If Only We Had Tracked Something Like This Before

Gabriel Brostow, UCL

Gabriel's Team (see homepage)

Maciej Gryka

Dr Ed Johns

Prof Mike Terry

Peter Rennert

Clément Godard

Oisin Mac Aodha

Malcolm Reynolds

Dr Tom Haines

Special thanks to:

Oisin Mac Aodha

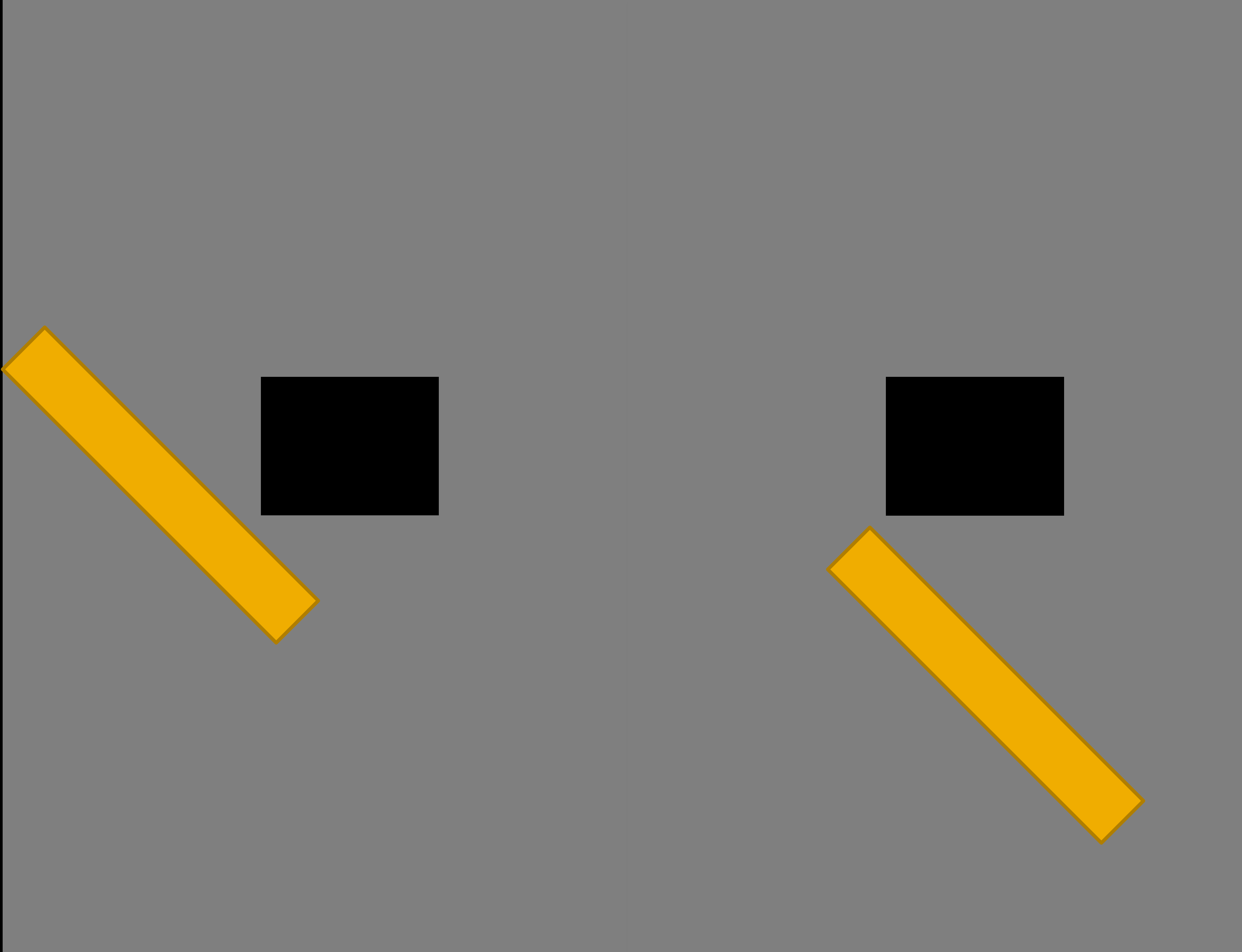
Dr Cristina Garcia Cifuentes

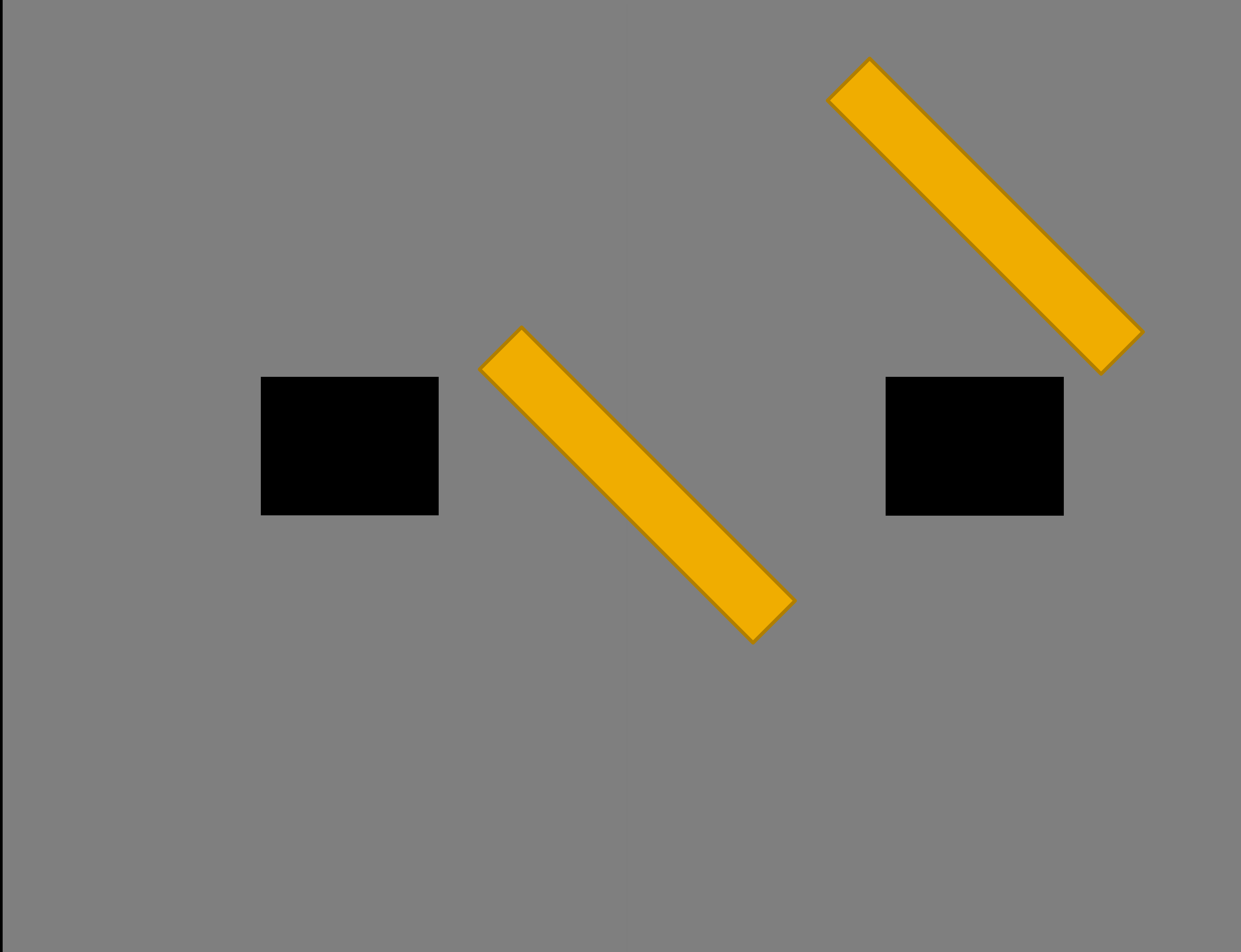
Ahmad Humayun

“A good hockey player plays where the puck is. A great hockey player plays where the puck is going to be."

- Wayne Gretzky

Photo from Arking.com





The Tracking Wishlist

- Monitor confidence of estimated $\mathbf{f}_i$
- Track real world motion → predict
- Cope with camouflage / invisibility
- Get accurate/faster with experience

The Tracking Wishlist

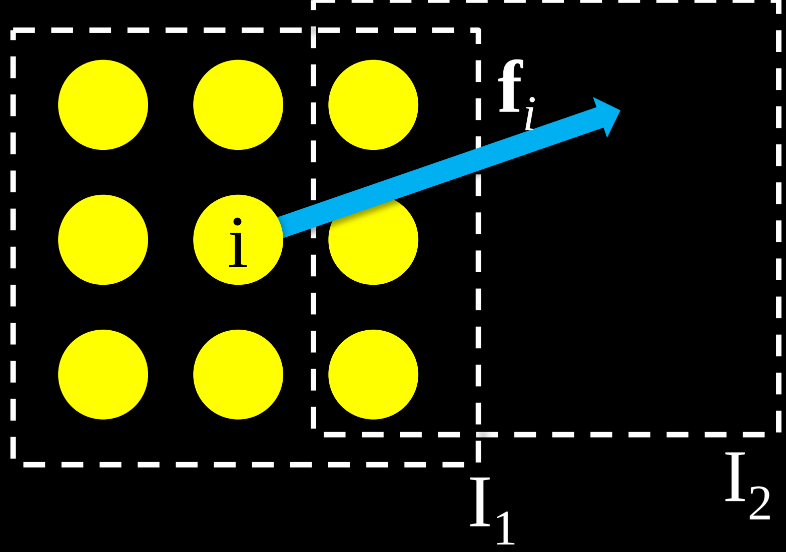
- *Monitor confidence of estimated* $\mathbf{f}_i$
- *Track real world motion → predict*
- Cope with camouflage / invisibility
- Get accurate/faster with experience


No
Ball Games

Infer flow confidence

[Mac Aodha et al. PAMI 2013]

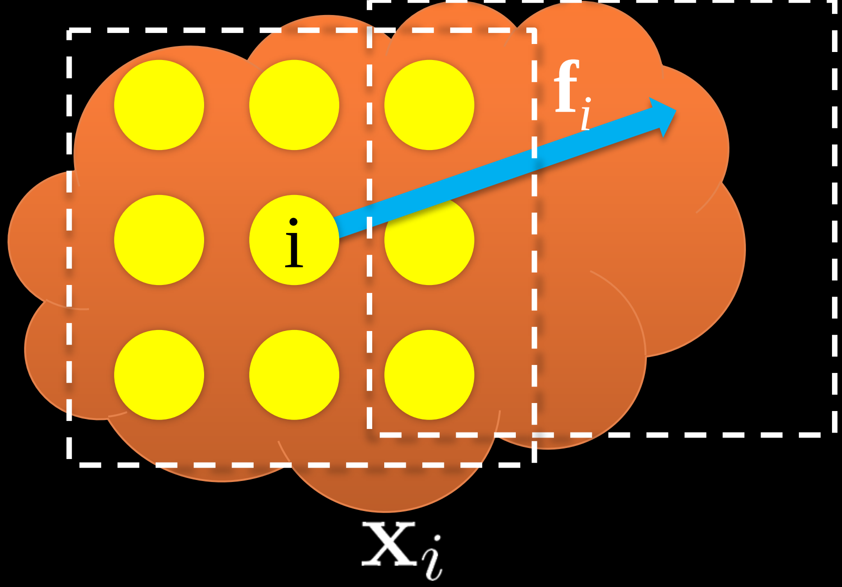
Approach



- Observe flow-algorithm-A's output $\mathbf{f}_i$ in different situations
- Compare results to ground truth flow of i
  - Is the End Point Error (epe) small? Sometimes?

Is there a correlation we can learn?

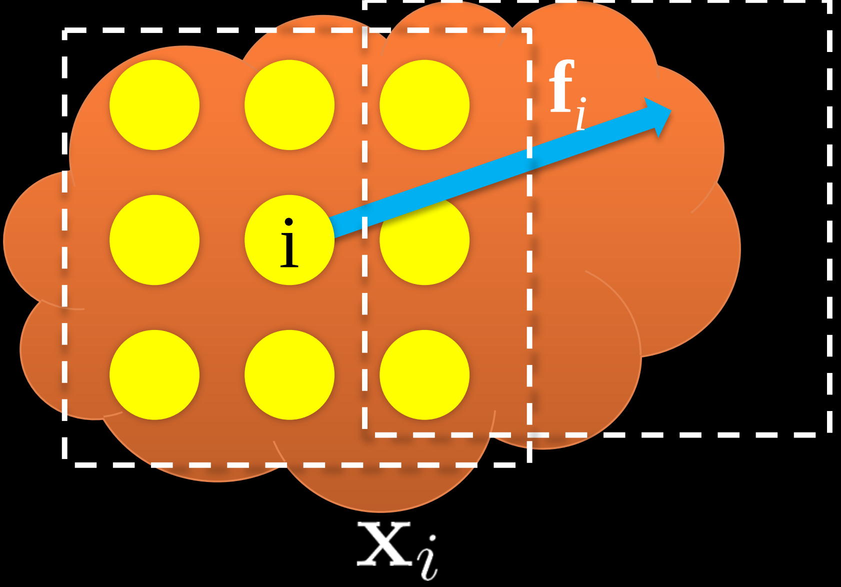
Model



- We define $\psi_i \in [0, 1]$ as our confidence that flow vector $\mathbf{f}_i$ is "reliable" (or not):

$$c_i = \begin{cases} 1 & \epsilon^i_{epe} \leqslant \epsilon^s_{epe} \\ 0 & \epsilon^i_{epe} > \epsilon^s_{epe} \end{cases}$$

Model

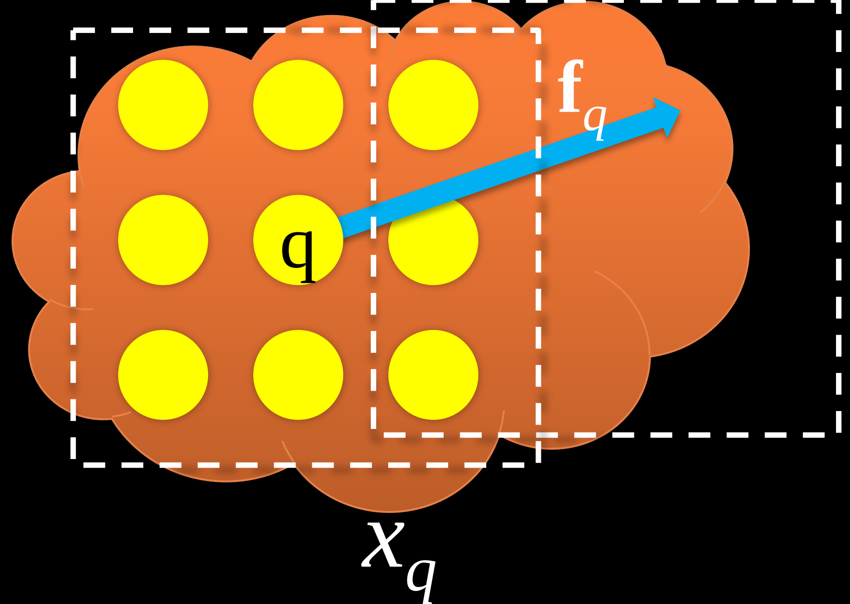


- We define $\psi_i \in [0,1]$ as our confidence that flow vector $\mathbf{f}_i$ is "reliable" (or not):

$$c_i = \begin{cases} 1 & \epsilon^i_{epe} \leqslant \epsilon^s_{epe} \\ 0 & \epsilon^i_{epe} > \epsilon^s_{epe} \end{cases}$$

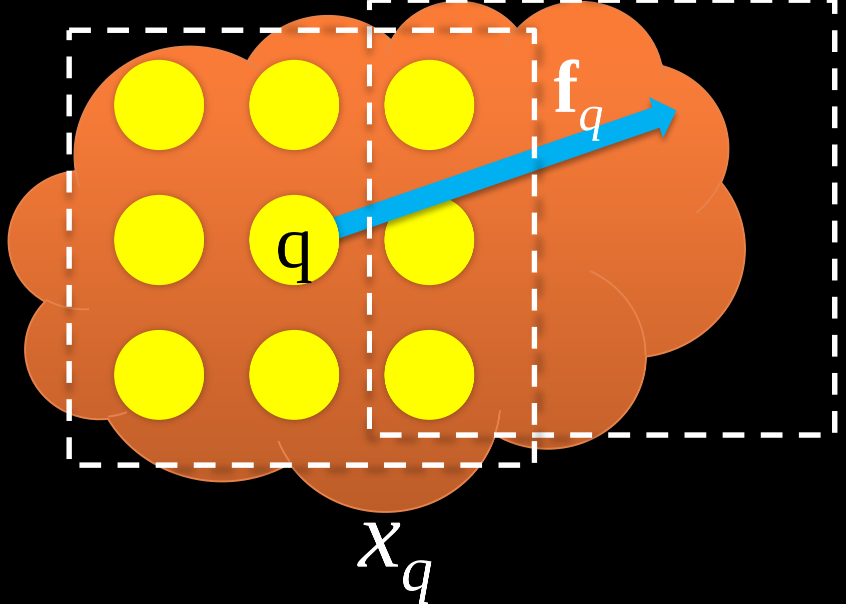
$$\mathcal{D} = \{(\mathbf{x}_i, c_i) | \mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^n$$

Infer Confidence



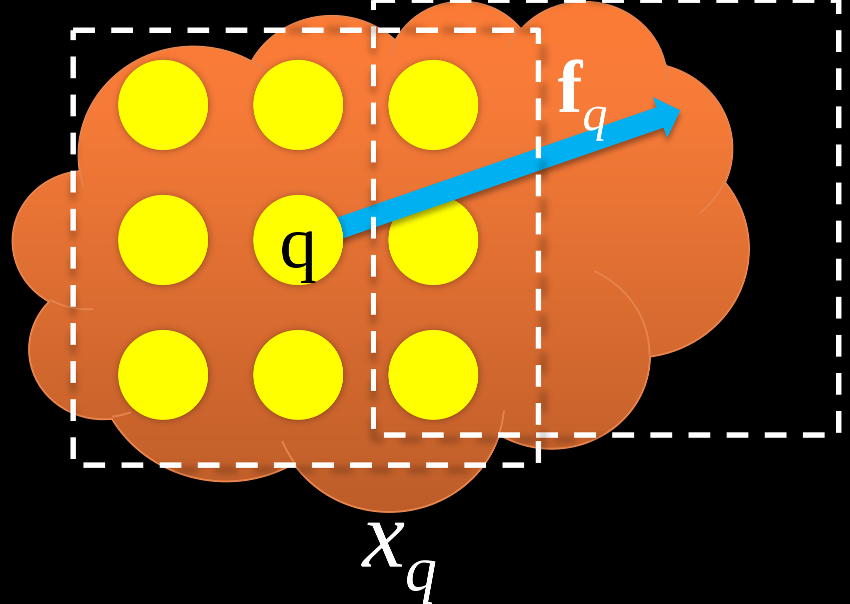
- At test time, $\psi_q = Pr(c_q = 1 | x_q)$

Infer Confidence



- At test time, $\psi_q = Pr(\, c_q = 1 | \, x_q \,)$
- Feature selection:
  -
  -
  -
  -

Infer Confidence



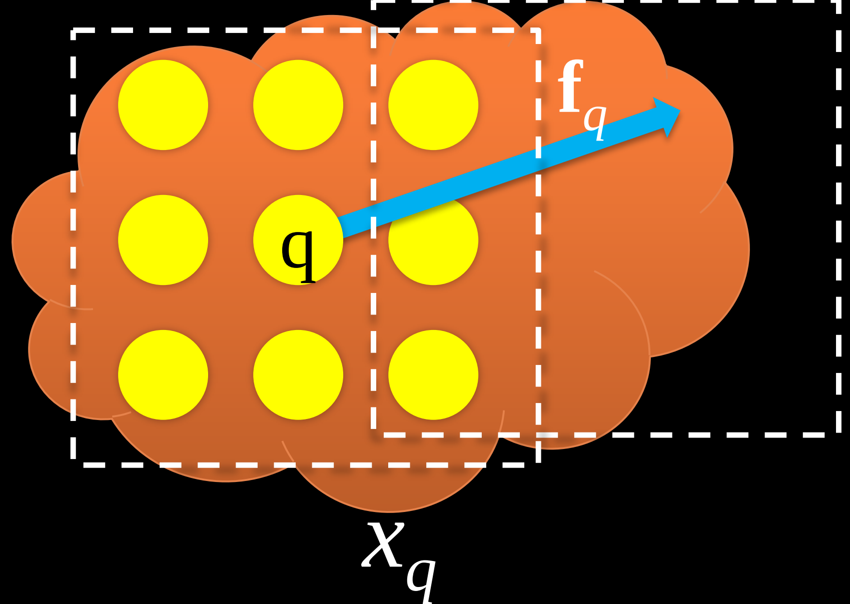
- At test time, $\psi_q = Pr(\,c_q = 1 |\, x_q)$

- Feature selection:

$$g(x, y, z) = ||\nabla I_1||$$

- Spatial Gradient
-
-
-

Infer Confidence



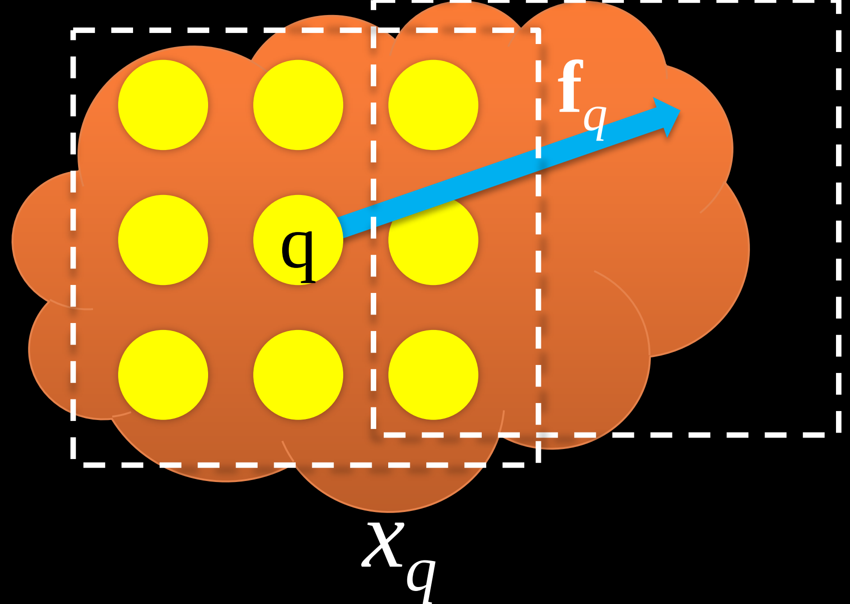
- At test time, $\psi_q = Pr(\, c_q = 1 |\, x_q)$

- Feature selection:

$$g(x,y,z) = ||\nabla I_1||$$
$$d(x,y,z) = disTrans(||\nabla I_1|| > \tau)$$

- Spatial Gradient
- Distance Transform
-
-

Infer Confidence



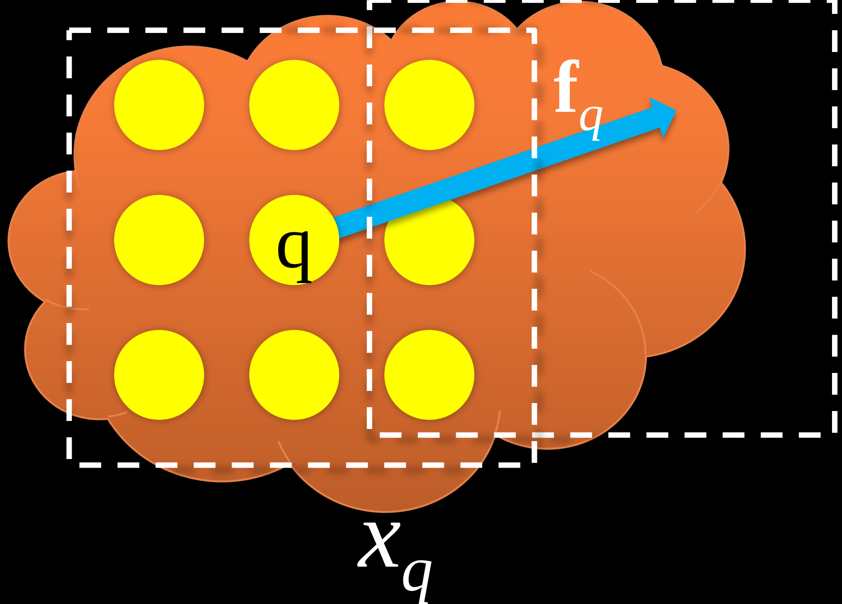
- At test time, $\psi_q = Pr(c_q = 1 | x_q)$

- Feature selection:

$$g(x,y,z) = ||\nabla I_1||$$
$$d(x,y,z) = disTrans(||\nabla I_1|| > \tau)$$
$$t_x(x,y,z) = ||\nabla \dot{u}||$$

- Spatial Gradient
- Distance Transform
- Temporal Gradient
-

Infer Confidence



- At test time, $\psi_q = Pr(c_q = 1 | x_q)$

- Feature selection:

$$g(x,y,z) = ||\nabla I_1||$$

$$d(x,y,z) = disTrans(||\nabla I_1|| > \tau)$$

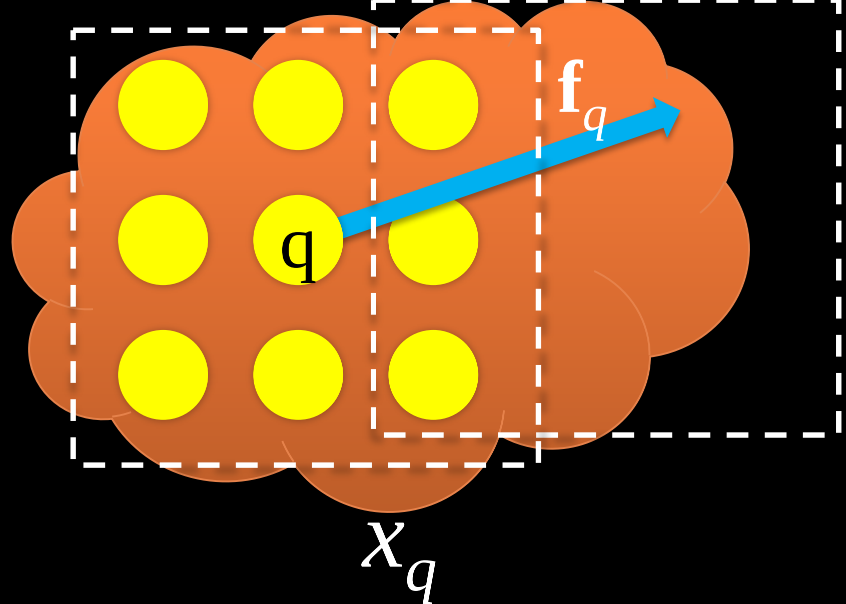
$$t_x(x,y,z) = ||\nabla \dot{u}||$$

$$r(x,y,k) = |I_1(x,y) - bicubic(I_2(x+u^k, y+v^k))|$$

- Spatial Gradient
- Distance Transform
- Temporal Gradient
- Residual Error

(most at multiple scales z)

Infer Confidence



- At test time, $\psi_q = Pr(\, c_q = 1 | \, x_q \,)$
- Feature selection:

nt

$d(x,y,$ … sform

t … dient

$r(x,y,k) = |I_1(x,y) - bicubic(I_2(x+u^k, y+v^k))|$

- Residual Error

(most at multiple scales z)

Let a Random Forest decide!

Generating more data

http://visual.cs.ucl.ac.uk/pubs/flowConfidence/

Generating more data

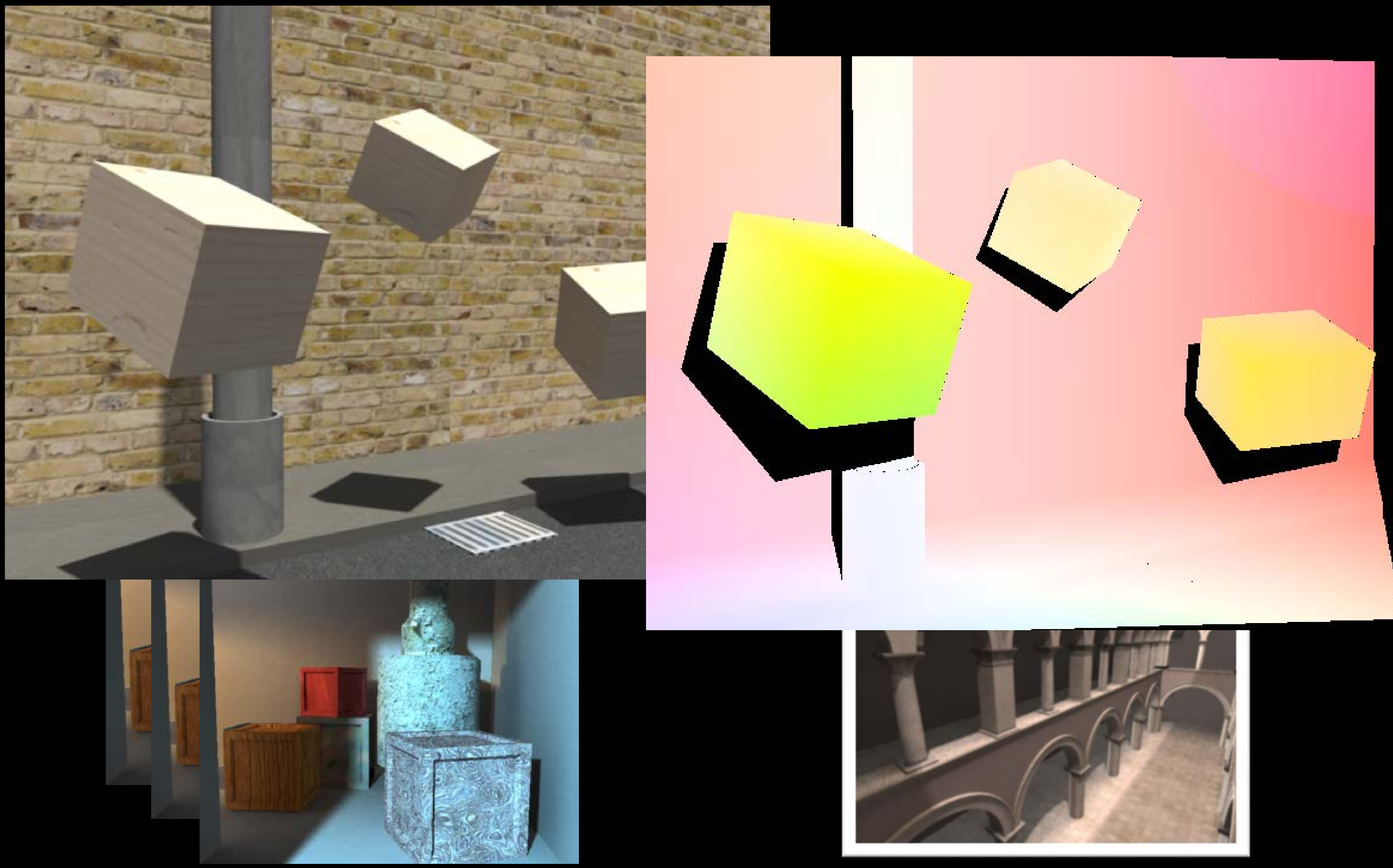
http://visual.cs.ucl.ac.uk/pubs/flowConfidence/

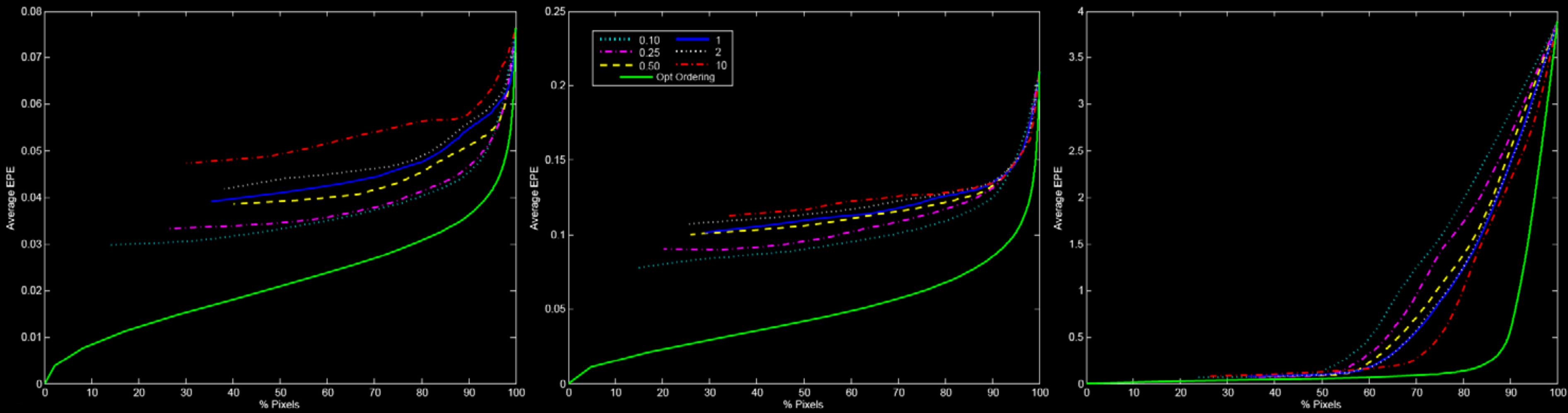
Generating more data

http://visual.cs.ucl.ac.uk/pubs/flowConfidence/

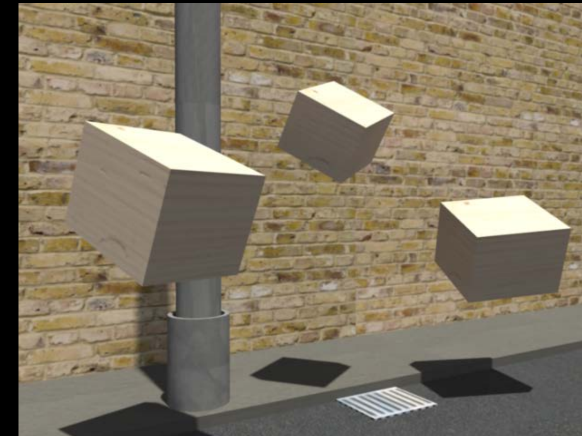
Generating more data

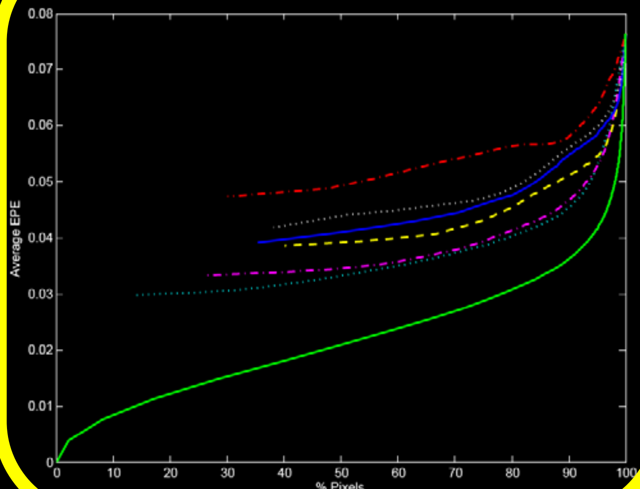
http://visual.cs.ucl.ac.uk/pubs/flowConfidence/

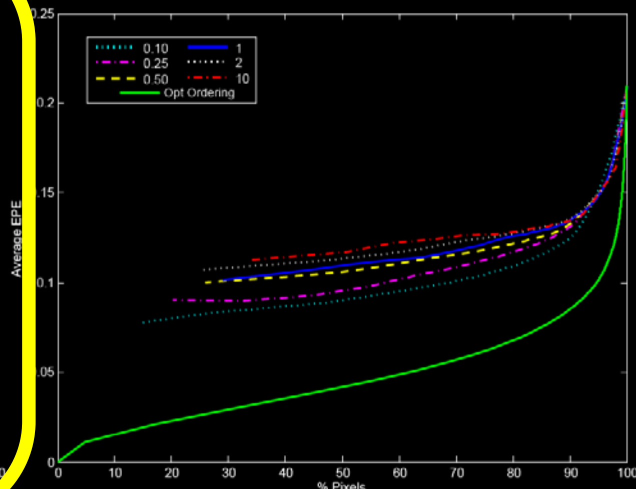


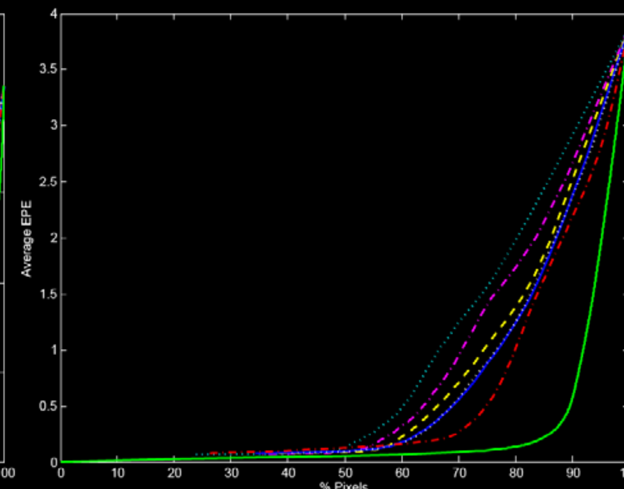


Here, using Secrets of Optical Flow Estimation, Sun et al. 2010

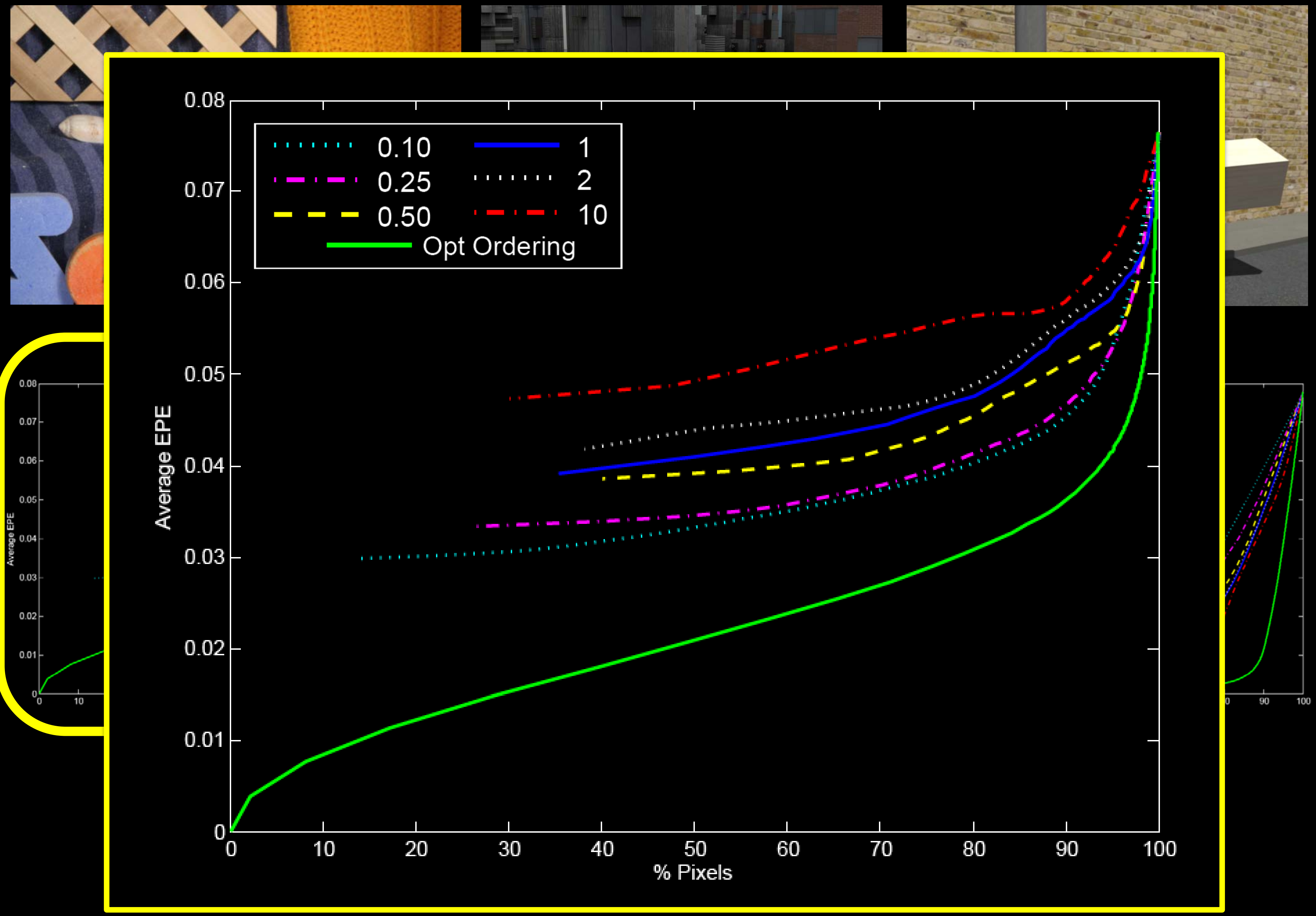




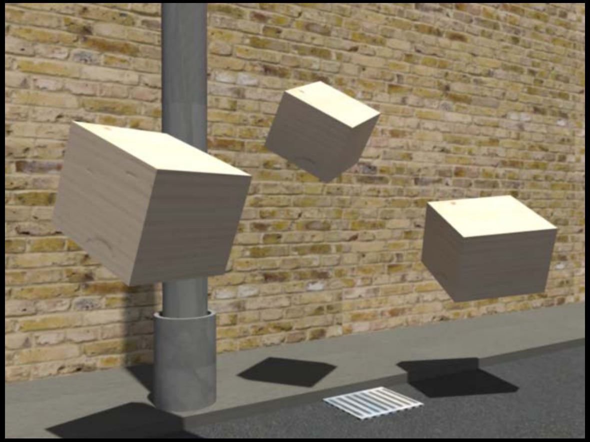


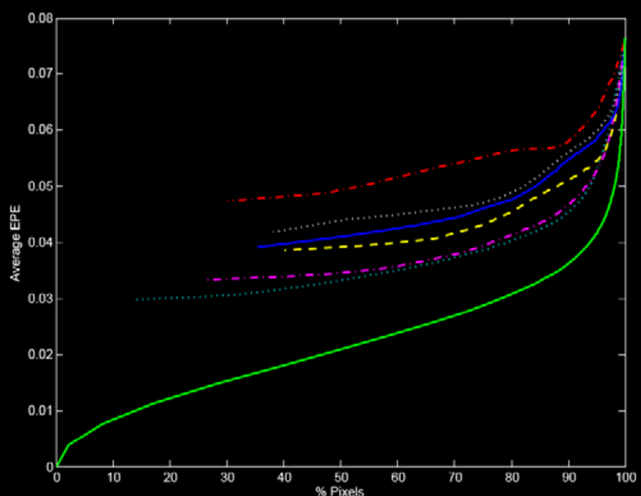


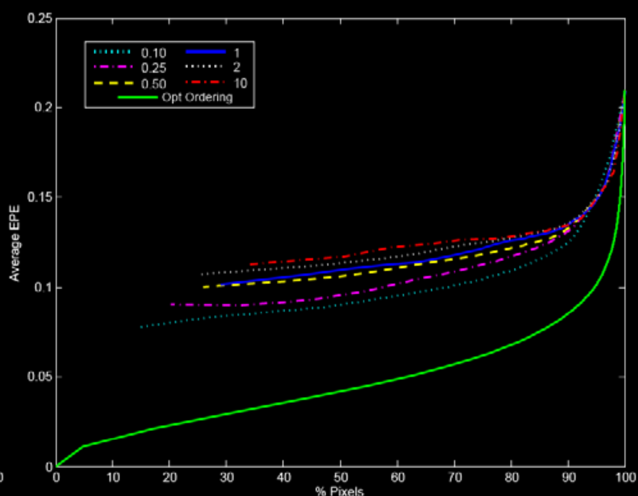
Here, using Secrets of Optical Flow Estimation, Sun et al. 2010

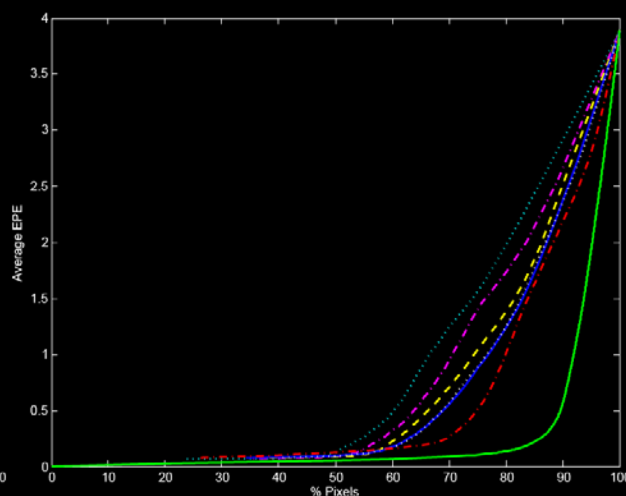


0.10
0.25
0.50
1
2
10
Opt Ordering
Average EPE
% Pixels
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
0
10
20
30
40
50
60
70
80
90
100

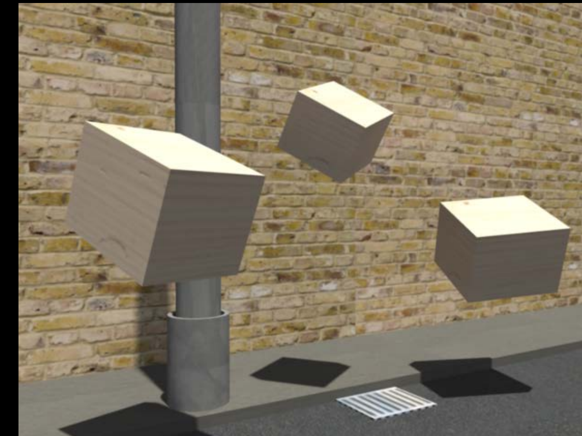


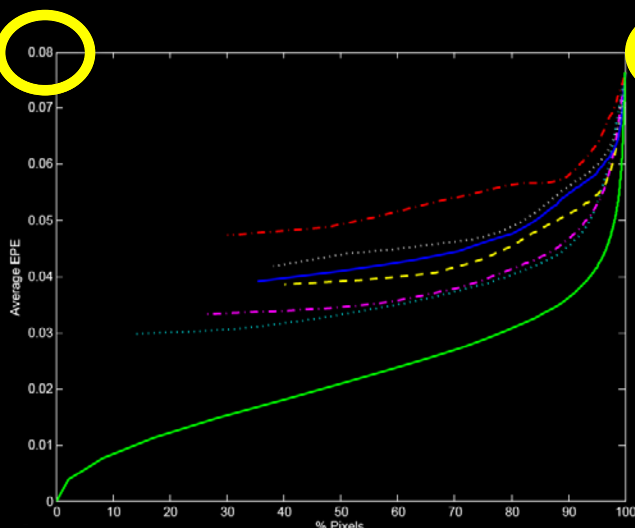

Average EPE
% Pixels


0.10
1
0.25
2
0.50
10
Opt Ordering
Average EPE
% Pixels

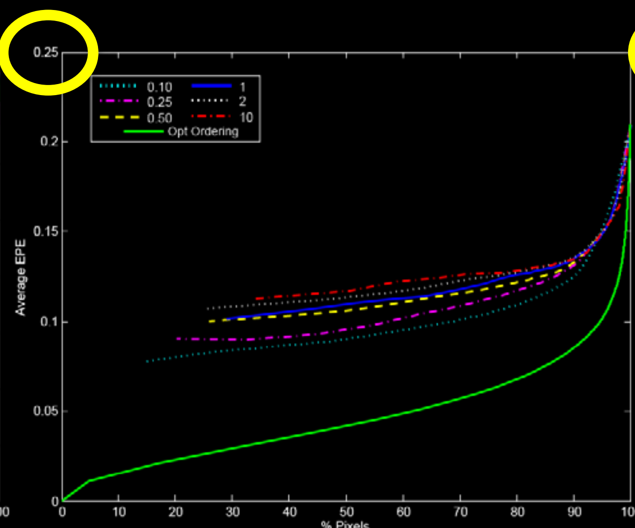

Average EPE
% Pixels



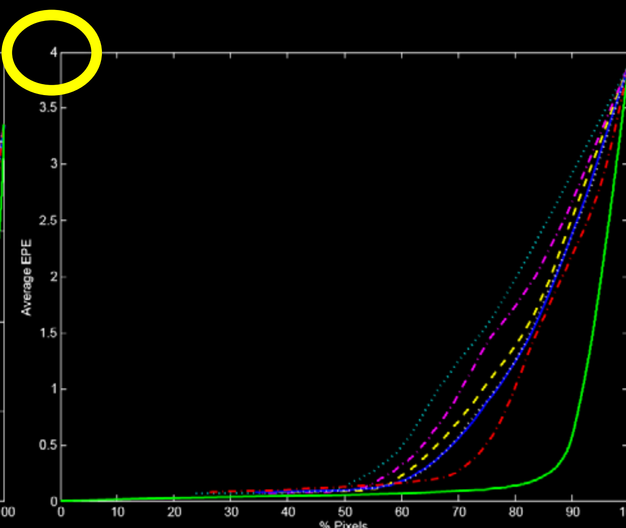




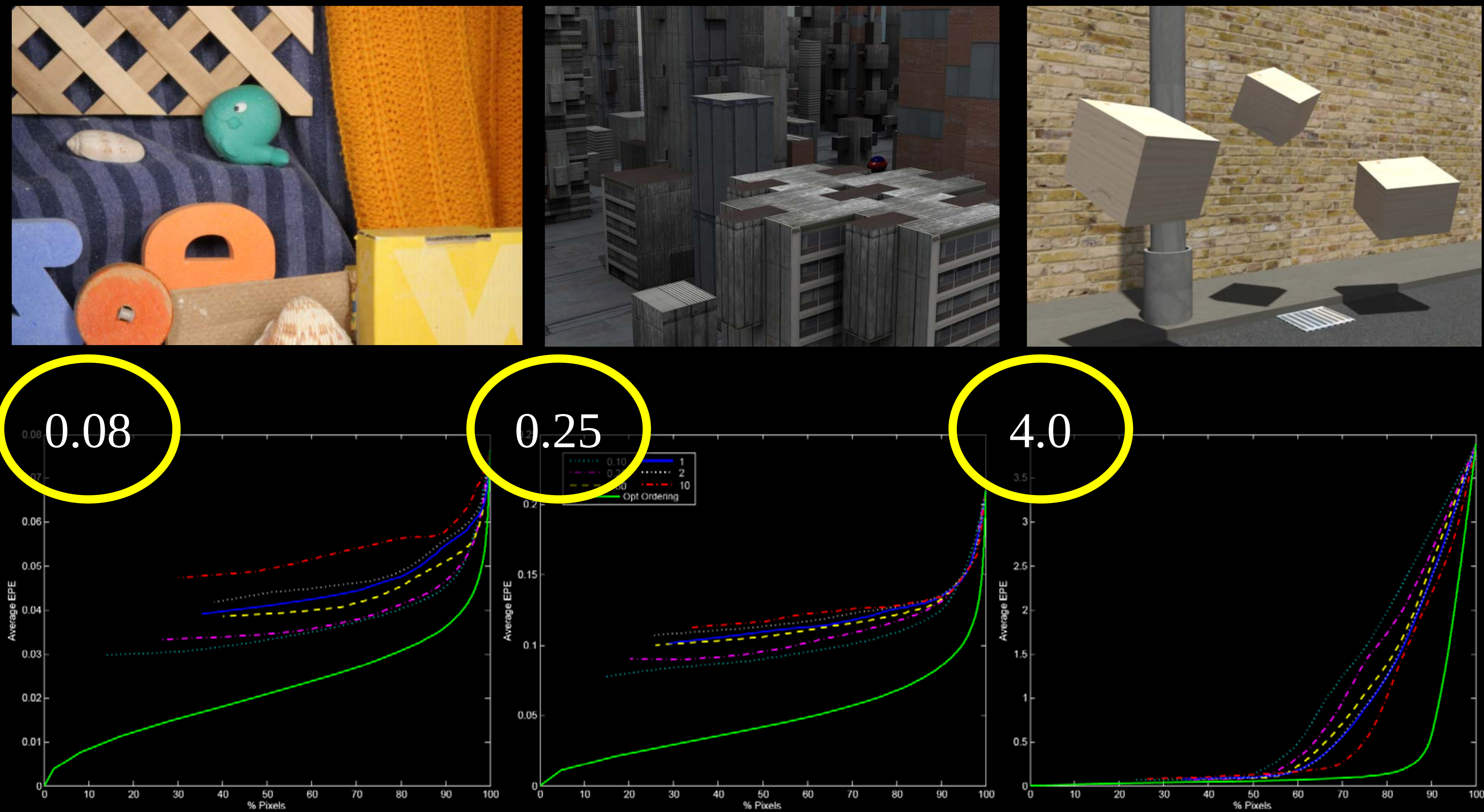
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
Average EPE
0 10 20 30 40 50 60 70 80 90 100
% Pixels

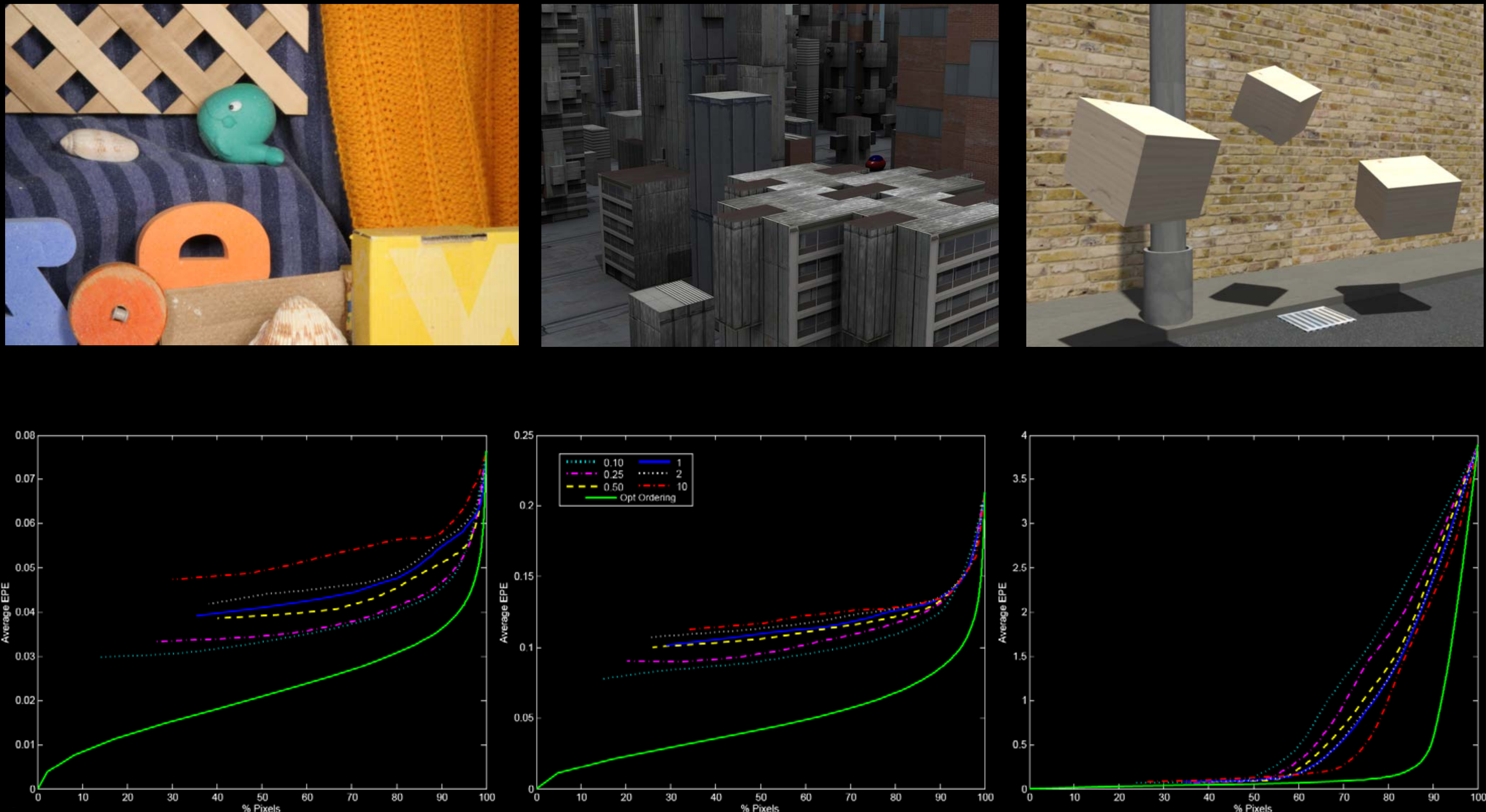


0.10
0.25
0.50
1
2
10
Opt Ordering
0.25
0.2
0.15
0.1
0.05
0
Average EPE
0 10 20 30 40 50 60 70 80 90 100
% Pixels

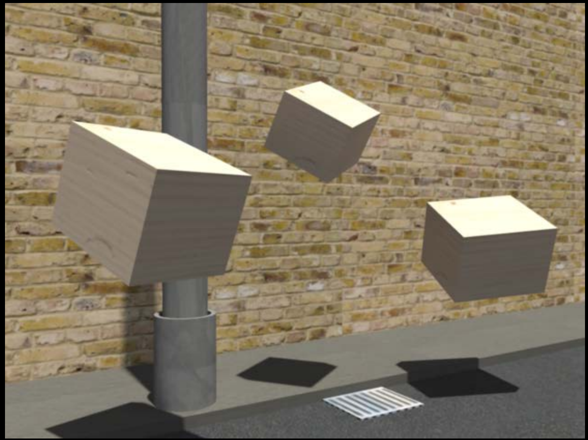


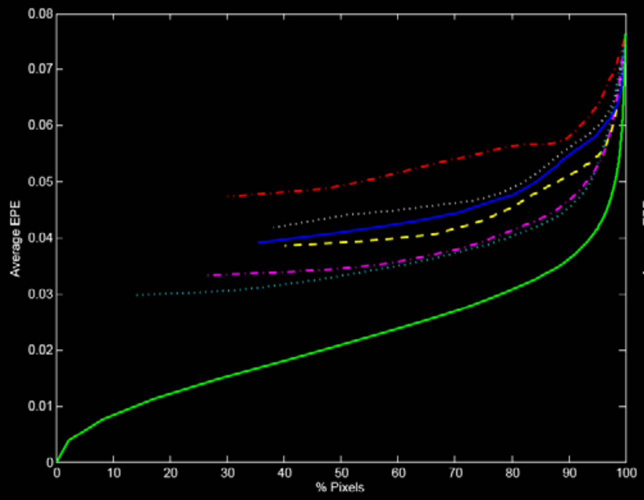
4
3.5
3
2.5
2
1.5
1
0.5
0
Average EPE
0 10 20 30 40 50 60 70 80 90 100
% Pixels

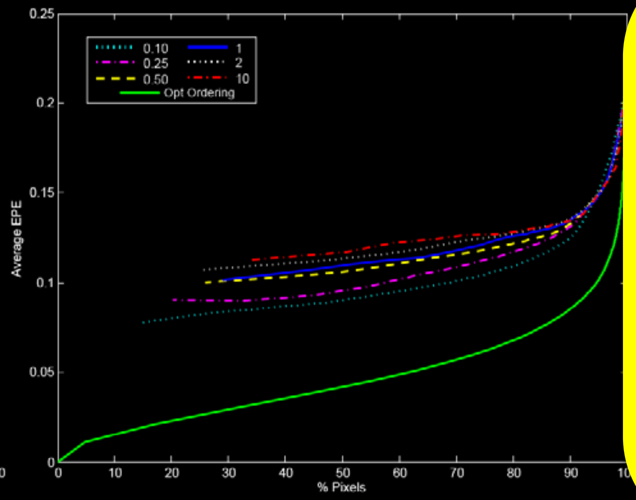

0.08
0.25
4.0
0.10
1
2
10
Opt Ordering
Average EPE
% Pixels

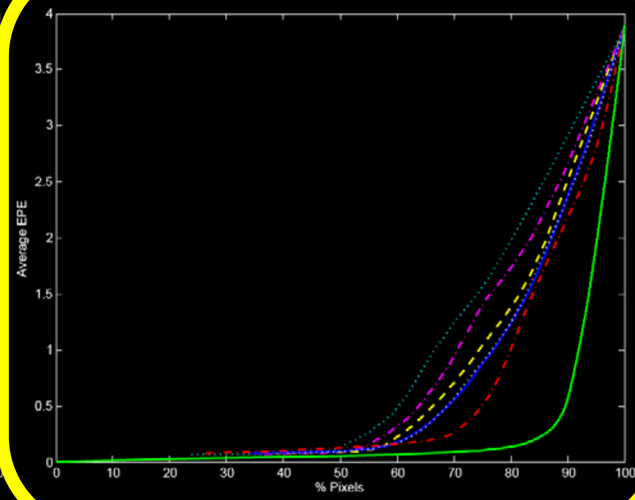


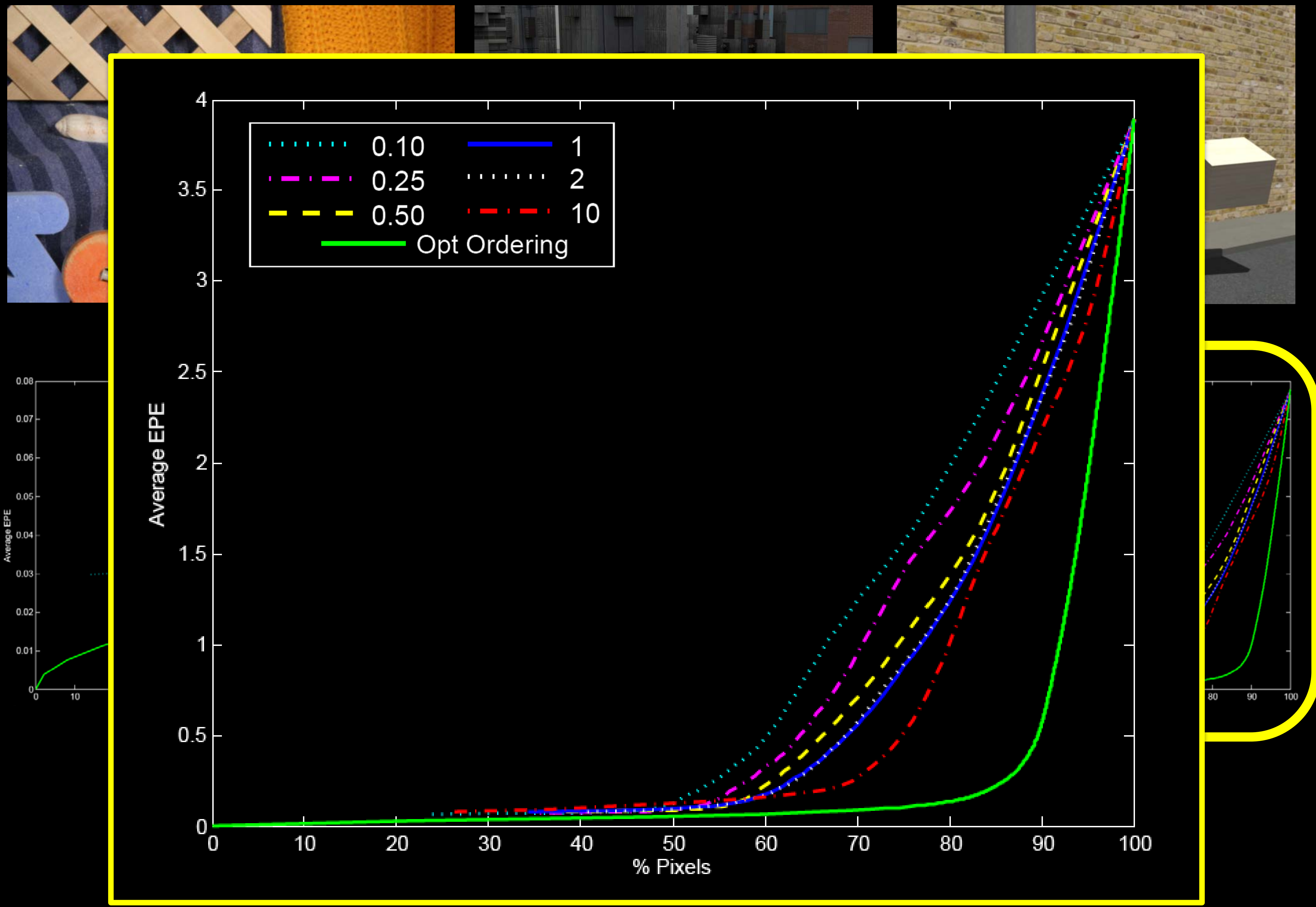
0.10
0.25
0.50
1
2
10
Opt Ordering
Average EPE
% Pixels



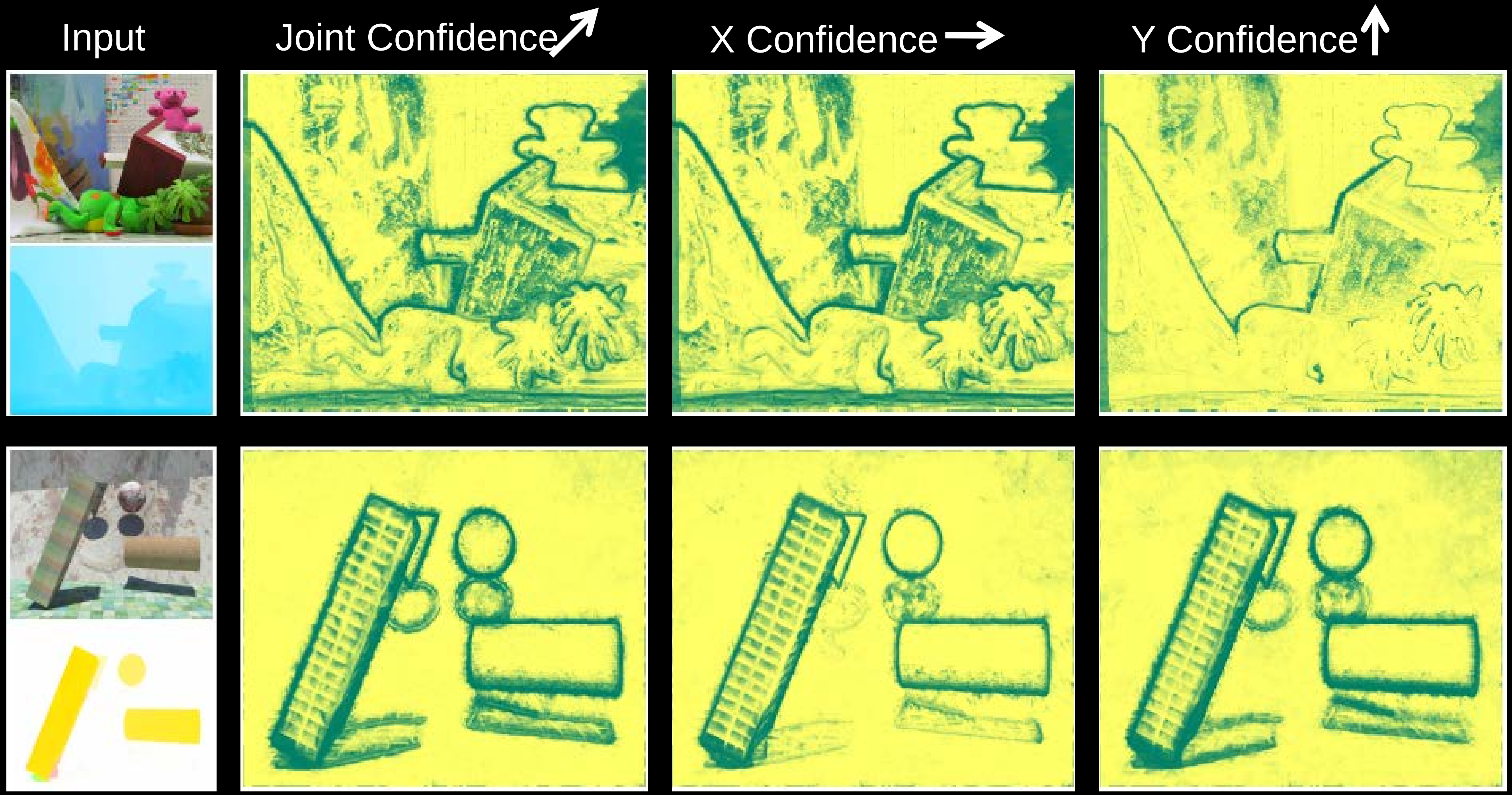

Average EPE
% Pixels


0.10
1
0.25
2
0.50
10
Opt Ordering
Average EPE
% Pixels

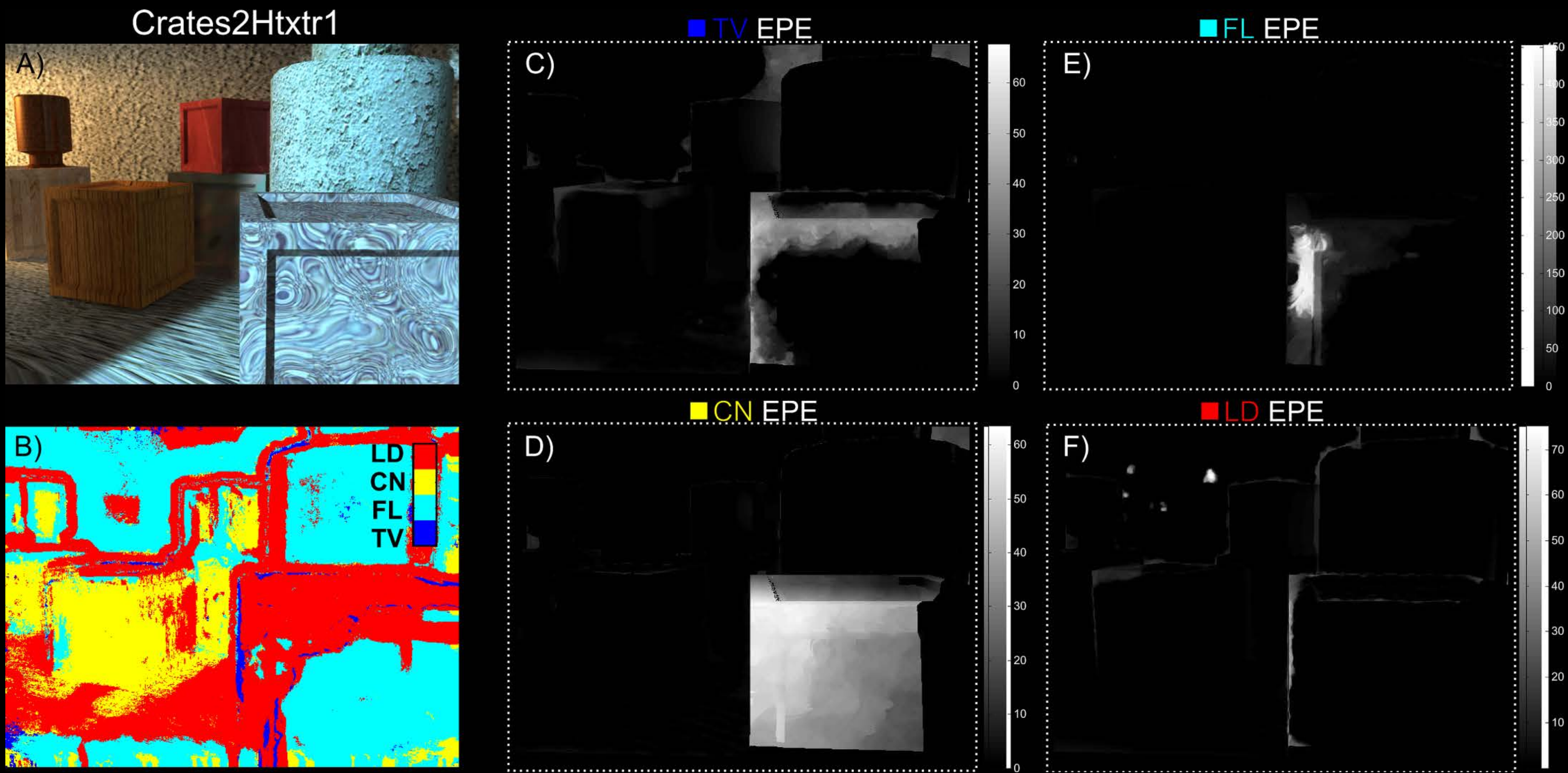

Average EPE
% Pixels



Average EPE
% Pixels
0.10
0.25
0.50
1
2
10
Opt Ordering


Input
Joint Confidence
X Confidence
Y Confidence

K Algorithms: OursKWay



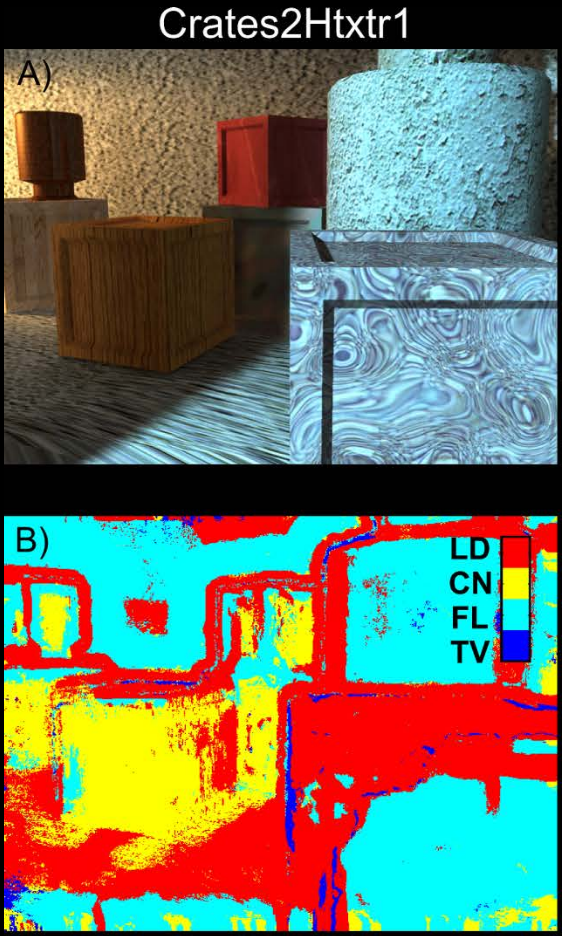
TV: Zach et al. DAGM 2007

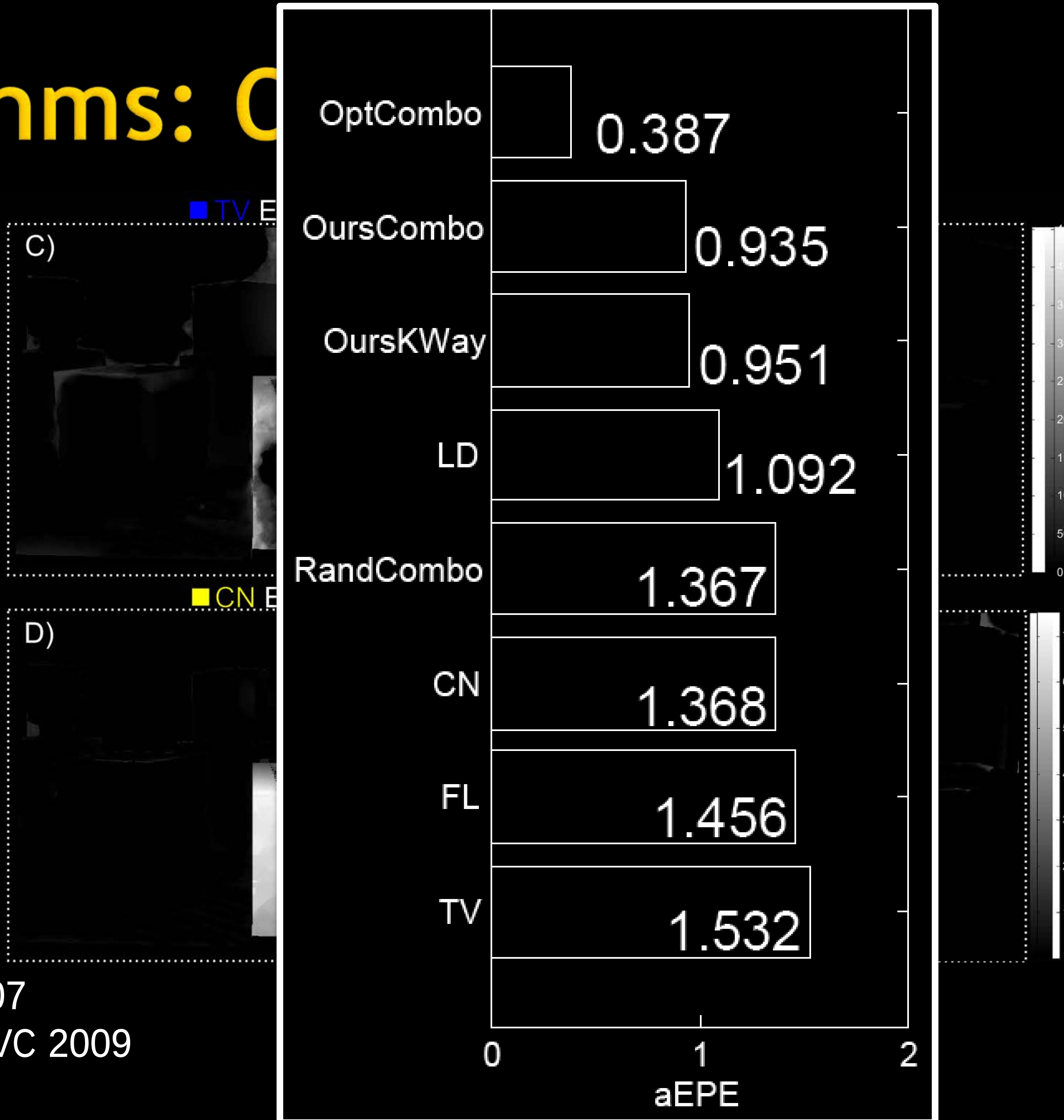
FL: Werlberger et al. BMVC 2009

CN: Sun et al. CVPR 2010

LD: Brox & Malik PAMI 2010

K Algorithms: C

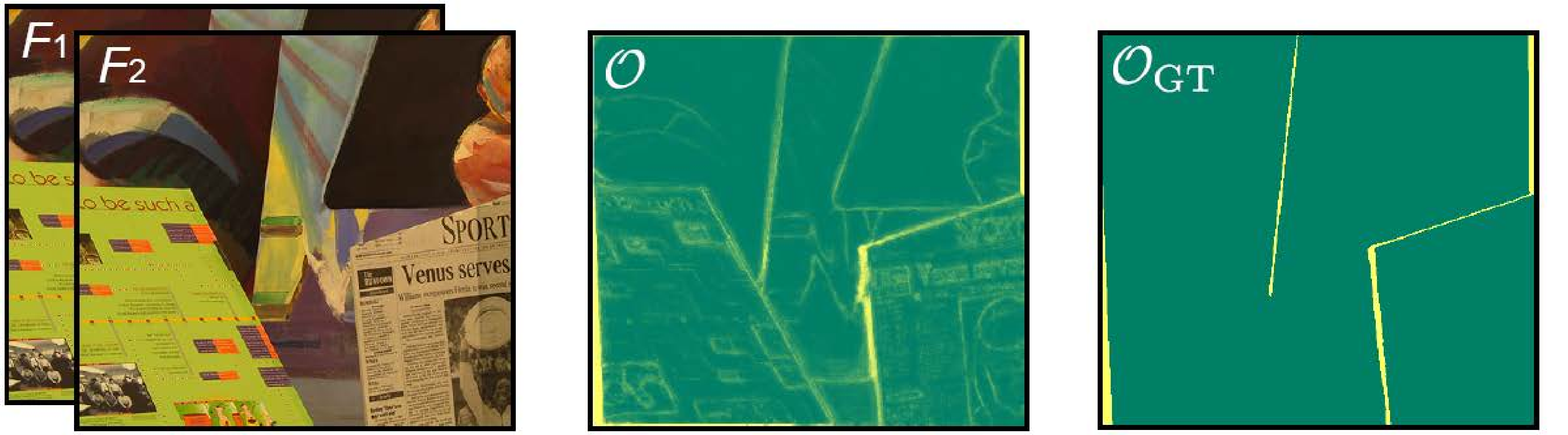




TV: Zach et al. DAGM 2007
FL: Werlberger et al. BMVC 2009
CN: Sun et al. CVPR 2010
LD: Brox & Malik PAMI 2010

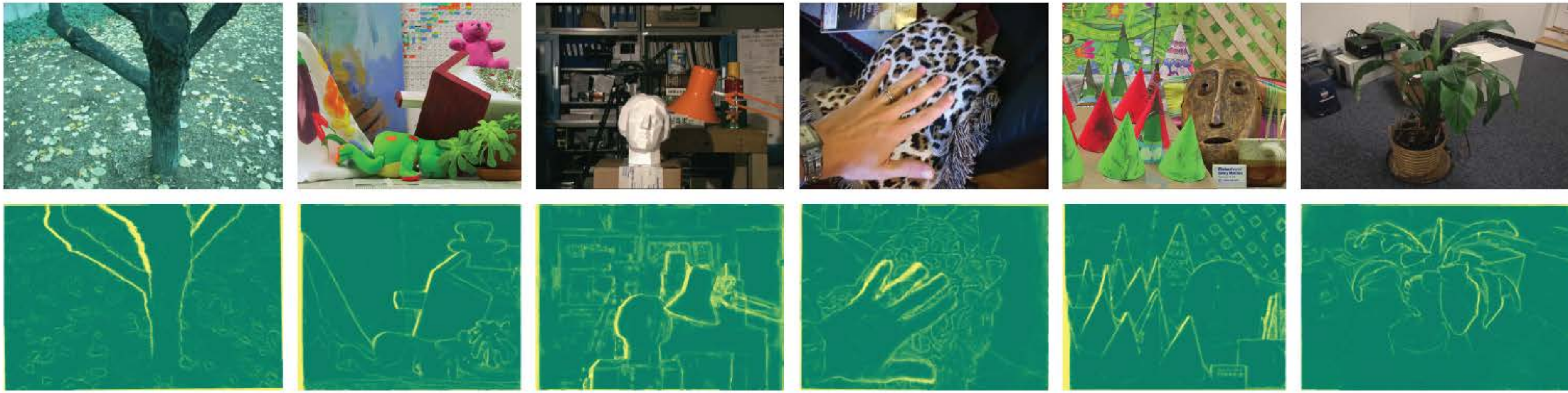
Example Application: Occlusion Regions

"Gets-occluded" posterior: $f(x_i) \rightarrow l_i = \{ ✓, ✗ \}$



Yellow indicates occlusion in frame two, Green shows visible pixels

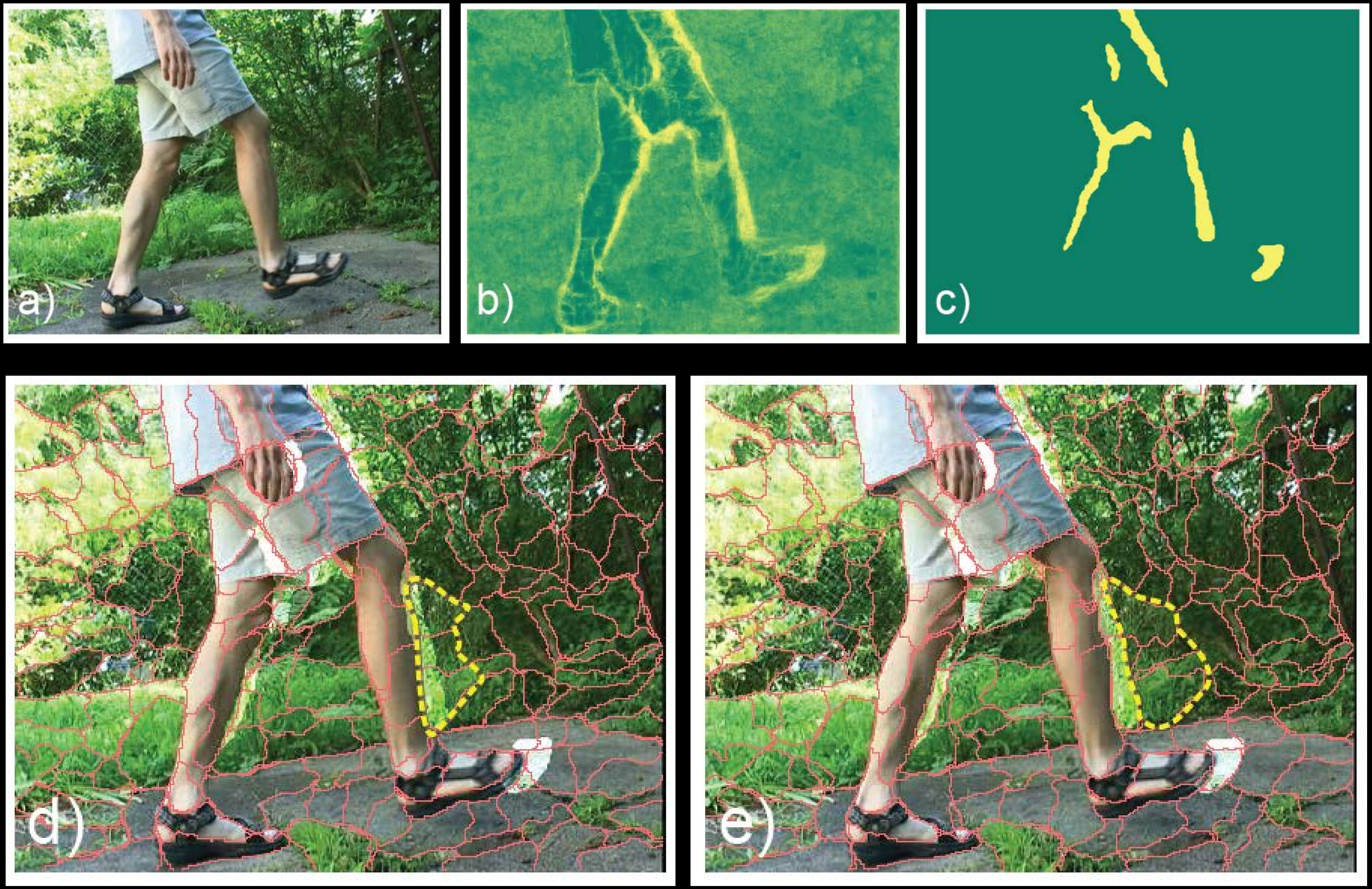
[Humayun et al. CVPR 2011]



Qualitative results of our occlusion-region classifier.

	Crates1	Crates2	Robot	Sponza1	Sponza2	Crates1txtr	Brickbox1t1	Brickbox2of	Mayan1	Text1
Single FGT	0.564	0.658	0.536	0.633	0.640	0.511	0.766	0.604	0.508	0.626
Lean CGT	0.826	0.758	0.851	0.945	0.814	0.981	0.979	0.641	0.952	0.973
Lean FGT	**0.950**	0.961	0.922	0.940	**0.929**	0.996	0.986	0.981	0.976	0.985
Ours CGT	0.744	0.682	0.876	**0.947**	0.812	0.991	0.989	0.618	0.973	0.982
Ours FGT	0.942	**0.969**	**0.936**	**0.947**	0.928	**0.997**	**0.992**	**0.991**	**0.986**	**0.991**

Leave-one-out scores reported as area under ROC curve.



Posterior probability of occlusion used as prior for superpixels.

But Pixels Aren’t Independent…

But Pixels Aren’t Independent…

Motion Models
(That Only Work Sometimes)

[Garcia Cifuentes et al. BMVC 2012]

Track + Predict Real World Motion

Data and code:

http://visual.cs.ucl.ac.uk/pubs/MotionModelPrediction

Does the Motion Model Matter?

	Br	CVel
Tracking robustness (10^{-2}):	42.3	43.2
± std dev random runs:	0.4	0.4

$$robustness = \frac{1}{N}\sum_{i=1}^{N} \triangle_{\theta}(p_i, \hat{p}_i)$$

$$\triangle_{\theta}(p_i, \hat{p}_i) = \begin{cases} 1 & \text{if } |x_i - \hat{x}_i| < \theta \text{ and } |y_i - \hat{y}_i| < \theta \\ 0 & \text{otherwise.} \end{cases}$$

Does the Motion Model Matter?

	Br	CVel		best{Br, CVel}
Tracking robustness (10^{-2}):	42.3	43.2	VS.	49.3
± std dev random runs:	0.4	0.4		0.6

$$robustness = \frac{1}{N}\sum_{i=1}^{N} \triangle_\theta(p_i, \hat{p}_i)$$

$$\triangle_\theta(p_i, \hat{p}_i) = \begin{cases} 1 & \text{if } |x_i - \hat{x}_i| < \theta \text{ and } |y_i - \hat{y}_i| < \theta \\ 0 & \text{otherwise.} \end{cases}$$

Recipe for Using Motion Models

1) Extract clip from video

2) Compute feature vector (BoW, *actions*)

3) Classify using pre-trained SVM → MM_i
   - Trained with inspection OR performance labels

4) Track clips using Particle Filter & MM_i

Fixed vs. Inferred Motion Category

Fixed for all videos	Br	CVel	TRight	TLeft	Fwd	Bwd
robustness (10^{-2}):	42.3	43.2	37.9	37.2	44.7	43.7
± std dev:	0.4	0.4	0.7	0.5	0.2	0.1

Fixed vs. Inferred Motion Category

Fixed for all videos	Br	CVel	TRight	TLeft	Fwd	Bwd
robustness (10^{-2}):	42.3	43.2	37.9	37.2	44.7	43.7
± std dev:	0.4	0.4	0.7	0.5	0.2	0.1

vs.

Chosen by classifier

robustness (10^{-2}): 51.9

± std dev: 0.1

Fixed vs. Inferred Motion Category

Fixed for all videos	Br	CVel	TRight	TLeft	Fwd	Bwd
robustness (10^{-2}):	42.3	43.2	37.9	37.2	44.7	43.7
± std dev:	0.4	0.4	0.7	0.5	0.2	0.1

vs.

Ideal predictions

best{all}	inspection labels
56.1	52.6
0.4	0.2

Chosen by classifier

robustness (10^{-2}): 51.9

± std dev: 0.1

Video: Infer Which Motion Model

Be like Wayne Gretzky!

Photo from
[Oilersnation]

Summary

- Narrow "specialist" tracking algorithms are great, <u>**if**</u> we can
  - Observe them in action
  - Learn where to trust them

- Much left to do:
  - Track + predict more motions
  - Cope with camouflage / invisibility
  - Get accurate/faster with experience

Thank you!

See visual.cs.ucl.ac.uk for code + data.

Reminder: We're looking for PhD students!