



# SuperPoint: Self-Supervised Interest Point Detection and Description

Daniel DeTone, Tomasz Malisiewicz, Andrew Rabinovich

**Research @ Magic Leap**

*CVPR 2018 Deep Learning for Visual SLAM Workshop*

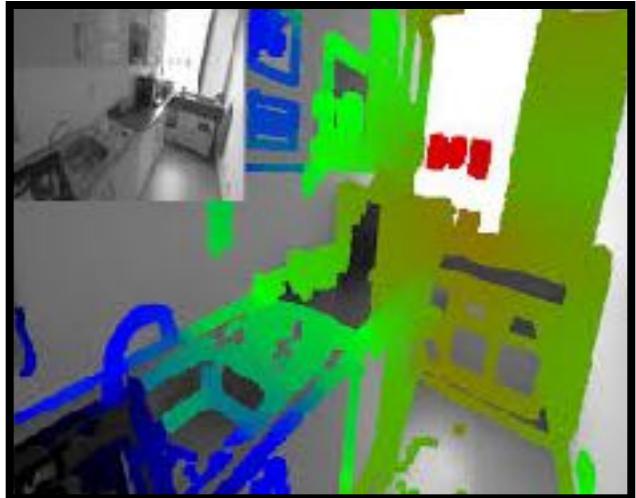
# Main Ideas

- “SuperPoint”
  - A Deep SLAM Frontend
  - Multi-task fully convolutional network
  - Designed for Real-time
- “Homographic Adaptation”
  - Self-supervised recipe to train keypoints
  - Synthetic pre-training
  - Homography-inspired domain adaptation

# 2000-2015 Visual SLAM

- Great Visual SLAM Research
- Real-time systems emerge
- Very few learned components

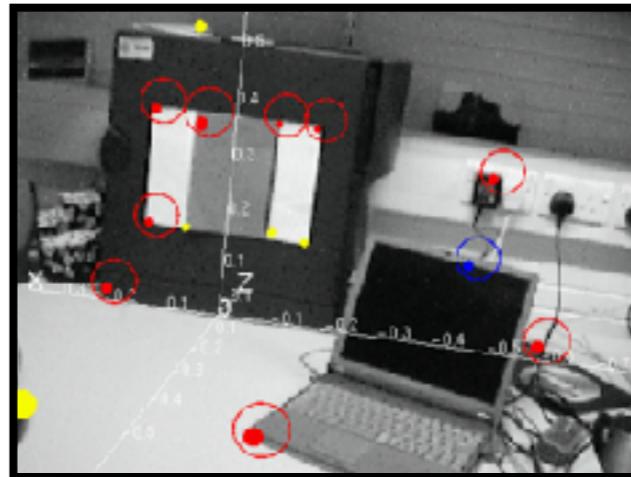
LSD-SLAM



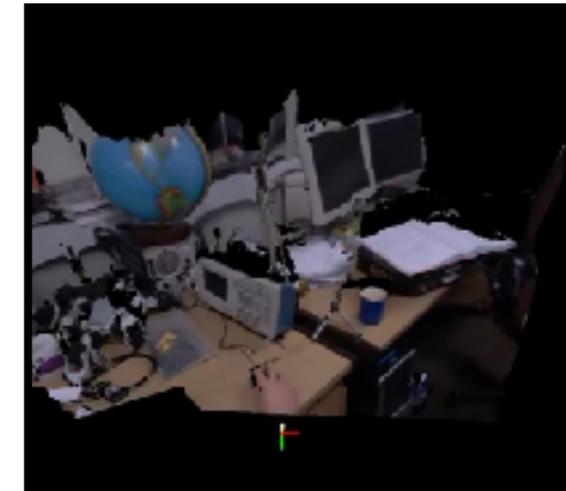
KinectFusion



MonoSLAM



DTAM



PTAM



ElasticFusion



Event-camera SLAM

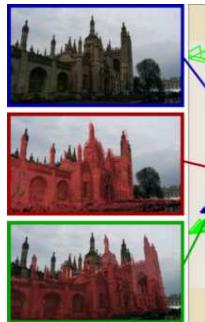


DynamicFusion

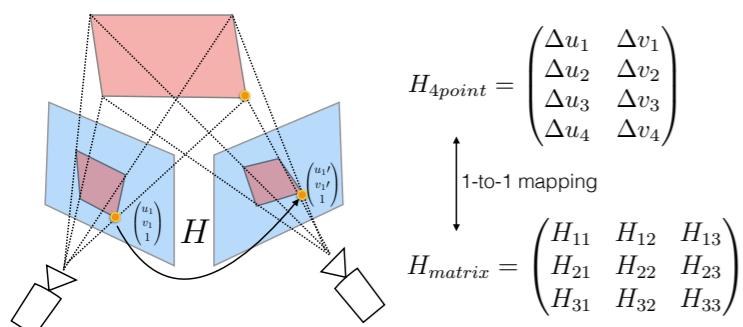


Collage courtesy: Andrew Davison's ICCV 2015 Future of Real-time SLAM workshop talk

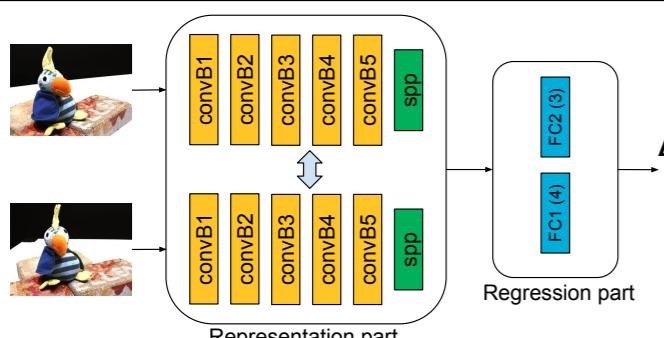
# 2015-2016: Simple End-to-End Deep SLAM?



PoseNet: A Convolutional Network for Real-time 6 DOF Localization



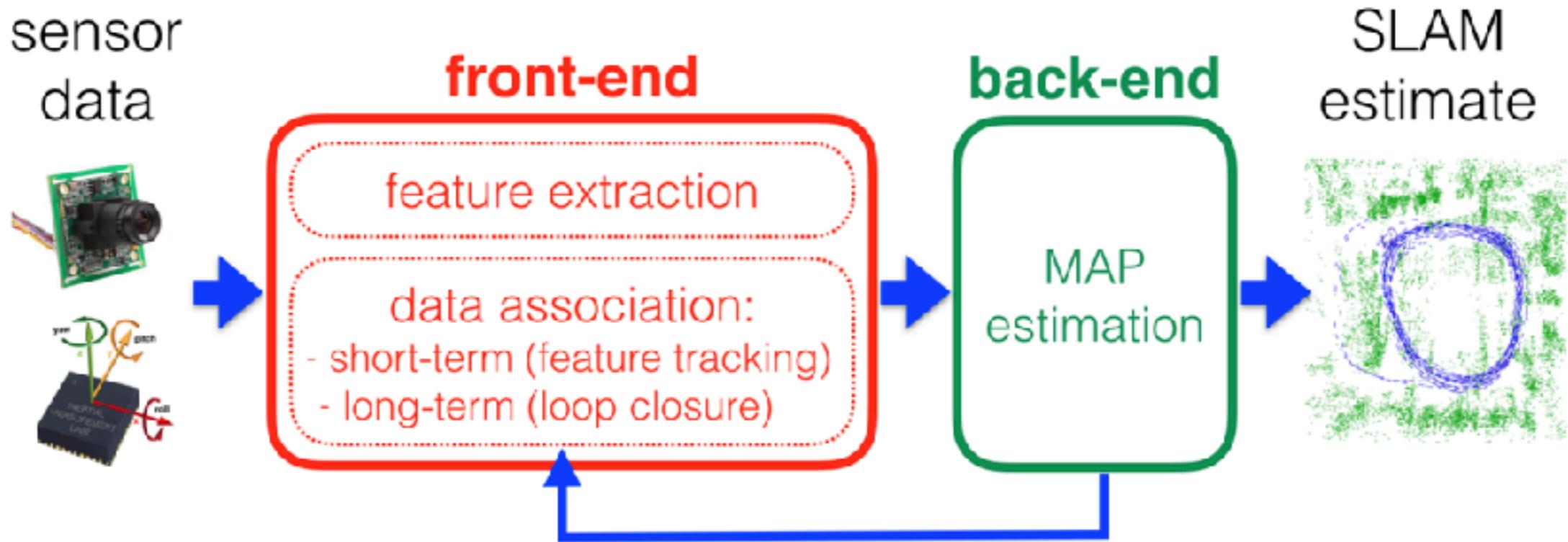
Deep Image Homography Estimation



Relative Camera Pose Estimation Using Convolutional Neural Networks

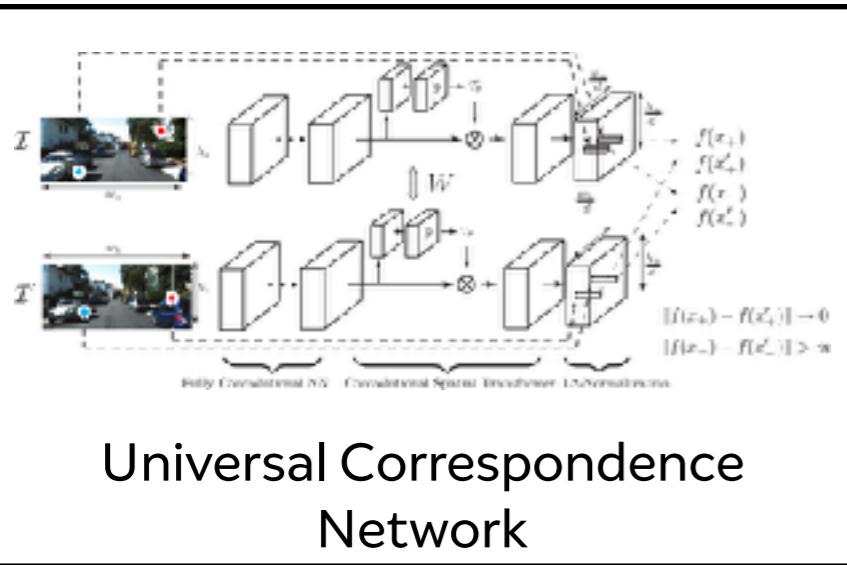
- Deep Learning excitement is very high
- Simple end-to-end setups work across many computer vision tasks
  - Purely data-driven, powerful
  - Very few heuristics / little hand-tuning
- Accuracy not yet competitive
  - Maybe due to lack of large-scale data

# 2017-2018: Splitting Up the Problem

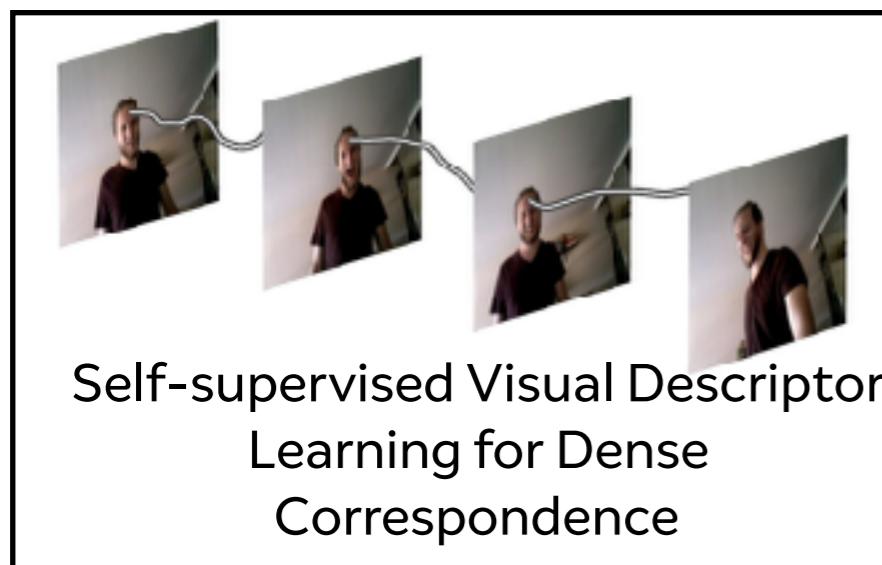
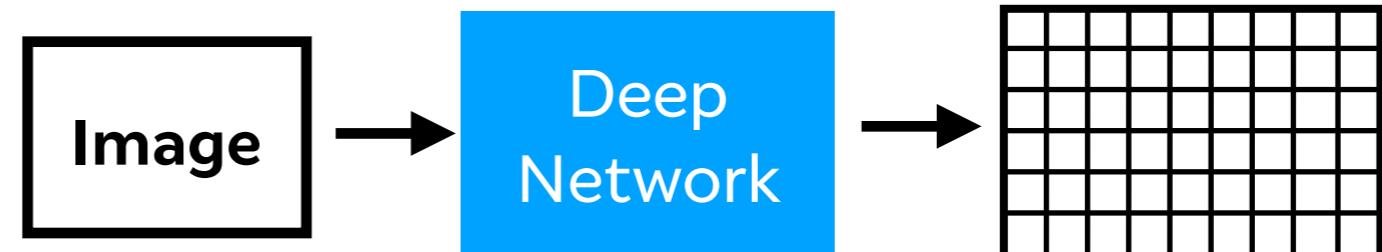


- **Frontend:** Image inputs
  - Deep Learning success: Images + ConvNets
  - Most of current work “deep-ifys” the Frontend -> Focus of this talk
- **Backend:** Optimization over pose and map quantities
  - 2018: Early deep learning work -> Focus of other oral at 12:05pm

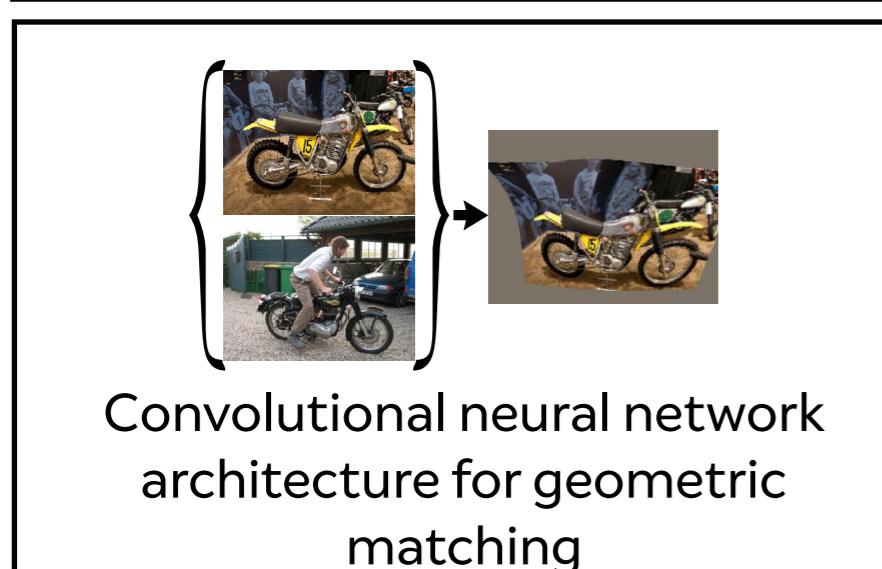
# 2017-2018 Deep Frontends: Dense



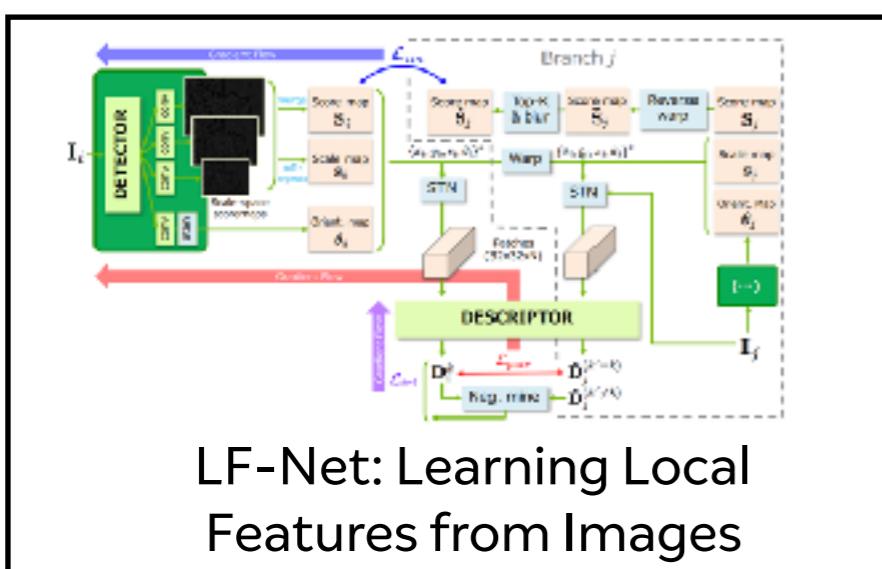
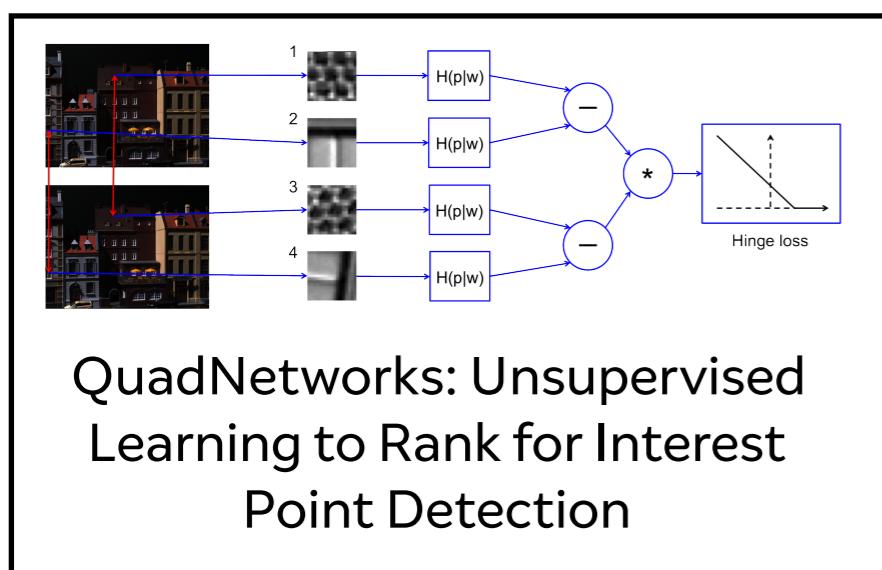
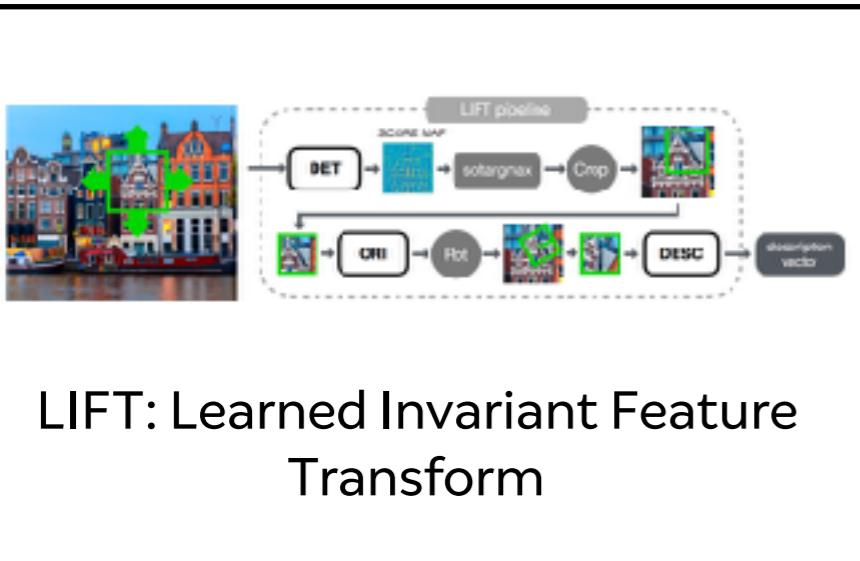
Dense or Semi-Dense Descriptors



- Dense output approaches
- Powerful Matchability
- Not practical in low-compute SLAM systems
- Too expensive for realtime BA

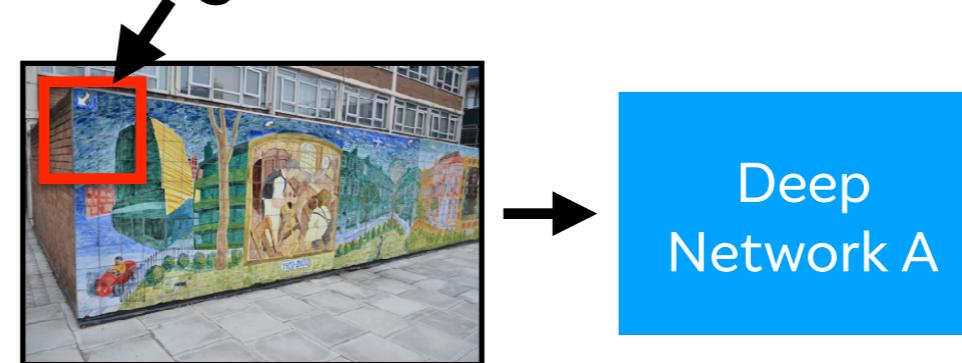


# 2017-2018 Deep Frontends: Sparse



## Existing Patch-based Systems

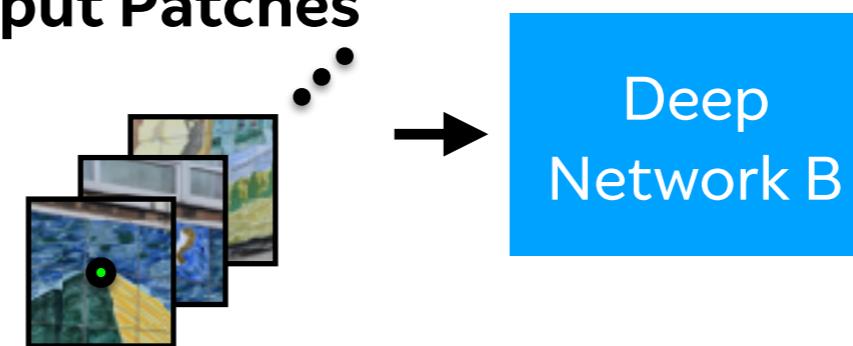
### Sliding Window



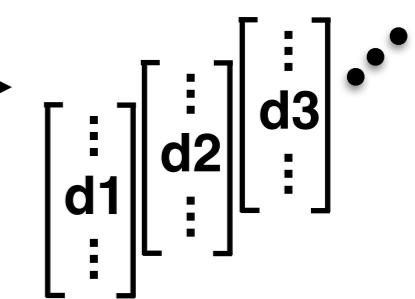
### Interest Points



### Input Patches



### Descriptors

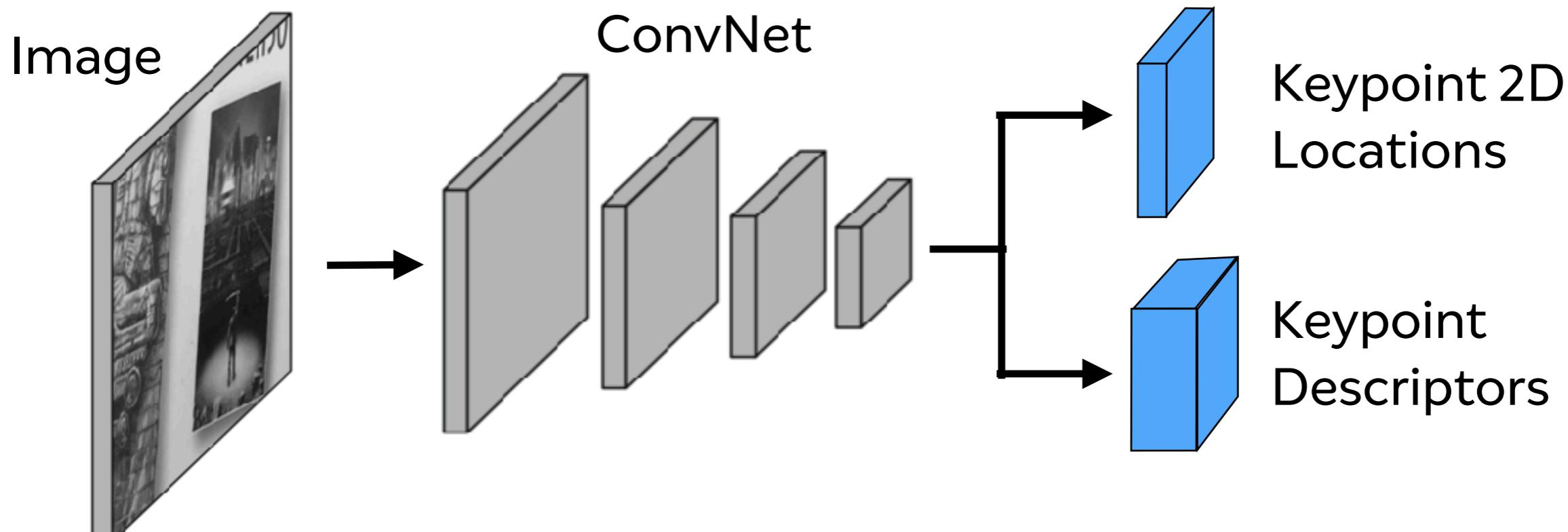


- Most low-compute Visual SLAM built on sparse frontends
- Extract points  $\rightarrow$  “Backend Ready”
- Most learned systems patch-based
  - Two separate networks
  - Lack powerful matchability of dense methods

# Question

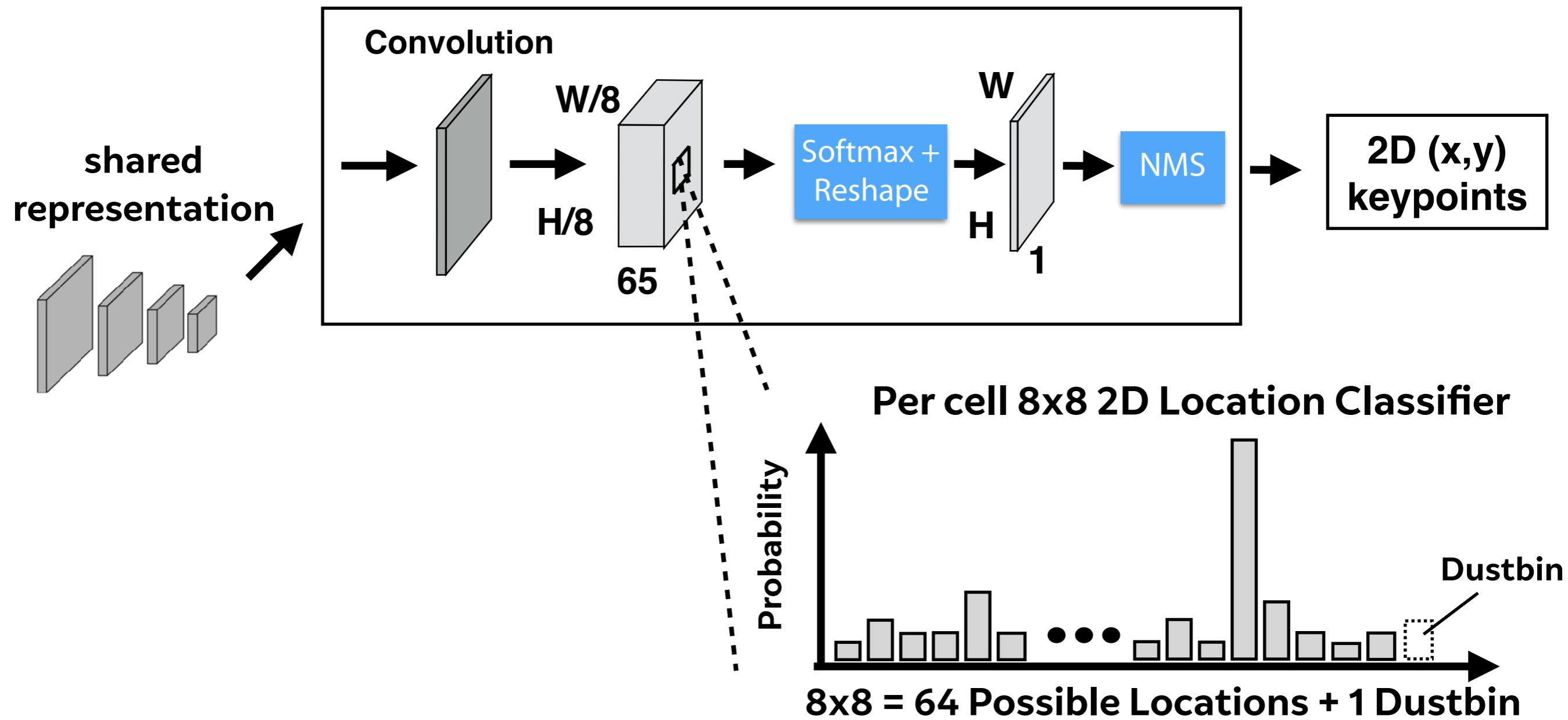
- How can we get the **power of dense matchability** and the **practicality of sparse output** in a **learnable framework**?

# SuperPoint: A Deep SLAM Front-end



- Powerful fully convolutional design
  - Points + descriptors computed jointly
  - Share VGG-like backbone
- Designed for real-time
  - Tasks share ~90% of compute
  - Two learning-free decoders: no deconvolution layers

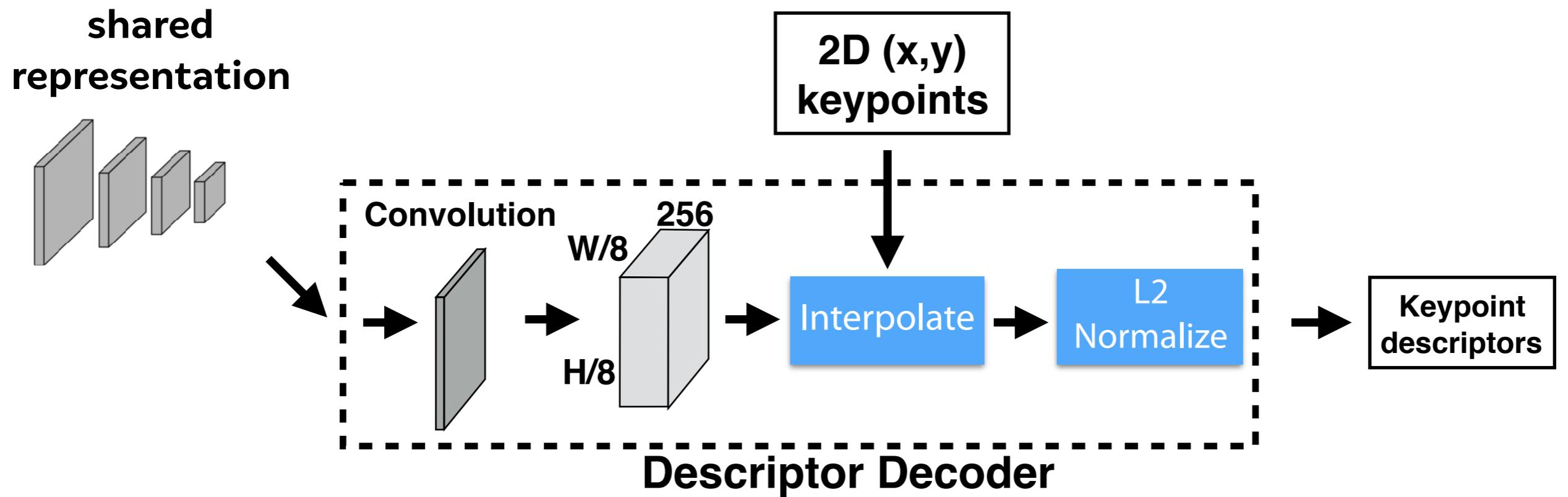
# Keypoint / Interest Point Decoder



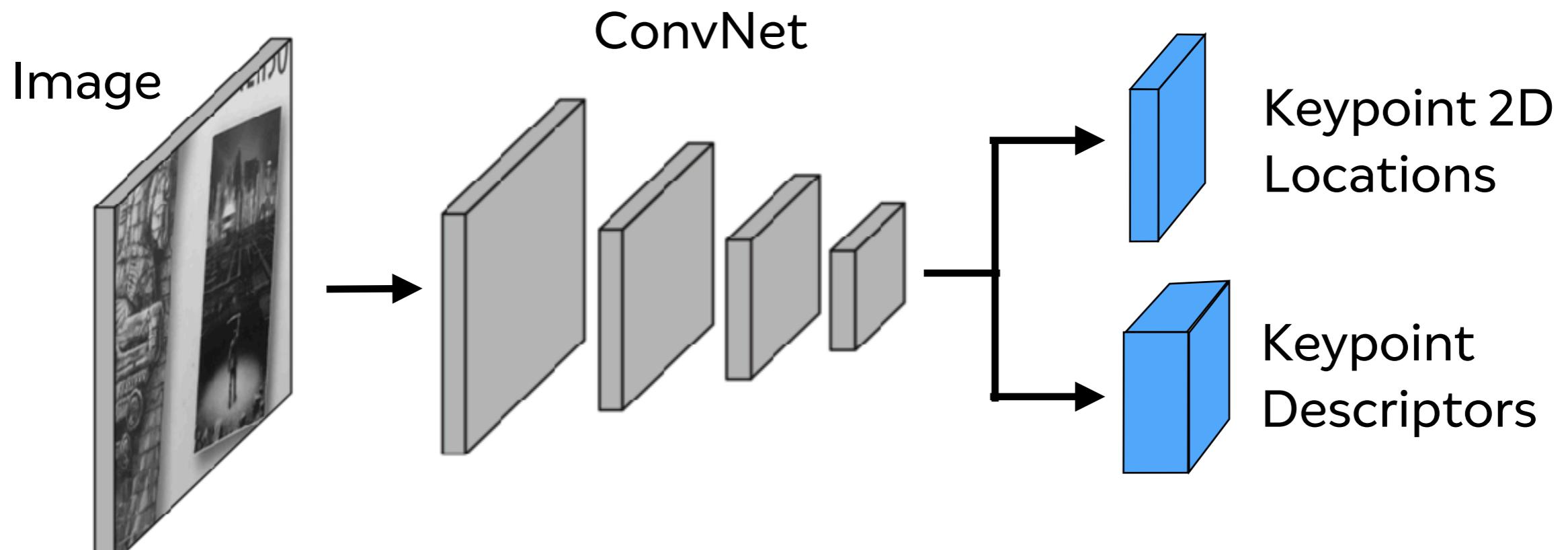
- No deconvolution layers
- Each output cell responsible for local 8x8 region

# Descriptor Decoder

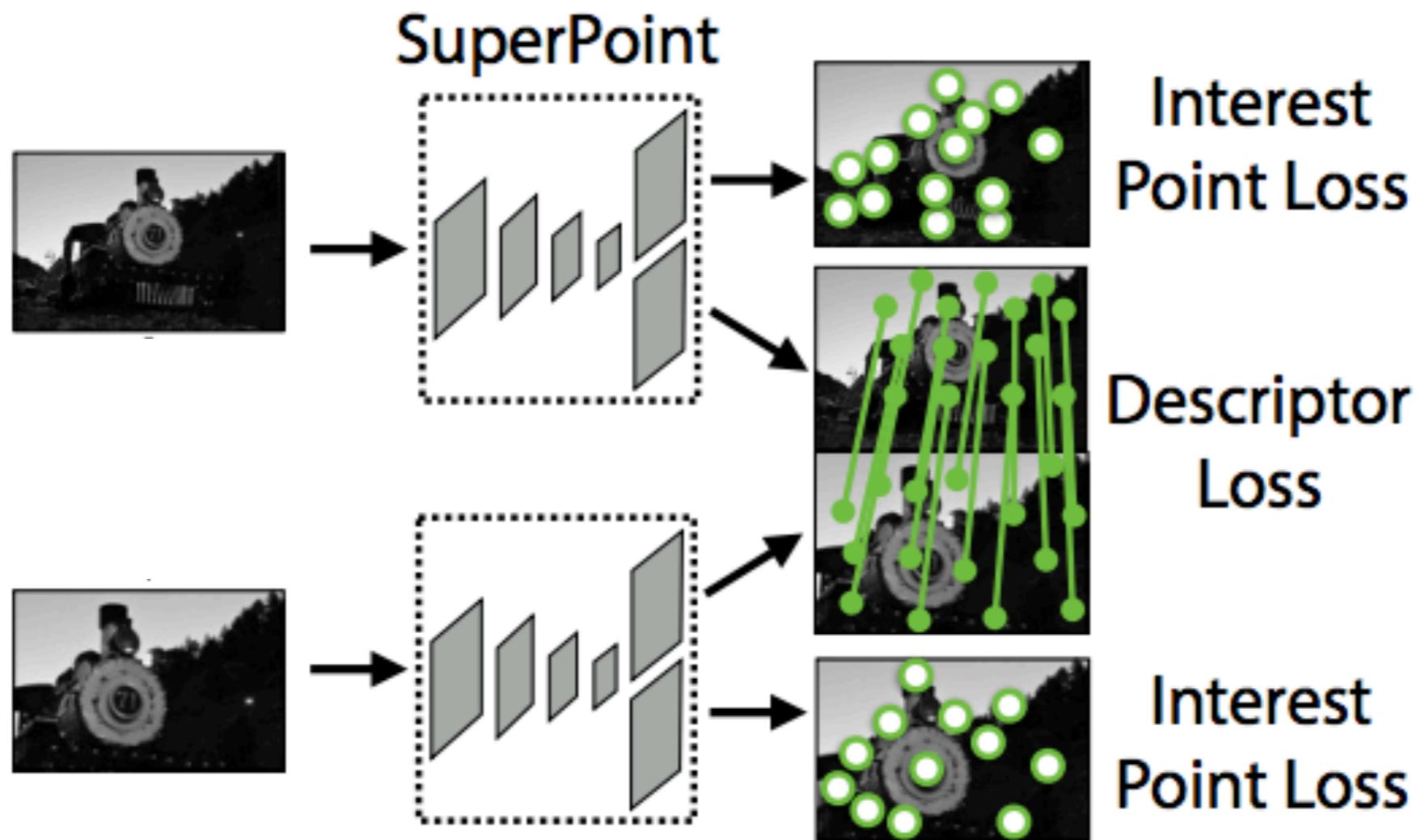
- Also no deconvolution layers
- Interpolate using 2D keypoint into coarse descriptor map



# How To Train SuperPoint?



# Setting up the Training



- Siamese training -> pairs of images
- Descriptor trained via metric learning
- Keypoints trained via supervised keypoint labels

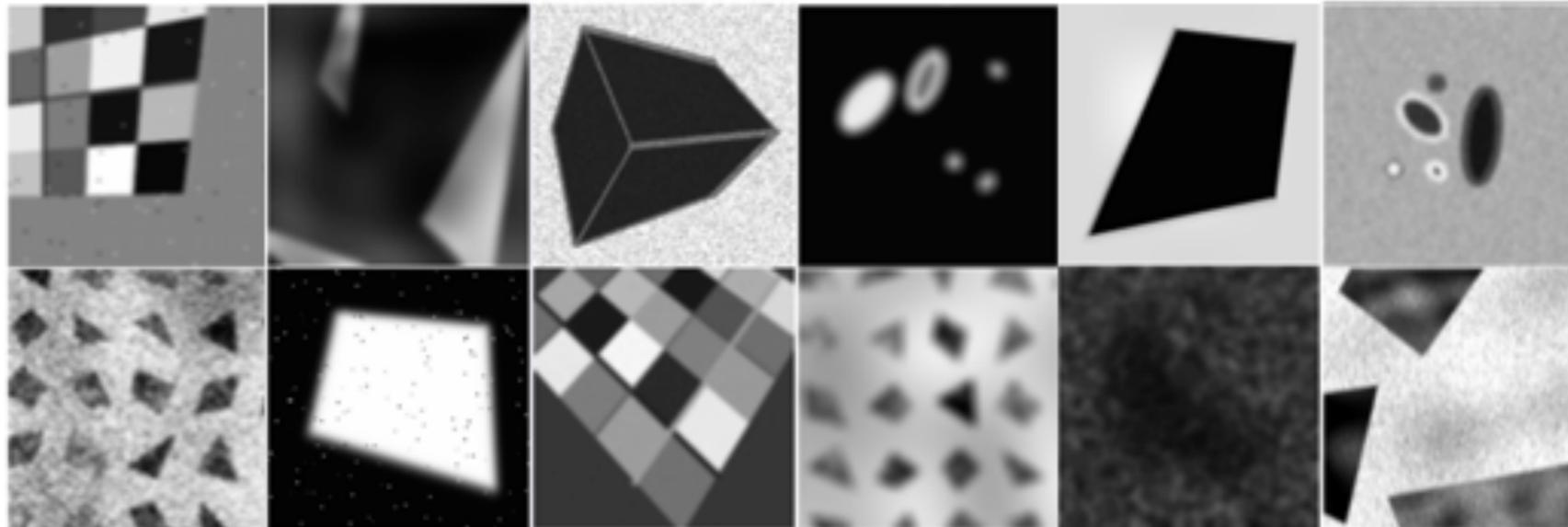
# How to get Keypoint Labels for Natural Images?



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

# Self-Supervised Approach

Synthetic Shapes (has interest point labels)



First train  
on this

MS-COCO (no interest point labels)

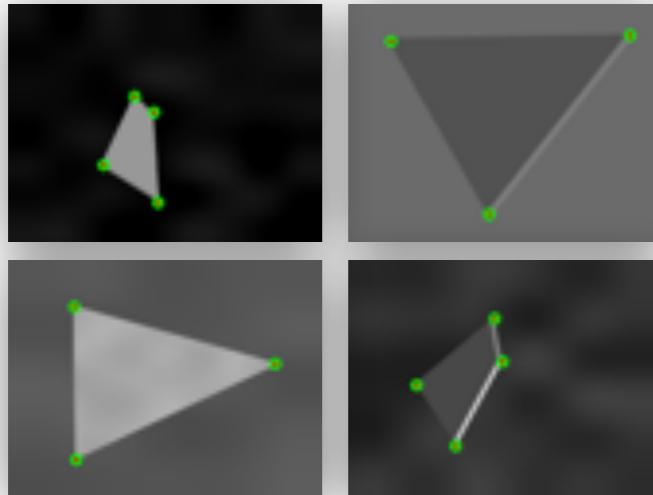


“Homographic  
Adaptation”

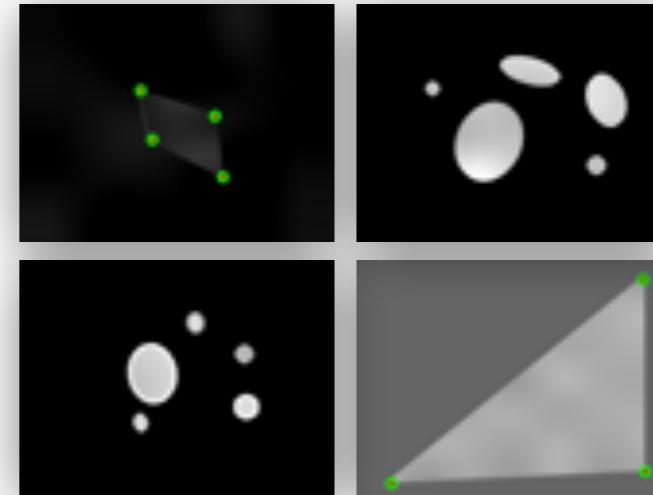
Use resulting  
detector to  
label this

# Synthetic Training

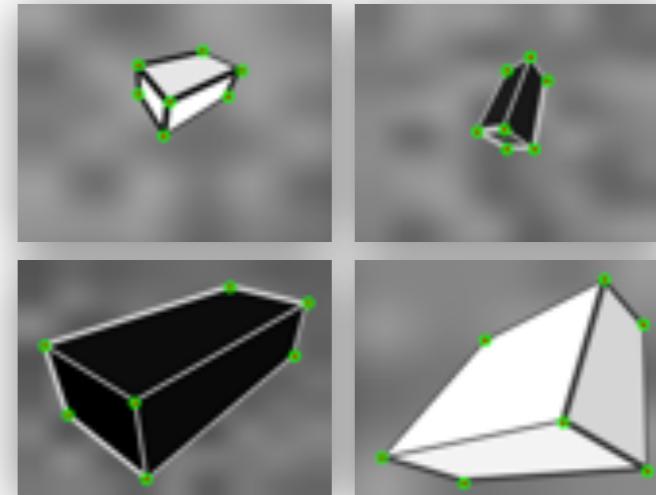
- Non-photorealistic shapes
- Heavy noise
- Effective and easy



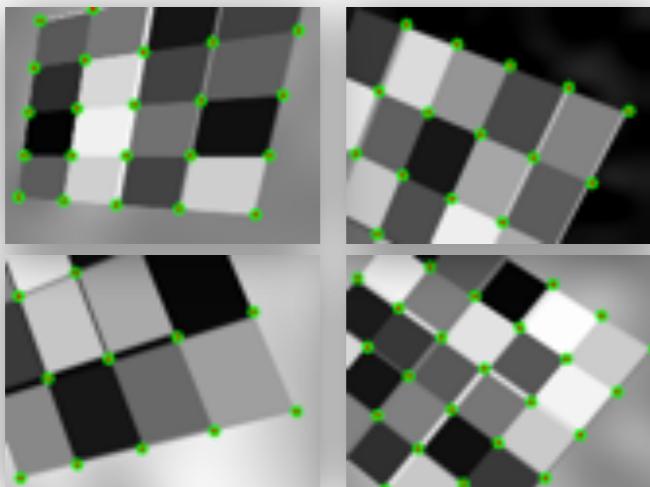
Quads/Tris



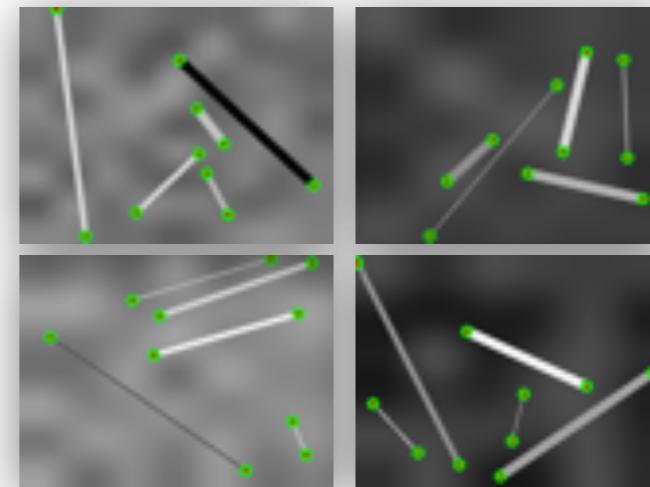
Quads/Tris/Ellipses



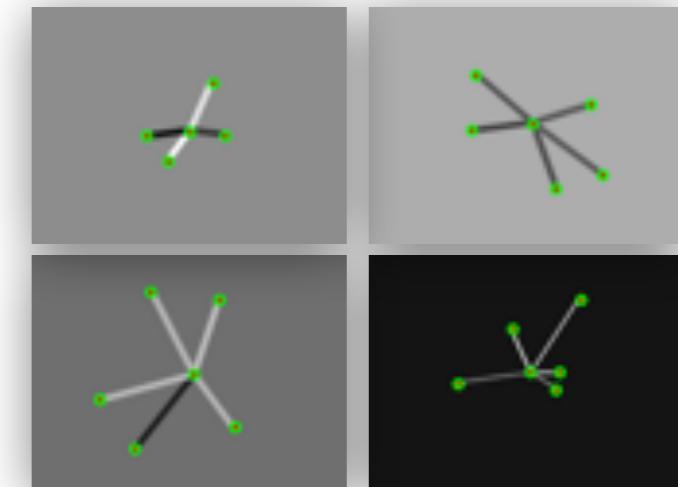
Cubes



Checkerboards



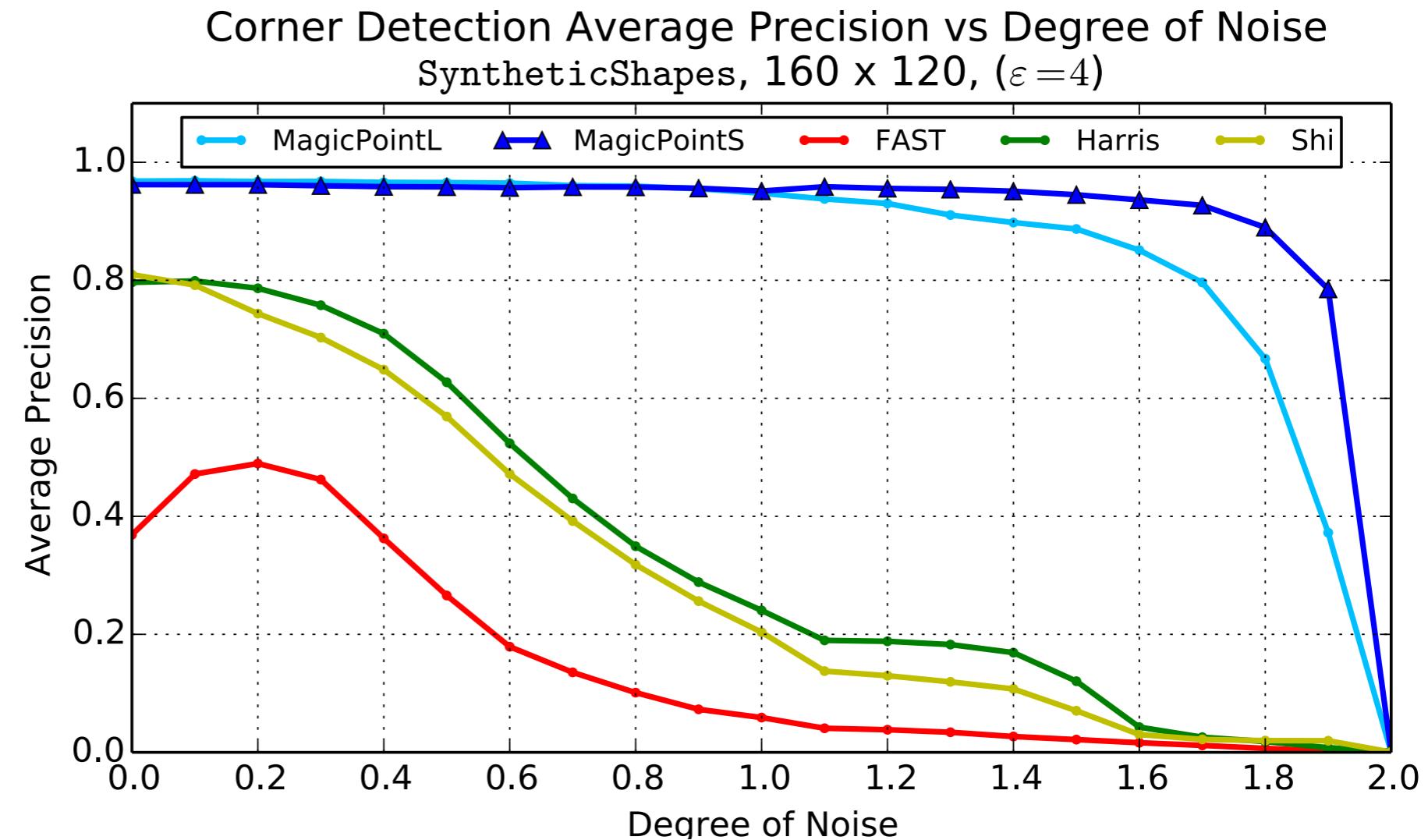
Lines



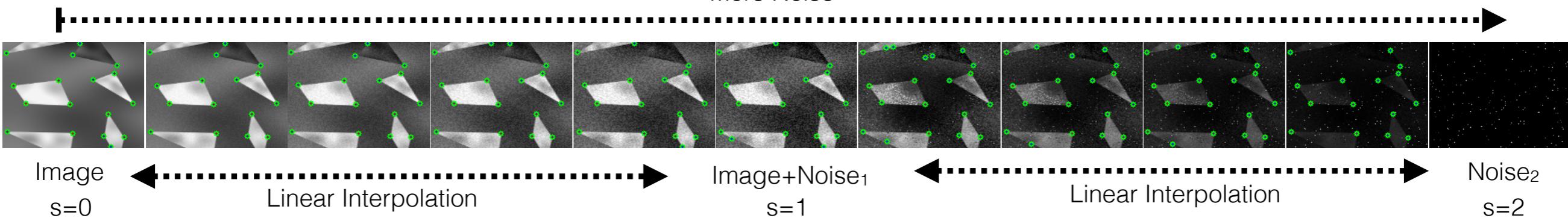
Stars

# Early Version of SuperPoint (MagicPoint)

“Toward Geometric  
Deep SLAM”  
DeTone et. al. 2017



Noise Legend



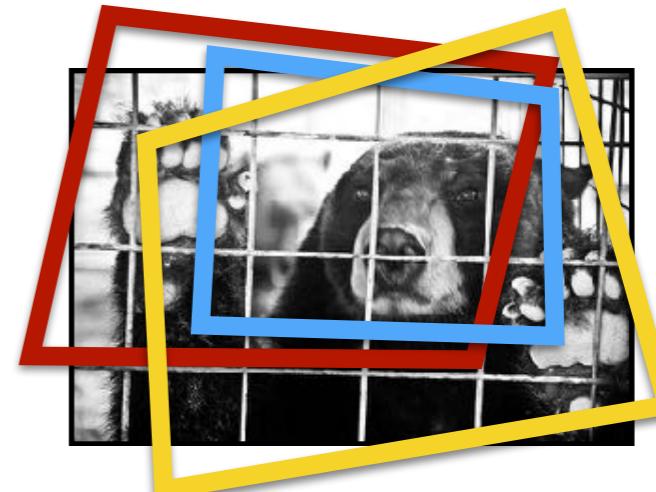
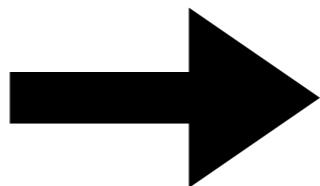
# Generalizing to Real Data

- Synthetically trained detector
  - Works! Despite large domain gap
  - Worked well on geometric structures
  - Under performed on certain textures unseen during training

Unlabeled  
Input  
Image



Synthetic Warp +  
Run Detector



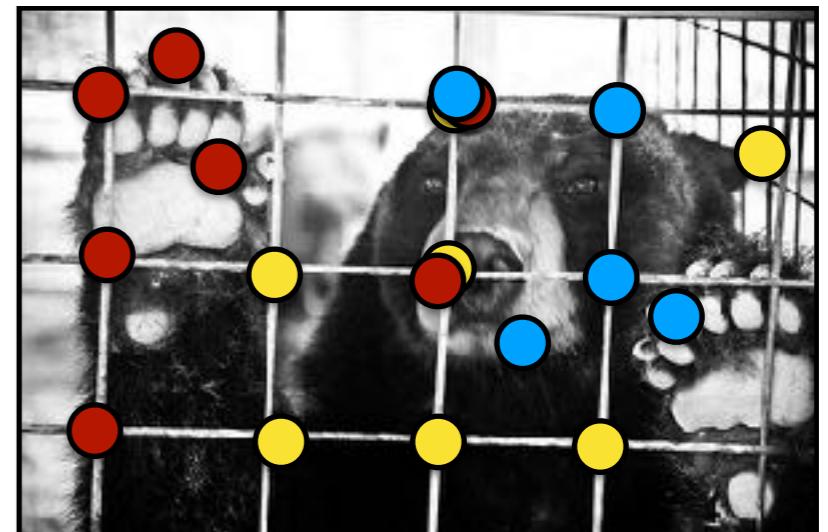
## Homographic Adaptation



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

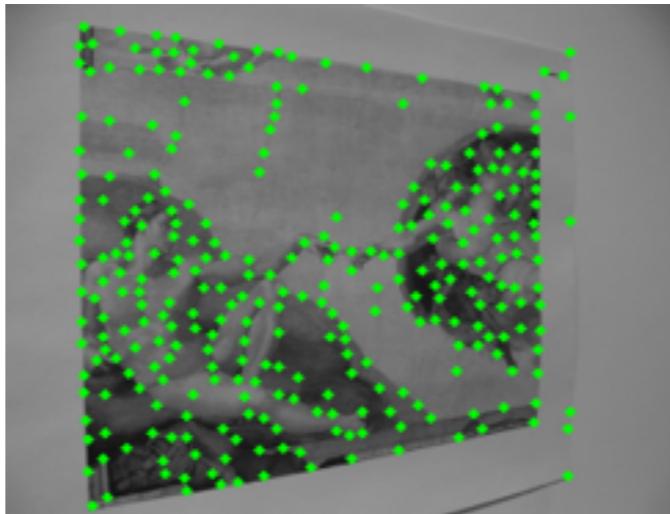
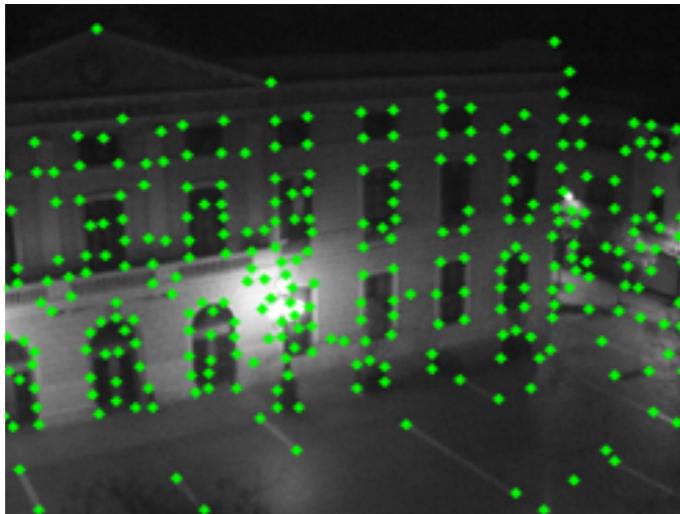
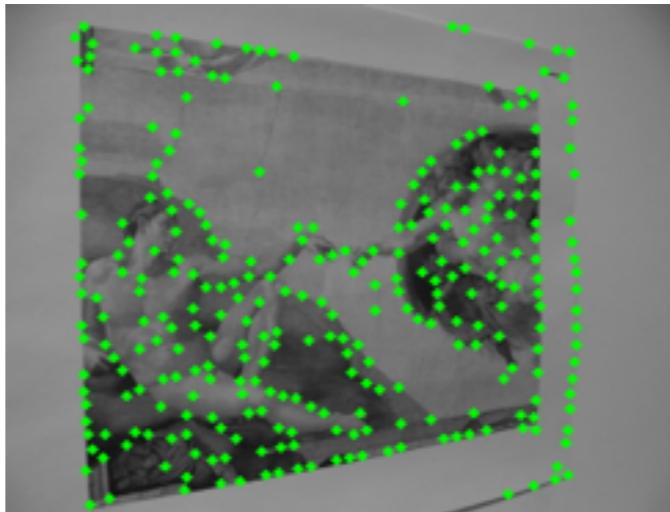
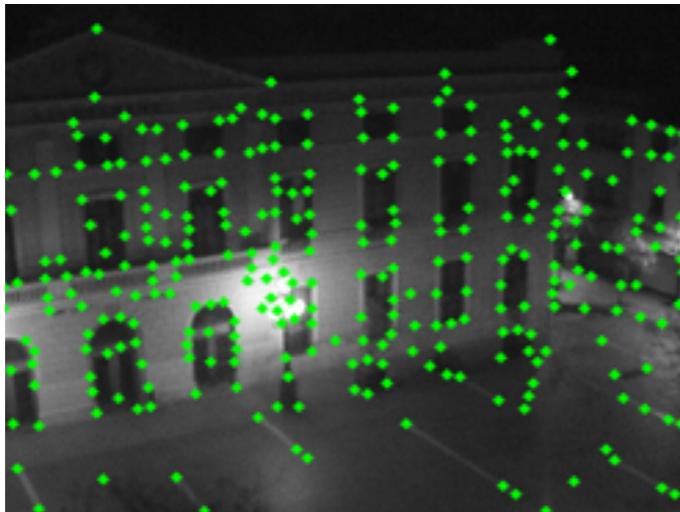
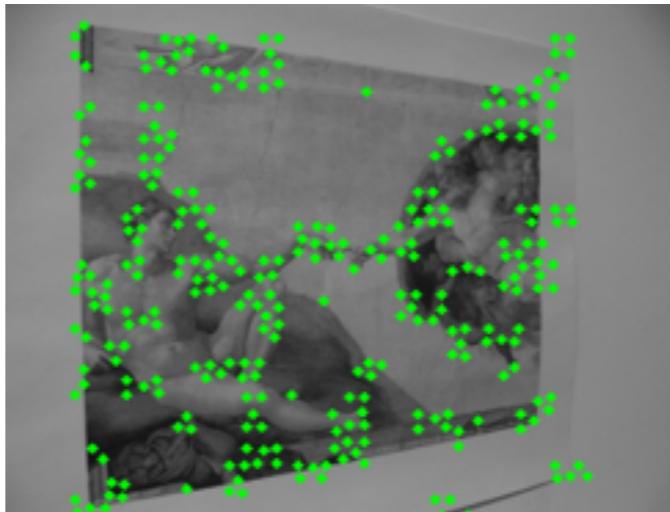
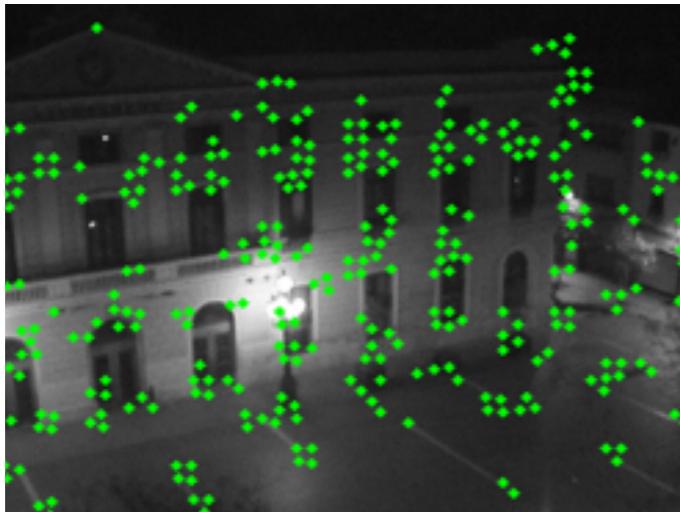
Point  
Aggregation

**Detected Point Superset**



# Iterative Homographic Adaptation

Homographic  
Adaptation  
↓



- Label, train, repeat, ...
- Resulting points:
  - Higher coverage
  - More repeatable

# HPatches Evaluation

- Homography estimation task
- Dataset of 116 scenes each with 6 images = 696 images
- Indoor and outdoor planar scenes
- Compared against LIFT, SIFT and ORB

50% of dataset:

Illumination

Change



50% of dataset:

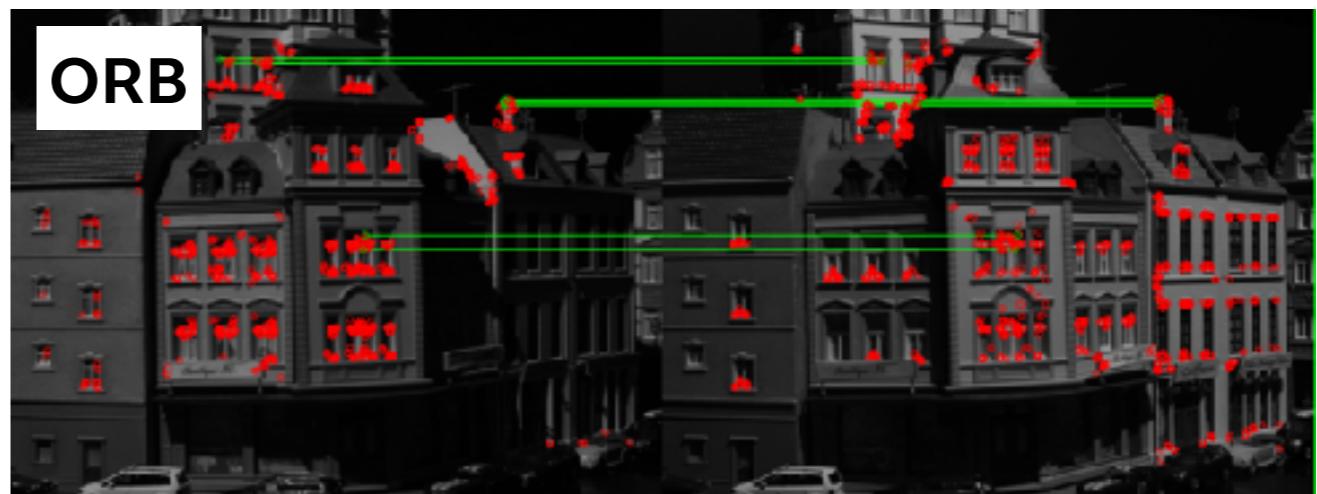
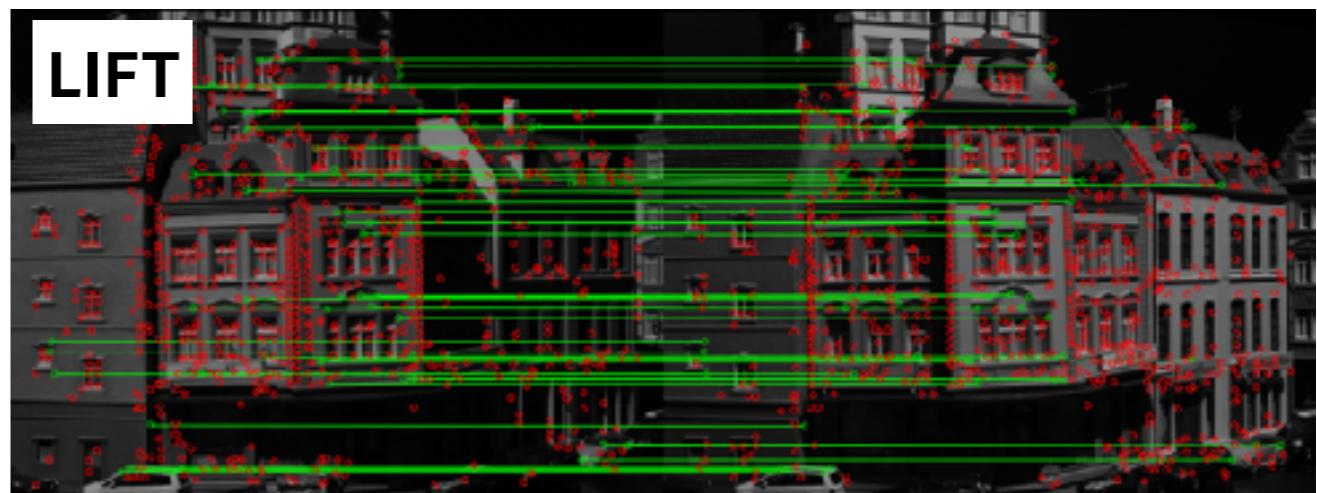
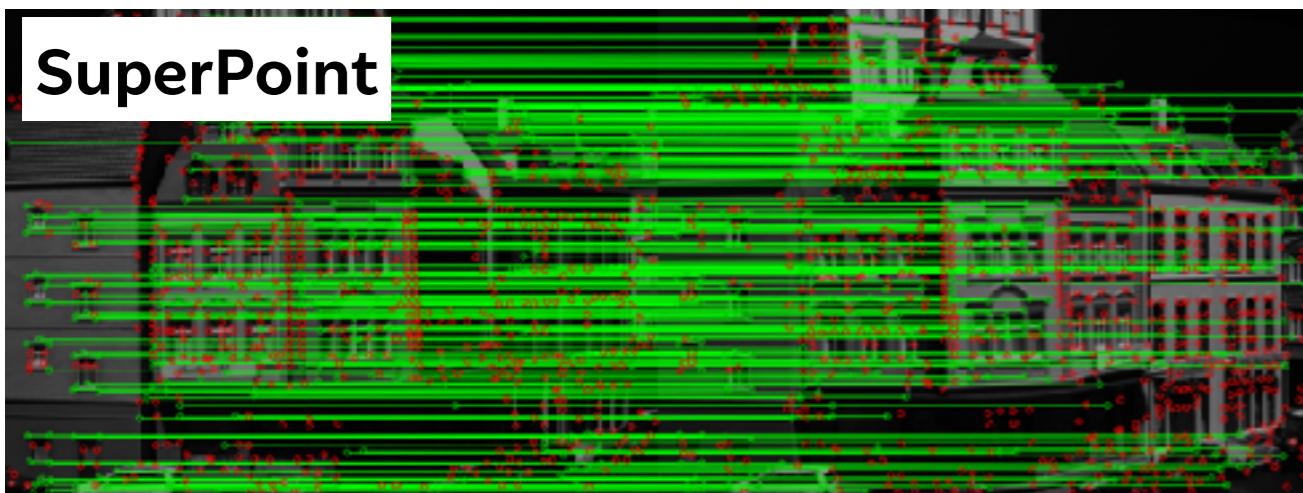
Viewpoint

Change



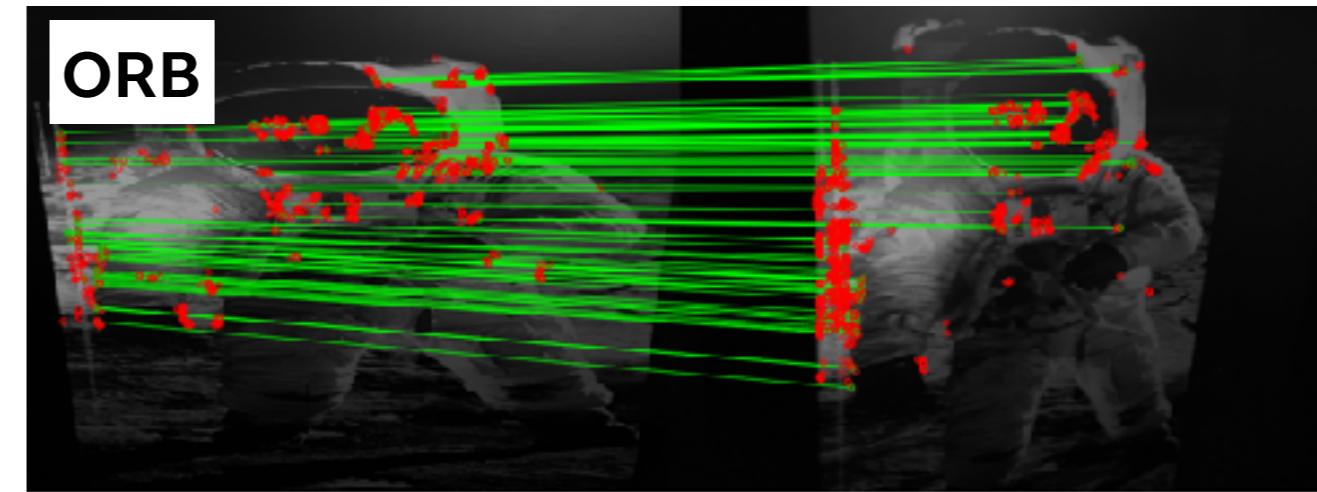
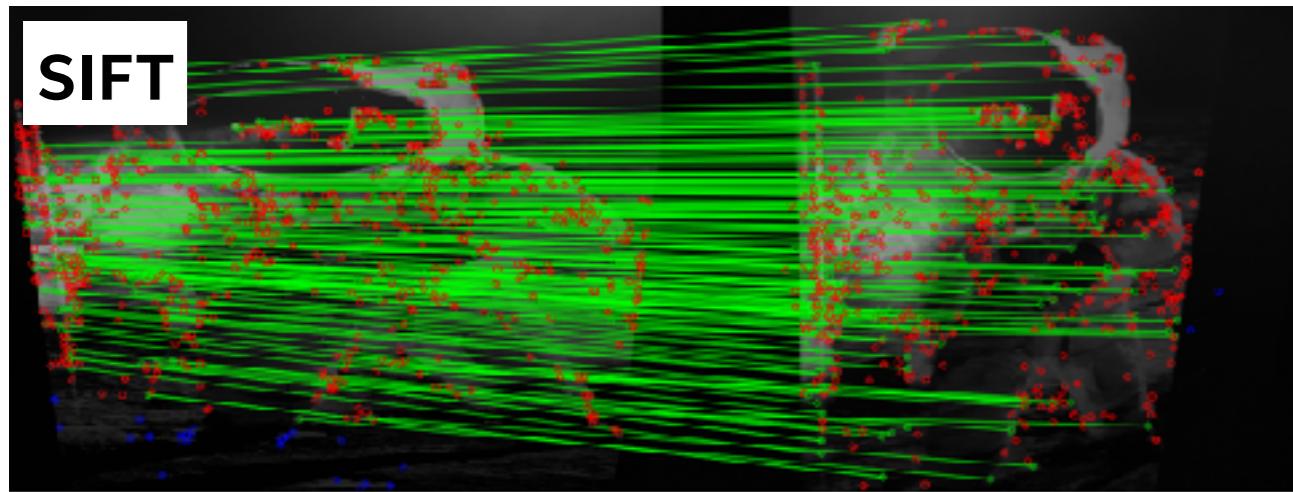
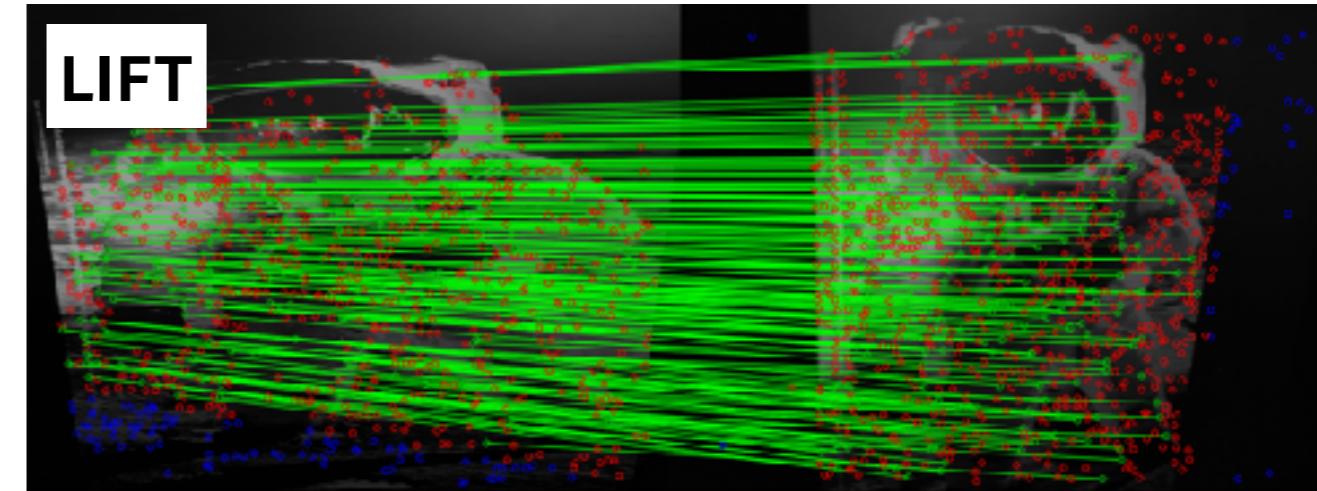
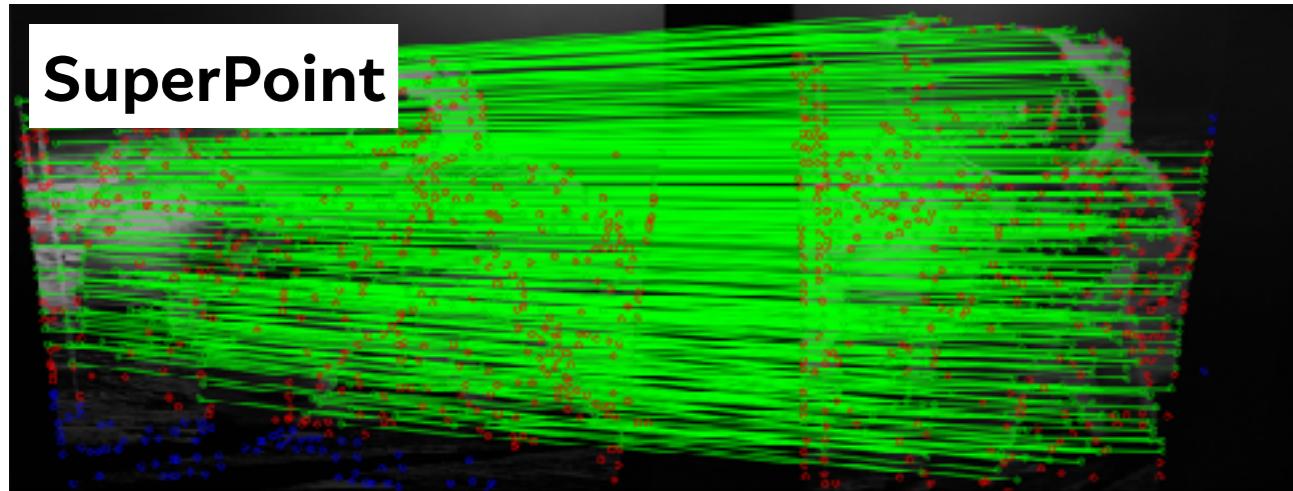
# Qualitative Illumination Example

- SuperPoint -> denser set of correct matches
- ORB -> highly clustered matches



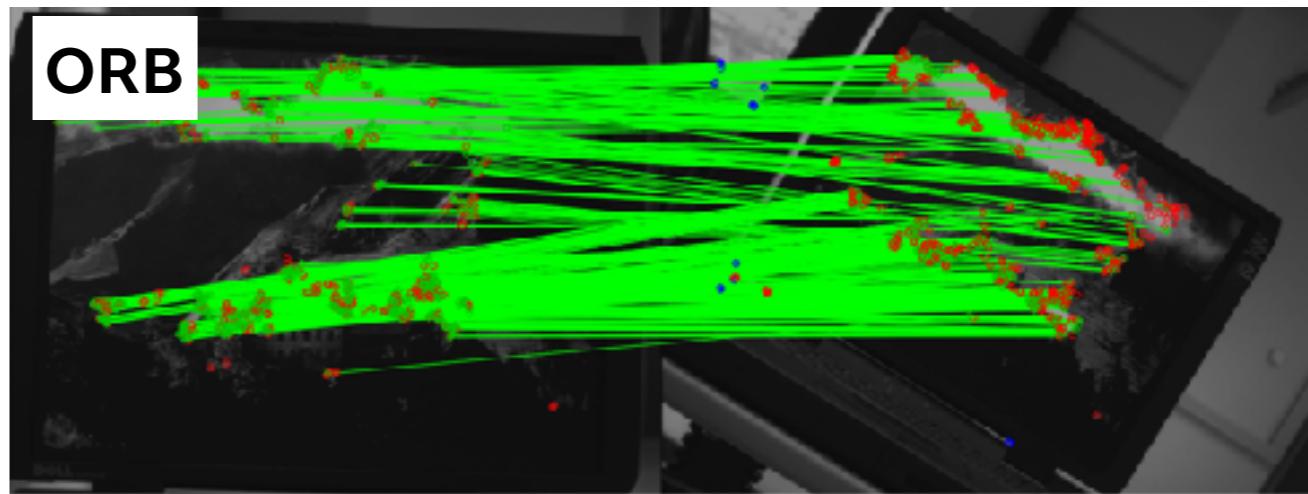
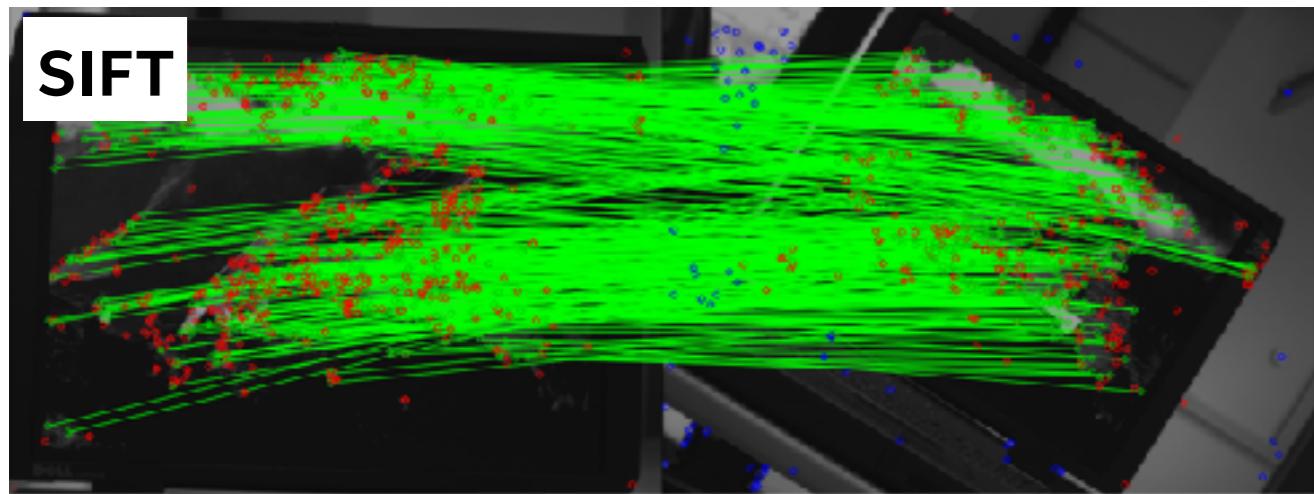
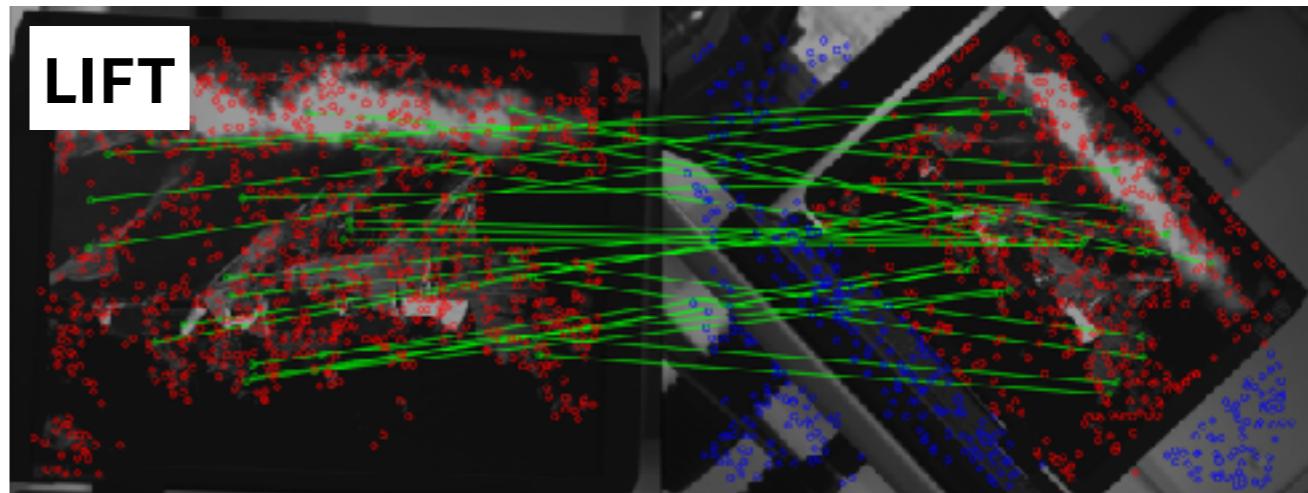
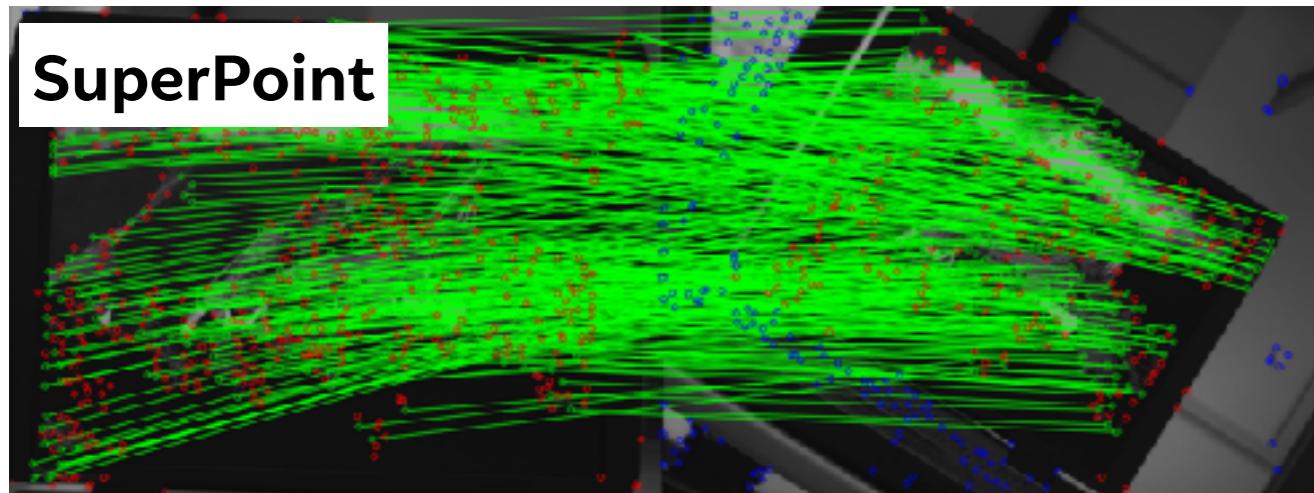
# Qualitative Viewpoint Example #1

- Similar story



# Qualitative Viewpoint Example #2

- In-plane rotation of ~35 degrees



# HPatches Evaluation

	Core Task	Sub-metrics			
		Descriptor Metrics		Detector Metrics	
	NN mAP	M. Score	Rep.	MLE	
SuperPoint	<b>0.684</b>	<b>0.821</b>	<b>0.470</b>	0.581	1.158
LIFT	0.598	0.664	0.315	0.449	1.102
SIFT	0.676	0.694	0.313	0.495	<b>0.833</b>
ORB	0.395	0.735	0.266	<b>0.641</b>	1.157

# Timing SuperPoint vs LIFT

- Speed important for low-compute Visual SLAM
  - SuperPoint total 640x480 time: ~ 33 ms
  - LIFT total 640x480 time: ~2 minutes

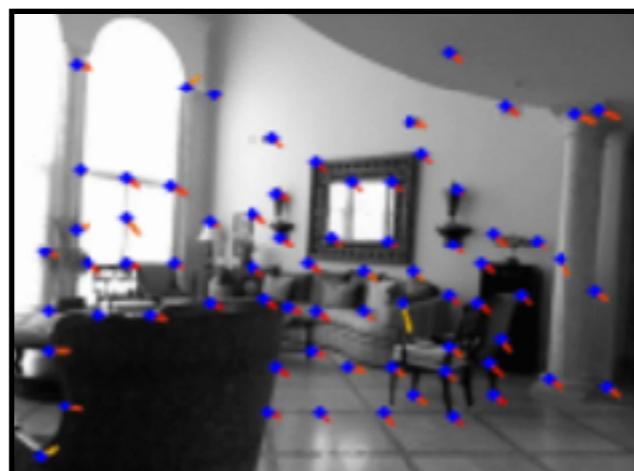
# 3D Generalizability of SuperPoint

- 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)*



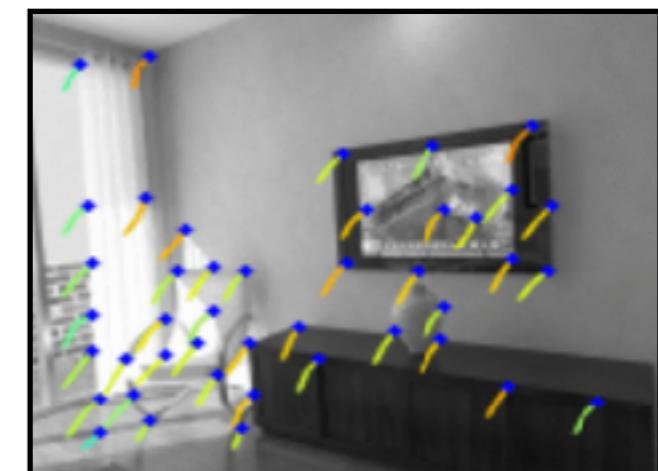
*NYU (Kinect)*



*MonoVO (fisheye)*



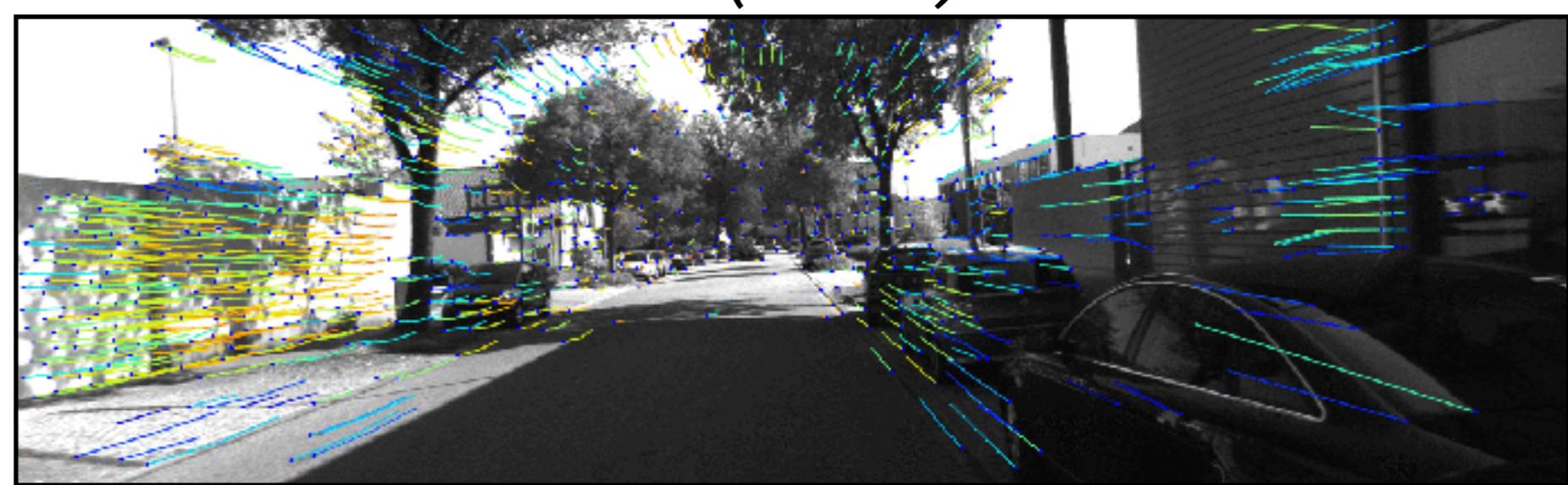
*ICL-NUIM (synth)*



*MS7 (Kinect)*



*KITTI (stereo)*



# New Announcement, Research @ MagicLeap

Public Release of Pre-trained Net:

**github.com/MagicLeapResearch/SuperPointPretrainedNetwork**

- Sparse Optical Flow Tracker Demo
- Implemented in Python + PyTorch
- Two files, minimal dependencies
- Easy to get up and running

The screenshot shows the GitHub repository page for "SuperPointWeightsFile and Demo Script". At the top is the Magic Leap logo. Below it is the repository name. Underneath is a section titled "Introduction" which includes a detailed description of the project, links to the full paper PDF and authors, and a summary of the demo. A thumbnail image below the introduction shows a scene from a Freiburg RGBD dataset with overlaid point tracks.

magic leap

Research @ Magic Leap

**SuperPoint Weights File and Demo Script**

**Introduction**

This repo contains the pretrained SuperPoint network, as implemented by the originating authors. SuperPoint is a research project at Magic Leap. The SuperPoint network is a fully convolutional deep neural network trained to detect interest points and compute their accompanying descriptors. The detected points and descriptors can thus be used for various image-to-image matching tasks. For more details please see

- + Full paper PDF: [SuperPoint: Self-Supervised Interest Point Detection and Description](#)
- + Authors: *Daniel DeTone, Tomasz Malisiewicz, Andrew Rabinovich*

This demo showcases a simple sparse optical flow point tracker that uses SuperPoint to detect points and match them across video sequences. The repo contains two core files (1) a PyTorch weights file and (2) a python deployment script that defines the network, loads images and runs the pytorch weights file on them, creating a sparse optical flow visualization. Here are videos of the demo running on various publicly available datasets:

Freiburg RGBD:

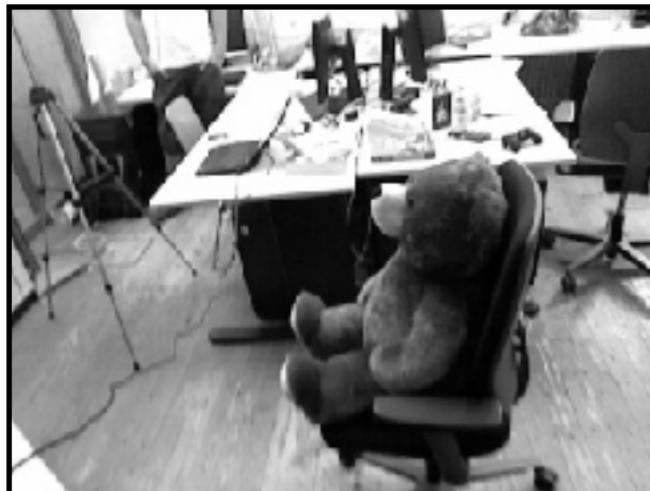
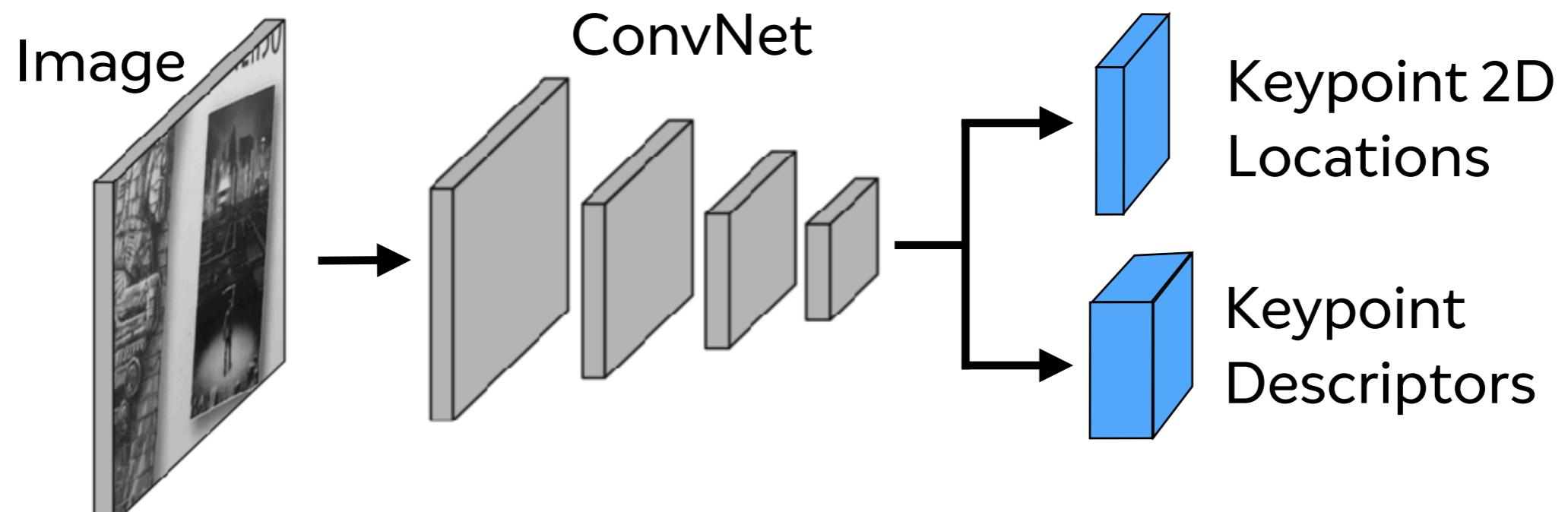
# Take-Aways

- “SuperPoint”: A Modern Deep SLAM Frontend
  - Non-patch based fully convolutional network
  - Real-time deployability
- Self-supervised recipe to train keypoints
  - Synthetic pre-training
  - Homography-inspired domain adaptation
- Public code available to run SuperPoint

**Thank You**

# Questions?

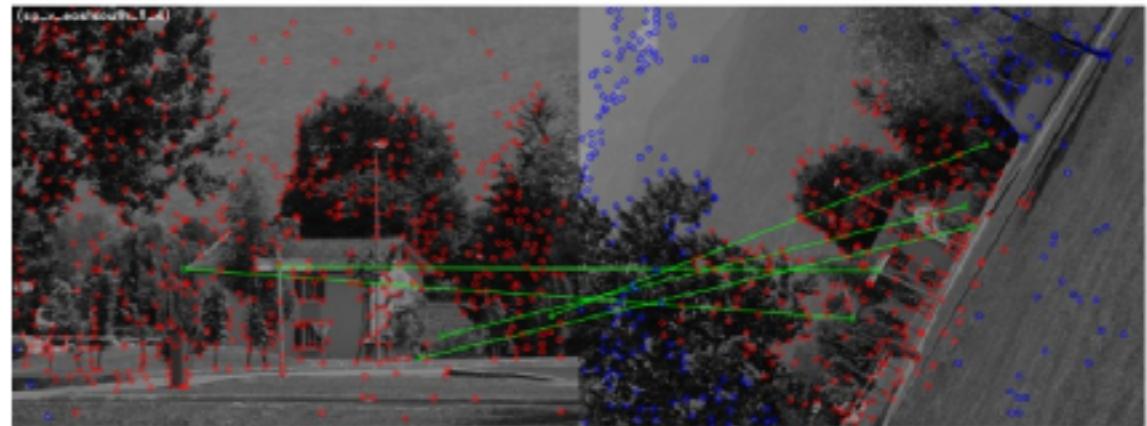
## SuperPoint: A Modern Deep SLAM Front-end



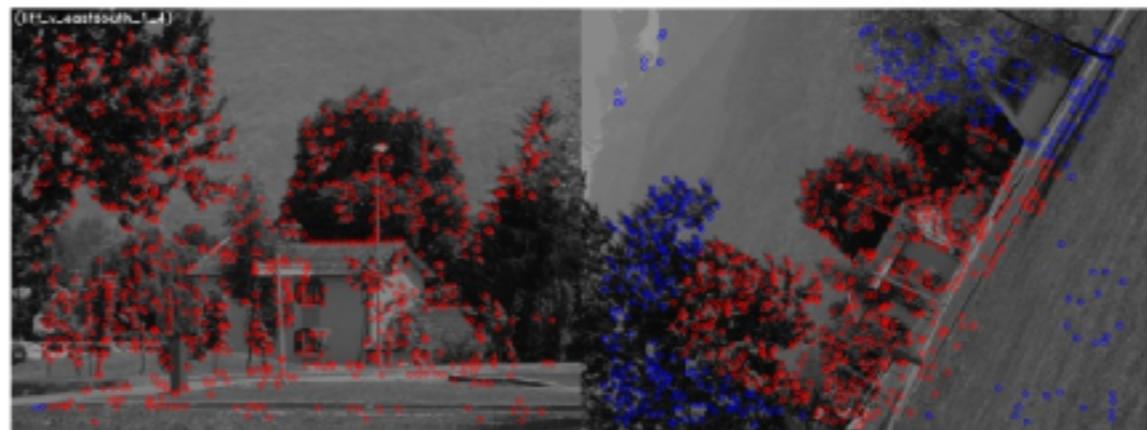
# Extra Slides

# Failure Mode: Extreme Rotation

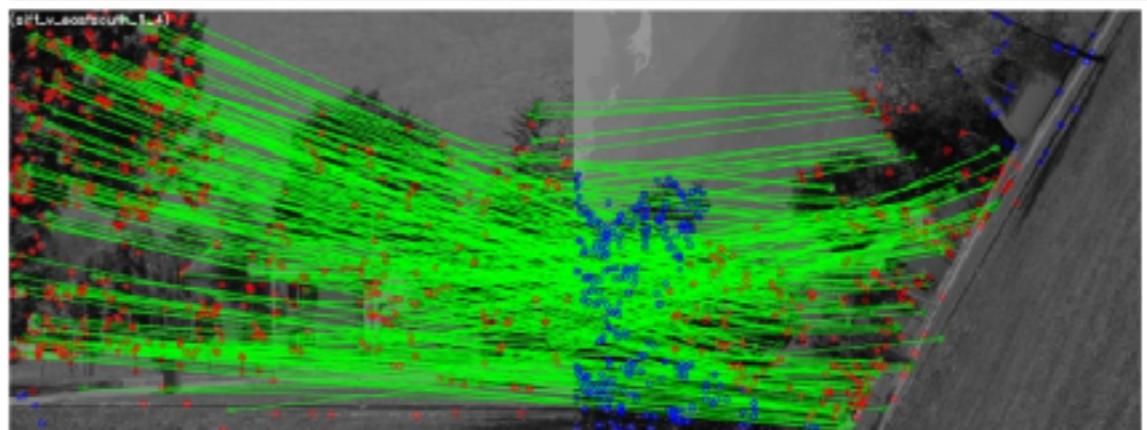
- Extreme in-plane rotations
- Trained for ~30 deg rotations
- Optimized tracking scenarios
- LIFT also struggles, despite learned orientation estimation



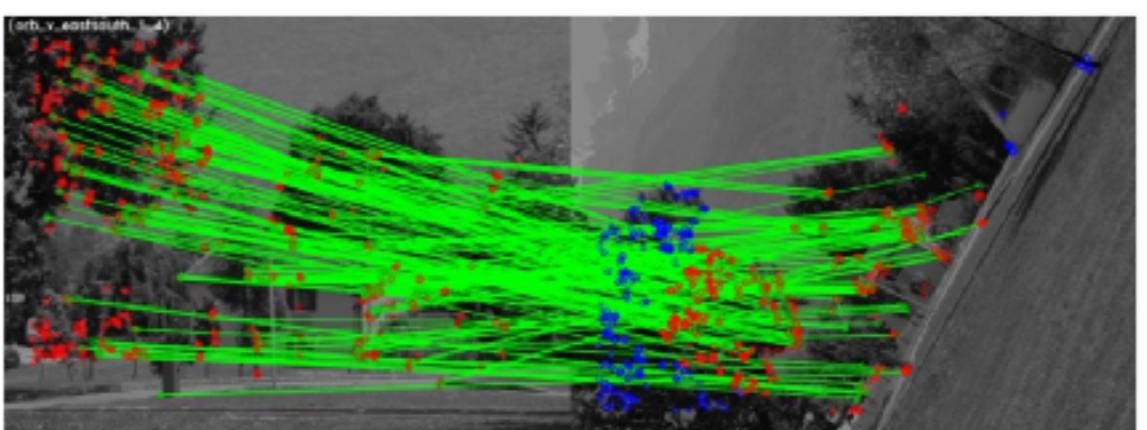
Super-  
Point



LIFT



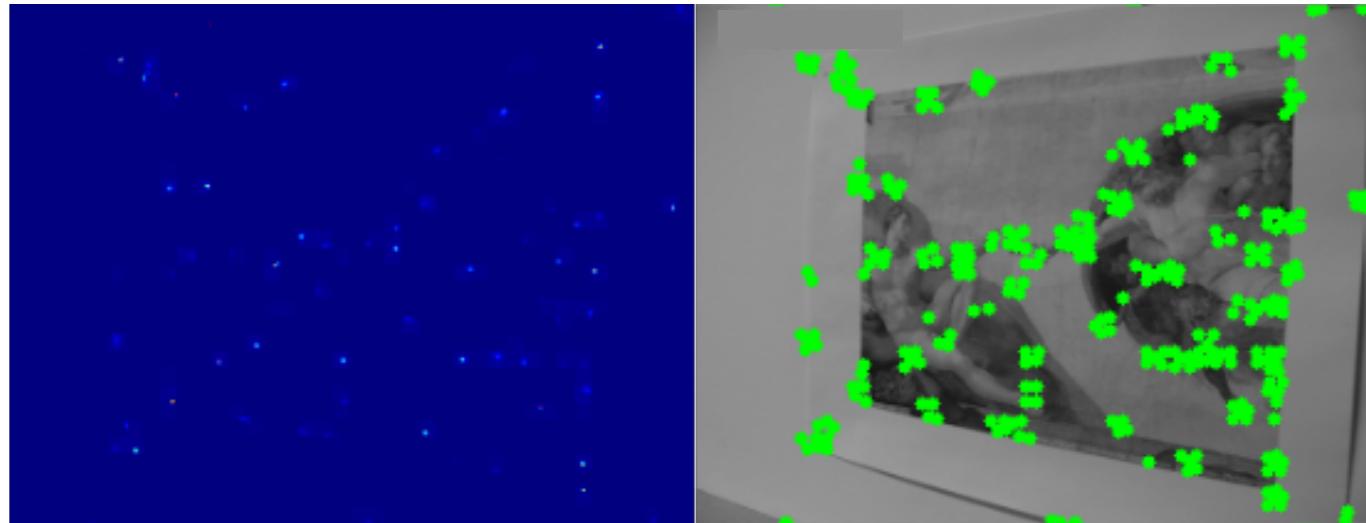
SIFT



ORB

# Iterative Homographic Adaptation

MagicPoint



↓ SuperPoint

