

Computer Vision

Introduction, Structure and Organization of the Course



Antonio Carta

2026-09-01

University of Pisa

antonio.cart@unipi.it

What is Computer Vision?

“What does it mean, to see? The plain man’s answer (and Aristotle’s, too). would be, to know what is where by looking.”—David Marr

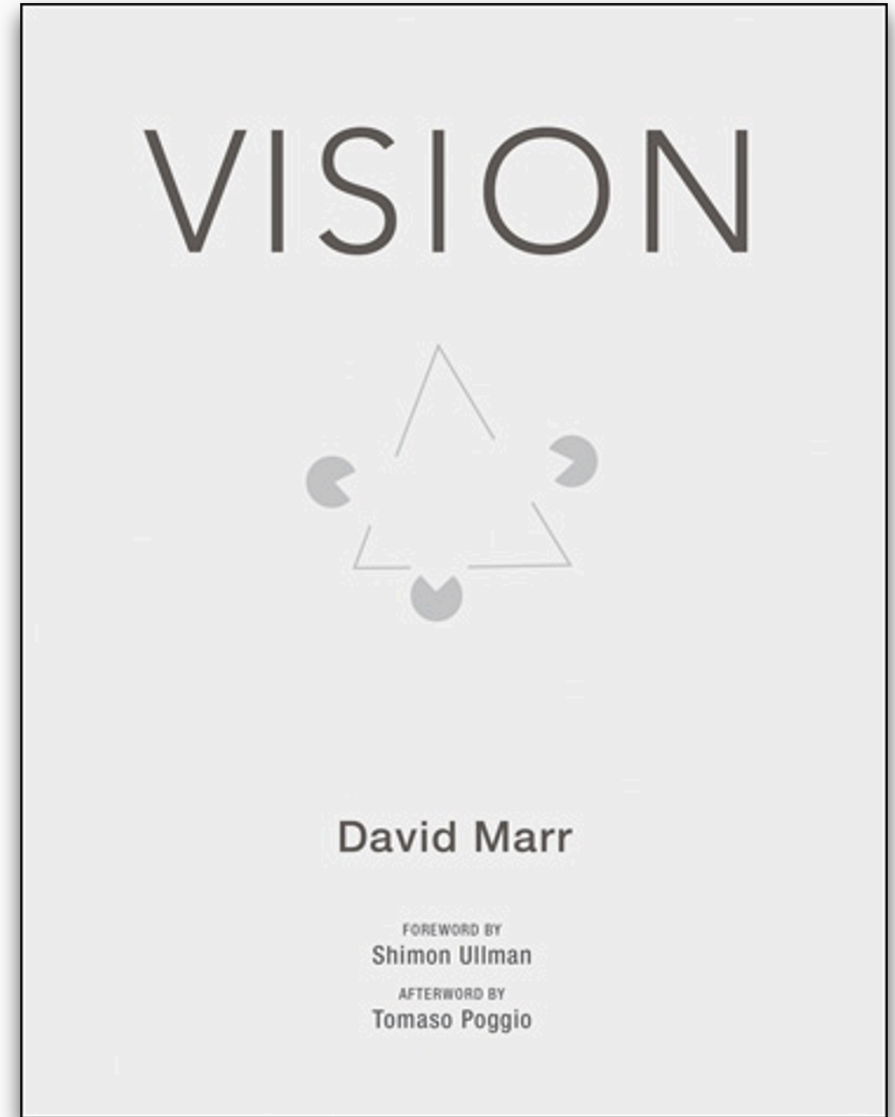


Figure 1: source: visionbook

- **Computer vision**: the study of how to make machines extract meaning, structure, and decisions from visual data (images, video, depth, multi-view)
- Spans a huge range of tasks that extract 2D and 3D structure from images
- Related fields: image processing (signals in/out), graphics (synthesis), robotics (action), machine learning (the modern toolbox)

The MIT Summer Vision Project

- <https://dspace.mit.edu/entities/publication/3f820aef-49d5-4d97-9faa-2ac178d4da00>
- in 1966, a “Seymour A. Papert”, proposed computer vision as a “summer project for students”
- **The final goal is OBJECT IDENTIFICATION which will actually name objects by matching them with a vocabulary of known objects.**

60 years later we are still trying to solve this summer project...

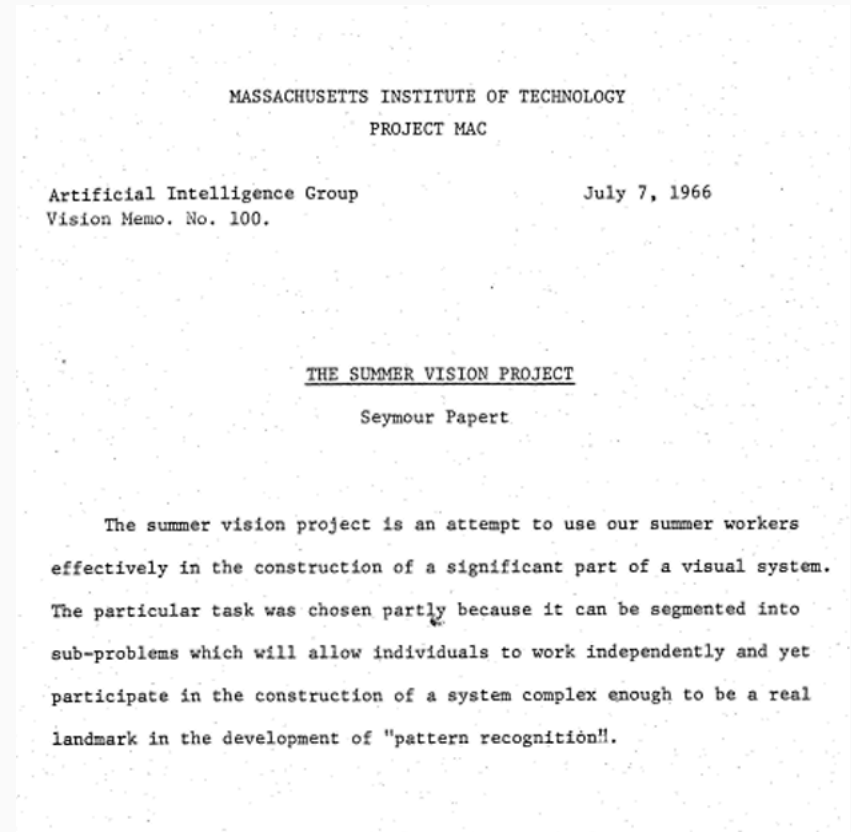


Figure 2:

- **Low-level**: denoising, edge detection, optical flow



Figure 3: source: ikomia.ai



Figure 4: source: ikomia.ai

- **Mid-level**: segmentation, depth, correspondence, 3D reconstruction



Figure 5: source: <https://www.youtube.com/watch?v=9vjyMCE-IbI>

- **High-level**: recognition, detection, captioning, generation, vision-language reasoning

draw an image of a rock band on a stage. Guitarist on the left, bassist on the right, singer front and center, drummer in the background.

Ha lavorato per 1min 9s >

Done — [download the rock band image.](#)



Figure 6: source: generated with ChatGPT

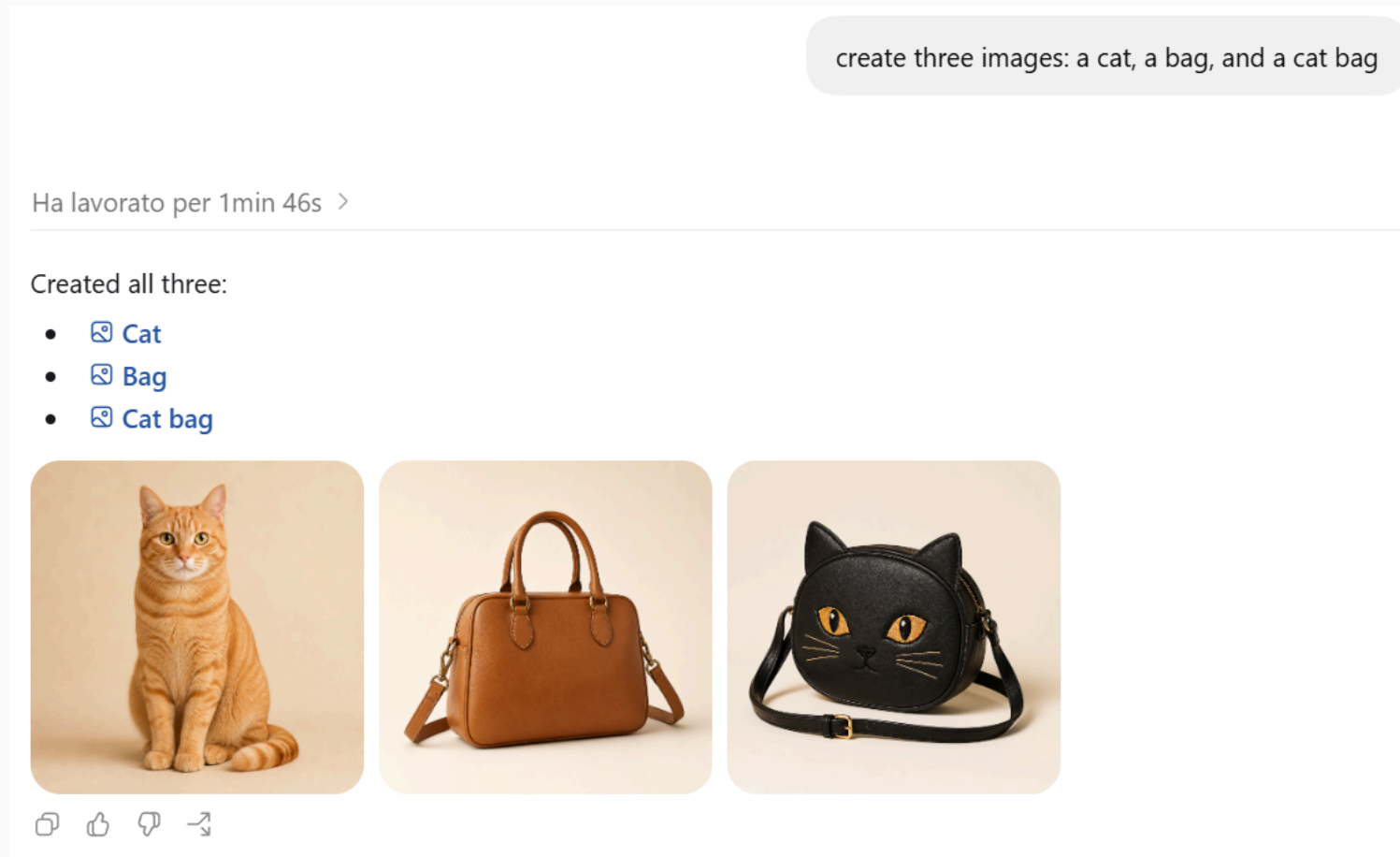


Figure 7: source: generated with ChatGPT

- **Phones**: face unlock, photo search, filters
- **Cars**: lane detection, pedestrian detection, depth estimation, SLAM for self-driving
- **Medical imaging**: tumor segmentation, registration across scans, diagnosis support
- **AR/VR**: 6-DoF tracking, scene reconstruction, hand/body pose estimation

Common system: raw pixels in, structured information out, often under tight latency/accuracy constraints

- **Graphics (forward model):** scene description $\rightarrow$ image
 - Given geometry, materials, lighting, camera, render an image
 - Well-posed: same input always produces the same output
- **Vision (inverse problem):** image $\rightarrow$ scene description
 - Given pixels, recover geometry, materials, lighting, camera, semantics
 - Ill-posed: many different scenes can produce the same image

Vision is graphics “run backwards”, but inversion is the hard part

- **Ill-posed problem**: a problem that lacks a unique, stable solution from the given data alone
- Classic example: a single 2D image is consistent with **infinitely many** 3D scenes
 - Same silhouette -> different shapes (depth ambiguity)
 - Same shading pattern -> different combinations of shape, material, and lighting
- vision algorithms need **priors** to pick a plausible answer among many possible ones

Why Vision is Hard: Illumination Changes

- The same object/scene can look drastically different under different lighting
 - Direction of light changes shading and shadows
 - Color temperature changes the recorded colors (white balance)
 - Specular highlights move with viewpoint and light position

A robust vision system must separate “what changed in the scene” from “what changed because of the light”



Figure 8: Illumination changes drastically alter pixel values for the same object.

Why Vision is Hard: Viewpoint/Pose Changes

- The same object viewed from different angles produces very different 2D projections
 - Self-occlusion (parts of the object hide other parts)
 - Foreshortening (shapes appear compressed)
 - Apparent size and aspect ratio change with distance and angle
- Recognition and matching must be (at least partially) invariant to viewpoint

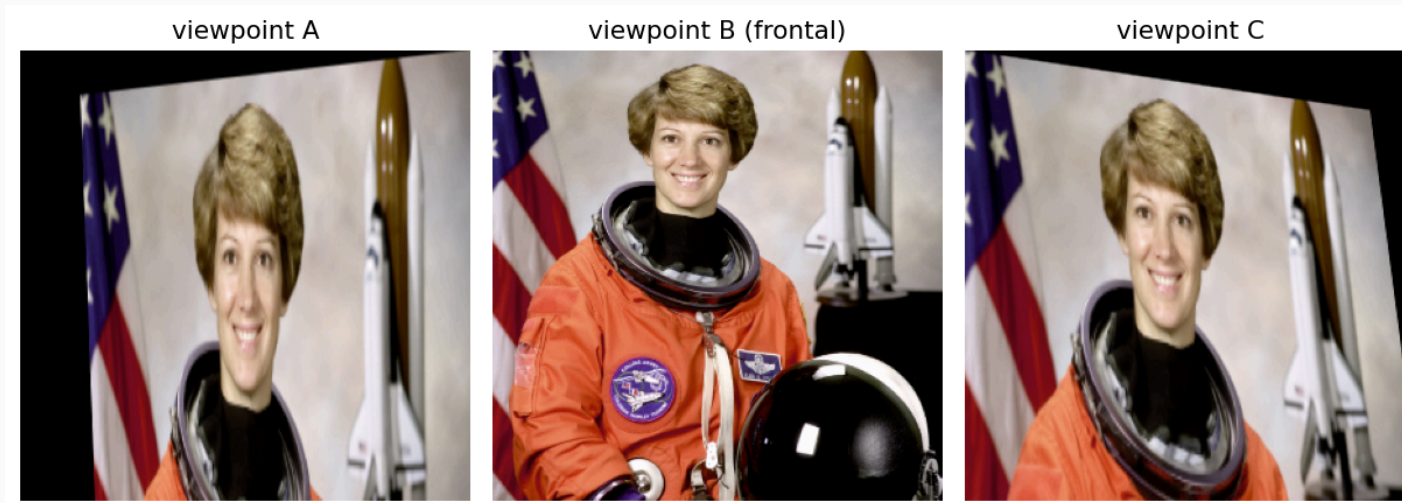


Figure 9: The same 3D object projects to very different 2D shapes from different viewpoints.

Why Vision is Hard: Occlusion

- Objects in the real world occlude each other and themselves
- Occlusion creates missing data: parts of the scene are simply not observed in a given image
- Algorithms must reason about what's hidden, often using context, priors, or multiple views



Figure 10: occlusion

Why Vision is Hard: Scale and Intra-Class Variation

- **Scale variation**: the same object can appear at vastly different sizes depending on distance to camera
- **Intra-class variation**: objects of the same category (e.g. “bird”) vary enormously in shape, color, texture, pose
- A useful representation must generalize across both axes simultaneously
- Solution: multi-scale processing and learned, invariant representations



Figure 11: source: visionbook.mit.edu

- Humans solve vision tasks effortlessly in milliseconds
 - rely on an enormous amount of implicit prior knowledge and computation (visual cortex, decades of “training data”)
- Modern deep learning systems, trained on large datasets, now match or exceed humans on many narrow vision tasks
- **Note:** Failure modes of ML (adversarial examples, distribution shift) differs from humans

- both learned and classic systems suffer from limitations
- learned systems may not generalize to novel objects and scenes
- Example: what happens when a classifier sees an unknown object? what happens when it sees a known object in a novel context?
- learned system usually are more robust to drastic changes in viewpoints/scenes/etc..
 - limitations of classic system are easier to understand, while DNN are more “blackbox” and it’s harder to predict where they will fail
 - it’s fundamental to try to understand when a technique works well and when it doesn’t. Will my model work with a low-resolution camera? will it work with images taken with bad lighting conditions?

Brief History of Computer Vision: Classical Era to Deep Learning

- **1960s-1980s**: early work on edge detection, blocks-world scene understanding, optical flow, stereo
- **1990s-2000s**: handcrafted features and geometry: SIFT, Harris corners, RANSAC, structure-from-motion, well-developed multi-view geometry
- **2010s**: deep learning revolution, CNNs dominate recognition (ImageNet), then detection, segmentation
- **2017-present**: transformers/ViTs, self-supervised learning, foundation models, vision-language models

Human Perception

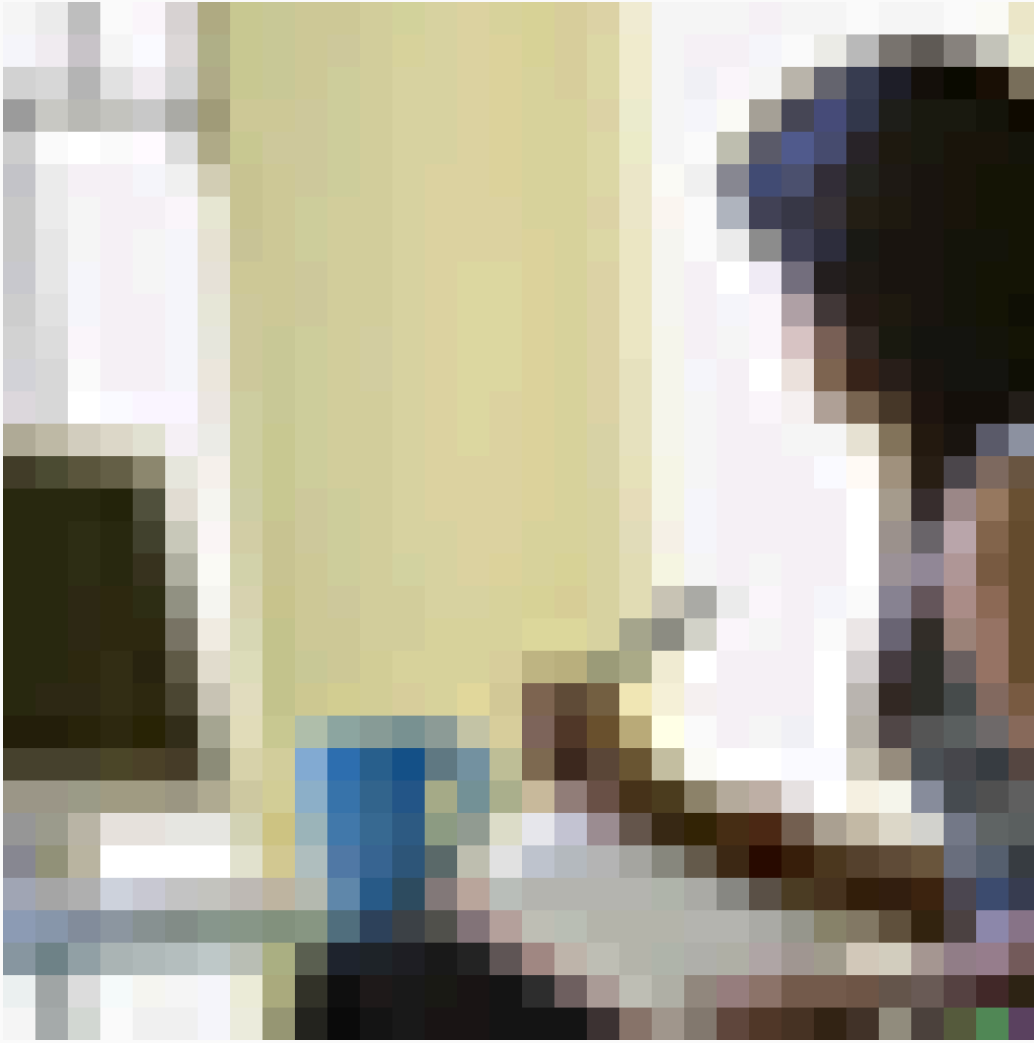


Figure 12: source: visionbook



Figure 13: source: visionbook

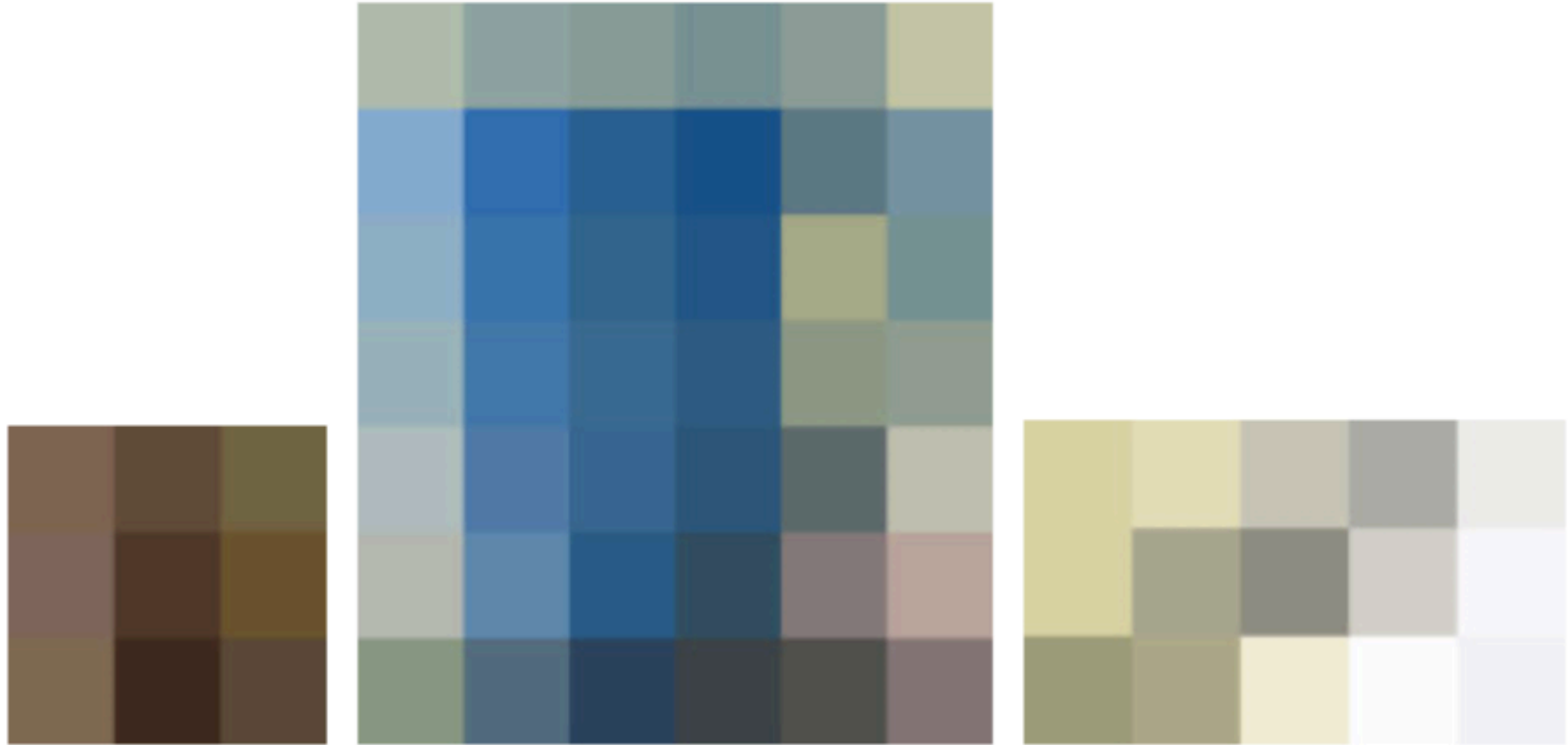


Figure 14: source: visionbook



Figure 15: source: visionbook



Figure 16: source: visionbook

Blind spot test

- human visual perception is a result of heavy processing from the brain
- Example: we have a blind spot in our eye that is always interpolated

Blind spot test [\[edit \]](#)

Demonstration of the blind spot

R

L

Instructions: Close one eye and focus the other on the appropriate letter (**R** for right or **L** for left). Place your eye a distance from the screen approximately equal to three times the distance between the **R** and the **L**. Move your eye towards or away from the screen until you notice the other letter disappear. For example, close your right eye, look at the "L" with your left eye, and the "R" will disappear.

Figure 17: source: [https://en.wikipedia.org/wiki/Blind_spot_\(vision\)](https://en.wikipedia.org/wiki/Blind_spot_(vision))

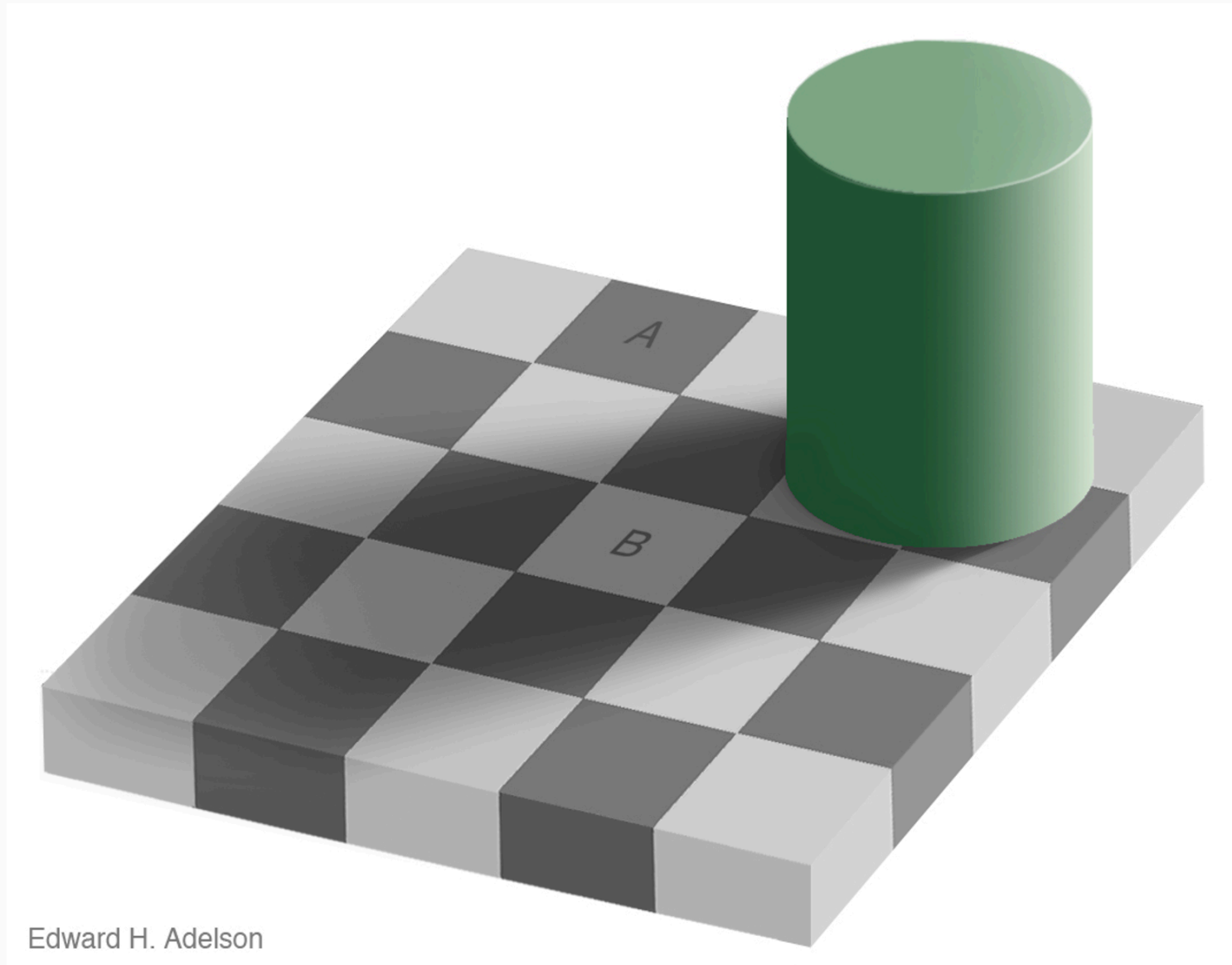


Figure 18: source: visionbook

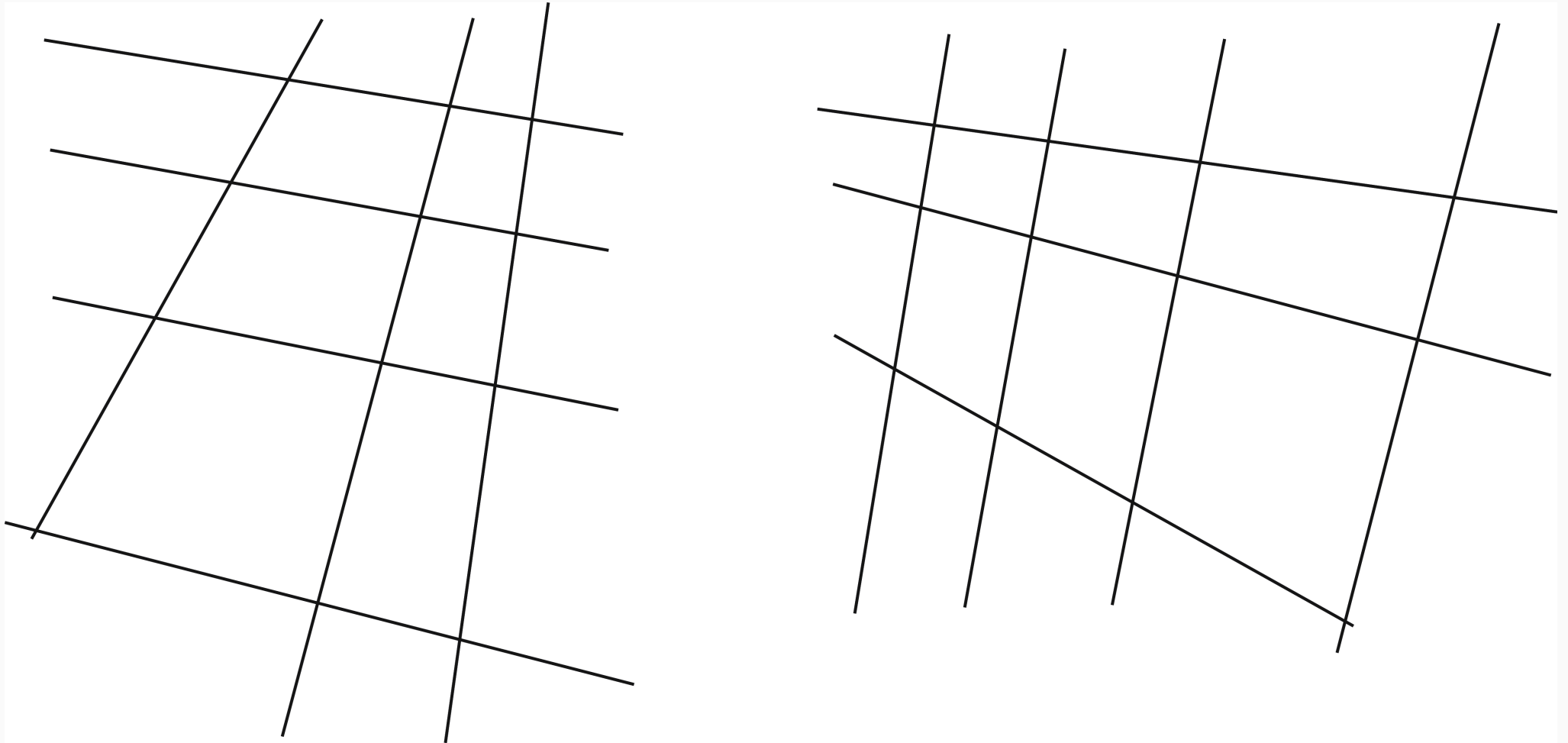


Figure 19: source: visionbook



Figure 20: source: visionbook

- human perception automatically uses contextual information and discounts nuisances (e.g. lighting effects)
- to build robust systems we need similar priors
- learned systems are the main solution used to build such priors

Unifying Themes in Computer Vision

- **Correspondence**: finding “the same point” across images
- **Geometric constraints**: cameras, multi-view geometry
- **Representation learning**: features and embeddings
- **Temporal consistency**: video, motion

- **Correspondence**: given two (or more) images of the same scene, find which pixel/region in one image corresponds to which pixel/region in the other
- Used in:
 - Panorama stitching (align overlapping photos)
 - Stereo and depth (match left/right views)
 - Tracking (match an object across video frames)
 - Structure-from-motion / SLAM (match across many views to recover 3D)
- Classical approach: detect distinctive keypoints, describe them, match by similarity
- Modern approach: learned dense or sparse matchers

Correspondence Example: Matching Points Between Two Photos

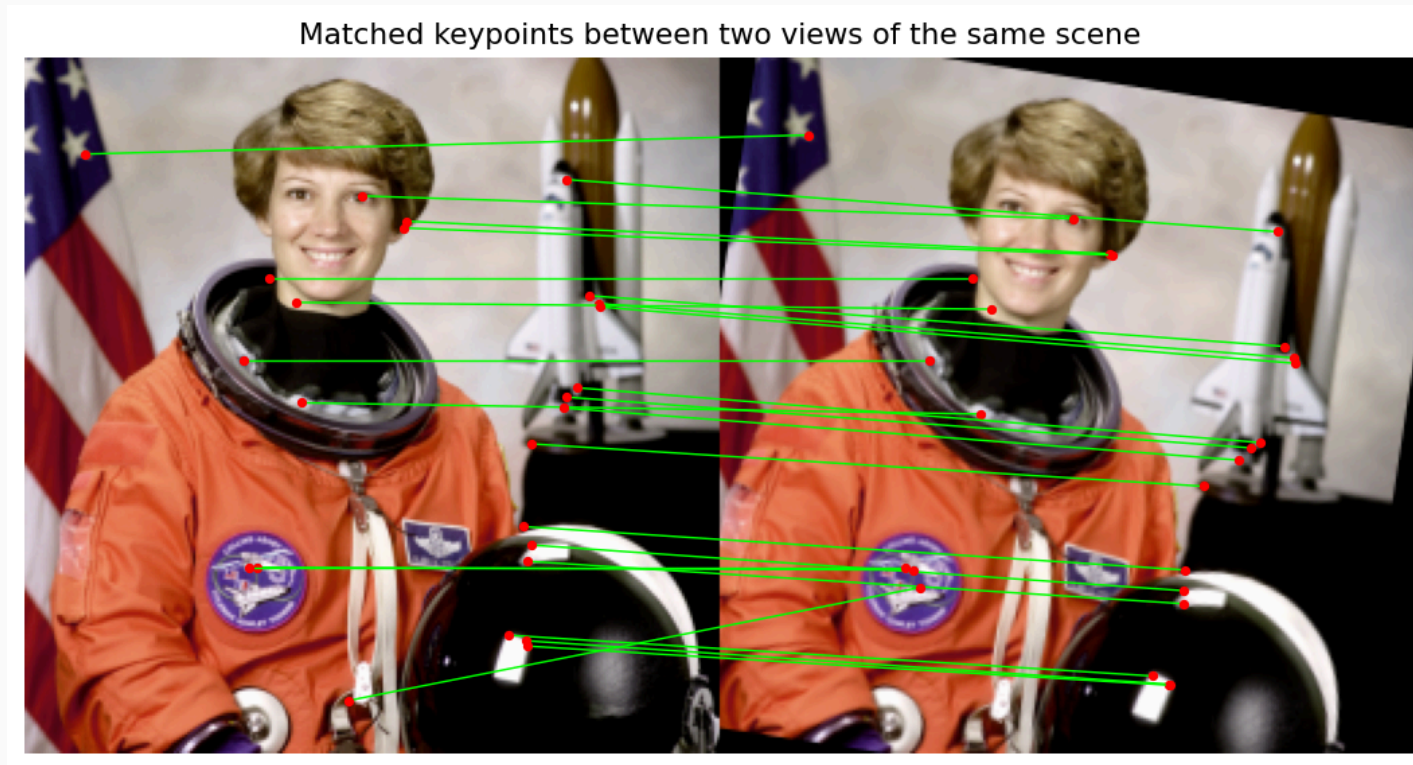


Figure 21: Corresponding keypoints detected and matched between two overlapping photos.

Geometric Constraints

- Cameras follow a **geometric model** (perspective projection)
- Perspective imposes hard constraints on how points can move between views:
 - A 3D point's projections in two images are related by the camera geometry (epipolar constraint)
 - A planar scene's projections are related by a single matrix (homography)
- These constraints let us go from 2D correspondences to 3D structure, and let us **reject incorrect matches** (geometric verification)

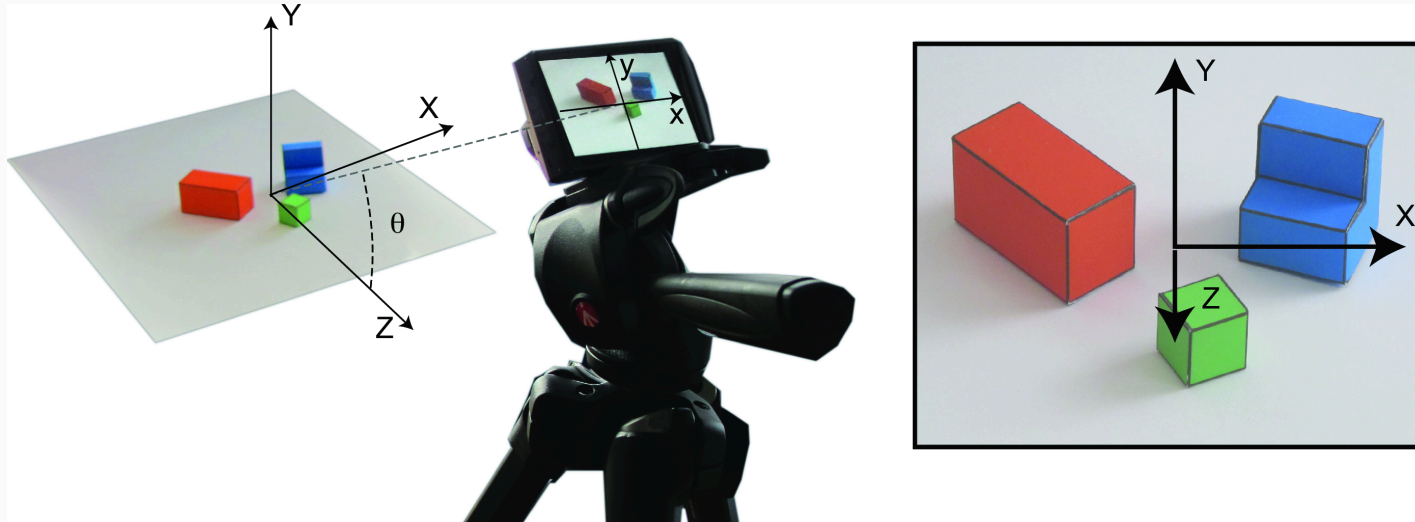


Figure 22: source: visionbook.mit.edu

- Given the same 3D point seen in two calibrated cameras, its two 2D projections plus the cameras' positions determine the 3D point uniquely (up to noise)
- This is **triangulation**: the geometric inverse of projection, made well-posed by using two views instead of one
- Generalizes to many views: structure-from-motion, stereo depth

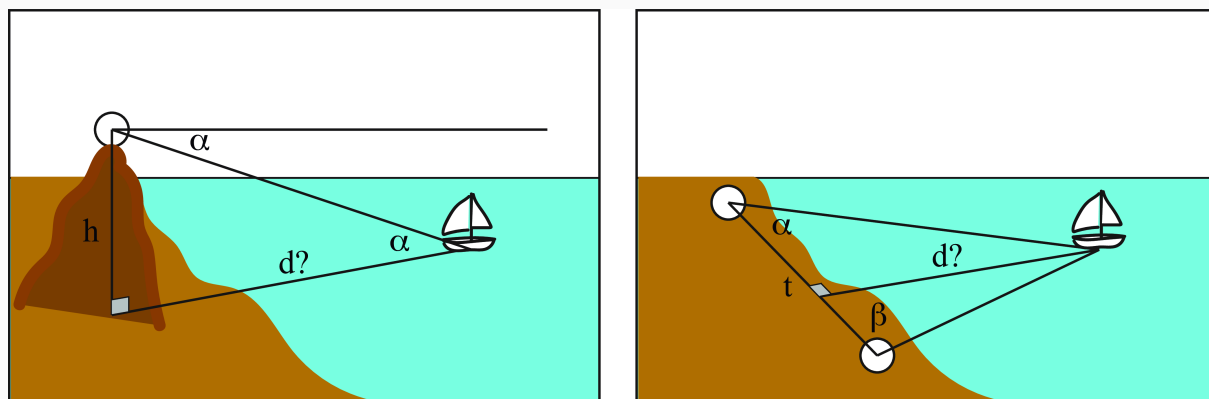
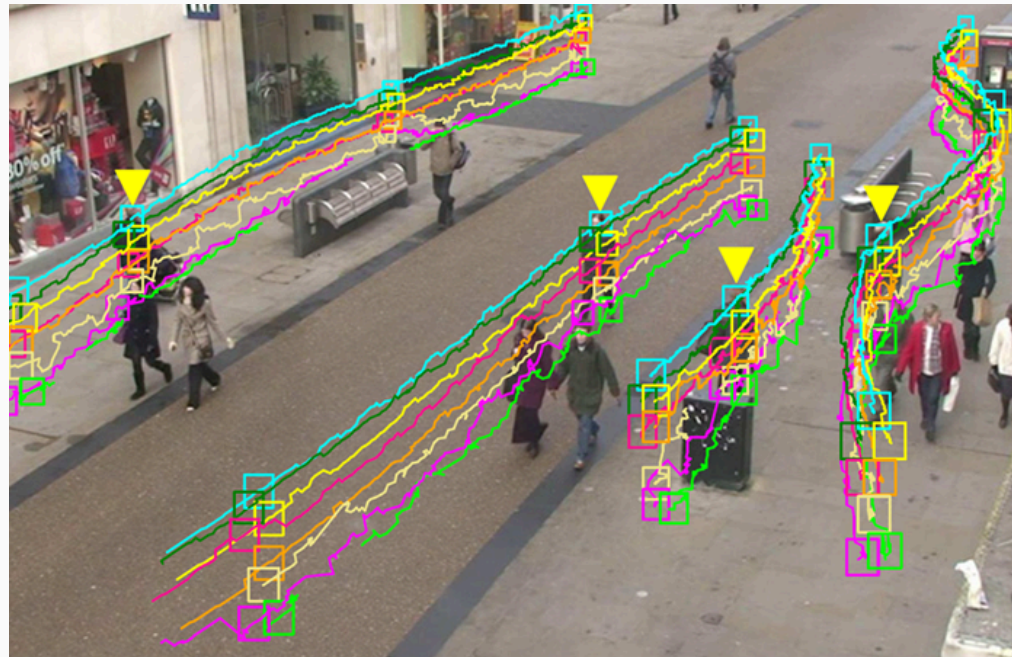


Figure 23: Two camera rays from corresponding image points intersect at the 3D point (triangulation).

- A **representation** (or feature/embedding) is a transformation of raw pixels into a vector that's more useful for a downstream task
- Good representations are:
 - **Invariant** to nuisance factors (lighting, viewpoint, scale) that shouldn't change the answer
 - **Discriminative** for factors that should (object identity, category)
- Classical: hand-design the transformation (e.g. SIFT: gradient histograms designed to be rotation/scale invariant)
- Modern: learn the transformation from data (CNN/ViT features, contrastive/self-supervised embeddings)
- Both solve the same underlying problem (a compact, invariant description of local appearance), with different design philosophies

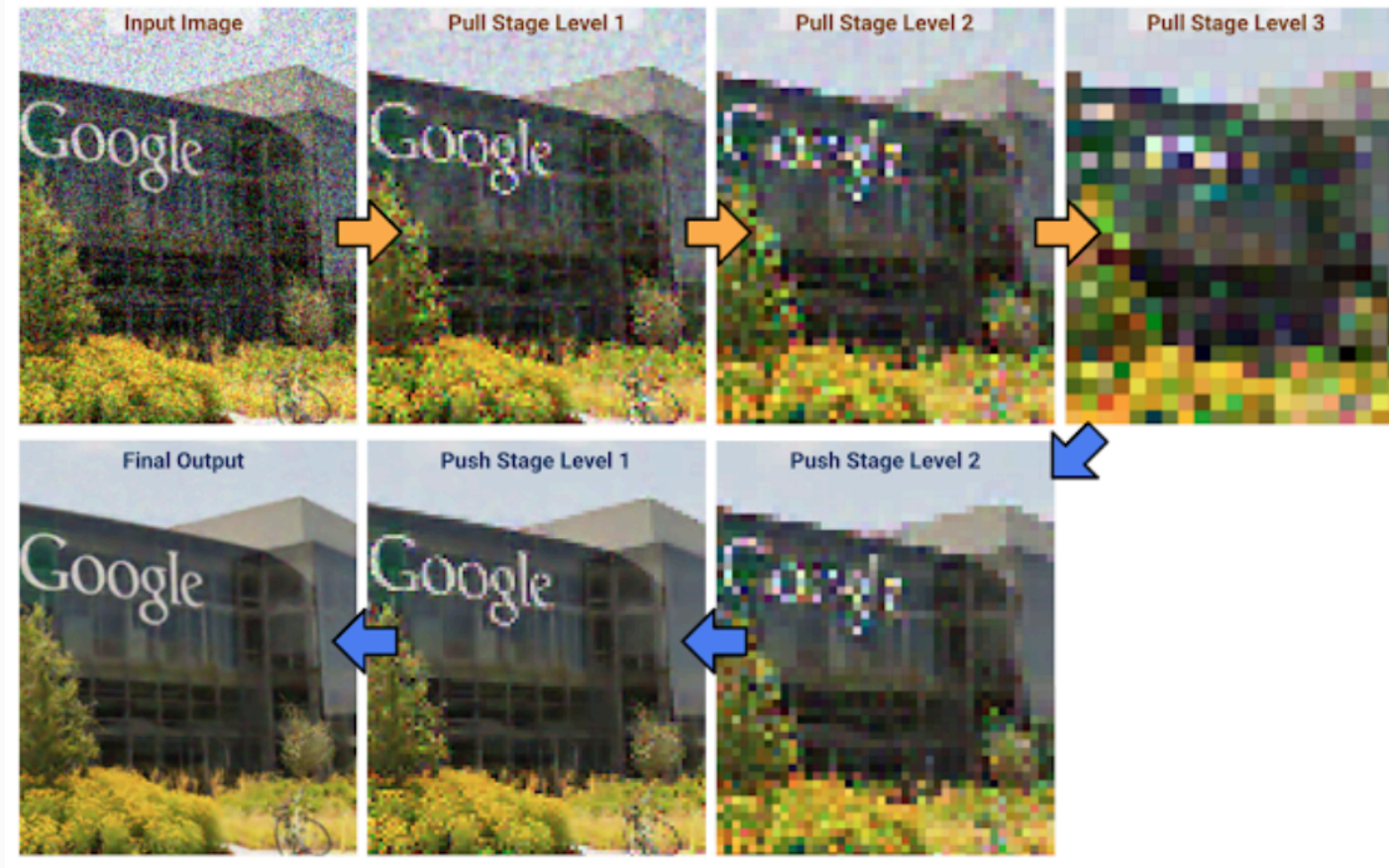
Temporal Consistency

- In videos, consecutive frames are highly correlated, objects and the camera move smoothly
- Temporal consistency: outputs are coherent across time (an object's identity, position, and label shouldn't flicker arbitrarily frame to frame)
- Connects correspondence to time: matching points/objects **across frames** rather than static viewpoints

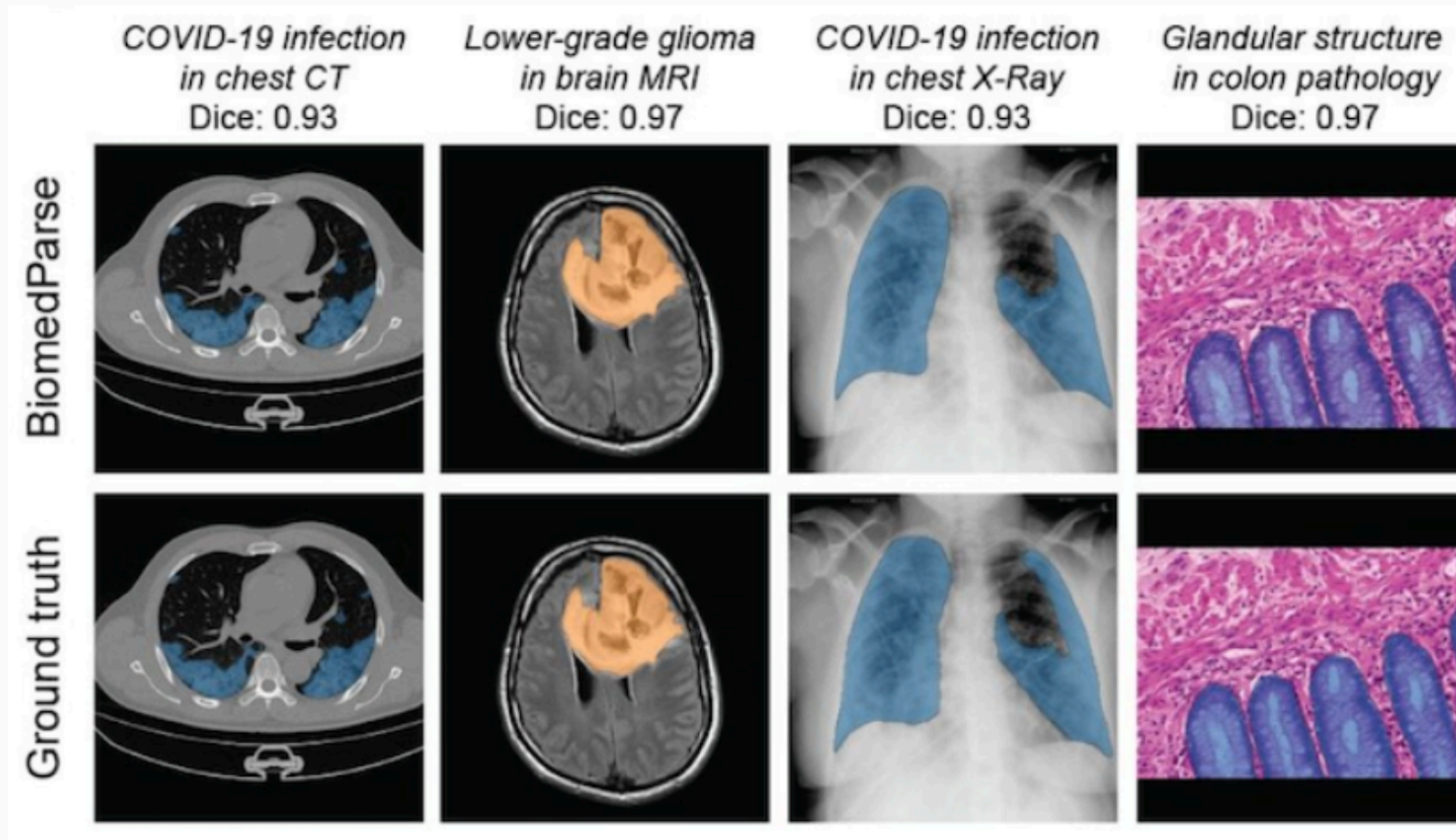


Samples of State-of-the-art Methods

<https://research.google/blog/take-all-your-pictures-to-the-cleaners-with-google-photos-noise-and-blur-reduction/>



<https://github.com/microsoft/BiomedParse>

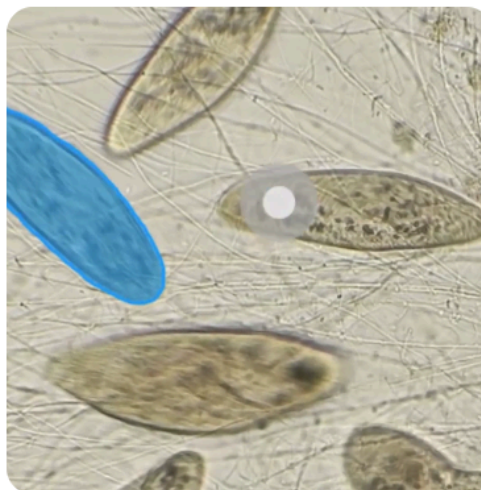


<https://ai.meta.com/research/sam2/>



Select objects and make adjustments across video frames

Using SAM 2, you can select one or multiple objects in a video frame. Use additional prompts to refine the model predictions.



Robust segmentation, even in unfamiliar videos

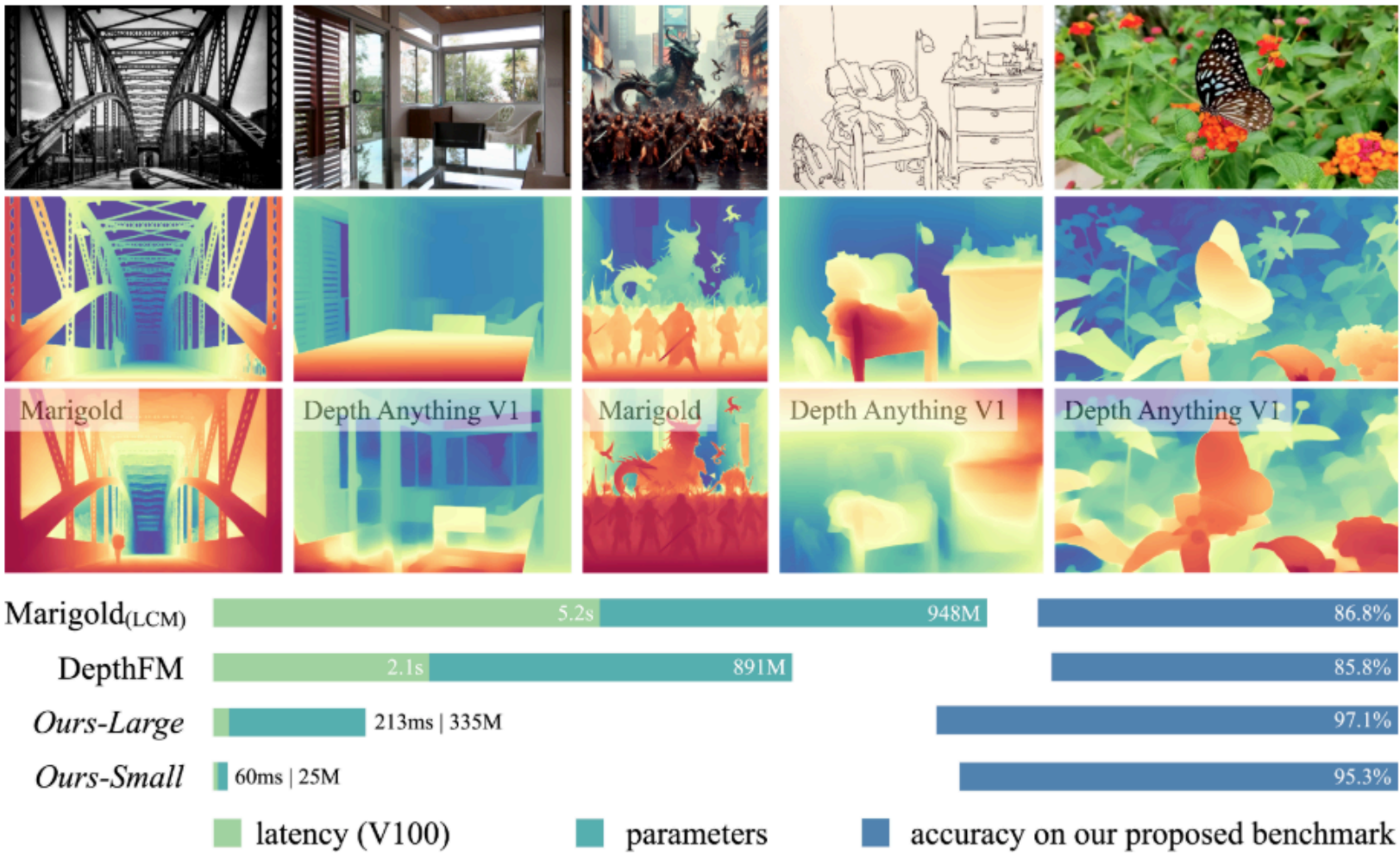
SAM 2 is capable of strong zero-shot performance for objects, images and videos not previously seen during model training, enabling use in a wide range of real-world applications.



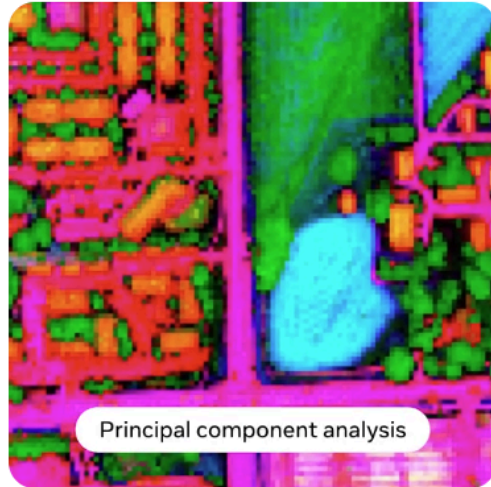
Real-time interactivity and results

SAM 2 is designed for efficient video processing with streaming inference to enable real-time, interactive applications.

<https://depth-anything-v2.github.io/>

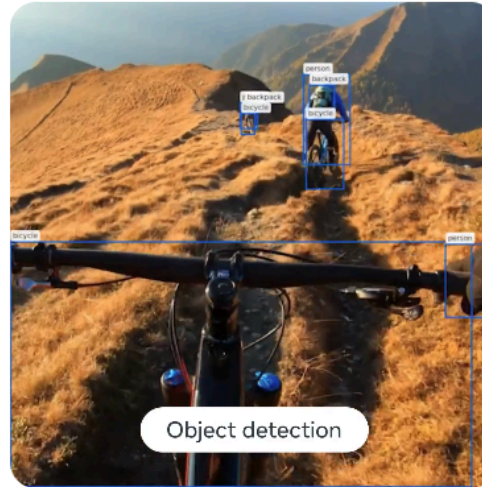


<https://ai.meta.com/research/dinov3/>



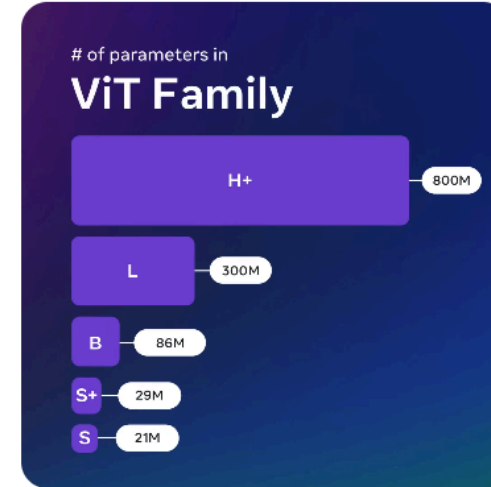
Exceptional performance across visual domains

SSL unlocks domains where annotations are scarce or costly. Backbones enable state-of-the-art results for tasks including object detection in web imagery, but also canopy height mapping in satellite and aerial imagery.



Versatile backbone with powerful dense image features

High-resolution dense features from a single DINOv3 backbone enable leading performance across vision tasks, including object detection, depth estimation, and segmentation, without any finetuning.



Efficient model sizes and architectures

We release a comprehensive model suite addressing a wide range of use cases, including broad coverage of ViT sizes and efficient ConvNeXt models for on-device deployment.

<https://research.google/blog/rt-1-robotics-transformer-for-real-world-control-at-scale/>

<https://research.google/blog/palm-e-an-embodied-multimodal-language-model/>

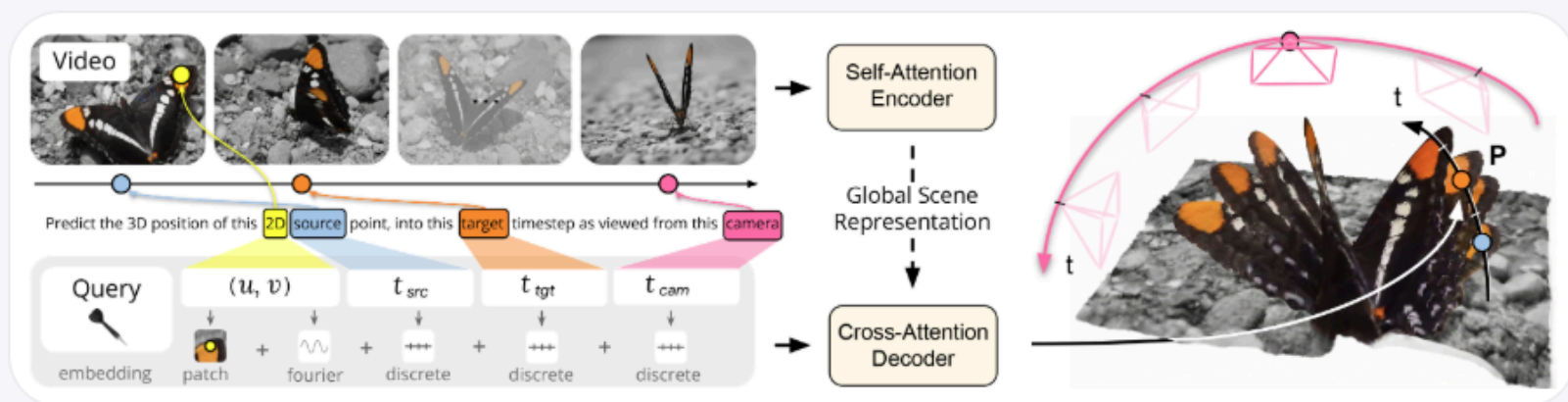


Efficiently Reconstructing Dynamic Scenes One 🎯 D4RT at a Time

CVPR 2026 best paper: <https://d4rt-paper.github.io/>

<https://research.google/blog/rt-1-robotics-transformer-for-real-world-control-at-scale/>

Method



A global self-attention encoder first transforms the input video into the latent *Global Scene Representation*, which is passed to a lightweight decoder. The decoder can be independently queried for the *3D position* P of any given 2D point (u, v) from the source timestep t_{src} at target timestep t_{tgt} in camera coordinate t_{cam} , unlocking full decoding at any point in space and time. The query also contains an embedding of the local frame patch centered around (u, v) , providing additional spatial context.

Course Organization

Signals and images:

- image formation, camera, and colors
- Convolution and Fourier transform
- Blur and image derivatives
- Aliasing and multi-scale representations
- labs

Correspondence and geometry:

- Keypoint detection
- Matching and homographies
- Camera models and perspective
- Camera calibration
- principles of stereo vision
- labs

Learned visual representations:

- CNNs
- ViTs
- detection/segmentation
- self-supervised learning
- vision-language models
- diffusion models
- pytorch labs

Advanced applications and challenges:

- personalization of pretrained models
- continual adaptation
- deployment
- robustness
- uncertainty and open world detection

- office at the CS department
- for office hours, write an email to book a timeslot



Figure 24: Antonio Carta



Figure 25: Donald Shenaj

No hard pre-requisites, but we expect familiarity with first-year courses

- linear algebra and optimization
- probability
- machine learning
- basic deep learning concepts
- basic familiarity with pytorch and the ability to train a simple DNN

- oral exam only
 - possibly a preliminary multiple choice questions test (e.g. 5 simple questions)
- no midterms this year

- **Labs:** hands-on notebooks
 - the goal of the labs is to let you explore and experiment to understand the limits of the method and see some simple applications and code recipes
 - write down your thoughts *before and after* running each experiment (hypothesis and analysis)
 - explore the notebooks, change hyperparameters and observe what changes in the results
 - the goal is not to memorize APIs or learn specific libraries
 - not evaluated. Feel free to use any AI tool

- **scipy/numpy**: classical image processing in Python (filters, transforms, feature detection)
- **OpenCV**: the standard classic CV and camera geometry library (C++/Python). We use the python bindings
- **Kornia**: differentiable computer vision in PyTorch, classical CV operations (filters, geometry, features) implemented as differentiable, GPU-friendly tensor ops
- **PyTorch/timm**: deep learning framework and a library of pretrained vision models (CNNs, ViTs)

Deep learning:

- Goodfellow, Bengio, Courville, *Deep Learning* <https://www.deeplearningbook.org/>
 - good reference for standard DL methods up to CNN (up to 2016). It does not contain recent methods (e.g. ViT and diffusion models)
 - you should know very well Part 1 and 2 of this book. Part 3 is useless for us.
 - for DL background if you need a refresher
- Simon J. D. Prince. “Understanding Deep Learning” <https://udlbook.github.io/udlbook/>
 - more modern DL book. Free pdf
 - websites provides several notebooks

Slides are the main reference

- Torralba et al., *Foundations of Computer Vision* <https://visionbook.mit.edu/>
 - free web version
 - main reference for everything on classic vision (with few exceptions)
 - tagged **visionbook** in the slides
- Szeliski, *Computer Vision: Algorithms and Applications* (2nd ed.), <https://szeliski.org/Book/>
 - free pdf on the website
 - good as a reference, light on details

if you want to know more on multi-view geometry (not necessary for this course):

- Hartley & Zisserman, *Multiple View Geometry in Computer Vision*, the definitive reference for Part 2's geometry (optional, advanced)

Conclusion

- Computer vision is the **inverse problem** of graphics: recovering scene structure and meaning from images, an inherently ill-posed problem requiring priors
- Vision is hard because of illumination, viewpoint, occlusion, scale, and intra-class variation, all at once
- Our intuitive understanding of images stems from strong priors, use of contextual information, and heavy preprocessing
- CV methods exploit correspondence, geometric constraints, representation learning, and temporal consistency

- visionbook Ch.1
- (optional) visionbook Ch.2 if you want a practical example of what classical techniques can and cannot do in a toy setting

Image formation

- Forward model and image formation
- The pinhole camera model
- Light and color