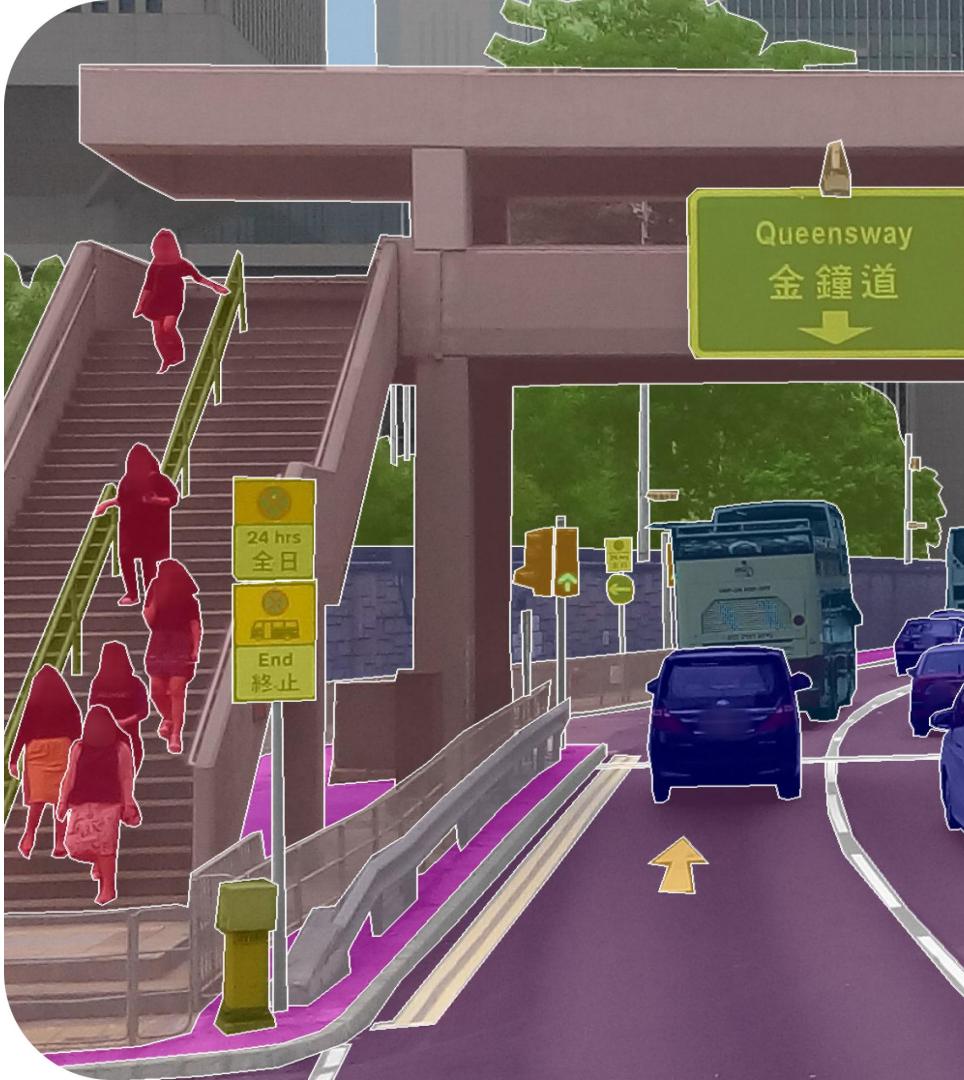
Computer Vision with Less Supervision

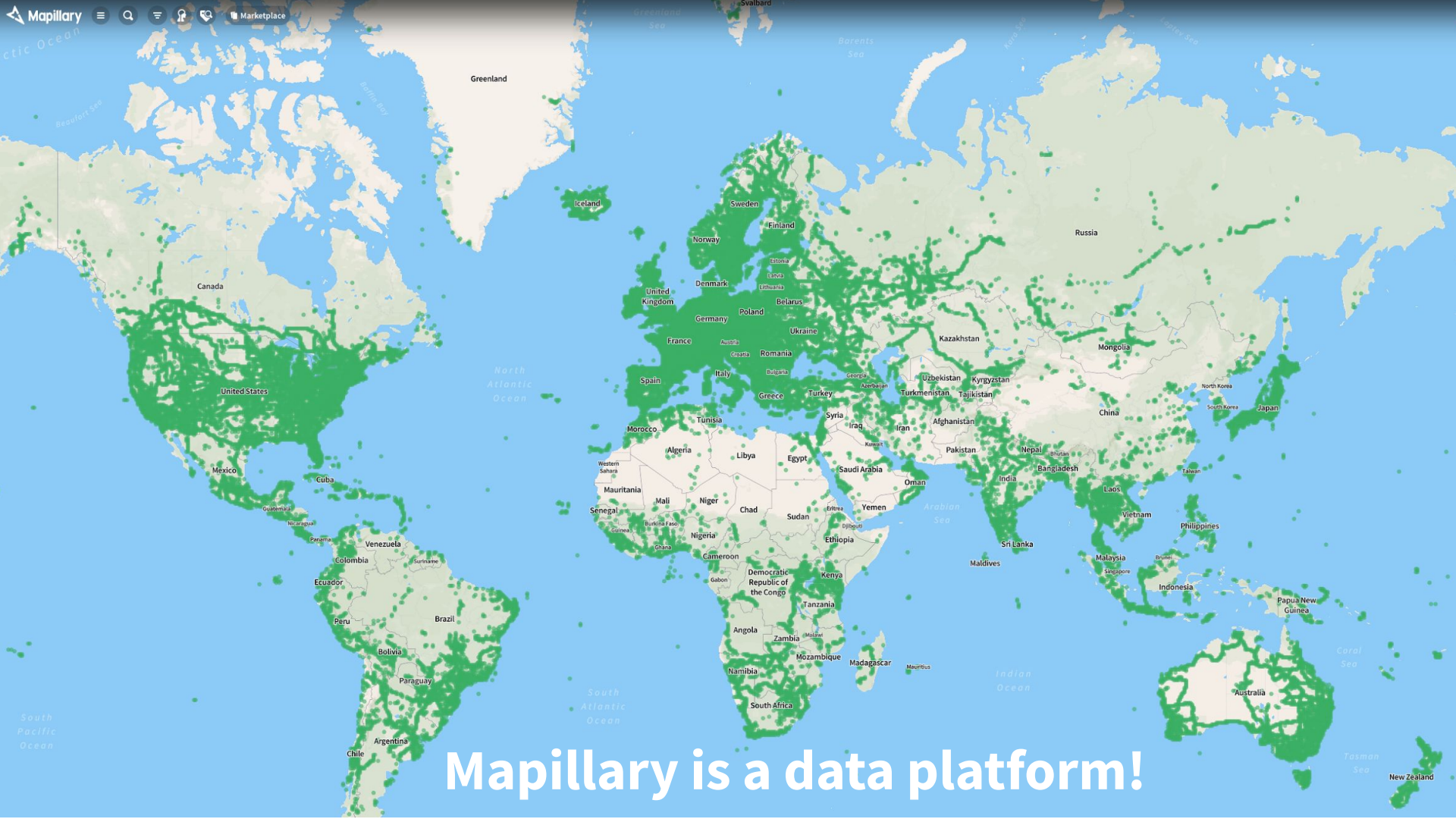
Peter Kontschieder

June 14, 2020

Mapillary



Mapillary is the street-level imagery platform that scales and automates mapping

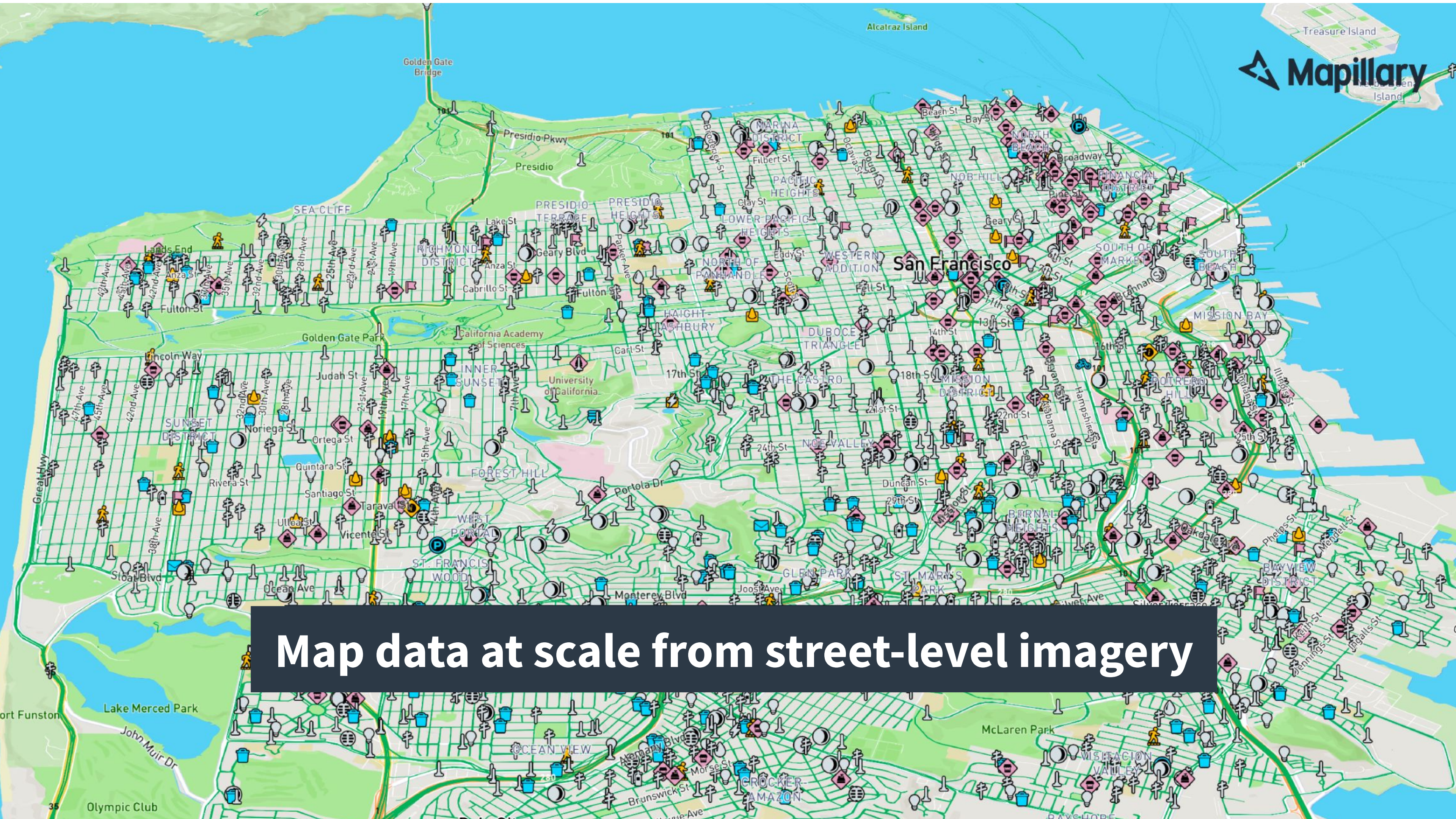


Mapillary is a data platform!

Anyone with Any Camera, Anywhere

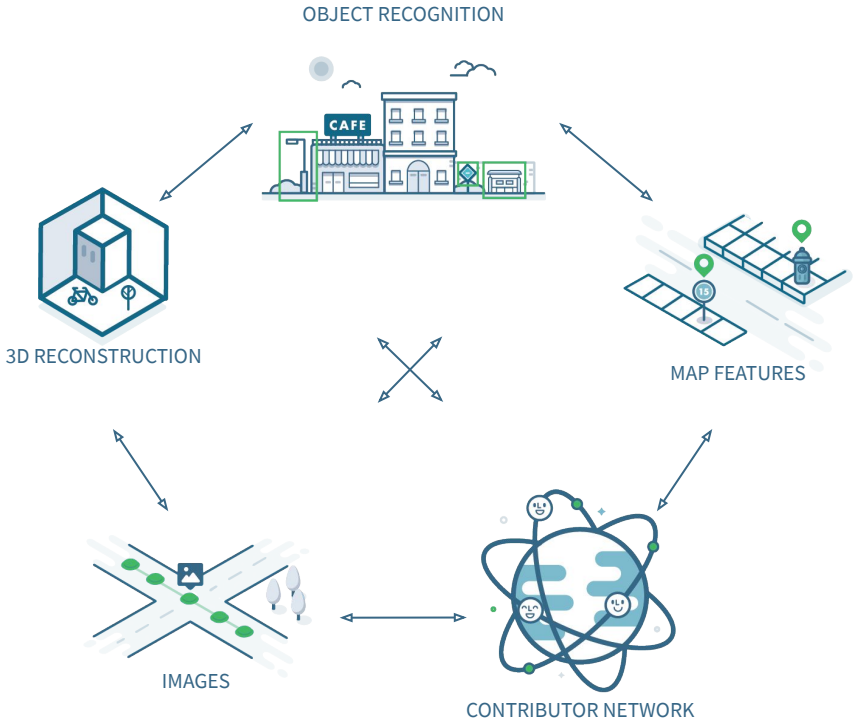
Phone | Action cam | Dash Cam | Vehicle Sensor | Pro Rig

1b+ images, >10 million road km mapped


Mapillary
Map data at scale from street-level imagery
San Francisco
Golden Gate Bridge
Alcatraz Island
Treasure Island
Presidio
Presidio Pkwy
SEA CLIFF
Lands End
RICHMOND DISTRICT
PRESIDIO TERRACE
PRESIDIO HEIGHTS
MARINA DISTRICT
PACIFIC HEIGHTS
NORTH BEACH
NOB HILL
FINANCIAL DISTRICT
LOWER PACIFIC HEIGHTS
WESTERN ADDITION
NORTH OF PANHANDLE
SOUTH OF MARKET
SOUTH BEACH
HAIGHT-ASHBURY
MISSION BAY
Golden Gate Park
California Academy of Sciences
DUBOCE TRIANGLE
INNER SUNSET
University of California
THE CASTRO
MISSION DISTRICT
POTRERO HILL
SUNSET DISTRICT
NOE VALLEY
FOREST HILL
WEST PORTAL
BERNAL HEIGHTS
ST. FRANCIS WOOD
GLEN PARK
ST. MARYS PARK
BAYVIEW DISTRICT
Lake Merced Park
John Muir Dr
Olympic Club
OCEAN VIEW
CROCKER-AMAZON
McLaren Park
VISITACION VALLEY
Great Hwy
Sloat Blvd
Monterey Blvd
Fulton St
Lincoln Way
Judah St
Noriega St
Ortega St
Quintara St
Rivera St
Santiago St
Taraval St
Ulloa St
Vicente St
Ocean Ave
Geary Blvd
Broadway
Market St

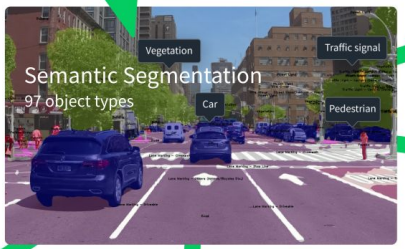
The Mapillary Ecosystem

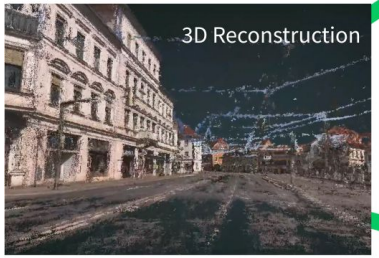
Schematic Data Lifecycle

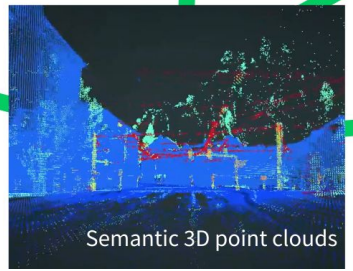


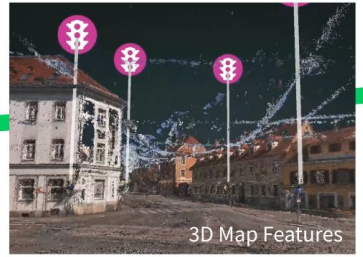
Strong Dependence on Recognition Algorithms

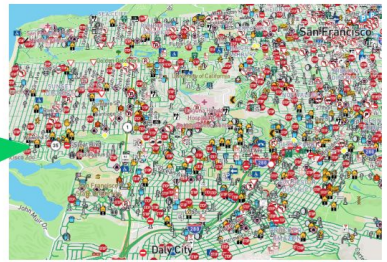
Images











Research @ Mapillary

Meet the Team!

Peter

Lorenzo

Arno

Manuel

Samuel

Aleksander

Andrea

Markus

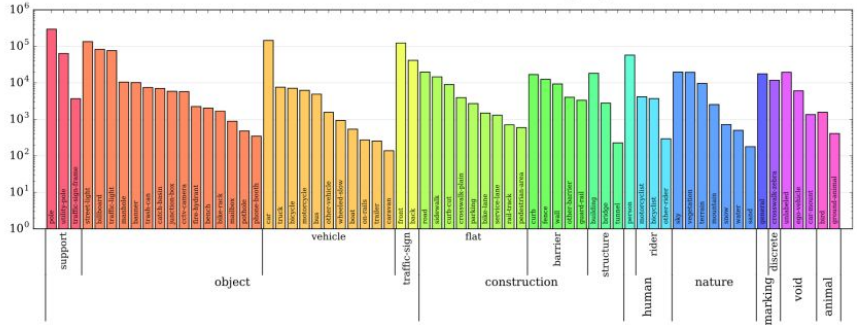
Mapillary Data Playground

The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes

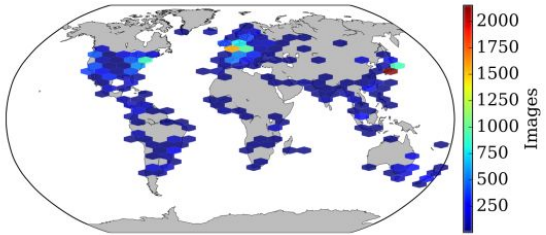
G. Neuhold, T. Ollmann, S. Rota Bulò, P. Kontschieder. (ICCV 2017)

Mapillary Research

Labeled instances per category

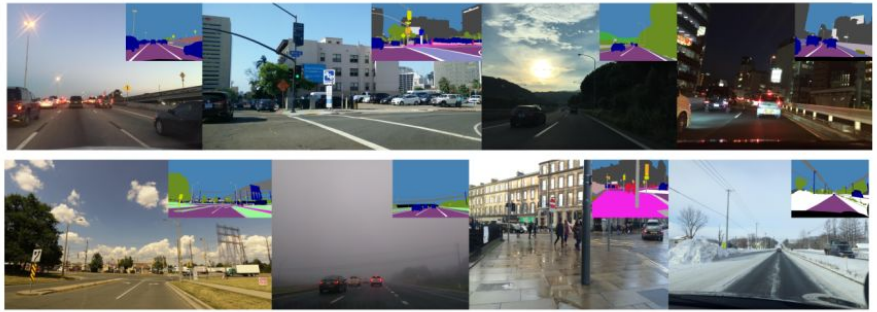


Global geographic reach (6 continents)

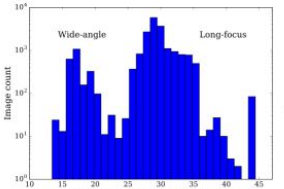


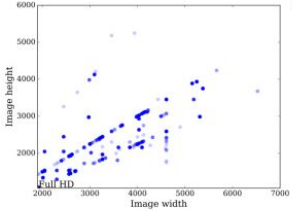
Diverse viewpoints from roads, sidewalks and off-road

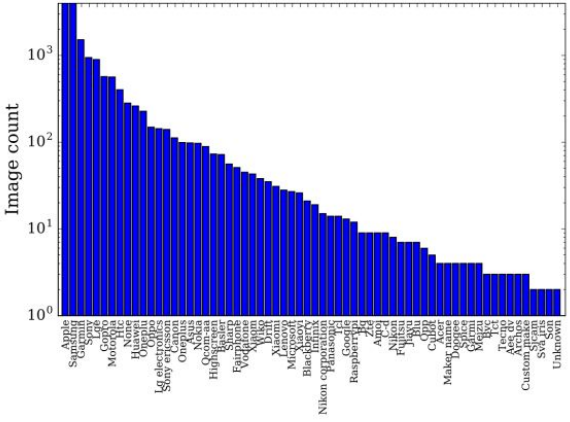
Various weather conditions and capture times



Wide variety of camera sensors, focal lengths image aspect ratios and types of camera noise







Selected projects in this talk:

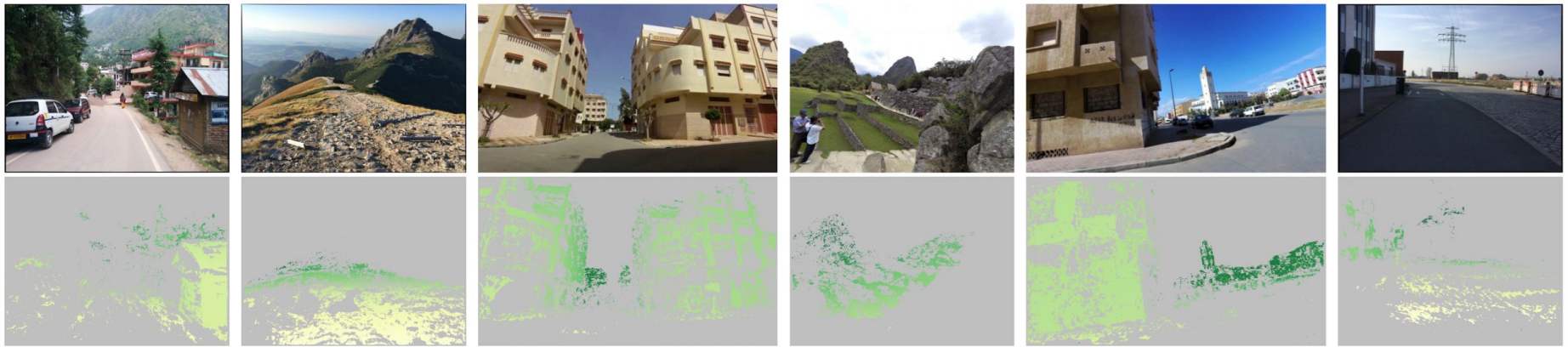
Single Image Depth Estimation
Multi-Object Tracking and Segmentation

Mapillary Planet Scale Depth Dataset (MPSD)

MPSD in a nutshell

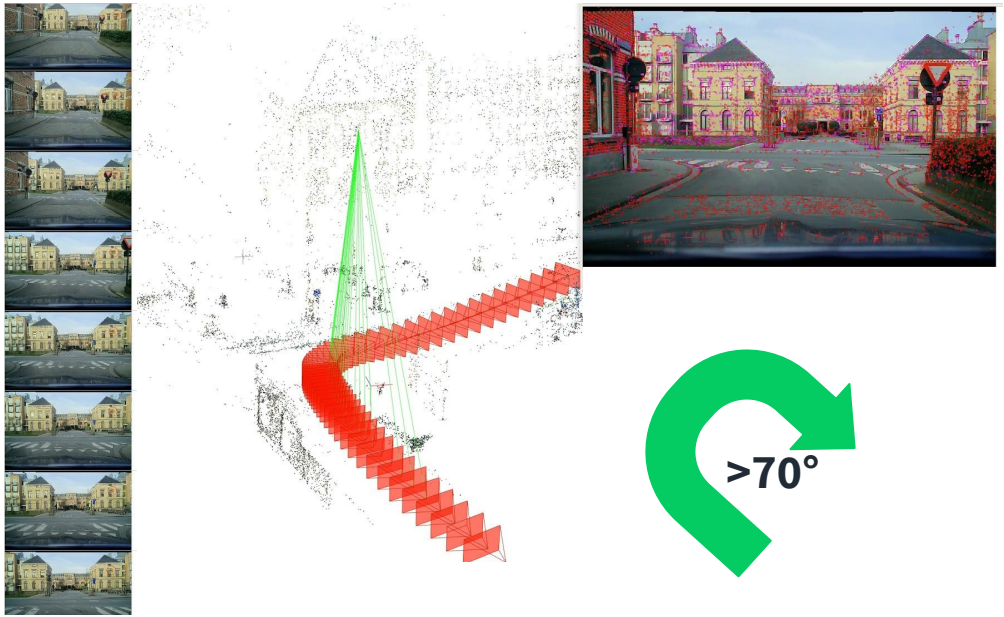
A scalable way to **create metrically accurate depth training data**, suitable for real-world applications, and that is

- ➜ larger, more complex and has diverse environments from around the world
- ➜ comprising many camera types, focal lengths and distortion characteristics
- ➜ containing diverse data for weather, time of day, viewpoint, motion blur, ...



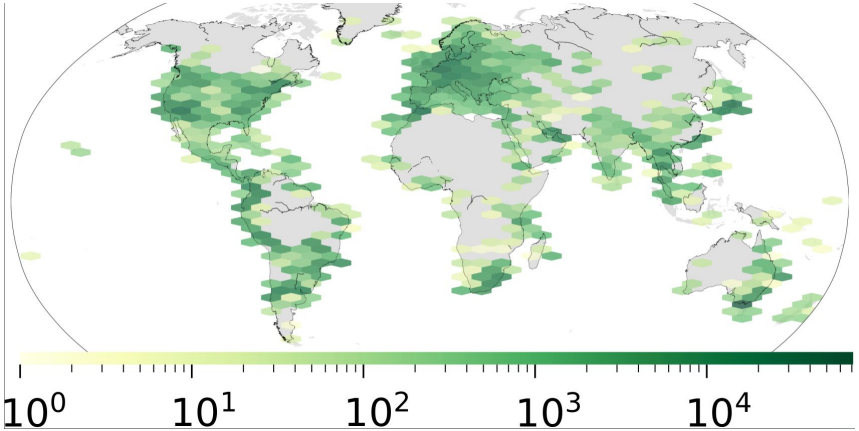
MPSD Data Selection Constraints

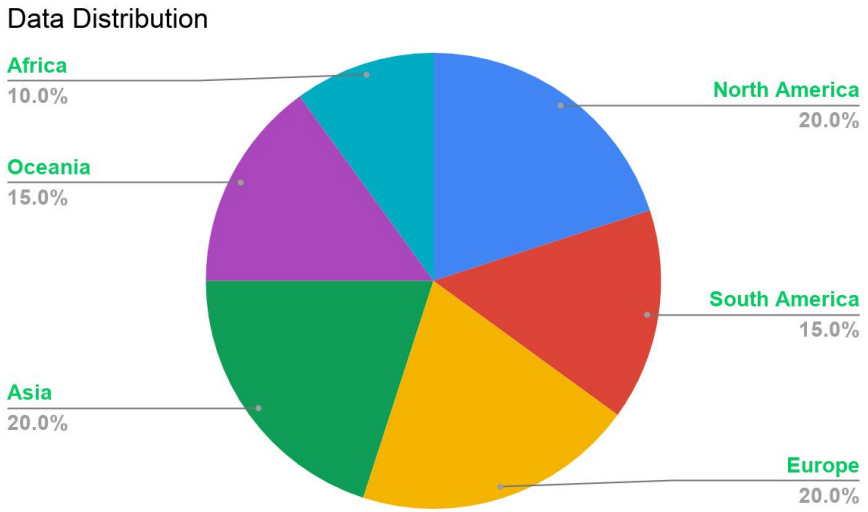
- Dense sampling available (at most 5m and <30° camera turning angle between frames)
- Cumulative trajectory of >70° for better constraining focal length
- Camera parameters are determined by iteratively running OpenSfM per sequence
- Same camera make, model, resolution and focal length data are assigned same parameters
- 10 reconstructions per camera before final set is hand-picked



Geographic data distribution

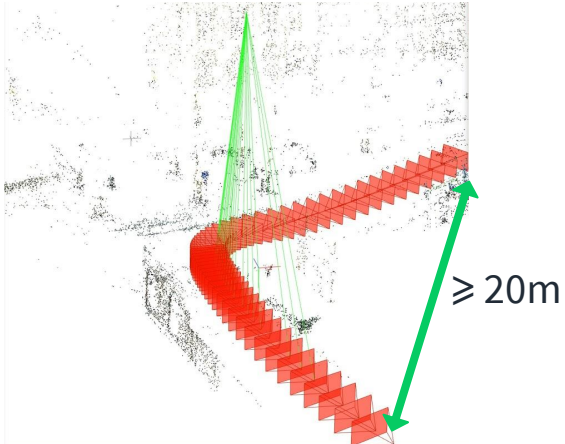
- Sampling from regular grid (156 km^2)
- 250 camera models in final dataset
- 750k images with depth training data





Obtaining metric scale and dense depth

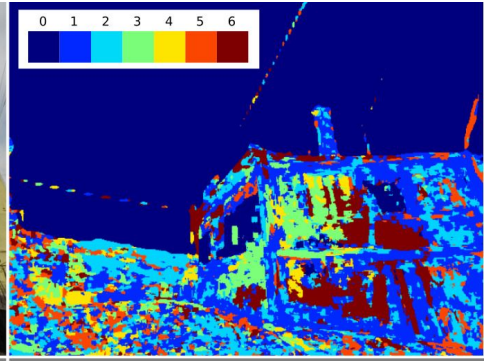
- Cost term proportional to squared distance between (noisy) GPS and estimated camera positions removes scale ambiguity
- Remove outliers due to short sequences and compact reconstructions by filtering (two most distant, resulting camera positions ⩾ 20m)
- Run patch-match multi-view stereo [Shen, 2013], i.e. a winner-takes-all approach based on normalized cross-correlation on depth & normals for corresponding pixels in adjacent images



Filtering dense depth

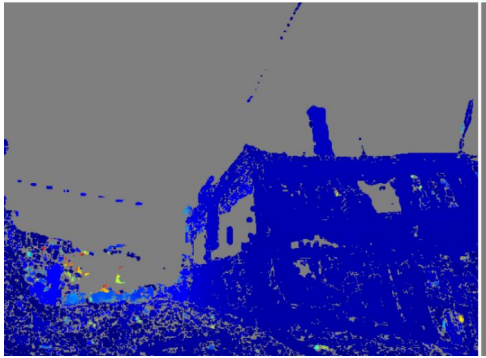
- Patch-match stereo algorithm may contain spurious results
- Cleanup based on consistency checks among three neighboring images

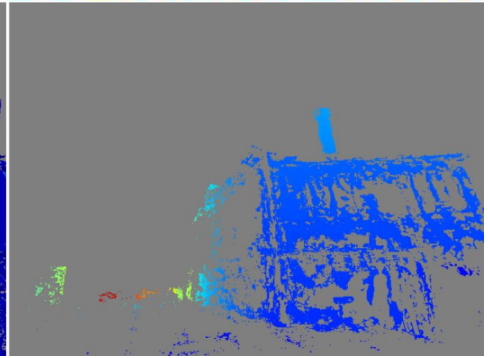
Candidate image



Covisibility

PatchMatch result



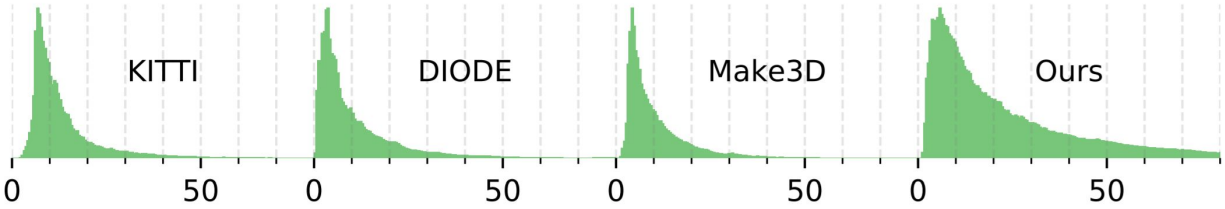


Final, cleaned depth

Dataset overview

Dataset	n. Images	Source	Extent	Metric
Make3D [24]	534	Lidar	Palo Alto	yes
iBims-1 [15]	100	Lidar	Various scenes	yes
DIODE [29]	26 k	Lidar	25 Scenes	yes
KITTI [11]	94 k	Lidar	Karlsruhe	yes
WSVD [30]	1.5M	Stereo	7k videos	no
Cityscapes [5]	25 k	Stereo	50 Cities	yes
MegaDepth [17]	130 k	SfM	200 Scenes	no
MPSD	750 k	SfM	50k Scenes	yes

Comparison of available depth datasets with MPSD



Distributions of volume-normalized depth (m) for several datasets

Training with multiple cameras

- Learning to predict absolute depth from a highly heterogeneous set of cameras negatively affects performance and impacts generalization
- Focal length normalization with per-pixel consideration $(y' = 1)$

$$z = f \frac{y}{y'}$$

z object depth

f focal length

y' object size in image plane [pix]

y real object size [m]

Camera normalization

We apply canonical camera model normalization and resize images by imposing

- Fixed focal length
- Square pixel sensor
- No radial distortion

Example: At a focal length of 720px, a real-world object with height 2m, the estimated depth is inversely proportional to the object size y' n the image.

$$z = 720 * 2/y'$$

Network “only” needs to learn real-world sizes of objects!

Experimental setup

- UNet architecture (ResNet-50 based)
- Dilation rates (1,1,2,4) and output stride x16
- InPlace-ABN to reduce training memory footprint
- DeepLabV3 head (12, 24, 36 dilation rates) + global feature
  - Upsampling to original input resolution in 3 stages
  - Concatenated with size-matching features from encoder
  - Skip-module (CONV+ACT)
- Final bilinear x2 upsampling
- Input size always fixed to 1216x352 @ batch size 64 (8 x V100, 32GB)
- Predicting log of focal-length normalized depth using Eigen-Loss

$$L(z, z^*, f) = \frac{1}{n}\sum_i d_i^2 - \frac{\lambda}{n^2}\left(\sum_i d_i\right)^2$$

$$d_i = \log(z) - \log(z^*)$$

Experimental results

#	Training set	Strategy		KITTI		MegaDepth	Cityscapes		DIODE (outdoor)		Make3D	
		Scale	Crop	SILog	rmse	SILog	SILog	rmse	SILog	rmse	SILog	rmse
1	MD-Ordinal	-	-	30.1	-	10.8	35.19	-	47.52	-	38.2	-
2	MegaDepth	Naive	C	25.61	-	11.86	65.11	-	42.91	-	59.89	-
3	MegaDepth	CC	R	26.92	-	10.67	62.92	-	50.3	-	54.24	-
4	MegaDepth	CC	C	23.79	-	11.51	60.08	-	47.28	-	55.9	-
5	MegaDepth	FF	C	26.79	-	**9.96***	36.73	-	48.28	-	41.64	-
6	mini MPSD	FF	C	14.89	4.87	17.85	22.61	9.05	44.43	8.44	29.55	5.99
7	MPSD	FF	C	**12.77**	4.21	14.68	19.77	**7.91**	42.2	7.78	**27.49**	**5.54**
8	MPSD	CC	C	13.33	**4.13**	21.5	34.83	12.77	43.04	8.05	54.66	59.45
9	MPSD	FF+C	C	12.8	4.39	**14.04**	**19.52**	8.13	**41.69**	**7.75**	28.07	5.67
10	MPSD+KITTI	FF	C	**9.23***	**3.04***	32.23	27.11	8.58	45.55	10.69	37.56	6.49

Prediction results on dynamic objects

Network trained on MPSD and tested on (previously unseen) KITTI data

Training set(s)	Static	Dynamic
MegaDepth	93.04	117.98
MPSD	**4.16**	**5.16**
MegaDepth+KITTI	3.74	4.29
MPSD+KITTI	**3.12**	**3.52**

RMSE on KITTI validation

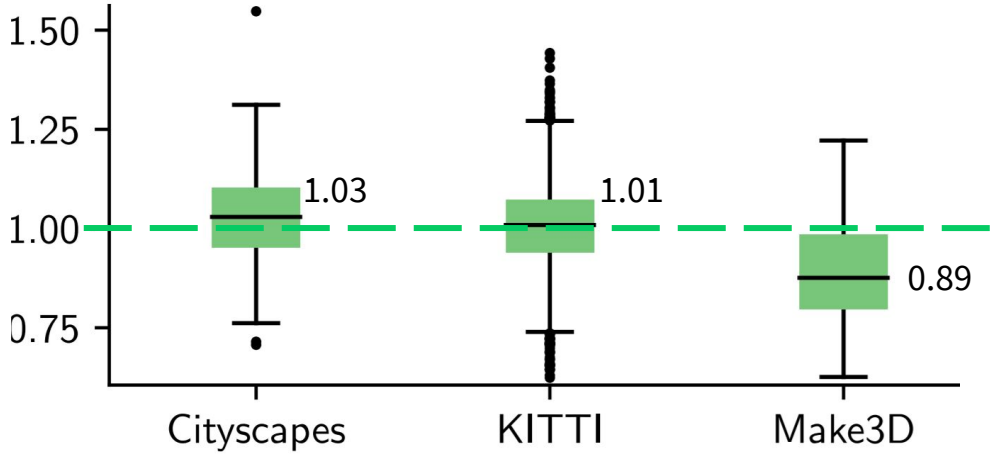
KITTI Depth prediction results

State-of-the-art on **KITTI test data** for 7 months!

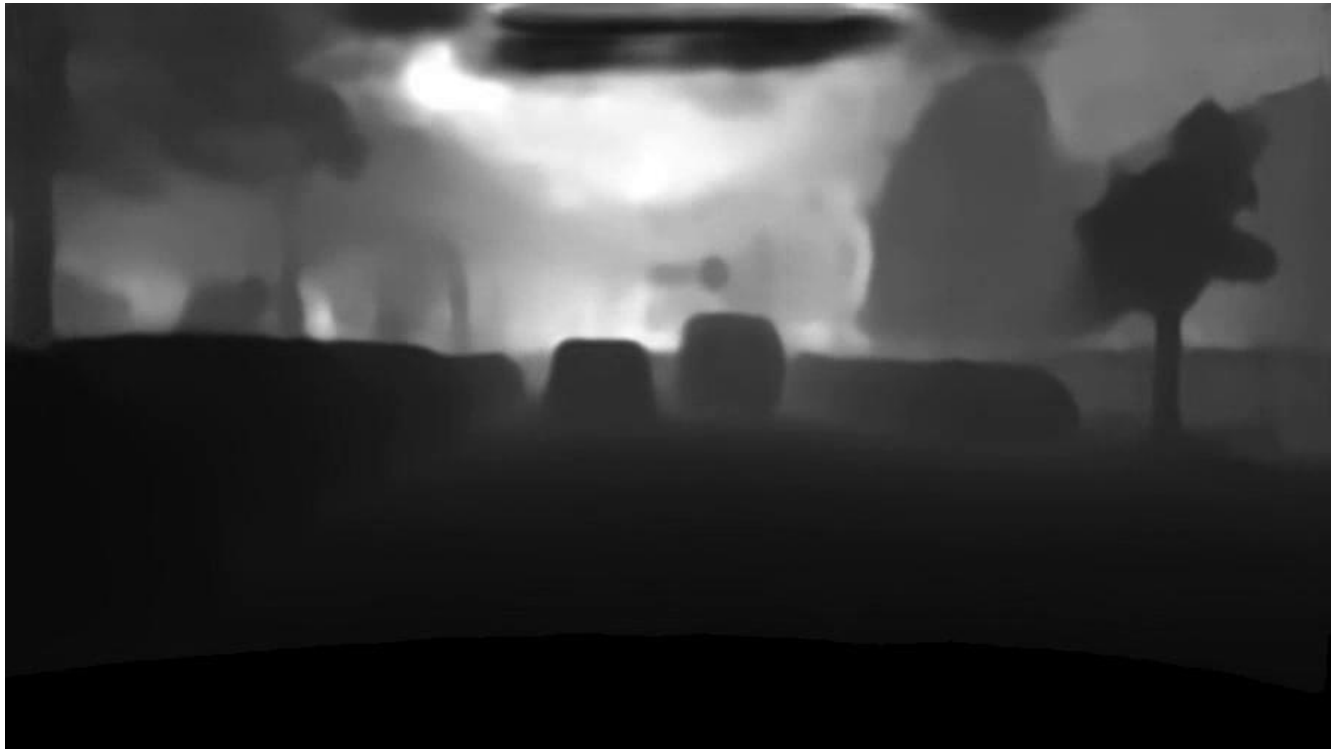
Rank	Method	SILog	sqErrorRel	absErrorRel	iRMSE
1	MPSD	**11.12**	**2.07** %	8.99 %	**11.56**
2	GSM (Anon.)	11.23	2.13 %	8.88 %	12.65
3	GSM (Anon.)	11.56	2.25 %	8.99 %	12.44
4	LCI (Anon.)	11.63	2.20 %	9.07 %	12.42
5	BTS [16]	11.67	2.21 %	9.04 %	12.23
6	AcED (Anon.)	11.70	2.45 %	9.54 %	12.51
7	DORN [10]	11.77	2.23 %	**8.78** %	12.98

Metric depth accuracy validation

Estimated least-square scale correction to describe depth scale bias for network exclusively trained on MPSD and tested on Cityscapes, KITTI, Make3D



Depth estimation in the wild



Learning Multi-Object Tracking and Segmentation from Automatic Annotations [CVPR 2020]

Overview

Joining multi-object tracking and instance segmentation brings mutual benefits, but ground truth data is rare and expensive to annotate

Main contributions:

- Completely **automated generation** of multi-object tracking and segmentation (MOTS) annotations from street-level videos
- **MOTSNet**: a multi-object tracking and segmentation network using a novel “Mask-Pooling” layer to achieve *SOTA* results on multiple benchmarks

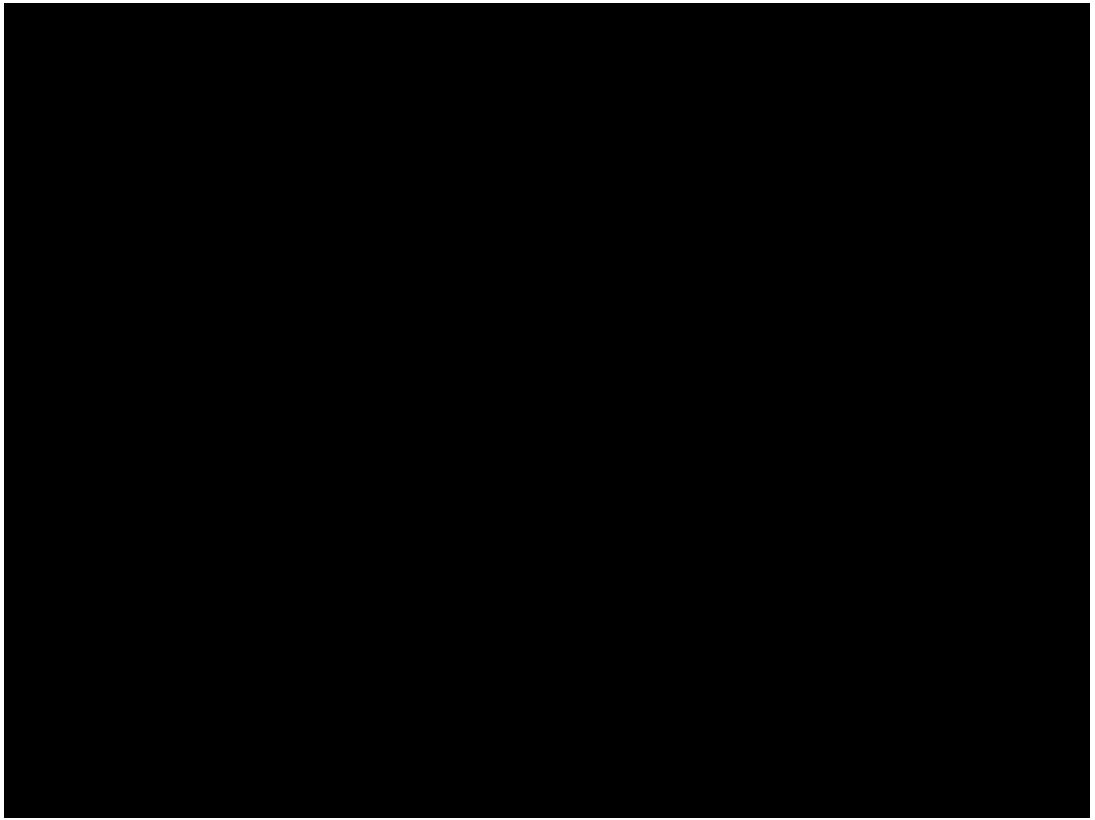
Automatic generation of MOTS annotations

- A Panoptic Segmentation network trained on Mapillary Vistas extracts object segmentations from the input videos
- An optical flow network trained on SfM-generated annotations predicts optical flow on the input videos
- Detected objects are matched across frames by tracking their motion based on the predicted optical flow

No human intervention needed!

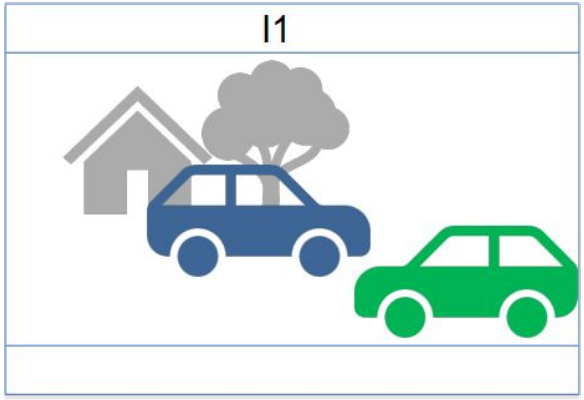


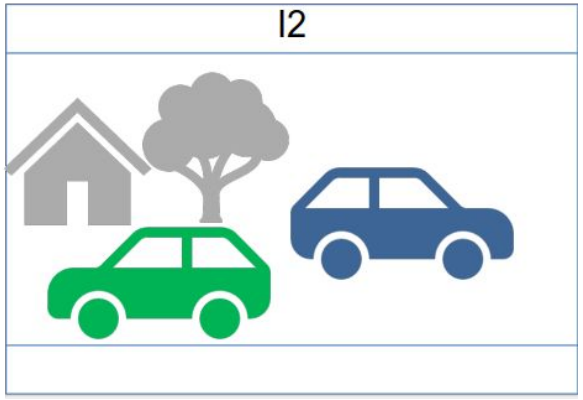
Why trust machine-generated segmentations

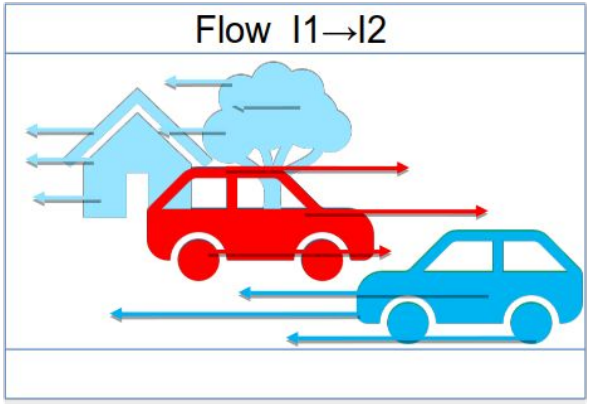


Optical Flow - Introduction

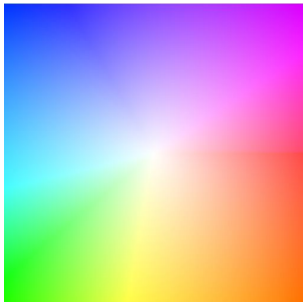
Apparent 2D motion of pixels in image pair







Camera and objects can move



Comparison to Structure-from-Motion

Optical Flow

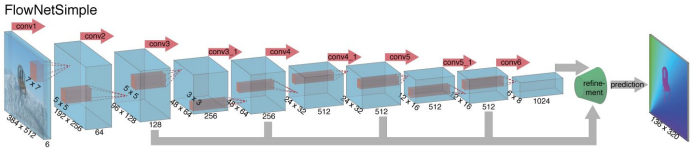
- ➜ Works with static cameras
- ➜ Establishes dense point-wise correspondences
- ➜ Usually from two consecutive images in a video (while there exist multi-frame methods)
- ➜ Can handle dynamic objects in scenes up to certain extent

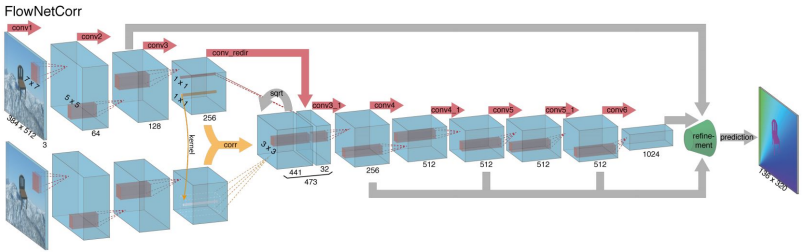
SfM

- ➜ Requires moving cameras
- ➜ Establishes sparse point-wise correspondences
- ➜ Usually based on multiple images
- ➜ Usually gets distracted by dynamic objects in scene

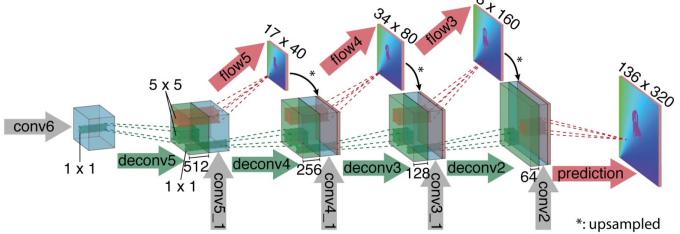
Complementary use cases!

Single-Slide Recap of Optical Flow

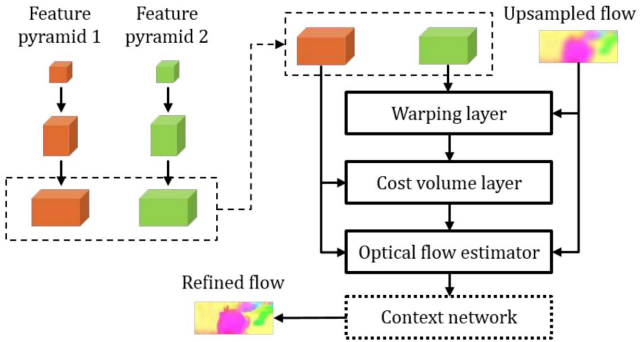




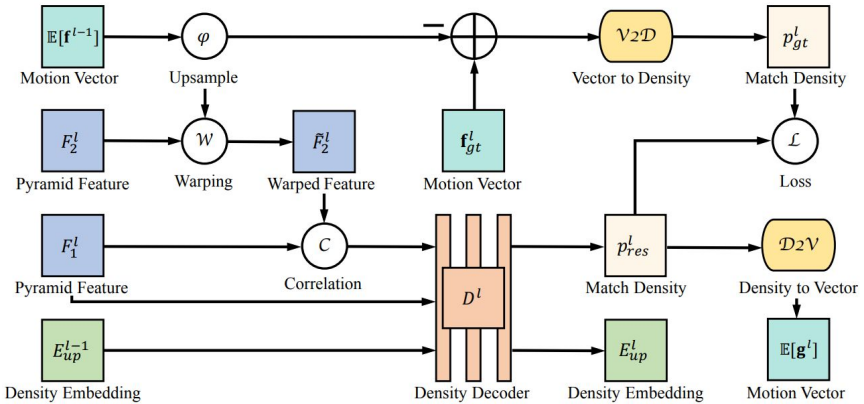
FlowNet:
Conventional Encoder + Decoder Stage



PWC-Net



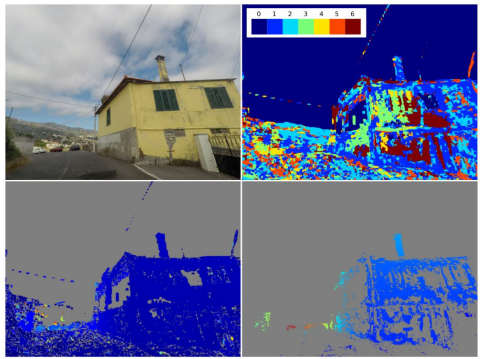
HD^3 (Hierarchical Discrete Distribution Decomposition)



Training data for optical flow networks?

Filtering dense depth

- Patch-match stereo algorithm may contain spurious results
- Cleanup based on consistency checks among three neighboring images

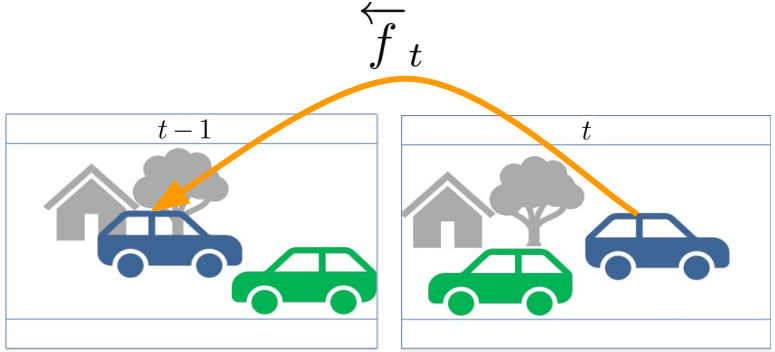


Cleaned covisibility maps can also be used to generate optical flow training data, i.e. we can exploit feature correspondences from multiple views to derive (sparse) flow data.

Leads to pairs of images with sparse flow information from matched points!

Training data for tracking task?

Inductive generation of tracklets per object



$\hat{s} \in \mathcal{S}_{t-1}$ segment in frame $t-1$

$s \in \mathcal{S}_t$ segment in frame t

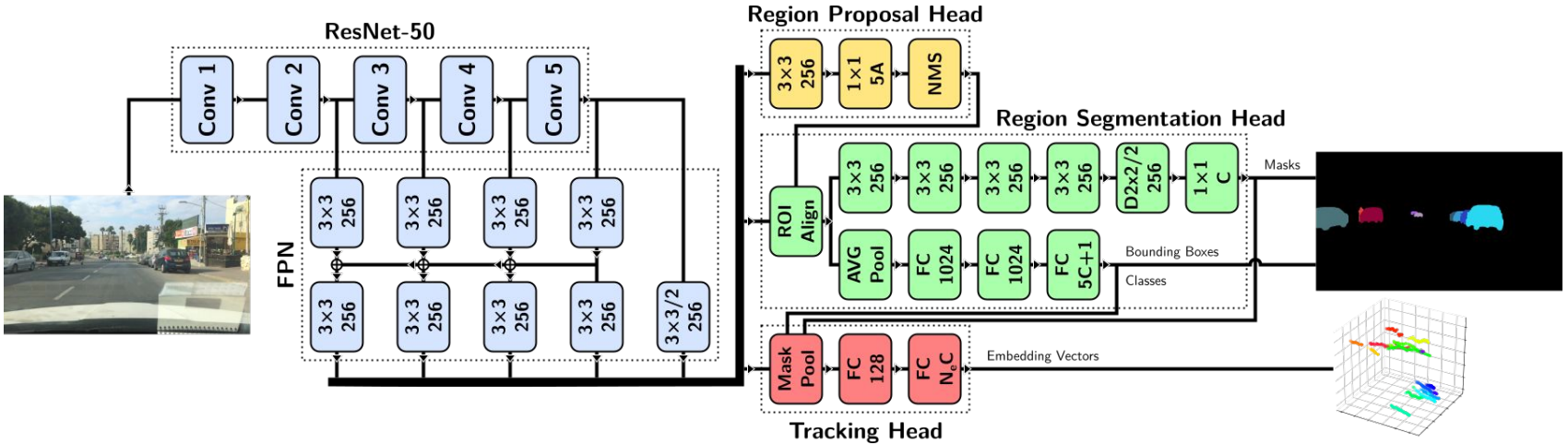
Payoff for linear assignment:

$$\pi(\hat{s}, s) = \mathrm{IoU}(\phi_s, \phi_{\hat{s}} \circ \overleftarrow{f}_t) + \eta(\hat{s}, s) \qquad \eta(\hat{s}, s) \in \{-\infty, 0\}$$

$\eta(\hat{s}, s)$ Encodes additional constraints like matching of segment class labels, minimal overlap checks, IoU differences for largest and second-largest segments, etc.

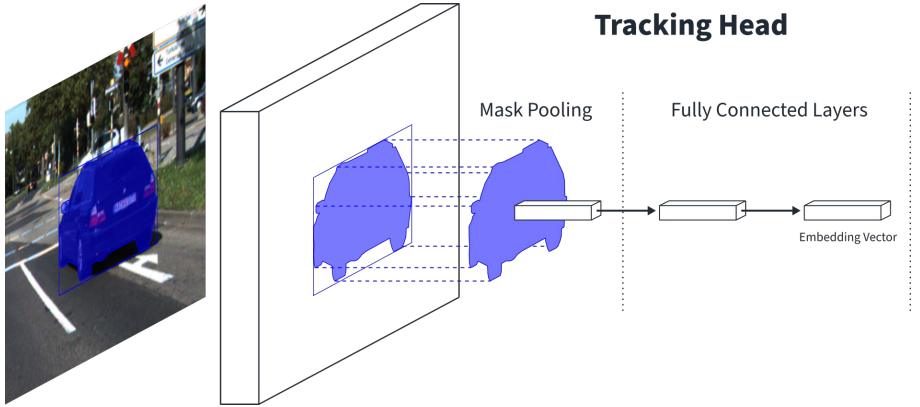
MOTSNet

- Mask R-CNN based architecture with an additional Tracking Head (TH)
- The TH maps detected objects to a learned embedding space for tracking



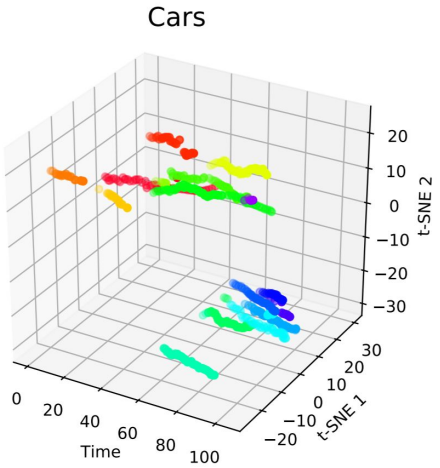
Tracking Head and Mask Pooling

- Pool features under the instance segmentation masks
- Process with FC layers to compute embedding vectors
- Compare embedding vectors across frames to match objects



Training and Inference

- Tracking-head optimization based on hard triplet loss [Hermans *et al.*, 2017], learning to generate object-specific embedding vectors that are similar for matching and dissimilar for non-matching objects.
- Inference based on embeddings, but similar to training tracklet generation



Experimental Setup

Evaluation on **KITTI MOTS**, **MOTSChallenge** (MOTS ground truth available) [Voigtländer et al., CVPR 2019] and BDD100k tracking data (bounding box tracking information available)

ResNet-50 backbone in all our experiments

Evaluation on KITTI MOTS:

- Quality assessment of dataset generation (KITTI Synth)
- MOTSNet ablation and evaluation

KITTI Synth Experiments

Generated training data from KITTI Raw (142 sequences, excluding validation set of KITTI MOTS), yields 1.25M object segments in ~44k images

Method	Pre-training	sMOTSA		MOTSA		MOTSP		mAP	
		Car	Ped	Car	Ped	Car	Ped	Box	Mask
KITTI Synth (val) + HD^3 [49] model zoo	inference only	65.4	45.7	77.3	66.3	**87.6**	76.6	–	–
KITTI Synth (val) + HD^3, KITTI-SfM	inference only	65.5	45.4	77.4	66.0	**87.6**	76.6	–	–
MOTSNet with:									
AVEBOX+TH	I	73.7	46.4	85.8	62.8	86.7	76.7	57.4	50.9
AVEMSK-TH	I	76.4	44.0	88.5	60.3	86.8	76.6	57.8	51.3
AVEBOX-TH	I	75.4	44.5	87.3	60.8	86.9	76.7	57.5	51.0
KITTI MOTS train sequences only	I	72.6	45.1	84.9	62.9	86.1	75.6	52.5	47.6
MOTSNet	I	77.6	49.1	89.4	65.6	87.1	76.4	58.1	51.8
MOTSNet	I, M	**77.8**	**54.5**	**89.7**	**70.9**	87.1	**78.2**	**60.8**	**54.1**

Results on KITTI MOTS validation data

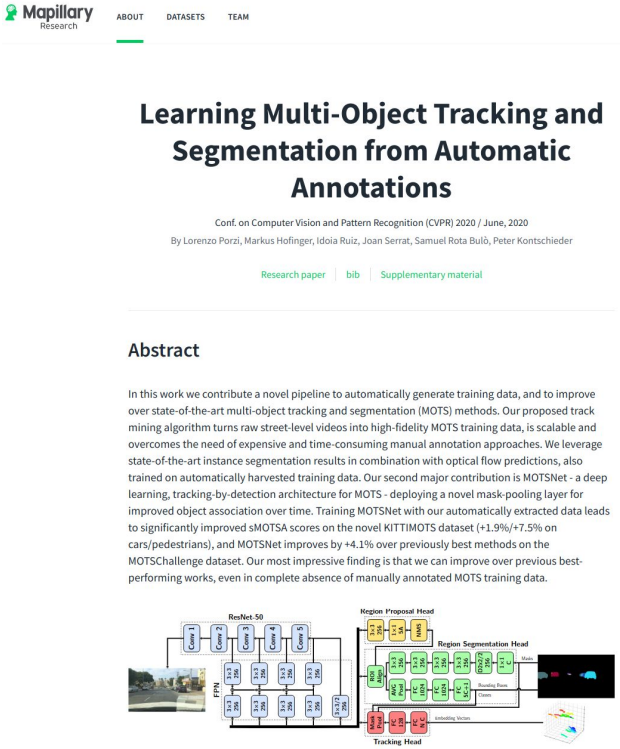
Method	Pre-training	sMOTSA		MOTSA		MOTSP		mAP	
		Car	Ped	Car	Ped	Car	Ped	Box	Mask
TrackR-CNN	I, C, M	76.2	47.1	87.8	65.5	87.2	75.7	–	–
CAMOT	I, C, M	67.4	39.5	78.6	57.6	86.5	73.1	–	–
CIWT	I, C, M	68.1	42.9	79.4	61.0	86.7	75.7	–	–
BeyondPixels	I, C, M	76.9	–	**89.7**	–	86.5	–	–	–
MOTSNet	I	69.0	45.4	78.7	61.8	88.0	76.5	55.2	49.3
	I, M	74.9	53.1	83.9	67.8	89.4	79.4	60.8	54.9
	I, KS	76.4	48.1	86.2	64.3	88.7	77.2	59.7	53.3
	I, M, KS	**78.1**	**54.6**	87.2	**69.3**	**89.6**	**79.7**	**62.4**	**55.7**

Results on MOTSChallenge

Method	Pre-training	sMOTSA	MOTSA	MOTSP
TrackR-CNN [7]	I, C, M	52.7	66.9	80.2
MHT-DAM [4]	I, C, M	48.0	62.7	79.8
FWT [2]	I, C, M	49.3	64.0	79.7
MOTDT [5]	I, C, M	47.8	61.1	80.0
jCC [3]	I, C, M	48.3	63.0	79.9
MOTSNet	I	41.8	55.2	78.4
	I, C	**56.8**	**69.4**	**82.7**

Results on KITTI MOTS / BDD100k

More Results



Drop by our virtual presentation at Poster Session 2.2 for more information!

Date: Wednesday, June 17 & Thursday, June 18 2020

Q&A Time: 1200–1400 and 0000–0200

Session: Poster 2.2 — Face, Gesture, and Body Pose; Motion and Tracking; Representation Learning

Presentation times 12:00 and 00:00 (Pacific Time Zone [Seattle time])

ID 5452

Summary

- Using less supervision, we obtain state-of-the-art results for
  - Single-Image depth estimation
  - Multi-object tracking and segmentation
- Mapillary-scale data for learning single-image depth estimation, extracted from multiple cameras and all around the globe, using SfM
- SOTA recognition algorithms for automatically mining training data is beneficial for MOTS. Even possible to outperform methods based on manually annotated data

Let’s create something amazing together!

@mapillary

Mapillary