Comprehensive Computer Vision Handbook with Flutter Integration

Abdelrahman Wael Ibrahim

September 7, 2025

Contents

1 Introduction **3**
- 1.1 What is Computer Vision? . . . 3
- 1.2 History . . . 3
- 1.3 Flutter Project Ideas . . . 3

2 Image Formation **4**
- 2.1 Camera Models . . . 4
- 2.2 Flutter Integration . . . 4

3 Image Processing **5**
- 3.1 Filtering . . . 5
- 3.2 Morphological Operations . . . 5
- 3.3 Pyramids and Fourier Transform . . . 5
- 3.4 Flutter Project . . . 5

4 Feature Detection and Matching **6**
- 4.1 Algorithms . . . 6
- 4.2 Flutter Integration . . . 6

5 Segmentation **7**
- 5.1 Algorithms . . . 7
- 5.2 Flutter Integration . . . 7

6 Feature-Based Alignment **8**
- 6.1 2D Alignment . . . 8
- 6.2 3D Pose Estimation (PnP) . . . 8
- 6.3 Flutter Integration . . . 8

7 Structure from Motion **9**
- 7.1 Triangulation . . . 9
- 7.2 Bundle Adjustment . . . 9
- 7.3 Flutter Integration . . . 9

8 Dense Motion Estimation **10**
- 8.1 Optical Flow . . . 10
- 8.2 Flutter Integration . . . 10

9 Image Stitching **11**
9.1 Pipeline 11
9.2 Flutter Integration 11

10 Computational Photography **12**
10.1 HDR Imaging 12
10.2 Flutter Integration 12

11 Stereo Correspondence **13**
11.1 Depth Estimation 13
11.2 Flutter Integration 13

12 3D Reconstruction **14**
12.1 Representations 14
12.2 Flutter Integration 14

13 Image-Based Rendering **15**
13.1 Light Fields and View Interpolation 15
13.2 Flutter Integration 15

14 Recognition **16**
14.1 Algorithms 16
14.2 Flutter Integration 16

15 General Flutter CV Guidelines **17**

Chapter 1

Introduction

1.1 What is Computer Vision?

Computer vision is the field that enables machines to interpret, process, and understand images or videos. Mathematically, it can be formulated as:

$$f : I \in \mathbb{R}^{H \times W \times C} \rightarrow Y$$

where H, W are image height and width, C is the number of channels, and Y is the output (class, segmentation map, 3D reconstruction, etc.)

1.2 History

- 1980s: Hough Transform, Harris Corners
- 1999: SIFT (Scale-Invariant Feature Transform)
- 2012: AlexNet introduced deep learning revolution in CV

1.3 Flutter Project Ideas

- Image Classification App using `tflite_flutter` and `camera` plugin
- Capture → Preprocess → Run TFLite model → Display prediction

Chapter 2

Image Formation

2.1 Camera Models

Pinhole camera model:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K[R|t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \quad K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

Lens distortion correction formulas:

$$r' = r(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

$$(x', y') = (x + [2p_1 xy + p_2(r^2 + 2x^2)], y + [p_1(r^2 + 2y^2) + 2p_2 xy])$$

2.2 Flutter Integration

- Camera Calibration App: capture checkerboard → compute intrinsic/extrinsic → undistort images
- Plugin: `flutter_opencv`

Chapter 3

Image Processing

3.1 Filtering

$$I'(x, y) = \sum_i \sum_j h(i, j) I(x - i, y - j)$$

Common filters: Gaussian, Sobel, Laplacian, Median

3.2 Morphological Operations

- Erosion, Dilation, Opening, Closing

3.3 Pyramids and Fourier Transform

- Gaussian Laplacian Pyramids
- FFT-based filtering for frequency domain

3.4 Flutter Project

- Real-time filter edge detection app
- Capture → Apply filters → Display live preview
- Tools: `image` package or `flutter_opencv`

Chapter 4

Feature Detection and Matching

4.1 Algorithms

- Harris Corner: $R = \det(M) - k \cdot (\text{trace}(M))^2$
- SIFT/ORB descriptors
- Feature matching + RANSAC

4.2 Flutter Integration

- Photo similarity app: highlight matched keypoints
- Workflow: detect → match → filter outliers → overlay matches
- Plugins: `flutter_opencv`, Dart Canvas for drawing

Chapter 5

Segmentation

5.1 Algorithms

- Graph Cuts, Active Contours, K-means
- Deep Learning: U-Net, Mask R-CNN
- Energy Minimization: $E(L) = \sum_p D_p(L_p) + \sum_{(p,q)} V_{p,q}(L_p, L_q)$

5.2 Flutter Integration

- Background removal app: capture → run TFLite segmentation → replace background
- Tools: `tflite_flutter`, `image`

Chapter 6

Feature-Based Alignment

6.1 2D Alignment

Compute homography H using DLT + RANSAC:

$$x' \sim Hx$$

6.2 3D Pose Estimation (PnP)

$$\min_{R,t} \sum_i \|x_i - K(RX_i + t)\|^2$$

6.3 Flutter Integration

- Panorama stitching app: align multiple images → warp → blend
- Plugin: `flutter_opencv`, `image`

Chapter 7

Structure from Motion

7.1 Triangulation

$$X = \arg\min \sum_i \|x_i - P_i X\|^2$$

7.2 Bundle Adjustment

$$\min_{R_i, t_i, X_j} \sum_{i,j} \|x_{ij} - \pi(R_i, t_i, X_j)\|^2$$

7.3 Flutter Integration

- 3D object reconstruction app: multiple images → backend SfM → display 3D model
- Tools: `flutter_3d_obj`, Python + OpenCV + COLMAP backend

Chapter 8

Dense Motion Estimation

8.1 Optical Flow

Lucas-Kanade / Horn-Schunck:

$$I_x u + I_y v + I_t = 0$$

$$E(u, v) = \iint (I_x u + I_y v + I_t)^2 + \alpha(\|\nabla u\|^2 + \|\nabla v\|^2)$$

8.2 Flutter Integration

- Motion tracking / video stabilization app
- Capture video → compute flow → overlay vectors
- Plugin: `flutter_opencv`

Chapter 9

Image Stitching

9.1 Pipeline

1. Detect features
2. Match features
3. Compute homography
4. Warp and blend

9.2 Flutter Integration

- 360° panorama creator app
- Capture multiple images → stitch → export panorama

Chapter 10

Computational Photography

10.1 HDR Imaging

$$I_{\text{HDR}} = \frac{\sum_i w(I_i) I_i / t_i}{\sum_i w(I_i)}$$

10.2 Flutter Integration

- HDR super-resolution app: multiple exposure capture → merge → enhance
- Tools: `tflite_flutter`, backend ML server

Chapter 11

Stereo Correspondence

11.1 Depth Estimation

$$Z = \frac{fB}{d}$$

11.2 Flutter Integration

- Depth map app: dual-camera → compute disparity → display 3D heatmap

Chapter 12

3D Reconstruction

12.1 Representations

Point cloud, mesh, voxel grid, TSDF

12.2 Flutter Integration

- 3D scanner app: multiple images → backend reconstruction → render in Flutter
- Plugins: `flutter_3d_obj`, platform channels for Unity/SceneKit

Chapter 13

Image-Based Rendering

13.1 Light Fields and View Interpolation

$$L(u, v, s, t)$$

13.2 Flutter Integration

- Interactive 3D viewer: change camera angles → interpolate views

Chapter 14

Recognition

14.1 Algorithms

- Object Detection: YOLO, Faster R-CNN
- Face Recognition: FaceNet embeddings, ArcFace
- Scene Understanding: combine semantic + instance segmentation

14.2 Flutter Integration

- Real-time object face detection app
- Capture image/video → detect → overlay bounding boxes labels
- Plugins: `tflite_flutter`, `camera`, `flutter_opencv`

Chapter 15

General Flutter CV Guidelines

- Camera Input: `camera`, `image_picker`
- On-device ML: `tflite_flutter`
- OpenCV: `flutter_opencv` or FFI for C++
- 3D Rendering: `flutter_3d_obj`, Unity/SceneKit via platform channel
- Backend Processing: Python + OpenCV + PyTorch/TensorFlow
- Mini-project phase template: Capture → Preprocess → Compute → Post-process → Display