# Direct Methods in Visual Odometry

July 24, 2017

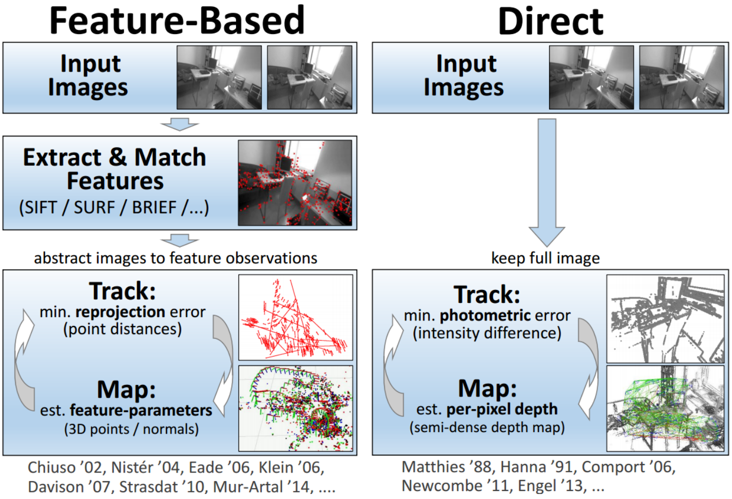
## Motivation for using Visual Odometry

- Wheel odometry is affected by wheel slip
- More accurate compared to wheel odometry
- Can be used to complement GPS, IMUs, Lidar
- Particularly useful in GPS-denied environments

## Visual Odometry Assumptions

- Sufficient Illumination in the environment
- Dominance of static scene over moving objects
- Enough texture to allow apparent motion to be extracted
- Sufficient scene overlap between consective frames

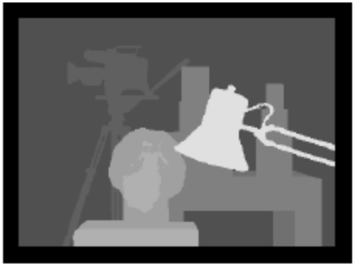
# Feature Based vs Direct


Feature-Based
Direct
Input Images
Input Images
Extract & Match Features
(SIFT / SURF / BRIEF /...)
abstract images to feature observations
keep full image
Track:
min. **reprojection** error
(point distances)
Map:
est. **feature-parameters**
(3D points / normals)
Track:
min. **photometric** error
(intensity difference)
Map:
est. **per-pixel depth**
(semi-dense depth map)
Chiuso '02, Nistér '04, Eade '06, Klein '06, Davison '07, Strasdat '10, Mur-Artal '14, ....
Matthies '88, Hanna '91, Comport '06, Newcombe '11, Engel '13, ...

## Feature Based vs Direct

| Feature-Based | Direct |
|---|---|
| can only use & reconstruct corners | can use & reconstruct whole image |
| faster | slower (but good for parallelism) |
| flexible: outliers can be removed retroactively. | inflexible: difficult to remove outliers retroactively. |
| robust to inconsistencies in the model/system (rolling shutter). | not robust to inconsistencies in the model/system (rolling shutter). |
| decisions (KP detection) based on less complete information. | decision (linearization point) based on more complete information. |
| no need for good initialization. | needs good initialization. |

## Stereo Matching



## Stereo Matching - Matching Cost

A **Matching Cost** measures the **similarity** of pixels, examples:

- Absolute Intensitiy Difference (AD):

$$|I_L(x, y) - I_R(x, y)| \tag{1}$$

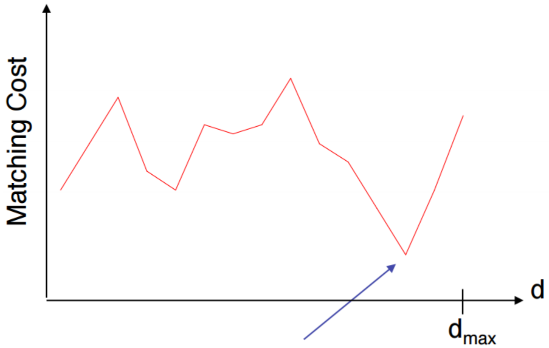
- Squared Intensitiy Difference (SD):

$$(I_L(x, y) - I_R(x, y))^2 \tag{2}$$

## Stereo Matching - Disparity Computation

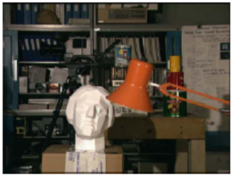
The corresponding pixel is chosen in a way that the similarity between the pixels is high ("dissimilarity" = cost). For example the "Winner Takes All" algorithm, where for every pixel select the disparity with the lowest cost.

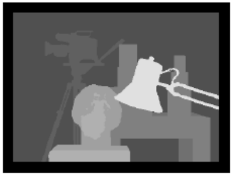
$$|I_L(x, y) - I_R(x + d, y)| \tag{3}$$

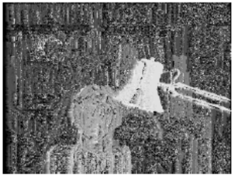


## Stereo Matching - Example Algorithm

Using the "Winner Takes All" algorithm the disparity map looks like this:

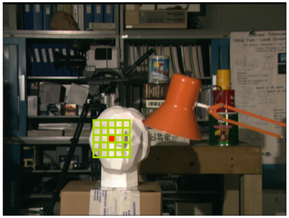

Left Camera Image

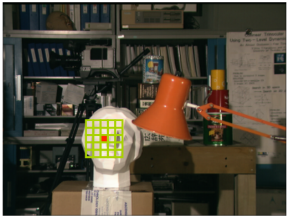

Optimal Result


Actual Result (Bad!)

The disparity map is very noisy, due to a low signal to noise ratio (SNR). To remedy this we use **Cost Aggregation** where we do not compare single pixels but small patches.

## Stereo Matching - Cost Aggregation





Using a "matching window" around the pixel of interest, and apply the **sum of absolute intensity differences** (SAD):

$$\sum_{(x,y)\in W} |I_R(x,y) - I_L(x+d,y)| \tag{4}$$

## Stereo Matching - Cost Aggregation

Examples for such area-based matching costs:

- **Sum of absolute differences** (SAD):

$$\sum_{(x,y)\in W} |I_R(x,y) - I_L(x+d,y)| \tag{5}$$

- **Sum of square differences** (SSD):

$$\sum_{(x,y)\in W} (I_R(x,y) - I_L(x+d,y))^2 \tag{6}$$

- **Normalized Cross Correlation** (NCC):

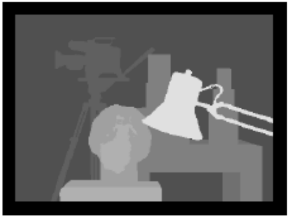
$$\frac{\sum_{(x,y)\in W}[I_R(x,y) - \bar{I}_L] \times [I_L(x+d,y) - \bar{I}_L]}{\sqrt{\sum_{(x,y)\in W}[I_R(x,y) - \bar{I}_L]^2} \times \sqrt{\sum_{(x,y)\in W}[I_L(x+d,y) - \bar{I}_L]^2}} \tag{7}$$

## Stereo Matching - Cross Correlation

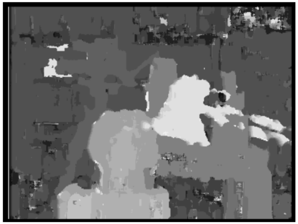
If we use both "Winner-Takes-All" algorithm and an area based matching cost (SAD) we get:

Left Camera Image

Optimal Result



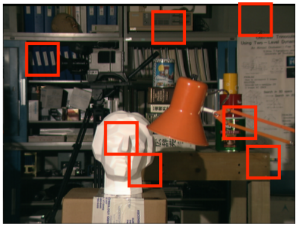
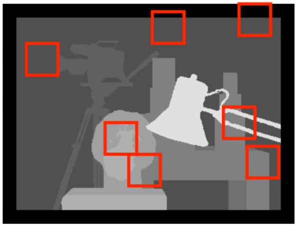
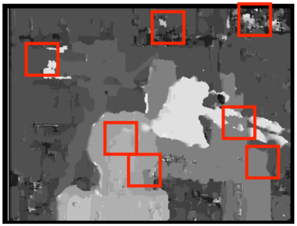
Actual Result (Still many errors)



## Stereo Matching - Problems with Fixed Windows

The area-based approach has other problems:

- Assumes constant depth with in the window
- Repetitive textures
- Uniform areas
- Thin structures

## Stereo Matching - Cross Correlation Summary

Despite drawbacks of area-based approaches, cross correlation (WTA with SAD) is often adpoted in practice. Because:

- Simple
- Fast
- Low memory requirements

Memory requirement is low, because we need no additional information except the disparity for every pixel.

## Stereo Matching - Inverse Depth Estimation

- Montiel, JM Martnez, Javier Civera, and Andrew J. Davison. "Unified inverse depth parametrization for monocular SLAM." Robotics: Science and Systems, 2006.
- Civera, Javier, Andrew J. Davison, and JM Martinez Montiel. "Inverse depth parametrization for monocular SLAM." IEEE transactions on robotics 24.5 (2008): 932-945.

## Direct Dense VO - DTAM

- 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
- Uses GPU to speed up optimization

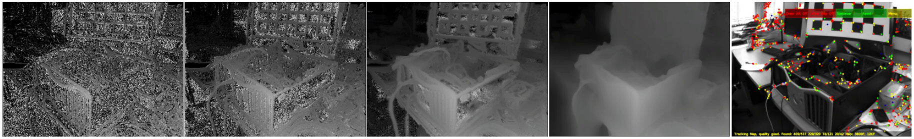
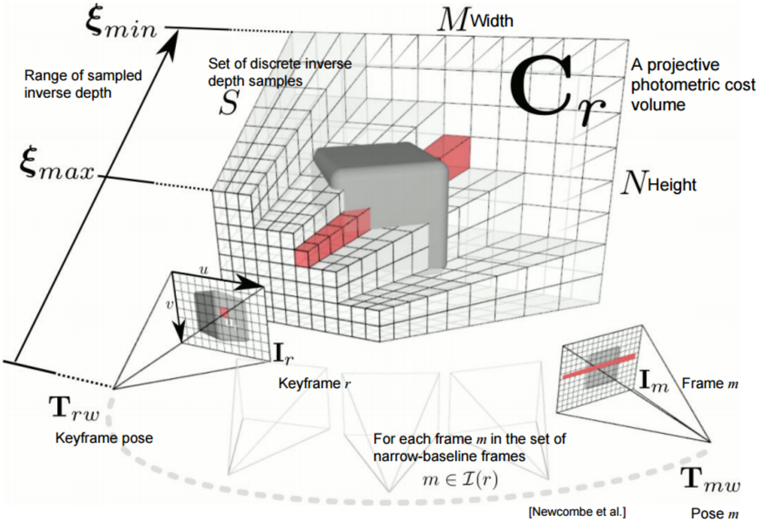


Figure 3. Incremental cost volume construction; we show the current inverse depth map extracted as the current minimum cost for each pixel row $d_{\mathbf{u}}^{min} = \arg\min_d \mathbf{C}(\mathbf{u}, d)$ as 2, 10 and 30 overlapping images are used in the data term (left). Also shown is the regularised solution that we solve to provide each keyframe inverse depth map (4th from left). In comparison to the nearly $300 \times 10^3$ points estimated in our keyframe, we show the $\approx 1000$ point features used in the same frame for localisation in PTAM ([6]). Estimating camera pose from such a fully dense model enables tracking robustness during rapid camera motion.


$\xi_{min}$
Range of sampled inverse depth
$\xi_{max}$
Set of discrete inverse depth samples $S$
$M$ Width
$\mathbf{C}_r$
A projective photometric cost volume
$N$ Height
$u$
$v$
$\mathbf{I}_r$
Keyframe $r$
$\mathbf{T}_{rw}$
Keyframe pose
$\mathbf{I}_m$
Frame $m$
For each frame $m$ in the set of narrow-baseline frames $m \in \mathcal{I}(r)$
$\mathbf{T}_{mw}$
Pose $m$
[Newcombe et al.]

$$\underbrace{\mathbf{C}_r(\mathbf{u}, d)}_{\text{Pixel coord, Inverse depth}} = \underbrace{\frac{1}{|\mathcal{I}(r)|}}_{\text{Number of frames}} \sum_{m \in \mathcal{I}(r)} \underbrace{\| \rho_r(\mathbf{I}_m, \mathbf{u}, d) \|_1}_{\text{Photometric error of frame } \mathbf{I}_m\text{, pixel } \mathbf{u}\text{, depth } d}$$

Minimize the regularized energy functional:

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

Non-convex! Approximate it:

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

$$+ \lambda \mathbf{C}\left(\mathbf{u}, \boldsymbol{\alpha}(\mathbf{u})\right) \Big\} \mathrm{d}\mathbf{u}$$

# Semi Dense Visual Odometry

- 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)
- Real time in CPU!

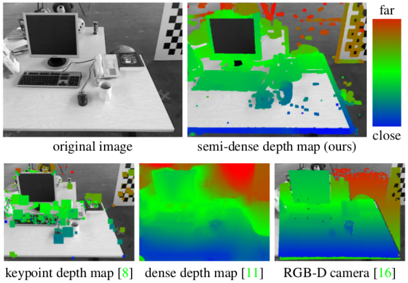


**Figure 2. Semi-Dense Approach:** Our approach reconstructs and tracks on a *semi-dense inverse depth map*, which is dense in all image regions carrying information (top-right). For comparison, the bottom row shows the respective result from a keypoint-based approach, a fully dense approach and the ground truth from an RGB-D camera.

## Semi Dense Visual Odometry - 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

## Semi Dense Visual Odometry - Geometric Disparity Error

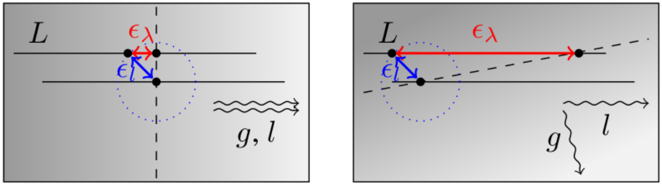


**Figure 5. Geometric Disparity Error:** Influence of a small positioning error $\epsilon_l$ of the epipolar line on the disparity error $\epsilon_\lambda$. The dashed line represents the isocurve on which the matching point has to lie. $\epsilon_\lambda$ is small if the epipolar line is parallel to the image gradient (left), and a large otherwise (right).

$$\sigma^2_{\lambda(\xi,\pi)} = \frac{\sigma^2_l}{\langle g, l\rangle^2} \tag{8}$$

## Semi Dense Visual Odometry - Photometric Disparity Error

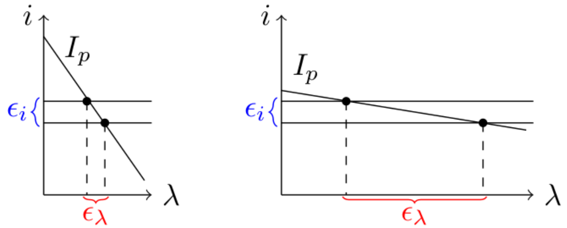


**Figure 6. Photometric Disparity Error:** Noise $\epsilon_i$ on the image intensity values causes a small disparity error $\epsilon_\lambda$ if the image gradient along the epipolar line is large (left). If the gradient is small, the disparity error is magnified (right).

$$\sigma^2_{\lambda(I)} = \frac{2\sigma_i^2}{g_p^2} \tag{9}$$

## Semi Dense Visual Odometry - Pixel to Inverse Depth Error

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

$$\alpha := \frac{\delta_d \rightarrow \text{Searched inverse depth range}}{\delta_\lambda \rightarrow \text{Searched epipolar line length}}$$

## Semi Dense Visual Odometry - 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

## Semi Dense Visual Odometry - Results

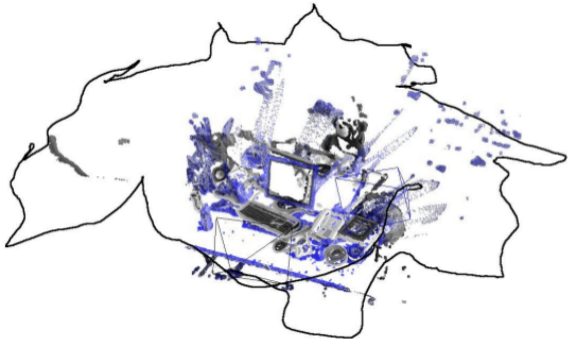
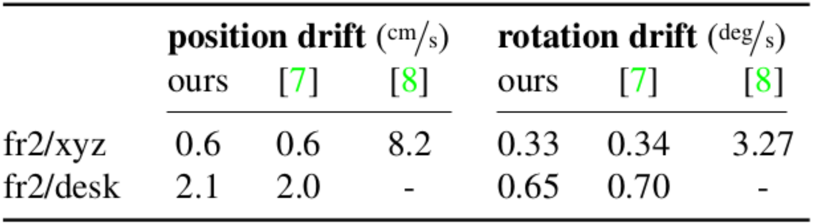


**Figure 9. RGB-D Benchmark Sequence fr2/desk:** Tracked camera trajectory (black), the depth map of the first frame (blue), and the estimated depth map (gray-scale) after a complete loop around the table. Note how well certain details such as the keyboard and the monitor align.

## Semi Dense Visual Odometry - Results

Table 1. Results on RGB-D Benchmark

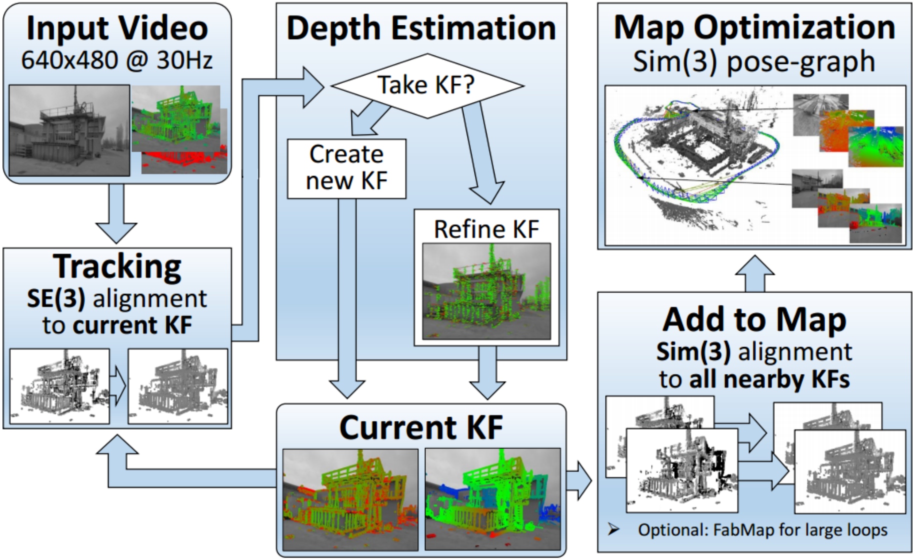


| | position drift ($^{cm}/_{s}$) | | | rotation drift ($^{deg}/_{s}$) | | |
|---|---|---|---|---|---|---|
| | ours | [7] | [8] | ours | [7] | [8] |
| fr2/xyz | 0.6 | 0.6 | 8.2 | 0.33 | 0.34 | 3.27 |
| fr2/desk | 2.1 | 2.0 | - | 0.65 | 0.70 | - |

## Large Scale Direct SLAM

- J. Engel, T. Schops, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in European Conference on Computer Vision, pp. 834849, Springer, 2014.
- Build large scale consistent maps in real time
- Novel direct tracking method that operates on sim(3), thereby explicitly detecting scale drift
- Probabilistic solution to include effect of noisy depth values into tracking

## LSD SLAM - Pipeline



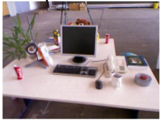
## LSD SLAM - Overview

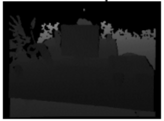
- **Tracking**: continuously tracks new camera images
- **Depth map estimation**: uses tracked frames to either refine or replace current keyframe
- **Map optimization**: once a keyframe is replaced as tracking reference (its depth map will no longer be refined further), it is incorporated into the global map by the map optimization component.

## LSD SLAM - Direct Tracking

$$E(\boldsymbol{\xi}) = \sum_{\mathbf{x}\in\Omega_{\mathrm{KF}}} \left(I_{\mathrm{KF}}(\mathbf{x}) - \underbrace{I(\omega(\mathbf{x}, D_{\mathrm{KF}}(\mathbf{x}), \boldsymbol{\xi}))}\right)^2 =: \|\mathbf{r}(\boldsymbol{\xi})\|_2^2$$

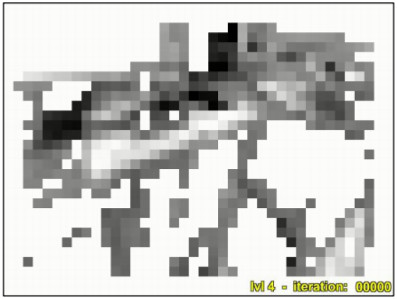
Camera Pose


KF image


KF depth

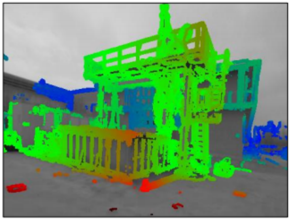
back-warped new frame

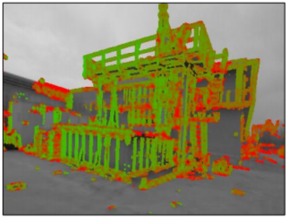
- minimize using **Gauss-Newton / LM** Algorithm
- Coarse-to-fine + Huber norm + statistical norm.



## LSD SLAM - Depth Estimation

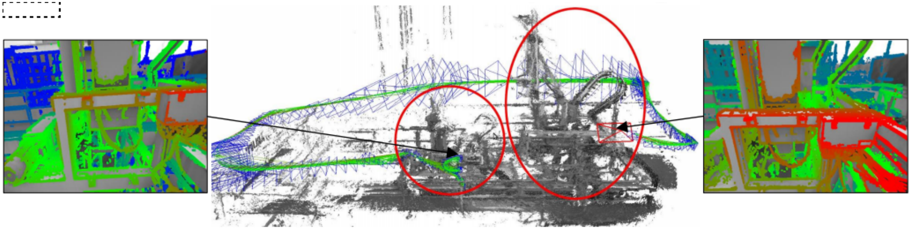
image


inverse depth


inverse depth variance

- filter over many (small-baseline) stereo-correspondences.
- small baseline + epipolar constraint + prior
  -> small search region („track " instead of „detect ")
- only use „good" (sufficiently constrained) pixel.
- Edge-preserving smoothing
- Distance-based KF selection

## LSD SLAM - Global Mapping



- **Direct Tracking with scale (on Sim(3)):**

$$E(\xi) = \sum_{\mathbf{x}\in\Omega_1}\Big(\big(I_1(\mathbf{x}) - I_2(\mathbf{x}')\big)^2 + \big([\mathbf{x}']_3 - D_2(\mathbf{x}')\big)^2\Big)$$

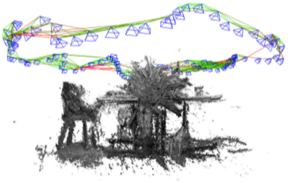
$\xi \leftarrow$ ~~$\mathfrak{se}(3)$~~ $\mathfrak{sim}(3)$ with $\mathbf{x}' := \omega(\mathbf{x}, D_1(\mathbf{x}), \xi)$ (warped point)

**+ GN optimization + multi-resolution + Huber norm + statistical norm.**

- **Optimize pose-graph on Sim(3)**

$$E(\xi_{1W}\dots\xi_{nW}) := \sum_{(\xi_{ij},\Sigma_{ij})\in\mathcal{E}} (\xi_{ij}\circ\xi_{iW}^{-1}\circ\xi_{jW})^T\Sigma_{ij}^{-1}(\xi_{ij}\circ\xi_{iW}^{-1}\circ\xi_{jW}).$$

## LSD SLAM - Results



| | 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 | - |

- [9]: Semi-Dense VO
- [15]: Keypoint Based Mono SLAM
- [14]: Direct RGB-D SLAM
- [7]: Keypoint based RGB-D SLAM

## Semi-Dense Visual Odometry (SVO)

- C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in Robotics and Automation (ICRA), 2014 IEEE International Conference on, pp. 1522, IEEE, 2014.
- Novel semi-direct VO pipeline that is faster and more accurate than state of the art
- Integration of a probabilistic mapping method that is robust to outlier measurements

## SVO - Architecture

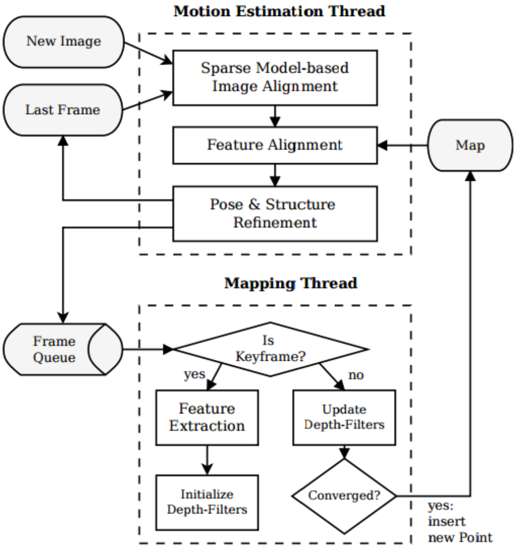


Fig. 1: Tracking and mapping pipeline

## SVO - Important Note

"SVO uses **feature-correspondence only as a result of direct motion estimation rather than of explicit feature extraction and matching**. Thus, feature extraction is only required when a keyframe is selected to initialize new 3d points."

# SVO - Sparse Model Based Image Alignment

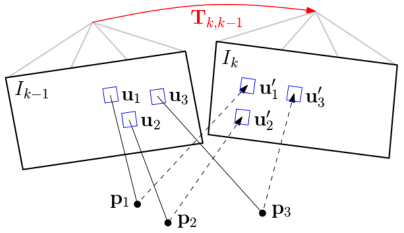


Fig. 2: Changing the relative pose $\mathbf{T}_{k,k-1}$ between the current and the previous frame implicitly moves the position of the reprojected points in the new image $\mathbf{u}'_i$. Sparse image alignment seeks to find $\mathbf{T}_{k,k-1}$ that minimizes the photometric difference between image patches corresponding to the same 3D point (blue squares). Note, in all figures, the parameters to optimize are drawn in red and the optimization cost is highlighted in blue.

Minimize the negative log-likelihood of the intensity residuals:

$$\mathbf{T}_{k,k-1} = \arg\min_{\mathbf{T}} \iint_{\bar{\mathcal{R}}} \rho\left[\delta I(\mathbf{T},\mathbf{u})\right] d\mathbf{u}.$$

## SVO - Feature Alignment

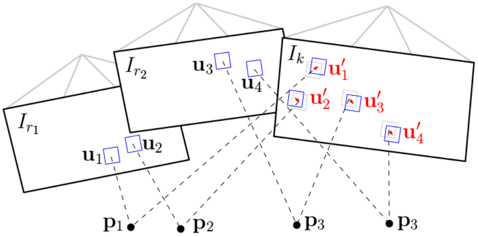


Fig. 3: Due to inaccuracies in the 3D point and camera pose estimation, the photometric error between corresponding patches (blue squares) in the current frame and previous keyframes $r_i$ can further be minimised by optimising the 2D position of each patch individually.

Minimize the photometric error of the patch in the current image with respect to the reference patch in the keyframe r:

$$\mathbf{u}'_i = \arg\min_{\mathbf{u}'_i} \frac{1}{2} \parallel \mathbf{I}_k(\mathbf{u}'_i) - \mathbf{A}_i \cdot \mathbf{I}_r(\mathbf{u}_i) \parallel^2, \quad \forall\ i.$$

## SVO - Pose and Structure Refinement

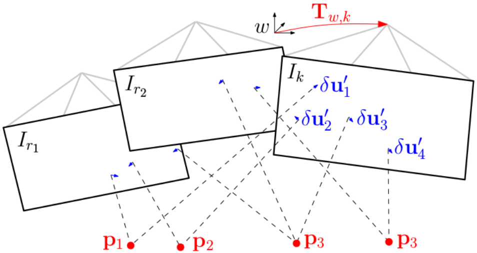


Fig. 4: In the last motion estimation step, the camera pose and the structure (3D points) are optimized to minimize the reprojection error that has been established during the previous feature-alignment step.

Minimize reprojection error (motion only bundle adjustment):

$$\mathbf{T}_{k,w} = \arg\min_{\mathbf{T}_{k,w}} \frac{1}{2} \sum_i \| \mathbf{u}_i - \pi(\mathbf{T}_{k,w}\ {}_w\mathbf{p}_i) \|^2 .$$

- The sparse model based image alignment and pose and structure refinement seems redundant.
- One could directly start establishing feature correspondence but the processing time would be higher. Further some features could be tracked inaccurately, the sparse image alignment step satisfies implicitly the epipolar constraint and ensures that there are no outliers.
- One may also argue that the sparse image alignment would be sufficient to estimate the camera motion, however the authors of SVO found empirically that using the first step only results in a significantly more drift compared to using all three steps together.

## SVO - Results

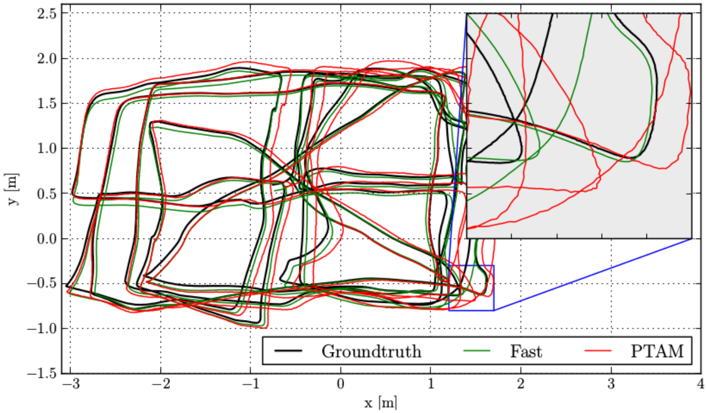


Fig. 7: Comparison against the ground-truth of SVO with the *fast* parameter setting (see Table I) and of PTAM. Zooming-in reveals that the proposed algorithm generates a smoother trajectory than PTAM.

# SVO - Results

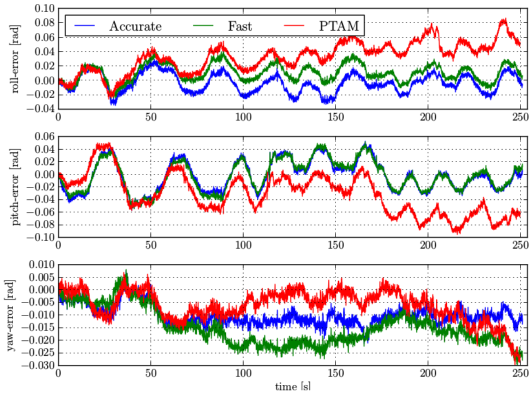


Fig. 9: Attitutde drifts of SVO with *fast* and *accurate* parameter setting and comparison against PTAM.

| | Pos-RMSE [m/s] | Pos-Median [m/s] | Rot-RMSE [deg/s] | Rot-Median [deg/s] |
|---|---|---|---|---|
| *fast* | 0.0059 | 0.0047 | 0.4295 | 0.3686 |
| *accurate* | 0.0051 | 0.0038 | 0.4519 | 0.3858 |
| PTAM | 0.0164 | 0.0142 | 0.4585 | 0.3808 |

TABLE II: Relative pose and rotation error of the trajectory in Figure 7

# SVO - Results

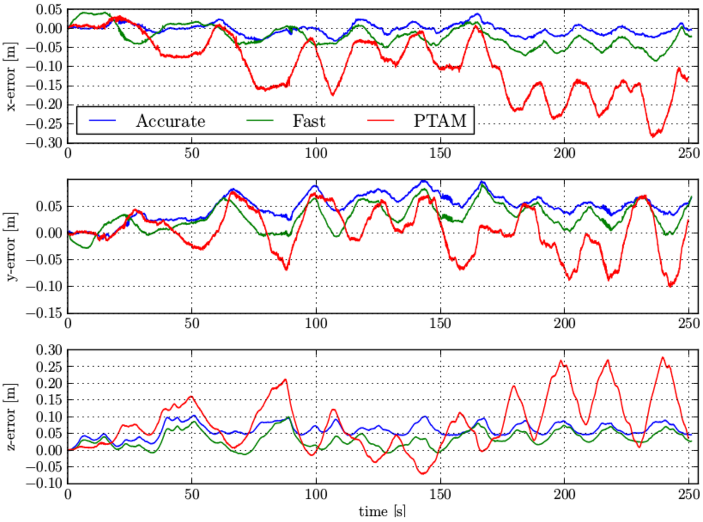


Fig. 8: Position drift of SVO with *fast* and *accurate* parameter setting and comparison against PTAM.

## Direct Sparse Odometry (DSO)

- J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," arXiv preprint arXiv:1607.02565, 2016.
- Proposes a Sparse + Direct method
- Continus optimization of the photometric error over a window of recent frames including geometry and camera motion
- Integrated photometric camera model: lens attenuation, gamma correction, and known exposure times
- Runs real time on CPU

## DSO - Sparse vs Dense Hessian Structure

- In dense approaches, the main drawback of adding a geometric prior is the introduction of correlations between geometry parameters, which render a statistically consistent joint optimization in real time infeasible.

## DSO - Sparse vs Dense Hessian Structure

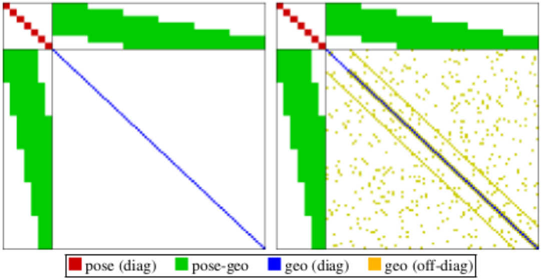


Figure 2. **Sparse vs. dense Hessian structure.** Left: Hessian structure of sparse bundle adjustment: since the geometry-geometry block is diagonal, it can be solved efficiently using the Schur complement. Right: A geometry prior adds (partially unstructured) geometry-geometry correlations – the resulting system is hence not only much larger, but also becomes much harder to solve. For simplicity, we do not show the global camera intrinsic parameters.

## Discussion

- Is front-end done? What more can we improve?
- Is it realistic to aim for a front-end that could perform in:
  - Poorly illuminated environments
  - Textureless environments
  - Low camera frame rate
- What is our goal with gimbal VO?