Eye Pose Tool – Full Baseline Technical Design Document (Bilingual)

眼位九宫格工具 —— 基线技术设计文档（中英文双语）

本文件综合了我们在本次对话中对 1 – 10 章的全部共识，是今后所有开发者和 AI 模型继续开发本项目的唯一权威基线。

Chapter 1. Project Overview / 第 1 章 项目概述

1.1 Purpose / 目的

- The Eye Pose Tool is an Android-based clinical utility to automatically process multi-direction gaze photos and generate a standardized 3×3 eye-position mosaic.

- 眼位九宫格工具是一个基于 Android 的临床辅助工具，自动处理多方向眼位照片，并生成标准化的 3×3 九宫格眼位图，用于病历记录和临床对比。

1.2 Target Users / 目标用户

- Ophthalmologists, orthoptists, clinical researchers.

- 眼科医生、斜弱视与视功能医生、临床研究人员。

1.3 Core Value / 核心价值

- Automatic cropping with a fixed 2:1 aspect ratio (眉毛→鼻梁，上下；脸廓→耳廓，左右)。

- Automatic gaze direction classification into 9 directions (UL, U, UR, L, C, R, DL, D, DR) regardless of input order.

- Automatic placement into a fixed 3×3 grid (B-area), with optional manual drag correction.

- High-quality JPEG export (quality=100, subsampling=0) suitable for medical records.

1.4 Design Principles / 设计原则

- 文档是唯一真相：所有行为以本 Baseline 文档为准。

- 机器自动为主，人为拖拽纠错为辅。

- 输入顺序任意（A 区），输出布局标准（B 区九宫格）。

- 算法可替换，接口和数据结构必须保持兼容。

Chapter 2. Core Workflow / 第 2 章 核心流程

2.1 High-level Workflow / 总体流程

1. User selects 1 – 9 photos in any order → A-area shows raw thumbnails.

2. Flutter calls Python to process all images:

- Face & eye detection

- Eye pose normalization (roll correction + centering)

- 2:1 cropping

- Gaze direction estimation (9 directions)

3. Flutter auto-fills B-area 3×3 grid based on directions.

4. Doctor may drag images in B-area to correct misplacements.

5. User taps “ Save Photos ” to export a final 3×3 mosaic to the gallery.

2.2 A-area (Raw Input) / A 区（原始输入）

- 3×3 grid showing up to 9 original photos.

- The order has **no meaning** for directions.

- A-area is purely for input preview.

2.3 Automatic Image Processing / 自动图像处理

- 每张图片都经过：人脸检测 → 眼睛与虹膜定位 → 眼位几何标准化（双眼连线水平、鼻梁居中）→ 2:1 裁剪 → 视线方向分类。

2.4 Gaze Direction Estimation / 视线方向判定

- For each eye, compute dx, dy:

dx = (pupil_x - eye_center_x) / eye_width

dy = (pupil_y - eye_center_y) / eye_height

- Use thresholds Tx=0.25, Ty=0.25 to classify Left/Center/Right and Up/Center/Down.

- Combine to 9 directions: upLeft, up, upRight, left, center, right, downLeft, down, downRight.

2.5 B-area Auto Layout / B 区自动布局

- Fixed layout:

Row1: upLeft | up | upRight

Row2: left | center | right

Row3: downLeft | down | downRight

- Each GazeResult is mapped to one cell according to its direction.

- If multiple images share same direction: keep the one with higher confidence, others go to a candidate list (optional).

2.6 Manual Drag Correction / 拖拽纠错

- Long-press to drag a cell in B-area; dropping onto another cell swaps the two.

- Manual operations override automatic classification for final export.

2.7 Export / 导出

- Combine B-area current layout into one 3×3 high-quality JPEG.

- Save to Android gallery; the exported mosaic must visually match B-area exactly.

Chapter 3. UI Specification / 第 3 章 UI 设计规范

3.1 Overall Layout / 整体布局

- Single scrollable page (SingleChildScrollView).

- Layout from top to bottom:

A-area (3×3 raw grid)

B-area (3×3 processed grid, draggable)

C-area (3 buttons: Select / Process / Save)

3.2 A-area / A 区

- 3×3 non-scrollable grid, 9 cells.
- Shows original photos or placeholders with “ + ” icon.
- Does not indicate gaze directions.

3.3 B-area / B 区

- 3×3 grid, fixed semantics per position:
(0,0)=upLeft, (0,1)=up, (0,2)=upRight,
(1,0)=left, (1,1)=center, (1,2)=right,
(2,0)=downLeft, (2,1)=down, (2,2)=downRight.
- Displays processed 2:1 cropped images.
- Supports long-press drag-and-drop to swap cells.

3.4 C-area Buttons / C 区按钮

- Select Photos: open system picker, allow 1 – 9 selection, refill A-area and clear B-area.
- Process Photos: send current A-area images to Python, fill B-area with results; disabled if no images.
- Save Photos: export B-area mosaic; disabled if B-area has no processed images.

3.5 State Management / 状态管理

- Recommended: Riverpod.
- Core states:
- rawPhotos: list of original paths.
- processedGrid: List length 9.
- processingState: idle / processing / done.
- errorMessage: last error to show.

3.6 Error UI / 错误提示

- Use SnackBar or AlertDialog.
- Examples:
- “ 请先选择照片 / Please select photos first. ”
- “ 未检测到人脸 / Face not detected. ”

Chapter 4. Python Processing Module / 第 4 章 Python 图像处理模块

4.1 Responsibilities / 职责

- Read input images.
- Detect face & landmarks (Mediapipe FaceMesh/Iris).

- Normalize head pose (roll correction; eye-eye line horizontal).
- Center nose line & eyes for cropping.
- Compute gaze direction dx/dy.
- Crop to 2:1 region (眉毛→鼻梁, 左右脸廓).
- Save processed images.
- Return JSON with direction, confidence, paths.

4.2 Face & Iris Detection / 人脸与虹膜检测

- Use MediaPipe FaceMesh with refine_landmarks=True.
- Key regions (indices may vary by version):
- Left eye contour
- Right eye contour
- Iris points for each eye
- Nose bridge top & nose tip

4.3 Eye Pose Normalization / 眼位几何标准化算法

目标：

- 双眼连线水平，鼻梁线大致垂直居中。

步骤：

1. 计算左眼、右眼中心：L=(Lx,Ly), R=(Rx,Ry)。
2. 使用 atan2 计算眼-眼线倾角 angle_deg = atan2(Ry-Ly, Rx-Lx) * 180/pi。
3. 将整张图像绕中心旋转 -angle_deg，使眼-眼线水平（roll 校正）。
4. 对所有 landmarks 应用同样的旋转（仿射变换），得到旋转后的 L’, R’, nose_top’, nose_tip’。
5. 计算鼻梁中点 X 坐标：nose_center_x = (nose_top’.x + nose_tip’.x)/2。
6. 计算双眼中心高度：eyes_center_y = (L’.y + R’.y)/2。
7. 将裁剪框中心设置为 (nose_center_x, eyes_center_y)，必要时对 Y
略作上移，让眼睛稍高于垂直中心。
8. 确定 2:1 裁剪框宽高（可根据人脸尺寸或固定值计算），并从裁剪中心向四周扩展：
- crop_width : crop_height = 2 : 1
- x1 = cx - crop_width/2, x2 = cx + crop_width/2
- y1 = cy - crop_height/2, y2 = cy + crop_height/2
9. 对 x1,y1,x2,y2 做边界裁剪，保证落在图像范围内。
10. 执行裁剪，得到最终标准化 2:1 图像，并输出 rotation_angle = -angle_deg 供调试使用。

4.4 Gaze Vector Calculation / 视线偏移计算

- For each eye, compute pupil center and eye center.
- dx = (pupil_x - eye_center_x) / eye_width
- dy = (pupil_y - eye_center_y) / eye_height

4.5 Direction Classification / 方向分类

- Thresholds: Tx=0.25, Ty=0.25.

- 水平：dx<-Tx→Left, dx>Tx→Right, |dx|<=Tx→Center。
- 垂直：dy<-Ty→Up, dy>Ty→Down, |dy|<=Ty→Center。
- Combine into 9 directions: upLeft, up, upRight, left, center, right, downLeft, down, downRight.

4.6 Confidence Score / 置信度
- Based on:
- Iris detection quality.
- Consistency between left and right eye dx/dy.
- Magnitude of deviation beyond thresholds.
- Range: 0.0 ~ 1.0.

4.7 JSON Response / JSON 返回结构
Example:
{
"results": [
{
"original_path": "/input/p1.jpg",
"cropped_path": "/output/p1_crop.jpg",
"direction": "upLeft",
"confidence": 0.92,
"rotation_angle": -3.2
}
],
"errors": []
}

4.8 Error Codes / 错误码
- "NoFaceDetected"
- "IrisNotFound"
- "InvalidGazeVector"
- "LowConfidence"
- "CropFailed"
- "RotateFailed"
- "PythonCrash"
- "FileMissing"
- "UnsupportedFormat"

Chapter 5. Frontend ↔ Python Protocol / 第 5 章 前后端交互协议

5.1 Communication / 通信方式

- Use serious_python as the bridge between Flutter and embedded Python.
- Flutter calls:

await SeriousPython.invoke("process_images", argsJson)

5.2 Request JSON / 请求 JSON

{
"images": [
"/path/to/photo1.jpg",
"/path/to/photo2.jpg"
]
}

5.3 Response JSON / 返回 JSON

见第 4 章 4.7。Flutter 需解析 results[] 并映射到 B 区。

5.4 Mapping / 映射规则

- 由 direction 映射到 0 – 8 索引。
- index = eyeDirectionToIndex(direction)

Chapter 6. Data Structures / 第 6 章 数据结构定义

6.1 EyeDirection Enum

- Values: upLeft, up, upRight, left, center, right, downLeft, down, downRight.

6.2 GazeResult (Dart)

class GazeResult {
final String originalPath;
final String croppedPath;
final EyeDirection direction;
final double confidence;
final double rotationAngle;
}

6.3 ProcessResult (Dart)

class ProcessResult {
final List results;
final List errors;
}

6.4 B-area Grid

- List grid = List.filled(9, null);
- 0..8 按照固定方向顺序映射。

6.5 ExportResult

```
class ExportResult {
final String outputPath;
final bool success;
final String? error;
}
```

Chapter 7. Detailed Modules / 第 7 章 模块详细说明

7.1 Flutter Modules
- ui/: home_page.dart, grid_a_area.dart, grid_b_area.dart, button_panel.dart, image_viewer.dart
- models/: gaze_result.dart, eye_direction.dart, process_result.dart
- providers/: raw_photo_provider.dart, grid_provider.dart, python_state_provider.dart
- services/: python_service.dart, export_service.dart

7.2 A-area Module
- Displays raw photos.
- No semantics about directions.

7.3 B-area Module
- Displays processed & auto-sorted photos.
- Supports drag-and-drop.

7.4 Drag Module
- Implemented via LongPressDraggable + DragTarget.
- Swaps GazeResult entries in grid list.

7.5 Python Module
- processor.py as main logic.
- face_detector.py, iris_detector.py, gaze_vector.py, direction_classifier.py, cropper.py.

7.6 Export Service
- Reads B-area grid.
- Asks Python (or Dart) to compose final 3×3 JPEG.
- Saves to gallery.

Chapter 8. Edge Cases & Exceptions / 第 8 章 边界条件与异常处理

8.1 Empty A-area

- Show “请先选择照片 / Please select photos first.”
- Disable Process button.

8.2 NoFaceDetected

- Skip that image, keep B-area cell empty.
- Show message including filename.

8.3 IrisNotFound, InvalidGazeVector, LowConfidence

- Do not auto-place into B-area.
- Optionally keep as candidate; doctor can correct by drag.

8.4 Multiple Images Same Direction

- Keep the higher confidence in main grid.
- Lower confidence ones go to candidates.

8.5 CropFailed, RotateFailed

- Show human-readable error; skip that image.

8.6 PythonCrash

- Show “处理失败，请重试 / Processing failed, please retry.”
- Optionally restart Python engine.

8.7 Empty Cells in Export

- Still export full 3×3 mosaic; empty cells use gray background.

8.8 Very Large Input

- Automatically downscale before processing.

8.9 OnlyOneEyeDetected, EyesClosed

- Mark as low confidence or error; require manual correction.

Chapter 9. Performance & Compatibility / 第 9 章 性能与设备适配

9.1 Performance Targets

- 9 images processed ≤ 3 seconds on a flagship device.
- Python engine init ≤ 1.2 seconds.
- Memory usage ≤ 400 MB peak.

9.2 Python Optimization

- Load Mediapipe model once.
- Batch process all images.
- Downscale very large images.

9.3 Flutter Optimization

- Use Riverpod to minimize rebuild.
- Use lightweight drag widgets.

9.4 Device Compatibility

- Android 10 – 15.
- ARM64-v8a as primary ABI.
- No dependency on Google Play Services.

9.5 Offline Requirement

- Entire system must run fully offline.
- No cloud calls, no external APIs.

Chapter 10. Future Extensions / 第 10 章 未来扩展

10.1 AI-based Gaze Classifier

- Replace or augment geometric dx/dy with a small CNN model.

10.2 Quality Assessment

- Score photo quality: blur, darkness, occlusion.

10.3 Multi-patient Management

- Store multiple mosaics with patient IDs & timestamps.

10.4 Annotation Tools

- Allow drawing arrows, notes on each image.

10.5 Video Mode

- Capture a short video, auto-select best frames for each direction.

End of Baseline Document / 基线文档结束
