

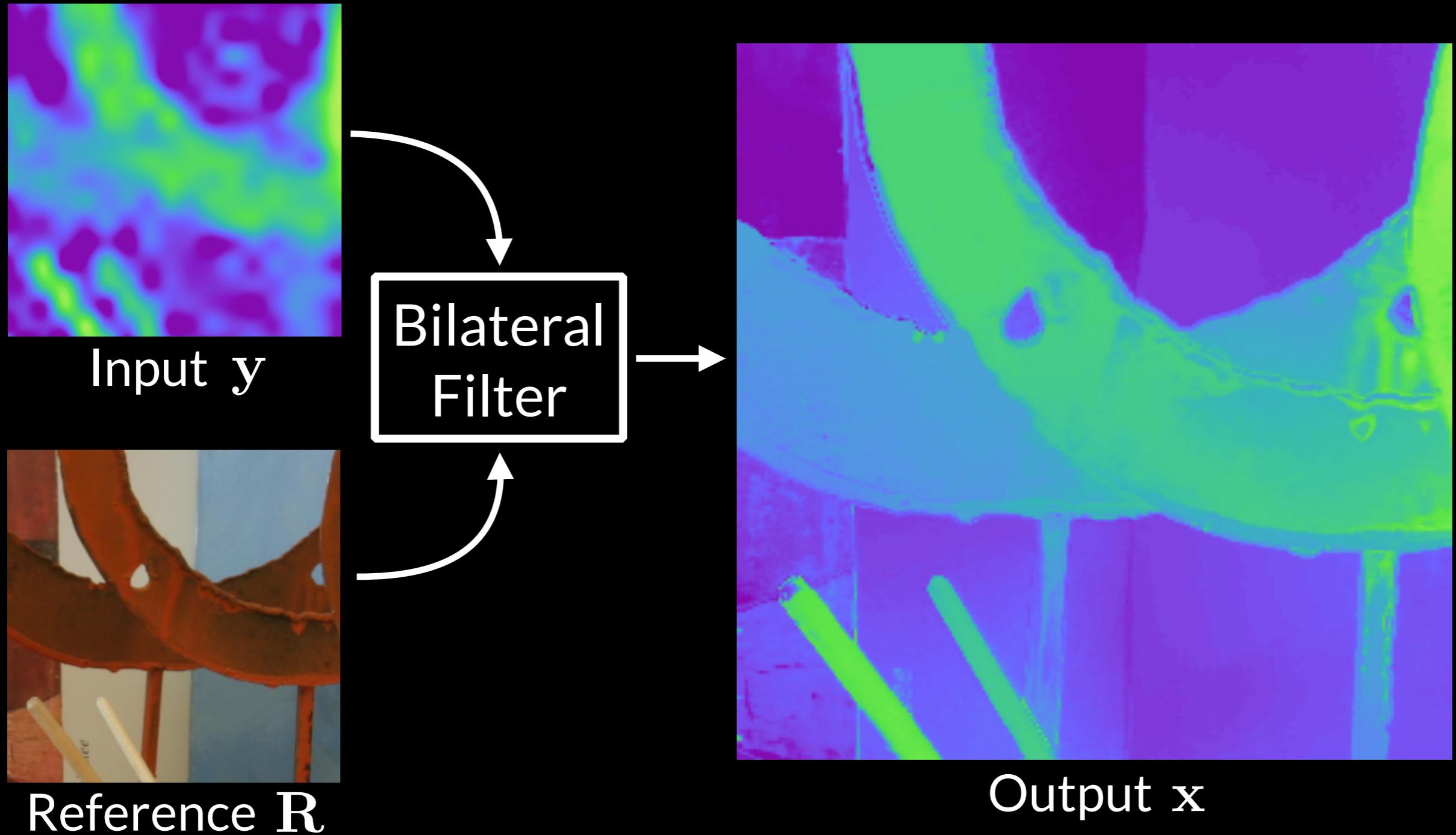
The Fast Bilateral Solver

Jonathan T. Barron
Google Research

Ben Poole
Stanford University

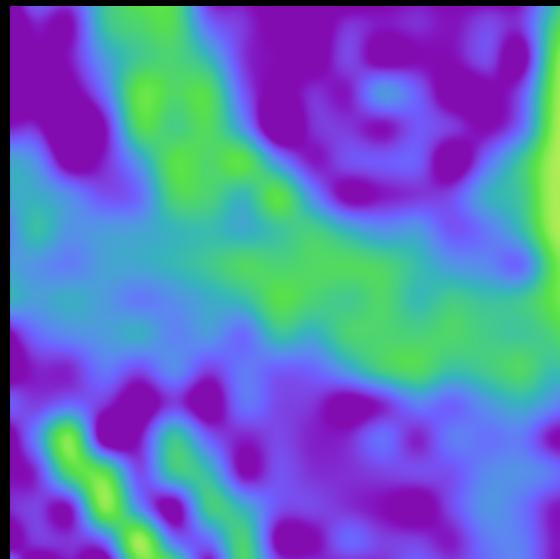
ECCV 2016

Bilateral Filter



Blur the input while respecting edges in the reference image.

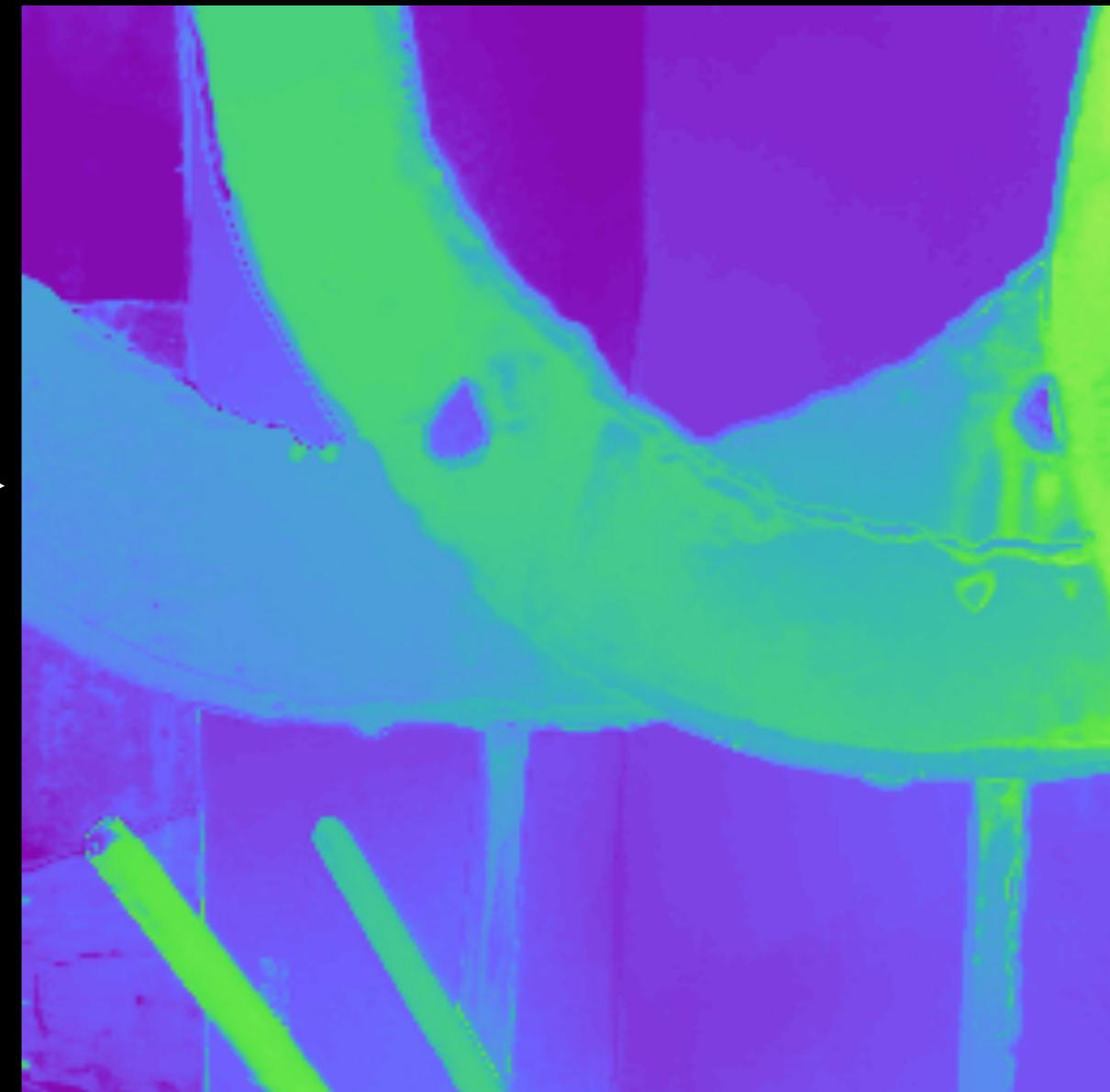
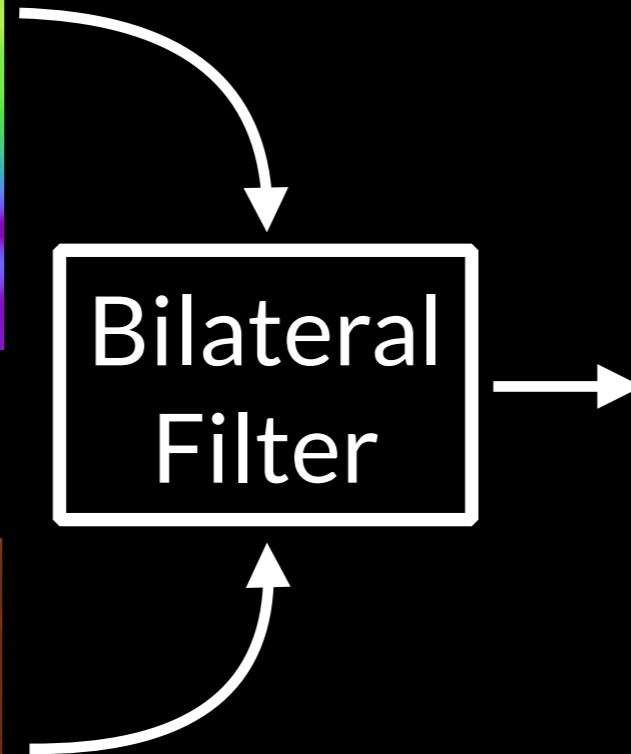
Bilateral Filter



Input y



Reference R

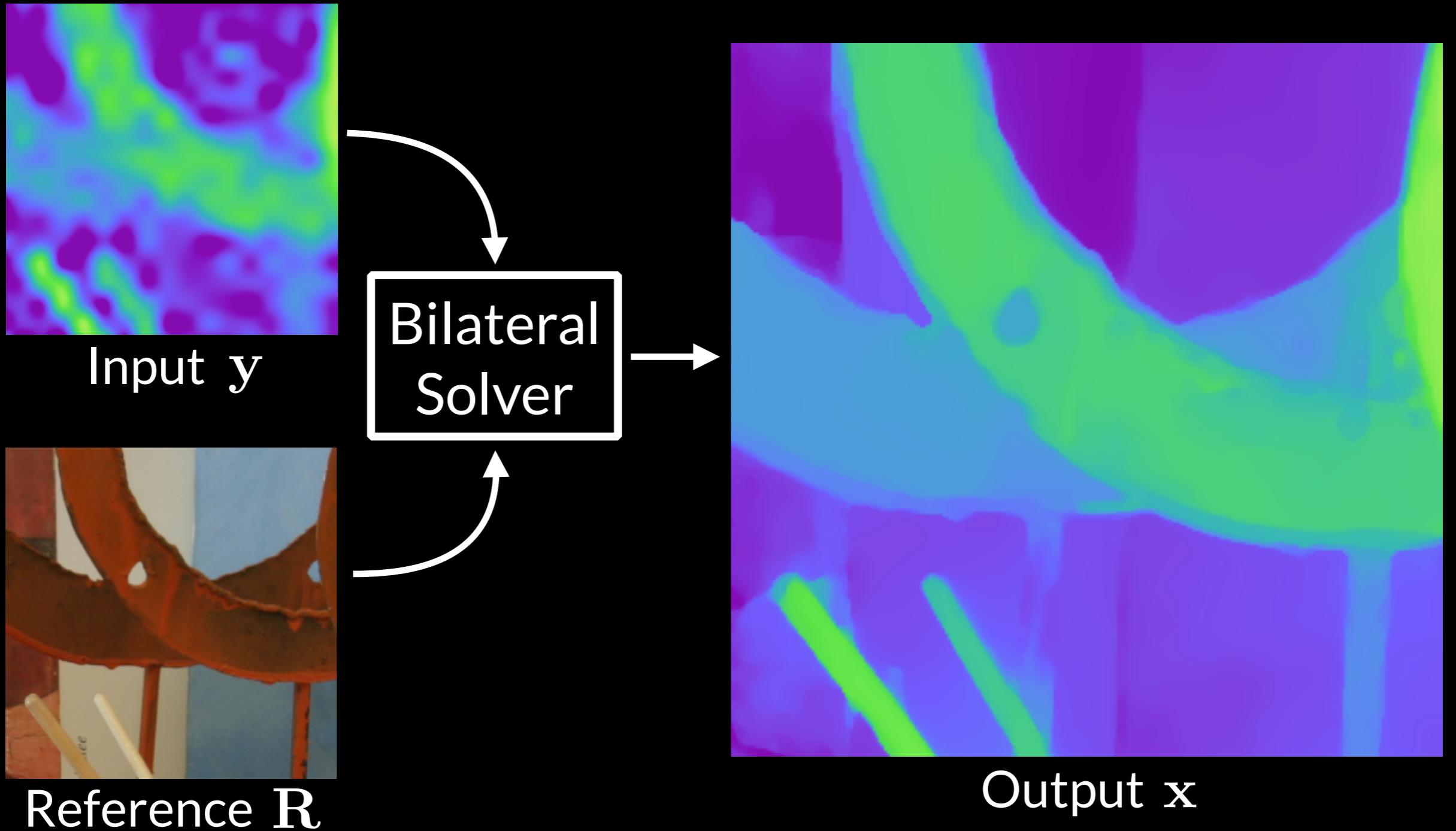


Output x

$$\mathbf{x} \leftarrow \mathbf{W}\mathbf{y}$$

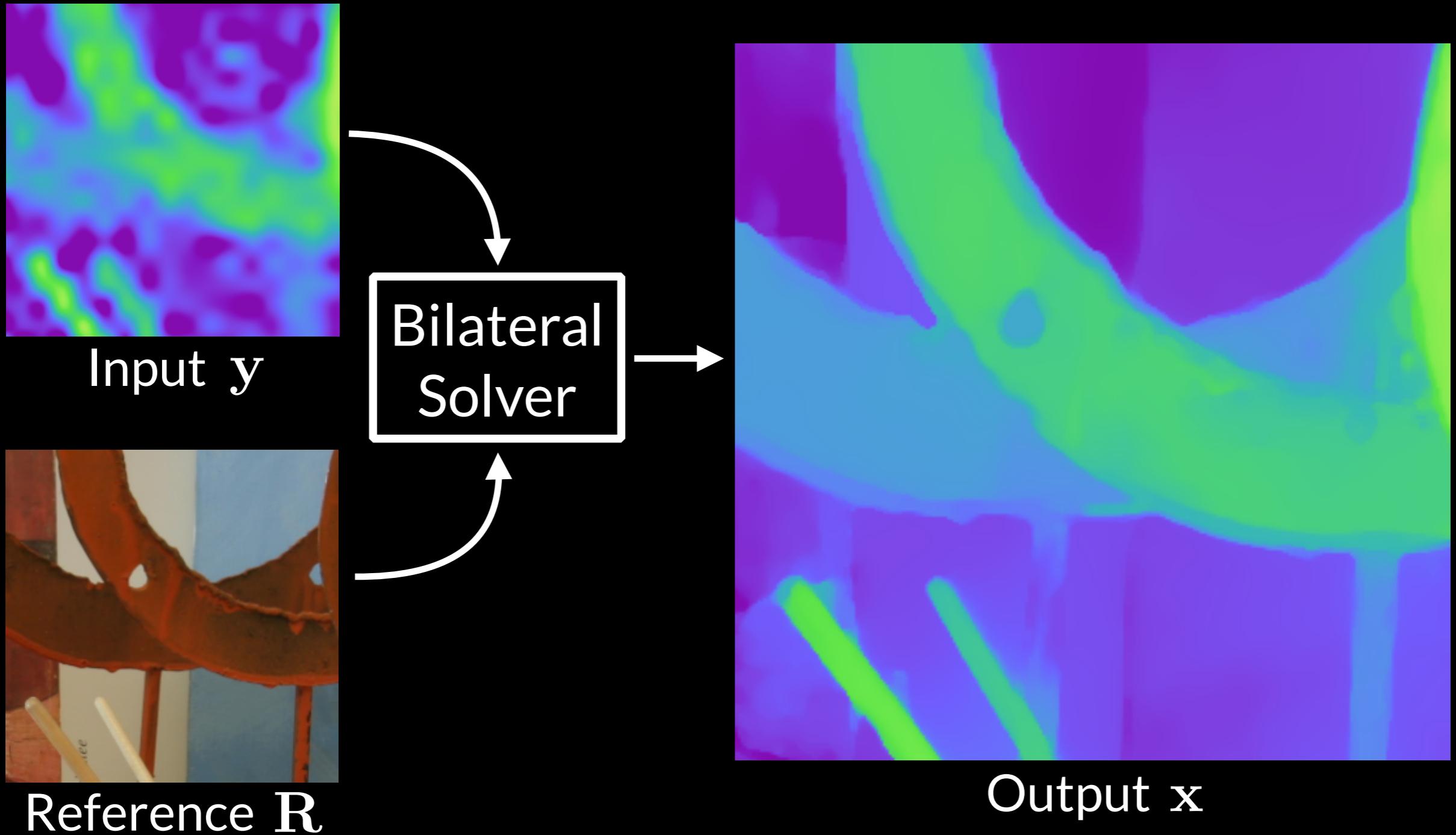
$$W_{i,j} = \exp \left(- \sum_d \frac{(R_{i,d} - R_{j,d})^2}{2\sigma_d^2} \right)$$

Bilateral Solver



Find the image that is as smooth as possible with respect to the reference image, and as close as possible to the input.

Bilateral Solver



$$\mathbf{x} \leftarrow \arg \min_{\mathbf{x}} \frac{\lambda}{2} \sum_{i,j} W_{i,j} (x_i - x_j)^2 + \sum_i (x_i - y_i)^2$$

Bilateral Solver

*The bilateral filter is
one gradient descent step
in the bilateral solver.*

(assuming: step size = 1, $\lambda = 1$, initial state = y)

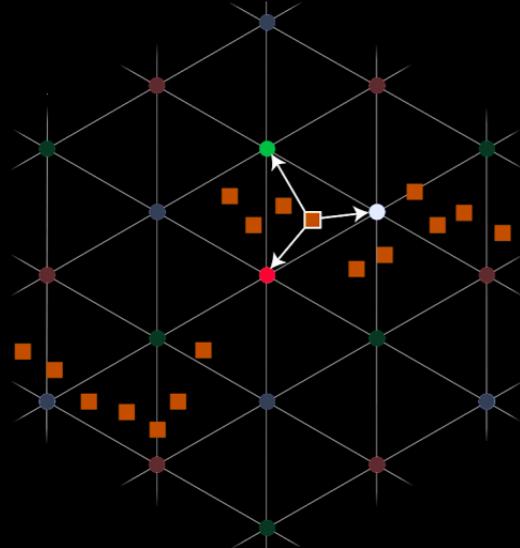
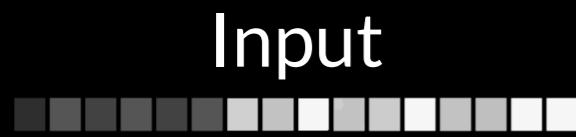
Bilateral Solver

Fast - Can be reformulated to be roughly as fast as a fast bilateral filter.

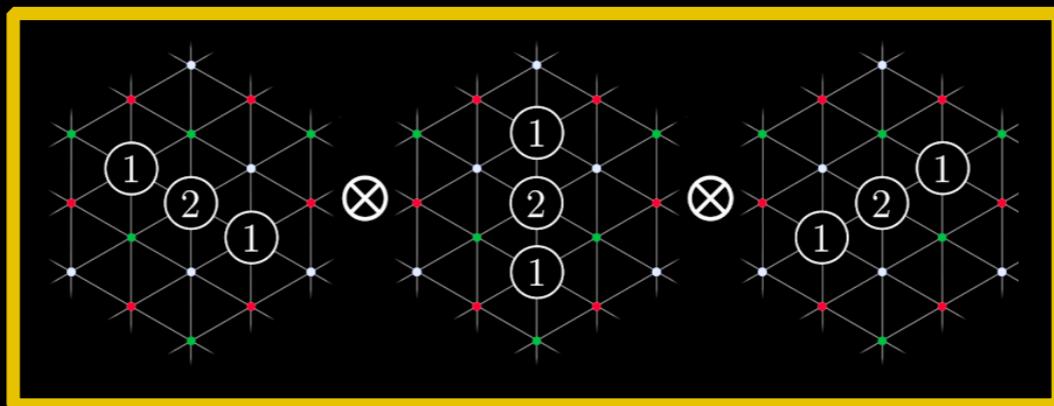
General - Can be easily generalized into weighted, robust, low rank, and differentiable variants.

Effective - Performs well on a variety of tasks:
depth superresolution
colorization
stereo
semantic segmentation, etc.

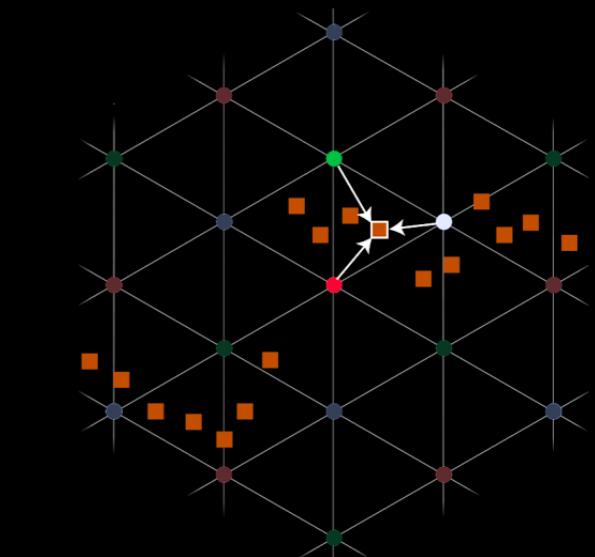
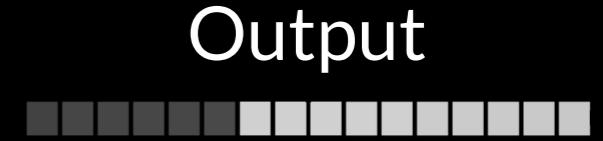
Review: Bilateral-Space Optimization



Splat
resample from pixels
into “bilateral-space”



Blur
apply a series of blurs in
bilateral-space



Slice
resample back into
pixel-space

Adams et al, Eurographics 2010

We can solve optimization problems based on
bilateral kernels in this “bilateral space”

Barron et al, CVPR 2015
Märki et al, CVPR 2016

Bilateral Solver

If we know that:

$$W_{i,j} = \exp\left(-\sum_d \frac{(R_{i,d} - R_{j,d})^2}{2\sigma^2}\right)$$

We can show that:

$$\mathbf{W} = \mathbf{S}^T \mathbf{D}_m^{-1} \mathbf{D}_n \mathbf{B} \mathbf{D}_n \mathbf{D}_m^{-1} \mathbf{S}$$

Bilateral Solver

If we know that:

$$W_{i,j} = \exp\left(-\sum_d \frac{(R_{i,d} - R_{j,d})^2}{2\sigma^2}\right)$$

We can show that:

$$\mathbf{W} = \mathbf{S}^T \mathbf{D}_m^{-1} \mathbf{D}_n \mathbf{B} \mathbf{D}_n \mathbf{D}_m^{-1} \mathbf{S}$$

$$\mathbf{x} \leftarrow \arg \min_{\mathbf{x}} \frac{\lambda}{2} \sum_{i,j} W_{i,j} (x_i - x_j)^2 + \sum_i (x_i - y_i)^2$$

Bilateral Solver

If we know that:

$$W_{i,j} = \exp\left(-\sum_d \frac{(R_{i,d} - R_{j,d})^2}{2\sigma^2}\right)$$

We can show that:

$$\mathbf{W} = \mathbf{S}^T \mathbf{D}_m^{-1} \mathbf{D}_n \mathbf{B} \mathbf{D}_n \mathbf{D}_m^{-1} \mathbf{S}$$

$$\mathbf{A} = \lambda (\mathbf{D}_m - \mathbf{D}_n \mathbf{B} \mathbf{D}_n) + \mathbf{S} \vec{\mathbf{l}}$$

$$\mathbf{x} \leftarrow \mathbf{S}^T (\mathbf{A}^{-1} (\mathbf{S} \mathbf{y}))$$

Bilateral Solver

If we know that:

$$W_{i,j} = \exp\left(-\sum_d \frac{(R_{i,d} - R_{j,d})^2}{2\sigma^2}\right)$$

We can show that:

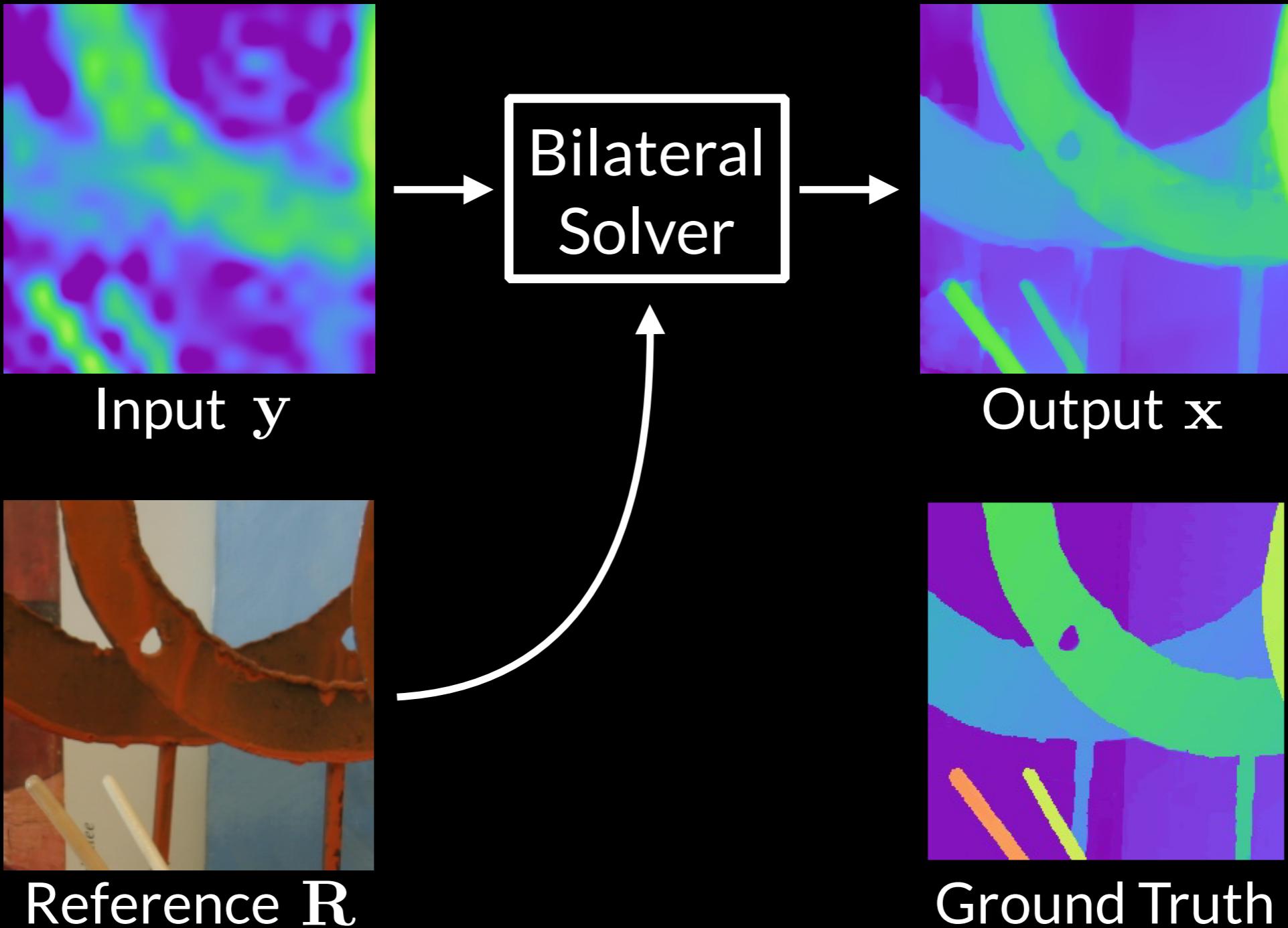
$$\mathbf{W} = \mathbf{S}^T \mathbf{D}_m^{-1} \mathbf{D}_n \mathbf{B} \mathbf{D}_n \mathbf{D}_m^{-1} \mathbf{S}$$

$$\mathbf{A} = \lambda (\mathbf{D}_m - \mathbf{D}_n \mathbf{B} \mathbf{D}_n) + \mathbf{S} \vec{\mathbf{l}}$$

$$\mathbf{x} \leftarrow \mathbf{S}^T (\mathbf{A}^{-1} (\mathbf{S} \mathbf{y}))$$

(see paper for details)

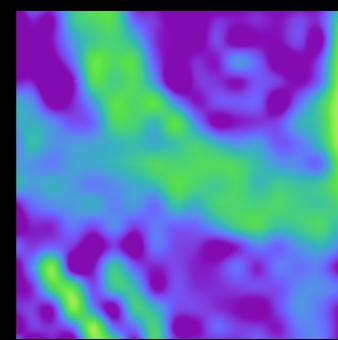
Depth Superresolution



task from Ferstl et al ICCV 2013, data from the Middlebury stereo dataset



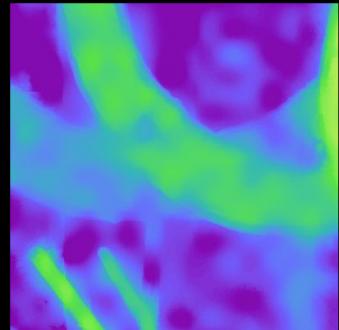
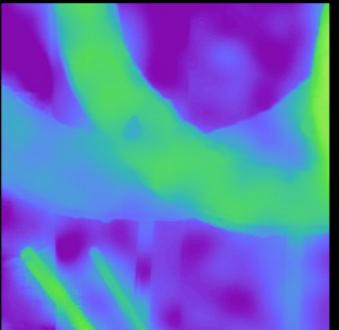
Reference Image



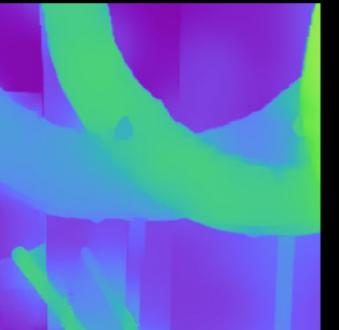
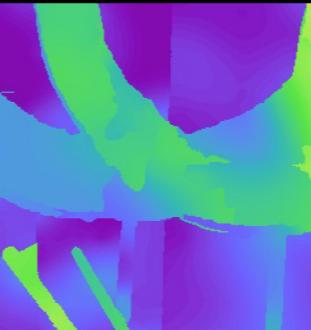
Input Depth



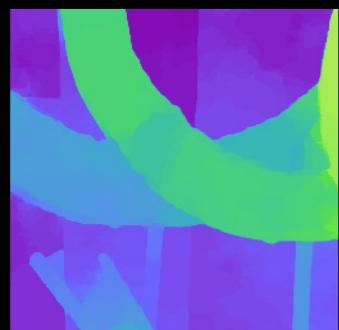
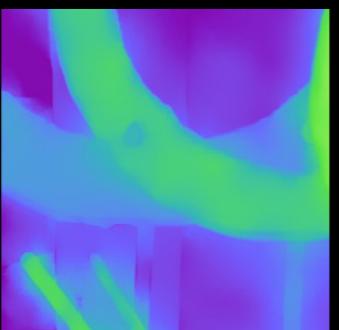
True Depth

Chan *et al.*

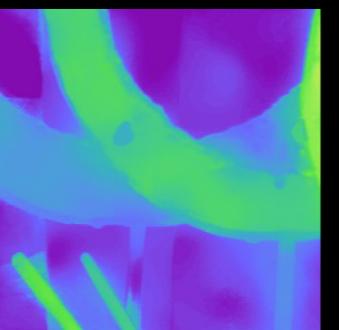
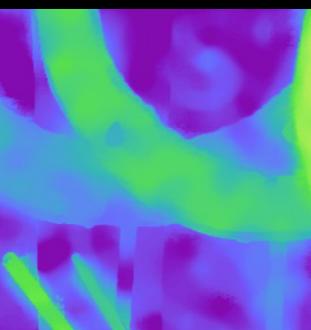
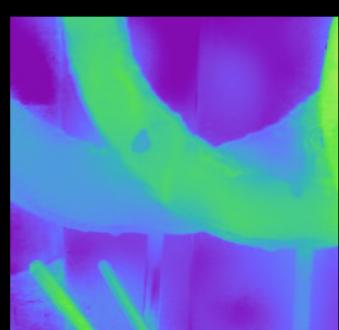
GF

Min *et al.*

† Lu

Park *et al.*

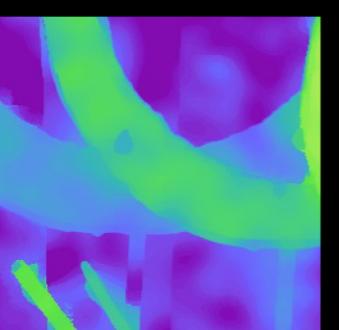
DT

Ma *et al.*Zhang *et al.*

FGF



Yang 2015



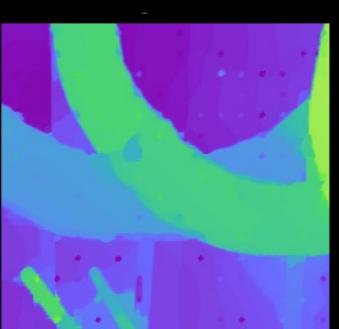
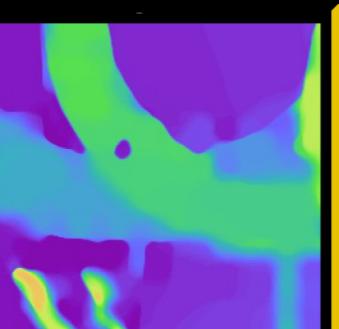
Yang 2007



WLS



JB

Ferstl *et al.*

† Liet al.

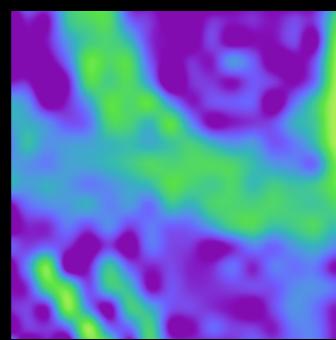


BS

Method	Err	Time (sec)
Nearest Neighbor	7.26	0.003
Bicubic	5.91	0.007
Kiechle <i>et al.</i>	5.86	450
Bilinear	5.16	0.004
Liu <i>et al.</i>	5.10	16.60
Shen <i>et al.</i>	4.24	31.48
Diebel & Thrun	3.98	—
Chan <i>et al.</i>	3.83	3.02
Guided Filter	3.76	23.89
Min <i>et al.</i>	3.74	0.383
Lu & Forsyth	3.69	20
Park <i>et al.</i>	3.61	24.05
Domain Transform	3.56	0.021
Ma <i>et al.</i>	3.49	18
GuidedFilter (Matlab)	3.47	0.434
Zhang <i>et al.</i>	3.45	1.346
Fast Guided Filter	3.41	0.225
Yang 2015	3.41	0.304
Yang <i>et al.</i> 2007	3.25	—
Farbman <i>et al.</i>	3.19	6.11
Joint Bilateral Upsample	3.14	1.98
Ferstl <i>et al.</i>	2.93	140
Li <i>et al.</i>	2.56	700
Kwon <i>et al.</i>	1.21	300
Bilateral Solver	2.70	0.234



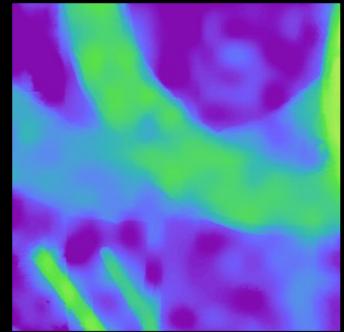
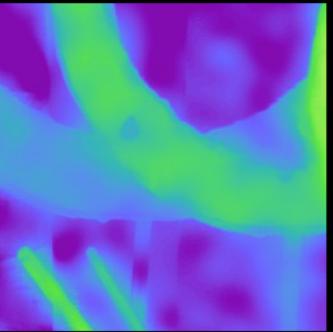
Reference Image



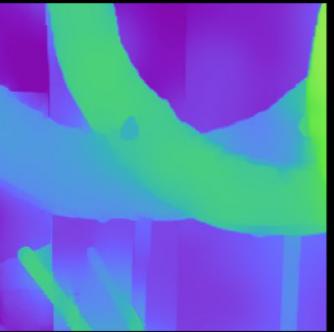
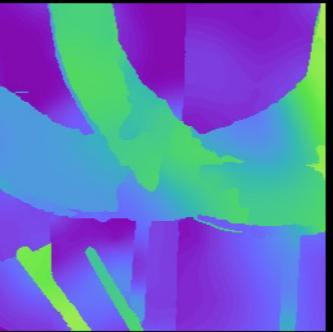
Input Depth



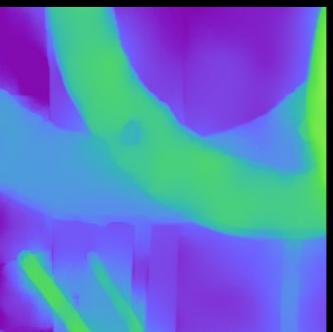
True Depth

Chan *et al.*

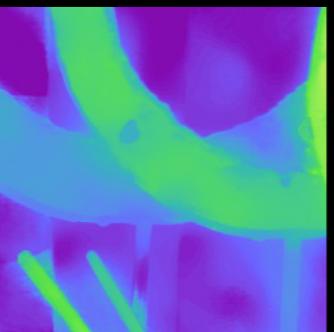
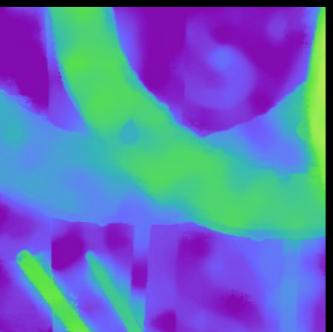
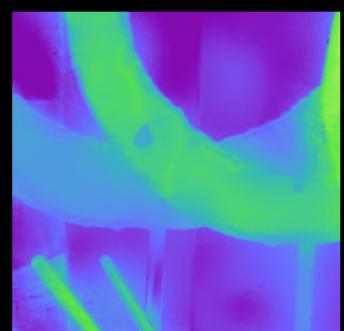
GF

Min *et al.*

† Lu

Park *et al.*

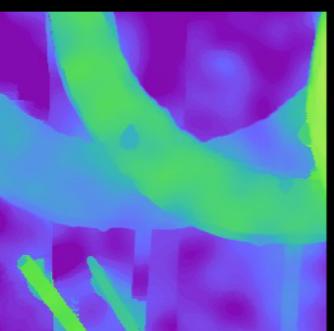
DT

Ma *et al.*Zhang *et al.*

FGF



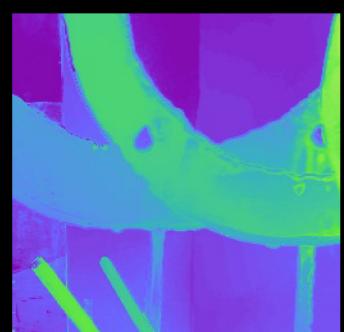
Yang 2015



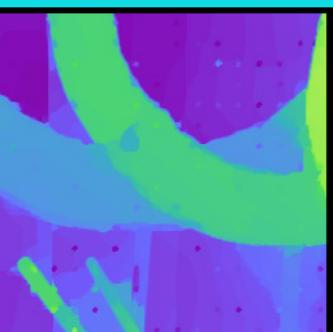
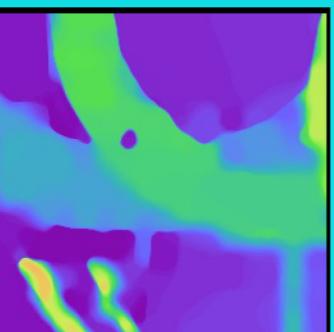
Yang 2007



WLS



JB

Ferstl *et al.*

† Liet al.



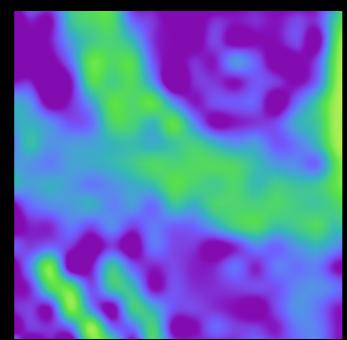
BS

Method

	Err	Time (sec)
Nearest Neighbor	7.26	0.003
Bicubic	5.91	0.007
Kiechle <i>et al.</i>	5.86	450
Bilinear	5.16	0.004
Liu <i>et al.</i>	5.10	16.60
Shen <i>et al.</i>	4.24	31.48
Diebel & Thrun	3.98	—
Chan <i>et al.</i>	3.83	3.02
Guided Filter	3.76	23.89
Min <i>et al.</i>	3.74	0.383
Lu & Forsyth	3.69	20
Park <i>et al.</i>	3.61	24.05
Domain Transform	3.56	0.021
Ma <i>et al.</i>	3.49	18
GuidedFilter (Matlab)	3.47	0.434
Zhang <i>et al.</i>	3.45	1.346
Fast Guided Filter	3.41	0.225
Yang 2015	3.41	0.304
Yang <i>et al.</i> 2007	3.25	—
Farbman <i>et al.</i>	3.19	6.11
Joint Bilateral Upsample	3.14	1.98
Ferstl <i>et al.</i>	2.93	140
Li <i>et al.</i>	2.56	700
Kwon <i>et al.</i>	1.21	300
Bilateral Solver	2.70	0.234



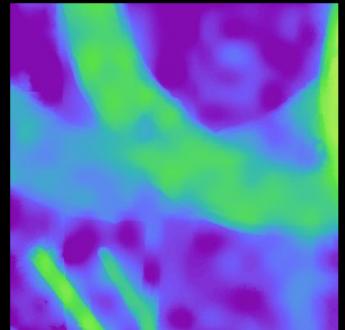
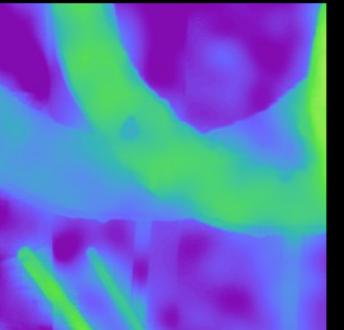
Reference Image



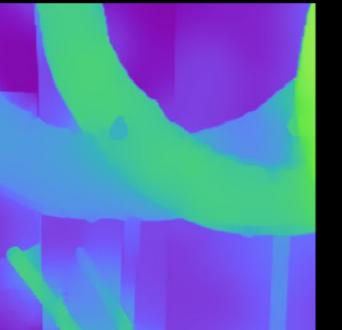
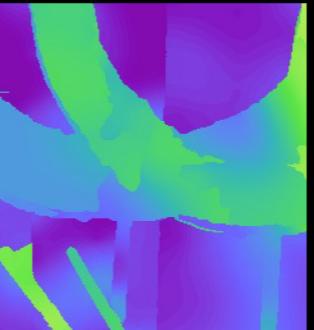
Input Depth



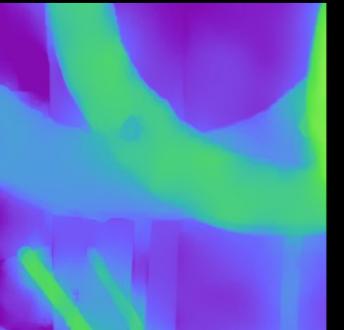
True Depth

Chan *et al.*

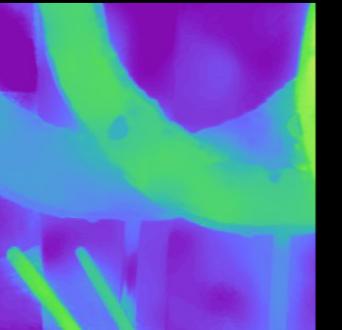
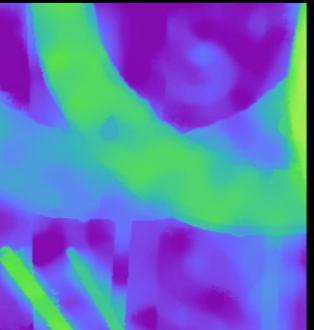
GF

Min *et al.*

† Lu

Park *et al.*

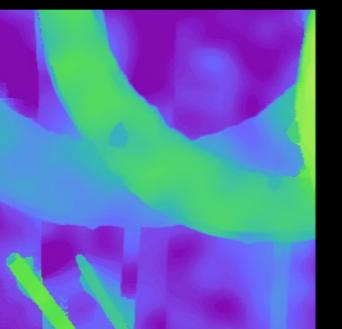
DT

Ma *et al.*Zhang *et al.*

FGF



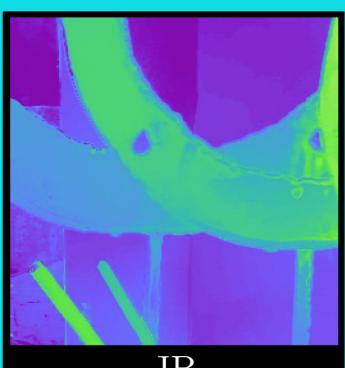
Yang 2015



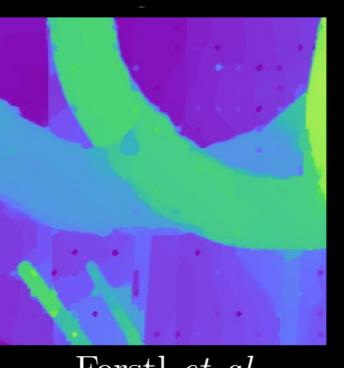
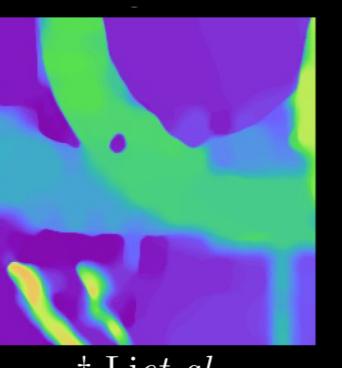
Yang 2007



WLS



JB

Ferstl *et al.*

† Liet al.

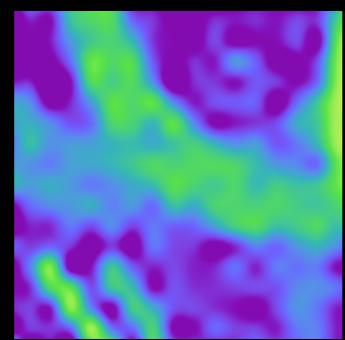


BS

Method	Err	Time (sec)
Nearest Neighbor	7.26	0.003
Bicubic	5.91	0.007
Kiechle <i>et al.</i>	5.86	450
Bilinear	5.16	0.004
Liu <i>et al.</i>	5.10	16.60
Shen <i>et al.</i>	4.24	31.48
Diebel & Thrun	3.98	—
Chan <i>et al.</i>	3.83	3.02
Guided Filter	3.76	23.89
Min <i>et al.</i>	3.74	0.383
Lu & Forsyth	3.69	20
Park <i>et al.</i>	3.61	24.05
Domain Transform	3.56	0.021
Ma <i>et al.</i>	3.49	18
GuidedFilter (Matlab)	3.47	0.434
Zhang <i>et al.</i>	3.45	1.346
Fast Guided Filter	3.41	0.225
Yang 2015	3.41	0.304
Yang <i>et al.</i> 2007	3.25	—
Farbman <i>et al.</i>	3.19	6.11
Joint Bilateral Upsample	3.14	1.98
Ferstl <i>et al.</i>	2.93	140
Li <i>et al.</i>	2.56	700
Kwon <i>et al.</i>	1.21	300
Bilateral Solver	2.70	0.234



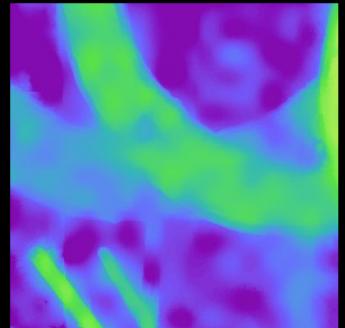
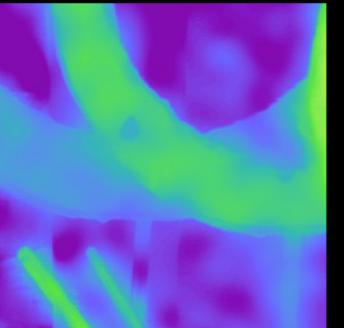
Reference Image



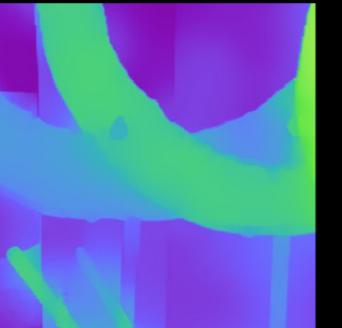
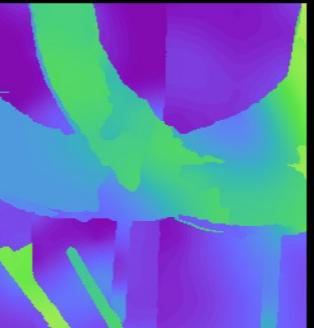
Input Depth



True Depth

Chan *et al.*

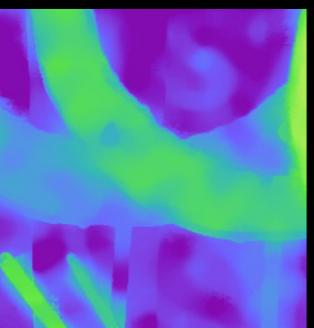
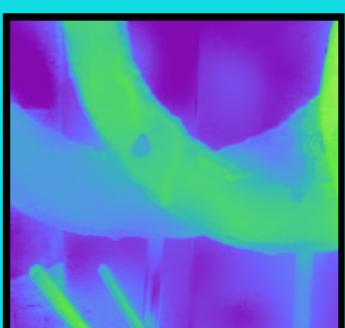
GF

Min *et al.*

† Lu

Park *et al.*

DT

Ma *et al.*Zhang *et al.*

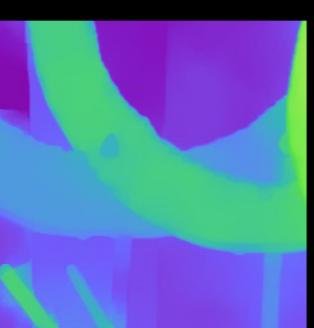
FGF



Yang 2015



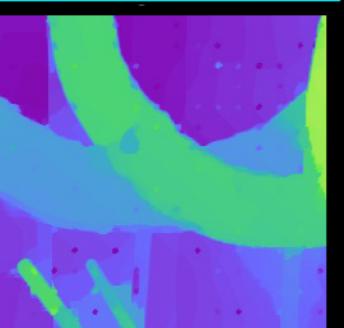
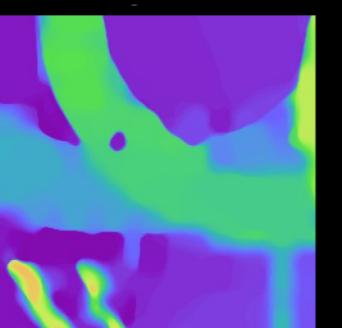
Yang 2007



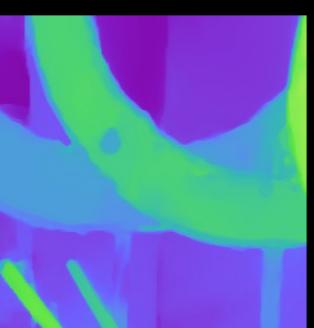
WLS



JB

Ferstl *et al.*

† Liet al.



BS

Method

Method	Err	Time (sec)
Nearest Neighbor	7.26	0.003
Bicubic	5.91	0.007
Kiechle <i>et al.</i>	5.86	450
Bilinear	5.16	0.004
Liu <i>et al.</i>	5.10	16.60
Shen <i>et al.</i>	4.24	31.48
Diebel & Thrun	3.98	—
Chan <i>et al.</i>	3.83	3.02
Guided Filter	3.76	23.89
Min <i>et al.</i>	3.74	0.383
Lu & Forsyth	3.69	20
Park <i>et al.</i>	3.61	24.05
Domain Transform	3.56	0.021
Ma <i>et al.</i>	3.49	18
GuidedFilter (Matlab)	3.47	0.434
Zhang <i>et al.</i>	3.45	1.346
Fast Guided Filter	3.41	0.225
Yang 2015	3.41	0.304
Yang <i>et al.</i> 2007	3.25	—
Farbman <i>et al.</i>	3.19	6.11
Joint Bilateral Upsample	3.14	1.98
Ferstl <i>et al.</i>	2.93	140
Li <i>et al.</i>	2.56	700
Kwon <i>et al.</i>	1.21	300
Bilateral Solver	2.70	0.234

Bilateral Solver

$$\mathbf{x} \leftarrow \arg \min_{\mathbf{x}} \frac{\lambda}{2} \sum_{i,j} W_{i,j} (x_i - x_j)^2 + \sum_i (x_i - y_i)^2$$

≡

$$\mathbf{A} = \lambda \left(\mathbf{D}_m - \mathbf{D}_n \mathbf{B} \mathbf{D}_n \right) + \mathbf{S} \vec{\mathbf{l}}$$

$$\mathbf{x} \leftarrow \mathbf{S}^T \left(\mathbf{A}^{-1} \left(\mathbf{S} \mathbf{y} \right) \right)$$

Weighted Bilateral Solver

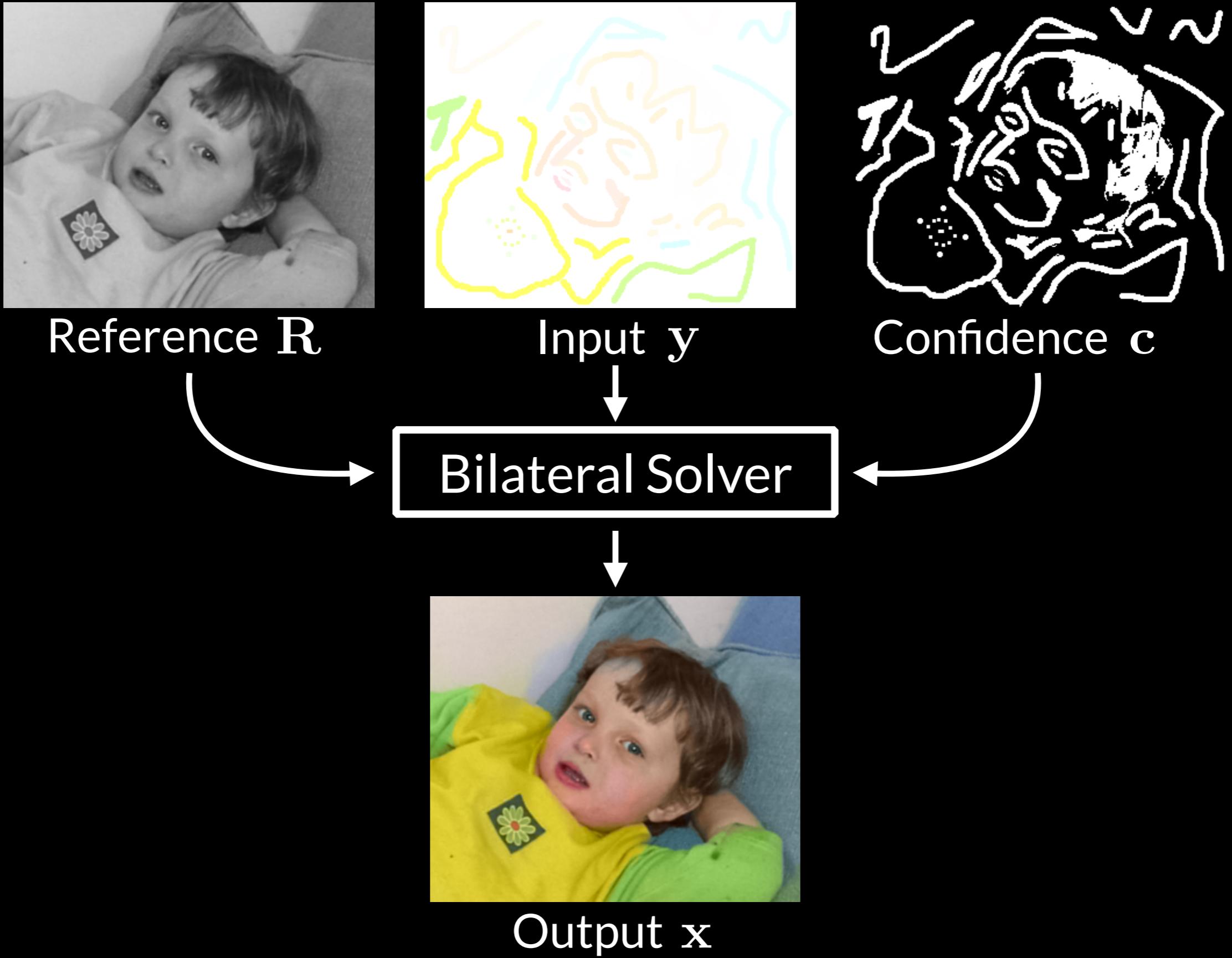
$$\mathbf{x} \leftarrow \arg \min_{\mathbf{x}} \frac{\lambda}{2} \sum_{i,j} W_{i,j} (x_i - x_j)^2 + \sum_i c_i (x_i - y_i)^2$$

≡

$$\mathbf{A} = \lambda (\mathbf{D}_m - \mathbf{D}_n \mathbf{B} \mathbf{D}_n) + \mathbf{S} \mathbf{c}$$

$$\mathbf{x} \leftarrow \mathbf{S}^T \left(\mathbf{A}^{-1} (\mathbf{S} \mathbf{y}) \right)$$

Weighted Bilateral Solver



Weighted Bilateral Solver



Bilateral Solver
0.85 sec/megapixel



Levin *et al*
81.0 sec/megapixel

95× speedup

Weighted Bilateral Solver

$$\mathbf{x} \leftarrow \arg \min_{\mathbf{x}} \frac{\lambda}{2} \sum_{i,j} W_{i,j} (x_i - x_j)^2 + \sum_i c_i (x_i - y_i)^2$$

≡

$$\mathbf{A} = \lambda (\mathbf{D}_m - \mathbf{D}_n \mathbf{B} \mathbf{D}_n) + \mathbf{S} \mathbf{c}$$

$$\mathbf{x} \leftarrow \mathbf{S}^T (\mathbf{A}^{-1} (\mathbf{S} \mathbf{y}))$$

Robust Bilateral Solver

$$\mathbf{x} \leftarrow \arg \min_{\mathbf{x}} \frac{\lambda}{2} \sum_{i,j} W_{i,j} (x_i - x_j)^2 + \sum_i \rho(x_i - y_i)$$

≡

while not converged :

$$\mathbf{c} \leftarrow \frac{\rho'(\mathbf{x} - \mathbf{y})}{\mathbf{x} - \mathbf{y}}$$

$$\mathbf{A} = \lambda (\mathbf{D}_m - \mathbf{D}_n \mathbf{B} \mathbf{D}_n) + \mathbf{S} \mathbf{c}$$

$$\mathbf{x} \leftarrow \mathbf{S}^T (\mathbf{A}^{-1} (\mathbf{S} \mathbf{y}))$$

Robust Bilateral Solver

$$\mathbf{x} \leftarrow \arg \min_{\mathbf{x}} \frac{\lambda}{2} \sum_{i,j} W_{i,j} (x_i - x_j)^2 + \sum_i \rho(x_i - y_i)$$

≡

while not converged :

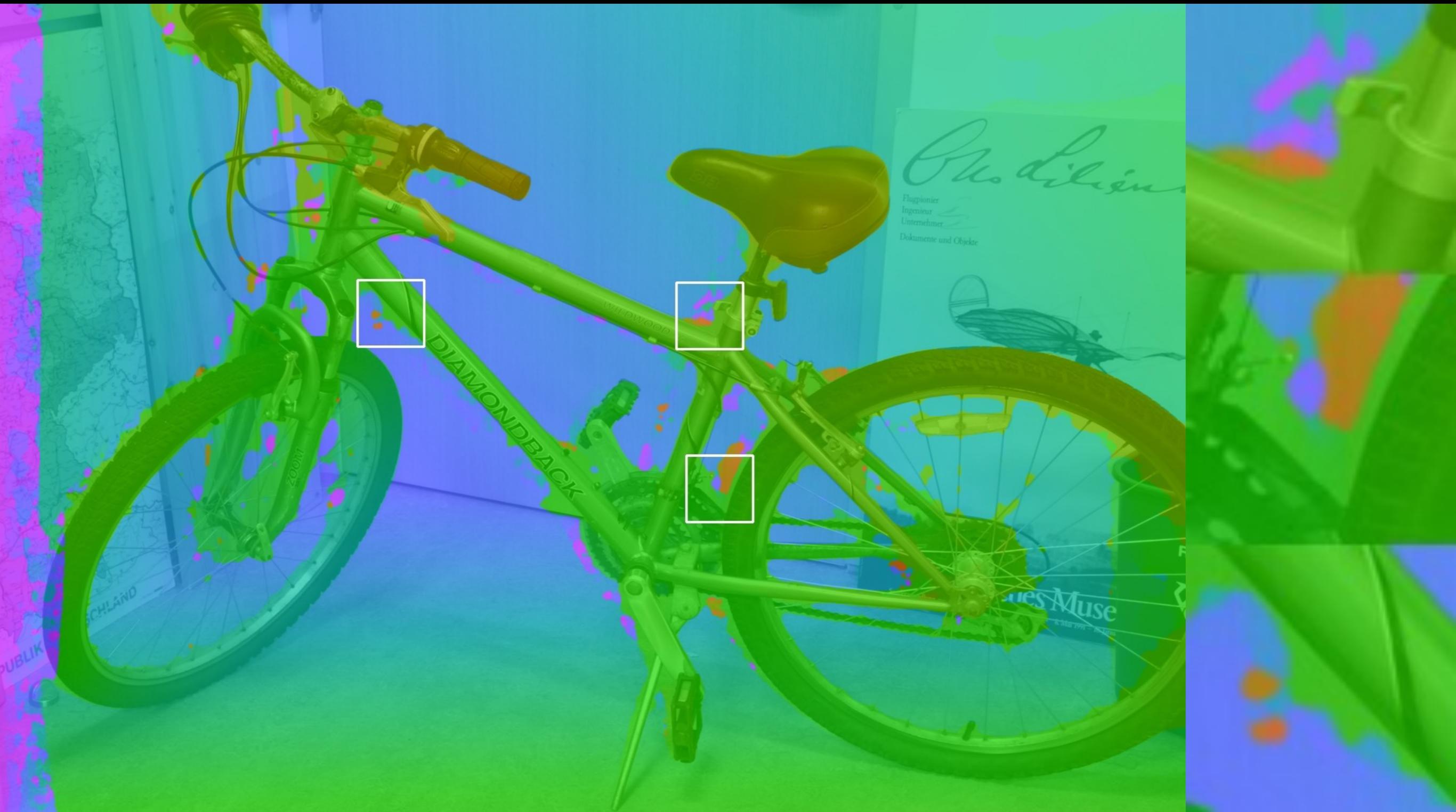
$$\mathbf{c} \leftarrow \frac{\rho'(\mathbf{x} - \mathbf{y})}{\mathbf{x} - \mathbf{y}}$$

$$\mathbf{A} = \lambda (\mathbf{D}_m - \mathbf{D}_n \mathbf{B} \mathbf{D}_n) + \mathbf{S} \mathbf{c}$$

$$\mathbf{x} \leftarrow \mathbf{S}^T (\mathbf{A}^{-1} (\mathbf{S} \mathbf{y}))$$

} can be
done in
bilateral
space

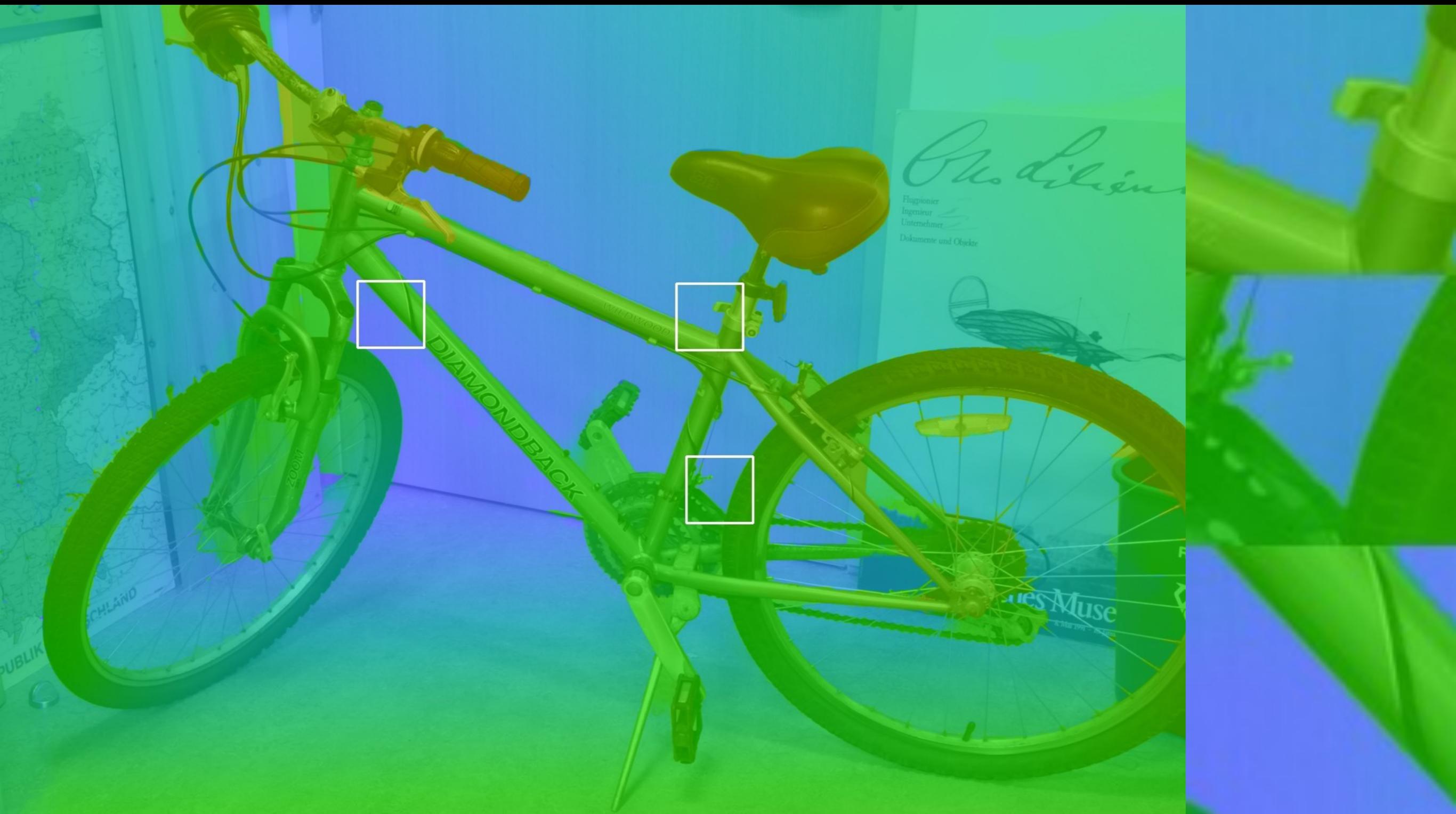
Stereo



MC-CNN

Zbontar & LeCun, CVPR 2015

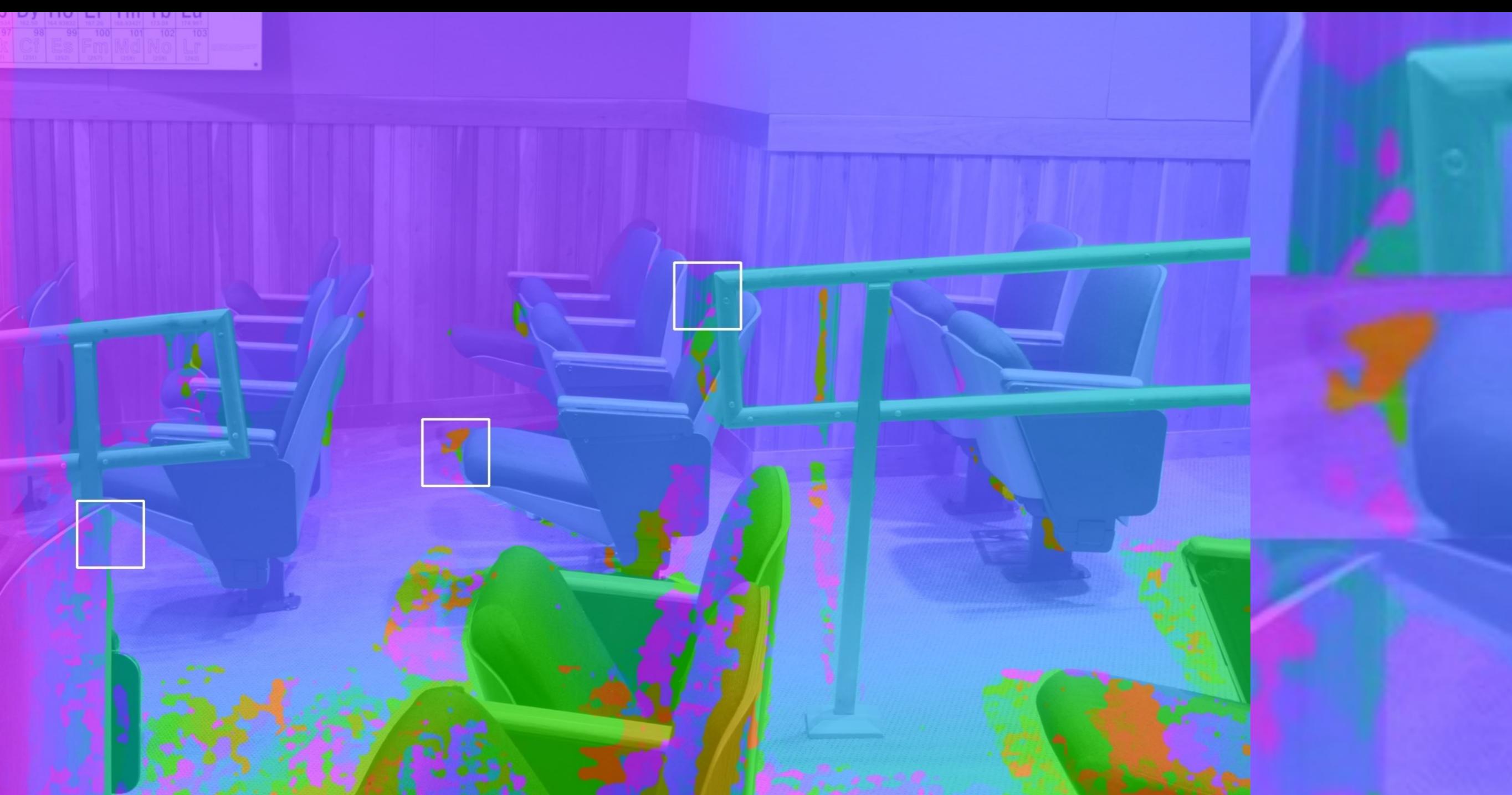
Stereo



MC-CNN + Robust Bilateral Solver

Zbontar & LeCun, CVPR 2015

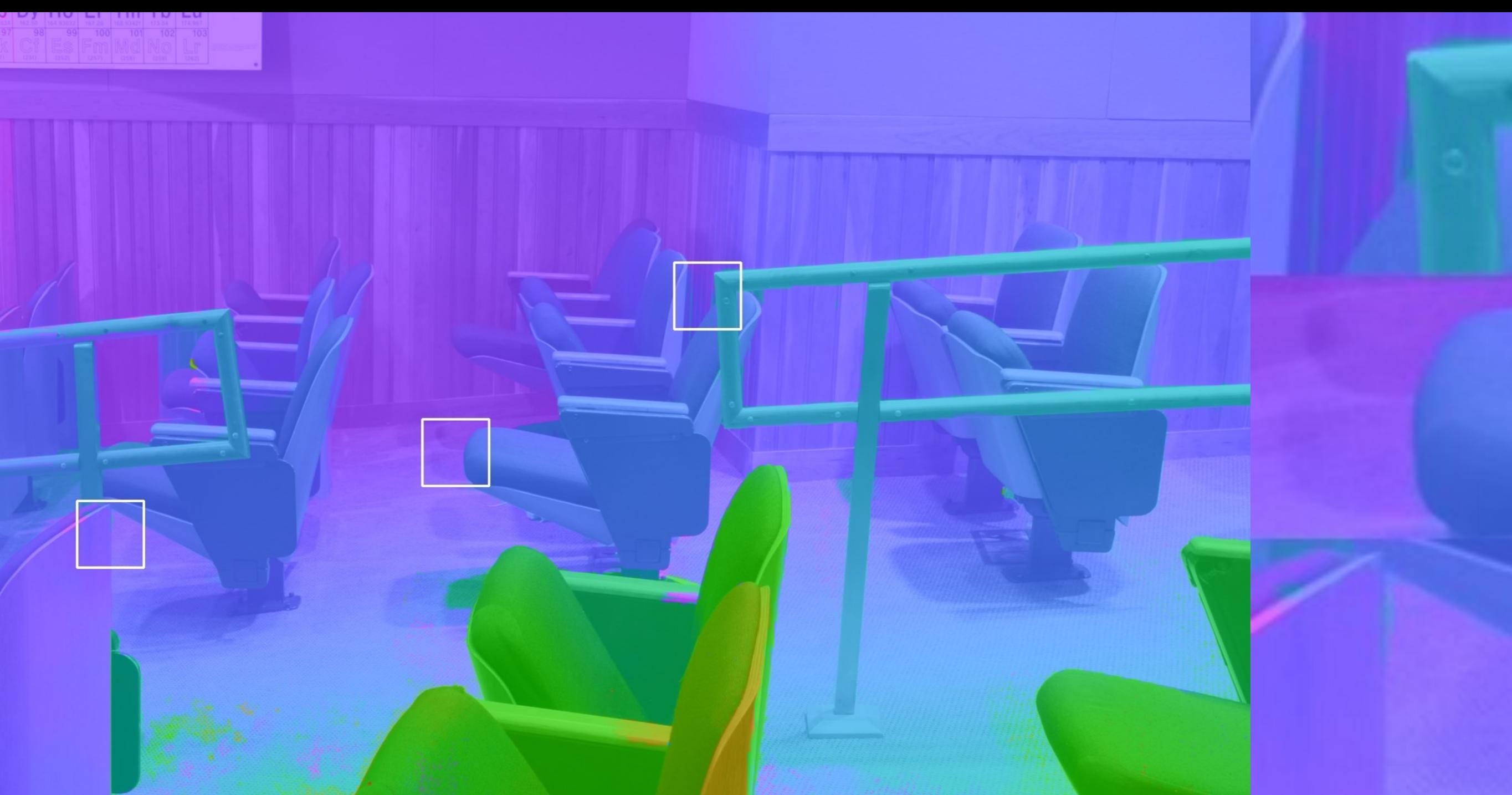
Stereo



MC-CNN

Zbontar & LeCun, CVPR 2015

Stereo



MC-CNN + Robust Bilateral Solver

Stereo

Middlebury Test Set V3

Method	MAE	RMSE
MC-CNN	17.9	55.0
MC-CNN + RBS	8.19	29.9

(Lowest test-set MAE and RMSE at submission time)

Stereo

Middlebury Test Set V3

Method	MAE	RMSE
MC-CNN	17.9	55.0
MC-CNN + RBS	8.19	29.9

Middlebury Training Set V3

Method	MAE	RMSE
MC-CNN	5.93	18.36
MC-CNN + TF	5.67	16.18
MC-CNN + FGF	5.91	16.32
MC-CNN + WMF	5.30	15.62
MC-CNN + DT	5.69	16.53
MC-CNN + RBS	2.81	8.44

Stereo

Middlebury Test Set V3

Method	MAE	RMSE
MC-CNN	17.9	55.0
MC-CNN + RBS	8.19	29.9

Middlebury Training Set V3

Method	MAE	RMSE
MeshStereo	3.83	10.75
MeshStereo + TF	3.81	9.91
MeshStereo + FGF	3.96	10.03
MeshStereo + WMF	3.87	10.10
MeshStereo + DT	3.77	10.12
MeshStereo + RBS	3.22	8.72

Stereo

Middlebury Test Set V3

Method	MAE	RMSE
MC-CNN	17.9	55.0
MC-CNN + RBS	8.19	29.9

Middlebury Training Set V3

Method	MAE	RMSE
TMAP	3.98	11.55
TMAP + TF	3.94	10.90
TMAP + FGF	4.17	10.79
TMAP + WMF	4.11	11.01
TMAP + DT	3.86	10.92
TMAP + RBS	3.31	9.44

Stereo

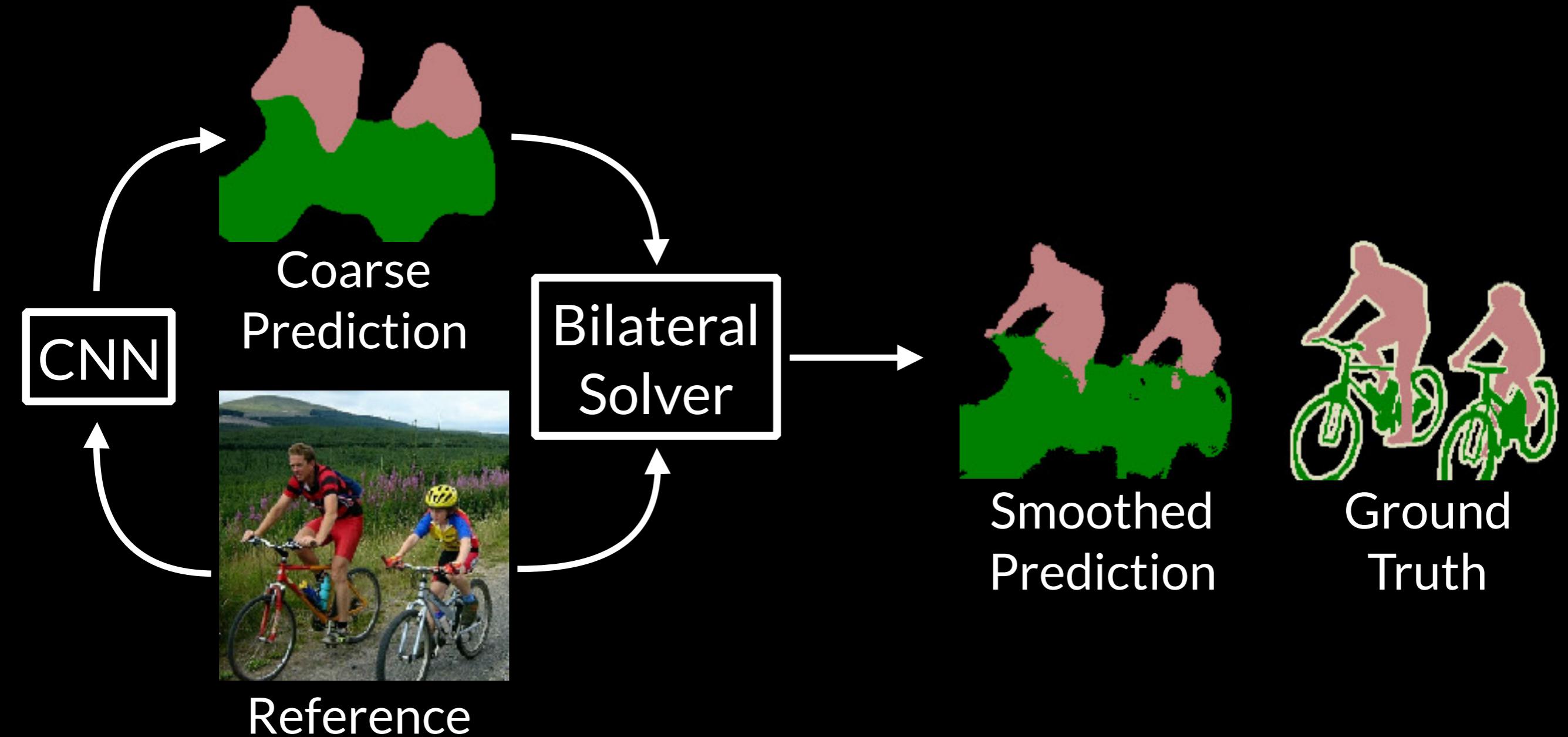
Middlebury Test Set V3

Method	MAE	RMSE
MC-CNN	17.9	55.0
MC-CNN + RBS	8.19	29.9

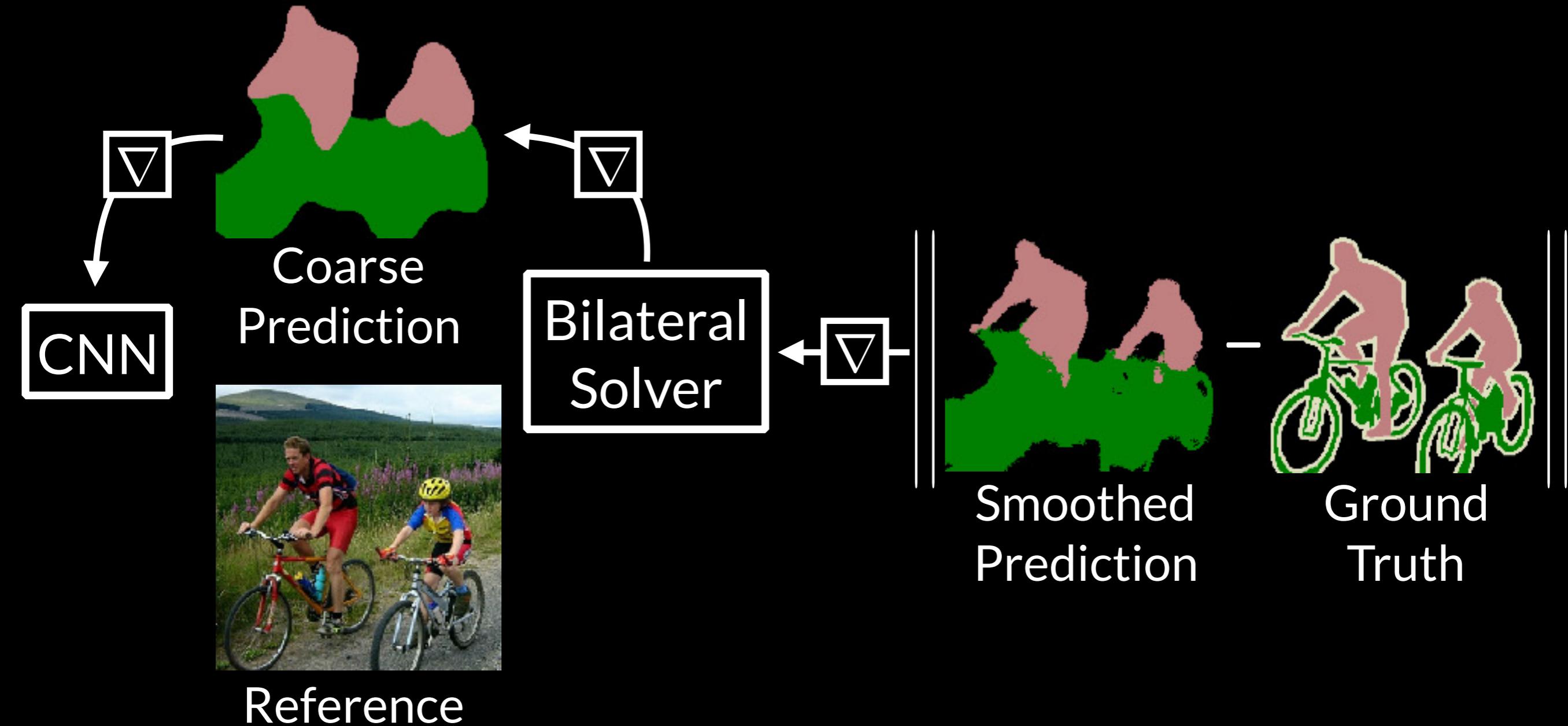
Middlebury Training Set V3

Method	MAE	RMSE
SGM	3.85	10.68
SGM + TF	3.82	9.55
SGM + FGF	4.05	9.66
SGM + WMF	3.97	9.99
SGM + DT	3.85	9.90
SGM + RBS	3.44	9.21

Bilateral Solver as a “Layer”



Bilateral Solver as a “Layer”



Bilateral Solver as a “Layer”

Forward: $\mathbf{x} \leftarrow \mathbf{S}^T (\mathbf{A}^{-1} (\mathbf{S}\mathbf{y}))$

Backward: $\nabla_{\mathbf{y}} \leftarrow \mathbf{S}^T (\mathbf{A}^{-1} (\mathbf{S}\nabla_{\mathbf{x}}))$

No annoying bookkeeping, “unrolling”, etc.

Semantic Segmentation



Reference

Semantic Segmentation



Deeplab

Accuracy (IOU)	Runtime (ms)
Deeplab 62.3%	58

Chen et al, ICLR 2015

Semantic Segmentation



Deeplab + DenseCRF

Accuracy (IOU)	Runtime (ms)
Deeplab 62.3%	58
Deeplab + DenseCRF 67.6%	58 + 918

Chen et al, ICLR 2015

Semantic Segmentation



Deeplab + Bilateral Solver

Accuracy (IOU)	Runtime (ms)
Deeplab 62.3%	58
Deeplab + DenseCRF 67.6%	58 + 918
Deeplab + Bilateral Solver 66.0%	58 + 85

70% as accurate
11x faster

To Conclude

The bilateral solver is a simple, fast, and effective tool.

Nearly as fast as a bilateral filter, but significantly more accurate.

Works well on a variety of tasks: depth superresolution, colorization, stereo, semantic segmentation, etc.

Can be easily integrated into deep learning pipelines.

Code Available*

*not the exact same code that was used for the paper

github.com/poolio/bilateral_solver

Thanks!