Learning Deep Visual SLAM
Frontends: SuperPoint++

Tomasz Malisiewicz
November 2, 2019
Research @ Magic Leap, Inc.
Main ideas

- 1. **Deep SLAM frontends and SuperPoint**: the tricks you need to know
- 2. Using VO/SLAM to train deep convolutional frontends
- 3. Quō vādis Visual SLAM? Some interesting and open problems in SLAM
Two parts of Visual SLAM

- **Frontend**: Image inputs
  - Deep Learning success: Images + ConvNets
- **Backend**: Optimization over pose and map quantities
  - Use Bundle Adjustment

Photo Credit: Cadena et al 2016
SuperPoint: A Deep SLAM Front-end

- Powerful fully convolutional design
  - Points + descriptors computed jointly, **No Patches**
  - Share VGG-like backbone
- Designed for real-time on a GPU
  - Medium-sized backbone
  - Tasks share ~90% of compute
Setting up the Training

- Siamese training -> pairs of images
- Descriptor trained via metric learning
  - Straightforward given correspondence
- Keypoints trained via supervised keypoint labels
  - Where do these come from?
How to get Keypoint Labels for Natural Images?

- Need large-scale dataset of annotated images
- Too hard for humans to label
Self-Supervised Training

Synthetic Shapes (has interest point labels)

First train on this

“Homographic Adaptation”

MS-COCO (no interest point labels)

Use resulting detector to label this
Synthetic Training

- Non-photorealistic shapes
- Heavy noise
- Effective and easy

![Synthetic Shapes](image)
Early Version of SuperPoint (MagicPoint)

Corner Detection Average Precision vs Degree of Noise
Synthetic Shapes, 160 x 120, ($\varepsilon = 4$)

Homographic Adaptation

- Simulate planar camera motion with homographies
- Self-labelling technique
  - Suppress spurious detections
  - Enhance repeatable points
Qualitative Illumination Example

SuperPoint

LIFT

SIFT

ORB
Qualitative Viewpoint Example #1

SuperPoint

LIFT

SIFT

ORB
Qualitative Viewpoint Example #2

- **SuperPoint**
- **LIFT**
- **SIFT**
- **ORB**
3D Generalizability of SuperPoint

- Trained+evaluated on planar, does it generalize to 3D?
- “Connect-the-dots” using nearest neighbor matches
- Works across many datasets / input modalities / resolutions!

Freiburg (Kinect)  NYU (Kinect)  MonoVO (fisheye)  ICL-NUIM (synth)

MS7 (Kinect)  KITTI (stereo)
Public Release of SuperPoint

- Sparse Optical Flow Tracker Demo
- Implemented in PyTorch
- Two files, minimal dependencies
- Get up and running in 5 minutes or less!
- Released in July 2018 at 1st Deep Learning for Visual SLAM Workshop

[Image of optical flow tracker output]

github.com/MagicLeapResearch/SuperPointPretrainedNetwork
SuperPointVO

What happens when we combine SuperPoint with a Visual Odometry backend?

DeTone, D., Malisiewicz, T., Rabinovich, A. Self-Improving Visual Odometry In arXiv: 1812.03245
VO Reconstruction on Freiburg-TUM RGBD
‘structure_texture_far’

Top-Down Trajectory

- **Green line**: Groundtruth
- **Blue line**: Estimate
VO Reconstruction on Freiburg-TUM RGBD ‘long_office_household’
Keypoint 2D Locations
Keypoint Stability
Keypoint Descriptors

ConvNet

Convolutional Frontend

Point Tracks

VO Backend

6DOF Trajectory

Labeled Point Tracks

Self-Supervision from VO

Supervision Signal

Input Monocular Sequence

#1
#2
#3
#4

3D Points

stable
unstable
ignore

ConvNet

[see Section 3]

[see Section 4]

[see Section 5]
How Does VO Help Learning?

- Learn correspondence across time
- Learn which points are stable and which are not
VO Stability Ground Truth Videos

t-junctions across depth aka “sliders”

lighting highlights

dynamic motion
How to Use Stability?

- For sufficiently long tracks, look at the reprojection error

\[ X_{\text{stable}} = \begin{cases} 
\text{Stable} & \text{, if reprojection error is } < 1 \text{ pixel} \\
\text{Not Stable} & \text{, if reprojection error is } > 5 \text{ pixels} \\
\text{Ignore} & \text{, else} 
\end{cases} \]

- **Stable Points:** Positives
- **Not Stable Points:** Negatives
- **Other Points:** Ignore
Siamese Training on Sequences

Labeled Sequence

Randomly Select Pair

Random Homography

$H_1$

$H_2$

SuperPointVO

Keypoint Loss

Descriptor Loss

SuperPointVO

Keypoint Loss
Pose Estimation on ScanNet

- Small baseline of ~1 second
Pose Estimation on ScanNet

- Medium baseline of ~2 seconds
Pose Estimation on ScanNet

- Widest baseline of ~3 seconds, biggest performance gap
Comparison to LF-Net

- SuperPointVO latches onto localizable corners
Comparison to SuperPoint

- SuperPointVO gets more wide-baseline matches
Qualitative Stability Results

Lighting Highlight Suppression

T-junction Suppression

Generalization on Freiburg Dataset
Epic Kitchens: Arm & Hand Suppression

Keypoint Stability Classification

Low Stability Heatmap

Stable keypoints are **green**, unstable keypoints shown in **red**

The system learns to reject points on **arms & hands**
Epic Kitchens: Shadow Suppression

Keypoint Stability Classification

Low Stability Heatmap

It also learns to suppress points on shadows

Stable keypoints are green, unstable keypoints shown in red
Training “Scene” Specific SuperPoints

What if our “Scene” is this?
CharucoNet

- We can modify the SuperPoint architecture to detect object specific keypoints
- In this work we trained it on a Charuco Pattern
Training Methodology

- First frame bootstrap with OpenCV detector
- Stationary camera
- Subsequent frames add light change, backgrounds, shadows, etc
CharucoNet can “see” in the dark

Increasingly Dark Images
Deep Matching on top of SuperPoint: How to get better correspondences?
Green/red:
RANSAC inliers/outliers
Green/red:
RANSAC inliers/outliers
Summary

- SuperPoint: A ConvNet Architecture for Visual SLAM
- Self-Supervised Learning Via:
  - Homographies
  - Visual Odometry Backend
- Pattern-specific SuperPoints (CharucoNet) and seeing in the dark
- New experiments with deep nets to get better matches
Quō vādis Visual SLAM?

(some open problems at the intersection of DL and SLAM)

1. Multi-user SLAM: Creating representations/maps that work across a large number of camera types (clients)

2. Integrating object recognition capabilities into SLAM frontends

3. Enabling life-long learning: letting the system automatically improve over time
Thank you

Tomasz Malisiewicz
@quantombone

Daniel DeTone
@ddetone

Paul-Edouard Sarlin
@pesarlin

Follow us on Twitter: