北京理工大學

信号与信息处理课程设计实验报告

信号产生、分析与处理软件系统设计

学	院：	信息与电子学院
专	业：	电子信息工程（徐特立英才班）
班	级：	[redacted]
姓	名：	[redacted]
学	号：	[redacted]

目录

实验六 VR 眼动数据的目标分类 1
一、 实验题目 1
1. 基本知识 1
2. 实验过程 1
3. 数据格式 2
4. 数据分析 2
5. 目标分类 3
二、 实验原理与方法 3
1. 眼动仪工作基础 3
2. 目标分类的核心算法 3
3. 关键 MATLAB 函数 4
三、 实验结果与分析 4
1. 软件系统界面设计 4
1.1. 控制面板 5
1.2. 结果显示面板 5
2. 10 个目标校准点坐标 5
2.1. 求解结果 5
2.2. 分析讨论 6
3. 正式实验阶段的目标分类 6
3.1. 频谱分析结果 6
3.2. 分析讨论 7
四、 实验总结 7
五、 程序代码 8

实验六 VR 眼动数据的目标分类

一、 实验题目

1. 基本知识

眼动仪是一种能够跟踪测量眼球位置及眼球运动信息的一种设备，在视觉系统、心理学、认知语言学的研究中有广泛的应用。通用设备原理为角膜反射法。

以最基本的桌面式眼动仪为例，由于眼动仪（含摄像机）的位置固定，屏幕光源（被试所看的刺激材料）的位置也固定、眼球中心位置不变（假设眼球为球状，且头部不动），普尔钦斑的绝对位置并不随眼球的转动而变化。但其相对于瞳孔和眼球的位置则是在不断变化 的——比如，当你盯着摄像头时，普尔钦斑就在你瞳孔之间；而当你抬起头时，普尔钦斑就 在你的瞳孔下方。这样一来，只要实时定位眼睛图像上的瞳孔和普尔钦斑的位置，计算出角膜反射向量，便能利用几何模型，估算得到用户的视线方向。再基于前期定标过程（即让用户注视电脑屏幕上特定的点）中所建立的用户眼睛特征与电脑屏幕呈现内容之间的关系，眼动仪就能判断出用户究竟在看屏幕上的什么内容了。

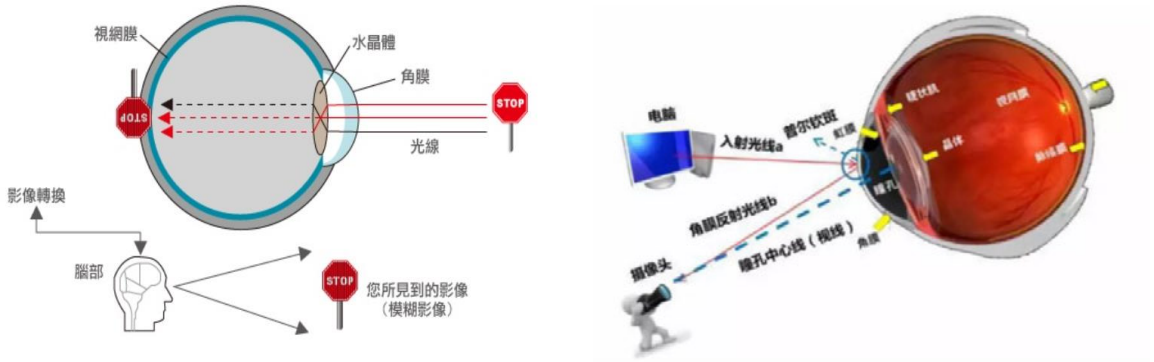

图 1：眼动信号采集原理

眼动的本质是人注意力资源的主动或被动分配，选择更有用或吸引力的信息。其应用方向有：用户体验与交互研究（网页可用性、移动端可用性、软件可用性、视线交互、游戏可用性研究）、市场研究与消费者调研等。

2. 实验过程

实验数据采集分为两个过程，第一个是眼动校准阶段，要求实验参与者依次观看界面上的数字 0-9，如下图 2 所示，每个目标 1 个试次，校准阶段共 10 个试次。所采集的数据作为参与者的模板数据，用于正式阶段的分类。第二个是正式实验阶段，采集过程连续进行，数据中包含开始和结束的位置索引，每个目标 2 个试次，共 20 个试次。本题任务是根据采集的眼动数据实现 10 个目标的分类。

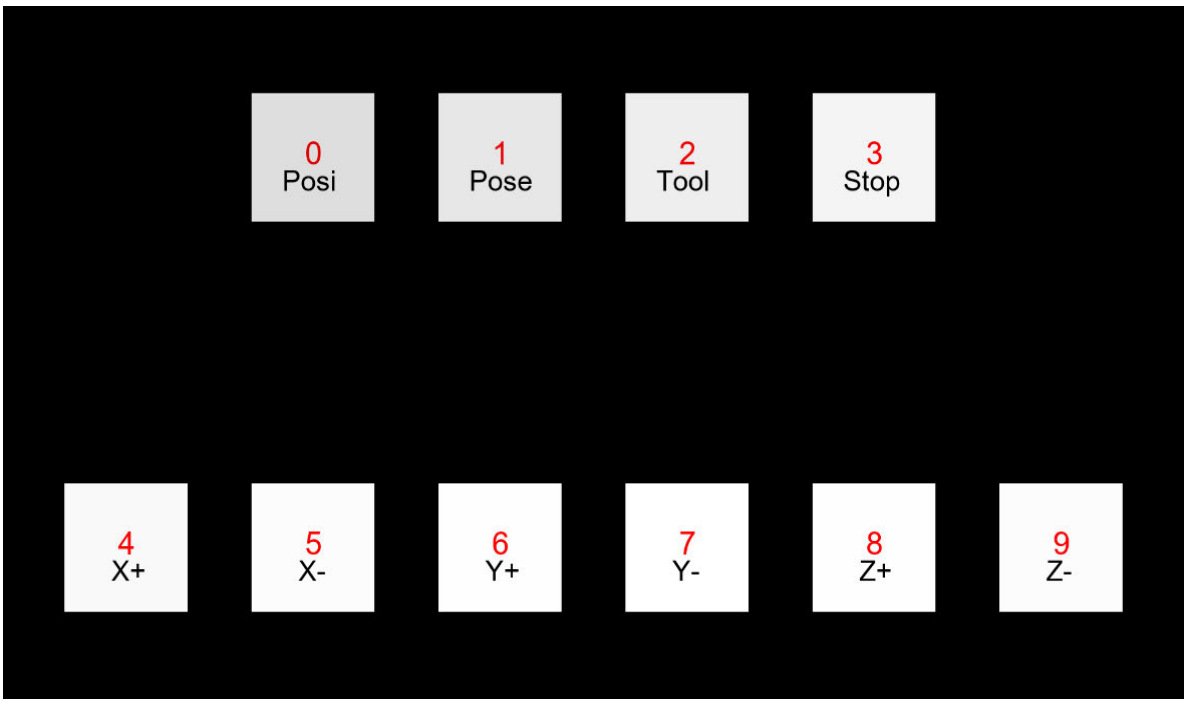

图 2：目标刺激位置的界面布局

3. 数据格式

利用校准阶段的眼动数据，对正式实验阶段的眼动数据进行分类，并计算目标分类的准确率。眼动数据包括左眼和右眼两部分数据，可参见附件数据包：calib_data.mat 文件包含 Left_pre_c、Right_pre_c; ——校准阶段预处理后的左右眼数据。online.mat 文件包含 Left_pre_o、Right_pre_o; ——正式阶段预处理后的左右眼数据。数据维度：试次*特征维度*时域采样点；一个试次对应一个指令(即一个目标)，特征维度为 X,Y 轴坐标数据，时域采样点为一段时间内的眼动数据。如 20*2*40 是指 20 个试次，一眼 XY 两轴坐标数据，40 个时间点。校准阶段的数据标签为[1 2 3 4 5 6 7 8 9 10]，正式实验阶段的数据标签为 20 个 1-10 的数值。

4. 数据分析

(1) 先对校准数据 10 个试次中每个试次 left_pre_c 40 个时间点的数据进行平均得到一个值，这样 10 个目标各自有一个平均值，作为 10 个目标 xy 坐标的校准点（10*2）。左右眼数据可分别处理。

(2) 对正式实验数据 left_pre_o，每个试次下 40 个时间点中每一时间点都与 10 个校准数据点求距离，查看距离最小的对应目标是什么，然后查看 40 个时间点中所得分类结果（目标投票最多的是哪一个），就确定是哪个目标。最后与标签 label 比对并计算准确率。相同方法对右眼数据进行处理，也可将左右眼的数据融合后按此方法处理。

5. 目标分类

(1) 求正式实验阶段目标分类的准确率，左右眼分别计算分类准确率，然后左右眼融合计算分类准确率。

(2) 分析不同时间点 10、20、30、40 条件下的分类准确率（理论上时间点越多准确率越高），并画出曲线。

(3) GUI 界面呈现校准阶段 10 个试次各自的中心位置（XY 坐标），以及正式实验阶段 20 个试次目标分类类别，如左眼分类准确率、右眼分类准确率、双眼融合分类准确率（可表格呈现）。另外呈现不同时间点下的分类准确率曲线（如 10 个点、20 个点、30 个点、40 个点）。

二、 实验原理与方法

1. 眼动仪工作基础

眼动仪是跟踪测量眼球位置及运动信息的设备，其核心工作原理依赖普尔钦斑（Purkinje 斑）——光束入射眼球过程中，在角膜、晶状体等各层膜的前后面形成的反射影像。通过捕捉该影像的位置变化，可精确获取眼球的 xy 坐标数据，进而反映视线指向。

眼动的本质是人体注意力资源的主动或被动分配，当实验参与者注视特定目标时，其眼球坐标会稳定在目标区域附近。基于这一特性，可通过对比眼动坐标与目标基准坐标的匹配程度，实现目标分类。

2. 目标分类的核心算法

VR 眼动数据目标分类的核心算法包含校准阶段、分类阶段两个步骤，具体原理如下：

校准阶段：通过让参与者依次注视 10 个目标（数字 0-9），采集每个目标对应的眼动坐标，得到校准数据 left_pre_c,right_pre_c，这两个数据分别为校准阶段左右眼在 10 个试次下 40 个时间点的 xy 坐标值。先对双眼校准数据 10 个试次每个试次 40 个时间点的数据求平均值，这样 10 个分类目标各自对应一个平均值，分别为这 10 个目标的校准点。

测试阶段：在正式实验中，需要对 left_pre_o,righ_pre_o 测试数据中 20 个试次下 40 个时间点的眼动数据进行分类。测试阶段分别选取了前 10、20、30、40 个时间点进行分类，理论上选取的时间点数越多分类准确率越高。每个试次每个时间点的眼动坐标与 10 个目标的校准点计算距离，距离最小的目标即为该试次该时间点的分类结果，并对该结果进行投票。统计该试次所有时间点的分类结果，投票数最多的目标即为最

终分类结果。实验中对左右眼的视线数据分别进行上述的投票分类，得到左右眼各自的分类结果，这就是单眼分类的方法。实验同时进行了双眼融合分类，将同一个试次左右眼的投票结果叠加后再进行统计，此时总票数变为单眼分类的两倍，投票数最多的目标即为最终分类结果。由于左右眼的视线数据具有互补性，双眼融合分类的结果准确度更高。

这里的距离可用欧氏距离来衡量，计算公式为：

$$d = \sqrt{(x - x_0)^2 + (y - y_0)^2} \tag{1}$$

其中(x, y)为正式实验中某一试次某一时间点的眼动坐标，(x_0, y_0)为某一目标的校准点坐标。欧氏距离能直观反映两点在二维平面的空间差异，距离越小说明视线越接近该目标。

3. 关键 MATLAB 函数

- squeeze：压缩矩阵单元素维度；
- mean：计算数组均值；
- sum：计算数组元素之和；
- sqrt：计算数组平方根；
- max：计算数组最大值及对应索引；
- size：获取数组维度信息；

三、 实验结果与分析

1. 软件系统界面设计

本系统基于 MATLAB App Designer 开发，采用左右分栏布局，左侧为控制面板，右侧为结果显示面板。软件系统界面简洁直观、功能分区明确，操作流程清晰，可视化效果直观，可以完成 VR 眼动数据的目标分类的基本功能。直接分类打开系统，未经任何操作的初始化软件界面如下图所示：

图 3：软件系统界面

1.1. 控制面板

控制面板根据实验的操作流程从上往下布局，各组件之间的功能和相互之间的逻辑关系紧密结合。最上面是“加载数据”按钮，点击该按钮即可选择文件夹加载其中的 VR 眼动数据集，若加载失败则会弹窗提示错误原因，数据加载完成后会显示数据所在路径。“加载数据”按钮下面是“选择时间点数”下拉框，通过该下拉框可选择正式实验中每试次的眼动数据中用于分类的时间点数。控制面板的中间为“开始处理”按钮，该按钮只有在数据加载成功时才被启用，点击该按钮进行眼动数据分类，分类完成后自动填写下方的“分类准确率”表格。控制面板的最下方是“关闭界面”按钮，点击即可关闭界面。

1.2. 结果显示面板

结果显示面板的上部包含两个水平排列的坐标区，下部包含两个水平排列的表格。左上方的坐标区用于绘制 10 个目标校准点坐标图，左下方的表格列举了 10 个目标校准点 xy 坐标值，点击“加载数据”按钮后即可自动绘制和填写。右上方的坐标区用于绘制不同分类方法下的准确率曲线，右下方的表格列举了不同分类方法的分类结果，分类时间点数由控制面板中的“选择时间点数”下拉框决定，点击“开始处理”按钮后即可自动绘制和填写。

2. 10 个目标校准点坐标

2.1. 求解结果

点击“加载数据”按钮加载眼动数据，系统可自动求解左右眼 10 个目标的校准点坐标并绘图展示，如图 4 所示：

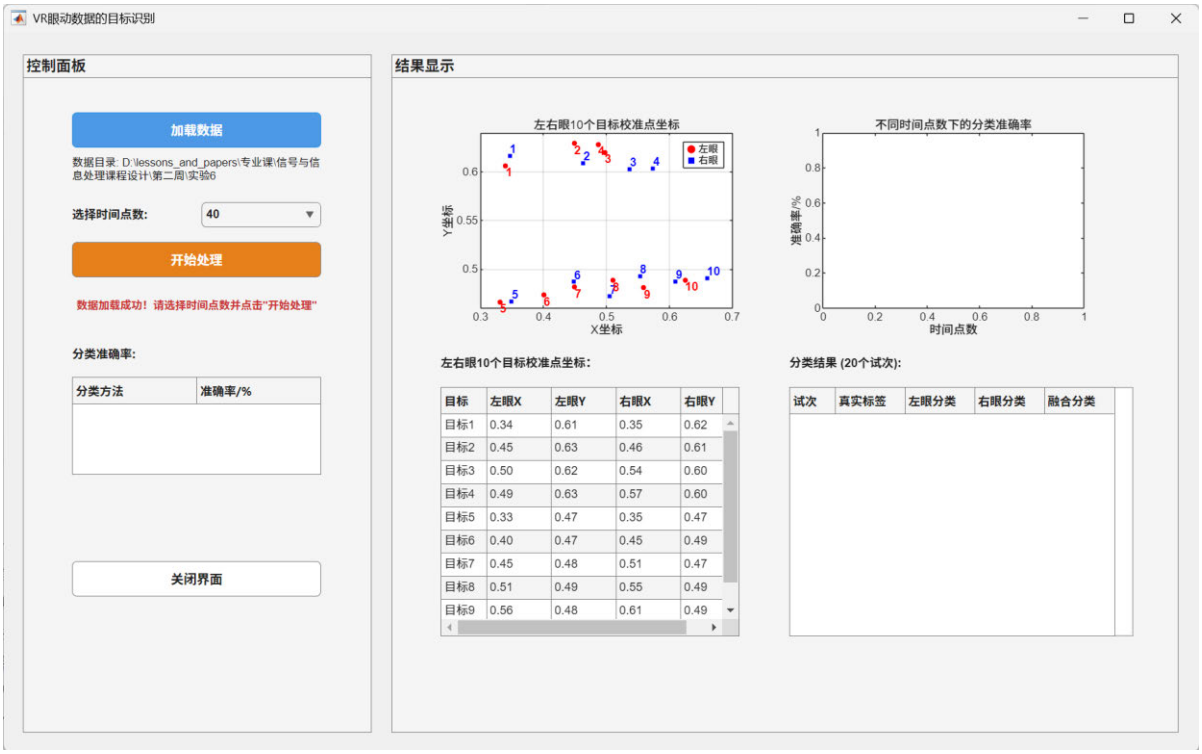

图 4：10 个目标校准点求解结果

2.2. 分析讨论

通过对校准阶段 10 个试次 40 个时间点的左右眼坐标求平均，可以得到 10 个目标的 xy 坐标校准点。校准阶段充分利用了每个试次各个时间点左右眼的坐标数据，求得的坐标点可以很好地反映每个目标左右眼 xy 坐标的平均位置，尽可能消除抖动等其他干扰的影响，为之后测试阶段的目标分类提供了可靠的依据。

3. 正式实验阶段的目标分类

3.1. 频谱分析结果

求解完 10 个目标的校准点坐标后，选择时间点数为 40，点击“开始处理”按钮，系统即可进行目标分类。分类结束后在控制面板的“分类准确率”表格展示三种分类方法的分类结果，在结果显示面板的“分类结果”表格展示 20 个试次三种分类方法的分类结果，与此同时绘制三种分类方法的准确率随时间点数变化曲线图，如图 5 所示：

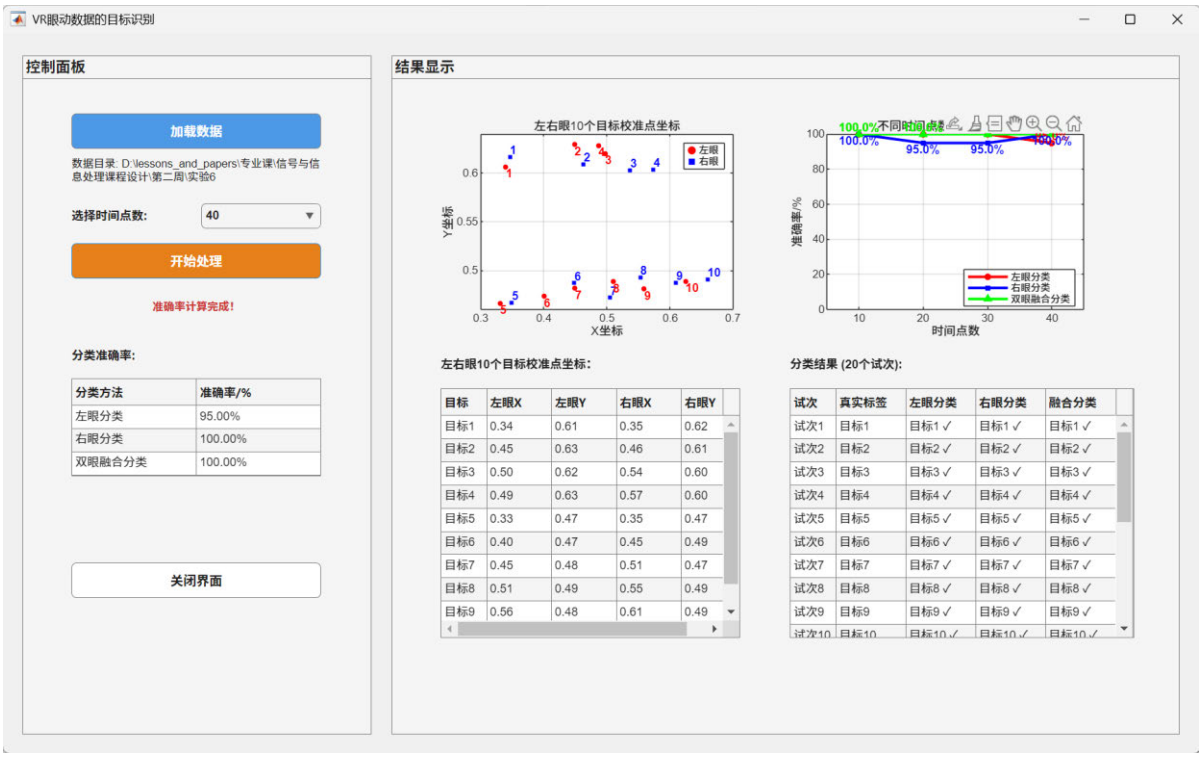

图 5：目标分类结果

注意上图中两个表格仅是时间点数为 40 的分类结果，将时间点数更改为 10、20、30 后分类准确率和分类结果表格会分别展示对应的结果，这里将不再展示。

3.2. 分析讨论

观察时间点数为 40 的分类结果，可知左眼分类、右眼分类和双眼融合分类的准确率分别达到了 95%、100%、100%，均具有很高的准确率，这充分说明了上述分类方法的合理性。观察结果显示面板右上方的三种分类方法的准确率随时间点数变化曲线图，发现左眼分类和右眼分类的准确率并不严格随着时间点数的增长而不断提高，而是出现了一定的波动。这是因为不同时间点采集到的眼动坐标可能受到不同的干扰，在一定程度上对分类结果造成了影响。但是由于分类的准确率均达到了 95%以上，表明这种分类方法的可靠性已经非常高，这种波动可以忽略不计。与此同时还可以发现，尽管左眼分类和右眼分类的准确率均存在波动，但双眼融合分类的准确率始终在 100%。这是因为左右眼的视线数据具有很强的互补性，将双眼在每个试次下不同时间点的分类结果放在一起投票可以更充分地利用左右眼的眼动坐标数据，消除其他干扰对分类结果的影响，从而提高分类的准确率。

四、 实验总结

通过本次 VR 眼动数据目标分类实验，我收获了信号处理和数据分类的专业知识与 MATLAB App Designer 开发实践经验，对虚拟现实人机交互核心技术的原理与数据处理方法有了更为透彻的理解。实验过程中，我不仅深入掌握了眼动仪普尔钦斑追踪

原理、注视点坐标特征提取方法，还熟练运用 MATLAB 实现了校准中心点计算、欧氏距离匹配、多时间点投票分类等核心算法，成功开发出模块化的 VR 眼动数据目标分类 GUI 系统。

其中，双眼融合分类的投票算法是我在实验中遇到的最大问题。在刚开始的开发实践中，我误以为双眼融合分类应当先将双眼的坐标数据求加权平均后再进行投票，后经[redacted]的指正下我意识到了自己的错误。由于双眼的坐标数据是相互独立的，如果直接加权平均会导致单眼数据的异常点对融合结果造成很大的不利影响。与此同时，这种方法在最后的投票决策时仍然只有 40 票，无法充分利用双眼的坐标数据。在[redacted]的指导下，我意识到正确的双眼融合方法应当是将左右眼分类投票的结果进行叠加后再统计票数最高的目标类别，这样总票数增加到了 80 票，可以充分利用双眼的坐标信息，并且可以尽可能降低数据异常点对分类的影响。

在本次实验中，我亲手编写程序实现了欧氏距离计算和度量以及投票分类算法，深切感受到了理论知识与编程实践相结合的重要意义。实验中我不仅验证了多时间点采样对干扰抑制的提升效果，观察到了校准精度对分类结果的决定性影响，还通过双眼融合技术验证了多模态数据融合的优势，进一步加深了对眼动轨迹特征与分类算法的理解，真切感受到该技术在 VR 购物、智能界面设计等领域的应用价值。

最后，我要衷心感谢指导[redacted]的悉心指导。[redacted]在课堂上对眼动仪校准原理和投票逻辑算法进行了细致讲解，并指出了我在实验中遇到的问题和错误，帮助我顺利完成了这次实验。通过本次实验，我不仅夯实了数字信号处理与眼动追踪技术的基础理论，积累了算法优化与 GUI 开发的实践经验，更培养了科学严谨的实验态度和自主调试的问题解决思维。这些收获将为我后续深入学习信号处理相关技术和机器学习进阶算法、开展虚拟现实人机交互相关科研实践奠定坚实基础，对未来的专业学习与发展产生积极而深远的影响。

五、 程序代码

```
classdef exp6 < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        VRUIFigure          matlab.ui.Figure
        RightPanel          matlab.ui.container.Panel
        TestResultTable     matlab.ui.control.Table
        TestResultLabel     matlab.ui.control.Label
        CalibLabel          matlab.ui.control.Label
        CalibTable          matlab.ui.control.Table
        AccuracyCurveAxes   matlab.ui.control.UIAxes
```

```
        CalibAxes            matlab.ui.control.UIAxes
        LeftPanel            matlab.ui.container.Panel
        DeleteButton         matlab.ui.control.Button
        AccuracyTable        matlab.ui.control.Table
        AccuracyLabel        matlab.ui.control.Label
        StatusLabel          matlab.ui.control.Label
        ProcessButton        matlab.ui.control.Button
        TimePointsDropdown   matlab.ui.control.DropDown
        TimePointsLabel      matlab.ui.control.Label
        FilePathLabel        matlab.ui.control.Label
        LoadDataButton       matlab.ui.control.Button
    end


    properties (Access = private)
        Property % Description
        % 数据属性
        data_dir                    char     % 数据目录
        left_pre_c                  double   % 校准阶段左眼数据 (10×2×40)
        right_pre_c                 double   % 校准阶段右眼数据 (10×2×40)
        left_pre_o                  double   % 测试阶段左眼数据 (20×2×40)
        right_pre_o                 double   % 测试阶段右眼数据 (20×2×40)
        labels                      double   % 标签数据 (20×1)

        % 计算结果
        calib_centers_left          double   % 左眼校准中心点 (10×2)
        calib_centers_right         double   % 右眼校准中心点 (10×2)
        results                     struct   % 分类结果结构体
    end

    methods (Access = private)

        function calculateCalibrationCenters(app)
            % 初始化中心点矩阵
            app.calib_centers_left = zeros(10, 2);
            app.calib_centers_right = zeros(10, 2);

            % 计算每个目标的平均坐标
            for i = 1:10
                % 左眼数据平均
                left_data = squeeze(app.left_pre_c(i, :, :));  % 2×40
                app.calib_centers_left(i, 1) = mean(left_data(1, :));  % X 坐标
                app.calib_centers_left(i, 2) = mean(left_data(2, :));  % Y 坐标

```

```
                % 右眼数据平均
                right_data = squeeze(app.right_pre_c(i, :, :));  % 2×40
                app.calib_centers_right(i, 1) = mean(right_data(1, :));  % X 坐标
                app.calib_centers_right(i, 2) = mean(right_data(2, :));  % Y 坐标
            end
        end

        function classifyTargets(app, time_points)
            num_trials = size(app.left_pre_o, 1);
            predictions_left = zeros(num_trials, 1);
            predictions_right = zeros(num_trials, 1);
            predictions_fused = zeros(num_trials, 1);

            % 对每个测试试次进行分类
            for trial = 1:num_trials
                % ========== 左眼分类 ==========
                left_votes = zeros(10, 1);
                left_data = squeeze(app.left_pre_o(trial, :, 1:time_points));  % 2×time_points

                for t = 1:time_points
                    point = left_data(:, t);  % 当前时间点的坐标
                    distances = zeros(10, 1);

                    % 计算与 10 个校准中心点的距离
                    for target = 1:10  % 目标编号 1-10
                        distances(target) = sqrt(sum((point - app.calib_centers_left(target, :)').^2));
                    end

                    % 找到最小距离对应的目标
                    [~, min_idx] = min(distances);
                    left_votes(min_idx) = left_votes(min_idx) + 1;
                end

                % 投票最多的目标作为预测结果
                [~, predictions_left(trial)] = max(left_votes);

                % ========== 右眼分类 ==========
                right_votes = zeros(10, 1);
                right_data = squeeze(app.right_pre_o(trial, :, 1:time_points));  % 2×time_points

                for t = 1:time_points
                    point = right_data(:, t);
```

```
98.                     distances = zeros(10, 1);
99.
100.                    for target = 1:10  % 目标编号 1-10
101.                        distances(target) = sqrt(sum((point - app.calib_centers_right(
    target, :)').^2));
102.                    end
103.
104.                    % 找到最小距离对应的目标
105.                    [~, min_idx] = min(distances);
106.                    right_votes(min_idx) = right_votes(min_idx) + 1;
107.                end
108.
109.                [~, predictions_right(trial)] = max(right_votes);
110.
111.                % ========== 双眼融合分类 ==========
112.                fused_votes = right_votes + left_votes;
113.                [~, predictions_fused(trial)] = max(fused_votes);
114.            end
115.
116.            % 计算准确率
117.            accuracy_left = sum(predictions_left == app.labels) / num_trials * 100;
118.            accuracy_right = sum(predictions_right == app.labels) / num_trials * 100;
119.            accuracy_fused = sum(predictions_fused == app.labels) / num_trials * 100;
120.
121.            % 保存结果
122.            app.results = struct(...
123.                'predictions_left', predictions_left, ...
124.                'predictions_right', predictions_right, ...
125.                'predictions_fused', predictions_fused, ...
126.                'accuracy_left', accuracy_left, ...
127.                'accuracy_right', accuracy_right, ...
128.                'accuracy_fused', accuracy_fused, ...
129.                'time_points', time_points);
130.        end
131.
132.        function [time_points_list, acc_left, acc_right, acc_fused] = calculateAccurac
    yCurve(app)
133.            time_points_list = [10, 20, 30, 40];
134.            acc_left = zeros(4, 1);
135.            acc_right = zeros(4, 1);
136.            acc_fused = zeros(4, 1);
137.
```

```
            % 显示进度
            app.StatusLabel.Text = '正在计算不同时间点的准确率...';
            drawnow;

            for i = 1:4
                classifyTargets(app, time_points_list(i));
                acc_left(i) = app.results.accuracy_left;
                acc_right(i) = app.results.accuracy_right;
                acc_fused(i) = app.results.accuracy_fused;
            end

            % 计算完成后更新状态
            app.StatusLabel.Text = '准确率计算完成！';
        end

        function updateCalibrationDisplay(app)
            % 清除坐标轴
            cla(app.CalibAxes);

            % 在坐标轴上显示校准中心点
            scatter(app.CalibAxes, ...
                app.calib_centers_left(:,1), app.calib_centers_left(:,2), ...
                15, 'ro', 'filled', 'DisplayName', '左眼');
            hold(app.CalibAxes, 'on');
            scatter(app.CalibAxes, ...
                app.calib_centers_right(:,1), app.calib_centers_right(:,2), ...
                15, 'bs', 'filled', 'DisplayName', '右眼');

            % 添加目标编号标签（1-10）
            for i = 1:10
                text(app.CalibAxes, app.calib_centers_left(i,1), ...
                    app.calib_centers_left(i,2), sprintf('%d', i), ...
                    'HorizontalAlignment', 'left', 'VerticalAlignment', 'top', ...
                    'FontWeight', 'bold', 'Color', 'r');
                text(app.CalibAxes, app.calib_centers_right(i,1), ...
                    app.calib_centers_right(i,2), sprintf('%d', i), ...
                    'HorizontalAlignment', 'left', 'VerticalAlignment', 'bottom', ...
                    'FontWeight', 'bold', 'Color', 'b');
            end

            hold(app.CalibAxes, 'off');

            % 设置坐标轴属性
            legend(app.CalibAxes, 'Location', 'northeast');
```

```
182.             xlabel(app.CalibAxes, 'X 坐标');
183.             ylabel(app.CalibAxes, 'Y 坐标');
184.             title(app.CalibAxes, '左右眼 10 个目标校准点坐标');
185.             grid(app.CalibAxes, 'on');
186.
187.             % 更新校准表格数据
188.             table_data = cell(10, 5);
189.             for i = 1:10
190.                 table_data{i, 1} = sprintf('目标%d', i);
191.                 table_data{i, 2} = sprintf('%.2f', app.calib_centers_left(i, 1));
192.                 table_data{i, 3} = sprintf('%.2f', app.calib_centers_left(i, 2));
193.                 table_data{i, 4} = sprintf('%.2f', app.calib_centers_right(i, 1));
194.                 table_data{i, 5} = sprintf('%.2f', app.calib_centers_right(i, 2));
195.             end
196.             app.CalibTable.Data = table_data;
197.         end
198.
199.         function updateResultsDisplay(app)
200.             % 更新准确率表格
201.             accuracy_data = {
202.                 '左眼分类', sprintf('%.2f%%', app.results.accuracy_left);
203.                 '右眼分类', sprintf('%.2f%%', app.results.accuracy_right);
204.                 '双眼融合分类', sprintf('%.2f%%', app.results.accuracy_fused)
205.                 };
206.             app.AccuracyTable.Data = accuracy_data;
207.
208.             % 更新测试结果表格
209.             test_data = cell(20, 5);
210.             for i = 1:20
211.                 test_data{i, 1} = sprintf('试次%d', i);
212.                 test_data{i, 2} = sprintf('目标%d', app.labels(i));
213.                 test_data{i, 3} = sprintf('目
    标%d', app.results.predictions_left(i));
214.                 test_data{i, 4} = sprintf('目
    标%d', app.results.predictions_right(i));
215.                 test_data{i, 5} = sprintf('目
    标%d', app.results.predictions_fused(i));
216.
217.                 % 高亮显示正确的结果
218.                 if app.results.predictions_left(i) == app.labels(i)
219.                     test_data{i, 3} = [test_data{i, 3}, ' ✓'];
220.                 else
221.                     test_data{i, 3} = [test_data{i, 3}, ' X'];
222.                 end
223.                 if app.results.predictions_right(i) == app.labels(i)
```

```
                    test_data{i, 4} = [test_data{i, 4}, ' ✓'];
                else
                    test_data{i, 3} = [test_data{i, 3}, ' X'];
                end
                if app.results.predictions_fused(i) == app.labels(i)
                    test_data{i, 5} = [test_data{i, 5}, ' ✓'];
                else
                    test_data{i, 3} = [test_data{i, 3}, ' X'];
                end
            end
            app.TestResultTable.Data = test_data;
        end

        function updateAccuracyCurve(app)
            % 计算不同时间点下的准确率
            [time_points_list, acc_left, acc_right, acc_fused] = calculateAccuracyCurve(app);

            % 清除坐标轴
            cla(app.AccuracyCurveAxes);

            % 绘制曲线
            plot(app.AccuracyCurveAxes, time_points_list, acc_left, ...
                'ro-', 'LineWidth', 2, 'MarkerSize', 3, 'DisplayName', '左眼分类');
            hold(app.AccuracyCurveAxes, 'on');
            plot(app.AccuracyCurveAxes, time_points_list, acc_right, ...
                'bs-', 'LineWidth', 2, 'MarkerSize', 3, 'DisplayName', '右眼分类');
            plot(app.AccuracyCurveAxes, time_points_list, acc_fused, ...
                'g^-', 'LineWidth', 2, 'MarkerSize', 3, 'DisplayName', '双眼融合分类');
            hold(app.AccuracyCurveAxes, 'off');

            % 设置坐标轴属性
            xlabel(app.AccuracyCurveAxes, '时间点数');
            ylabel(app.AccuracyCurveAxes, '准确率/%');
            title(app.AccuracyCurveAxes, '不同时间点数下的分类准确率');
            legend(app.AccuracyCurveAxes, 'Location', 'best');
            grid(app.AccuracyCurveAxes, 'on');
            xlim(app.AccuracyCurveAxes, [5, 45]);
            ylim(app.AccuracyCurveAxes, [0, 100]);

            % 添加准确率数值标签
            for i = 1:4
                text(app.AccuracyCurveAxes, time_points_list(i), acc_left(i), ...
```

```
                    sprintf('%.1f%%', acc_left(i)), 'HorizontalAlignment', 'center', ...
                    'VerticalAlignment', 'bottom', 'Color', 'r', 'FontWeight', 'bold');
                text(app.AccuracyCurveAxes, time_points_list(i), acc_right(i), ...
                    sprintf('%.1f%%', acc_right(i)), 'HorizontalAlignment', 'center', ...
                    'VerticalAlignment', 'top', 'Color', 'b', 'FontWeight', 'bold');
                text(app.AccuracyCurveAxes, time_points_list(i), acc_fused(i), ...
                    sprintf('%.1f%%', acc_fused(i)), 'HorizontalAlignment', 'center', ...
                    'VerticalAlignment', 'bottom', 'Color', 'g', 'FontWeight', 'bold');
            end
        end
    end


    % Callbacks that handle component events
    methods (Access = private)

        % Button pushed function: LoadDataButton
        function LoadDataButtonPushed(app, event)
            try
                % 显示状态
                app.StatusLabel.Text = '正在选择数据目录...';
                drawnow;

                % 选择数据目录
                app.data_dir = uigetdir(pwd, '选择数据文件所在目录');
                if app.data_dir == 0
                    app.StatusLabel.Text = '用户取消选择目录';
                    return;
                end

                % 显示状态
                app.StatusLabel.Text = '正在加载数据...';
                app.FilePathLabel.Text = ['数据目录: ', app.data_dir];
                drawnow;

                % 加载 calib_data.mat
                calib_file = fullfile(app.data_dir, 'calib_data.mat');
                if exist(calib_file, 'file')
                    calib_data = load(calib_file);

```

```
306.                     % 注意变量名是 Left_pre_c 和 Right_pre_c
307.                     if isfield(calib_data, 'Left_pre_c')
308.                         app.left_pre_c = calib_data.Left_pre_c;
309.                     elseif isfield(calib_data, 'left_pre_c')
310.                         app.left_pre_c = calib_data.left_pre_c;
311.                     else
312.                         error('calib_data.mat 中找不到左眼校准数据');
313.                     end
314.
315.                     if isfield(calib_data, 'Right_pre_c')
316.                         app.right_pre_c = calib_data.Right_pre_c;
317.                     elseif isfield(calib_data, 'right_pre_c')
318.                         app.right_pre_c = calib_data.right_pre_c;
319.                     else
320.                         error('calib_data.mat 中找不到右眼校准数据');
321.                     end
322.                 else
323.                     error('找不到 calib_data.mat 文件');
324.                 end
325.
326.                 % 加载 online.mat
327.                 online_file = fullfile(app.data_dir, 'online.mat');
328.                 if exist(online_file, 'file')
329.                     online_data = load(online_file);
330.
331.                     % 注意变量名是 Left_pre_o 和 Right_pre_o
332.                     if isfield(online_data, 'Left_pre_o')
333.                         app.left_pre_o = online_data.Left_pre_o;
334.                     elseif isfield(online_data, 'left_pre_o')
335.                         app.left_pre_o = online_data.left_pre_o;
336.                     else
337.                         error('online.mat 中找不到左眼测试数据');
338.                     end
339.
340.                     if isfield(online_data, 'Right_pre_o')
341.                         app.right_pre_o = online_data.Right_pre_o;
342.                     elseif isfield(online_data, 'right_pre_o')
343.                         app.right_pre_o = online_data.right_pre_o;
344.                     else
345.                         error('online.mat 中找不到右眼测试数据');
346.                     end
347.                 else
348.                     error('找不到 online.mat 文件');
349.                 end
350.
```

```
                % 加载 label.mat
                label_file = fullfile(app.data_dir, 'label.mat');
                if exist(label_file, 'file')
                    label_data = load(label_file);

                    % 注意变量名是 label
                    if isfield(label_data, 'label')
                        label_matrix = label_data.label;  % 10×2 的标签矩阵

                        % 将 10×2 的标签矩阵转换为 20×1 的向量（按列顺序）
                        % 第一列：前 10 个试次，第二列：后 10 个试次
                        app.labels = label_matrix(:);  % 按列展开

                        % 确保标签在 1-10 范围内
                        if min(app.labels) < 1 || max(app.labels) > 10
                            error('标签值超出 1-10 范围');
                        end
                    else
                        error('label.mat 中找不到 label 变量');
                    end
                else
                    error('找不到 label.mat 文件');
                end

                % 验证数据维度
                if ~isequal(size(app.left_pre_c), [10, 2, 40])
                    error('left_pre_c 维度应为 10×2×40，实际维度为 %s', mat2str(size(app.left_pre_c)));
                end
                if ~isequal(size(app.right_pre_c), [10, 2, 40])
                    error('right_pre_c 维度应为 10×2×40，实际维度为 %s', mat2str(size(app.right_pre_c)));
                end
                if ~isequal(size(app.left_pre_o), [20, 2, 40])
                    error('left_pre_o 维度应为 20×2×40，实际维度为 %s', mat2str(size(app.left_pre_o)));
                end
                if ~isequal(size(app.right_pre_o), [20, 2, 40])
                    error('right_pre_o 维度应为 20×2×40，实际维度为 %s', mat2str(size(app.right_pre_o)));
                end
                if ~isequal(size(label_matrix), [10, 2])
                    error('label 矩阵维度应为 10×2，实际维度为 %s', mat2str(size(label_matrix)));
                end
```

```
391.
392.                 % 计算校准中心点
393.                 calculateCalibrationCenters(app);
394.
395.                 % 更新显示
396.                 updateCalibrationDisplay(app);
397.
398.                 % 启用处理按钮
399.                 app.ProcessButton.Enable = 'on';
400.
401.                 % 更新状态
402.                 app.StatusLabel.Text = '数据加载成功！请选择时间点数并点击"开始处理"';
403.
404.             catch ME
405.                 % 更新状态
406.                 app.StatusLabel.Text = '数据加载失败！';
407.
408.                 % 显示错误消息
409.                 uialert(app.VRUIFigure, sprintf('数据加载失败：%s', ME.message), '错误
    ', 'Icon', 'error');
410.                 return;
411.             end
412.         end
413.
414.         % Button pushed function: ProcessButton
415.         function ProcessButtonPushed(app, event)
416.             % 获取选择的时间点数
417.             time_points = str2double(app.TimePointsDropdown.Value);
418.
419.             % 显示处理中消息
420.             d = uiprogressdlg(app.VRUIFigure, 'Title', '处理中', ...
421.                 'Message', '正在进行目标分类处理...', 'Indeterminate', 'on');
422.
423.             try
424.                 % 进行分类
425.                 classifyTargets(app, time_points);
426.
427.                 % 更新显示
428.                 updateResultsDisplay(app);
429.
430.                 % 更新准确率曲线
431.                 updateAccuracyCurve(app);
432.
433.                 % 关闭进度对话框
434.                 close(d);
```

```
435.
436.             catch ME
437.                 close(d);
438.                 uialert(app.VRUIFigure, sprintf('处理失败: %s', ME.message), '错误
    ', 'Icon', 'error');
439.                 return;
440.             end
441.         end
442.
443.         % Button pushed function: DeleteButton
444.         function DeleteButtonPushed(app, event)
445.             delete(app)
446.         end
447.     end
448.
449.     % Component initialization
450.     methods (Access = private)
451.
452.         % Create UIFigure and components
453.         function createComponents(app)
454.
455.             % Create VRUIFigure and hide until all components are created
456.             app.VRUIFigure = uifigure('Visible', 'off');
457.             app.VRUIFigure.Color = [0.9412 0.9412 0.9412];
458.             app.VRUIFigure.Position = [100 100 1200 720];
459.             app.VRUIFigure.Name = 'VR 眼动数据的目标识别 ';
460.
461.             % Create LeftPanel
462.             app.LeftPanel = uipanel(app.VRUIFigure);
463.             app.LeftPanel.Title = '控制面板';
464.             app.LeftPanel.FontWeight = 'bold';
465.             app.LeftPanel.FontSize = 15;
466.             app.LeftPanel.Position = [20 20 350 680];
467.
468.             % Create LoadDataButton
469.             app.LoadDataButton = uibutton(app.LeftPanel, 'push');
470.             app.LoadDataButton.ButtonPushedFcn = createCallbackFcn(app, @LoadDataButto
    nPushed, true);
471.             app.LoadDataButton.BackgroundColor = [0.302 0.6 0.902];
472.             app.LoadDataButton.FontSize = 13;
473.             app.LoadDataButton.FontWeight = 'bold';
474.             app.LoadDataButton.FontColor = [1 1 1];
475.             app.LoadDataButton.Position = [50 587 250 35];
476.             app.LoadDataButton.Text = '加载数据';
477.
```

```
            % Create FilePathLabel
            app.FilePathLabel = uilabel(app.LeftPanel);
            app.FilePathLabel.VerticalAlignment = 'top';
            app.FilePathLabel.WordWrap = 'on';
            app.FilePathLabel.FontSize = 11;
            app.FilePathLabel.Position = [50 547 250 32];
            app.FilePathLabel.Text = '数据目录: 未选择';

            % Create TimePointsLabel
            app.TimePointsLabel = uilabel(app.LeftPanel);
            app.TimePointsLabel.FontWeight = 'bold';
            app.TimePointsLabel.Position = [50 507 120 25];
            app.TimePointsLabel.Text = '选择时间点数:';

            % Create TimePointsDropdown
            app.TimePointsDropdown = uidropdown(app.LeftPanel);
            app.TimePointsDropdown.Items = {'10', '20', '30', '40'};
            app.TimePointsDropdown.FontWeight = 'bold';
            app.TimePointsDropdown.Position = [180 507 120 25];
            app.TimePointsDropdown.Value = '40';

            % Create ProcessButton
            app.ProcessButton = uibutton(app.LeftPanel, 'push');
            app.ProcessButton.ButtonPushedFcn = createCallbackFcn(app, @ProcessButtonP
    ushed, true);
            app.ProcessButton.BackgroundColor = [0.902 0.502 0.102];
            app.ProcessButton.FontSize = 13;
            app.ProcessButton.FontWeight = 'bold';
            app.ProcessButton.FontColor = [1 1 1];
            app.ProcessButton.Enable = 'off';
            app.ProcessButton.Position = [50 457 250 35];
            app.ProcessButton.Text = '开始处理';

            % Create StatusLabel
            app.StatusLabel = uilabel(app.LeftPanel);
            app.StatusLabel.HorizontalAlignment = 'center';
            app.StatusLabel.FontSize = 11;
            app.StatusLabel.FontWeight = 'bold';
            app.StatusLabel.FontColor = [0.8 0.2 0.2];
            app.StatusLabel.Position = [50 417 250 25];
            app.StatusLabel.Text = '请点击"加载数据"按钮';

            % Create AccuracyLabel
            app.AccuracyLabel = uilabel(app.LeftPanel);
            app.AccuracyLabel.FontWeight = 'bold';
```

```
            app.AccuracyLabel.Position = [50 367 120 25];
            app.AccuracyLabel.Text = '分类准确率:';

            % Create AccuracyTable
            app.AccuracyTable = uitable(app.LeftPanel);
            app.AccuracyTable.ColumnName = {'分类方法'; '准确率/%'};
            app.AccuracyTable.RowName = {};
            app.AccuracyTable.Position = [50 259 250 98];

            % Create DeleteButton
            app.DeleteButton = uibutton(app.LeftPanel, 'push');
            app.DeleteButton.ButtonPushedFcn = createCallbackFcn(app, @DeleteButtonPus
    hed, true);
            app.DeleteButton.BackgroundColor = [1 1 1];
            app.DeleteButton.FontSize = 13;
            app.DeleteButton.FontWeight = 'bold';
            app.DeleteButton.Position = [50 137 250 35];
            app.DeleteButton.Text = '关闭界面';

            % Create RightPanel
            app.RightPanel = uipanel(app.VRUIFigure);
            app.RightPanel.Title = '结果显示';
            app.RightPanel.FontWeight = 'bold';
            app.RightPanel.FontSize = 15;
            app.RightPanel.Position = [390 20 790 680];

            % Create CalibAxes
            app.CalibAxes = uiaxes(app.RightPanel);
            title(app.CalibAxes, '左右眼 10 个目标校准点坐标')
            xlabel(app.CalibAxes, 'X 坐标')
            ylabel(app.CalibAxes, 'Y 坐标')
            zlabel(app.CalibAxes, 'Z')
            app.CalibAxes.Box = 'on';
            app.CalibAxes.FontSize = 11;
            app.CalibAxes.Position = [50 397 300 220];

            % Create AccuracyCurveAxes
            app.AccuracyCurveAxes = uiaxes(app.RightPanel);
            title(app.AccuracyCurveAxes, '不同时间点数下的分类准确率')
            xlabel(app.AccuracyCurveAxes, '时间点数')
            ylabel(app.AccuracyCurveAxes, '准确率/%')
            zlabel(app.AccuracyCurveAxes, 'Z')
            app.AccuracyCurveAxes.Box = 'on';
            app.AccuracyCurveAxes.FontSize = 11;
            app.AccuracyCurveAxes.Position = [400 397 300 220];
```

```
566.
567.             % Create CalibTable
568.             app.CalibTable = uitable(app.RightPanel);
569.             app.CalibTable.ColumnName = {'目标'; '左眼 X'; '左眼 Y'; '右眼 X'; '右眼
    Y'};
570.             app.CalibTable.ColumnWidth = {45, 65, 65, 65, 65};
571.             app.CalibTable.RowName = {};
572.             app.CalibTable.Position = [50 97 300 250];
573.
574.             % Create CalibLabel
575.             app.CalibLabel = uilabel(app.RightPanel);
576.             app.CalibLabel.FontWeight = 'bold';
577.             app.CalibLabel.Position = [50 360 180 22];
578.             app.CalibLabel.Text = '左右眼 10 个目标校准点坐标：';
579.
580.             % Create TestResultLabel
581.             app.TestResultLabel = uilabel(app.RightPanel);
582.             app.TestResultLabel.FontWeight = 'bold';
583.             app.TestResultLabel.Position = [400 360 180 22];
584.             app.TestResultLabel.Text = '分类结果 (20 个试次):';
585.
586.             % Create TestResultTable
587.             app.TestResultTable = uitable(app.RightPanel);
588.             app.TestResultTable.ColumnName = {'试次'; '真实标签'; '左眼分类'; '右眼分类
    '; '融合分类'};
589.             app.TestResultTable.ColumnWidth = {45, 70, 70, 70, 70};
590.             app.TestResultTable.RowName = {};
591.             app.TestResultTable.Position = [400 97 345 250];
592.
593.             % Show the figure after all components are created
594.             app.VRUIFigure.Visible = 'on';
595.         end
596.     end
597.
598.     % App creation and deletion
599.     methods (Access = public)
600.
601.         % Construct app
602.         function app = exp6
603.
604.             % Create UIFigure and components
605.             createComponents(app)
606.
607.             % Register the app with App Designer
608.             registerApp(app, app.VRUIFigure)
```

```
609.
610.             if nargout == 0
611.                 clear app
612.             end
613.         end
614.
615.         % Code that executes before app deletion
616.         function delete(app)
617.
618.             % Delete UIFigure when app is deleted
619.             delete(app.VRUIFigure)
620.         end
621.     end
622. end
```