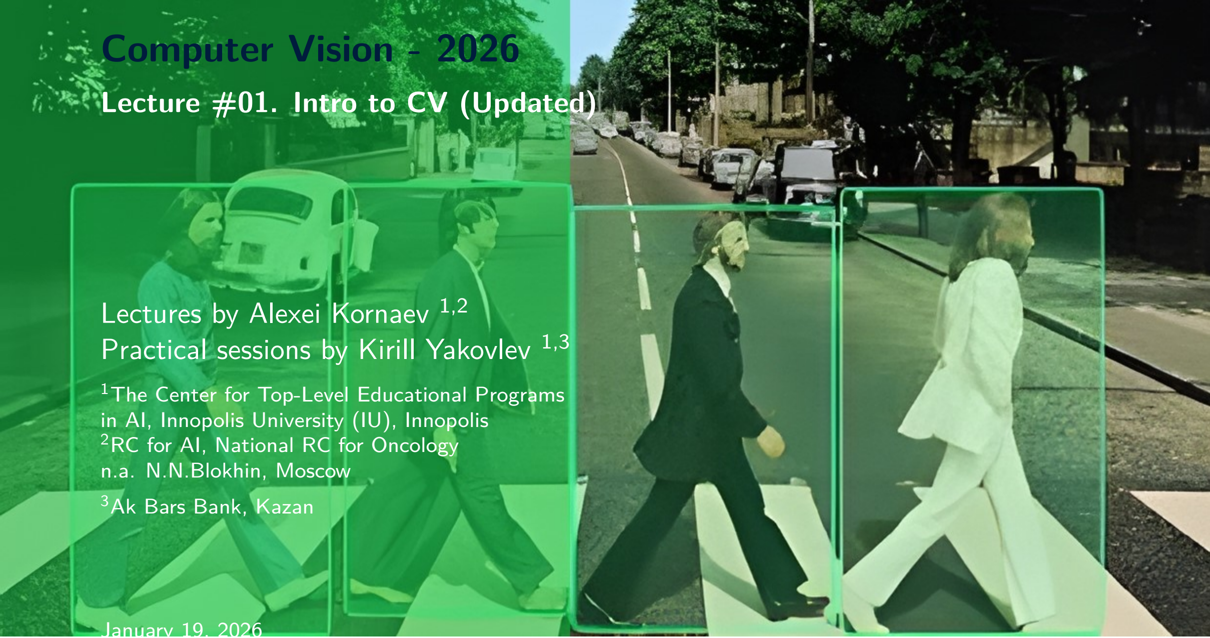


Computer Vision - 2026

Lecture #01. Intro to CV (Updated)

Lectures by Alexei Kornaev [1,2]

Practical sessions by Kirill Yakovlev [1,3]

[1]The Center for Top-Level Educational Programs in AI, Innopolis University (IU), Innopolis

[2]RC for AI, National RC for Oncology n.a. N.N.Blokhin, Moscow

[3]Ak Bars Bank, Kazan

January 19, 2026

Agenda

Section 1. Learning Outcomes and Course Map

Outcomes (Why this course)

This course is about **building robust CV systems** — not only training networks. We go from **pixels** and **classical CV** to **CNN/ViT**, then to **VLMs** and finally **Vision-Language-Action** models.

By the end you should be able to:

1. **Explain** key CV ideas (data, models, losses, metrics, failure modes)
2. **Build** strong baselines and improve them with modern open-source models
3. **Run experiments** reproducibly (logs, ablations, error analysis) and report results

Outcomes (3 levels)

Level 1 — Concepts you must understand

- Images as signals; distortions and dataset shift
- Representation learning: CNNs vs Transformers; inductive biases
- Core CV tasks: classification, detection, segmentation, tracking, depth, 3D
- What changed recently: **prompting**, **open-vocabulary**, **foundation models**

Level 2 — Practical skills

- Train / fine-tune models, use transfer learning, and evaluate correctly
- Use OpenCV for debugging and strong baselines
- Use Lightning + ClearML for reproducible experiments

Level 3 — Real-life / research skills

- Choose the right tool (classic CV vs DL vs VLM), justify the choice, and measure trade-offs
- Build a pipeline: perception $\rightarrow$ decision $\rightarrow$ (optionally) action
- Deploy a model and document limitations (latency, robustness, safety, privacy)

Course map (where Lecture #01 fits)

- **Lectures 1–3:** foundations + CNN/ViT (representations, training, evaluation)
- **Lectures 4–10:** generative models, segmentation/detection, depth/pose, video/3D, point clouds
- **Lectures 11–13: Vision-Language Models → Vision-Language-Action** (VLA/VLAM)

Main message: modern CV is increasingly **promptable** and **open-vocabulary**; the endgame is **perception + language + action**.

Course materials

The course content is available on Moodle and GitHub:

1. Moodle
2. Git project [CV-2026]

Rule: every assignment must include (1) baseline, (2) metrics, (3) failure analysis, (4) ClearML logs.

Section 2. Place and Role of CV Today

Where CV is used (fast reality check)

CV is a core component in:

- **Robotics:** perception for manipulation, navigation, autonomy (2D/3D + tracking)
- **Medicine:** radiology, pathology, endoscopy (segmentation, detection, triage)
- **Industry:** quality control, inspection, anomaly detection
- **Security & retail:** re-identification, counting, event detection
- **Consumer apps:** photo enhancement, search, AR filters

Key trend: the boundary between "CV" and "ML" is now **multimodal** (vision + language) [Radford et al., 2021; Alayrac, 2022].

CV: from closed-set to open-vocabulary

Classic pipeline: fixed classes, fixed labels, fixed evaluation.

Modern pipeline: promptable models that accept **text** as an interface:

- Open-vocabulary classification / retrieval (CLIP-style) [Radford et al., 2021]
- Language-guided detection / grounding (GroundingDINO-style) [Liu et al., 2024]
- Promptable segmentation (SAM-style, including video) [Kirillov et al., 2023; Ravi et al., 2024]

Why it matters for robotics: you cannot pre-enumerate all objects you will see in the world.

AI Index in Russia by MIPT (# 14, 2024)

INNOPOLIS UNIVERSITY

Place and role of CV (R. B. Weide slide)

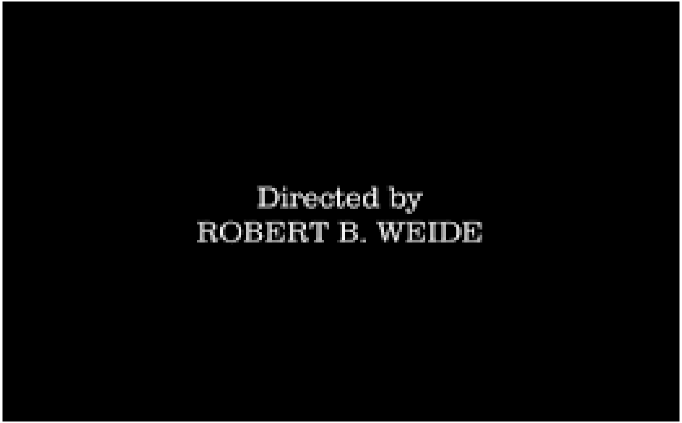


Figure: CV is a bridge between physical world and digital decision making.

Section 3. What is an Image and What Can Go Wrong

What is an image (engineering view)

Image = a tensor with structure:

- Grid of pixels: $H \times W$
- Channels: grayscale (1), RGB (3), multispectral ($C > 3$), depth, thermal, etc.
- Data type and scale: 8-bit, 16-bit, float; linear vs gamma-compressed

In code: image $\approx$ `torch.Tensor`$[C, H, W]$ (or `numpy`[H,W,C]).

Types of images you will meet in robotics/CV

- **RGB** (camera): appearance, texture, color
- **Grayscale**: simplified signal (often enough for geometry)
- **Depth** (stereo/ToF): geometry, scale, occlusions
- **Thermal/IR**: night vision, heat signatures
- **Medical** (CT/MRI/US): different physics, different artifacts

Important: "image" is not always visible-light photography.

Dataset shift: the default enemy

Models often fail not because of “bad architecture” but because the world changes:

- Lighting, viewpoint, motion blur, camera intrinsics, compression
- Background changes, rare objects, occlusion, clutter
- Domain shift: sim $\rightarrow$ real, lab $\rightarrow$ street, hospital A $\rightarrow$ hospital B

Lecture rule: always ask “what will be different at test time?” and design evaluation accordingly.

Section 4. Deterministic Models in CV (Why Classic CV Still Matters)

Why deterministic CV is still in the course

Classical CV is not "obsolete" — it is:

- A strong baseline for simple tasks (fast and interpretable)
- A debugging tool (to understand what the camera sees)
- A component inside modern pipelines (preprocessing, geometry, tracking primitives)

Examples you should be able to implement quickly:

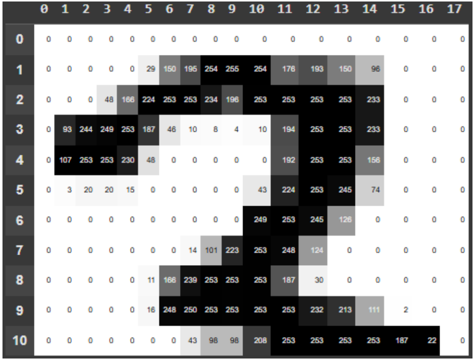
- Filtering / denoising, edges, corners, optical flow (conceptually)
- Homography / calibration basics (as a system component)

A minimal CV workflow (system view)

1. **Define the task:** what is input/output? what errors matter?
2. **Baseline:** simplest method (often classical CV + simple ML)
3. **Improve:** CNN/ViT / segmentation / tracking / VLM
4. **Evaluate:** metrics + stress tests + failure analysis
5. **Deploy:** latency, memory, monitoring, dataset drift

This course repeatedly loops through this workflow.

Hands-on: fastai as a reference workflow



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	29	150	195	254	255	254	176	193	150	96	0	0	0
2	0	0	0	48	166	224	253	253	234	196	253	253	253	253	233	0	0	0
3	0	93	244	249	253	187	46	10	8	4	10	194	253	253	233	0	0	0
4	0	107	253	253	230	48	0	0	0	0	0	192	253	253	156	0	0	0
5	0	3	20	20	15	0	0	0	0	0	43	224	253	245	74	0	0	0
6	0	0	0	0	0	0	0	0	0	0	249	253	245	126	0	0	0	0
7	0	0	0	0	0	0	0	14	101	223	253	248	124	0	0	0	0	0
8	0	0	0	0	0	11	166	239	253	253	253	187	30	0	0	0	0	0
9	0	0	0	0	0	16	248	250	253	253	253	253	232	213	111	2	0	0
10	0	0	0	0	0	0	0	43	98	98	208	253	253	253	253	187	22	0

Figure: fastbook: a clean practical reference (e.g., MNIST basics) [Howard and Gugger, 2020]

INNOPOLIS UNIVERSITY

Section 5. How We Code and How We Grade (Reproducibility)

Support of coding: tools and expectations

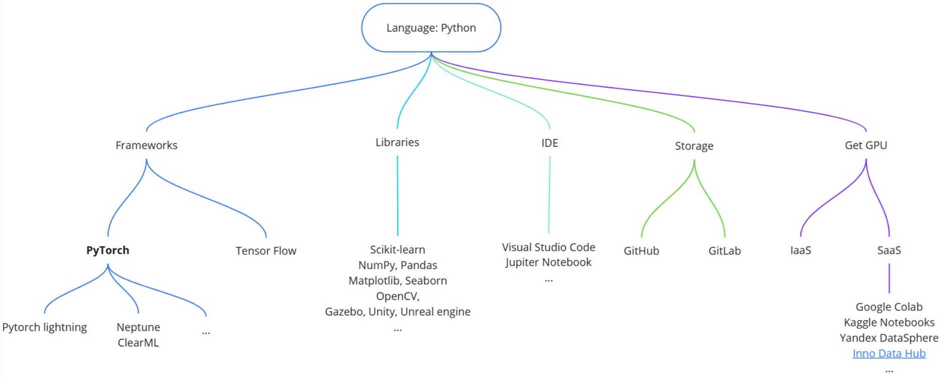


Figure: From idea to reproducible experiment: code, logs, and reports.

Support of coding (practical requirements)

Two frameworks should be met:

1. Review the Lightning framework (Core API is enough to start)
2. Review ClearML basics: experiments, datasets, artifacts

Minimal course standard for every homework:

- fixed seeds (where possible), saved config, logged metrics, and at least one ablation
- at least 5–10 failure cases shown (images with explanations)

What you will stop doing after this course

- "I trained a model and got accuracy $= 0.93$" (without specifying dataset split, metrics, and failure cases)
- Comparing methods without the same evaluation protocol
- Treating training curves as proof of generalization

Instead you will do: baseline + protocol + stress tests + report.

Section 6. Conclusion

Conclusion

Today you should remember:

- An image is a tensor, but the world adds noise, artifacts, and shift.
- Classical CV is still useful: baselines + debugging + geometry.
- Modern CV trends: **promptable** and **open-vocabulary** models, then **VLM** $\rightarrow$ **VLA/VLAM**.
- Reproducibility and evaluation are part of the model.

Next lecture: CNNs as inductive bias + modern training recipes.

Bibliography

Jean-Baptiste et al. Alayrac. Flamingo: a visual language model for few-shot learning. **NeurIPS**, 2022.

CV-2026. The course repository. https://github.com/Mechanics-Mechatronics-and-Robotics/CV-2026, 2026. Accessed: January 19, 2026.

J. Howard and S. Gugger. **Deep Learning for Coders with Fastai and Pytorch: AI Applications Without a PhD**. O'Reilly Media, Incorporated, 2020. ISBN 9781492045526. URL https://books.google.no/books?id=xd6LxgEACAAJ.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In **Proceedings of the IEEE/CVF International Conference on Computer Vision**, pages 4015–4026, 2023.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding dino: Marrying dino with grounded pre-training for open-set object detection, 2024. URL https://arxiv.org/abs/2303.05499.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman RÃ¤dle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr DollÃ¡r, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos, 2024. URL https://arxiv.org/abs/2408.00714.