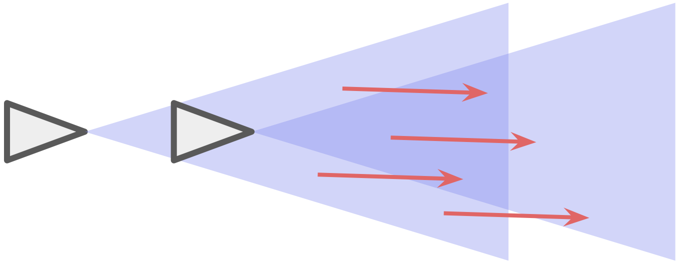
SLAM
Simultaneous Localization and Mapping

CSC 2541 -- Visual Perception for Autonomous Driving
Presented by Kirk MacTavish

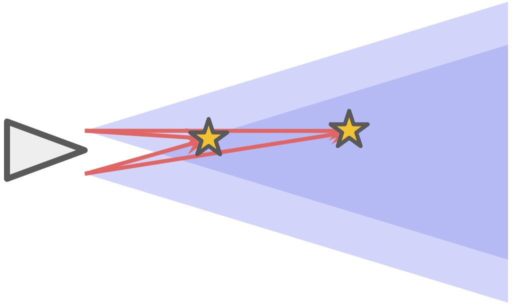
Outline

- How does SLAM fit in?
- Relative continuous-time SLAM
  - Motivation
  - Cubic B-splines
  - Gaussian Process Regression
- Long-term lidar SLAM
  - Map Maintenance
  - Scene Flow

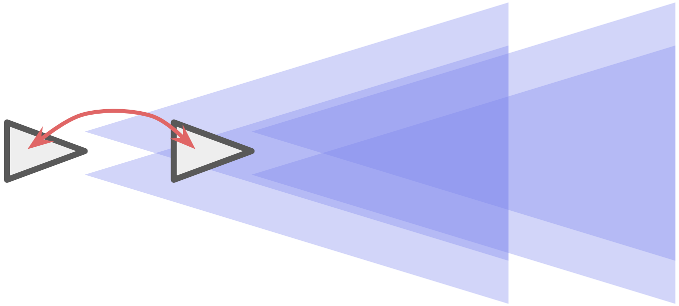
How does SLAM fit in?



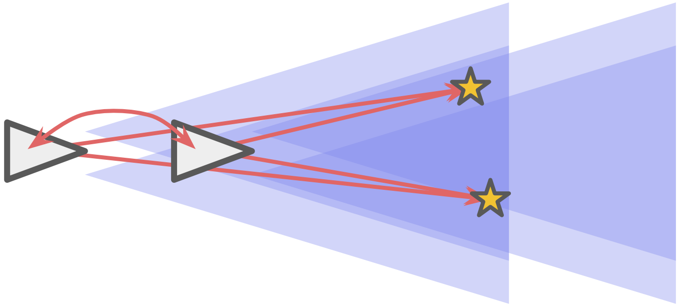
Optical Flow (2D) and Scene Flow (3D)



Stereo

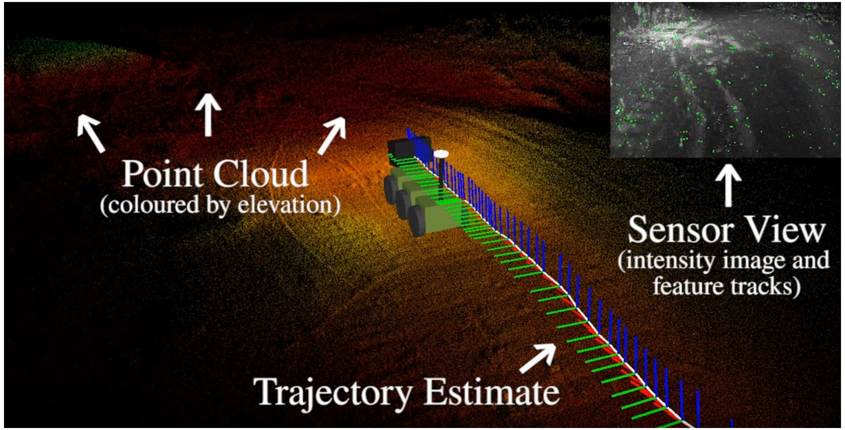


Visual Odometry



SLAM

How does SLAM fit in?



SLAM aims to:

- build an accurate map of the world
- localize the camera within that world

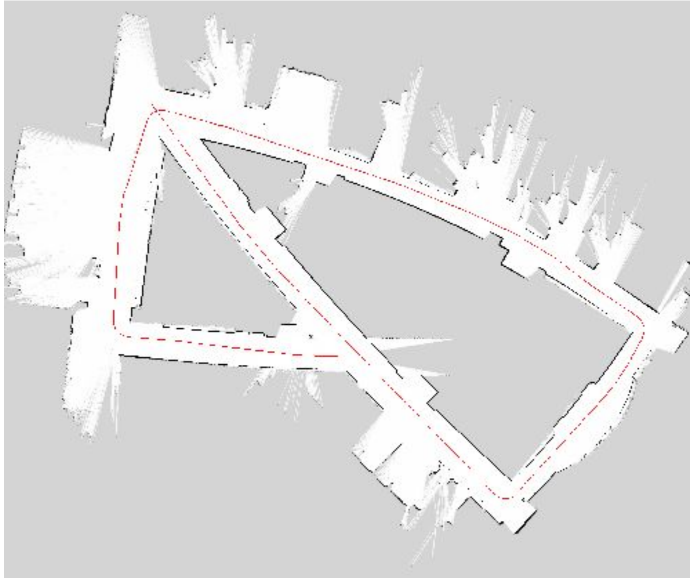
Some defining characteristics:

- The *map is used over an extended period* (for loop closure, a localization reference, for survey) as opposed to VO, which only uses it instantaneously.
- The *egomotion of the vehicle is estimated* as opposed to scene flow/optical flow, which are more concerned with the motion at each pixel.

How does SLAM fit in? - Terminology

loop closure: identifying when the camera is revisiting a previous location.

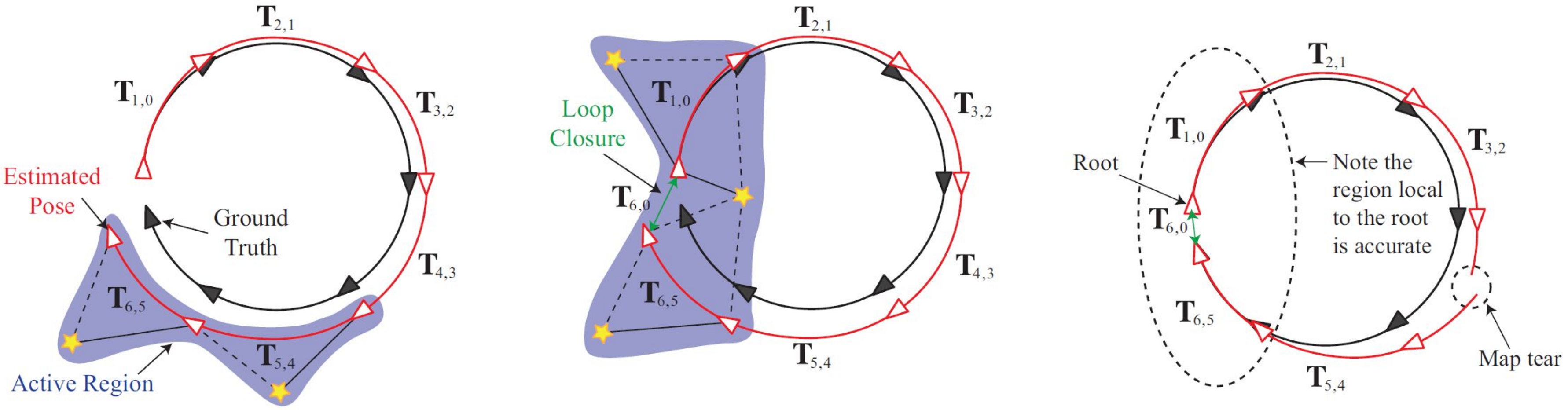
image credit: Michael Kaess



Relative Continuous-time SLAM - Motivation

We often only need a relative map—not a single privileged coordinate frame.

Can perform loop closure in constant time, not growing with the size of our map.



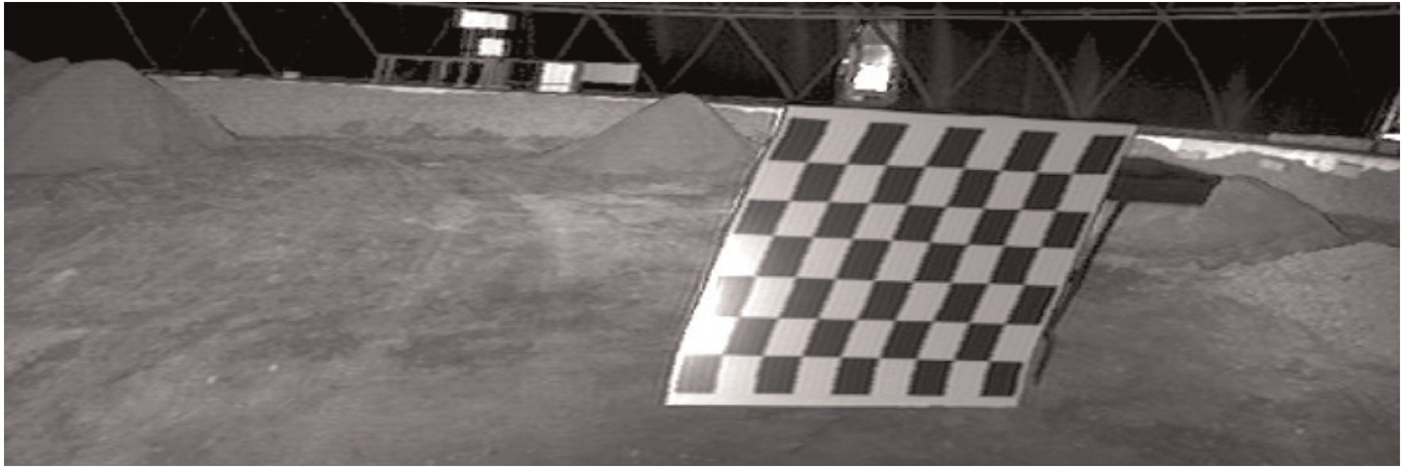
(a) Sliding Window Filter (b) Loop-Closure Optimization (c) Map Consistency

Relative Continuous-time SLAM - Motivation

Discrete-time estimation makes it difficult to deal with

- high-rate sensors (e.g., IMU, LIDAR)
- fusion with different-rate sensors (e.g., LIDAR + Camera)

since a discrete pose estimate must be available at each measurement time.



Relative Continuous-time SLAM - Overview

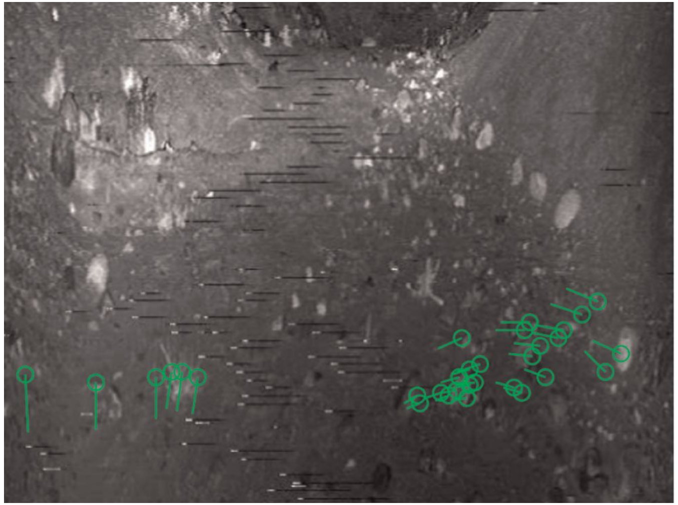
Continuous-time estimation allows us to

- interpolate between state variables (at *key times*) to
  - process arbitrarily-timed measurements
  - query the pose estimate at arbitrary times
- use fewer state variables (at *key times*) to represent the egomotion
  - smooth, predictable motion needs little adjustment from interpolation

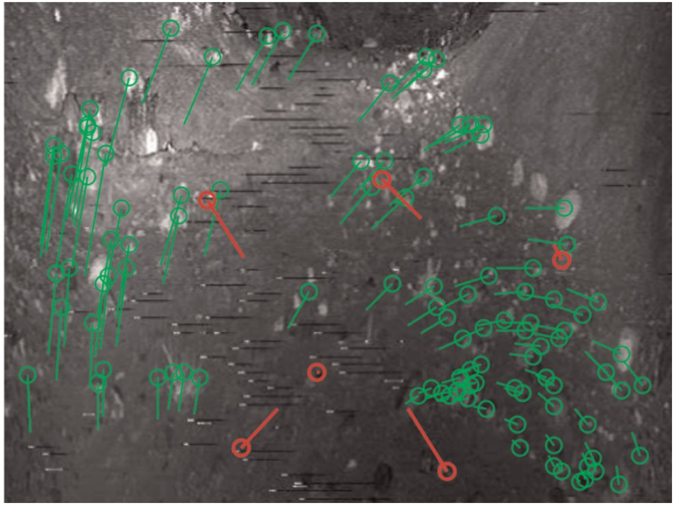
Relative Continuous-time SLAM - Proof of Concept

Anderson, MacTavish et al. *Relative continuous-time SLAM* (IJRR 2015)

An appearance-based lidar algorithm (SURF on intensity images)



Rigid RANSAC



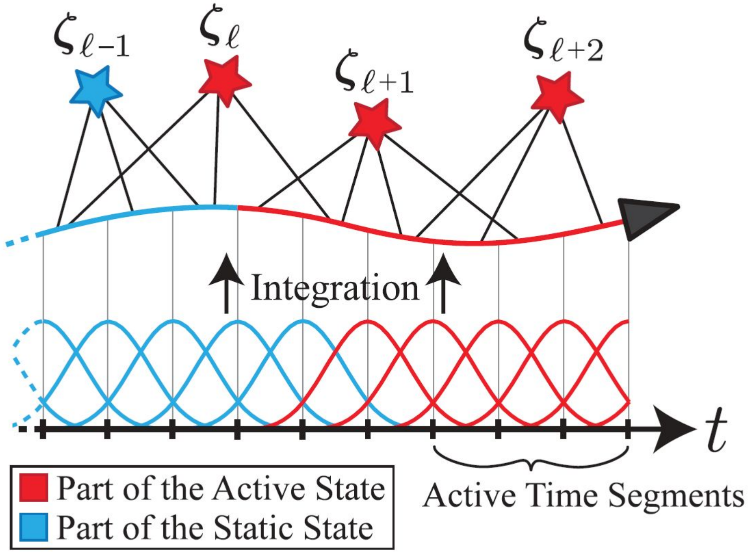
Motion-compensated RANSAC

Relative Continuous-time SLAM - Proof of Concept

Anderson, MacTavish et al. *Relative continuous-time SLAM* (IJRR 2015)

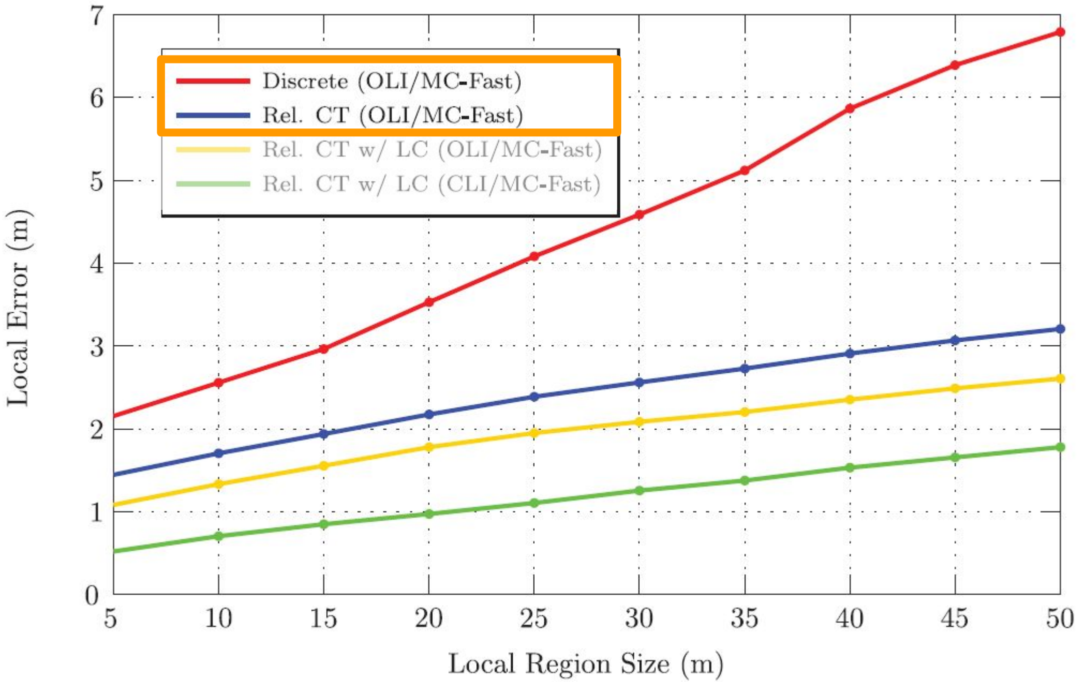
Use weights on cubic B-splines to represent continuous state variables

- Differentiable to the n-th degree
- Local support (only adjust local weights during optimization)
- Implicit trajectory prior is arbitrary :(



Relative Continuous-time SLAM - Proof of Concept

Discrete assumes no distortion, to maintain a tractable number of state variables



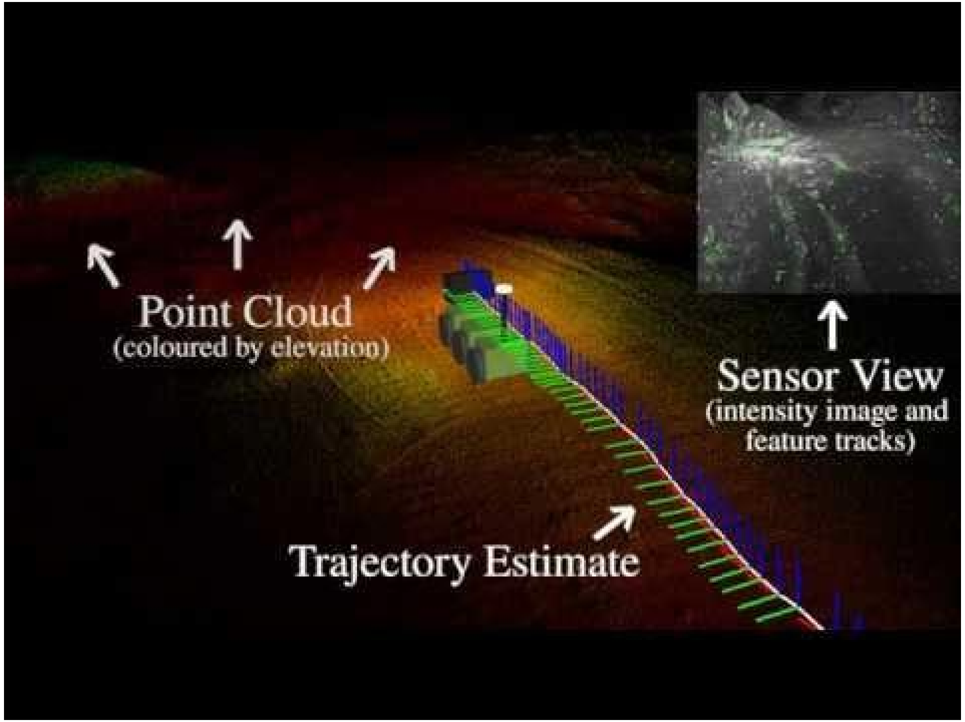
Simultaneous *Trajectory* Estimation and Mapping

Anderson and Barfoot. *Full STEAM Ahead: Exactly Sparse Gaussian Process Regression for Batch Continuous-Time Trajectory Estimation on SE(3)* (IROS 2015)

Instead of cubic B-splines, use Gaussian Process (GP) regression.

- Incredibly slow for dense kernels, but careful selection can result in realistic sparse GP kernels that are very fast
- Interpolates between conventional state parameterizations at *key times*.
- The trajectory prior has physical meaning (e.g. constant velocity/acceleration)
- Uncertainty estimates even at interpolated times

Simultaneous *Trajectory* Estimation and Mapping

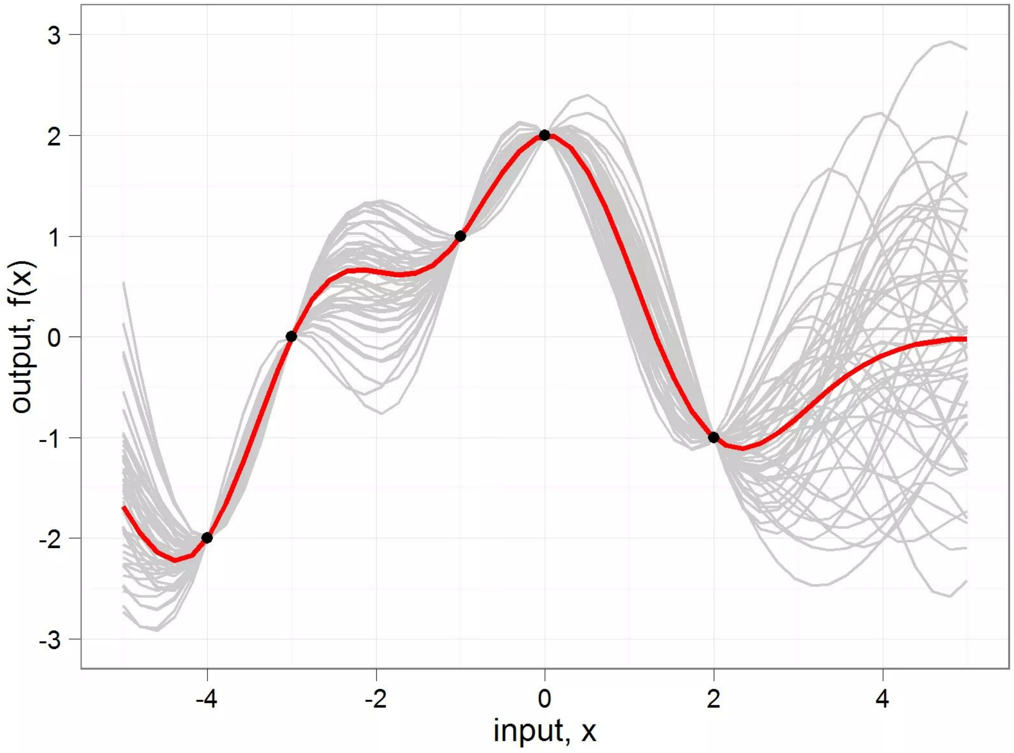


Simultaneous *Trajectory* Estimation and Mapping

What is GP regression?

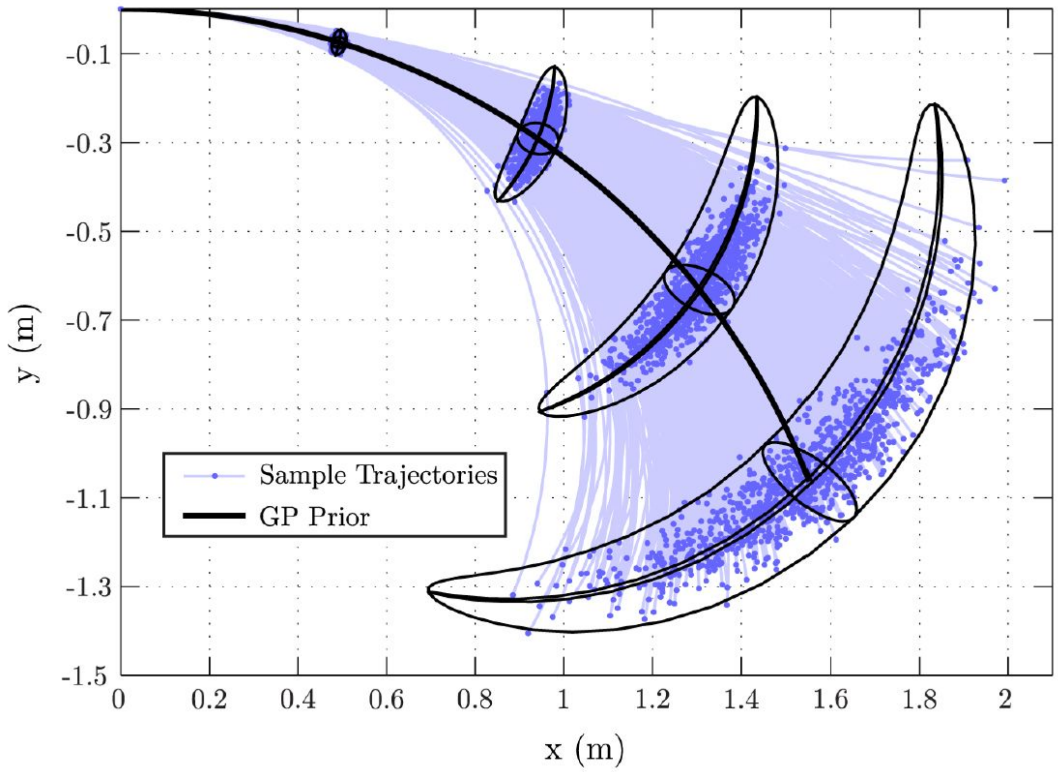
A Gaussian process is a distribution over continuous functions.

Used for regression, represents the posterior likelihood of the state, given the measurements.



Simultaneous *Trajectory* Estimation and Mapping

When applied to SE3
(a way to represent rigid 3D transformations),
this parameterization can represent realistic probabilistic trajectories obeying nonlinear, nonholonomic motion models.

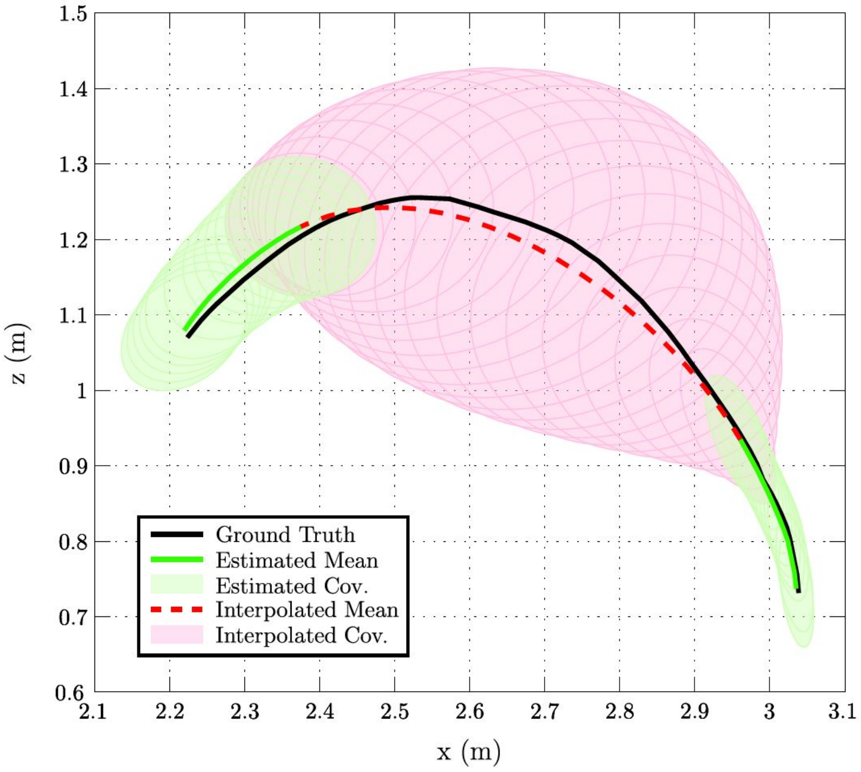


Simultaneous *Trajectory* Estimation and Mapping

Green - measurements are being used for the pose estimate

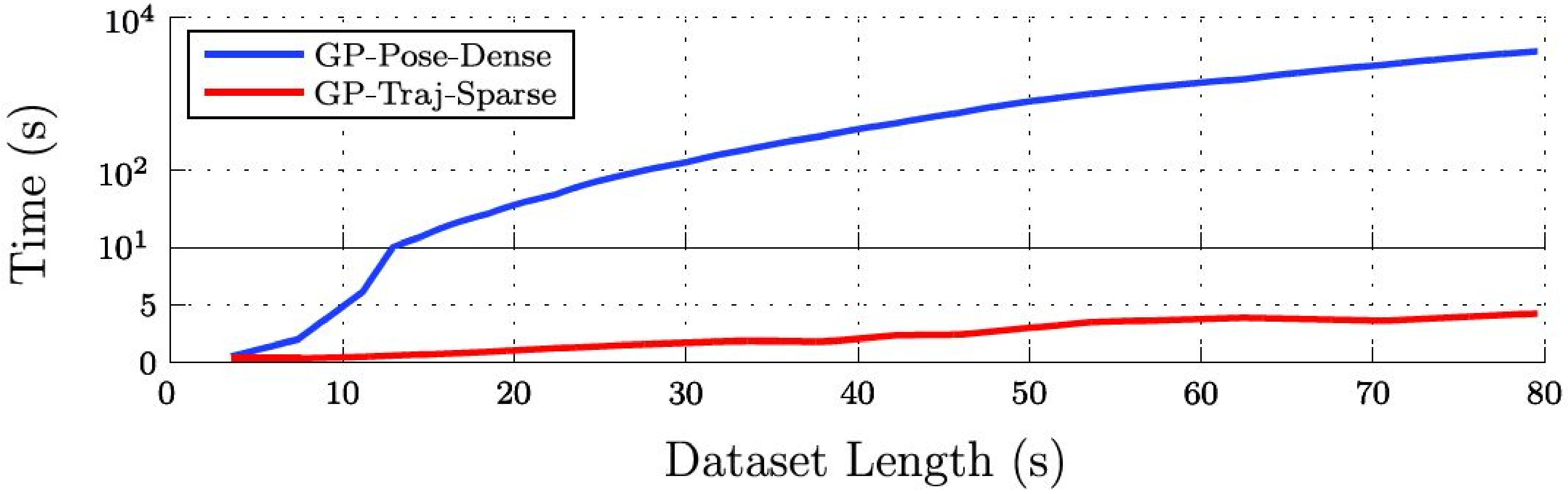
Red - no measurements are used

The interpolation is performs well, and the uncertainty grows the further the interpolation is from evidence.



Simultaneous *Trajectory* Estimation and Mapping

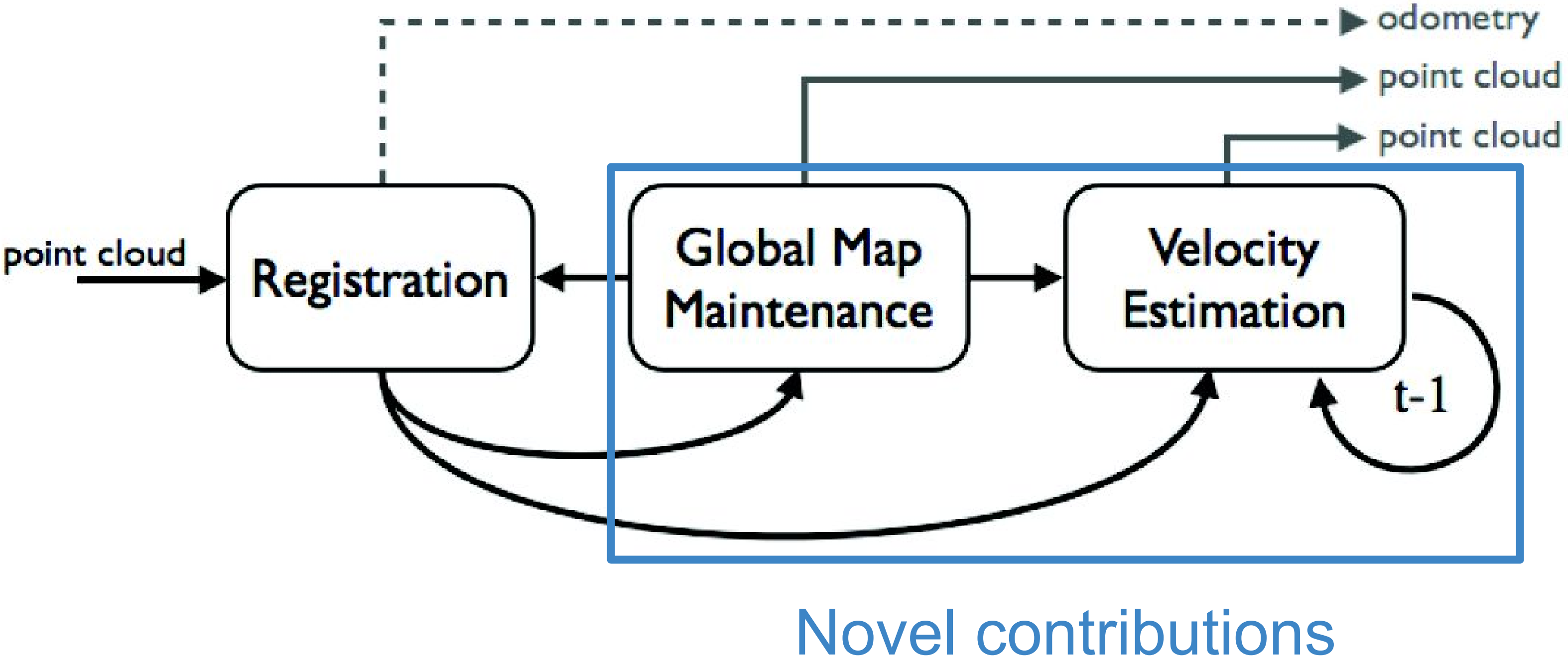
The sparse kernel, based on a realistic motion model, allows the GP regression to be very fast (easily real-time)



Long-term Lidar SLAM

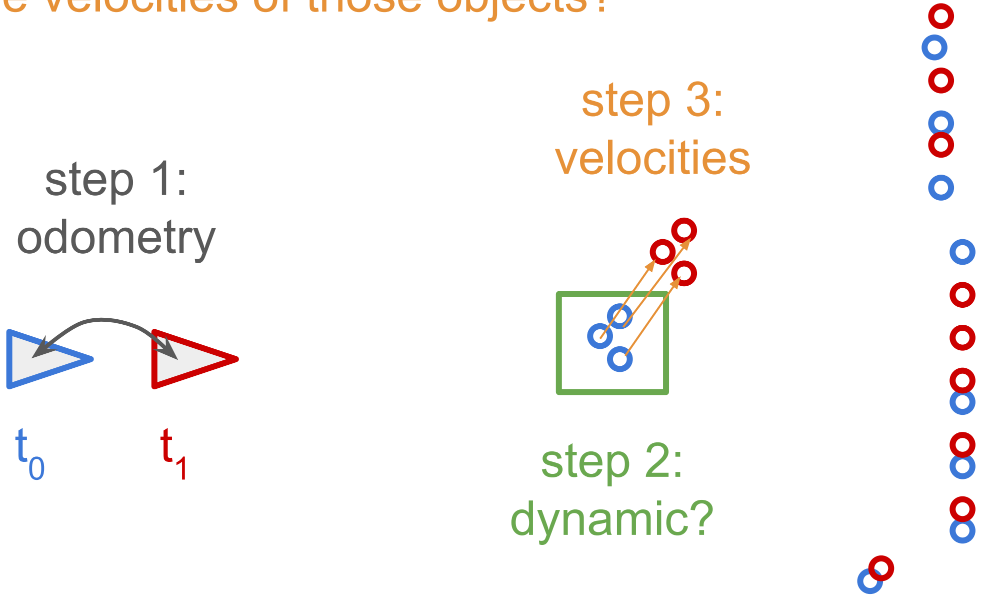
Pomerleau, Krusi et al. *Long-term 3D map maintenance in dynamic environments* (ICRA 2014)

Long-term Lidar SLAM



Long-term Lidar SLAM

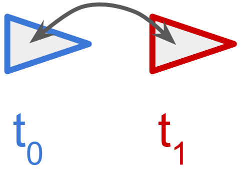
1. Estimate the sensor egomotion to align the recent scan
2. Are any points in the map dynamic objects?
3. What are the velocities of those objects?

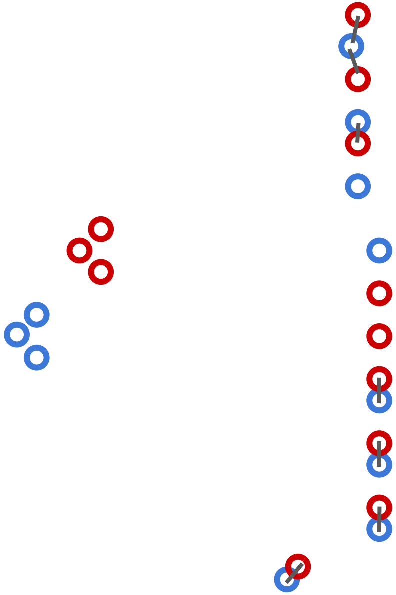


Long-term Lidar SLAM

Odometry: Use ICP to align the current scan to the previous map to estimate sensor egomotion (not the main focus of this paper).

step 1:
odometry

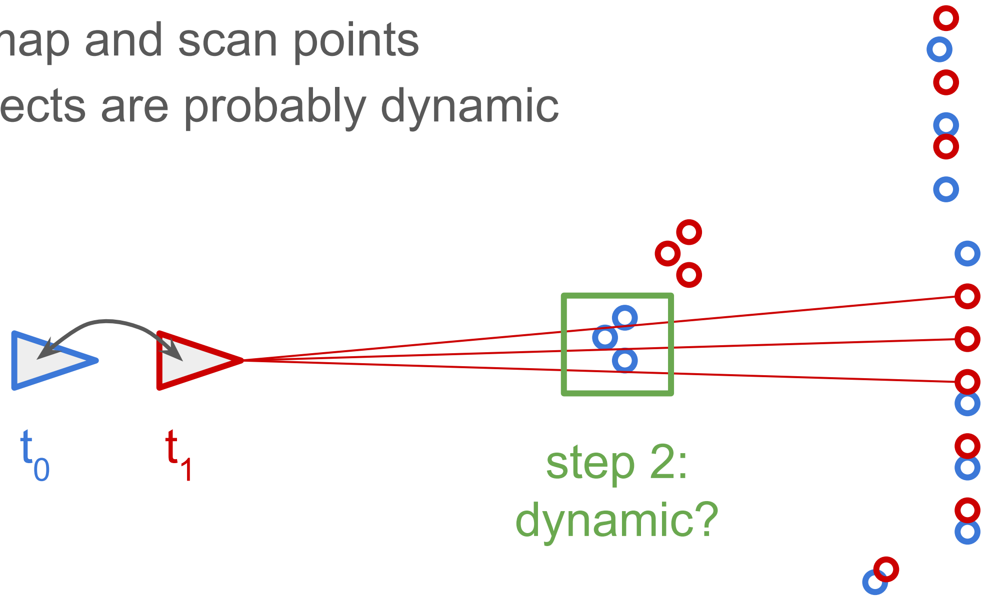




Long-term Lidar SLAM - Map Maintenance

Map maintenance: Figuring out which points are dynamic (and can be removed from the long-term map).

- Ray trace map and scan points
- Missing objects are probably dynamic



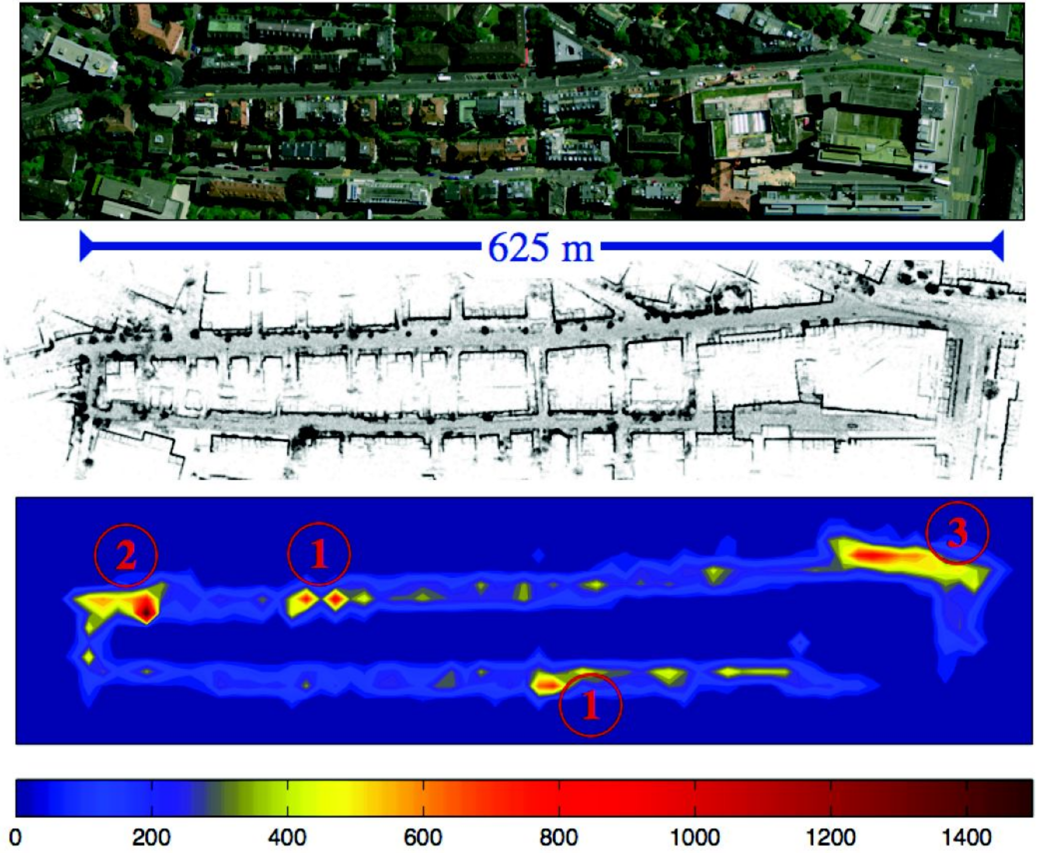
Long-term Lidar SLAM - Map Maintenance

Aerial view of 1.3 km surveyed over 7 months (top)

The static map (middle)

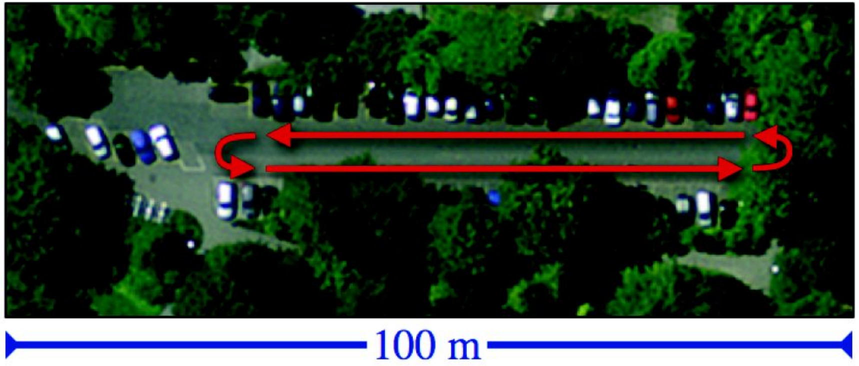
Annotations show the following dynamic scenes (bottom):

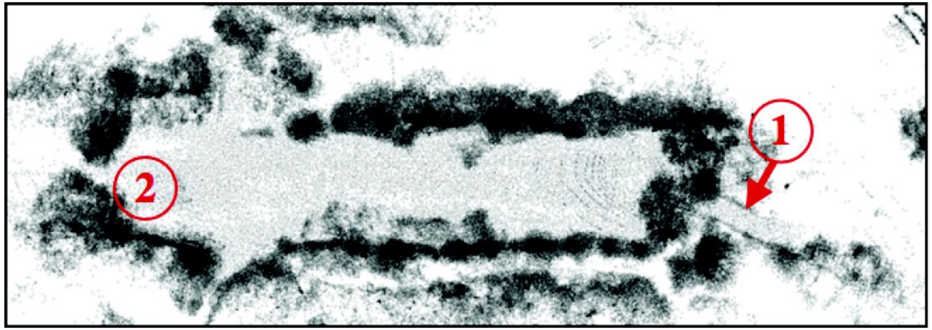
1. Construction sites
2. A large tree
3. A busy intersection

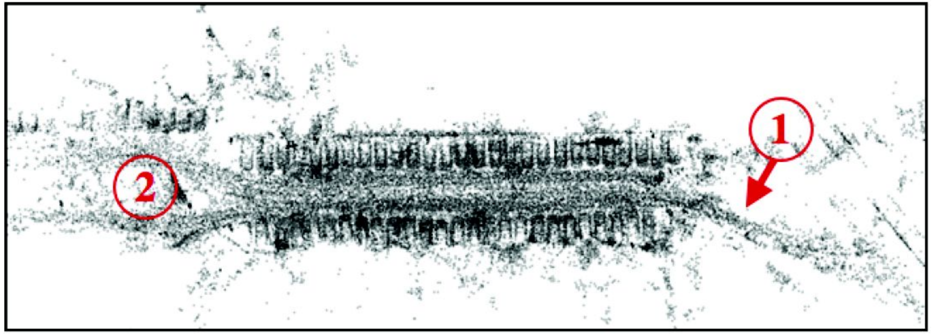


Long-term Lidar SLAM - Map Maintenance

- 9 surveys over 3 days
- classification of static (top right) vs dynamic points (bottom right)

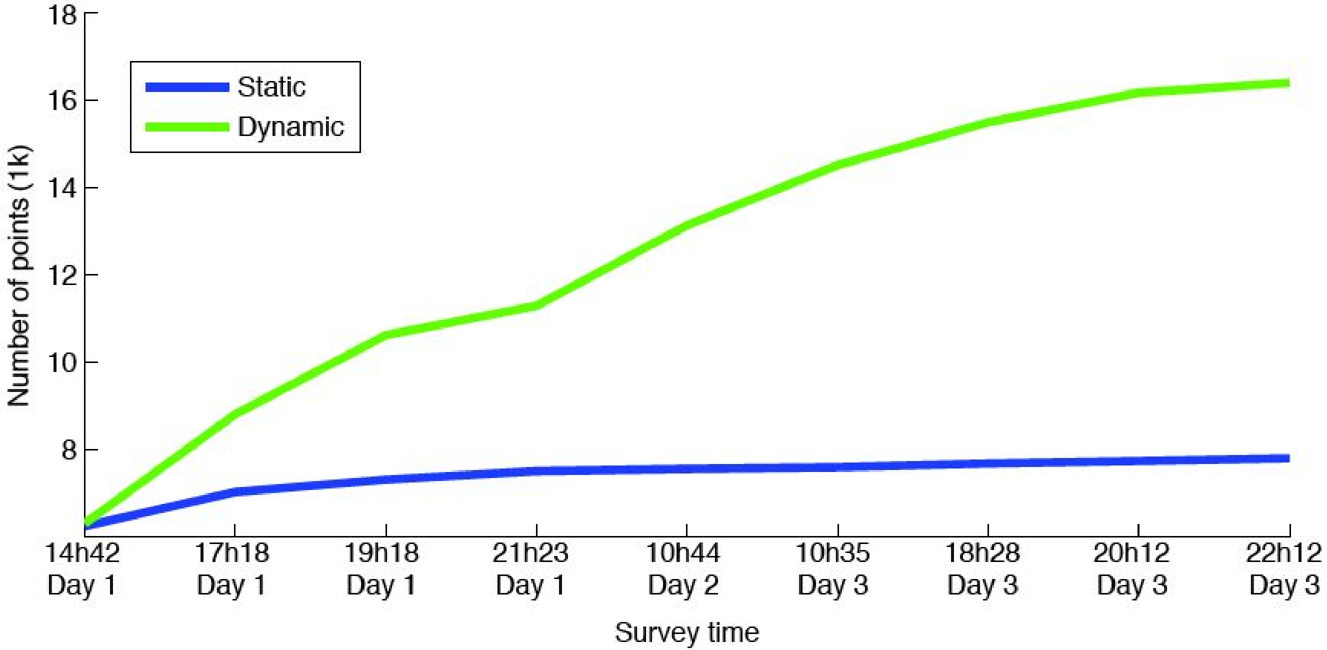






Long-term Lidar SLAM - Map Maintenance

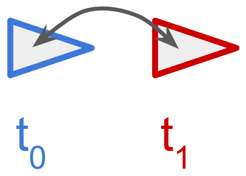
The static map stabilizes very quickly, memory requirements are bounded

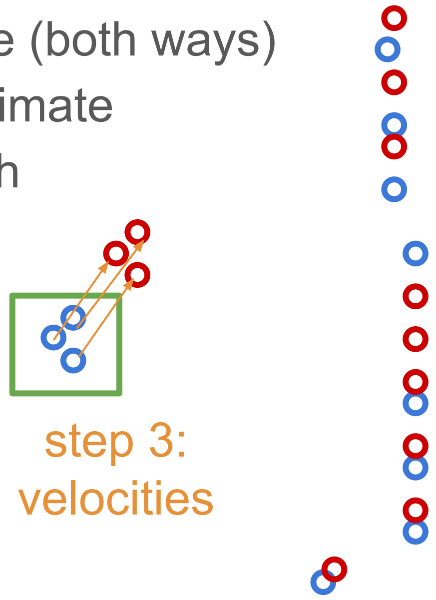


Long-term Lidar SLAM - Scene Flow

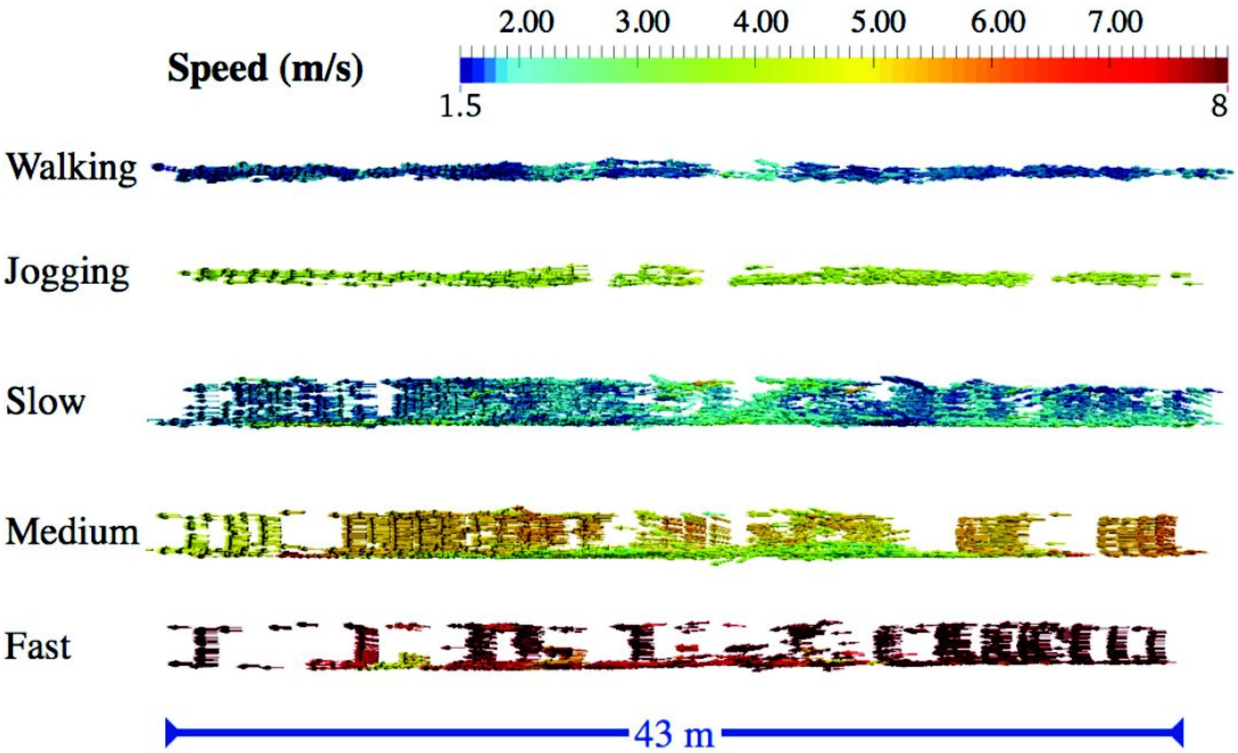
Estimate velocities: for all of the dynamic points, propose assignments to dynamic objects from the previous scan. Iterates the following steps:

1. Project points using previous estimate (both ways)
2. Nearest k neighbours for a robust estimate
3. Use a windowed mean filter to smooth





Long-term Lidar SLAM - Scene Flow



Long-term Lidar SLAM

Pomerleau, Krusi et al. *Long-term 3D map maintenance in dynamic environments* (ICRA 2014)

Dense Techniques for SLAM

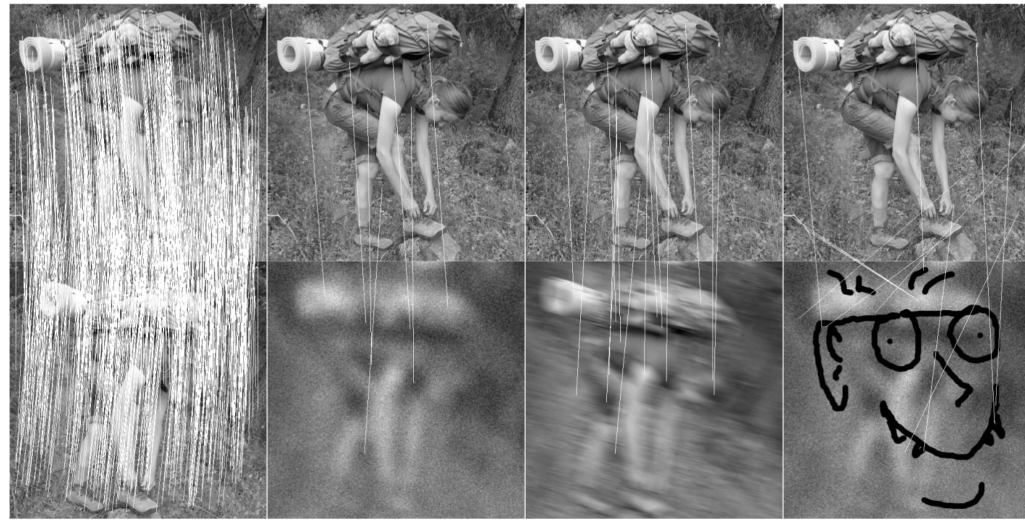
Lingzhu Xiang
Feb 23, 2016

Why Dense?

Robust to scaling and rotation, occlusions, or motion blurs

High quality correspondence

Traditionally considered expensive
→ New ways to reduce computation, or compute on GPU



SIFT feature correspondence failure
Richard Newcombe. Dense Visual SLAM. Thesis 2012.

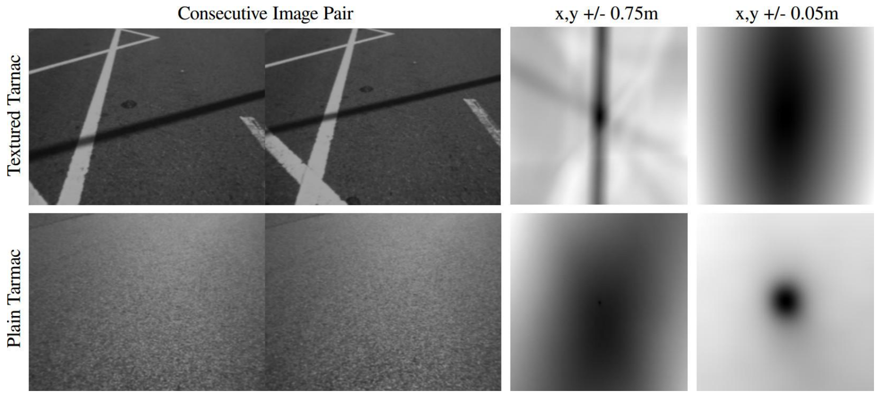
Some Background

Andrew Davison's group at ICL:

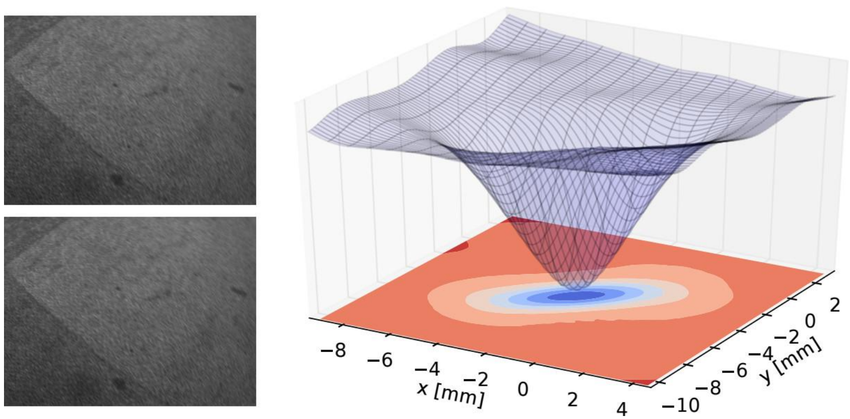
- 2011, Use pixel SSD for VO on SE(2)
- 2013, Dense VO with autocalibration

→ Pose estimation is really possible with dense cost functions!

KinectFusion, ElasticFusion, dense planar VO, articulated models, deformable models, indoor environments
→ Surreal Vision acquired by Oculus



Steven Lovegrove, Andrew Davison, Javier Ibanez-Guzman. Accurate Visual Odometry from a Rear Parking Camera. IV 2011.



Jacek Zienkiewicz, Robert Lukierski, Andrew Davison. Dense, Auto-Calibrating Visual Odometry from a Downward-Looking Camera. BMVC 2013.

DTAM: Dense Tracking and Mapping in Real-Time

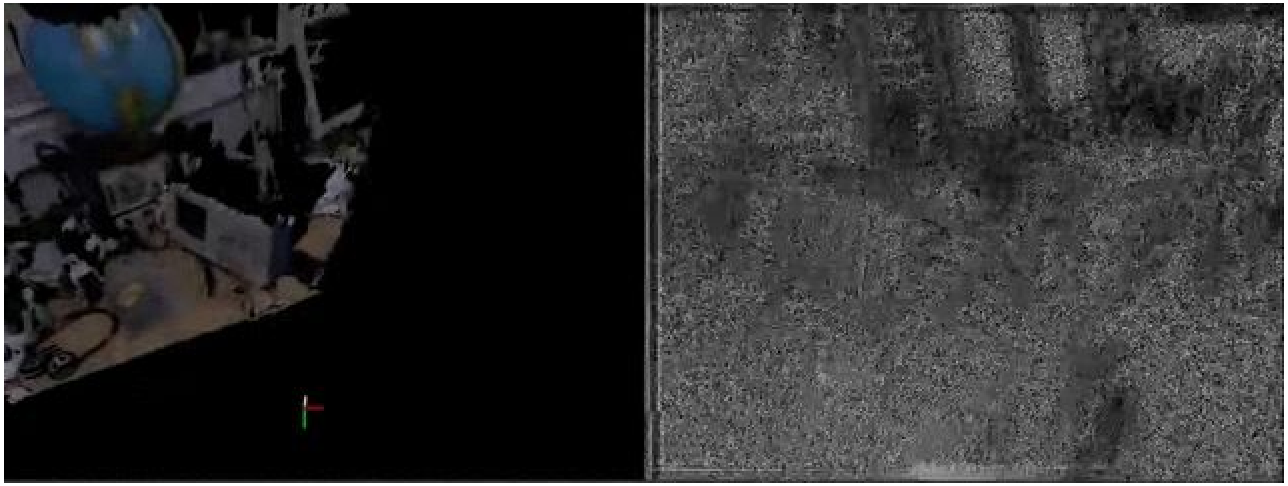
Richard Newcombe, Steven Lovegrove, Andrew Davison - ICCV 2011

Monocular cameras

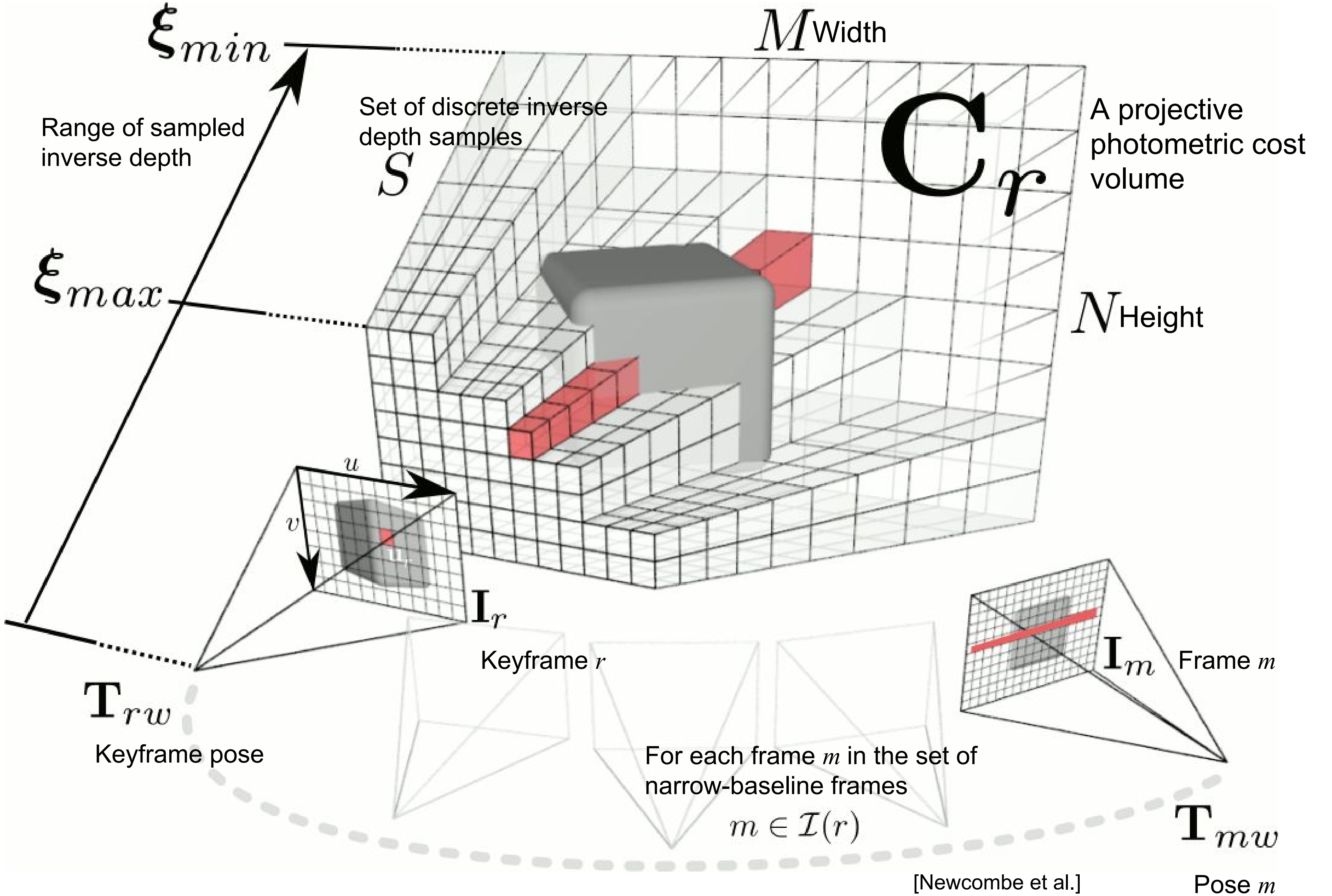
No feature extraction

Superior tracking performance than feature based methods

Oriented “for real-time scene interaction in a physics-enhanced augmented reality application”



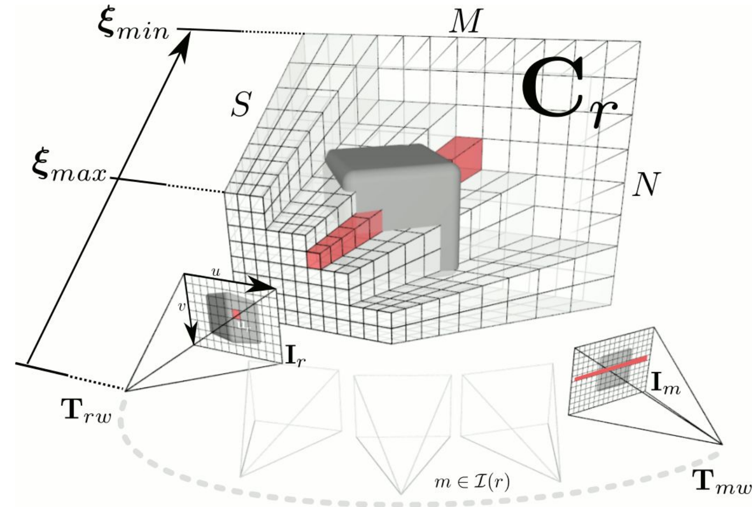
[Newcombe et al.]



$$\mathbf{C}_r(\mathbf{u}, d) = \frac{1}{|\mathcal{I}(r)|} \sum_{m \in \mathcal{I}(r)} \| \rho_r \left(\mathbf{I}_m, \mathbf{u}, d \right) \|_1$$

Pixel coord Inverse depth Number of frames Photometric error of frame $\mathbf{I}_m$, pixel $\mathbf{u}$, depth d

The Photometric Error



[Newcombe et al.]

$$\mathbf{C}_r(\mathbf{u}, d) = \frac{1}{|\mathcal{I}(r)|} \sum_{m \in \mathcal{I}(r)} \|\rho_r\left(\mathbf{I}_m, \mathbf{u}, d\right)\|_1$$

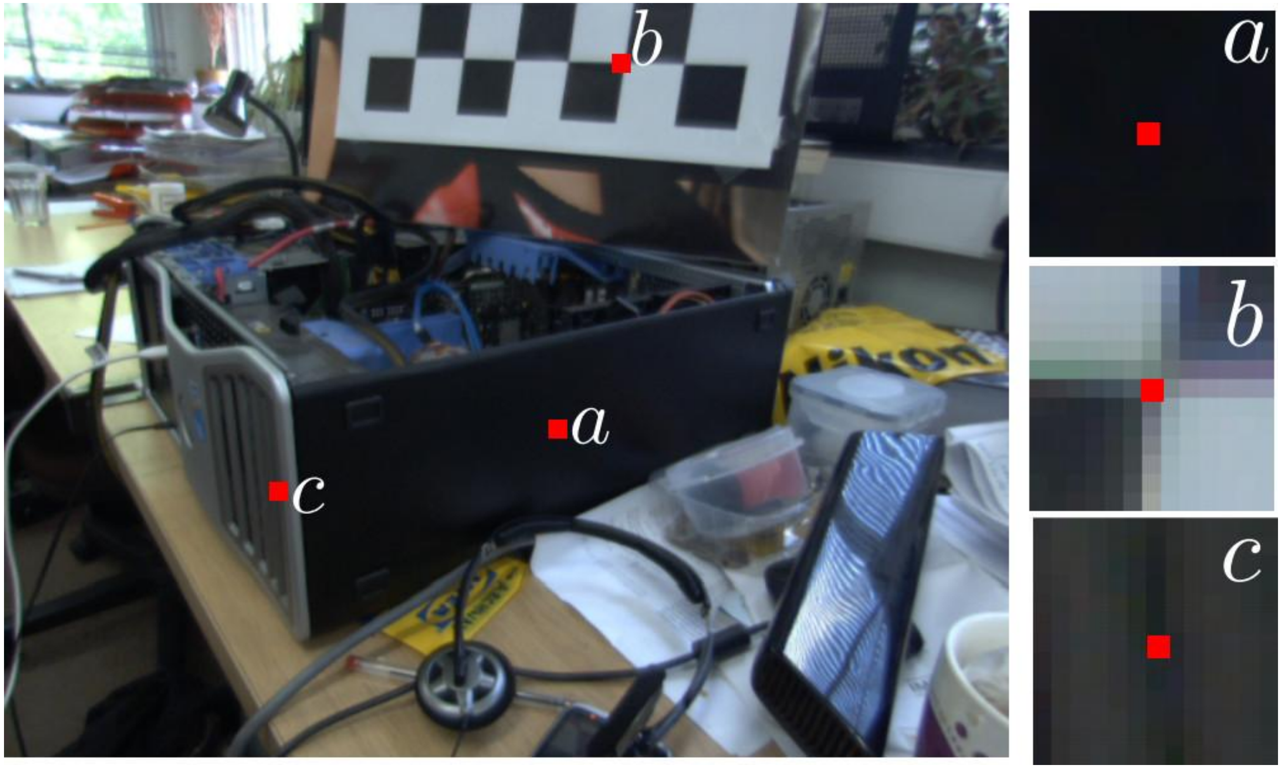
$$\underbrace{\rho_r\left(\mathbf{I}_m, \mathbf{u}, d\right)} = \mathbf{I}_r\left(\mathbf{u}\right) - \mathbf{I}_m\left(\underbrace{\pi\left(\mathrm{KT}_{mr}\pi^{-1}\left(\mathbf{u}, d\right)\right)}\right)$$

Difference of image intensity between keyframe pixel $\mathbf{u}$, and the corresponding pixel in frame $\mathbf{I}_m$

Project 3D point represented by keyframe pixel $\mathbf{u}$, inverse depth d onto frame $\mathbf{I}_m$

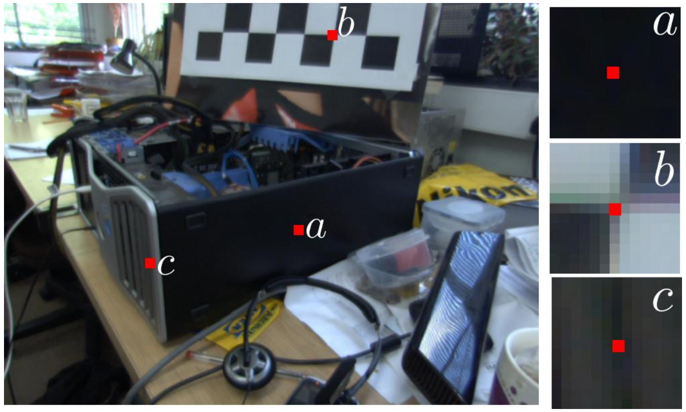
Is the photometric error valid for depth estimation?

Three test cases each with a single pixel:
(a) textureless; (b) strongly textured; (c) linear repeating texture



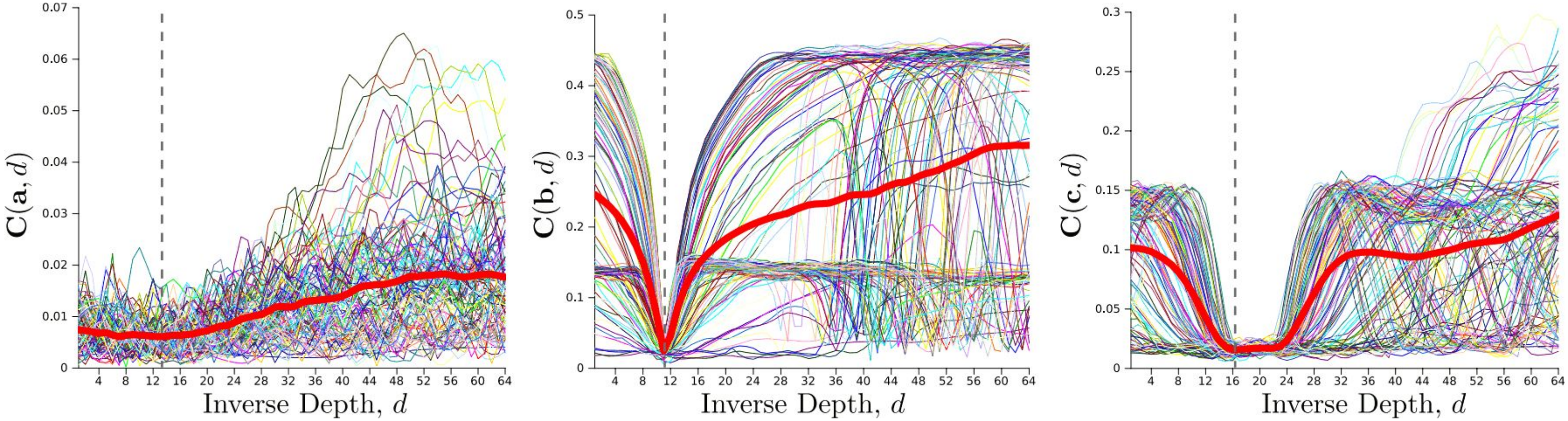
[Newcombe et al.]

Total cost shows clear global minimum except for textureless regions



Inverse depth vs cost plot $\|\rho_r(\mathbf{I}_m, \mathbf{u}, d)\|_1$ for each m $\mathbf{C}_r(\mathbf{u}, d) = \sum_m \|\rho_r(\mathbf{I}_m, \mathbf{u}, d)\|_1$

(a) Textureless (b) Strongly textured (c) Repeating texture



[Newcombe et al.]

Really need many views to avoid local minima in the total cost.

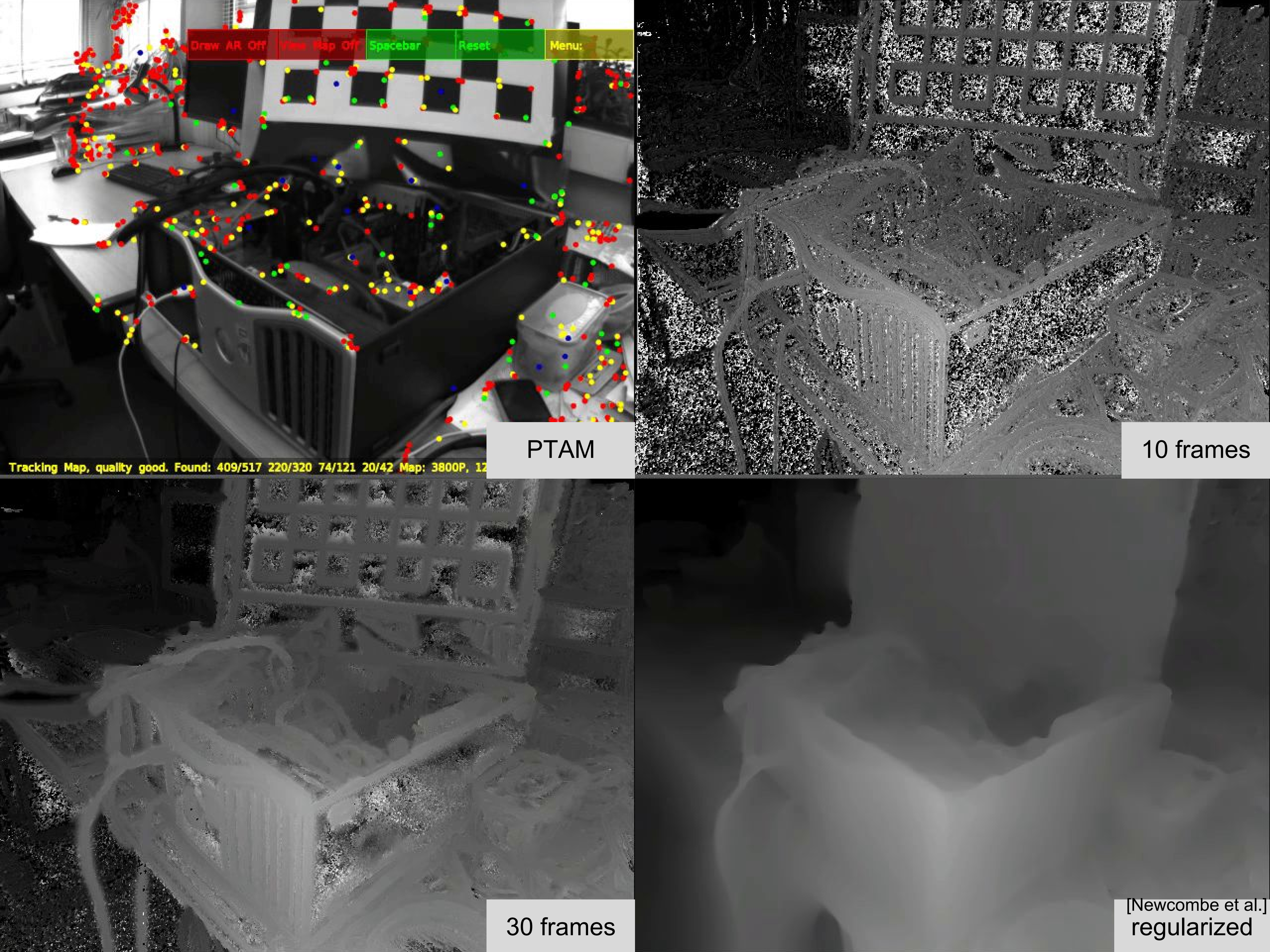
Dense Mapping: the Big Batch Optimization

Minimize the regularized energy functional:

$$\underbrace{E_{\boldsymbol{\xi}}}_{\substack{\text{Inverse depth map}\\ \text{to minimize over}}} = \int_{\Omega} \Big\{ \overbrace{\underbrace{g(\mathbf{u})}_{\text{Do not smooth edges}} \underbrace{\|\nabla \boldsymbol{\xi}(\mathbf{u})\|_{\epsilon}}_{\substack{\text{Huber norm, to make the depth}\\ \text{map smoother}}}}^{\text{Regularization term}} + \overbrace{\lambda \mathbf{C}\left(\mathbf{u}, \boldsymbol{\xi}(\mathbf{u})\right)}^{\text{“Cost volume” data term}} \Big\} \mathrm{d}\underbrace{\mathbf{u}}_{\text{Integrate over each pixel}}$$

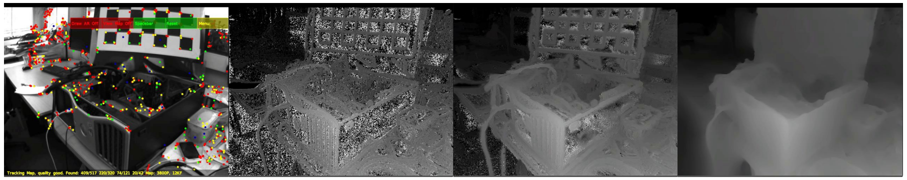
Non-convex! Approximate it:

$$\mathbf{E}_{\boldsymbol{\xi},\boldsymbol{\alpha}} = \int_{\Omega} \Big\{ g(\mathbf{u}) \|\nabla \boldsymbol{\xi}(\mathbf{u})\|_{\epsilon} + \overbrace{\frac{1}{2\theta} \left(\boldsymbol{\xi}(\mathbf{u}) - \underbrace{\boldsymbol{\alpha}}_{\text{Auxiliary variable}}(\mathbf{u})\right)^2}^{\substack{\text{Coupling term}\\ \text{Enforce } \xi = \alpha \text{ as } \theta \to 0}}$$
$$+ \lambda \mathbf{C}\left(\mathbf{u}, \boldsymbol{\alpha}(\mathbf{u})\right) \Big\} \mathrm{d}\mathbf{u}$$



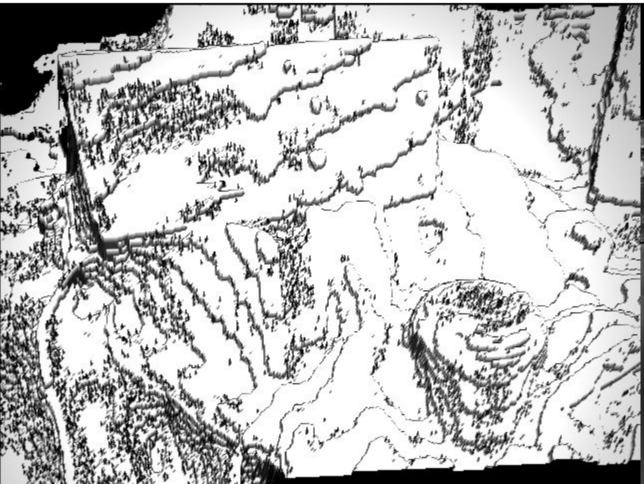
Draw AR Off
View Map Off
Spacebar
Reset
Menu:
Tracking Map, quality good. Found: 409/517 220/320 74/121 20/42 Map: 3800P, 12
PTAM
10 frames
30 frames
[Newcombe et al.]
regularized

Algorithms

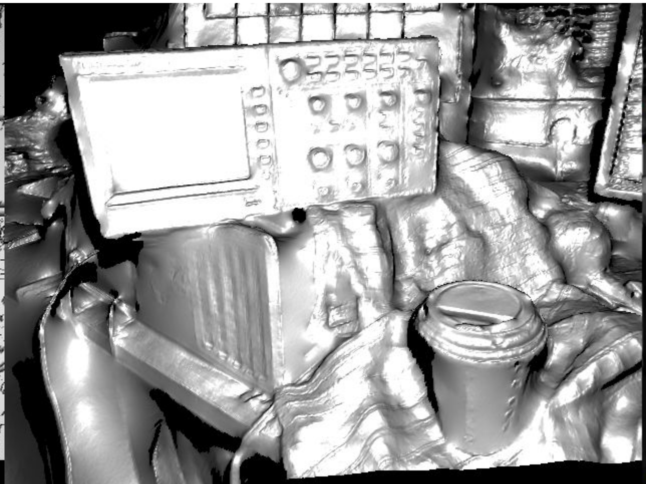


- Primal-dual method optimization
- Incremental cost volume construction - $O(1)$ for any number of frames
- Parallel per pixel optimization on GPU
- Exhaustive search over discrete inverse depth samples
  - Accelerate by showing deterministically decreasing feasible region
- Subpixel refinement

Without subpixel refinement

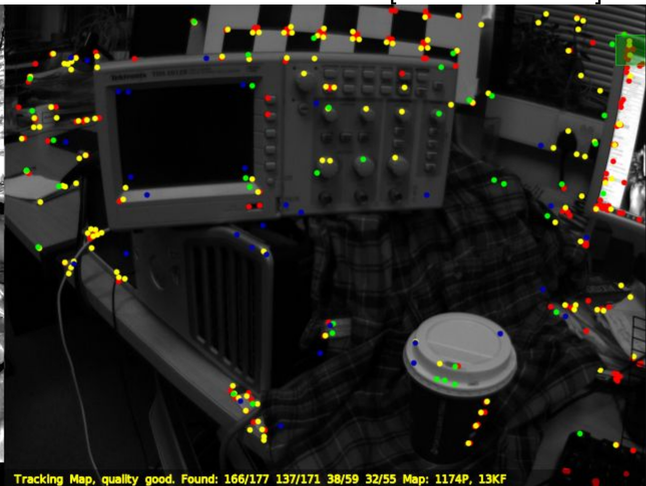


With



PTAM

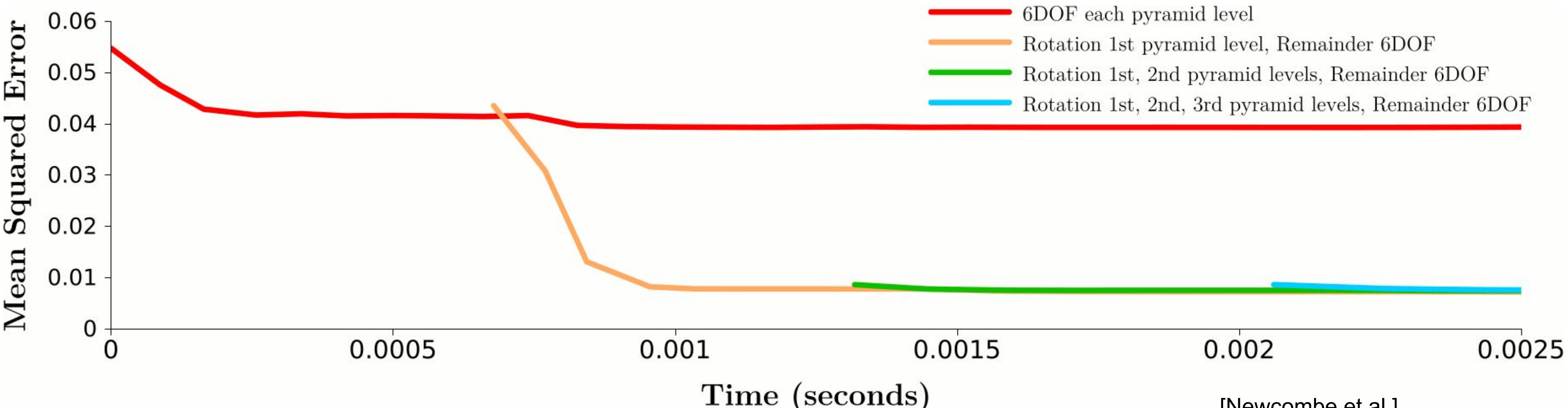
[Newcombe et al.]



Dense Tracking

- Dense cost function: project the map onto a virtual camera, and compute photometric error between synthetic images and real images
- Must initialize within convex basin of true solution
- Coarse-to-fine iterative Lucas-Kanade, two stages:
  a. Constrained inter-frame rotation estimation
  b. 6DOF full pose refinement against the map

Estimating rotation first can help to avoid local minima.



[Newcombe et al.]

Results

640×480 30Hz calibrated RGB camera, Nvidia GTX 480 (1345GFlops), i7 quad-core CPU

[Newcombe et al.]

Caveats

- No lighting changes
- No moving objects
- Requires a lot of views to be accurate
- Can’t self-bootstrap - Initializing with feature method until a keyframe is built

Textureless regions do not perform well. Regularization removes details.

Can we do something else?

[Newcombe et al.]

Semi-dense Methods

Daniel Cremers's group at TUM:

Jakob Engel

Daniel Cremers

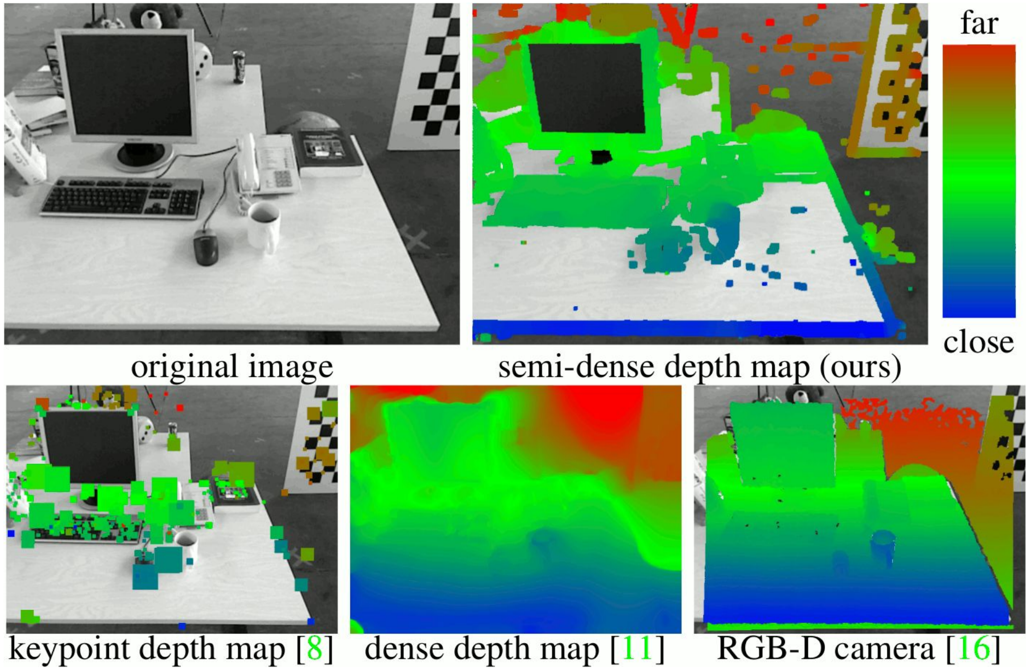
- 2013, Semi-dense monocular VO
  J. Engel, J. Sturm, D. Cremers. Semi-Dense Visual Odometry for a Monocular Camera. ICCV 2013.
- 2014, LSD-SLAM
  J. Engel, T. Schöps, D. Cremers. LSD-SLAM: Large-Scale Direct Monocular SLAM. ECCV 2014.
- 2015, Stereo LSD-SLAM
  J. Engel, J. Stueckler, D. Cremers. Large-Scale Direct SLAM with Stereo Cameras. IROS 2015.
- 2015, Omnidirectional LSD-SLAM
  D. Caruso, J. Engel, D. Cremers. Large-Scale Direct SLAM for Omnidirectional Cameras. IROS 2015.
- 2016, Semi-dense visual-inertial odometry
  V. Usenko, J. Engel, J. Stueckler, D. Cremers. Direct Visual-Inertial Odometry with Stereo Cameras. ICRA 2016.

Semi-dense VO

J. Engel, J. Sturm, D. Cremers. Semi-Dense Visual Odometry for a Monocular Camera. ICCV 2013.

- Do not track “low gradient” pixels (the semi- part)
- Probabilistic depth map representation (not in DTAM)
- Dense tracking

→ Real-time on CPU!



[Engel et al.]

Semi-dense Depth Estimation

- Estimate a depth map for the *current* image
  - DTAM: Estimate the depth map for the previous keyframe
- Propagate and refine the depth map from frame to frame (filtering like)
  - DTAM: (Incremental) batch optimization over several frames
- One depth hypothesis (Gaussian) per pixel in the current image

Stereo-based algorithm:

1. Use uncertainty criteria to select “good” pixels
2. Select adaptively a reference frame for each pixel
3. Do disparity search on the epipolar line

Error Modeling: Geometric

L: Epipolar line segment, derived from estimated motion

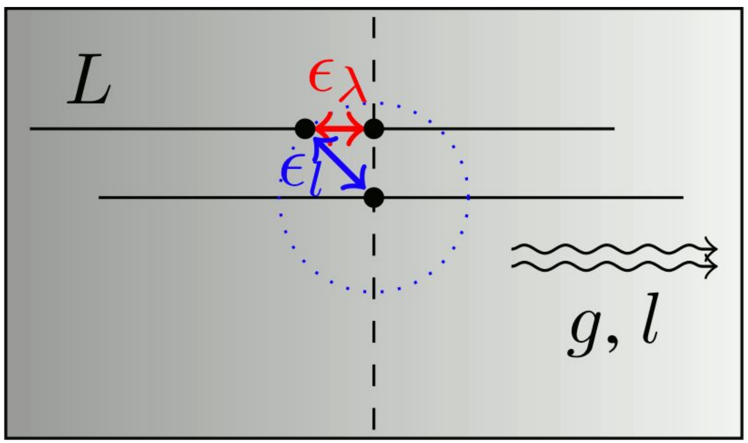
ε_l: Position error of L caused by errors in motion estimation and calibration (isotropic Gaussian, translation-only)

ε_λ: Error in estimated disparity

Isolines: lines of pixels with equal intensity

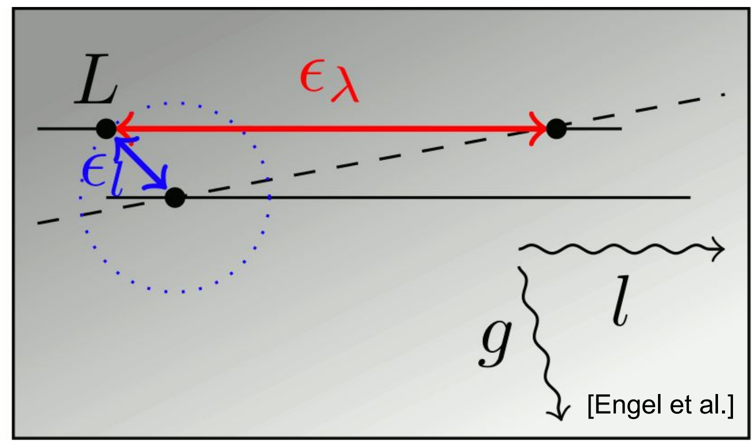
g: Direction of the gradient

l: Direction of the epipolar line



Epipolar line parallel to gradient: good
Unique match on the epipolar line

Small ε_λ



Epipolar line perpendicular to gradient: bad
All same pixels on the epipolar line

Large ε_λ

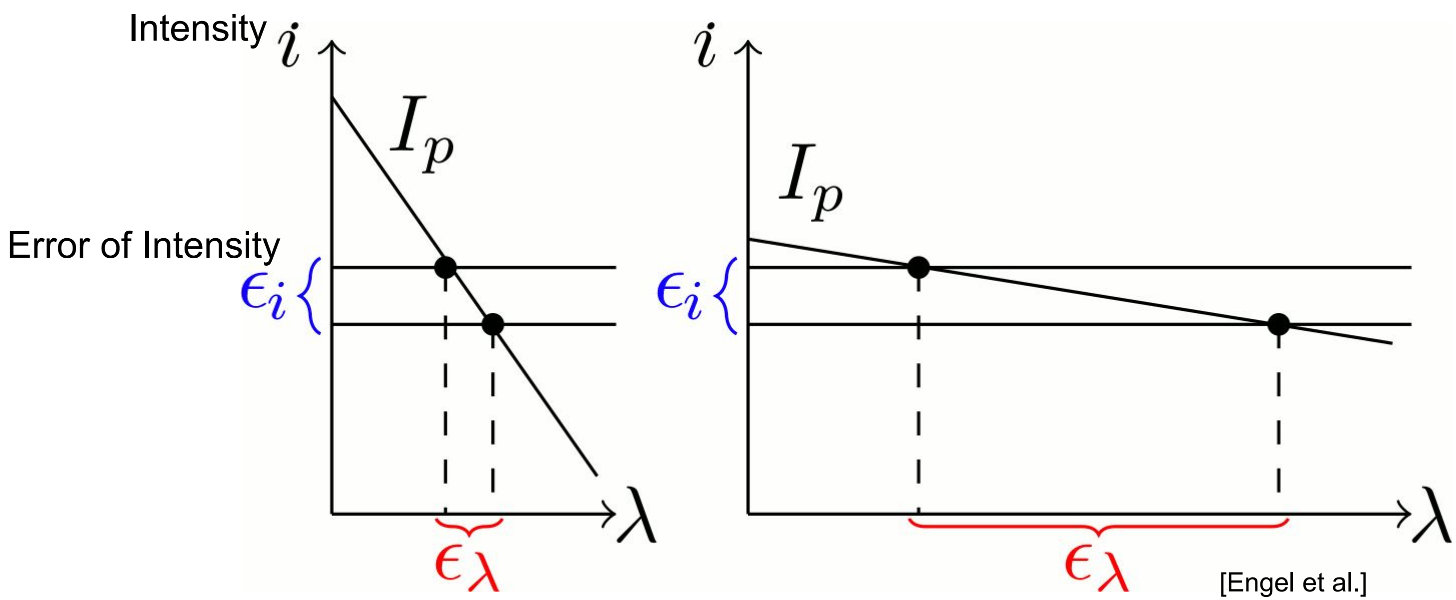
$$\sigma^2_{\lambda(\xi,\pi)} = \frac{\sigma^2_l}{\langle g, l\rangle^2}$$

Error Modeling: Photometric

ε_i: Error in image intensity

ε_λ: Error in estimated disparity

I_p: Image intensity along the epipolar line



[Engel et al.]

Large gradient: small ε_λ

Small gradient: large ε_λ

$$\sigma^2_{\lambda(I)} = \frac{2\sigma_i^2}{g_p^2}$$

Error Modeling: “pixel to inverse depth conversion”

Observation variance of the inverse depth $\leftarrow$ $$\sigma^2_{d,\text{obs}} = \alpha^2 \Big(\underbrace{\sigma^2_{\lambda(\xi,\pi)}}_{\text{Geometric}} + \underbrace{\sigma^2_{\lambda(I)}}_{\text{Photometric}} \Big)$$

$$\alpha := \frac{\delta_d}{\delta_\lambda}$$

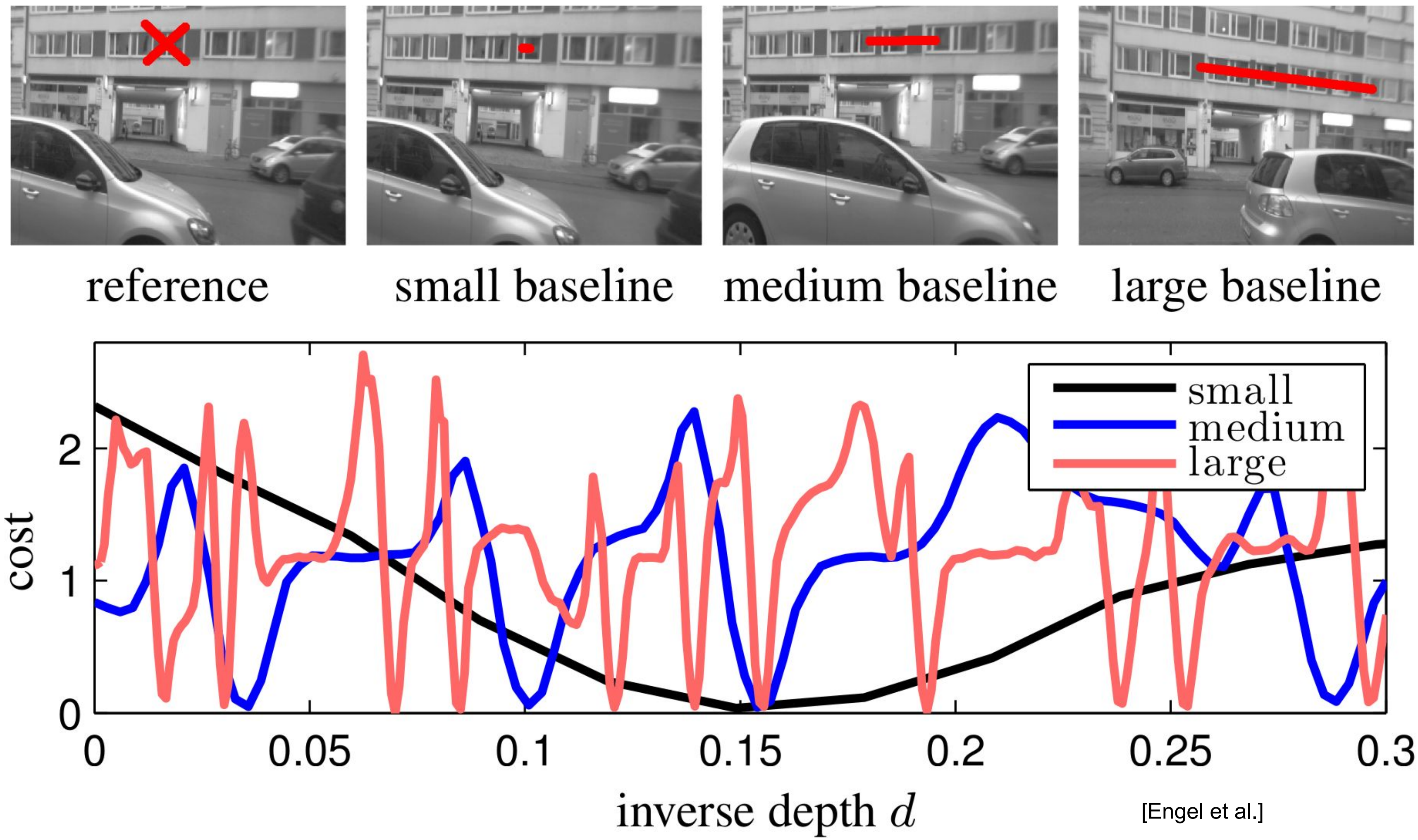
$\delta_d \rightarrow$ Searched inverse depth range

$\delta_\lambda \rightarrow$ Searched epipolar line length

Depth estimation step 1: 3 pixel selection criteria

- Low geometric disparity error
  → The epipolar line being parallel to the image gradient
- Low photometric disparity error
  → High gradient along the epipolar line
- Pixel to inverse depth ratio α

Trade-off Between Multiple Baselines



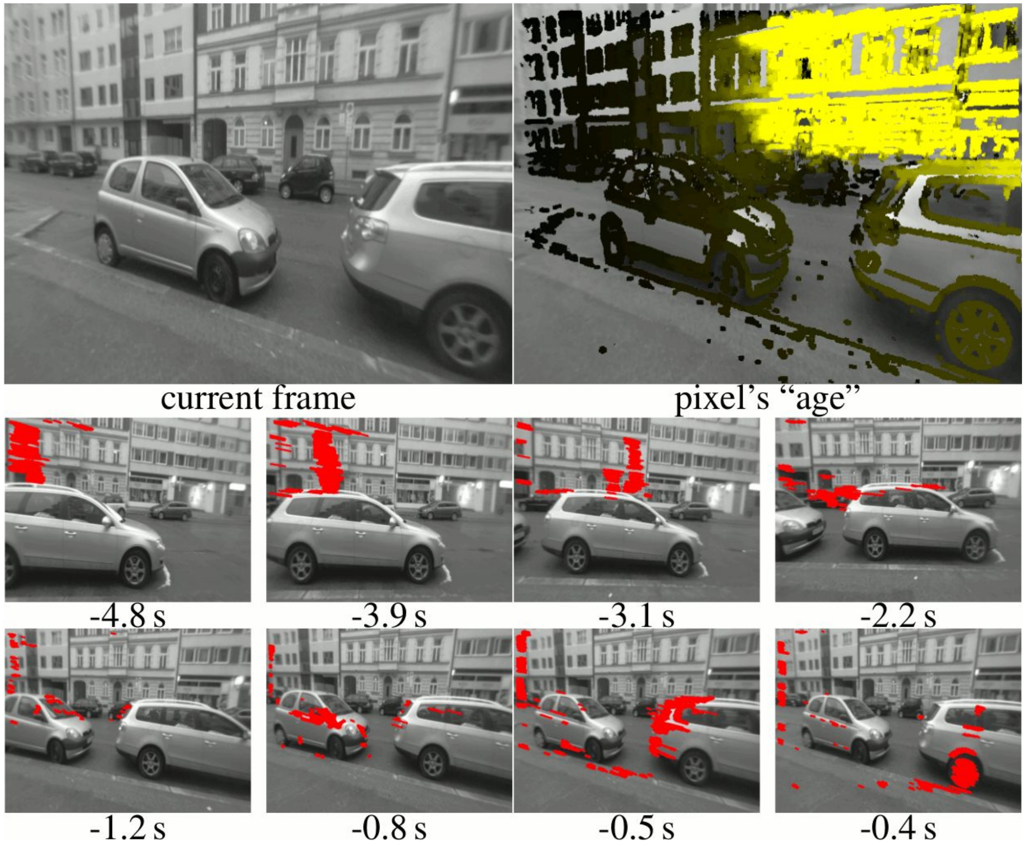
[Engel et al.]

Depth Estimation Step 2: Adaptive Baseline

For each pixel:

1. Select the oldest frame
2. Do the disparity search
3. If the search fails:

    Increase the pixel age



[Engel et al.]

Depth Estimation Step 3: Stereo Matching

- Exhaustive search along the epipolar line
- Sub-pixel refinement
- Limit the search range using the uncertainty estimates

    Otherwise search the full range

Probabilistic Depth Map Filtering

- Update step ("Kalman filter") of a depth estimate:
  - Given the prior and current observation of the depth distribution
  - Produce a posterior distribution (all Gaussians)
- Predict step:
  - Given the motion estimate of a new frame
  - Project the posterior onto the new frame as its prior
- Pixel contention: two depth hypotheses may be projected onto one pixel
  - If similar, treat independently
  - Otherwise, discard the farther one
- Regularization: edge-preserving smoothing using the uncertainty estimates, outlier removal

(Semi-)Dense Tracking

Robust weighting

$$E(\xi) := \sum_i \frac{\alpha(r_i(\xi))}{\sigma^2_{d_i}} r_i(\xi)$$

Pose to optimize

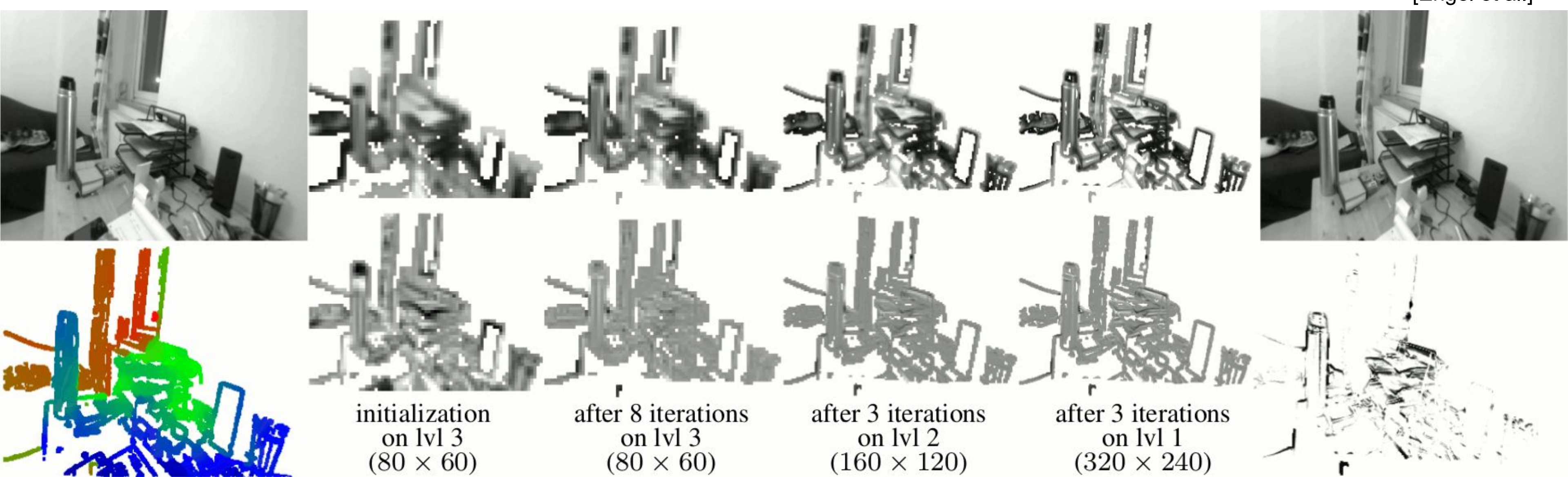
For all good pixels

Estimated variance of the inverse depth

SSD photometric error

Solve with a coarse-to-fine iterative reweighted Gauss-Newton algorithm.

[Engel et al.]



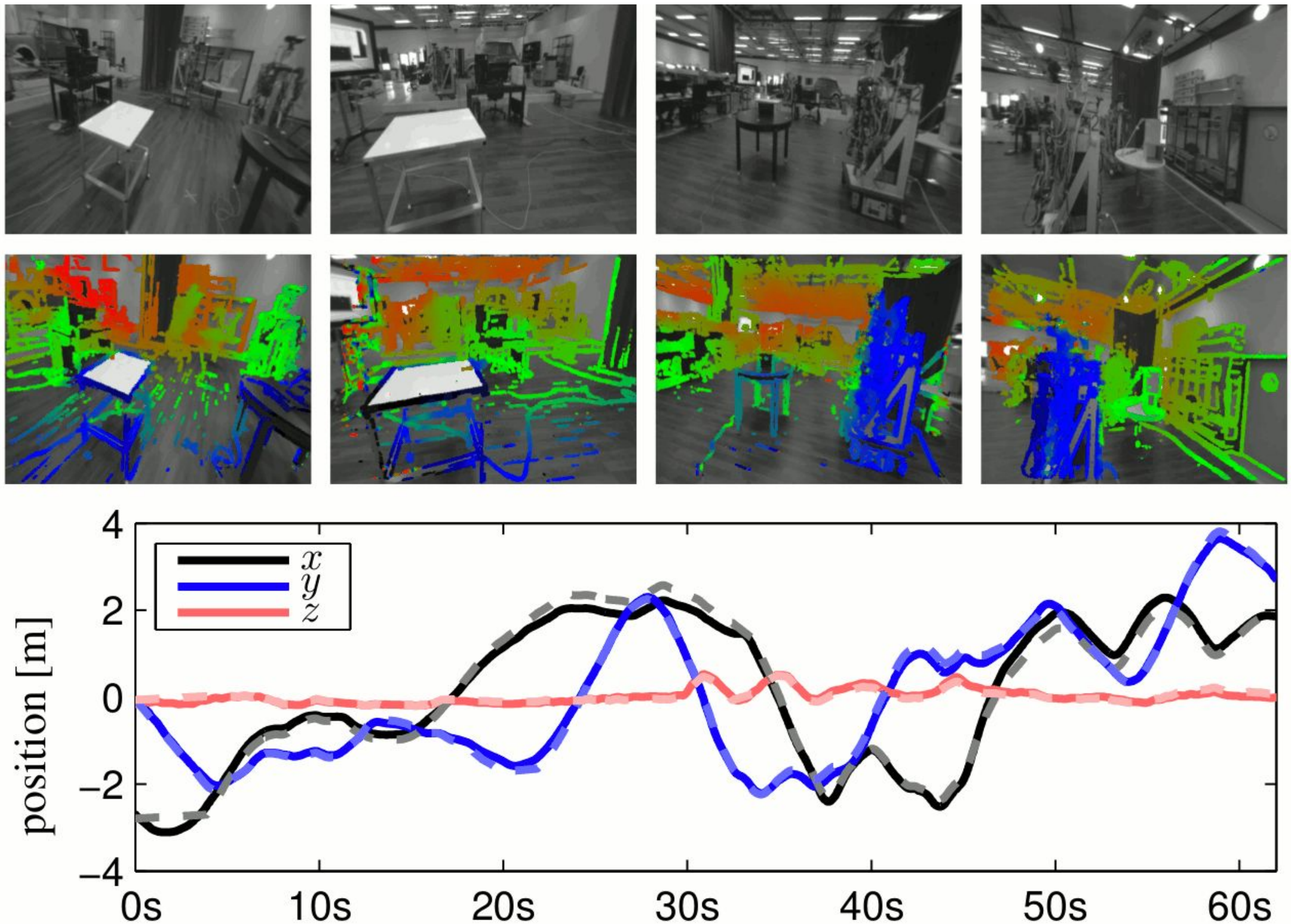
Recap of the Pipeline

1. Get a new frame
2. Estimate motion with coarse-to-fine iterative optimization against the map
3. Predict the next depth estimate with the motion estimate
4. Select “high gradient” good pixels
5. Do disparity search with the largest baseline and within the prior
6. Sub-pixel refinement to produce depth estimate
7. Update depth estimate posterior
8. Go to 1

Implementation

- “Parallel tracking and mapping”: tracking @ 30Hz, mapping @ 15Hz
- i7 quad-core CPU, a calibrated camera
- Adaptive baseline buffer: 100 frames
- Use a feature-based method to obtain initial motion, then self-sufficient
    - Still work without the initialization (?!)
    - DTAM: feature-based stereo until the depth map is built

Results

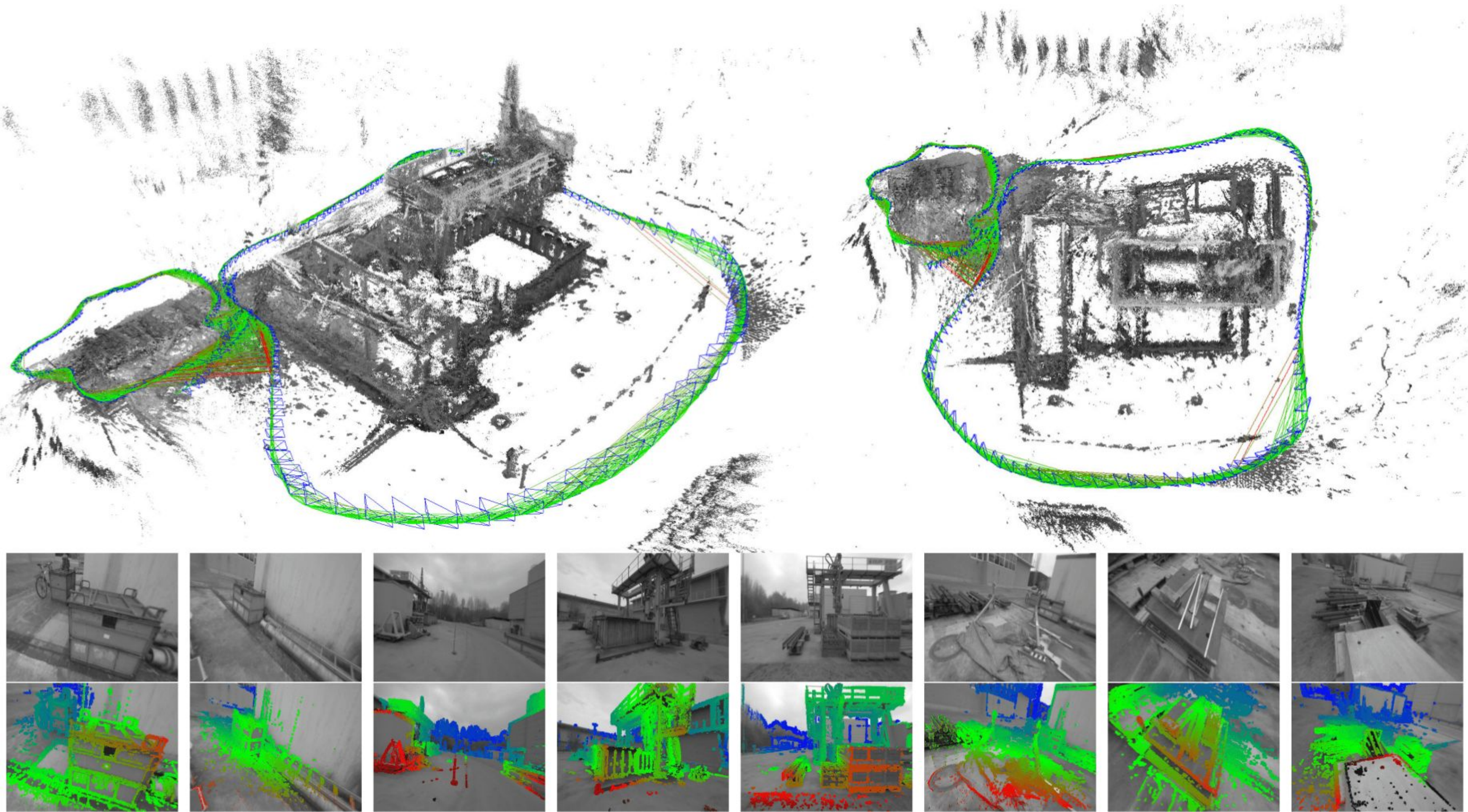


[Engel et al.]

Figure 10. Additional Sequence: Estimated camera trajectory and ground truth (dashed) for a long and challenging sequence.

LSD-SLAM: Large-Scale Direct Monocular SLAM

Jakob Engel, Thomas Schöps, Daniel Cremers. ECCV 2014.



What’s New?

New from semi-dense VO:

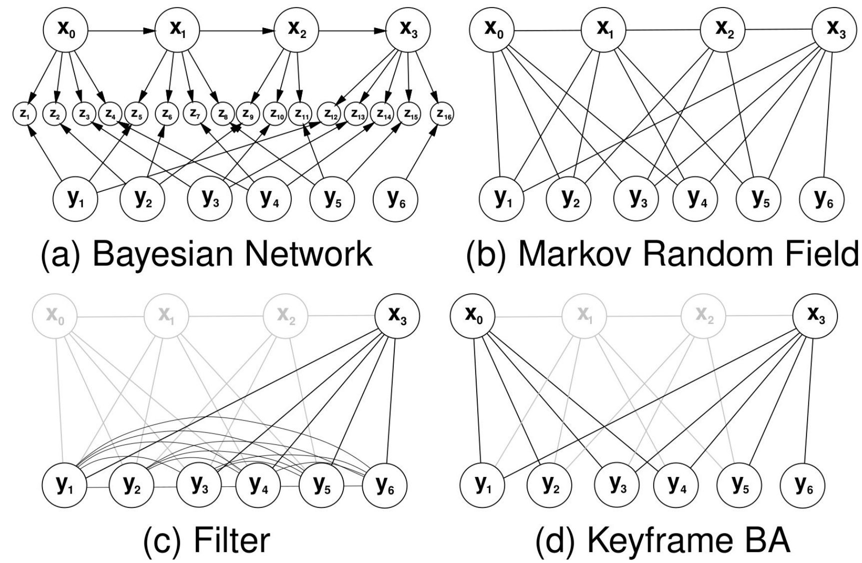
- Keyframe based pose graph map
- Scale-aware image alignment
- No initialization required

Why Filter?

“[I]n most modern applications keyframe optimisation gives the most accuracy per unit of computing time.

[I]n order to increase the accuracy of monocular SLAM it is more profitable to increase the number of features than the number of frames.

[F]ilter-based SLAM frameworks might be beneficial if a small processing budget is available, but that BA optimisation is superior elsewhere.”



Real-time Monocular SLAM: Why Filter? Hauke Strasdat, J. M. M. Montiel, Andrew J. Davison. ICRA 2010.

LSD-SLAM: Keyframe Depth Estimation

- Keyframe selection
  If too far away from the map (scale relative), create a new keyframe
- Depth creation of the keyframe
  Project previous keyframes onto the new keyframe
  Scale the depth map to have a mean of one
- Depth refinement of the keyframe

→ Optimize the keyframe depth map, not the new frame
→ The same dense tracking for the new frame, though

Pose Graph Construction

Direct tracking on *sim*(3)

3D Similarity Transformations. A 3D similarity transform $\mathbf{S} \in \mathrm{Sim}(3)$ denotes rotation, scaling and translation, i.e. is defined by

$$\mathbf{S} = \begin{pmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \quad \text{with} \quad \mathbf{R} \in \mathrm{SO}(3),\ \mathbf{t} \in \mathbb{R}^3 \ \text{ and } \ s \in \mathbb{R}^+. \tag{4}$$

$$E(\boldsymbol{\xi}_{ji}) := \sum_{\mathbf{p} \in \Omega_{D_i}} \left\| \underbrace{\frac{r_p^2(\mathbf{p}, \boldsymbol{\xi}_{ji})}{\sigma^2_{r_p(\mathbf{p},\boldsymbol{\xi}_{ji})}}}_{\text{Variance-normalized photometric error}} + \underbrace{\frac{r_d^2(\mathbf{p}, \boldsymbol{\xi}_{ji})}{\sigma^2_{r_d(\mathbf{p},\boldsymbol{\xi}_{ji})}}}_{\text{Variance-normalized photometric error of the inverse depth map}} \right\|_{\delta}$$

Variance-normalized photometric error of the inverse depth map → Required to constrain the scale

δ → Huber norm

Loop closure detection:

- Search within 10 closest frames and appearance-based candidates
- Reciprocal tracking check

Improve convergence radius for large loop closures:

- Initial guess with keypoints
- Efficient Second Order Minimization
- Coarse-to-fine

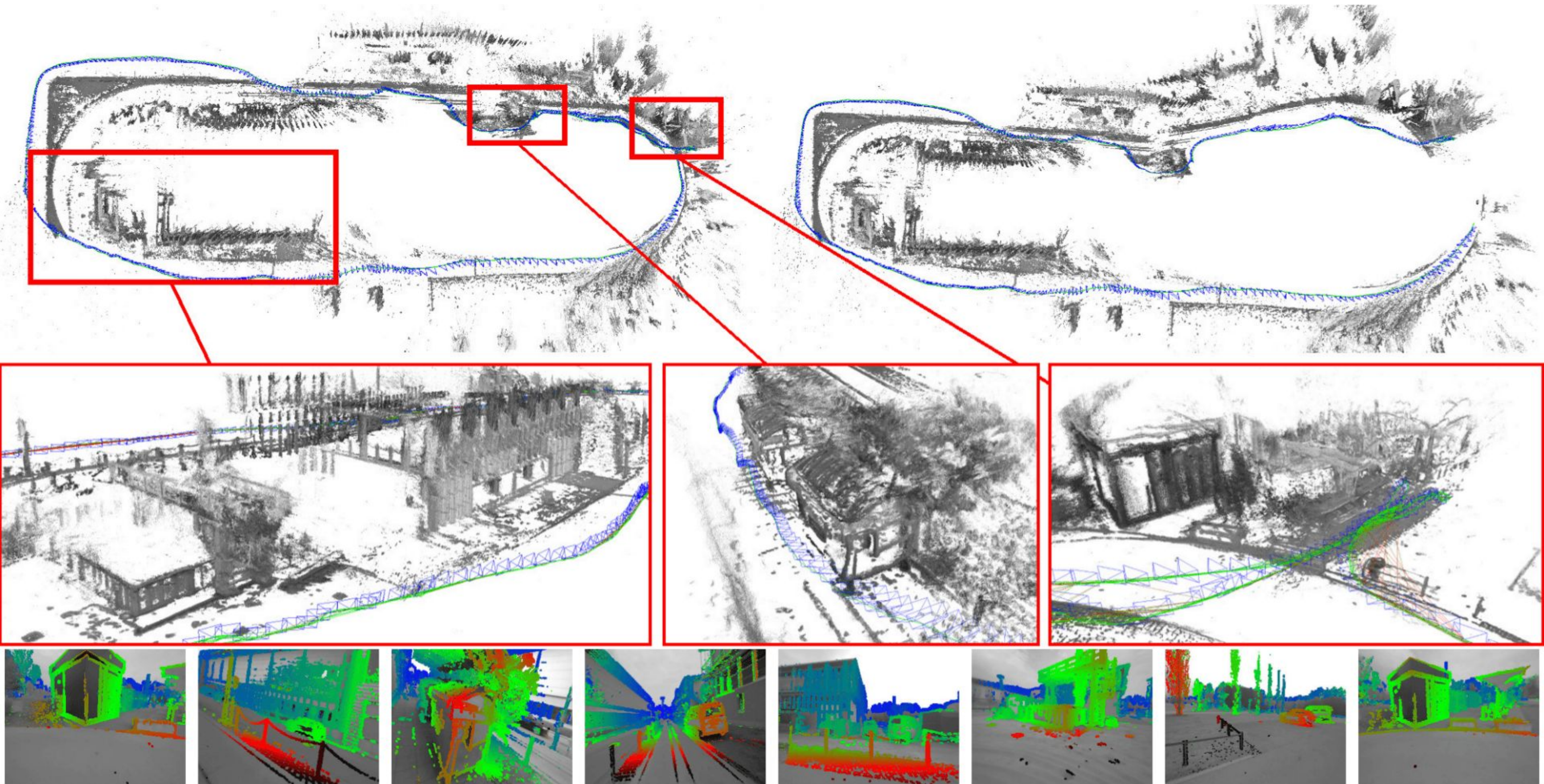
Then Optimize the Pose Graph

With g2o.

[Engel et al.]

Loop closure found

Before loop closure



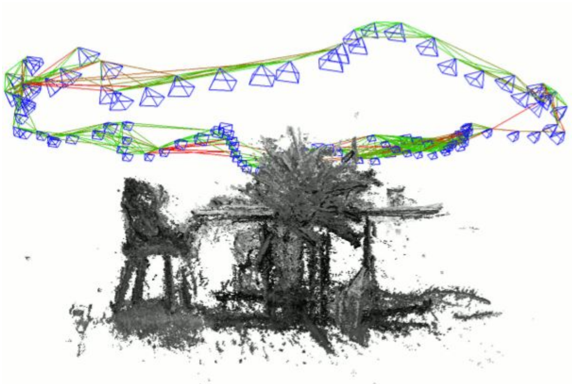
Evaluation

RGBDSLAM (ICRA'12)

Dense RGBD VSLAM (IROS'13)

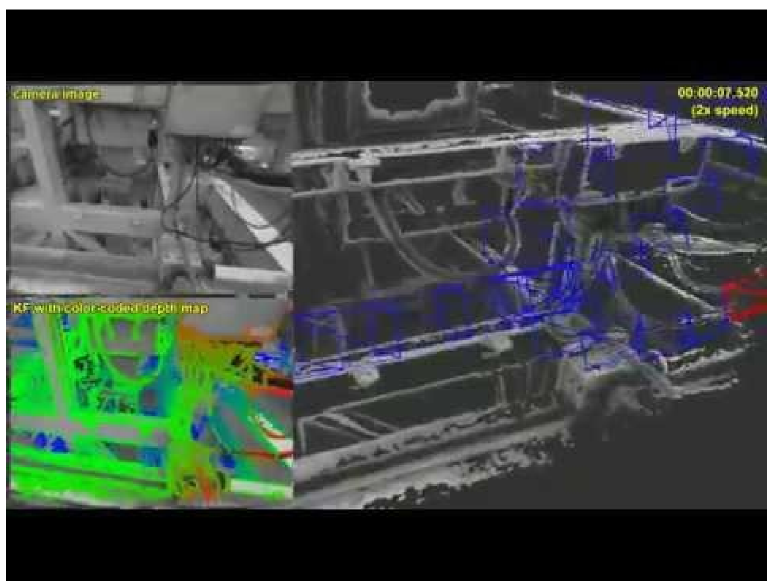
PTAM (ISMAR'07)

Semi-dense VO (ICCV'13)



	LSD-SLAM (#KF)	[9]	[15]	[14]	[7]
fr2/desk	4.52 (116)	13.50	x	1.77	9.5
fr2/xyz	1.47 (38)	3.79	24.28	1.18	2.6
sim/desk	0.04 (39)	1.53	-	0.27	-
sim/slowmo	0.35 (12)	2.21	-	0.13	-

Absolute trajectory RMSE (cm)



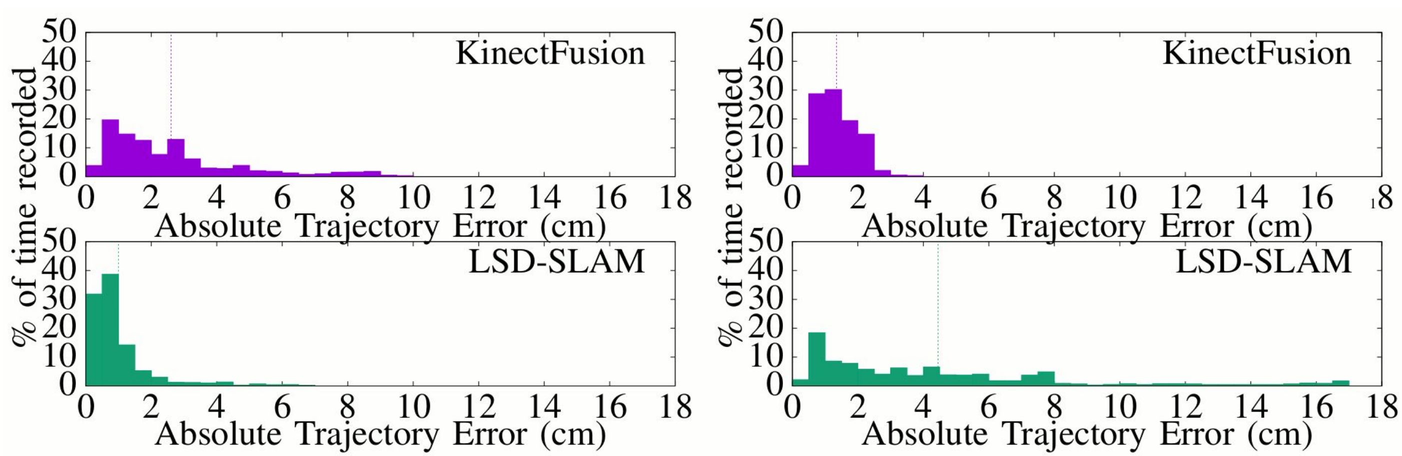
[Engel et al.]

LSD-SLAM vs KinectFusion

Comparative Design Space Exploration of Dense and Semi-Dense SLAM.
M. Zeeshan Zia, Luigi Nardi, Andrew Jack, Emanuele Vespa, Bruno Bodin, Paul H.J. Kelly, Andrew J. Davison. ICRA 2016.

Platform	Seq.	Time/frame (s)		ATE (cm)		Energy/frame (J)	
		KF	LSD	KF	LSD	KF	LSD
Desktop	Syn.	0.18	0.03	1.36	4.44	12.51	0.80
Desktop	Real	0.15	0.03	2.62	0.99	10.62	1.21
ODROID	Syn.	0.89	0.20	1.35	4.37	4.90	0.38
ODROID	Real	0.93	0.20	2.62	1.14	4.99	0.50

TABLE I: Holistic comparison table.



(a) Real Scene (b) Synthetic Scene

Runtime Profiling of LSD-SLAM

Zeeshan Zia et al.

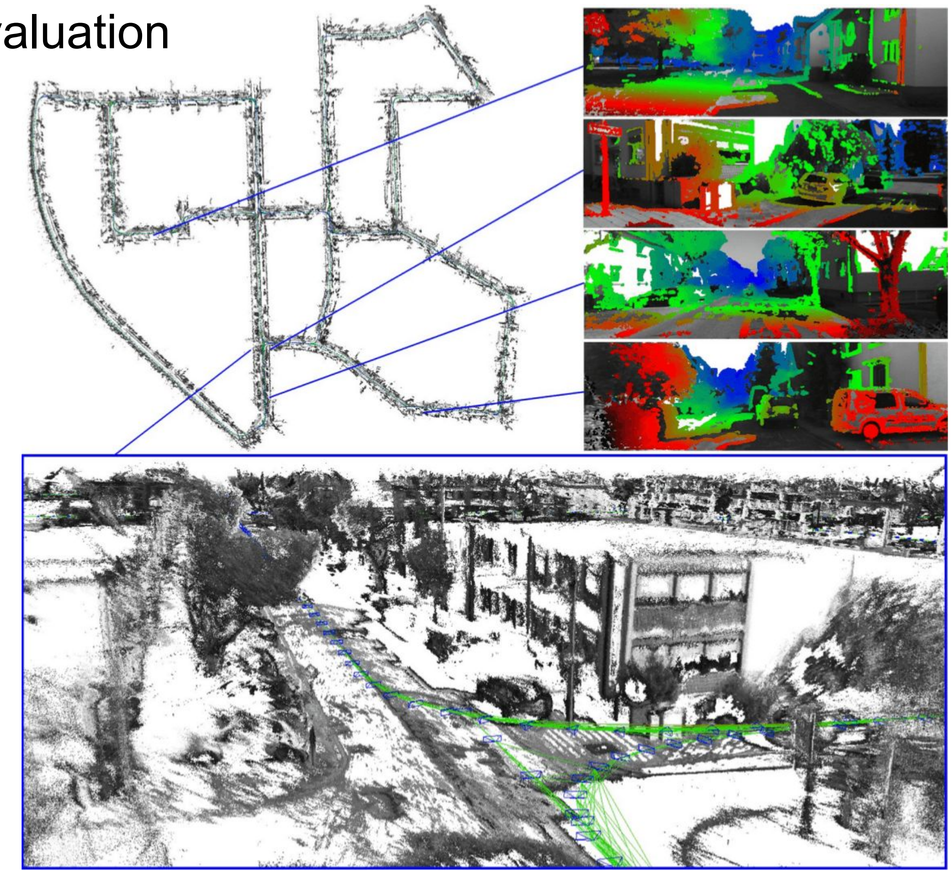
A lot of slow image processing - room for acceleration

Thread name	**Major kernels**	**Description**	**Percent**
Tracking (vectorized)	*Calc. Residuals*		72%
	Calc. Weights and Residuals	Calculate components of the Levenberg-	4%
	Calc. Jacobian Matrix		9%
	Solve	Evaluate the LM algorithm given the abov	0%
Total			**34 s**
Depth	*Stereo Line Search*	Epipolar line search	43%
	Fill Holes	Increase density of depth map	20%
	Regularize Depth Map	Denoise the depth map	28%
	Copy Depth Map to Frame	Implementation specific overhead	6%
Total			**48 s**
Constraint Search	*Find Euclidean Overlaps*	Get neighbour frames from graph, to inser	6%
	Filter and Sorting	Remove less optimal frames from results	4%
	Calc. Residuals		71%
	Calc. Weights and Residuals	Calculate components of the Levenberg	7%
	Calc. Jacobian Matrix	between keyframe and neighbour frame	12%
Total			**19 s**
Optimization	*g2o Call*	Run iterations of global optimization	99%
	Update Graph	Incorporate improvements from g2o into g	1%
Total			**3 s**

Large-Scale Direct SLAM with Stereo Cameras

Jakob Engel, Jorg Stueckler, Daniel Cremers. IROS 2015.

- Couple *temporal stereo* from monocular with *static stereo*
- Get depth from static stereo, recover scale
- Model illumination changes during direct image alignment
- Systematic evaluation



[Engel et al.]

Depth Estimation

- Use static stereo for keyframe depth estimation
- Use temporal stereo to refine keyframe depth
  Larger baseline than static stereo
- Modify the photometric error to include affine lighting correction

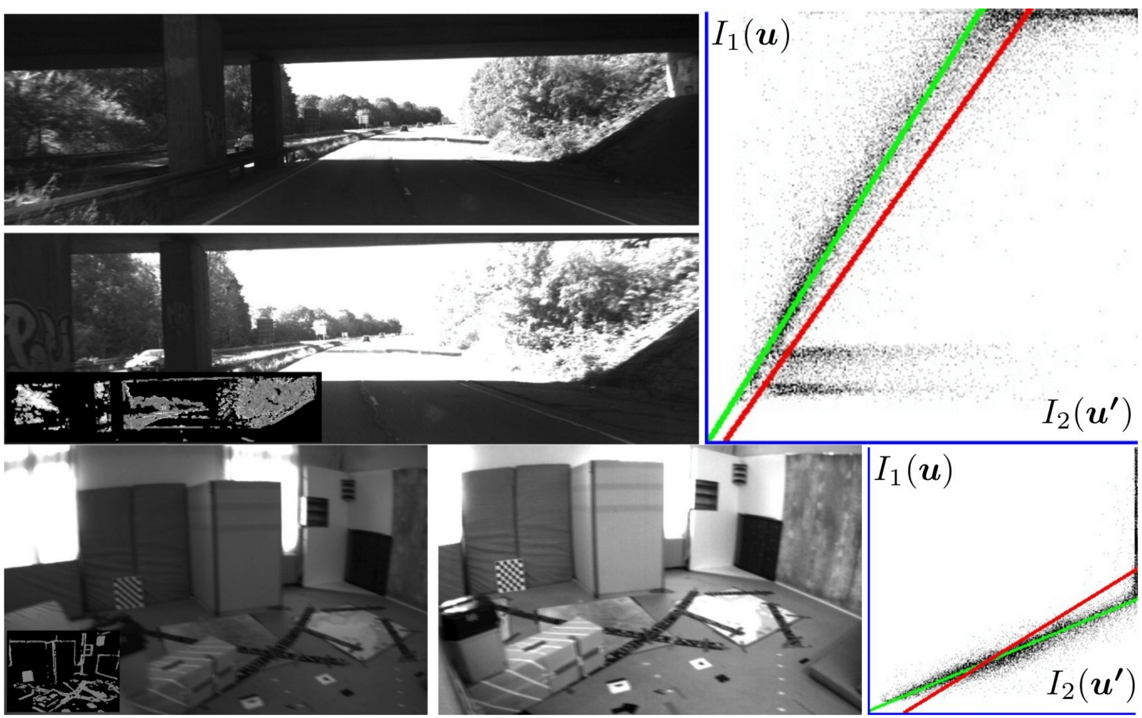
Affine Lighting Correction

Model the photometric error with additional affine parameters a, b

$$r_{\boldsymbol{u}}^{I}(\boldsymbol{\xi}) := aI_1^l(\boldsymbol{u}) + b - I_2^l(\boldsymbol{p}')$$

Iteratively optimize over all of $\boldsymbol{\xi}$, a, b.

a and b can be estimated by robust linear least-squares.



Scatter plot of old vs new image intensity on matched pixels

Robust fit

Total fit

[Engel et al.]

Evaluation

KITTI Visual Odometry / SLAM Evaluation 2012 (only showing stereo)

Submitted after S-LSD-SLAM

	Method	Setting	Code	Translation	Rotation	Runtime
3	SOFT			0.88 %	0.0022 [deg/m]	0.1 s
I. Cvišić and I. Petrović: Stereo odometry based on careful feature selection and tracking. European Conference on Mobile F						
4	RoCROCC			0.98 %	0.0028 [deg/m]	0.3 s
5	ROCC			0.98 %	0.0028 [deg/m]	0.3 s
6	cv4xv1-sc			1.09 %	0.0029 [deg/m]	0.145 s
M. Persson, T. Piccini, R. Mester and M. Felsberg: Robust Stereo Visual Odometry from Monocular Techniques. IEEE Intell						
8	ORB-SLAM2		code	1.15 %	0.0027 [deg/m]	0.06 s
9	NOTF			1.17 %	0.0035 [deg/m]	0.45 s
Anonymous submission						
10	S-LSD-SLAM		code	1.20 %	0.0033 [deg/m]	0.07 s
J. Engel, J. St\"uckler and D. Cremers: Large-Scale Direct SLAM with Stereo Cameras. Int.~Conf.~on Intelligent Robot Syst						

Submitted on Sep 26, 2015

Problems:

Test sequences 00-10 have moving objects, can't handle them

Dataset framerate is too low (10Hz at 80km/h) for direct methods

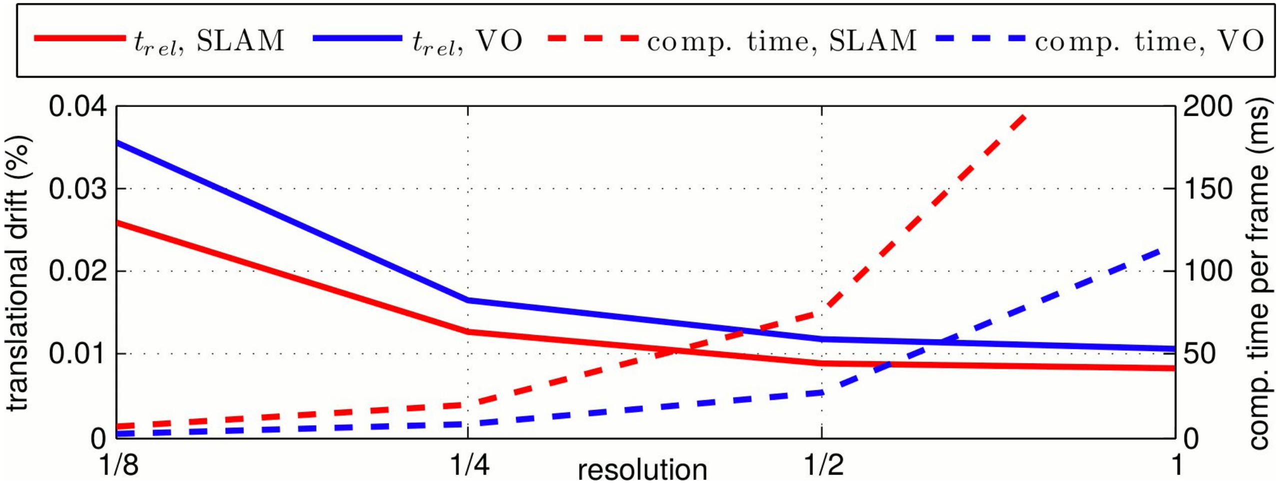
Performance Analysis

154x46 resolution:

Error 2.5% (SLAM) 3.5% (VO)

Runtime 15x (SLAM) 40x (VO) real-time

Feature-based methods will not work under such low resolution.



[Engel et al.]

Limitations

- No moving objects (yet) → rigid body motion segmentation
- No reflection
- No models for surfaces (can't use for collision avoidance)