LocateDemoCs

使用说明

在法律允许的最大范围内，本文档是“按照现状”提供，可能存在瑕疵或错误。本公司不对本文档提供任何形式的明示或默示保证，包括但不限于适销性、质量满意度、适合特定目的、不侵犯第三方权利等保证；亦不对使用或是分发本文档导致的任何特殊、附带、偶然或间接的损害进行赔偿，包括但不限于商业利润损失、系统故障、数据或文档丢失产生的损失。

前　言

本节内容的目的是确保用户通过本手册能够正确使用产品，以避免操作中的危险或财产损失。在使用此产品之前，请认真阅读产品手册并妥善保存以备日后参考。

概述

本手册适用于 LocateDemoCs 示例 Demo。

符号约定

对于文档中出现的符号，说明如下所示。

符号	说明
	说明类文字，表示对正文的补充和解释。
	注意类文字，表示提醒用户一些重要的操作或者防范潜在的伤害和财产损失危险。
	警告类文字，表示有潜在风险，如果不加避免，有可能造成伤害事故、设备损坏或业务中断。
	危险类文字，表示有高度潜在风险，如果不加避免，有可能造成人员伤亡的重大危险。

目　录

第1章 环境篇

1.1 应用场景

本案例作为 VM 二次开发的定位引导案例，适用场景为单相机拍照，引导机械手抓取产品的情形。

典型场景包括：

- 相机固定安装，拍照时相机固定不动。
- 机械手旋转时旋转中心不在抓取点上，而是绕某个轴的中心旋转。如下图所示，机械手的旋转轴中心线如图中红线所示，和抓取中心并不重合。

图1-1 定位抓取

说明

本案例演示重点为如何使用 VM 模块去实现定位抓取，并未考虑图像透视和畸变带来的抓取精度问题，因此若实际项目中照搬案例中的方法去引导机械手抓取，被抓产品可能越靠近视野边缘抓取误差越大，越靠近视野中心区域误差越小，此为正常现象。

1.2 运行环境

- 操作系统：Win7 及以上 64 位操作系统，不支持 XP 系统和 32 位系统。
- VisionMaster 版本：VM4.2.0 及以上
- Visual Studio 版本：2015 及以上
- 编程环境：C#
- 界面平台：Winform

- .NET Framework：4.6.1 及以上，建议使用 4.6.1
- 加密狗要求：支持标定和定位功能的加密狗

注意

- 运行程序前请先插加密狗。
- 该示例程序对显卡无要求。

第2章 方案篇

示例方案路径：./PlatformSDKSampleCS/LocateDemoCs/LocateDemoCs.sol

2.1 方案思路

本演示案例为应用案例，软件和具体方案是强相关的。可加载设计好的方案，该演示方案为 Demo 文件夹下的 LocateDemo.sol 方案。若导入其他案例提供的方案，则需进行相关配置方可运行出正确结果。

本方案包含一个标定流程和生产流程，流程的整体框架如下图所示。

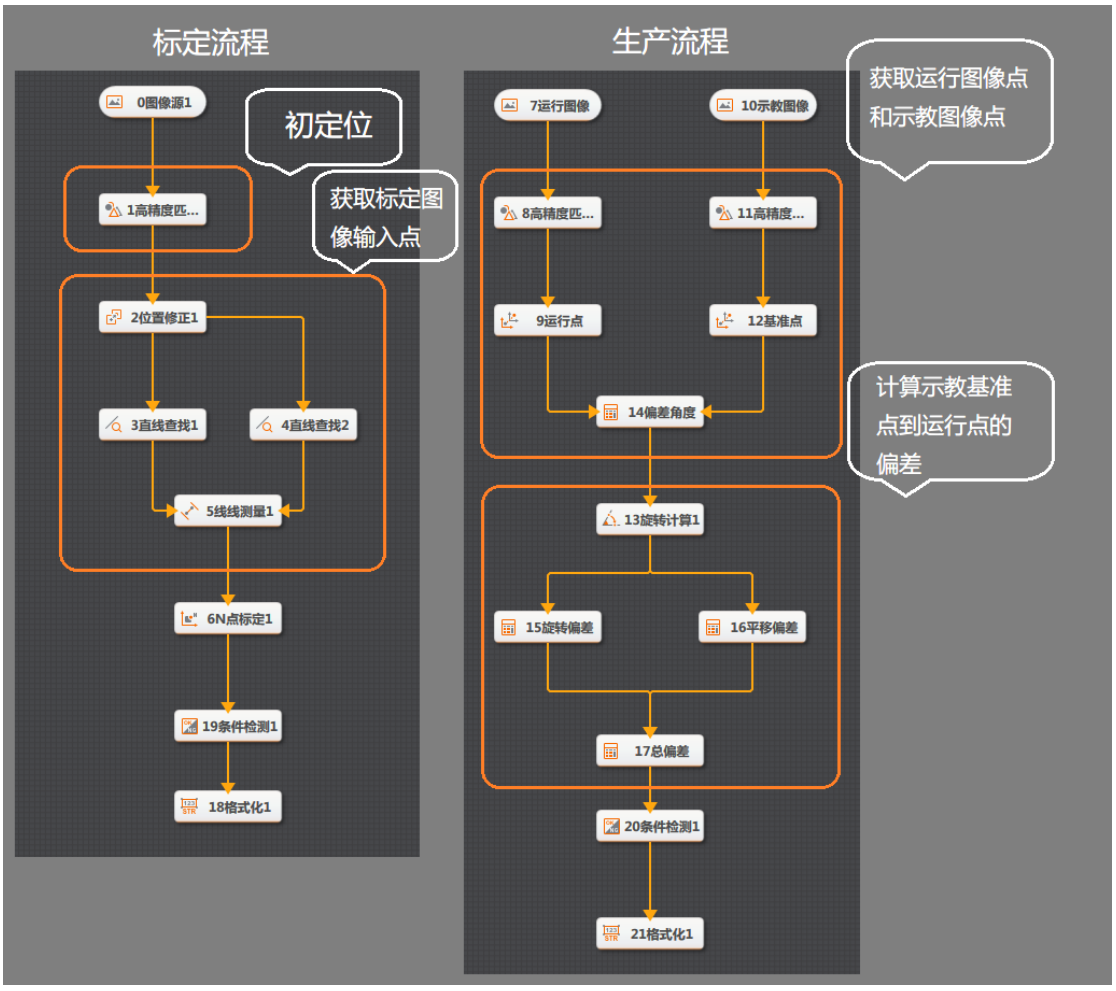

图2-1 方案流程

- 标定流程使用标定板，通过查找标定物中两条边缘直线的交点作为标定图像输入点。

- 生产流程通过旋转计算模块来计算运行图像点和示教图像点的偏差。偏差包含两部分：一部分是由纯平移（不考虑旋转）产生的平移偏差，另一部分是由机械手不共轴旋转产生的偏差。两者偏差加在一起便是示教点与运行点的坐标偏差

2.2 耦合模块

方案的数据结果经过格式化模块汇总，并在流程“输出设置”中订阅，可在二次开发中拿流程输出结果，降低二次开发软件与具体模块的耦合性。

开发获取结果的是流程输出的“out”参数，开发渲染控件 RenderControl 绑定的渲染为图像源图像、DL 字符定位输出矩形框、格式化文本，具体操作如下：

1. 点击流程图标旁边的小扳手图标进入流程配置窗口。

图2-2 配置流程

2. 点击“输出设置”，其中参数名称下面的“out”（区分大小写）即为开发获取的参数名，点击“订阅”，选择格式化作为绑定数据，具体如下图所示。

图2-3 输出设置

3. 点击“显示设置”，点“加号”增加需要的渲染，点“订阅”按钮绑定渲染内容，配置方式如下图所示。

图2-4 显示设置

4. 配置格式化模块内容。

格式化模块必须配置正确。若使用时修改了案例自带的方案，特别是修改格式化模块订阅的数据，则必须参照案例自带的方案，进行格式化模块配置，格式为：**[条件检测结果 INT],[数据项 1],[数据项 2],[数据项 3]**

说明

[ ]内的数据订阅流程中的模块结果输出，各个条目用英文逗号分隔，标定流程的格式化配置如下图所示。

图2-5 格式化配置

说明

生产流程的格式化模块的配置类似，由条件检测结果、数据项组成，各条目用英文逗号分隔，在此不再赘述，请参照生产流程的格式化模块配置。

第3章 开发篇

3.1 运行逻辑

在 Visual Studio 直接调试，或在 bin 目录下双击 LocateDemoCs.exe 运行。

操作逻辑：弹出软件运行界面后，先在方案操作区选择方案、选择成功后点击加载方案；然后在操作区选择需要执行的流程名，即可执行流程，执行方式支持单次执行，连续执行和通讯触发执行三种。

图3-1 Demo 界面

3.2 主要函数

本案例主要函数包含：方案加载、单次执行、连续执行、注册工作结束回调、结果获取、方案保存和渲染界面绑定。

- 方案加载

```
VmSolution.Load(currentSolutionPath);
```

例如：

```
VmSolution.Load("D:\\Test\\Test.sol","abc123");
```

- 单次执行

```
procedure.Run();
```

- 连续运行

```
procedure.ContinuousRunEnable = true;//开启连续执行
procedure.ContinuousRunEnable = false;//停止连续执行
```

- 注册工作结束回调：注册回调是基于使用事件模型，使用户获取流程运行的结果

```
foreach (var vmProcedure in processList)
{
    vmProcedure.OnWorkEndStatusCallBack                                    +=
VmProcedure_OnWorkEndStatusCallBack;
}
```

- 结果获取

```
private   void   VmProcedure_OnWorkEndStatusCallBack(object   sender,
EventArgs e)
{
    try
    {
        VmProcedure procedure = sender as VmProcedure;
        if (procedure != null)
       {
            //此处添加获取结果代码
        }
     }
    catch (Exception ex)
    {
      AppendLog(ex.Message);
    }
}
```

- 方案保存

```
VmSolution.Save();
```

- 渲染界面绑定

```
renderControl.ModuleSource  =  (VmProcedure)VmSolution.Instance[“流程
1”];
```

3.3 控件说明

本案例涉及 3 个控件，分别为 VmRenderControl 控件，VmMainViewConfigControl 控件和 VmGlobalToolControl 控件。

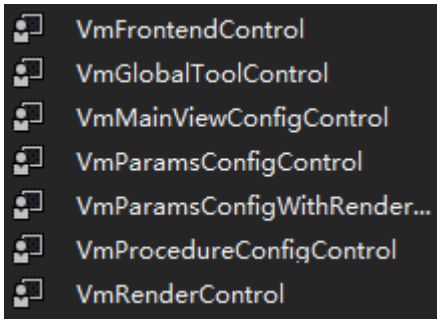

图3-2 相关控件

- VmRenderControl 控件：用于渲染图像数据。

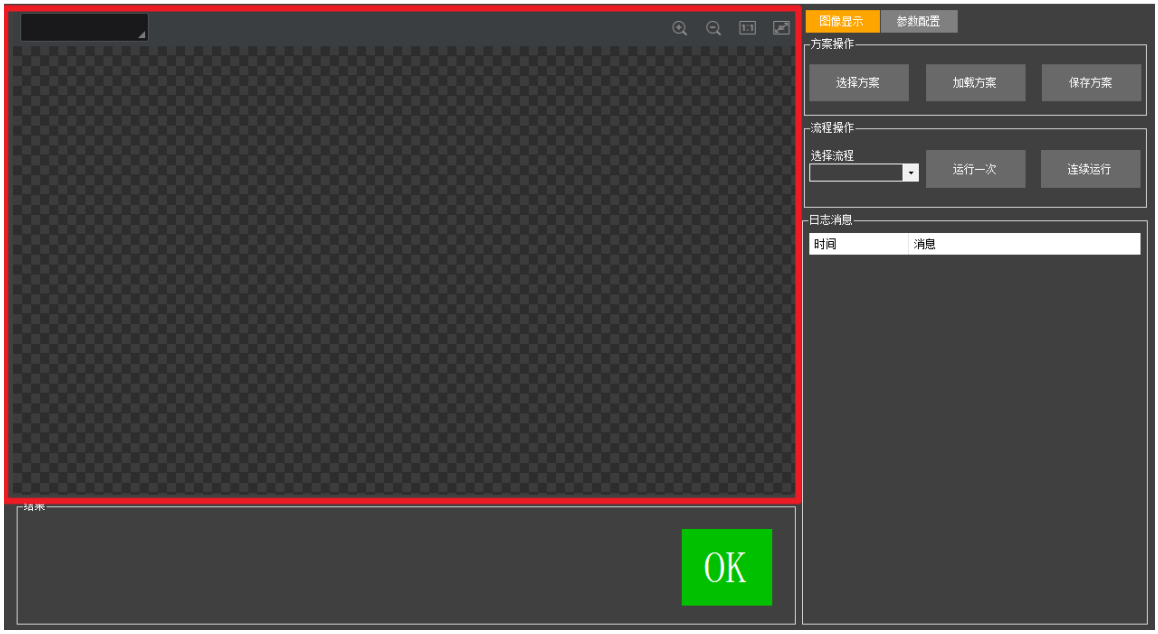

图3-3 VmRenderControl 控件效果

- VmMainViewConfigControl 控件：用于显示流程配置。

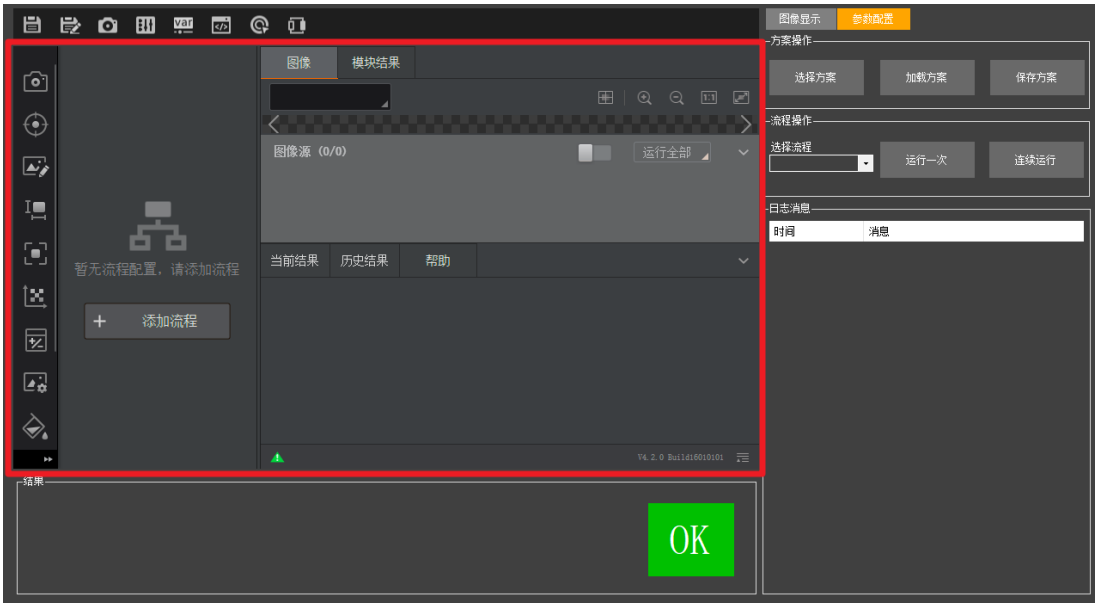

图3-4 VmMainViewConfigControl 控件

- VmGlobalToolControl 控件：用于显示全局配置工具。

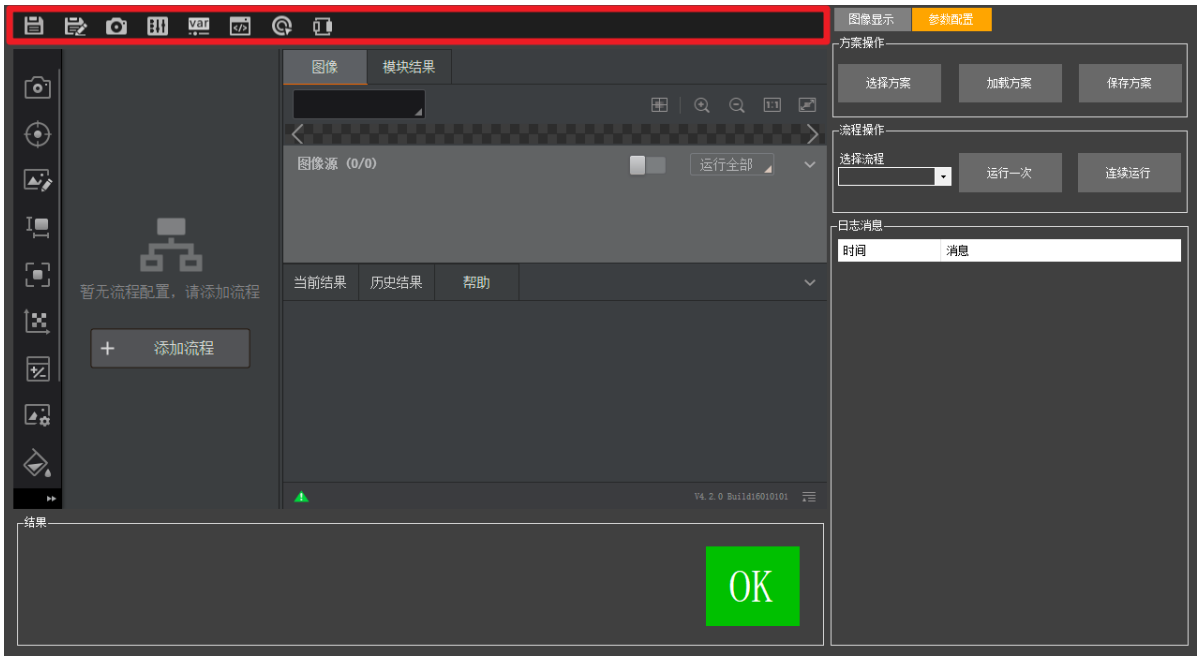

图3-5 VmGlobalToolControl 控件