

Efficient Large-Scale Stereo Matching

Andreas Geiger*, Martin Roser* and Raquel Urtasun**

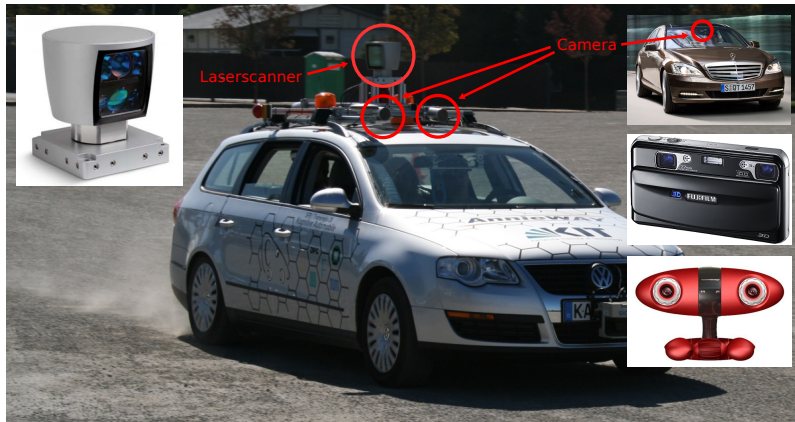
*KARLSRUHE INSTITUTE OF TECHNOLOGY

**TOYOTA TECHNOLOGICAL INSTITUTE AT CHICAGO



- 1 Motivation and Related Work
- 2 Efficient Large-Scale Stereo Matching
- 3 Experimental Evaluation
- 4 Summary and Future Work

- 1 Motivation and Related Work
- 2 Efficient Large-Scale Stereo Matching
- 3 Experimental Evaluation
- 4 Summary and Future Work



Why is 3D from Stereo hard?

- Ambiguities
- Textureless regions
- Sensor saturation
- Non-Lambertian surfaces
- Δz grows quadratically
- Computational burden



Why is 3D from Stereo hard?

- Ambiguities
- Textureless regions
- Sensor saturation
- Non-Lambertian surfaces
- Δz grows quadratically
- Computational burden



Why is 3D from Stereo hard?

- Ambiguities
- Textureless regions
- Sensor saturation
- Non-Lambertian surfaces
- Δz grows quadratically
- Computational burden



$$|\Delta z| \approx \frac{z^2}{f \cdot b} |\Delta d|$$

distance error focal length baseline disparity error

Local Methods

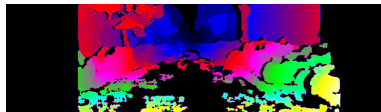
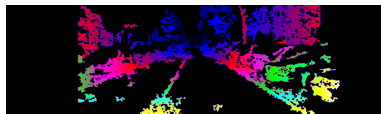
- Winner-takes-All

Examples

- Block matching
(Scharstein 02)
- Adaptive windows
(Kanade 94, Yoon 06)
- Plane-sweep
(Collins 96, Gallup 07)

Problems

- Small matching ratios
- Border bleeding



Global Methods

- Minimize 1D/2D energy $E(d) = E_{\text{data}}(d) + \lambda E_{\text{smooth}}(d)$

Examples

- Graph cuts, Belief propagation
(Kolmogorov 02, Felzenszwalb 06)
- Variational methods
(Pock 07, Zach 09)
- Fusion moves
(Woodford 08, Bleyer 10)



Problems

- Computational and memory requirements
- Pairwise potentials
can not model planarity

Seed-and-Grow Methods

- Grow disparity components from random seeds

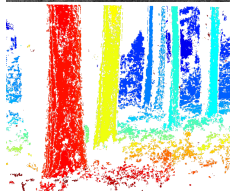
Examples

- (Cech 07)
- (Sara 03)



Problems

- Slanted/textureless surfaces
- No dense disparity maps



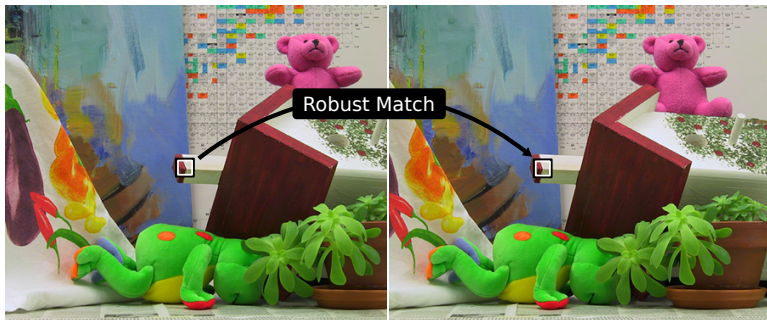
- 1 Motivation and Related Work
- 2 Efficient Large-Scale Stereo Matching**
- 3 Experimental Evaluation
- 4 Summary and Future Work



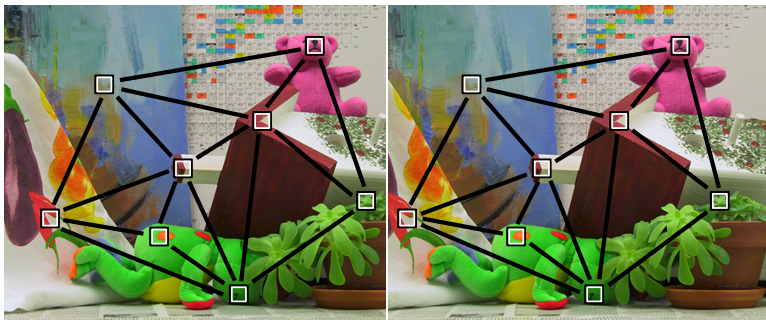
- Assumption: rectified images
- Image pairs contain 'easy' and 'hard' correspondences
- Robustly match 'easy' correspondences on regular grid
- Build prior on dense search space $\Rightarrow$ dense matching



- Assumption: rectified images
- Image pairs contain 'easy' and 'hard' correspondences
- Robustly match 'easy' correspondences on regular grid
- Build prior on dense search space $\Rightarrow$ dense matching



- Assumption: rectified images
- Image pairs contain 'easy' and 'hard' correspondences
- Robustly match 'easy' correspondences on regular grid
- Build prior on dense search space $\Rightarrow$ dense matching



- Assumption: rectified images
- Image pairs contain 'easy' and 'hard' correspondences
- Robustly match 'easy' correspondences on regular grid
- Build prior on dense search space $\Rightarrow$ dense matching

Notation

- Robust support points $\mathbf{S} = \{\mathbf{s}_1, \dots, \mathbf{s}_M\}$
with $\mathbf{s}_m = (u_m \ v_m \ d_m)^T$
- Disparity $d_n \in \mathbb{N}$
- Observations $\mathbf{o}_n = (u_n \ v_n \ \mathbf{f}_n)^T$
- Local image features $\mathbf{f}_n$

Algorithm

- Split image domain into support points $\mathbf{S}$ and dense pixels
- Assume factorization of distribution over disparity, observations and support points into ...

Notation

- Robust support points $\mathbf{S} = \{\mathbf{s}_1, \dots, \mathbf{s}_M\}$
with $\mathbf{s}_m = (u_m \ v_m \ d_m)^T$
- Disparity $d_n \in \mathbb{N}$
- Observations $\mathbf{o}_n = (u_n \ v_n \ \mathbf{f}_n)^T$
- Local image features $\mathbf{f}_n$

Algorithm

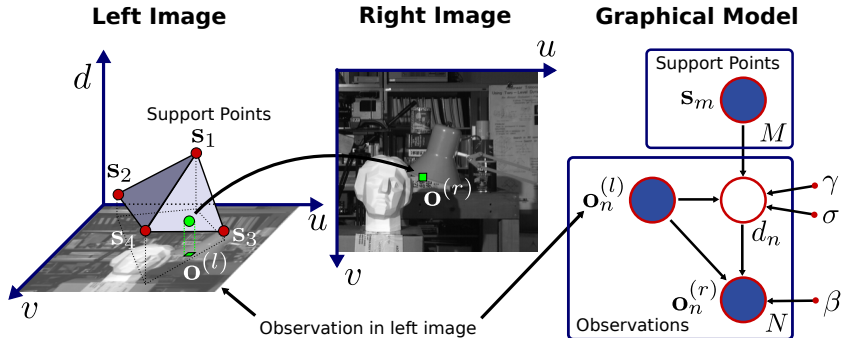
- Split image domain into support points $\mathbf{S}$ and dense pixels
- Assume factorization of distribution over disparity, observations and support points into ...

Notation

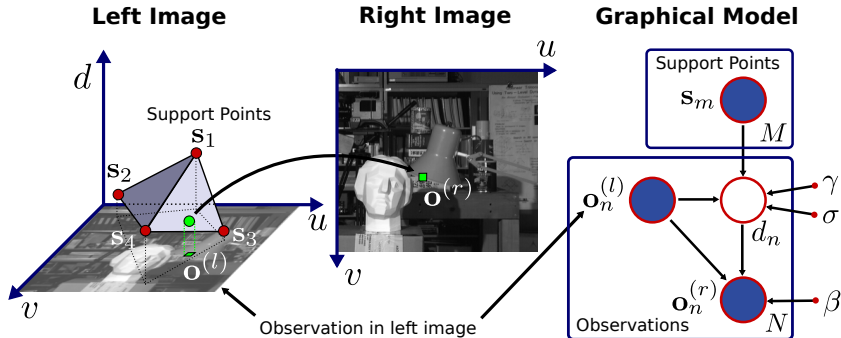
- Robust support points $\mathbf{S} = \{\mathbf{s}_1, \dots, \mathbf{s}_M\}$
with $\mathbf{s}_m = (u_m \ v_m \ d_m)^T$
- Disparity $d_n \in \mathbb{N}$
- Observations $\mathbf{o}_n = (u_n \ v_n \ \mathbf{f}_n)^T$
- Local image features $\mathbf{f}_n$

Algorithm

- Split image domain into support points $\mathbf{S}$ and dense pixels
- Assume factorization of distribution over disparity, observations and support points into ...



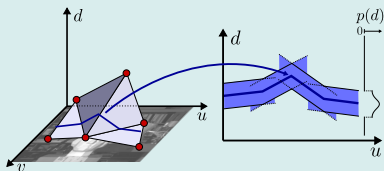
$$p(d_n, \mathbf{o}_n^{(l)}, \mathbf{o}_n^{(r)}, \mathbf{S}) \propto \underbrace{p(d_n | \mathbf{S}, \mathbf{o}_n^{(l)})}_{\text{Prior}} \underbrace{p(\mathbf{o}_n^{(r)} | \mathbf{o}_n^{(l)}, d_n)}_{\text{Likelihood}}$$



$$p(d_n, \mathbf{o}_n^{(l)}, \mathbf{o}_n^{(r)}, \mathbf{S}) \propto \underbrace{p(d_n | \mathbf{S}, \mathbf{o}_n^{(l)})}_{\text{Prior}} \underbrace{p(\mathbf{o}_n^{(r)} | \mathbf{o}_n^{(l)}, d_n)}_{\text{Likelihood}}$$

Prior $p(d_n | \mathbf{S}, \mathbf{o}_n^{(l)})$

- Support pt. triangulation
- Piecew. linear manifold
- Local extrapolation

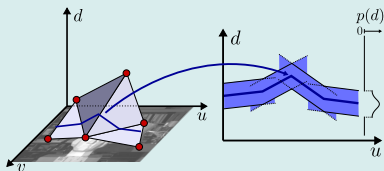


Likelihood $p(\mathbf{o}_n^{(r)} | \mathbf{o}_n^{(l)}, d_n)$

- Laplace distribution
- 5×5 block window
- 3×3 Sobel filter

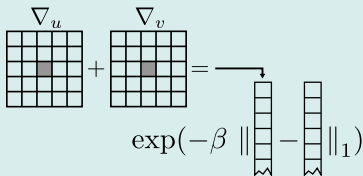
Prior $p(d_n | \mathbf{S}, \mathbf{o}_n^{(l)})$

- Support pt. triangulation
- Piecew. linear manifold
- Local extrapolation



Likelihood $p(\mathbf{o}_n^{(r)} | \mathbf{o}_n^{(l)}, d_n)$

- Laplace distribution
- 5×5 block window
- 3×3 Sobel filter



The diagram illustrates the likelihood calculation. It shows two 5×5 block windows, ∇_u and ∇_v , which are summed and then passed through a Laplace distribution function: $\exp(-\beta \| \cdot \|_1)$. The result is shown as a vertical vector.

Sampling from the model

Left image



Sample mean



Sampling from the model

Left image



Sample mean



Right image



- 1 Motivation and Related Work
- 2 Efficient Large-Scale Stereo Matching
- 3 Experimental Evaluation**
- 4 Summary and Future Work

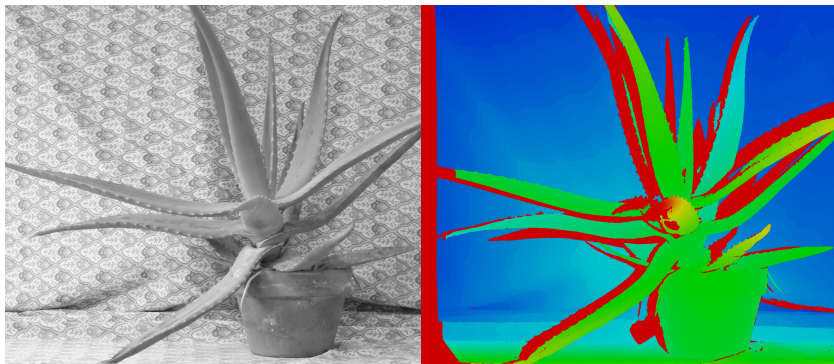


900 x 750 pixels, ground truth

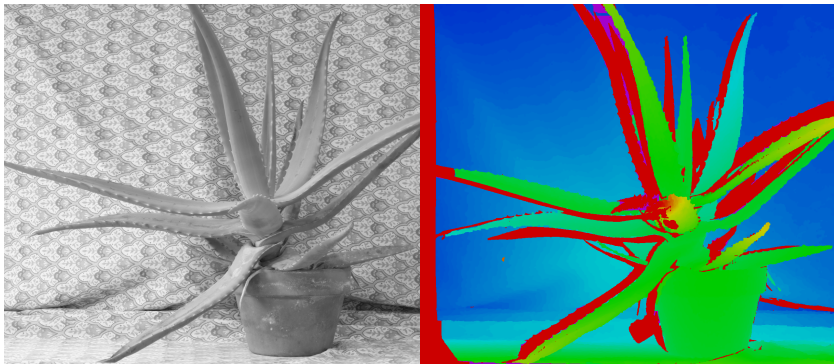
Middlebury Benchmark



900 x 750 pixels, 0.4 seconds

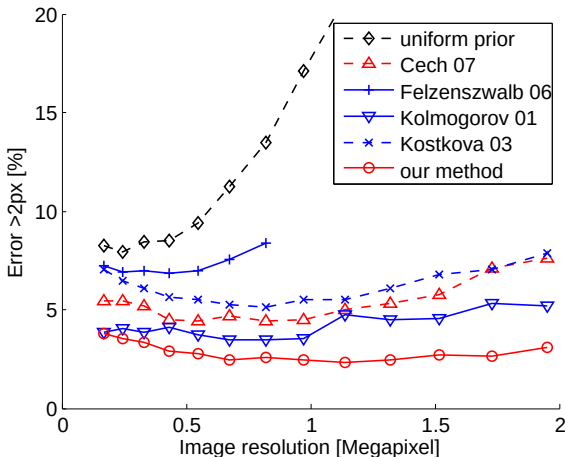


1300 x 1100 pixels, ground truth

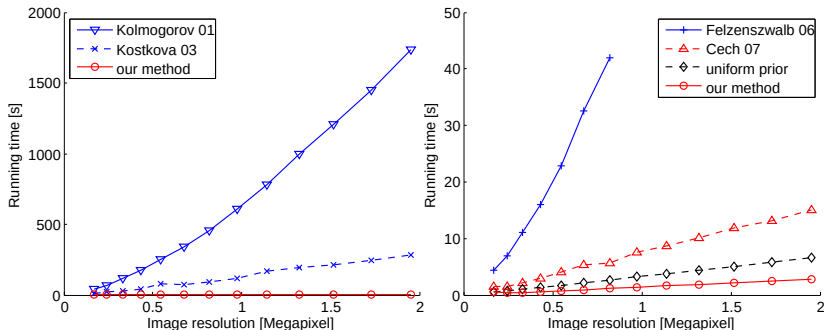


1300 x 1100 pixels, 1 second

Accuracy (on cones image pair)



Running times (on cones image pair)



[For more details see: Geiger et al., ACCV 2010]

3D Reconstruction: Brussels



2 seconds

[<http://cvlab.epfl.ch/data/strechamvs/>]

3D Face Reconstruction

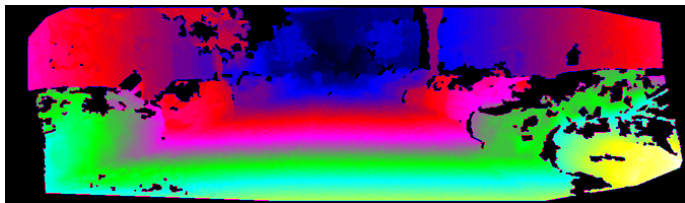


[<http://www.fujifilm.com/products/3d>]

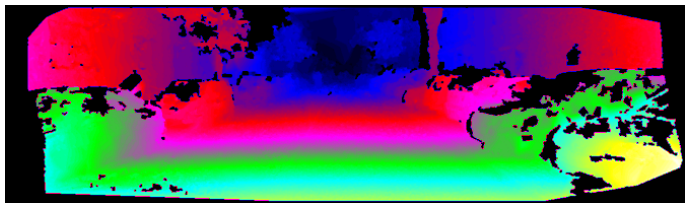
Urban Scene Reconstruction



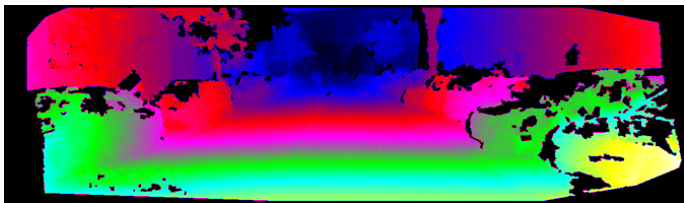
- 1 Motivation and Related Work
- 2 Efficient Large-Scale Stereo Matching
- 3 Experimental Evaluation
- 4 Summary and Future Work**



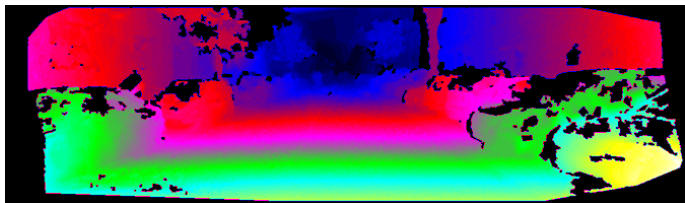
- Simple prior based on sparse feature matches
- Reduced ambiguities and run-time
- Takes into account slanted surfaces
- Real-time 3D reconstruction of static scenes on CPU
- C++ / MATLAB code available at <http://cvlibs.net>



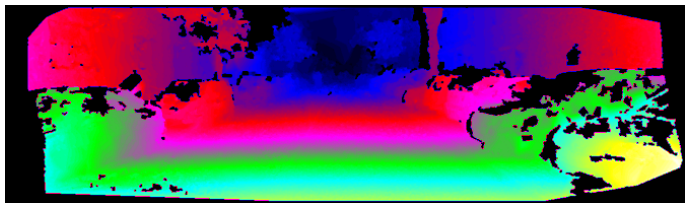
- Simple prior based on sparse feature matches
- Reduced ambiguities and run-time
- Takes into account slanted surfaces
- Real-time 3D reconstruction of static scenes on CPU
- C++ / MATLAB code available at <http://cvlibs.net>



- Simple prior based on sparse feature matches
- Reduced ambiguities and run-time
- Takes into account slanted surfaces
- Real-time 3D reconstruction of static scenes on CPU
- C++ / MATLAB code available at <http://cvlibs.net>



- Simple prior based on sparse feature matches
- Reduced ambiguities and run-time
- Takes into account slanted surfaces
- Real-time 3D reconstruction of static scenes on CPU
- C++ / MATLAB code available at <http://cvlibs.net>



- Simple prior based on sparse feature matches
- Reduced ambiguities and run-time
- Takes into account slanted surfaces
- Real-time 3D reconstruction of static scenes on CPU
- C++ / MATLAB code available at **<http://cvlibs.net>**

- Develop better priors
- Incorporate segmentation / global reasoning on lines
- GPU implementation
(goal: 20 fps at 1-2 megapixels)
- Employ as unitary potentials on global methods
⇒ smaller label sets
- **Thank you!**

- Develop better priors
 - Incorporate segmentation / global reasoning on lines
 - GPU implementation
(goal: 20 fps at 1-2 megapixels)
 - Employ as unitary potentials on global methods
⇒ smaller label sets
- **Thank you!**

- Develop better priors
- Incorporate segmentation / global reasoning on lines
- GPU implementation
(goal: 20 fps at 1-2 megapixels)
- Employ as unitary potentials on global methods
⇒ smaller label sets

■ Thank you!

- Develop better priors
- Incorporate segmentation / global reasoning on lines
- GPU implementation
(goal: 20 fps at 1-2 megapixels)
- Employ as unitary potentials on global methods
⇒ smaller label sets

■ **Thank you!**