Low-level Vision for Autonomous Driving

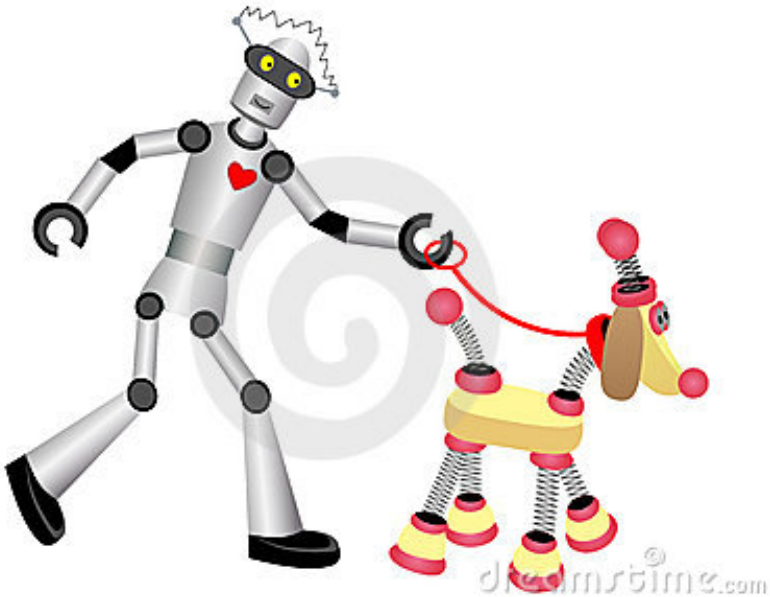
Raquel Urtasun

TTI Chicago

Sep 4, 2013

What excites me ... and hopefully you too ...

Developing autonomous systems that are able to help us in everyday's tasks



My view of how to get there ...

Autonomous systems need to:

- Sense the environment
- Recognize the 3D world
- Interact with it

What's important?

- Representation
- Learning
- Inference
- Data

I believe in **holistic approaches** that solve multiple tasks, and for that MRFs provide a great mathematical framework

Autonomous Driving

State of the art

- Localization, path planning, obstacle avoidance

Autonomous Driving

State of the art

- Localization, path planning, obstacle avoidance
- Heavy usage of Velodyne and detailed (recorded) maps

Autonomous Driving

3D Laser-scanner

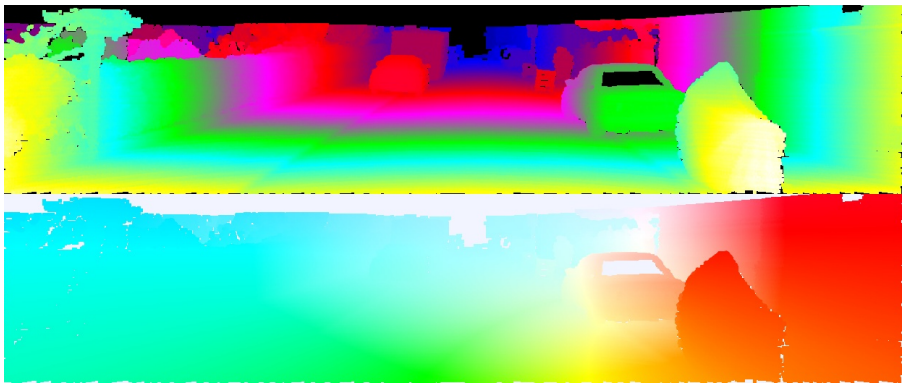
State of the art

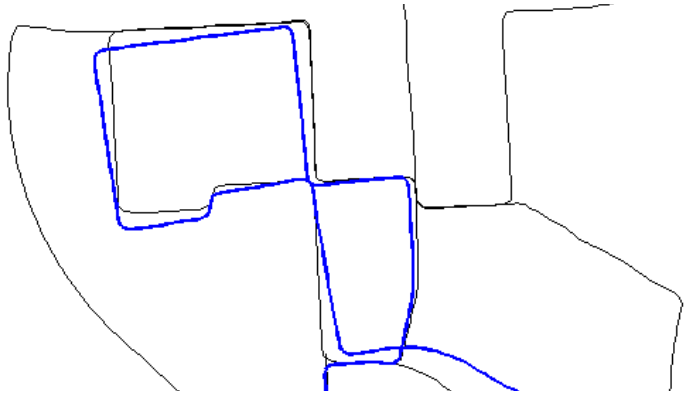
- Localization, path planning, obstacle avoidance
- Heavy usage of Velodyne and detailed (recorded) maps

Problems for computer vision

- Stereo, optical flow, visual odometry, structure-from-motion

Autonomous Driving





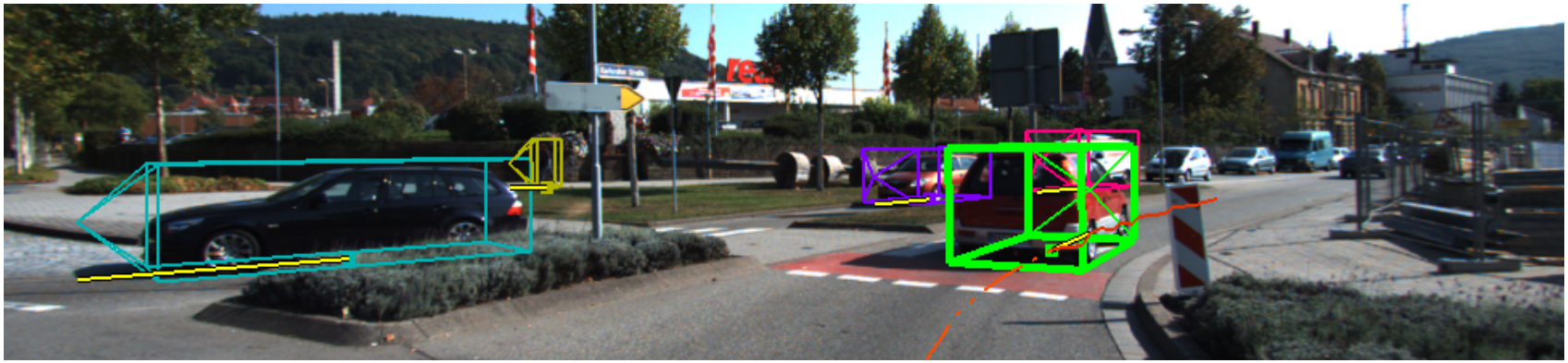
State of the art

- Localization, path planning, obstacle avoidance
- Heavy usage of Velodyne and detailed (recorded) maps

Problems for computer vision

- Stereo, optical flow, visual odometry, structure-from-motion
- Object detection, recognition and tracking

Autonomous Driving



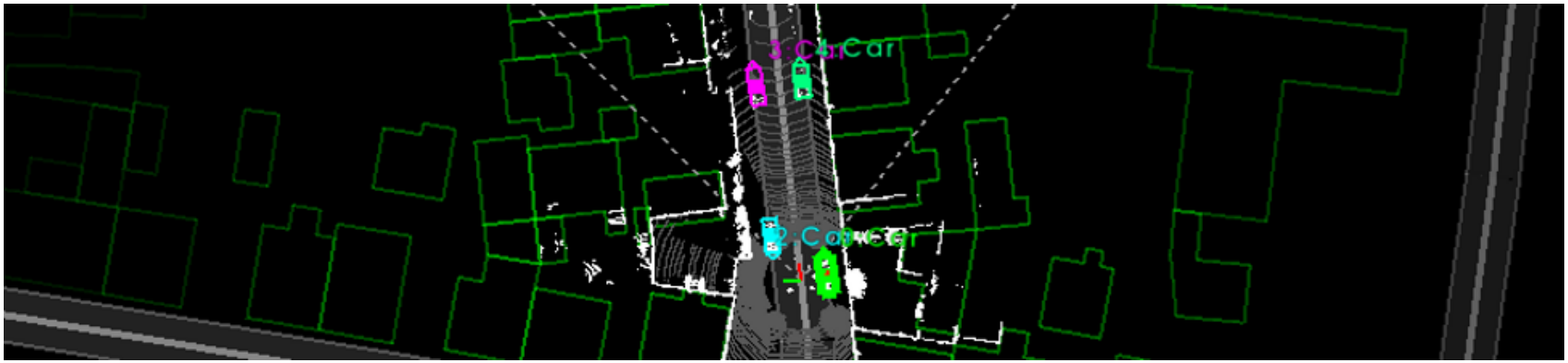
State of the art

- Localization, path planning, obstacle avoidance
- Heavy usage of Velodyne and detailed (recorded) maps

Problems for computer vision

- Stereo, optical flow, visual odometry, structure-from-motion
- Object detection, recognition and tracking
- 3D scene understanding

Autonomous Driving



State of the art

- Localization, path planning, obstacle avoidance
- Heavy usage of Velodyne and detailed (recorded) maps

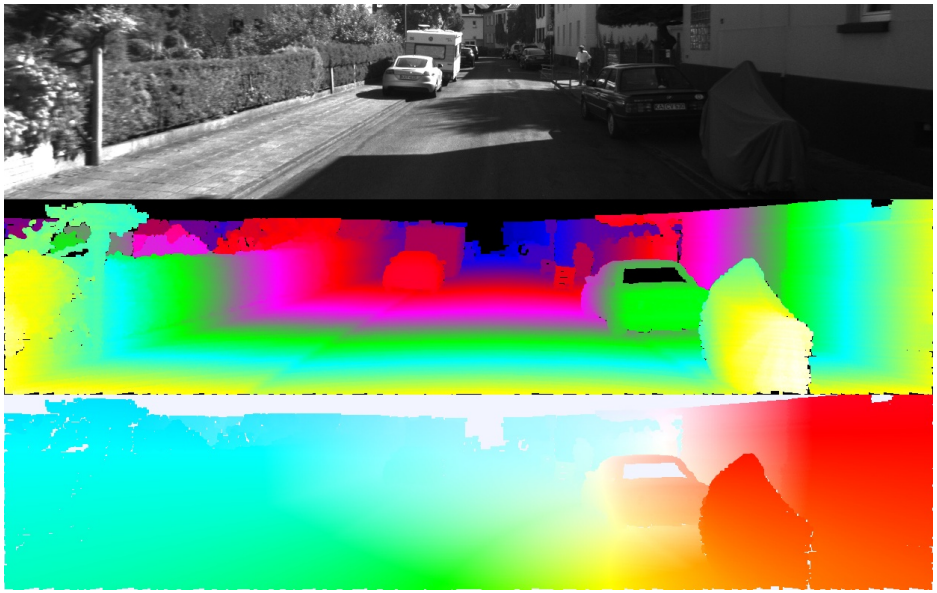
Problems for computer vision

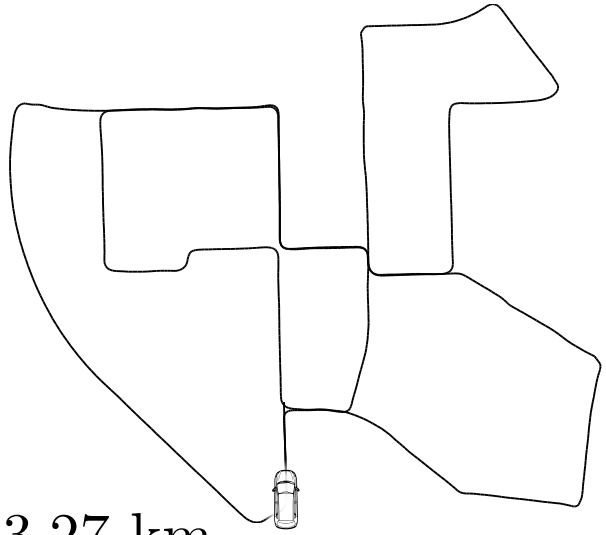
- Stereo, optical flow, visual odometry, structure-from-motion
- Object detection, recognition and tracking
- 3D scene understanding

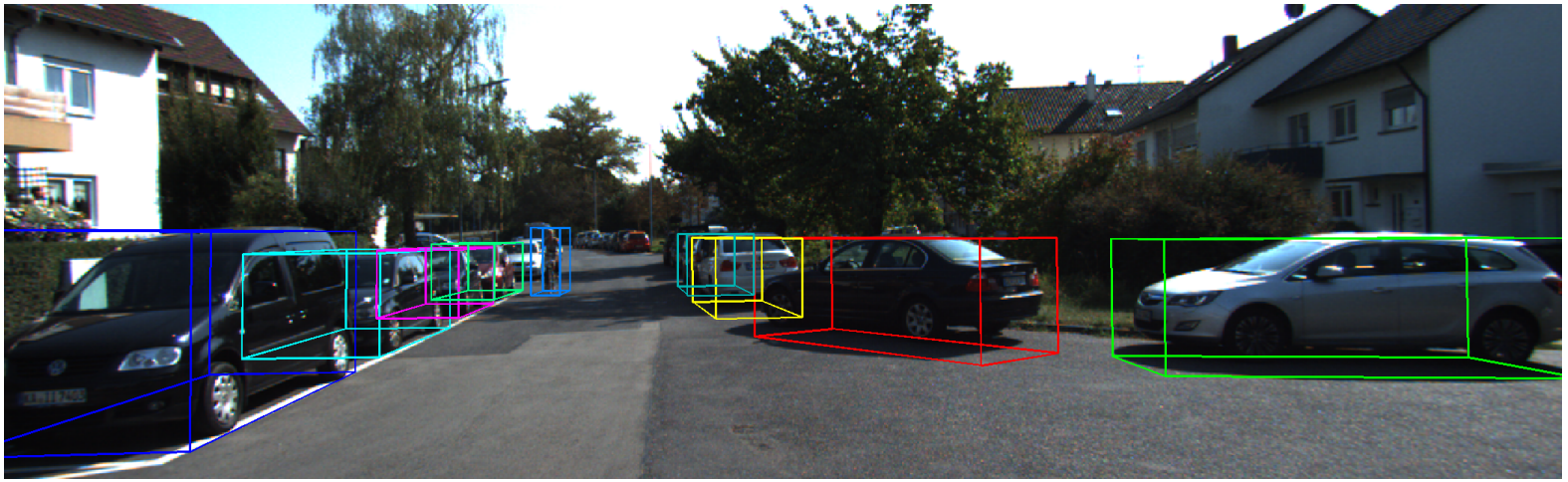
Benchmarks: KITTI Data Collection

- **Two stereo rigs** (1392 $\times$ 512 px, 54 cm base, 90° opening)
- **Velodyne** laser scanner, **GPS+IMU** localization
- **6 hours** at 10 frames per second!

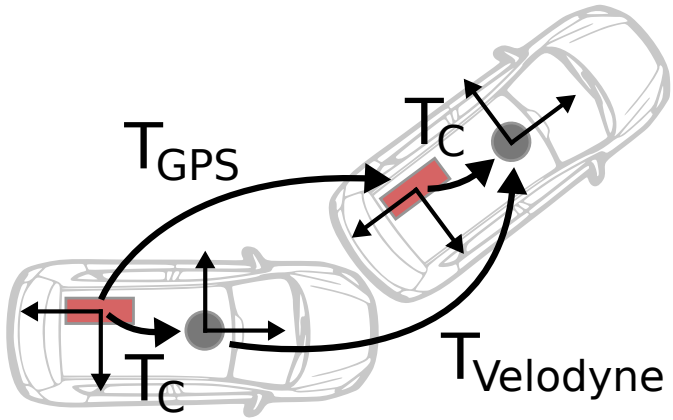
The KITTI Vision Benchmark Suite





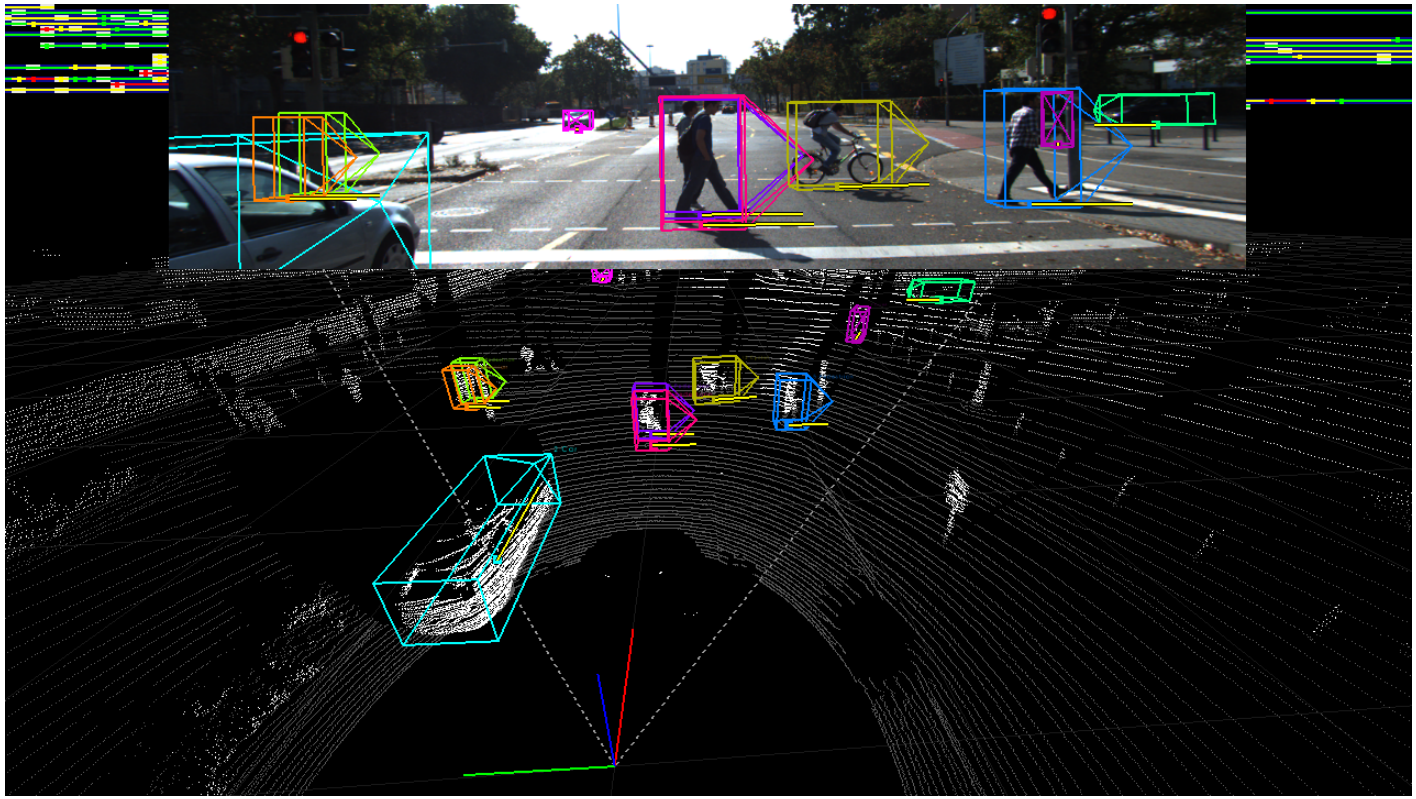


First Difficulty: Sensor Calibration



- Camera calibration [Geiger et al., ICRA 2012]
- Velodyne $\leftrightarrow$ Camera registration
- GPS+IMU $\leftrightarrow$ Velodyne registration

Second Difficulty: Object Annotation



- **3D object labels:** Annotators (undergrad students from KIT working for months)
- **Occlusion labels:** Mechanical Turk

One more Difficulty: Evaluation

- More than 100 submissions since CVPR 2012!
- Important to not have access to the test set ground truth!

Sense the Environment

- Goal: given 2 cameras mounted on top of the car, reconstruct the environment in 3D.

An MRF view of Stereo

- **Local methods**: try to locally match pixels in one image to the other
  - Suffer in texture-less regions
  - No global consistency
- **MRFs at the pixel level**
  - Still too local
  - Computationally expensive: large number of labels
- **Slanted-plane MRFs** assume a 3D piece-wise planar representation of the world
  - If segments small enough this assumption is good
  - Computationally expensive

Keys for Success

- Good **over-segmentation** is key for slanted-plan MRF methods to work well
- We developed an algorithm that produces over-segmentation that respects **depth boundaries**
- Jointly solves for depth (piecewise-planar) and over-segments the image
- Works in only a few seconds in a single core
- Given the over-segmentation, we can build our sophisticated MRFs

SLIC superpixels

- The simplest unsupervised segmentation algorithm is to use K-means
- Let $\mathbf{S} = \{s_1, \cdots, s_N)$ be the set of superpixel **assignments** for each pixel

SLIC superpixels

- The simplest unsupervised segmentation algorithm is to use K-means
- Let $\mathbf{S} = \{s_1, \cdots, s_N)$ be the set of superpixel **assignments** for each pixel
- We define $\mu = \{\mu_1, \cdots, \mu_m\}$ as the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor.

SLIC superpixels

- The simplest unsupervised segmentation algorithm is to use K-means
- Let $\mathbf{S} = \{s_1, \cdots, s_N)$ be the set of superpixel **assignments** for each pixel
- We define $\mu = \{\mu_1, \cdots, \mu_m\}$ as the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor.
- We can define the total energy of a pixel as

$$E(p) = E_{\text{col}}(\mathbf{p}, c_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p})$$

SLIC superpixels

- The simplest unsupervised segmentation algorithm is to use K-means
- Let $\mathbf{S} = \{s_1, \cdots, s_N)$ be the set of superpixel **assignments** for each pixel
- We define $\mu = \{\mu_1, \cdots, \mu_m\}$ as the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor.
- We can define the total energy of a pixel as

$$E(p) = E_{\text{col}}(\mathbf{p}, c_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p})$$

- The problem of unsupervised segmentation becomes

$$\min_{\mathbf{S},\mu,\mathbf{c}} \sum_{\mathbf{p}} E(\mathbf{p}, s_p, \mu_{s_p}, c_{s_p}).$$

SLIC superpixels

- The simplest unsupervised segmentation algorithm is to use K-means
- Let $\mathbf{S} = \{s_1, \cdots, s_N)$ be the set of superpixel **assignments** for each pixel
- We define $\mu = \{\mu_1, \cdots, \mu_m\}$ as the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor.
- We can define the total energy of a pixel as

$$E(p) = E_{\text{col}}(\mathbf{p}, c_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p})$$

- The problem of unsupervised segmentation becomes

$$\min_{\mathbf{S},\mu,\mathbf{c}} \sum_{\mathbf{p}} E(\mathbf{p}, s_p, \mu_{s_p}, c_{s_p}).$$

- Simple iterative algorithm
  - Solve for the assignments $\mathbf{S}$
  - Solve in parallel for the positions μ and appearances $\mathbf{c}$

SLIC superpixels

- The simplest unsupervised segmentation algorithm is to use K-means
- Let $\mathbf{S} = \{s_1, \cdots, s_N)$ be the set of superpixel **assignments** for each pixel
- We define $\mu = \{\mu_1, \cdots, \mu_m\}$ as the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor.
- We can define the total energy of a pixel as

$$E(p) = E_{\text{col}}(\mathbf{p}, c_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p})$$

- The problem of unsupervised segmentation becomes

$$\min_{\mathbf{S},\mu,\mathbf{c}} \sum_{\mathbf{p}} E(\mathbf{p}, s_p, \mu_{s_p}, c_{s_p}).$$

- Simple iterative algorithm
  - Solve for the assignments $\mathbf{S}$
  - Solve in parallel for the positions μ and appearances $\mathbf{c}$

Joint Segmentation and Depth

- Let $\mathbf{S} = \{s_1, \cdots, s_m)$ be the set of superpixel **assignments**,
- We define $\mu = \{\mu_1, \cdots, \mu_m\}$ as the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor
- Let $\Theta = \{\theta_1, \cdots, \theta_m\}$ be the set of plane parameters with

$$d(p, \theta_i) = \alpha_i p_x + \beta_i p_y + \gamma_i$$

Joint Segmentation and Depth

- Let $\mathbf{S} = \{s_1, \cdots, s_m)$ be the set of superpixel **assignments**,
- We define $\mu = \{\mu_1, \cdots, \mu_m\}$ as the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor
- Let $\Theta = \{\theta_1, \cdots, \theta_m\}$ be the set of plane parameters with

$$d(p, \theta_i) = \alpha_i p_x + \beta_i p_y + \gamma_i$$

- We can define the total energy of a pixel as

$$E(p) = E^{l,r}_{\text{col}}(\mathbf{p}, c_{s_p}, \theta_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p}) + \lambda_{\text{disp}} E^{l,r}_{\text{disp}}(\mathbf{p}, \theta_{s_p}),$$

Joint Segmentation and Depth

- Let $\mathbf{S} = \{s_1, \cdots, s_m)$ be the set of superpixel **assignments**,
- We define $\mu = \{\mu_1, \cdots, \mu_m\}$ as the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor
- Let $\Theta = \{\theta_1, \cdots, \theta_m\}$ be the set of plane parameters with

$$d(p, \theta_i) = \alpha_i p_x + \beta_i p_y + \gamma_i$$

- We can define the total energy of a pixel as

$$E(p) = E^{I,r}_{\text{col}}(\mathbf{p}, c_{s_p}, \theta_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p}) + \lambda_{\text{disp}} E^{I,r}_{\text{disp}}(\mathbf{p}, \theta_{s_p}),$$

- We can use:

$$E_{pos}(\mathbf{p}, \mu_{s_p}) = ||\mathbf{p} - \mu_{s_p}||_2^2 / g \qquad E^I_{col}(\mathbf{p}, c_{s_p}) = (I_t(\mathbf{p}) - c_{s_p})^2$$

and

$$E_{disp}(\mathbf{p}, \theta_{s_p}) = \begin{cases} (d(\mathbf{p}, \theta_{s_p}) - \hat{d}(\mathbf{p}))^2 & \text{if } \mathbf{p} \in \mathcal{F} \\ \lambda & \text{otherwise} \end{cases}$$

Joint Segmentation and Depth

- Let $\mathbf{S} = \{s_1, \cdots, s_m)$ be the set of superpixel **assignments**,
- We define $\mu = \{\mu_1, \cdots, \mu_m\}$ as the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor
- Let $\Theta = \{\theta_1, \cdots, \theta_m\}$ be the set of plane parameters with

$$d(p, \theta_i) = \alpha_i p_x + \beta_i p_y + \gamma_i$$

- We can define the total energy of a pixel as

$$E(p) = E^{l,r}_{\text{col}}(\mathbf{p}, c_{s_p}, \theta_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p}) + \lambda_{\text{disp}} E^{l,r}_{\text{disp}}(\mathbf{p}, \theta_{s_p}),$$

- We can use:

$$E_{pos}(\mathbf{p}, \mu_{s_p}) = ||\mathbf{p} - \mu_{s_p}||_2^2 / g \qquad E^l_{col}(\mathbf{p}, c_{s_p}) = (I_t(\mathbf{p}) - c_{s_p})^2$$

and

$$E_{disp}(\mathbf{p}, \theta_{s_p}) = \begin{cases} (d(\mathbf{p}, \theta_{s_p}) - \hat{d}(\mathbf{p}))^2 & \text{if } \mathbf{p} \in \mathcal{F} \\ \lambda & \text{otherwise} \end{cases}$$

Joint Segmentation and Depth

- We can define the total energy of a pixel as

$$E(p) = E^{l,r}_{\text{col}}(\mathbf{p}, c_{s_p}, \theta_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p}) + \lambda_{\text{disp}} E^{l,r}_{\text{disp}}(\mathbf{p}, \theta_{s_p}),$$

- Joint unsupervised segmentation and flow estimation as

$$\min_{\Theta, \mathbf{S}, \mu, \mathbf{c}} \sum_{\mathbf{p}} E(\mathbf{p}, s_p, \theta_{s_p}, \mu_{s_p}, c_{s_p}).$$

Joint Segmentation and Depth

- We can define the total energy of a pixel as

$$E(p) = E^{l,r}_{\text{col}}(\mathbf{p}, c_{s_p}, \theta_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p}) + \lambda_{\text{disp}} E^{l,r}_{\text{disp}}(\mathbf{p}, \theta_{s_p}),$$

- Joint unsupervised segmentation and flow estimation as

$$\min_{\Theta, \mathbf{S}, \mu, \mathbf{c}} \sum_{\mathbf{p}} E(\mathbf{p}, s_p, \theta_{s_p}, \mu_{s_p}, c_{s_p}).$$

- Simple iterative algorithm
  - Solve for the assignments $\mathbf{S}$
  - Solve in parallel for the planes Θ, positions μ and appearances $\mathbf{c}$

Joint Segmentation and Depth

- We can define the total energy of a pixel as

$$E(p) = E^{l,r}_{\text{col}}(\mathbf{p}, c_{s_p}, \theta_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p}) + \lambda_{\text{disp}} E^{l,r}_{\text{disp}}(\mathbf{p}, \theta_{s_p}),$$

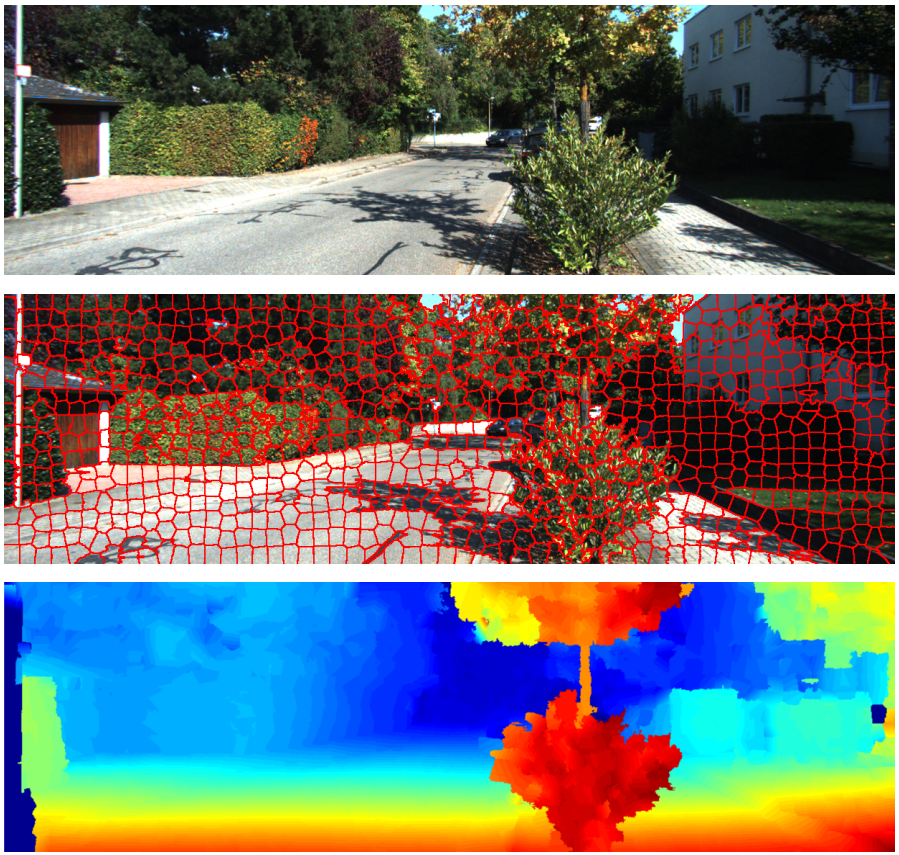
- Joint unsupervised segmentation and flow estimation as

$$\min_{\Theta, \mathbf{S}, \mu, \mathbf{c}} \sum_{\mathbf{p}} E(\mathbf{p}, s_p, \theta_{s_p}, \mu_{s_p}, c_{s_p}).$$

- Simple iterative algorithm
  - Solve for the assignments $\mathbf{S}$
  - Solve in parallel for the planes Θ, positions μ and appearances $\mathbf{c}$

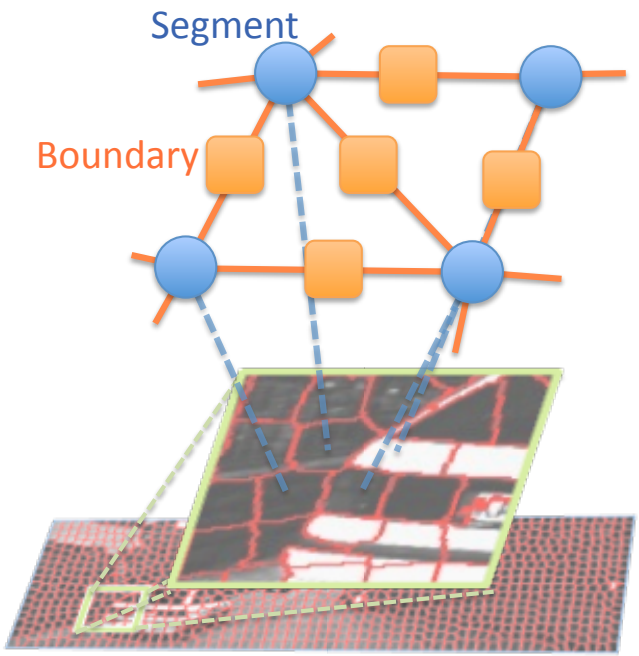
Results Stereo SLIC

[K. Yamaguchi, D. McAllester and R. Urtasun, CVPR13]



Slanted-Plane MRFs

- Slanted-plane MRF with explicit occlusion handling
- Start with an over-segmentation of the image into superpixels
- MRF on continuous variables (slanted planes) and discrete var. (boundary)



Superpixels (UCM [Arbelaez, et al. 2011] and SLIC [Achanta, et al. 2010])

Segment variable $\mathbf{y}_i = (\alpha_i, \beta_i, \gamma_i)$

Slanted 3D plane of segment

Continuous variable

Boundary variable o_{ij}

Relationship between segments

4 states

Occlusion Hinge Coplanar

Discrete variable

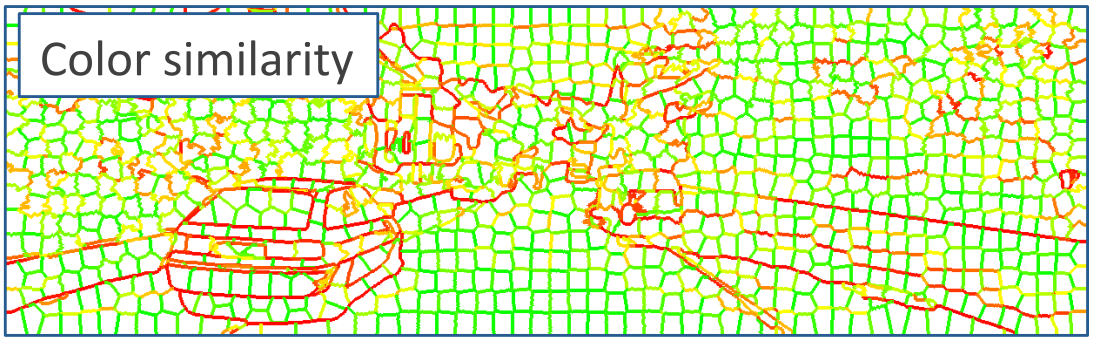
- Takes as input disparities computed by any local algorithm

Energy of PCBP-Stereo

- **y** the set of slanted 3D planes, **o** the set of discrete boundary variables

$$E(\mathbf{y},\mathbf{o}) = E_{color}(\mathbf{o}) + E_{match}(\mathbf{y},\mathbf{o}) + E_{compatibility}(\mathbf{y},\mathbf{o}) + E_{junction}(\mathbf{o})$$

Similar color ⇒ Likely to be coplanar

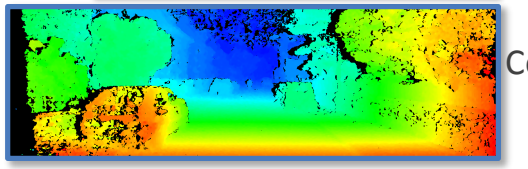


Energy of PCBP-Stereo

- $\mathbf{y}$ the set of slanted 3D planes, $\mathbf{o}$ the set of discrete boundary variables

$$E(\mathbf{y},\mathbf{o}) = E_{color}(\mathbf{o}) + E_{match}(\mathbf{y},\mathbf{o}) + E_{compatibility}(\mathbf{y},\mathbf{o}) + E_{junction}(\mathbf{o})$$

Agreement with result of input disparity map

Computed by any matching method
(Modified semi-global matching)

Truncated quadratic function $\phi_i^{\mathrm{TP}}(\mathbf{p},\mathbf{y}_i,K) = \min\left(|\mathcal{D}(\mathbf{p}) - \hat{d}_i(\mathbf{p},\mathbf{y}_i)|, K\right)^2$

Disparity map Slanted plane

On boundary

"Occlusion" – Foreground segment owns boundary

Energy of PCBP-Stereo

- **y** the set of slanted 3D planes, **o** the set of discrete boundary variables

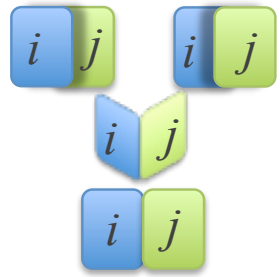
$$E(\mathbf{y},\mathbf{o}) = E_{color}(\mathbf{o}) + E_{match}(\mathbf{y},\mathbf{o}) + E_{compatibility}(\mathbf{y},\mathbf{o}) + E_{junction}(\mathbf{o})$$

(1) Preference of boundary label (Coplanar > Hinge > Occlusion)

Impose penalty $\lambda_{\text{occ}} > \lambda_{\text{hinge}} > 0$

(2) Boundary labels ⟷ (match) Slanted planes

- "Occlusion" ⟷ $\hat{d}_{\text{front}}(\mathbf{p}) > \hat{d}_{\text{back}}(\mathbf{p})$
- "Hinge" ⟷ $\hat{d}_i(\mathbf{p}) = \hat{d}_j(\mathbf{p})$ on boundary
- "Coplanar" ⟷ $\hat{d}_i(\mathbf{p}) = \hat{d}_j(\mathbf{p})$ in both segments



Energy of PCBP-Stereo

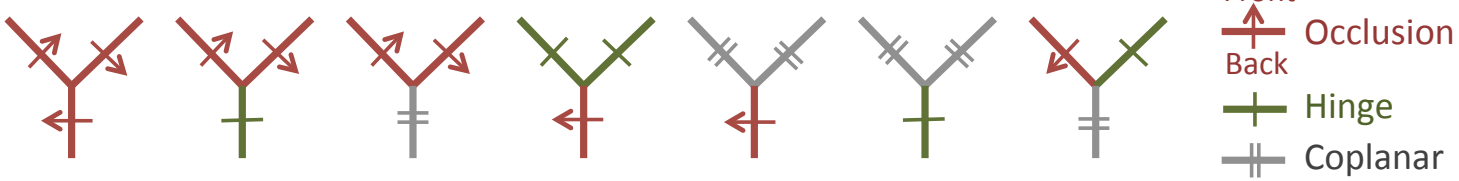
- $\mathbf{y}$ the set of slanted 3D planes, $\mathbf{o}$ the set of discrete boundary variables

$$E(\mathbf{y},\mathbf{o}) = E_{color}(\mathbf{o}) + E_{match}(\mathbf{y},\mathbf{o}) + E_{compatibility}(\mathbf{y},\mathbf{o}) + E_{junction}(\mathbf{o})$$

Occlusion boundary reasoning [Malik 1987]

Penalize impossible junctions

Impossible cases



Distributed Inference for Large Graphs

Learning: using our primal-dual algorithm [Hazan & Urtasun, NIPS 2010]

- Don't need to compute the partition function or the MAP at each iteration

Inference: use particle convex-BP [Peng, Hazan, McAllester, Urtasun, ICML 2011]

- Iterate sampling to discretize MRF and solving the discrete MRF

Algorithm 1 PCBP for stereo estimation and occlusion boundary reasoning

Set N
Initialize slanted planes $\mathbf{y}_i^0 = (\alpha_i^0, \beta_i^0, \gamma_i^0)$ via local fitting $\forall i$
Initialize $\sigma_\alpha, \sigma_\beta$ and σ_γ
for $t = 1$ to #iters **do**
  Sample N times $\forall i$ from $\alpha_i \sim \mathcal{N}(\alpha_i^{t-1}, \sigma_\alpha)$, $\beta_i \sim \mathcal{N}(\beta_i^{t-1}, \sigma_\beta)$, $\gamma_i \sim \mathcal{N}(\gamma_i^{t-1}, \sigma_\gamma)$
  $(\mathbf{o}^t, \mathbf{y}^t) \leftarrow$ Solve the discretized MRF using convex BP
  Update $\sigma_\alpha^c = \sigma_\beta^c = 0.5 \times \exp(-c/10)$ and $\sigma_\gamma^c = 5.0 \times \exp(-c/10)$
end for
Return $\mathbf{o}^t$, $\mathbf{y}^t$

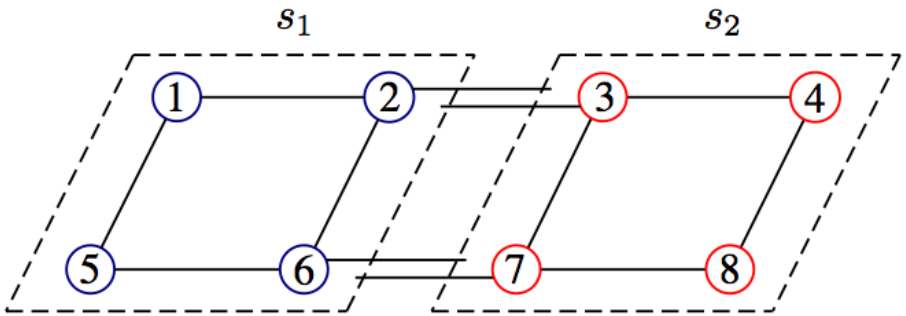
Distributed Inference for Large Graphs

Learning: using our primal-dual algorithm [Hazan & Urtasun, NIPS 2010]

- Don't need to compute the partition function or the MAP at each iteration

Inference: use particle convex-BP [Peng, Hazan, McAllester, Urtasun, ICML 2011]

- Iterate sampling to discretize MRF and solving the discrete MRF
- We use our distributed convex BP, which uses dual-decomposition to solve with **distributed memory and computation** while maintaining the theoretical guarantees [Schwing, Hazan, Pollefeys, Urtasun, CVPR 2011]



solve LP relaxation for each subgraph

subject to:

marginalization constraints

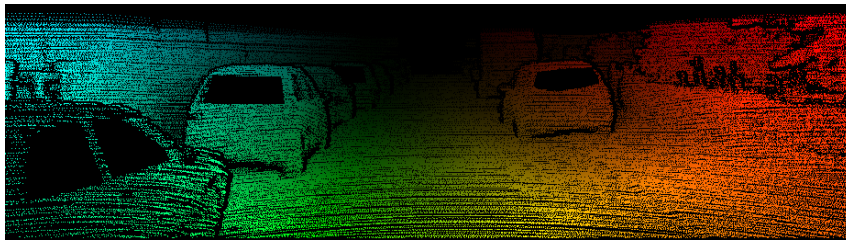
$$\forall s, \alpha \in N_{\mathcal{P}}(s), x_\alpha, \qquad b^s_\alpha(x_\alpha) = b_\alpha(x_\alpha)$$

KITTI Stereo/Flow Dataset

KITTI dataset (Geiger et al. 12)

- Real-world stereo/flow dataset
- Accurate ground truth
- High-resolution (1237x374 pixels)
- 10 train, 184 validation, 195 test images

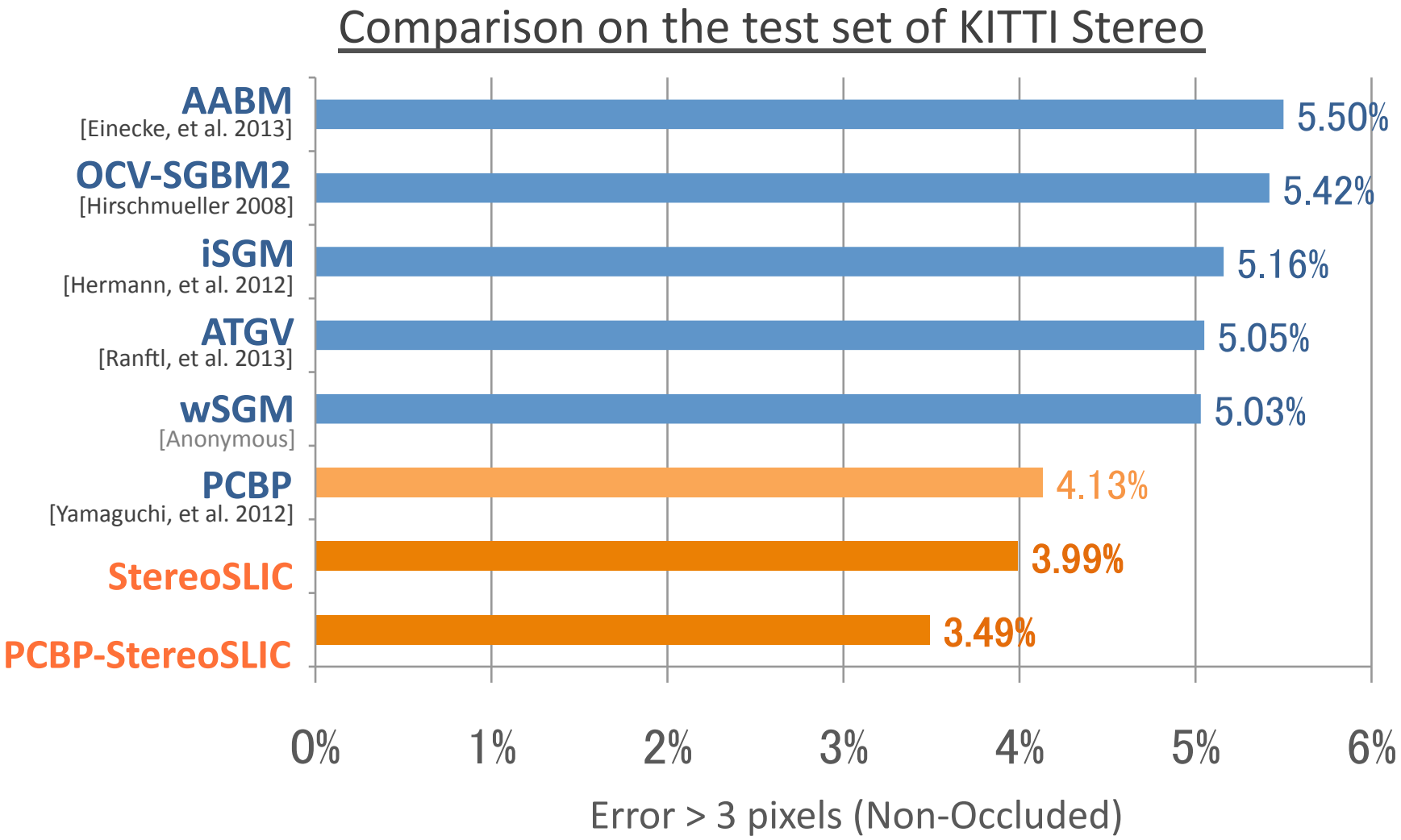
Ground truth



Stereo Evaluation

Rank	Method	Setting	Out-Noc	Out-All	Avg-Noc	Avg-All	Density	Runtime	Environment	Compare
1	PCBP		4.13 %	5.45 %	0.9 px	1.2 px	100.00 %	5 min	4 cores @ 2.5 Ghz (Matlab + C/C++)	☐
	Koichiro Yamaguchi, Tamir Hazan, David McAllester and Raquel Urtasun. Continuous Markov Random Fields for Robust Stereo Estimation. ECCV 2012.									
2	iSGM		5.16 %	7.19 %	1.2 px	2.1 px	94.70 %	8 s	2 cores @ 2.5 Ghz (C/C++)	☐
	Simon Hermann and Reinhard Klette. Iterative Semi-Global Matching for Robust Driver Assistance Systems. ACCV 2012.									
3	SGM		5.83 %	7.08 %	1.2 px	1.3 px	85.80 %	3.7 s	1 core @ 3.0 Ghz (C/C++)	☐
	Heiko Hirschmueller. Stereo Processing by Semi-Global Matching and Mutual Information. IEEE Transactions on Pattern Analysis and Machine Intelligence 2008.									
4	SNCC		6.27 %	7.33 %	1.4 px	1.5 px	100.00 %	0.27 s	1 core @ 3.0 Ghz (C/C++)	☐
	N. Einecke and J. Eggert. A Two-Stage Correlation Method for Stereoscopic Depth Estimation. DICTA 2010.									
5	ITGV		6.31 %	7.40 %	1.3 px	1.5 px	100.00 %	7 s	1 core @ 3.0 Ghz (Matlab + C/C++)	☐
	Rene Ranftl, Stefan Gehrig, Thomas Pock and Horst Bischof. Pushing the Limits of Stereo Using Variational Stereo Estimation. IEEE Intelligent Vehicles Symposium 2012.									
6	BSSM		7.50 %	8.89 %	1.4 px	1.6 px	94.87 %	20.7 s	1 core @ 3.5 Ghz (C/C++)	☐
	Anonymous submission									
7	OCV-SGBM		7.64 %	9.13 %	1.8 px	2.0 px	86.50 %	1.1 s	1 core @ 2.5 Ghz (C/C++)	☐
	Heiko Hirschmueller. Stereo processing by semiglobal matching and mutual information. PAMI 2008.									
8	ELAS		8.24 %	9.95 %	1.4 px	1.6 px	94.55 %	0.3 s	1 core @ 2.5 Ghz (C/C++)	☐
	Andreas Geiger, Martin Roser and Raquel Urtasun. Efficient Large-Scale Stereo Matching. ACCV 2010.									
9	MS-DSI		10.68 %	12.11 %	1.9 px	2.2 px	100.00 %	10.3 s	>8 cores @ 2.5 Ghz (C/C++)	☐
	Anonymous submission									
10	SDM		10.98 %	12.19 %	2.0 px	2.3 px	63.58 %	1 min	1 core @ 2.5 Ghz (C/C++)	☐
	Jana Kostkova. Stratified dense matching for stereopsis in complex scenes. BMVC 2003.									
11	GCSF		12.06 %	13.26 %	1.9 px	2.1 px	60.77 %	2.4 s	1 core @ 2.5 Ghz (C/C++)	☐
	Jan Cech, Jordi Sanchez-Riera and Radu P. Horaud. Scene Flow Estimation by Growing Correspondence Seeds. CVPR 2011.									
12	GCS		13.37 %	14.54 %	2.1 px	2.3 px	51.06 %	2.2 s	1 core @ 2.5 Ghz (C/C++)	☐
	Jan Cech and Radim Sara. Efficient Sampling of Disparity Space for Fast And Accurate Matching. BenCOS 2007.									
13	CostFilter		19.96 %	21.05 %	5.0 px	5.4 px	100.00 %	4 min	1 core @ 2.5 Ghz (Matlab)	☐
	Christoph Rhemann, Asmaa Hosni, Michael Bleyer, Carsten Rother and Margrit Gelautz. Fast Cost-Volume Filtering for Visual Correspondence and Beyond. CVPR 2011.									
14	OCV-BM		25.39 %	26.72 %	7.6 px	7.9 px	55.84 %	0.1 s	1 core @ 2.5 Ghz (C/C++)	☐
	G. Bradski. The OpenCV Library. Dr. Dobb's Journal of Software Tools 2000.									
15	GC+occ		33.50 %	34.74 %	8.6 px	9.2 px	87.57 %	6 min	1 core @ 2.5 Ghz (C/C++)	☐
	Vladimir Kolmogorov and Ramin Zabih. Computing Visual Correspondence with Occlusions using Graph Cuts. ICCV 2001.									

Results KITTI



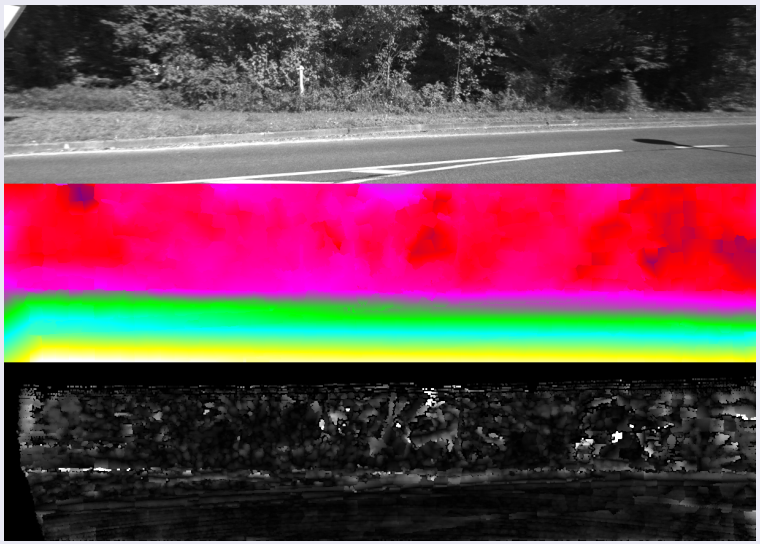
Stereo Evaluation

[K. Yamaguchi, T. Hazan, D. McAllester and R. Urtasun, ECCV12]

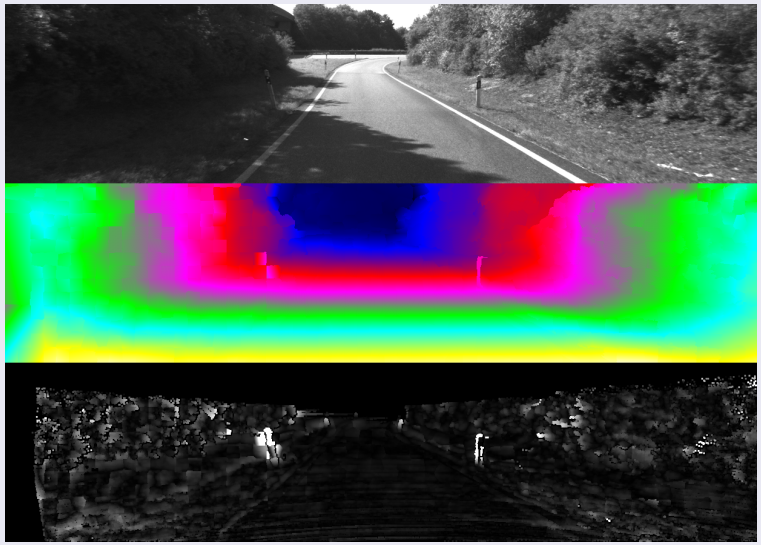
Easy Scenarios:

- Natural scenes, lots of texture, no objects
- A couple of errors at thin structures (poles)

Errors: $< 0.5\%$



Errors: $< 0.5\%$



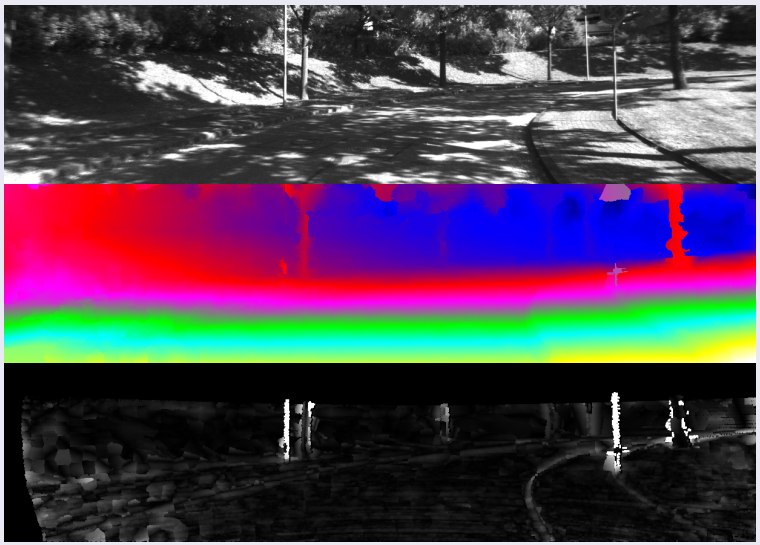
Stereo Evaluation

[K. Yamaguchi, T. Hazan, D. McAllester and R. Urtasun, ECCV12]

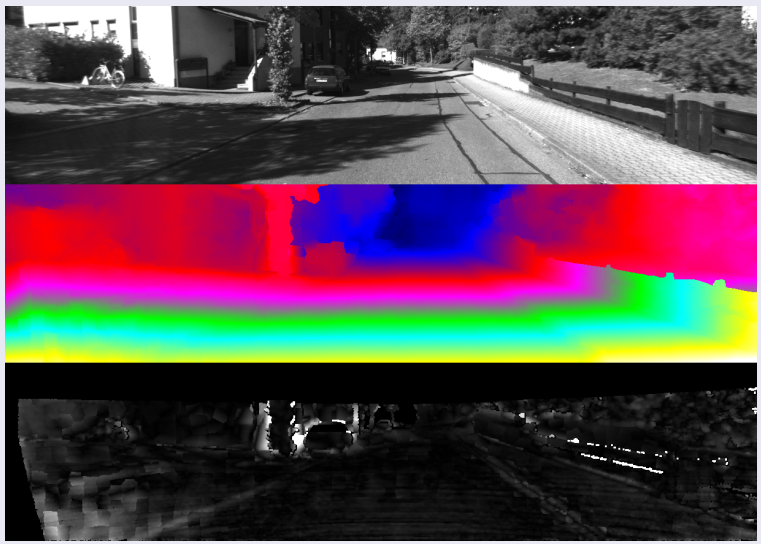
Easy Scenarios:

- Shadows help the disambiguation process
- Errors at thin structures and far away textureless regions

Errors: $< 0.5\%$



Errors: $< 0.5\%$



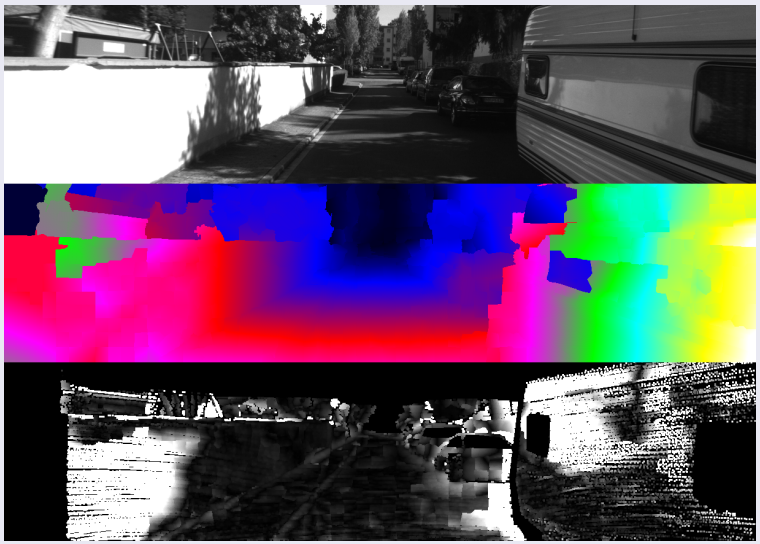
Stereo Evaluation

[K. Yamaguchi, T. Hazan, D. McAllester and R. Urtasun, ECCV12]

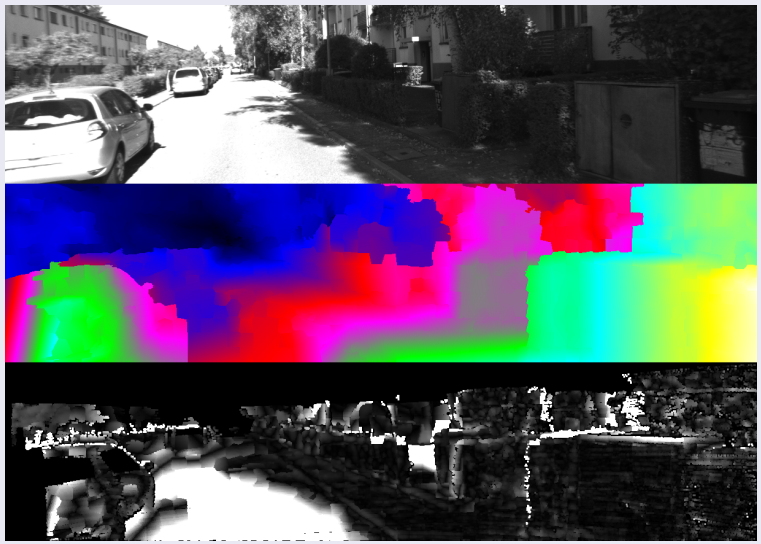
Hard Scenarios:

- Textureless or saturated areas
- Ambiguous reflections

Errors: 22.1%



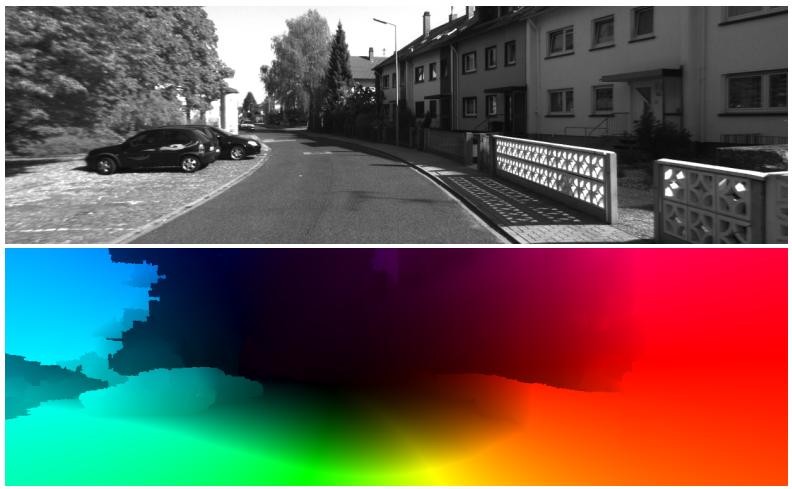
Errors: 17.4%



Sense the Environment

[K. Yamaguchi, D. McAllester and R. Urtasun, CVPR 2013]

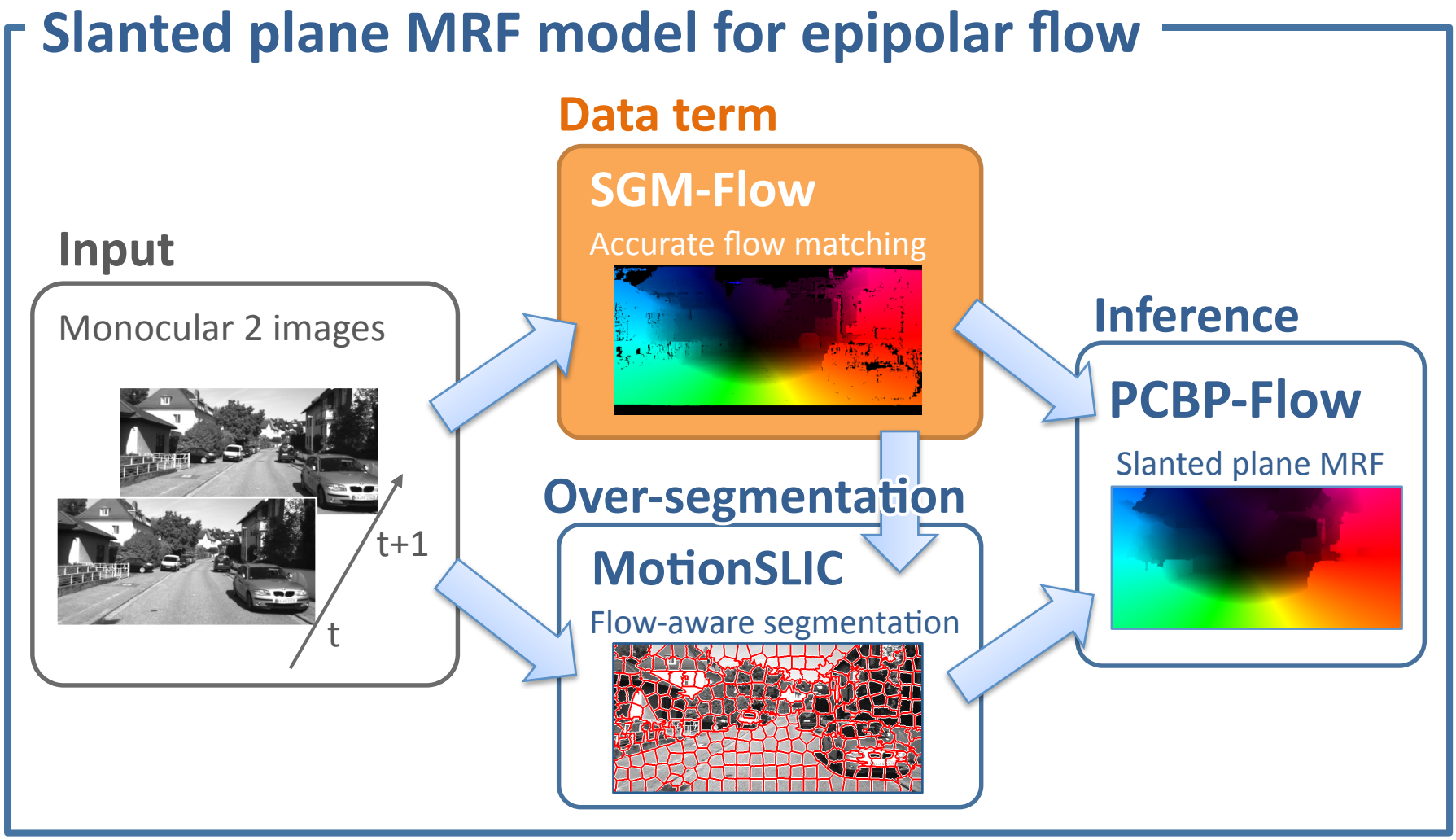
- Depth is not all!
- Recover the motion of the scene given a single camera



Flow for Autonomous Driving

- Most of the flow is due to the vehicle's **ego-motion**
- Goal: compute the **epipolar flow** by doing matching along the epipolar lines
- The problem is very similar to stereo
- Previous work does not have big performance gains with epipolar constraints
- Can we exploit what we learned about the stereo problem?

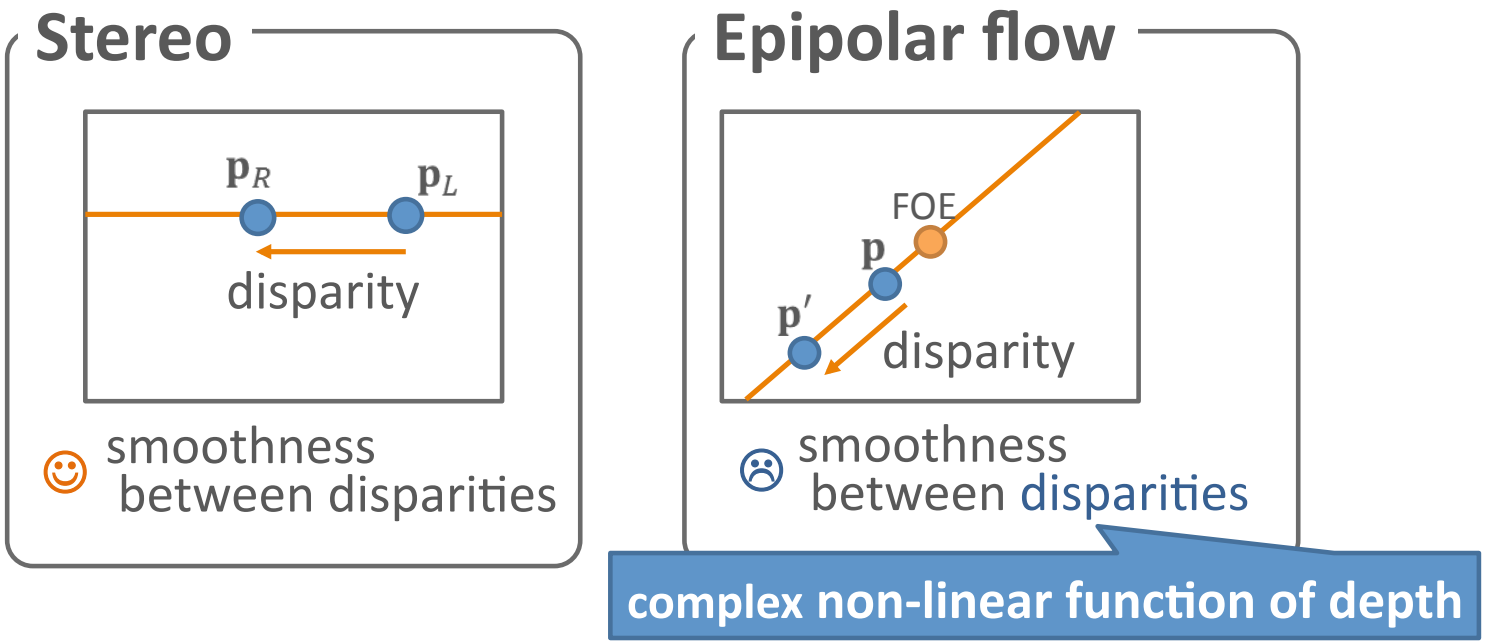
Diagram Flow



SGM for Epipolar Flow

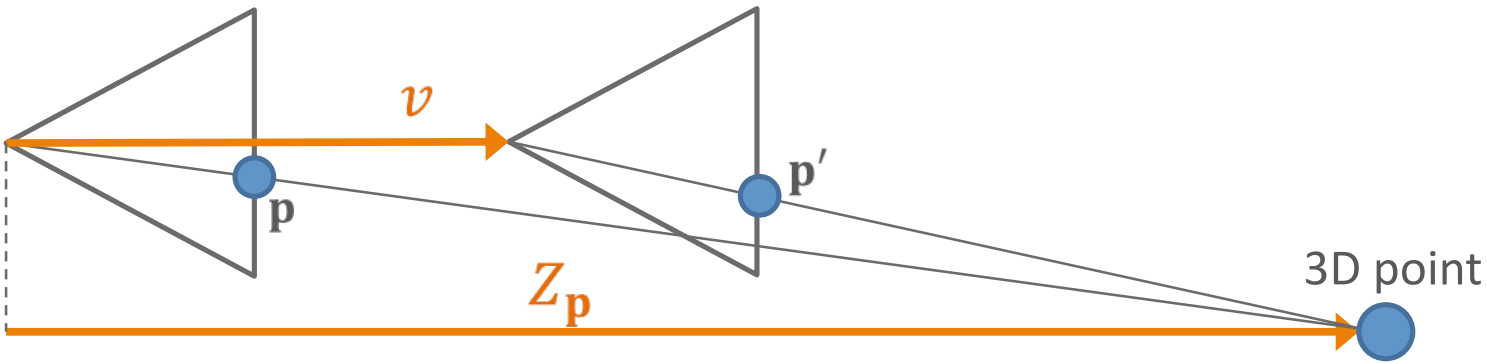
- Adapt semi-global block matching (SGM) [Hirschmueller PAMI2008] to the **epipolar flow problem**

$$\text{Energy} = \text{Unary term} + \text{Smoothness Term}$$



- Disparity is not a good representation to impose smoothness

SGM for Epipolar Flow



- **VZ-ratio** is a linear function of $\frac{1}{Z}$

$$\omega_p = \frac{v}{Z_p} = \frac{p' - p}{p} = \frac{d}{p + d}$$

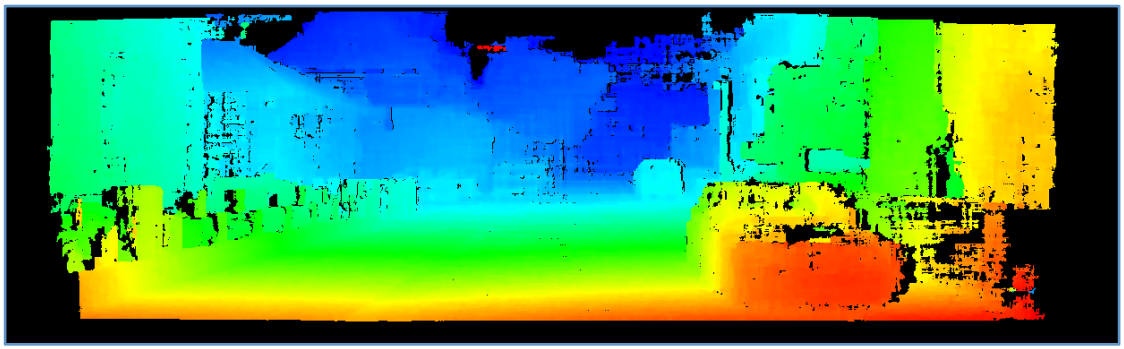
- Much better representation to impose smoothness than disparity

$$E(\omega) = \sum_p C(p, \omega_p) + \sum_{(p,q) \in \mathcal{N}} S(\omega_p, \omega_q)$$

SGM Flow

Input image

SGM-Flow
VZ-ratio map



Flow field

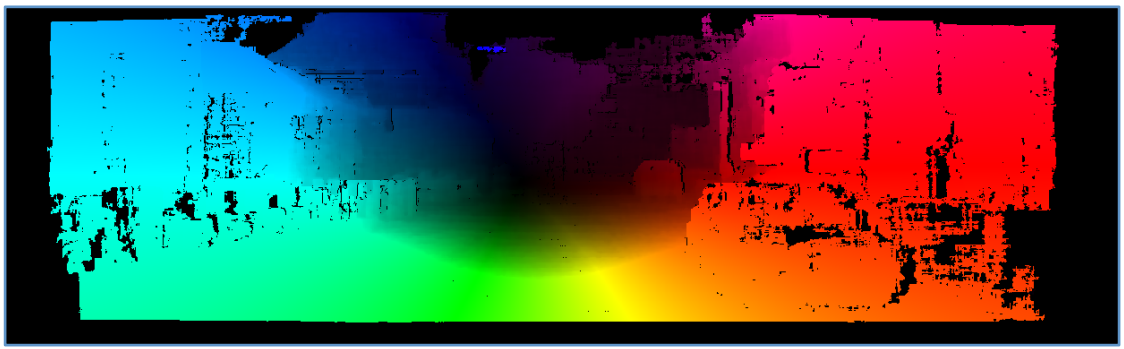
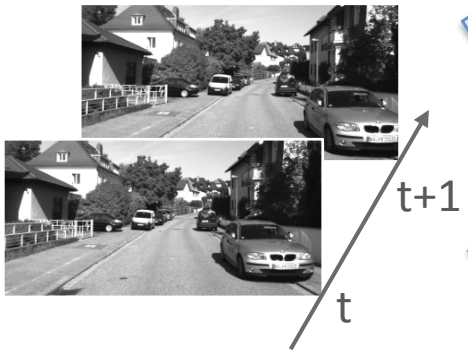


Diagram Flow

Slanted plane MRF model for epipolar flow

Input

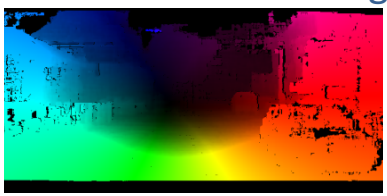
Monocular 2 images



Data term

SGM-Flow

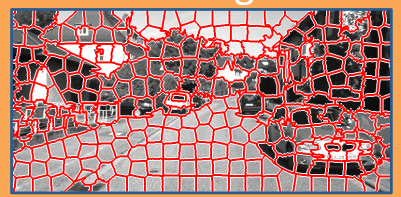
Accurate flow matching



Over-segmentation

MotionSLIC

Flow-aware segmentation



Inference

PCBP-Flow

Slanted plane MRF

Joint Segmentation and Epipolar Flow

- Let $\mathbf{S} = \{s_1, \cdots, s_m)$ be the set of superpixel **assignments**, $\mu = \{\mu_1, \cdots, \mu_m\}$ the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor
- Let $\Theta = \{\theta_1, \cdots, \theta_m\}$ be the set of plane parameters with

$$\omega(p, \theta_i) = \alpha_i p_x + \beta_i p_y + \gamma_i$$

Joint Segmentation and Epipolar Flow

- Let $\mathbf{S} = \{s_1, \cdots, s_m)$ be the set of superpixel **assignments**, $\mu = \{\mu_1, \cdots, \mu_m\}$ the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor
- Let $\Theta = \{\theta_1, \cdots, \theta_m\}$ be the set of plane parameters with

$$\omega(p, \theta_i) = \alpha_i p_x + \beta_i p_y + \gamma_i$$

- We can define the total energy of a pixel as

$$E(p) = E_{\text{col}}^{t,t+1}(\mathbf{p}, c_{s_p}, \theta_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p}) + \lambda_{\text{vz}} E_{\text{vz}}^{t,t+1}(\mathbf{p}, \theta_{s_p}),$$

with $E_{\text{vz}}^{t,t+1}(\mathbf{p}, \theta_{s_p}) = (\omega(p, \theta_{s_p}) - \omega_{SGM})^2$

Joint Segmentation and Epipolar Flow

- Let $\mathbf{S} = \{s_1, \cdots, s_m)$ be the set of superpixel **assignments**, $\mu = \{\mu_1, \cdots, \mu_m\}$ the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor
- Let $\Theta = \{\theta_1, \cdots, \theta_m\}$ be the set of plane parameters with

$$\omega(p, \theta_i) = \alpha_i p_x + \beta_i p_y + \gamma_i$$

- We can define the total energy of a pixel as

$$E(p) = E_{\text{col}}^{t,t+1}(\mathbf{p}, c_{s_p}, \theta_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p}) + \lambda_{\text{vz}} E_{\text{vz}}^{t,t+1}(\mathbf{p}, \theta_{s_p}),$$

  with $E_{\text{vz}}^{t,t+1}(\mathbf{p}, \theta_{s_p}) = (\omega(p, \theta_{s_p}) - \omega_{SGM})^2$
- Joint unsupervised segmentation and flow estimation by

$$\min_{\Theta, \mathbf{S}, \mu, \mathbf{c}} \sum_{\mathbf{p}} E(\mathbf{p}, s_p, \theta_{s_p}, \mu_{s_p}, c_{s_p}).$$

Joint Segmentation and Epipolar Flow

- Let $\mathbf{S} = \{s_1, \cdots, s_m)$ be the set of superpixel **assignments**, $\mu = \{\mu_1, \cdots, \mu_m\}$ the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor
- Let $\Theta = \{\theta_1, \cdots, \theta_m\}$ be the set of plane parameters with

$$\omega(p, \theta_i) = \alpha_i p_x + \beta_i p_y + \gamma_i$$

- We can define the total energy of a pixel as

$$E(p) = E_{\text{col}}^{t,t+1}(\mathbf{p}, c_{s_p}, \theta_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p}) + \lambda_{\text{vz}} E_{\text{vz}}^{t,t+1}(\mathbf{p}, \theta_{s_p}),$$

  with $E_{\text{vz}}^{t,t+1}(\mathbf{p}, \theta_{s_p}) = (\omega(p, \theta_{s_p}) - \omega_{SGM})^2$
- Joint unsupervised segmentation and flow estimation by

$$\min_{\Theta, \mathbf{S}, \mu, \mathbf{c}} \sum_{\mathbf{p}} E(\mathbf{p}, s_p, \theta_{s_p}, \mu_{s_p}, c_{s_p}).$$

- Simple iterative algorithm
  - Solve for the assignments $\mathbf{S}$
  - Solve in parallel for the planes Θ, positions μ and appearances $\mathbf{c}$

Joint Segmentation and Epipolar Flow

- Let $\mathbf{S} = \{s_1, \cdots, s_m)$ be the set of superpixel **assignments**, $\mu = \{\mu_1, \cdots, \mu_m\}$ the mean **location** of each superpixel, and $\mathbf{c} = \{c_1, \cdots, c_m\}$ as the mean **appearance** descriptor
- Let $\Theta = \{\theta_1, \cdots, \theta_m\}$ be the set of plane parameters with

$$\omega(p, \theta_i) = \alpha_i p_x + \beta_i p_y + \gamma_i$$

- We can define the total energy of a pixel as

$$E(p) = E_{\text{col}}^{t,t+1}(\mathbf{p}, c_{s_p}, \theta_{s_p}) + \lambda_{\text{pos}} E_{\text{pos}}(\mathbf{p}, \mu_{s_p}) + \lambda_{\text{vz}} E_{\text{vz}}^{t,t+1}(\mathbf{p}, \theta_{s_p}),$$

with $E_{\text{vz}}^{t,t+1}(\mathbf{p}, \theta_{s_p}) = (\omega(p, \theta_{s_p}) - \omega_{SGM})^2$

- Joint unsupervised segmentation and flow estimation by

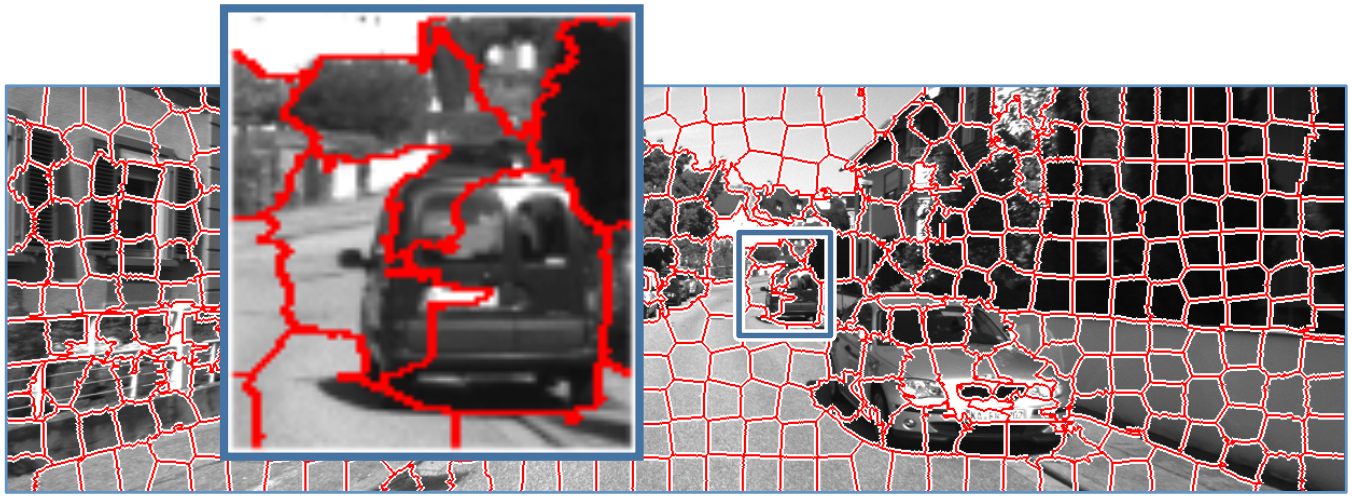
$$\min_{\Theta, \mathbf{S}, \mu, \mathbf{c}} \sum_{\mathbf{p}} E(\mathbf{p}, s_p, \theta_{s_p}, \mu_{s_p}, c_{s_p}).$$

- Simple iterative algorithm
  - Solve for the assignments $\mathbf{S}$
  - Solve in parallel for the planes Θ, positions μ and appearances $\mathbf{c}$

Motion Slic

SLIC

Energy = Position
+ Color



MotionSLIC

Energy = Position
+ Color
+ VZ-ratio

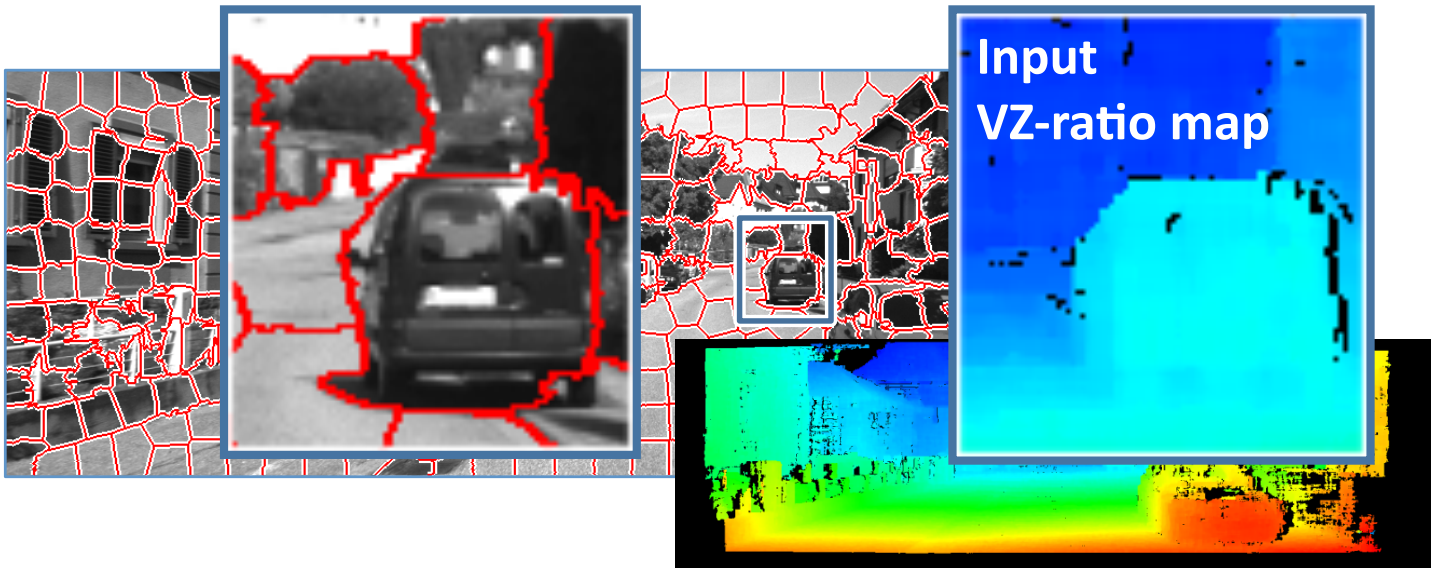
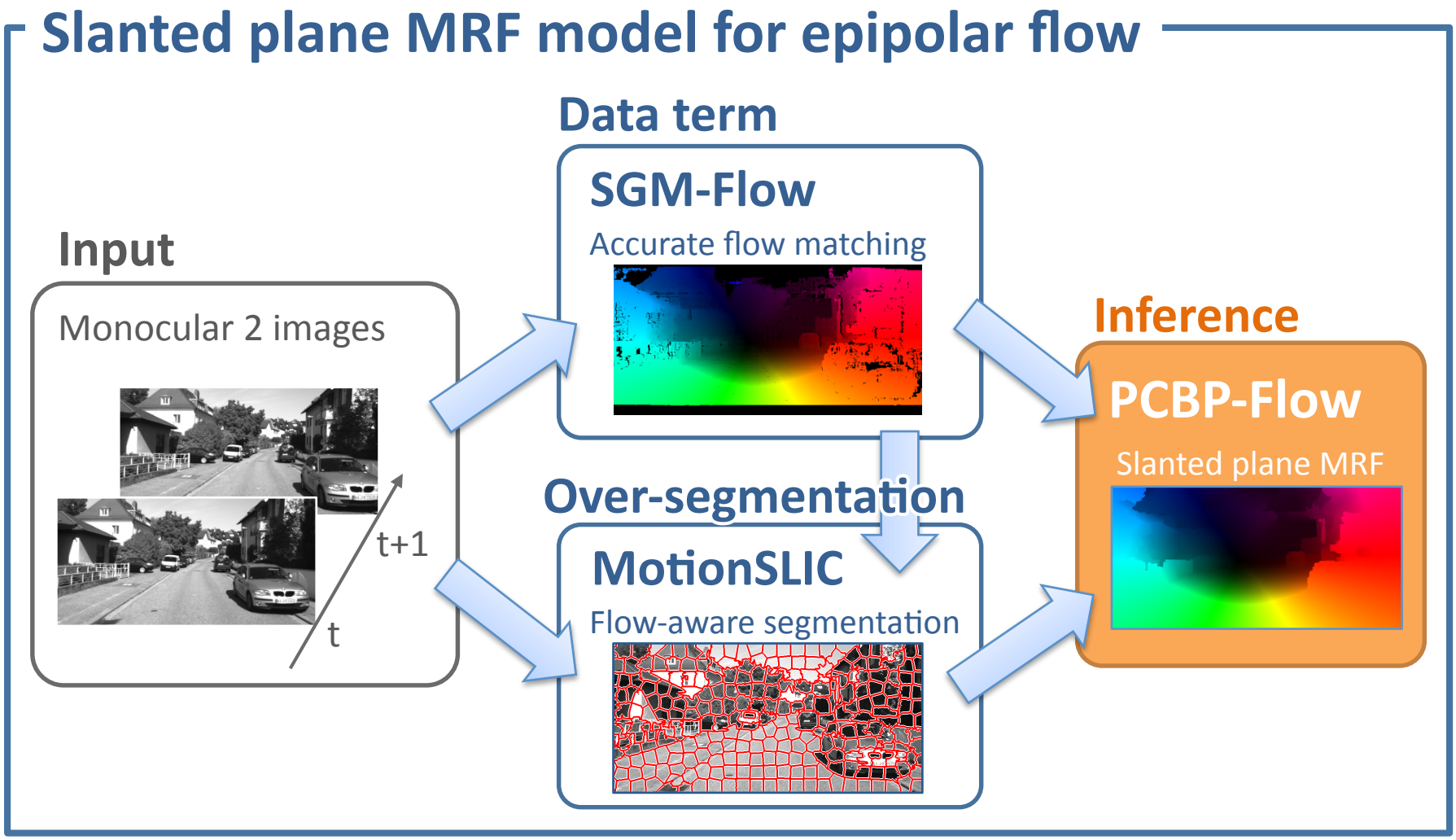
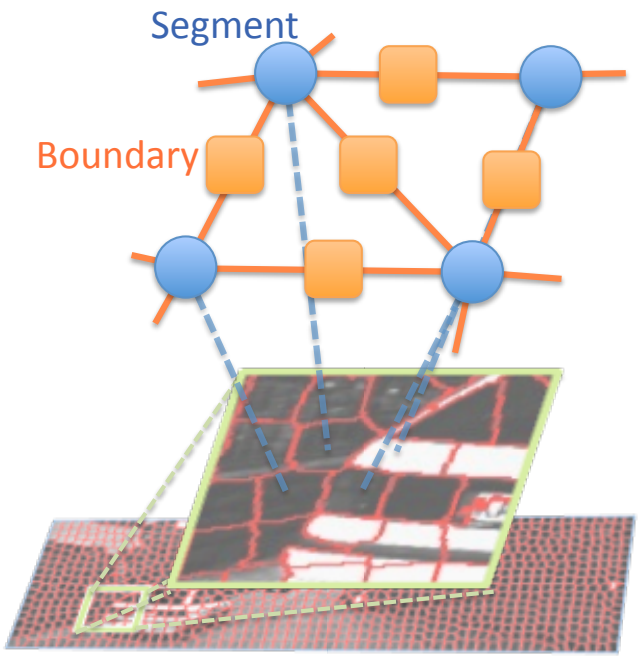


Diagram Flow



Slanted-Plane MRFs for Flow

- Slanted-plane MRF with explicit occlusion handling
- Start with an over-segmentation of the image into superpixels
- MRF on continuous variables (slanted planes) and discrete var. (boundary)



Superpixels (UCM [Arbelaez, et al. 2011] and SLIC [Achanta, et al. 2010])

Segment variable $\mathbf{y}_i = (\alpha_i, \beta_i, \gamma_i)$

Slanted 3D plane of segment

Continuous variable

Boundary variable o_{ij}

Relationship between segments

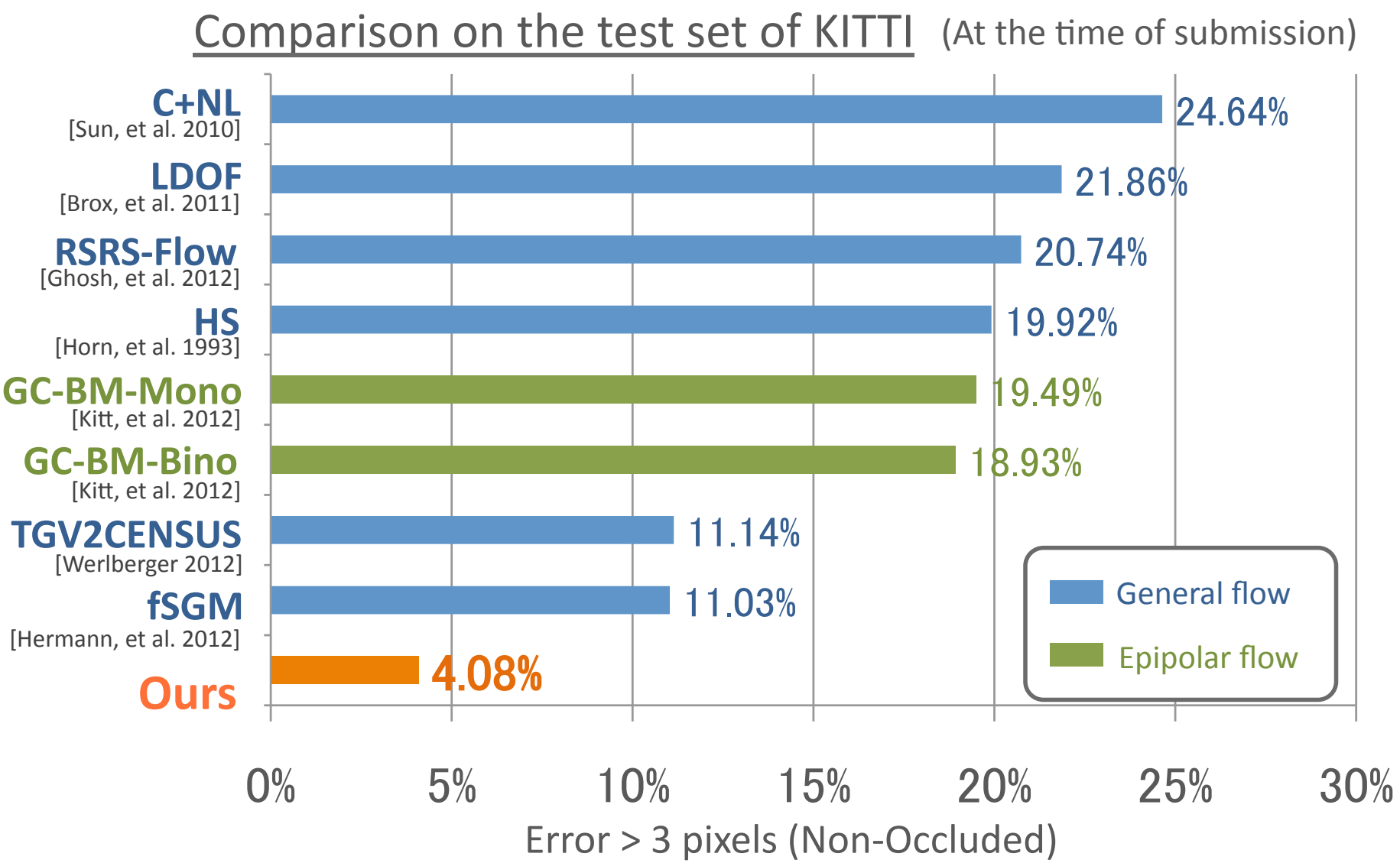
4 states

Occlusion Hinge Coplanar

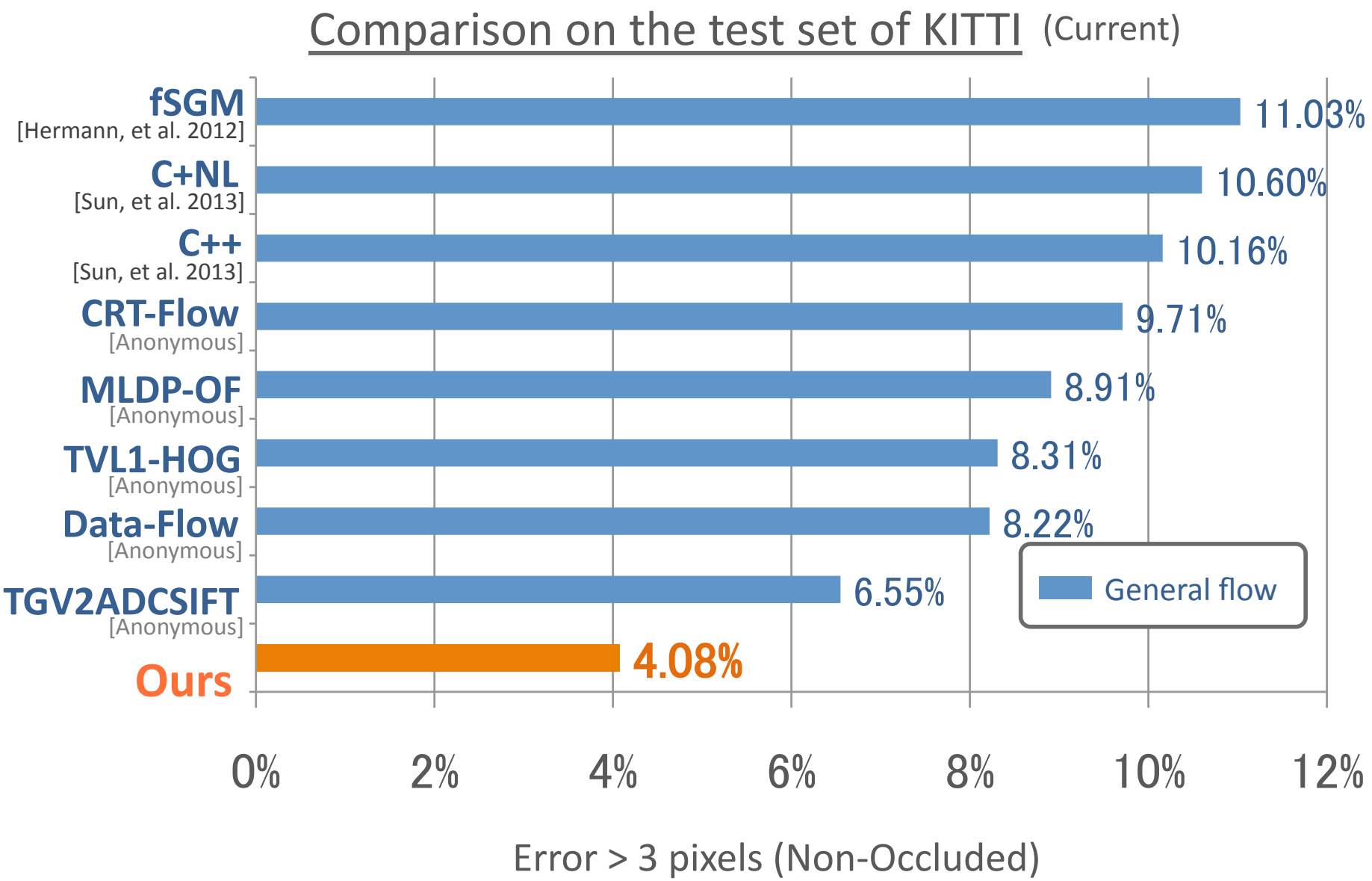
Discrete variable

- Takes as input VZ-ratio SGM for flow

Results KITTI



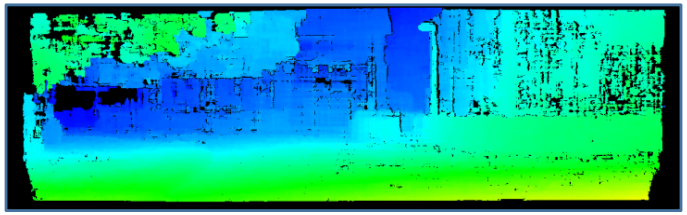
Results KITTI

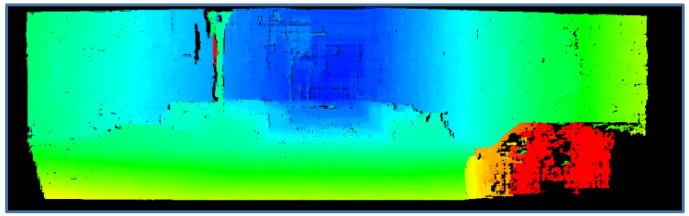


Results KITTI

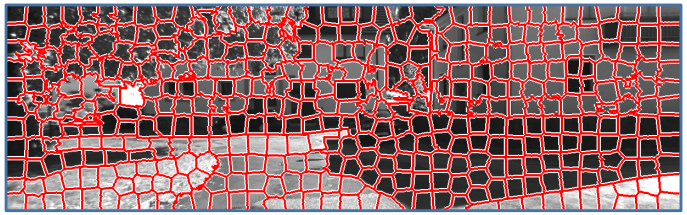
Input image

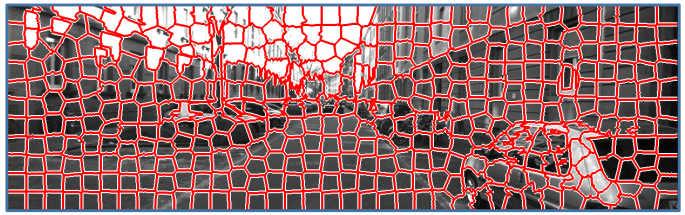
SGM-Flow
VZ-ratio map



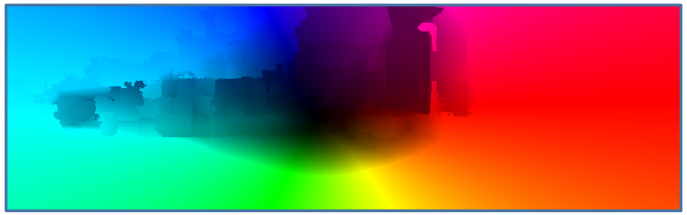


MotionSLIC
Superpixels



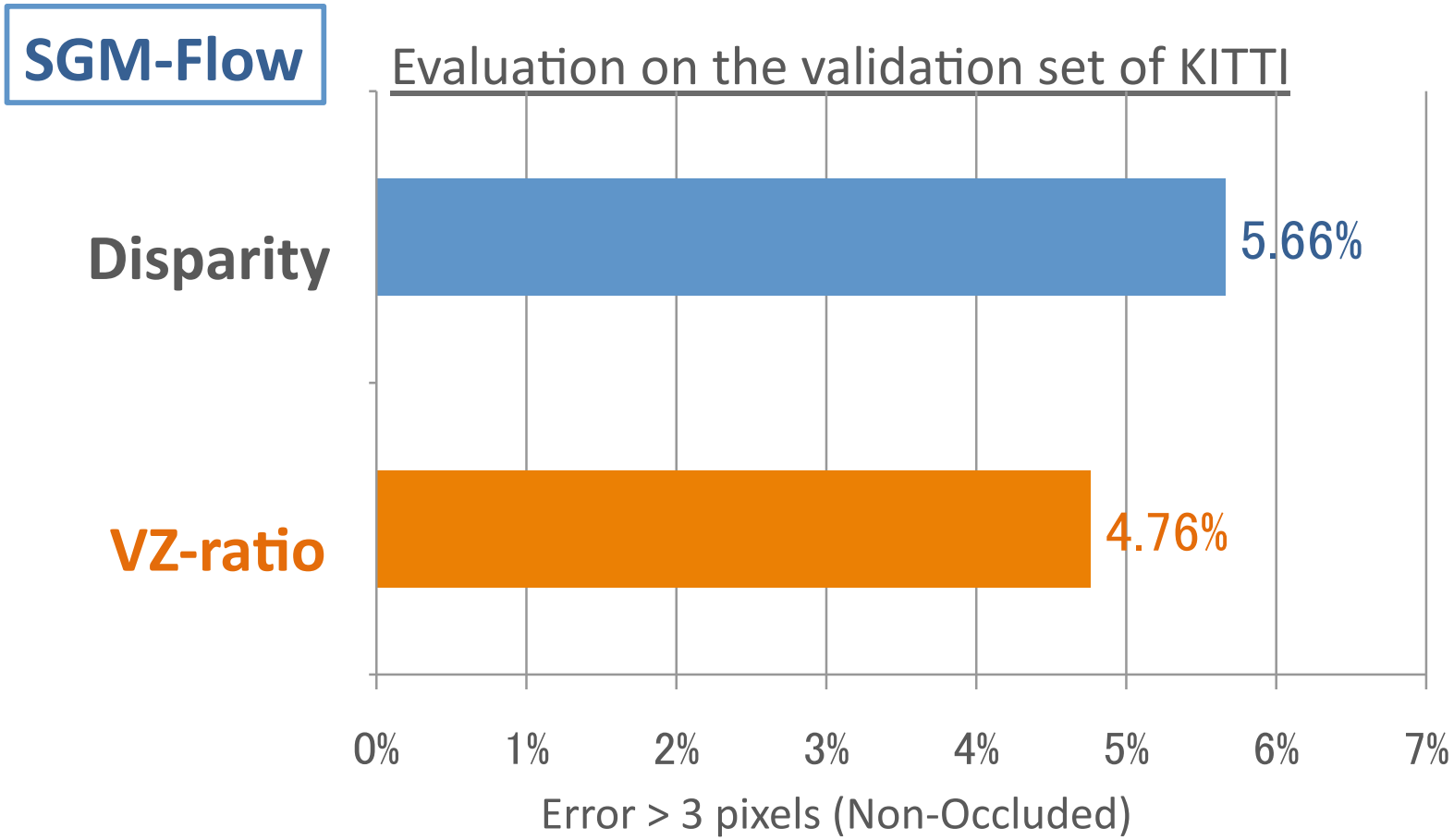


PCBP-Flow
Flow field



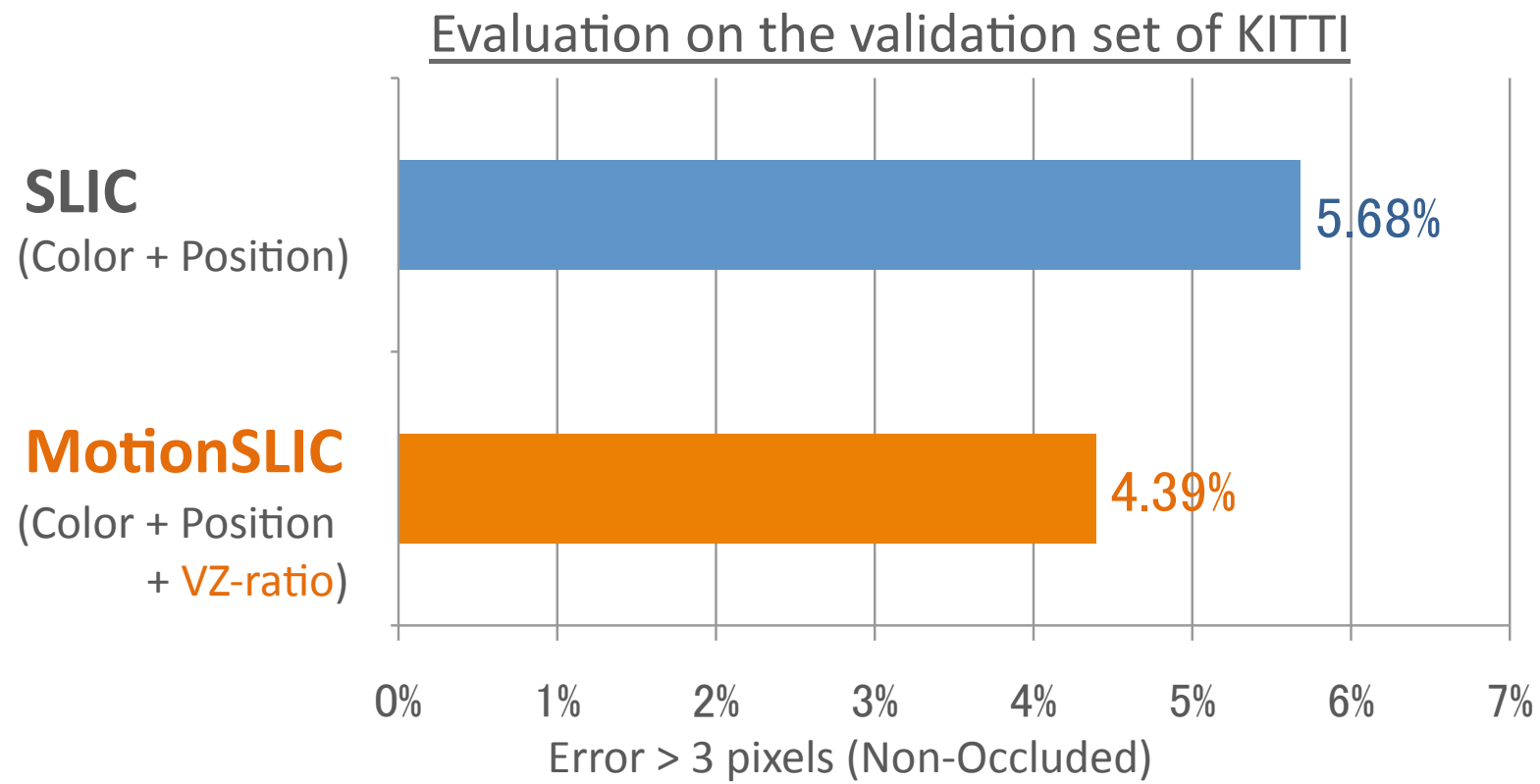
Error

Results KITTI

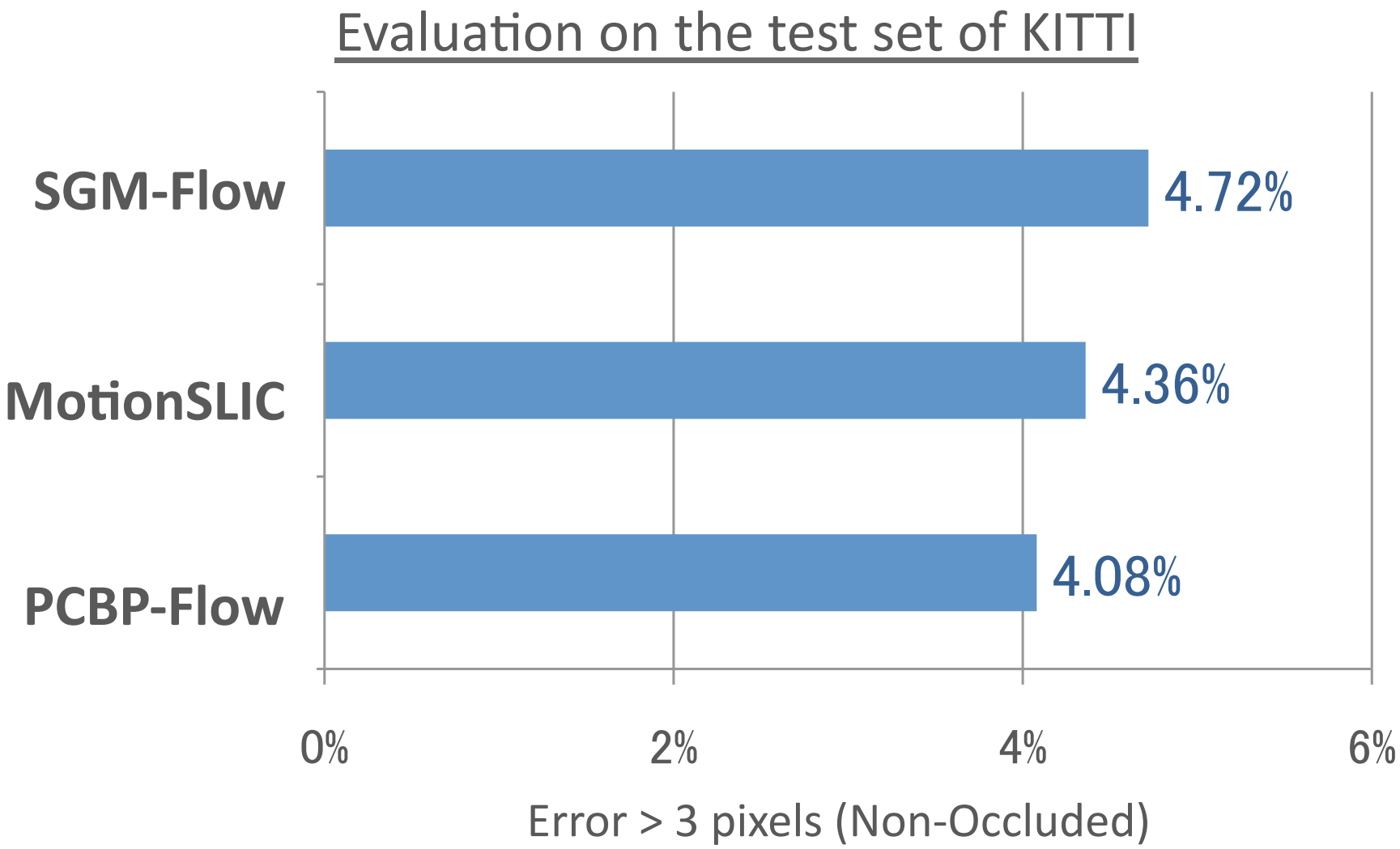


Results KITTI

Superpixel segmentation & Local fitting of slanted planes



Results KITTI



Results of the Challenge

Illumination Changes:

	#44		#11		#15		#74		Average BP
SGM-Flow	16.84%	3.86 px	24.82%	9.45 px	18.90%	4.36 px	26.94%	8.15 px	21.87%
MotionSLIC	9.03%	2.34 px	18.56%	4.78 px	13.18%	3.80 px	23.36%	4.08 px	16.03%
PCBP-Flow	6.42%	2.23 px	15.34%	3.75 px	10.00%	2.42 px	19.15%	3.37 px	12.73%

Large Displacements:

	#147		#117		#144		#181		Average BP
SGM-Flow	9.66%	1.43 px	6.84%	1.64 px	23.10%	6.56 px	29.31%	12.59 px	17.23%
MotionSLIC	9.88%	1.70 px	5.02%	1.47 px	16.65%	3.19 px	22.22%	7.62 px	13.44%
PCBP-Flow	9.02%	1.04 px	6.29%	1.77 px	15.23%	2.59 px	18.32%	7.77 px	12.22%

Next Steps

- Joint **Recognition** and **Reconstruction** (i.e., Stereo / Flow / Scene-flow)
- It will allows us to segment into coherent (piece-wise rigid) motions
- We can impose priors on depth/flow
- It makes recognition much easier
- What is the ultimate goal?

Conclusions

- Data is important for learning and for benchmarking
- Autonomous driving is a fantastic real-world problem to test our algorithms
- Generalization, generalization, generalization!

- Joint segmentation and depth/flow estimation that allows real-time inference
- Slanted plane MRFs to reason properly about occlusion
- State of the art on KITTI stereo and flow

- **Reconstruction meets Recognition Challenge** at ICCV 13, composed of KITTI and NYU-RGBD datasets

Acknowledgements

- Andreas Geiger $\rightarrow$ KITTI
- Tamir Hazan $\rightarrow$ MRF learning and inference
- Philip Lenz $\rightarrow$ KITTI
- David McAllester $\rightarrow$ Stereo/flow and MRF inference
- Alex Schwing $\rightarrow$ MRF learning and inference
- Koichiro Yamaguchi $\rightarrow$ Stereo/flow

Conclusions

Autonomous systems should

- Sense the environment: stereo, flow, layout estimation
- Recognize the 3D world: detection, segmentation
- Interact with it

I advocate for **MRF holistic models** which require

- Representation
- Learning
- Inference
- Data

Build holistic models to properly reason about uncertainty of the different tasks

An autonomous system has to self-localize

Self-localization

- **GPS** does not work all the time, or has errors. Good if a human drives, but not if is an autonomous system!
- **Place recognition approaches**: drive once manually, and try to recognize where you are (e.g., point clouds or visual features).
- Very successful (e.g., Google car), but has scaling and privacy issues (e.g., Germany).
- Humans use a **cartographic map** and look around
- Can we do something similar?

Visual GPS

[M. Brubaker, A. Geiger and R. Urtasun, best paper runner-up at CVPR13]

- Visual Self-localization (1 or 2 cameras) and a map of the environment
- Utilize only visual odometry
- Localization in places you have **NOT** seen or driven before
- Validated in KITTI visual odometry (50km of driving)
- We can self localize in only a few seconds of driving with precision of 3m in maps that contain 2,150km of drivable roads!