CS231N: Low-Level Vision

Jia Deng
Optical Flow

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

Robotic Applications:
- Self-driving cars (Waymo)
- Everydayrobots.com
- Project starline (Google)

AR/VR Applications:
- Hololens (Microsoft)

Intersecting with:
- 3D Vision
- Graphics
Optical Flow as Optimization

- Objective: appearance constancy + plausibility of flow field

\[ E(\Delta x) = \text{Distance}(I(x_i), I(x_i + \Delta x_i)) + \text{Regularization}(I, \Delta x) \]
Optical Flow

- Classical approaches:

The Model of Horn and Schunck [1]

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

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

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


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, ...

\[
\min_{u,v} \left\{ E = \int_{\Omega} |\nabla u| + |\nabla v| \, dx + \lambda \int_{\Omega} |\rho(u, v)| \, dx \right\}
\]

+ Allows for discontinuities in the flow field
+ Robust to some extent to outliers in the OFC
+ Still convex
- Much harder to solve

Optical Flow

- Classical approaches: DeepFlow

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)
Optical Flow

- Deep Learning: FlowNet

FlowNet C (Correlation) architecture

- Input: two images ([image(t), image(t-1)])
- Correlation layer calculating “correlation” of two images
  - NO trainable weights
  - Inner product
  - Neighborhood size = 41 x 41
  - Striding = 2
  - Z-dimension of output
    \[ \frac{(41 - 1)}{2} + 1 = 21^2 = 441 \]

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

- Deep Learning: FlowNet

Refinement layers in FlowNet S/C

1. 4 De-convolution layers & 4 Upsampled prediction layers
   - De-convolution: Transposed convolution + LeakyReLU
   - Upsampled prediction: Transposed convolution (evaluated)
   - De-conv + Previous feature map + Upsampled prediction

2. Bilinear upsampling (4x)
   - Cheaper & Adding more refinement layers did not improve the result

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

- Deep Learning: FlowNet 2.0

Deep Learning and Optical Flow

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

[FlowNet] [Dosovitskiy et al., 2015]
[PWC-Net] [Sun et al., 2018]

[Ranjan and Black, 2017] [Maurer and Bruhn, 2018] [Bar-Haim and Wolf, 2020] [Yang and Ramanan, 2018]
[Ilg et al., 2017] [Ilg et al., 2017] [Zhao et al., 2020] [Lu et al., 2020]
[Hui et al, 2018] [Neoral et al, 2018] [Z Yin et al, 2019]
Optical Flow

- Deep Learning: PWC-Net

Optical Flow

- Deep Learning: VCN

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:

81D 81D 81D 81D

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 \((w_1, ..., w_9)\)
Convex Upsampling

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

• Supervised directly on sequence of full resolution flow fields

\[
Loss = \sum_{i}^{N} \frac{1.25^i}{1.25^N} \| f_{gt} - f_i \|_1
\]
RAFT versus VCN

RAFT [Teed & Deng, 2020]

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

VCN [Yang & Ramanan, 2019]

• Construct 4D cost volume
• 4D convolution on entire cost volume
<table>
<thead>
<tr>
<th>Method</th>
<th>% of Outliers</th>
</tr>
</thead>
<tbody>
<tr>
<td>LiteFlowNet2 [Hui et al 2019]</td>
<td>16%</td>
</tr>
<tr>
<td>PWC-Net+ [Sun et al, 2019]</td>
<td>16%</td>
</tr>
<tr>
<td>IRR-PWC [Hur and Roth, 2019]</td>
<td>16%</td>
</tr>
<tr>
<td>ScopeFlow [Bair-Haim and Wolf, 2019]</td>
<td>16%</td>
</tr>
<tr>
<td>HD3 [Yin et al, 2019]</td>
<td>16%</td>
</tr>
<tr>
<td>VCN [Yang and Ramanan, 2019]</td>
<td>16%</td>
</tr>
<tr>
<td>MaskFlowNet [Zhao et al, 2020]</td>
<td>16%</td>
</tr>
<tr>
<td>Ours</td>
<td>16%</td>
</tr>
</tbody>
</table>

Cross-Dataset Generalization

![Sintel (clean) and KITTI 2015 bar charts]

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

![Graph 1](image1.png)

- **Error** vs. **# of Flow Updates**
- **Sintel (train) Pass**
  - Clean
  - Final

![Graph 2](image2.png)

- **Update Magnitude** vs. **# of Flow Updates**
- **Sintel (train) Pass**
  - Clean
  - Final
Convergence Visualized

1 Iteration  2 Iterations  5 Iterations  32 Iterations
RAFT can recover the motion of small, fast moving objects

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

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

Robust Vision Challenge ECCV 2020

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
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

\[ \text{disparity} = x - x' = \frac{B \cdot f}{z} \]
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 $\text{depth}(x) = Bf/(x - x')$
Failures of correspondence search

Textureless surfaces

Occlusions, repetition
Failures of correspondence search

- Textureless surfaces
- Occlusions, repetition
- Non-Lambertian surfaces, specularities
Results with window search

Window-based matching

Ground truth
Better methods exist...

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

\[ E(D) = \sum_i \left( W_1(i) - W_2(i + D(i)) \right)^2 + \lambda \sum_{\text{neighbors } i, j} \rho(D(i) - D(j)) \]
Stereo matching as energy minimization

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

- **data term**
- **smoothness term**
Stereo matching as energy minimization

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

- **Data term**
- **Smoothness term**
Stereo matching as energy minimization

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

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

Active stereo with structured light

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

Active stereo with structured light

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

Kinect: Structured infrared light

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

Middlebury: bad 1.0 (%)

- HSM-Net [Yang et al.]
- LEAStereo [Cheng et al.]
- MC-CNN [Drouyer et al.]
- HITNet [Tankovich et al.]
- Ours

[Scharstein et al., 2014]
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

\[
X \xrightarrow{\mathbf{T}} X' \quad \mathbf{f} = F(X, T)
\]

\[
\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \in SE(3)
\]

Step 1. Estimate 2D flow \( f \)

\( \rightarrow \) Match pixels by manual features

Step 2. Solve for 3D given flow

\[
\min_{X,T} \| f - F(X, T) \|^2_{X,T}
\]

**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
DROID-SLAM

DROID: Differentiable Recurrent Optimization-Inspired Design

- **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: Differentiable Recurrent Optimization-Inspired Design

\[ T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \in SE(3) \]

\[ f = F(X, T) \]

- 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

End-to-end neural architecture
DROID-SLAM

DROID: Differentiable Recurrent Optimization-Inspired Design

\[ T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \in SE(3) \]

\[ f = F(X, T) \]

**Step 1. Estimate 2D flow** \( f \)

\( \rightarrow \) 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

End-to-end neural architecture
Estimate 2D motion (optical flow)

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

DROID: Differentiable Recurrent Optimization-Inspired Design

\[
T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \in SE(3)
\]

\[ f = F(X, T) \]

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

End-to-end neural architecture

RAFT [TD, ECCV20]
DROID-SLAM

DROID: Differentiable Recurrent Optimization-Inspired Design

\[ T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \in SE(3) \]

\[ f = F(X, T) \]

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

End-to-end neural architecture
Dense Bundle Adjustment (DBA)

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

```
min | \mathbf{d}, \mathbf{T} |
\quad \not\in \mathcal{E} f_i, f_j, (\mathbf{d}, \mathbf{T})
```

Predicted flow $f_{ij}^{\text{pred}}$

Co-visibility graph

frame $i$

predicted flow $f_{ij}^{\text{pred}}$

frame $j$
Dense Bundle Adjustment (DBA)

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

• **Want:** depth maps \(d = (d_1, ..., d_i, ...)\), camera poses \(T = (T_1, ..., T_i, ...)\)
Dense Bundle Adjustment (DBA)

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

\[
\min_{d,T} \sum_{(i,j) \in \mathcal{E}} \left\| f_{ij}^{\text{pred}} - f_{ij}^{\text{ind}}(d, T) \right\|^2
\]

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

\(f_{ij}^{\text{pred}}\) predicted flow
\(f_{ij}^{\text{ind}}\) induced flow

---

Co-visibility graph

Frame $i$

Predicted flow $f_{ij}^{\text{pred}}$

Frame $j$
Dense Bundle Adjustment (DBA)

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

\[
\min_{d,T} \sum_{(i,j) \in \mathcal{E}} \left\| f_{ij}^{\text{pred}} - f_{ij}^{\text{ind}}(d,T) \right\|^2
\]
Dense Bundle Adjustment (DBA)

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

\[
\min_{d,T} \sum_{(i,j) \in E} \left\| f_{ij}^{pred} - f_{ij}^{ind}(d, T) \right\|^2
\]

- 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 poorly, because of outliers: visibility, dynamic objects, prediction error
- How to exclude outliers?

New Idea: Integrating RAFT and DBA iterations

\[
\min_{x, y} E(x, y)
\]

Feedback

\[
T^*, d^*
\]
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

\[
d \in \mathbb{R}^{H \times W \times |V|}, T \in SE(3)^{|V|}
\]
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

\[ d \in \mathbb{R}^{H \times W \times |V|}, T \in SE(3)^{|V|} \]

\[ \Delta d, \Delta T \]

\[ \Delta d, \Delta T \]

\[ \Delta d, \Delta T \]

\[ \text{Visual similarities} \]

\[ \text{hidden state} \]

\[ \text{flow revision } r_{ij} \]

\[ \text{confidence map } w_{ij} \]

\[ f_{ij}^{\text{ind}}(d, T) \]

\[ \text{all pairs } (i, j) \text{ in co-vis graph } (V, E) \]

\[ \Delta d, \Delta T \]
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

\[ d \in \mathbb{R}^{H \times W \times |V|}, T \in SE(3)^{|V|} \]

Visual similarities

all pairs \((i, j)\) in graph \((\mathcal{V}, \mathcal{E})\)

\[ \Delta d, \Delta T \]

hidden state

convGRU

flow revision \(r_{ij}\)

confidence map \(w_{ij}\)

induced flow

Visual similarities

all pairs \((i, j)\) in co-vis graph \((\mathcal{V}, \mathcal{E})\)
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

\[ d \in \mathbb{R}^{HWV}, T \in SE(3)^{|V|} \]

**Visual similarities**

- Recurrent Updates + Analytical Layer

DBA Layer

ConvGRU

hidden state

flow revision \( r_{ij} \)

confidence map \( w_{ij} \)

\( f_{ij}^{ind}(d, T) \)

induced flow

all pairs \((i, j)\) in co-vis graph \((V, E)\)

\( \Delta d, \Delta T \)
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

\[ d \in \mathbb{R}^{H \times W \times |V|}, T \in SE(3)^{|V|} \]

Visual similarities

ConvGRU

hidden state

flow revision \( r_{ij} \)

confidence map \( w_{ij} \)

DBA Layer

all pairs \((i, j)\) in graph \((V, E)\)

\( f_{ij}^{ind}(d, T) \)

induced flow

all pairs \((i, j)\) in co-vis graph \((V, E)\)
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

*all pairs (i, j) in graph (V, E)*

\[ d \in \mathbb{R}^{H \times W \times |V|}, T \in SE(3)^{|V|} \]

\[ \Delta d, \Delta T \]

\[ \Delta d, \Delta T \]

\[ \Delta d, \Delta T \]

\[ \Delta d, \Delta T \]

Visual similarities

hidden state

flow revision \( r_{ij} \)

confidence map \( w_{ij} \)

DBA Layer

induced flow

all pairs (i, j) in co-vis graph (V, E)

\[ \Delta d, \Delta T \]
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

\[
\begin{align*}
\min_{\Delta d, \Delta T} \sum_{(i,j) \in \mathcal{E}} \left\| f_{ij}^{ind}(d, T) + r_{ij} - f_{ij}^{ind}(d + \Delta d, T + \Delta T) \right\|_2^2 \\
\end{align*}
\]
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

\[ \min_{\Delta d, \Delta T} \sum_{(i,j) \in \mathcal{E}} \left\| f_{ij}^{ind}(d, T) + r_{ij} - f_{ij}^{ind}(d + \Delta d, T + \Delta T) \right\|_2^{2} \text{diag}(w_{ij}) \]
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

\[
\begin{align*}
\text{all pairs } (i, j) \text{ in graph } (\mathcal{V}, \mathcal{E}) \\
\end{align*}
\]

\[
\begin{align*}
d \in \mathbb{R}^{H \times W \times |\mathcal{V}|}, T \in SE(3)^{|\mathcal{V}|} \\
\end{align*}
\]

\[
\begin{align*}
\min_{\Delta d, \Delta T} \sum_{(i, j) \in \mathcal{E}} \left\| f_{ij}^{\text{ind}}(d, T) + r_{ij} - f_{ij}^{\text{ind}}(d + \Delta d, T + \Delta T) \right\|_{\text{diag}(w_{ij})}^2
\end{align*}
\]

Current induced flow between frame \( i, j \)
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

\[ d \in \mathbb{R}^{H \times W \times |V|}, T \in SE(3)^{|V|} \]

\[ \Delta d, \Delta T \]

\[ \sum_{(i,j) \in \mathcal{E}} \left\| f_{ij}^{ind}(d, T) + r_{ij} - f_{ij}^{ind}(d + \Delta d, T + \Delta T) \right\|_2^2 \text{diag}(w_{ij}) \]

Current induced flow between frame \( i,j \)

Flow revision

Visual similarities
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

\[
d \in \mathbb{R}^{H \times W \times |\mathcal{V}|}, T \in SE(3)^{|\mathcal{V}|}
\]

\[
\Delta d, \Delta T \quad \Delta d, \Delta T \quad \ldots
\]

\[
\min_{\Delta d, \Delta T} \sum_{(i,j) \in \mathcal{E}} \left\| f_{ij}^{\text{ind}}(d, T) + r_{ij} - f_{ij}^{\text{ind}}(d + \Delta d, T + \Delta T) \right\|_2^2_{\text{diag}(w_{ij})}
\]

Current induced flow between frame \(i, j\)

Flow revision

New induced flow between frame \(i, j\)
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

Current induced flow between frame $i,j$

new induced flow between frame $i,j$

flow revision
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

\[
d \in \mathbb{R}^{H \times W \times |V|}, T \in SE(3)^{|V|}
\]

all pairs \((i, j)\) in graph \((V, E)\)

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

\[
\min_{\Delta d, \Delta T} \sum_{(i,j) \in E} \left\| r_{ij} - \frac{\partial f_{ij}^{ind}(d, T)}{\partial d} \Delta d - \frac{\partial f_{ij}^{ind}(d, T)}{\partial T} \Delta T \right\|_{diag(w_{ij})}^2
\]
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

\[ d \in \mathbb{R}^{H \times W \times |\mathcal{V}|}, T \in SE(3)^{|\mathcal{V}|} \]

all pairs \((i, j)\) in graph \((\mathcal{V}, \mathcal{E})\)

\[ L \]

hidden state

ConvGRU

flow revision \(r_{ij}\)

confidence map \(w_{ij}\)

DBA Layer

induced flow

all pairs \((i, j)\) in co-vis graph \((\mathcal{V}, \mathcal{E})\)

\[ \Delta d, \Delta T \]

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

linearize

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

Linear least squares

Differentiable closed-form solution
i.e. Gauss-Newton step
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer

\[
d \in \mathbb{R}^{H \times W \times |V|}, T \in SE(3)^{|V|}
\]

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

Linear least squares
Differentiable closed-form solution
i.e. Gauss-Newton step
DROID-SLAM: Architecture

- Recurrent Updates + Analytical Layer
DROID-SLAM Architecture

- Iteratively updating all depth and poses
- Analytical constrained updates

\[
\begin{align*}
\Delta d, \Delta T \\
\end{align*}
\]

- DBA Layer
  - induced flow
  - flow revision \( r_{ij} \)
  - confidence map \( w_{ij} \)

No direct supervision

Visual similarities

- horizontal flow confidence

Confidence

ConvGRU

all pairs \((i, j)\) in co-vis graph \((\mathcal{V}, \mathcal{E})\)

\( f_{ij}^{\text{ind}}(d, T) \)

\( d \)

\( d, T \)

\( L \)

\( \frac{\%}{\%} \)

Pose loss

Flow loss

\( \Delta d, \Delta T \)
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

No retraining needed for stereo or RGB-D
Our system trained on TartanAir (training split) with monocular input
• 66% lower error on monocular, 60% lower error on stereo, 16x faster
• Our system trained only on TartanAir
• **82% less error** among methods with zero failures
• **43% less error** than ORB-SLAM3 on its successful sequences
• Our system trained only on monocular TartanAir
• 71% less error than ORB-SLAM3
Our system trained only on monocular TartanAir

83% lower error than DeepFactors
• 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.
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

Accuracy ($\delta < 0.05$)

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>FlowNet3D++ [Wang et al. 2020]</td>
<td>30.33</td>
</tr>
<tr>
<td>Ours</td>
<td>83.71</td>
</tr>
</tbody>
</table>

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

FlyingThings3D [Mayer et al. 2016]
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)