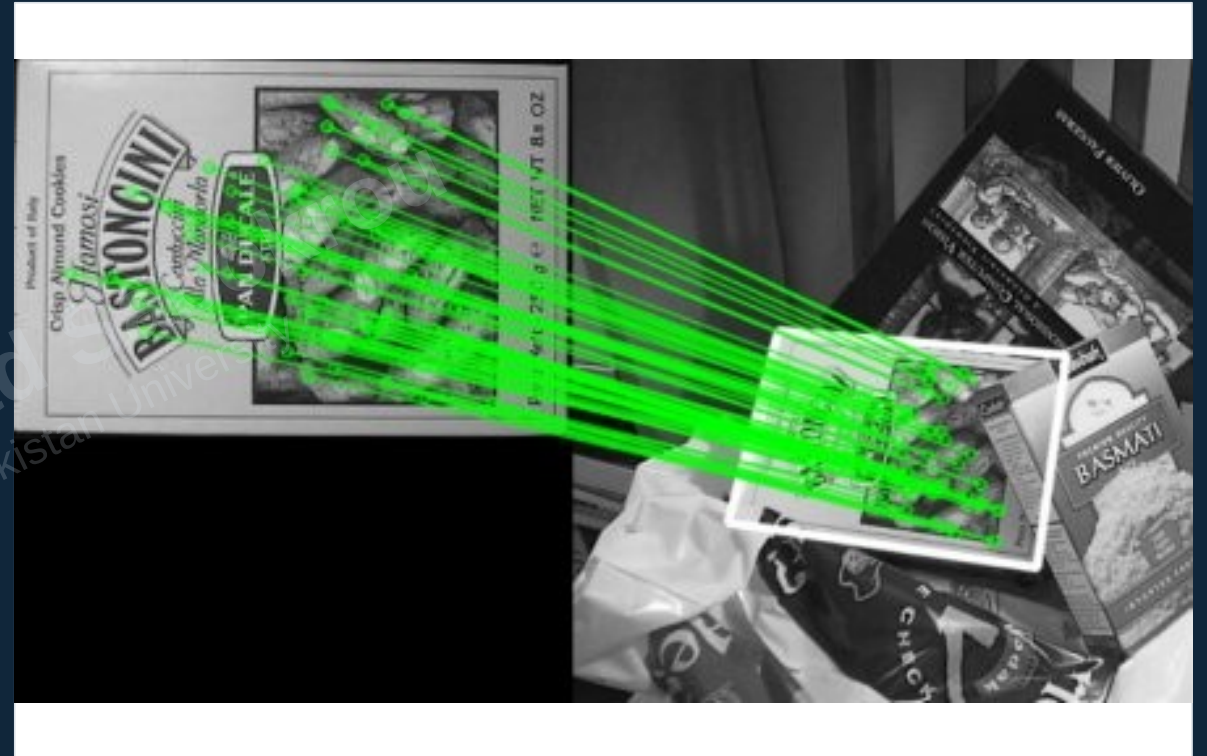


Features & matching

Same place. Different pixels.

Mohammad Sabokrou
New Uzbekistan University

120 minutes • 2 × 5-minute activities • Visual reasoning first



HOW CAN A SYSTEM FIND THE SAME BOOK AGAIN?

From edges to corresponding points

Week 3

Smooth → gradients → edges

The missing question

Which exact point is this?

Week 4

Locate → describe → match → verify

Week 3 measured local change. Week 4 uses it to find the same place again.

Four goals you can see

LOCATE



A point is marked at a repeatable image location.

DESCRIBE



[18, 0, 5, ...]
local appearance

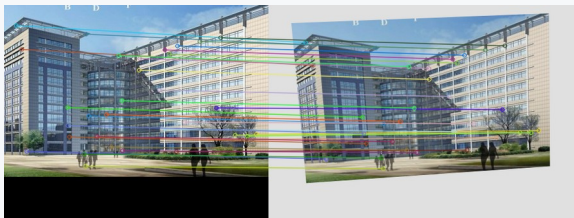
The surrounding patch becomes a vector of numbers.

DIAGNOSE



Double edges reveal a deliberately wrong alignment.

VERIFY



Keep pairs supported by the same geometry.

Explain the evidence, not only the algorithm name.

Our revised two-hour route

0–8	Problem, visual goals and one pipeline
8–35	Patch motion → gradients → Harris → feature quality
35–40	ACTIVITY 1 • Recall checkpoint + find the point
40–58	Scale, orientation, SIFT and ORB
58–63	Break
63–77	Distances, ratio test and ambiguous matches
77–96	Homography, RANSAC and verification
96–101	ACTIVITY 2 • Can the fitted model mislead us?
101–113	Warping, blending and panorama failures
113–120	Saved demonstration, lab bridge and recap

Both activities stay five minutes; added Harris examples replace some longer demonstration time.

A match links the same physical point



OpenCV feature-introduction example: some small patches have many possible homes.

Coordinates can change

A scene point might move from (80, 40) to (135, 65).

Identity should not change

We want the same physical location, not merely a similar patch.

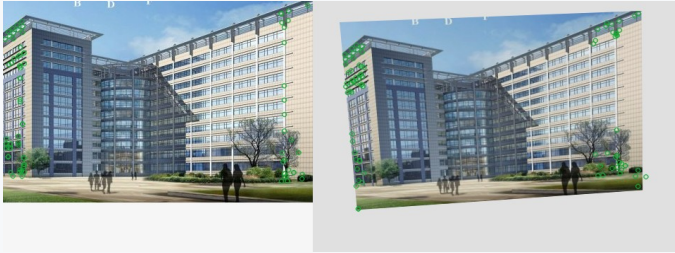
Why pixels alone struggle

Rotation, scale, lighting and occlusion alter appearance.

Similar-looking pixels are evidence, not proof of identity.

The same building through all six steps

1 • Detect



Mark candidate locations in A and B.

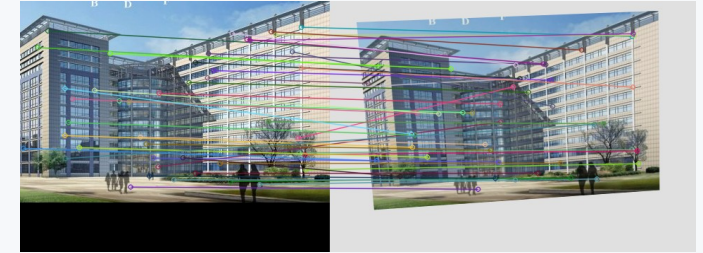
2 • Describe



patch
→ 128 values

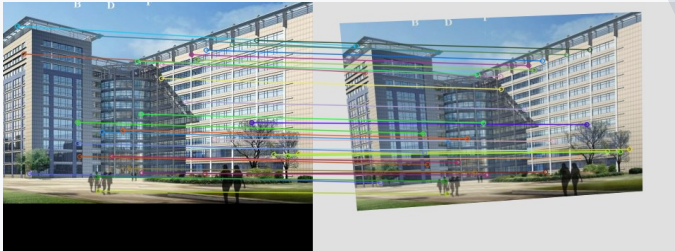
Describe a patch around each mark.

3 • Match



Compare vectors; propose pairs.

4 • Verify



RANSAC keeps a consistent subset.

5 • Warp



Map B into A's coordinate frame.

6 • Blend



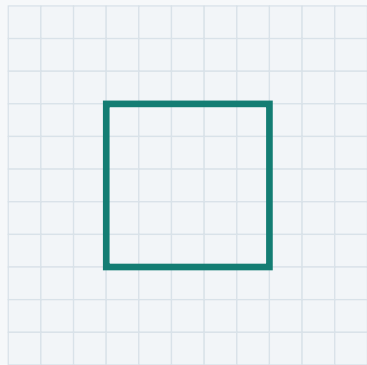
Combine aligned intensities in overlap.

Controlled example: B is a transformed copy of A, not a second independently captured photograph.

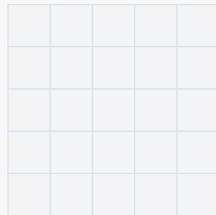
The image stays still. We move a window.

Flat region

WHOLE IMAGE • simplified intensity scene



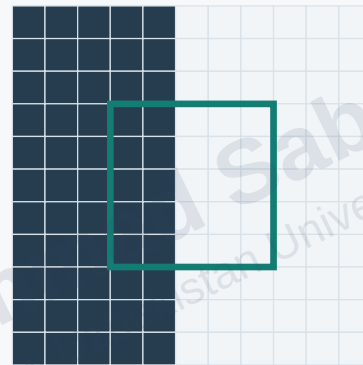
Selected
pixels →



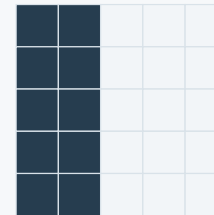
Little structure to locate.

Edge

WHOLE IMAGE • simplified intensity scene



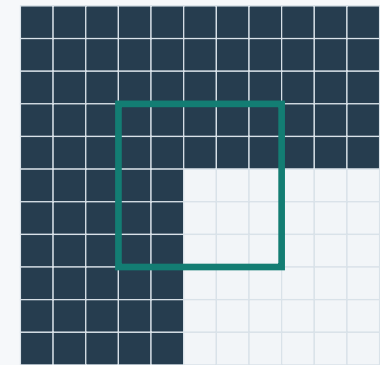
Selected
pixels →



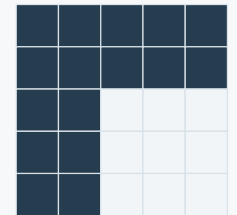
Boundary, but ambiguity along it.

Corner

WHOLE IMAGE • simplified intensity scene



Selected
pixels →

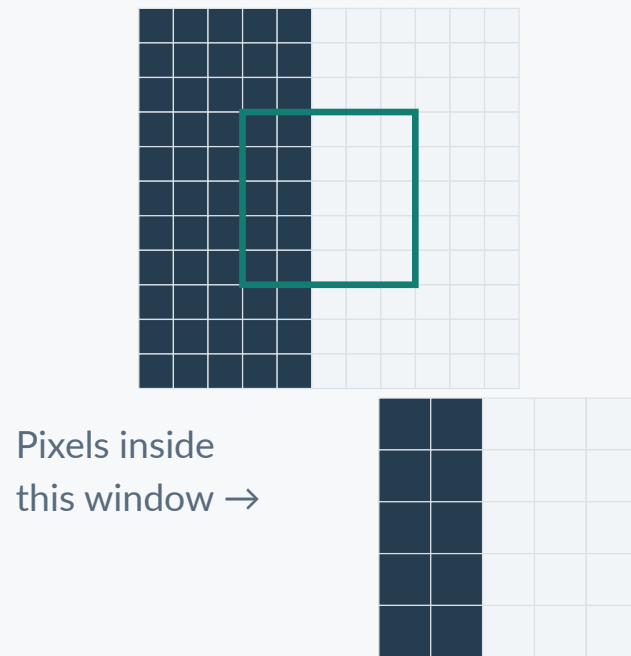


Two-direction positional evidence.

Image = the whole grid. Patch = the small group of pixels inside the outlined window.

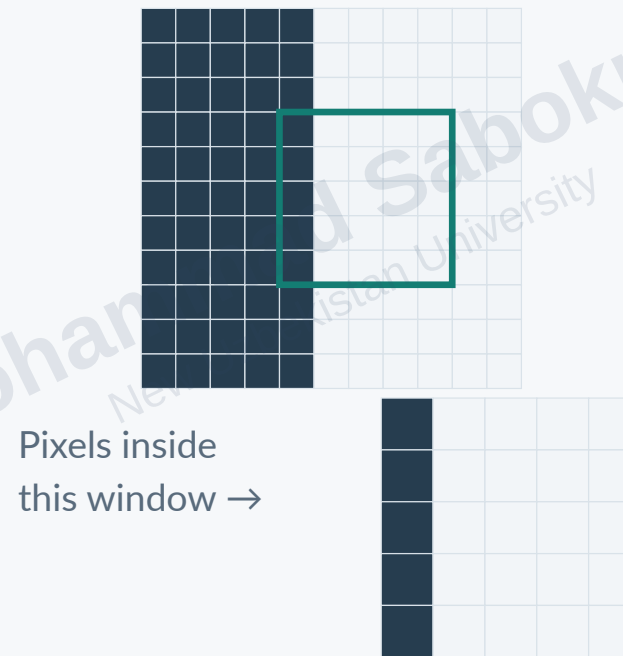
A straight edge: right changes, down does not

Reference window



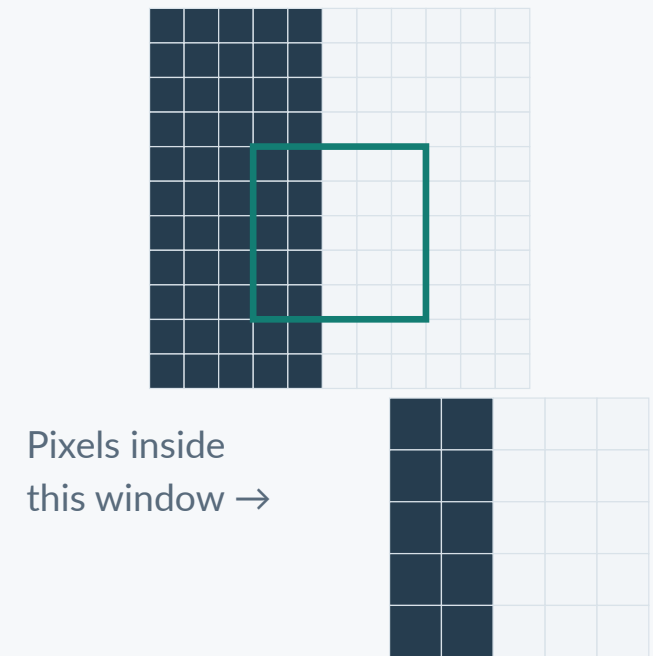
Reference

Move 1 pixel right



5 of 25 pixel values change

Move 1 pixel down

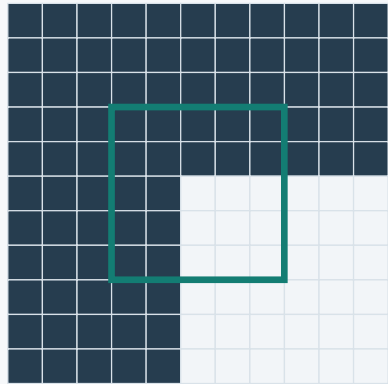


0 of 25 pixel values change

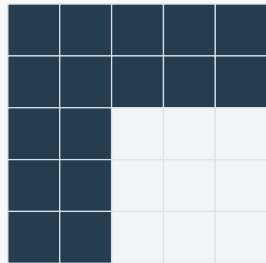
Today's demonstration uses a 5×5 -pixel window. This is an example size, not a Harris rule.

A corner: right and down both change the patch

Reference window

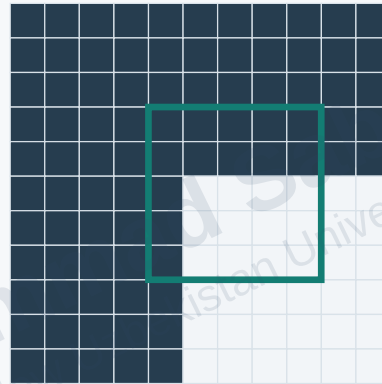


Pixels inside
this window →

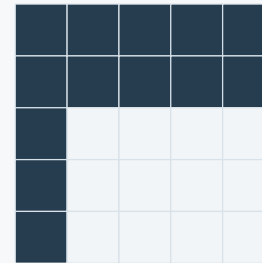


Reference

Move 1 pixel right

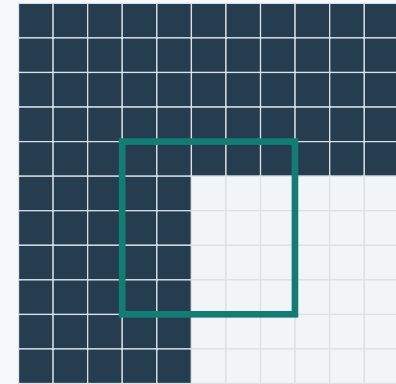


Pixels inside
this window →

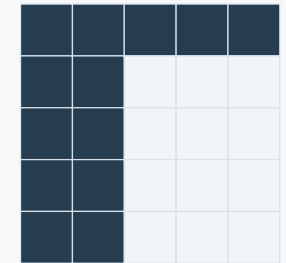


3 of 25 pixel values change

Move 1 pixel down



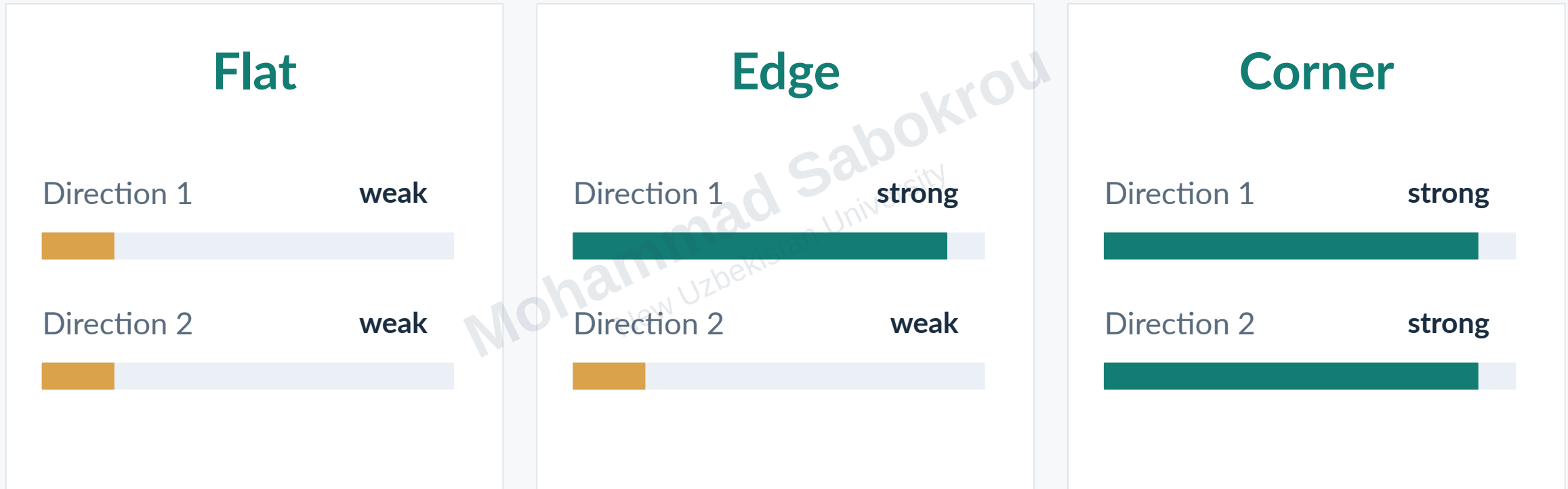
Pixels inside
this window →



3 of 25 pixel values change

Today's demonstration uses a 5×5 -pixel window. This is an example size, not a Harris rule.

Harris asks about change in two directions



The two strengths are the eigenvalues of M ; no eigenvalue calculation is needed today.

Large change in only one direction suggests an edge, not a well-localised point.

First turn neighbouring brightness into gradients

One shared numerical recipe

$$I_x = (\text{right} - \text{left}) / 2$$

$$I_y = (\text{below} - \text{above}) / 2$$

$$\text{Example: } (2 - 0) / 2 = 1$$

0	0	0	0	0
0	0	0	0	0
0	0	2	2	2
0	0	2	2	2
0	0	2	2	2

0 = dark
2 = bright
Teal = nine
evaluation sites

The outer 5×5 supplies neighbours.
Compute gradients at the inner 3×3 sites.
Use all nine sites, with weight 1.

$$A = \sum I_x^2 \quad B = \sum I_x I_y \quad C = \sum I_y^2$$

I_x and I_y are brightness changes, not coordinates. We sum products of these changes.

Flat: intensity information, little position evidence

Image patch I

2	2	2	2	2
2	2	2	2	2
2	2	2	2	2
2	2	2	2	2
2	2	2	2	2

Horizontal I_x

0	0	0
0	0	0
0	0	0

Vertical I_y

0	0	0
0	0	0
0	0	0

$$A = \sum I_x^2 = 0 \times 1^2 = 0$$

$$C = \sum I_y^2 = 0 \times 1^2 = 0$$

$$B = \sum I_x I_y = 0 \times 1 = 0$$

$M =$

0	0
0	0

Centred differences at the nine inner sites; all weights = 1.

$$\det(M) = 0 \times 0 - 0^2 = 0 \quad \text{trace}(M) = 0$$

$$\text{With } k = 0.04: R = 0 - 0.04 \times 0^2 = 0$$

A flat region describes brightness or material, but is a poor point to localise precisely.

Edge: strong across the boundary, weak along it

Image patch I

0	0	2	2	2
0	0	2	2	2
0	0	2	2	2
0	0	2	2	2
0	0	2	2	2

Horizontal I_x

1	1	0
1	1	0
1	1	0

Vertical I_y

0	0	0
0	0	0
0	0	0

$$A = \sum I_x^2 = 6 \times 1^2 = 6$$

$$C = \sum I_y^2 = 0 \times 1^2 = 0$$

$$B = \sum I_x I_y = 0 \times 1 = 0$$

$M =$

6	0
0	0

Centred differences at the nine inner sites; all weights = 1.

$$\det(M) = 6 \times 0 - 0^2 = 0 \quad \text{trace}(M) = 6$$

$$\text{With } k = 0.04: R = 0 - 0.04 \times 6^2 = -1.44$$

An edge is useful boundary evidence; sliding along it can leave location ambiguous.

Corner: two directions contribute to the matrix

Image patch I

0	0	0	0	0
0	0	0	0	0
0	0	2	2	2
0	0	2	2	2
0	0	2	2	2

Horizontal I_x

0	0	0
1	1	0
1	1	0

Vertical I_y

0	1	1
0	1	1
0	0	0

$$A = \sum I_x^2 = 4 \times 1^2 = 4$$

$$C = \sum I_y^2 = 4 \times 1^2 = 4$$

$$B = \sum I_x I_y = 1 \times 1 = 1$$

$M =$

4	1
1	4

Centred differences at the nine inner sites; all weights = 1.

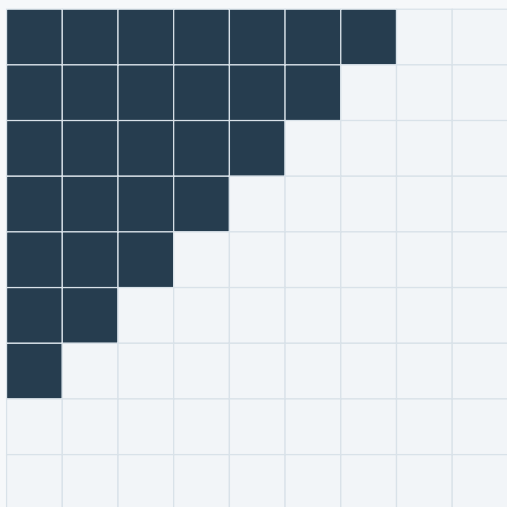
$$\det(M) = 4 \times 4 - 1^2 = 15 \quad \text{trace}(M) = 8$$

$$\text{With } k = 0.04: R = 15 - 0.04 \times 8^2 = 12.44$$

A corner supplies precise local position evidence; matching still needs a distinctive patch.

Why do we keep the cross-term B?

A diagonal edge can fool a two-sum test



Gradients can all point in the same diagonal direction.

Example: (1, 1), repeated 3 times.

$$A = 3 \quad C = 3 \quad B = 3$$

$$M = \begin{bmatrix} 3 & 3 \\ 3 & 3 \end{bmatrix}$$

$$\det(M) = 3 \times 3 - 3^2 = 0$$

$$\text{trace}(M) = 6$$

$$R = 0 - 0.04 \times 6^2 = -1.44$$

The tangent direction is still weak.

Illustrative gradient set; not a new Sobel calculation.

Large I_x^2 and I_y^2 do not by themselves prove two independent directions.

Why this score, rather than total change alone?

Class question: why $R = \det(M) - k \cdot \text{trace}(M)^2$?

$\det(M)$

$\lambda_1 \lambda_2$: both strengths must contribute.

$\text{trace}(M)$

$\lambda_1 + \lambda_2$: total directional variation.

k

Penalty strength, not confidence or a probability.

Case	$\det(M)$	$\text{trace}(M)$	R at $k = 0.04$
Flat	0	0	0
Edge	0	6	-1.44
Corner	15	8	12.44

The product rewards two strong directions; the trace-squared penalty discourages imbalance.

Harris in one compact, labelled recipe

$$R = \det(M) - k \cdot \text{trace}(M)^2$$

$$M = \begin{bmatrix} A & B \\ B & C \end{bmatrix}$$

$$A = \sum w I_x^2 \quad B = \sum w I_x I_y$$

$$C = \sum w I_y^2$$

$$\det(M) = A \cdot C - B^2$$

$$\text{trace}(M) = A + C$$

$k = 0.04$ in our examples

$w = 1$ in our worked sums

Flat → weak score. Edge-like → often negative. Corner-like → sufficiently strong positive peak.

M is a summary of many nearby gradients. R is one score for the window location.

A response map is not yet a list of corners

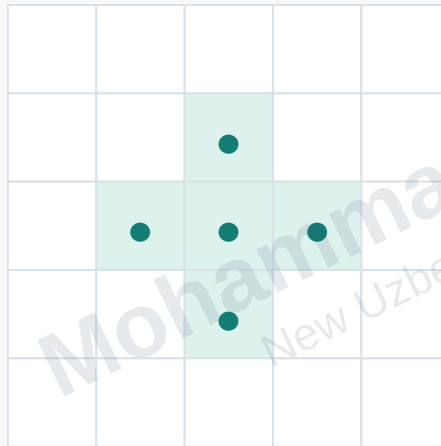
1 • Response map

0	0	1	0	0
0	3	8	5	0
1	7	12	8	1
0	3	7	4	0
0	0	0	0	0

Several nearby pixels score highly.

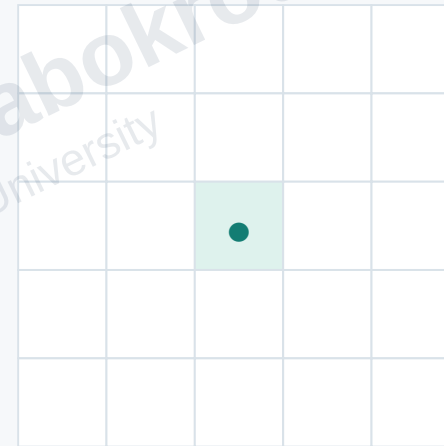
Illustrative map and refinement coordinates; not a second measurement of the preceding toy patch.

2 • Threshold > 6



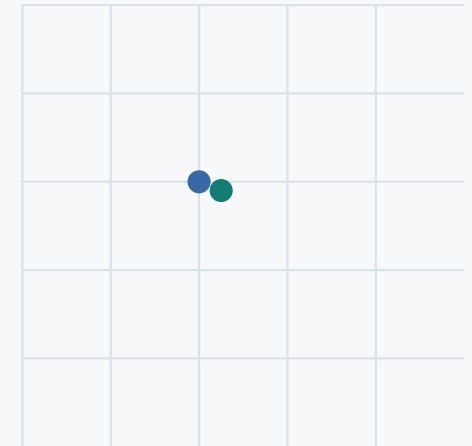
Keep strong candidates.

3 • Local maximum



Keep 12, not the entire cluster.

4 • Refine position



$(2, 2) \rightarrow (2.25, 2.10)$

Fit a location between sample pixels.

Threshold removes weak responses; non-maximum suppression removes nearby duplicates.

More detections does not mean better features



Would 500 points on the same repeated windows be enough?

Repeatability

Does the same scene structure reappear across views?

Distinctiveness

Does its surrounding patch differ from competing patches?

Localisation accuracy

Are the detected coordinates close to the true image position?

Spatial distribution

Do useful points cover the overlap, rather than one tiny cluster?

Judge repeatability, distinctiveness, localisation accuracy and spatial distribution—not count alone.

A point locates. A surrounding patch describes.

The descriptor comes from the neighbourhood around the keypoint—not the corner pixel alone.



[18, 0, 5, ...]
appearance vector

Dot = one position.

Outline = nearby pixels used for description.

Detector

Find a location.

Descriptor

Summarise a patch.

Matching

Compare vectors.

Verification

Check geometry.

Recall one detail. Then explain the best point.

LECTURE CHECK • ONE SHORT ANSWER FROM MEMORY

Lecture check — from memory: what were the dimensions, in pixels, of the window in our earlier sliding-window demonstration? Give one short answer; do not reopen the slides.

REASONING QUESTION • KEEP THE ORIGINAL TASK

Two photographs show the same wall after a small camera movement. Consider three small patches: A — a blank wall; B — the middle of a straight vertical edge; C — an isolated L-shaped corner. Which patch is easiest to relocate precisely? Give one reason that compares a small horizontal shift with a small vertical shift. Answer in two sentences, using your own reasoning.

Class Engage • CE 8480 • Keep lecture recall separate from conceptual reasoning

First answer: recall 15 seconds + reasoning 75 seconds. Both stay within the original 90 seconds.

Read feedback. Improve the reason.

FEEDBACK 30S → REVISION 60S

Compare moving a patch along an edge with moving it across the edge. Which movement could leave the patch looking almost unchanged? Check that your explanation accounts for both horizontal and vertical position, rather than only brightness or the strength of one intensity change.

Revise or keep your answer to the reasoning question only. State what changed in your explanation, or why your original reasoning still holds. Your first lecture-check answer remains the recorded recall response.

Answer

Feedback

Revise

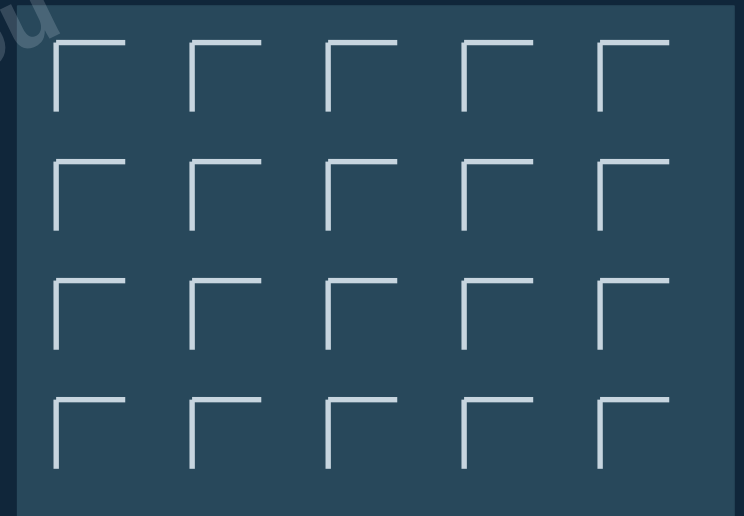
Changed case

Explain

Changed case: twenty identical corners

INDEPENDENT CHANGED CASE • 60 SECONDS

Changed case — work independently. The L-shaped corner is now one of twenty identical corners in a tiled wall. A detector finds every corner very accurately. Does this guarantee that the same physical corner can be identified in the other photograph? Explain, and name one extra visual cue that would help.



20 repeated corners • schematic

Answer

Feedback

Revise

Changed case

Explain

Localisation is not identity

EXPLANATION · FINAL 60 SECONDS

Initially, C is easiest to localise: moving the small window horizontally or vertically changes the corner pattern. A vertical edge leaves movement along the edge uncertain; a flat patch leaves both directions uncertain. In the changed case, all twenty corners can be well localised but still look alike. Localisation and identity are different problems. A distinctive surrounding mark or wider context helps identify the same physical location. Lecture check: the demonstration used a 5×5 -pixel window, or 25 pixels. That size was chosen for this example, not required by Harris.



Answer

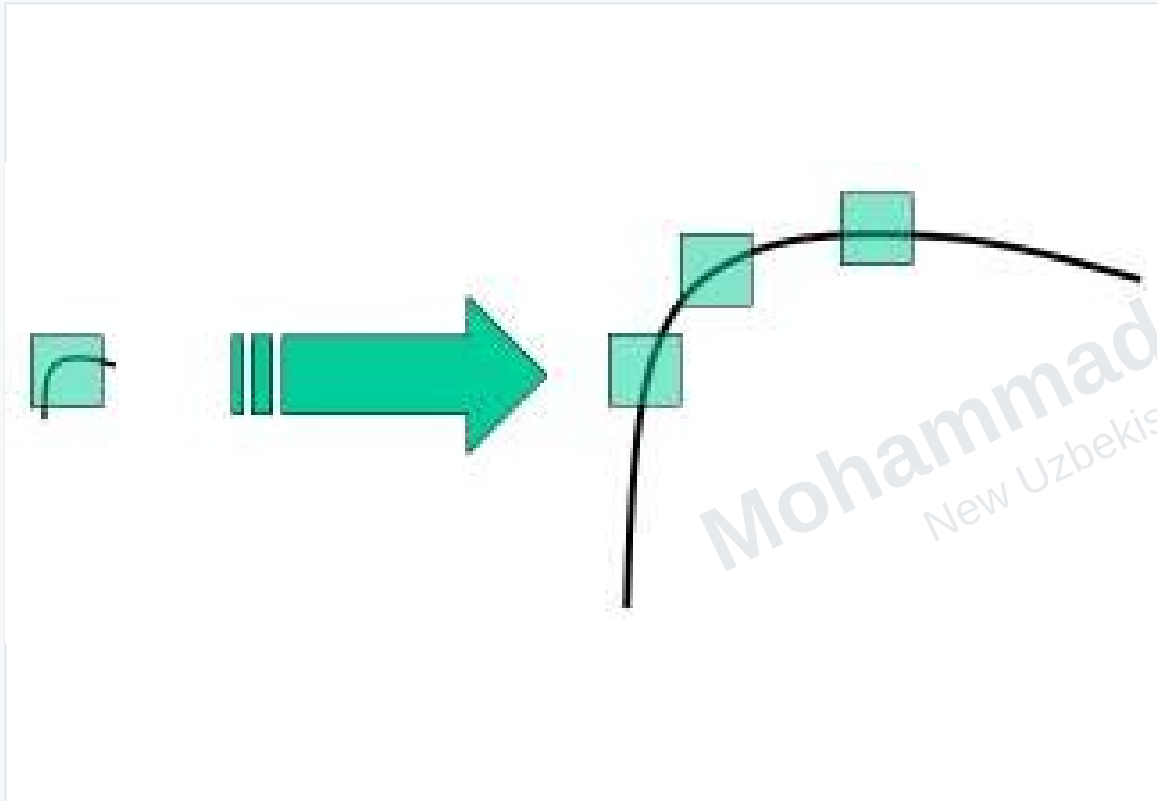
Feedback

Revise

Changed case

Explain

A fixed-size window does not fit every scale



Move closer

A small feature occupies more pixels.

Keep the same window

It may cover only a smooth part of a larger structure.

Better approach

Search over multiple sizes and record a useful scale.

OpenCV SIFT introduction: the same fixed window sees different structure after enlargement.

Scale should be part of the feature, not an accidental camera setting.

A keypoint has a small “passport”



position • radius • direction

x, y

The position in image pixels.

Scale

How large a neighbourhood to describe.

Orientation

Which direction counts as the local reference.

Descriptor

A separate numerical summary of that neighbourhood.

Position locates the patch; scale and orientation define its local frame.

SIFT in four plain-language steps

1 • Search sizes

Find candidates over image position and scale.

2 • Keep stable points

Refine location; reject weak and edge-like candidates.

3 • Pick a direction

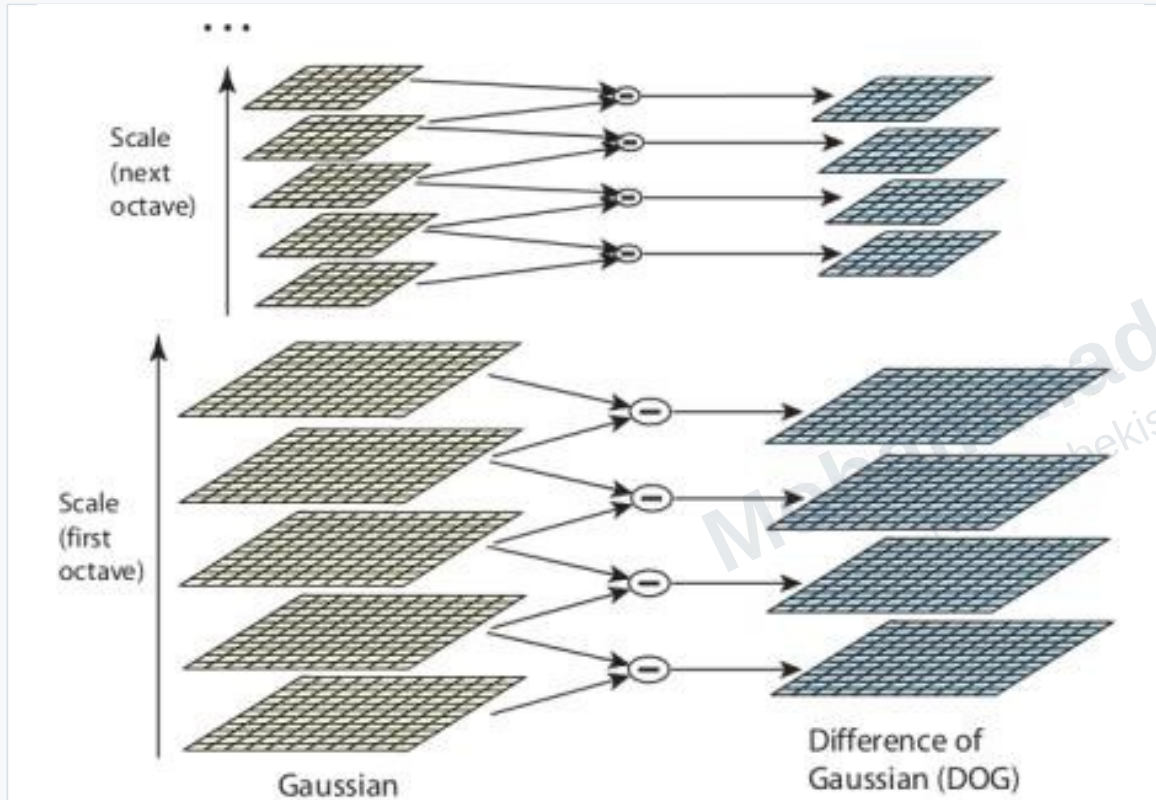
Assign a dominant local orientation.

4 • Describe the patch

Record the distribution of local gradient directions.

SIFT detects features and describes them; it is not just “Harris plus a vector.”

Look at a family of blurred images



Gaussian scale space and differences of nearby blur levels. OpenCV illustration.

Gaussian images

Smooth the image by different amounts.

Difference of Gaussians

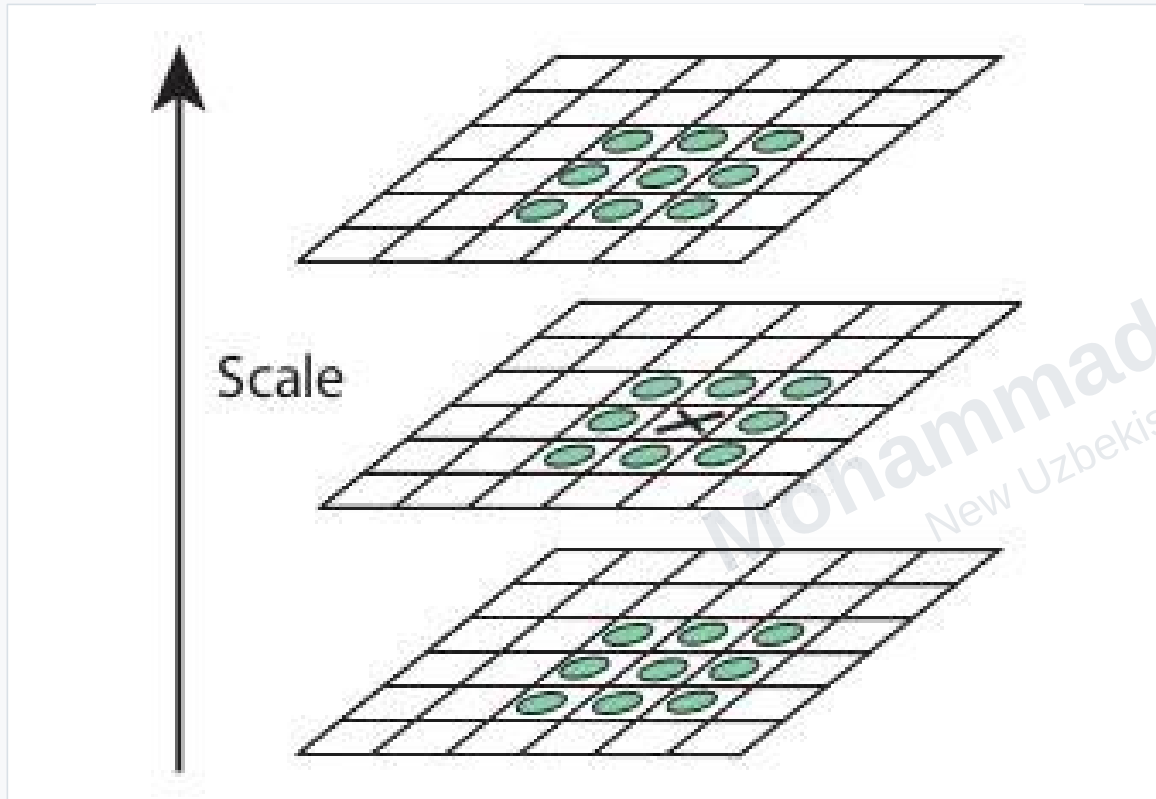
Subtract neighbouring blur levels.

Octaves

Repeat at a smaller image size to cover larger structures.

The useful question is “at what size is this structure distinctive?”

A candidate must stand out across space and scale



A candidate is compared with neighbours in its own level and adjacent levels.

Across space

Compare nearby pixels in the current level.

Across scale

Also compare the levels above and below.

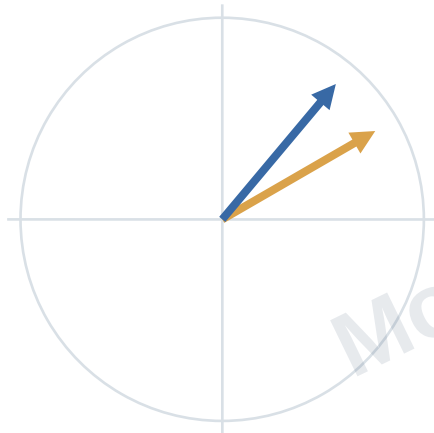
Then check stability

Reject weak contrast and poorly localised edge responses.

A local extremum is a candidate, not automatically a final keypoint.

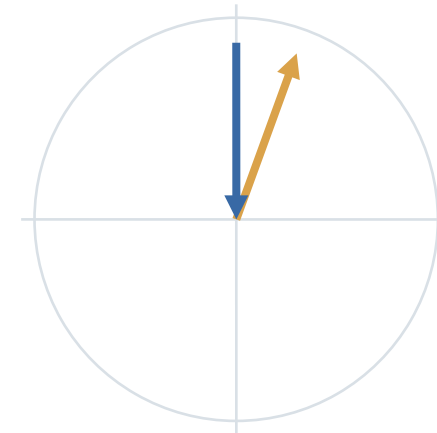
Describe directions relative to the feature

Original patch



$$50^\circ - 30^\circ = 20^\circ$$

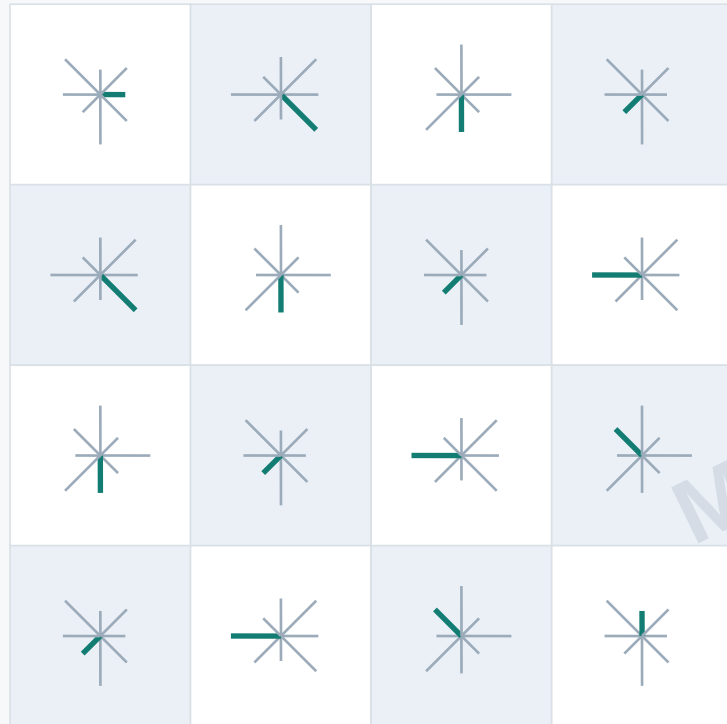
Image rotated by 40°



$$90^\circ - 70^\circ = 20^\circ$$

Subtracting the local reference direction can cancel a shared rotation.

A SIFT descriptor is a pattern of directions



16 spatial cells

8 bins
per cell

128

descriptor values

Schematic gradient bins; not measured data.

4×4 spatial cells $\times$ 8 orientation bins = 128 values.

Normalisation reduces some lighting effects

Before gain

$$(3, 4) \div 5 = (0.6, 0.8)$$

After ×2 gain

$$(6, 8) \div 10 = (0.6, 0.8)$$

More tolerant does not mean immune to shadows, clipping or occlusion.

ORB asks many tiny yes/no questions

ONE COMPARISON



p = 30



q = 90

30 < 90 → 1

Find and orient the patch

FAST finds candidate points; a pyramid handles size. ORB assigns orientation.

Compare two sample locations

In the oriented patch, is intensity at p smaller than at q?

Store binary answers

30 < 90 → 1. Repeating selected tests makes the descriptor.

ORB = Oriented FAST and Rotated BRIEF. Default: 256 bits (32 bytes), Hamming distance.

Choose a descriptor and the matching distance together

Method	Description	Distance	Practical focus
SIFT	128 numeric values	L2	Scale/orientation-normalised gradients
ORB (default)	256 bits / 32 bytes	Hamming	Compact binary tests; efficient matching

Use the descriptor's meaning, not merely the array data type, to choose a metric.

Five-minute break

05:00

Next: how do we decide whether two descriptions belong together?

Distance means “how different?”

L2 • numerical vectors

(1, 2) and (4, 6)

$$\sqrt{3^2 + 4^2} = 5$$

Hamming • binary tests

10110010 and 10010110

2 differing bits

Small descriptor distance suggests similarity, not guaranteed correspondence.

Nearest neighbours propose candidates

Query descriptor

One patch in image A

Image B candidates

distance 20 • distance 40 • distance 75

Candidate match

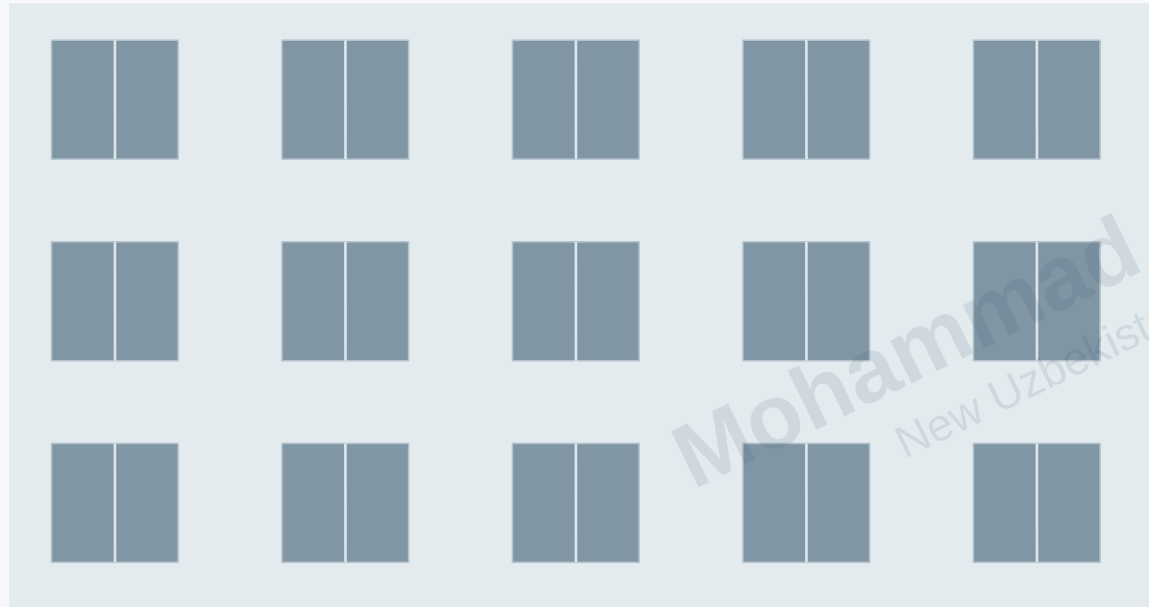
Choose the nearest, then ask whether it is distinctive.



Candidate lines in an OpenCV tutorial example.

Keep the best and second-best options so that you can inspect ambiguity.

Repeated windows create lookalikes



Candidate 1

$$d_1 = 40$$

Candidate 2

$$d_2 = 42$$

Interpretation

Two almost equally plausible descriptions.

“Best” is weak evidence when another candidate is almost as good.

The ratio test asks for a clear winner

d_1 = nearest distance d_2 = second-nearest distance

d_1	d_2	Ratio	Decision
20	40	0.50	Keep
40	45	0.89	Reject
15	20	0.75	Reject (strict <)
3	20	0.15	Keep

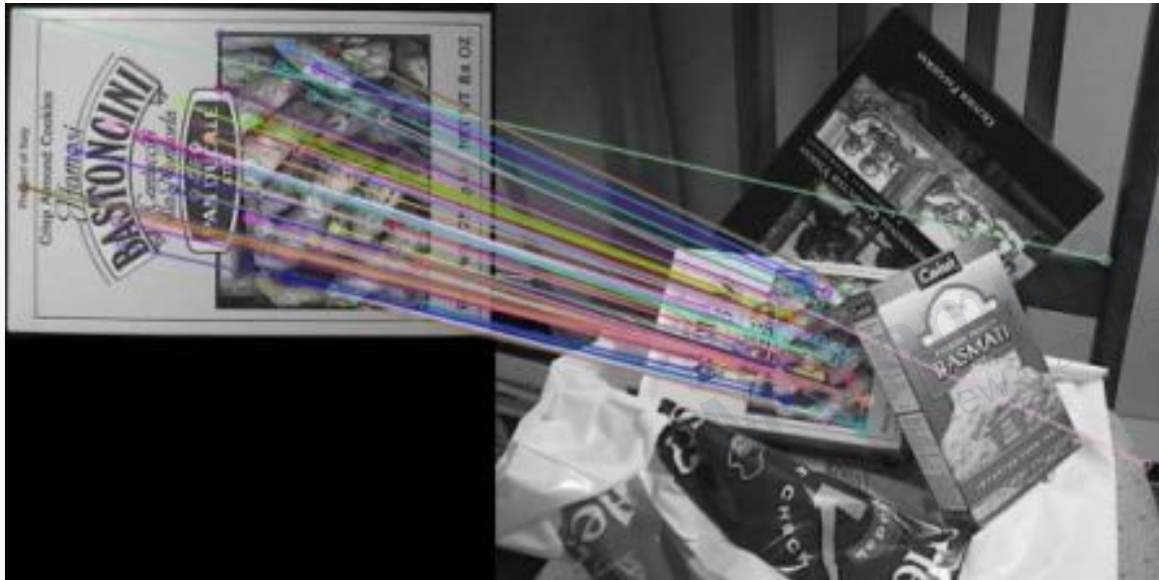
Accept only when $d_1 / d_2 < 0.75$ in this worked example.

Cross-check asks whether the choice is mutual



Mutual nearest neighbours can still be a mutually wrong pair.

Filtered matches are still hypotheses



OpenCV SIFT ratio-filtered matches; lines are candidate correspondences.

A useful filter

Remove many ambiguous descriptor matches.

A remaining question

Do the point locations agree with plausible geometry?

What not to conclude

A page full of lines is not an accuracy measurement.

Descriptor evidence and geometric evidence answer different questions.

Before blaming geometry, check the earlier stages

What you observe	Possible cause	Useful check
Very few keypoints	Low texture or heavy blur	Inspect the image and keypoint map
Many points, few matches	Wrong metric / limited overlap / large change	Check the metric and shared scene
Many similar candidates	Repeated appearance	Look for distinctive context

Count → inspect → form a hypothesis → change one thing.

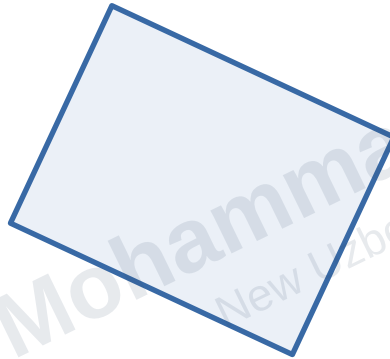
Geometry describes how points move together

Translation



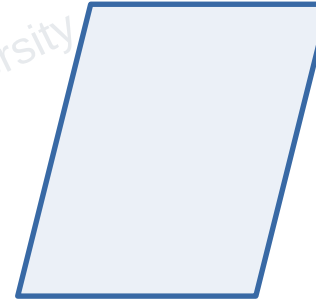
2 free parameters

Similarity



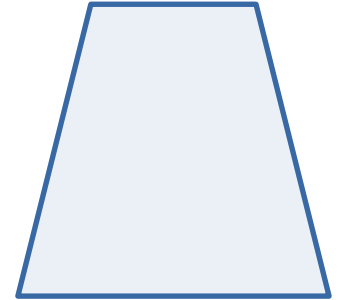
4 free parameters

Affine



6 free parameters

Homography



8 free parameters

Choose a transformation because of the scene, not because it has more parameters.

When can one homography explain the image pair?

Good model: a plane

A poster, a page or a nearly flat wall.

Good model: pure rotation

The camera rotates about a fixed optical centre.

Warning: depth + translation

Nearby and distant objects can move differently.

A flat surface or pure camera rotation is the key ideal case.

Homography: a three-by-three mapping recipe

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

$$[x, y, 1] \rightarrow [a, b, c]$$

Output point: $(a/c, b/c)$

h_{ij} : a coefficient in row i , column j of H

x, y : input coordinates • a, b, c : the matrix-product outputs
 c must be nonzero for a finite output point.

Multiply first. Divide by the third coordinate second.

Two small numerical examples

TRANSLATION

$$\begin{bmatrix} 1 & 0 & 20 \\ 0 & 1 & 10 \\ 0 & 0 & 1 \end{bmatrix}$$

$$(30, 40) \rightarrow (50, 50)$$

PROJECTIVE DIVISION

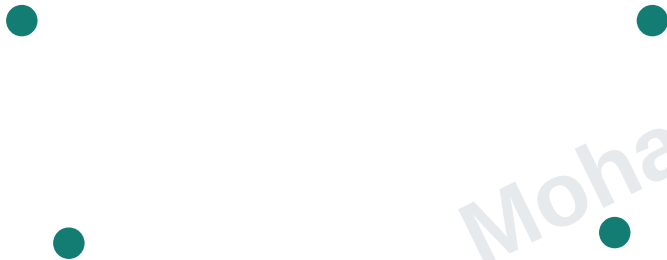
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0.01 & 0 & 1 \end{bmatrix}$$

$$(100, 50) \rightarrow (100, 50, 2) \\ \rightarrow (50, 25)$$

The final division matters whenever the third output is not 1.

Four pairs are a minimum, not a quality guarantee

Well spread



Constrains a 2D region.

On one line



Does not determine the mapping away from the line.

For a minimal four-pair estimate, use general position: no three collinear.

A few bad pairs can pull an ordinary fit away

Candidate matches

A mixture of useful pairs and bad pairs.

Fit everything equally

Large errors can distort the estimate.

Seek consensus

Find a plausible model supported by many pairs.

We need a method that can fit a model despite some incorrect candidates.

RANSAC: sample, fit, test, repeat

1

Sample 4
suitable pairs

2

Fit a
homography

3

Test all
candidate
pairs

4

Repeat and
compare
support

5

Refit using
the inliers

A consensus is a hypothesis to inspect, not an unconditional truth guarantee.

Reprojection error measures a geometric miss

Error = distance between the observed point and H's prediction

Observed	Predicted	Error (px)	At < 3 px
(50, 50)	(50, 50)	0	Inlier
(22, 12)	(20, 10)	2.83	Inlier
(42, 20)	(30, 20)	12	Outlier
(83, 64)	(80, 60)	5	Outlier

Here, error < 3 pixels means “inlier to this model.”

Trust several kinds of evidence together

1 Support

How many candidate pairs fit?

2 Coverage

Do inliers span the overlap?

3 Residuals

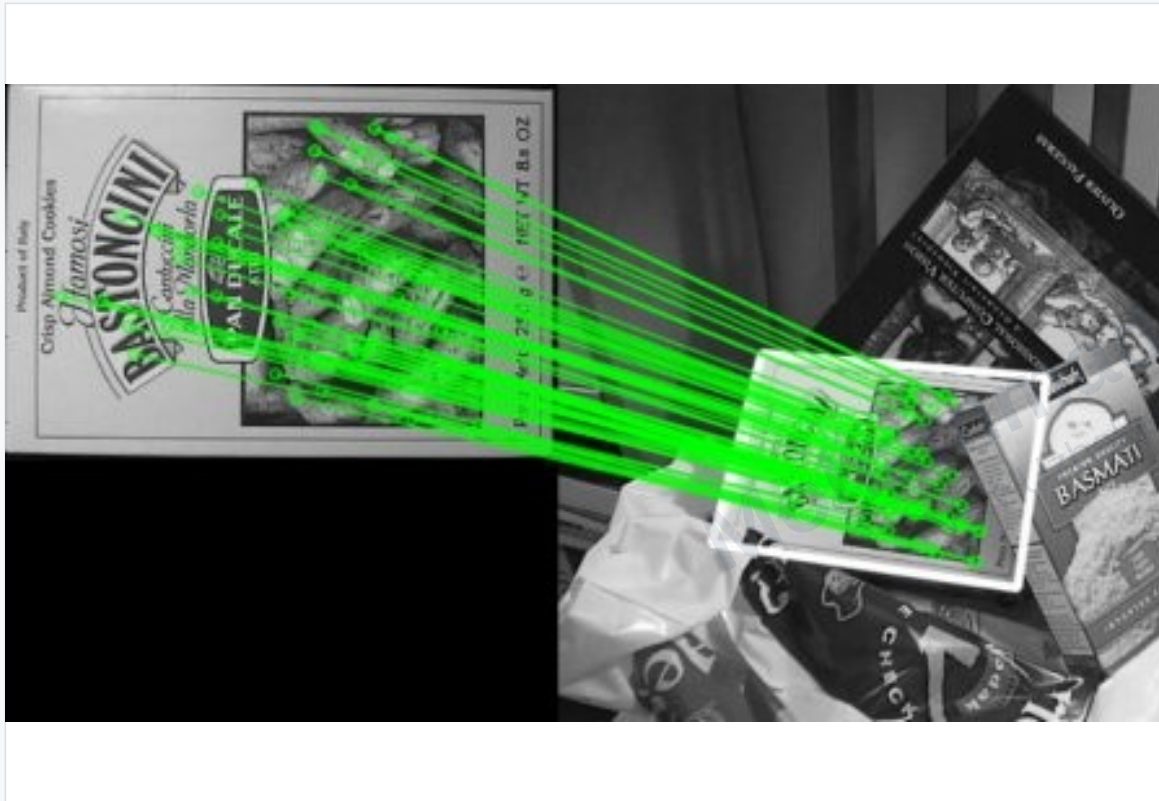
How far are predicted points from observations?

4 Plausibility

Do scene assumptions and unique landmarks agree?

High support + good spread + low error + plausible assumptions—not one number alone.

The geometric model predicts a boundary



OpenCV homography example: the reference boundary is projected into the scene.

What H does

Maps the corners of the reference image into the scene.

What the outline means

A model prediction supported by selected matches.

What still needs checking

Correct object identity, coverage and physical plausibility.

This resolves the opening example—but does not remove the need to verify it.

Can 80% inliers still be wrong?

INITIAL ANSWER · 90 SECONDS

A panorama uses repeated, identical windows. RANSAC finds that 60 of 75 candidate matches fit one homography within 3 pixels: an 80% inlier fraction. However, the panorama appears shifted by one window. A student says, “80% inliers proves the alignment is correct.” Do you agree? Give a reason and one independent visual check. Answer in two sentences.

80%

60 of 75 pairs
fit one model

Answer

Feedback

Revise

Changed case

Explain

Check what the percentage actually proves.

FEEDBACK 30S → REVISION 60S

Ask what the 80% actually measures: agreement with a model, or proof that each pair represents the same physical window? Imagine the whole repeated pattern shifted by one window. Identify a non-repeating landmark or boundary that could test the proposed alignment independently of those repeated matches.

Revise or keep your decision. State what the 80% tells you and name the independent evidence you would check. Explain any change in your reasoning.

Answer

Feedback

Revise

Changed case

Explain

Changed case: a nearby pole and a distant wall

INDEPENDENT CHANGED CASE • 60 SECONDS

Changed case — work independently. The matches are now physically correct. A nearby pole and a distant wall are photographed while the camera moves sideways. The wall aligns well, but the pole appears twice. Can one homography necessarily align both depths? Name one change to image capture that would reduce the problem.

Answer

Feedback

Revise

Changed case

Explain

Wrong identity and wrong geometry are different

EXPLANATION · FINAL 60 SECONDS

The 80% describes agreement with one model, not proof of correct identity. Identical windows may agree on the wrong offset; check a unique landmark and whether the alignment is physically plausible. In the changed case, correct matches still do not make one homography suitable for a translated camera viewing multiple depths. Parallax can double the pole while the wall aligns. Rotate with minimal translation or use a nearly planar/distant scene.

Answer

Feedback

Revise

Changed case

Explain

From verified matches to a panorama

Capture overlap

Both images must share useful scene content.

Estimate H

Use filtered matches and robust geometry.

Choose a canvas

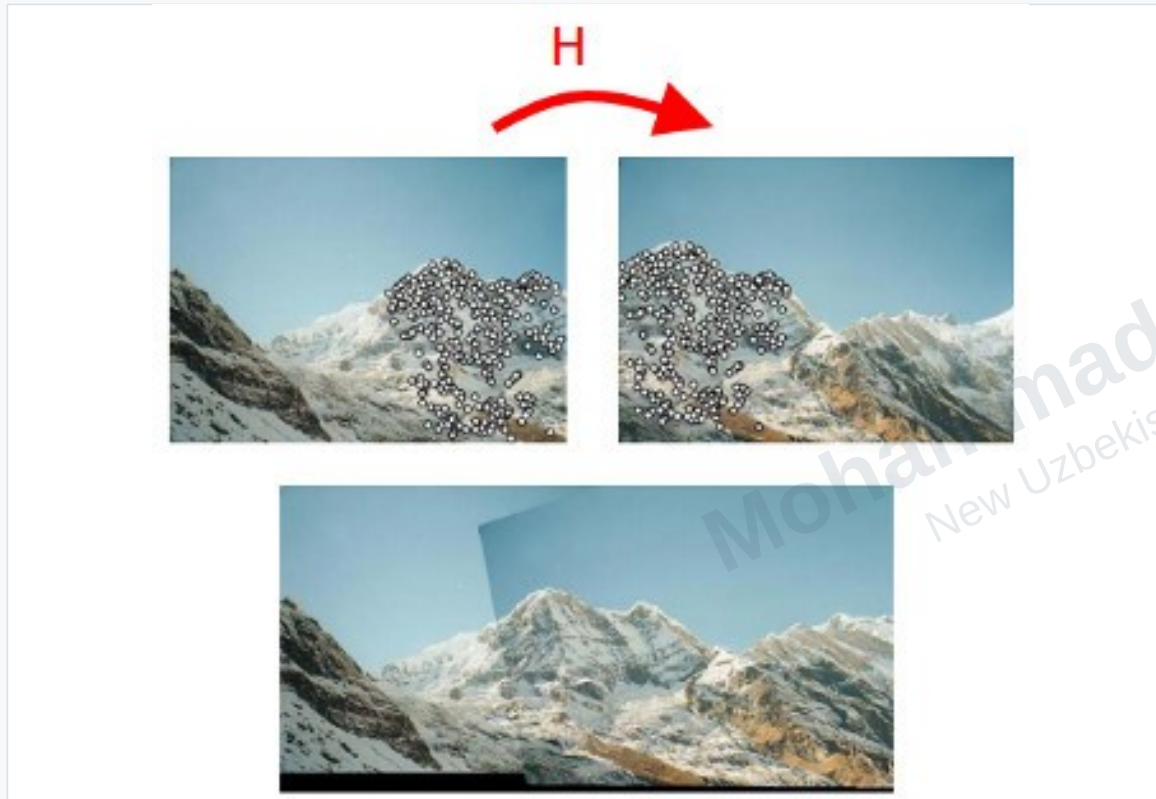
Include the transformed image boundaries.

Warp and blend

Resample, combine overlap and inspect the result.

Align geometry first; combine appearance second.

Overlapping views become a wider scene



Shared content

The overlap links the coordinate systems.

New content

Each view adds another part of the scene.

Inspection

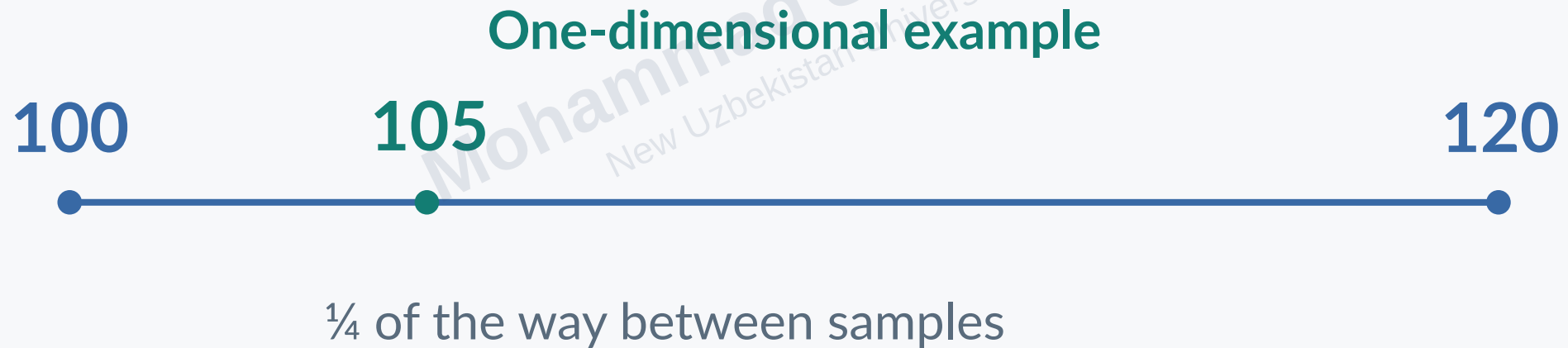
Look at boundaries and structures in the overlap.

OpenCV homography tutorial panorama example; not a student experiment.

A wider image is useful only when shared structures are aligned sensibly.

Warping also means resampling pixels

Output pixel → inverse map → source location → interpolate



Geometry gives a sampling location; interpolation gives an intensity.

Blending can soften an exposure seam

Image A: 100

Image B: 140

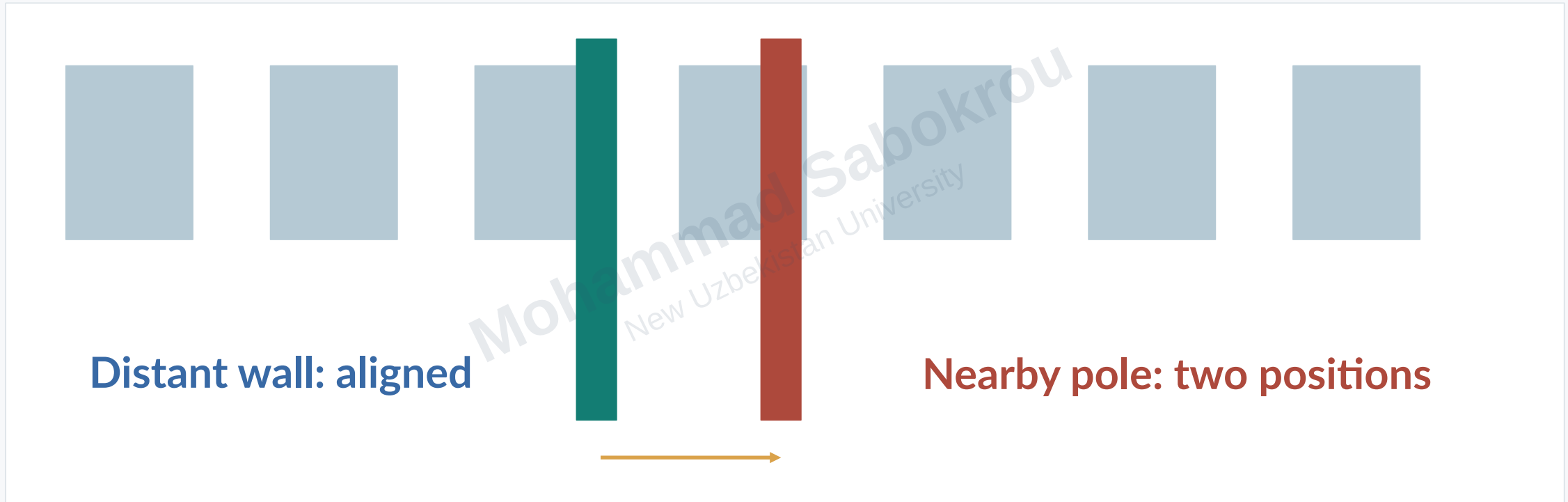


At equal weights: $0.5 \times 100 + 0.5 \times 140 = 120$

A smooth brightness transition is not a geometric accuracy test.

Blending can soften a brightness jump. It cannot repair mismatched geometry.

Parallax: the wall aligns, the pole doubles



Depth variation plus camera translation can violate the one-homography model.

Diagnose the failure before choosing the repair

Symptom	Likely issue to inspect	A targeted repair
Few usable points	Low texture / blur	Add textured context; hold steady
One-window offset	Repeated appearance	Include a unique landmark
Doubled near object	Parallax / wrong model	Reduce translation or depth variation
Brightness seam	Exposure mismatch	Stabilise exposure; blend after alignment
Only moving objects double	Scene motion	Capture a static scene / retake

A repair should address the cause, not merely make the output look smoother.

Capture a pair that gives matching a fair chance

Share useful content

Keep substantial overlap and distinctive texture.

Stabilise capture

Hold steady; avoid motion blur and changing exposure.

Respect the model

Prefer a plane or near-pure rotation.

Keep it safe

Use permitted scenes; avoid unnecessary personal information.

For a comparison, change one condition deliberately and keep the others similar.

A measured walk through the matching pipeline

1200

Candidates

648

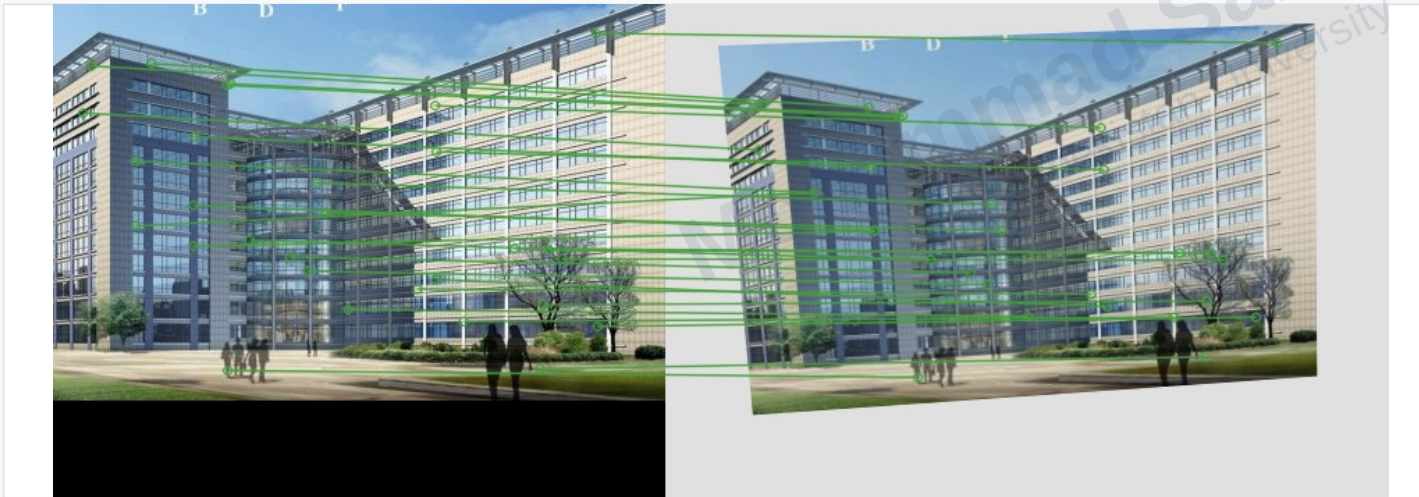
Ratio survivors

565

RANSAC inliers

87.2%

Inlier fraction



Filtered pairs (up to 30 drawn)



Boundary predicted by fitted H

This is a controlled transformed photograph—not two independent camera captures.

Lab bridge: measure, break, repair

Measure

Keypoints, matches, inliers, errors and coverage.

Break

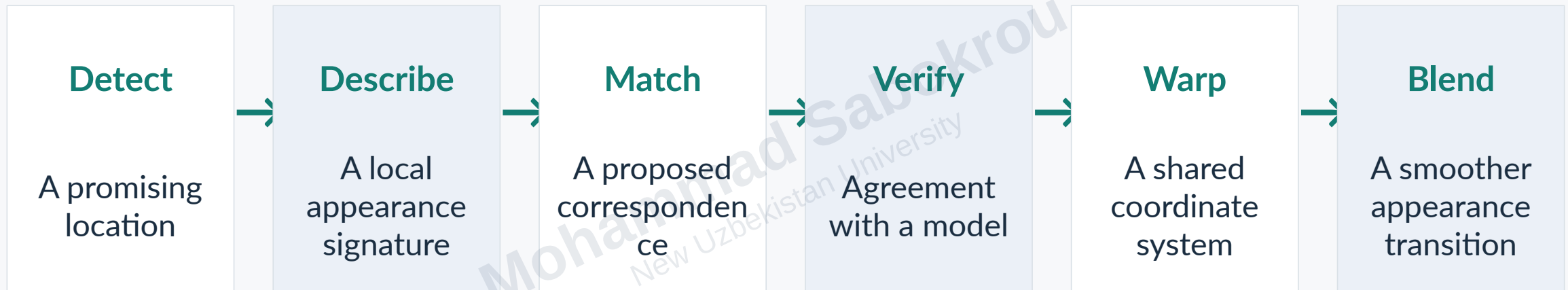
Low texture, repetition or extra translation—one at a time.

Repair

Choose a change that addresses the observed cause.

Report observations that support your explanation, not only a final image.

Say what each stage contributes



Appearance evidence

Geometry and construction

A reliable pipeline joins appearance evidence with geometric assumptions.

Three ideas to take into the lab

- 01** **A strong feature is not necessarily unique.**
Localisation and identity are different.
- 02** **A similar descriptor is not necessarily a correct match.**
A ratio threshold is not a probability.
- 03** **A strong consensus is not necessarily physical truth.**
Inspect assumptions and independent evidence.

Measure what happened. Explain why. Change the right thing.

Appendix • From patch shifts to gradients

$$E(u, v) = \sum w(x, y) [I(x+u, y+v) - I(x, y)]^2$$

$$I(x+u, y+v) \approx I(x, y) + I_x u + I_y v$$

$$E(u, v) \approx \sum w(x, y) (I_x u + I_y v)^2$$

I : intensity • u, v : small shifts • w : window weight • I_x, I_y : gradients

The approximation is for small shifts and a sufficiently smooth local image.

Appendix • A complete Harris calculation

Gradients: (2, 0), (0, 2), (2, 2) • all weights = 1

M entries: $\sum w |x|^2$, $\sum w |x|y$, $\sum w |y|^2 \rightarrow 8, 4, 8$

$$M = \begin{bmatrix} 8 & 4 \\ 4 & 8 \end{bmatrix}$$

$$\det(M) = 8 \times 8 - 4 \times 4 = 48$$

$$\text{trace}(M) = 8 + 8 = 16$$

$$k = 0.04$$

$$R = 48 - 0.04 \times 16^2 = 37.76$$

All values are illustrative. Thresholds depend on scale and image data.

Appendix • Why outliers increase RANSAC work

$$N \geq \log(1 - p) / \log(1 - w^s)$$

p: desired all-inlier-sample probability w: assumed inlier fraction s = 4

With p = 0.99	Assumed inliers	Trials (round up)
Easier case	w = 0.50	72
More outliers	w = 0.20	2,876

These trial counts are conditional on the assumptions, not a guarantee of correct geometry.

Appendix • A safe matching recipe

```
sift = cv.SIFT_create()
k1, d1 = sift.detectAndCompute(gray1, None)
k2, d2 = sift.detectAndCompute(gray2, None)
if d1 is None or d2 is None or len(d2) < 2:
    raise ValueError("Not enough descriptors")

pairs = cv.BFMatcher(cv.NORM_L2).knnMatch(d1, d2, k=2)
good = []
for pair in pairs:
    if len(pair) != 2:
        continue
    m, n = pair
    if n.distance > 0 and m.distance < 0.75 * n.distance:
        good.append(m)
```

No coding is required during the lecture. This is a reference for the lab.

Appendix • Fit, check and only then warp

```
if len(good) < 4:
    raise ValueError("At least four suitable pairs are needed")
src = np.float32([k1[m.queryIdx].pt for m in good])
dst = np.float32([k2[m.trainIdx].pt for m in good])
src = src.reshape(-1, 1, 2)
dst = dst.reshape(-1, 1, 2)
H, mask = cv.findHomography(src, dst, cv.RANSAC, 3.0)
if H is None or mask is None or not np.isfinite(H).all():
    raise ValueError("A usable homography was not found")
inliers = mask.ravel().astype(bool)
if inliers.sum() < 4:
    raise ValueError("Insufficient geometric support")
# Next: inspect spread, residuals and assumptions before warping.
```

A returned matrix is not sufficient: inspect support, spread, errors and scene assumptions.

Appendix • A small vocabulary map

Patch

A small neighbourhood of image pixels.

Descriptor

A numerical summary of the nearby appearance.

Homography

A projective mapping between image coordinates.

Parallax

Different image displacement for different depths under translation.

Keypoint

A detected position with optional scale and orientation.

Correspondence

A pair intended to represent the same physical point.

Inlier

A pair that agrees with a chosen model within a tolerance.

Blending

Combining appearance in an overlap after alignment.

Use these terms to describe evidence precisely.

Sources • Foundations and further reading

- [H]** Harris & Stephens (1988)
A Combined Corner and Edge Detector.
<https://www.bmva-archive.org.uk/bmvc/1988/avc-88-023.html>
- [S]** Lowe (2004), IJCV
Distinctive Image Features from Scale-Invariant Keypoints.
<https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf>
- [O]** Rublee et al. (2011), ICCV
ORB: An Efficient Alternative to SIFT or SURF.
https://docs.opencv.org/4.x/d1/d89/tutorial_py_orb.html
- [R]** Fischler & Bolles (1981), CACM
Random Sample Consensus, 24(6), 381–395.
- [B]** Szeliski (2022), 2nd edition
Computer Vision: Algorithms and Applications.
<https://szeliski.org/Book/>

Sources • OpenCV tutorials and visual credits

[F]

Introduction to Features

Patch examples and building photograph.

https://docs.opencv.org/4.x/df/d54/tutorial_py_features_meaning.html

[H]

Harris Corner Detection

Response-map and refinement illustrations.

https://docs.opencv.org/4.x/dc/d0d/tutorial_py_features_harris.html

[SI]

Introduction to SIFT

Scale-space, extrema and scale-change diagrams.

https://docs.opencv.org/4.x/da/df5/tutorial_py_sift_intro.html

[M /
FL]

Feature Matching / FLANN

Candidate-match images and descriptor metrics.

https://docs.opencv.org/4.x/dc/dc3/tutorial_py_matcher.html

[G /
GH]

Homography concepts / object localisation

Panorama illustration and projected book boundary.

https://docs.opencv.org/4.x/d9/dab/tutorial_homography.html