Photometric Bundle Adjustment for Globally Consistent Mapping

Simon Klenk

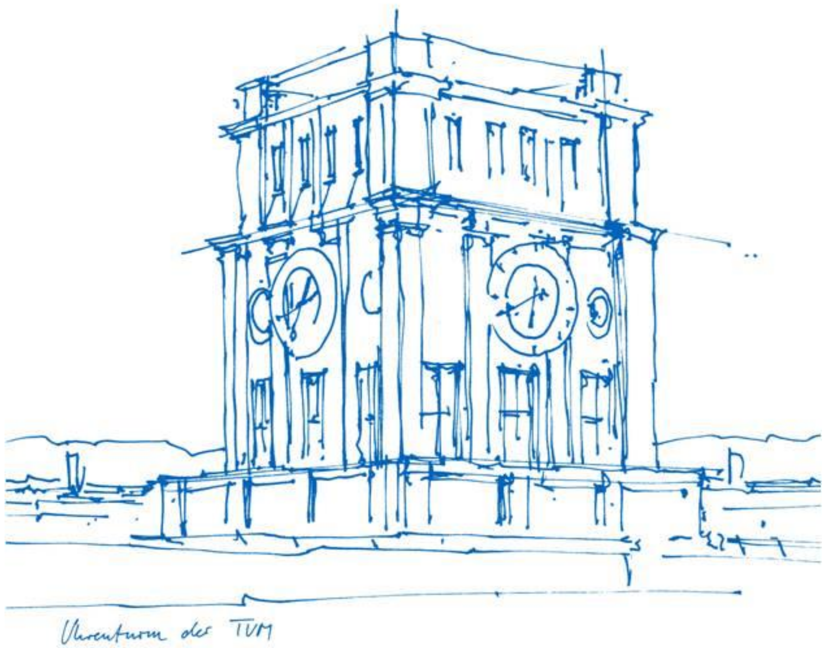
Technische Universität München

Chair of Computer Vision & AI

Master Thesis

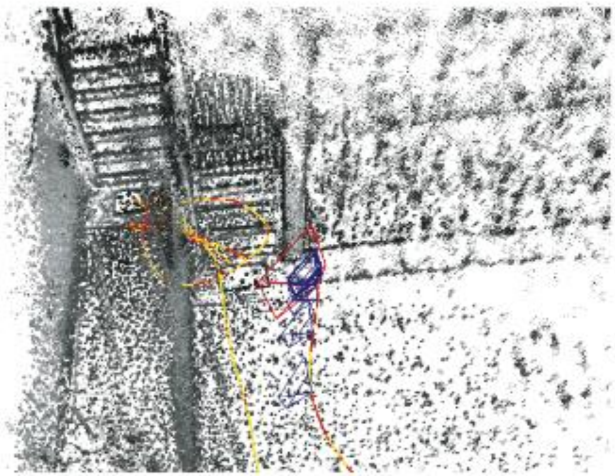
Advisor: M.Sc. Nikolaus Demmel

Supervisor: Prof. Dr. Daniel Cremers

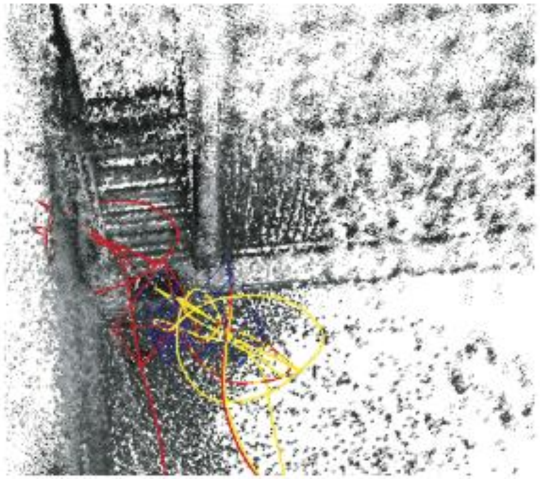


Motivation: Improving Photometric Maps

Before Loop Closure



After Loop Closure



Direct Sparse Odometry with Loop Closure [1]

Stairs converges to one object

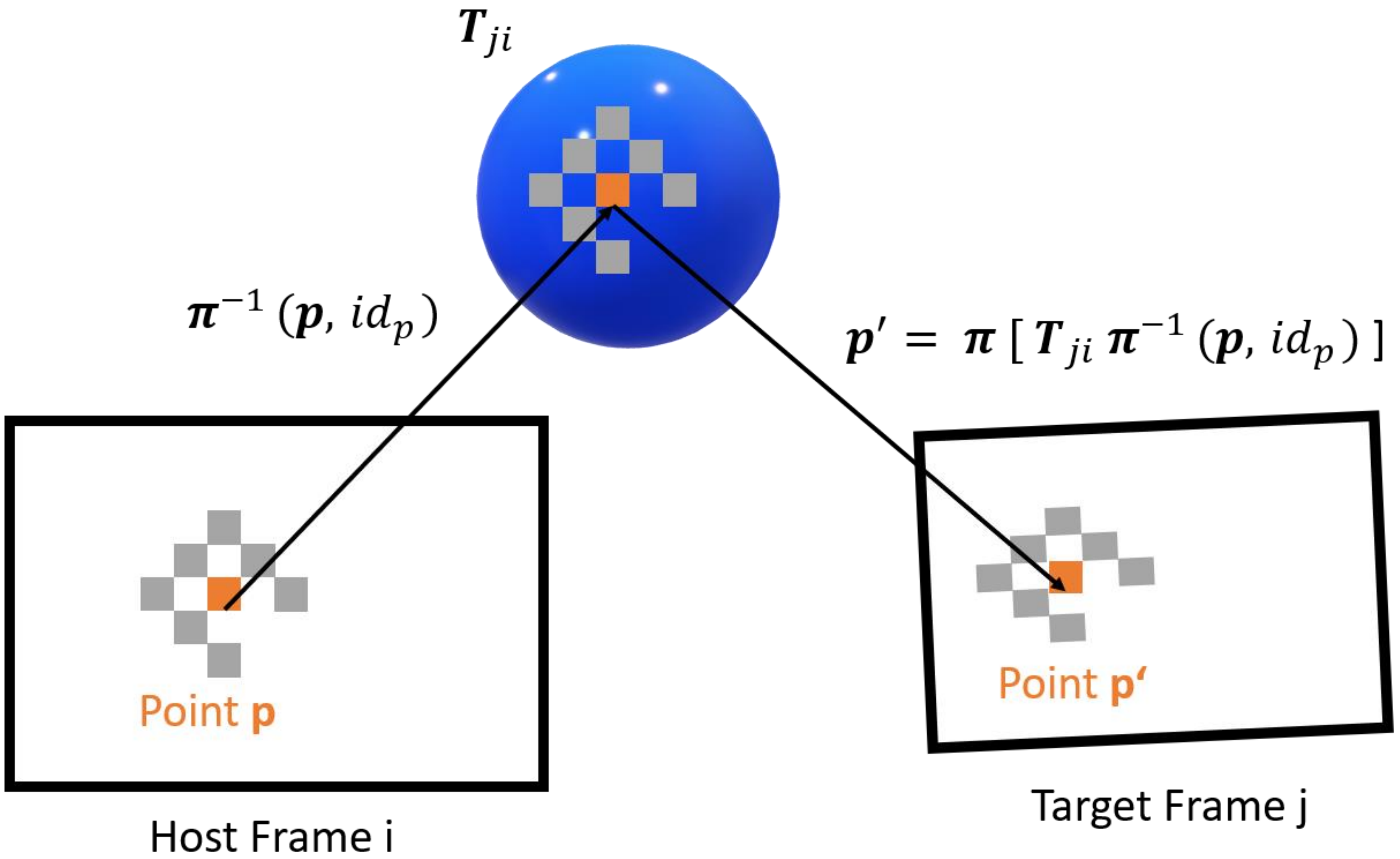
Even more Improvement:

Photometric **B**undle **A**djustment

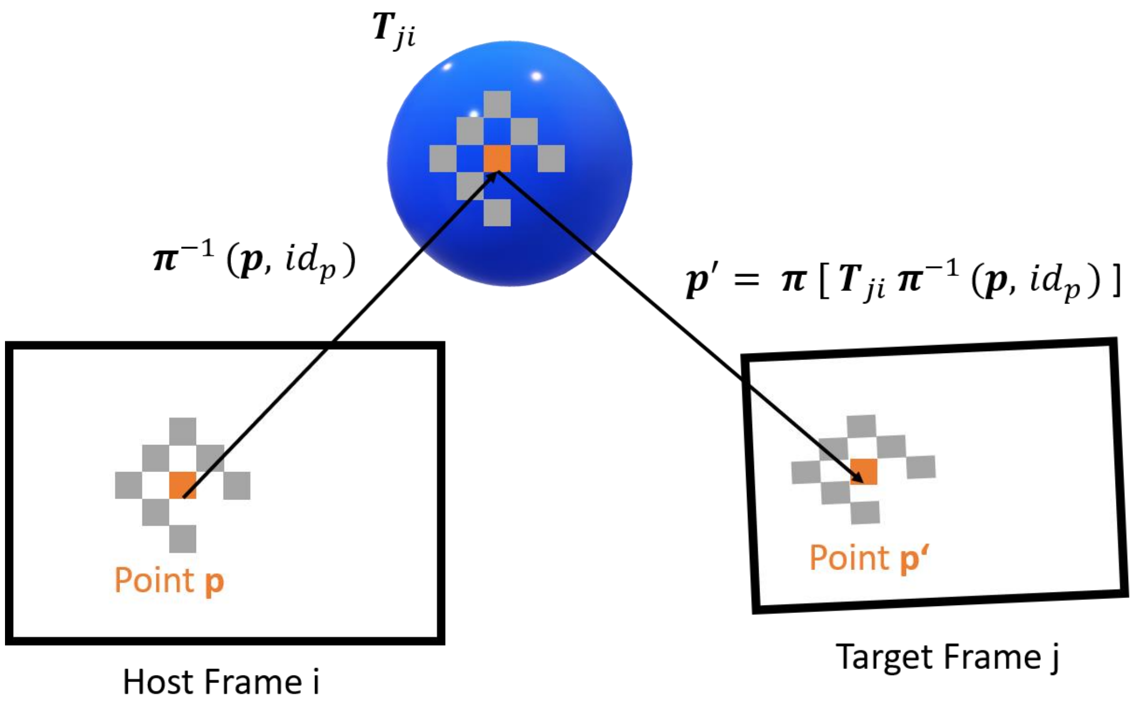
Research Question:
Is the current implementation without alternative?

Evaluation:
Kitti odometry 00-10
Euroc MAV

PBA Cost Formulation: Direct Image Error



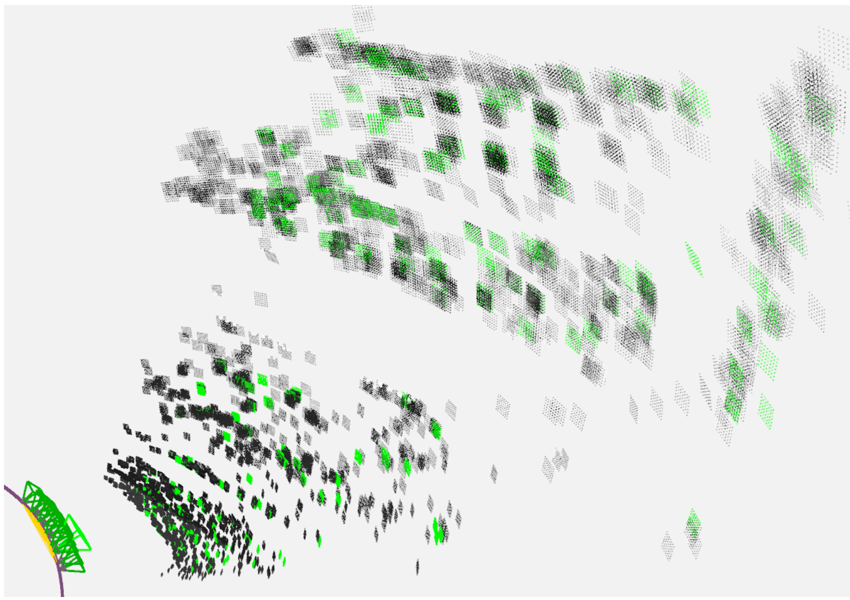
PBA Cost Formulation: Direct Image Error

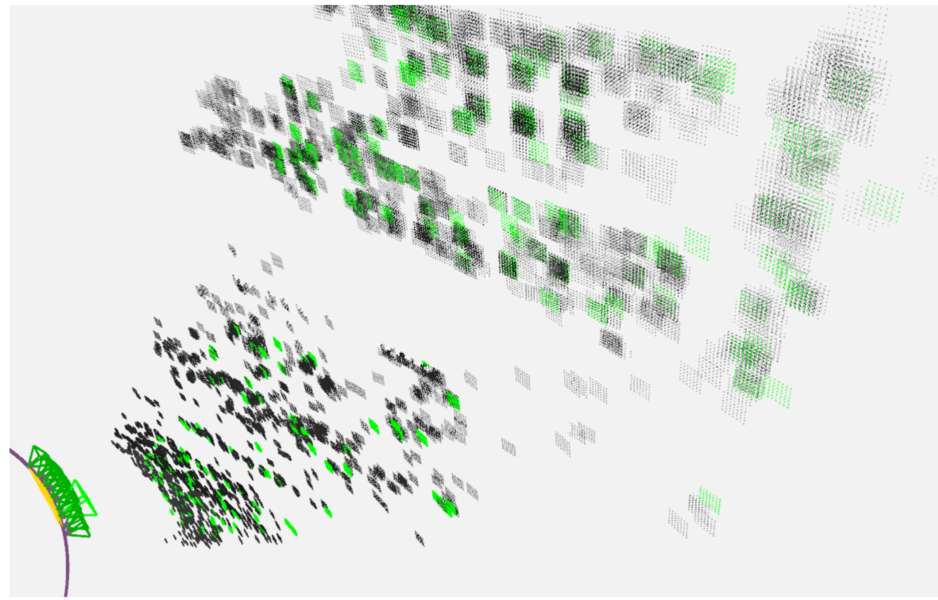


$$E_{photo} = \sum_{frames} \sum_{points} \sum_{obs} \sum_{pattern} \left\| I_j[p'] - I_i[p] \right\|_{Huber}$$

Residual

Residual Pattern Geometry



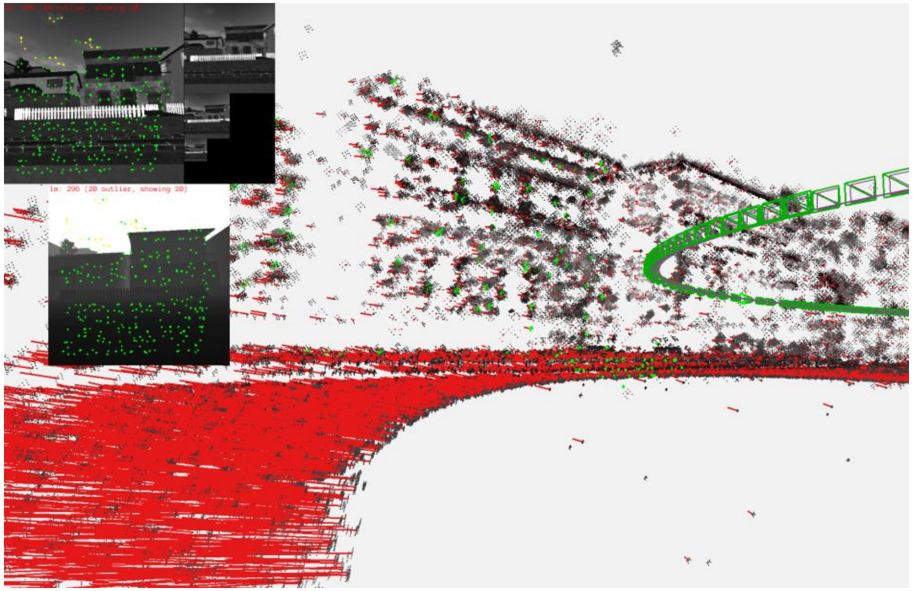


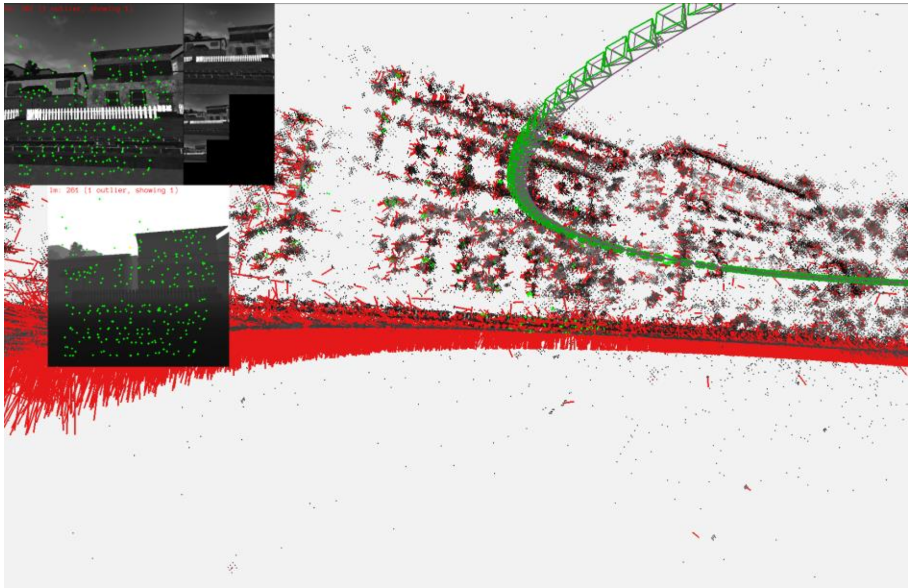
Spherical Patterns
(inverse distance)
$\mathbf{0.675}\ \boldsymbol{ATE_{avg}}$

Which is better?

Planar Patterns
(inverse depth)
$0.684\ ATE_{avg}$

Residual Pattern: Normal Vectors





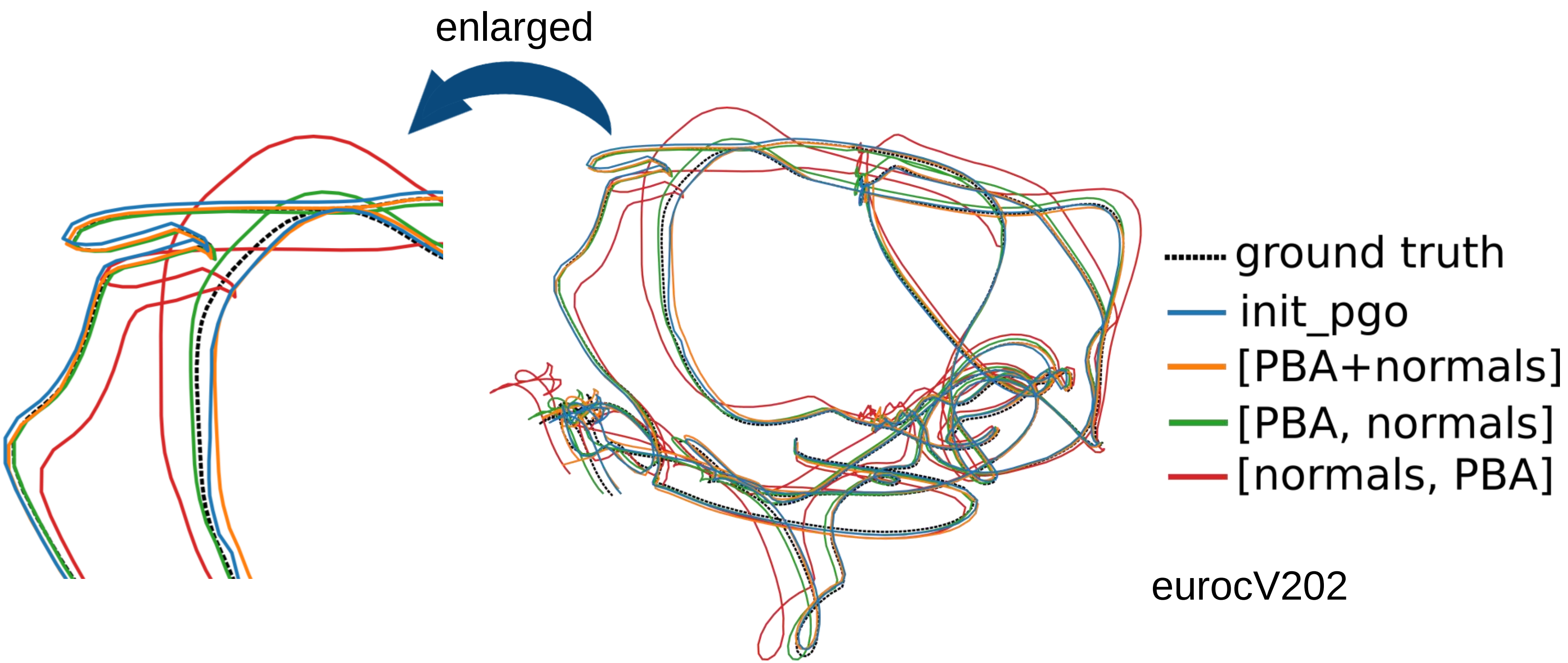
Initialization

After normal vector optimization

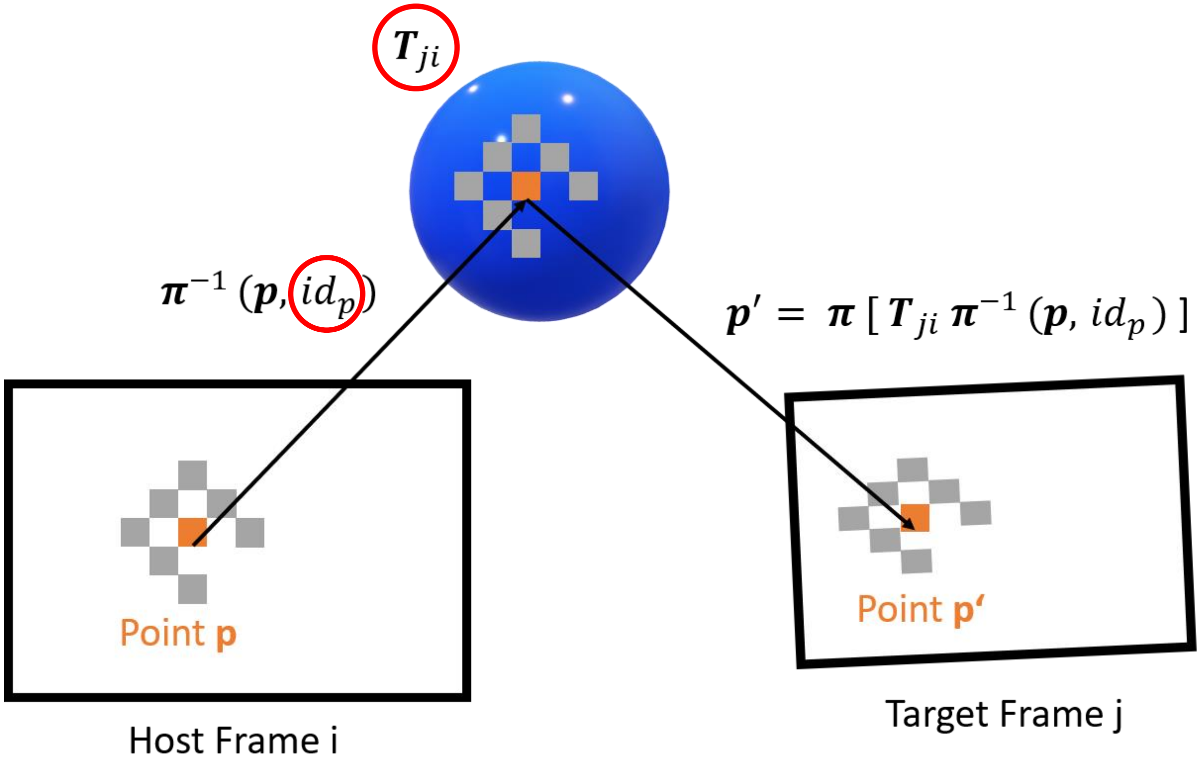
How to optimize the normal vectors?

Residual Pattern: Normal Vectors

	$[PBA, normals]$	$[normals, PBA]$	$[PBA + normals]$
all sequences	**0.672**	0.762	0.731

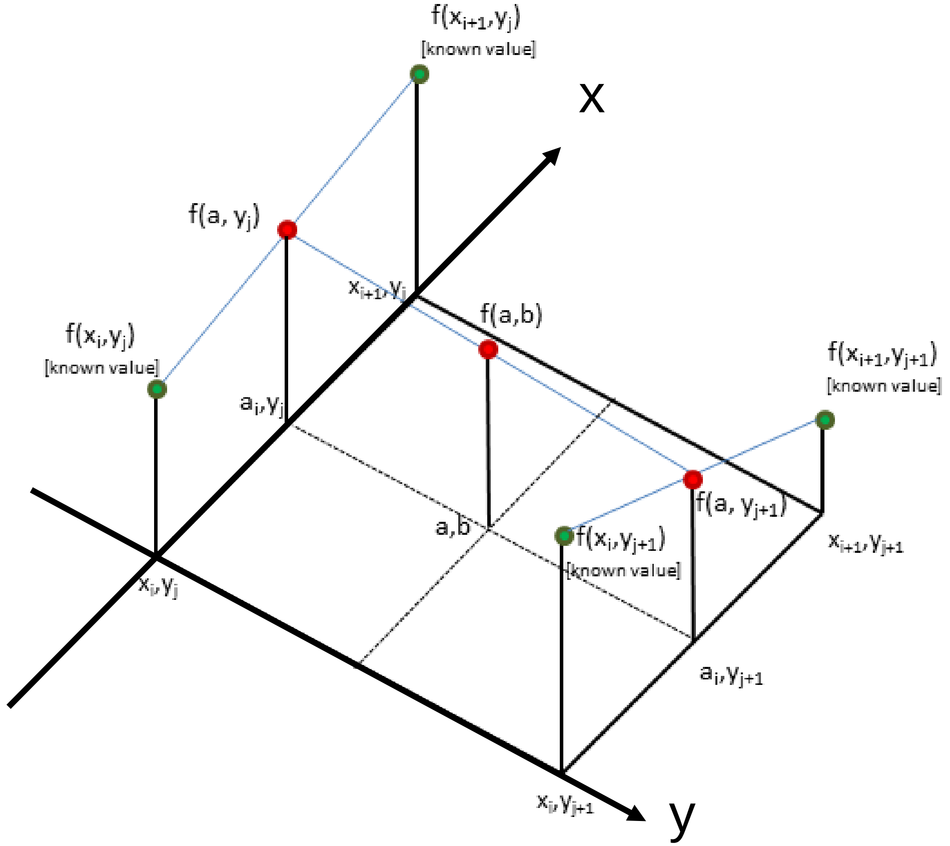


Where else did we have a closer look?



$$E_{photo} = \sum_{frames} \sum_{points} \sum_{obs} \sum_{pattern} \left\| I_j\,[p'] - I_i\,[p] \right\|_{Huber}$$

Host-Target Transformation: Interpolation in Target



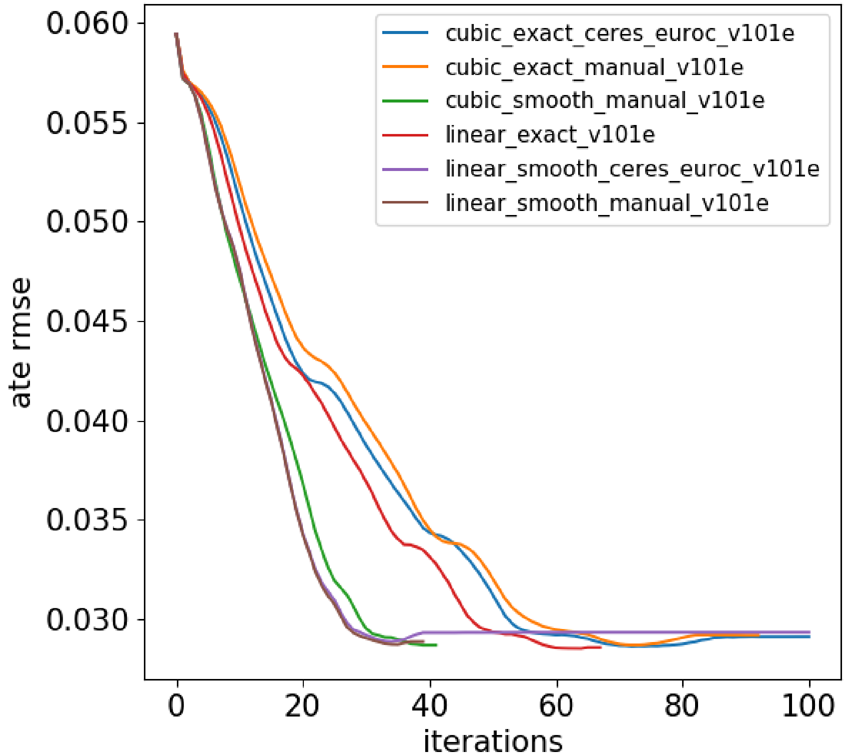
Bilinear interpolation [2]

Computing **exact gradients**

↕

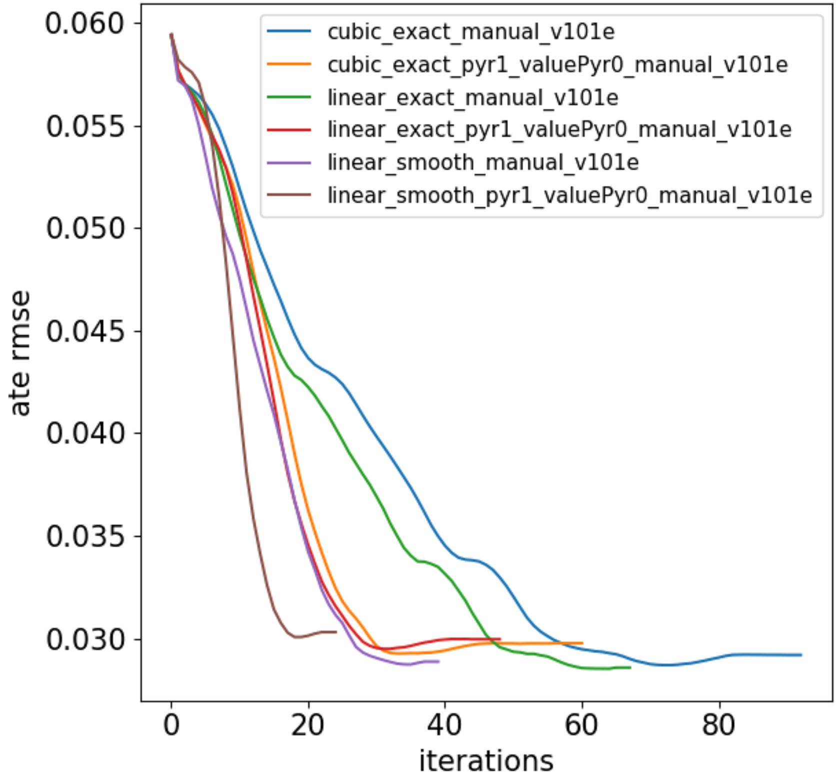
Computing **smooth gradients**: using gradient image (central differences)

Host-Target Transformation: Interpolation in Target



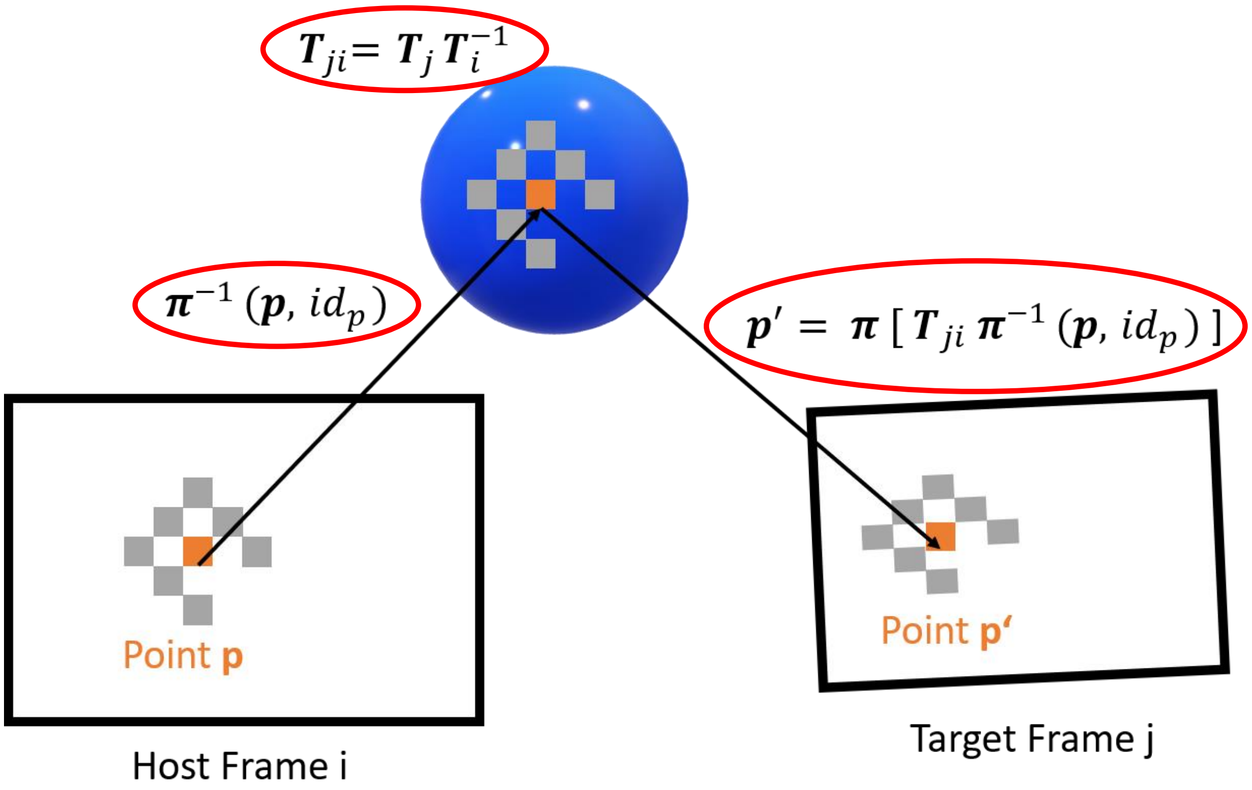
	init_pgo	bilin	bilin_s	bicubic	bicubic_s	bicubic_smooth@20It
$ATE_{rmse,geo}$	1.0	0.671	0.673	0.669	0.666	**0.657**

Host-Target Transformation: Interpolation in Target



Smooth gradients are similar to interpolating on image pyramid

Where else did we have a closer look?

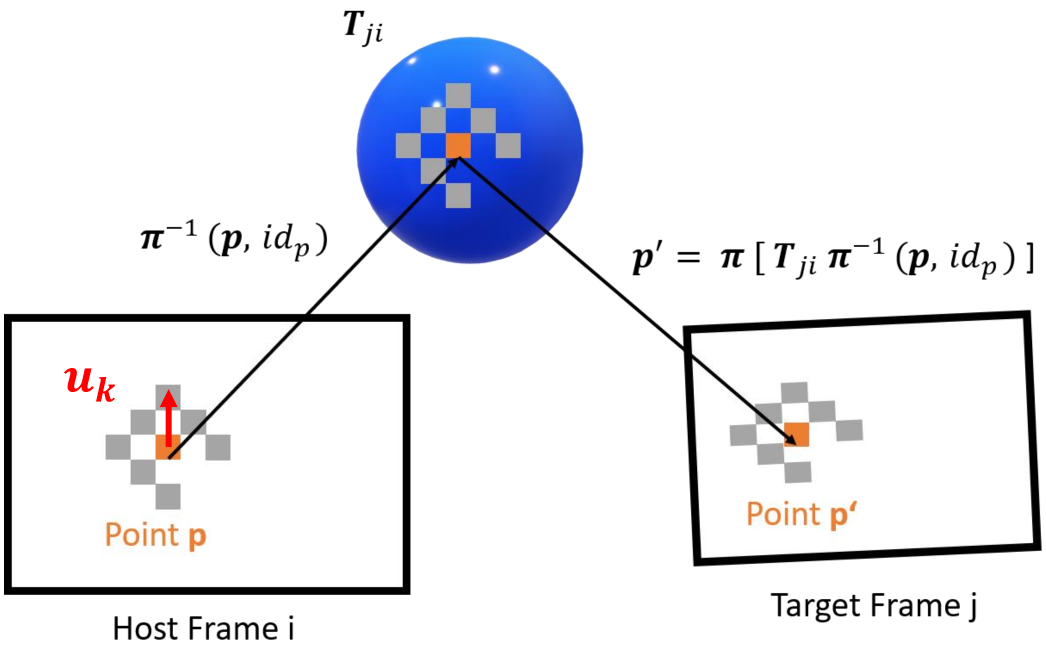


$$E_{photo} = \sum_{frames} \sum_{points} \sum_{obs} \sum_{pattern} \left\| I_j [p'] - I_i [p] \right\|_{Huber}$$

Host-Target Transformation: Approximation

full warp:

$$p_k' = \pi\,[T_{ji}\,\pi^{-1}\,(p_0 + u_k,\, id_p)]$$



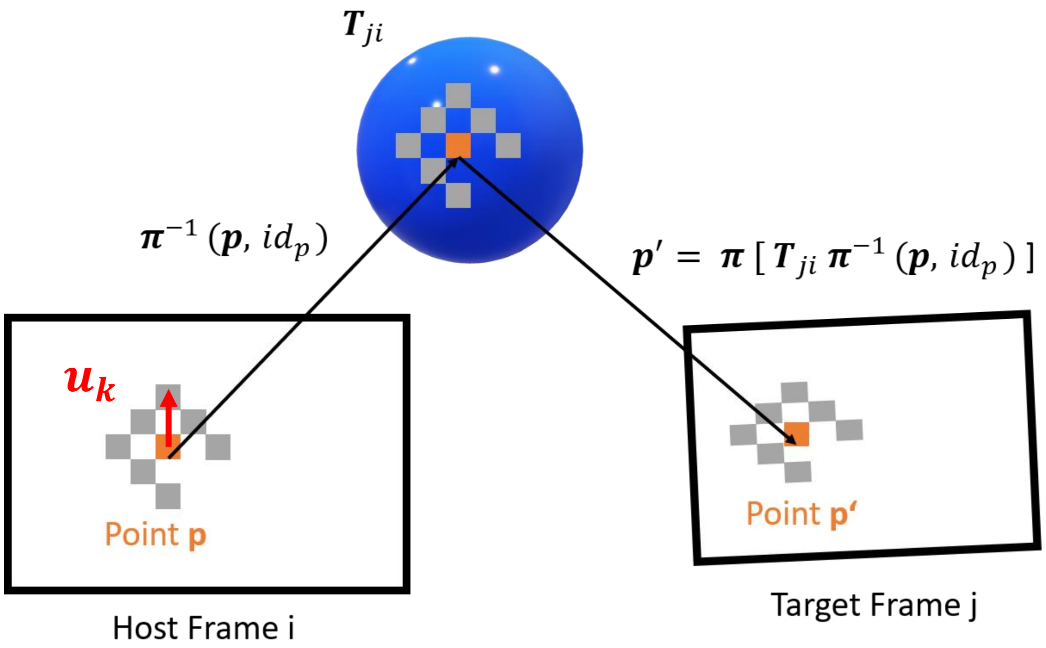
Host-Target Transformation: Approximation

full warp:

$$\boldsymbol{p_k}' = \boldsymbol{\pi}\,[\boldsymbol{T}_{ji}\,\boldsymbol{\pi}^{-1}\,(\boldsymbol{p_0} + \boldsymbol{u_k},\, id_p)]$$

simple warp:

$$\boldsymbol{p_k}' = \boldsymbol{\pi}\,[\boldsymbol{T}_{ji}\,\boldsymbol{\pi}^{-1}\,(\boldsymbol{p_0},\, id_p)] + \boldsymbol{u_k}$$



Host-Target Transformation: Approximation

full warp:

$$\boldsymbol{p_k}' = \boldsymbol{\pi}\,[\boldsymbol{T}_{ji}\,\boldsymbol{\pi}^{-1}\,(\boldsymbol{p_0} + \boldsymbol{u_k},\, id_p)]$$

simple warp:

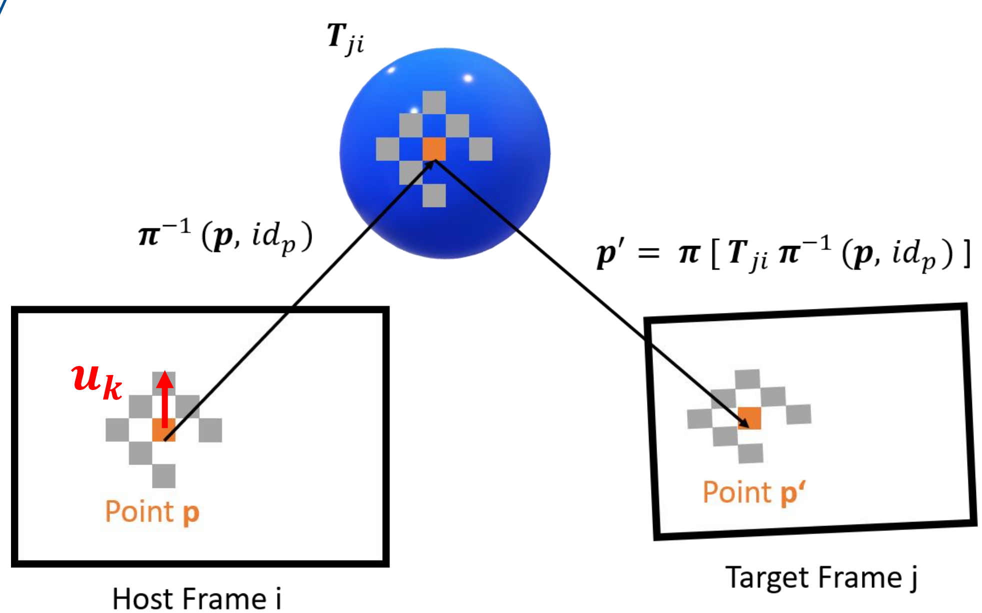
$$\boldsymbol{p_k}' = \boldsymbol{\pi}\,[\boldsymbol{T}_{ji}\,\boldsymbol{\pi}^{-1}\,(\boldsymbol{p_0},\, id_p)] + \boldsymbol{u_k}$$

exact full warp:

1) Warp all exactly

approximate full warp:

1) Warp by 1st order Taylor at $\boldsymbol{p_0}$
2) Jacobian only for central pixel $\boldsymbol{p_0}$



Host-Target Transformation: Approximation

full warp:

$$\boldsymbol{p_k}' = \boldsymbol{\pi}\,[\boldsymbol{T}_{ji}\,\boldsymbol{\pi}^{-1}\,(\boldsymbol{p_0} + \boldsymbol{u_k},\, id_p)]$$

exact ⟷ **approximate**

⟷

simple warp:

$$\boldsymbol{p_k}' = \boldsymbol{\pi}\,[\boldsymbol{T}_{ji}\,\boldsymbol{\pi}^{-1}\,(\boldsymbol{p_0},\, id_p)] + \boldsymbol{u_k}$$

	DSO pattern		
warp:	exact	approx	simple
all	0.707	**0.693**	**0.693**

Host-Target Transformation: Approximation

full warp:

$$p_k' = \pi\, [T_{ji}\, \pi^{-1}\, (p_0 + u_k,\, id_p)]$$

exact ⟷ **approximate**

⟷

simple warp:

$$p_k' = \pi\, [T_{ji}\, \pi^{-1}\, (p_0,\, id_p)] + u_k$$

	DSO pattern		
warp:	exact	approx	simple
all	0.707	**0.693**	**0.693**
euroc-ok	**0.688**	0.691	0.690
euroc-fail	1.183	**0.997**	1.006

Host-Target Transformation: Approximation

full warp:

$$\boldsymbol{p}_k{}' = \boldsymbol{\pi}\,[\boldsymbol{T}_{ji}\,\boldsymbol{\pi}^{-1}\,(\boldsymbol{p}_0 + \boldsymbol{u}_k,\, id_p)]$$

exact ⟷ **approximate**

⟷

simple warp:

$$\boldsymbol{p}_k{}' = \boldsymbol{\pi}\,[\boldsymbol{T}_{ji}\,\boldsymbol{\pi}^{-1}\,(\boldsymbol{p}_0,\, id_p)] + \boldsymbol{u}_k$$

	DSO pattern		
warp:	exact	approx	simple
all	0.707	**0.693**	**0.693**
euroc-ok	**0.688**	0.691	0.690
euroc-fail	1.183	**0.997**	1.006
kit-no-loop	0.719	0.724	**0.702**
kit-loop	**0.548**	0.568	0.579

Host-Target Transformation: Approximation

full warp:

$$\boldsymbol{p}_k{}' = \boldsymbol{\pi}\,[\boldsymbol{T}_{ji}\,\boldsymbol{\pi}^{-1}\,(\boldsymbol{p}_0 + \boldsymbol{u}_k,\, id_p)]$$

exact ⟷ **approximate**

⟷

simple warp:

$$\boldsymbol{p}_k{}' = \boldsymbol{\pi}\,[\boldsymbol{T}_{ji}\,\boldsymbol{\pi}^{-1}\,(\boldsymbol{p}_0,\, id_p)] + \boldsymbol{u}_k$$

	DSO pattern			9x9 sparse			13x13 sparse		
warp:	exact	approx	simple	exact	approx	simple	exact	approx	simple
all	0.707	**0.693**	**0.693**	**0.739**	0.743	0.787	**0.785**	0.796	0.847
euroc-ok	**0.688**	0.691	0.690						
euroc-fail	1.183	**0.997**	1.006						
kit-no-loop	0.719	0.724	**0.702**						
kit-loop	**0.548**	0.568	0.579						

Host-Target Transformation: Approximation

full warp:

$$\boldsymbol{p}_k{}' = \boldsymbol{\pi}\,[\boldsymbol{T}_{ji}\,\boldsymbol{\pi}^{-1}\,(\boldsymbol{p}_0 + \boldsymbol{u}_k,\, id_p)]$$

exact ⟷ **approximate**

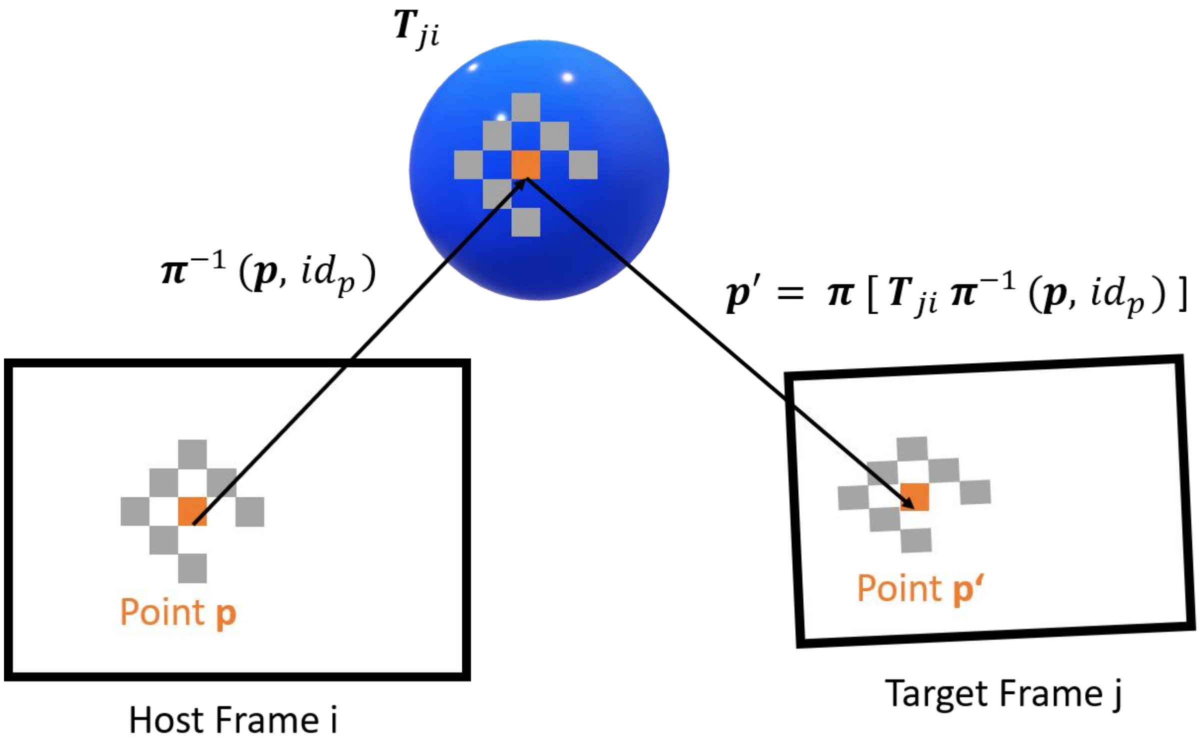
⟷

simple warp:

$$\boldsymbol{p}_k{}' = \boldsymbol{\pi}\,[\boldsymbol{T}_{ji}\,\boldsymbol{\pi}^{-1}\,(\boldsymbol{p}_0,\, id_p)] + \boldsymbol{u}_k$$

	DSO pattern			9x9 sparse			13x13 sparse		
warp:	exact	approx	simple	exact	approx	simple	exact	approx	simple
all	0.707	**0.693**	**0.693**	**0.739**	0.743	0.787	**0.785**	0.796	0.847
euroc-ok	**0.688**	0.691	0.690	0.738	**0.731**	0.779	**0.795**	0.796	0.930
euroc-fail	1.183	**0.997**	1.006	0.996	0.996	**0.986**	1.015	**1.013**	1.010
kit-no-loop	0.719	0.724	**0.702**	0.778	0.781	**0.766**	0.736	0.774	**0.699**
kit-loop	**0.548**	0.568	0.579	**0.566**	0.566	0.603	**0.579**	0.582	0.640

Where else did we have a closer look?

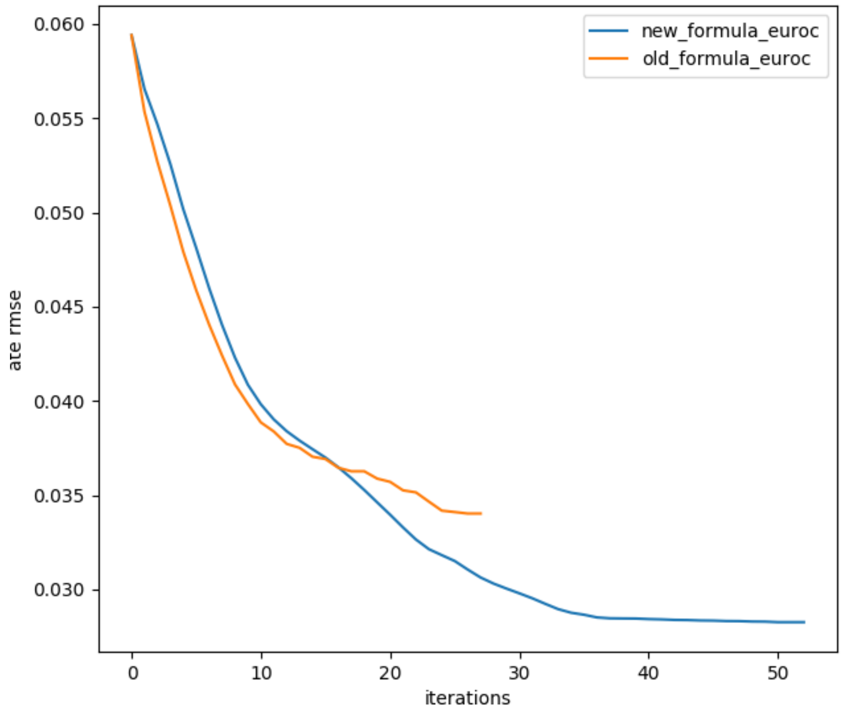


$$E_{photo} = \sum_{frames} \sum_{points} \sum_{obs} \sum_{pattern} \left\| I_j\,[p'] - I_i\,[p] \right\|_{t-distribution}$$

Robust Norms: t-distribution

ours: $w_{i,t} = \frac{1}{\sigma_t{}^2} \frac{v+1}{v+\left(\frac{r_i}{\sigma_t}\right)^2}$

old [3]: $w_{i,t} = \frac{v+1}{v+\left(\frac{r_i}{\sigma_t}\right)^2}$



cost: $c = \sum_i w_{i,t} r_i{}^2$

Robust Norms: t-distribution

$$c = \sum_i w_{i,t} {r_i}^2$$

weight:	TDist weight $w_{i,corrected}$	TDist weight $w_{i,old}$
all sequences	**0.708**	**0.708**
euroc-ok	**0.627**	0.702
euroc-fail&eurocV202	1.355	**1.053**
kit-no-loop	**0.520**	0.584
kit-loop	0.720	**0.683**

ours: $$w_{i,t} = \frac{1}{{\sigma_t}^2} \frac{v+1}{v+\left(\frac{r_i}{\sigma_t}\right)^2}$$

↔

old [3]: $$w_{i,t} = \frac{v+1}{v+\left(\frac{r_i}{\sigma_t}\right)^2}$$

Robust Norms: t-distribution

$$c = \sum_i w_{i,t} {r_i}^2$$

weight:	TDist weight $w_{i,corrected}$	$w_{i,old}$	average CoV
all sequences	**0.708**	**0.708**	0.58
euroc-ok	**0.627**	0.702	0.65
euroc-fail&eurocV202	1.355	**1.053**	0.74
kit-no-loop	**0.520**	0.584	0.51
kit-loop	0.720	**0.683**	0.41

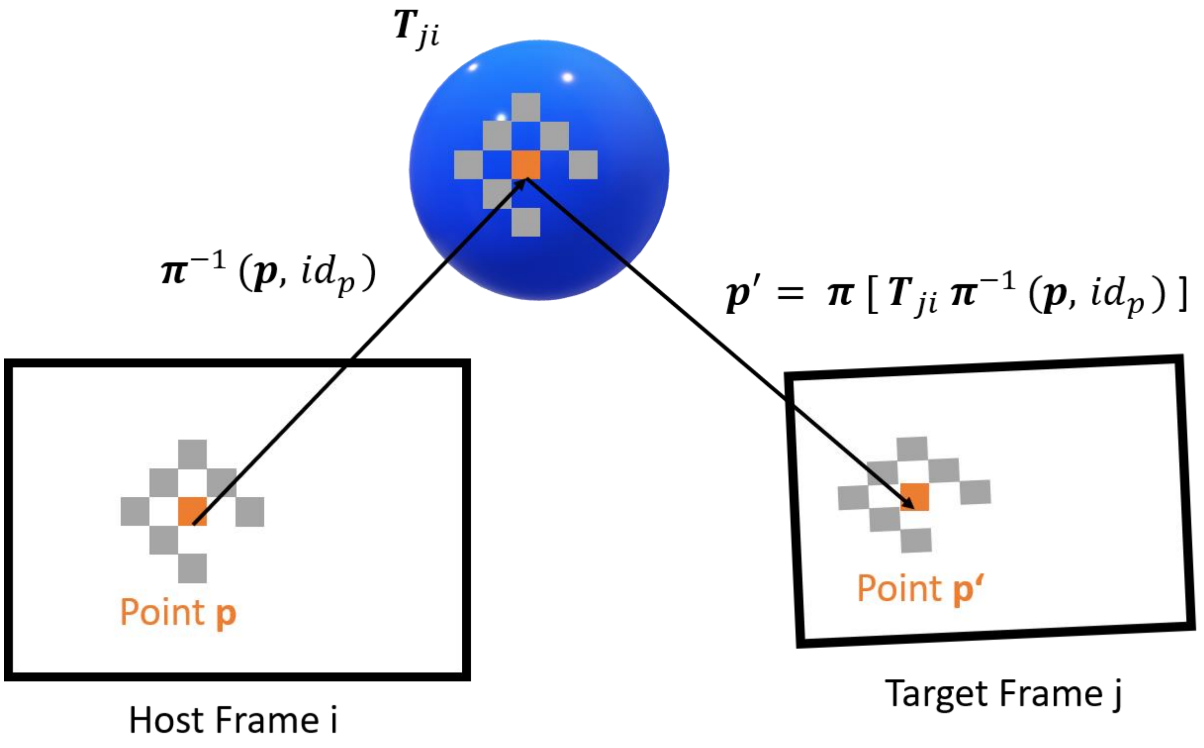
$$\mathbf{CoV} = \frac{\mathbf{var}(\{\sigma\})}{\mathbf{mean}(\{\sigma\})}$$

ours: $$w_{i,t} = \frac{1}{{\sigma_t}^2} \frac{v+1}{v+\left(\frac{r_i}{\sigma_t}\right)^2}$$

⟺

old [3]: $$w_{i,t} = \frac{v+1}{v+\left(\frac{r_i}{\sigma_t}\right)^2}$$

Where else did we have a closer look?



$$\boldsymbol{E}_{photo} = \sum_{frames} \sum_{points} \sum_{obs} \sum_{pattern} \left\| I_j[\boldsymbol{p}'] - I_i[\boldsymbol{p}] \right\|_{Huber}$$

Residual Formulations

- **Explicit brightness model (per image): *ABOPT***

$$\mathbf{r}_{ab}^{(k)} = (I_j[\mathbf{p}'_k] - b_j) - \frac{e^{a_j}}{e^{a_i}}(I_i[\mathbf{p}_k] - b_i)$$

- **Implicit brightness model (per patch): *LSSD, LNSSD, ZNCC/ZNSSD***

$$\mathbf{r}_{lssd}^{(k)} = I_j[\mathbf{p}'_k] - \frac{\overline{\mathbf{I}}_j}{\overline{\mathbf{I}}_i} I_i[\mathbf{p}_k]$$

residuals:	SSD	LSSD	LNSSD	ABOPT
all sequences	0.693	**0.658**	0.670	0.666

Residual Formulations

- **Explicit brightness model (per image): *ABOPT***

$$\mathbf{r}_{ab}^{(k)} = (I_j[\mathbf{p}'_k] - b_j) - \frac{e^{a_j}}{e^{a_i}}(I_i[\mathbf{p}_k] - b_i)$$

- **Implicit brightness model (per patch): *LSSD, LNSSD, ZNCC/ZNSSD***

$$\mathbf{r}_{lssd}^{(k)} = I_j[\mathbf{p}'_k] - \frac{\bar{\mathbf{I}}_j}{\bar{\mathbf{I}}_i} I_i[\mathbf{p}_k]$$

$$2 * (1 - ZNCC) = ZNSSD$$

residuals:	SSD	LSSD	LNSSD	ABOPT	ZNCC	ZNSSD
all sequences	0.693	**0.658**	0.670	0.666	0.751	0.676

Overview of other experiments

- **Huber:**
  - Per-target frame works, with different scale estimator (same as for t-distribution, MAD, or sample standard deviation tested)
- **Self-tuning M-estimation [4]:**
  - Achieves very good results for t-distribution
  - most general and therefore preferred
- **LM dampening:**
  - No big difference between options, most efficient should be used, e.g. only landmark dampening (identity or original Hessian or Schur)
- **LM step criteria:**
  - Okay to evaluate PBA cost or linearized costs, theoretically OLS correct
- **Triggs correction**:
  - Second order correction of Hessian for robust loss
  - Small improvement for t-distribution, for Huber not because only outlier contribute to corrected Hessian
- **Occlusion geometric & photometric**:
  - Simple approaches results only in very minor improvement

Conclusions

- **Use** residuals which account for brightness changes
- **Use** smooth gradients in the beginning, exact gradients in the end
- **Use** full warp: approximated version is usually fine, simple warp is too simple
- **Use** normal optimization as separate step after PBA
- **Use** self-tuning approach (or corrected formula for t-distribution)
- **Use** Triggs-correction for t-distribution case
- **Use** any kind of dampening (diagonal of Hessian/Schur or identity)

- **Future Work:**
  - different metrics required! (especially map evaluation)
  - Numerical properties
  - Occlusion detections / Deduplication
  - Benchmark on more data & against DL / feature-based

Thanks for listening and asking questions!

Sources

[1] X. Gao, R. Wang, N. Demmel and D. Cremers,LDSO: Direct Sparse Odometry with Loop Closure,iros, October 2018

[2] Stackoverflow Answer by Niek Sanders (user: nsanders) on 10.01.2012. Question: „Bilinear interpolation to enlarge bitmap images asked on 10.01.2012. Link: https://stackoverflow.com/questions/8808996/bilinear-interpolation-to-enlarge-bitmap-images [last acces 13.06.2020]

[3] J. Zubizarreta, I. Aguinaga, and J. M. M. Montiel. “Direct Sparse Mapping.” In: CoRR abs/1904.06577 (2019). arXiv: 1904.06577.

[4] G. Agamennoni, P. Furgale, and R. Siegwart. “Self-tuning M-estimators.” In: 2015 IEEE International Conference on Robotics and Automation (ICRA). May 2015, pp. 4628–4635. doi: 10.1109/ICRA.2015.7139840.