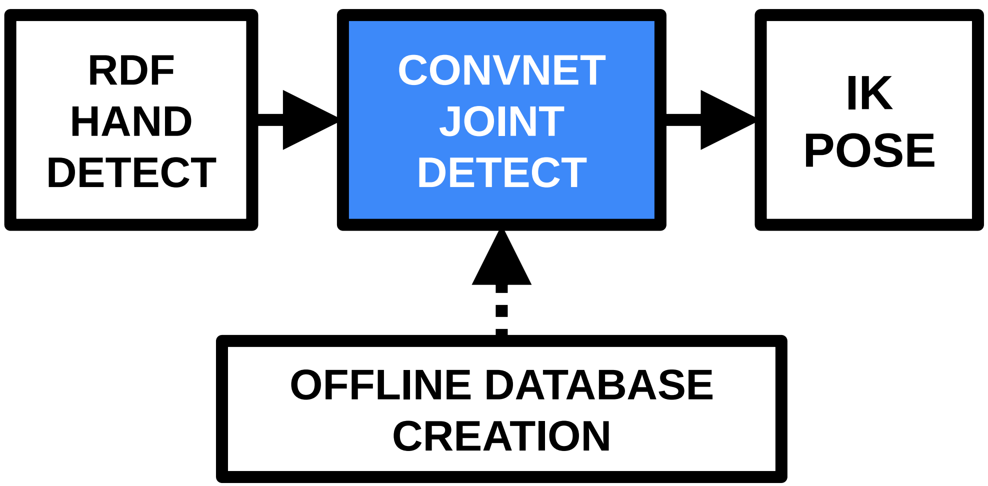


PIPELINE

Supervised learning based approach

- Needs labeled dataset + machine learning
- Existing datasets had limited pose information for hands

Architecture

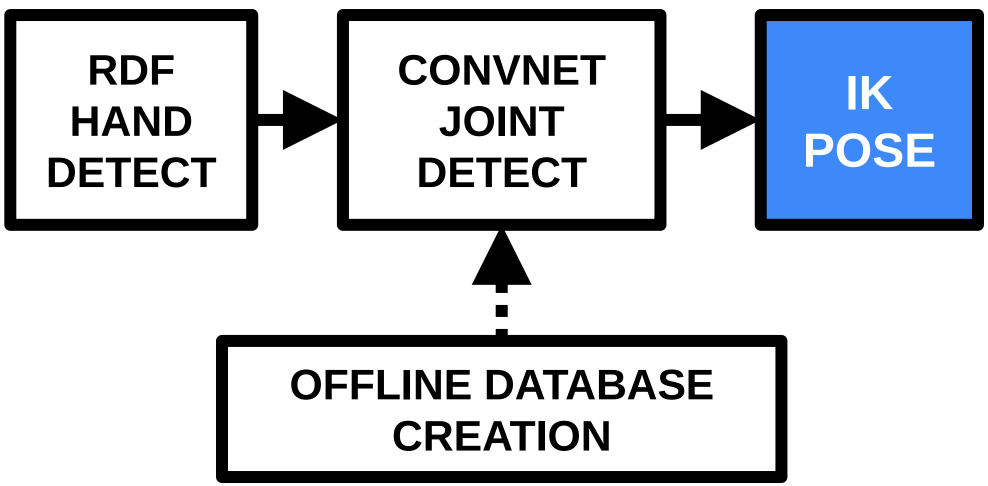


PIPELINE

Supervised learning based approach

- Needs labeled dataset + machine learning
- Existing datasets had limited pose information for hands

Architecture

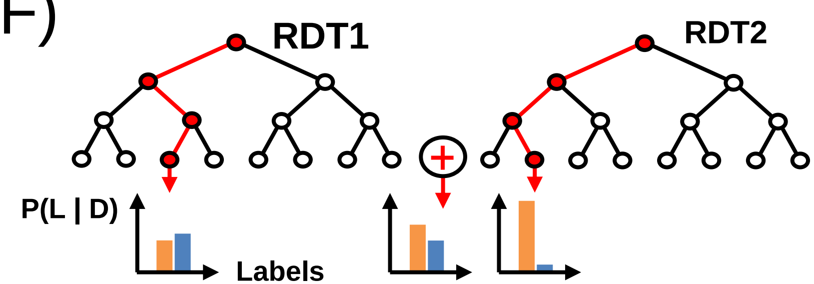


RDF Hand Detection

Per-pixel binary classification → Hand centroid location

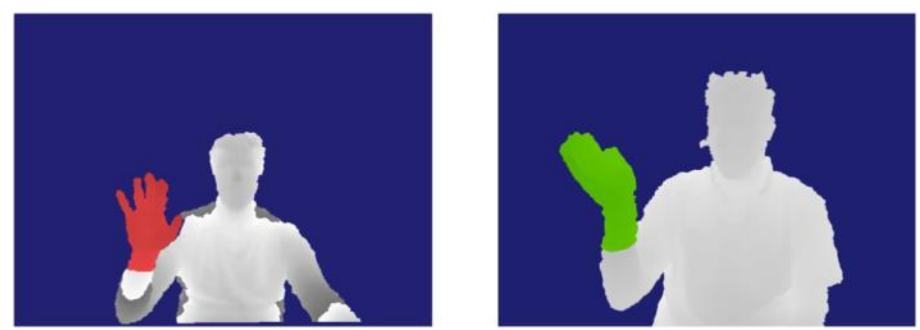
Randomized decision forest (RDF)

- Shotton et al.[1]
- Fast (parallel)
- Generalize
- 12 hour training, depth 25



Dataset

- 7500 images (1000 for test)



[1] J. Shotten et al., Real-time human pose recognition in parts from single depth images, CVPR 11

NYU

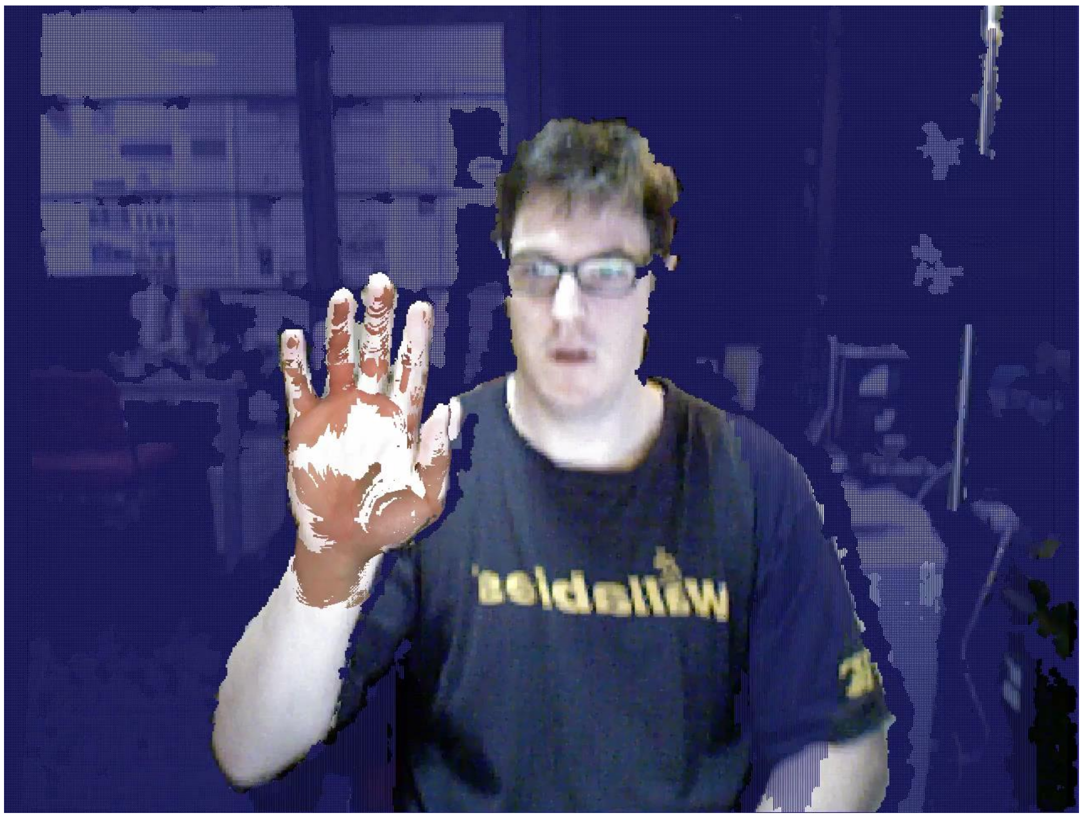
DATASET CREATION

Goal: labeled RGBD images

- $\{\{D_1, \text{pose}_1\}, \{D_2, \text{pose}_2\}, \ldots\}$, pose_i in R^{42}
- Synthetic data doesn't capture device noise

pose_1 pose_2 pose_3

pose_4 pose_5 pose_6



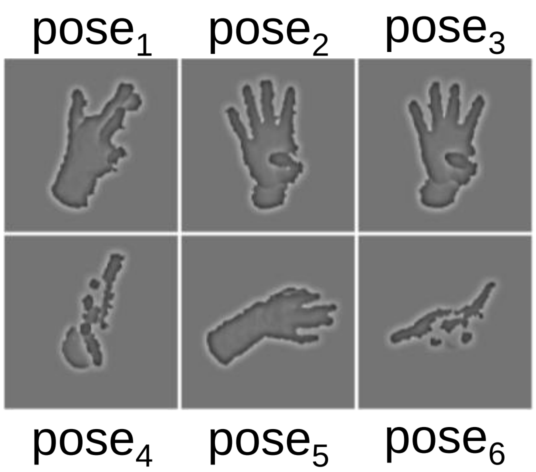
NYU

Dataset Creation

Goal: labeled RGBD images

$\{\{D_1, pose_1\}, \{D_2, pose_2\}, \ldots\}$, $pose_i$ in R^{42}

Synthetic data doesn't capture device noise

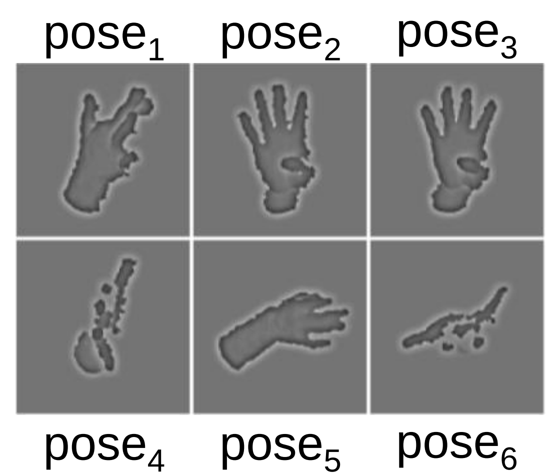


Analysis-by-synthesis Oikonomidis et al.[1]

With many improvements

[1] I. Oikonomidis et al., Efficient model-based 3D tracking of hand articulations using Kinect, BMVC '11

Dataset Creation

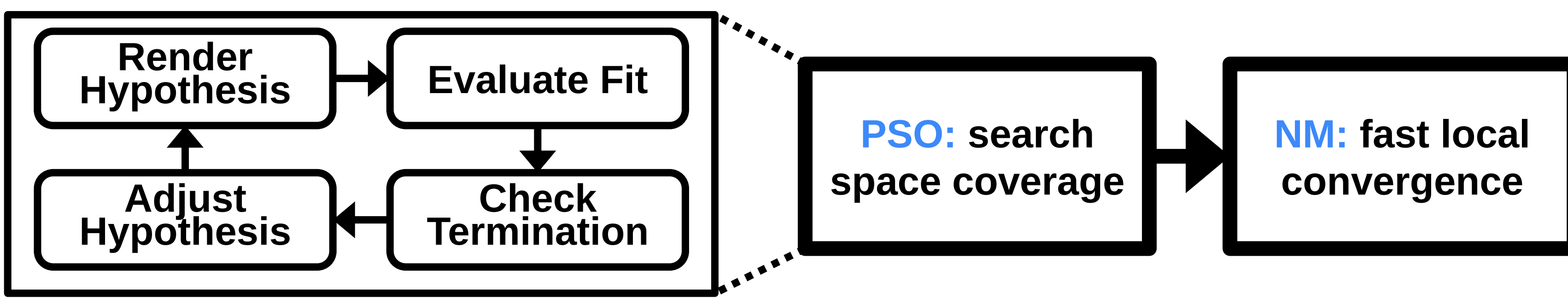


Goal: labeled RGBD images

- $\{\{D_1, pose_1\}, \{D_2, pose_2\}, \ldots\}$, $pose_i$ in R^{42}
- Synthetic data doesn't capture device noise

Analysis-by-synthesis Oikonomidis et al.[1]

- With many improvements



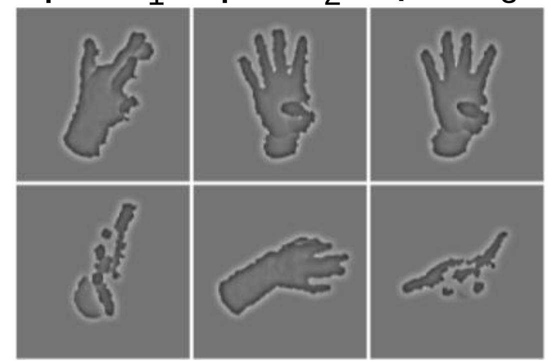
[1] I. Oikonomidis et al., Efficient model-based 3D tracking of hand articulations using Kinect, BMVC '11

Dataset Creation

Goal: labeled RGBD images

- $\{\{D_1, pose_1\}, \{D_2, pose_2\}, \ldots\}$, $pose_i$ in R^{42}
- Synthetic data doesn't capture device noise

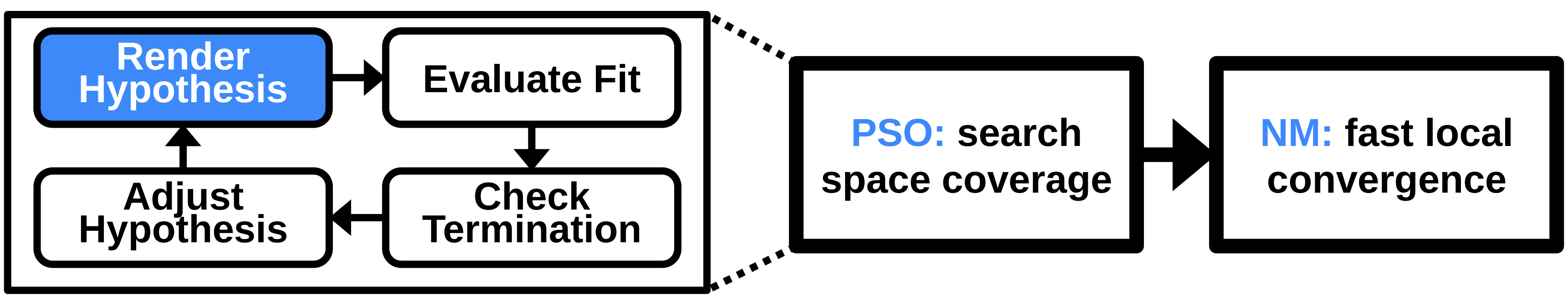
$pose_1$ $pose_2$ $pose_3$



$pose_4$ $pose_5$ $pose_6$

Analysis-by-synthesis Oikonomidis et al.[1]

- With many improvements



[1] I. Oikonomidis et al., Efficient model-based 3D tracking of hand articulations using Kinect, BMVC '11

DATASET CREATION

$pose_1$ $pose_2$ $pose_3$

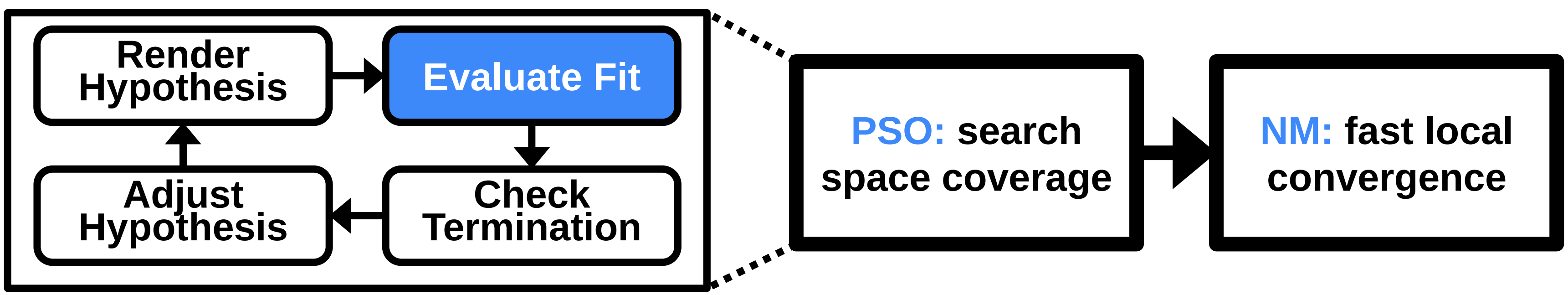
$pose_4$ $pose_5$ $pose_6$

Goal: labeled RGBD images

- $\{\{D_1, pose_1\}, \{D_2, pose_2\}, \ldots\}, pose_i \text{ in } R^{42}$
- Synthetic data doesn't capture device noise

Analysis-by-synthesis Oikonomidis et al.[1]

- With many improvements



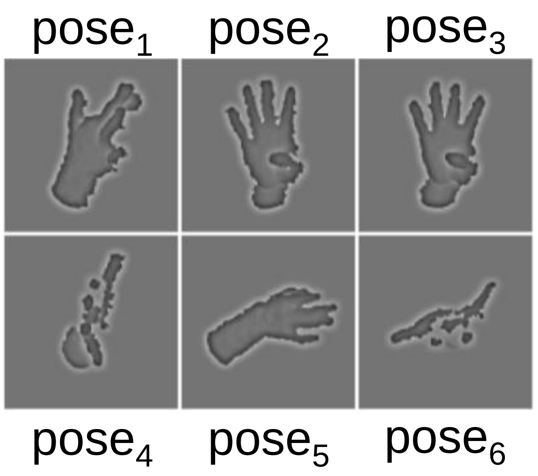
[1] I. Oikonomidis et al., Efficient model-based 3D tracking of hand articulations using Kinect, BMVC '11

DATASET CREATION

Goal: labeled RGBD images

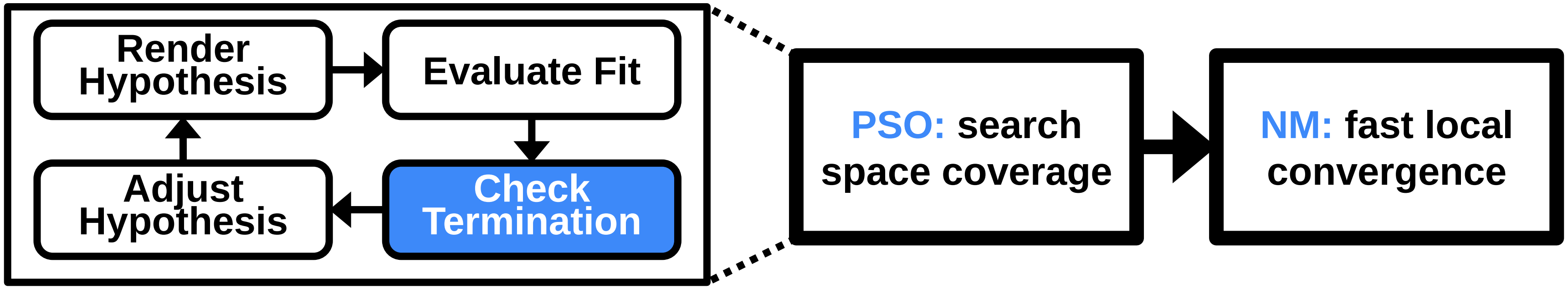
{{D_1, $pose_1$}, {D_2, $pose_2$}, …}, $pose_i$ in R^{42}

Synthetic data doesn't capture device noise



Analysis-by-synthesis Oikonomidis et al.[1]

With many improvements



[1] I. Oikonomidis et al., Efficient model-based 3D tracking of hand articulations using Kinect, BMVC '11

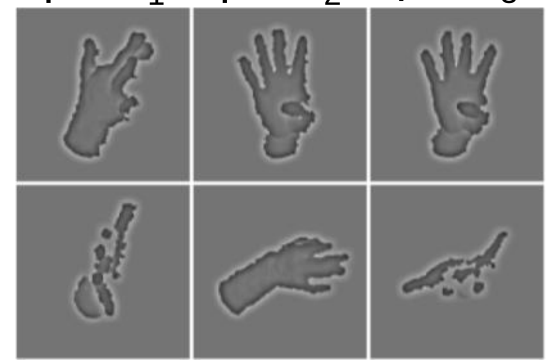
DATASET CREATION

Goal: labeled RGBD images

$\{\{D_1, pose_1\}, \{D_2, pose_2\}, \ldots\}$, $pose_i$ in R^{42}

Synthetic data doesn't capture device noise

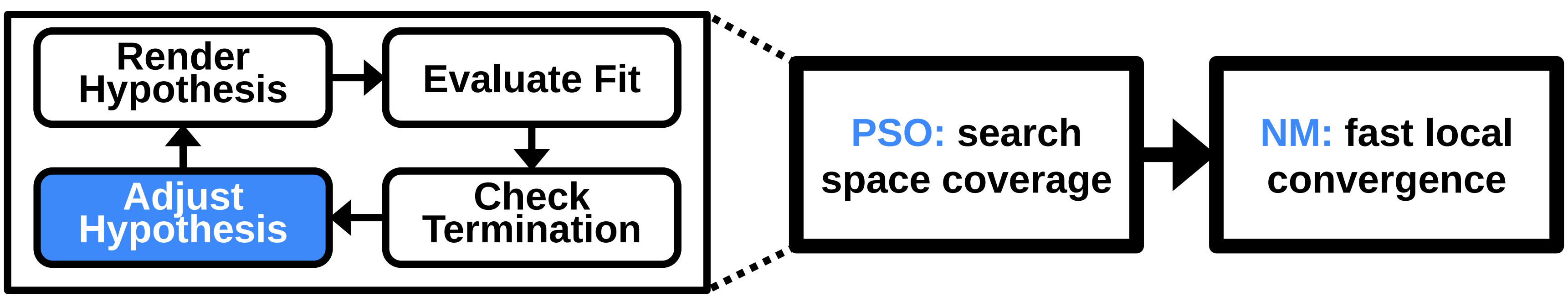
$pose_1$ $pose_2$ $pose_3$



$pose_4$ $pose_5$ $pose_6$

Analysis-by-synthesis Oikonomidis et al.[1]

With many improvements

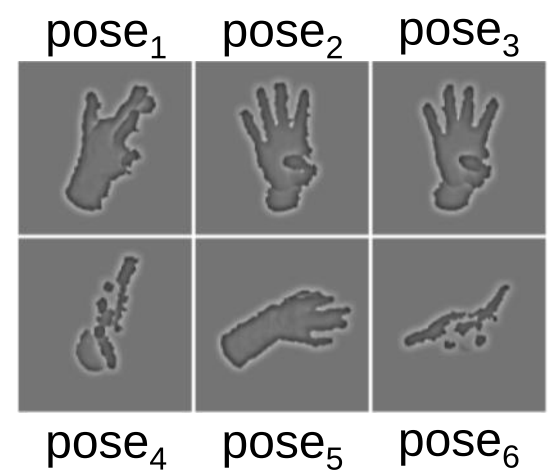


[1] I. Oikonomidis et al., Efficient model-based 3D tracking of hand articulations using Kinect, BMVC '11

Dataset Creation

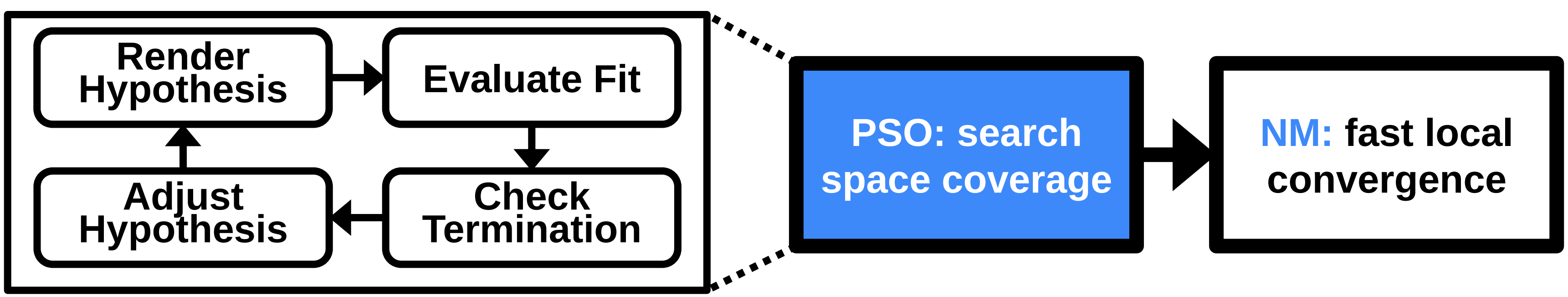
Goal: labeled RGBD images

- $\{\{D_1, pose_1\}, \{D_2, pose_2\}, \ldots\}$, $pose_i$ in R^{42}
- Synthetic data doesn't capture device noise



Analysis-by-synthesis Oikonomidis et al.[1]

- With many improvements



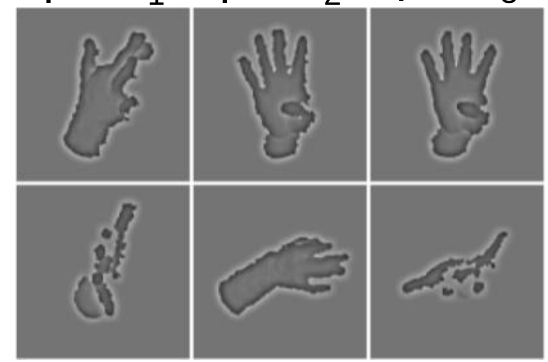
[1] I. Oikonomidis et al., Efficient model-based 3D tracking of hand articulations using Kinect, BMVC '11

DATASET CREATION

Goal: labeled RGBD images

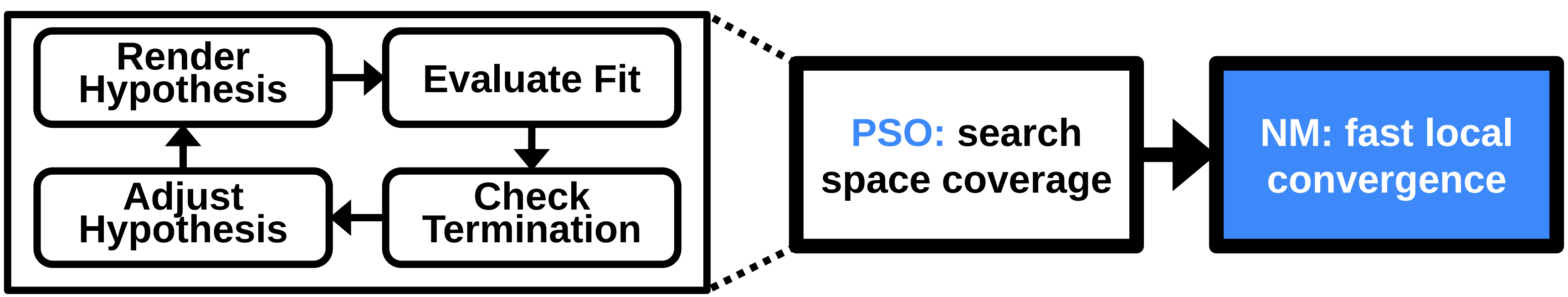
{{D_1, $pose_1$}, {D_2, $pose_2$}, …}, $pose_i$ in R^{42}

Synthetic data doesn't capture device noise



Analysis-by-synthesis Oikonomidis et al.[1]

With many improvements



[1] I. Oikonomidis et al., Efficient model-based 3D tracking of hand articulations using Kinect, BMVC '11

Feature Detection

CN has difficulty learning (U,V) positions directly

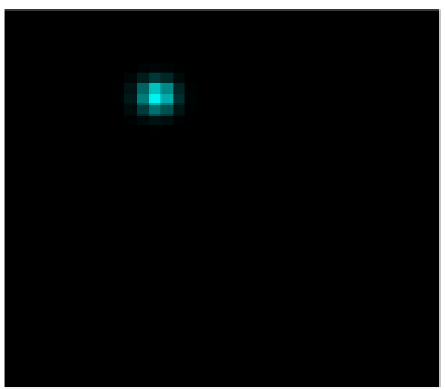
- Require learned integration
- Possible in theory (**never works**)

Recast pose-recognition

- Learn feature distributions

$P_{part1}(x, y)$

y

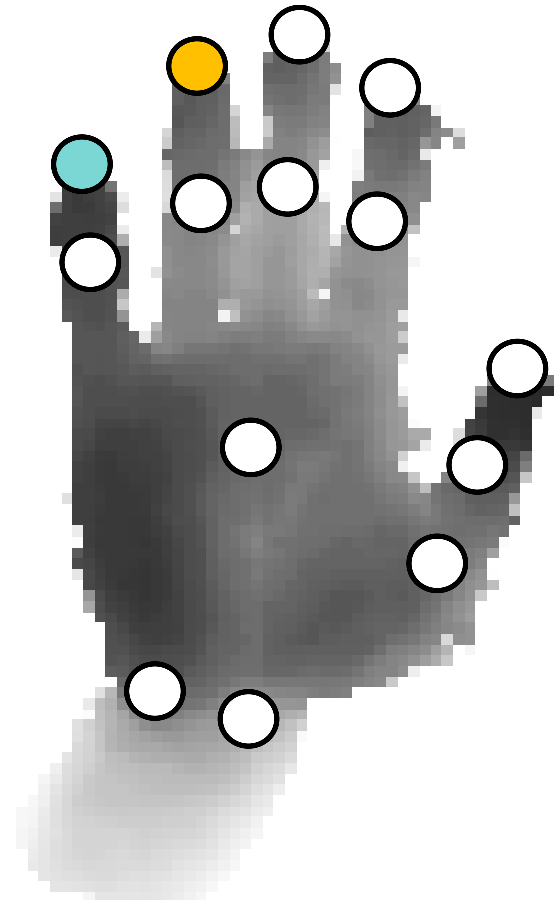


x

$P_{part2}(x, y)$

y

x

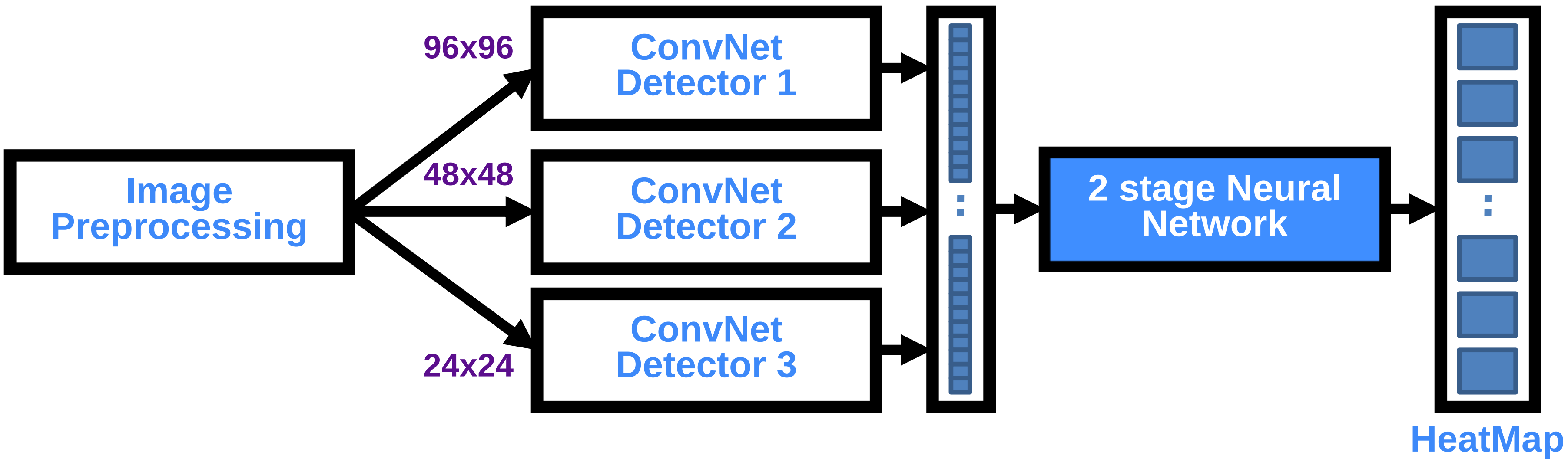


NYU

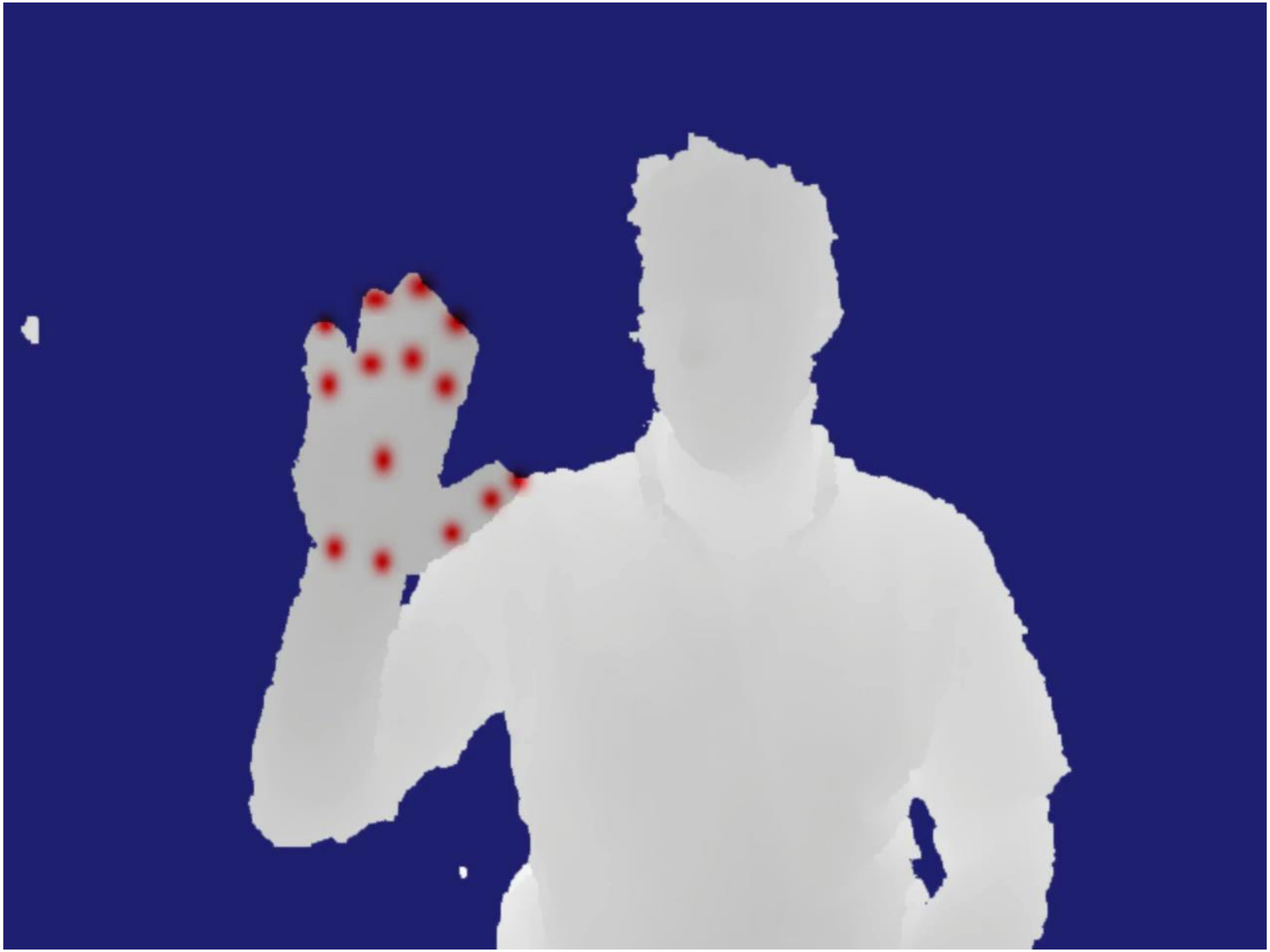
Detection Architecture

Inspired by Farabet et al. (2013)

Multi-resolution convolutional banks



INFERRED JOINT POSITIONS



POSE RECOVERY (IK)

Goal

2D heat-maps + 3D depth → 3D skeletal pose

Solution

A variant of Inverse Kinematics (IK)

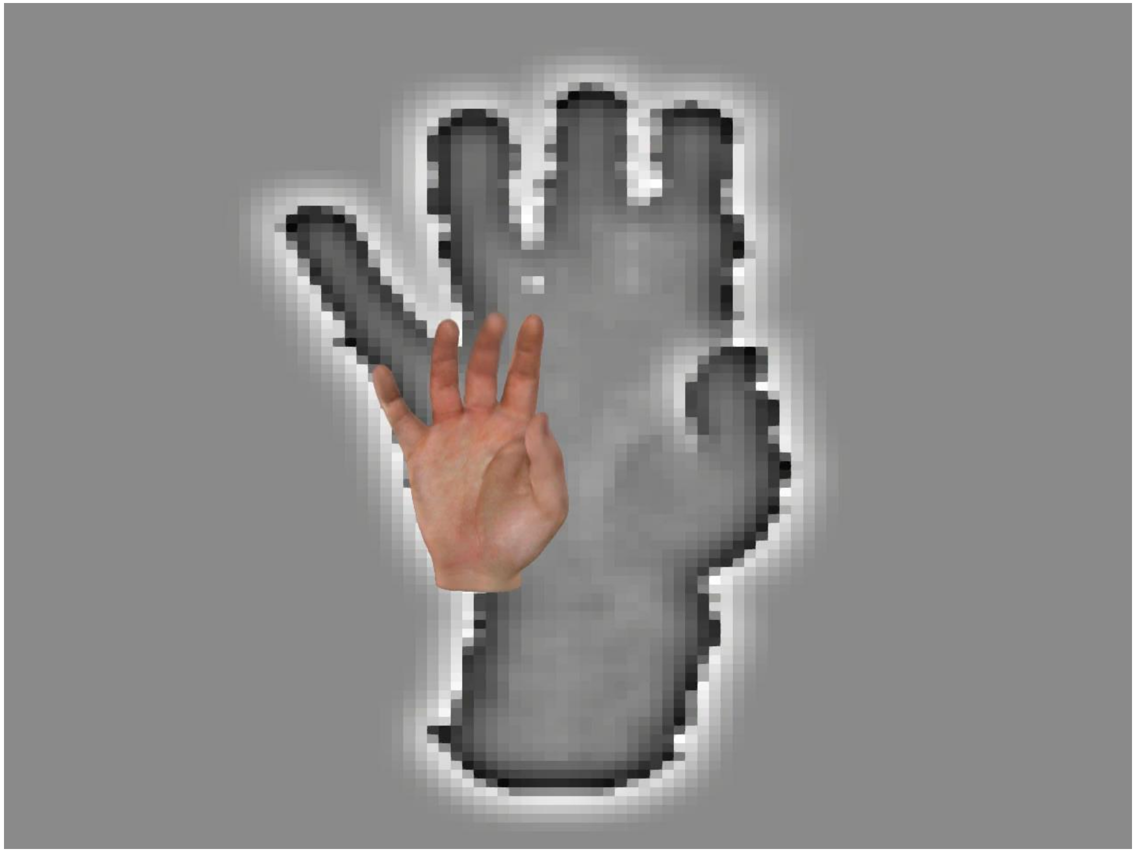
1. Fit a 2D Gaussian to heat-map (Levenberg-Marquardt)
2. Sample depth at Gaussian mean

Fit skeleton at UVD/UV locations using IK

POSE RECOVERY (IK)

Entire Pipeline: 24.9ms

- DF: 3.4ms
- CNN: 5.6ms
- PSO pose: 11.2ms



TALK OUTLINE

1. Hand Tracking
   a) J. Tompson, M. Stein, Y. LeCun, K. Perlin, “Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks”, ACM TOG/SIGGRAPH 2014
2. Body Tracking
   a) J. Tompson, A. Jain, Y. LeCun, C. Bregler, “Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation”, NIPS 2014
   b) A. Jain, J. Tompson, Y. LeCun, C. Bregler, “MoDeep: A Deep Learning Framework Using Motion Features for Human Pose Estimation”, ACCV 2014
   c) J. Tompson, R. Goroshin, A. Jain, Y. LeCun, C. Bregler, “Efficient Object Localization Using Convolution Networks”, CVPR 2015

HUMAN BODY POSE INFERENCE

Motivation:

- From my hand tracking work: do similar techniques work in RGB?
- How about full-body tracking?

Problem definition:

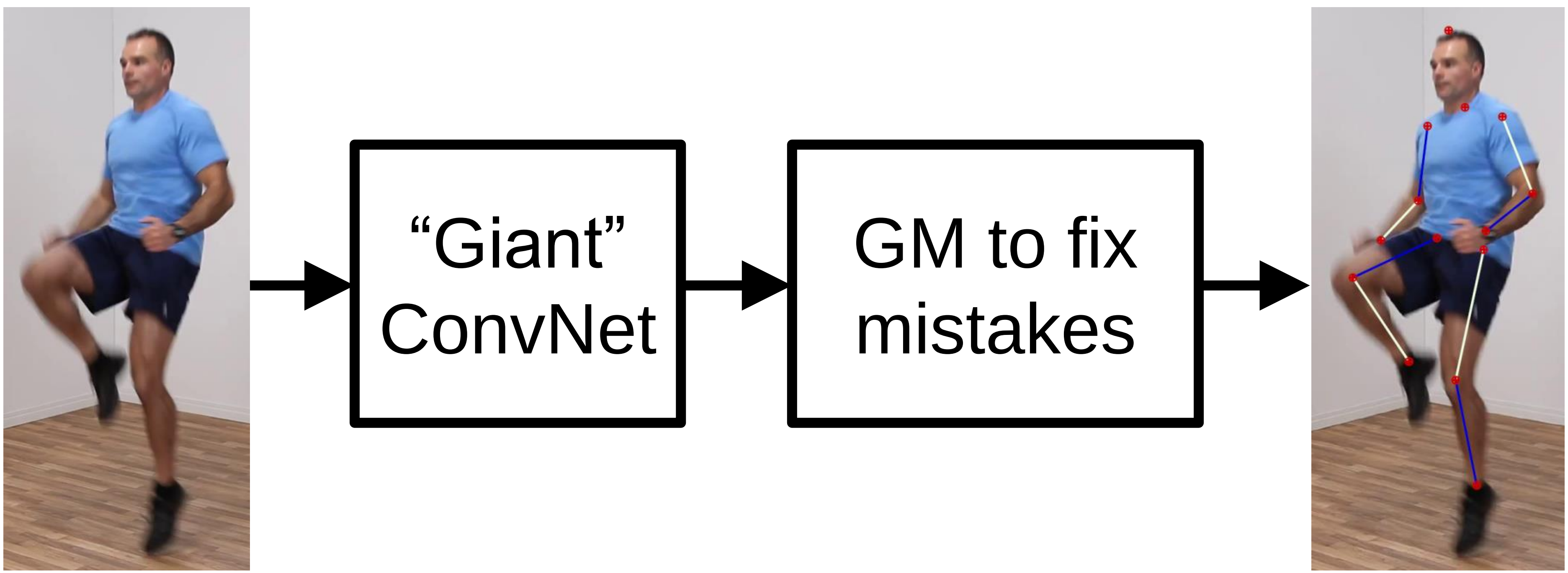
- Track human body joints on a monocular RGB image
- Arbitrary pose and background

My Basic Idea

Two Parts:

- ConvNet to track joints
- Graphical Model to stitch them together

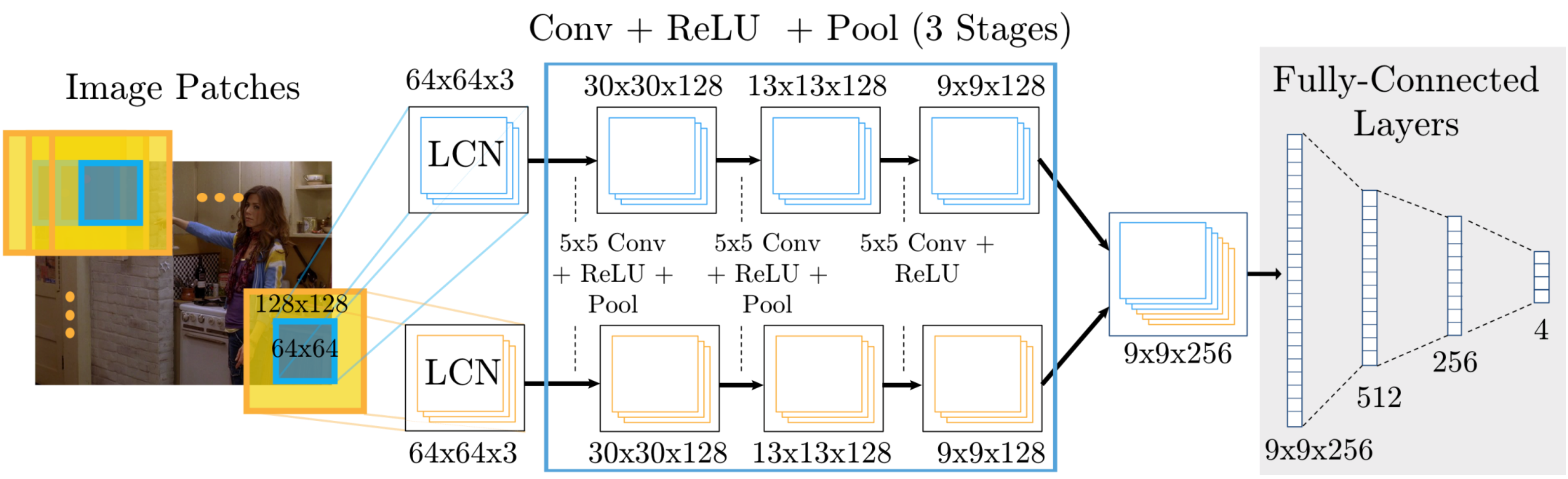
JOINTLY TRAIN THEM!



PART DETECTOR

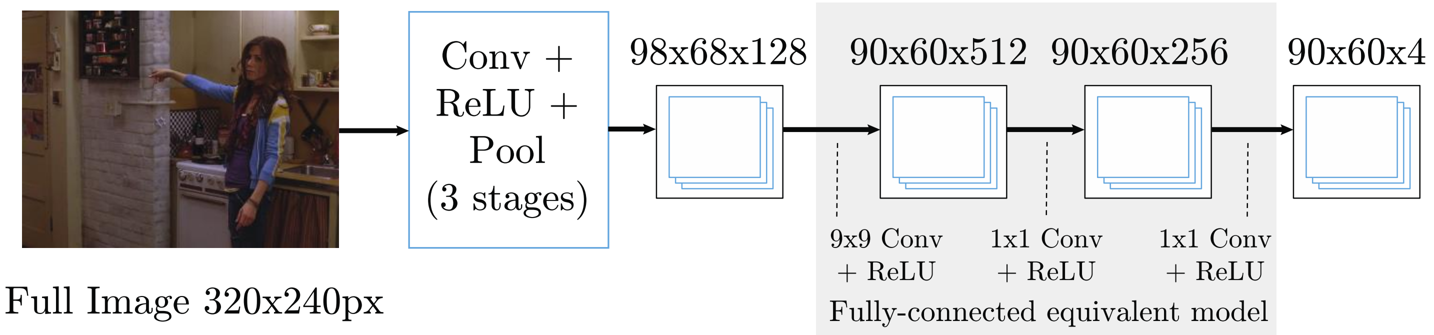
ConvNet Architecture:

Multi-resolution sliding window with overlapping receptive fields

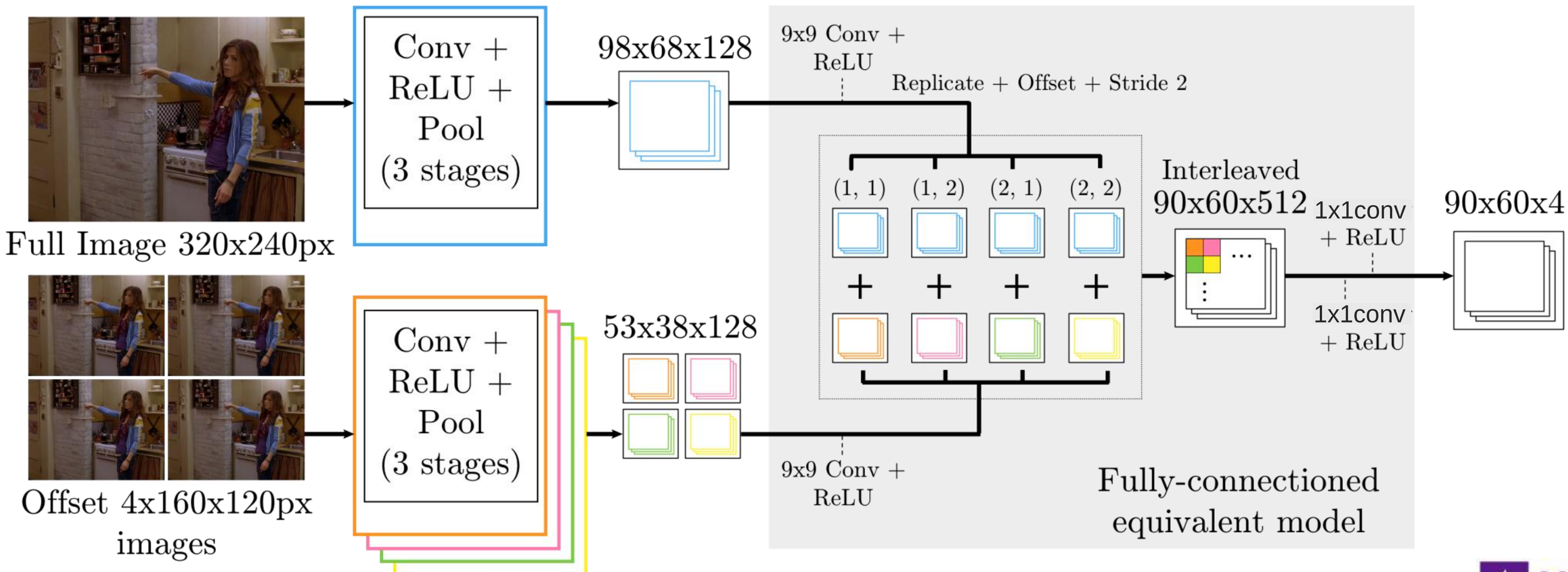


Part Detector

Efficient model (P. Sermanet and others):



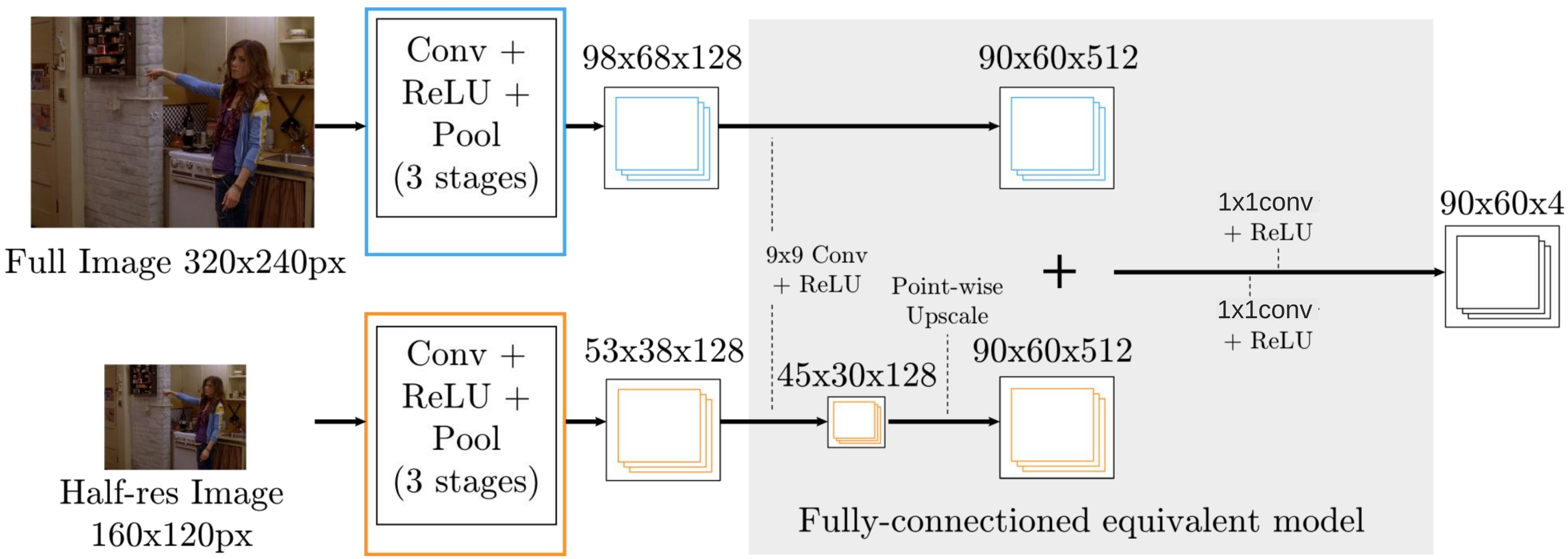
Multi-resolution efficient model (my work):



PART DETECTOR

Simplified model:

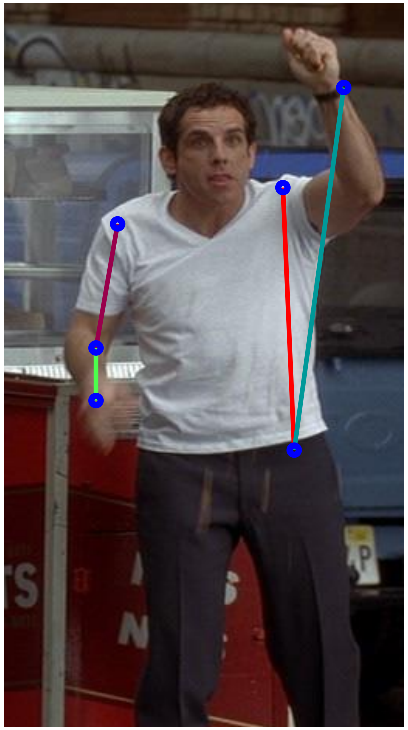
- Performance is close to the "full" model
- Use this for real-time demo



Part Detector Results

What's wrong so far

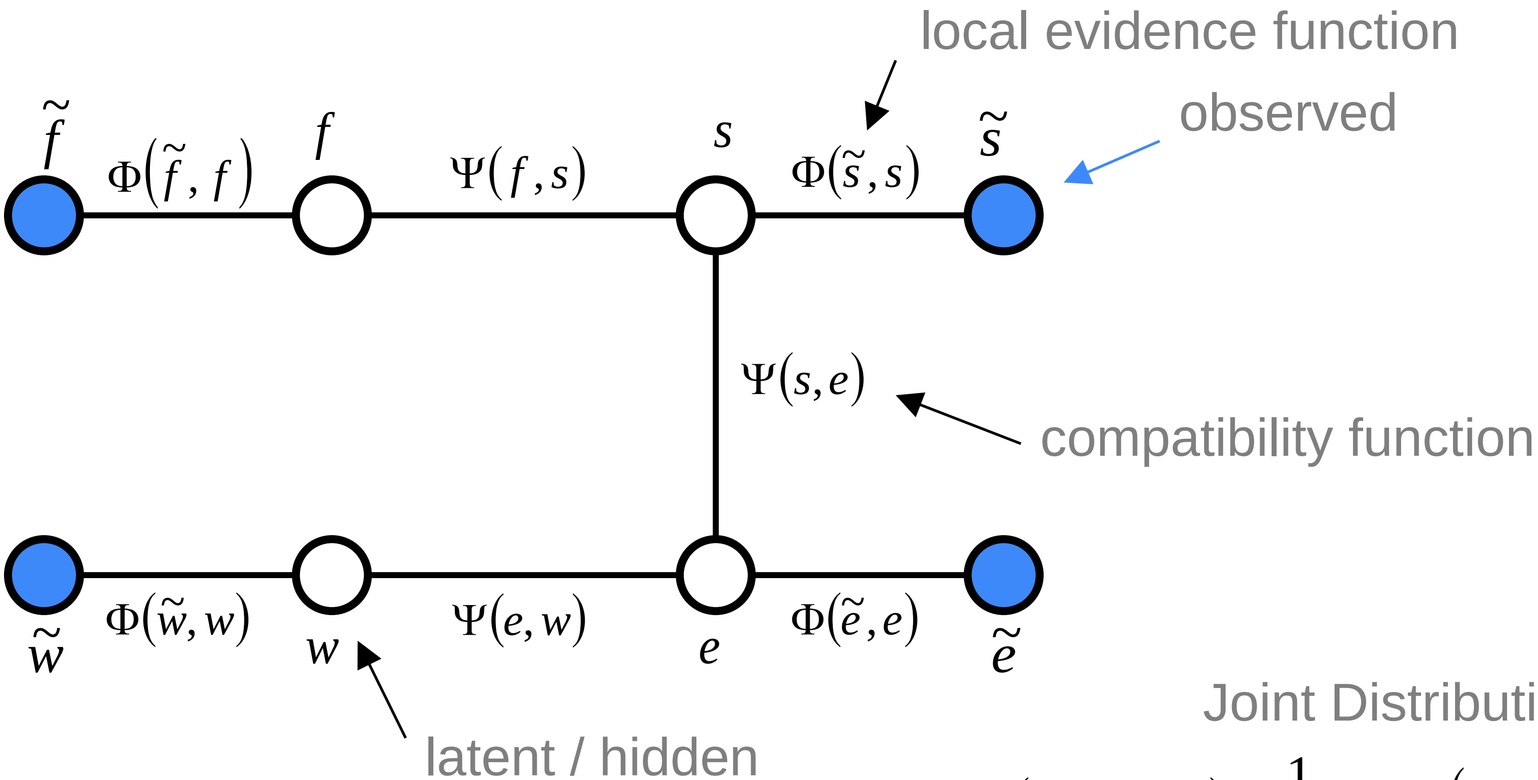
- Independent joint terms in objective function
- We're hoping the network implicitly learns joint consistency
- Failure cases are usually stupid:



Spatial Model

Start with my GM from ICLR 2013 paper

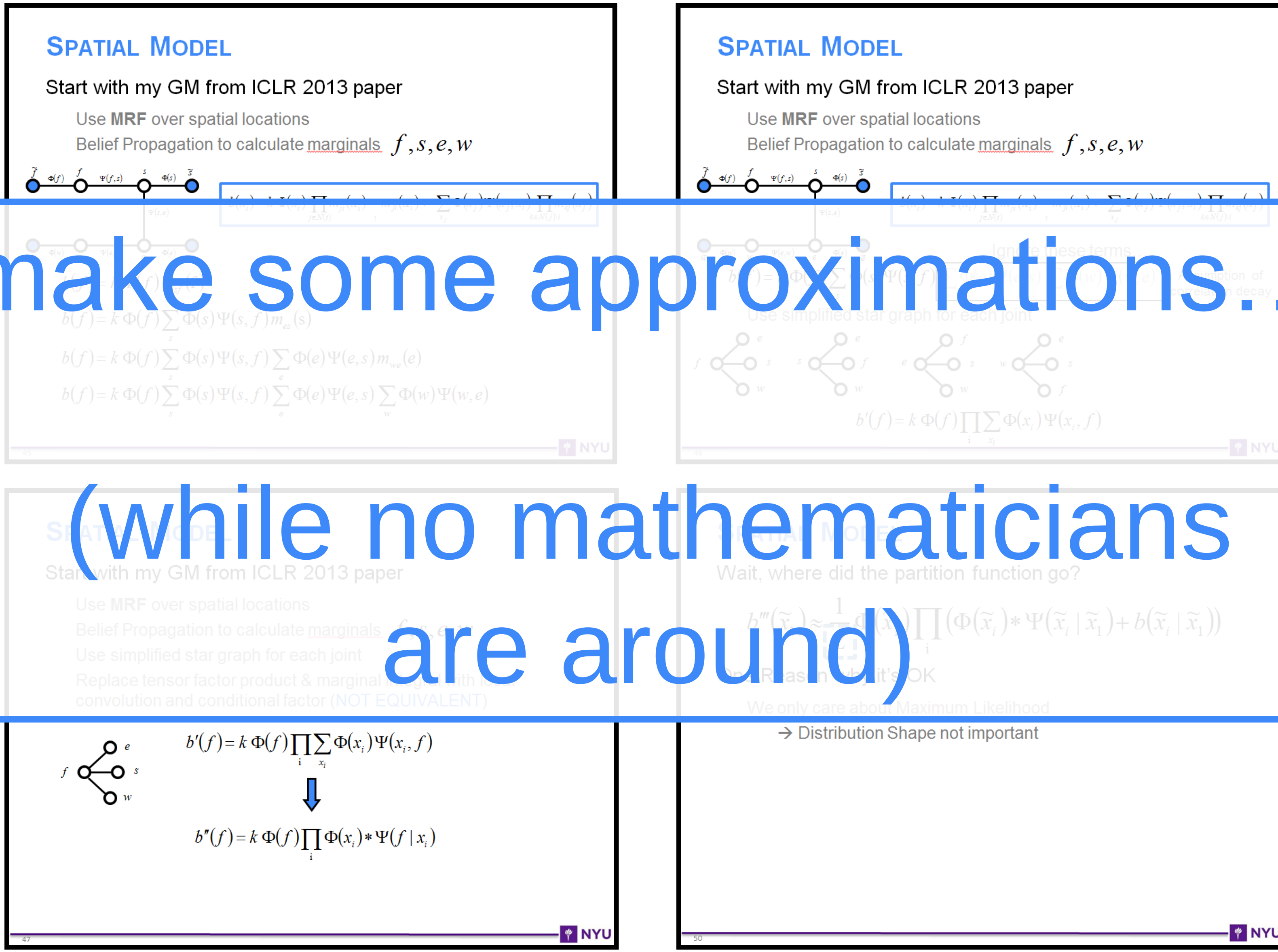
MRF over spatial locations



Joint Distribution:

$$P(f, s, e, w) = \frac{1}{Z} \prod_{i,j} \Psi(x_i, x_j) \prod_i \Phi(x_i, \tilde{x}_i)$$

SPATIAL MODEL



make some approximations…

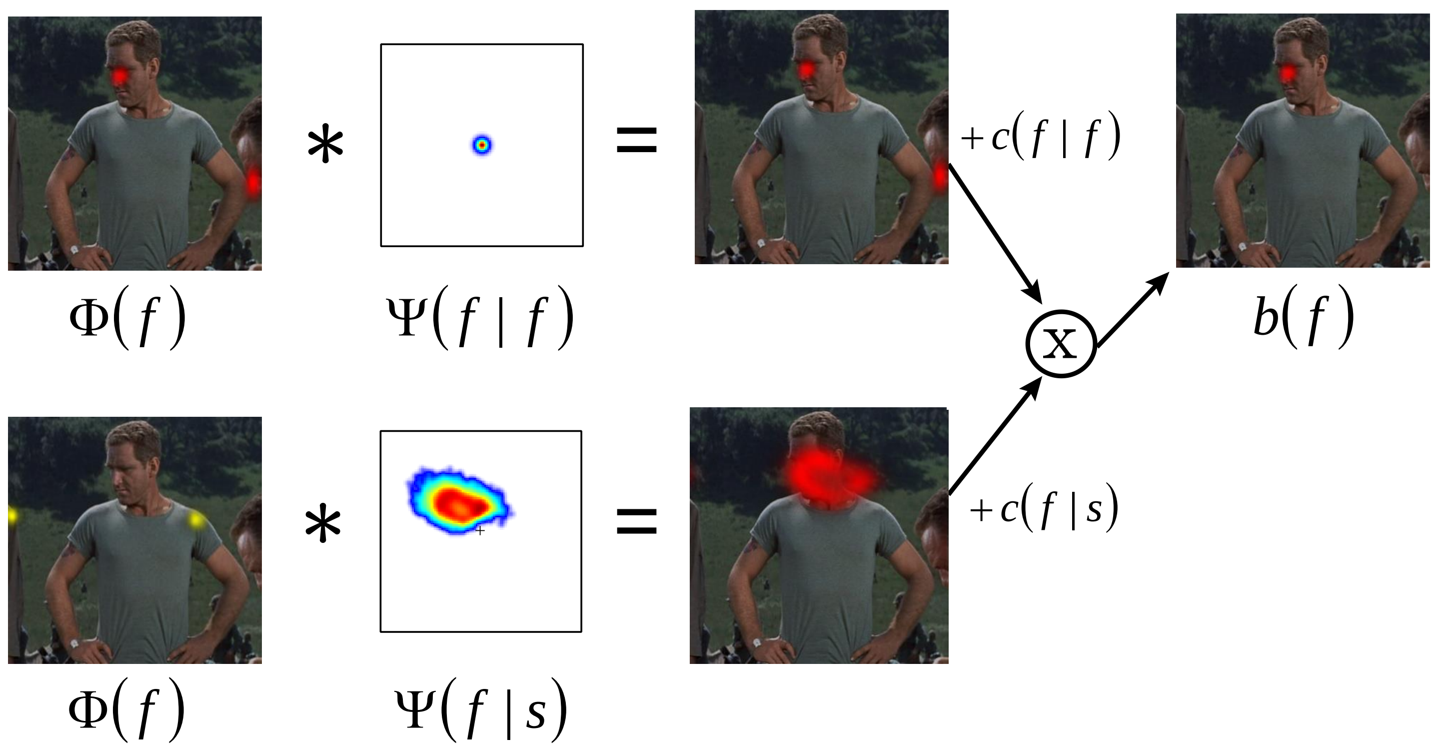
(while no mathematicians are around)

SPATIAL MODEL

Start with my GM from ICLR 2013 paper

… And approximate it

$$b(f)=\Phi(f)\prod_{\mathrm{i}}\left(\Phi(x_i)*\Psi(f\mid x_i)+c(f\mid x_i)\right)$$

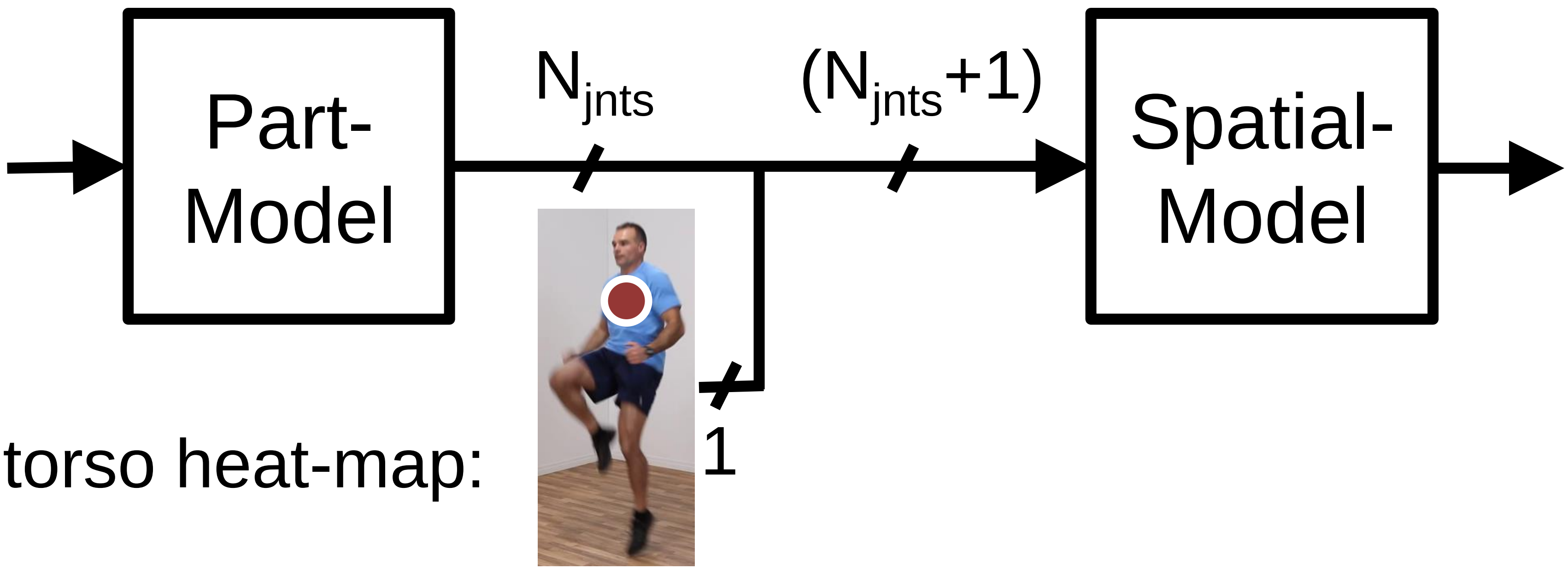


SPATIAL MODEL

Two additional details

1. Spatial model kernel size is 128x128! → Have to use FFT[1]
2. For standard datasets → add (noisy) torso location

torso heat-map:



[1] M. Mathieu, M. Henaff, and Y. LeCun. Fast training of convolutional networks through FFTs. 2013.

JOINT TRAINING

Joint training

- Pre-train both models separately
- Joint train (BPROP) through both models

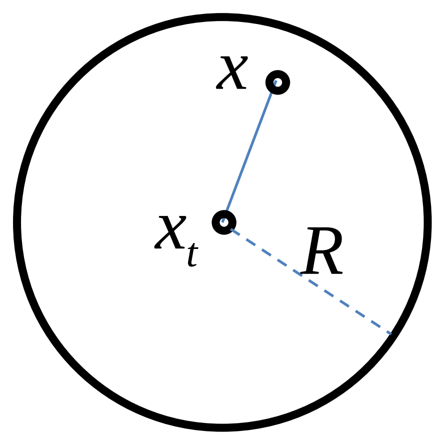
JOINT TRAINING

Joint training

- Pre-train both models separately
- Joint train (BPROP) through both models

Performance (Wrist):

$$\mathrm{DetectionRate}\,(R) = \frac{100}{N}\sum_{t=1}^{N}\left(\frac{\|x \;\; - x^{t}\|_2}{(\text{torso height t})\,/100} \leq R\right)$$



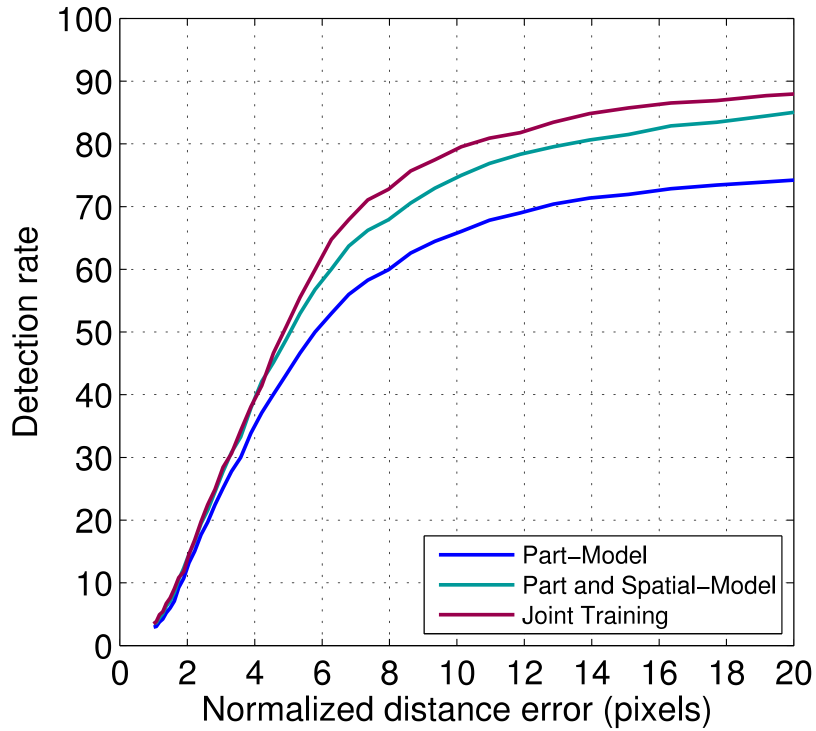
JOINT TRAINING

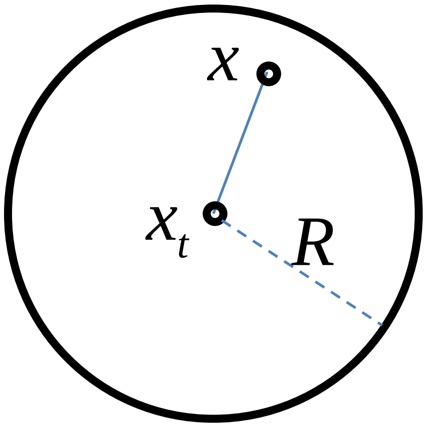
Joint training

- Pre-train both models separately
- Joint train (BPROP) through both models

Performance (Wrist):

$$\mathrm{DetectionRate}\,(R) = \frac{100}{N}\sum_{t=1}^{N}\left(\frac{\|x \;\; - x^t\|_2}{(\text{torso height t})\,/100} \leq R\right)$$

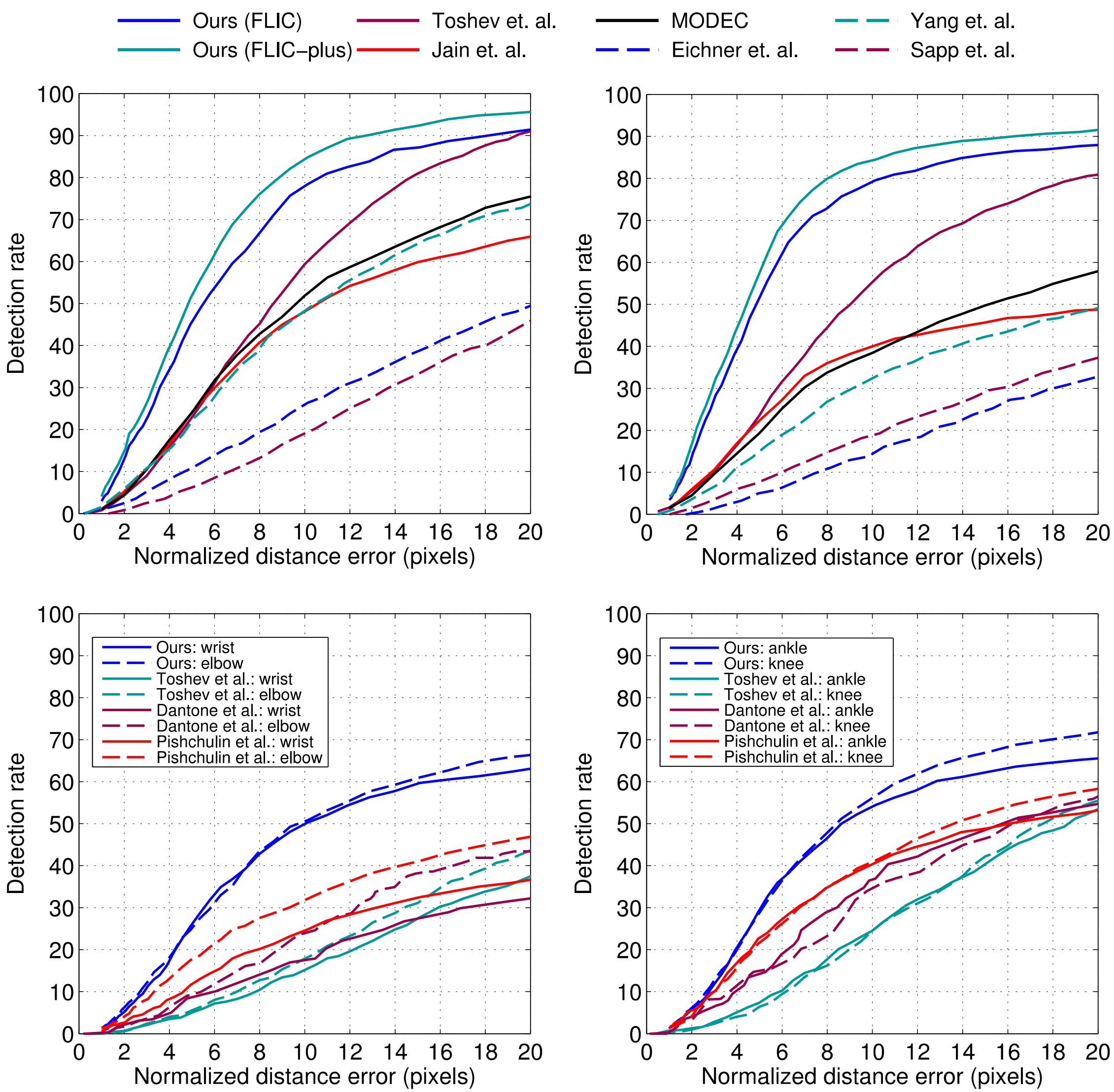




RESULTS

FLIC(1) Elbow

FLIC(1) Wrist



LSP(2) Arms

LSP(1) Legs

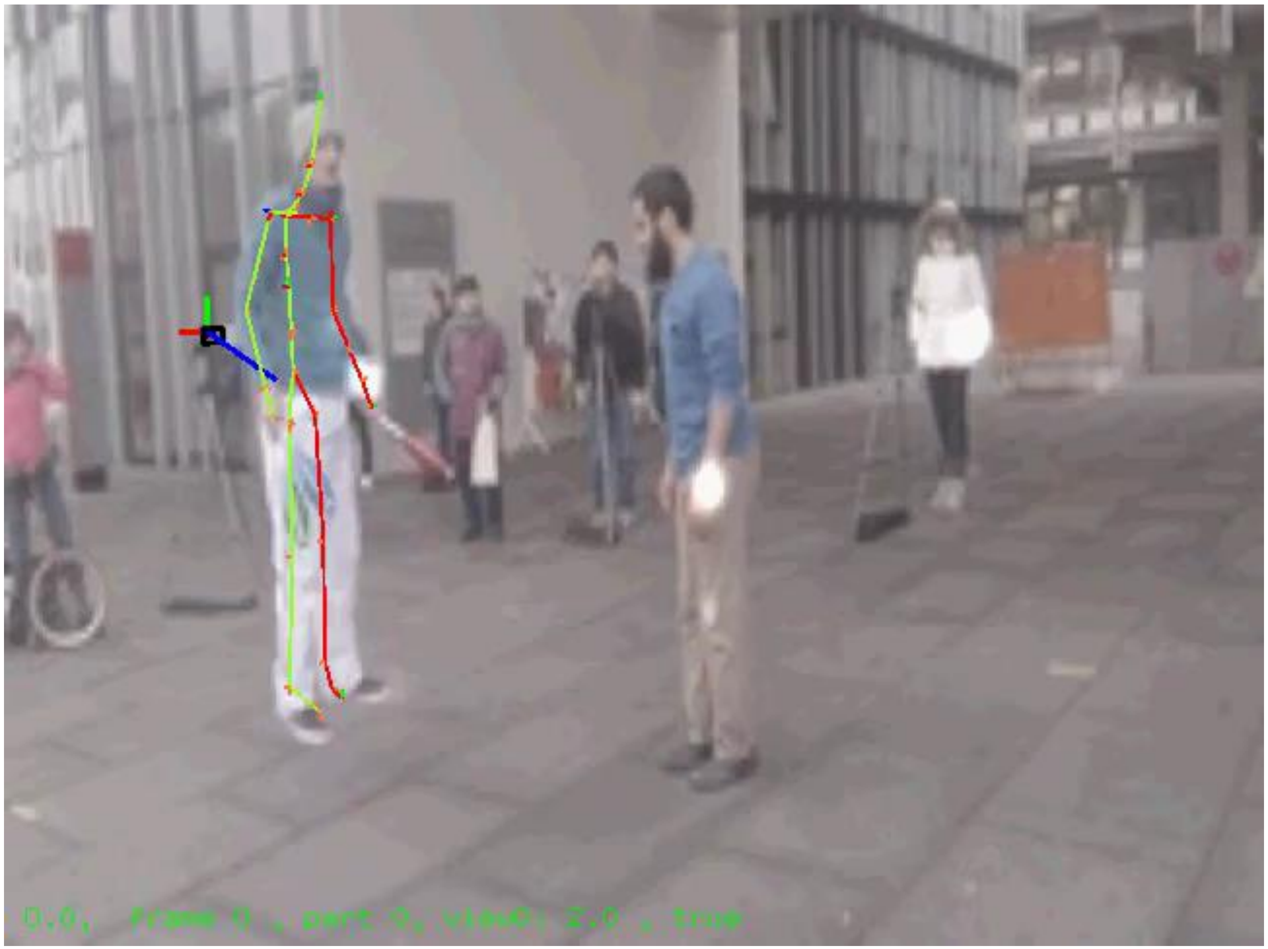
(1) B. Sapp and B. Taskar. MODEC: Multimodel decomposition models for human pose estimation. CVPR’13

(2) S. Johnson and M. Everingham. Learning Effective Human Pose Estimation for Inaccurate Annotation. CVPR’11

RESULTS

Joint work with MPII

Use our detector + analysis by synthesis technique



RESULTS

Joint work with MPII

- Use our detector + analysis by synthesis technique

TALK OUTLINE

1. Hand Tracking
   a) J. Tompson, M. Stein, Y. LeCun, K. Perlin, “Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks”, ACM TOG/SIGGRAPH 2014
2. Body Tracking
   a) J. Tompson, A. Jain, Y. LeCun, C. Bregler, “Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation”, NIPS 2014
   b) A. Jain, J. Tompson, Y. LeCun, C. Bregler, “MoDeep: A Deep Learning Framework Using Motion Features for Human Pose Estimation”, ACCV 2014
   c) J. Tompson, R. Goroshin, A. Jain, Y. LeCun, C. Bregler, “Efficient Object Localization Using Convolution Networks”, CVPR 2015

INCORPORATING TEMPORAL FEATURES

Single frame detection is overly constrained

- Ambiguities can be resolved using multiple frames

Our ACCV Paper - A short investigation:

- How to use temporal features with a ConvNet detector
- Performance is improved using motion features

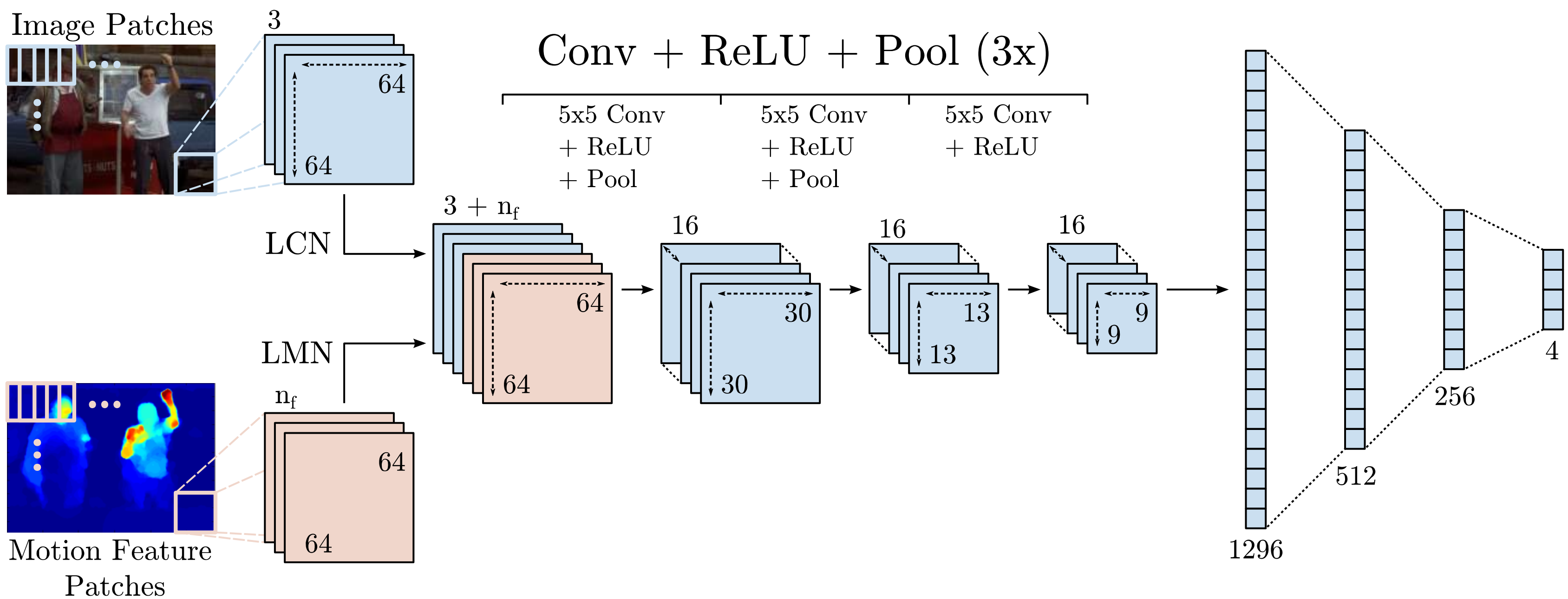
motion features

baseline

Motion Features

Brainless:

Just concat some form of "motion feature" with the RGB



Question:

What features should we use?

Motion Features

Try many and see what works:

RGB image pair	$\{f_i, f_{i+\delta}\}$
RGB difference	$\{f_i, f_{i+\delta} - f_i\}$
RGB + 2D Optical Flow	$\{f_i, \mathrm{FLOW}(f_{i+\delta}, f_i)\}$
RGB + 2D Optical Flow Mag	$\{f_i, \lVert\mathrm{FLOW}(f_{i+\delta}, f_i)\rVert_2\}$

... etc

Motion Features

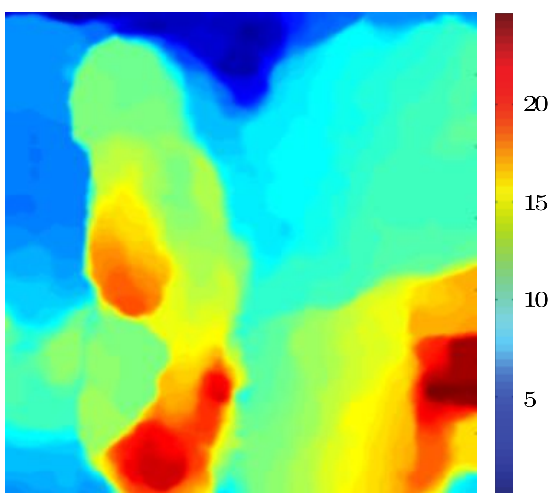
Try many and see what works:

RGB image pair	$\{f_i, f_{i+\delta}\}$
RGB difference	$\{f_i, f_{i+\delta} - f_i\}$
RGB + 2D Optical Flow	$\{f_i, \mathrm{FLOW}(f_{i+\delta}, f_i)\}$
RGB + 2D Optical Flow Mag	$\{f_i, \lVert \mathrm{FLOW}(f_{i+\delta}, f_i) \rVert_2\}$

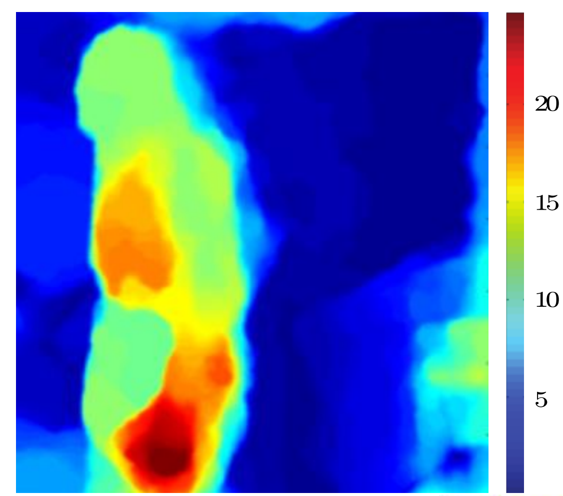
Annoying detail:

SIFT + RANSAC + Bundle adjustment → planar camera motion
Then subtract it out:

Optical flow mag



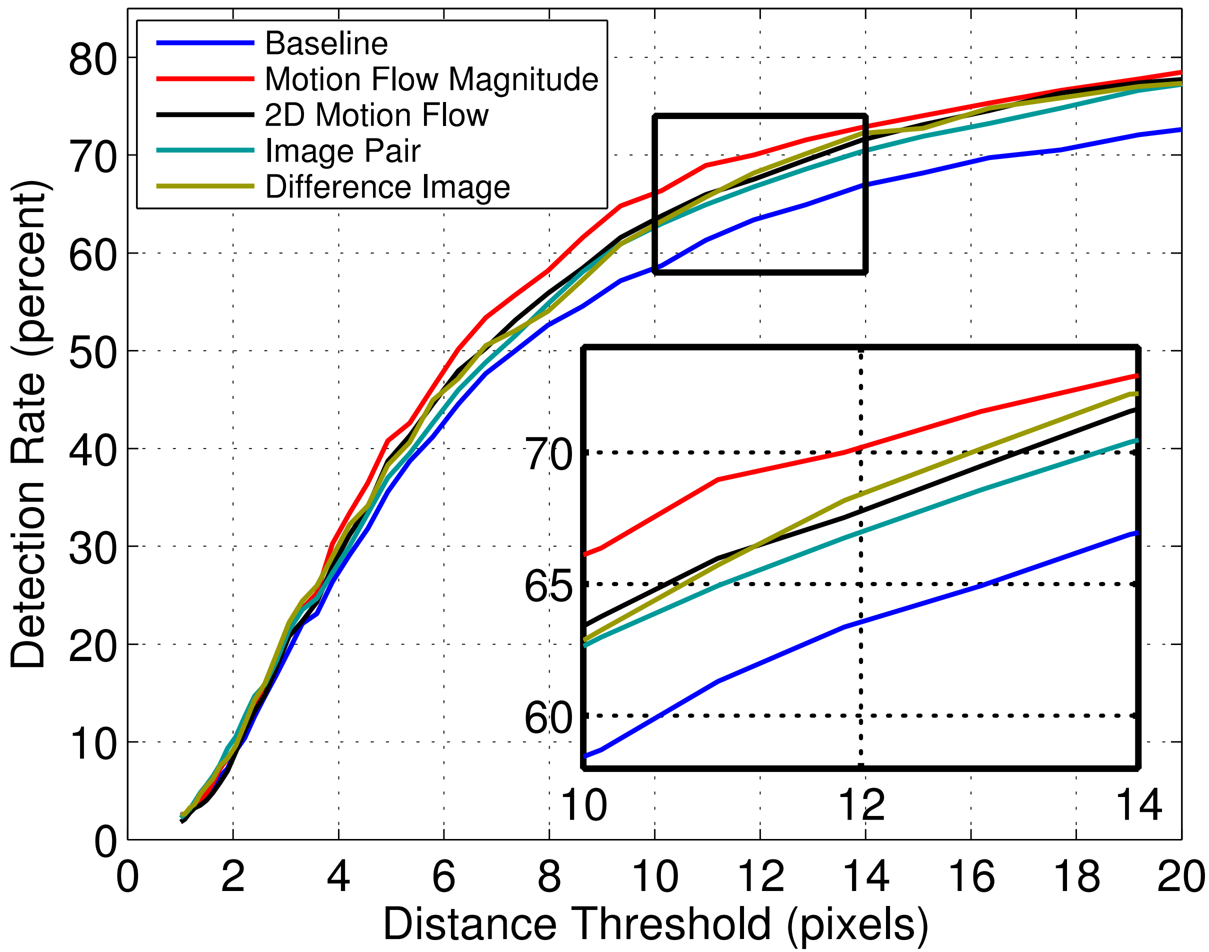
Optical flow mag w.o. camera



Motion Features: Results

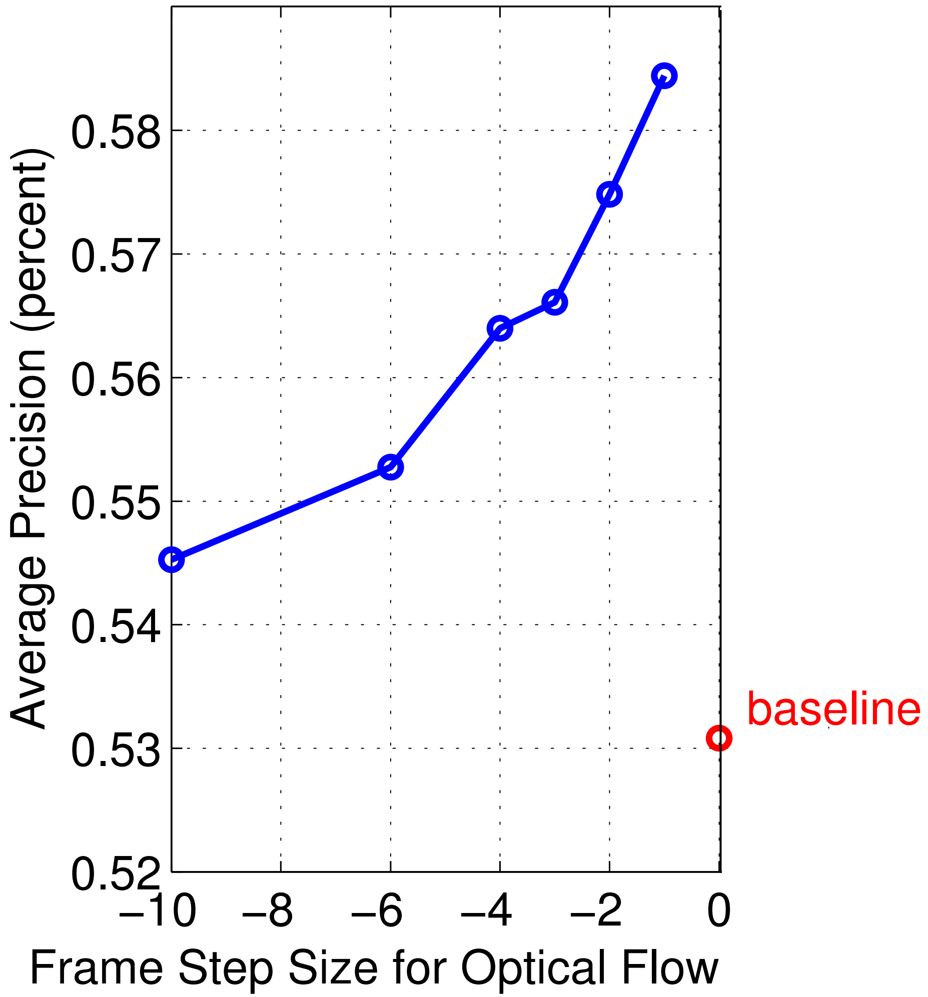
Unsurprising… Optical flow is the best

Magnitude prevents overtraining on motion direction



MOTION FEATURES

How sensitive is it to time?



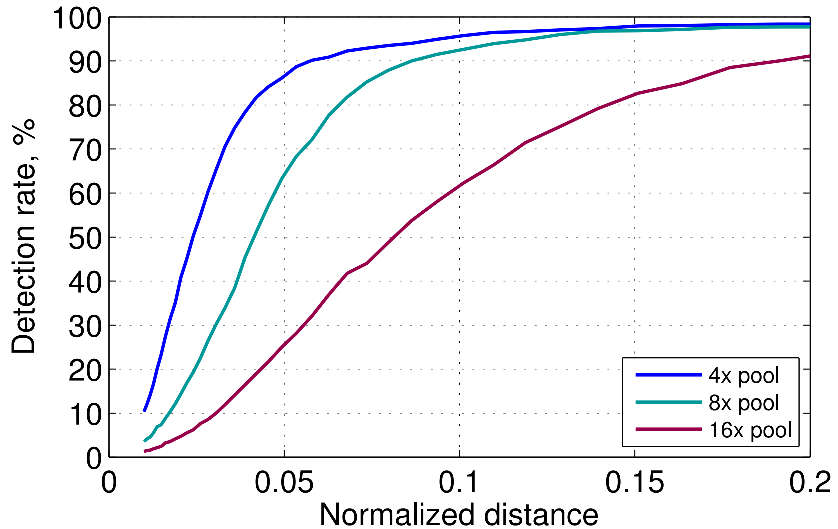
TALK OUTLINE

1. Hand Tracking
    a) J. Tompson, M. Stein, Y. LeCun, K. Perlin, “Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks”, ACM TOG/SIGGRAPH 2014
2. Body Tracking
    a) J. Tompson, A. Jain, Y. LeCun, C. Bregler, “Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation”, NIPS 2014
    b) A. Jain, J. Tompson, Y. LeCun, C. Bregler, “MoDeep: A Deep Learning Framework Using Motion Features for Human Pose Estimation”, ACCV 2014
    c) J. Tompson, R. Goroshin, A. Jain, Y. LeCun, C. Bregler, “Efficient Object Localization Using Convolution Networks”, CVPR 2015

MOTIVATION

We use pooling

- Too much: bad spatial accuracy



- Too little: Over-trains, **<u>slow</u>**, not enough context

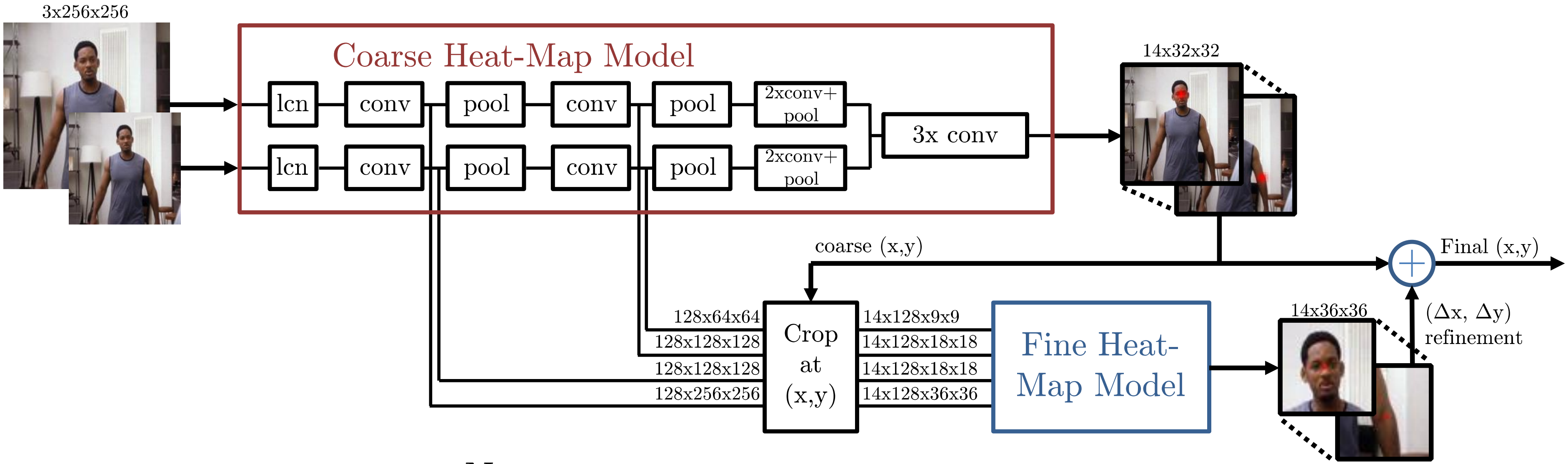
Idea:

- Use lots of pooling but recover spatial accuracy
- A “smart” cascade (highly engineered)

High Level Architecture

Highly “engineered”

Coarse to Fine architecture with shared features



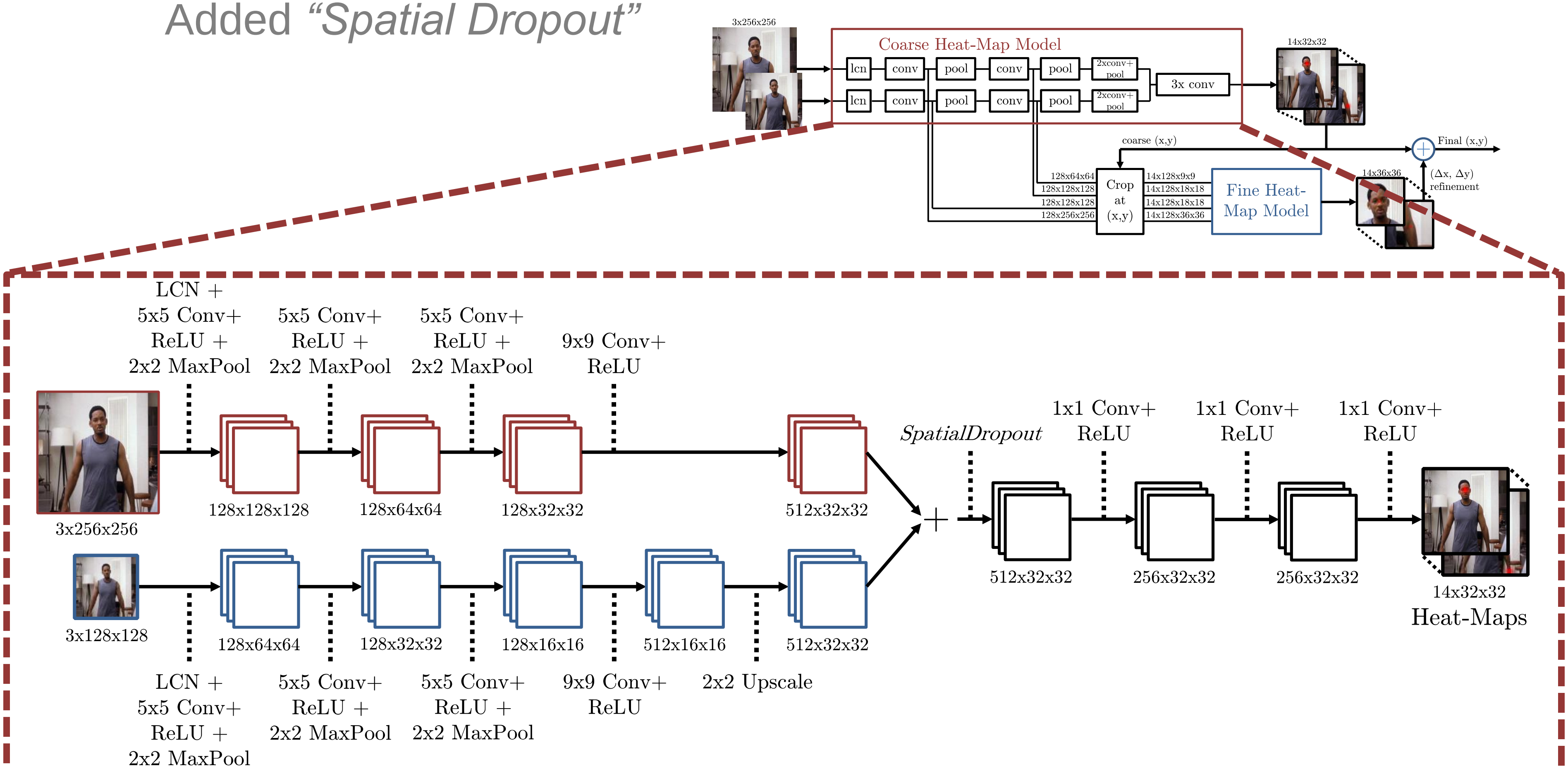
$$E_1 = \frac{1}{N} \sum_{j=1}^{N} \sum_{xy} \left\| H'_j(x,y) - H_j(x,y) \right\|^2$$

$$E_2 = E_1 + \lambda \frac{1}{N} \sum_{j=1}^{N} \sum_{x,y} \left\| G'_j(x,y) - G_j(x,y) \right\|^2$$

Coarse Model

Straight from the NIPS paper

- Many more features (more engineering)
- Added *"Spatial Dropout"*



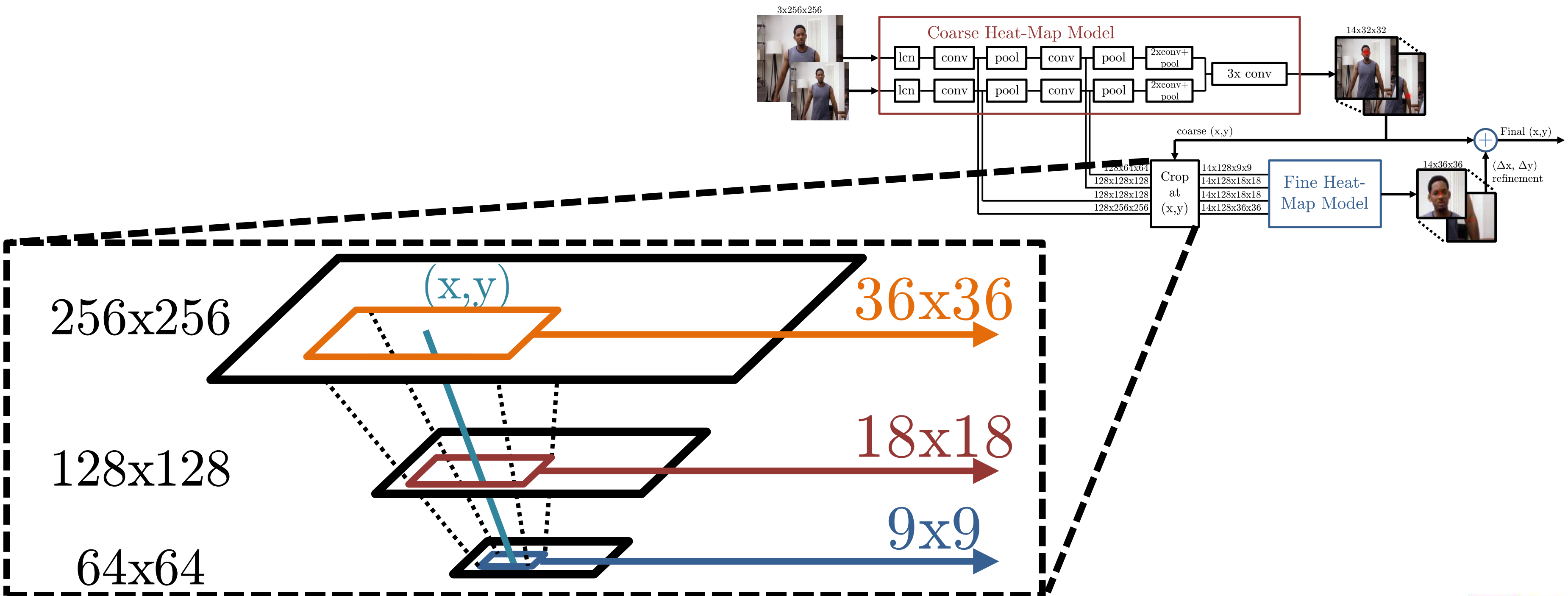
CASCADE CROP

What do we want?

- To refine the coarse heat-map by cropping around the approx. UV

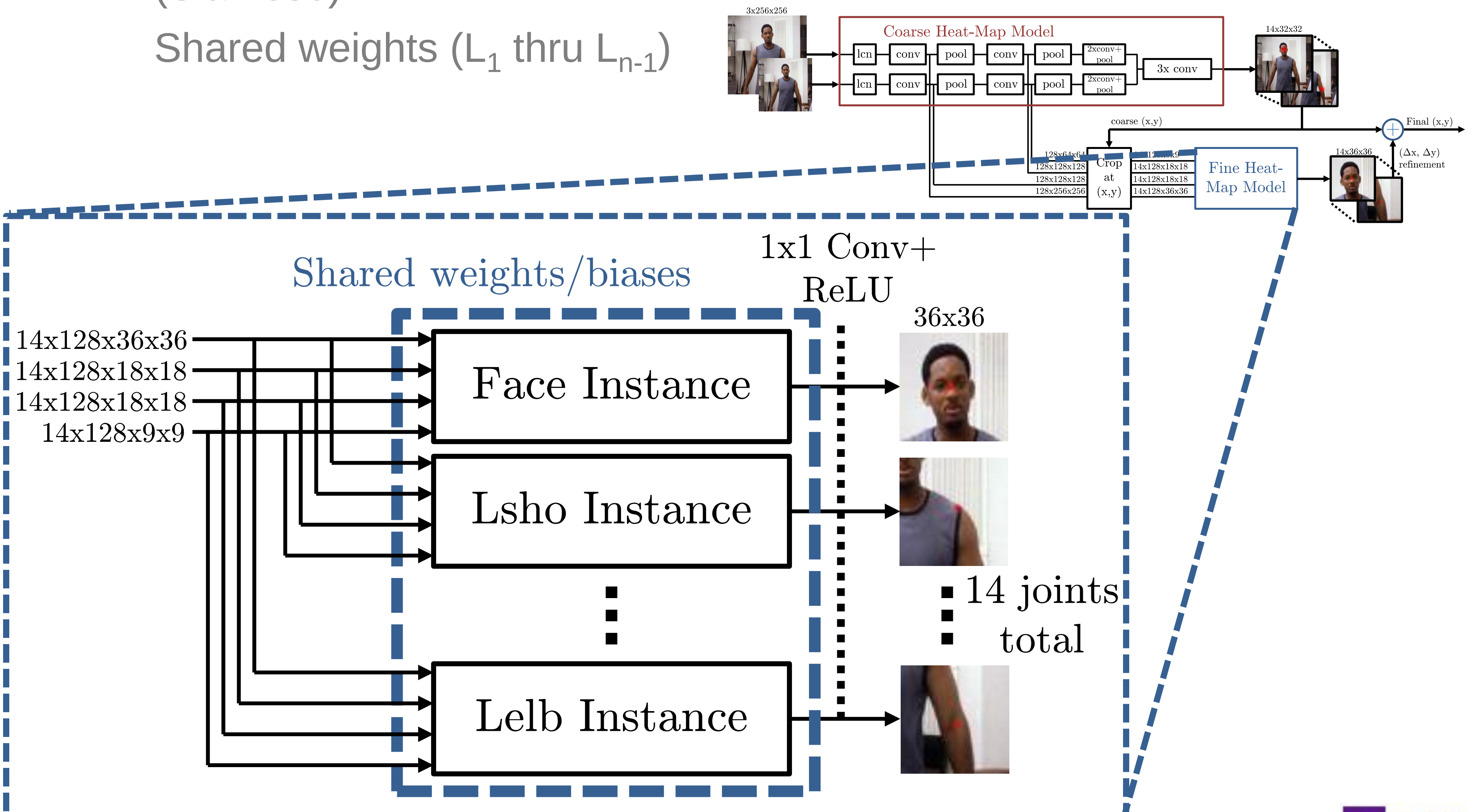
What features do we need?

- Early layers that haven't specialized
- Consistent Spatial Context from all layers



Fine Model

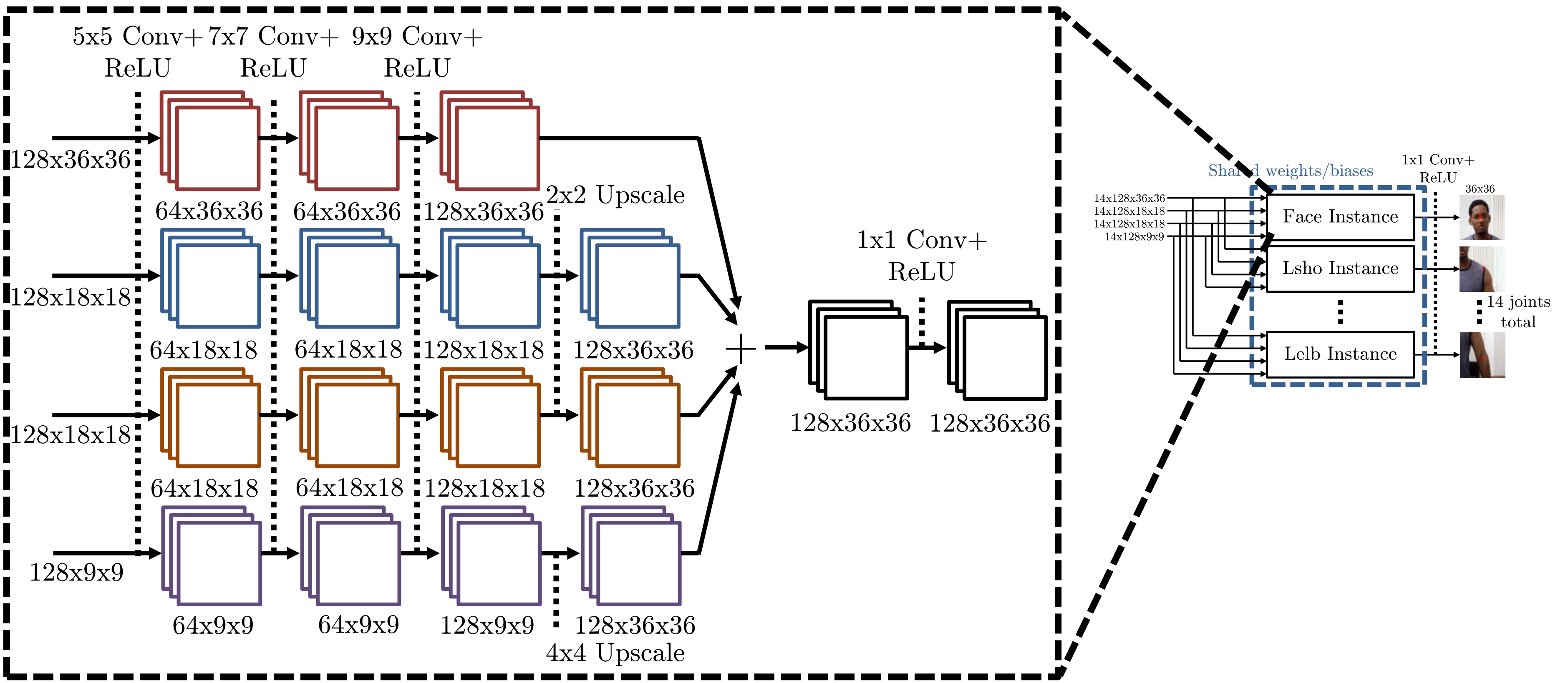
- Inputs sampled from multiple locations → Need separate networks (Siamese)
- Shared weights (L_1 thru L_{n-1})



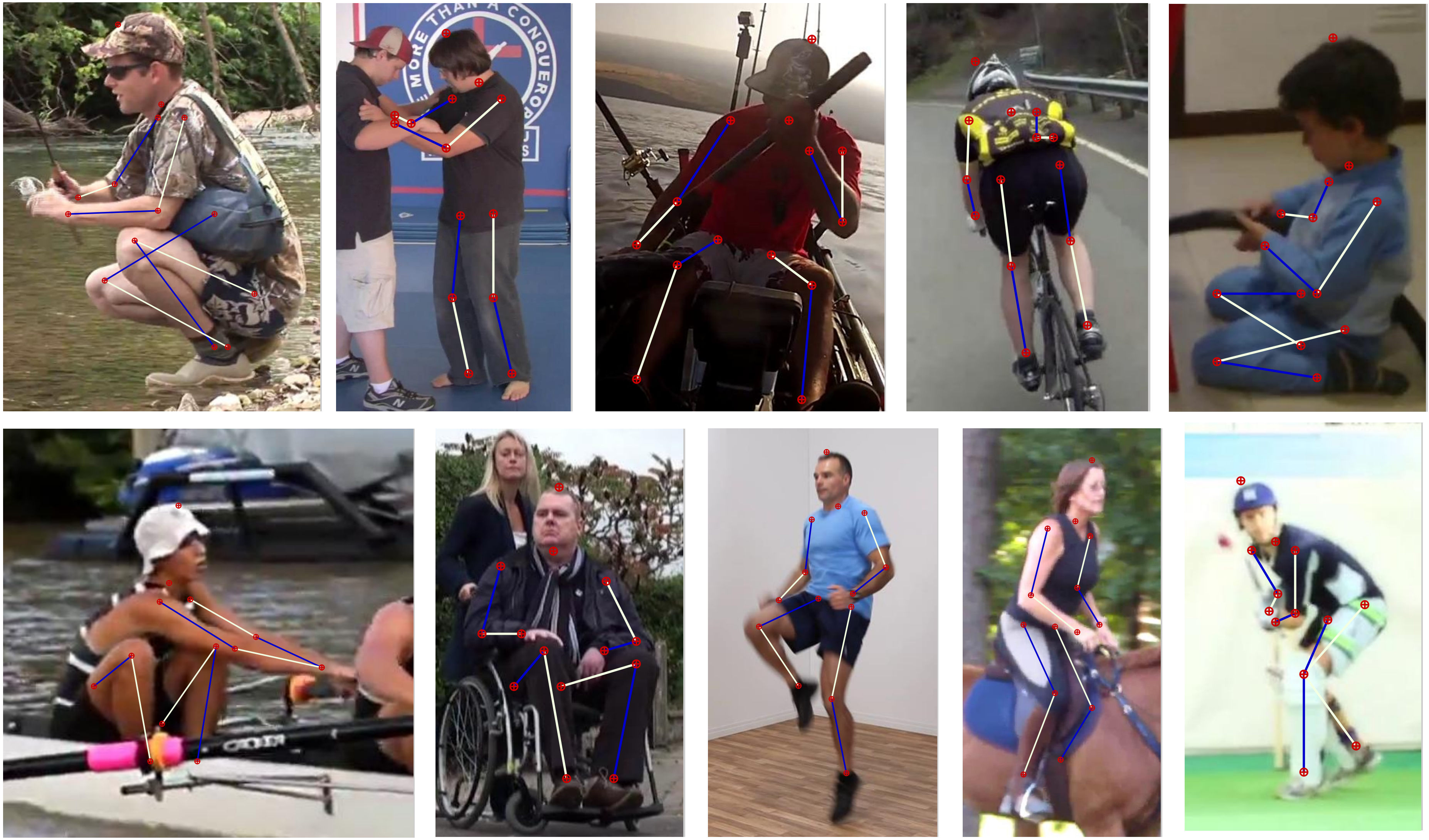
Fine Model – Replicated Instance

Use same strategy as the Coarse Model

- Fully Convolutional (with 1x1 conv layers)
- Up-sample to bring features into canonical resolution



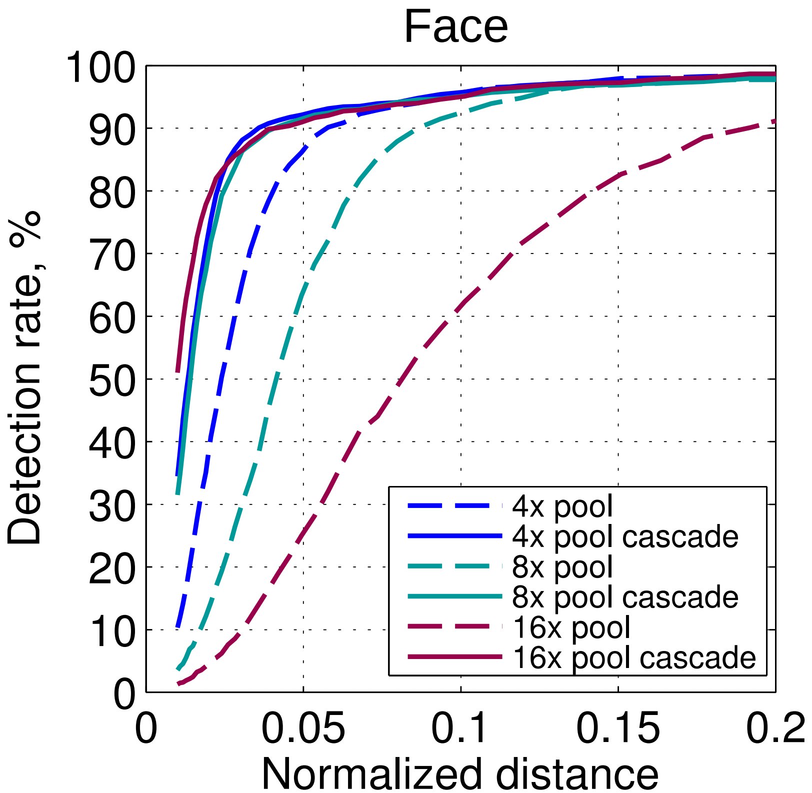
RESULTS



RESULTS

FLIC Dataset again

- Small improvement for wrist:
  - Not enough context in fine model
  - Too many mistakes in the coarse model

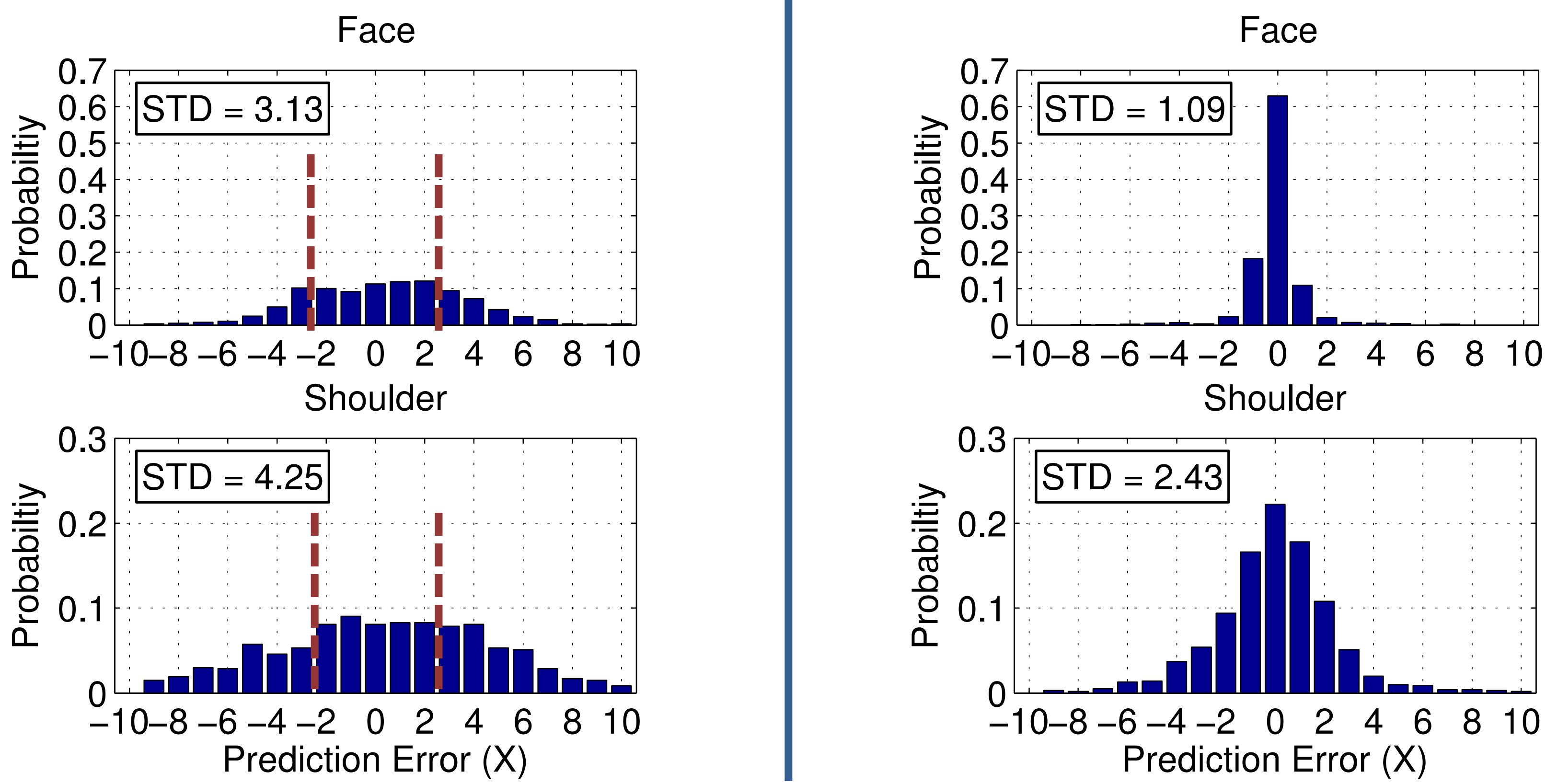


How Well Are We Doing?

Evaluation of our model

Here we use a part model with 4x pooling (+/- 2 pixel uncertainty)

Part-Model | Cascade



How Well Are We Doing?

Informal study to evaluate human performance

1. Show users examples and explain desired joint location
2. Ask users to label 10 images from FLIC
3. Compare this to our performance




How Well Are We Doing?

Comparing our ConvNet to Human Variance

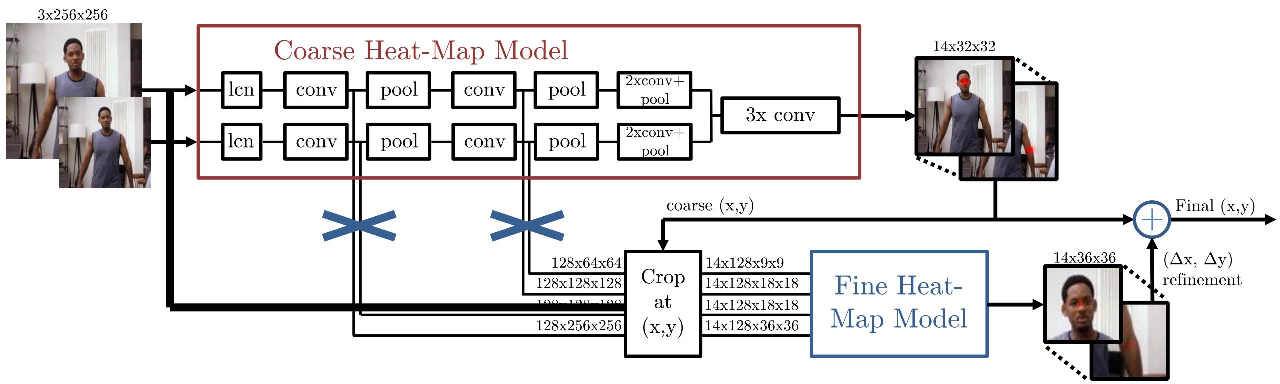
The 16x pooling module actually does quite well despite pooling.

	Face	Shoulder	Elbow	Wrist
Label Noise (10 images)	0.65	2.46	2.14	1.57
This work 4x (test-set)	1.09	2.43	2.59	2.82
This work 8x (test-set)	1.46	2.72	2.49	3.41
This work 16x (test-set)	1.45	2.78	3.78	4.16

Well, What About a Standard Cascade?

Baseline experiment

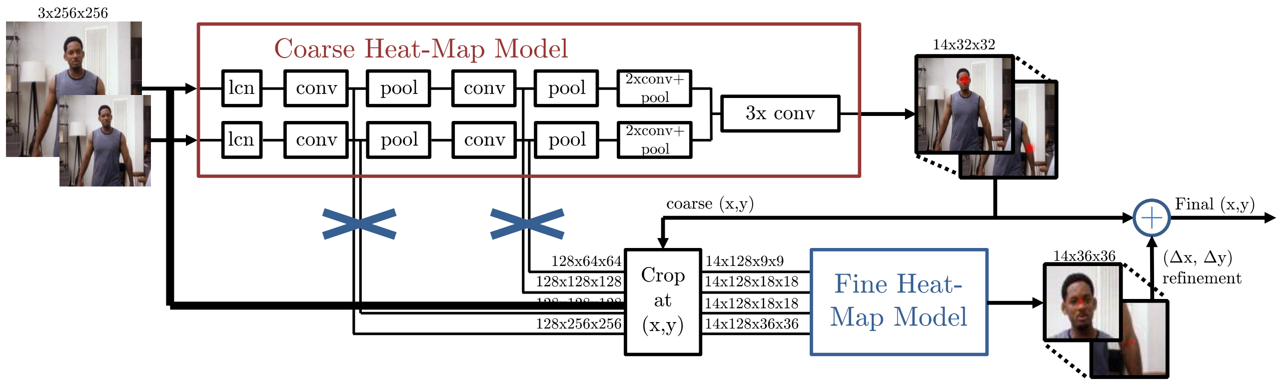
Construct a ConvNet that ONLY samples from the RGB



Well, What About a Standard Cascade?

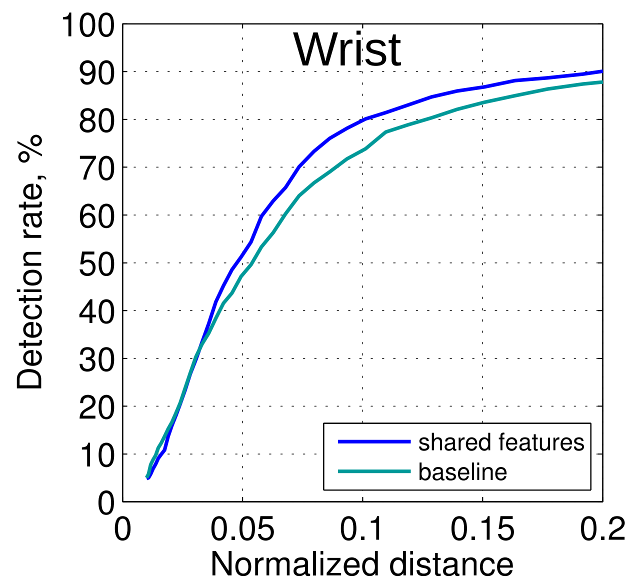
Baseline experiment

Construct a ConvNet that ONLY samples from the RGB



Shared features help

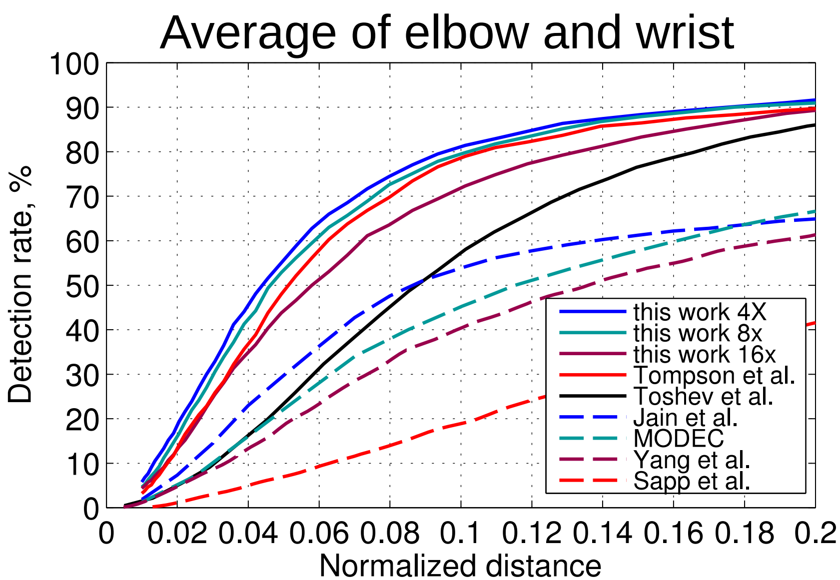
- Regularization
- Shared capacity
- More pooling → Higher delta



NYU

COMPARISON WITH OTHER MODELS

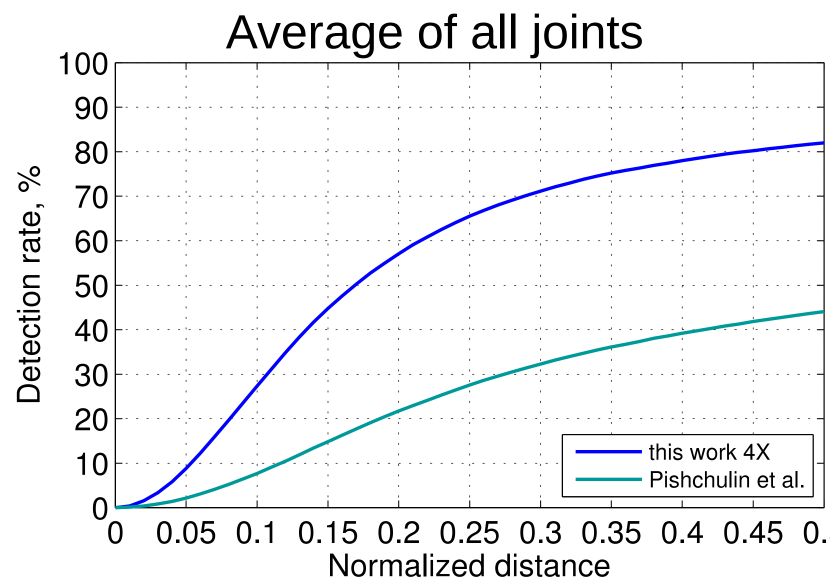
FLIC



PCK @ 0.05

	Head	Shoulder	Elbow	Wrist
Yang et al.	-	-	22.6	15.3
Sapp et al.	-	-	6.4	7.9
Eichner et al.	-	-	11.1	5.2
MODEC et al.	-	-	28.0	22.3
Toshev et al.	-	-	25.2	26.4
Jain et al.	-	42.6	24.1	22.3
Tompson et al.	90.7	70.4	50.2	55.4
This work 4x	**92.6**	73.0	**57.1**	**60.4**
This work 8x	92.1	**75.8**	55.6	56.6
This work 16x	91.6	73.0	47.7	45.5

MPII



PCK @ 0.2

	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Upper Body	Full Body
Gkioxari et al.	-	36.3	26.1	15.3	-	-	-	25.9	-
Sapp & Taskar	-	38.0	26.3	19.3	-	-	-	27.9	-
Yang & Ramanan	73.2	56.2	41.3	32.1	36.2	33.2	34.5	43.2	44.5
Pishchulin et al.	74.2	49.0	40.8	34.1	36.5	34.4	35.1	41.3	44.0
This work 4x	**96.0**	**91.9**	**83.9**	**77.7**	**80.9**	**72.2**	**64.8**	**84.5**	**82.0**

CONCLUSION

CONCLUSIONS

What have we shown?

- ConvNets are *really* good at Object Recognition
  - Even with real-time constraints

CONCLUSIONS

What have we shown?

- ConvNets are *really* good at Object Recognition
  - Even with real-time constraints
- Simple networks in limited domains work well (hand tracking)
  - Can leverage “real” models to create amazing data

CONCLUSIONS

What have we shown?

- ConvNets are *really* good at Object Recognition
  - Even with real-time constraints
- Simple networks in limited domains work well (hand tracking)
  - Can leverage “real” models to create amazing data
- For complex inputs we need domain optimizations
  - “belief propagation” within a neural network – enforce structure
  - Use multi-frame input – human motion is well defined and “learnable”
  - Multi-res input + Multi-res output – Offset spatial precision loss

ACKNOWLEDGEMENTS

QUESTIONS?

THANK YOU!

Some Related Work

Wang et al. & 3Gear (2009 to present)

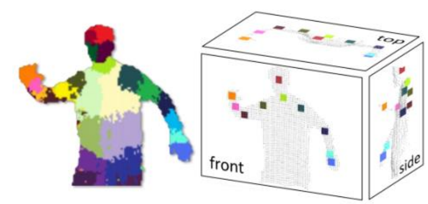
Tiny images (nearest-neighbor)

Oikonomidis et al. (2011,2012)

PSO search using synthetic depth images

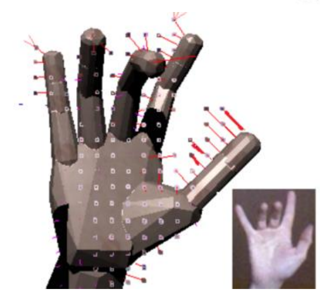
Shotton et al. (2011 and 2014-15)

Randomized Decision Forests



Melax et al. (2013)

Physics simulation (LCP)



Many more in the paper…

(Li, Weise, LeCun, Balan, Keskin, …)

Feature Detection

Infer 2D feature locations

- Fingertips, palm, knuckles, etc.

Convolutional network (ConvNet) for feature inference

- Efficient arbitrary function learner
- Reasonably fast using modern GPUs
- Self-similar features share learning capacity

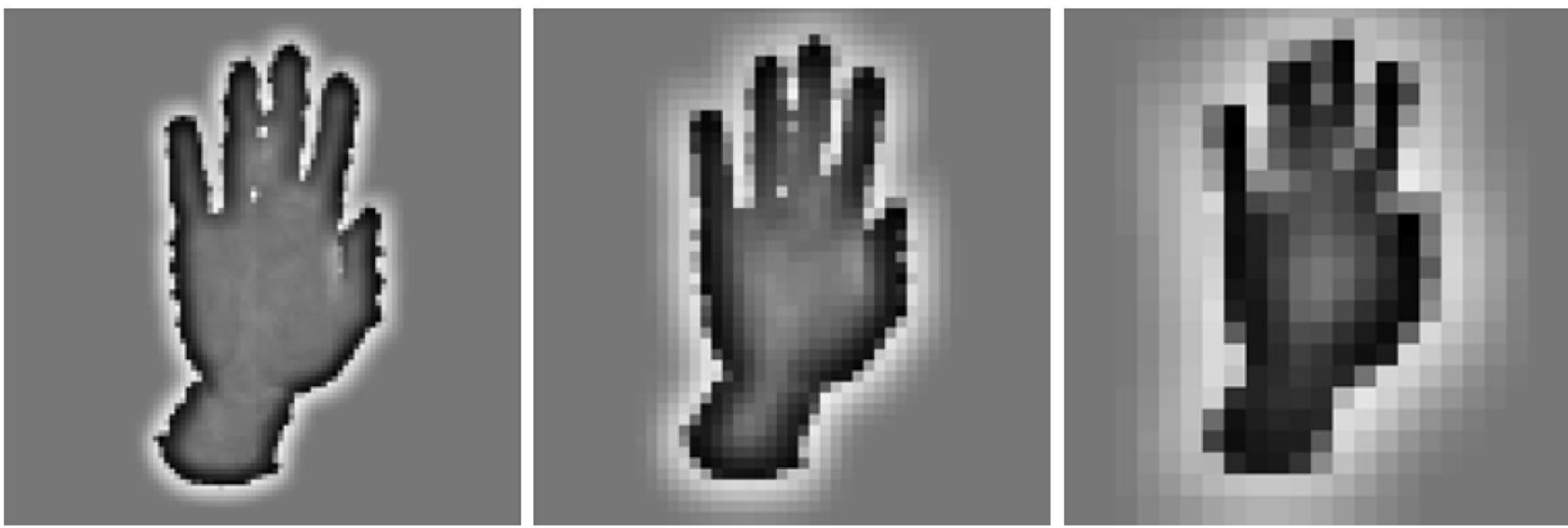
MULTI-RESOLUTION CONVNET

Downsampling (low pass) & local contrast normalization (high pass)

- 3 x banks with band-pass spectral density

CN convolution filter sizes constant

- CN bandwidth context is high without the cost of large (expensive) filter kernels



PSO/NM Objective Function

$$F\left(C\right)=\sum_{s=1}^{3}\left(\Delta(I_s,C)\right)+\Phi\left(C\right)+P\left(C\right)$$

L1 Depth comparison (multiple cameras)

$$\Delta\left(I_s,C\right)=\sum_{u,v}\min\left(\left|I_s(u,v)-R_s(C,u,v)\right|,d_{\max}\right)$$

Coefficient prior (out-of-bound penalty)

$$\Phi\left(C\right)=\sum_{k=1}^{n}w_k\left[\max\left(C_k-C_{k\max},0\right)+\max\left(C_{k\min}-C_k,0\right)\right]$$

Interpenetration constraint

- Sum of bounding sphere interpenetrations

NYU

IK Objective Function

$$f(m) = \sum_{i=1}^{n} [\Delta_i(m)] + \Phi(C)$$

$$\Delta_i(m) = \begin{cases} \left\| (u,v,d)_i^t - (u,v,d)_i^m \right\|_2 & \text{If } d_i^t \neq 0 \\ \left\| (u,v)_i^t - (u,v)_i^m \right\|_2 & \text{otherwise} \end{cases}$$

$\Delta_i(m)$ is a L2 norm in 2D or 3D if there is depth image support for that pixel

Lots of problems... But it works

Use PrPSO to minimize $f(m)$: hard to parameterize and multi-modal (so gradient descent methods fail)

Some Related Work

Andriluka et al. (CVPR 2009)

Pictorial Structures Revisited: People Detection and Articulated Pose Estimation

Shape context descriptors trained using ADABOOST + generative model and belief propagation

Felzenszwalb and Sirshick (PAMI 2010)

Object Detection with Discriminatively Trained Part-Based Models

Deformable Part Model (DPM) → “structure as latent variable”

Sapp and Taskar (CVPR 2013)

Modec: Multimodal decomposable models for human pose estimation

HOG + Graphical Model (and FLIC dataset)

Jain et al. (including me) (ICLR 2014)

Learning Human Pose Estimation Features with Convolutional Networks

Simple ConvNet with simple MRF spatial model

Toshev & Szegedy (CVPR 2014)

DeepPose: Human pose estimation via deep neural networks

Large ConvNet cascade to perform direct regression on UV locations

More in the paper (and more recent)…

(Pishchulin, Andriluka, LeCun, Johnson, Chen, …)

Spatial Model

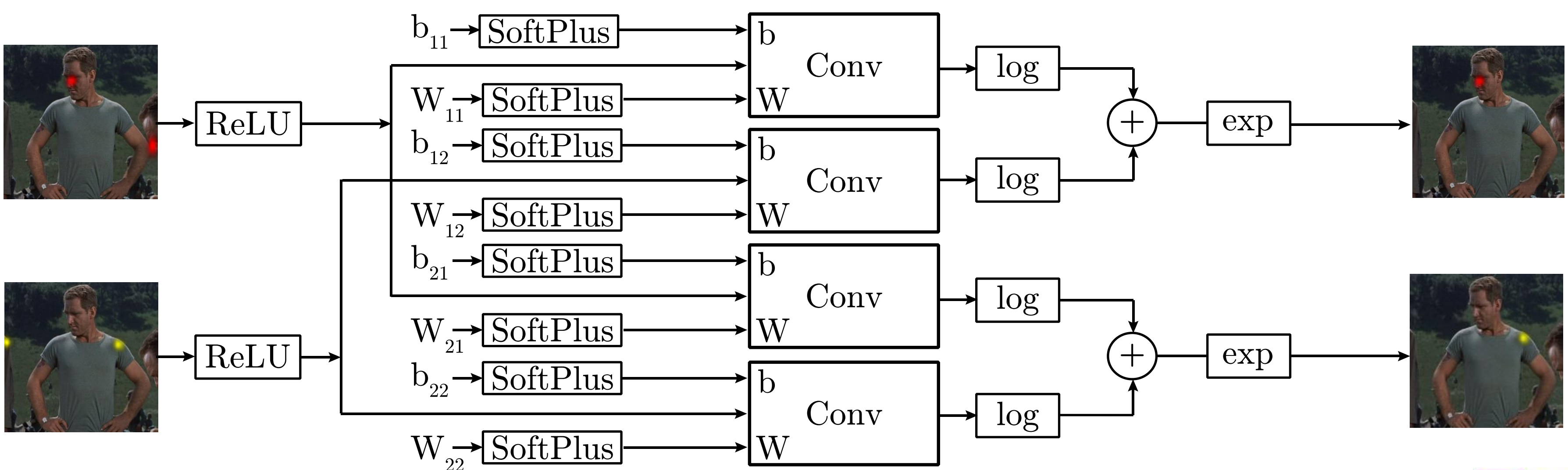
Implement it as a network

$$b(f)=\Phi(f)\prod_{i}\left(\Phi(x_i)*\Psi(f\mid x_i)+c(f\mid x_i)\right)$$

$$\bar{e}_A = \exp\left(\sum_{v\in V}\left[\log\left(\mathrm{SoftPlus}\left(e_{A|v}\right)*\mathrm{ReLU}\left(e_v\right)+\mathrm{SoftPlus}\left(b_{v\to A}\right)\right)\right]\right)$$

$$\text{where:}\quad \mathrm{SoftPlus}(x)={}^{1}\!/\!_{\beta}\log\left(1+\exp(\beta x)\right),\ {}^{1}\!/\!_{2}\leq\beta\leq 2$$

$$\mathrm{ReLU}(x)=\max(x,\epsilon),\ 0<\epsilon\leq 0.01$$



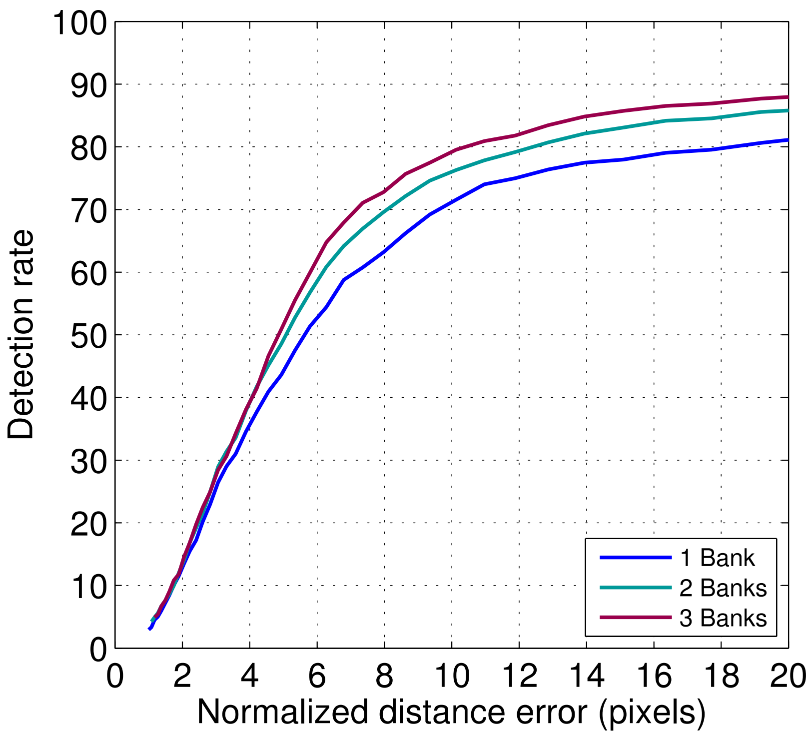
JOINT TRAINING

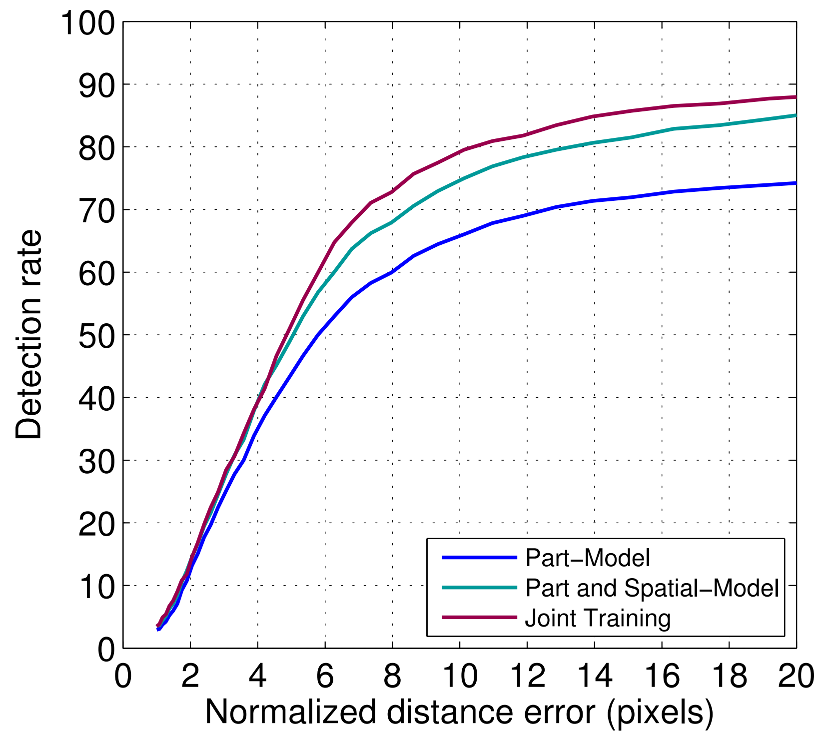
Joint training

- Pre-train both models separately
- Joint train (BPROP) through both models

Wrist Performance:

$$\text{DetectionRate}\,(R) = \frac{100}{N}\sum_{t=1}^{N}\left(\frac{\|x \;\; - x^t\|_2}{(\text{torso height t})\,/100} \leq R\right)$$





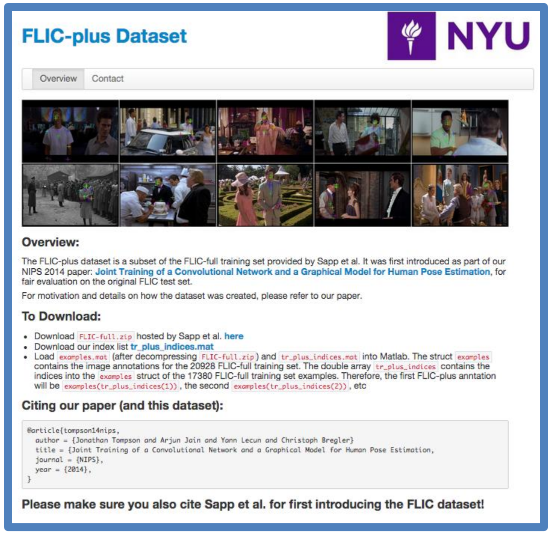
FLIC-PLUS

FLIC-Extended[1] is not fair!

- 20928 training / 1014 test samples
- 800 out of the 1014 test images (~80%) have an image in the training set that is at most 40 frames away

"FLIC-Plus":

- Use Mechanical Turk to find scenes in training and test sets
- Reject training set images from the same scene


FLIC-plus Dataset

NYU

Overview Contact

Overview:

The FLIC-plus dataset is a subset of the FLIC-full training set provided by Sapp et al. It was first introduced as part of our NIPS 2014 paper: **Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation**, for fair evaluation on the original FLIC test set.

For motivation and details on how the dataset was created, please refer to our paper.

To Download:

- Download `FLIC-full.zip` hosted by Sapp et al. **here**
- Download our index list **tr_plus_indices.mat**
- Load `examples.mat` (after decompressing `FLIC-full.zip`) and `tr_plus_indices.mat` into Matlab. The struct `examples` contains the image annotations for the 20928 FLIC-full training set. The double array `tr_plus_indices` contains the indices into the `examples` struct of the 17380 FLIC-full training set examples. Therefore, the first FLIC-plus anntation will be `examples(tr_plus_indices(1))`, the second `examples(tr_plus_indices(2))`, etc

Citing our paper (and this dataset):

```
@article{tompson14nips,
  author = {Jonathan Tompson and Arjun Jain and Yann Lecun and Christoph Bregler}
  title = {Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation,
  journal = {NIPS},
  year = {2014},
}
```

Please make sure you also cite Sapp et al. for first introducing the FLIC dataset!

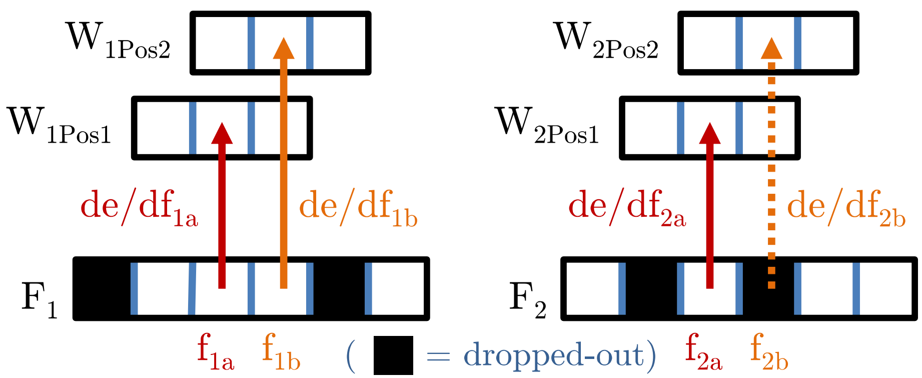
cims.nyu.edu/~tompson/flic_plus.htm

(1) B. Sapp and B. Taskar. MODEC: Multimodel decomposition models for human pose estimation. CVPR'13

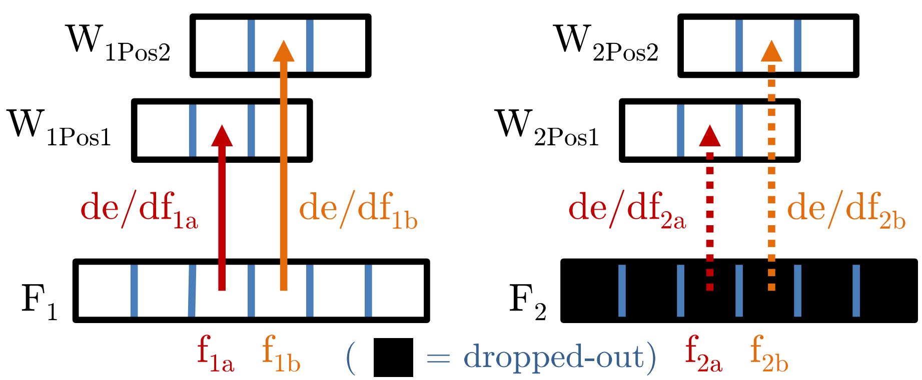
Spatial Dropout

Dropout fails for fully-convolutional networks

- Gradients are correlated because pixels are highly correlated
- Dropout just scales the learning rate



Instead Dropout the entire feature



HOW WELL ARE WE DOING?

Time (ms) for FPROP

For 16x pooling: first 2 layers dominates runtime

	4x pool	8x pool	16x pool
Coarse-Model	140.0	74.9	54.7
Fine-Model	17.2	19.3	15.9
Cascade	157.2	94.2	70.6