


# The whole picture

## A free roadmap from Python backend to computer vision in defence tech

A model sees a blob of pixels. An engineer has to see everything that made it, and everything that happens next.

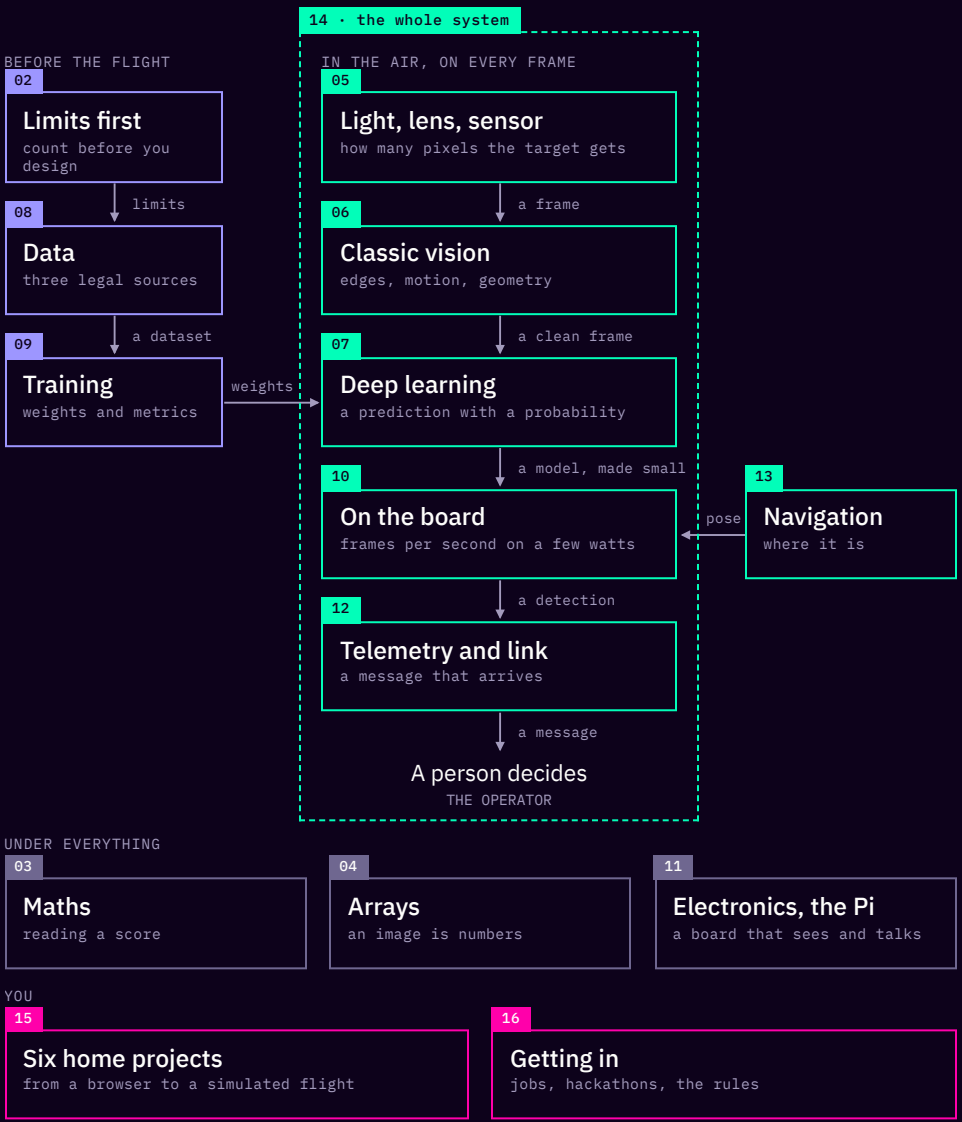
Daria Chuprina
Computer Vision Engineer

Version 2 · 23 November 2026

Українською ↗

# The map

The eye of a small drone is a chain. If one link is weak, the model at the end sees nothing. Every block is a chapter: tap it.



00 HOW TO USE THIS

# How I learned, and how to read this map

I came to computer vision as a Python backend developer. I could solder, I had built LLM agents, and I did not have the maths. I never finished a course from the first lecture to the last. What worked was simpler.

01 Take a task that is slightly too hard for you.

02 Read the documentation and the papers that task needs. Nothing else yet.

03 Ask an LLM to explain what you do not understand. Then check its answer with a measurement: it is a patient tutor, and it is sometimes confidently wrong.

04 Write down what failed and why. That page is your portfolio.

So this is not a syllabus. It is a map of one system, the eye of a small drone. For every part of it you get four things: why it matters, where to read, what to build at home, and how to know you are done. Start from any block that touches a task you already have. If you have none, start with the six home projects.

WHAT "FREE" MEANS HERE

FREE — Free, with no time limit. Most of this map.

FREE TIER — A free plan with limits: a GPU quota, or projects that everyone can see.

AUDIT — Free to watch for a limited time, without graded work.

Coursera is missing on purpose. Since August 2025 only the first module of a Coursera course is free. Where a Coursera course matters, this map links the same lectures on the author's own site. Hours are rough estimates. Almost everything here is in English.

02 BEFORE THE FLIGHT

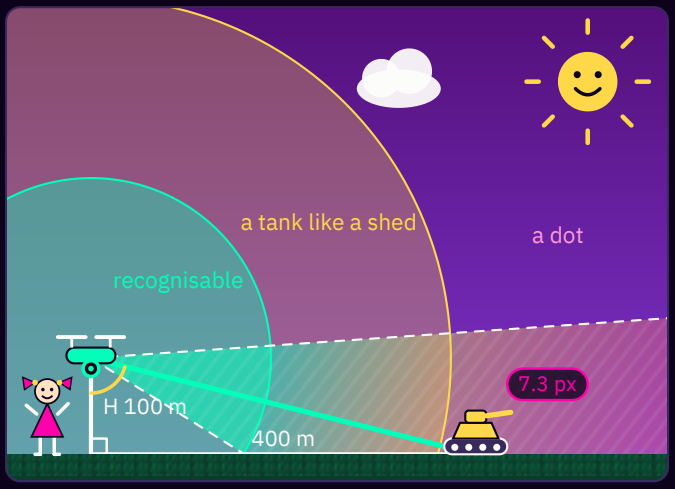
# See like Wolowitz, not like Sheldon

IN
a camera, a board, the weather, a map

OUT
→ the limits inside which a model has a chance

Sheldon derives the theory of everything on a whiteboard. Wolowitz works out whether the bracket survives the launch, and that is why his hardware is the one that flies to space. In vision from a drone, Wolowitz wins: the architecture of the model is chosen last. First physics, optics and geography say what can be seen at all.

Below are the questions worth counting before the first line of code. Every example uses one camera: a frame 1280 px wide, a 60° lens, a height of 100 m, the camera looking straight down. One pixel covers 9 cm of ground.




The scene from the front page of the roadmap. The discs around the camera show how many pixels are left on the target: 16 and more in the green one, from 8 in the yellow one, from 2 in the pink one. The numbers in the scene are rough, an order of magnitude.

## HOW MANY PIXELS

01 **How many pixels does it take to see, to recognise and to name?**

Johnson's criteria, 1958: a person needs about 2 px across the target to notice it, 8 px to tell a vehicle from a person, and 13 px to name the type. These are thresholds with a 50% probability on a clean image. For a network on a blurred frame, leave a margin.

**notice 2 px · recognise 8 px · name 13 px**

### 02 The object is smaller than 6 pixels. What can be done with it at all?

There is no class there, only “something is there”. A box does not work either: a shift of one pixel along both axes leaves an IoU of 0.53, and the metric notices. Time helps: the mean of N frames cuts the noise √N times, and motion tells a target from a stone. Look for a track before a detection and a mask, not a box.

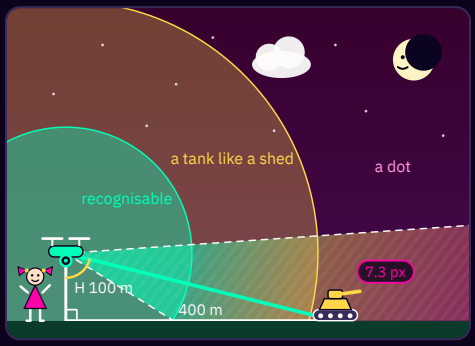
**IoU(6 px, shift 1 px) = 25 / 47 = 0.53 · noise / √N**

### 03 Will a camera with more megapixels save me?

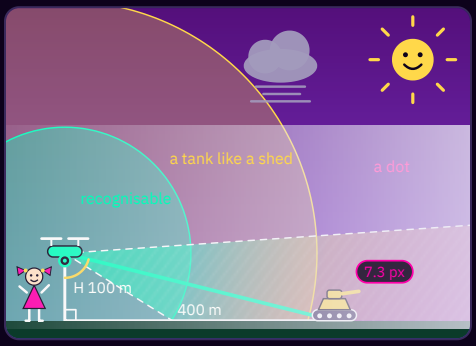
Not always. A lens draws a point as a blob, and a pixel smaller than the blob adds nothing. In visible light at f/2.8 the blob is 3.8 µm. In a thermal camera at f/1.0 it is 24 µm, twice the size of a 12 µm pixel: a thermal sensor sees less than its resolution promises. And count the pixels at the network input, not on the sensor.

**d = 2.44 · λ · N**

LIGHT AND AIR




At night this camera is blind. Take a thermal one.




The fog ate the target. Come closer.

The same scene at night and in fog. The discs have not moved: the pixels are the same, the contrast is gone.

### 04 It is dark. What breaks first?

From noon to a moonless night there is a hundred million times less light: 100,000 lx against 0.001. The camera answers with a longer exposure or with gain. The first gives blur, the second noise: signal to noise grows only as the square root of the number of photons. A thermal camera does not care about the dark, but in the morning and in the evening the target can match the background in temperature and vanish.

**SNR = √photons: 10,000 → 100, 100 → 10**

### 05 What smears the frame: speed or rotation?

More often rotation. Flying at 20 m/s with a 1/100 s exposure smears the frame by 2 px. Rolling at 30° per second with the same exposure smears it by 6 px, and height does not help here. That is why a gimbal and a short exposure matter more than the model.

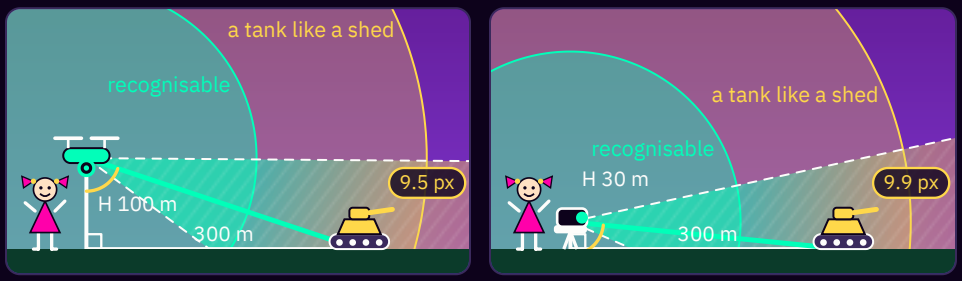
$\text{blur} = v \cdot t / \text{GSD}$ $\text{blur} = \omega \cdot t \cdot f$

### 06 Fog and haze: at what distance does the target vanish?

Air eats contrast exponentially. Meteorological visibility V is the distance at which 2% of the contrast is left. At half of that distance 14% is left: with a visibility of 1 km a target at 500 m is already grey on grey. Haze with a visibility of 10 km takes a third of the contrast per kilometre.

$C = C0 \cdot \exp(-3.912 \cdot d / V)$

THE PLATFORM




On the left the camera flies at 100 m, on the right it stands on a 30 m tower. From the tower the angle to the same target is almost the horizon.

### 07 A wing or a copter: what does it change for vision?

Time on target. A camera looking down from 100 m sees a strip of 65 m along the track. A wing at 20 m/s crosses it in 3 seconds: 97 frames in all. A copter can hover. A wing also banks in a turn: a circle of 100 m radius at 20 m/s means 22° of bank, and without a gimbal the camera axis slides 40 m sideways on the ground.

$t = \text{footprint} / v$ $\tan(\text{bank}) = v^2 / (g \cdot R)$

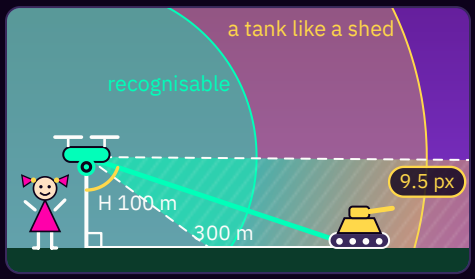
### 08 The drone tilted. What happened to my pixel formula?

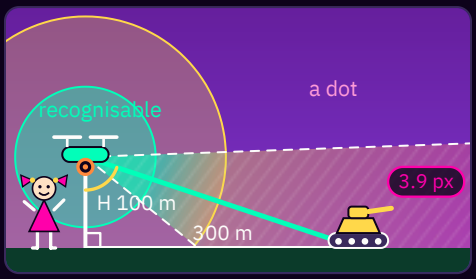
The angle from nadir changed, and everything with it. Roll and pitch from the telemetry must be added to the mounting angle of the camera, or the map from chapter 5 shows the wrong frame. The price of an error grows towards the horizon: one degree of error from 100 m moves the point on the ground by 1.7 m under the drone and by 7 m at 60°.

$\Delta = H \cdot \Delta\theta / \cos^2\theta$

02 See like Wolowitz, not like Sheldon · continued

## THE LENS



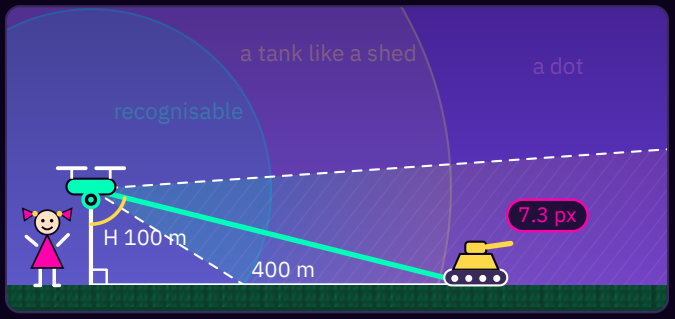

The cone is the frame: everything between its two edge rays. A plain camera on the left, a thermal one on the right: it has fewer pixels, so the discs are tighter.

### 09 A wide lens. Why is everything off at the edges?

Because f · tan θ is honest only for a lens without distortion. An ideal 120° lens gives four times more pixels per degree at the edge than in the centre, while a fisheye keeps them equal and bends straight lines. Calibrate the camera and count after the correction: it stretches the edges, but it creates no new pixels.

**r = f · tan θ against r = f · θ**

## GROUND, MAP AND 3D




The triangle from question 10: the height H, the ground range, the slant range and the angle from straight down.

### 10 Where on the ground is the thing I see in this pixel?

A point on the ground is the position of the drone plus the height times the tangent of the ray angle. There are three “heights” here: above the take-off point, above sea level and above the GPS ellipsoid, and in Europe the last two differ by tens of metres. You need a fourth one, above the ground under the target, and there are terrain maps for it. A height error of 10 m at 60° gives 17 m on the ground. And a compass points at magnetic north, not at true north.

**GR = H · tan θ**

02 See like Wolowitz, not like Sheldon · continued

### 11 Why do distances on a web map lie?

The Mercator projection stretches the map more the further you are from the equator. At a latitude of 50° a metre on the map is 1.56 times longer than a real one, and areas are 2.4 times larger. A degree of longitude there is 71.6 km instead of 111.3. Measure in metres on the ground, not in pixels of the map.

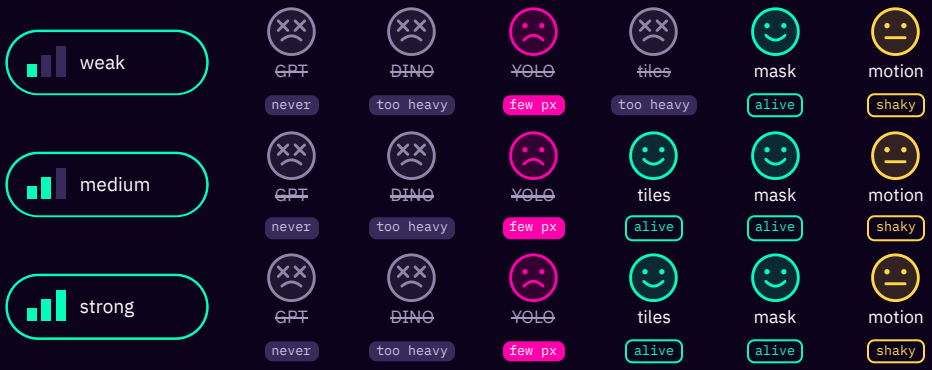
**scale = 1 / cos(latitude)** **m/px = 156,543 · cos(latitude) / 2^z**

### 12 A 3D model for synthetic data: where do the warped surface and the torn texture come from?

From the same mathematics as Mercator: a curved surface cannot be unfolded onto a plane without stretching, and Gauss proved it. UV unwrapping stretches the texture where the surface curves and tears it at the seams. Put the seams where the camera does not look, and check the density: a 2048 px texture on a seven-metre model gives 293 texels per metre, while a camera at 100 m sees 11 px per metre. Pass the render through the same camera you counted: size, blur, noise.

**texels per metre = texture size / model size**

THE BOARD




The same scene on three classes of hardware. For the weak one the tiles are already too heavy.

### 13 How many frames per second does the board need?

Enough for the target not to escape the tracker between frames. The ground under a camera at 20 m/s from 100 m moves 7 px per frame at 30 fps and 44 px at 5. Six tiles at 40 ms each make 4 fps. A delay of 200 ms at this speed is 4 m that the drone has already flown. A microcontroller is too small for a network: see chapter 10.

**shift = v / (fps · GSD)** **fps = 1 / (tiles · network time)**

02 See like Wolowitz, not like Sheldon · continued

THE CARD OF LIMITS: FILL IT IN BEFORE YOU CHOOSE AN ARCHITECTURE

| WHAT YOU COUNTED | WHAT IT DECIDES |
|---|---|
| **Pixels on target** | The method: a track and a mask, tiles, or an ordinary detector. |
| **Blur and noise** | The exposure, the need for a gimbal, and which augmentations are honest. |
| **Contrast in fog and in the dark** | The camera, and the conditions the dataset must hold. |
| **Time on target, frames per second** | The input size, the number of tiles and the board. |
| **Error of the point on the ground** | What you can promise the operator. |
| **Projection and heights** | How to stitch detections to the map. |
| **Unwrapping and texture density** | Whether the 3D model suits this height. |

**the order**

Limits first, then data, then the model. If the target has 5 px, the best detector will not help. If it has 40, a simple one is enough.

ALSO When the sun is low and shadows are long: the NOAA calculator · Blender

**at home**

Take your calculator from chapter 5 and add four lines: blur from speed, blur from rotation, contrast from visibility, time on target. Fill in the card of limits for one camera and two flight profiles: a wing at 100 m and a copter at 30 m. Then play with the limits on the front page.

☐ DONE WHEN You have one page of numbers that shows which method makes sense, before the first labelled frame.

03 UNDER EVERYTHING

# The maths you actually use

IN
school maths, half forgotten

OUT
→ reading a score, a matrix, a loss curve

A detector does not say “tank”. It says 0.88. Reading that number, a loss curve or a camera matrix takes school probability and a little linear algebra. Not a degree.

start here

FREE

## Essence of Linear Algebra

3BLUE1BROWN · 4 H · BEGINNER

Vectors, matrices and transforms as moving pictures. Watch it before you meet a single formula.

FREE

## Seeing Theory

BROWN UNIVERSITY · 3–5 H · BEGINNER

Probability and Bayes as things you drag with a mouse.

FREE

## Think Stats, 3rd edition

ALLEN DOWNEY · 20 H · BEGINNER

Statistics taught through Python code, for people who think in code.

FREE

## Essence of Calculus

3BLUE1BROWN · 3 H · BEGINNER

Derivatives and the chain rule: all the calculus that training a network needs.

FREE

## 18.06SC Linear Algebra

MIT OPENCOURSEWARE · 40+ H · INTERMEDIATE

The rigorous course, for the day you want proofs and problem sets.

ALSO Mathematics for Machine Learning (book) · Khan Academy, to fill gaps

at home

Take the output of any detector, with scores. Plot one histogram of scores for correct detections and one for wrong ones. Move the threshold and count what you lose and what you let in.

☐ DONE WHEN You can explain to a friend why 0.5 is a choice and not a law.

04 UNDER EVERYTHING

# From backend to arrays

IN
Python lists, loops, JSON

OUT
→ an image as an array you can slice

Backend code moves records. Vision code moves arrays. An image is a block of numbers, height by width by three, and most bugs of your first month are a wrong axis, a wrong type or a wrong colour order.

start here

FREE

## NumPy: the absolute basics for beginners ↗

NUMPY · 3 H · BEGINNER

Arrays, slicing, broadcasting. Three hours that pay back every day.

FREE

## A crash course on NumPy for images ↗

SCIKIT-IMAGE · 1 H · BEGINNER

Rows and columns, channels, masks: the conventions that bite.

FREE

## Python / NumPy tutorial ↗

STANFORD CS231N · 2 H · BEGINNER

A fast bridge for people who already program.

FREE

## Python Data Science Handbook ↗

JAKE VANDERPLAS · 8 H · BEGINNER

The NumPy and Matplotlib chapters, when you want depth.

at home

Load a photo as an array. Without calling an image library: crop it, flip it, swap red and blue, make it grey, and draw a rectangle by assigning to a slice.

☐ DONE WHEN You read `img[y1:y2, x1:x2, ::-1]` without stopping.

05 IN THE AIR, ON EVERY FRAME

# Light, lens, sensor

| IN | OUT |
|---|---|
| light from the scene | → a frame, and a number of pixels on the target |

Before any network runs, optics has already decided what can be seen. A vehicle four metres long, filmed from a hundred metres through a wide lens, may be thirty pixels. Through the wrong lens it is eight, and no model brings it back.

start here

FREE

### First Principles of Computer Vision ↗

SHREE NAYAR, COLUMBIA UNIVERSITY · 40+ H · BEGINNER

Two modules to start with: Image Formation and Image Sensing.

FREE

### Understanding focal length and field of view ↗

EDMUND OPTICS · 30 MIN · BEGINNER

Focal length, sensor size and field of view, on two pages.

FREE

### How much ground does one pixel cover?

CALCULATORS

Two calculators to check your own arithmetic: ground sample distance · field of view

FREE

### CMOS rolling shutter cameras ↗

BASLER · 30 MIN · BEGINNER

Rolling and global shutter, and what motion does to each.

FREE

### Thermal cameras (white paper) ↗

AXIS · 1 H · BEGINNER

Thermal cameras: sensitivity, range, and what it takes to detect or identify.

ALSO Pi camera specifications, to practise on · Edmund Optics guide · Calibration in OpenCV

at home

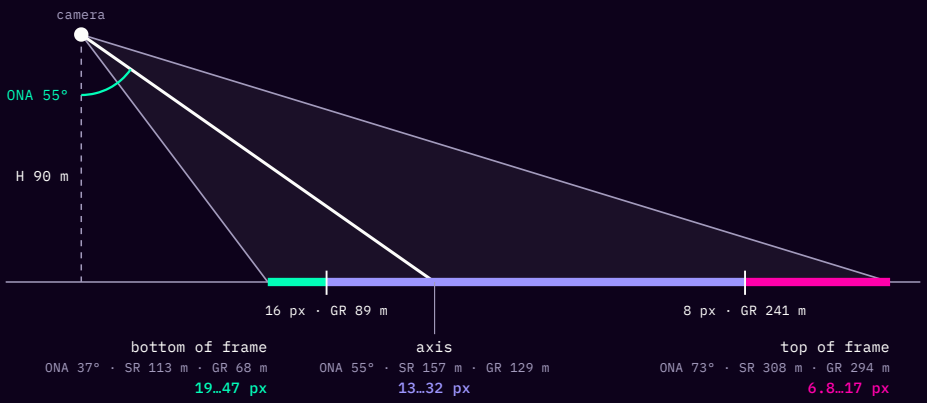
Photograph a one-metre object from 2, 5 and 10 metres. Count its height in pixels. Compare with what the lens and sensor numbers predict.

☐ DONE WHEN Given a camera, a height and a target size, you can say how many pixels the target gets.

05 Light, lens, sensor · continued

PIXELS ON TARGET: COUNT BEFORE YOU TRAIN

A model never sees a vehicle. It sees a number of pixels, and that number is fixed before take-off by four things: the height above the ground, the angle of the camera, the lens and the size of the target. Put them into one line of arithmetic and you know what can be found, and where in the frame, before a single image is labelled.




Height 90 m, camera axis 55° from nadir, a 60° lens, a frame 1280 px wide. The target is a vehicle 4.5 × 1.8 m: the first number is nose-on, the second side-on.

| | |
|---|---|
| `f = (W / 2) / tan(HFOV / 2)` | focal length in pixels, from the frame width W and the horizontal field of view |
| `SR = H / cos(ONA)` | slant range, the distance along the ray. ONA is the angle from nadir, from straight down |
| `px = f · L / SR` | pixels across a target of size L in the middle of the frame |

From 100 m at 60° from nadir the slant range is 200 m: the vehicle is 10 px nose-on and 25 px side-on. Tilt the same camera down to 30° and the same 1.8 metres become 17 px. Nothing about the model has changed.

05 Light, lens, sensor · continued

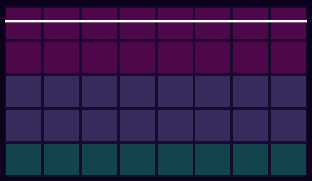
THE FRAME IS NOT ONE NUMBER

The top of the frame looks far and the bottom looks near, so the same target is a dot in one place and an object in another. Cut the frame into bricks, run the formula for the centre of each, and colour it by the worst view of the target. Draw this map before you choose a model.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | 2.1…5.3<br>SR 986 m | 3.5…8.8<br>SR 584 m | 4.9…12<br>SR 418 m | 6.3…16<br>SR 332 m | 7.7…19<br>SR 282 m | 9.1…23<br>SR 251 m |
| 4…10<br>SR 557 m | 5.4…14<br>SR 393 m | 6.8…17<br>SR 301 m | 8.2…21<br>SR 245 m | 9.6…24<br>SR 209 m | 11…28<br>SR 186 m | 12…31<br>SR 172 m | 14…35<br>SR 163 m |
| 8.7…22<br>SR 256 m | 10…25<br>SR 209 m | 12…29<br>SR 177 m | 13…32<br>SR 154 m | 14…36<br>SR 139 m | 16…39<br>SR 130 m | 17…43<br>SR 124 m | 19…46<br>SR 120 m |
| 13…34<br>SR 167 m | 15…37<br>SR 144 m | 16…41<br>SR 127 m | 18…44<br>SR 114 m | 19…48<br>SR 106 m | 20…51<br>SR 101 m | 22…55<br>SR 98 m | 23…58<br>SR 97 m |
| 18…45<br>SR 126 m | 20…49<br>SR 112 m | 21…52<br>SR 101 m | 22…56<br>SR 92 m | 24…59<br>SR 87 m | 25…63<br>SR 84 m | 27…66<br>SR 82 m | 28…70<br>SR 82 m |

under 8 px · 8 to 16 px · 16 px and more · sky · horizon

Height 50 m, axis 70° from nadir, 15° of roll. In each brick: pixels nose-on … side-on, and the slant range to the ground at that point.



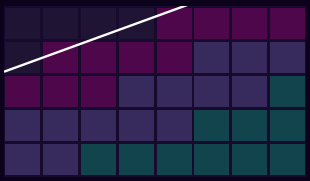
LEVEL, LOOKING FAR · ONA 75°

A dot out to 245 m, a small object out to 116 m. Most of the frame is too far.



TILTED DOWN · ONA 45°

Every brick has 16 px or more. The frame now ends 98 m ahead.



20° OF ROLL, NO GIMBAL

The good bricks slide into one lower corner. A stabilised camera gives the band back.

Read the map as two zones. Far, under the horizon: a dot that says only "something is there". Near, at the bottom edge: enough pixels to say what it is. A target that comes close enough to classify is already sliding out of the bottom of the frame. Height, tilt and a gimbal move these zones. A model does not.

05 Light, lens, sensor · continued

### WHERE THE FOUR NUMBERS COME FROM

| NUMBER | SOURCE | WATCH OUT |
|---|---|---|
| **H, height above ground** | Barometer, GNSS or a rangefinder. In MAVLink: `relative_alt` in `GLOBAL_POSITION_INT`. | It is the height above the take-off point. Over a hill or a valley the ground is somewhere else. |
| **ONA, angle from nadir** | Pitch and roll from `ATTITUDE`, plus the mounting angle of the camera. With a gimbal: its own status message. | Without a gimbal the camera tilts with the airframe in every turn. |
| **f, focal length in pixels** | The field of view and the frame width, or a calibration. | Count pixels at the detector input, not on the sensor. |
| **L, size of the target** | A table of dimensions. | Take the narrow side: nose-on is the view you will get. |

### PIXELS CHOOSE THE METHOD

| NARROW SIDE | WHAT IS THERE | WHAT WORKS |
|---|---|---|
| `under 4 px` | Noise. In one frame the target cannot be told from a stone. | Motion only: the difference between frames, a track before a detection. |
| `4–8 px` | A dot: "something is there", with no class. | A network that paints a mask or a heat map instead of a box, at full resolution. |
| `8–16 px` | A small object, a coarse class at best. | A detector on tiles (SAHI) with a head for small scales. |
| `16–32 px` | An object with a class. | An ordinary detector, and a box you can track. |
| `32 px and more` | Detail: the type, the parts, the heading. | Anything, classic methods included. |

The limits are not pulled from the air. A human observer needs about 2 px across the narrow side to detect a target, 8 px to recognise it and 13 px to identify it: Johnson's criteria. The AI-TOD benchmark calls 2–8 px very tiny, 8–16 px tiny and 16–32 px small.

| | |
|---|---|
| `SR_max = f · L / px_min` | Every pixel of the threshold costs range. For the 1.8 m side: 8 px reach 249 m, 16 px reach 125 m, 32 px reach 62 m. |

05 Light, lens, sensor · continued

## WHAT THE FORMULA LEAVES OUT

- **Blur.** The camera moves while the shutter is open. Blur in pixels is speed times exposure, divided by metres per pixel: 20 m/s at 1/100 s over 10 cm pixels smears every edge across 2 px.
- **Lens and air.** The count assumes perfect optics. A real lens, haze and video compression dissolve a small target into the background first.
- **Ground.** The height is measured from the take-off point, and the ground is not flat. On a slope the whole map shifts.
- **Input size.** Count at the detector input. A 1920 px frame squeezed to 640 px keeps a third of every number above.
- **The target itself.** Trees, shadow, camouflage, a turned hull. Count the narrow side, then expect less.

FREE

### MAVLink common messages ↗

MAVLINK · REFERENCE · INTERMEDIATE

The messages that carry height and attitude: `GLOBAL_POSITION_INT`, `ATTITUDE`, the gimbal status.

FREE

### Dissecting the camera matrix: the intrinsic matrix ↗

KYLE SIMEK · 30 MIN · INTERMEDIATE

What the focal length in pixels is, and where it sits in the camera matrix.

FREE

### AI-TOD: tiny objects in aerial images ↗

WANG ET AL. · DATASET · INTERMEDIATE

A benchmark where the average object is 12.8 px: what "tiny" does to a detector. Research use only.

ALSO Metres per pixel · Slant range · Pitch, roll and focal length from one frame: GeoCalib · Why IoU breaks at 8 px · Small targets as masks: BasicIRSTD · Gimbals · Terrain following

**at home**

Write the calculator: thirty lines of Python. Height, angle, field of view, frame size and target size go in; pixels nose-on and side-on come out. Cut the frame into bricks and colour the map. Then check one brick against a photo taken from a balcony or a hill.

☐ DONE WHEN For any camera and any flight profile you can draw the map of the frame and name the method for each zone.

06 IN THE AIR, ON EVERY FRAME

# Classic vision: Python, then C++

| IN | OUT |
| --- | --- |
| a frame | → edges, shapes, motion, geometry |

Not everything needs a network. Thresholds, edges, optical flow and geometry run on a tiny processor and never hallucinate. They still stabilise the video, calibrate the camera and check what the network claims. On board, this code is often C++.

start here

FREE

### First Principles of Computer Vision ↗

SHREE NAYAR, COLUMBIA UNIVERSITY · 40+ H · BEGINNER

The whole course is about why each method works: filtering, edges, features, homography, optical flow.

FREE

### OpenCV-Python tutorials ↗

OPENCV · 15 H · BEGINNER

The same algorithms, hands on, in Python.

FREE

### OpenCV Bootcamp ↗

OPENCV UNIVERSITY · 3 H · BEGINNER

A three-hour tour, if you have never opened OpenCV.

FREE

### Modern C++ for Computer Vision (2020)

VIZZO AND STACHNISS, UNIVERSITY OF BONN · 20 H · BEGINNER

Modern C++ taught for vision people, with build tools and an OpenCV tutorial: course page · lectures

FREE

### CMake Tutorial ↗

KITWARE · 4–6 H · BEGINNER

How a C++ project is built, step by step.

ALSO OpenCV C++ tutorials · learncpp.com, as a reference

at home

Track a bright object in webcam video with Python: threshold, contours, centroid. Then write the same thing in C++ with CMake, and time both.

☐ DONE WHEN Your C++ version builds from a clean checkout with one command.

07 IN THE AIR, ON EVERY FRAME

# Deep learning for vision

| IN | OUT |
| --- | --- |
| pixels | → a prediction with a probability |

A network is a function with millions of knobs, turned by data. You need enough theory to pick one, train it, and notice when it lies. Take one practical course first and one rigorous course later.

start here

FREE

## Practical Deep Learning for Coders ↗

FAST.AI · 9 LESSONS · INTERMEDIATE

Top down: a working model in the first lesson, the theory after.

FREE

## CS231n: Deep Learning for Computer Vision, 2025 ↗

STANFORD · 30–60 H · ADVANCED

The rigorous one: convolutions, detection, transformers.

FREE

## Dive into Deep Learning ↗

D2L.AI · REFERENCE · INTERMEDIATE

A textbook where every formula runs. Chapter 14 builds a detector.

FREE

## Computer Vision Course ↗

HUGGING FACE · SELF-PACED · INTERMEDIATE

A free course that reaches the current architectures.

ALSO CS231n course notes

at home

Fine-tune a small classifier on two classes of your own photos. Then find twenty images it gets wrong and say why, one by one.

☐ DONE WHEN You can sketch what happens to a 640 by 640 image between the input and the answer.

08 BEFORE THE FLIGHT

# Data

| IN | OUT |
| --- | --- |
| public sets, renders, the open web | → a reviewed, labelled dataset |

In defence nobody hands you a dataset. Most of the work is here, and so are most of the mistakes. There are three legal sources: public datasets, synthetic renders, and images collected from open sources and reviewed by a person.

## PUBLIC DATASETS, SEEN FROM ABOVE

start here

FREE

### VisDrone ↗

TIANJIN UNIVERSITY · DATASET

Drone video and photos with boxes. The place to start for the view from above. No licence is stated: treat it as research only.

FREE

### DOTA ↗

WUHAN UNIVERSITY · DATASET

Aerial images with rotated boxes. Academic use only.

FREE

### xView ↗

DIUX / NGA · DATASET

Satellite imagery at 0.3 metres per pixel, 60 classes.

FREE

### DroneVehicle (RGB + infrared) ↗

VISDRONE TEAM · DATASET

Paired colour and infrared frames from a drone, day and night.

ALSO COCO · Open Images · Roboflow Universe (read each licence) · Anti-UAV, thermal video

08 Data · continued

## SYNTHETIC AND COLLECTED

FREE

### Blender ↗

BLENDER FOUNDATION · TOOL

Render what does not exist. The lesson that cost me most: make the object exact and everything else random. Chasing realism is the wrong lever.

FREE

### Poly Haven ↗

POLY HAVEN · TEXTURES, SKIES

Textures and skies under CC0, for the random part.

FREE

### CVAT ↗

CVAT.AI · TOOL · BEGINNER

Free labelling: boxes, polygons, video, with model assistance.

ALSO Label Studio

## FIVE TRAPS

- **Class imbalance.** Ten thousand tanks, two hundred radars: the model knows only tanks.
- **Train and validation leak.** Neighbouring frames of one video on both sides: 0.95 on paper, 0.4 in the field. Split by video, not by frame.
- **Domain gap.** Trained on renders or on photos from the ground, then flown from a drone.
- **Small objects.** The target is fifteen pixels wide. Tile the image, raise the resolution, or change the lens.
- **Hardware decides the model.** A few watts on board. Choose the model for the chip, not for the leaderboard.

at home

Download the VisDrone validation set. Count the objects of each class and the size of their boxes in pixels. Draw both histograms.

☐ DONE WHEN You can say what your dataset does not contain.

09 BEFORE THE FLIGHT

# Training, and reading the metrics

IN
a dataset

OUT
→ weights, and numbers you can defend

Training is the easy part: one command. The skill is reading what comes out, and knowing the licence of what you trained.

start here

FREE
**Ultralytics documentation** ↗
ULTRALYTICS · 10 H · INTERMEDIATE
Train, validate and export in a few lines. VisDrone downloads itself.

FREE
**Performance metrics, explained** ↗
ULTRALYTICS · 1 H · BEGINNER
Precision, recall, mAP and the confusion matrix, in plain words.

FREE
**SAHI: sliced inference for small objects** ↗
OBSS · 2 H · INTERMEDIATE
Slice the image into tiles, so that small objects survive.

FREE
**Detectors under Apache-2.0**
CODE AND WEIGHTS YOU MAY SHIP
Three current detectors with a permissive licence: RF-DETR · D-FINE · RT-DETR

FREE TIER
**A free GPU**
WEEKLY QUOTAS AND LIMITS APPLY
Enough for every exercise in this map: Kaggle Notebooks · Google Colab

read the licence

Ultralytics code and weights are AGPL-3.0: a device or a commercial product needs their paid licence. If you plan to ship, start from an Apache-2.0 detector.

at home

Train a nano detector on VisDrone for a few epochs on a free GPU. Find the class and the box size where recall collapses.

☐ DONE WHEN You can name your model's weakest point with a number.

10 IN THE AIR, ON EVERY FRAME

# On the board

| IN | OUT |
| --- | --- |
| trained weights | → frames per second on a few watts |

A model is trained on a large GPU and never flies with it. On board there is a small computer and an accelerator on a few watts, so the model is exported, quantised to 8-bit integers and compiled for that chip.

start here

FREE

## Raspberry Pi AI documentation ↗

RASPBERRY PI · 3 H · BEGINNER

A Raspberry Pi 5 with a Hailo accelerator: install it and run a detector.

FREE

## hailo-apps ↗

HAILO · 5 H · INTERMEDIATE

Thirty ready pipelines: detection, pose, segmentation.

FREE

## Quantize ONNX models ↗

ONNX RUNTIME · 2 H · INTERMEDIATE

What quantisation does to a model, and how to debug it.

FREE

## TinyML courseware ↗

HARVARD TINYMLX · SELF-PACED · INTERMEDIATE

The Harvard TinyML materials: slides and notebooks, with no access window.

AUDIT

## Tiny Machine Learning ↗

HARVARDX ON EDX · 3 COURSES · INTERMEDIATE

The same course with video and a schedule. Audit access ends after a few weeks.

ALSO Hailo Model Zoo: compile your own model (v2 branch for Hailo-8) · Exporting to ONNX · Edge Impulse, free plan

at home

Export your detector to ONNX and run it on a laptop CPU with ONNX Runtime. Quantise it. Measure speed and accuracy before and after. If you have a board, repeat there.

☐ DONE WHEN You have one table: model, megabytes, frames per second, accuracy.

10 On the board · continued

## THREE FAMILIES OF BOARDS

The chip decides which tool turns your ONNX file into something it can run, and that tool decides which layers survive. Choose a board by its toolchain first, by watts second, and by the number on the box last.

| BOARD | ACCELERATOR | FROM ONNX TO THE CHIP | THE PRICE |
|---|---|---|---|
| **Raspberry Pi 5 with Hailo** | A separate chip on PCIe: Hailo-8L, 13 TOPS, or Hailo-8, 26 TOPS. | Hailo Dataflow Compiler on a Linux PC. Integers only, with calibration. | A few watts. The compiler is strict with unusual layers. |
| **NVIDIA Jetson Orin Nano** | A GPU with CUDA, up to 67 TOPS. | TensorRT builds the engine on the board itself. 16-bit needs no calibration. | 7 to 25 W, heat and money. Runs almost any model. |
| **Orange Pi 5 and other RK3588 boards** | An NPU inside the processor, up to 6 TOPS. | RKNN-Toolkit2 on an x86 Linux PC. 16-bit, or 8-bit integers with calibration. | Usually the cheapest. Thinner documentation. |

FREE

### Hello AI World ↗

NVIDIA · 6–10 H · BEGINNER

Classification, detection and segmentation on a Jetson with TensorRT, step by step.

FREE

### NVIDIA Jetson: setup and benchmarks ↗

ULTRALYTICS · 2 H · BEGINNER

Setting up a Jetson, and what each precision gives: 32-bit, 16-bit, 8-bit.

FREE

### RKNN-Toolkit2 ↗

ROCKCHIP · 3 H · INTERMEDIATE

The converter for Rockchip NPUs, with the lists of supported chips and layers.

ALSO Jetson Orin Nano: quick start · TensorRT quick start · Jetson AI Lab · Ready models for RK3588 · Exporting to RKNN · Raspberry Pi: benchmarks · AI HAT+

**before you buy**

Export your own model to ONNX and read the list of layers the chip supports. One unsupported layer in the middle of a network costs more than any benchmark gives.

11 UNDER EVERYTHING

# Electronics, and a Pi at home

IN
a 5 V supply and some pins

OUT
→ a board that sees and talks

A vision engineer who cannot read a pinout waits for someone else all day. Voltage, current and three serial buses cover most of what a companion computer needs, and you can learn them in a browser before you buy a board.

start here

FREE TIER **Wokwi simulator** ↗
WOKWI · 2–10 H · BEGINNER
ESP32, Arduino and Pi Pico simulated in the browser, with a logic analyser. Free projects are public.

FREE **Four short tutorials**
SPARKFUN · 1 H EACH · BEGINNER
The clearest explanations of the basics: Ohm's law · UART · I2C · SPI

FREE **Camera software** ↗
RASPBERRY PI · 3–5 H · BEGINNER
The camera stack of the Pi: the rpicam tools and Picamera2.

FREE **The Picamera2 Library** ↗
RASPBERRY PI · 4–8 H · BEGINNER
The Python camera library, with OpenCV and Hailo examples.

FREE **Getting started with the Camera Module** ↗
RASPBERRY PI FOUNDATION · 1 H · BEGINNER
A first camera project in an hour.

ALSO All About Circuits, a free textbook

at home

In Wokwi, build an ESP32 project in MicroPython: blink an LED through a resistor, read an I2C sensor such as an MPU6050, print the readings over UART, and watch the bus with the logic analyser.

☐ DONE WHEN You can say what every wire in your project carries.

12 IN THE AIR, ON EVERY FRAME

# Telemetry and the link

IN
a detection

OUT
→ a message that reaches a person in time

A detection that stays on the drone is worth nothing. It has to become a message, cross a radio link that is weaker than you hoped, and arrive in time. You can practise all of it with a simulated drone.

start here

FREE

### ArduPilot SITL simulator ↗

ARDUPILOT · 3–4 H · BEGINNER

A complete autopilot on your laptop. No drone needed.

FREE

### MAVLink

MAVLINK · 3 H · INTERMEDIATE

The message protocol small drones speak, and its Python library: the protocol guide · pymavlink

FREE

### QGroundControl user guide ↗

DRONECODE · 1–2 H · BEGINNER

A free ground station to watch your simulated drone.

FREE

### ExpressLRS: signal health ↗

EXPRESSLRS · 1 H · BEGINNER

Link quality, sensitivity and range, explained by an open radio project.

FREE

### PySDR: a guide to SDR and DSP using Python ↗

MARC LICHTMAN · 3–10 H · INTERMEDIATE

Link budgets worked in Python: path loss, antenna gain, signal to noise.

ALSO A 3D world with a camera: ArduPilot with Gazebo · PX4 simulation · MAVSDK · Companion computers · CRSF protocol

at home

Start ArduPilot SITL and connect QGroundControl. Write thirty lines of pymavlink that log HEARTBEAT, ATTITUDE and GLOBAL_POSITION_INT to a CSV file, then arm and take off to ten metres.

☐ DONE WHEN Your script and the ground station show the same altitude.

13 IN THE AIR, ON EVERY FRAME

# Estimation and navigation

| IN | OUT |
| --- | --- |
| noisy sensors | → where the drone is, and where the target is |

Every sensor lies a little. A filter blends them into one best guess of where the drone is, and geometry turns a pixel into a place on the ground. Without this, a detection has no coordinates.

start here

FREE **Kalman and Bayesian Filters in Python** ↗
ROGER LABBE · 20–40 H · BEGINNER
Kalman filters built up in Python notebooks that run in the browser.

FREE **kalmanfilter.net tutorial** ↗
ALEX BECKER · 3–5 H · BEGINNER
The same idea from plain numbers, one step at a time.

FREE **REP 103: units and coordinate conventions** ↗
ROS · 30 MIN · BEGINNER
Coordinate frames and units, on one page.

FREE **Inertial Navigation Primer** ↗
VECTORNAV · 4–6 H · INTERMEDIATE
Inertial sensors, attitude, error budgets.

FREE **An Introduction to GNSS** ↗
NOVATEL · 3–4 H · BEGINNER
How satellite positioning works, and where it fails.

FREE **Small Unmanned Aircraft: book draft and simulator** ↗
BEARD AND MCLAIN, BYU · 30+ H · INTERMEDIATE
A full textbook draft with a Python simulator: frames, dynamics, autopilot.

at home

Take the CSV from the telemetry project. Add noise to the altitude, then write a one-dimensional Kalman filter that recovers it. Plot the true, the noisy and the filtered line.

☐ DONE WHEN You can explain why the filter trusts a sensor less when you raise its noise.

14 IN THE AIR, ON EVERY FRAME

# The whole system

| IN | OUT |
| --- | --- |
| everything above | → one decision, on time |

The best model behind the wrong lens, with a late telemetry packet, on a board that overheats, is a failed system. Someone has to hold the whole chain in their head, write down what each part must deliver, and test the joints. It is rarely the person who only trains models.

start here

FREE

## NASA Systems Engineering Handbook

NASA · 3 H FOR CHAPTER 2 · BEGINNER

Read chapter 2: requirements, trade-offs, life cycle. Written for rockets, true for a drone.

FREE

## GStreamer tools

GSTREAMER · 2 H · BEGINNER

How video moves through a small computer: build a pipeline from the command line.

FREE

## ROS 2: first steps

OPEN ROBOTICS · 4–6 H · BEGINNER

How robots split software into parts that talk to each other.

FREE

## 16.842 Fundamentals of Systems Engineering

MIT OPENCOURSEWARE · 10–30 H · ADVANCED

A full graduate course, for the day you want the method.

at home

Draw your own version of the map for one of your projects. On every arrow write what goes through it, how often, and what happens when it fails.

☐ DONE WHEN You can name the weakest link of your own project and prove it with a measurement.

15 YOU

# Six home projects

One ladder through the whole map. Each step takes an evening or a weekend, and each one ends with something you can show.

## 1 A board in the browser

YOU NEED A browser.

BUILD An ESP32 in Wokwi: an LED, a sensor on I2C, readings over UART.

PROOF The logic analyser shows the address of your sensor.

READ 11

## 2 Classic vision, twice

YOU NEED A laptop with a webcam.

BUILD A tracker of a bright object in Python. Then the same in C++ with CMake.

PROOF Both print the same coordinates, and you know which is faster.

READ 04 06

## 3 How many pixels

YOU NEED A Raspberry Pi with a camera, or any camera with a specification sheet.

BUILD Capture frames, measure the field of view, predict the pixels on a target at 2, 5 and 10 metres. Then draw the brick map from chapter 5 for the view from your window.

PROOF Your prediction is within ten percent of the measurement.

READ 05 11

15 Six home projects · continued

## 4 Your own detector

YOU NEED A free GPU on Kaggle or Colab.

BUILD A nano detector trained on VisDrone. Read the metrics, collect the failures.

PROOF One page: the confusion matrix and twenty images the model got wrong.

READ 07 08 09

## 5 Make it fit

YOU NEED A laptop. A Raspberry Pi if you have one.

BUILD Export to ONNX, quantise, measure before and after.

PROOF A table: megabytes, frames per second, accuracy.

READ 10

## 6 Close the loop

YOU NEED A laptop with Linux, or WSL on Windows.

BUILD A simulated drone. Read its telemetry, draw altitude and attitude over the video, send every detection to the ground station as a message.

PROOF The detection appears on the ground station, next to the position of the drone.

READ 12 13 14

**then**

Put each project on GitHub with a README, a short video and honest numbers, including the place where it fails. That place is what an interviewer will ask about.

16 YOU

# Getting in

The entry point is lower than it looks, and the field needs more of us. Teams hire people who have built something small that works and can say where it breaks.

## WHERE THE WORK IS

start here

FREE

### DOU DefTech jobs

DOU · JOB BOARD · IN UKRAINIAN

The largest board of defence-tech jobs in Ukraine. Filter by AI / ML and read what teams ask for.

FREE

### Who defence tech hires: the numbers

DOU · 20 MIN · IN UKRAINIAN

A data study of who gets hired, and for what.

FREE

### Brave1

UKRAINE'S DEFENCE-TECH CLUSTER · HUB · IN UKRAINIAN

The state defence-tech cluster: what is being built and funded.

ALSO DOU jobs, filtered by AI / ML

## BUILD WITH OTHERS

FREE

### European Defense Tech Hub

EDTH · HACKATHONS ACROSS EUROPE

Weekend hackathons with a computer vision and edge AI track. Applications are reviewed: about · calendar

FREE

### CASSINI Hackathons

EUROPEAN UNION · HACKATHON

An EU hackathon on space data, with sites in Poland and Ukraine. No space experience needed.

ALSO EUDIS Defence Hackathon: watch for the next call

16 Getting in · continued

## LEARN THE AIRFRAME

FREE

### Інженер БПЛА. Базовий курс

PROMETHEUS, VICTORY DRONES, NAU · 6 H · BEGINNER · IN UKRAINIAN

A basic course in UAV engineering: aerodynamics, onboard electronics, navigation, failures in the field.

FREE

### Народний FPV

PROMETHEUS, VICTORY DRONES · 8 H · BEGINNER · IN UKRAINIAN

Build an FPV drone yourself. The course is free, the parts are not.

ALSO How to become a UAV engineer (DOU, in Ukrainian)

## WHAT IS DIFFERENT IN DEFENCE

FREE

### Exporting dual-use items

EUROPEAN COMMISSION · 20 MIN

Software and know-how count as dual-use goods. Read this before you publish or travel with your work.

FREE

### Brave1 Dataroom announcement

MINISTRY OF DEFENCE OF UKRAINE · 5 MIN

Why work datasets and weights stay closed.

FREE

### Think before you share

UK MINISTRY OF DEFENCE · 10 MIN

Old, short and clear: photos, geotags, job details.

## FOR WOMEN

ALSO Perspektywy Women in Tech (Poland): mentoring, scholarships · Women in Defense Tech · Women4Cyber mentorship

portfolio

Job adverts repeat the same list: Python and C++, OpenCV, PyTorch, ONNX, inference on a small board, tests on a real device. Two or three projects with honest numbers say more than a certificate. Labelling, dataset review and field tests are real doors into vision teams.

☐ DONE WHEN You have sent one application, or signed up for one hackathon.

— SMALL PRINT

# Credits

Text and layout © 2026 Daria Chuprina. You may share and adapt this roadmap under the Creative Commons Attribution 4.0 licence.

The shape on the cover is the mask of one vehicle in a synthetic frame, 415 by 222 pixels in the original. 3D model: Abrams M1A1 [Free] by DerpDude, CC BY 4.0. Typeface: IBM Plex, SIL Open Font Licence.

Found a dead link, or a better resource? Tell me: linkedin.com/in/dchuprina.

The latest version and the source files: chuprinadaria.github.io/wawtech-2026-roadmap.

## THANK YOU

To Anar Lavrenov, for the science. Separately, to the CV + ML department at BBTech and to the heads of departments. To the whole open-source community, and to everyone who adds to it, however small. And to all the people with unpronounceable surnames whose dissertations and papers I read through long nights.

## WHAT IS NOT HERE

Paid courses, programmes that only companies can join, and anything I could not open. Work datasets, work models and work footage are not here either, and they will not be. This is MilTech, baby 😉