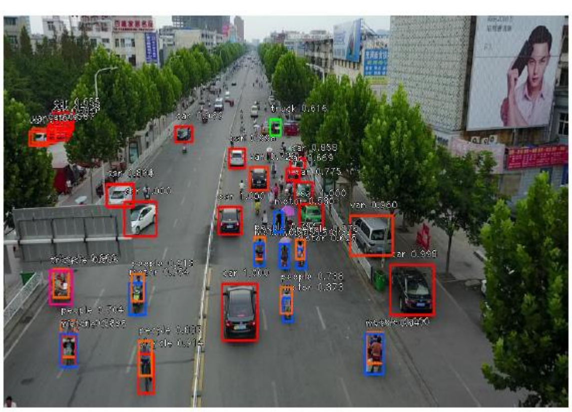
Robotic Vision

E.M. Bakker



From [10], S. Vaddi et al., 2019.

Honda Asimo (From: zdnet.com)

Overview

- OpenCV
- Some Neural Networks and AlexNet

Computer Vision and Pattern Recognition (CVPR)
- Object Tracking
- Human Robot Interaction
- Pose Estimation, Face Recognition, …
- Some problems with Neural Networks
- Data fusion …

OpenCV

- Low level image processing.
- Convolutional Kernels: filters, edge detectors, etc.

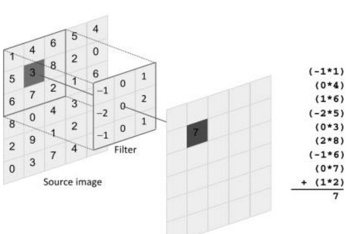


Image from [1].

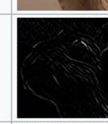
The general expression of a convolution is

$$g(x,y) = \omega * f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s,t) f(x-s, y-t),$$

where $g(x,y)$ is the filtered image, $f(x,y)$ is the original image, ω is the filter kernel. Every element of the filter kernel is considered by $-a \leq s \leq a$ and $-b \leq t \leq b$.

Wikipedia

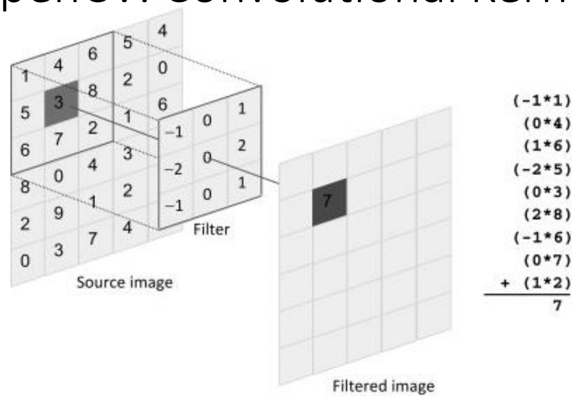
- Blob tracking
- Face and people detector
- Neural networks

Operation	Kernel ω	Image result g(x,y)
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	

Wikipedia

[1] https://www.sciencedirect.com/topics/computer-science/convolution-filter

OpenCV: Convolutional Kernels



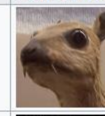
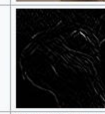
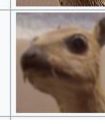
[1] https://www.sciencedirect.com/topics/computer-science/convolution-filter

The general expression of a convolution is

$$g(x,y) = \omega * f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s,t) f(x-s, y-t),$$

where $g(x,y)$ is the filtered image, $f(x,y)$ is the original image, ω is the filter kernel. Every element of the filter kernel is considered by $-a \leq s \leq a$ and $-b \leq t \leq b$.

Wikipedia

Operation	Kernel ω	Image result g(x,y)
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	

Wikipedia

OpenCV

- Low level image processing.
- Convolutional Kernels: filters, edge detectors, etc.

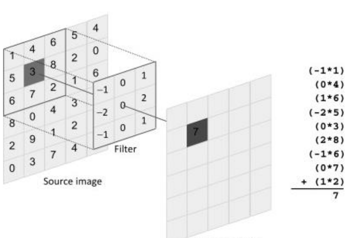


Image from [1].

The general expression of a convolution is

$$g(x,y) = \omega * f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s,t) f(x-s, y-t),$$

where $g(x,y)$ is the filtered image, $f(x,y)$ is the original image, ω is the filter kernel. Every element of the filter kernel is considered by $-a \leq s \leq a$ and $-b \leq t \leq b$.

Wikipedia

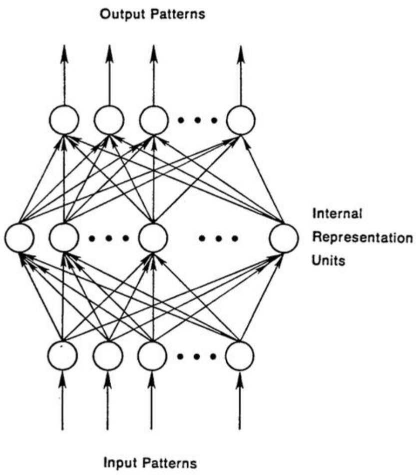
- Blob tracking
- Face and people detector
- Neural networks

Operation	Kernel ω	Image result g(x,y)
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	

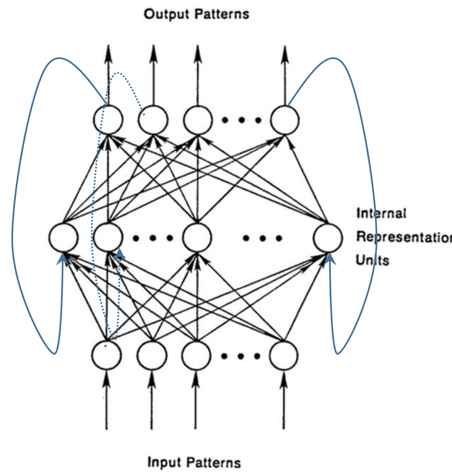
[1] https://www.sciencedirect.com/topics/computer-science/convolution-filter

Wikipedia

Some Neural Networks



Feed Forward Neural Network



Recurrent Neural Network

… -> To the ZOO

DNN: AlexNet, VGG16, ResNet, etc.

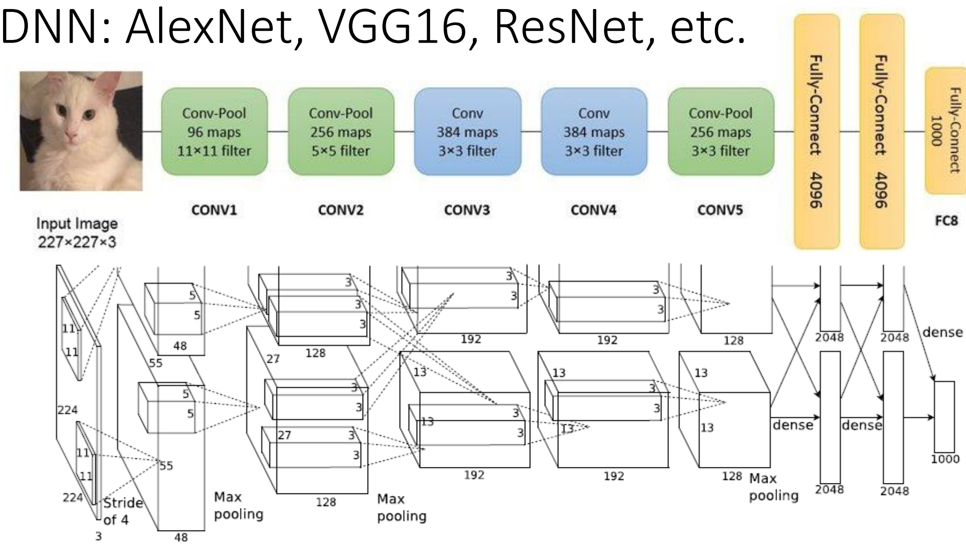
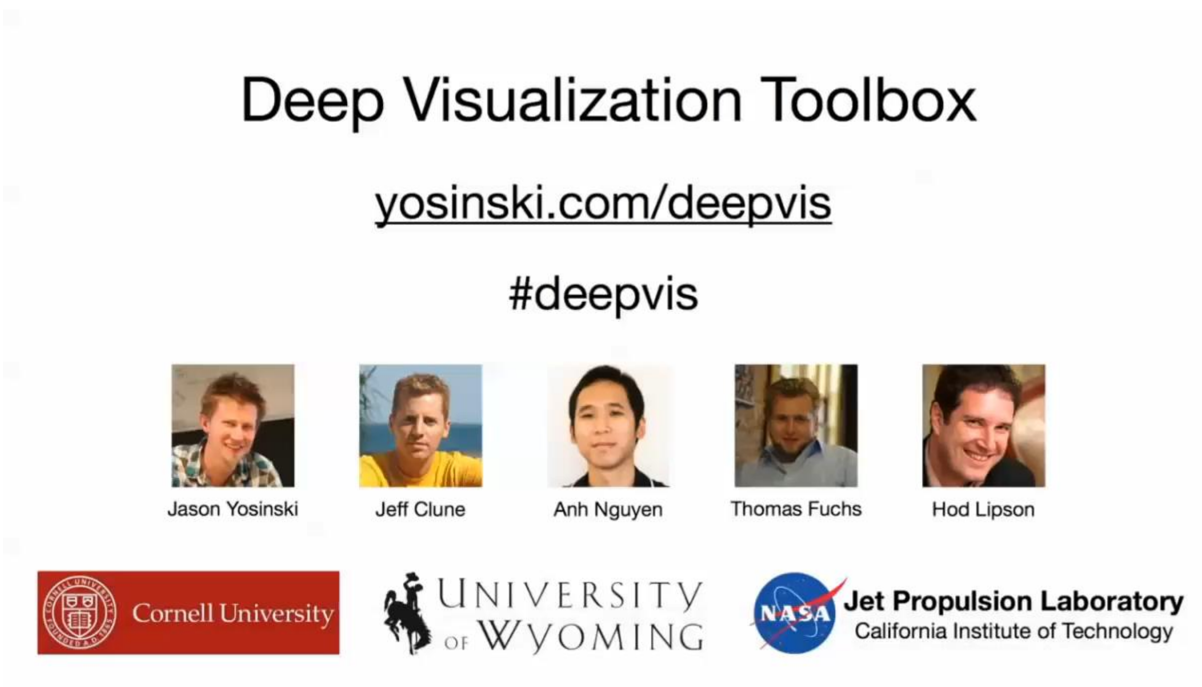


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

Krizhevsky, Alex; Sutskever, Ilya; Hinton, Geoffrey E. "ImageNet classification with deep convolutional neural networks" Communications of the ACM. 60 (6): 84–90. 2012



ImageNet

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, **ImageNet: A Large-Scale Hierarchical Image Database.** *IEEE Computer Vision and Pattern Recognition (CVPR), 2009.* pdf | BibTex

- #images: 14,197,122
- # non-empty WordNet synsets: 21,841
- # images with bounding box: 1,034,908
- # synsets with SIFT features: 1000
- # images with SIFT features: 1.2 million

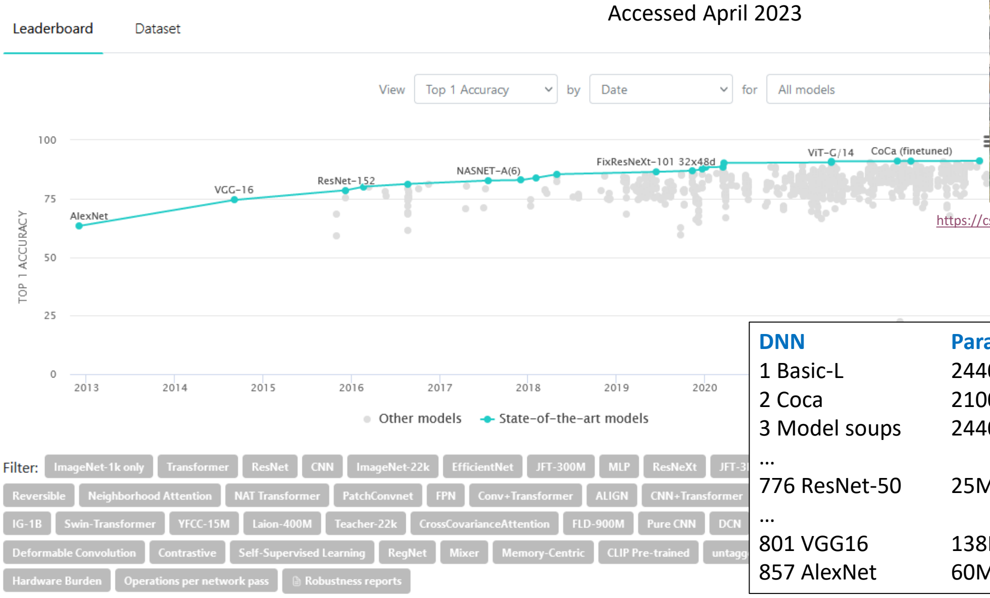
synset = set of one or more synonyms

https://cs.stanford.edu/people/karpathy/cnnembed/

Image Classification on ImageNet

(https://www.image-net.org/)

Accessed April 2023



https://cs.stanford.edu/people/karpathy/cnnembed/

DNN	Param	Top-1 Accuracy
1 Basic-L	2440M	91.1%
2 Coca	2100M	91%
3 Model soups	2440M	90,98%
…		
776 ResNet-50	25M	75.3% (2016)
…		
801 VGG16	138M	74.4% (2014)
857 AlexNet	60M	63.3% (2012)

https://paperswithcode.com/paper/imagenet-classification-with-deep

Object Tracking

- Conference on Computer Vision and Pattern Recognition (CVPR)

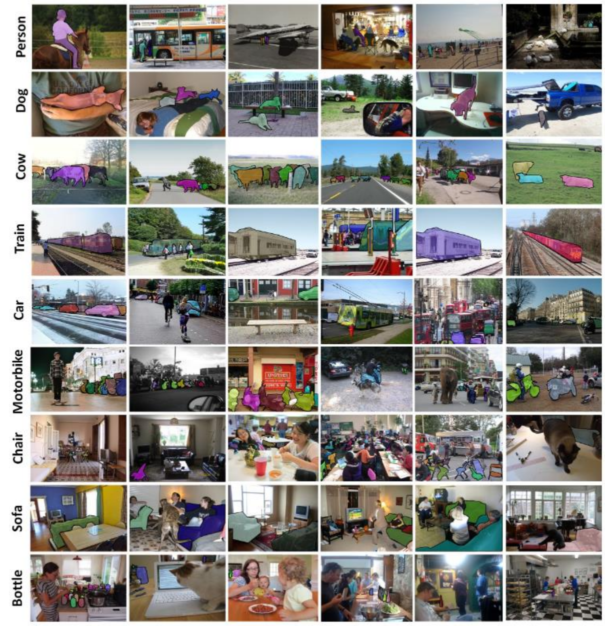
Real-Time Tracking

- A. He et al. A Twofold Siamese Network for Real-Time Object Tracking
- B. Yang et al. PIXOR: Real-Time 3D Object Detection From Point Clouds
- ….
- MEMOT: Multi Object Tracking with Memory CVPR2022
- Etc.

COCO:
Common Objects in Context
https://cocodataset.org

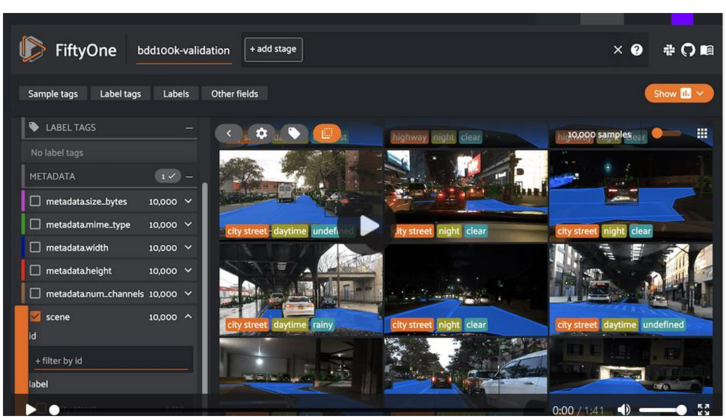
COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

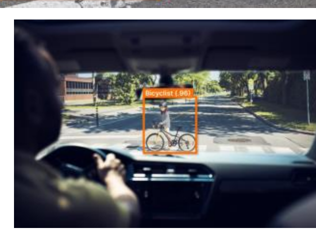
- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with keypoints

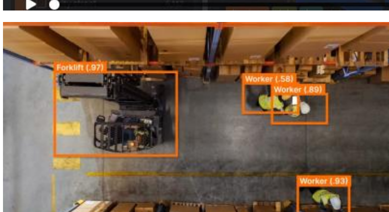


T.-Y. Lin et al. Microsoft COCO: Common Objects in Context., Computer Vision and Pattern Recognition, 2015.

FiftyOne: https://voxel51.com

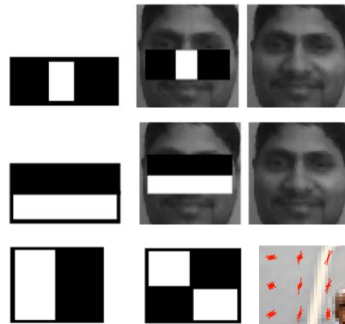


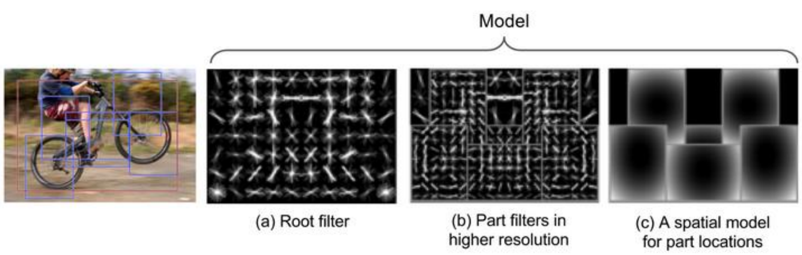




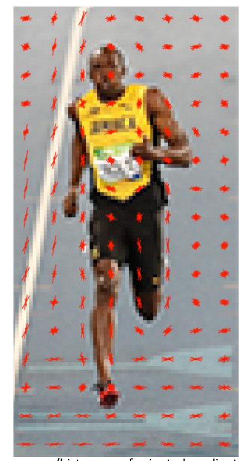
Object Detection

- Viola, Jones, Robust Real-time Object Detection, IJCV 2001.
- Histogram of Oriented Gradients (HOG) Detector, ECCV 2006
- Deformable Parts Model (Felzenswalb et al. 2010)





Deformable Parts Model (DPB), using Markov Random Fields

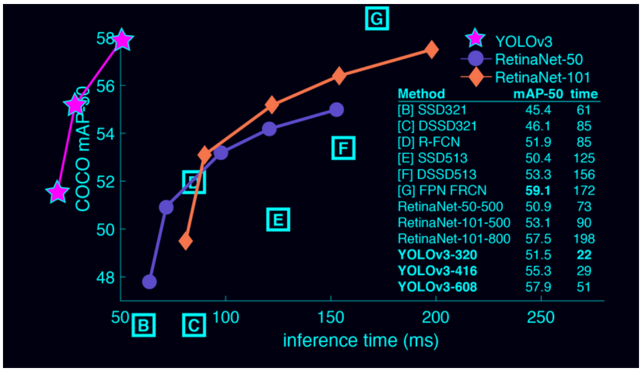


https://learnopencv.com/histogram-of-oriented-gradients/

Object Detection

- COCO Data Set
  - https://cocodataset.org/#explore
  - https://cocodataset.org/#detection-leaderboard
- MMDetection
  - https://github.com/open-mmlab/mmdetection
  - https://platform.openmmlab.com/web-demo/demo/detection
- YOLO v1 – v3
  - https://pjreddie.com/darknet/yolo/
  - Joseph Redmon, Ali Farhadi, YOLOv3: An Incremental Improvement, Tech Report, 2018 (See: https://pjreddie.com/publications/)
- Yolo v5
  - https://pytorch.org/hub/ultralytics_yolov5/

Object Detection: Yolo v1 – v3, …, Yolo v5



https://pjreddie.com/darknet/yolo/

better

YOLO: You Only Look Once

Performance on the COCO Dataset

Model	Train	Test	mAP	FLOPS	FPS	Cfg	Weights
SSD300	COCO trainval	test-dev	41.2	-	46		link
SSD500	COCO trainval	test-dev	46.5	-	19		link
YOLOv2 608x608	COCO trainval	test-dev	48.1	62.94 Bn	40	cfg	weights
Tiny YOLO	COCO trainval	test-dev	23.7	5.41 Bn	244	cfg	weights
SSD321	COCO trainval	test-dev	45.4	-	16		link
DSSD321	COCO trainval	test-dev	46.1	-	12		link
R-FCN	COCO trainval	test-dev	51.9	-	12		link
SSD513	COCO trainval	test-dev	50.4	-	8		link
DSSD513	COCO trainval	test-dev	53.3	-	6		link
FPN FRCN	COCO trainval	test-dev	59.1	-	6		link
Retinanet-50-500	COCO trainval	test-dev	50.9	-	14		link
Retinanet-101-500	COCO trainval	test-dev	53.1	-	11		link
Retinanet-101-800	COCO trainval	test-dev	57.5	-	5		link
YOLOv3-320	COCO trainval	test-dev	51.5	38.97 Bn	45	cfg	weights
YOLOv3-416	COCO trainval	test-dev	55.3	65.86 Bn	35	cfg	weights
YOLOv3-608	COCO trainval	test-dev	57.9	140.69 Bn	20	cfg	weights
YOLOv3-tiny	COCO trainval	test-dev	33.1	5.56 Bn	220	cfg	weights
YOLOv3-spp	COCO trainval	test-dev	60.6	141.45 Bn	20	cfg	weights

https://pytorch.org/hub/ultralytics_yolov5/

Yolo v5x6 mAP 54.4 22.4 ms on V100 GPU, 141.8 Mparams, 222.9 FLOPS

C.W. Corsel, YOLO-based Obstacle Avoidance for Drones. BSc Thesis, 2020.

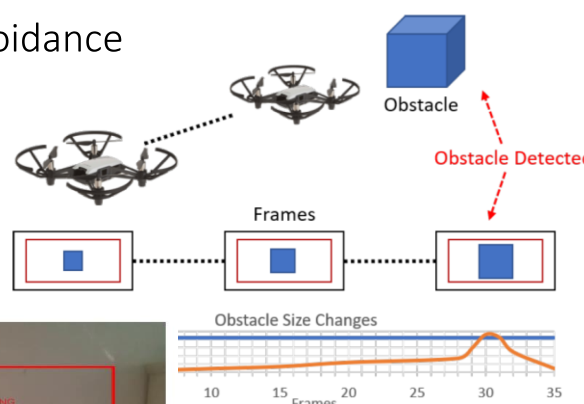


Figure 3.1: Size expansion concept

(a) SIFT (b) YOLO

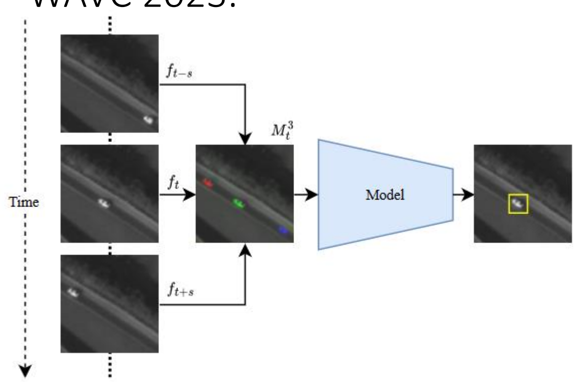
Figure 6.6: Object detection on multiple obstacles

C.W. Corsel et al. Exploiting Temporal Context for Tiny Object Detection, WAVC 2023.

WPAFB

Datasets: TwinCam, VIRAT and selected area of interests from the WPAFB Dataset.

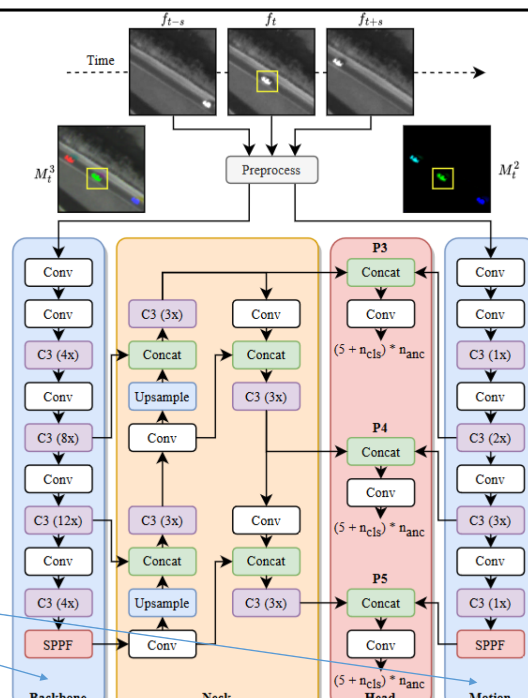
C.W. Corsel et al. Exploiting Temporal Context for Tiny Object Detection, WAVC 2023.



Three video frames are combined into a 3-channel image. A deep learning **object detector** detects objects by exploiting the temporal context

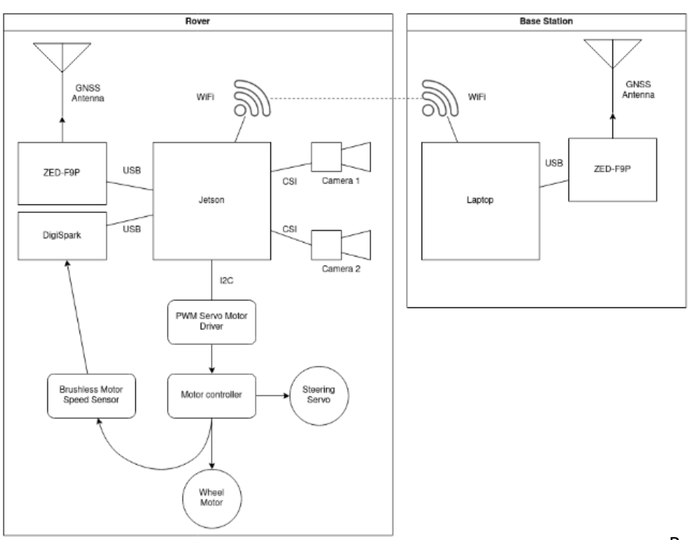
Yolov5s

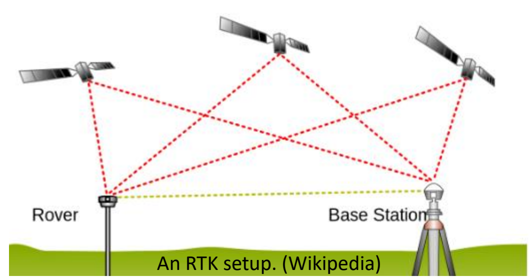
Yolov5 Backbones



Ouderijn t-yolov5x results

M. Delzenne, Autonomous navigation in pedestrian spaces. MSc Thesis 2023.

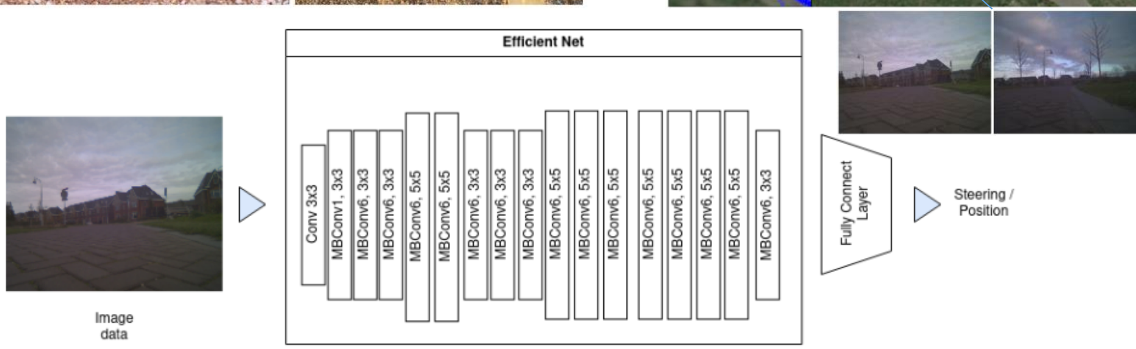




An RTK setup. (Wikipedia)

Real Time Kinematic Global Navigation Satellite System (RTK-GNSS)

M. Delzenne, Autonomous navigation in pedestrian spaces. MSc Thesis 2023.



W. Stokman, Obstacle detection and avoidance using image processing on embedded systems. BSc Thesis, 2020.

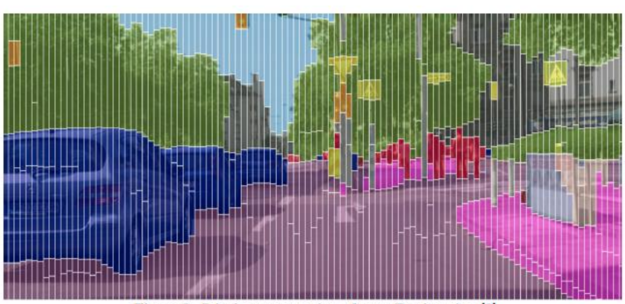


Figure 3: Stixel representation of a traffic situation [2]

Figure 2: Workflow of optimization using tensorflow in combination with TensorRT [17]

Figure 15: The Jetson Nano test setup

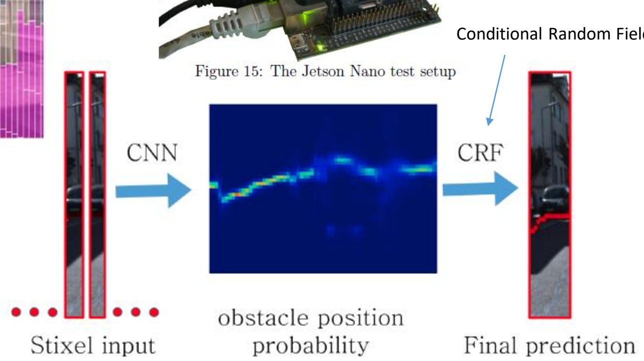


Figure 6: Sample output in a real world application

A. Tonioni et al. Real-time self-adaptive deep stereo. CVPR2019

https://github.com/CVLAB-Unibo/Real-time-self-adaptive-deep-stereo

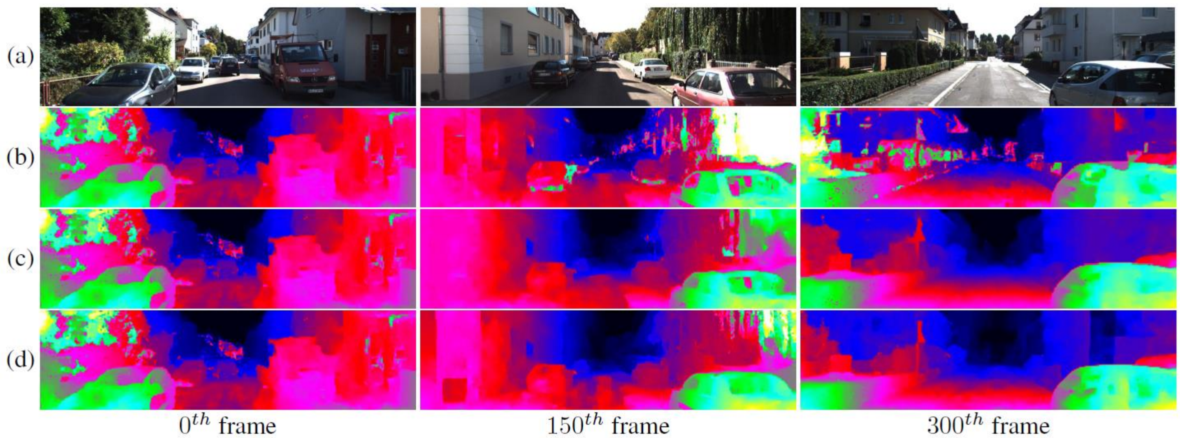
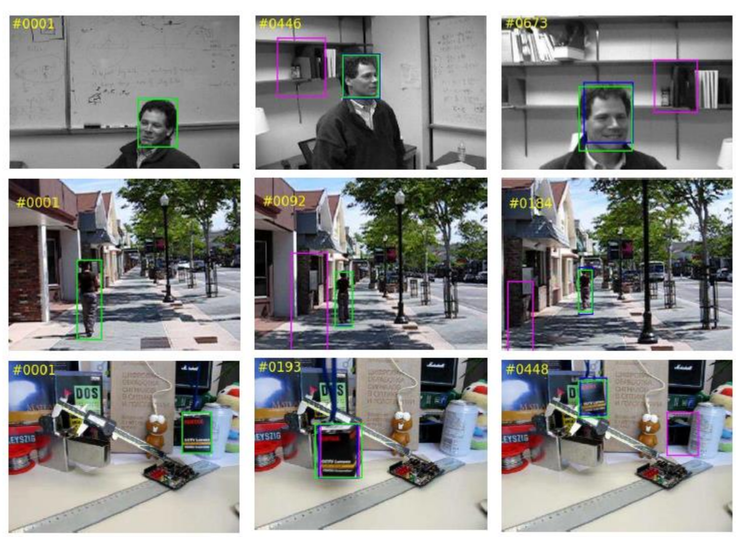


Figure 1. Disparity maps predicted by *MADNet* on a KITTI sequence [7]. Left images (a), no adaptation (b), online adaptation of the *whole* network (c), online adaptation by *MAD* (d). Green pixel values indicate larger disparities (*i.e.*, closer objects).

A. He et al. A Twofold Siamese Network for Real-Time Object Tracking, CVPR2018.

- Green is ground truth.
- Purple is tracked by *SiamFC*.
- Blue is tracked by the novel twofold Siamese network *2FSiamFC*.
- *2FSiamFC* is more robust to shooting angle change and scale change.

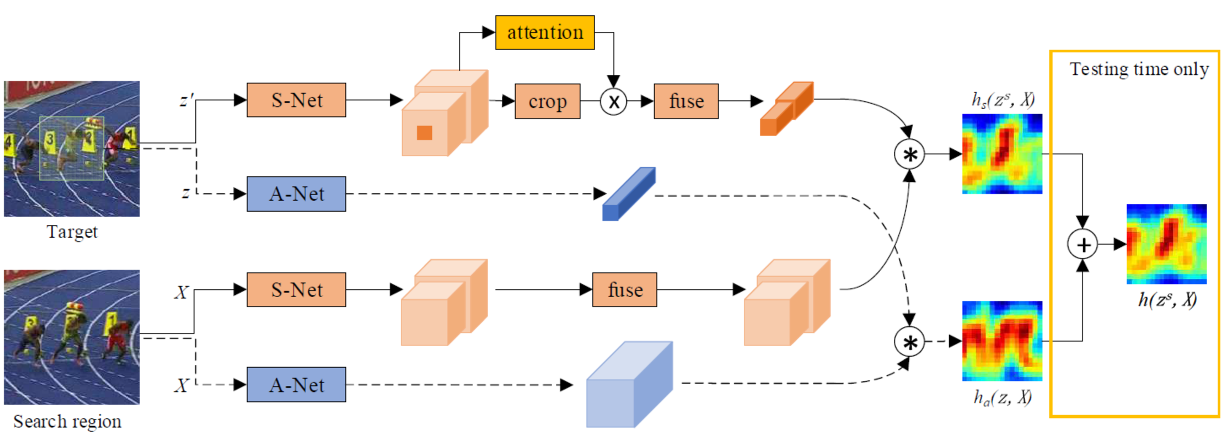


A. He et al. A Twofold Siamese Network for Real-Time Object Tracking, CVPR2018.

Object Tracking is a similarity learning problem

- Compare target image patch with candidate patches in a search region
- Track object to the location whit highest similarity score

- Similarity learning with deep CNNs use so called Siamese architectures (SiamFC).
- CNNs can process a larger search image where all sub-windows are evaluated as similarity candidates. (Efficient.)

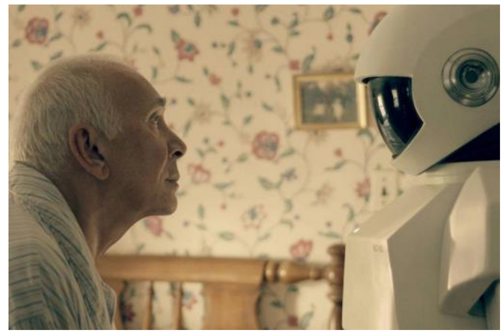


- A-Net is an appearance network, and S-Net is a semantic Network. (Branches trained separately.)
- The dotted lines is a SiamFC (Fully Convolutional Siamese Network Bertinetto et al. 2016.)
- The channel attention module determines the weight for each feature channel based on both target and context information.

(See also: J. Schonenberg, Differential Siamese Network for the Avoidance of Moving Obstacles. BSc, 2020.)

Human Robot Interaction

- Face Recognition
- Pose Recognition
- Hand Tracking
- Person Tracking
- Emotion Recognition
- Action Recognition



Face Recognition

- Yancheng Bai, et al., Finding Tiny Faces in the Wild With Generative Adversarial Network, CVPR, 2018.
- Xuanyi Dong, et al., Aggregated Network for Facial Landmark Detection, CVPR, 2018.
- Yaojie Liu, et al., Learning Deep Models for Face Anti-Spoofing: Binary or Auxiliary Supervision, CVPR, 2018.

- CVPR2018 58 papers on Face Recognition
- CVPR2019 and CVPR2020 similar numbers
- CVPR2021 ~50 papers related to Face Recognition
- CVPR2022 ~110 papers related to Face Recognition

https://openaccess.thecvf.com/CVPR2021

Yancheng Bai, et al., Finding Tiny Faces in the Wild With Generative Adversarial Network, CVPR2018.

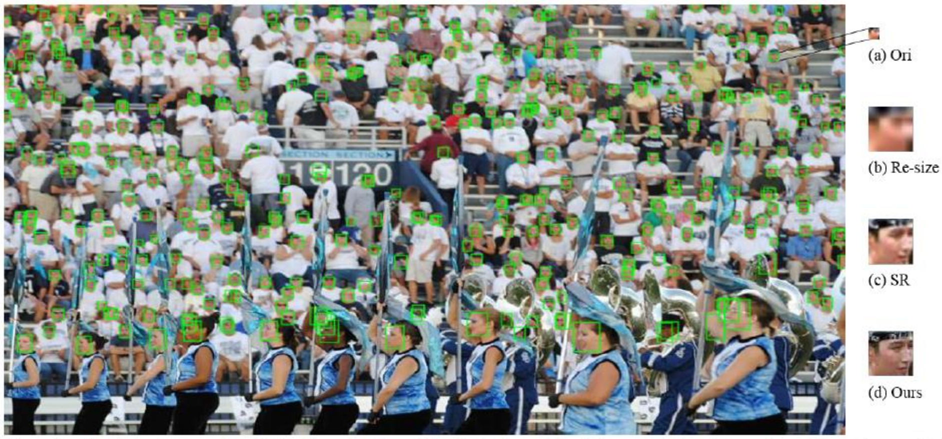
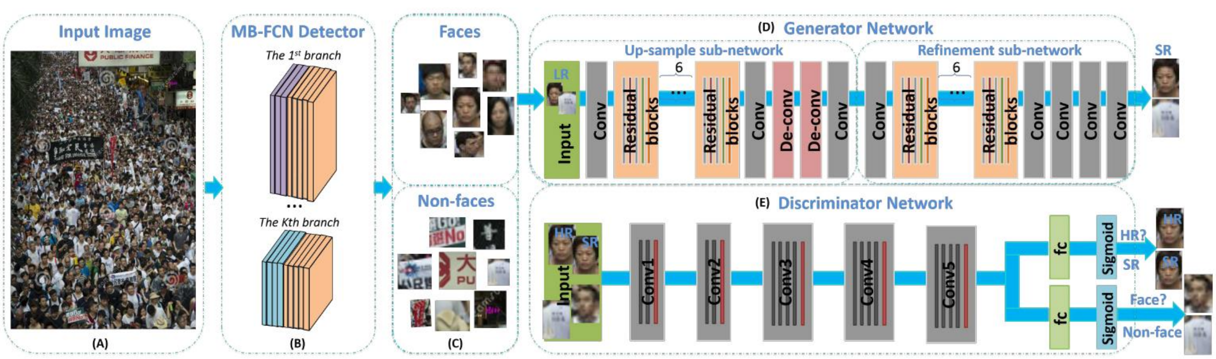
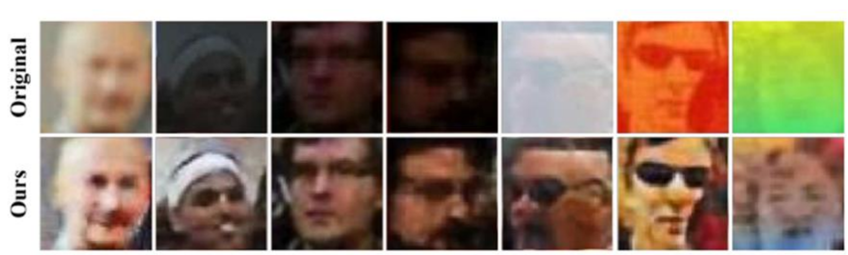


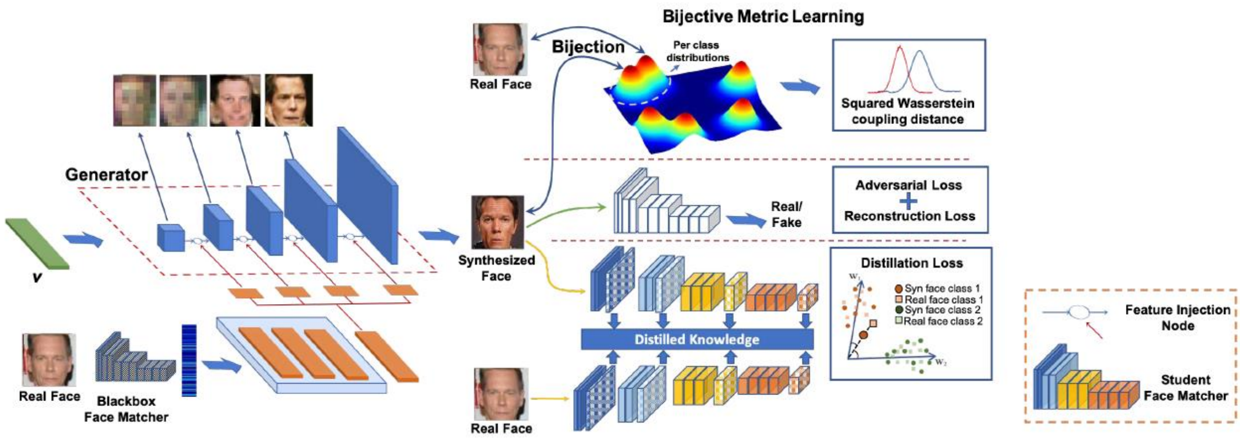
Figure1. The detection results of tiny faces in the wild. (a) is the original low-resolution blurry face, (b) is the result of re-sizing directly by a bi-linear kernel, (c) is the generated image by the super-resolution method, and our result (d) is learned by the super-resolution (×4 upscaling) and refinement network simultaneously. Best viewed in color and zoomed in.

Generative Adversarial Network.





See also:
C.N. Duong at al. Vec2Face: Unveil Human Faces from their Blackbox Features in Face Recognition, CVPR 2020



Some Qualitative Results

Green ground truth, red selected by the network.

Some Qualitative Results

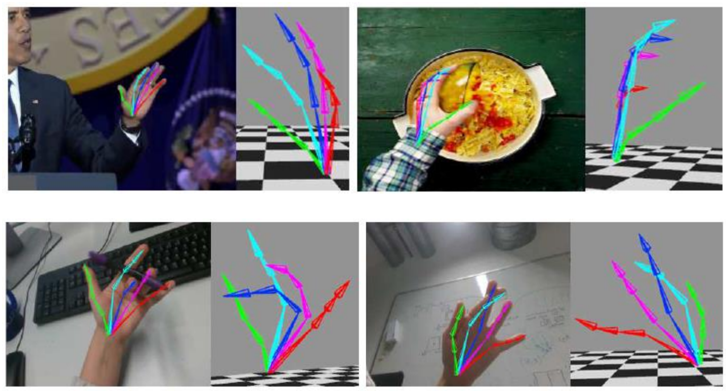
Green ground truth, red selected by the network.

Hand Pose Recogntion

F. Mueller, et al., **GANerated Hands for Real-Time 3D Hand Tracking From Monocular RGB**, CVPR2018.

G. Garcia-Hernando, et al., **First-Person Hand Action Benchmark With RGB-D Videos and 3D Hand Pose Annotations**, CVPR2018.

F. Mueller, et al., **GANerated Hands for Real-Time 3D Hand Tracking From Monocular RGB**, CVPR2018.

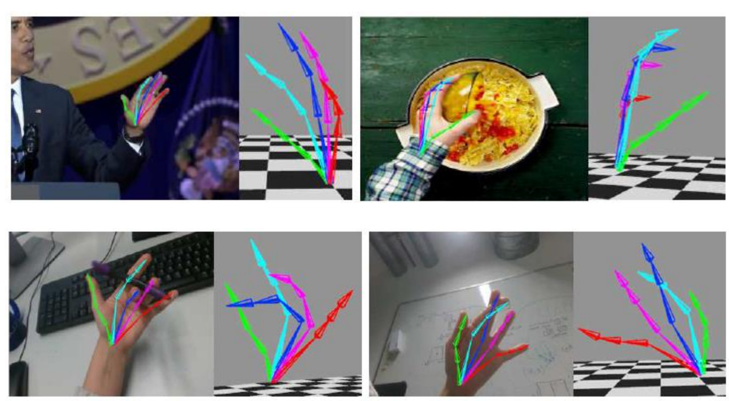


Input: RGB Image
Output: Hand Pose Skeleton.

F. Mueller, et al., **GANerated Hands for Real-Time 3D Hand Tracking From Monocular RGB**, CVPR2018.

Real-time 3D hand tracking from monocular RGB-only input.

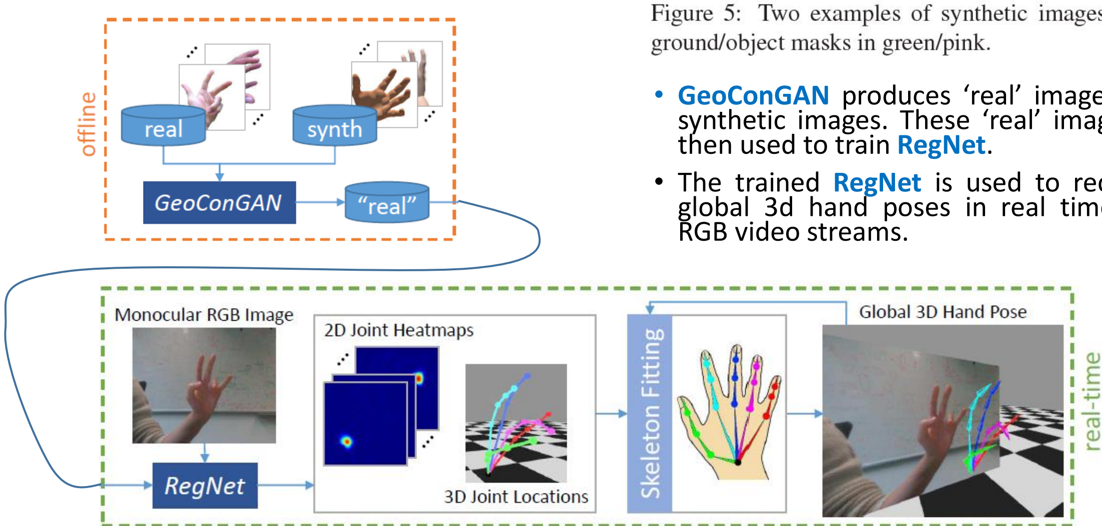
- Works on unconstrained videos from YouTube
- Is robust to occlusions.
- Real-time 3D hand tracking using an off-the-shelf RGB webcam in unconstrained setups.



F. Mueller, et al., **GANerated Hands for Real-Time 3D Hand Tracking From Monocular RGB**, CVPR2018.

Figure 5: Two examples of synthetic images with background/object masks in green/pink.

- **GeoConGAN** produces 'real' images from synthetic images. These 'real' images are then used to train **RegNet**.
- The trained **RegNet** is used to recognize global 3d hand poses in real time from RGB video streams.



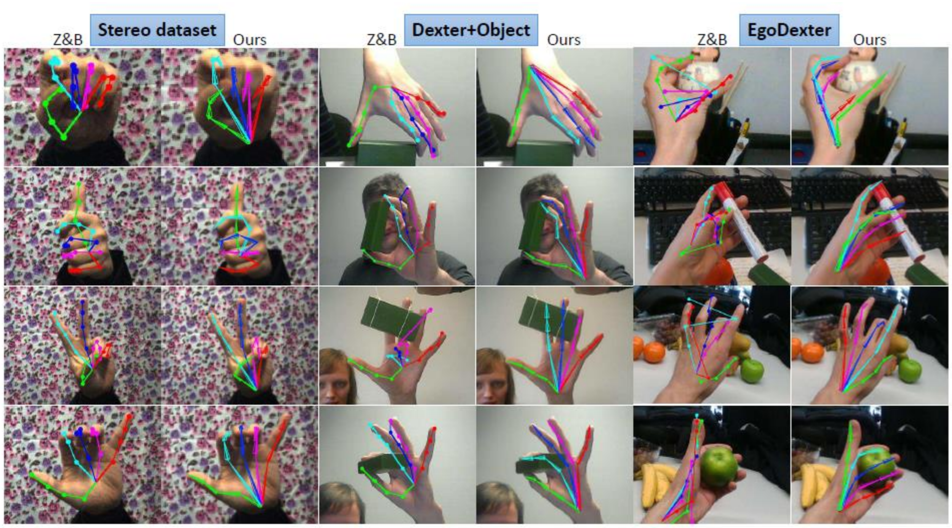
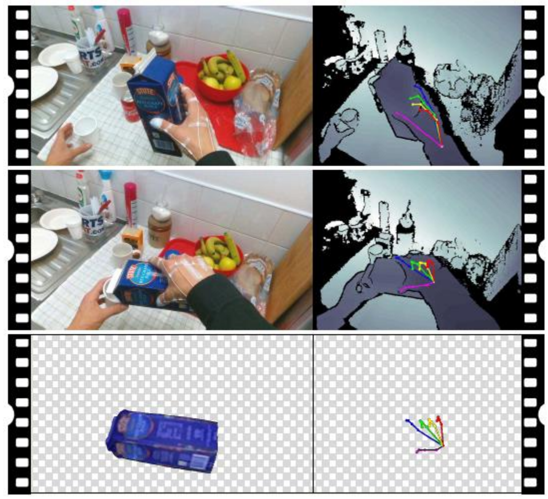


Figure 8: We compare our results with Zimmermann and Brox [63] on three different datasets. Our method is more robust in cluttered scenes and it even correctly retrieves the hand articulation when fingers are hidden behind objects.

Garcia-Hernando, et al., First-Person Hand Action Benchmark With RGB-D Videos and 3D Hand Pose Annotations, CVPR2018.

Pouring Juice

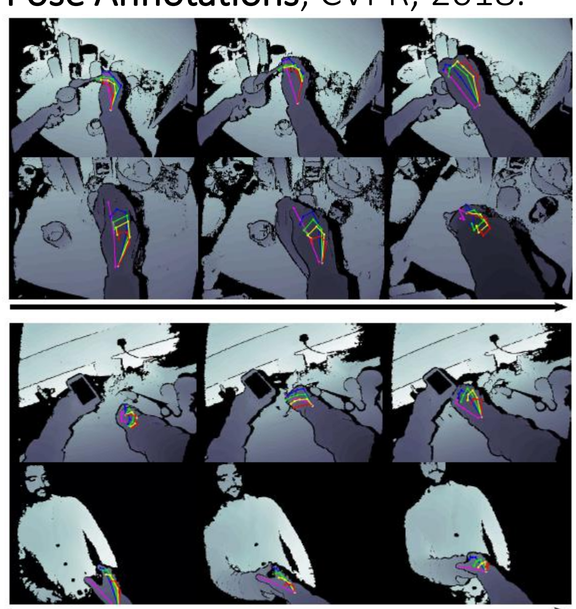
- A novel firstperson action recognition dataset with RGB-D videos and 3D hand pose annotations.
- Magnetic sensors and inverse kinematics to capture the hand pose.
- Also captured 6D object pose for some of the actions



Garcia-Hernando, et al., **First-Person Hand Action Benchmark With RGB-D Videos and 3D Hand Pose Annotations**, CVPR, 2018.

A novel first person action recognition dataset with RGB-D videos and 3D hand pose annotations.

- Put sugar.
- Pour milk.
- Charge cell-phone.
- Shake hand



Garcia-Hernando, et al., **First-Person Hand Action Benchmark With RGB-D Videos and 3D Hand Pose Annotations**, CVPR, 2018.

Visual data: Intel RealSense SR300 RGB-D camera on the shoulder of the subject (RGB 30 fps at 1920×1080 and Depth 640×480.)

Pose annotation:

hand pose

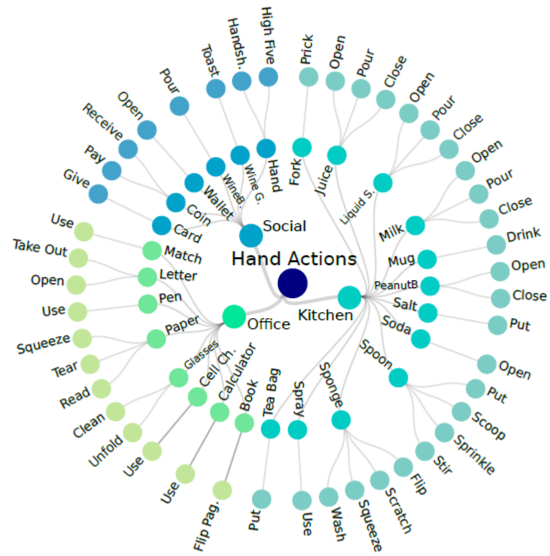
- captured using six magnetic sensors (6DOF) attached to the user's hand, five fingertips and one wrist, following [84].
- the hand pose is inferred using inverse kinematics over a defined 21-joint hand model

object pose

- 1 6DOF magnetic sensor attached to the closest point to the center of mass.

Recording process:

- 6 people, all right handed performed the actions.



Garcia-Hernando, et al., **First-Person Hand Action Benchmark With RGB-D Videos and 3D Hand Pose Annotations**, CVPR2018.

Baseline: RNN LSTM 100 neurons.

1:3 25% training 75% testing

1:1 50% - 50%

3:1 75% - 25%

Cross-person

Leave one of the 6 persons out of the training and test on the person left out.

Tensorflow and Adam optimizer.

Baseline Action recognition results

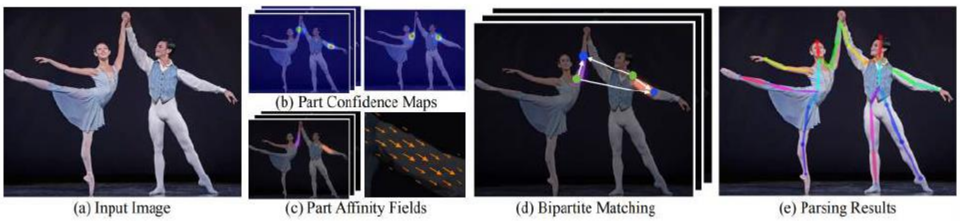
Protocol	1:3	1:1	3:1	cross-person
Acc. (%)	58.75	78.73	84.82	62.06

Hand pose recognition

Method	Year	Color	Depth	Pose	Acc. (%)
Two stream-color [15]	2016	✓	✗	✗	61.56
Two stream-flow [15]	2016	✓	✗	✗	69.91
Two stream-all [15]	2016	✓	✗	✗	75.30
HOG^2-depth [40]	2013	✗	✓	✗	59.83
HOG^2-depth+pose [40]	2013	✗	✓	✓	66.78
HON4D [43]	2013	✗	✓	✗	70.61
Novel View [47]	2016	✗	✓	✗	69.21
1-layer LSTM	2016	✗	✗	✓	78.73
2-layer LSTM	2016	✗	✗	✓	80.14
Moving Pose [85]	2013	✗	✗	✓	56.34
Lie Group [64]	2014	✗	✗	✓	82.69
HBRNN [12]	2015	✗	✗	✓	77.40
Gram Matrix [86]	2016	✗	✗	✓	85.39
TF [17]	2017	✗	✗	✓	80.69
JOULE-color [19]	2015	✓	✗	✗	66.78
JOULE-depth [19]	2015	✗	✓	✗	60.17
JOULE-pose [19]	2015	✗	✗	✓	74.60
JOULE-all [19]	2015	✓	✓	✓	78.78

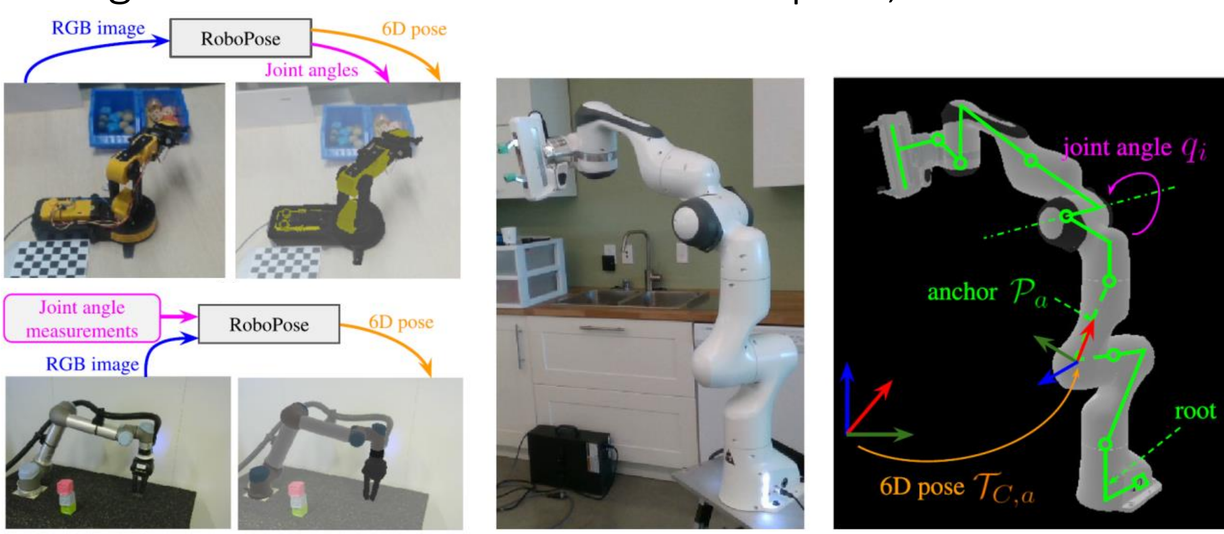
Table 4: Hand action recognition performance by different evaluated approaches on our proposed dataset.

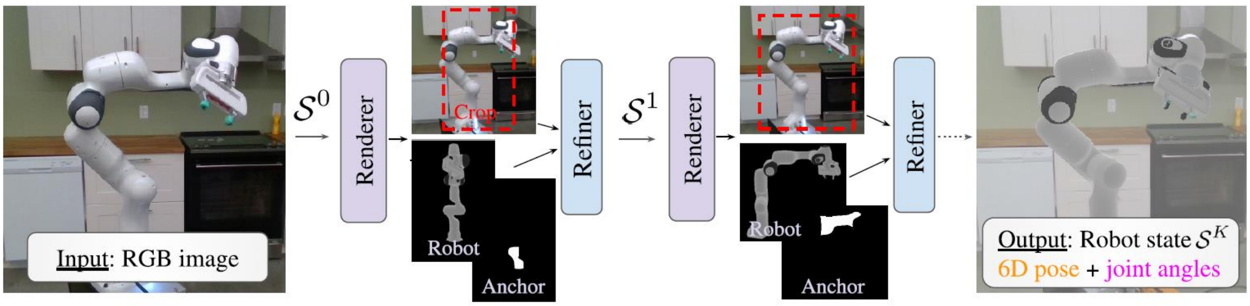
K. Maas, Full-Body Action Recognition from Monocular RGB-Video: A multi-stage approach using OpenPose and RNNs, BSc Thesis, 2020.



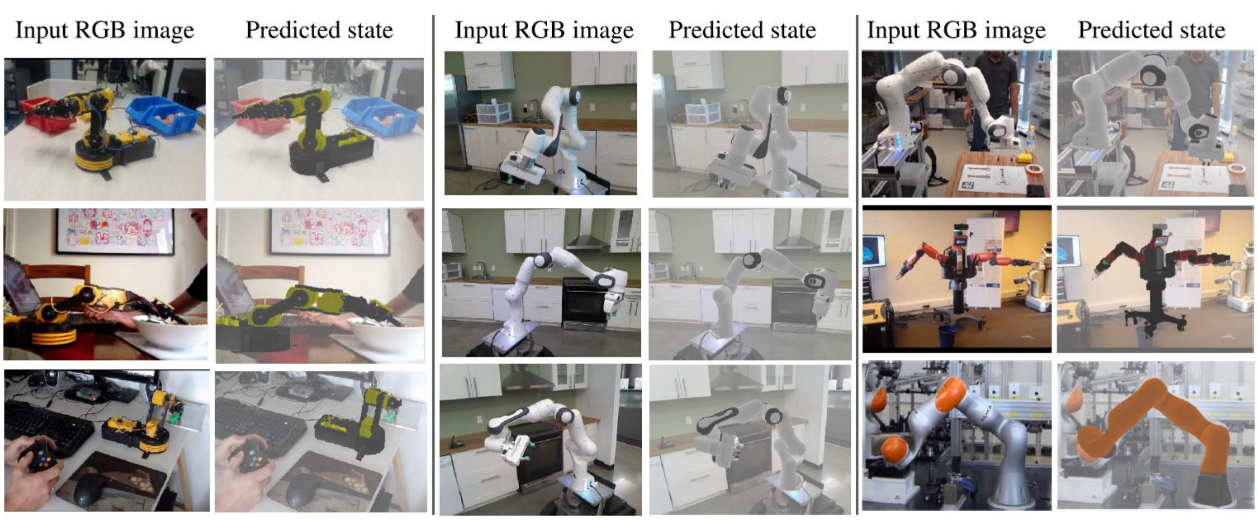
(a) *Start state* (b) *Raise Arm* (c) *Swipe*

Y. Labbe et al. Single-view robot pose and joint angle estimation via render & compare, CVPR2021





- Iteratively updating using a renderer and refiner until the rendered robot matches the input image.



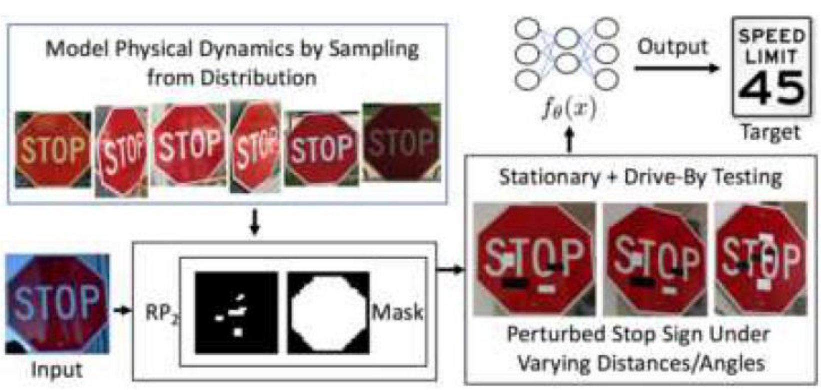
Some Problems with Deep Neural Networks

K. Eykholt, et al. Dawn Song Robust Physical-World Attacks on Deep Learning Visual Classification, CVPR2018.

K. Eykholt, et al. Dawn Song Robust Physical-World Attacks on Deep Learning Visual Classification, CVPR2018.

Robust Physical Perturbations (RP2):

- generate physical perturbations for physical-world objects such that a DNN-based classifier produces a designated misclassification.
- This under a range of dynamic physical conditions, including different viewpoint angles and distances.



K. Eykholt, et al. Dawn Song Robust Physical-World Attacks on Deep Learning Visual Classification, CVPR2018.

Two types of attacks showing that RP2 produces robust perturbations for real road signs.

- poster attacks are successful in 100% of stationary and drive-by tests against LISA-CNN
- sticker attacks are successful in 80% of stationary testing conditions

Target

K. Eykholt, et al. Dawn Song Robust Physical-World Attacks on Deep Learning Visual Classification, CVPR2018.

This is a micro-wave. This is not a micro-wave.

Yuxin Xiong, Adversarial Detection and Defense in Deep learning, 2021

Adversarial attacks on DNNs in e.g. autonomous driving and facial recognition.

- Adversarial examples constructed by shapeshifter
- robust to distortions at different distances and angles, etc.

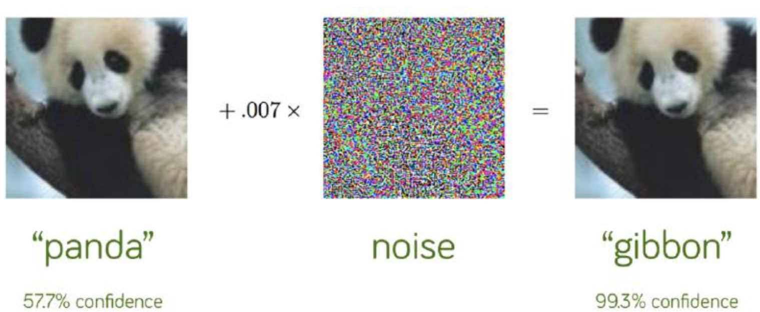
UNMASK[15] a framework to detect and defend against attacks:

- extract features by semantic segmentation technique.
- compare extracted features to detect if input image is benign
- counter against attacks by refining to the correct class.

Modified UNMASK model for Resnet101:

- add 4 feature denoising blocks: robust to various attacks
- improves UNMASK against several types of attacks

Adversarial Examples



Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. in ICLR, 2015.

Shapeshifter

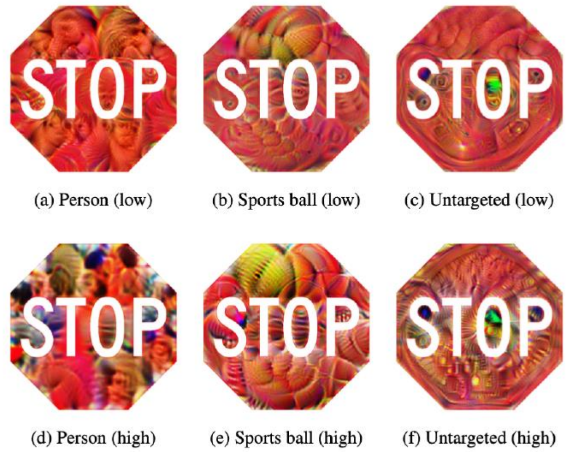


Figure 4: Adversarial examples generated by Shapeshifter with "low" and "high" confidence(perturbation strength). Shapeshifter can perform both targeted attacks and non-target attacks.

Framework D

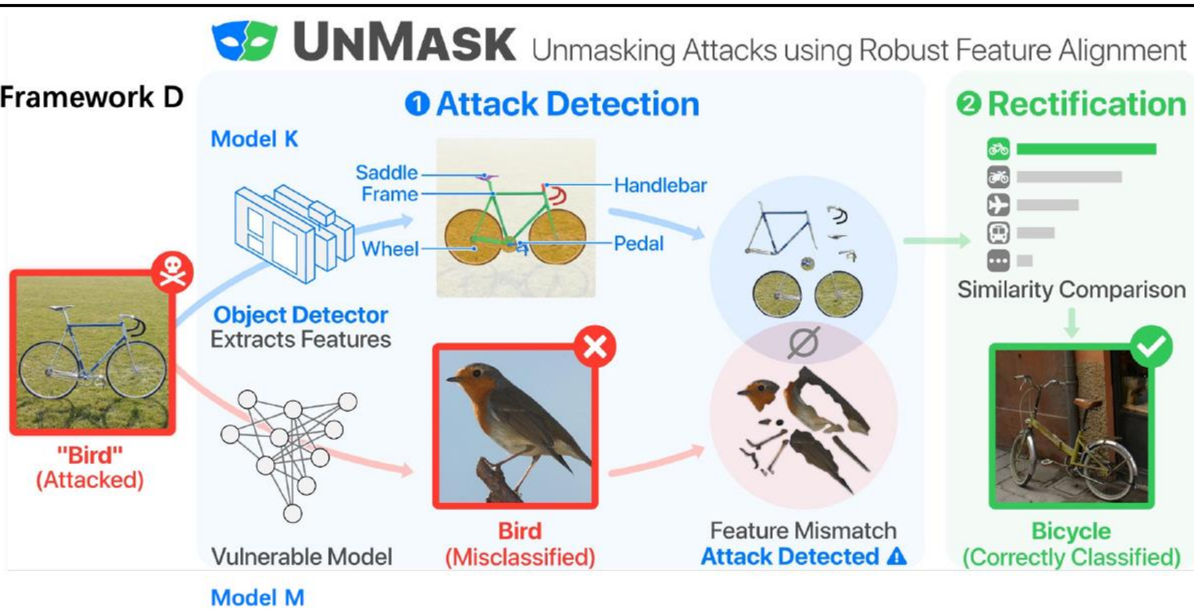
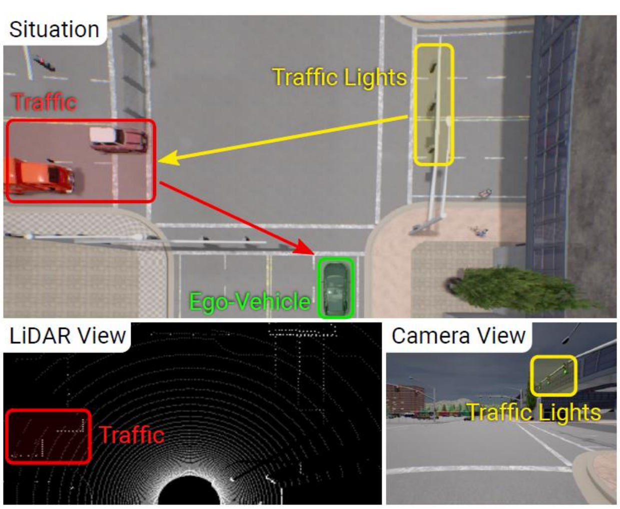
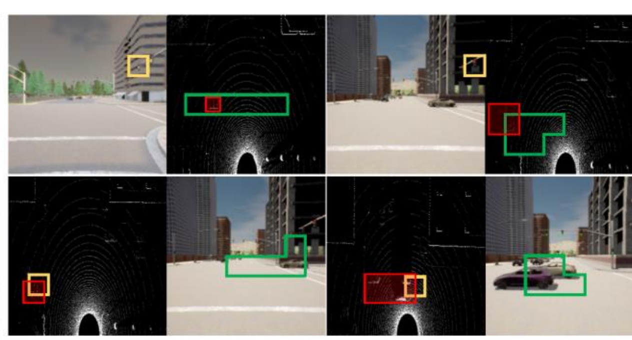


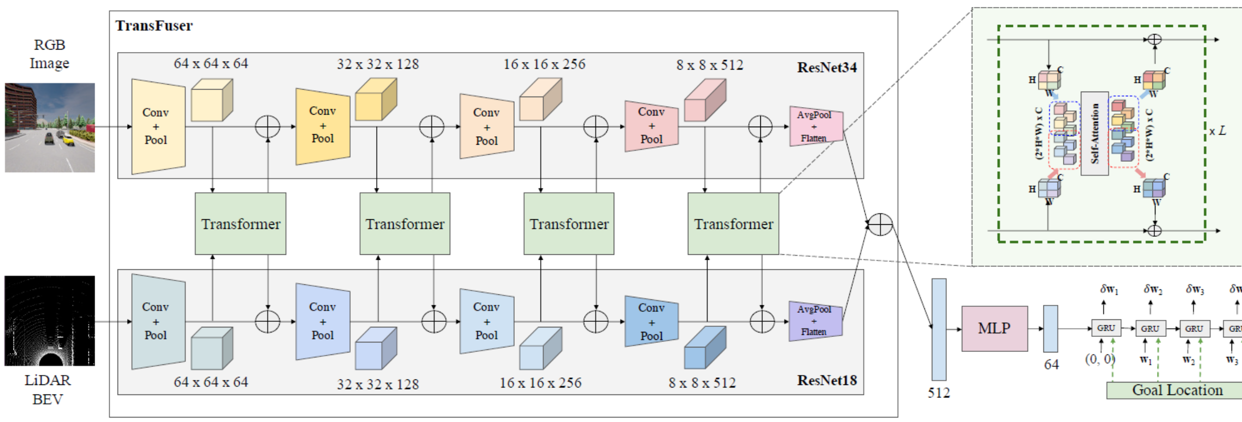
Figure 6: An overview of UNMASK framework. [3]

A. Prakash et al., Multi-Modal Fusion Transformer for End-to-End Autonomous Driving, CVPR2021





Attention Maps: yellow query token, red vehicle in lidar.
=> Green top-5 attended tokens.



The TransFuser uses attention to capture the global 3D scene context and focuses on dynamic agents and traffic lights, resulting in state-of-the-art performance on CARLA.

Method	Town05 Short		Town05 Long	
	DS ↑	RC ↑	DS ↑	RC ↑
CILRS [16]	7.47 ± 2.51	13.40 ± 1.09	3.68 ± 2.16	7.19 ± 2.95
LBC [8]	30.97 ± 4.17	55.01 ± 5.14	7.05 ± 2.13	32.09 ± 7.40
AIM	49.00 ± 6.83	81.07 ± 15.59	26.50 ± 4.82	60.66 ± 7.66
Late Fusion	51.56 ± 5.24	83.66 ± 11.04	31.30 ± 5.53	68.05 ± 5.39
Geometric Fusion	54.32 ± 4.85	**86.91** ± 10.85	25.30 ± 4.08	**69.17** ± 11.07
TransFuser (Ours)	**54.52** ± 4.29	78.41 ± 3.75	**33.15** ± 4.04	56.36 ± 7.14
Expert	*84.67 ± 6.21*	*98.59 ± 2.17*	*38.60 ± 4.00*	*77.47 ± 1.86*

Mean and stdev on Route Completion (RC) and Driving Score (DS) in 2 Town Settings with high densities of dynamic agents and scenario's over a total of 9 runs.

References

Papers can be obtained from http://openaccess.thecvf.com/CVPR2018.py

Real-Time Tracking

[1] A. He et al. A Twofold Siamese Network for Real-Time Object Tracking, CVPR, 2018.

[2] B. Yang et al. PIXOR: Real-Time 3D Object Detection From Point Clouds, CVPR, 2018.

[3] B. Tekin et al., Real-Time Seamless Single Shot 6D Object Pose Prediction, CVPR, 2018.

Face Recognition

[4] Yancheng Bai, et al., Finding Tiny Faces in the Wild With Generative Adversarial Network, CVPR, 2018.

[5] Xuanyi Dong, et al., Aggregated Network for Facial Landmark Detection, CVPR, 2018.

[6] Yaojie Liu, et al., Learning Deep Models for Face Anti-Spoofing: Binary or Auxiliary Supervision, CVPR, 2018.

Hand Pose Recognition

[7] F. Mueller, et al., GANerated Hands for Real-Time 3D Hand Tracking From Monocular RGB, CVPR, 2018.

[8] G. Garcia-Hernando, et al., First-Person Hand Action Benchmark With RGB-D Videos and 3D Hand Pose Annotations, CVPR, 2018.

Problems with Deep Learning Classification

[9] K. Eykholt, et al. Dawn Song Robust Physical-World Attacks on Deep Learning Visual Classification, CVPR, 2018.

References

For further papers see also:

Conference on Computer Vision and Pattern Recognition (CVPR)

- http://openaccess.thecvf.com/CVPR2018.py
- http://openaccess.thecvf.com/CVPR2019.py
- http://openaccess.thecvf.com/CVPR2020.py
- https://openaccess.thecvf.com/CVPR2021
- https://openaccess.thecvf.com/CVPR2022