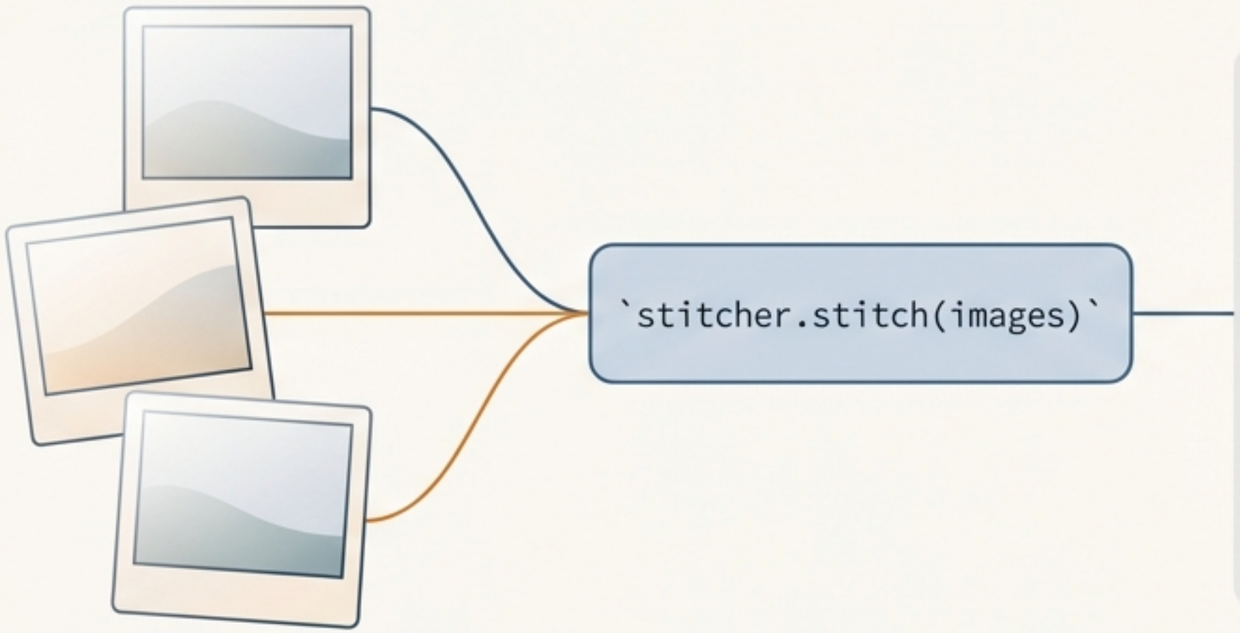
Module 11: The Art & Science of Image Stitching

From a Single Command to a Seamless Panorama

The One-Liner Panorama




```
# Create stitcher instance
# Mode can be PANORAMA (default) or SCANS
stitcher = cv2.Stitcher_create(cv2.Stitcher_PANORAMA)

# Stitch the images
(status, panorama) = stitcher.stitch(images)

# Check for success
if status == cv2.Stitcher_OK:
    # Display the result
```

Creating a panorama can be deceptively simple. The real power lies in understanding the engine inside this command.

Deconstructing the Stitch: The Panorama Pipeline

1. Detect Features

Find distinctive points in each image.

2. Match Features

Find correspondences between images.

3. Estimate Transformation

Compute the geometric transformation.

4. Warp Images

Transform images to a common plane.

5. Blend Pixels

Seamlessly combine the warped images.

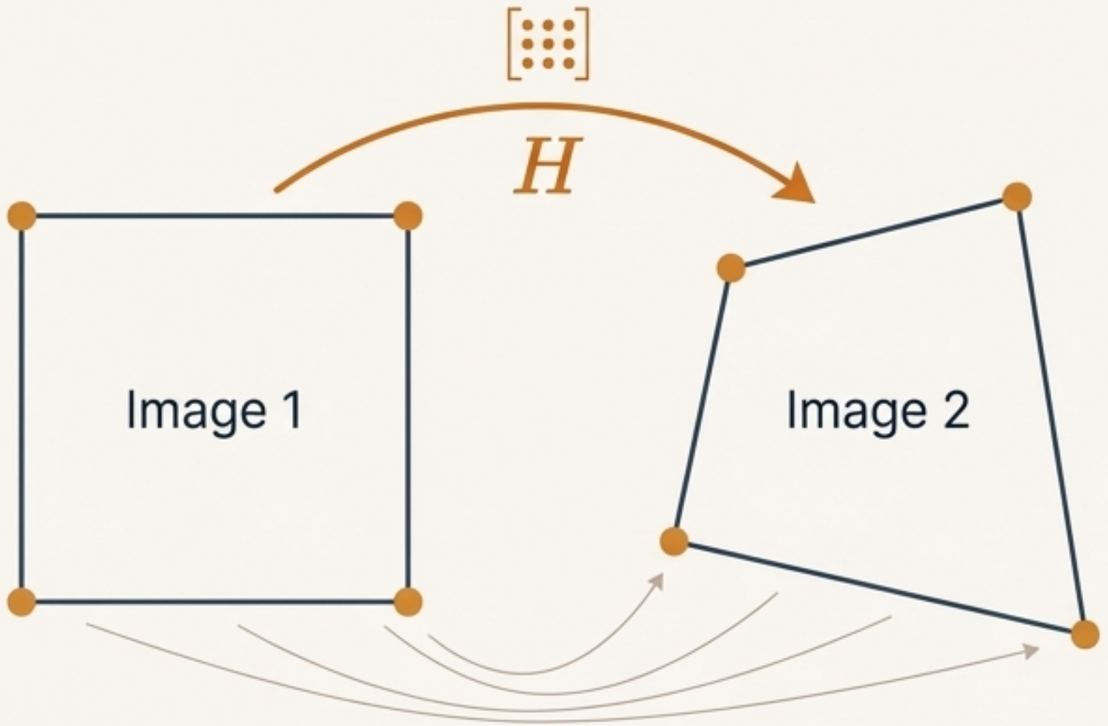
Step 1 & 2: Finding Common Ground

The process begins by finding unique, stable points (keypoints) in each image and then matching them across the overlap. We need at least 4 good matches to proceed.

```
# SIFT for high accuracy, ORB for speed
detector = cv2.SIFT_create()
# or detector = cv2.ORB_create(nfeatures=1000)
keypoints, descriptors = detector.detectAndCompute
        (image, None)
```

```
# Brute-force matcher finds candidate pairs
bf = cv2.BFMatcher()
matches = bf.knnMatch(desc1, desc2, k=2)

# Lowe's Ratio Test filters for unambiguous matches
good_matches = []
for m, n in matches:
    if m.distance < 0.75 * n.distance:
        good_matches.append(m)
```

Step 3: Aligning Perspectives with Homography

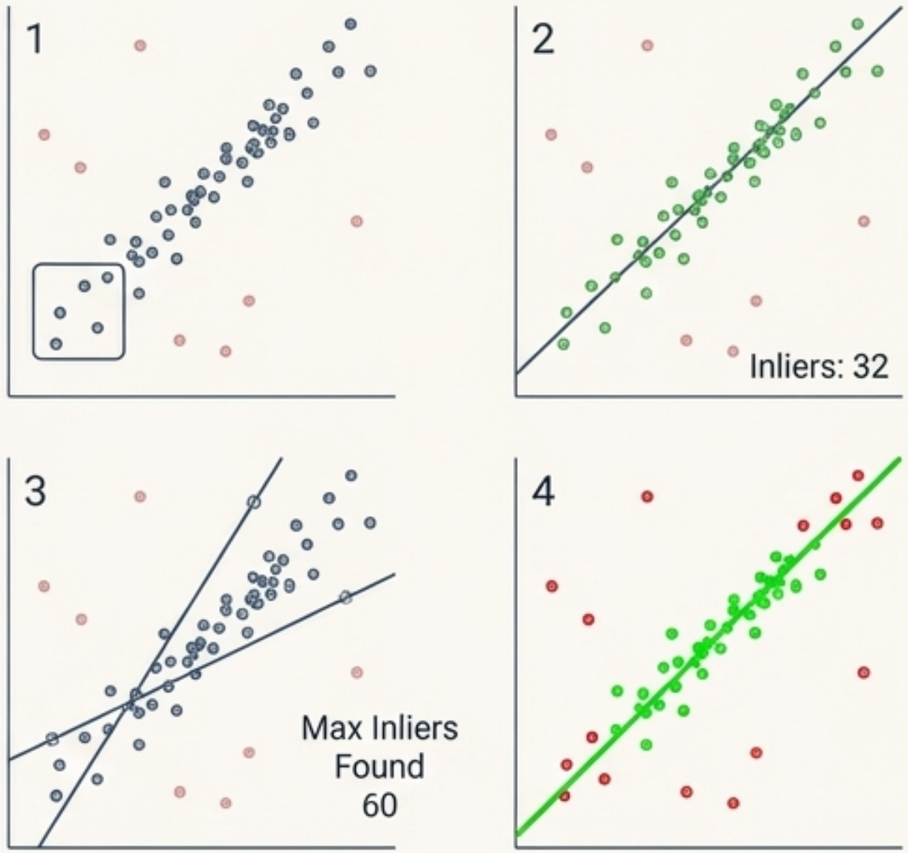


A **Homography matrix** (**H**) is a 3×3 matrix that describes the perspective warp between two planes. It has 8 degrees of freedom and allows us to map any point from one image to its corresponding position in the other.

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

Lowe's Ratio Test helps, but some 'good' matches can still be wrong.
How do we compute a reliable Homography from an imperfect set of points?

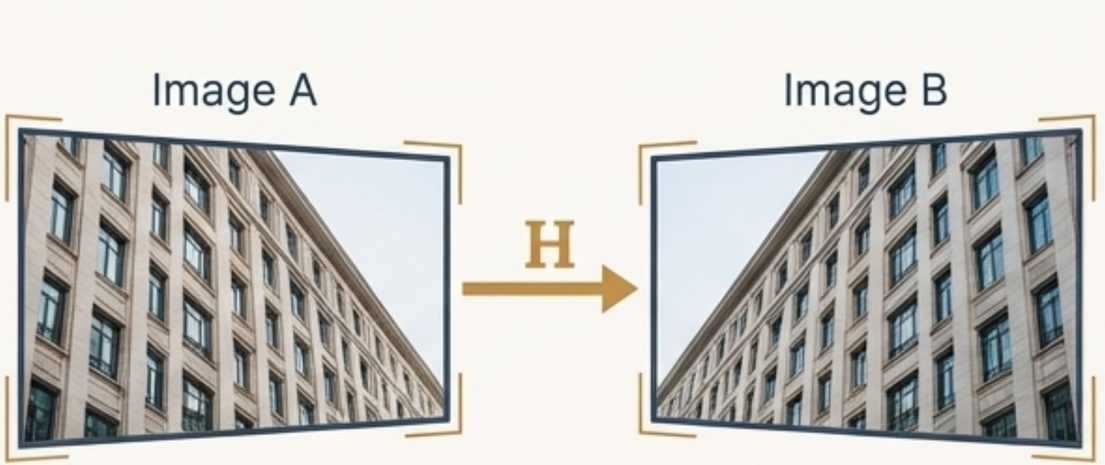
Deep Dive: Robust Matching with RANSAC



RANSAC (RANdom SAmple Consensus) is an iterative algorithm that finds the optimal Homography. It builds models from random subsets of matches and finds the one supported by the most data (the inliers), effectively ignoring outliers.

```
H, mask = cv2.findHomography(src_pts,
dst_pts, cv2.RANSAC, 5.0)
# The 'mask' returns which points were
identified as inliers
```

Step 4: Warping to a Unified Canvas



Concept

Using the robust Homography matrix H, we apply a perspective warp to one of the images. This transforms its pixels so they align perfectly with the other image in a shared coordinate system.

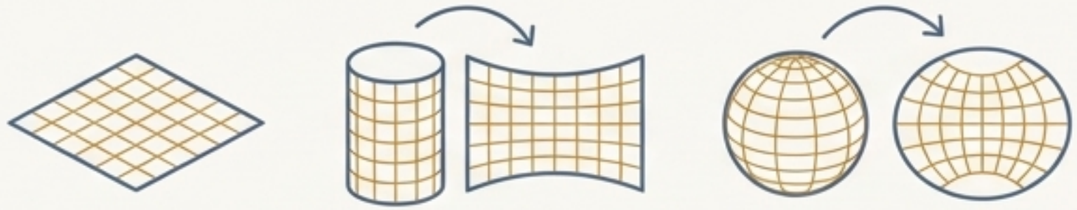
Canvas Calculation

To determine the size of the final panorama, we transform the corners of the source image, find the minimum and maximum coordinates, and create a canvas large enough to hold both images without clipping.

The Code

```
# Warp the image's perspective using the Homography
warped_image = cv2.warpPerspective(image, H, (width, height))
```

Choosing the Right Projection Surface



Warper	Description	Use Case
`Plane`	Planar projection	Small rotations, architectural shots
Cylindrical	Cylindrical surface projection	360° horizontal panoramas
Spherical	Spherical surface projection	Full 360° × 180° VR panoramas
Fisheye	Fisheye lens correction	Wide-angle and fisheye lenses

Step 5: The Challenge of the Seam

Simple Overlay

Seamless Blend

Geometric alignment is solved. But how do we make the transition between images invisible?

A Spectrum of Blending Strategies

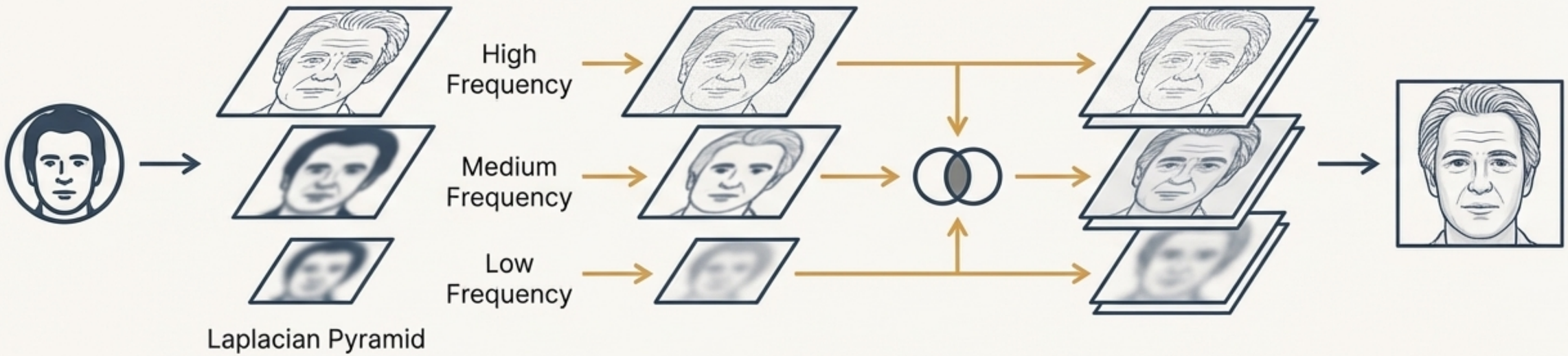
No Blending

Alpha Blending

Multi-Band Blending

Method	Quality	Speed	Artifacts
No blending	Poor	Fast	Visible seams
Alpha	Medium	Fast	Ghosting, gradient artifacts
Feather	Good	Medium	Slight blur at the seam
Multi-band	Best	Slow	Minimal to none

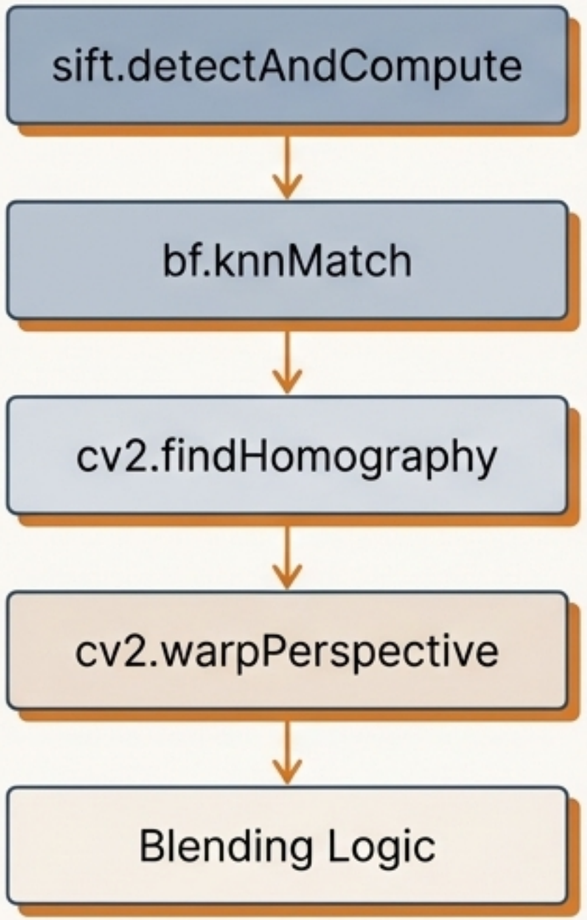
The Gold Standard: Blending Frequencies, Not Pixels



The technique's brilliance is blending **low frequencies** (colors, tones) over a **wide area and high frequencies** (edges, textures) **over a narrow area**. This avoids both ghosting and hard seams.

```
blender = cv2.detail.MultiBandBlender()
```

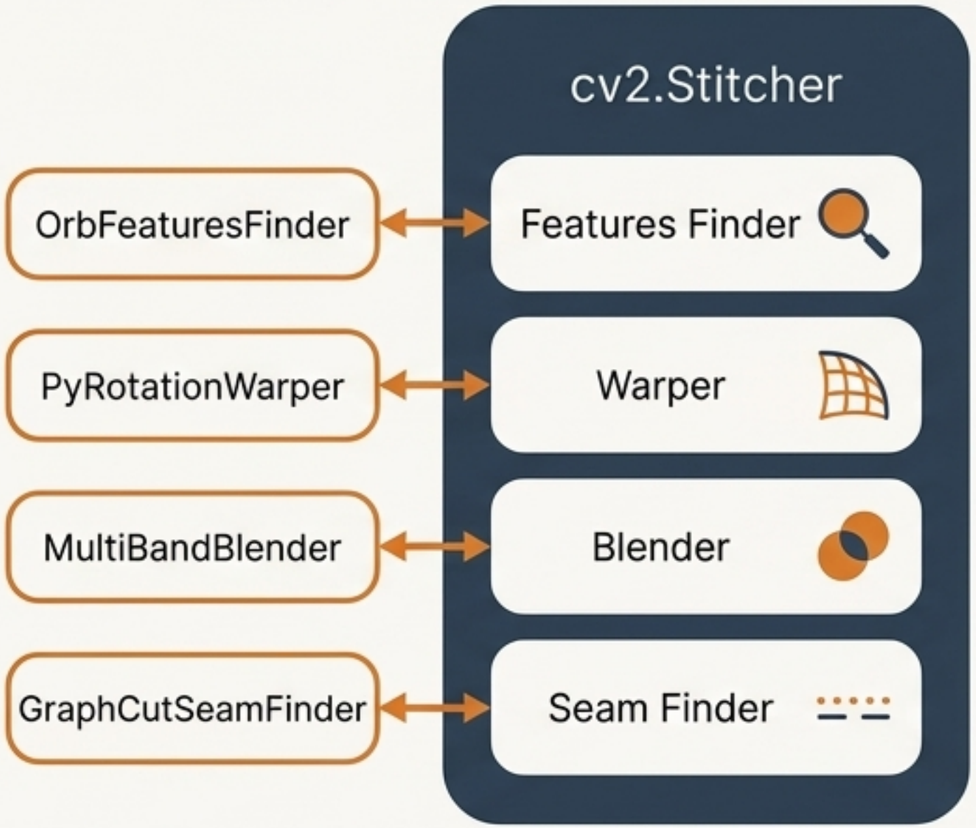
Full Control: The Manual Stitching Pipeline




```python
def stitch_two_images(img1, img2):
    # 1. Detect & describe features (SIFT/ORB)
    kp1, desc1 = sift.detectAndCompute(img1, None)
    kp2, desc2 = sift.detectAndCompute(img2, None)

    # 2. Match features & apply ratio test
    good_matches = match_features(desc1, desc2)

    # 3. Find homography with RANSAC
    H, _ = cv2.findHomography(src_pts, dst_pts, cv2.RANSAC, 5.0)

    # 4 & 5. Warp image and blend onto a combined canvas
    result = warp_and_blend(img2, img1, H)
    return result
```

Now that you understand each component, you can build and customize the entire pipeline for maximum control and performance.

Fine-Tuning the Automated Stitcher




```python
stitcher = cv2.Stitcher_create()

# Swap the default feature detector for ORB
stitcher.setFeaturesFinder(cv2.detail.OrbFeaturesFinder())

# Set a spherical warper for a 360° panorama
stitcher.setWarper(cv2.PyRotationWarper('spherical', 1.0))

# Upgrade to the highest quality blender
stitcher.setBlender(cv2.detail.MultiBandBlender())

# Use a graph-cut based seam finder
stitcher.setSeamFinder(cv2.detail.GraphCutSeamFinder('COST_COLOR'))
```

Field Guide to Great Stitching

Sufficient Overlap (20-40%)

Ensures enough common features for robust matching.

Rotate, Don't Pan

Rotate the camera around its optical center to minimize parallax error.

Consistent Exposure

Lock focus and exposure settings to prevent brightness and color shifts between shots.

Avoid Moving Objects

Movement in the overlap region creates ghosting and stitching artifacts.

Seek Textured Surfaces

Feature detectors work best on images with rich textures and distinct details.

Your Journey Into Stitching

We started with a simple command and uncovered the powerful, five-step pipeline within.

Further Exploration

- OpenCV Stitching Module Documentation
- OpenCV Panorama Tutorial
- Original Multi-Band Blending Paper (Burt & Adelson, 1983)

You now have the art and the science to create stunning panoramas.