VisualRobot 调试信息手册

概述

本手册详细说明了VisualRobot项目中调试信息的使用方法和规范。系统使用 `appendLog` 函数记录不同级别的调试信息，帮助开发人员和用户追踪程序运行状态。

调试信息等级

系统定义了三个调试信息等级：

```
#define ERROR 2      // 错误级别
#define WARNNING 1   // 警告级别
#define INFO 0       // 信息级别
```

显示格式

调试信息的基本格式为：

```
[时间戳] : [信息内容] [可选的数值]
```

- 时间戳格式：yyyy-MM-dd hh:mm:ss
- 不同级别信息使用不同颜色显示：
  - ERROR（红色）：错误信息
  - WARNNING（橙色）：警告信息
  - INFO（黑色）：普通信息

常见调试信息示例

1. 设备操作相关

- 设备枚举

```
appendLog(QString("枚举到 %1 个设备").arg(deviceCount), INFO);
appendLog("未知设备被枚举", WARNNING);
appendLog("没有设备", WARNNING);
```

- 设备打开/关闭

```
appendLog("设备打开成功", INFO);
appendLog("设备关闭成功", INFO);
appendLog("相机对象无效", ERROR);
```

2. 图像采集相关

- 采集控制

```
appendLog("开始抓图成功", INFO);
appendLog("停止抓图成功", INFO);
appendLog("开始抓图失败", ERROR);
appendLog("停止抓图失败", ERROR);
```

- 触发模式

```
appendLog("连续模式已开启", INFO);
appendLog("触发模式已开启", INFO);
appendLog("软件触发已开启", INFO);
appendLog("软件触发已关闭", INFO);
```

3. 参数设置相关

- 曝光参数

```
appendLog(QString("获取曝光时间成功: %1").arg(value), INFO);
appendLog(QString("设置曝光时间成功: %1").arg(value), INFO);
appendLog("获取曝光时间失败", ERROR);
appendLog("设置曝光时间失败", ERROR);
```

- 增益参数

```
appendLog(QString("获取增益成功: %1").arg(value), INFO);
appendLog(QString("设置增益成功: %1").arg(value), INFO);
appendLog("获取增益失败", ERROR);
appendLog("设置增益失败", ERROR);
```

- 帧率参数

```
appendLog(QString("获取帧率成功: %1").arg(value), INFO);
appendLog(QString("设置帧率成功: %1").arg(value), INFO);
appendLog("获取帧率失败", ERROR);
```

```
4. appendLog("设置帧率失败", ERROR);
```

4. 图像处理相关

- 图像保存

```
1. appendLog(QString("保存成功，地址为：%1").arg(path), INFO);
2. appendLog("暂无可用图像，请先开始采集", WARNNING);
3. appendLog("内存不足（编码缓冲）", ERROR);
```

- 角点检测

```
1. appendLog("待检测图像读取成功，角点检测算法执行完毕", INFO);
2. appendLog("角点检测算法执行失败，请调整曝光或者重选待测物体", ERROR);
3. appendLog("未能正确检测到角点", ERROR);
4. appendLog(QString("第%1个角点(x,y)像素坐标为：(%2, %3)").arg(i+1).arg(x).arg(y),
   INFO);
```

5. 坐标变换相关

- 矩阵操作

```
1. appendLog("成功读取变换矩阵，详见终端显示", INFO);
2. appendLog("坐标矩阵变换成功", INFO);
3. appendLog("变换矩阵计算并保存成功", INFO);
4. appendLog(matrixStr, INFO);  // 显示变换矩阵内容
```

- 坐标转换验证

```
1. appendLog(QString("点 %1: 理论世界坐标(%2, %3) -> 变换后世界坐标(%4, %5)").arg(i)
2.           .arg(WorldCoord[i].x(), 0, 'f', 3)
3.           .arg(WorldCoord[i].y(), 0, 'f', 3)
4.           .arg(x_transformed, 0, 'f', 3)
5.           .arg(y_transformed, 0, 'f', 3), INFO, total_error);
```

- 测量结果

```
1. appendLog(QString("物件对角线（mm）：%1").arg(value), INFO);
2. appendLog(QString("物件长度（mm）：%1").arg(value), INFO);
3. appendLog(QString("物件宽度（mm）：%1").arg(value), INFO);
4. appendLog(QString("物件倾角（°）：%1").arg(value), INFO);
```

6. 图像处理增强功能

- 多边形绘制功能

```
appendLog("多边形绘制功能已初始化", INFO);
appendLog(QString("已添加第%1个点，坐标(%2, %3)").arg(pointCount).arg(x).arg(y), INFO);
appendLog(QString("需要至少3个点才能绘制多边形，当前只有%1个点").arg(pointCount), WARNNING);
appendLog(QString("开始绘制多边形，共%1个点").arg(pointCount), INFO);
appendLog("多边形绘制完成，顶点坐标：", INFO);
appendLog(QString("顶点%1: (%2, %3)").arg(i+1).arg(x).arg(y), INFO);
appendLog(QString("多边形区域图像裁剪完成，尺寸: %1x%2 像素").arg(width).arg(height), INFO);
appendLog(QString("背景颜色: RGB(%1, %2, %3)").arg(red).arg(green).arg(blue), INFO);
appendLog("裁剪后的图像已显示", INFO);
appendLog("已清除多边形显示", INFO);
appendLog("请在widgetDisplay_2上显示图片后再进行点击", WARNNING);
appendLog("多边形区域超出图像范围", WARNNING);
appendLog("需要至少3个点才能裁剪图像", WARNNING);
```

- 图像显示状态

```
appendLog("检测后图像显示成功", INFO);
appendLog("原始图像显示成功", INFO);
appendLog("裁剪区域检测后图像显示成功", INFO);
```

7. 检长算法相关

- 标准检测模式

```
appendLog("检测后图像显示成功", INFO);
appendLog("检长算法执行完成", INFO);
appendLog(QString("物件长度（mm）：%1").arg(length), INFO);
appendLog(QString("物件宽度（mm）：%1").arg(width), INFO);
appendLog(QString("物件倾角（°）：%1").arg(angle), INFO);
```

- 裁剪区域检测模式

```
appendLog("检测到多边形区域，将处理裁剪后的图像", INFO);
appendLog("裁剪区域检测后图像显示成功", INFO);
appendLog("裁剪区域检长算法执行完成", INFO);
appendLog(QString("裁剪区域物件长度（mm）：%1").arg(length), INFO);
appendLog(QString("裁剪区域物件宽度（mm）：%1").arg(width), INFO);
appendLog(QString("裁剪区域物件倾角（°）：%1").arg(angle), INFO);
```

8. 系统监控相关

- 清晰度计算

```
appendLog(QString("当前图像清晰度: %1").arg(sharpness, 0, 'f', 2), INFO);
```

- 深度学习功能

```
appendLog("深度学习二分类示例窗口已打开", INFO);
```

9. 文档操作相关

```
appendLog("已尝试打开调试信息手册", INFO);
```

使用建议

1. 对于重要的操作步骤，应该使用INFO级别记录成功信息
2. 当出现可恢复的异常情况时，使用WARNNING级别提示用户
3. 对于导致程序无法继续执行的错误，使用ERROR级别通知用户
4. 涉及具体数值的信息，建议使用QString的arg()函数格式化输出
5. 在开发新功能时，建议多添加调试信息，便于问题定位

注意事项

1. 所有调试信息都会显示在主界面的日志区域
2. 不同级别的信息使用不同颜色区分，便于快速识别
3. 每条信息都会自动添加时间戳
4. 当日志较多时，会自动滚动到最新信息
5. 错误级别的信息通常还会通过QMessageBox弹窗提示用户