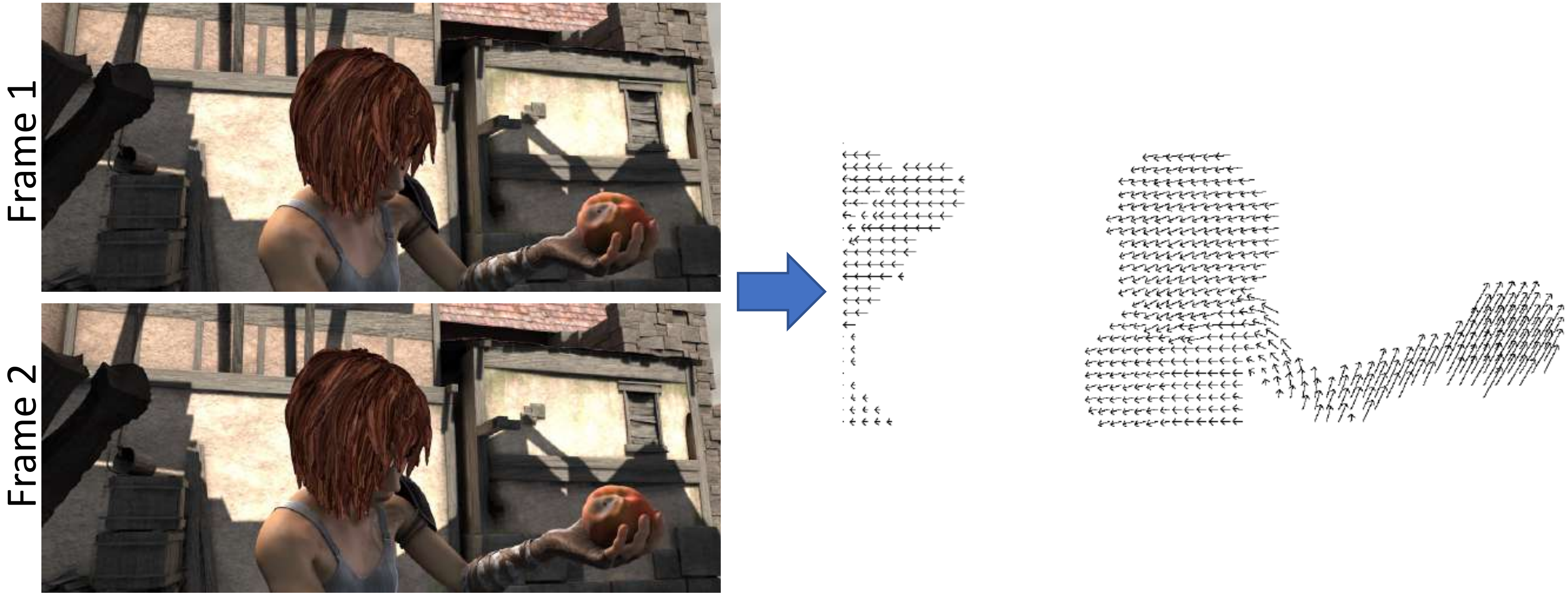
CS231N: Low-Level Vision

Jia Deng

Optical Flow

- Predict per-pixel 2D motion between a pair of frames



Applications

Robotics

AR/VR

Self-driving cars (Waymo)

Everydayrobots.com

Project starline (Google)

Hololens (Microsoft)

Optical Flow as Optimization

- Objective: appearance constancy + plausibility of flow field

$$E(\Delta x) = Distance\big(I(x_i), I(x_i + \Delta x_i)\big) + Regularization(I, \Delta x)$$

[Horn and Schunck, 1981]
[Black and Anandan, 1993]
[Zach et al. 2007]

[Brox et al. 2004]
[Brox and Malik, 2010]
[Weinzaepfel et al, 2013]

[Liu et al. 2009]
[Roth et al. 2009]
[Menze et al, 2015]
[Sun et al, 2010]

[Bailer et al. 2015]
[Chen and Koltun, 2016]
[Xu et al, 2017]

Optical Flow

- Classical approaches:

The Model of Horn and Schunck [1]

$$\min_{u,v}\left\{E = \underbrace{\int_\Omega |\nabla u|^2 + |\nabla v|^2 \; d^2x}_{\text{Regularization Term}} + \lambda \underbrace{\int_\Omega \rho(u,v)^2 \; d^2x}_{\text{Data Term (OFC)}}\right\}$$

$$\rho(u,v) = I_t + (u,v)\cdot\nabla I \approx 0$$

+ Convex
+ Easy to solve
- Does not allow for sharp edges in the solution
- Sensitive to outliers violating the OFC

[1] Horn and Schunck. Determinig Optical Flow. Artificial Intelligence, 1981

Slide credit: Thomas Pock and Daniel Cremers

Optical Flow

- Classical approaches: TV-L1 Flow (TV: total variation)
  - Replace quadratic functions by L_1 – norms
  - Done by Cohen, Aubert, Brox, Bruhn, ...

$$\min_{u,v} \left\{ E = \int_\Omega |\nabla u| + |\nabla v| \; d^2x + \lambda \int_\Omega |\rho(u,v)| \; d^2x \right\}$$

+Allows for discontinuities in the flow field

+Robust to some extent to outliers in the OFC

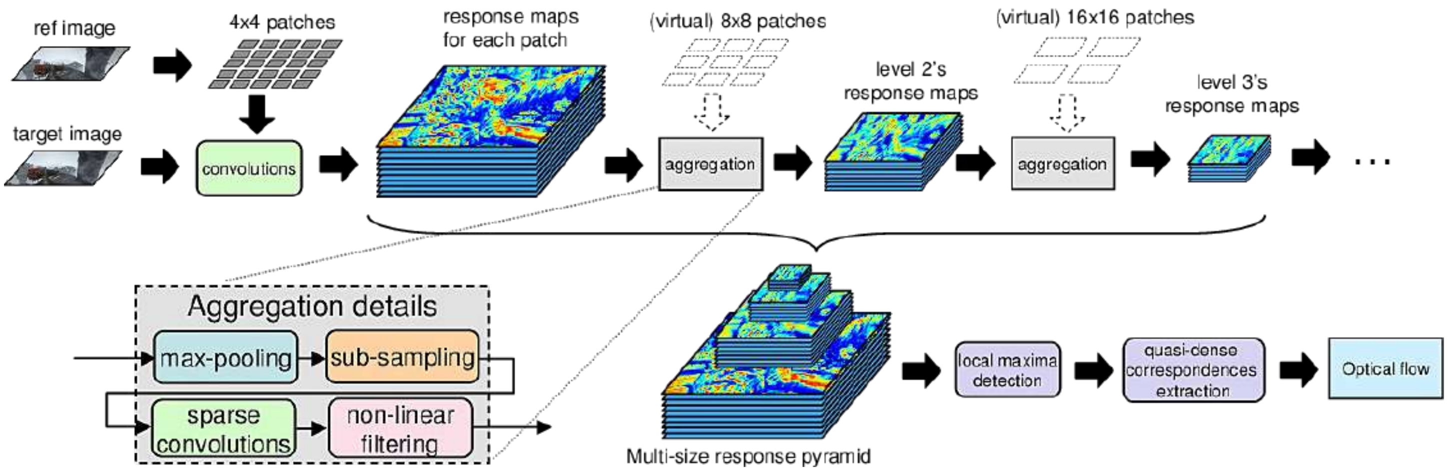
+Still convex

- Much harder to solve

See: Zach, C., Pock, T. and Bischof, H., 2007, September. A duality based approach for realtime TV-L 1 optical flow. In Joint pattern recognition symposium (pp. 214-223). Springer, Berlin, Heidelberg.

Optical Flow

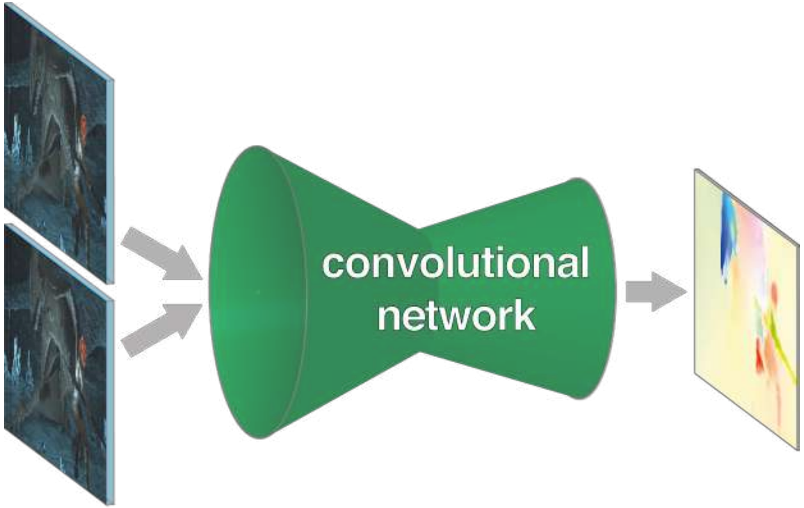
- Classical approaches: DeepFlow



Weinzaepfel P, Revaud J, Harchaoui Z, Schmid C. DeepFlow: Large displacement optical flow with deep matching. InProceedings of the IEEE international conference on computer vision 2013 (pp. 1385-1392).

FlowNet [Dosovitskiy et al. 2015]

- First optical flow network
- U-Net on concatenated frames
- Simple and Fast -- but underperforming the best optimization approaches



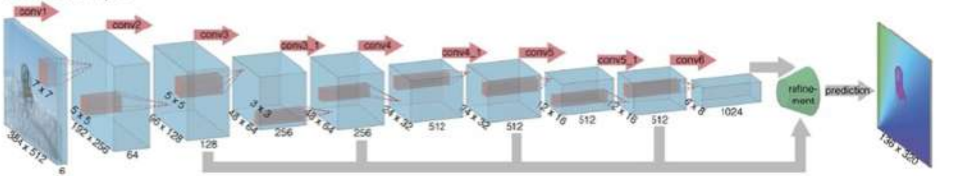
Optical Flow

- Deep Learning: FlowNet

FlowNet S (Simple) architecture

- Input: two **stacked** images (**[image(t), image(t-1)]**)
- Encode: 9 Convolutional layers (strides: 2)
  - conv 7*7: 1 layers
  - conv 5*5: 2 layers
  - conv 3*3: 6 layers
- Decode: Refinement layers (described later)

FlowNetSimple



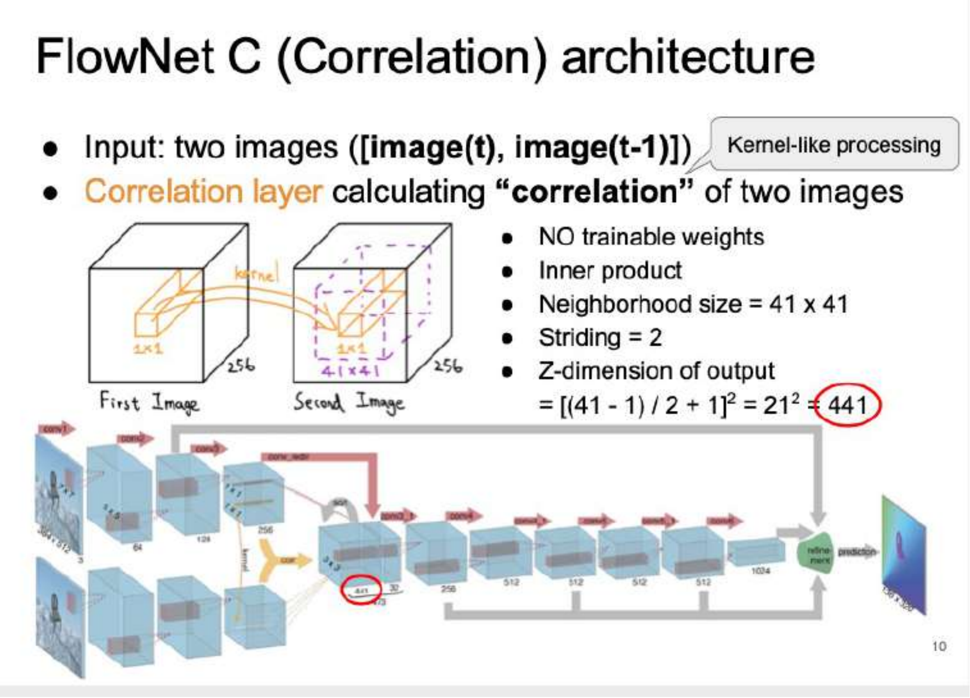
9

Slide credit: K-Inoue @ki42 & Oscar @wang

Dosovitskiy A, Fischer P, Ilg E, Hausser P, Hazirbas C, Golkov V, Van Der Smagt P, Cremers D, Brox T. Flownet: Learning optical flow with convolutional networks. InProceedings of the IEEE international conference on computer vision 2015 (pp. 2758-2766).

Optical Flow

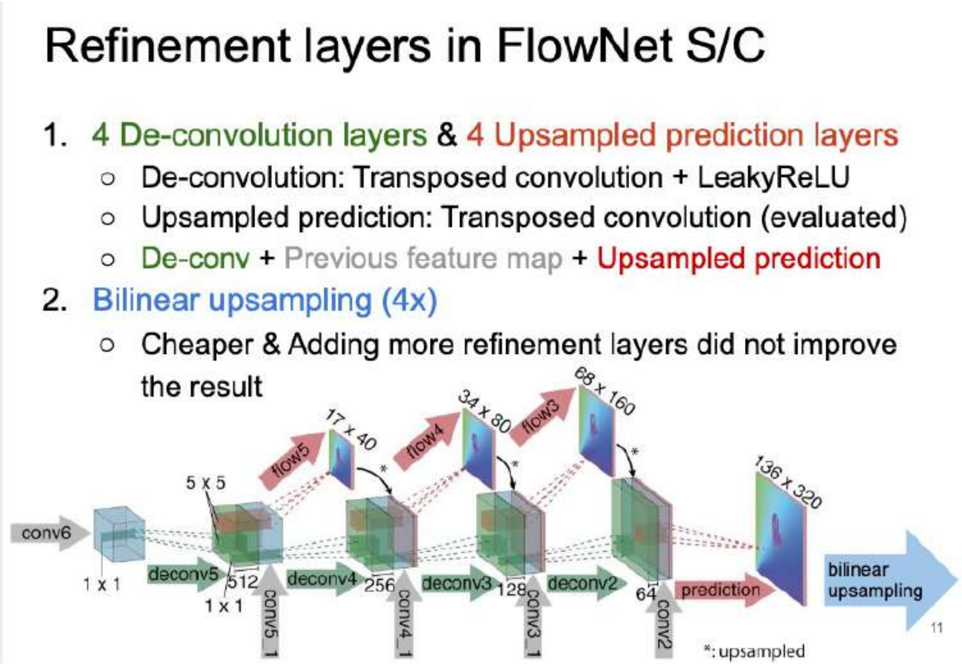
- Deep Learning: FlowNet



Slide credit: K-Inoue @ki42 &
Oscar @wang

Optical Flow

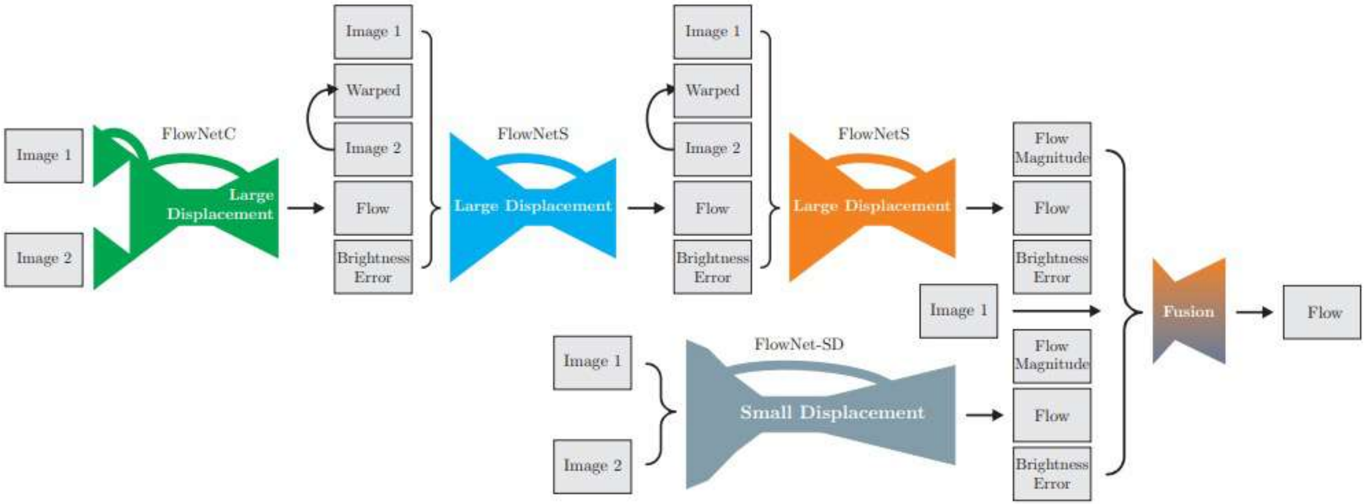
- Deep Learning: FlowNet



Slide credit: K-Inoue @ki42 & Oscar @wang

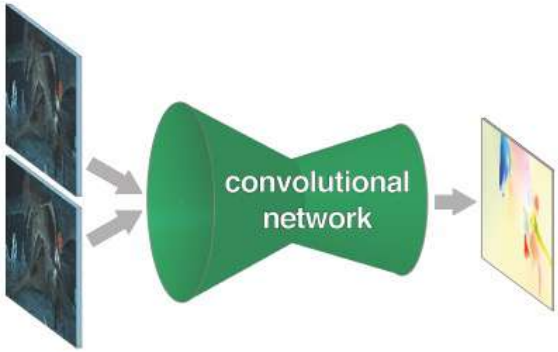
Optical Flow

- Deep Learning: FlowNet 2.0

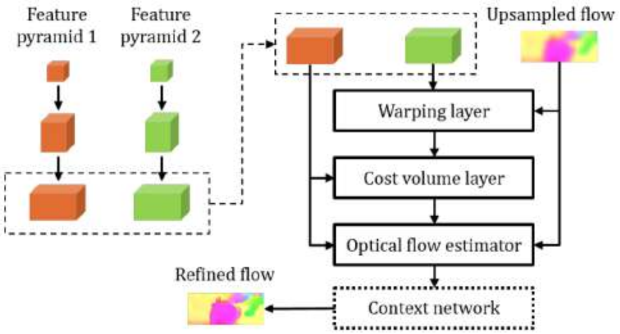


Ilg E, Mayer N, Saikia T, Keuper M, Dosovitskiy A, Brox T. Flownet 2.0: Evolution of optical flow estimation with deep networks. InProceedings of the IEEE conference on computer vision and pattern recognition 2017 (pp. 2462-2470).

Deep Learning and Optical Flow



FlowNet [Dosovitskiy et al. 2015]



PWC-Net [Sun et at al., 2018]

- Inductive bias: warping, cost volume
- Iterative refinement limited to pyramid levels

[Ranjan and Black, 2017]
[Ilg et al., 2017]
[Hui et al, 2018]

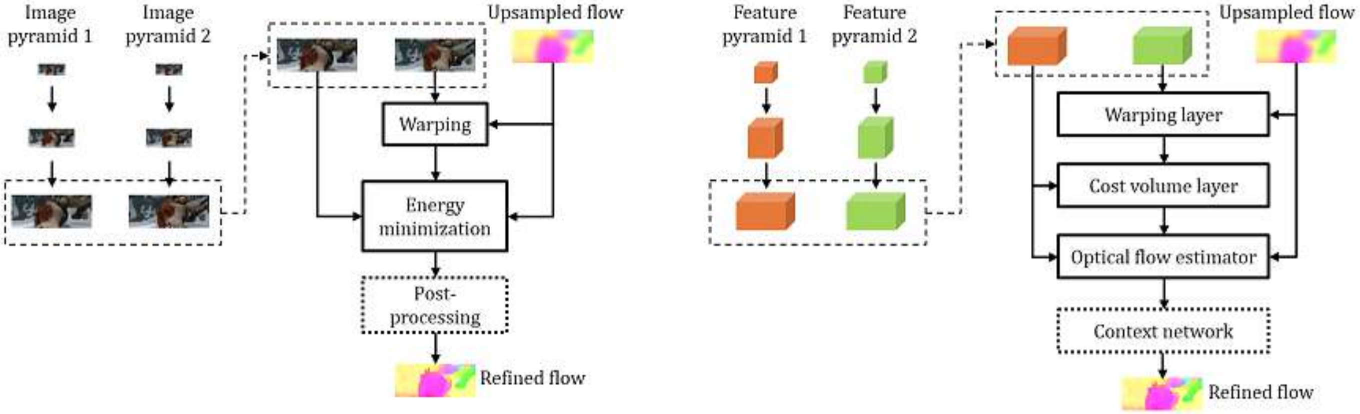
[Maurer and Bruhn, 2018]
[Ilg et al., 2017]
[Neoral et al, 2018]

[Bar-Haim and Wolf, 2020]
[Zhao et al, 2020]
[Z Yin et al, 2019]

[Yang and Ramanan, 2018]
[Lu et al, 2020]

Optical Flow

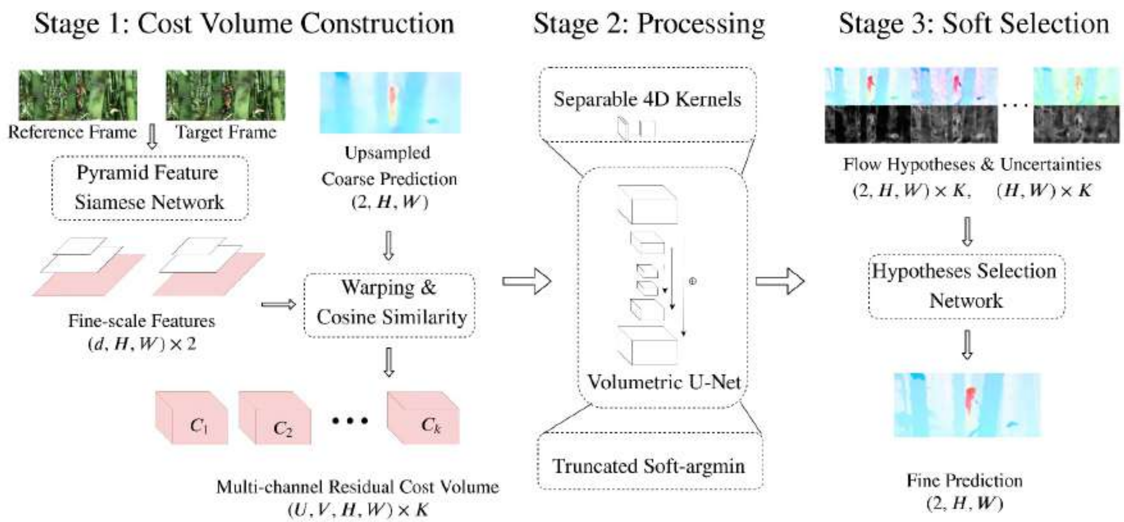
- Deep Learning: PWC-Net



Sun D, Yang X, Liu MY, Kautz J. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2018 (pp. 8934-8943).

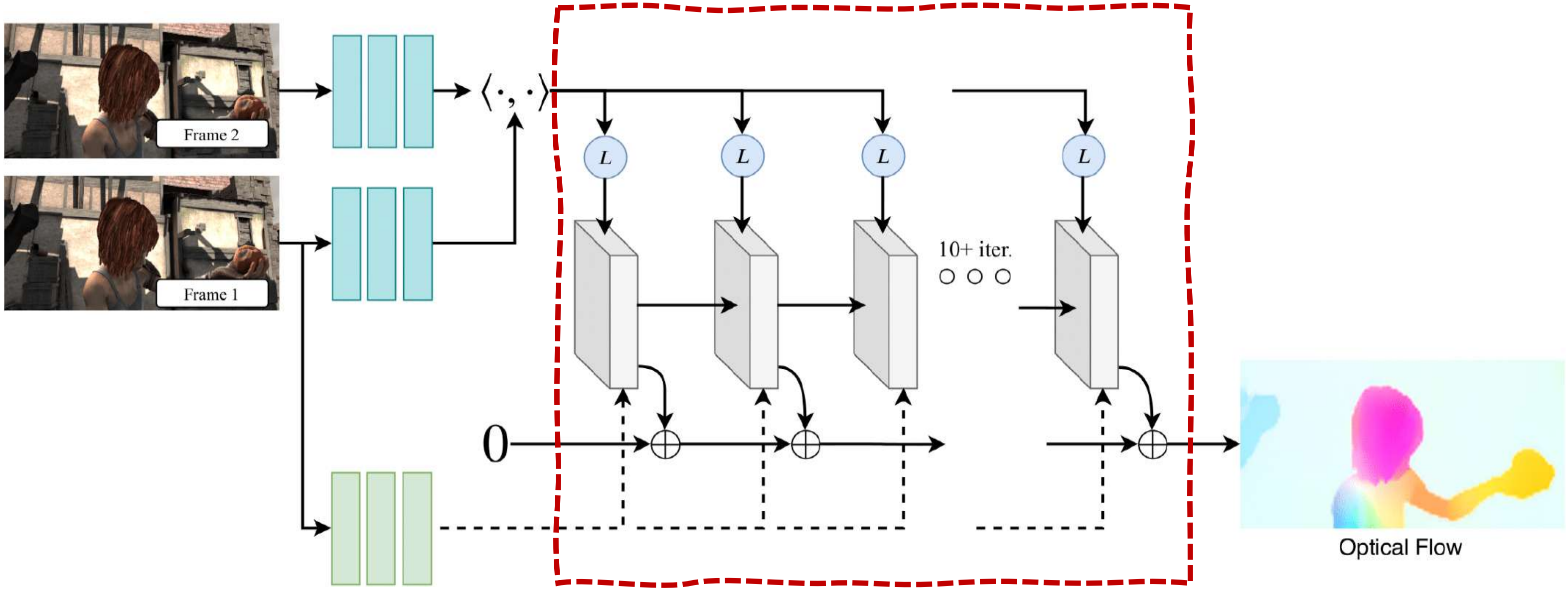
Optical Flow

- Deep Learning: VCN



Yang G, Ramanan D. Volumetric Correspondence Networks for Optical Flow. In Advances in Neural Information Processing Systems 2019 (pp. 793-803).

RAFT: Recurrent All-Pairs Field Transforms

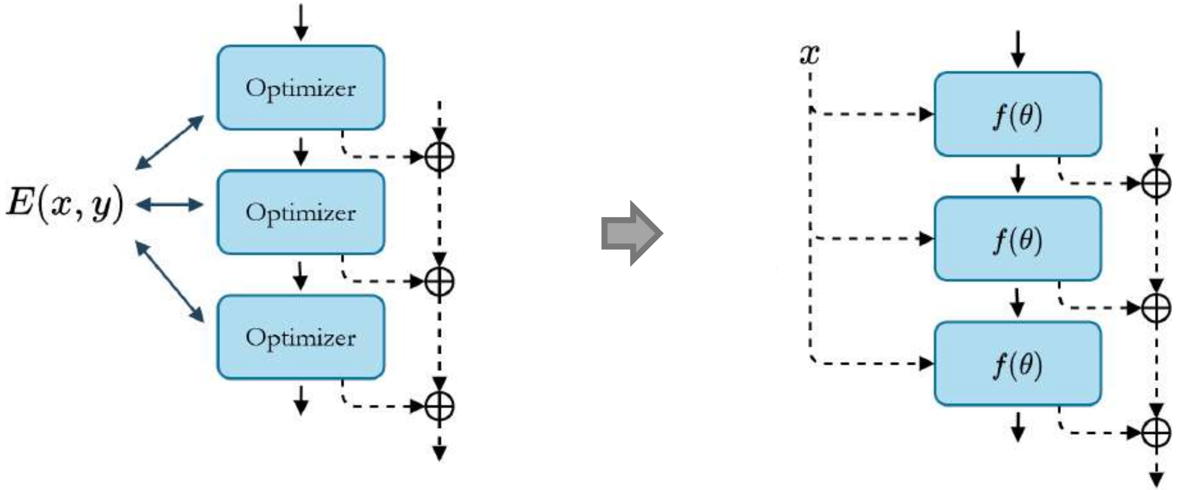


Iterative updates of a single high-res flow field

[Teed & Deng, ECCV 2020] Best Paper Award

Strategy: Optimization-Inspired Neural Architectures

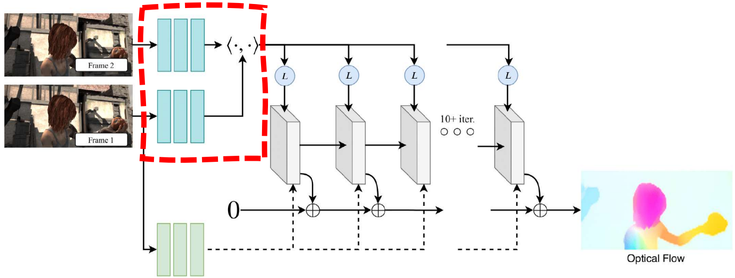
Design neural networks to behave like classical optimization algorithms



+ Recurrent iterative updates

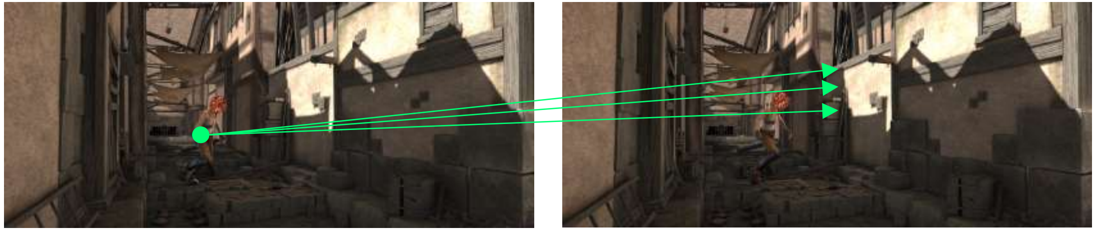
RAFT: Recurrent All-Pairs Field Transforms

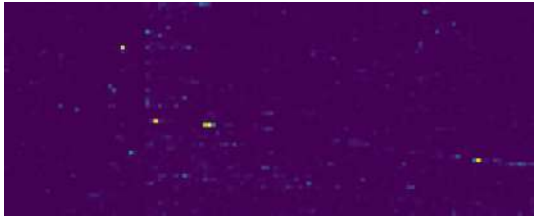
- ***State-of-the-art accuracy***: 16% better on KITTI, 30% better on Sintel
- ***High efficiency***: 10x faster training, 10fps on 436x1088 video
- ***Strong Generalization***: 40% better synthetic to real generalization



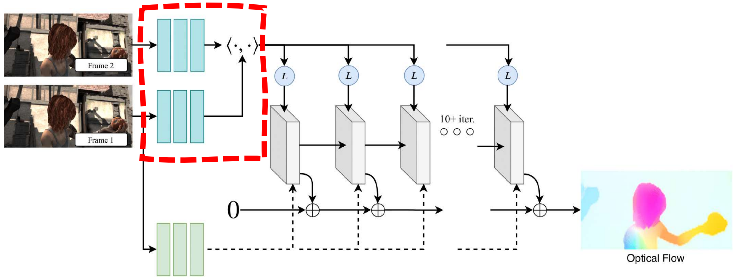
All-Pairs Visual Similarities

- Dot product between all pairs

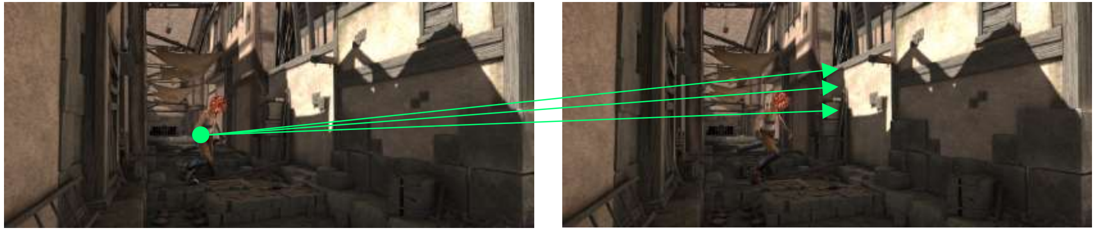




All-Pairs Visual Similarities

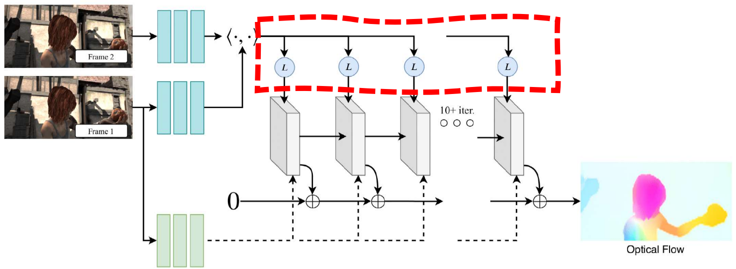


- Dot product between all pairs
- Repeated pooling of last two dimensions

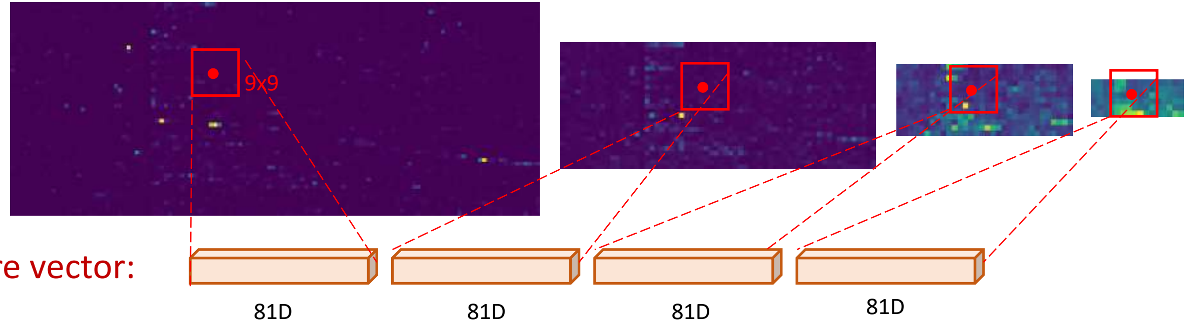


All-Pairs Visual Similarities

- Dot product between all pairs
- Repeated pooling of last two dimensions
- Use current flow estimate to retrieve a feature vector



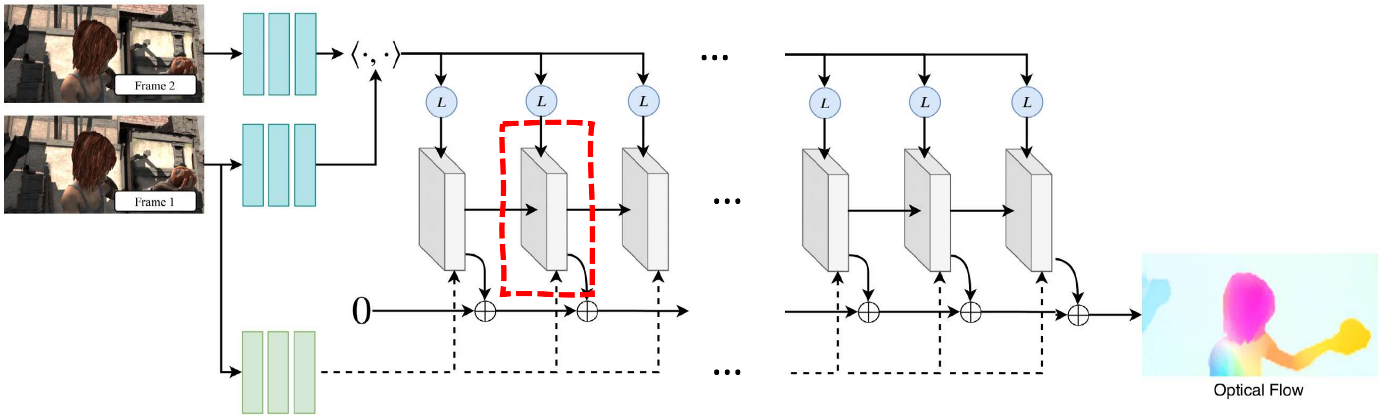
retrieved feature vector:



cues on how good the current flow is and where are better similarities

Update Operator

- GRU-Based recurrent update operator

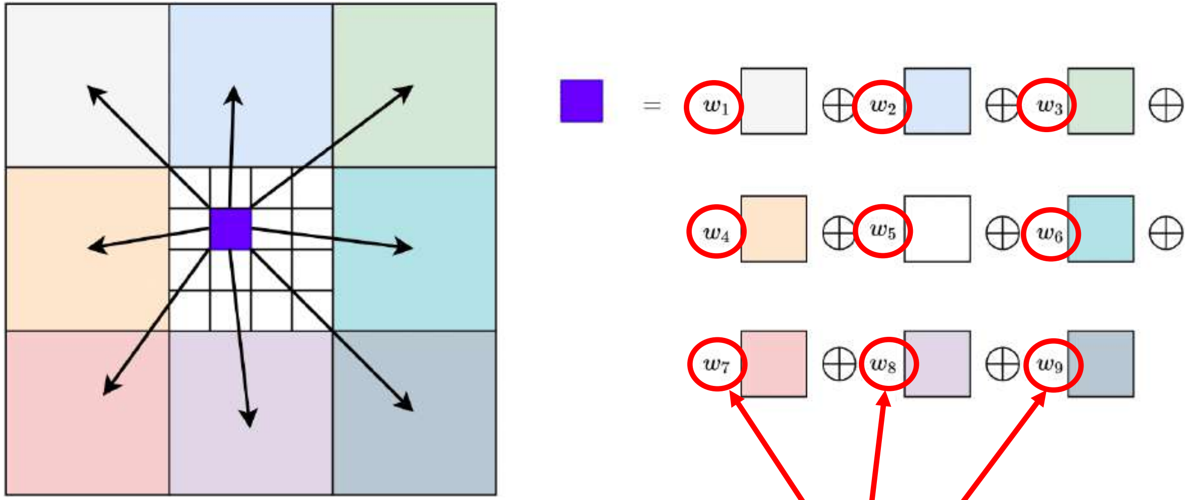


- Designed to mimic updates of first order optimization algorithm [1]
- But no explicit objective or gradient

[1] Adler, Jonas, and Ozan Öktem. "Learned primal-dual reconstruction."2018

Convex Upsampling

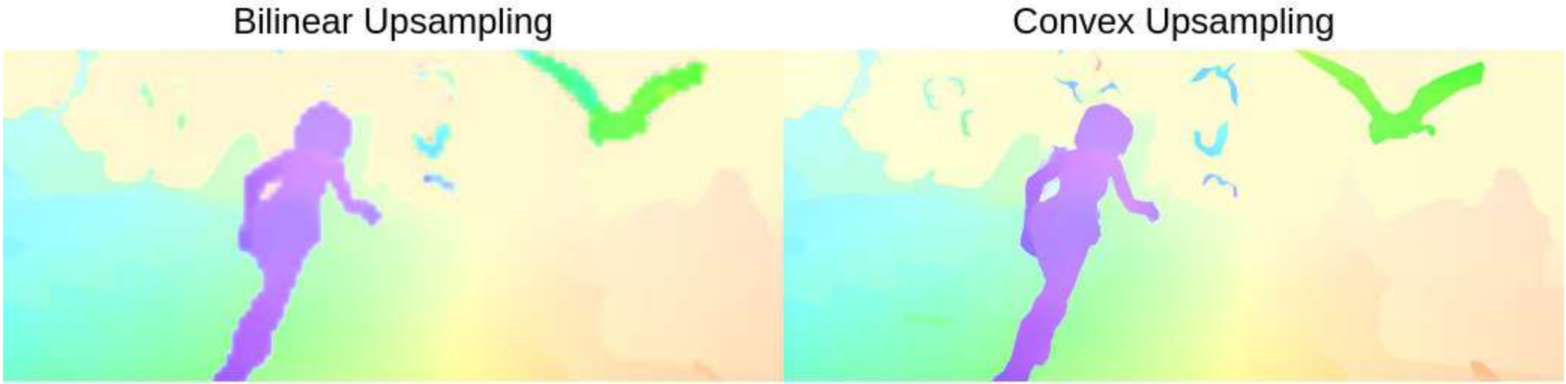
- Upsamples flow to **full resolution**
- Convex combination of 3x3 coarse resolution neighbors



Coefficients Predicted by Network (w1, …, w9)

Convex Upsampling

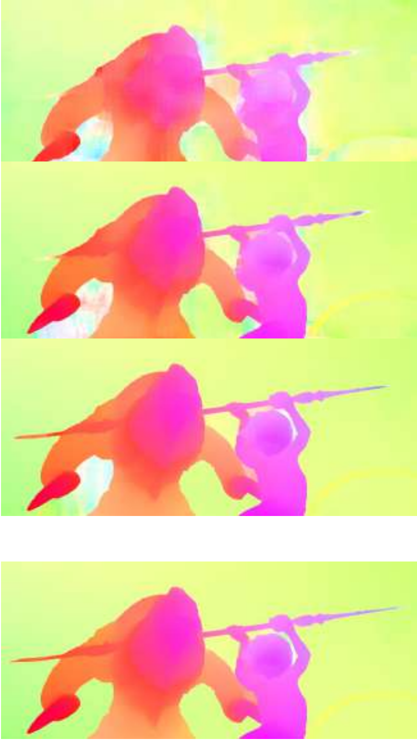
- Upsamples flow to **full resolution**
- Convex combination of 3x3 coarse resolution neighbors



Training

- Supervised directly on sequence of full resolution flow fields

$$Loss = \sum_{i}^{N} \frac{1.25^{i}}{1.25^{N}} \left|\left| f_{gt} - f_i \right|\right|_1$$



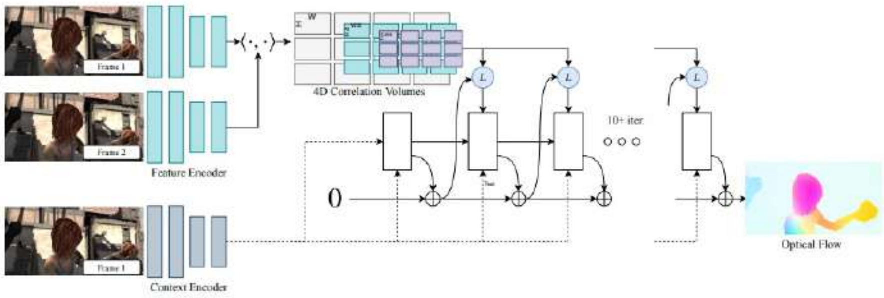
f_1

f_2

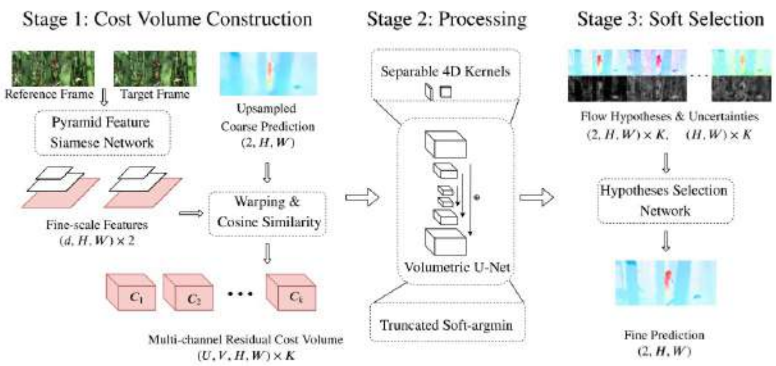
f_3

f_{12}

RAFT versus VCN



RAFT [Teed & Deng, 2020]

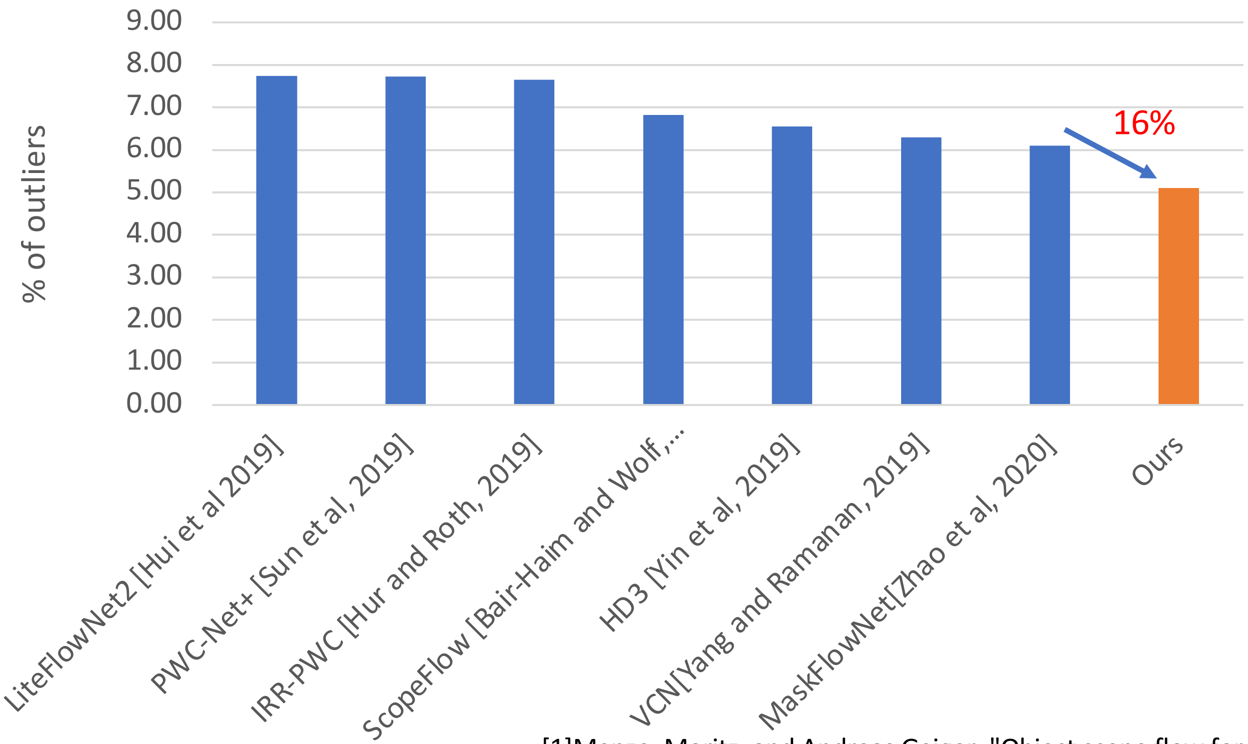


VCN [Yang & Ramanan, 2019]

- Construct 4D cost volume
- 2D convolution on slices of cost volume

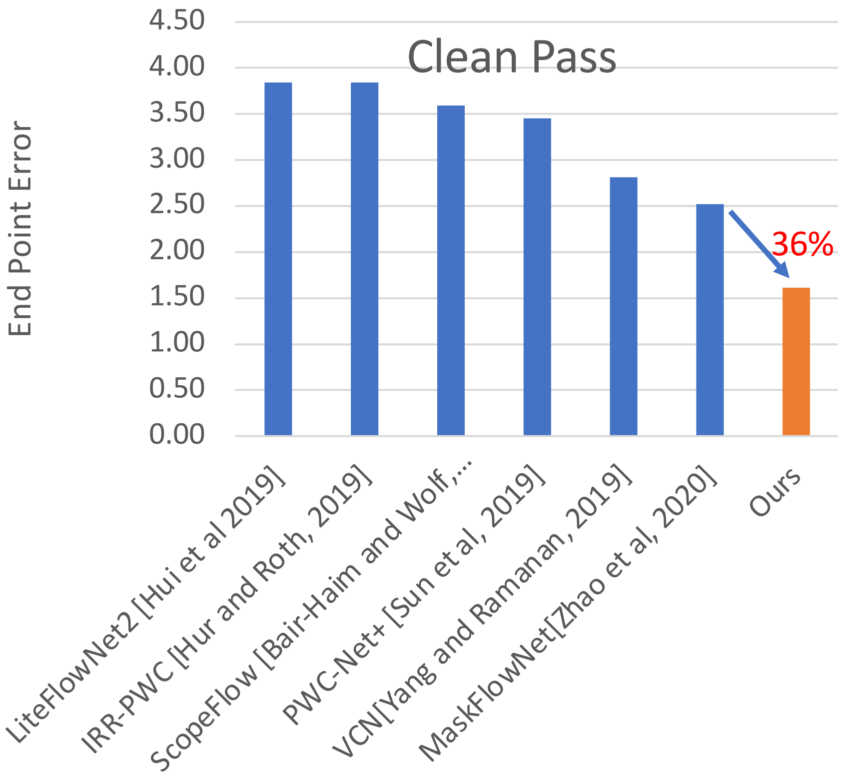
- Construct 4D cost volume
- 4D convolution on entire cost volume

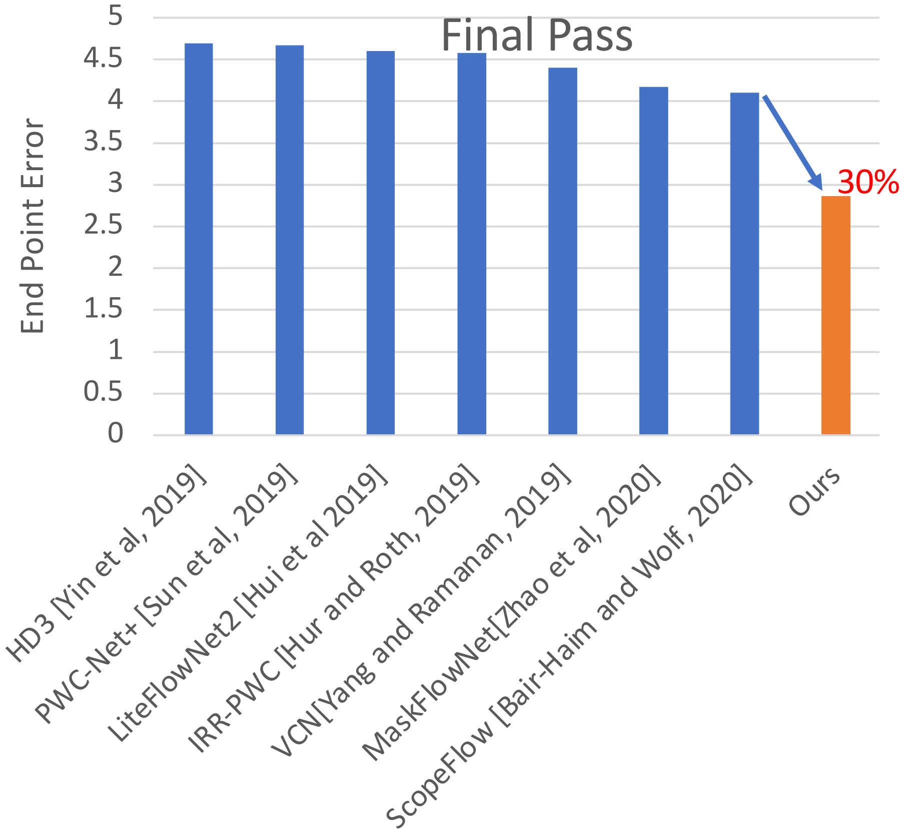
KITTI-2015[1] Results



[1]Menze, Moritz, and Andreas Geiger. "Object scene flow for autonomous vehicles" 2015

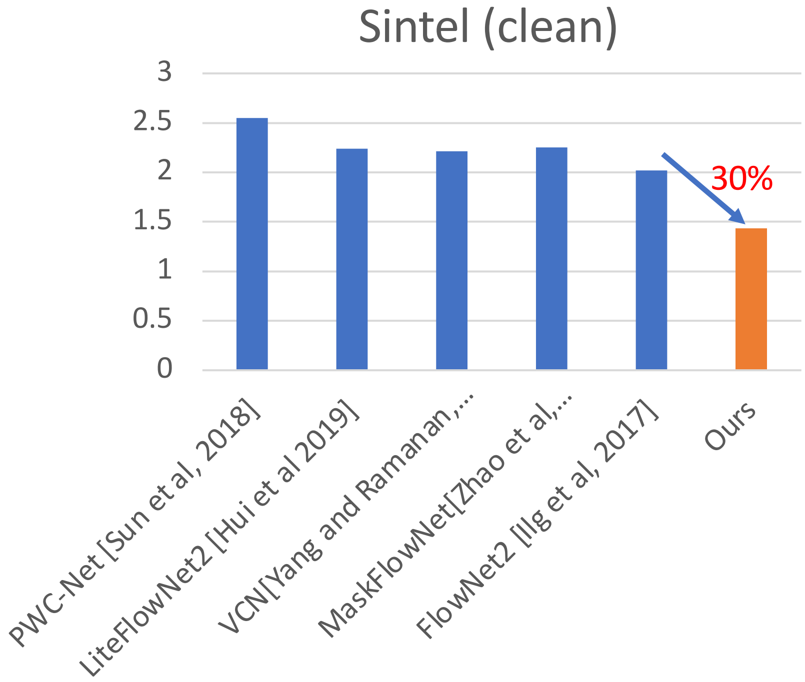
Sintel Results

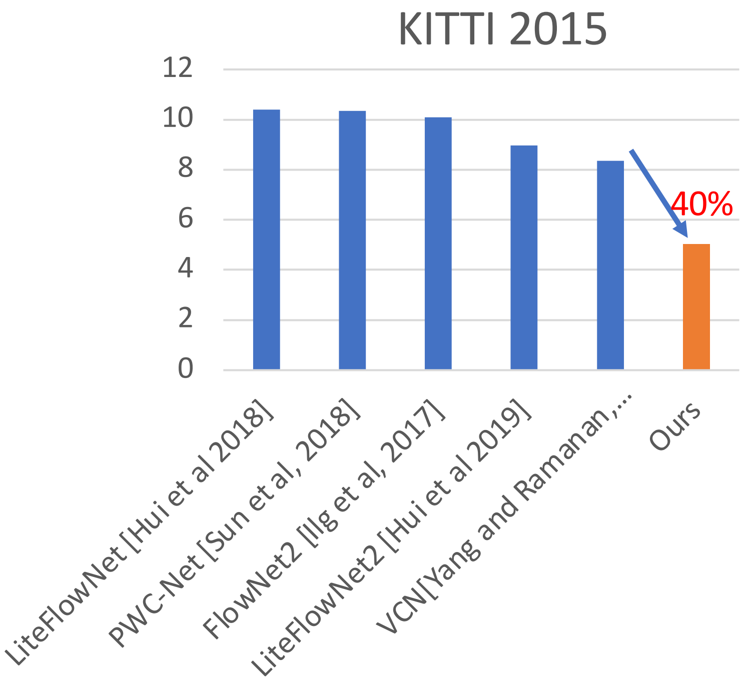




Butler, Daniel J., et al. "A naturalistic open source movie for optical flow evaluation." *ECCV* 2012.

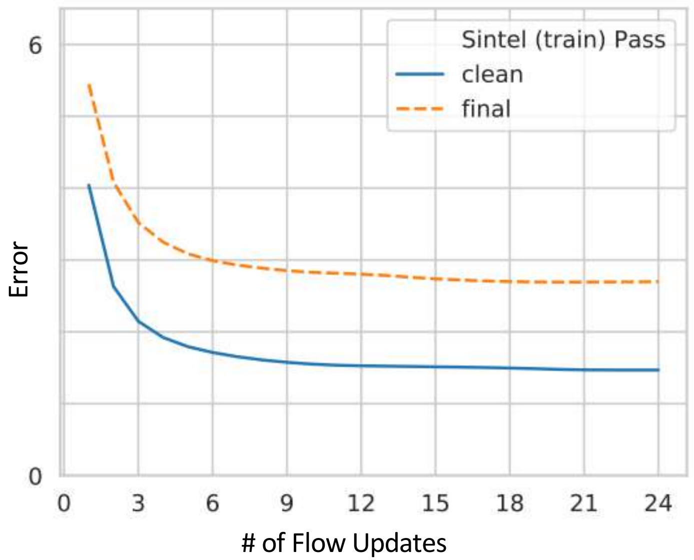
Cross-Dataset Generalization

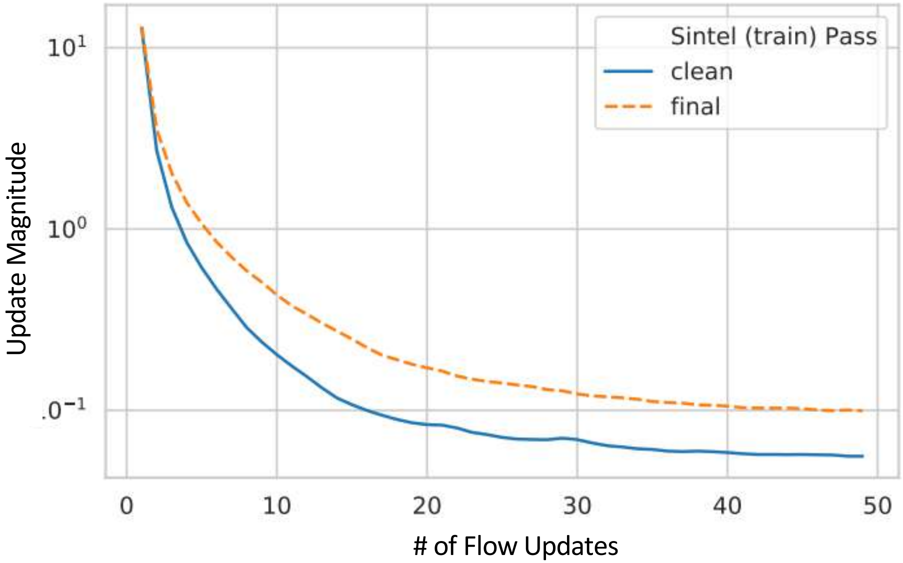




Models trained on **FlyingChairs** (Fischer et al. 2015) and **FlyingThings3D** (Mayer et al, 2016)

Convergence





Convergence Visualized

1 Iteration 2 Iterations 5 Iterations 32 Iterations

RAFT can recover the motion of small, fast moving objects

Ground Truth

VCN [Yang & Ramanan, 2019]

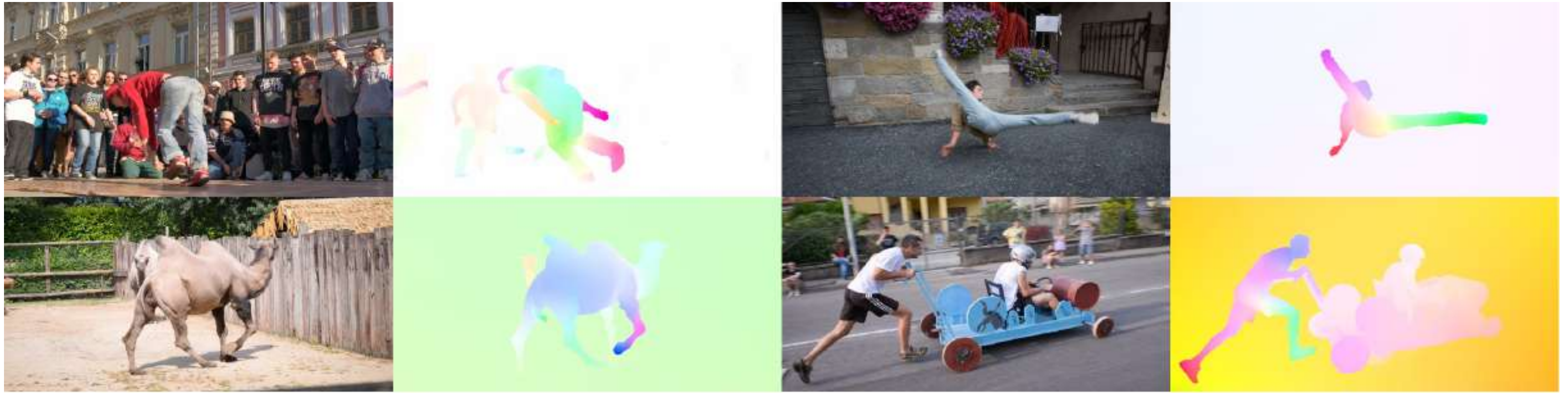
IRR-PWC [Hur & Roth, 2019]

Ours

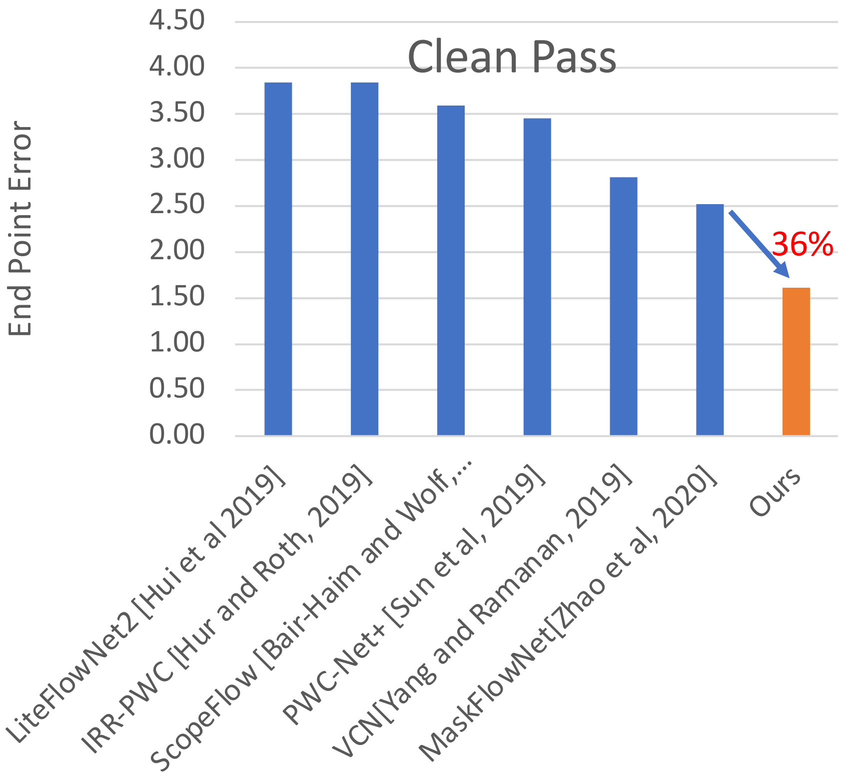
http://sintel.is.tue.mpg.de/

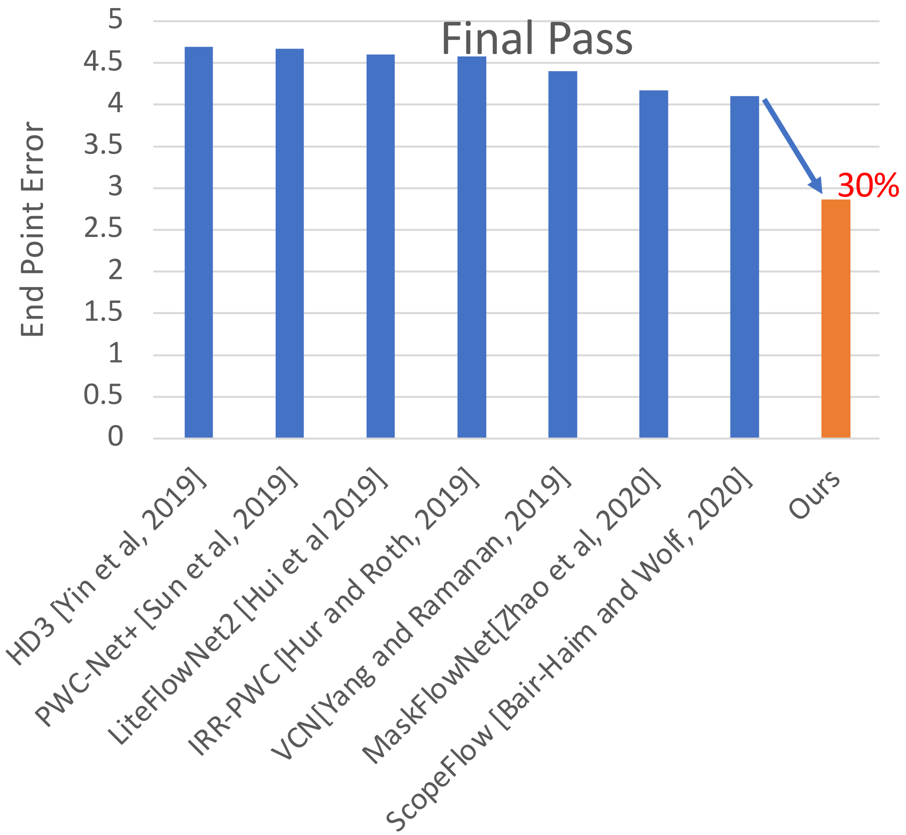
KITTI-2015: http://www.cvlibs.net/datasets/kitti/index.php

DAVIS (1080p) https://davischallenge.org/



Sintel Results





Butler, Daniel J., et al. "A naturalistic open source movie for optical flow evaluation." *ECCV* 2012.

Robust Vision Challenge ECCV 2020

	Method	Middlebury (Detailed subrankings)	KITTI (Detailed subrankings)	MPI Sintel (Detailed subrankings)	VIPER (Detailed subrankings)	
1	RAFT-TF_RVC	1	2	1	1	Submitted by Deqing Sun (Google)
2	PRAFlow_RVC	2	1	2	3	Submitted by Zhexiong Wan (Northwestern Polytechnical University, Xi'an, China)
3	C-RAFT_RVC	5	3	3	4	Submitted by Henrique Morimitsu (Tsinghua University)
4	VCN_RVC	3	6	5	5	Volumetric Correspondence Networks for Optical Flow, NeurIPS 2019. [Project page] - Submitted by Gengshan Yang (CMU)
5	IRR-PWC_RVC	7	5	7	2	Iterative Residual Refinement for Joint Optical Flow and Occlusion Estimation [Project page] - Submitted by Junhwa Hur (TU Darmstadt)
5	LSM_FLOW_RVC	6	4	4	7	LSM: Learning Subspace Minimization for Low-Level Vision [Project page] - Submitted by Chengzhou Tang (Simon Fraser University)
7	PWC-Net_RVC	4	7	6	6	PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume, CVPR 2018. [Project page] - Submitted by Deqing Sun (Google)
8	TVL1_RVC	8	8	8	8	Baseline - Submitted by Toby Weed (Middlebury College)
9	H+S_RVC	9	9	9	9	Baseline - Submitted by Toby Weed (Middlebury College)

All top 3 submissions used RAFT

Winner

A TensorFlow Implementation of RAFT

Deqing Sun, Charles Herrmann, Varun Jampani, Mike Krainin, Forrester Cole, Austin Stone, Rico Jonschkowski, Ramin Zabih, William Freeman, and Ce Liu

Google Research

Google

Stereo

Many slides adapted from Steve Seitz and Svetlana Lazebnik

Binocular stereo

- Given a calibrated binocular stereo pair, fuse it to produce a depth image

image 1

image 2

Dense depth map

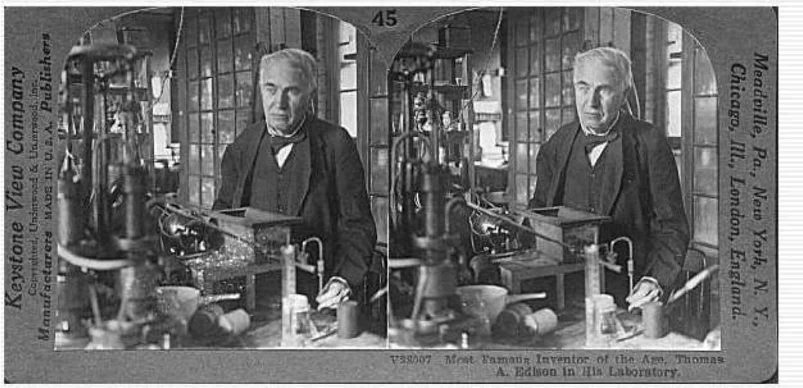
Binocular stereo

- Given a calibrated binocular stereo pair, fuse it to produce a depth image

Where does the depth information come from?

Binocular stereo

- Given a calibrated binocular stereo pair, fuse it to produce a depth image
  - Humans can do it



Stereograms: Invented by Sir Charles Wheatstone, 1838

Binocular stereo

- Given a calibrated binocular stereo pair, fuse it to produce a depth image
    - Humans can do it

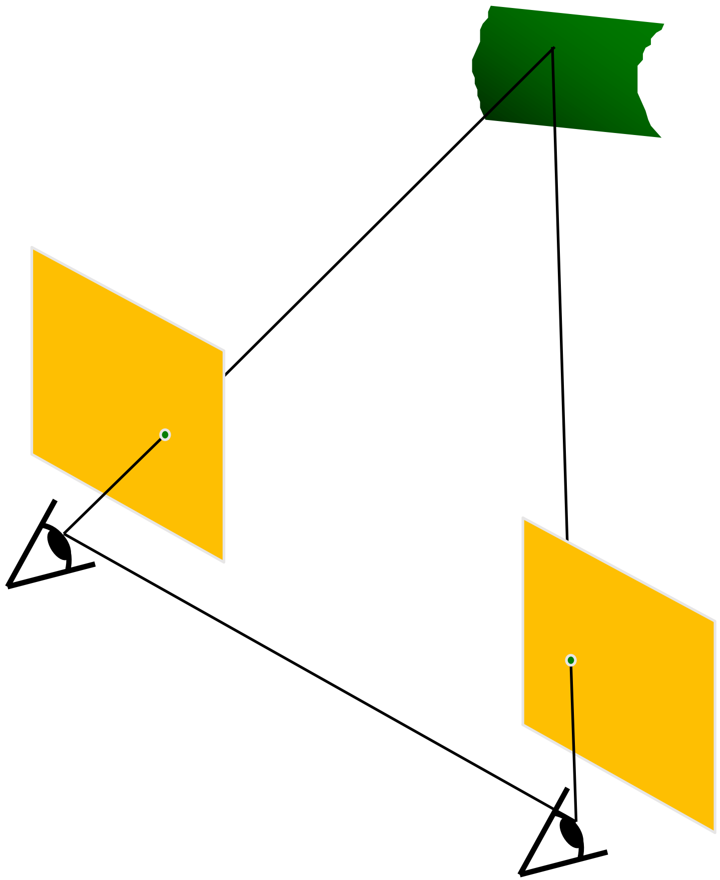
Autostereograms: www.magiceye.com

Binocular stereo

- Given a calibrated binocular stereo pair, fuse it to produce a depth image
  - Humans can do it

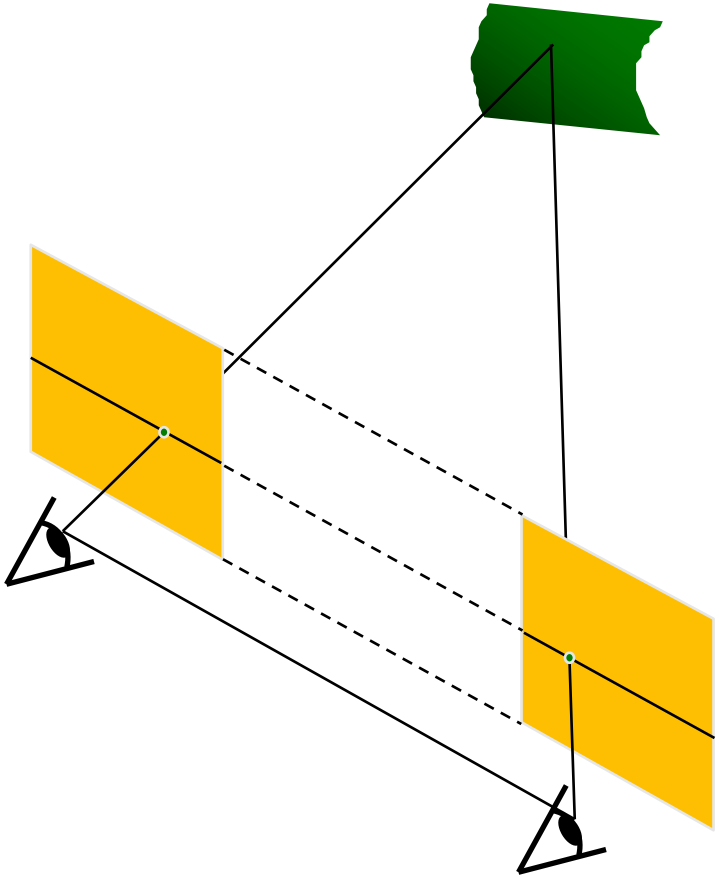
Autostereograms: www.magiceye.com

Simplest Case: Parallel images



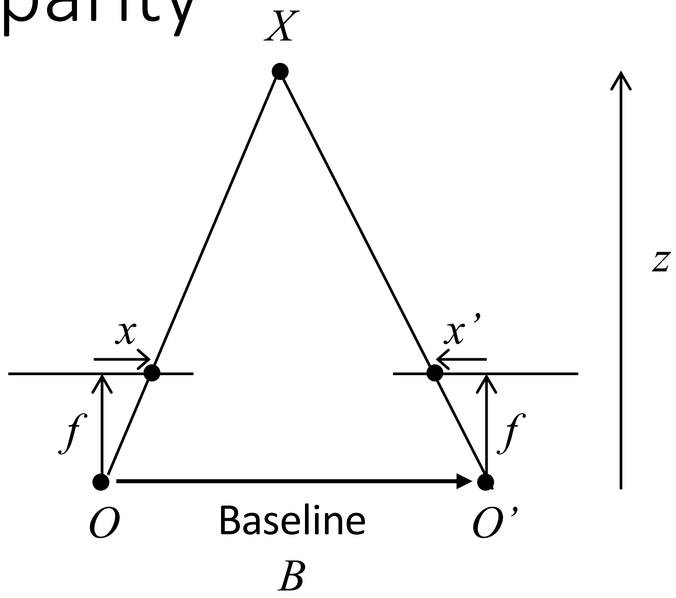
- Image planes of cameras are parallel to each other and to the baseline
- Camera centers are at same height
- Focal lengths are the same

Simplest Case: Parallel images

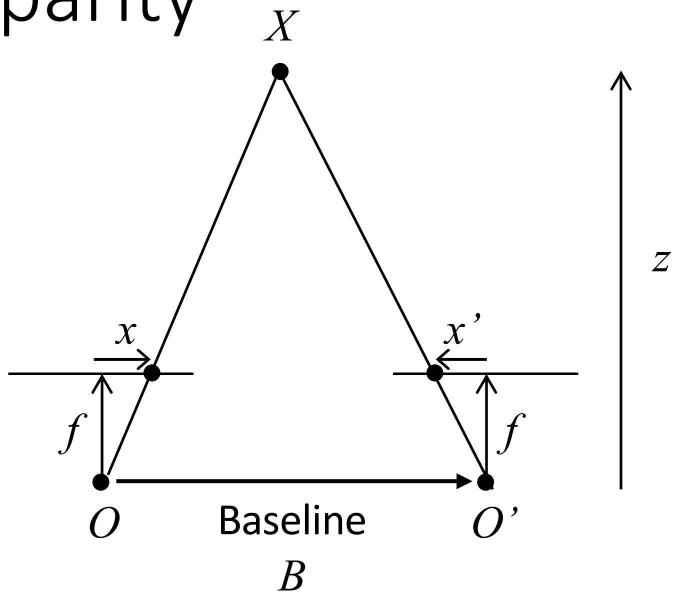


- Image planes of cameras are parallel to each other and to the baseline
- Camera centers are at same height
- Focal lengths are the same
- Then epipolar lines fall along the horizontal scan lines of the images

Depth from disparity

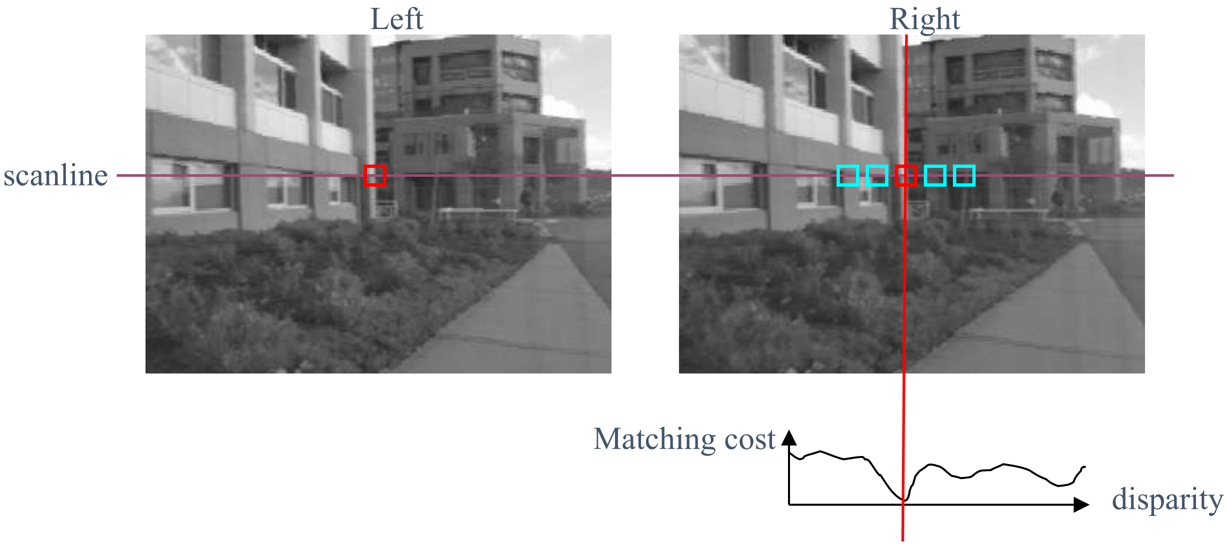


Depth from disparity



$$disparity = x - x' = \frac{B \cdot f}{z}$$

Correspondence search

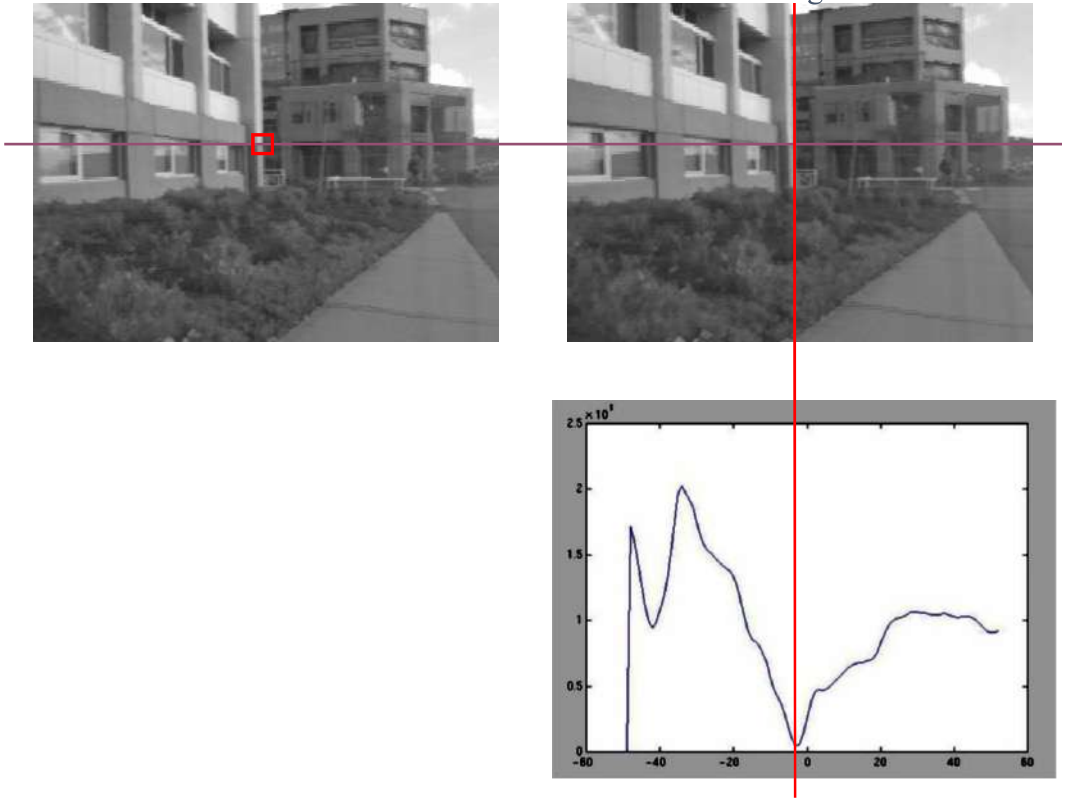


- Slide a window along the right scanline and compare contents of that window with the reference window in the left image
- Matching cost: SSD or normalized correlation

Correspondence search

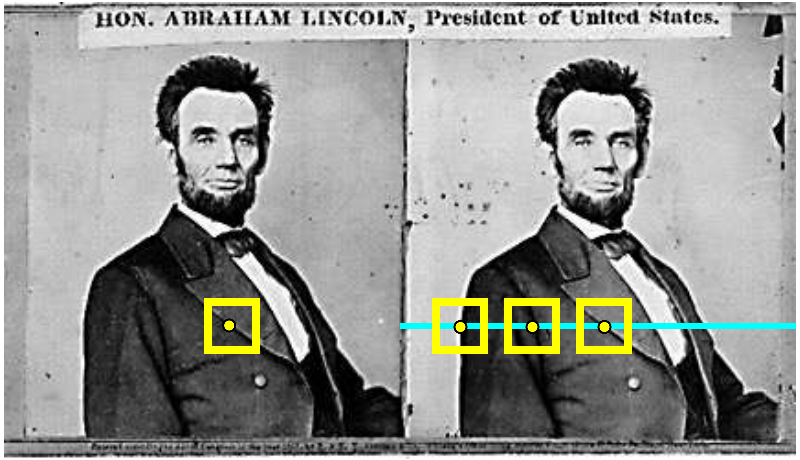
Left Right

scanline



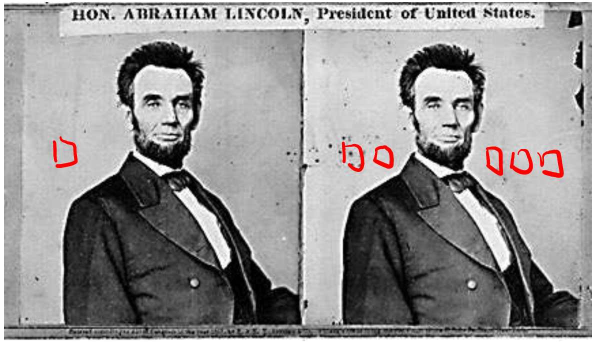
SSD

Basic stereo algorithm



- For each pixel *x* in the first image
  - Find corresponding epipolar scanline in the right image
  - Examine all pixels on the scanline and pick the best match *x'*
  - Compute disparity $x-x'$ and set $\text{depth}(x) = B*f/(x-x')$

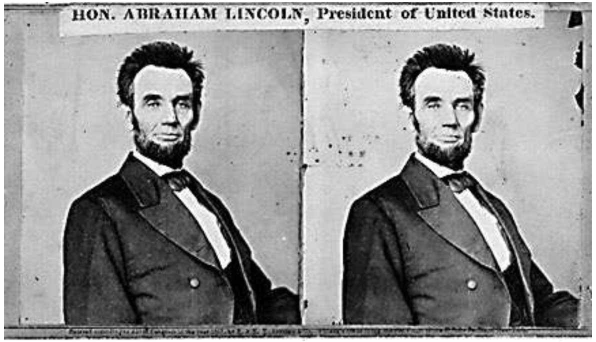
Failures of correspondence search



Textureless surfaces

Occlusions, repetition

Failures of correspondence search



Textureless surfaces

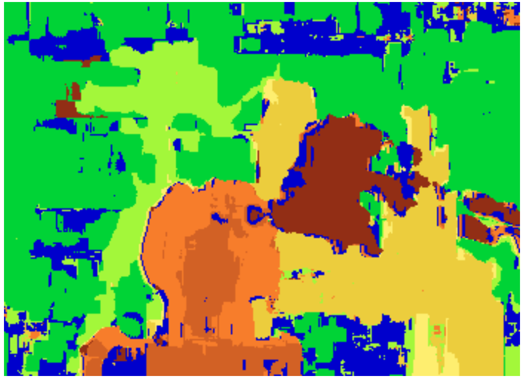
Occlusions, repetition

Non-Lambertian surfaces, specularities

Results with window search

Data

Window-based matching



Ground truth

Better methods exist...

Graph cuts

Ground truth

Y. Boykov, O. Veksler, and R. Zabih, Fast Approximate Energy Minimization via Graph Cuts, PAMI 2001

For the latest and greatest: http://www.middlebury.edu/stereo/

How can we improve window-based matching?

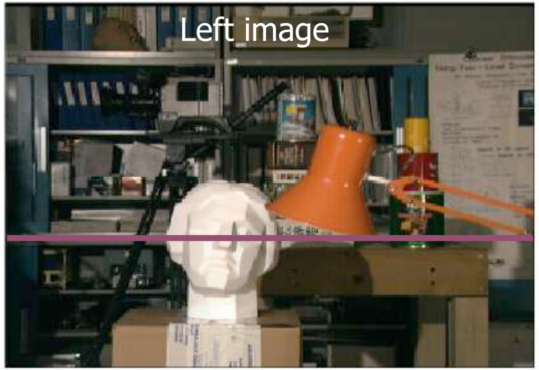
- The similarity constraint is **local** (each reference window is matched independently)
- Need to enforce **non-local** correspondence constraints

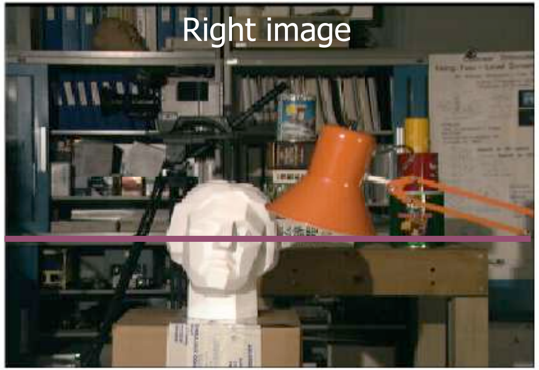
Non-local constraints

- Uniqueness
  - For any point in one image, there should be at most one matching point in the other image
- Ordering
  - Corresponding points should be in the same order in both views
- Smoothness
  - We expect disparity values to change slowly (for the most part)

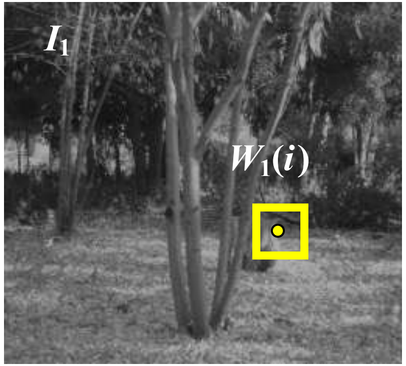
Scanline stereo

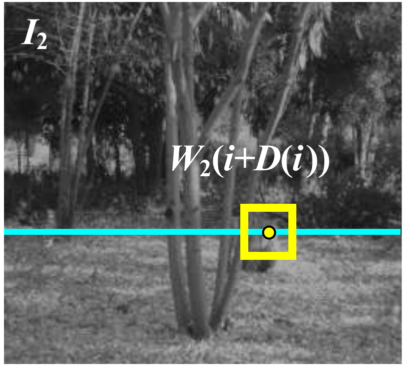
- Try to coherently match pixels on the entire scanline
- Different scanlines are still optimized independently

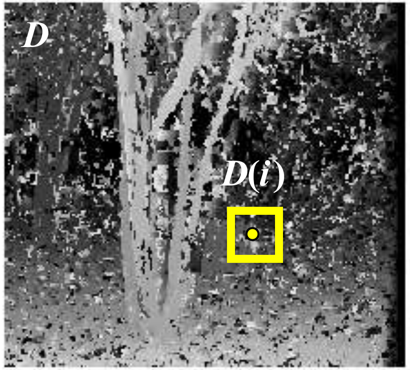




Stereo matching as energy minimization

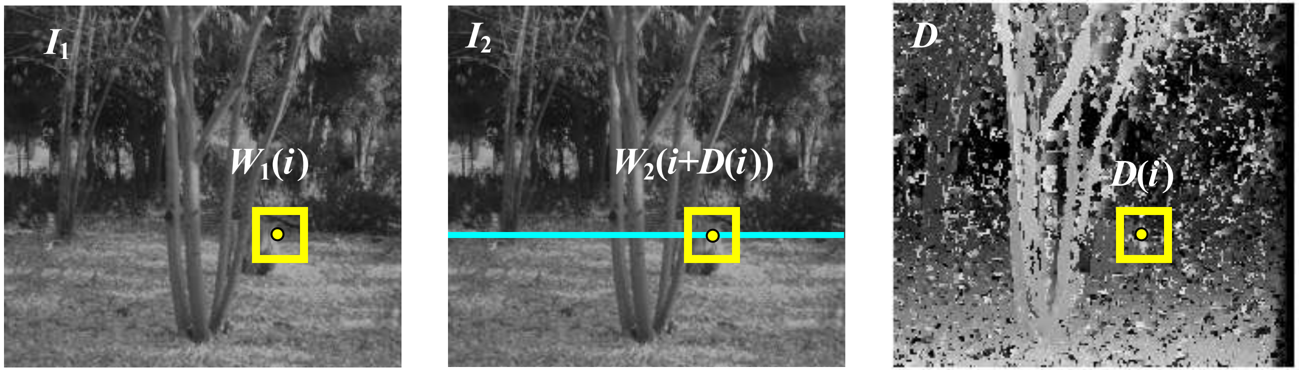






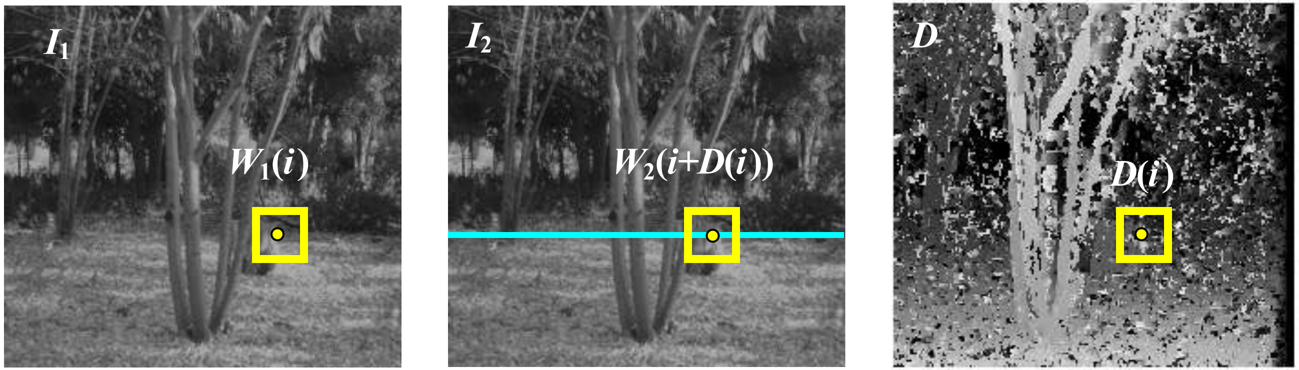
$$E(D) = \sum_i \left(W_1(i) - W_2(i + D(i))\right)^2 + \lambda \sum_{\text{neighbors } i,j} \rho\left(D(i) - D(j)\right)$$

Stereo matching as energy minimization



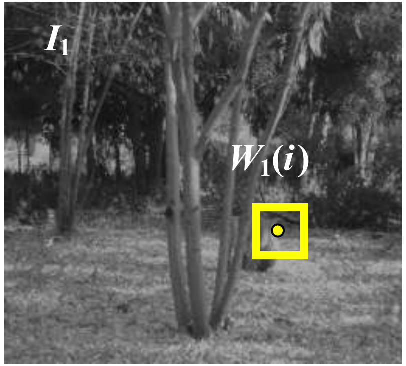
$$E(D) = \underbrace{\sum_i \left(W_1(i) - W_2(i + D(i))\right)^2}_{\textit{data term}} + \lambda \underbrace{\sum_{\text{neighbors } i,j} \rho\left(D(i) - D(j)\right)}_{\textit{smoothness term}}$$

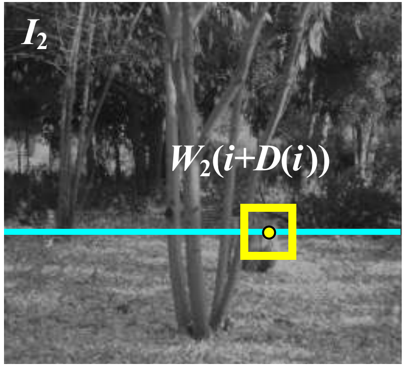
Stereo matching as energy minimization

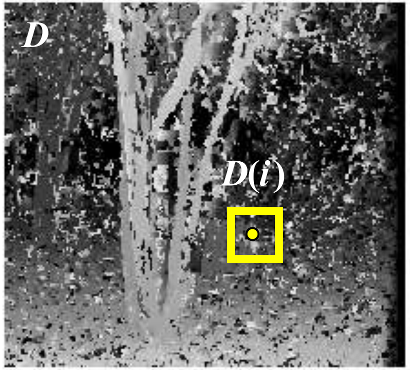


$$E(D) = \underbrace{\sum_i \left(W_1(i) - W_2(i + D(i))\right)^2}_{\textit{data term}} + \lambda \underbrace{\sum_{\text{neighbors } i,j} \rho\left(D(i) - D(j)\right)}_{\textit{smoothness term}}$$

Stereo matching as energy minimization







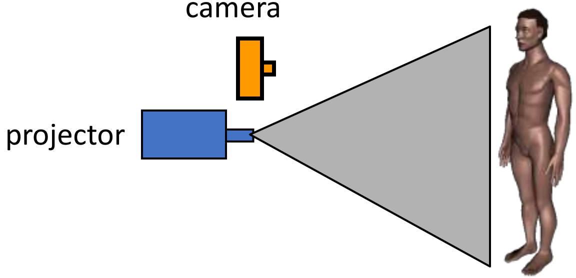
$$E(D) = \underbrace{\sum_i \left(W_1(i) - W_2(i + D(i))\right)^2}_{\textit{data term}} + \lambda \underbrace{\sum_{\text{neighbors } i,j} \rho\left(D(i) - D(j)\right)}_{\textit{smoothness term}}$$

- Energy functions of this form can be minimized using *graph cuts*

Y. Boykov, O. Veksler, and R. Zabih, [Fast Approximate Energy Minimization via Graph Cuts](), PAMI 2001

Active stereo with structured light

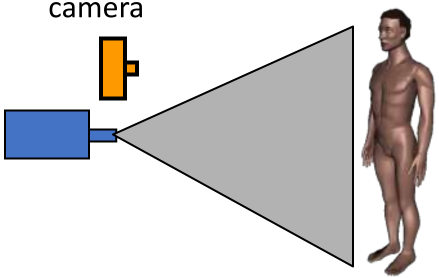
- Project "structured" light patterns onto the object
  - Simplifies the correspondence problem
  - Allows us to use only one camera



L. Zhang, B. Curless, and S. M. Seitz. Rapid Shape Acquisition Using Color Structured Light and Multi-pass Dynamic Programming. *3DPVT* 2002

Active stereo with structured light

- Project "structured" light patterns onto the object
  - Simplifies the correspondence problem
  - Allows us to use only one camera

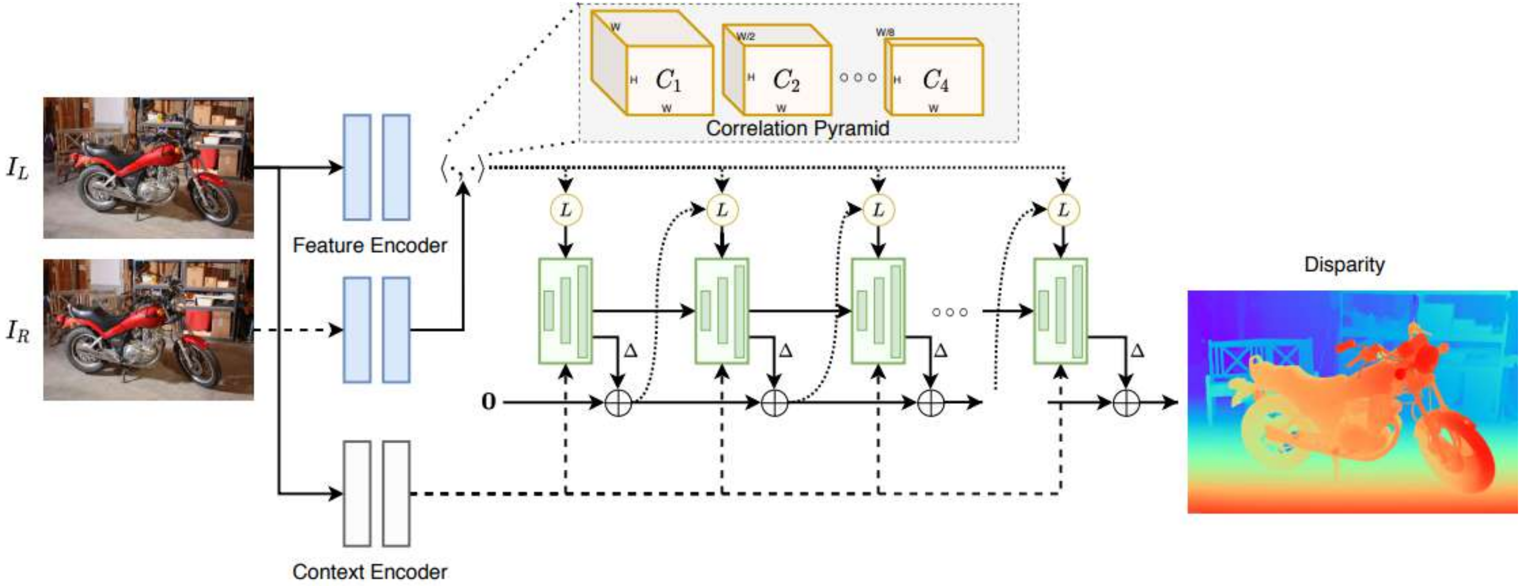


L. Zhang, B. Curless, and S. M. Seitz. Rapid Shape Acquisition Using Color Structured Light and Multi-pass Dynamic Programming. *3DPVT* 2002

Kinect: Structured infrared light

http://bbzippo.wordpress.com/2010/11/28/kinect-in-infrared/

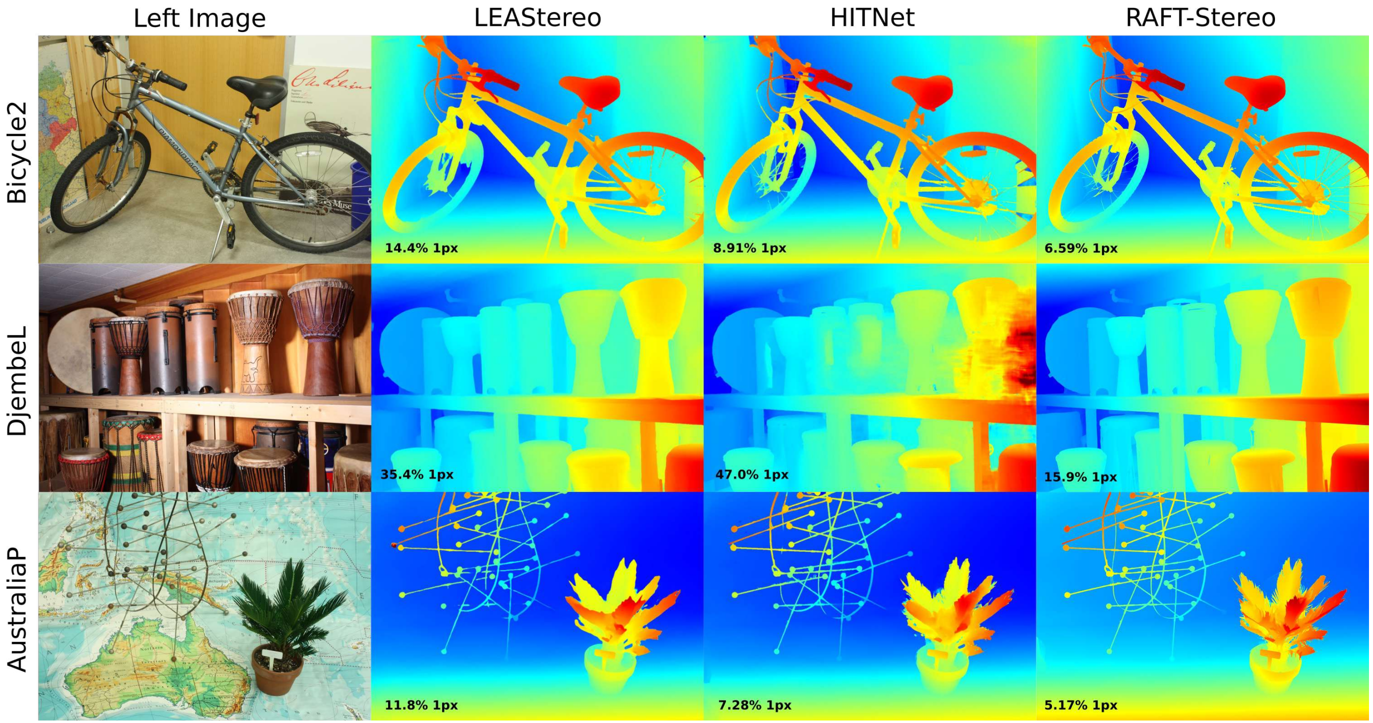
RAFT-Stereo: RAFT for rectified two-view stereo

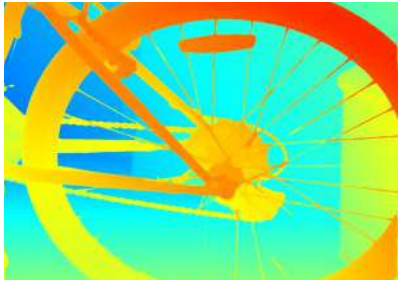


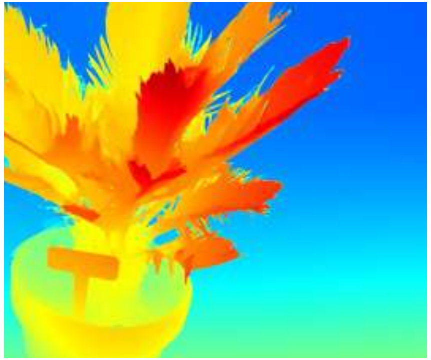
[Teed, Lipson, Deng, 2020]

RAFT-Stereo: 1st on Middlebury

[Scharstein et al, 2014]

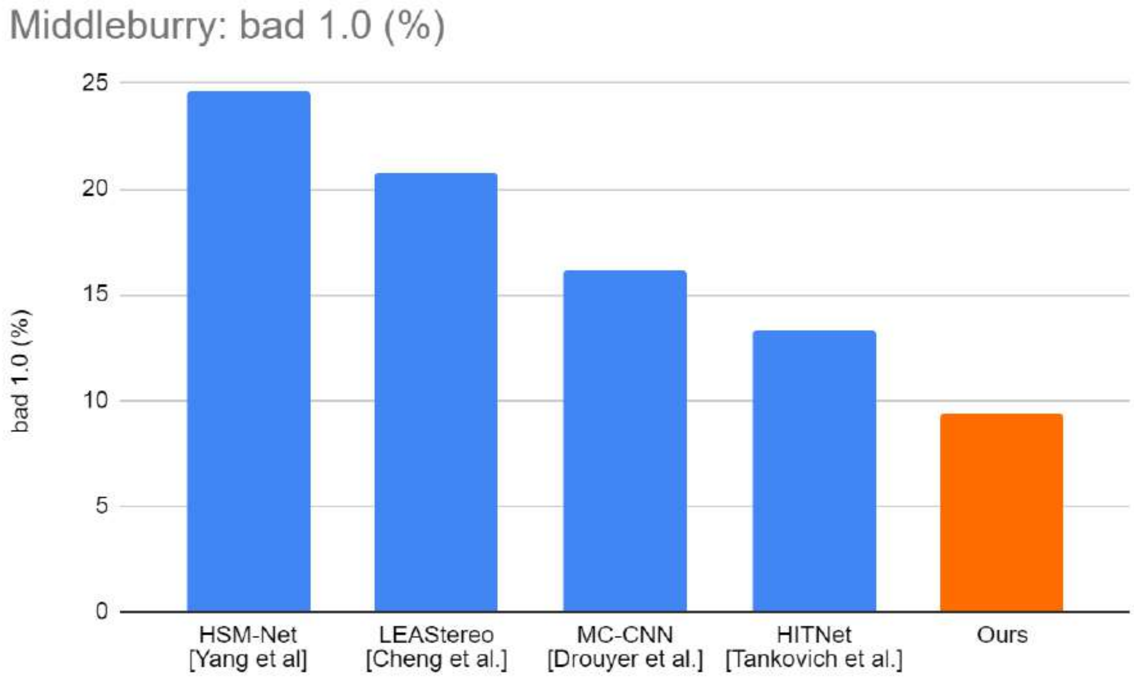






[Lipson, Teed, Deng, 3DV 2021] **Best Student Paper Award**

Middlebury Stereo Benchmark

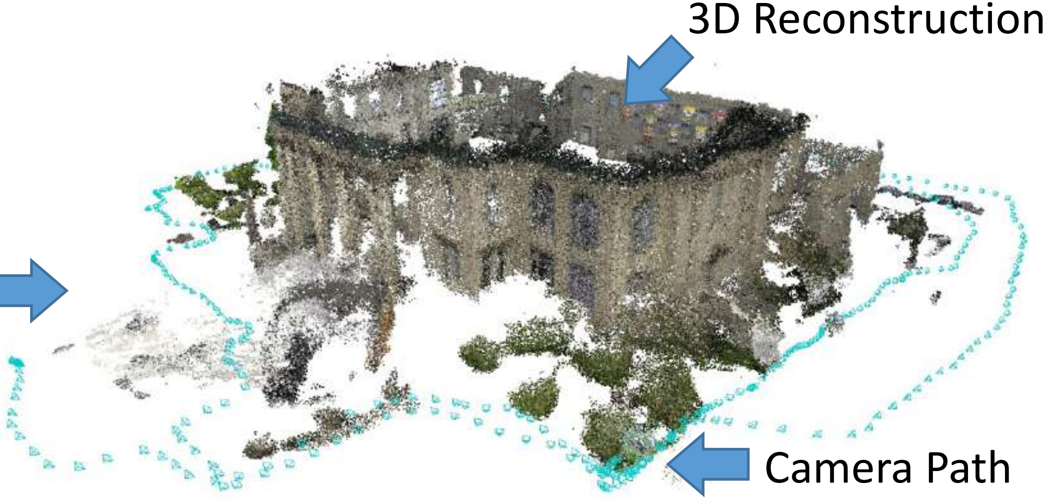


[Scharstein et al, 2014]


DIAMONDBACK

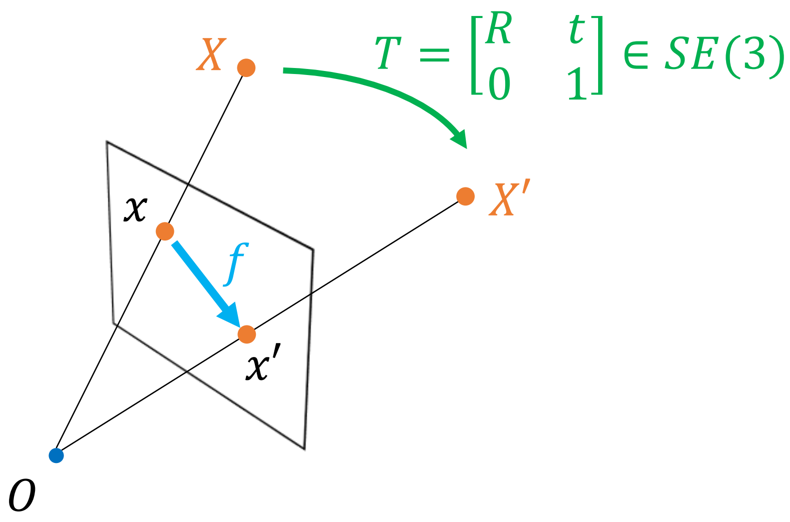
Visual SLAM:
Simultaneous Localization and Mapping

- Input: video of (largely) static scene
- Output: 3D map and camera trajectory



Classical Approach: Optimization with Multiview Geometry

2D motion (optical flow) is a known analytical function of 3D points and 3D motion



$$f = F(X, T)$$

Step 1. Estimate 2D flow f

→ Match pixels by manual features

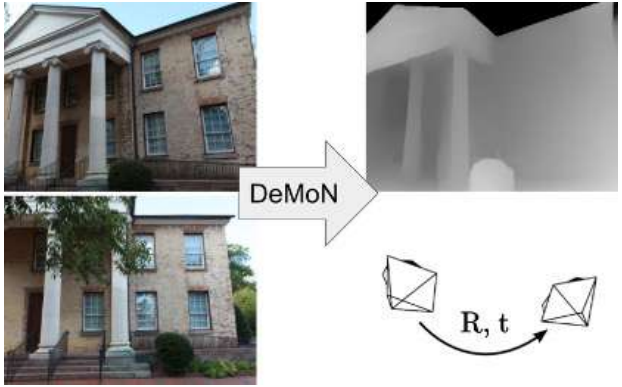
Step 2. Solve for 3D given flow

$$\min_{X,T} \| f - F(X,T) \|^2$$

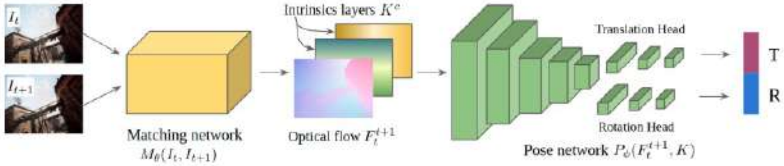
Insufficient Robustness: Failures are frequent and catastrophic

Deep Visual SLAM

Train a network to directly regress ***3D points*** (depth) and **3D motion**



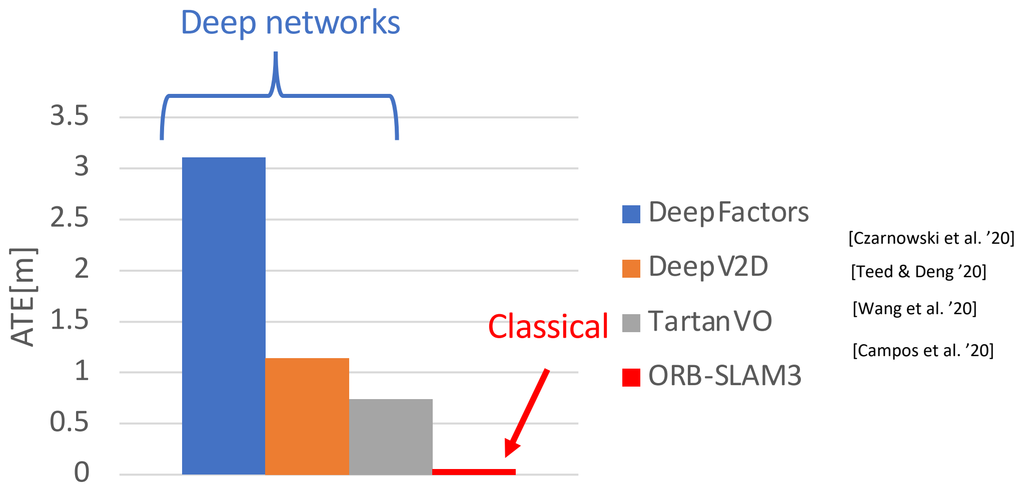
DeMoN [Ummenhofer et al., 2017]



TartanVO [Wang et al., 2021]

Problems with Deep Visual SLAM

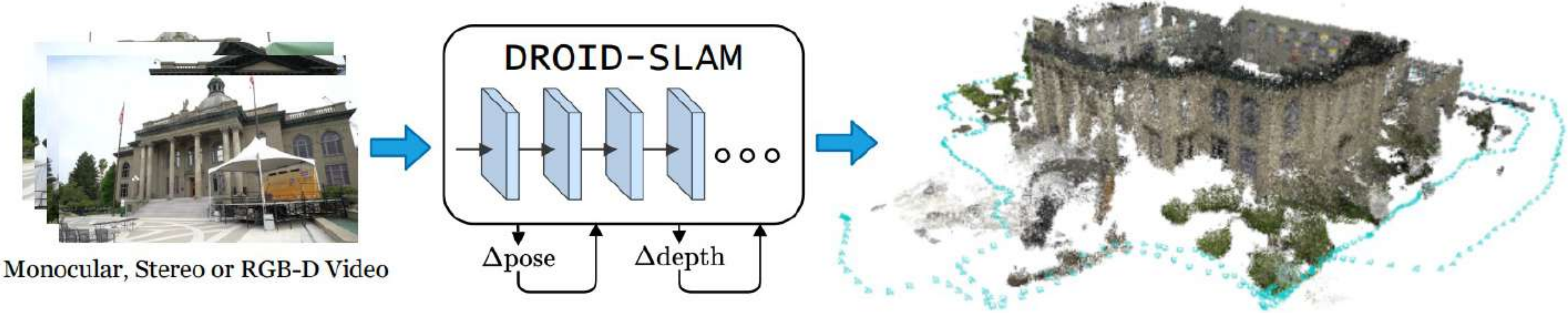
- ***Lower Accuracy:*** large amounts of drift, global inconsistency
- ***Weaker Generalization:*** doesn't generalize to new datasets or cameras



EuRoC MAV Benchmark (Monocular) [Burri et al. 2016]

DROID-SLAM

DROID: **D**ifferentiable **R**ecurrent **O**ptimization-**I**nspired **D**esign

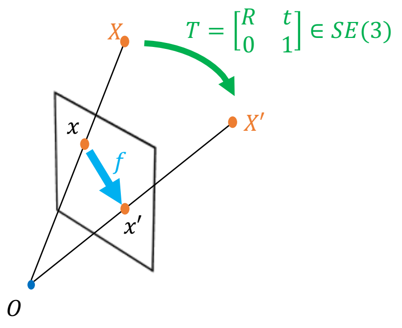


- ***Accurate*** – reduce error by *60%-80%* over prior systems
- ***Robust*** – *6X* fewer catastrophic failures
- ***Generalizable*** – trained only on synthetic data

[Teed & Deng, NeurIPS 2021]

DROID-SLAM

DROID: **D**ifferentiable **R**ecurrent **O**ptimization-**I**nspired **D**esign



$$f = F(X, T)$$

Step 1. Estimate 2D flow f

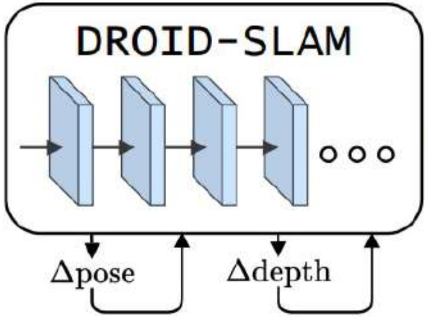
→ Match pixels by manual features

Step 2. Solve for 3D given flow

$$\min_{X,T} \| f - F(X, T) \|^2$$

Symbolic knowledge from classical approaches

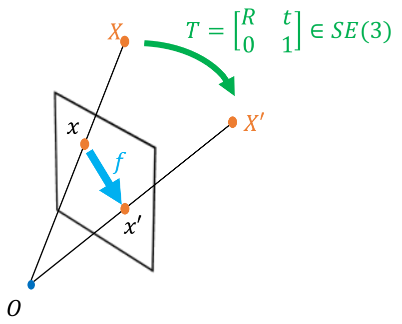
Embed →



End-to-end neural architecture

DROID-SLAM

DROID: **D**ifferentiable **R**ecurrent **O**ptimization-**I**nspired **D**esign



$$f = F(X, T)$$

Step 1. Estimate 2D flow *f*

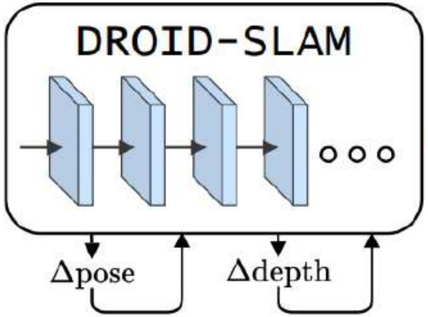
→ Match pixels by manual features

Step 2. Solve for 3D given flow

$$\min_{X,T} \| f - F(X,T) \|^2$$

Embed

→

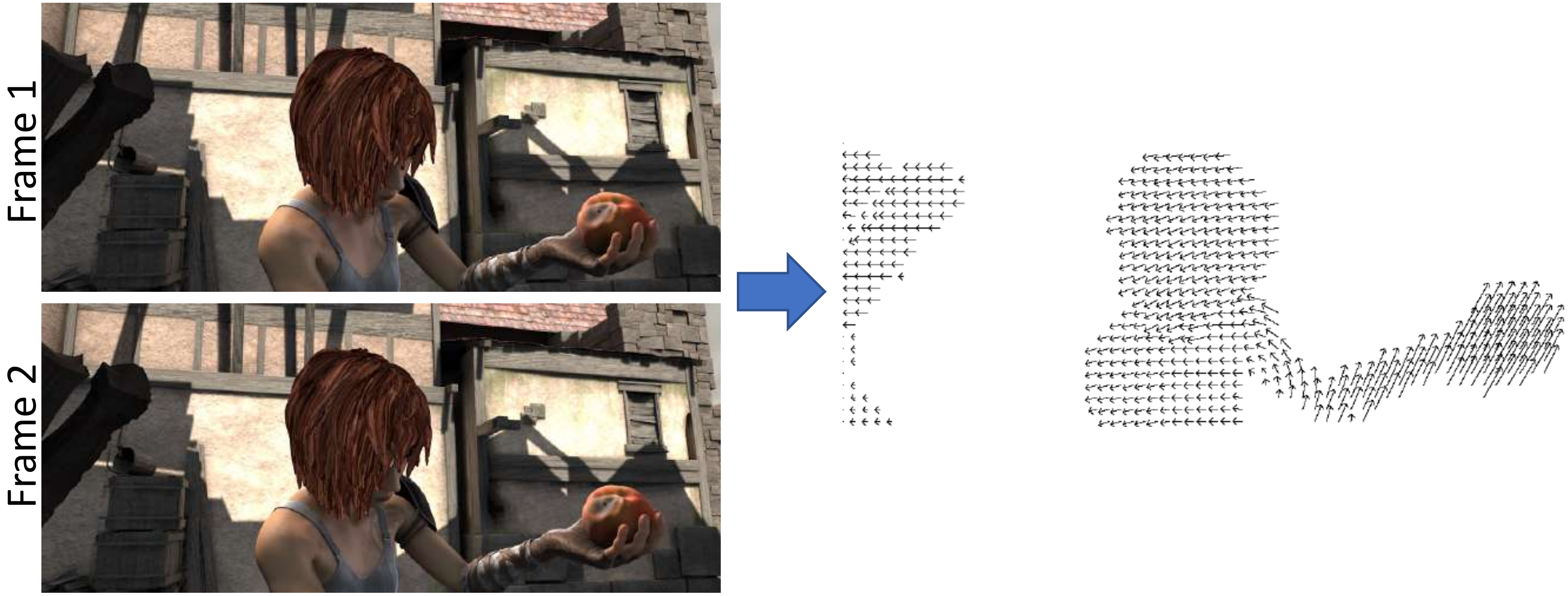


Symbolic knowledge from classical approaches

End-to-end neural architecture

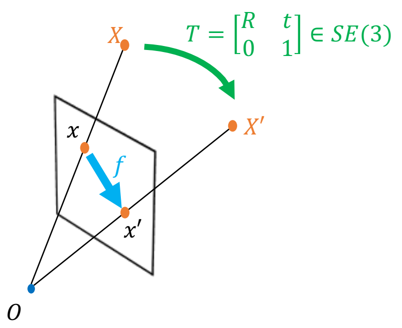
Estimate 2D motion (optical flow)

- Predict per-pixel 2D motion between a pair of frames



DROID-SLAM

DROID: **D**ifferentiable **R**ecurrent **O**ptimization-**I**nspired **D**esign



$$f = F(X, T)$$

Step 1. Estimate 2D flow f

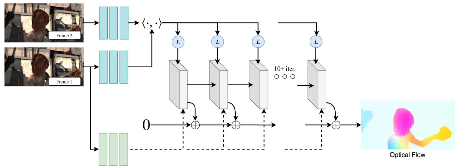
→ Match pixels by manual features

Step 2. Solve for 3D given flow

$$\min_{X,T} \| f - F(X,T) \|^2$$

Symbolic knowledge from classical approaches

Embed →



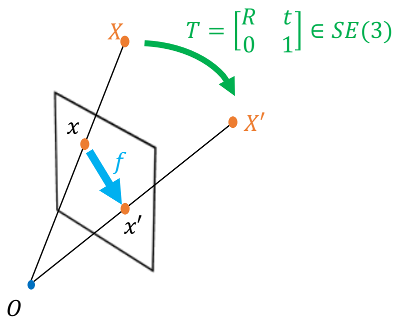
RAFT [T**D**,ECCV20]

?

End-to-end neural architecture

DROID-SLAM

DROID: **D**ifferentiable **R**ecurrent **O**ptimization-**I**nspired **D**esign



$$f = F(X, T)$$

Step 1. Estimate 2D flow f

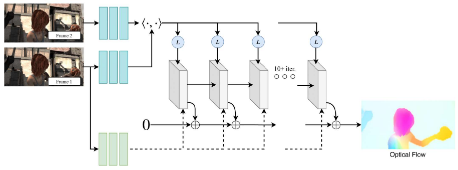
→ Match pixels by manual features

Step 2. Solve for 3D given flow

$$\min_{X,T} \|f - F(X,T)\|^2$$

Symbolic knowledge from classical approaches

Embed →



RAFT [T**D**,ECCV20]

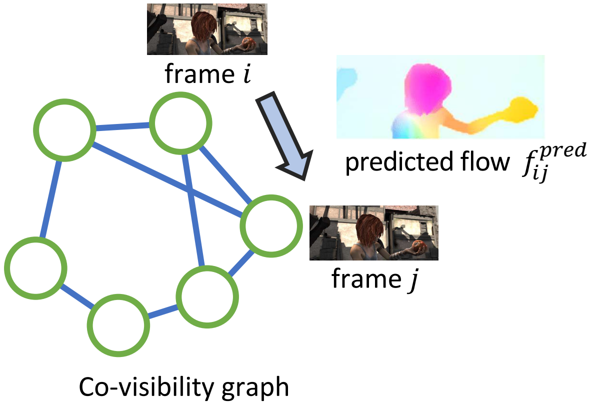
Step 2. Solve for 3D given flow

$$\min_{X,T} \|f - F(X,T)\|^2$$

End-to-end neural architecture

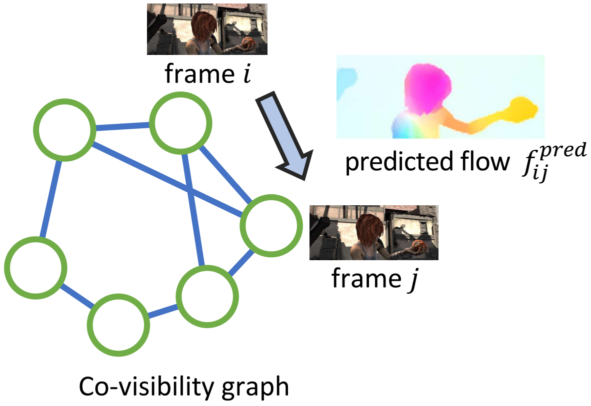
Dense Bundle Adjustment (DBA)

- ***Given:*** co-visibility graph $(\mathcal{V}, \mathcal{E})$, predicted flow f_{ij}^{pred}



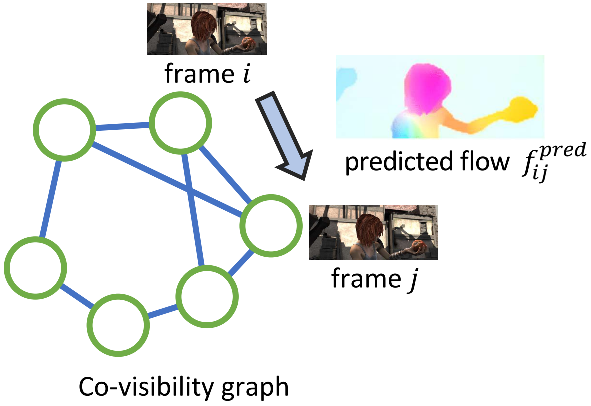
Dense Bundle Adjustment (DBA)

- ***Given:*** co-visibility graph $(\mathcal{V}, \mathcal{E})$, predicted flow f_{ij}^{pred}
- ***Want:*** depth maps $\boldsymbol{d} = (d_1, \ldots, d_i, \ldots)$, camera poses $\boldsymbol{T} = (T_1, \ldots, T_i, \ldots)$



Dense Bundle Adjustment (DBA)

- ***Given:*** co-visibility graph $(\mathcal{V}, \mathcal{E})$, predicted flow f_{ij}^{pred}
- ***Want:*** depth maps $\boldsymbol{d} = (d_1, \dots, d_i, \dots)$, camera poses $\boldsymbol{T} = (T_1, \dots, T_i, \dots)$

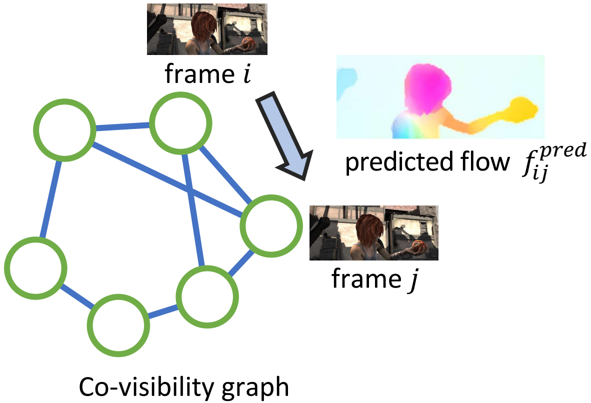


Dense Bundle Adjustment

$$\min_{\boldsymbol{d},\boldsymbol{T}} \sum_{(i,j)\in\mathcal{E}} \left\| \underbrace{f_{ij}^{pred}}_{\text{Predicted flow}} - \underbrace{f_{ij}^{ind}(\boldsymbol{d},\boldsymbol{T})}_{\text{Induced flow}} \right\|^2 \Rightarrow \boldsymbol{T}^*, \boldsymbol{d}^*$$

Dense Bundle Adjustment (DBA)

- ***Given:*** co-visibility graph $(\mathcal{V}, \mathcal{E})$, predicted flow f_{ij}^{pred}
- ***Want:*** depth maps $\boldsymbol{d} = (d_1, \dots, d_i, \dots)$, camera poses $\boldsymbol{T} = (T_1, \dots, T_i, \dots)$



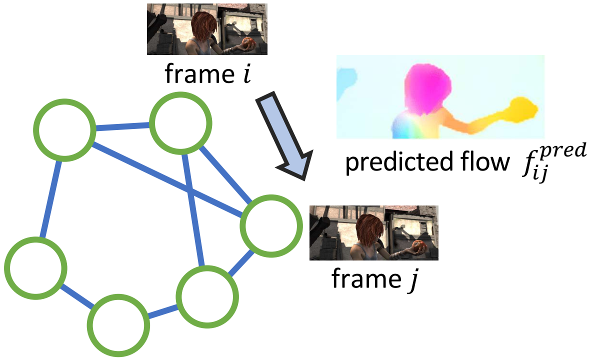
Dense Bundle Adjustment

$$\min_{\boldsymbol{d},\boldsymbol{T}} \sum_{(i,j)\in\mathcal{E}} \left\| \underbrace{f_{ij}^{pred}}_{\text{Predicted flow}} - \underbrace{f_{ij}^{ind}(\boldsymbol{d},\boldsymbol{T})}_{\text{Induced flow}} \right\|^2$$

$\boldsymbol{T}^*, \boldsymbol{d}^*$

Dense Bundle Adjustment (DBA)

- ***Given:*** co-visibility graph $(\mathcal{V}, \mathcal{E})$, predicted flow f_{ij}^{pred}
- ***Want:*** depth maps $\boldsymbol{d} = (d_1, \dots, d_i, \dots)$, camera poses $\boldsymbol{T} = (T_1, \dots, T_i, \dots)$

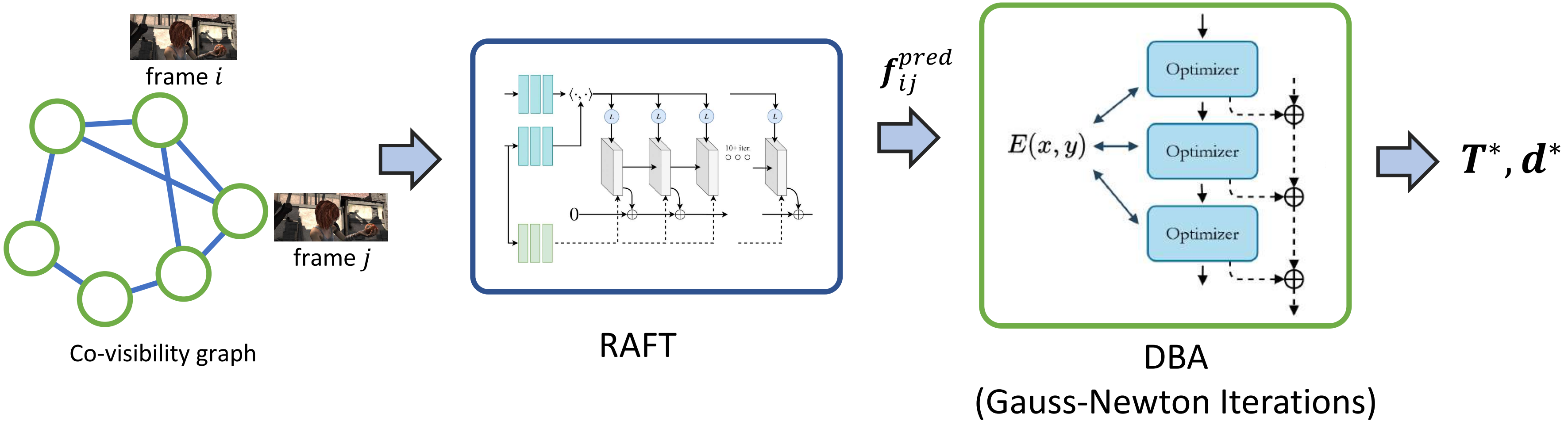


Dense Bundle Adjustment

$$\min_{\boldsymbol{d},\boldsymbol{T}} \sum_{(i,j)\in\mathcal{E}} \left\| \underbrace{f_{ij}^{pred}}_{\text{Predicted flow}} - \underbrace{f_{ij}^{ind}(\boldsymbol{d},\boldsymbol{T})}_{\text{Induced flow}} \right\|^2 \Rightarrow \boldsymbol{T}^*, \boldsymbol{d}^*$$

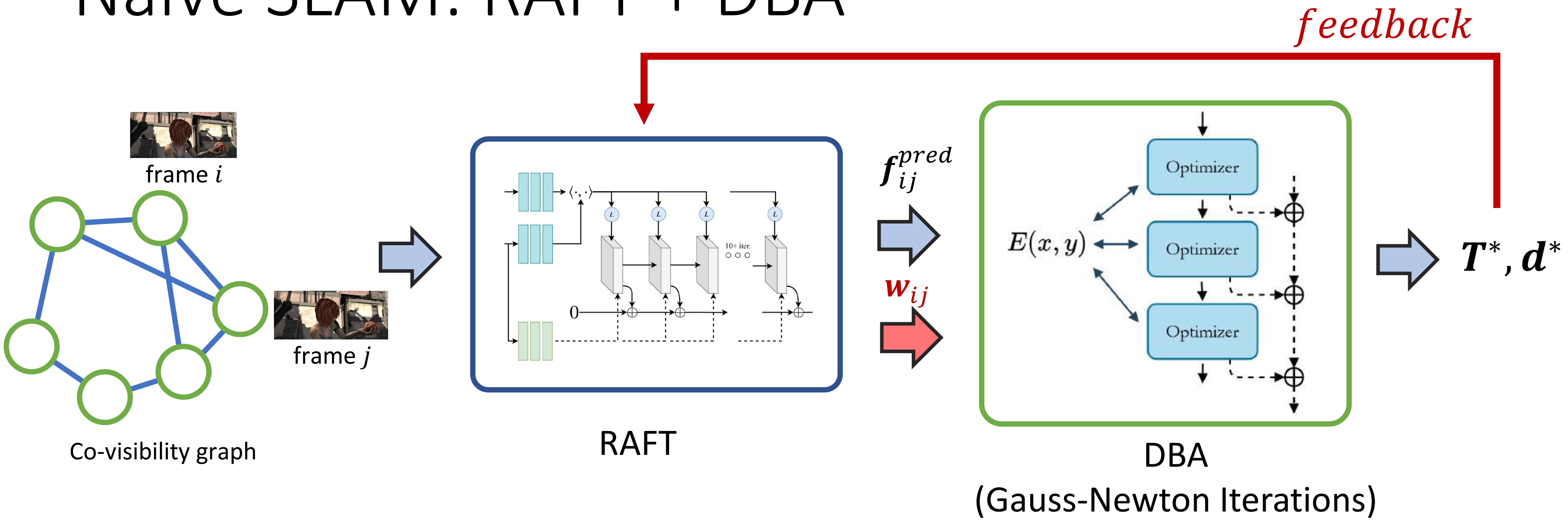
- Non-linear least squares
- Iterative algorithms like Gauss-Newton

Naïve SLAM: RAFT + DBA



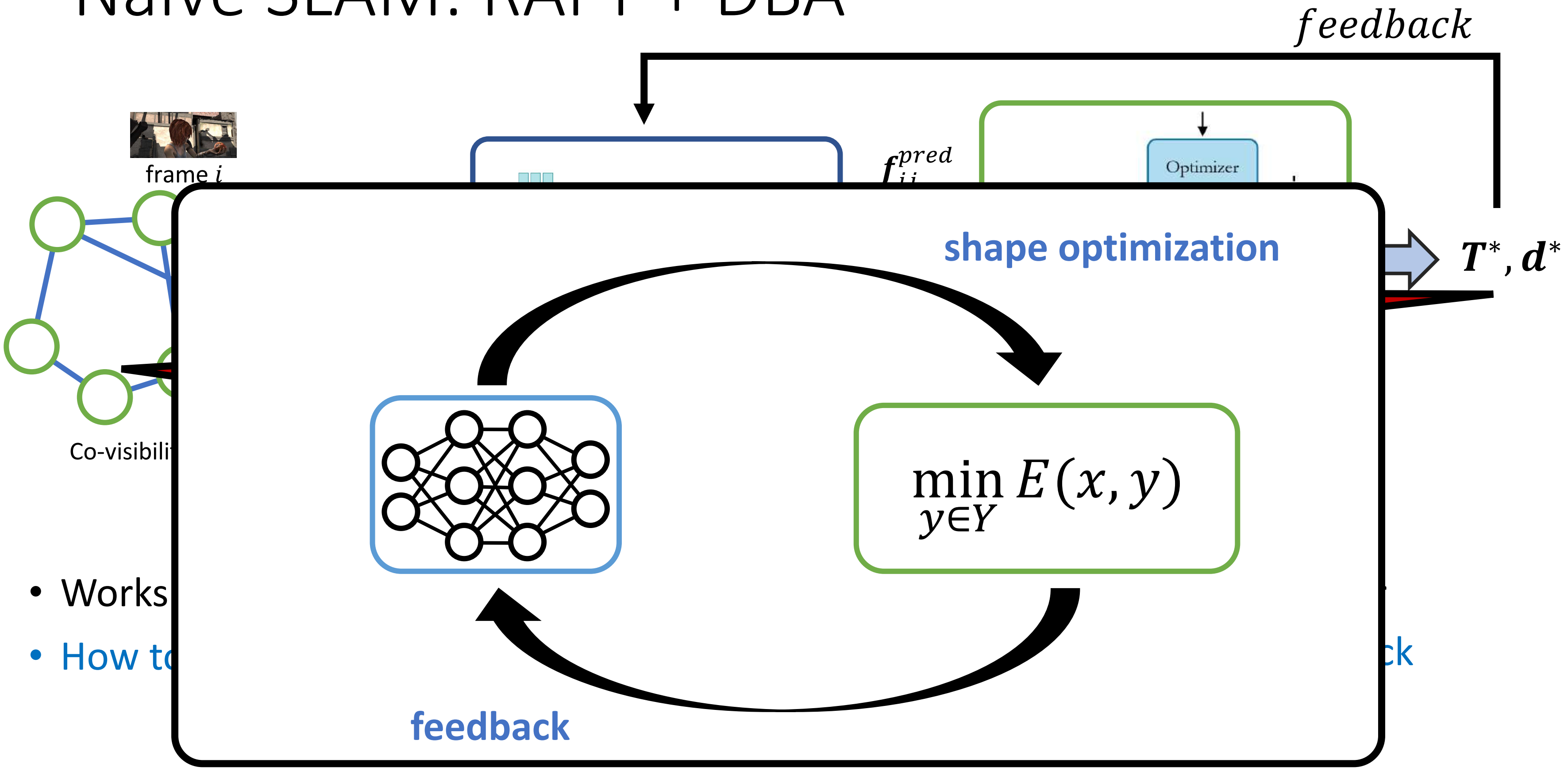
- Works poorly, because of outliers: visibility, dynamic objects, prediction error

Naïve SLAM: RAFT + DBA



- Works poorly, because of outliers: visibility, dynamic objects, prediction error
- How to exclude outliers? (1) Predicted Confidence Map (2) Feedback

Naïve SLAM: RAFT + DBA

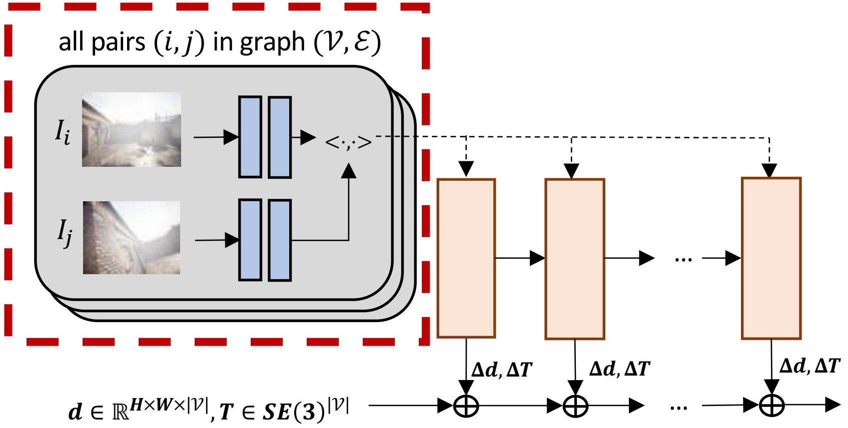


- Works
- How t

ck

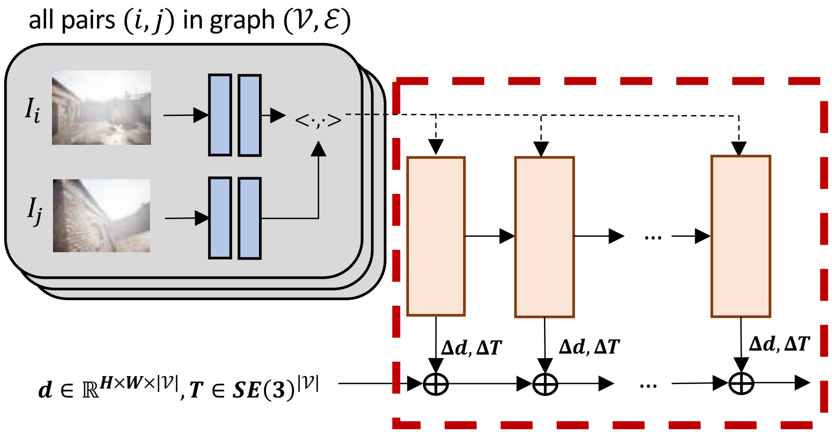
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer



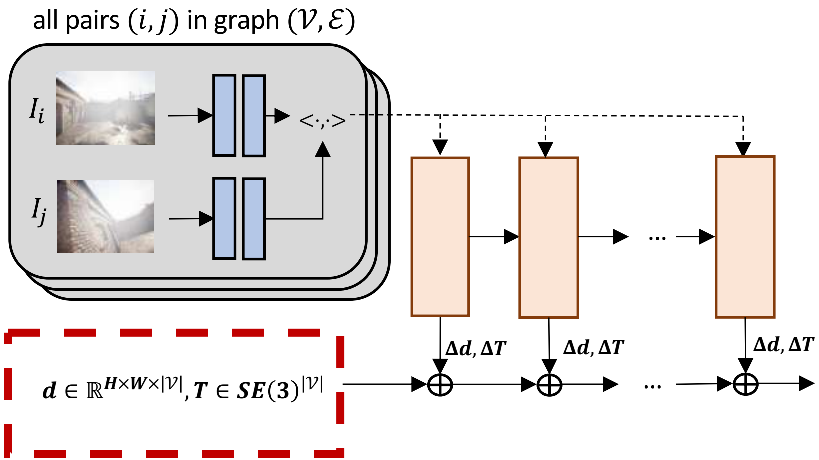
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer



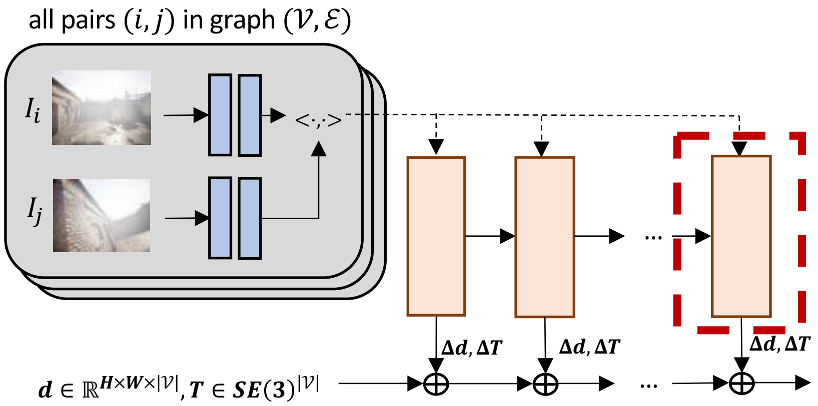
DROID-SLAM: Architecture

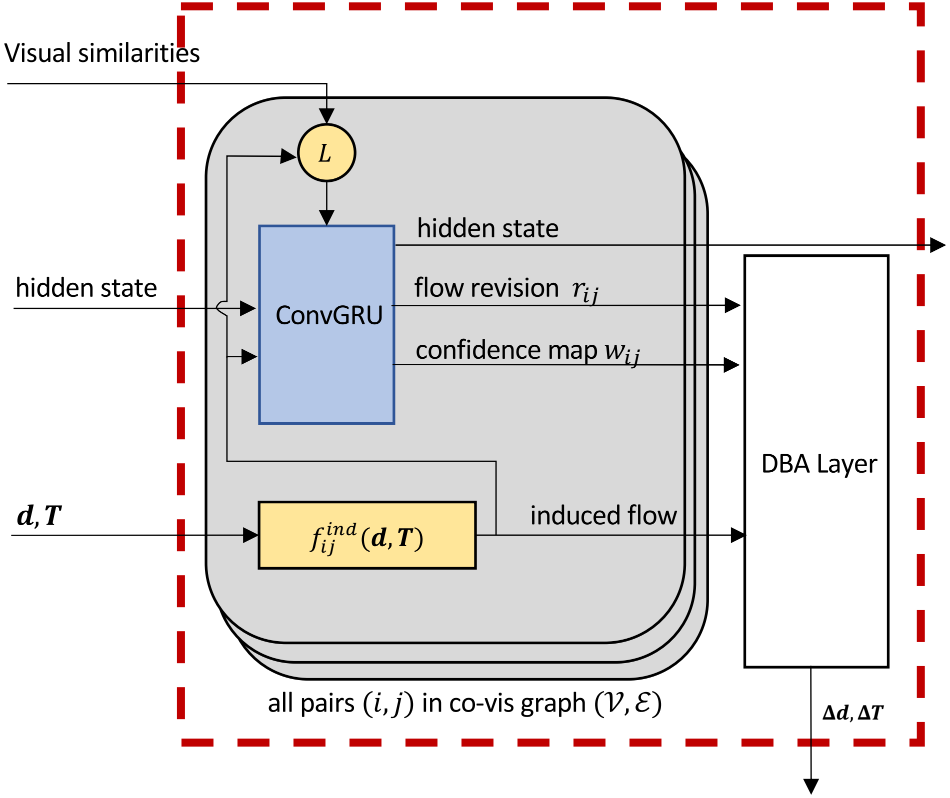
- Recurrent Updates + Analytical Layer



DROID-SLAM: Architecture

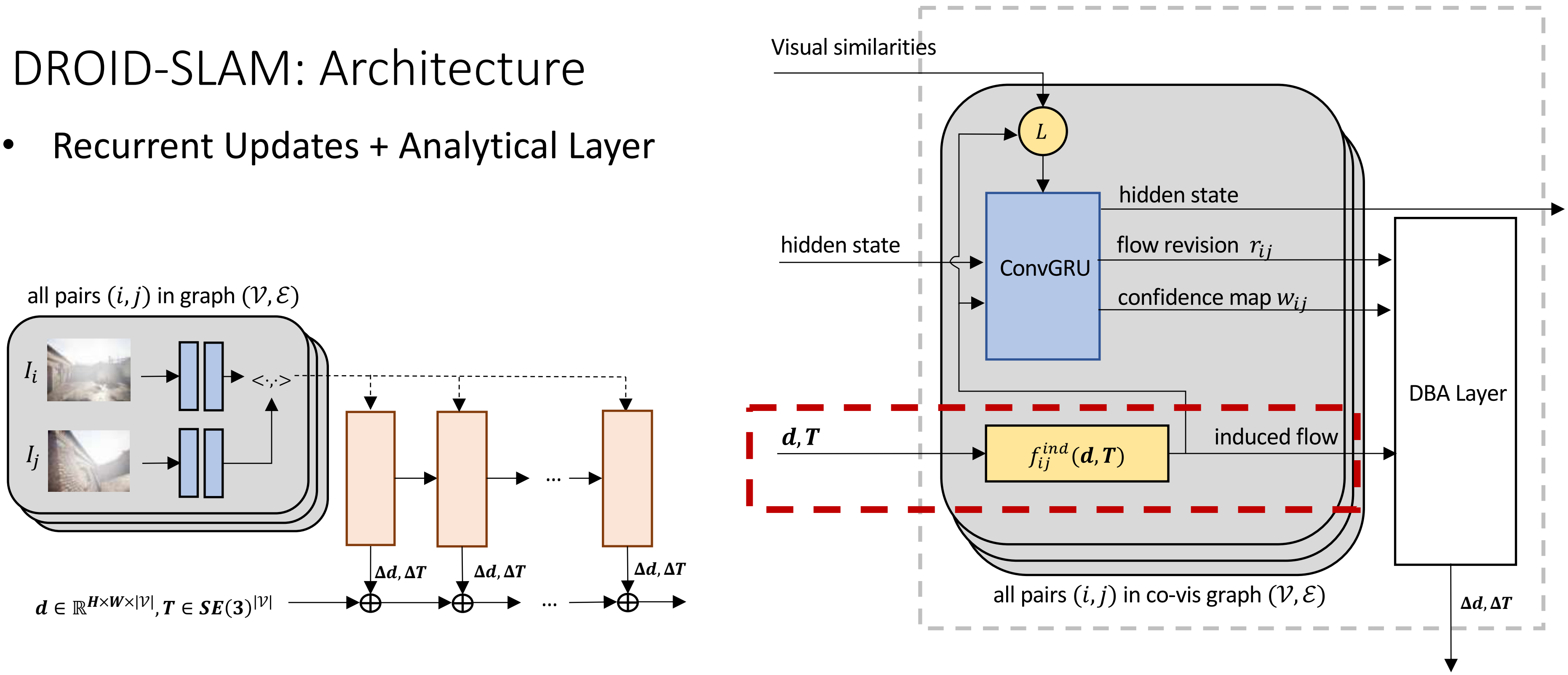
- Recurrent Updates + Analytical Layer





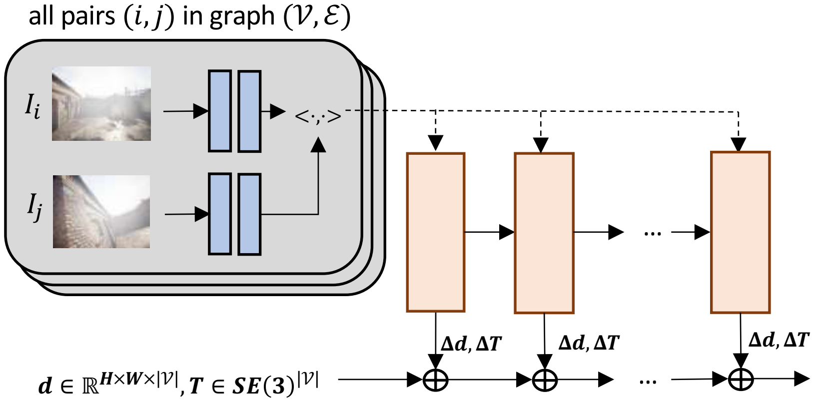
DROID-SLAM: Architecture

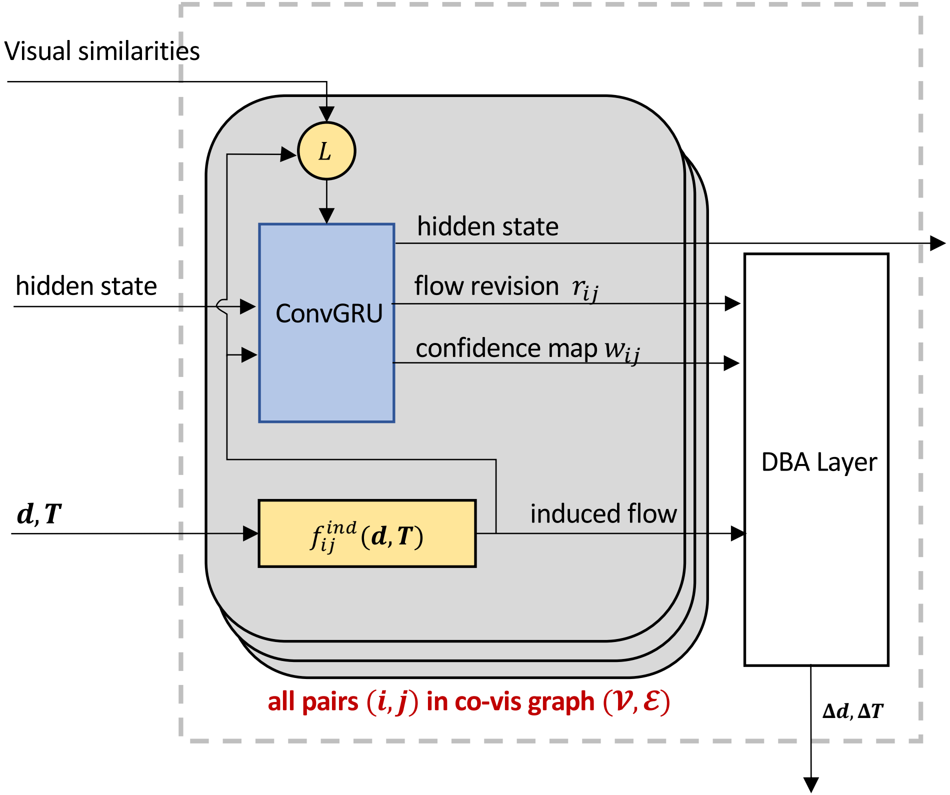
- Recurrent Updates + Analytical Layer



DROID-SLAM: Architecture

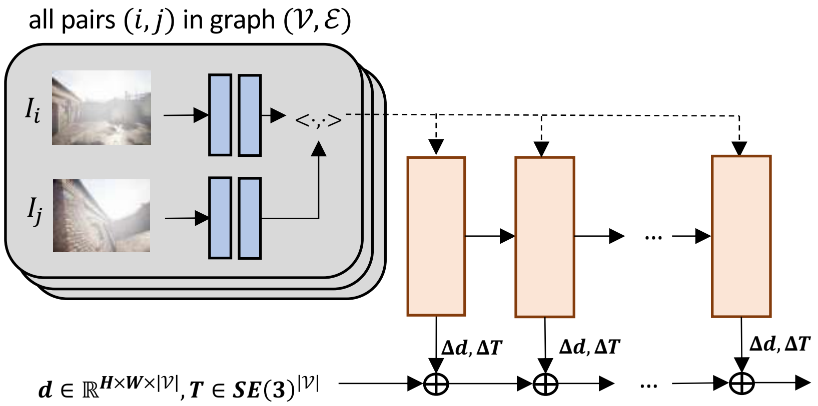
- Recurrent Updates + Analytical Layer

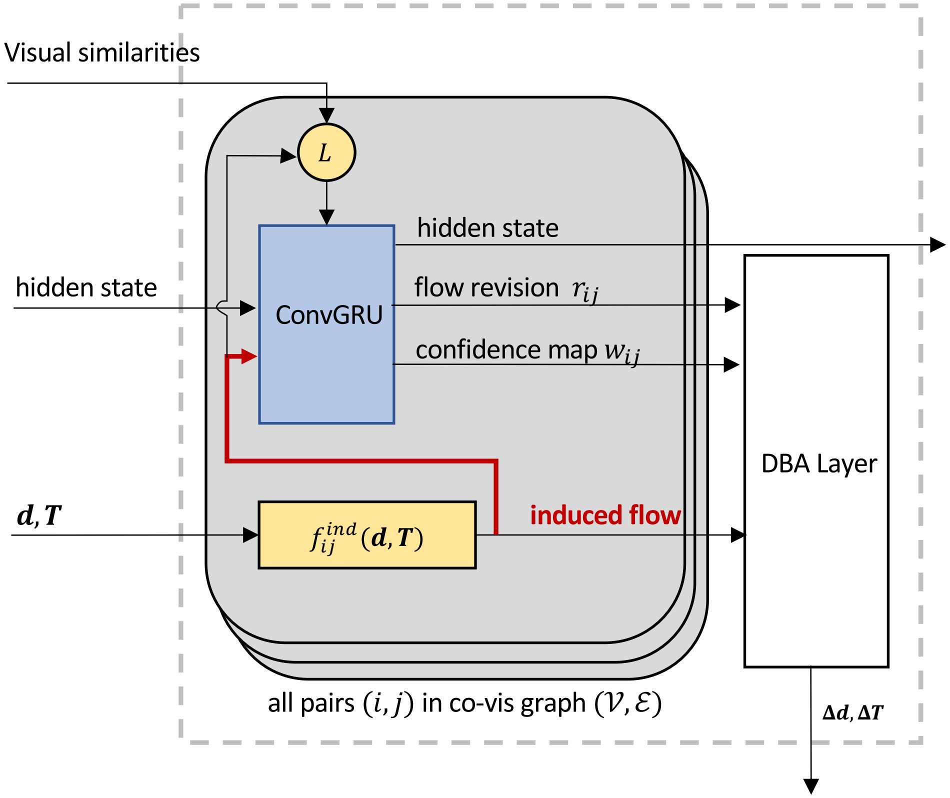




DROID-SLAM: Architecture

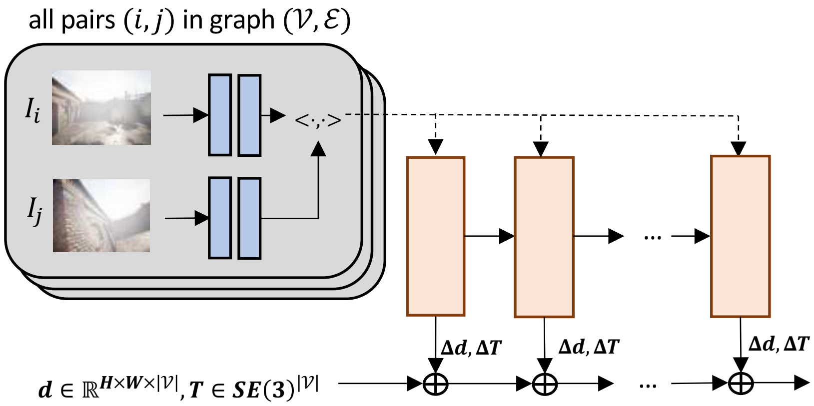
- Recurrent Updates + Analytical Layer

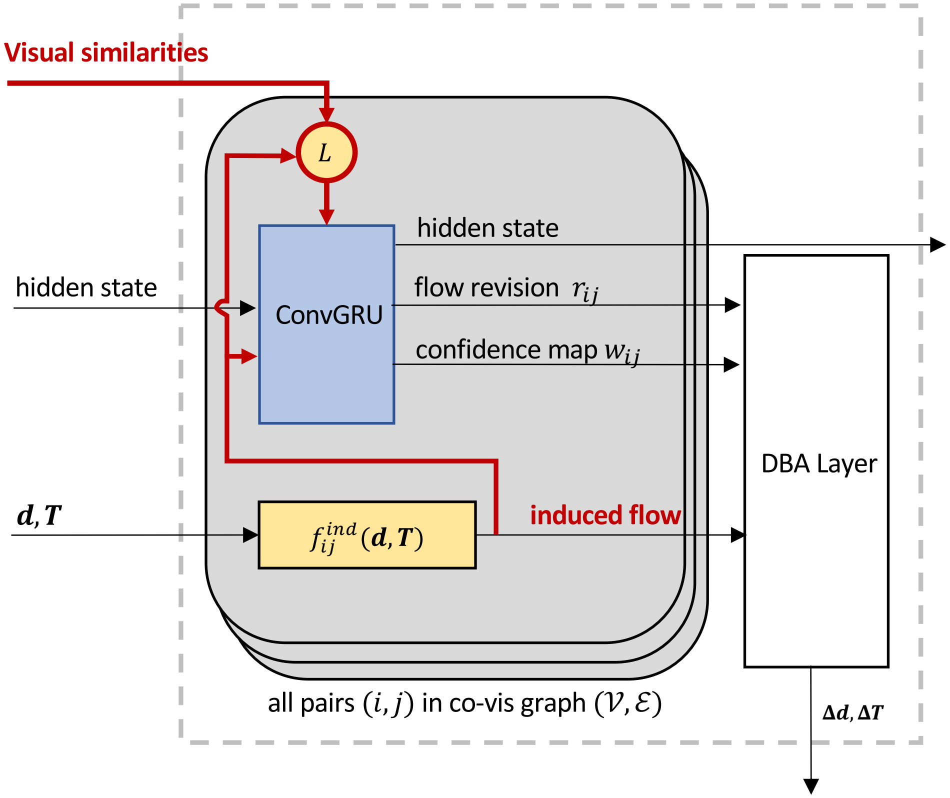




DROID-SLAM: Architecture

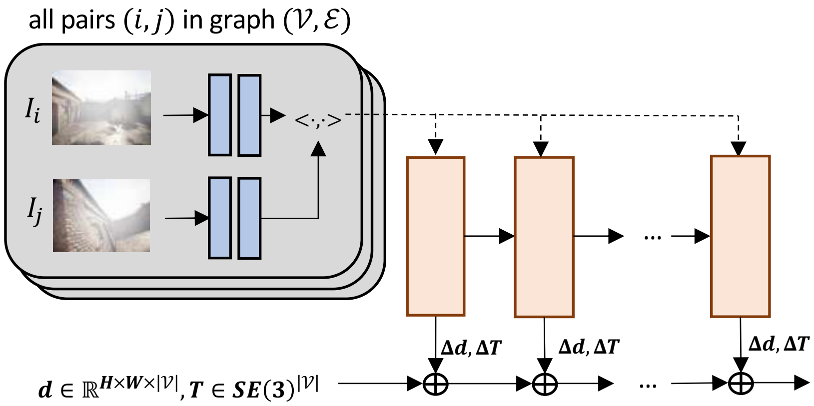
- Recurrent Updates + Analytical Layer

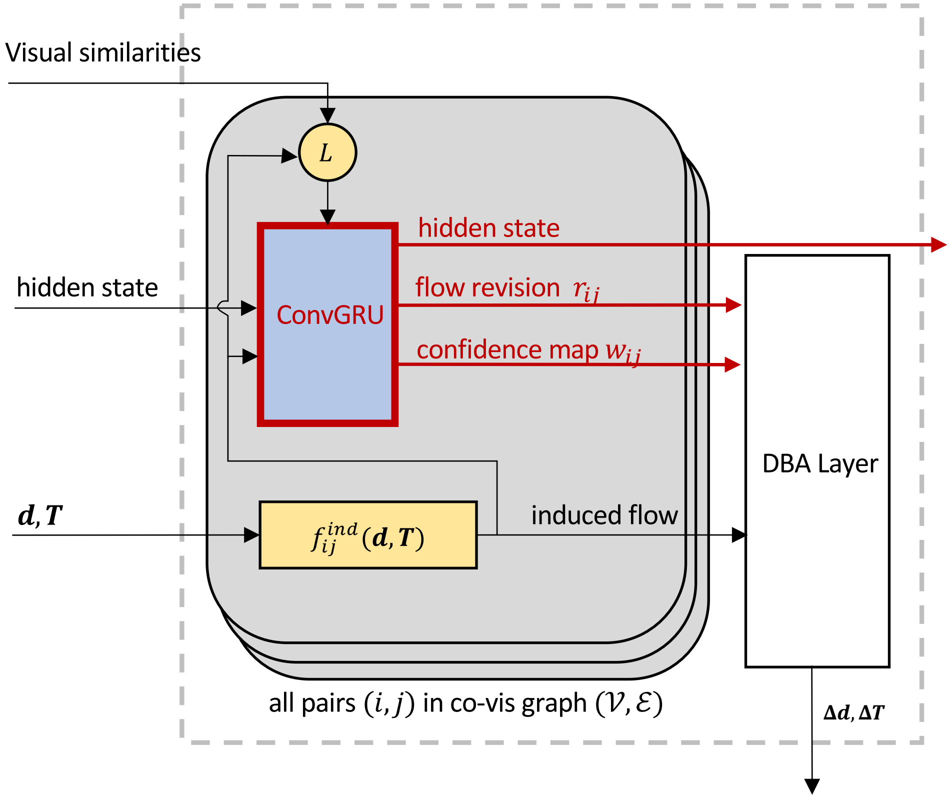




DROID-SLAM: Architecture

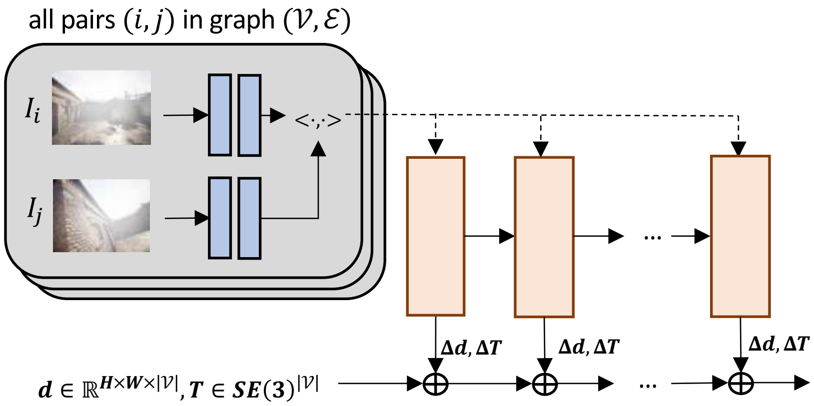
- Recurrent Updates + Analytical Layer

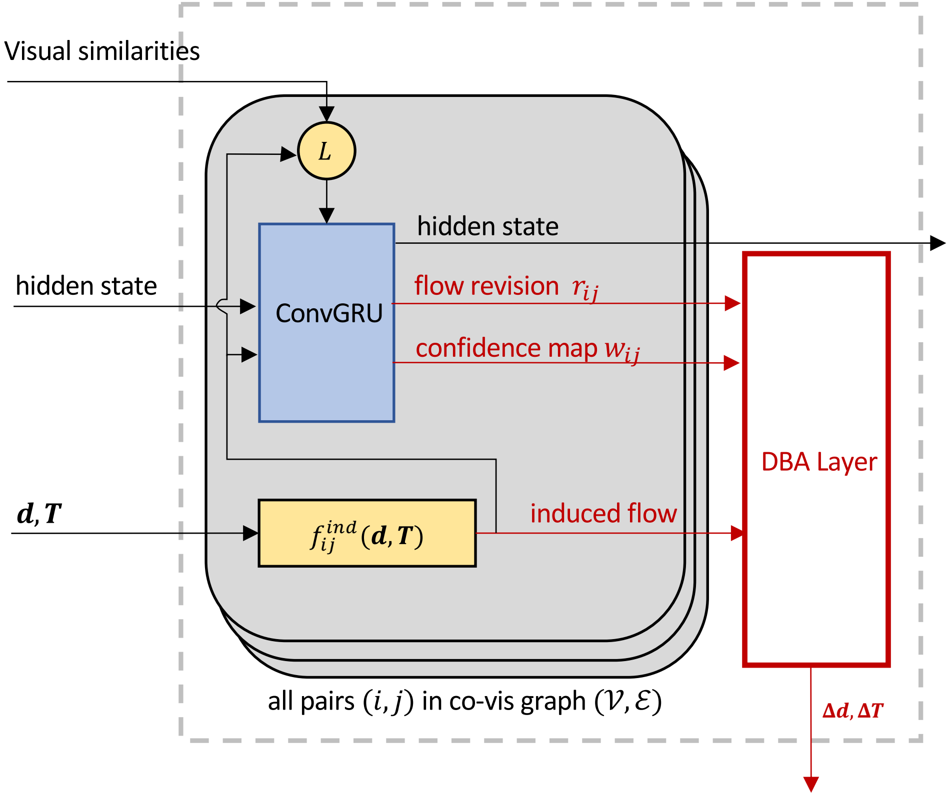




DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

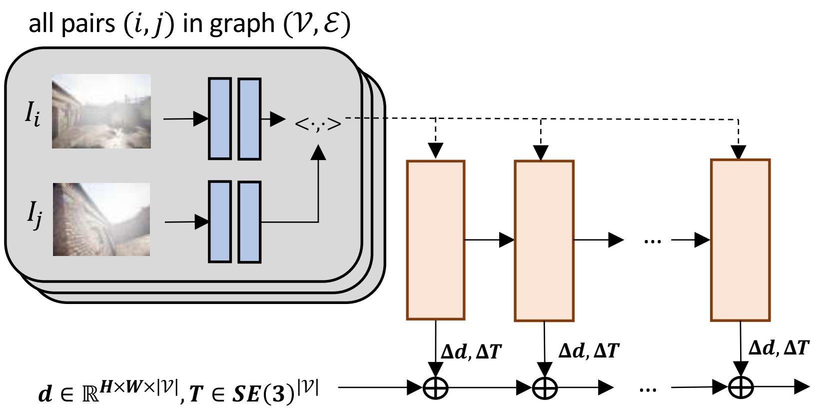


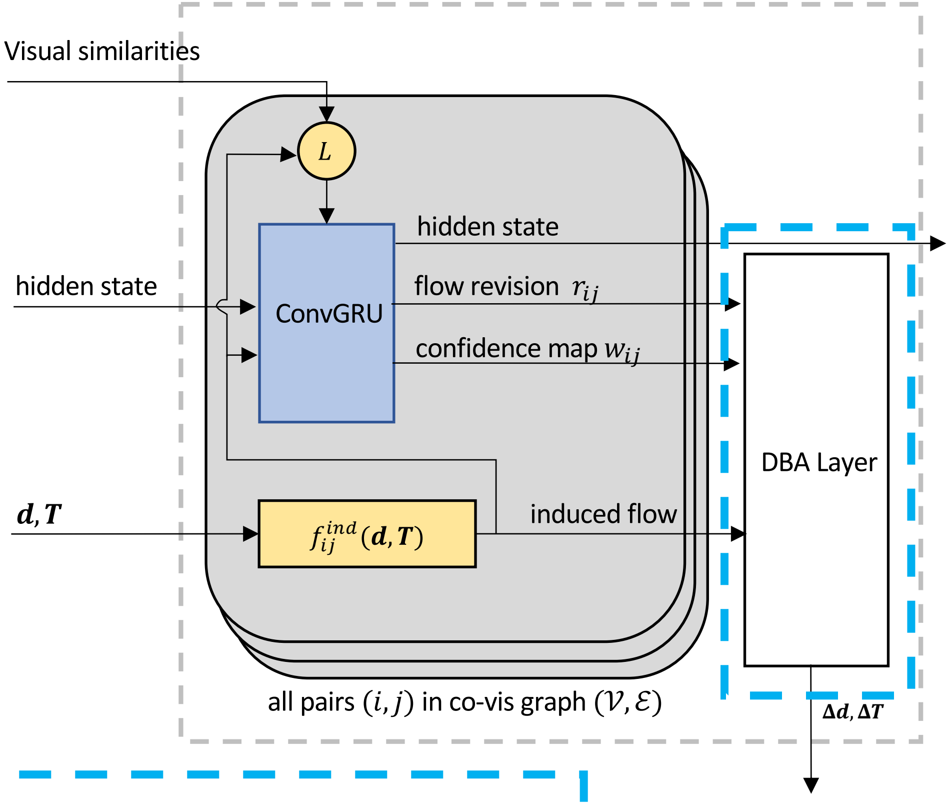


DBA Layer: how to update depth and poses to make induced flow better?

DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

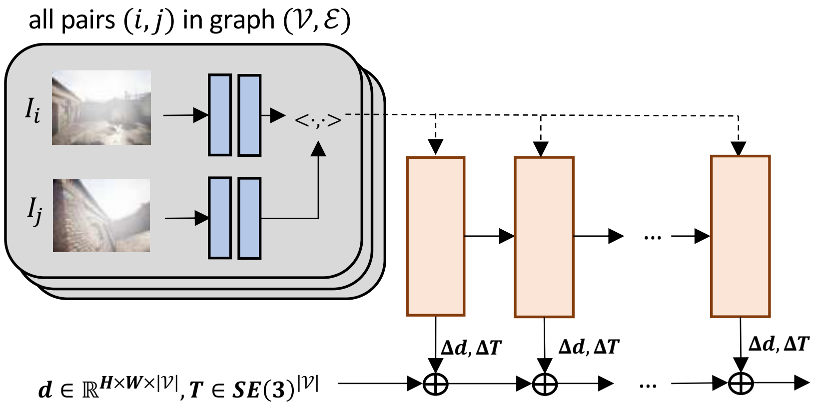


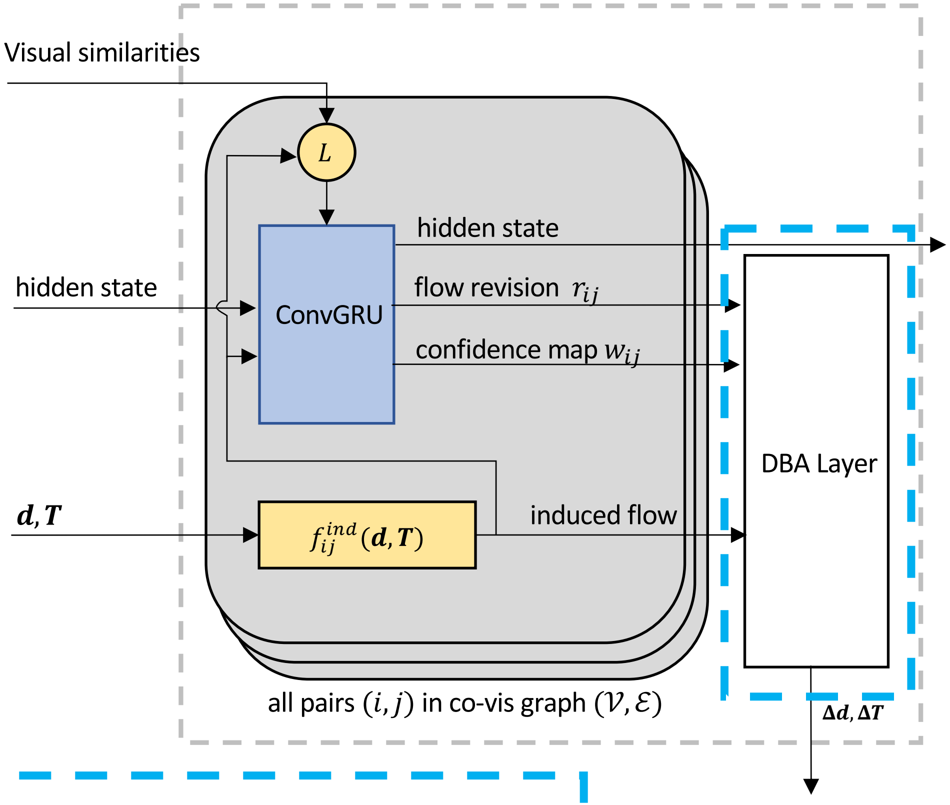


$$\min_{\Delta \boldsymbol{d},\Delta \boldsymbol{T}} \sum_{(i,j)\in\mathcal{E}} \left\| f_{ij}^{ind}(\boldsymbol{d},\boldsymbol{T}) + r_{ij} - f_{ij}^{ind}(\boldsymbol{d}+\Delta\boldsymbol{d}, \boldsymbol{T}+\Delta\boldsymbol{T})\right\|^2_{diag(w_{ij})}$$

DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

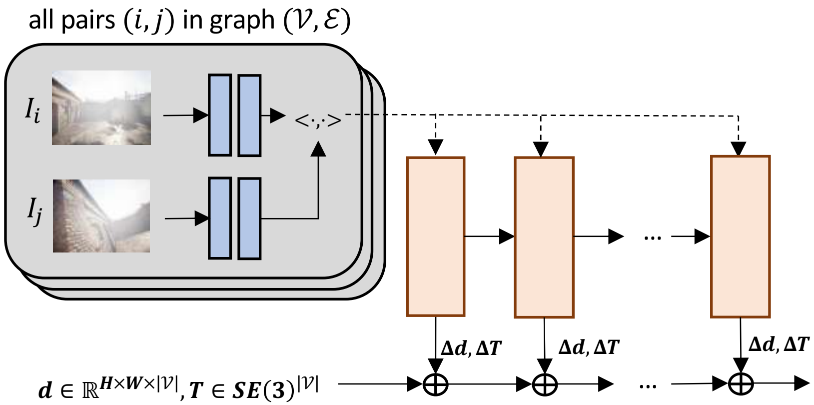


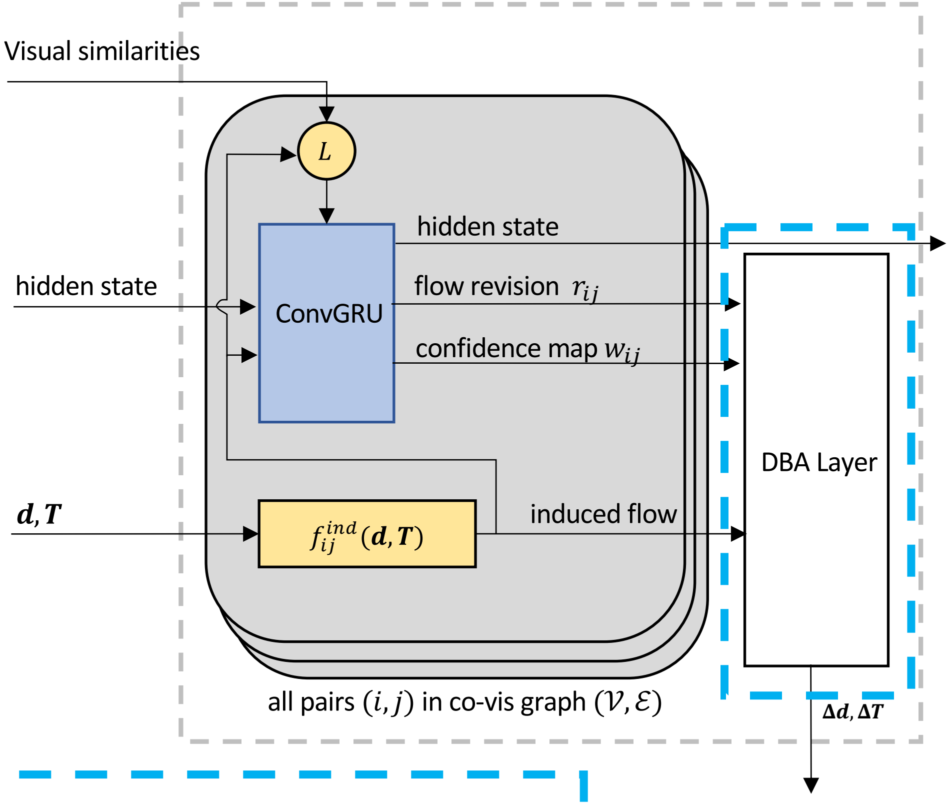


$$\min_{\Delta\boldsymbol{d},\Delta\boldsymbol{T}} \sum_{(i,j)\in\mathcal{E}} \left\| f_{ij}^{ind}(\boldsymbol{d},\boldsymbol{T}) + r_{ij} - f_{ij}^{ind}(\boldsymbol{d}+\Delta\boldsymbol{d},\boldsymbol{T}+\Delta\boldsymbol{T}) \right\|^2_{diag(w_{ij})}$$

DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer



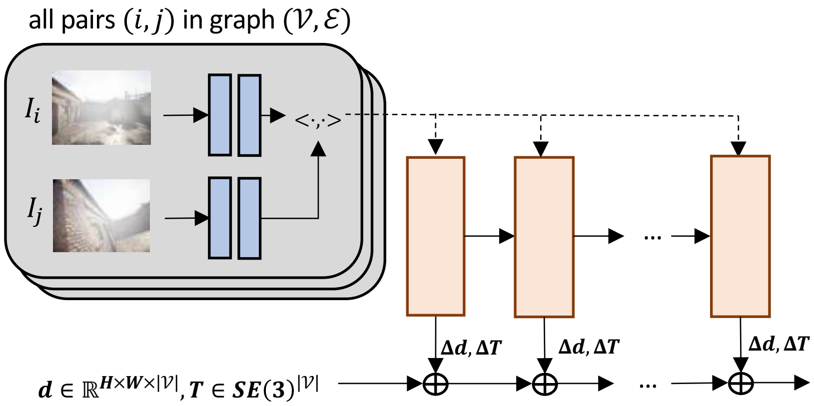


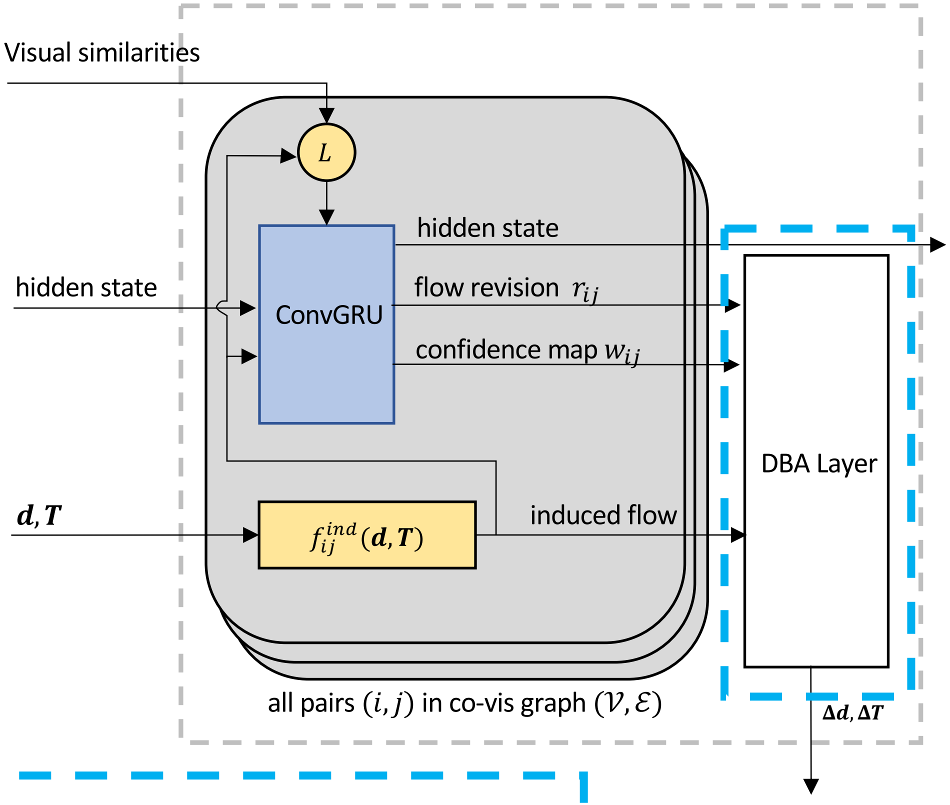
$$\min_{\Delta d, \Delta T} \sum_{(i,j)\in\mathcal{E}} \left\| f_{ij}^{ind}(d, T) + r_{ij} - f_{ij}^{ind}(d + \Delta d, T + \Delta T) \right\|_{diag(w_{ij})}^2$$

Current induced flow between frame i, j

DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer





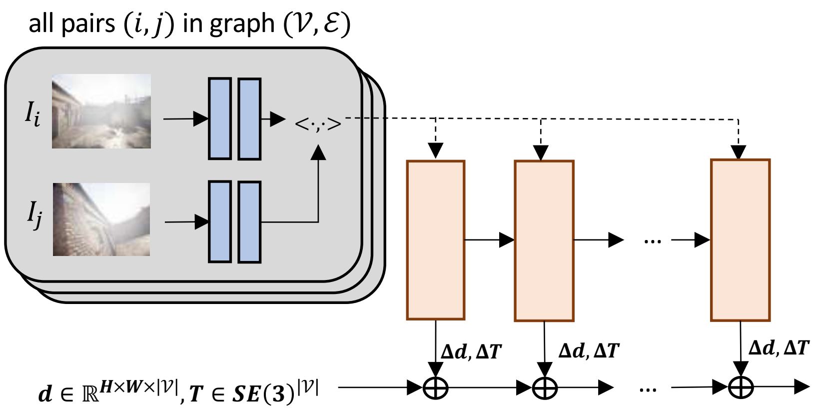
$$\min_{\Delta \boldsymbol{d}, \Delta \boldsymbol{T}} \sum_{(i,j)\in\mathcal{E}} \left\| f_{ij}^{ind}(\boldsymbol{d}, \boldsymbol{T}) + r_{ij} - f_{ij}^{ind}(\boldsymbol{d} + \Delta \boldsymbol{d}, \boldsymbol{T} + \Delta \boldsymbol{T}) \right\|^2_{diag(w_{ij})}$$

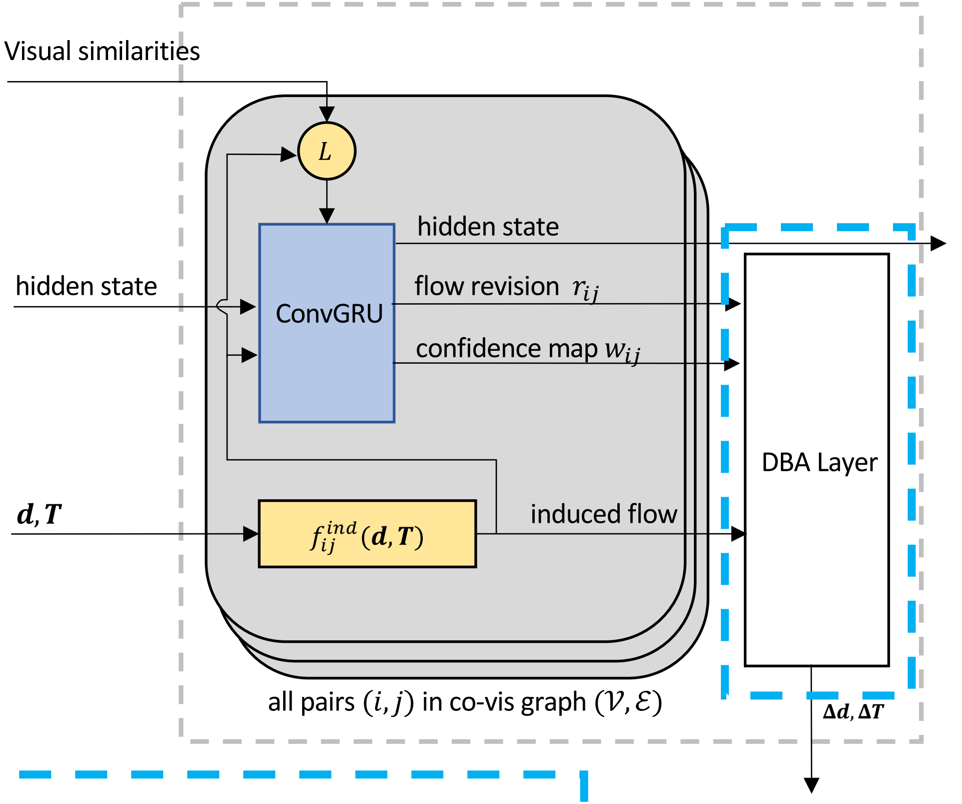
Current induced flow between frame i, j

flow revision

DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer





$$\min_{\Delta d, \Delta T} \sum_{(i,j)\in\mathcal{E}} \left\| f_{ij}^{ind}(d, T) + r_{ij} - f_{ij}^{ind}(d + \Delta d, T + \Delta T) \right\|_{diag(w_{ij})}^2$$

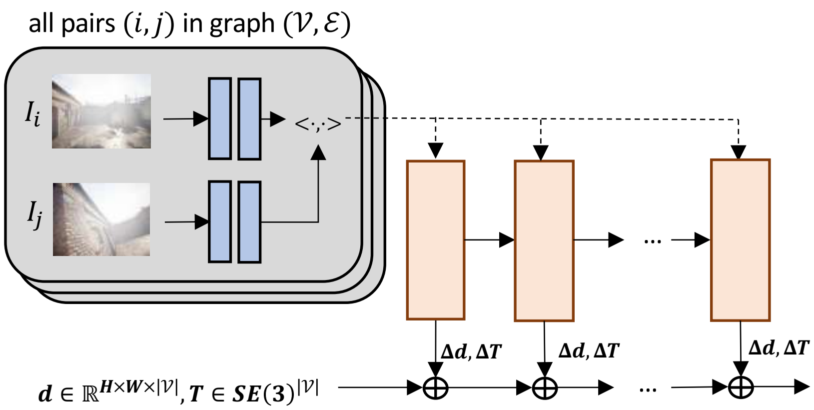
Current induced flow between frame i, j

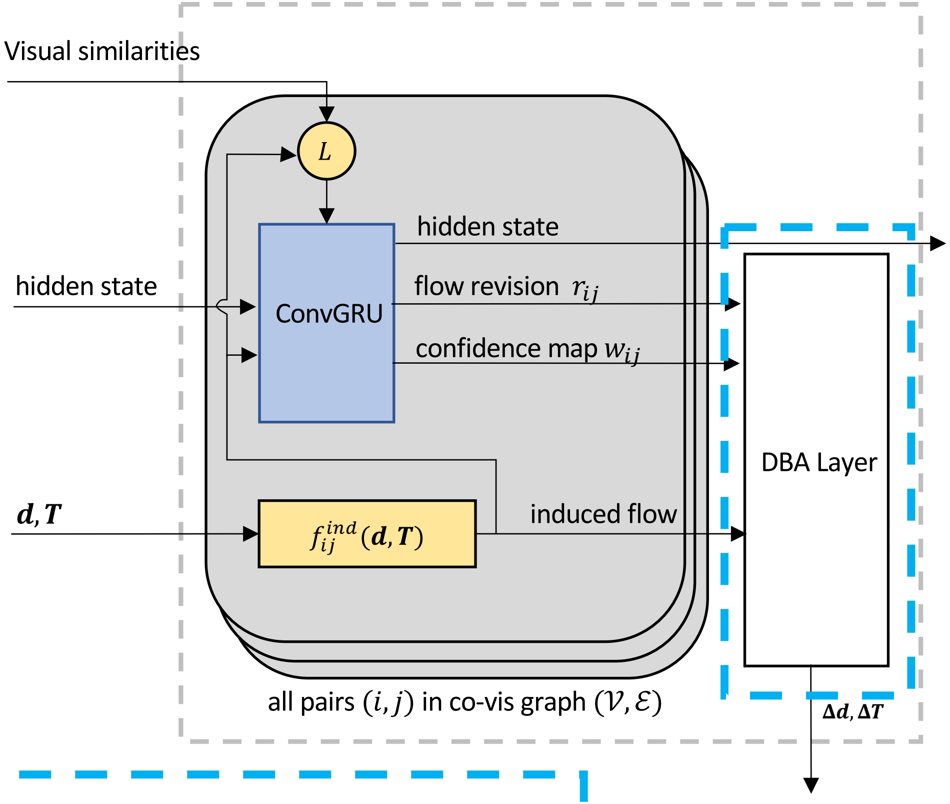
flow revision

new induced flow between frame i, j

DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer





$$\min_{\Delta d, \Delta T} \sum_{(i,j)\in\mathcal{E}} \left\| f_{ij}^{ind}(d, T) + r_{ij} - f_{ij}^{ind}(d + \Delta d, T + \Delta T) \right\|^2_{diag(w_{ij})}$$

Current induced flow between frame i, j

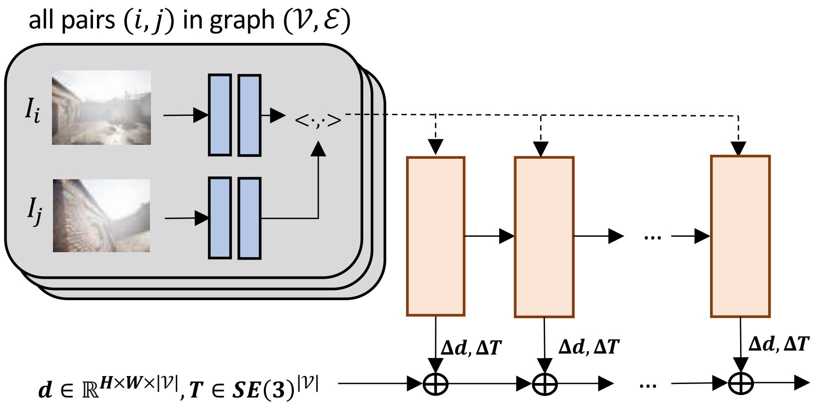
flow revision

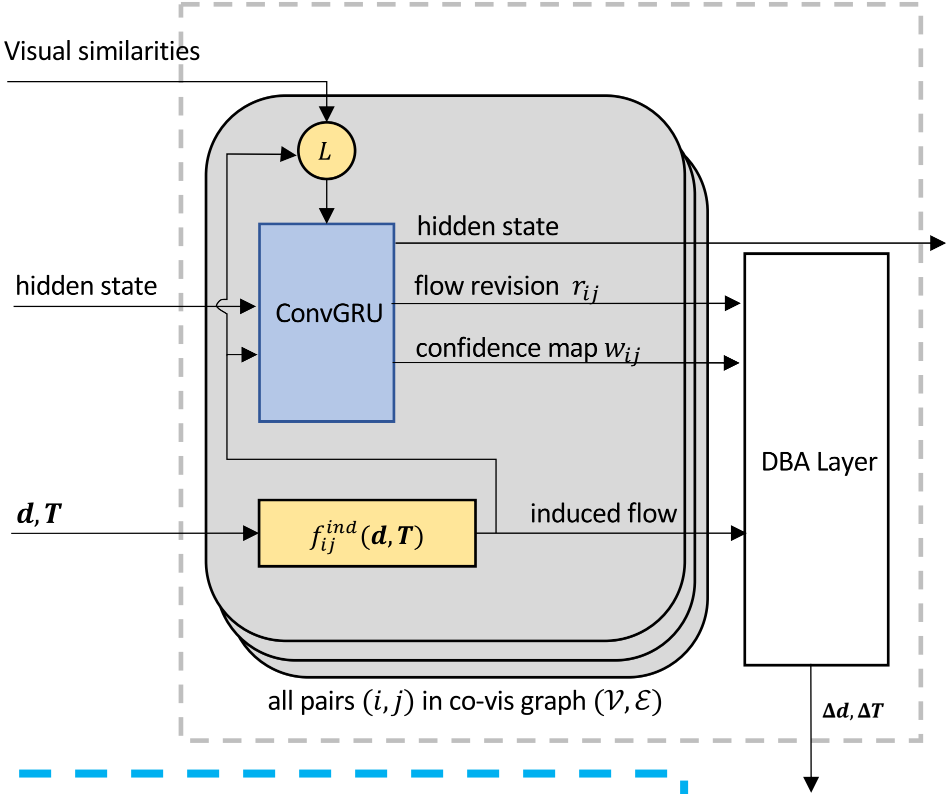
new induced flow between frame i, j

pixel confidence

DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer





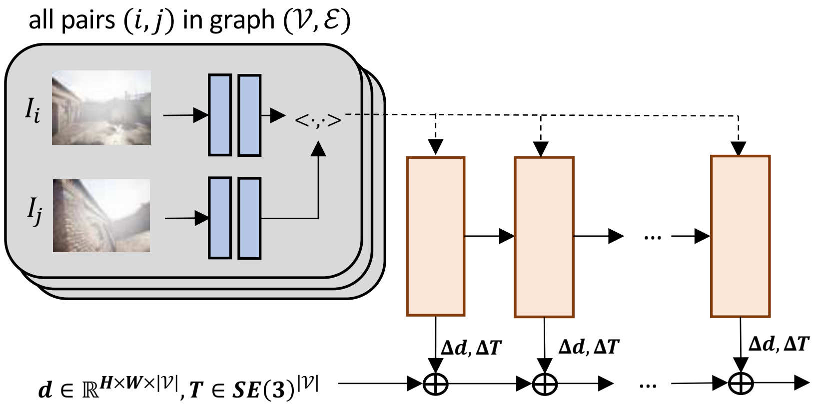
$$\min_{\Delta d, \Delta T} \sum_{(i,j) \in \mathcal{E}} \left\| f_{ij}^{ind}(d, T) + r_{ij} - f_{ij}^{ind}(d + \Delta d, T + \Delta T) \right\|_{diag(w_{ij})}^2$$

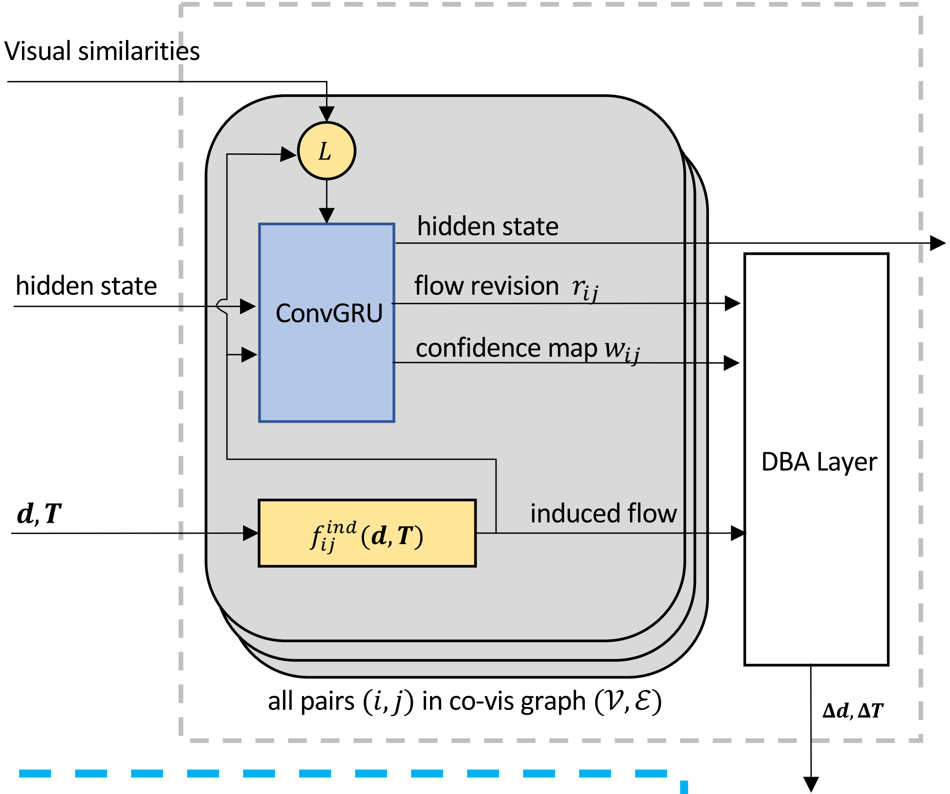
linearize

$$\min_{\Delta d, \Delta T} \sum_{(i,j) \in \mathcal{E}} \left\| r_{ij} - \frac{\partial f_{ij}^{ind}(d, T)}{\partial d} \Delta d - \frac{\partial f_{ij}^{ind}(d, T)}{\partial T} \Delta T) \right\|_{diag(w_{ij})}^2$$

DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer





$$\min_{\Delta d,\Delta T} \sum_{(i,j)\in\mathcal{E}} \left\| f_{ij}^{ind}(d,T) + r_{ij} - f_{ij}^{ind}(d+\Delta d, T+\Delta T)\right\|^2_{diag(w_{ij})}$$

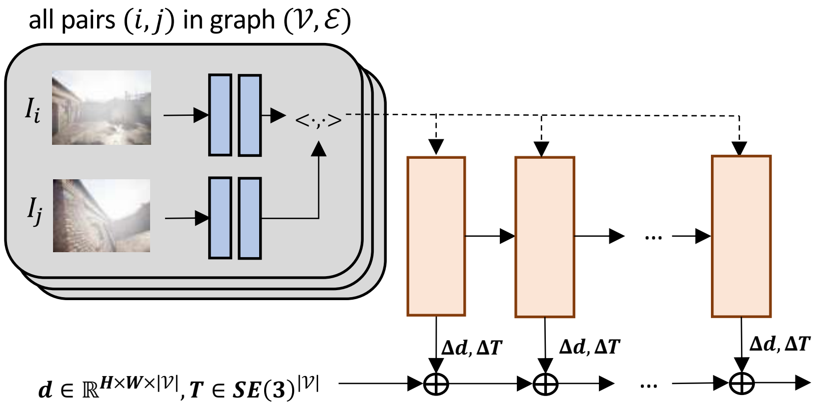
$\xrightarrow{\text{linearize}}$

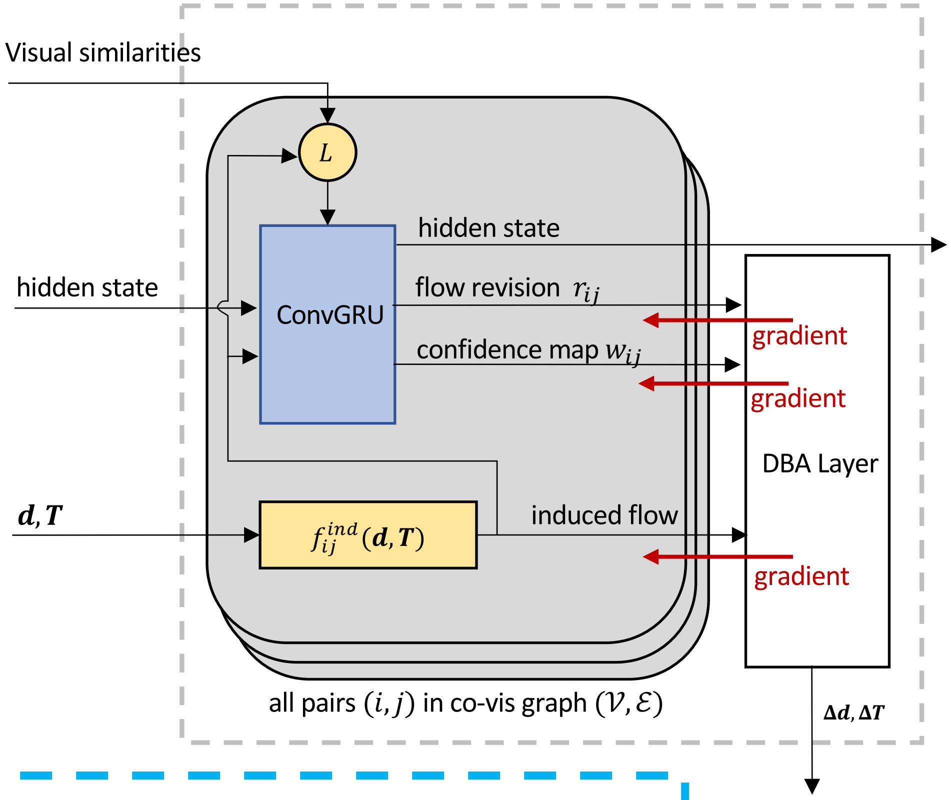
$$\min_{\Delta d,\Delta T} \sum_{(i,j)\in\mathcal{E}} \left\| r_{ij} - \frac{\partial f_{ij}^{ind}(d,T)}{\partial d}\Delta d - \frac{\partial f_{ij}^{ind}(d,T)}{\partial T}\Delta T)\right\|^2_{diag(w_{ij})}$$

Linear least squares
Differentiable closed-form solution
i.e. Gauss-Newton step

DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer





$$\min_{\Delta\boldsymbol{d},\Delta\boldsymbol{T}} \sum_{(i,j)\in\mathcal{E}} \left\| f_{ij}^{ind}(\boldsymbol{d},\boldsymbol{T}) + r_{ij} - f_{ij}^{ind}(\boldsymbol{d}+\Delta\boldsymbol{d},\boldsymbol{T}+\Delta\boldsymbol{T}) \right\|^2_{diag(w_{ij})}$$

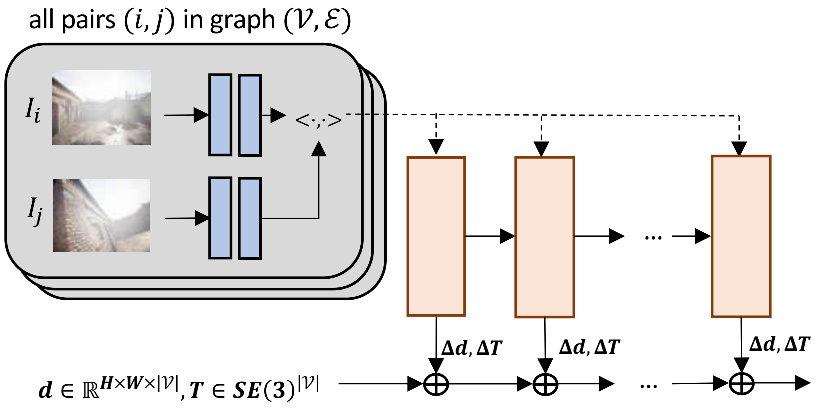
linearize $\longrightarrow$

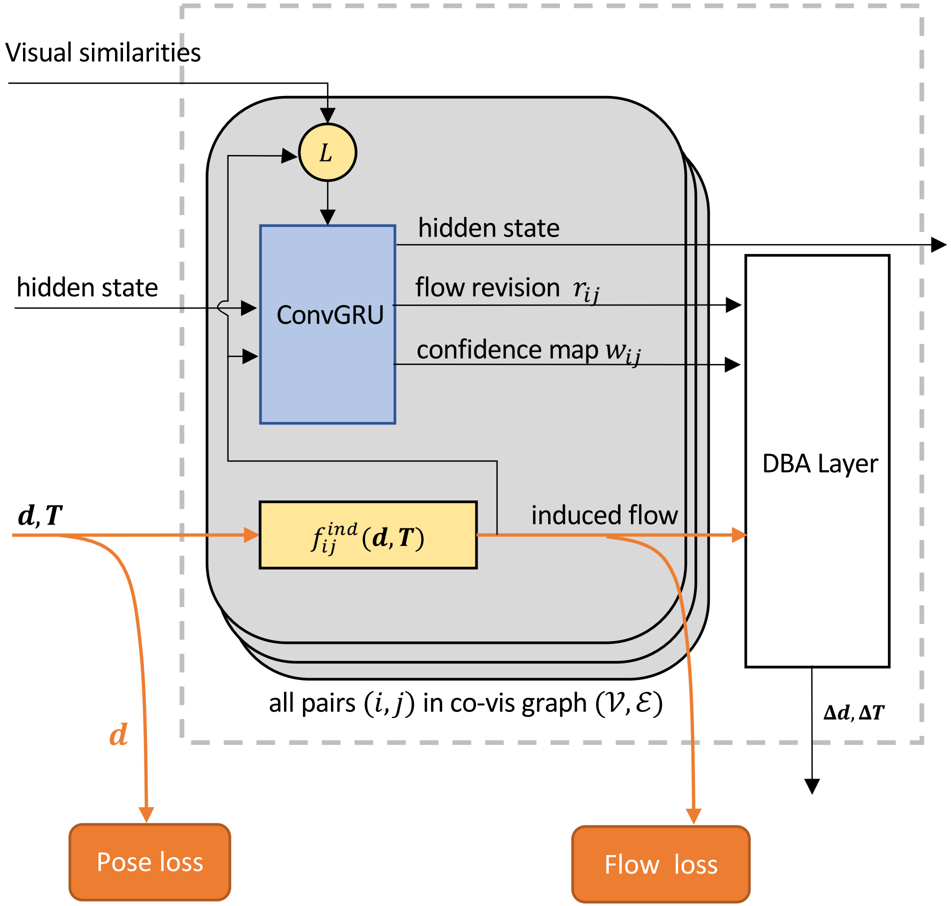
$$\min_{\Delta\boldsymbol{d},\Delta\boldsymbol{T}} \sum_{(i,j)\in\mathcal{E}} \left\| r_{ij} - \frac{\partial f_{ij}^{ind}(\boldsymbol{d},\boldsymbol{T})}{\partial \boldsymbol{d}}\Delta\boldsymbol{d} - \frac{\partial f_{ij}^{ind}(\boldsymbol{d},\boldsymbol{T})}{\partial \boldsymbol{T}}\Delta\boldsymbol{T}) \right\|^2_{diag(w_{ij})}$$

Linear least squares
Differentiable closed-form solution
i.e. Gauss-Newton step

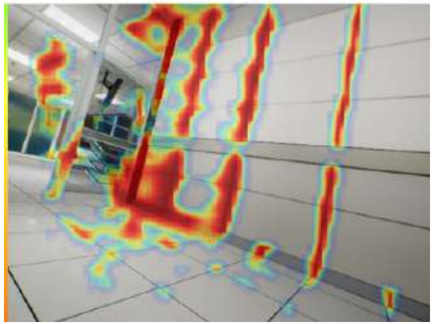
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

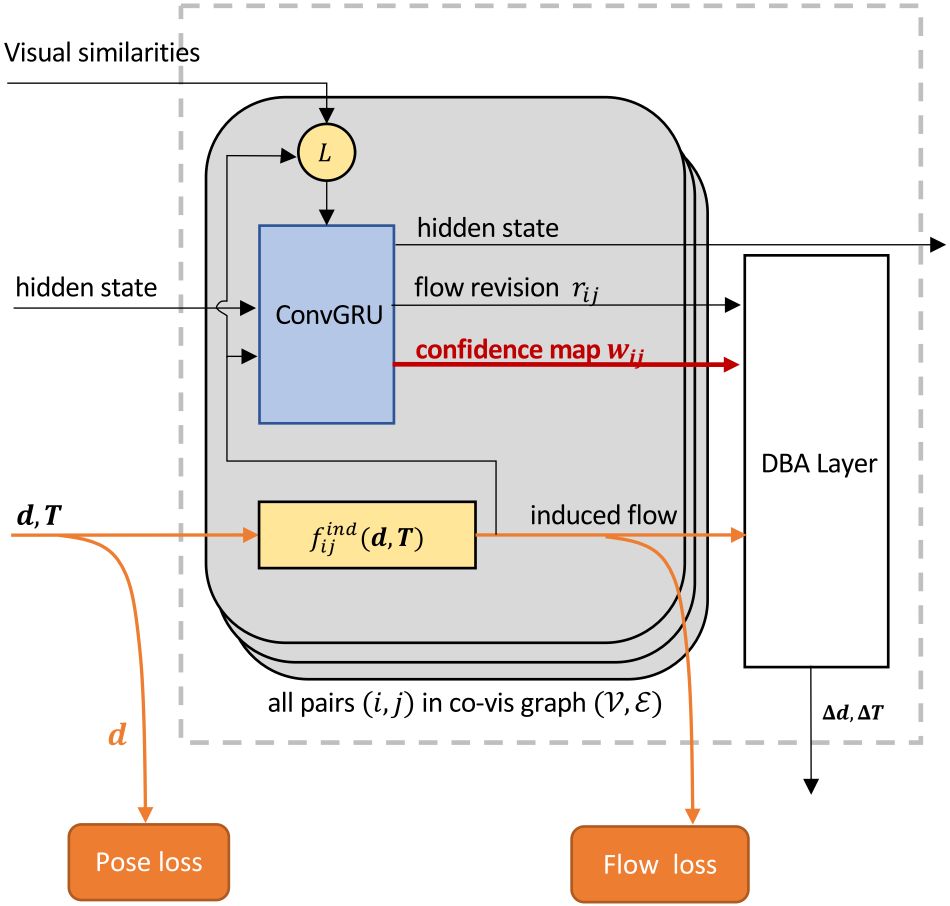




horizontal flow confidence

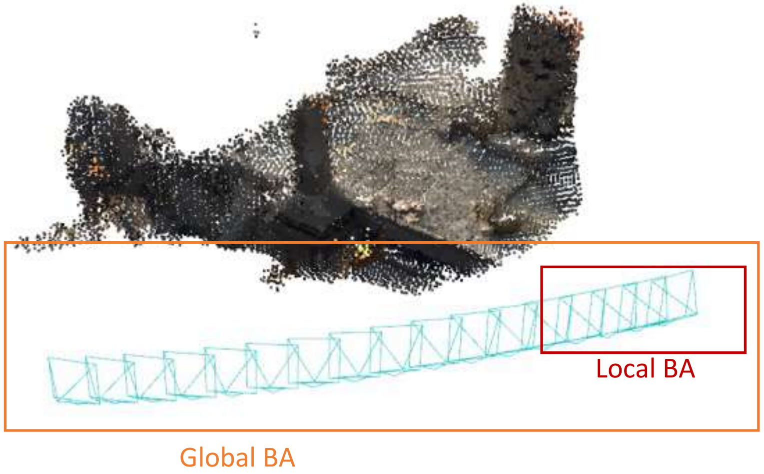


No direct supervision



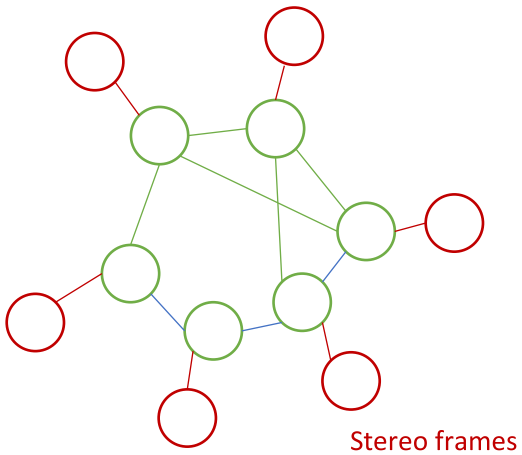
DROID-SLAM: Full System

- **Frontend**: feature extraction, local bundle adjustment
- **Backend**: global bundle adjustment
- **Building covisibility graph**: thresholding inter-frame flow magnitude
- Real time on 2 3090 GPUs (with custom GPU kernels)
- Trained only on monocular input



DROID-SLAM: extension to stereo and RGB-D

- **Stereo**: double the frames in graph, fixing relative poses between left & right frames



Co-visibility graph for stereo

DROID-SLAM: extension to stereo and RGB-D

- **Stereo**: double the frames in graph, fixing relative poses between left & right frames
- **RGB-D**: still estimate depth, but use sensor depth as a prior in DBA layer
  - Sensor depth can have noise and missing observations

Co-visibility graph for stereo

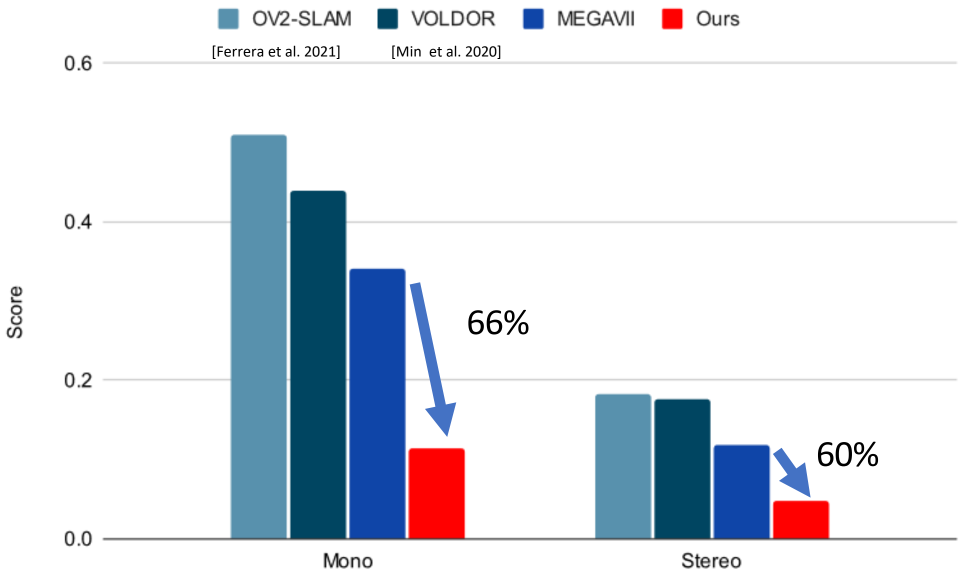
$$\min_{\Delta \boldsymbol{d}, \Delta \boldsymbol{T}} \sum_{(i,j)\in\mathcal{E}} \left\| f_{ij}^{ind}(\boldsymbol{d}, \boldsymbol{T}) + r_{ij} - f_{ij}^{ind}(\boldsymbol{d} + \Delta \boldsymbol{d}, \boldsymbol{T} + \Delta \boldsymbol{T}) \right\|_{diag(w_{ij})}^2 + \left\| \boldsymbol{d} + \Delta \boldsymbol{d} - \hat{\boldsymbol{d}} \right\|^2$$

Sensor depth $\hat{\boldsymbol{d}}$ as a prior

DBA layer

No retraining needed for stereo or RGB-D

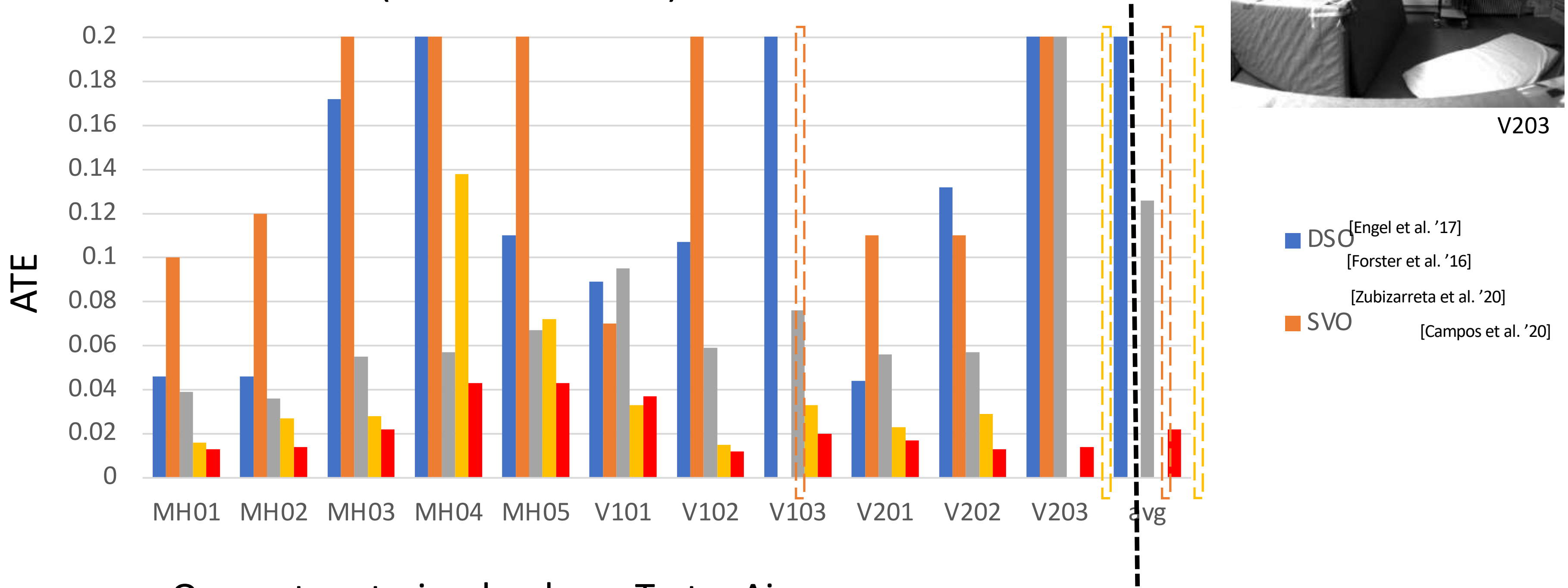
TartanAir – SLAM Challenge [Wang et al. 2020]



- Our system trained on TartanAir (training split) with monocular input
- 66% lower error on monocular, 60% lower error on stereo, 16x faster

EuRoC MAV (Monocular)

[Burri et al. 2016]

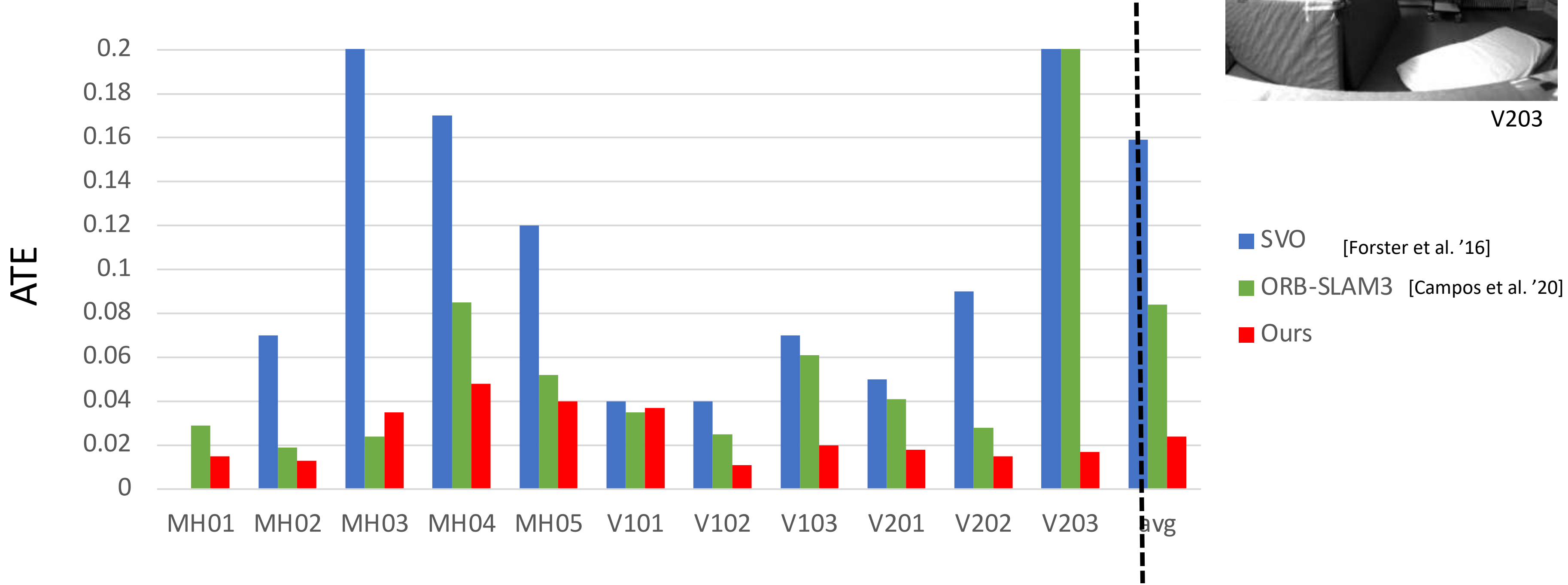


V203

- Our system trained only on TartanAir
- ***82% less error*** among methods with zero failures
- ***43% less error*** than ORB-SLAM3 on its successful sequences

EuRoC MAV (Stereo)

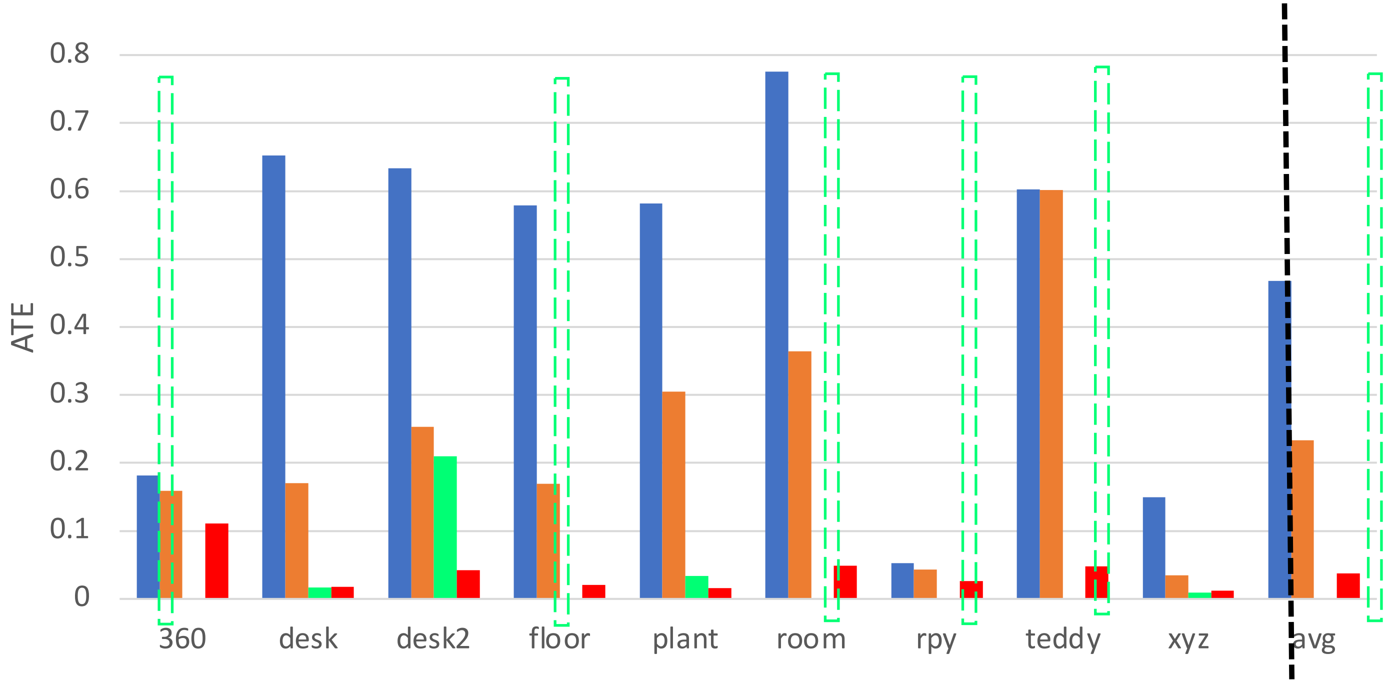
[Burri et al. 2016]



V203

- Our system trained only on monocular TartanAir
- ***71% less error*** than ORB-SLAM3

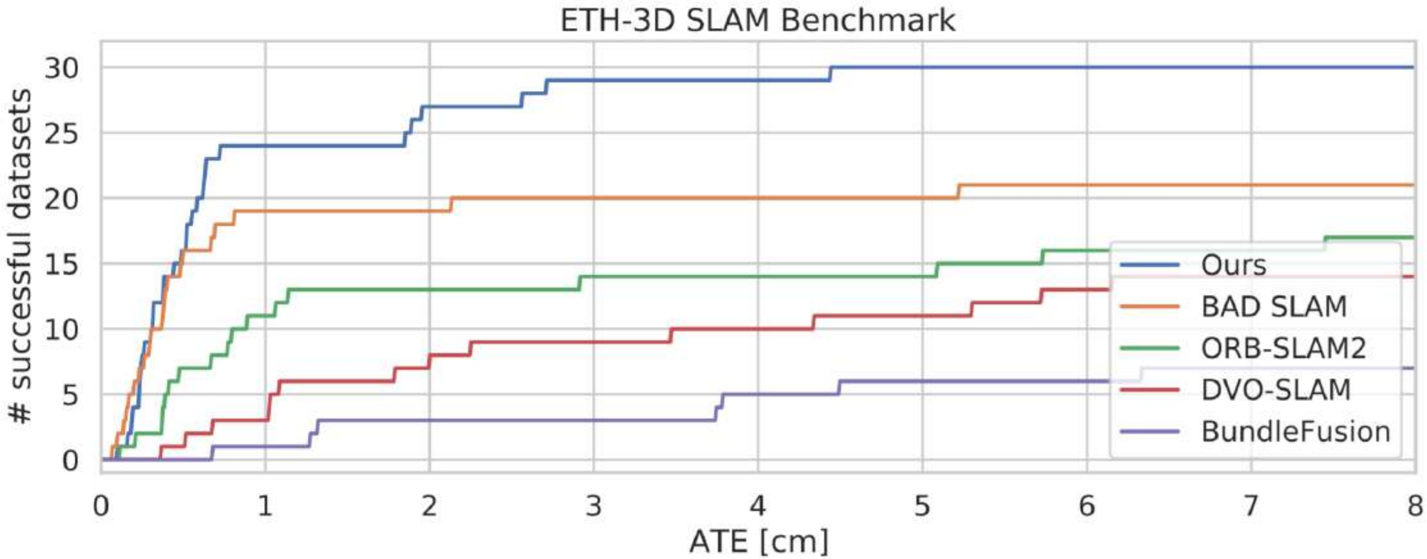
TUM-RGBD (Monocular) [Sturm et al. 2012]



room

- Our system trained only on monocular TartanAir
- ***83% lower error*** than DeepFactors

ETH-3D SLAM (RGB-D)



[Schops et al. '19]
[Mur-Artal & Tardos '17]
[Kerl et al. '13]
[Dai et al. '17]

- Our system trained only on monocular TartanAir
- Ranks 1st, 35% better AUC
- Successfully track 30/32 RGB-D datasets, next best method tracks 19/32

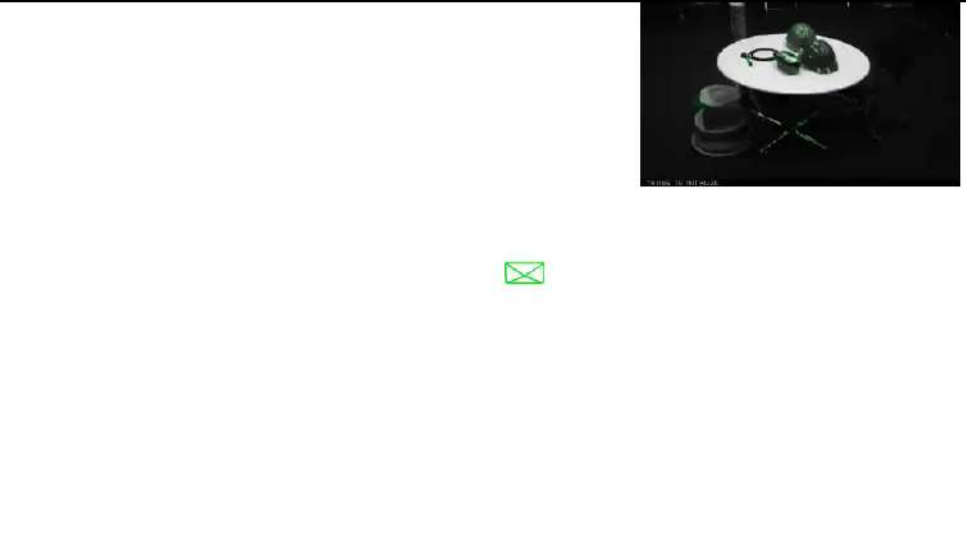
Strong Generalization

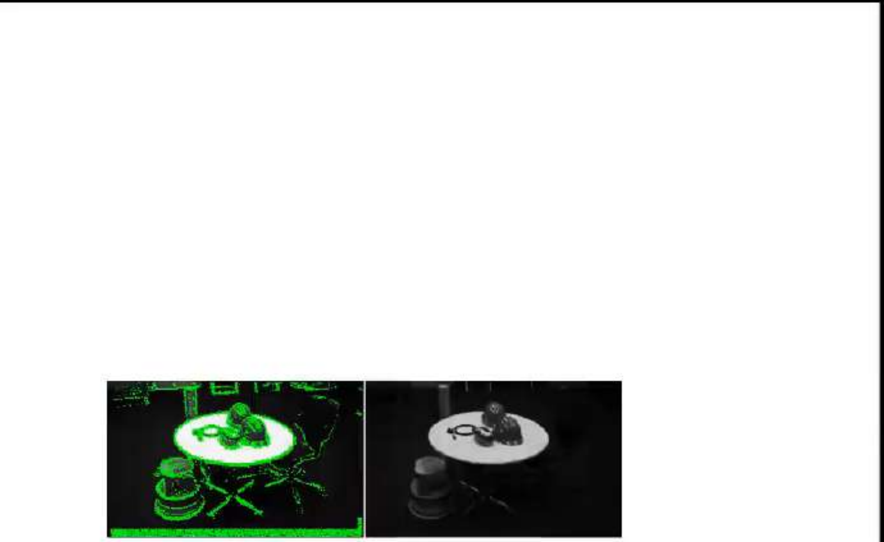
All results, across datasets and modalities (monocular, stereo, RGB-D),

are by ***a single model***, trained only once, on synthetic data.

[ORB-SLAM3, Campos et al]

[DSO, Engel et al]





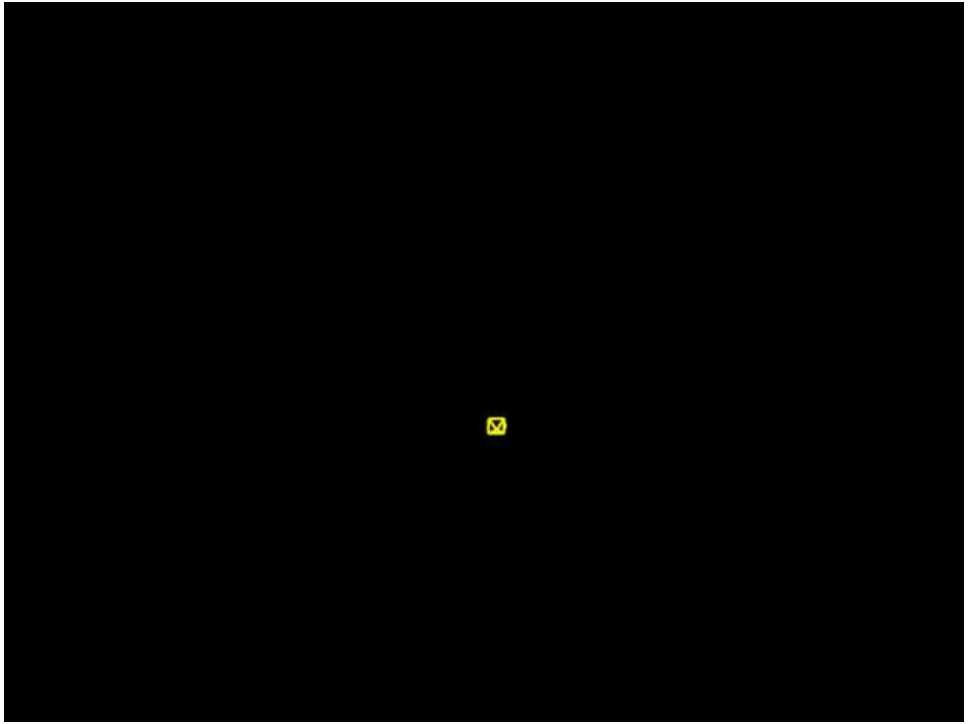

Tanks and Temples

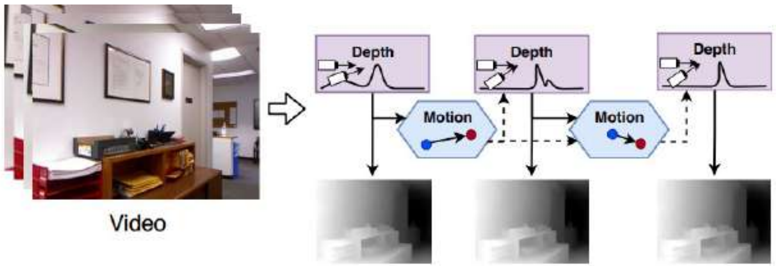
iPhone 12

iPhone 12

iPhone 12

DeepV2D [ICLR 2020]: Video to Depth



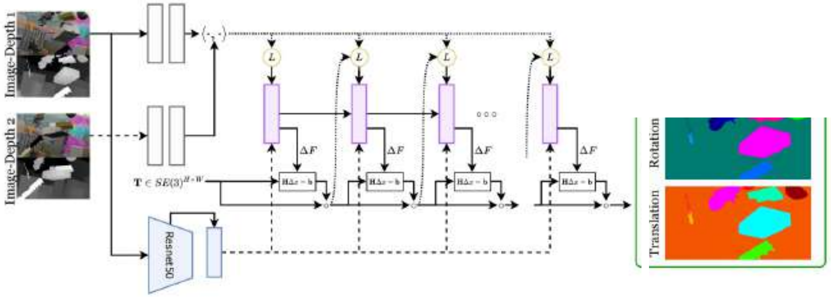


Recurrent unit + analytical layer (PnP)

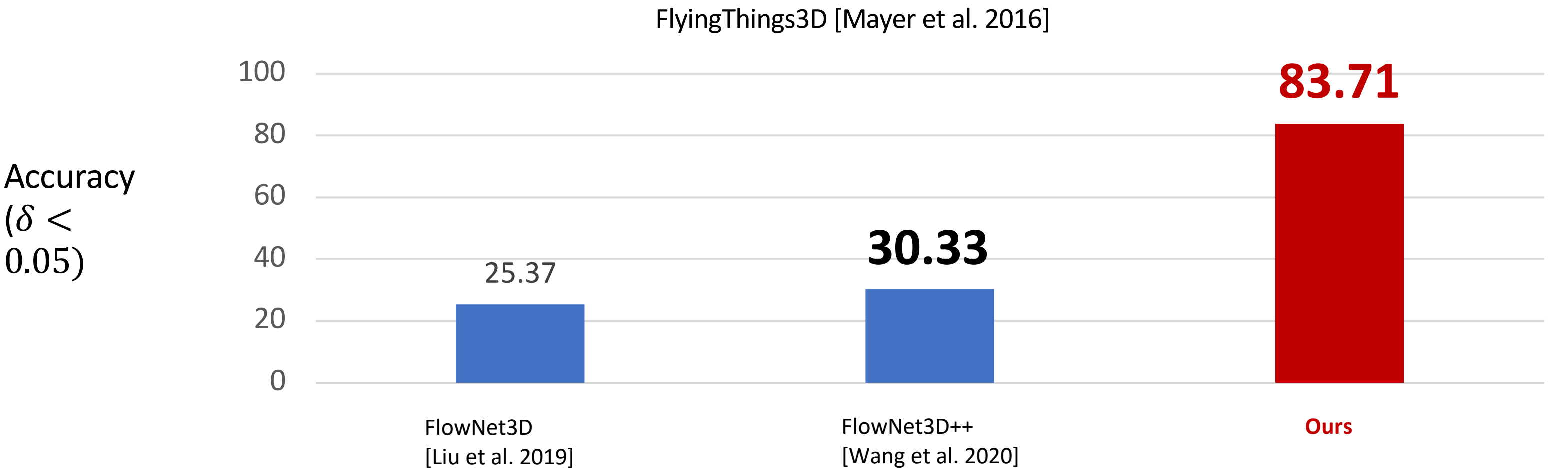
53% less error over prior SOTA on NYU Depth

RAFT-3D [CVPR 2021]: Scene Flow

Input: RGB-D video of dynamic scene
Output: per-pixel 3D motion

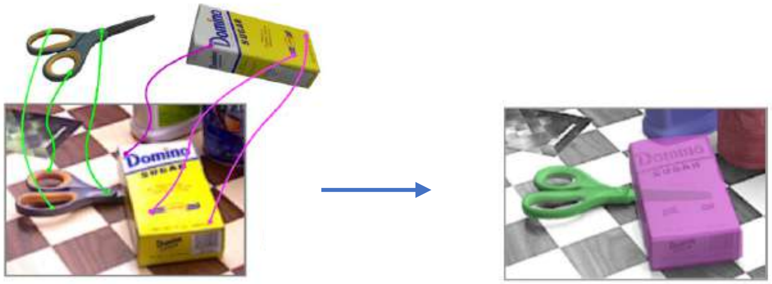


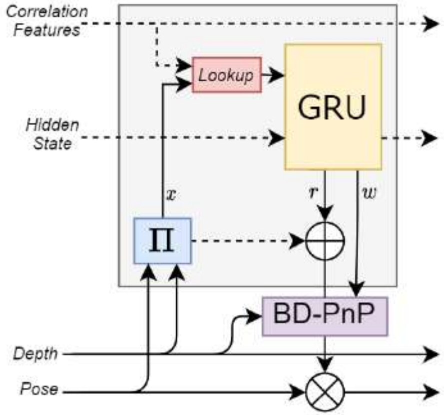
Recurrent unit + analytical layer (DBA w/ soft pixel grouping)



6D Multi-Object Pose [Lipson, Teed, Deng, CVPR 2022]

Input: RGB-D + known 3D models
Output: 6D object poses





Recurrent unit + analytical layer (Bidirectional PnP)

SOTA on the BOP benchmark (YCB-V, T-LESS, LINEMOD-Occluded)