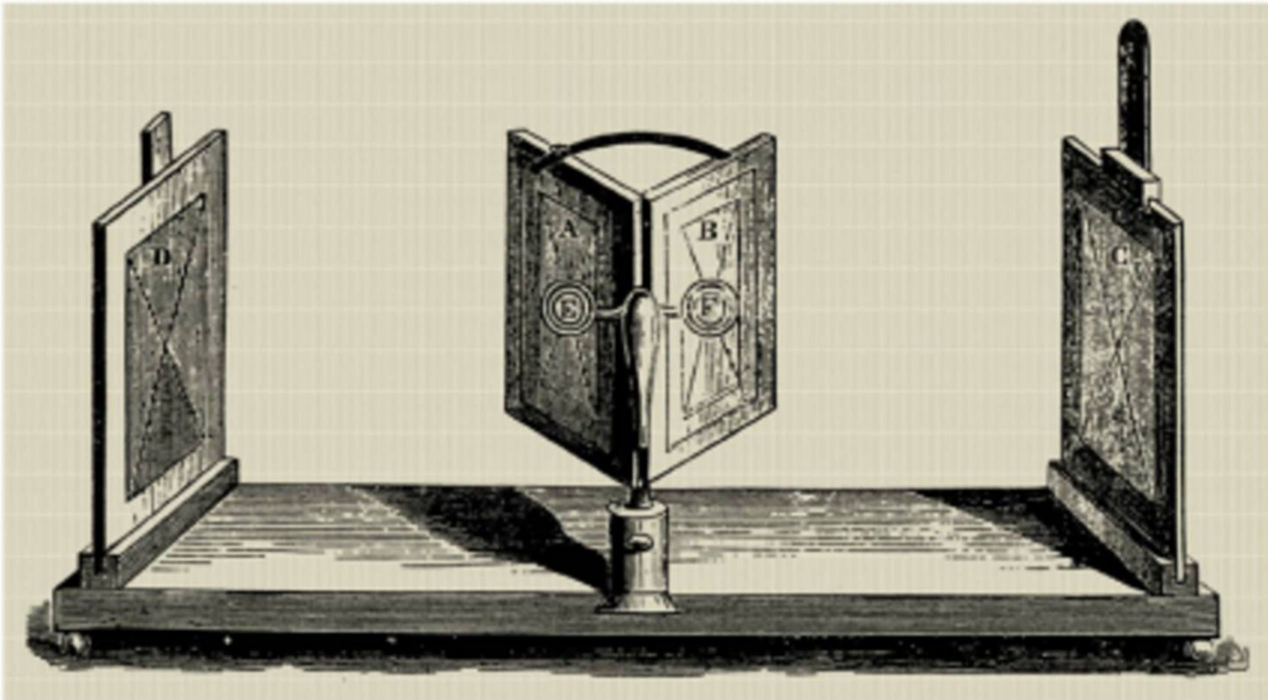
Engaging Content
Engaging People

Learning to See in 3D: From Lidar to Synthetic Data Generation

Hossein Javidnia, PhD
ADAPT Centre, Trinity College Dublin

Early Stage of Depth Sensing

- First Stereoscope was invented in 1932 by Sir Charles Wheatstone.



- Tremendous improvement since the invention of stereoscope in depth sensing technologies.

- Early 2000 was the beginning of the new era so-called “3D Revolution”.

	Time of flight	Stereoscopic vision	Fixed structured light	Programmable structured light	LIDAR	Learning based
Operational principle	IR pulse, measure light transit time	Two 2D sensors emulate human eyes	Single pattern visible or IR illumination, detects distortion	Multiple pattern visible or IR illumination, detects distortion	Laser illumination	Trained model using deep learning
Point cloud generation	Direct out of chipset	High SW Processing	Medium SW processing	SW processing scales with # of patterns	Direct out of chipset	High SW Processing
Active illumination	Yes	No	Yes	Yes – customizable spectrum	Yes	No
Low light performance	Good	Weak	Good	Good	Good	Weak - Unless trained for that
Bright light performance	Medium	Good	Medium / weak Depends on illumination	Medium / weak Depends on illumination power	Good	Medium
Power consumption	Medium/high / Scales with distance	Low	Medium	Medium / Scales with distance	High	Low/Medium
Range	Short to long range Depends on laser power & modulation	Mid range Depends on spacing between cameras	Very short to mid range Depends on illumination power	Very short to mid range Depends on illumination power	Short to very long range	Mid range
Resolution	QQVGA, QVGA	Camera Dependent	Projected pattern dependent	WVGA to 1080p	Depends on the laser module	QQVGA, QVGA
Depth accuracy	mm to cm Depends on resolution of sensor	mm to cm Difficulty with smooth surface	mm to cm	μm to cm	mm	mm to m / Depends on the trained model
Scanning speed	Fast Limited by sensor speed	Medium Limited by SW complexity	Medium Limited by SW complexity	Fast / medium Limited by camera speed	Fast/medium Limited by sensor speed	Medium Limited by SW complexity
Other Strengths	*The scene is recorded all at once and doesn't have to be scanned *2D and 3D information in a multi-part image *Compact system without moving components *No structure or contrast required *Large working distances are possible with a sufficiently strong light source *Low overall system costs *High real-time capability	*Possibility to achieve high accuracy at short range *2D area scan cameras can be used	*Possibility to achieve high accuracy at short range *2D area scan cameras can be used *Can be optimized for real-time applications		*Very high accuracy *Difficult lighting conditions are not a problem *No problems with mirroring or highly reflective surfaces *Suitable for real-time applications	*Can be optimized for low resolution real-time applications *There is a potential to estimate depth from one camera
Other Weaknesses	*Sensitive to scattered light *Difficulties with sunlight	*Will not work on homogeneous surfaces *High computing load makes real-time capability difficult *Exposure to sunlight is a problem *Will Not work with highly reflective surfaces	*High overall system costs due to complex setup and high installation cost *Limited to short scanning range		*Very expensive individual components *High overall system costs due to complex setup and high installation cost	*Highly dependent on graphics processing unit power *High computing load makes real-time capability difficult for high resolution images *There is no guarantee to achieve a specific accuracy

Deep Learning and Depth Sensing

Deep learning methods are applied to 3 categories of applications:

1. Depth from Stereo Camera
2. Depth from Monocular Camera
3. SLAM+Deep Learning (Recent)

KITTI Benchmark by:

Karlsruhe Institute of Technology and Toyota Technological Institute at Chicago

http://www.cvlibs.net/datasets/kitti/eval_scene_flow.php?benchmark=stereo

	Method	Setting	Code	D1-bg	D1-fg	D1-all	Density	Runtime	Environment
1	M2S_CSPN			1.51 %	2.88 %	1.74 %	100.00 %	0.5 s	GPU @ 2.5 Ghz (C/C++)
X. Cheng, P. Wang and R. Yang: Learning Depth with Convolutional Spatial Propagation Network. arXiv preprint arXiv:1810.02695 2018.									
2	Samsung_System_LSI			1.55 %	3.82 %	1.93 %	100.00 %	0.4 s	GPU @ 2.5 Ghz (Python)
3	MS_CSPN			1.56 %	3.78 %	1.93 %	100.00 %	0.5 s	GPU @ 2.5 Ghz (C/C++)
X. Cheng, P. Wang and R. Yang: Learning Depth with Convolutional Spatial Propagation Network. arXiv preprint arXiv:1810.02695 2018.									
4	NCA-Net			1.68 %	3.28 %	1.94 %	100.00 %	0.5 s	GPU @ 2.5 Ghz (Python)
5	PSMNet_R			1.62 %	3.79 %	1.98 %	100.00 %	0.5 s	GPU @ 2.5 Ghz (Python)
6	DSHNet			1.65 %	4.29 %	2.09 %	100.00 %	0.7 s	Nvidia GTX Titan Xp
7	EMCUA			1.66 %	4.27 %	2.09 %	100.00 %	0.9 s	1 core @ 2.5 Ghz (C/C++)
8	KesonStereo_V1			1.77 %	3.74 %	2.09 %	100.00 %	0.4 s	GPU @ 2.5 Ghz (Python)
9	open-depth			1.76 %	3.84 %	2.10 %	100.00 %	0.51 s	NVIDIA TITAN Xp (PyTorch 0.4.0)
10	GwcNet-g			1.74 %	3.93 %	2.11 %	100.00 %	0.32 s	GPU @ 2.0 Ghz (Python + C/C++)
11	IPSM Net			1.72 %	4.11 %	2.12 %	100.00 %	0.4 s	1 core @ 2.5 Ghz (C/C++)
12	DM-Net			1.69 %	4.29 %	2.12 %	100.00 %	0.9s	1 core @ 2.5 Ghz (Python)
13	DM-Net-i			1.69 %	4.38 %	2.14 %	100.00 %	0.40s	Titan XP
14	HSM			1.80 %	3.85 %	2.14 %	100.00 %	0.15 s	GPU @ 2.5 Ghz (Python)
15	Stereo-fusion-SJTU			1.87 %	3.61 %	2.16 %	100.00 %	0.7 s	Nvidia GTX Titan Xp
X. Song, X. Zhao, H. Hu and L. Fang: EdgeStereo: A Context Integrated Residual Pyramid Network for Stereo Matching. Asian Conference on Computer Vision (ACCV) 2018.									
16	EdgeStereo-V2			1.87 %	3.99 %	2.23 %	100.00 %	0.31 s	Nvidia GTX Titan Xp
X. Song, X. Zhao, H. Hu and L. Fang: EdgeStereo: A Context Integrated Residual Pyramid Network for Stereo Matching. Asian Conference on Computer Vision (ACCV) 2018.									
17	TinyStereo_V2			1.93 %	3.76 %	2.24 %	100.00 %	0.4 s	GPU @ 2.5 Ghz (Python)
18	SegStereo			1.88 %	4.07 %	2.25 %	100.00 %	0.6 s	Nvidia GTX Titan Xp
G. Yang, H. Zhao, J. Shi, Z. Deng and J. Jia: SegStereo: Exploiting Semantic Information for Disparity Estimation. arXiv preprint arXiv:1807.11699 2018.									
19	HDU-LJJ-Group			1.82 %	4.42 %	2.25 %	100.00 %	0.47 s	GPU @ 1.5 Ghz (Python)
20	Stereo-DRNet			1.72 %	4.95 %	2.26 %	100.00 %	0.23 s	Nvidia GTX 1080 Ti (Pytorch)
21	PASM			1.78 %	4.64 %	2.26 %	100.00 %	0.52 s	1 core @ 2.5 Ghz (C/C++)
22	MPSMNet			1.78 %	4.63 %	2.26 %	100.00 %	1.0 s	GPU @ 2.5 Ghz (Python)
23	MSDC-Net			1.96 %	3.77 %	2.26 %	100.00 %	0.6 s	GPU @ 2.5 Ghz (Python)
24	TinyStereo			1.92 %	4.13 %	2.28 %	100.00 %	0.39 s	1 core @ 2.5 Ghz (C/C++)
25	PSMNet_ROB			1.79 %	4.92 %	2.31 %	100.00 %	0.41 s	1 core @ 2.5 Ghz (Python)
26	MeituNet			1.88 %	4.48 %	2.31 %	100.00 %	0.51 s	GPU @ 2.5 Ghz (Python)

Challenges

1. Computationally expensive for LIDAR and stereo cameras

2. Optically not possible for Monocular cameras

3. Power resources limitation

VGG16: (3, 224, 224)

Layer (type)	Output Shape	Param #	MACC #
Conv2d-1	[-1, 64, 224, 224]	1,792	0.56%
ReLU-2	[-1, 64, 224, 224]	0	-
Conv2d-3	[-1, 64, 224, 224]	36,928	11.96%
ReLU-4	[-1, 64, 224, 224]	0	-
MaxPool2d-5	[-1, 64, 112, 112]	0	-
Conv2d-6	[-1, 128, 112, 112]	73,856	5.98%
ReLU-7	[-1, 128, 112, 112]	0	-
Conv2d-8	[-1, 128, 112, 112]	147,584	11.96%
ReLU-9	[-1, 128, 112, 112]	0	-
MaxPool2d-10	[-1, 128, 56, 56]	0	-
Conv2d-11	[-1, 256, 56, 56]	295,168	5.98%
ReLU-12	[-1, 256, 56, 56]	0	-
Conv2d-13	[-1, 256, 56, 56]	590,080	11.96%
ReLU-14	[-1, 256, 56, 56]	0	-
Conv2d-15	[-1, 256, 56, 56]	590,080	11.96%
ReLU-16	[-1, 256, 56, 56]	0	-
MaxPool2d-17	[-1, 256, 28, 28]	0	-
Conv2d-18	[-1, 512, 28, 28]	1,180,160	5.98%
ReLU-19	[-1, 512, 28, 28]	0	-
Conv2d-20	[-1, 512, 28, 28]	2,359,808	11.96%
ReLU-21	[-1, 512, 28, 28]	0	-
Conv2d-22	[-1, 512, 28, 28]	2,359,808	11.96%
ReLU-23	[-1, 512, 28, 28]	0	-
MaxPool2d-24	[-1, 512, 14, 14]	0	-
Conv2d-25	[-1, 512, 14, 14]	2,359,808	2.99%
ReLU-26	[-1, 512, 14, 14]	0	-
Conv2d-27	[-1, 512, 14, 14]	2,359,808	2.99%
ReLU-28	[-1, 512, 14, 14]	0	-
Conv2d-29	[-1, 512, 14, 14]	2,359,808	2.99%
ReLU-30	[-1, 512, 14, 14]	0	-
MaxPool2d-31	[-1, 512, 7, 7]	0	-
Linear-32	[-1, 4096]	102,764,544	0.66%
ReLU-33	[-1, 4096]	0	-
Dropout-34	[-1, 4096]	0	-
Linear-35	[-1, 4096]	16,781,312	0.11%
ReLU-36	[-1, 4096]	0	-
Dropout-37	[-1, 4096]	0	-
Linear-38	[-1, 1000]	4,097,000	0.03%

Total params: 138,357,544

Total MACC: 15,470,264,320

Trainable params: 138,357,544

Non-trainable params: 0

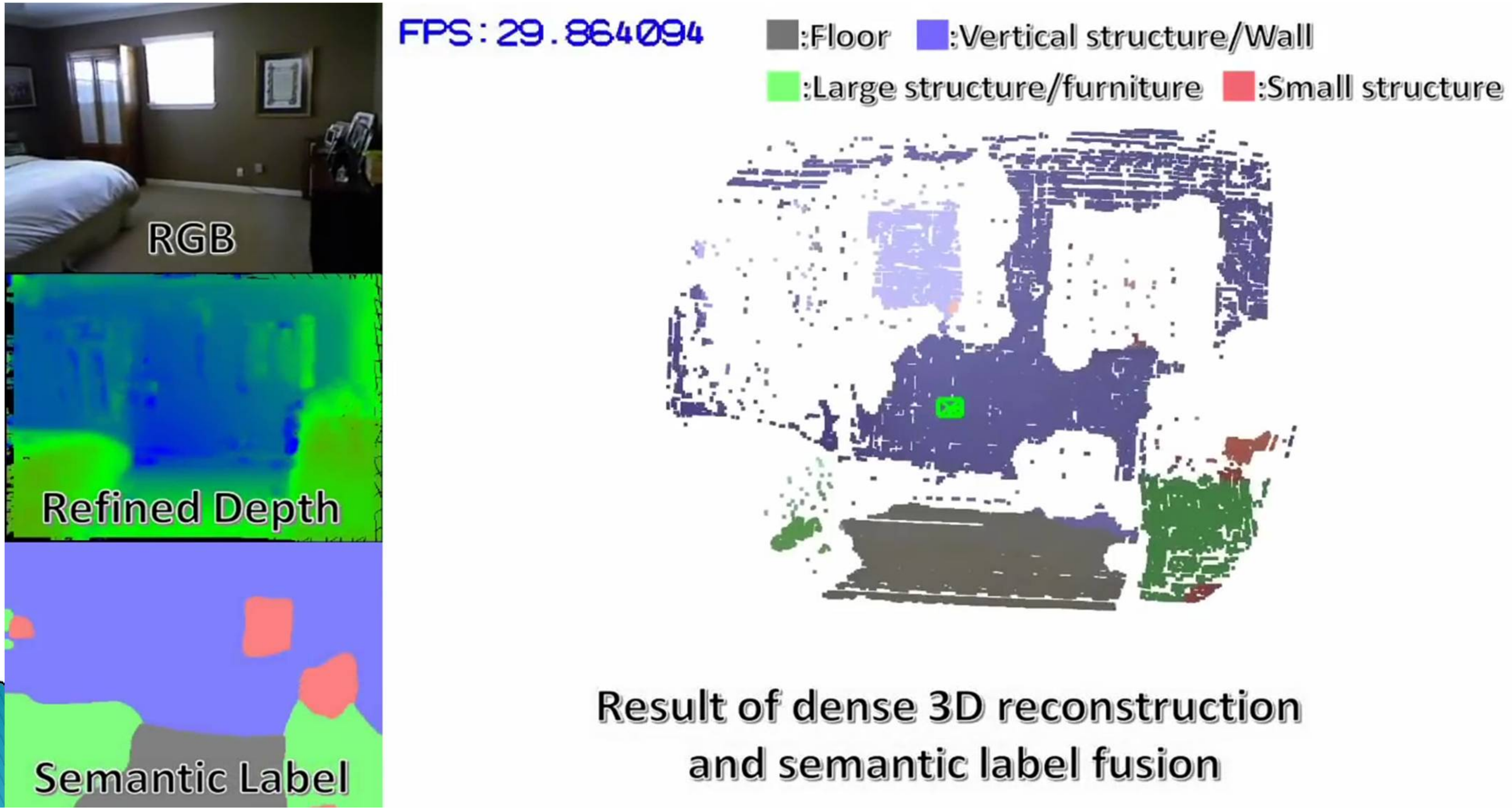
Input size (MB): 0.57

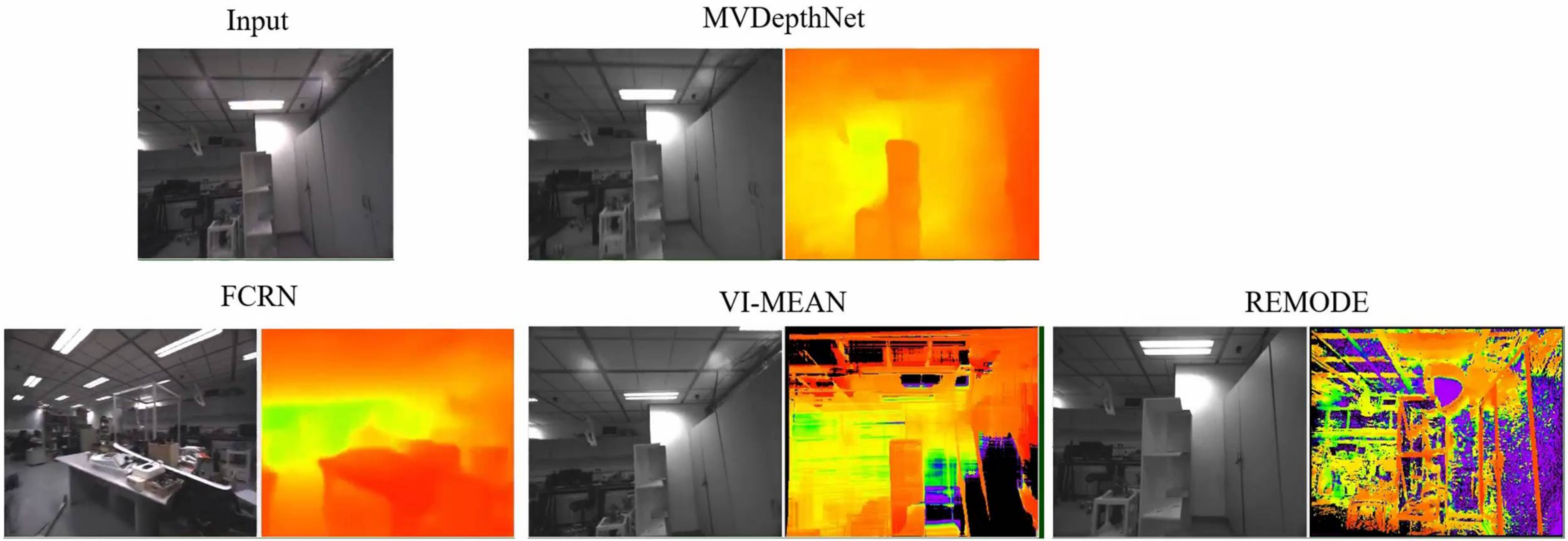
Forward/backward pass size (MB): 218.59

Params size (MB): 527.79

Estimated Total Size (MB): 746.96

CNN SLAM https://arxiv.org/abs/1704.03489





https://arxiv.org/abs/1807.08563

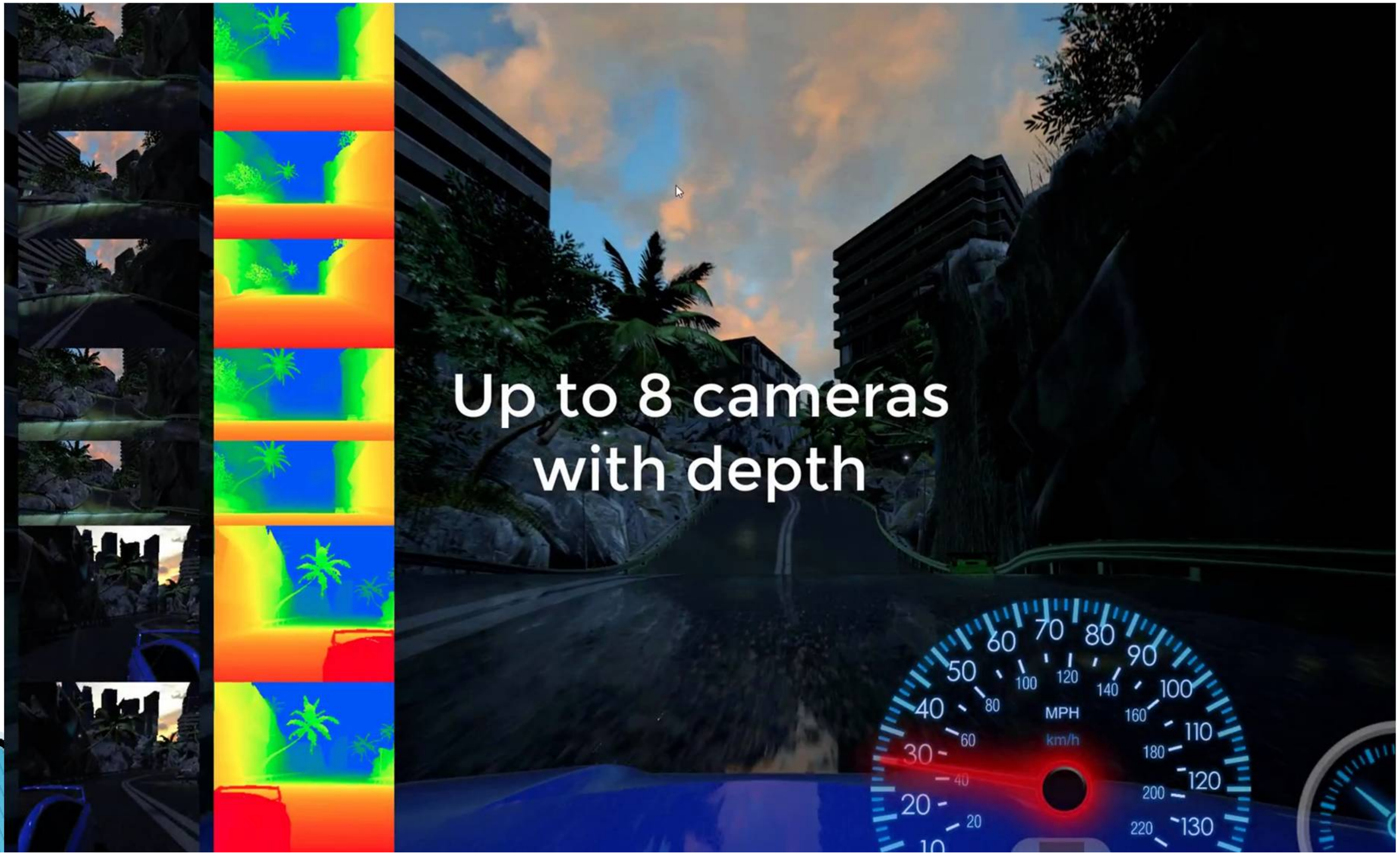
- The networks are trained on the data captured by a LIDAR scanner or consumer depth sensors.

- The main challenges with this type of ground-truth generation are the data sparsity and expensive components.

Simulation Platforms

- Open source
- Easy to implement
- Capturing thousands/millions of frames within a short time frame
- Variety of testing environment

DeepDrive.io



Microsoft AirSim