University of Zurich[UZH]

ROBOTICS & PERCEPTION GROUP

Vision Algorithms for Mobile Robotics

Lecture 12b
Deep Learning Tutorial

Jiaxu Xing

https://rpg.ifi.uzh.ch

Outline

- Introduction
- Supervised Learning
- Unsupervised Learning
- Applications to Computer Vision
- Conclusions
- Machine Learning for Drones

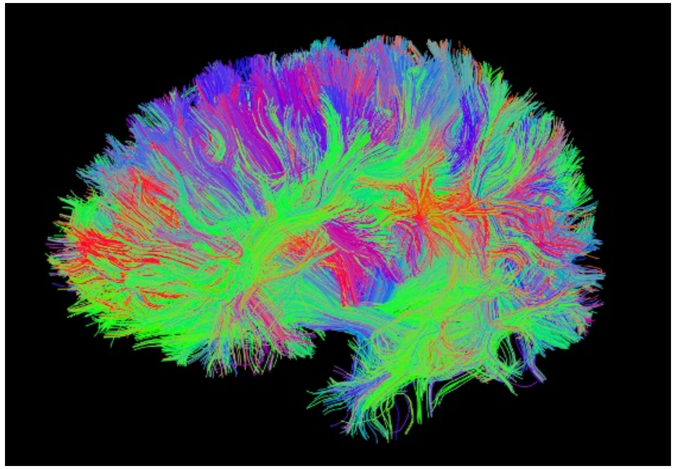
Relevant for the exam

Outline

- Introduction
- Supervised Learning
- Unsupervised Learning
- Applications to Computer Vision
- Conclusions
- Machine Learning for Drones

The Deep Learning Revolution

Medicine

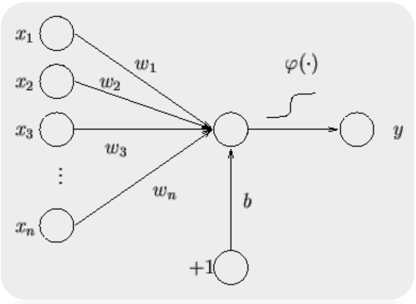


Media & Entertainment

Surveillance & Security

Autonomous Driving

Some History

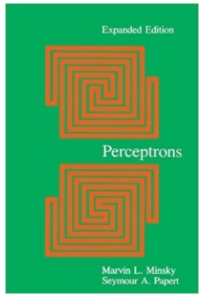


Perceptron

1958

1969

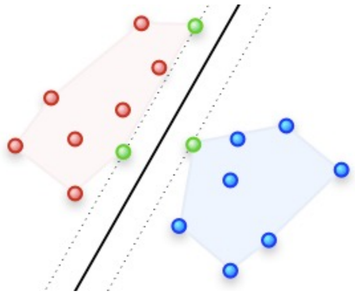
Critics



Back-Propagation

1974

AI Winter

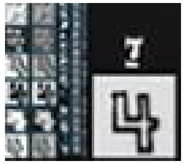


SVM

1995

1998

Convolutional Neural Networs for Handwritten Digits Recognition



2006

Restricted Boltzman Machines

AlexNet

2012

What changed?

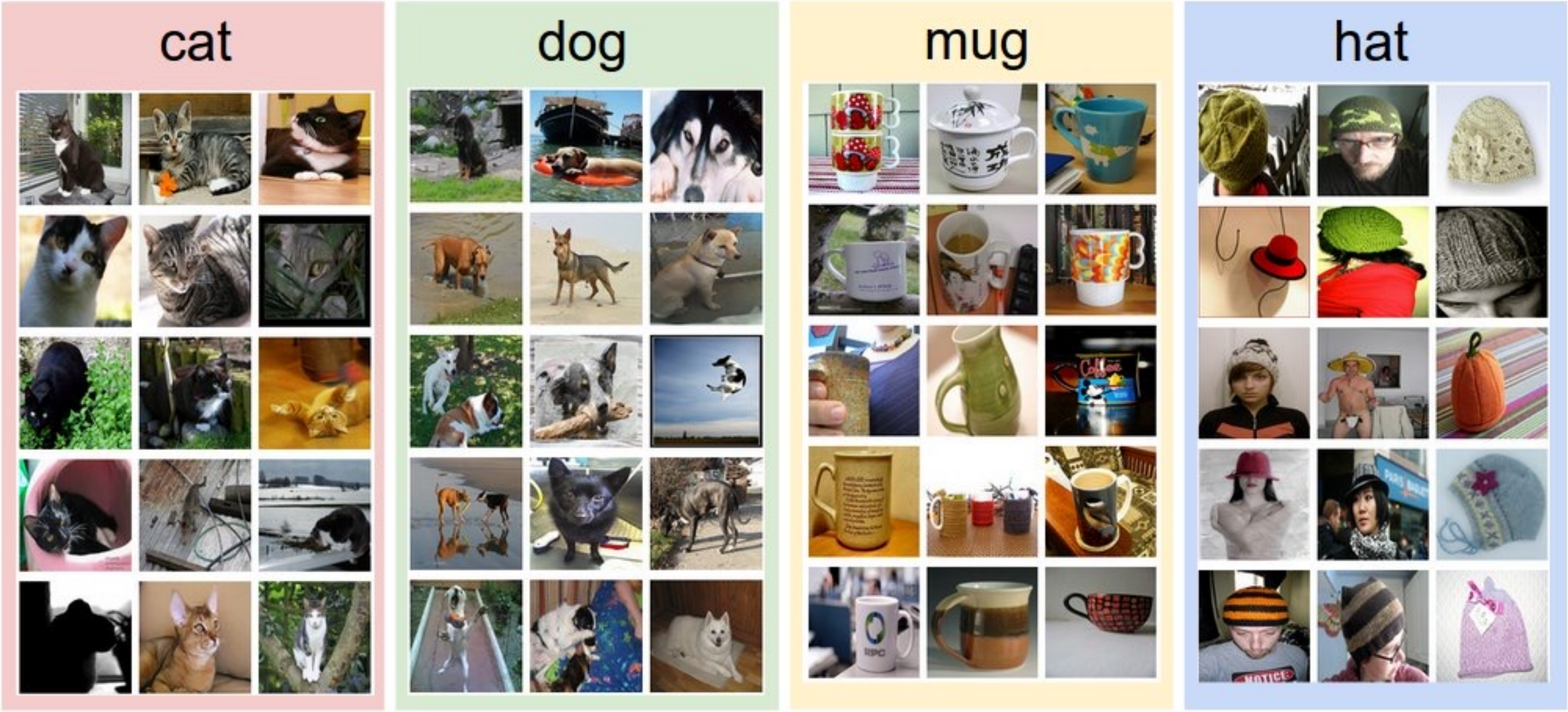
1. Hardware Improvements

2. Big Data Available

3. Algorithmic Progress

Image Classification

Task of assigning an input image a label from a fixed set of categories.



[1] Slide adapted from CNNs for Visual Recognition (Stanford) [Website]

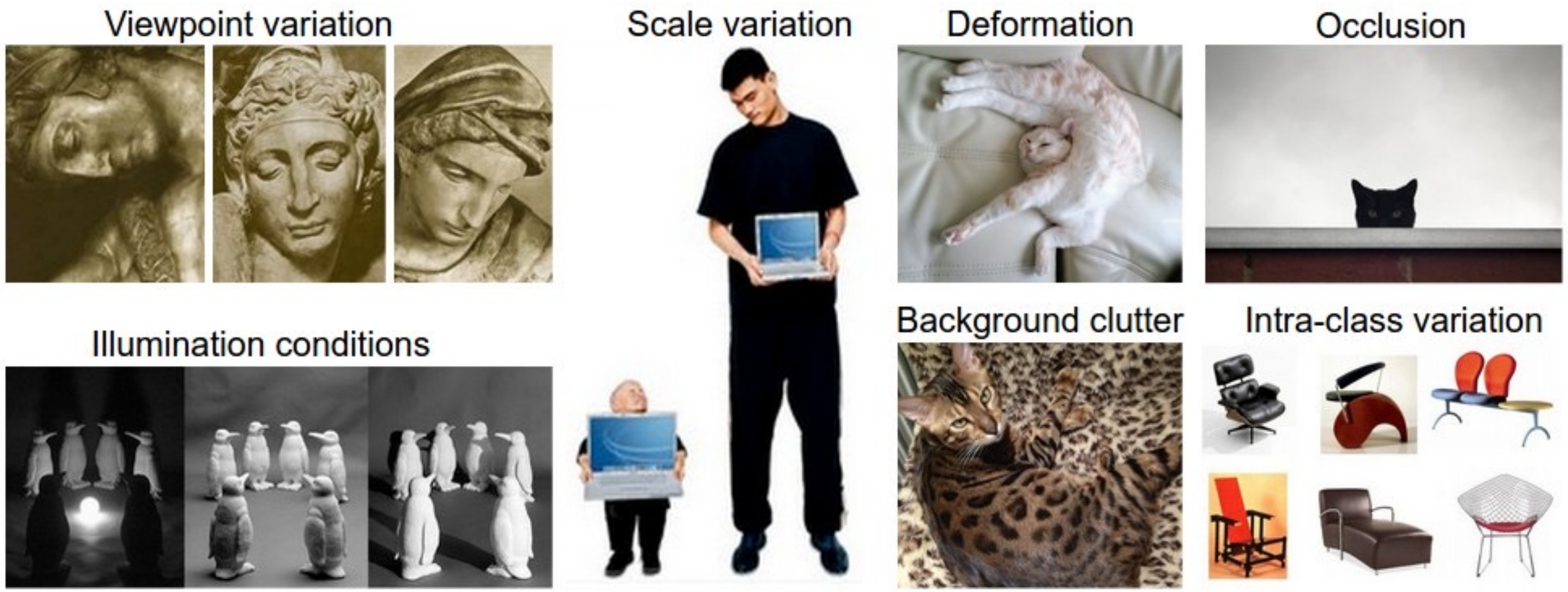
The semantic gap

What computers see compared to what we see

401	402	403	404	405	406	407	408	409	410
411	412	413	414	415	416	417	418	419	420
421	422	423	424	425	426	427	428	429	430
431	432	433	434	435	436	437	438	439	440
441	442	443	444	445	446	447	448	449	450
451	452	453	454	455	456	457	458	459	460
461	462	463	464	465	466	467	468	469	470
471	472	473	474	475	476	477	478	479	480
481	482	483	484	485	486	487	488	489	490
491	492	493	494	495	496	497	498	499	500

Classification Challenges

Directly specifying how a category looks like is impossible.



We need use a **Data Driven Approach**

Supervised Learning

Find function $f(x, \theta)$ that imitates a ground truth signal

$f(x, \theta)$

N numbers representing class scores

Predicted

$$\begin{pmatrix} 0.1 \\ 0.7 \\ \cdots \\ 0.0 \end{pmatrix}$$

Ground truth, y_i

$$\begin{pmatrix} 1.0 \\ 0.0 \\ \cdots \\ 0.0 \end{pmatrix}$$

Function parameters or weights

Update

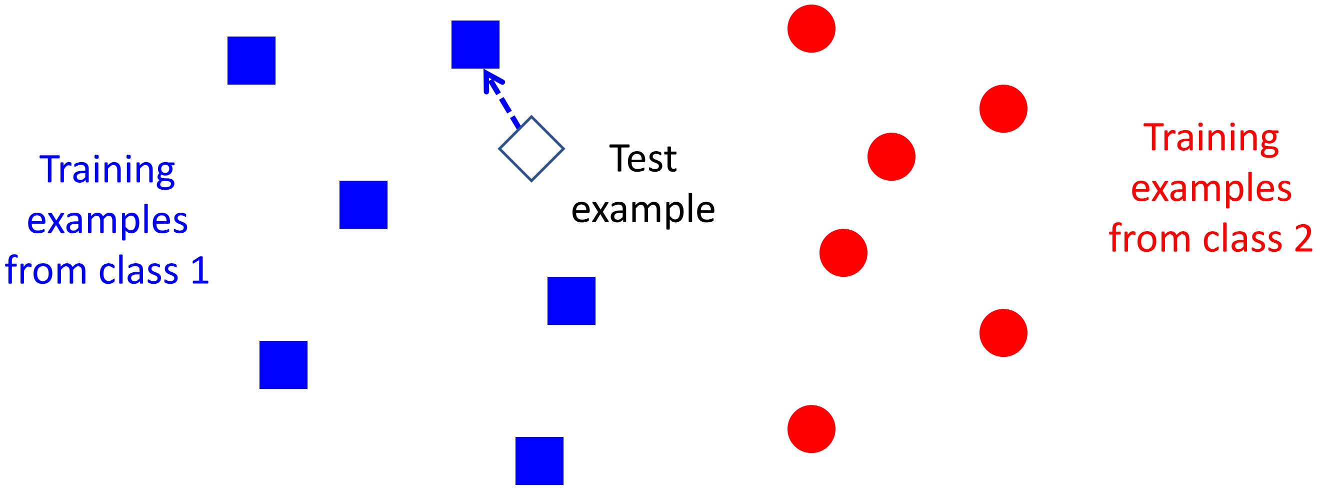
$$Loss(f(x_i, \theta), y_i)$$

Machine Learning Keywords

1. Loss: Quantify how good θ are
2. Optimization: The process of finding θ that minimize the loss
3. Function: Problem modelling $\rightarrow$ Deep networks are highly non-linear $f(x, \theta)$

Classifiers: K-Nearest neighbor

Features are represented in the descriptor space



$f(\boldsymbol{x}, \theta)$ = label of the K training examples nearest to $\boldsymbol{x}$

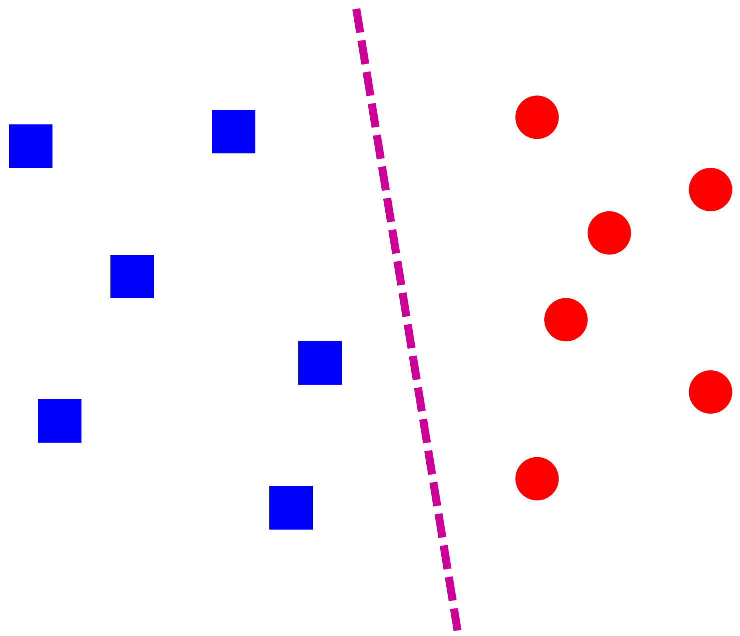
How fast is training? How fast is testing?

- O(1), O(n)

What is a good distance metric ? What K should be used? ☹

Classifiers: Linear

Find a *linear function* to separate the classes:



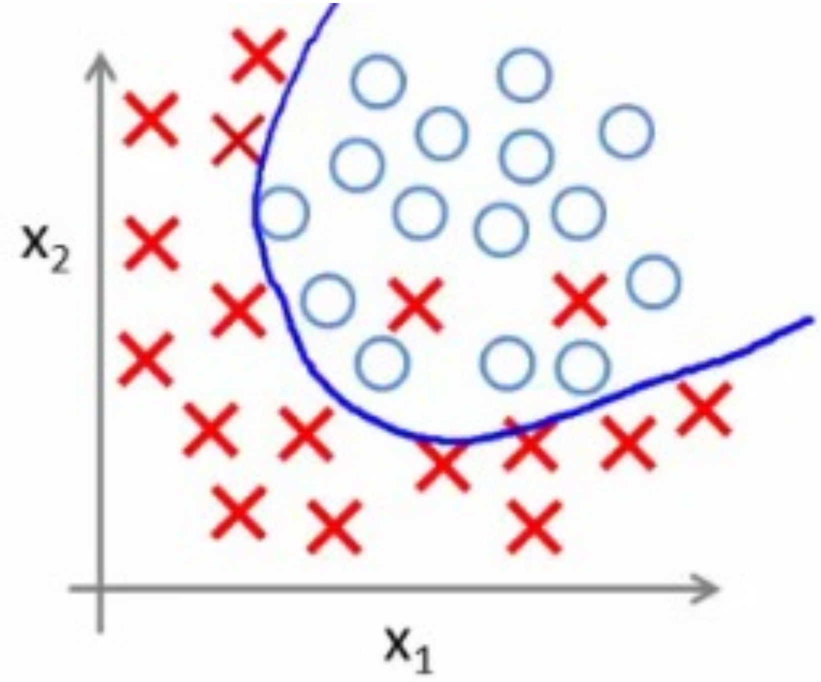
$$f(\boldsymbol{x}, \theta) = sgn(\theta \cdot \boldsymbol{x} + b)$$

What is θ? What is the dimensionality of images?

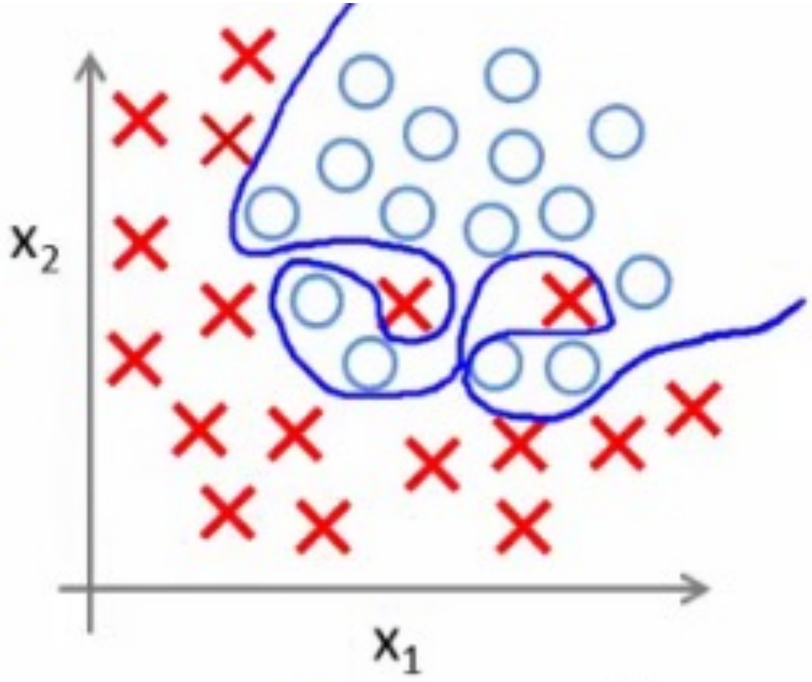
Classifiers: non-linear

What is $f(x, \theta)$?

Good classifier

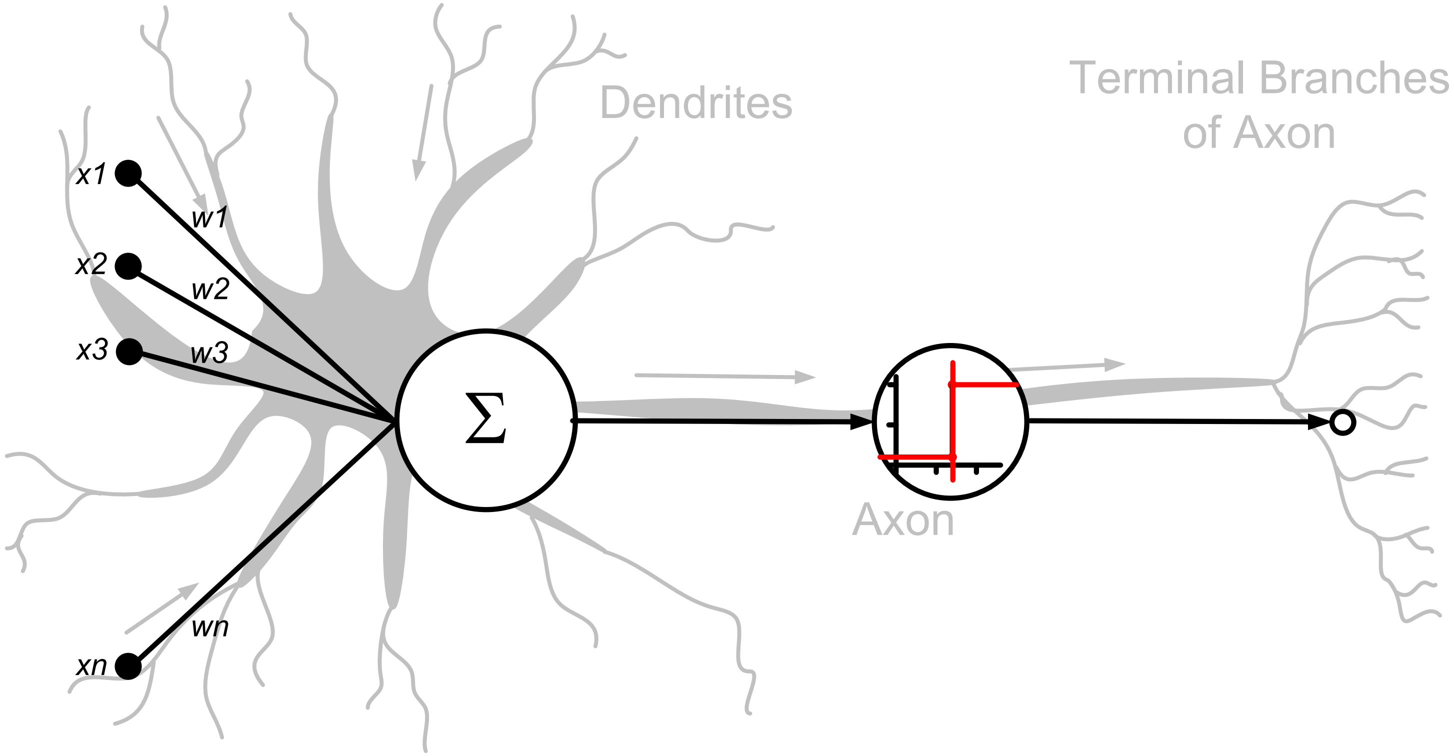


Bad classifier (over fitting)



Biological Inspiration

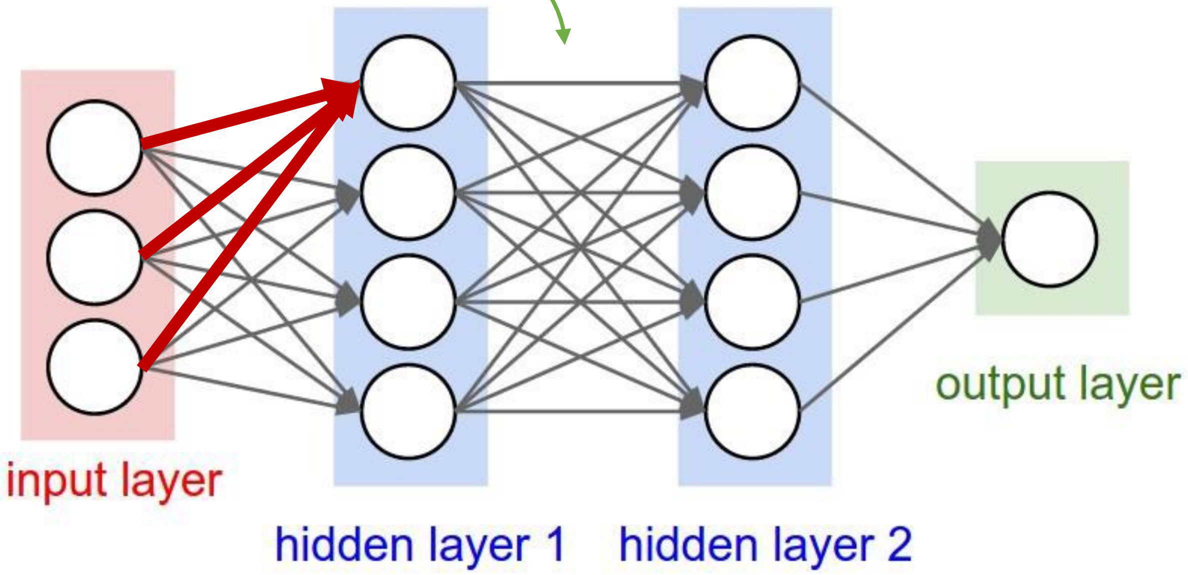
$f(x, \theta) = F(\theta x)$, F is a non-linear activation function (Step, ReLU, Sigmoid)



[1] Rosenblatt, *The perceptron: a probabilistic model for information storage and organization in the brain*, Psychological Review, 1958. PDF

Multi Layer Perceptron

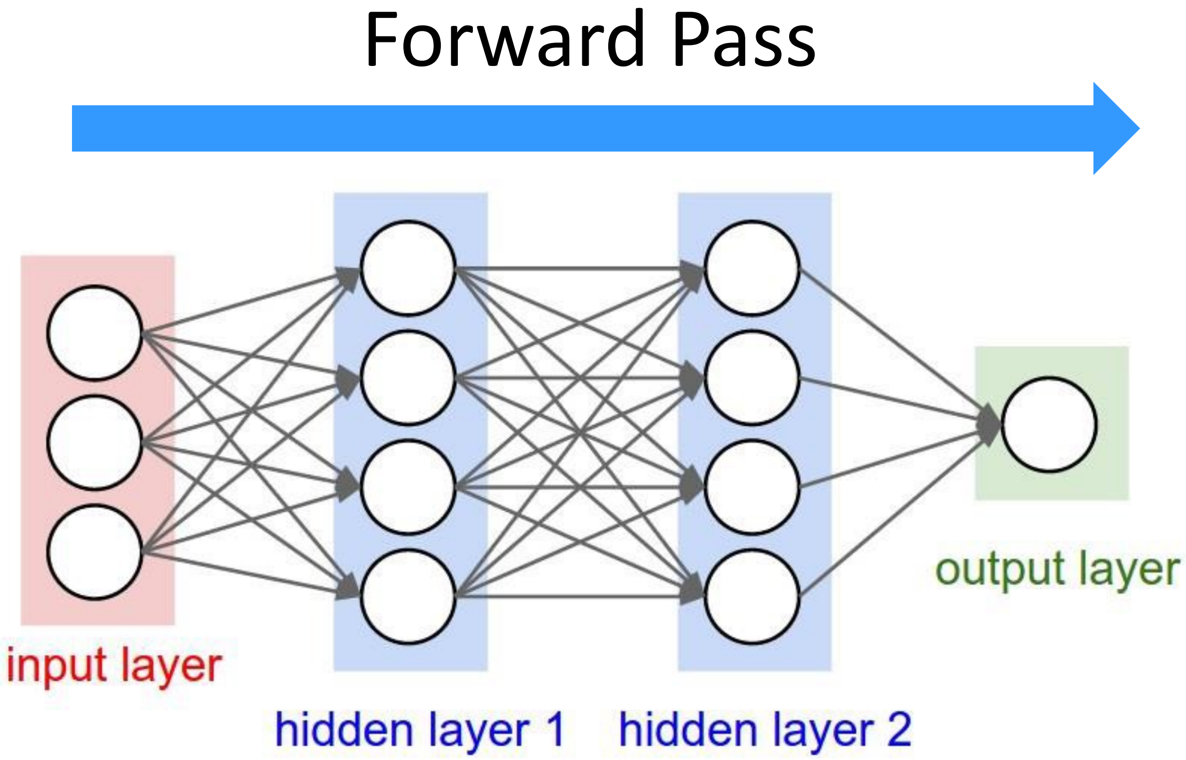
$f(x, \theta)$



Non-linear Activation functions (ReLU, sigmoid, etc.)

[1] Michael Nielsen, *Neural Networks and Deep Learning, Chapter 2* PDF

Forward Propagation



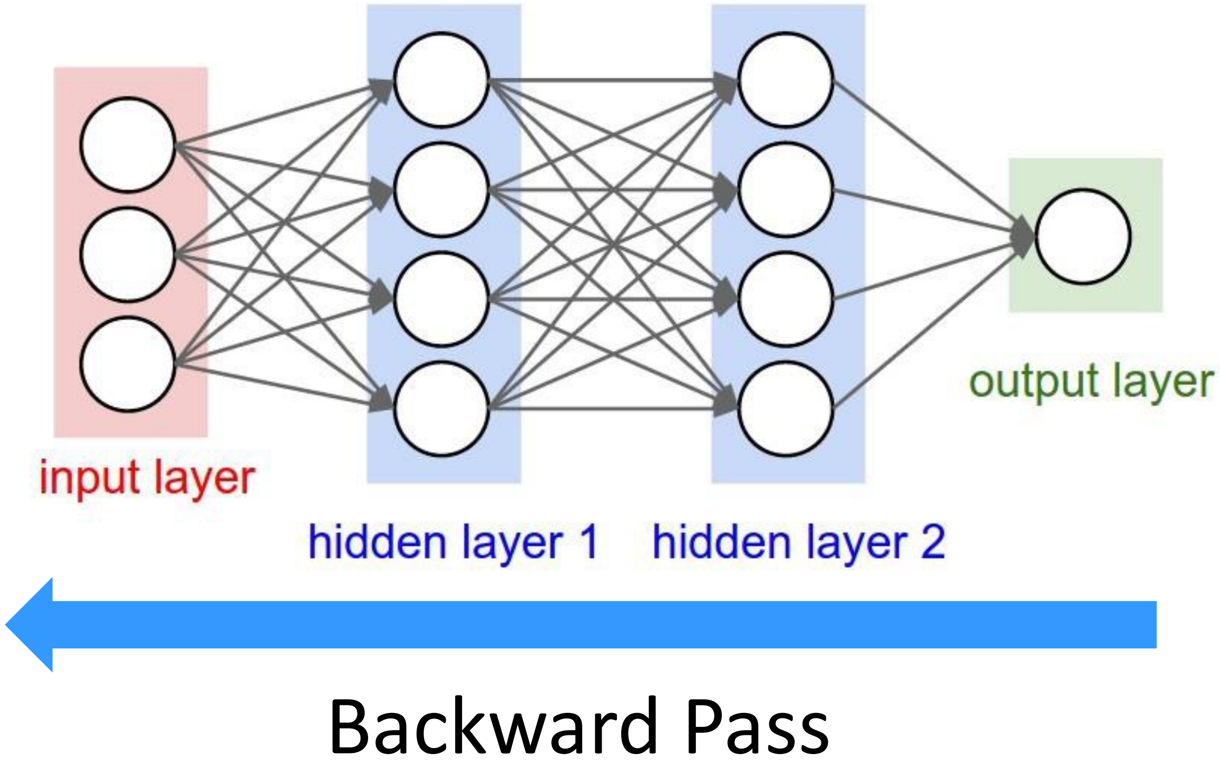
$$Loss(f(x_i, \theta), y_i)$$

[1] Michael Nielsen, *Neural Networks and Deep Learning, Chapter 2* PDF

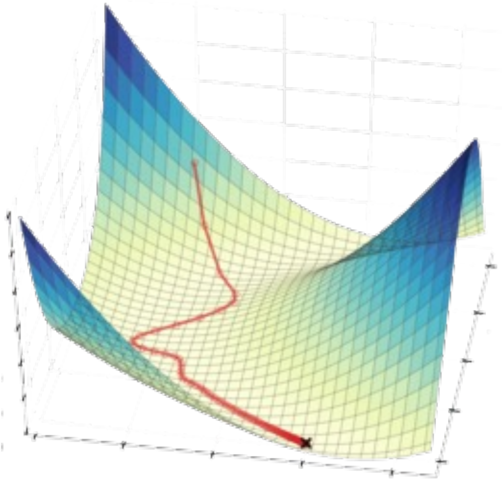
Optimization: Back-propagation

Compute gradients with respect to all parameters and perform **gradient descent**

$$\theta_{new} = \theta_{old} - \mu \nabla_{\theta} Loss$$



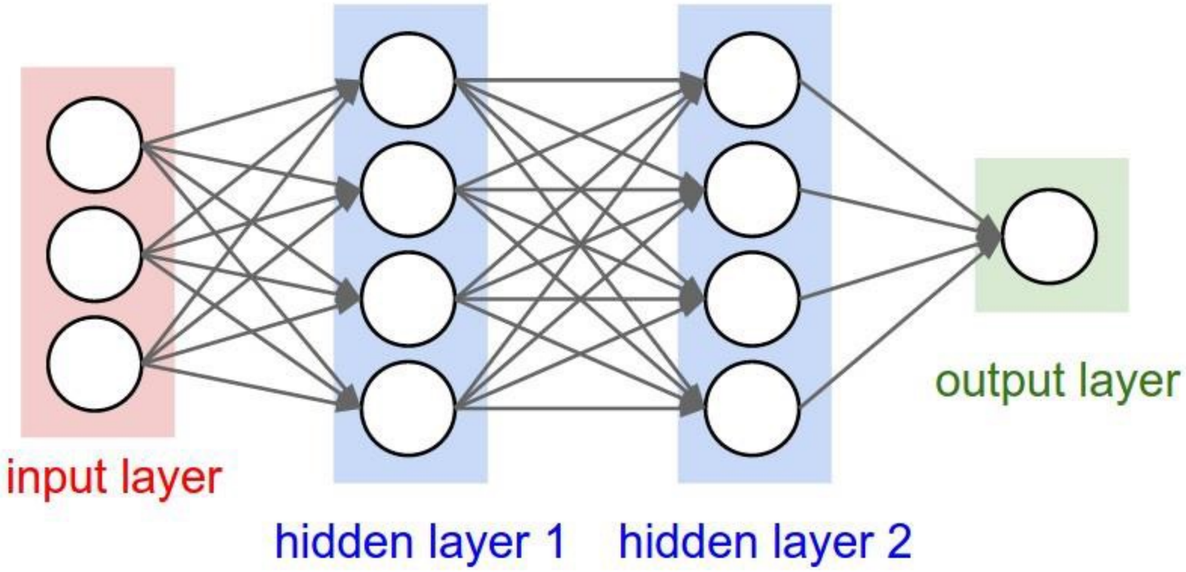
$$Loss(f(x_i, \theta), y_i)$$



[1] Michael Nielsen, *Neural Networks and Deep Learning, Chapter 2* PDF
[2] Dreyfus, *Artificial Neural Networks, Back Propagation and the Kelley-Bryson Gradient Procedure,* Journal of Guidance, 1989. *PDF*

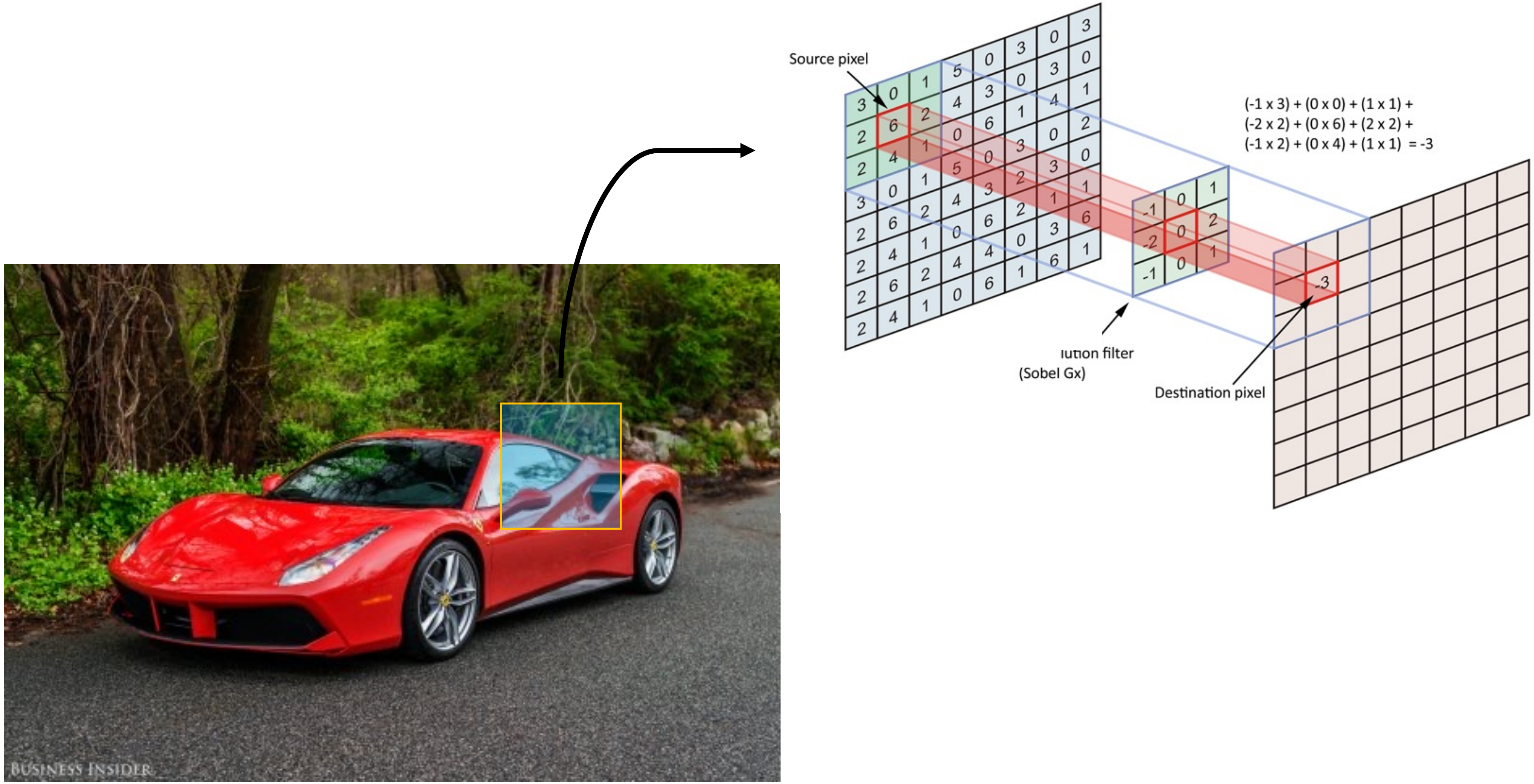
Problems of fully connected network

Too many parameters → **possible overfitting**.



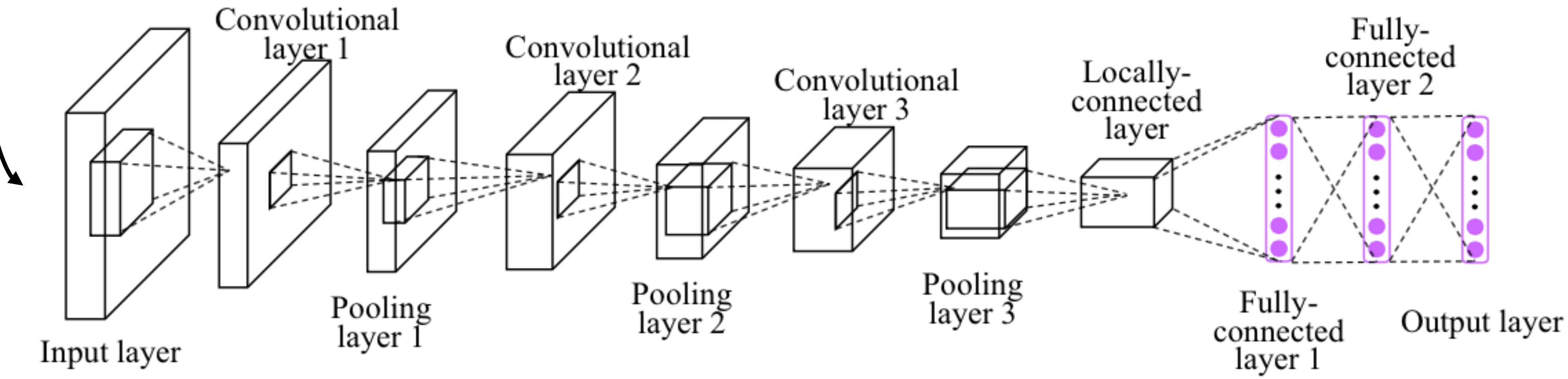
However, we are not using the fact that inputs are images!

Convolutional Neural Networks

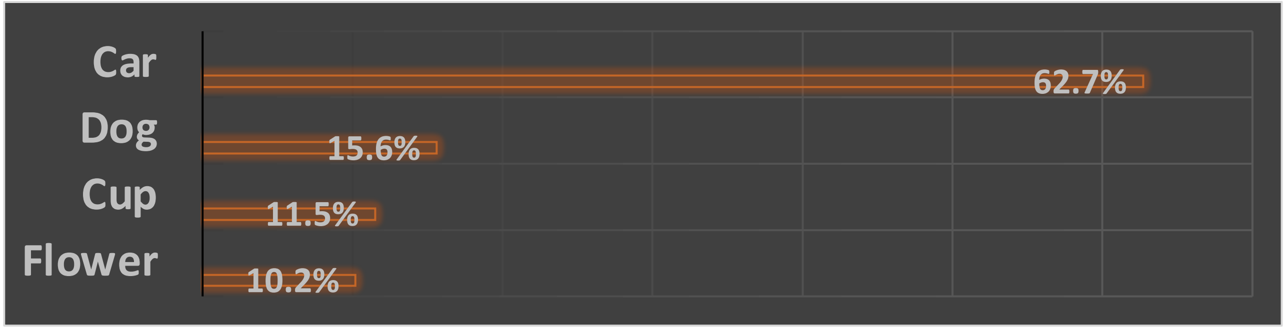


[1] LeCun, Bottou, Bengio, Haffner, *Gradient-based learning applied to document recognition*, Proceedings of the IEEE, 1998. PDF

Going Deep



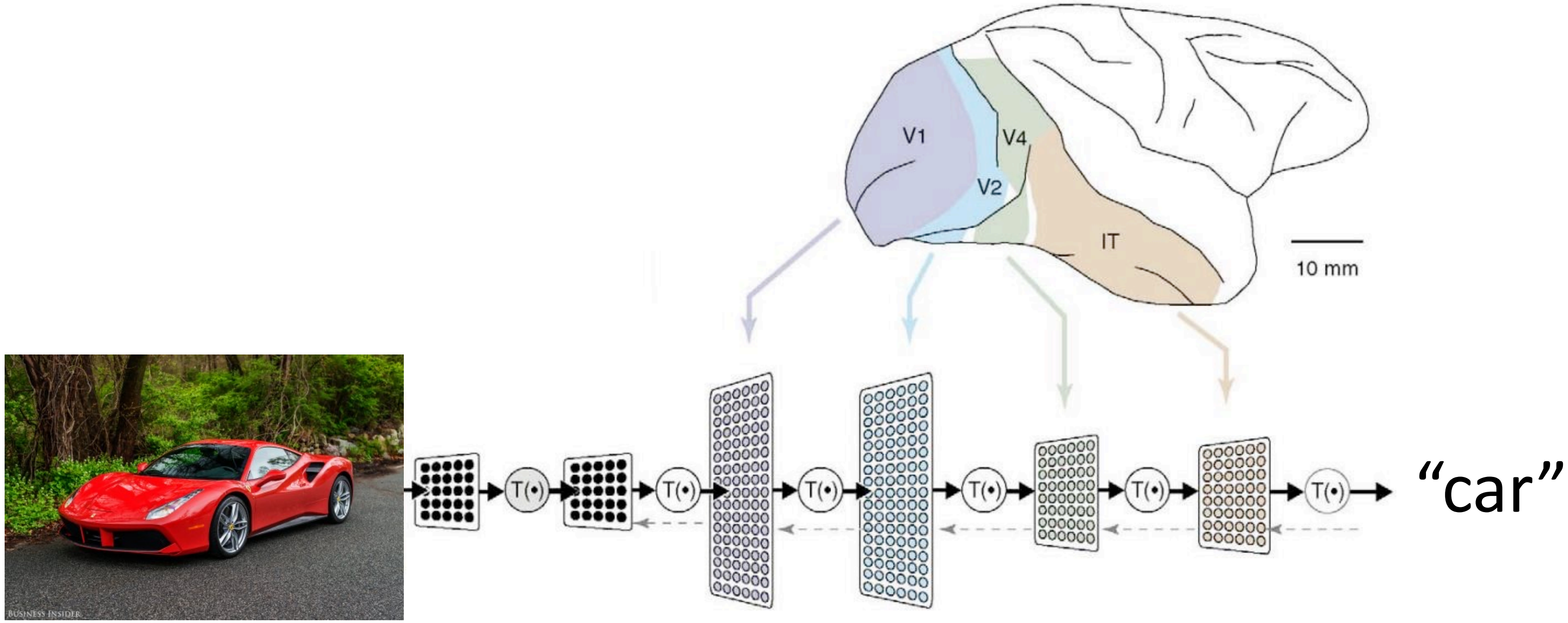
Input layer
Convolutional layer 1
Pooling layer 1
Convolutional layer 2
Pooling layer 2
Convolutional layer 3
Pooling layer 3
Locally-connected layer
Fully-connected layer 1
Fully-connected layer 2
Output layer



Car
62.7%
Dog
15.6%
Cup
11.5%
Flower
10.2%

Why Deep?

1. Inspired by the **human visual system**
2. Learn **multiple layers** of transformations of input
3. Extract progressively more **sophisticated representations**

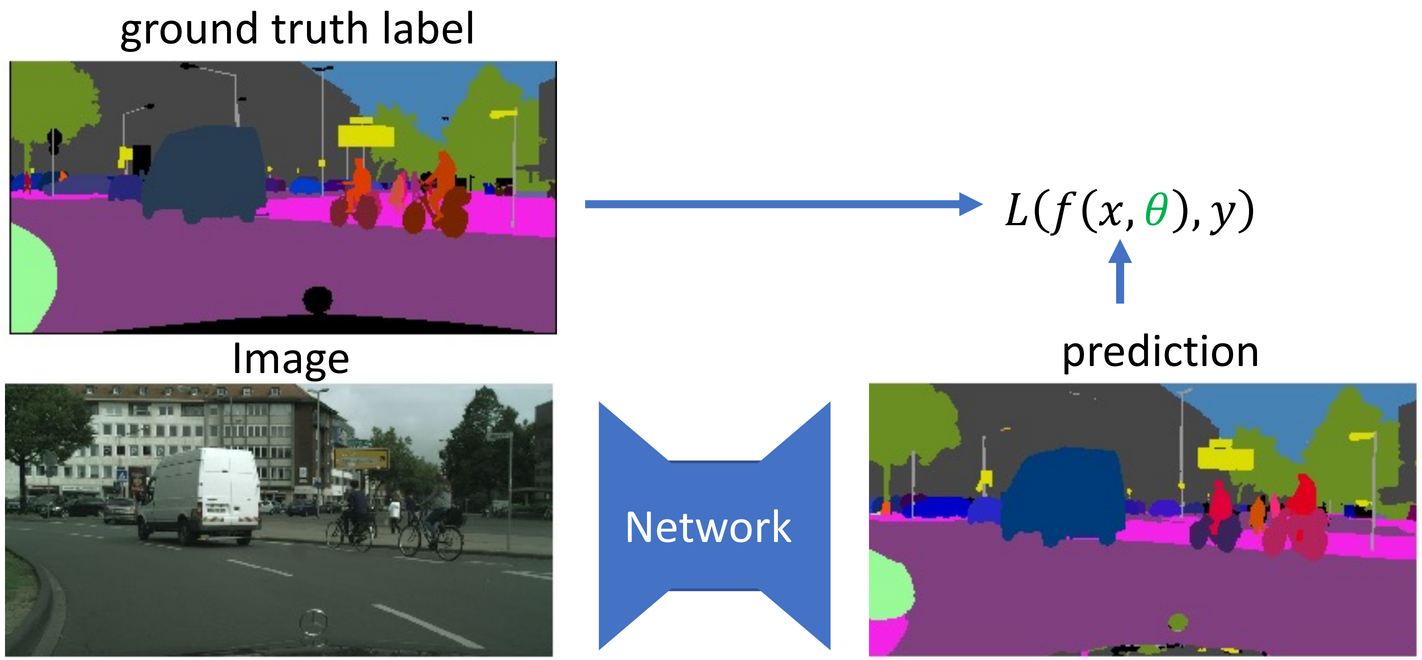


Outline

- Introduction
- Supervised Learning
- Unsupervised Learning
- Applications to Computer Vision
- Conclusions
- Machine Learning for Drones

Supervised Learning

- In supervised learning we assume have access to both **input data or images** and **ground truth labels**.
- Networks trained with supervision usually perform best
- However, getting ground truth is hard, since it often must be hand-labelled



Supervised Learning

- Image Segmentation

[1] Long, Shelhamer, *Fully Convolutional Networks for Semantic Segmentation,* Conference of Computer Vision and Pattern Recognition *(CVPR)*, 2015. PDF

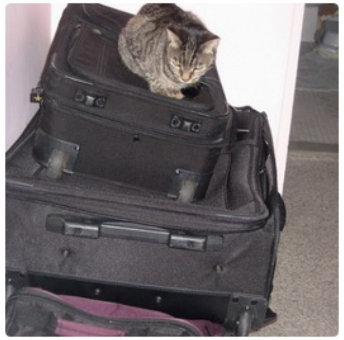
Supervised Learning

- Image Captioning

"little girl is eating piece of cake."

"baseball player is throwing ball in game."

"woman is holding bunch of bananas."



"black cat is sitting on top of suitcase."

"a young boy is holding a baseball bat."

"a cat is sitting on a couch with a remote control."

"a woman holding a teddy bear in front of a mirror."

"a horse is standing in the middle of a road."

[1] Karpathy, Fei-Fei, *Deep Visual-Semantic Alignments for Generating Image Descriptions,* Conference of Computer Vision and Pattern Recognition (CVPR), 2015. [PDF]

Supervised Learning

- Image Localization

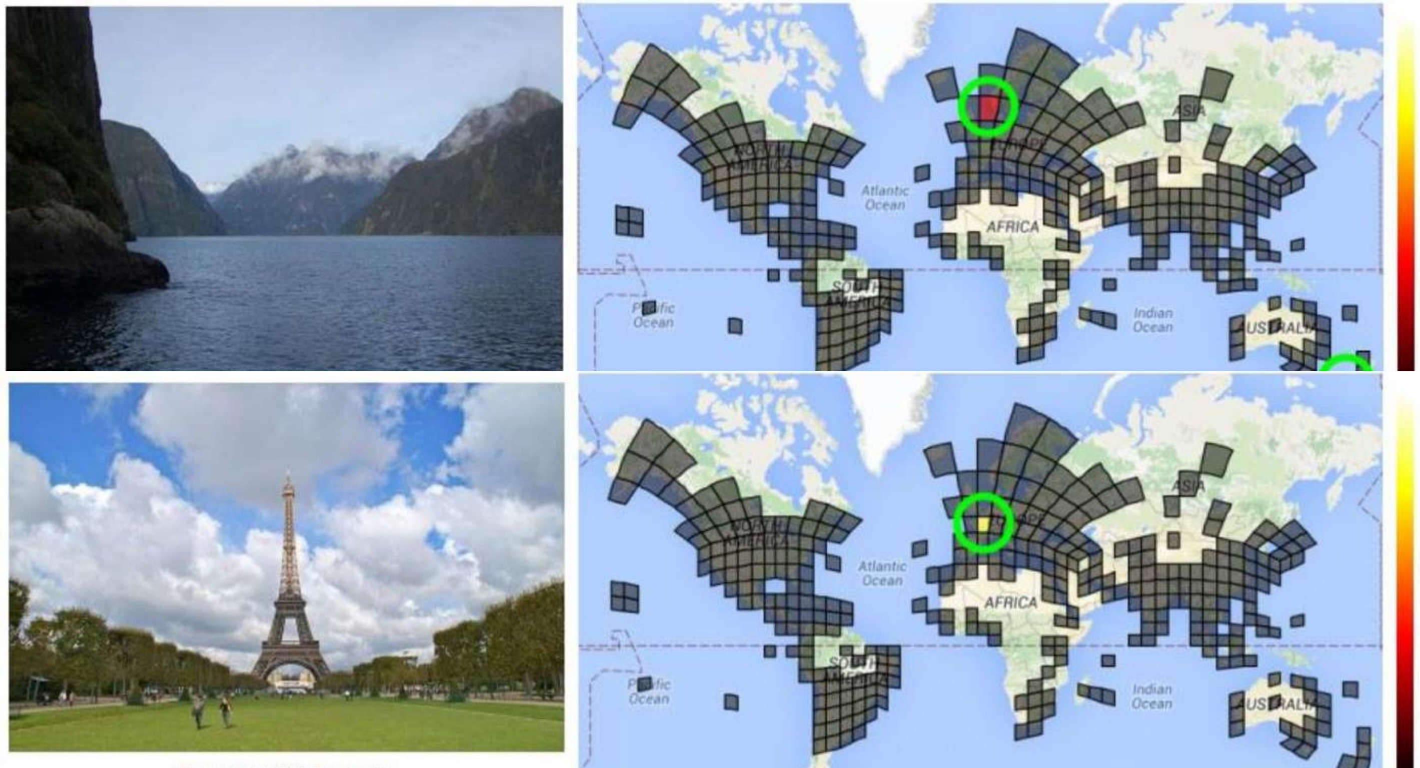


Photo CC-BY-NC by stevekc

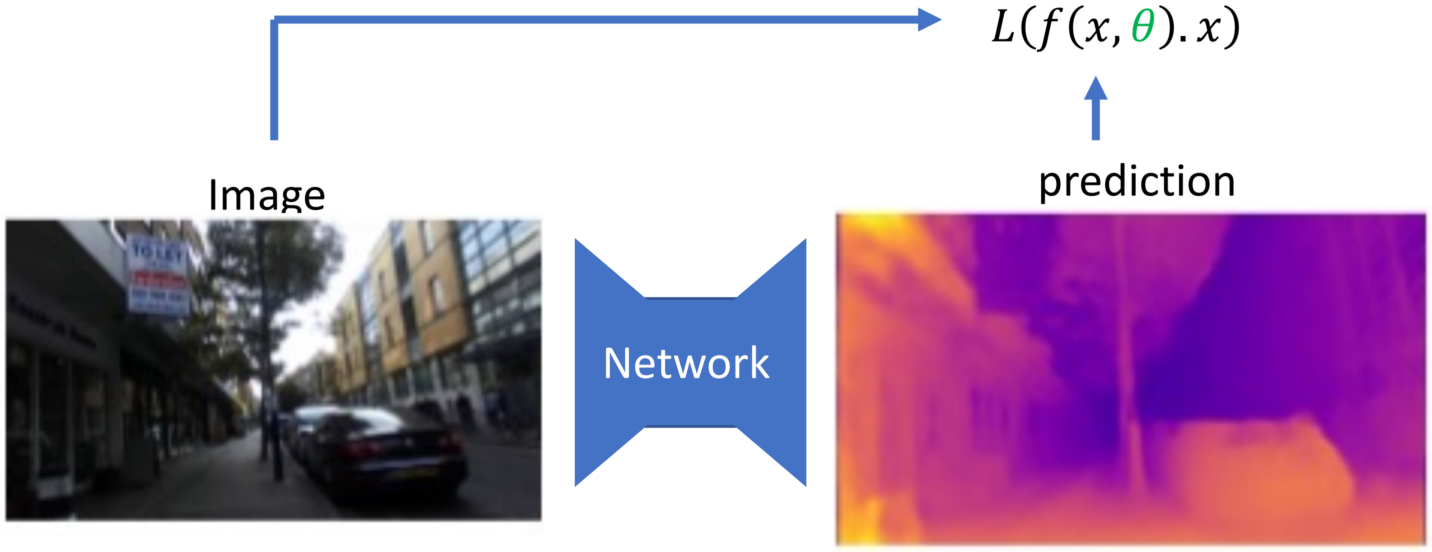
[1] Weyland, Kostrikov, Philbin, *PlaNet - Photo Geolocation with Convolutional Neural Networks,* European Conference on Computer Vision (ECCV), 2016. PDF

Outline

- Introduction
- Supervised Learning
- Unsupervised Learning
- Applications to Computer Vision
- Conclusions
- Machine Learning for Drones

Unsupervised Learning

- In unsupervised learning we only have access to input data or **images**.
- Usually, these methods are more popular because they can use much larger datasets that do not need to be manually labelled.



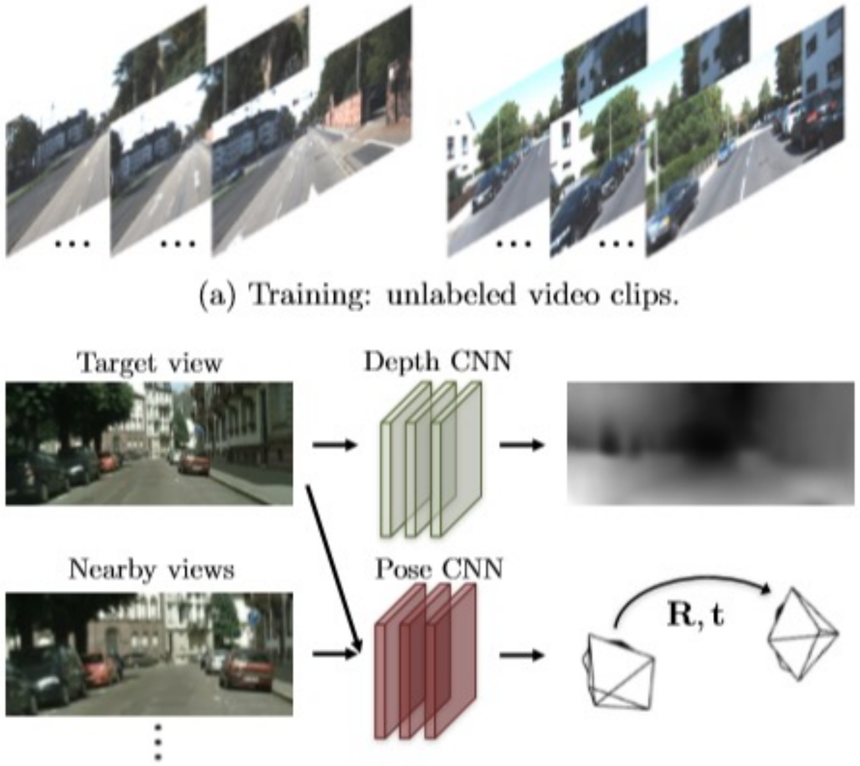
Unsupervised Learning

- Monocular Depth Estimation

[1] Godard, Mac Aodha, Brostow, *Unsupervised Monocular Depth Estimation with Left-Right Consistency* , Conference of Computer Vision and Pattern Recognition (CVPR), 2017. [PDF]

Unsupervised Learning

- Structure from Motion



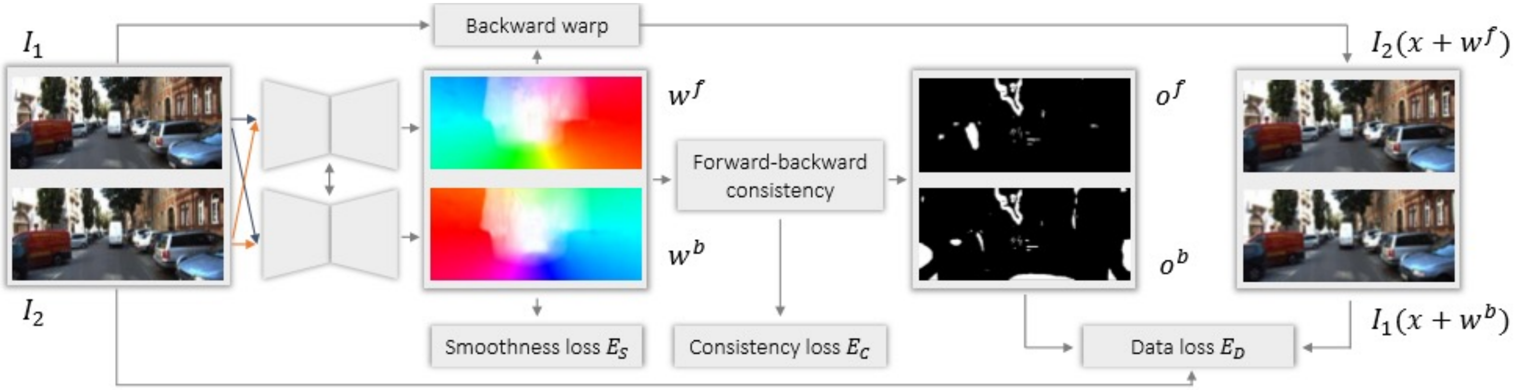
[1] Zhou, Brown, Snavely, Lowe, *Unsupervised Learning of Depth and Ego-Motion from Video,* Conference of Computer Vision and Pattern Recognition (CVPR), 2017. PDF

Unsupervised Learning

- Dense Optical Flow

Characteristic of the learned flow:

- Robustness against light changes (Census Transform)
- Occlusion handling (Bi-directional Flow)
- Smooth flow



[1] Meister, Hur, Roth, *Unsupervised Learning of Optical Flow with a Bidirectional Census Loss*, Association for the Advancement of Artificial Intelligence (AAAI), 2018. PDF

Unsupervised vs. Supervised learning

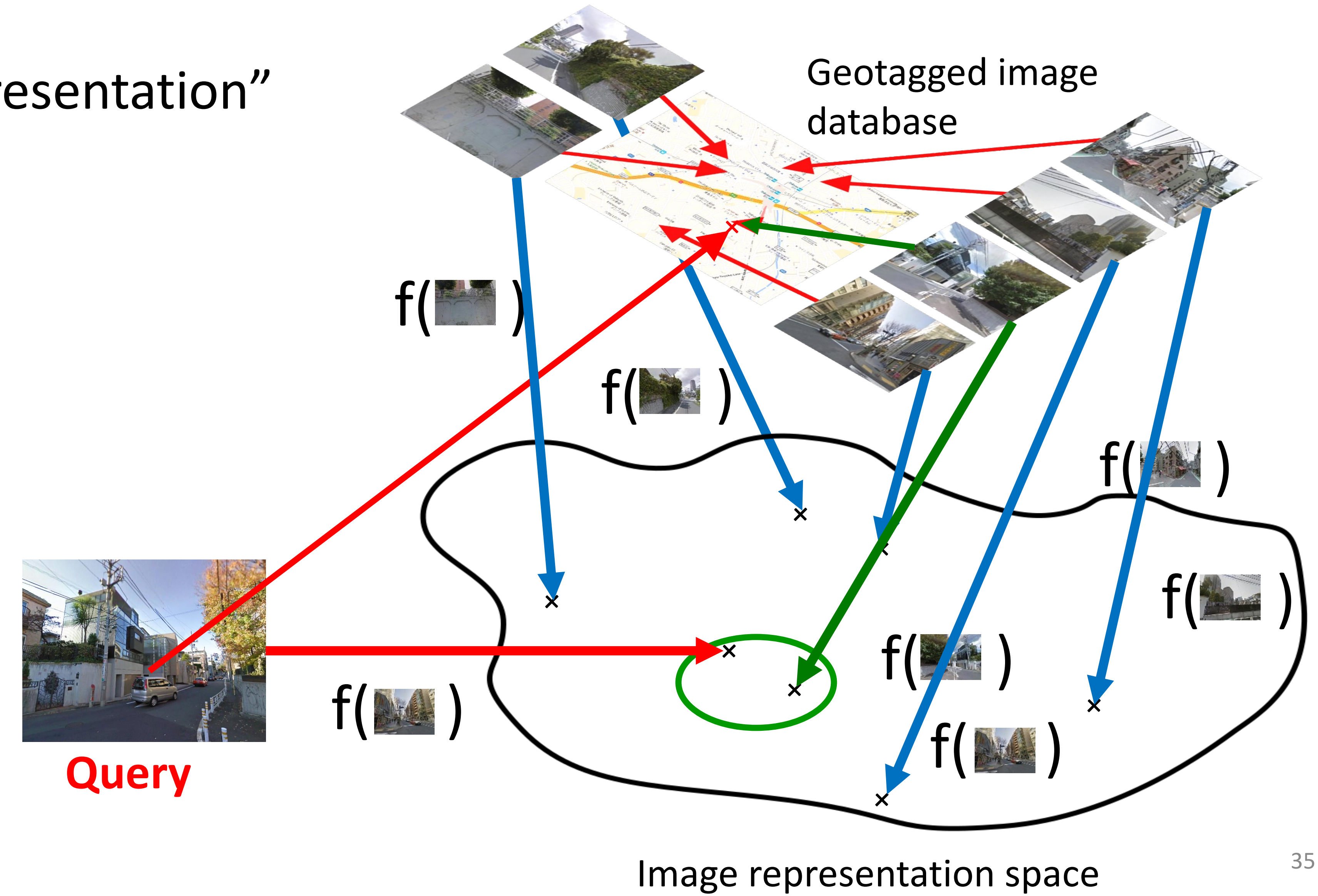
	Supervised	Unsupervised
Performance	Usually better for the same dataset size.	Usually worse, but can outperform supervised methods due to larger data availability.
Data availability	Low, due to manual labelling.	High, no labelling required.
Training	Simple, ground truth gives a strong supervision signal.	Sometimes difficult, loss functions have to be engineered to get good results.
Generalizability	Good, although sometimes the network learns to blindly copy the labels provided, leading to poor generalizability.	Better, since unsupervised losses often encode the task in a more fundamental way.

Outline

- Introduction
- Supervised Learning
- Unsupervised Learning
- Applications to Computer Vision
- Conclusions
- Machine Learning for Drones

Place Recognition – NetVLAD

- Design an "image representation" extractor $f(I, \theta)$

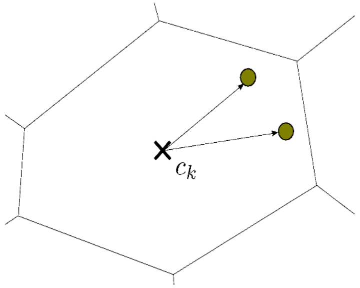


NetVLAD – Method

- Mimic the classical pipeline with deep learning

Image I

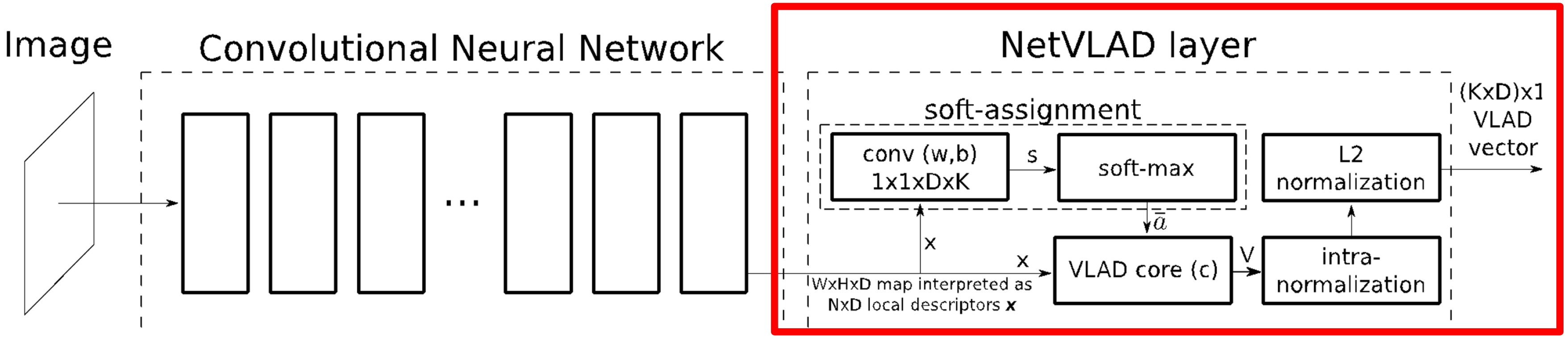
Extract local features (SIFT)



Aggregate (BoW, **VLAD**, FV)

$$\begin{pmatrix} 2 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ \cdots \end{pmatrix}$$

F(I)



Trainable pooling layer

Arandjelović, Gronat, Torii, Pajdla, Sivic, NetVLAD: CNN architecture for weakly supervised place recognition, Conference of Computer Vision and Pattern Recognition (CVPR), 2016. PDF

NetVLAD – Loss

- Triplet loss formulation

$D_p = ||F_\theta($  $) - F_\theta($  $)||^2 \longrightarrow$ Matching samples

$D_n = ||F_\theta($  $) - F_\theta($  $)||^2 \longrightarrow$ Non matching samples

$$L_\theta = \sum_{samples} \max(D_{p(\theta)} + \underset{\text{margin}}{m} - D_{n(\theta)}, 0)$$

Disclaimer: The actual NetVlad loss is a slightly more complicated version of the one above

Arandjelović, Gronat, Torii, Pajdla, Sivic, NetVLAD: CNN architecture for weakly supervised place recognition, Conference of Computer Vision and Pattern Recognition (CVPR), 2016. PDF

NetVLAD – Results

- Code, dataset and trained network online: give it a try here!

Query

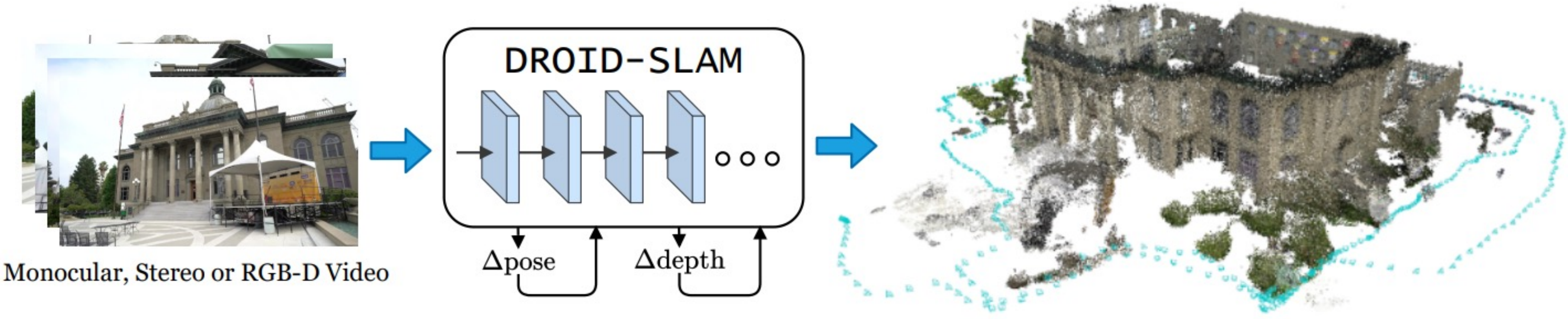
Top result

Green: Correct Red: Incorrect

Arandjelović, Gronat, Torii, Pajdla, Sivic, NetVLAD: CNN architecture for weakly supervised place recognition, Conference of Computer Vision and Pattern Recognition (CVPR), 2016. PDF

Simultaneous Localization and Mapping – DROID-SLAM

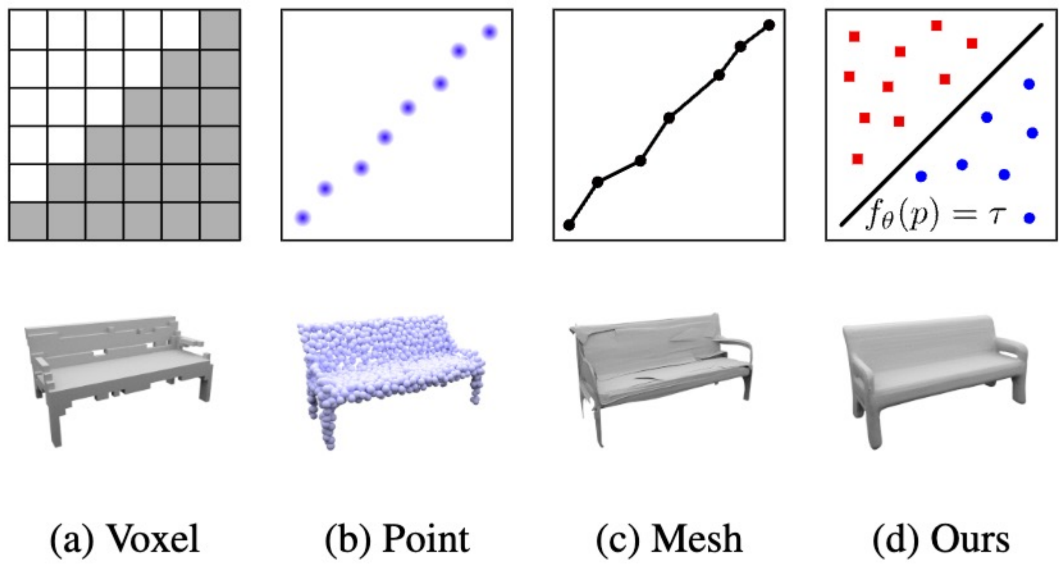
- End-to-end trained method that computes camera pose and depth from images directly
- Depth and pose refinement through recurrent connection
- **Uses learned, dense bundle adjustment as a key building block**

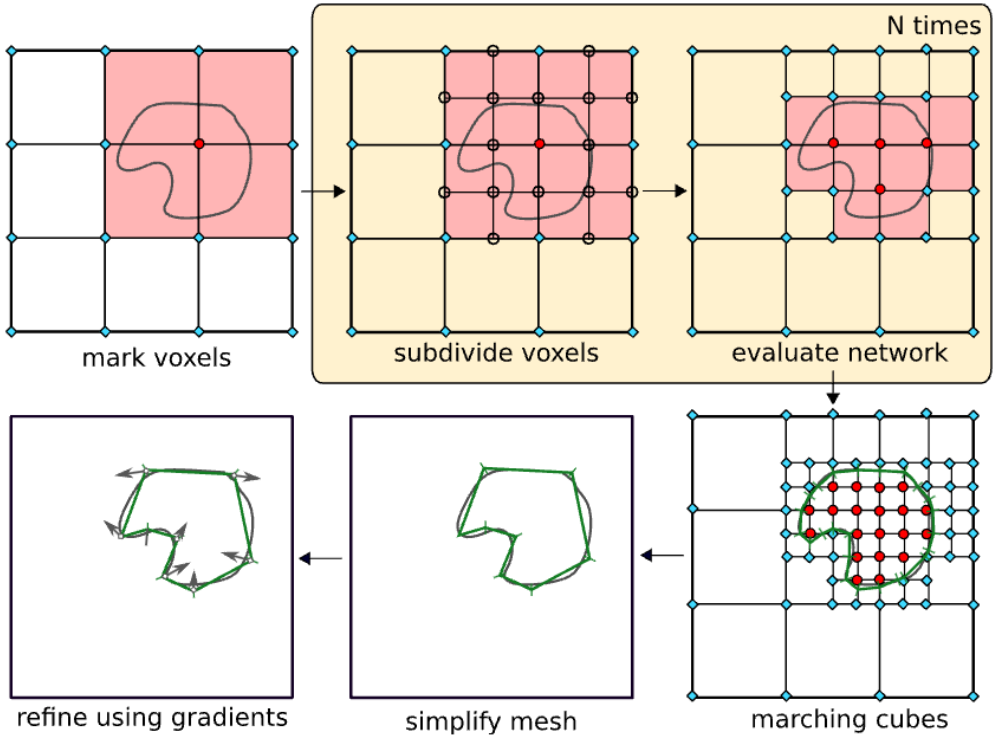


[1] Teed, Deng, *DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras*, Conference on Neural Information Processing Systems (NeurIPS) 2021 [PDF]

Occupancy Networks – 3D Shape as Decision Boundary

Dense 3D Reconstruction as Classification





[1] Mescheder, Oechsle, Niemeyer, Nowozin, Geiger: Occupancy Networks: Learning 3D Reconstruction in Function Space, CVPR 2019, [PDF]

Novel View Synthesis – Neural Radiance Fields (NeRF)

Render new views from a set of images with corresponding poses.

images with poses

reconstructed neural radiance field

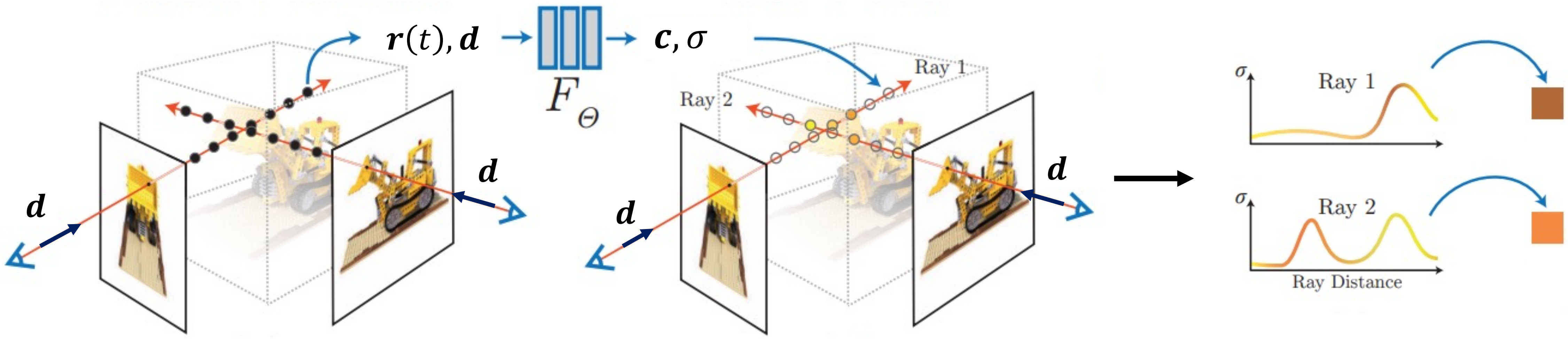
synthesized novel viewpoints

[1] Mildenhall, Srinivasan, Tancik, Barron, Ramamoorthi, Ng, *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis*, European Conference of Computer Vision (ECCV), 2020. PDF

[2] An overview and a reference for many follow-up works can be found here

Neural Radiance Fields (NeRF): Method

Images are rendered by integrating *transmittance and color* along a ray both of which are modelled with a *multilayer perceptron*



Position along ray: $\boldsymbol{r}(t) = \boldsymbol{o} + t\boldsymbol{d}$

Transmittance: $T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\boldsymbol{r}(s))ds\right)$

Ray direction: $\boldsymbol{d}$

Color: $C(r) = \int_{t_n}^{t_f} T(t)\sigma(\boldsymbol{r}(t))\boldsymbol{c}(\boldsymbol{r}(t), \boldsymbol{d})dt$

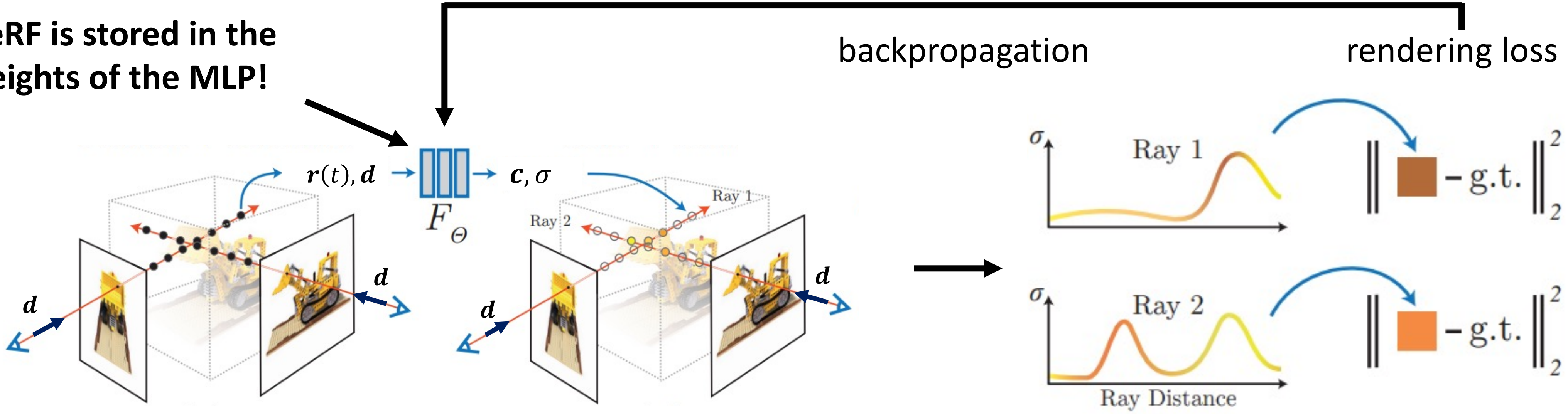
[1] Mildenhall, Srinivasan, Tancik, Barron, Ramamoorthi, Ng, *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis*, European Conference of Computer Vision (ECCV), 2020. PDF

[2] An overview and a reference for many follow-up works can be found here

Neural Radiance Fields (NeRF): Training

We train this multilayer perceptron by minimizing the rendering loss on the input images, thereby effectively overfitting.

NeRF is stored in the weights of the MLP!



[1] Mildenhall, Srinivasan, Tancik, Barron, Ramamoorthi, Ng, *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis*, European Conference of Computer Vision (ECCV), 2020. PDF
[2] An overview and a reference for many follow-up works can be found here

Neural Radiance Fields (NeRF): Results

Compared to previous approaches, NeRF generates highly photorealistic, and consistent novel views.

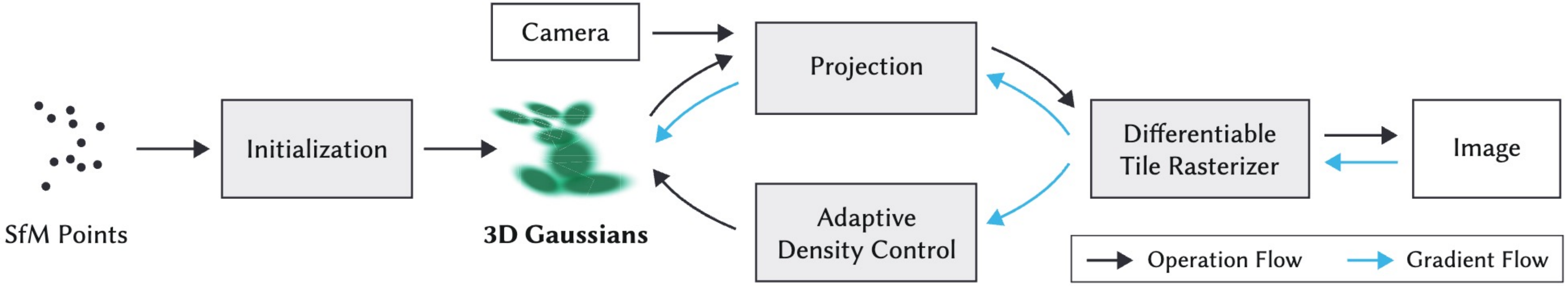
3D Gaussian Splatting

Scenes represented as a collection of **3D Gaussian primitives**, enabling efficient reconstruction and visualization of complex environments with high fidelity.

[1] Kerbl, Kopanas, Leimkühler, Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering, SIGGRAPH 2023, PDF

3D Gaussian Splatting: Method

The approach begins with a sparse SfM point cloud to initialize 3D Gaussians, optimizing their density using a differentiable tile-based rasterizer.



[1] Kerbl, Kopanas, Leimkühler, Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering, SIGGRAPH 2023, PDF

3D Gaussian Splatting: Results

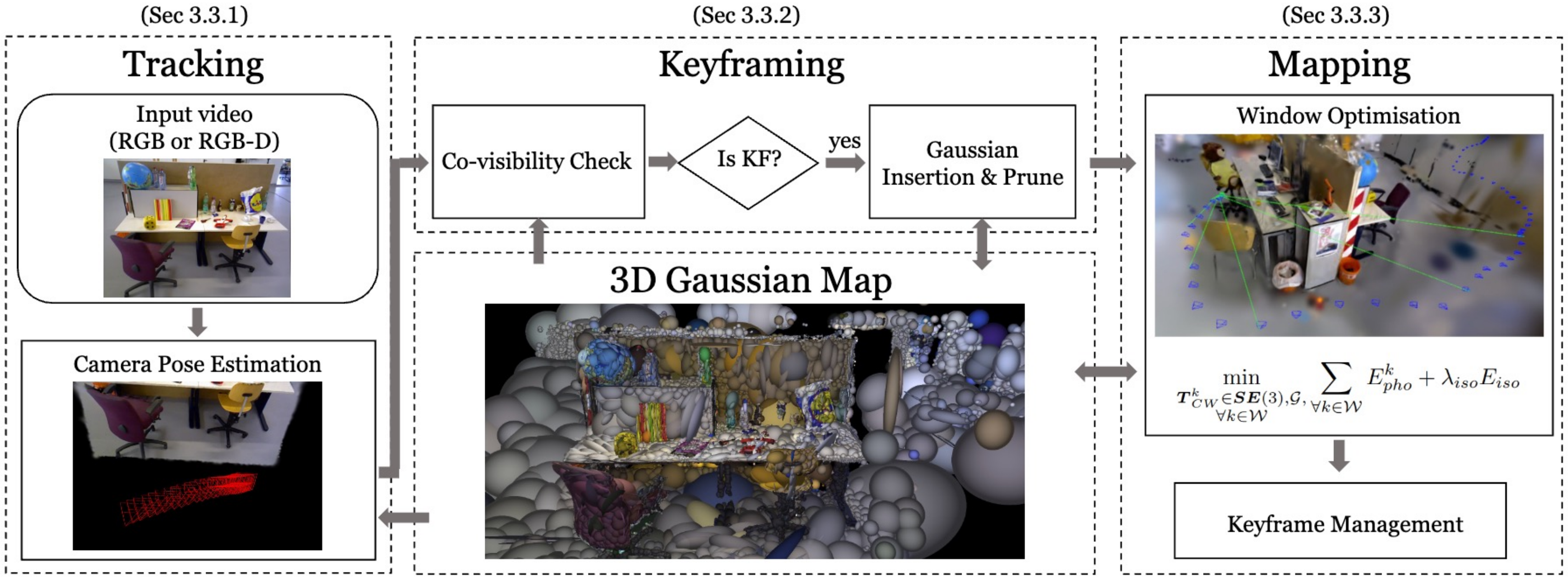
3D GS shows both better **reconstruction quality** and **higher efficiency**,
While NeRF suffers from slow **rendering** and **training**.

Dataset	Mip-NeRF360						Tanks&Temples						Deep Blending					
Method|Metric	*SSIM*↑	*PSNR*↑	*LPIPS*↓	Train	FPS	Mem	*SSIM*↑	*PSNR*↑	*LPIPS*↓	Train	FPS	Mem	*SSIM*↑	*PSNR*↑	*LPIPS*↓	Train	FPS	Mem
Plenoxels	0.626	23.08	0.463	25m49s	6.79	2.1GB	0.719	21.08	0.379	25m5s	13.0	2.3GB	0.795	23.06	0.510	27m49s	11.2	2.7GB
INGP-Base	0.671	25.30	0.371	5m37s	11.7	13MB	0.723	21.72	0.330	5m26s	17.1	13MB	0.797	23.62	0.423	6m31s	3.26	13MB
INGP-Big	0.699	25.59	0.331	7m30s	9.43	48MB	0.745	21.92	0.305	6m59s	14.4	48MB	0.817	24.96	0.390	8m	2.79	48MB
M-NeRF360	0.792†	27.69†	0.237†	48h	0.06	8.6MB	0.759	22.22	0.257	48h	0.14	8.6MB	0.901	29.40	0.245	48h	0.09	8.6MB
Ours-7K	0.770	25.60	0.279	6m25s	160	523MB	0.767	21.20	0.280	6m55s	197	270MB	0.875	27.78	0.317	4m35s	172	386MB
Ours-30K	0.815	27.21	0.214	41m33s	134	734MB	0.841	23.14	0.183	26m54s	154	411MB	0.903	29.41	0.243	36m2s	137	676MB

[1] Kerbl, Kopanas, Leimkühler, Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering, SIGGRAPH 2023, PDF

Gaussian Splatting SLAM

- 3D Gaussians as the only 3D representation
- VO pipeline instead of SLAM (no loop closure)



Monocular SLAM Results (x20)

[1] Matsuki, Murai, Kelly, Davison. Gaussian Splatting SLAM, CVPR 2024, PDF

Gaussian Splatting SLAM

- Better tracking performance than DROID SLAM → GS provides better view synthesis
- Underperform when comparing with full SLAM pipeline like ORB SLAM

Input	Loop-closure	Method	fr1/desk	fr2/xyz	fr3/office	Avg.
Monocular	w/o	DSO [4]	22.4	**1.10**	9.50	11.0
		DROID-VO [36]	5.20	10.7	7.30	7.73
		DepthCov-VO [3]	5.60	1.20	68.8	25.2
		Ours	**3.78**	4.60	**3.50**	**3.96**
	w/	DROID-SLAM [36]	**1.80**	**0.50**	2.80	1.70
		ORB-SLAM2 [20]	1.90	0.60	**2.40**	**1.60**

[1] Matsuki, Murai, Kelly, Davison. Gaussian Splatting SLAM, CVPR 2024, PDF

Outline

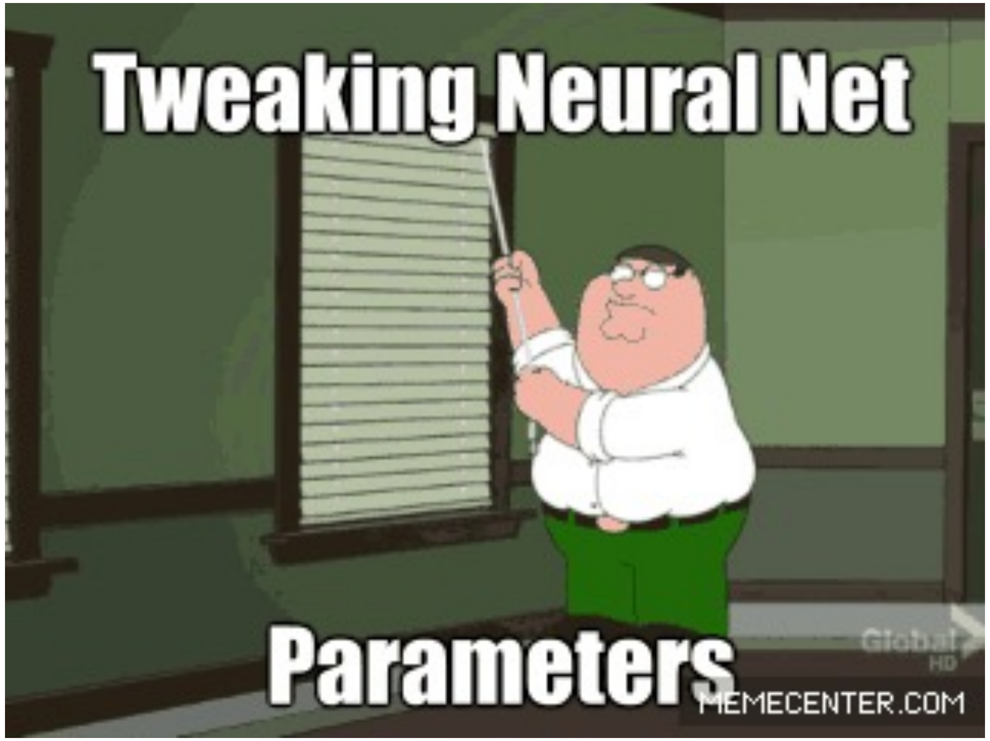
- Introduction
- Supervised Learning
- Unsupervised Learning
- Applications to Computer Vision
- Conclusions
- Machine Learning for Drones

Conclusions

Deep learning, when applied in the correct circumstances, can achieve remarkable performance on a variety of tasks by learning patterns from data

It works especially well when
- Sufficient data is available
- All operations are differentiable

Make sure to avoid the following pitfalls:
- Make sure to optimize the correct metric
- Test your model to an inch of its life
- Always monitor generalization



Additional Readings

- Nielsen, *Neural Networks and Deep Learning*, 2018. PDF
- Bengio, *Practical Recommendations for Gradient-Based Training of Deep Architectures*, 2012. PDF
- Goodfellow, Bengio, Courville, *Deep Learning*, 2016 Website

Outline

- Introduction
- Supervised Learning
- Unsupervised Learning
- Applications to Computer Vision
- Conclusions
- Machine Learning for Drones

The drone market is valued $24 billions today

Inspection

Agriculture

Transport

Search and Rescue

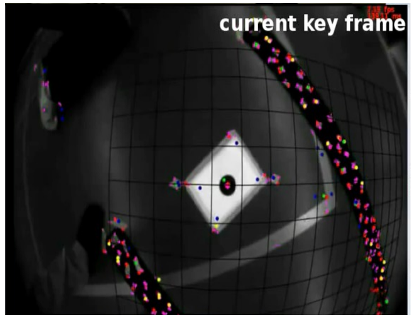
Source: Swiss Drone Industry Report 2021, p. 22:
https://drive.google.com/file/d/1ljesolDoUu1-IVX14nqJRCT-wpEQB22_/view

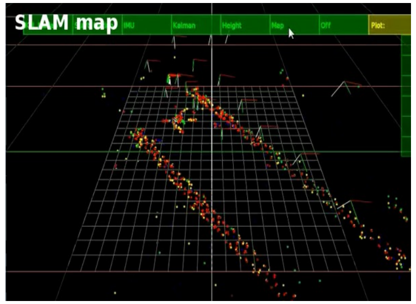
How are current commercial drones controlled?

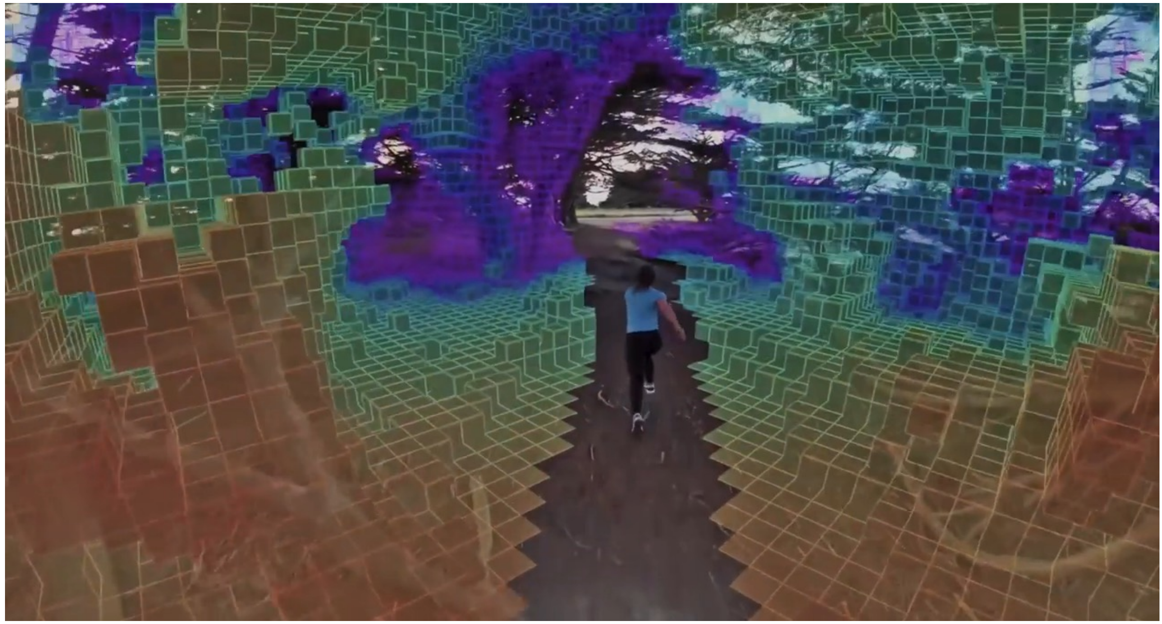
- **By a human pilot**
  - requires **line of sight** or **video link**
  - requires a **lot of training**

- **By an autopilot:** autonomous navigation
  - **GPS**: doesn’t work in GPS denied or degraded environments
  - **Lidar** (e.g., Exyn): expensive, heavy, power hungry
  - **Cameras** (e.g., Parrot, DJI, Skydio): cheap, lightweight, passive (i.e., low power)

Last 10-years Progress on Autonomous Vision-based Flight







2010

EU SFLY Project (2009-2012)

[Bloesch, ICRA 2010]

1st onboard goal-oriented vision-based flight

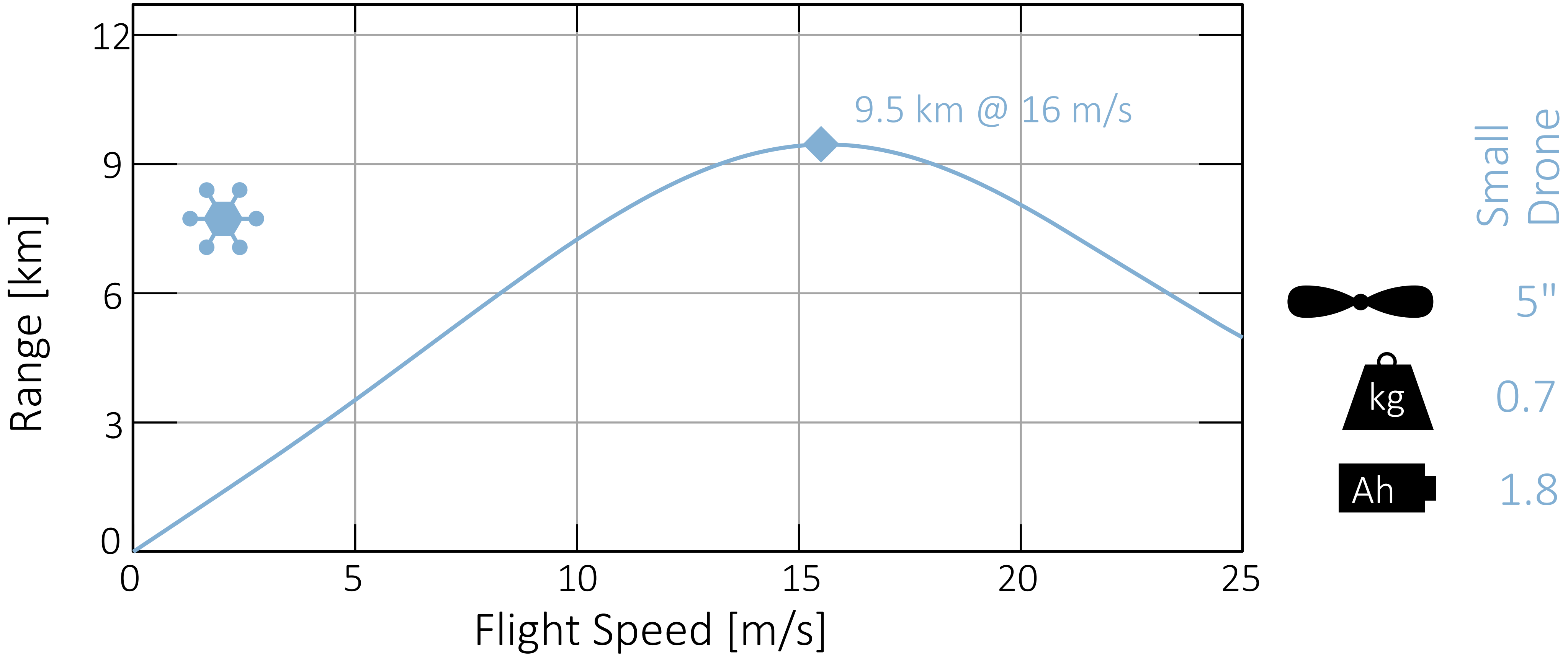
(previous research focused on reactive navigation)

2020

- **Skydio** (2018-2020),
- **DJI** (2018-2020),
- **NASA Mars Helicopter** (2020)

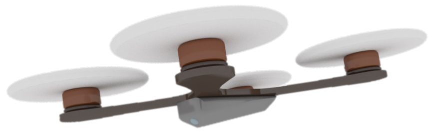
1st products in the market or sent to another planet ☺

Flying Fast to Fly Far



Bauersfeld, Scaramuzza, *Range, Endurance, and Optimal Speed Estimates for Multicopters*, IEEE RAL, 2022. PDF

Related Work: The Traditional Approach



Images
IMU

→ Perception → Planning → Control →

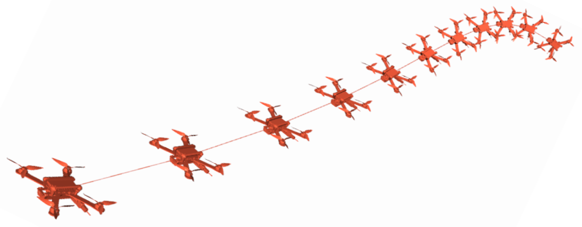
Command

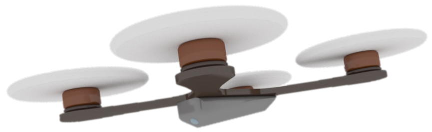
State Estimation
Mapping

Polynomial Trajectories
Trajectory Optimization

PID Control
Model Predictive Control
Reinforcement Learning
Iterative Learning Control

Related Work: The Problem





Images
IMU

Perception → Planning → Control

Command

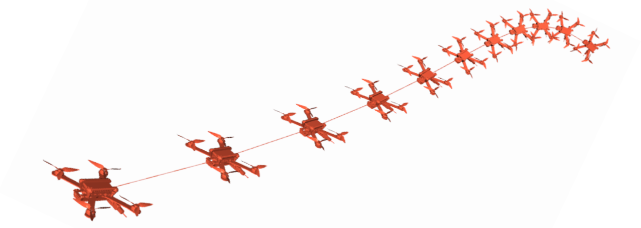
Mature Algorithms, but brittle during high speed due to motion blur.

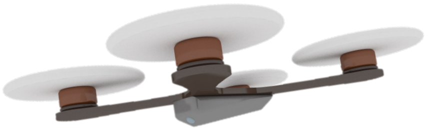
Require strong assumptions about the environment (e.g, CAD of scene).

Needs significant tuning, especially at high speed.

! This fine-grained modularity makes the robotic system fragile: The modules do not interact with each other.

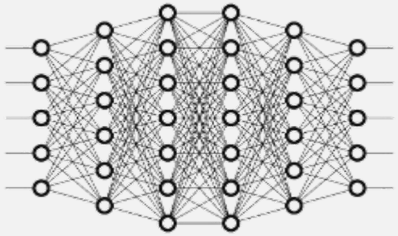
Related Work: End-to-End Learning





Images
IMU

Neural Network



Command

High-Level commands (forward, left, right)

Low-Level commands (collective thrust, bodyrates)

! Don't exploit the agile dynamics of the drone

! Too sample inefficient to be used on a physical drone. Only shown in Sim.

Our Research

Augment the traditional robotic cycle with learning-based methods.

Hypothesis:
Neural Networks can distill the knowledge of mature robotics algorithms into computationally efficient and robust sensorimotor policies.

Projects

- Learning High-Speed Flight in the Wild

- NeuroBEM: Hybrid Aerodynamic Quadrotor Model

- Autonomous Drone Racing

Learning High-Speed Flight in the Wild

What does it take to achieve similar **spatial awareness** to a human **with comparable sensing (and computing)** in the context of **high-speed flight**?

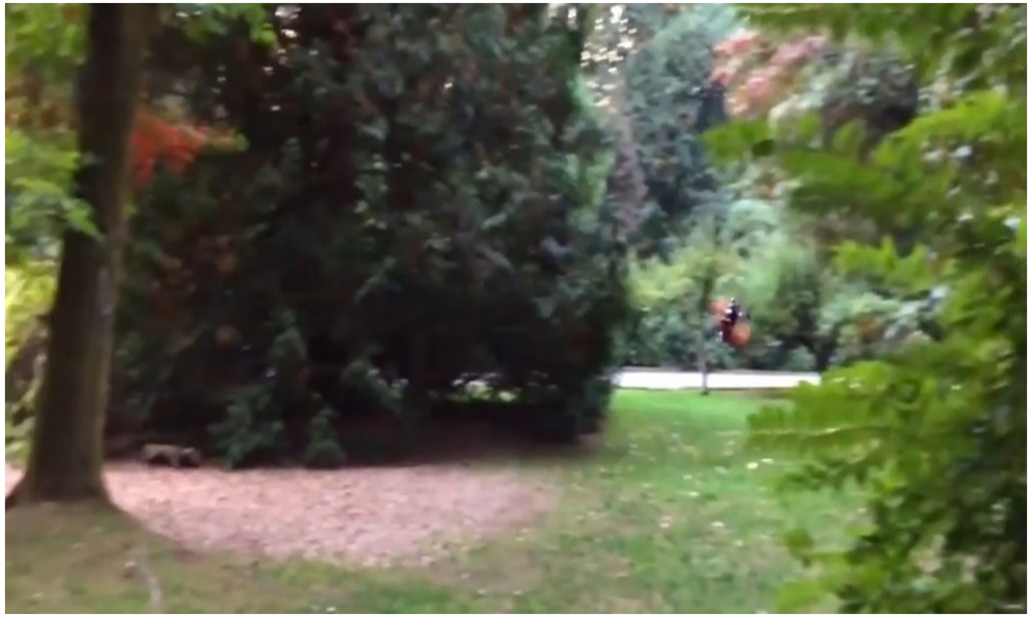
Assumptions:

- No external sensing or computing.
- Test environment not seen in advance.
- Possibly dynamic environment.

Available Information:

- Visual Feedback (multiple cameras).
- Inertial Feedback.
- An intention (e.g. fly straight).

Human pilots fly under similar assumptions!

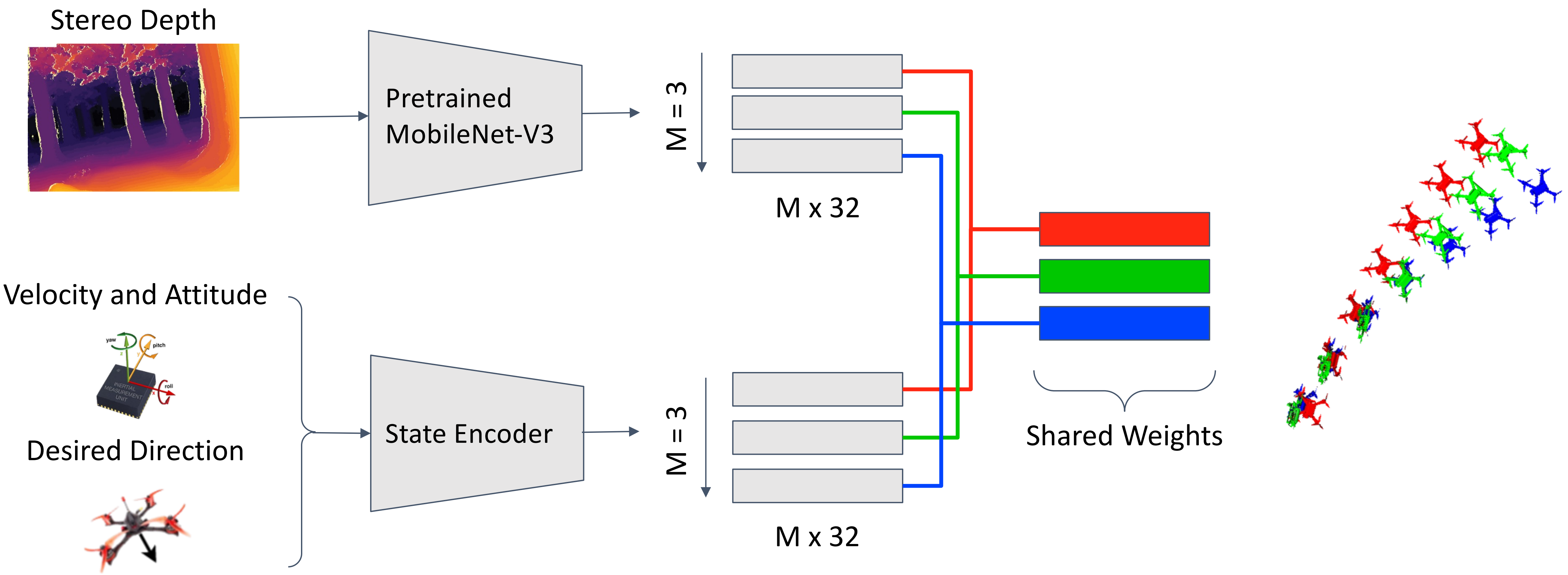


Loquercio, Kaufmann, Ranftl, Mueller, Koltun, Scaramuzza: Learning, High-Speed Flight in the Wild, Science Robotics, 2021. [PDF](), [Video](), [Code]()

Loquercio, Kaufmann, Ranftl, Mueller, Koltun, Scaramuzza: Learning, High-Speed Flight in the Wild, Science Robotics, 2021. PDF, Video, Code

Multiple-Hypothesis Action Prediction

We predict collision-free **receding-horizon trajectories** using a **neural network** with access to visual and inertial observations, as well as a reference velocity.



Loquercio, Kaufmann, Ranftl, Mueller, Koltun, Scaramuzza: Learning, High-Speed Flight in the Wild, Science Robotics, 2021. PDF, Video, Code

Training Procedure

We follow the **privileged learning paradigm** to train the network **purely in simulation***.

1. **Design an <u>expert planner</u> with access to full knowledge of the environment.**
   This expert uses a fine-grained point-cloud of the scene to find collision-free trajectories with sampling.

2. **Distill the knowledge of the expert into a deep neural network**.
   Basically do imitation learning from a set of expert demonstrations.

This simple idea hides quite some challenges!

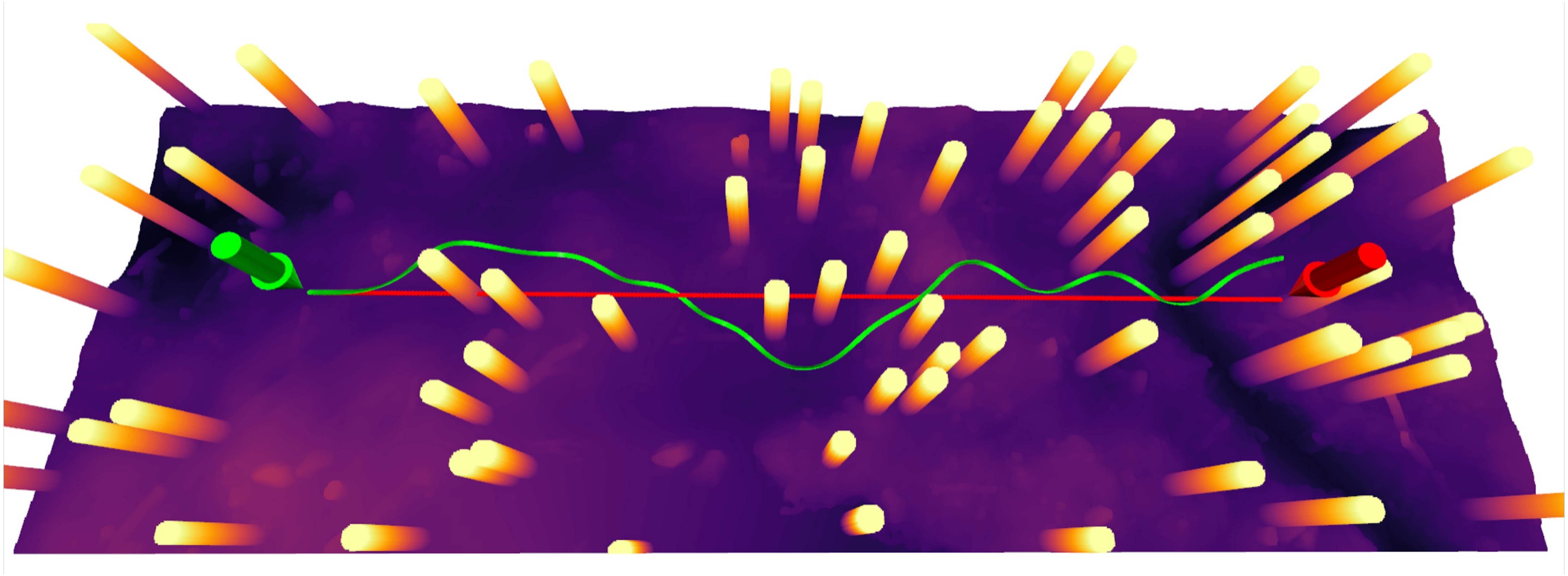
* Impossible to collect a dataset of real-world demonstrations since it is not possible (or very expensive) to have a perfect map of the environment.

Loquercio, Kaufmann, Ranftl, Mueller, Koltun, Scaramuzza: Learning, High-Speed Flight in the Wild, Science Robotics, 2021. PDF, Video, Code

Loquercio, Kaufmann, Ranftl, Mueller, Koltun, Scaramuzza: Learning, High-Speed Flight in the Wild, Science Robotics, 2021. PDF, Video, Code

Controlled Experiments

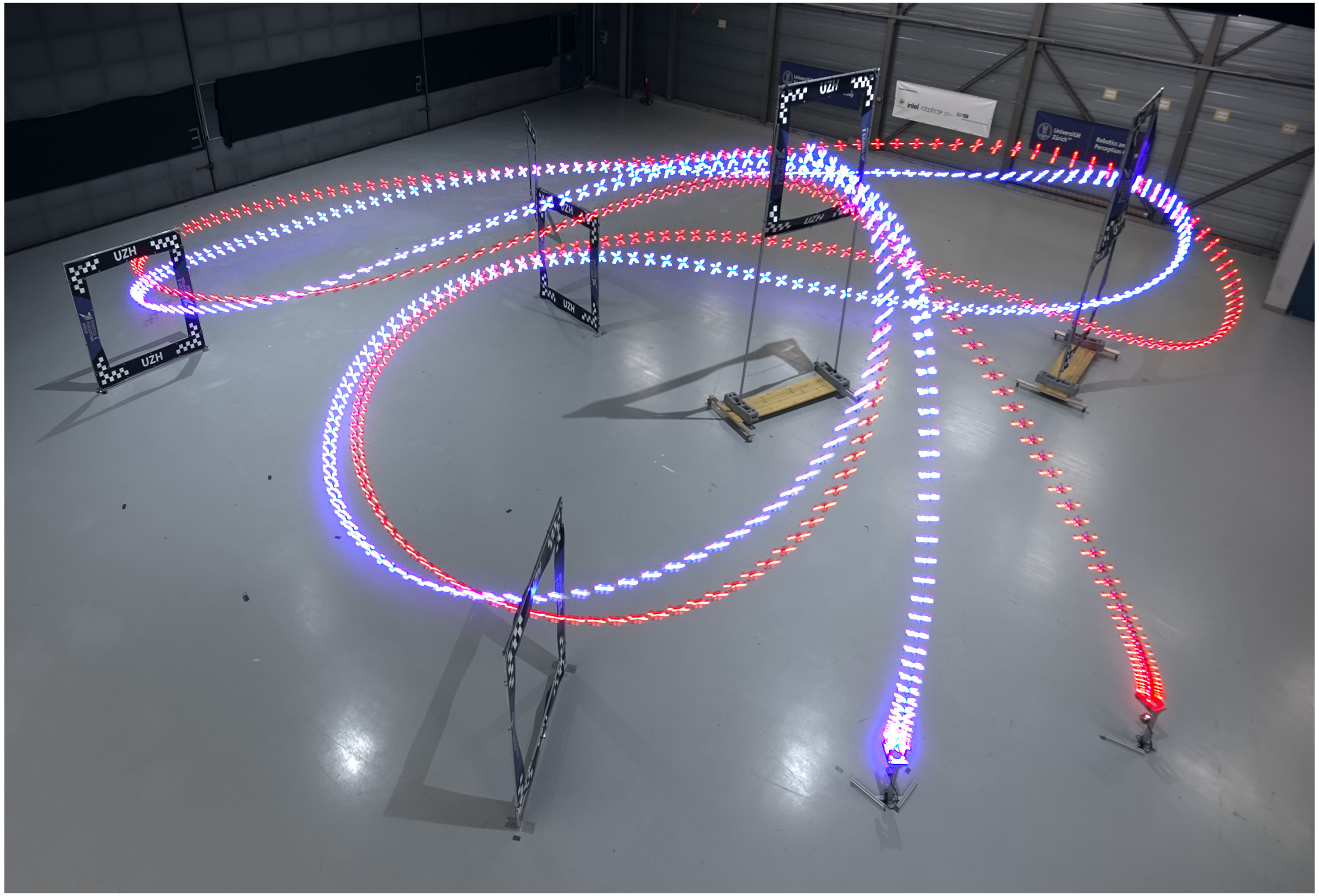
Evaluate on the task of reaching a goal with no prior knowledge about the scene.



Loquercio, Kaufmann, Ranftl, Mueller, Koltun, Scaramuzza: Learning, High-Speed Flight in the Wild, Science Robotics, 2021. PDF, Video, Code

Drone Racing – A Proxy Task

- pass a sequence of gates in the correct order
- fly a given number of laps in minimum time
- be quicker than the opponent



Our "Swift" Drone

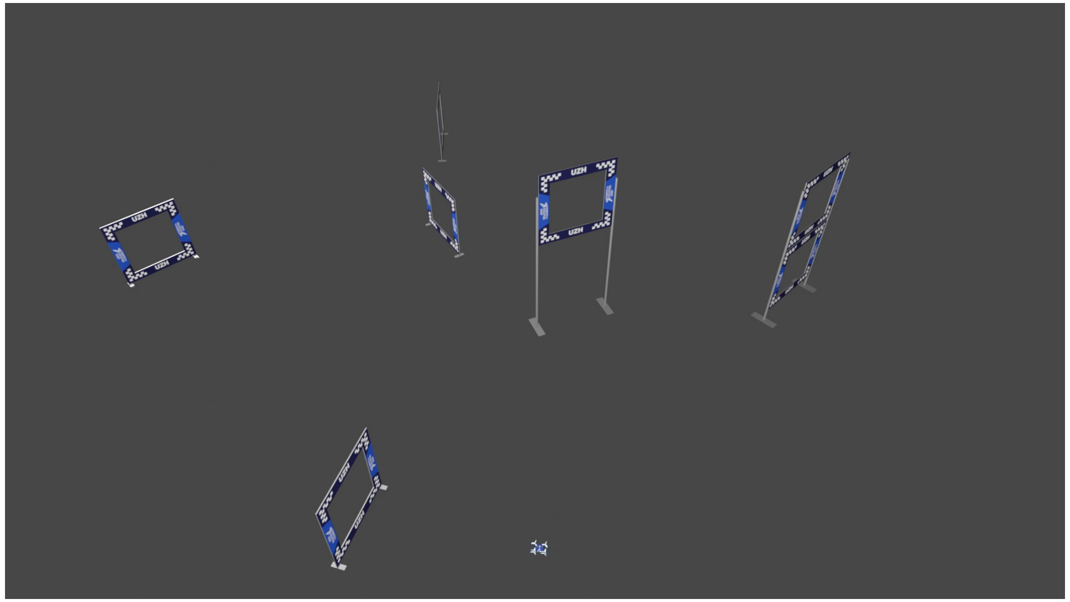
- Jetson TX2
- Realsense T265
  - Images
  - IMU/VIO

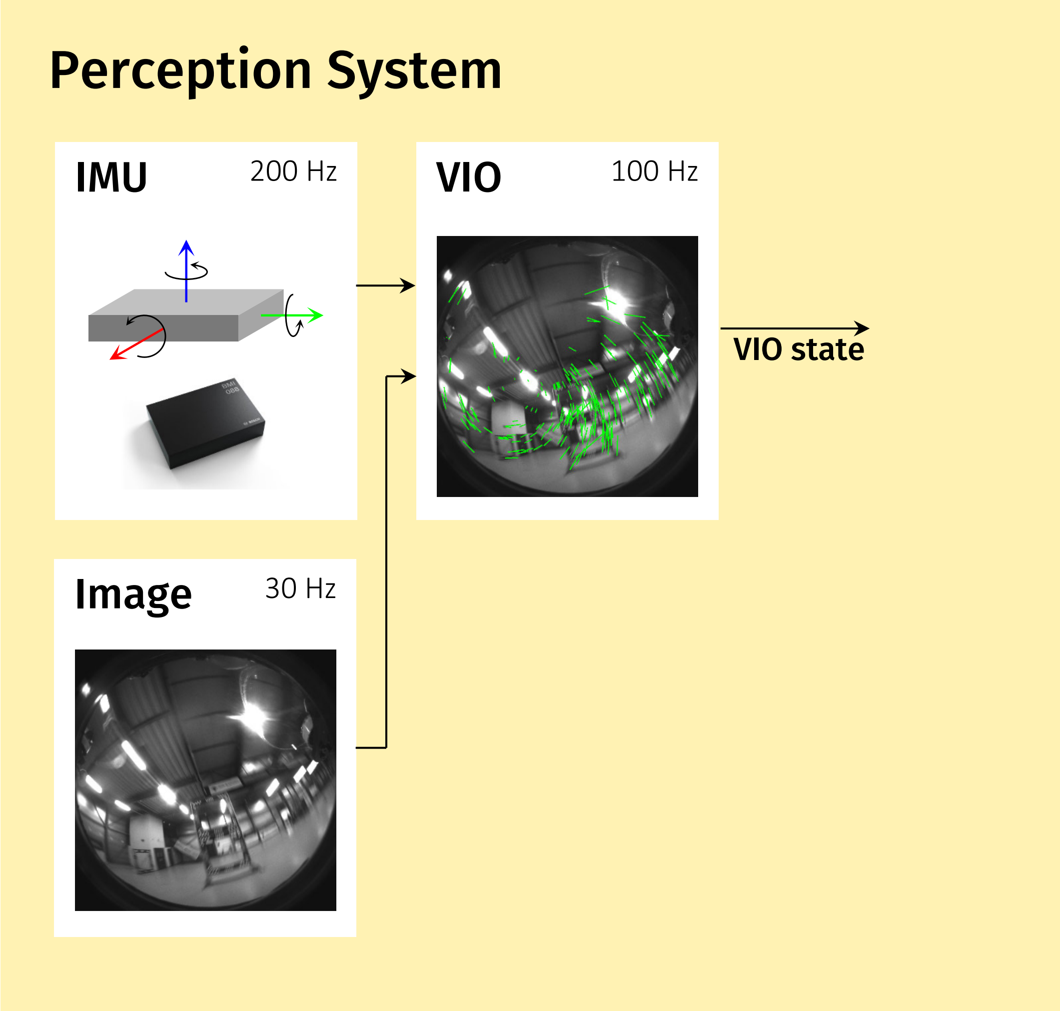
- Weight: 870g
- Thrust: 39N
- TWR: 4.5

Localization

VIO Performance

- VIO drift accumulates over time
- no robust feature tracking
- IMU forward integration

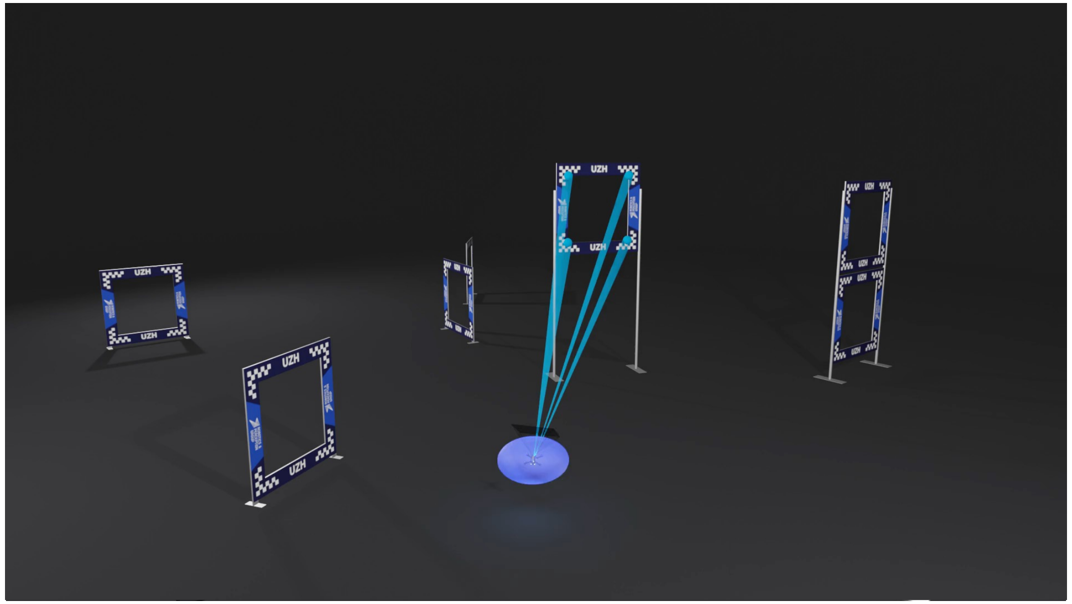


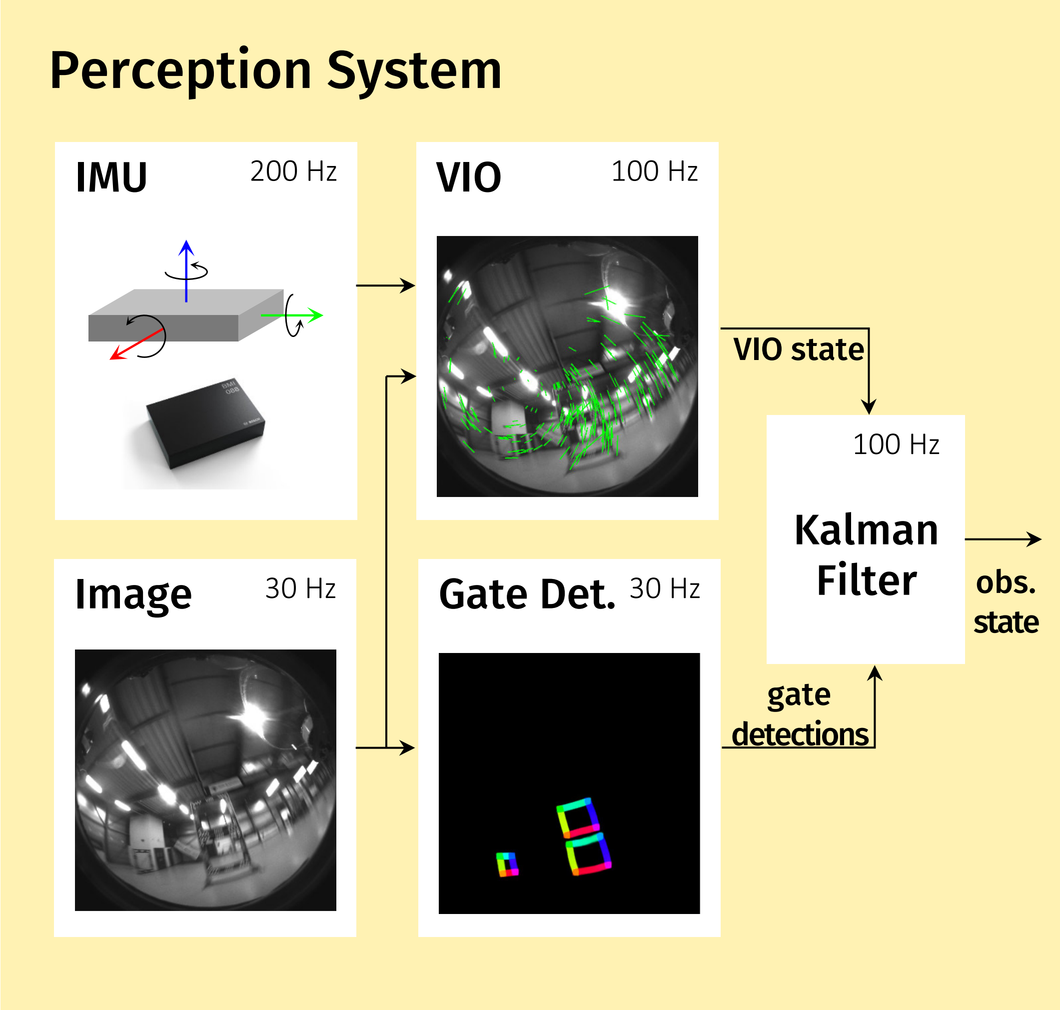


Localization

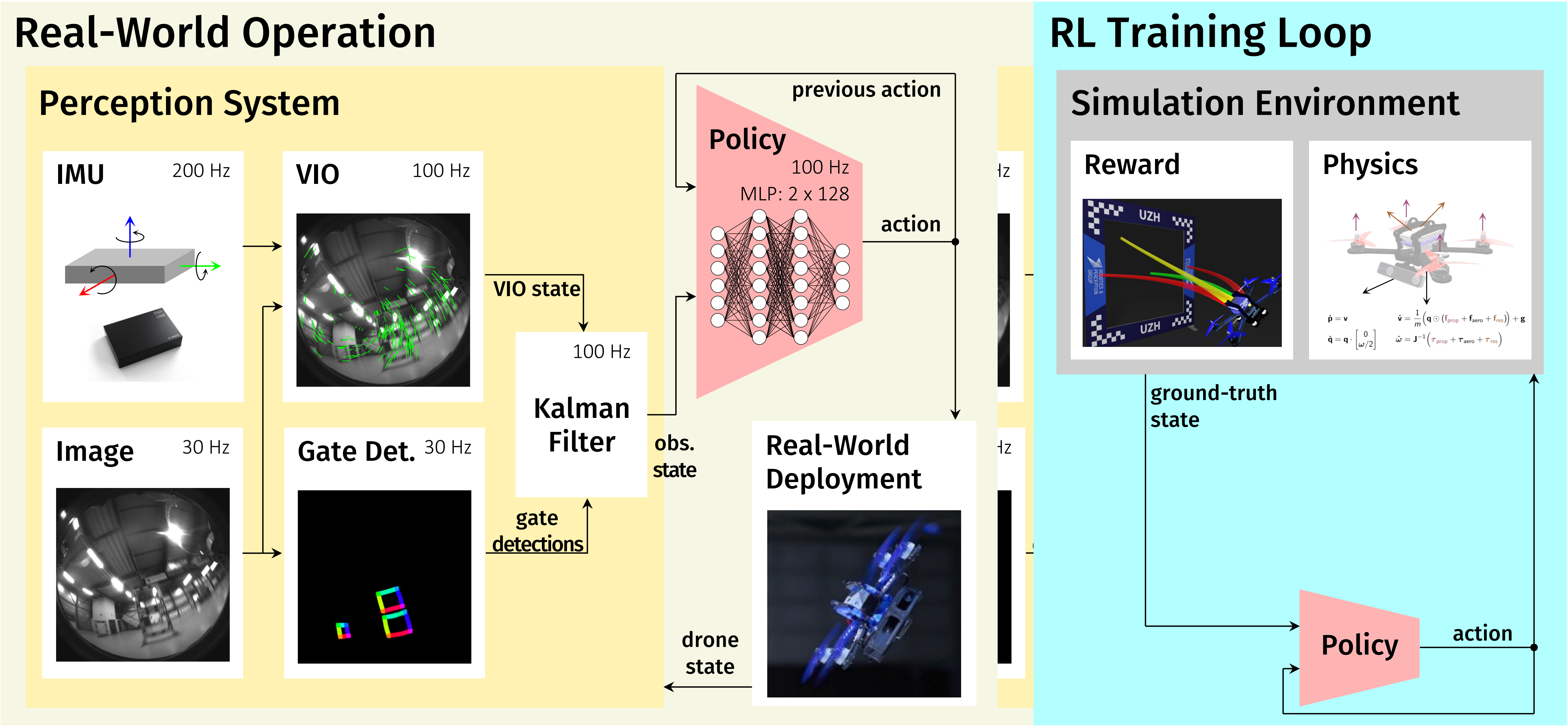
Gate Detections

- CNN-Unet detecting gates
- PnP for localization
- Kalman filter to fuse VIO+Gates





RL Policy Training



RL Policy Training

Reward

- progress reward

$$r_t^{prog} = \lambda_1 \left(d_{t-1}^{Gate} - d_t^{Gate}\right)$$

- perception reward

$$r_t^{perc} = \lambda_2 \exp(\lambda_3 \, \delta_{cam}^4)$$

- command reward

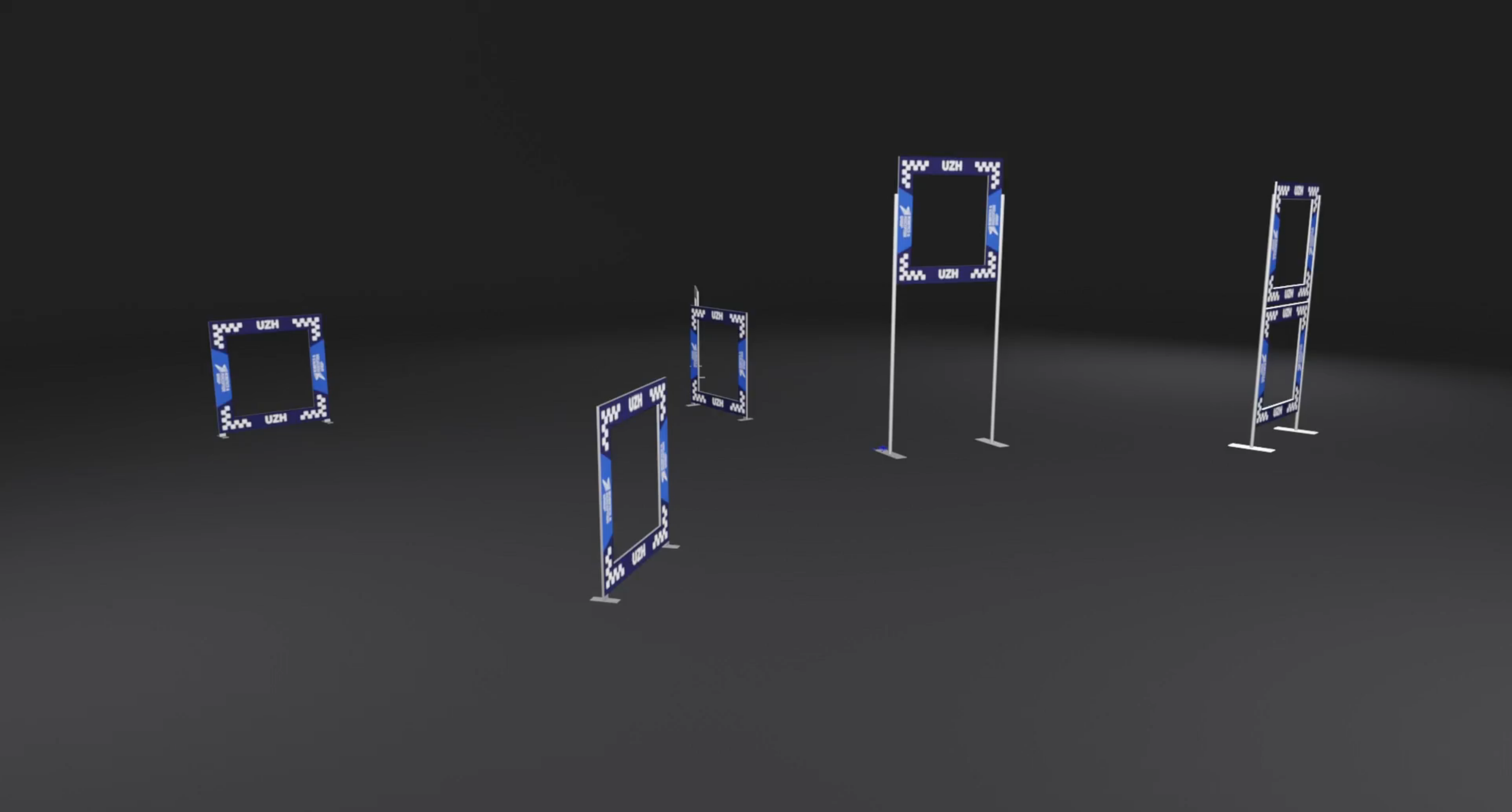
$$r_t^{cmd} = \lambda_4 |a_t^{\omega}| + \lambda_5 |a_t - a_{t-1}|^2$$

- crash penalty

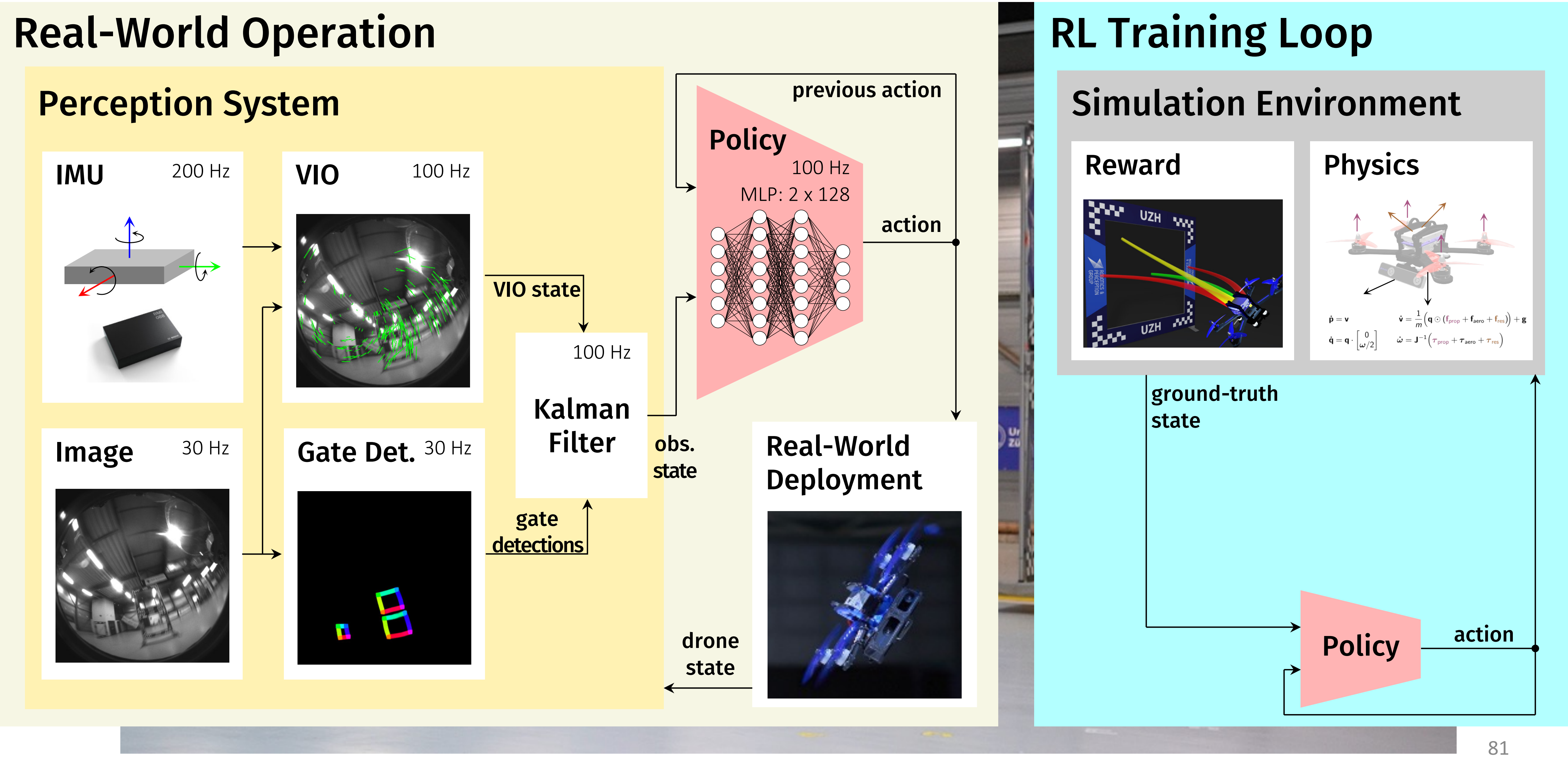
$$r_t^{crash} = \begin{cases} -5.0, & if\ crash \\ 0, & otherwise \end{cases}$$

Training Details

- training with PPO
  - 100M environment interactions
  - 50 min wall-time
  - 23 days sim-time
- value and policy network share architecture
- network:
  - 2 layer MLP
  - 128 nodes per layer
  - activation: LeakyReLU
  - optimizer: Adam


UZH

Is it good enough?

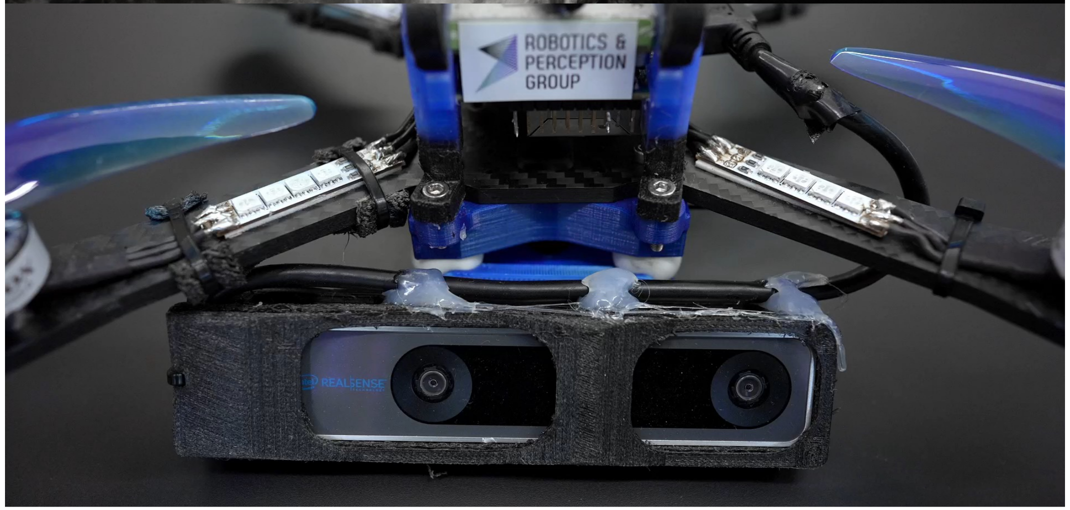


Residual Models

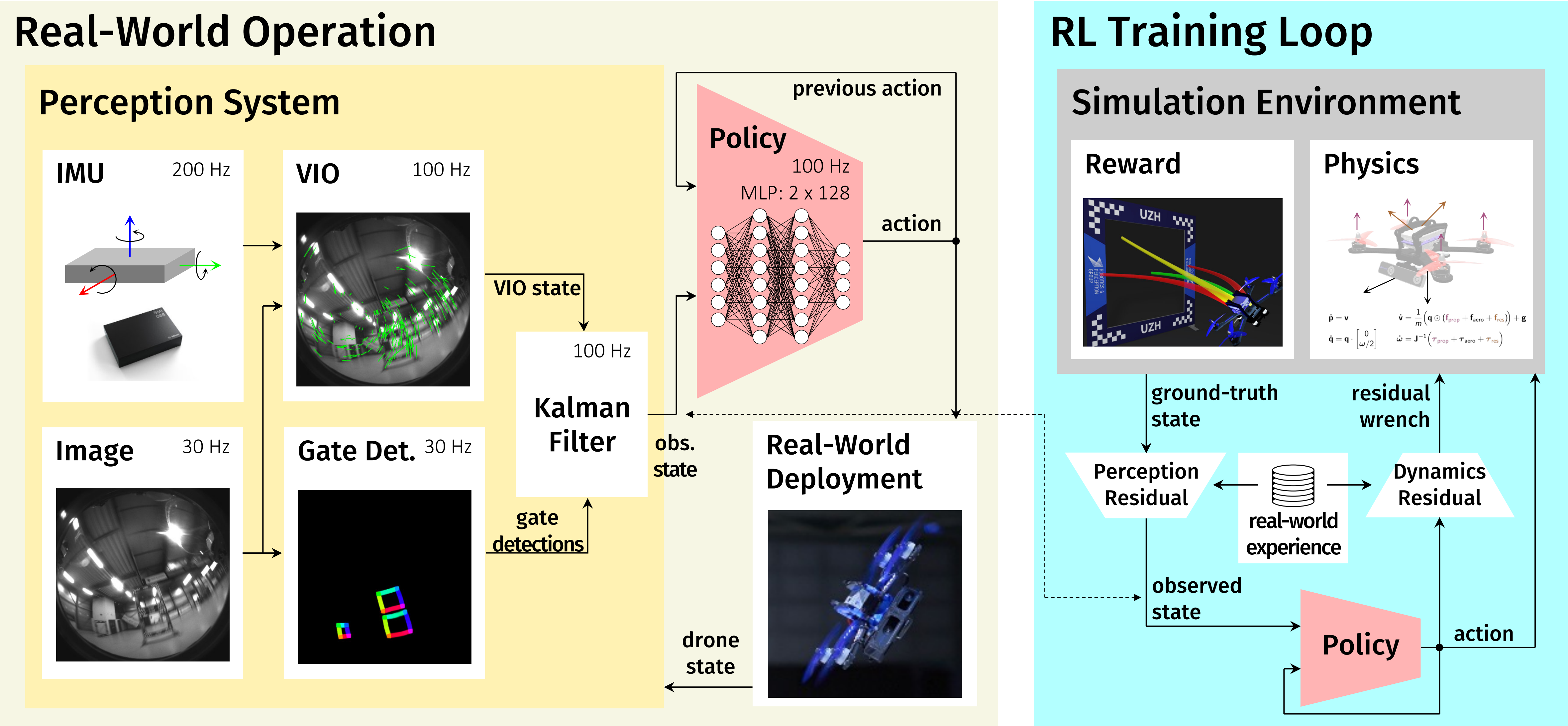
Unmodelled Effects

- Aerodynamic effects
  - turbulence & downwash
  - ground effect
- Mechanical effects
  - soft dampers to shield IMU from motor vibrations
  - camera moves w.r.t. drone body
- Perception effects
  - illumination & background changes

need policy-specific models



The Swift System



Head-to-Head Race Results

	Number of Races	Best time-to-finish	Wins	Losses	Win ratio
A. Vanover vs. Swift	9	17.956	4	5	0.44
T. Bitmatta vs. Swift	7	18.746	3	4	0.43
M. Schäpper vs. Swift	9	21.160	3	6	0.33
Swift vs. human pilots	25	17.465	15	10	0.60

A. Vanover

T. Bitmatta

M. Schäpper

Conclusions and Takeaways

- Autonomous **vision-based agile flight** as a new research topic (at least 10 years to solve it)

  Pushes the limit of existing algorithms in extreme situations

  Raises **fundamental problems** for robotics **research**

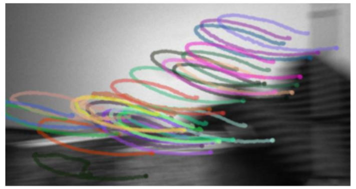
Come over for projects in DL!

- Visit our webpage for projects! http://rpg.ifi.uzh.ch/student_projects.php

Improving Event-Based Vision with Energy-Efficient Neural Networks - Available

Description: Event-based cameras, also known as neuromorphic vision sensors, capture visual information through asynchronous pixel-level brightness changes, offering high temporal resolution, low latency, and a wide dynamic range. These characteristics make them ideal for applications requiring rapid response times and efficient data processing. However, deploying deep learning models on resource-constrained devices remains challenging due to computational overhead and energy consumption. This project explores novel approaches to developing energy-efficient neural networks tailored for event-based vision tasks. By designing models that significantly reduce computational demands and memory footprint while maintaining high performance, we can make real-time processing on embedded hardware feasible. The focus will be on balancing training efficiency and model accuracy, minimizing energy consumption without sacrificing the quality of results.

Event Keypoint Extraction for Real-Time Pose Estimation - Available



Description: Neuromorphic cameras, known for their high dynamic range (HDR) capabilities, high-temporal resolution, and low power consumption, have opened up new possibilities in camera pose estimation, especially in fast-moving and challenging environments. This project aims to enhance camera pose estimation by developing a data-driven approach for keypoint extraction from event data, building on recent advancements in frame-based keypoint extraction. The project will also integrate a Visual Odometry (VO) pipeline to enable real-time feedback and tracking.

Hybrid Spiking-Deep Neural Network System for Efficient Event-Based Vision Processing - Available

Description: Event cameras are innovative sensors that capture changes in a scene dynamically, unlike standard cameras that capture images at fixed intervals. They detect pixel-level brightness changes, providing high temporal resolution and low latency. This results in efficient data processing and reduced power consumption, typically just 1 mW. Spiking Neural Networks (SNNs) process information as discrete events or spikes, mimicking the brain's neural activity. They differ from standard Neural Networks (NNs) that process information continuously. SNNs are highly efficient in power consumption and well-suited for event-driven data from event cameras. In collaboration with SynSense, this project aims to integrate the rapid processing capabilities of SNNs with the advanced analytic powers of deep neural networks. By distilling higher-level features from raw event data, we aim to significantly reduce the volume of events needing further processing by traditional NNs, improving data quality and transmission efficiency. System will be tested on computer vision tasks like object detection and tracking, gesture recognition, and high-speed motion estimation.

Learned Event Generation from Images - Available

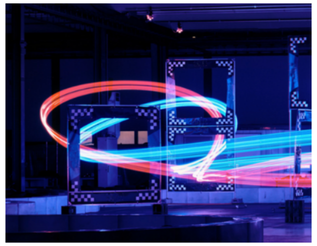
Description: Event cameras offer a unique approach to capturing scenes, detecting changes in light intensity rather than using fixed time intervals like traditional cameras. This project focuses on overcoming the scarcity of event-based datasets by generating synthetic event data from standard frame-based images. Using advanced deep learning techniques, the goal is to create high-quality synthetic events that closely resemble real-world data, helping to bridge the gap between simulated and actual event-based data.

Goal: In this project, you will apply cutting-edge deep learning models to generate artificial events from conventional image frames. You will gain a strong understanding of how event cameras work and how to produce realistic event data. Since the project involves exploring multiple state-of-the-art deep learning methods, a solid background in deep learning is essential. If you're interested, we would be happy to provide further details.

Check out our student projects!

- Visit our webpage: https://rpg.ifi.uzh.ch/student_projects.php

Fine-tuning Policies in the Real World with Reinforcement Learning - Available



Description: Training sub-optimal policies is relatively straightforward and provides a solid foundation for reinforcement learning (RL) agents. However, these policies cannot improve online in the real world, such as when racing drones with RL. Current methods fall short in enabling drones to adapt and optimize their performance during deployment. Imagine a drone equipped with an initial sub-optimal policy that can navigate a race course but not with maximum efficiency. As the drone races, it learns to optimize its maneuvers in real-time, becoming faster and more agile with each lap.

Goal: This project aims to explore online fine-tuning in the real world of sub-optimal policies using RL, allowing racing drones to improve continuously through real-world interactions.

Vision-based End-to-End Flight with Obstacle Avoidance - Available

Description: Recent progress in drone racing enables end-to-end vision-based drone racing, directly from images to control commands _without explicit state estimation_. In this project, we address the challenge of unforeseen obstacles and changes to the racing environment. The goal is to develop a control policy that can race through a predefined track but is robust to minor track layout changes and gate placement changes. Additionally, the policy should avoid obstacles that are placed on the racetrack, mimicking real-world applications where unforeseen obstacles can be present at any time. Requirements: - Machine learning experience (PyTorch) - Excellent programming skills in C++ and Python

Sim-to-real transfer of event-camera-based RL policies - Available

Description: This project aims to develop and evaluate drone navigation policies using event-camera inputs, focusing on the challenges of transferring these policies from simulated environments to the real world. Event cameras, known for their high temporal resolution and dynamic range, offer unique advantages over traditional frame-based cameras, particularly in high-speed and low-light conditions. However, the sim-to-real gap—differences between simulated environments and the real world—poses significant challenges for the direct application of learned policies. In this project we will look try to understand the sim-to-real gap for event cameras and how this gap influences downstream control tasks, such as flying in the dark, dynamic obstacle avoidance and, object catching. This would include learning representations for event data (ideally while reducing the sim-real domain gap) and training navigation policies using either reinforcement or imitation learning methods.

Learning Rapid UAV Exploration with Foundation Models - Available

Description: In this project, our objective is to efficiently explore unknown indoor environments using UAVs. Recent research has demonstrated significant success in integrating foundational models with robotic systems. Leveraging these foundational models, the drone will employ learned semantic relationships from large-world-scale data to actively explore and navigate through unknown environments. While most prior research has focused on ground-based robots, this project aims to investigate the potential of integrating foundational models with aerial robots to introduce more agility and flexibility. Applicants should have a solid understanding of mobile robot navigation, machine learning experience (PyTorch), and programming experience in C++ and Python.