Gesture Recognition

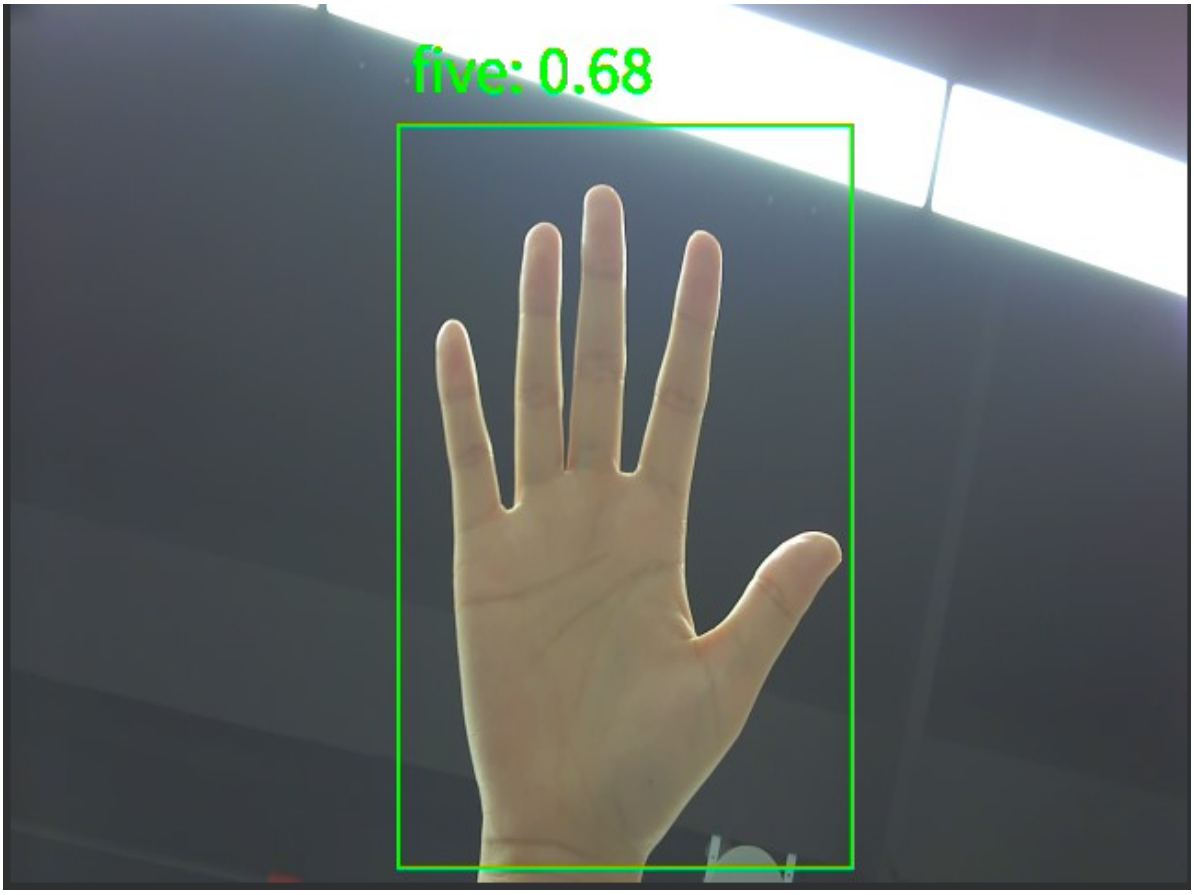
Routine Experiment Effect

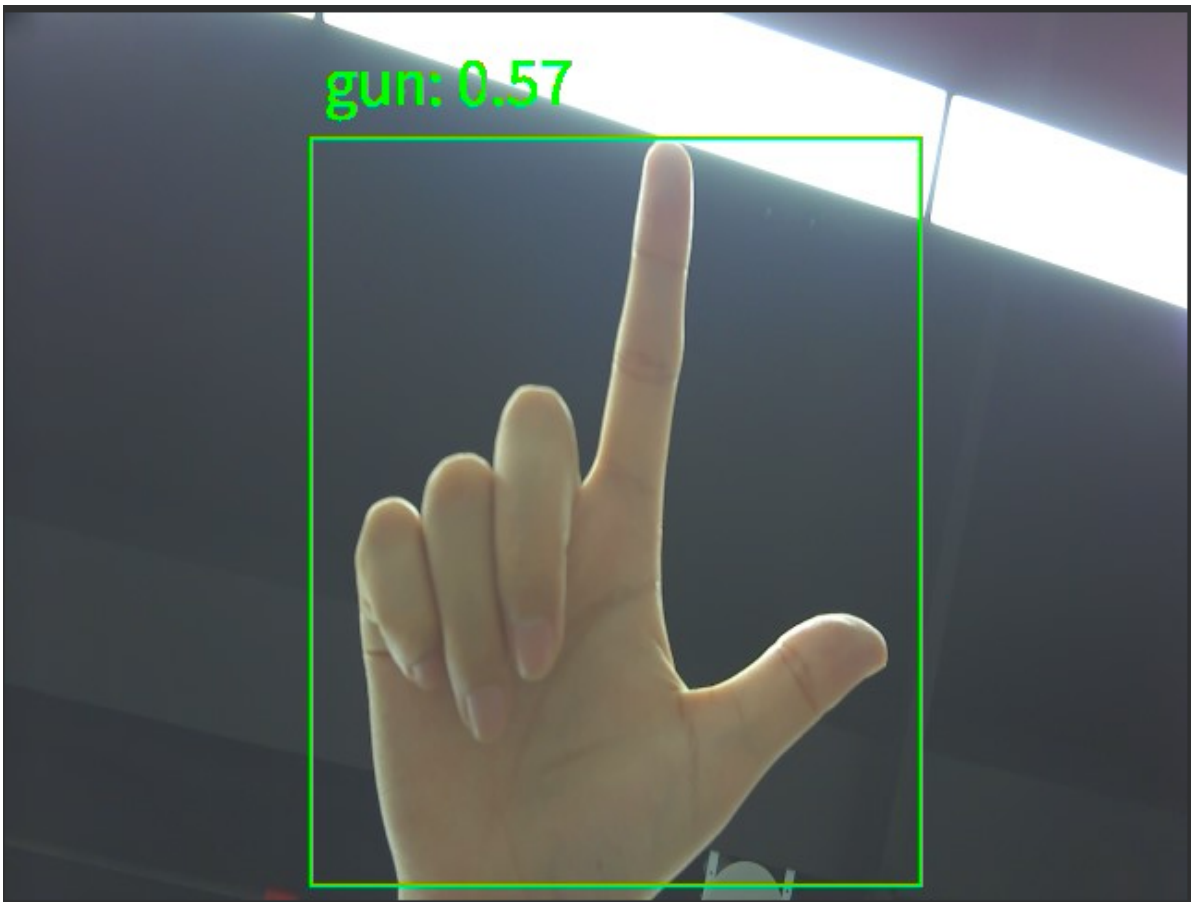
In this section, we will learn how to use K230 to implement gesture recognition.

The example code is in [Source code/08.Body/08.hand_recognition.py]

In fact, in the previous section, our palm key point detection classification has realized the gesture recognition function. In this section, we remove the lines of the key point recognition and directly display the recognized content.

> The implementation of gesture recognition in this section is different from that of gesture key point classification in the previous section.
>
> The gesture key point classification in the previous section is to distinguish the gestures according to the position of the fingers.
>
> The gesture recognition in this section uses a large number of photos of different gestures as a training set, and the trained model is used for recognition and judgment.
>
> In this section, the gesture direction must be upward (as shown in the figure below)





Code Explanation

Code structure

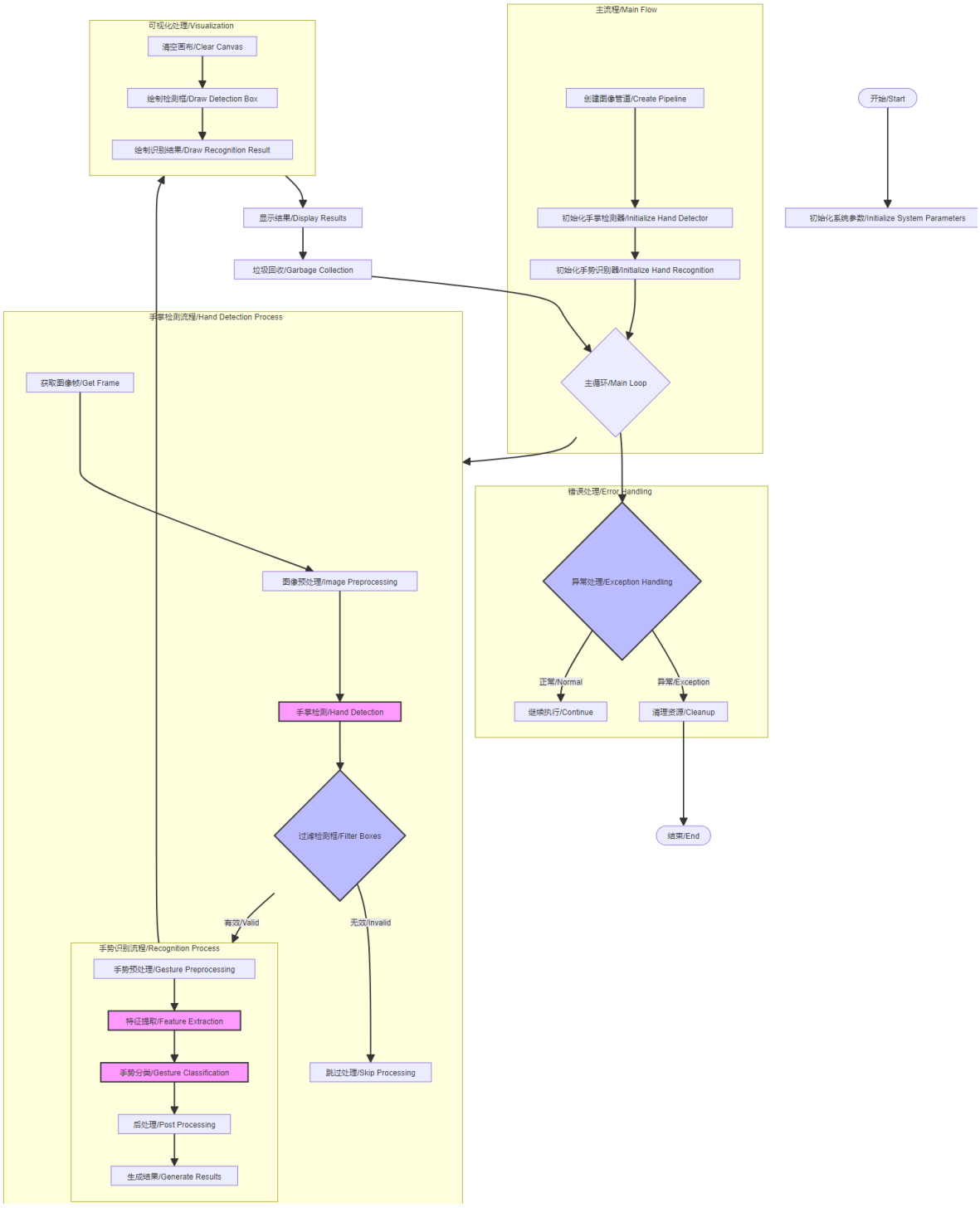
- Initialization Phase:
  - System parameter configuration
  - Create image pipeline
  - Initialize detectors and recognizers
- Hand Detection Process:
  - Get image frame
  - Image preprocessing
  - Hand detection
  - Detection box filtering
- Gesture Recognition Process:
  - Gesture preprocessing
  - Feature extraction
  - Gesture classification
  - Post processing
- Visualization:
  - Clear canvas
  - Draw detection box
  - Draw recognition results
  - Display results
- Error Handling:
  - Exception catching
  - Resource cleanup

Key part code

```
    # run函数 - 执行手掌检测和手势识别的主流程
    # Run function - main process for hand detection and gesture recognition
    def run(self, input_np):
        # 执行手掌检测
        # Perform hand detection
        det_boxes = self.hand_det.run(input_np)
        hand_rec_res = []  # 存储手势识别结果 / Store gesture recognition results
        hand_det_res = []  # 存储有效的手掌检测结果 / Store valid hand detection
results

        for det_box in det_boxes:
            # 对检测到的每一个手掌执行手势识别
            # Perform gesture recognition for each detected hand
            x1, y1, x2, y2 = det_box[2], det_box[3], det_box[4], det_box[5]
            w, h = int(x2 - x1), int(y2 - y1)

            # 过滤掉一些不合理的检测框
            # Filter out unreasonable detection boxes

            # 过滤高度太小的检测框 / Filter detection boxes with too small height
            if (h < (0.1 * self.rgb888p_size[1])):
                continue

            # 过滤在边缘且宽度较小的检测框 / Filter detection boxes at edges with
small width
```

```
            if (w < (0.25 * self.rgb888p_size[0]) and ((x1 < (0.03 *
self.rgb888p_size[0])) or (x2 > (0.97 * self.rgb888p_size[0])))):
                continue

            # 过滤在极端边缘且宽度很小的检测框 / Filter detection boxes at extreme
edges with very small width
            if (w < (0.15 * self.rgb888p_size[0]) and ((x1 < (0.01 *
self.rgb888p_size[0])) or (x2 > (0.99 * self.rgb888p_size[0])))):
                continue

            # 为当前检测框配置预处理并执行手势识别
            # Configure preprocessing for current detection box and perform
gesture recognition
            self.hand_rec.config_preprocess(det_box)
            text = self.hand_rec.run(input_np)

            # 保存有效的检测结果和识别结果
            # Save valid detection and recognition results
            hand_det_res.append(det_box)
            hand_rec_res.append(text)

        return hand_det_res, hand_rec_res
```

flow chart


主流程/Main Flow
开始/Start
初始化系统参数/Initialize System Parameters
创建图像管道/Create Pipeline
初始化手掌检测器/Initialize Hand Detector
初始化手势识别器/Initialize Hand Recognition
主循环/Main Loop
手掌检测流程/Hand Detection Process
获取图像帧/Get Frame
图像预处理/Image Preprocessing
手掌检测/Hand Detection
过滤检测框/Filter Boxes
有效/Valid
无效/Invalid
手势识别流程/Recognition Process
手势预处理/Gesture Preprocessing
特征提取/Feature Extraction
手势分类/Gesture Classification
后处理/Post Processing
生成结果/Generate Results
跳过处理/Skip Processing
可视化处理/Visualization
清空画布/Clear Canvas
绘制检测框/Draw Detection Box
绘制识别结果/Draw Recognition Result
显示结果/Display Results
垃圾回收/Garbage Collection
错误处理/Error Handling
异常处理/Exception Handling
正常/Normal
异常/Exception
继续执行/Continue
清理资源/Cleanup
结束/End