



OCRDemoCs

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 OCRDemoCs 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	说明类文字，表示对正文的补充和解释。
 注意	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	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	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
 - 1.1 应用场景.....1
 - 1.2 运行环境.....1
- 第 2 章 方案篇.....2
 - 2.1 方案思路.....2
 - 2.2 耦合模块.....3
- 第 3 章 开发篇.....6
 - 3.1 运行逻辑.....6
 - 3.2 主要函数.....7
 - 3.3 控件说明.....8

第1章 环境篇

1.1 应用场景

本案例作为 VM 二次开发的 OCR 识别案例，适用场景为需要使用深度学习进行 OCR 识别的场景。

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：V4.2.0 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
- VM 深度学习补丁包版本：V4.2.0 及以上
- 加密狗要求：同时支持定位和识别功能的加密狗



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/OCRDemoCs/OCRDemo.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 OCRDemo.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。

方案具体流程如下：

1. 使用 VisionTrain1.4.1 训练平台进行深度学习模型训练得到两个训练模型，文件后缀为 bin。分别为 CnnCharDetect.bin（字符定位模型）和 CnnOcrRecog.bin（字符识别模型）。
2. 在 VisionMaster 中拖拽图像源模块加载图像。
3. 使用 DL 字符定位模块进行字符串的定位。
4. 使用 DL 字符识别模块进行识别

方案整体流程如下图所示。

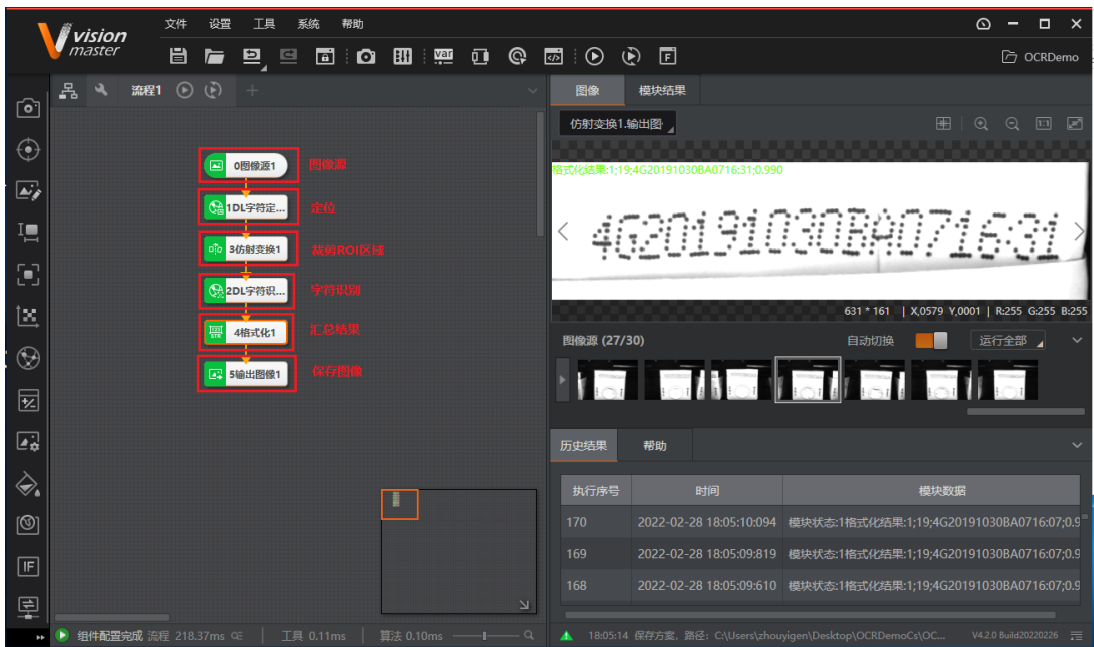


图2-1 方案流程

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，格式如下图所示，主要包含：DL 字符识别模块的模块状态、DL 字符识别模块结果中的最优字符个数、DL 字符识别模块结果中的最佳字符串信息、DL 字符识别模块的最优字符串置信度。



图2-5 格式化配置

第3章 开发篇

3.1 运行逻辑

在 Visual Studio 中直接调试运行，或则在 bin 目录下直接双击 OCRDemoCs.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。

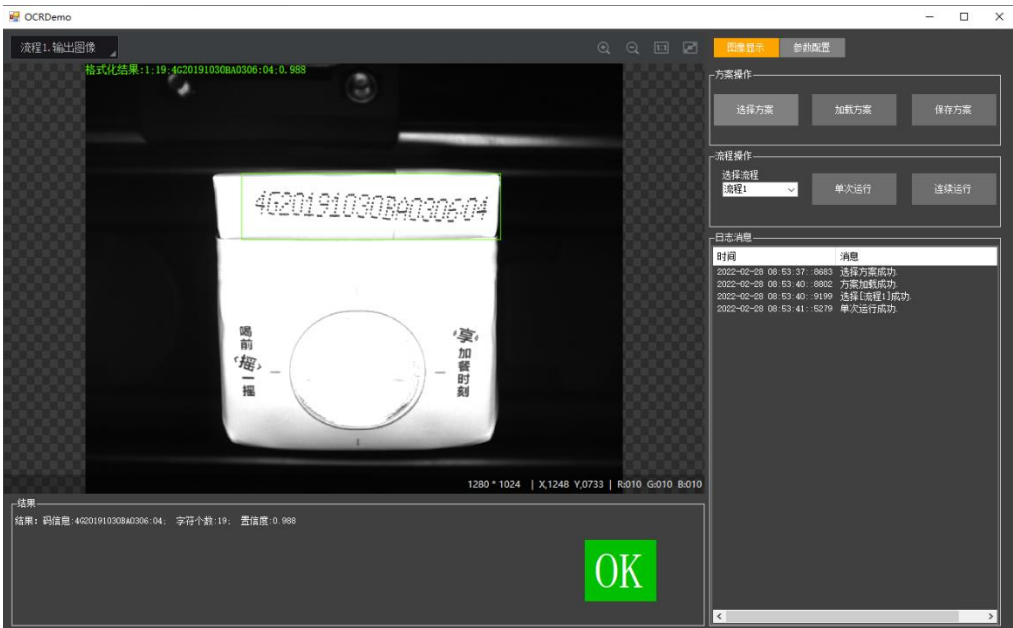


图3-1 Demo 界面

相关操作函数如下图所示，先是选择方案函数和加载方案函数，然后是执行一次函数，最后是流程执行回调函数（用于取算法执行结果），其中流程执行回调需要在构造函数中进行注册。

<pre>/// <summary> 选择方案 1 个引用 private void buttonSelectSolu_Click(object sender, EventArgs e)...</pre>	选择.sol方案文件
<pre>/// <summary> 加载方案 1 个引用 private void buttonLoadSolu_Click(object sender, EventArgs e)...</pre>	加载方案
<pre>/// <summary> 单次运行 1 个引用 private void buttonRunOnce_Click(object sender, EventArgs e)...</pre>	运行相应的流程
<pre>/// <summary> 方案状态回调 1 个引用 private void VmSolution_OnWorkStatusEvent(VM.PlatformSDKCS.ImvsSdkDefine.IMVS_MODULE_WORK_STAUS workStatusInfo)...</pre>	回调中获取流程执行的结果

图3-2 操作函数

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、注册方案状态回调、获取结果、方案保存和渲染界面绑定。

● 方案加载

```

VmSolution.Load(strSolutionPath); //参数为选择的方案路径

VmSolution.Load(strSolutionPath, "***"); //参数为选择的方案路径，方案密码

```

● 单次执行

```

procedure = VmSolution.Instance["流程1"] as VmProcedure; //[]内参数填流名

procedure.Run(); //运行流程

```

● 连续运行

```

procedure.ContinuousRunEnable = true; //开启连续运行

procedure.ContinuousRunEnable = false; //停止连续运行

```

● 方案状态回调

```

VmSolution.OnWorkStatusEvent += VmSolution_OnWorkStatusEvent; //注册方案状态回调

private void
VmSolution_OnWorkStatusEvent(VM.PlatformSDKCS.ImvsSdkDefine.IMVS_MODULE_WORK_STA
US workStatusInfo) //回调函数实现
{
    if (workStatusInfo.nWorkStatus == 0 && workStatusInfo.nProcessID == 10000) //流程执
    行完毕，且为第一个流程
    {
        //获取结果
    }
}

```

● 结果获取

```

string strResult =
procedure.ModuResult.GetOutputString("out").Value.astStringVal[0].strValue; //()
内参数为流程输出的参数名

```

● 方案保存

```

VmSolution.Save();

```

● 渲染界面绑定

```

renderControll1.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。

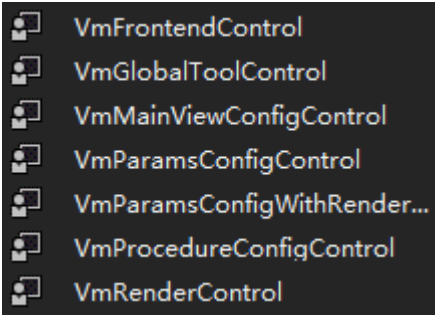


图3-3 相关控件

- VmRenderControl 控件：用于渲染图像数据。

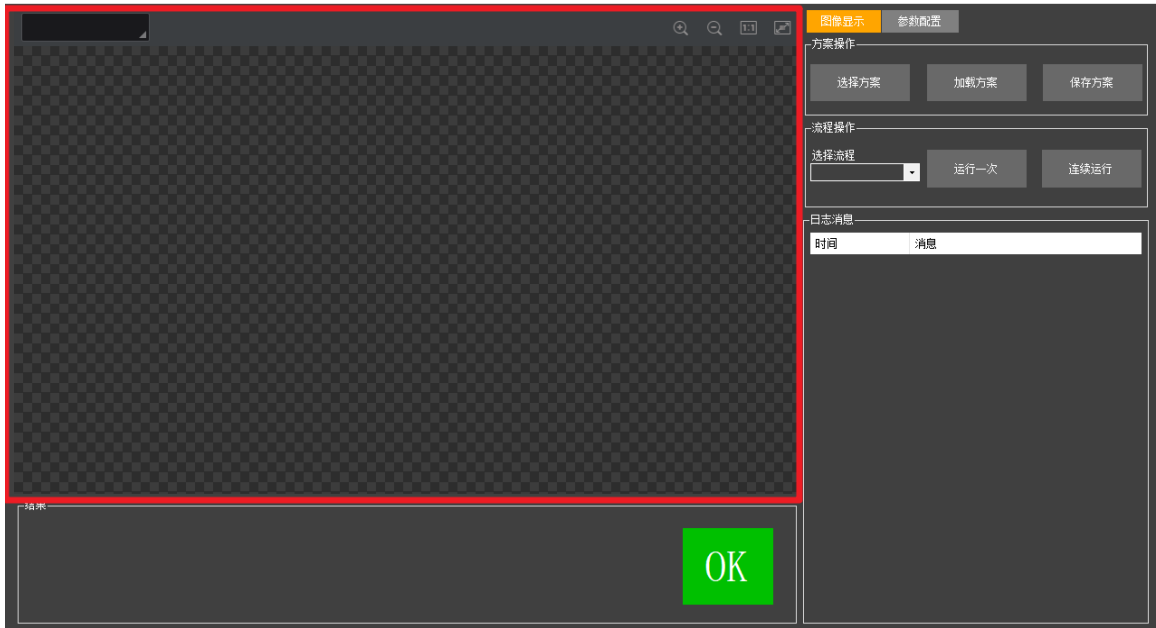


图3-4 VmRenderControl 控件效果

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

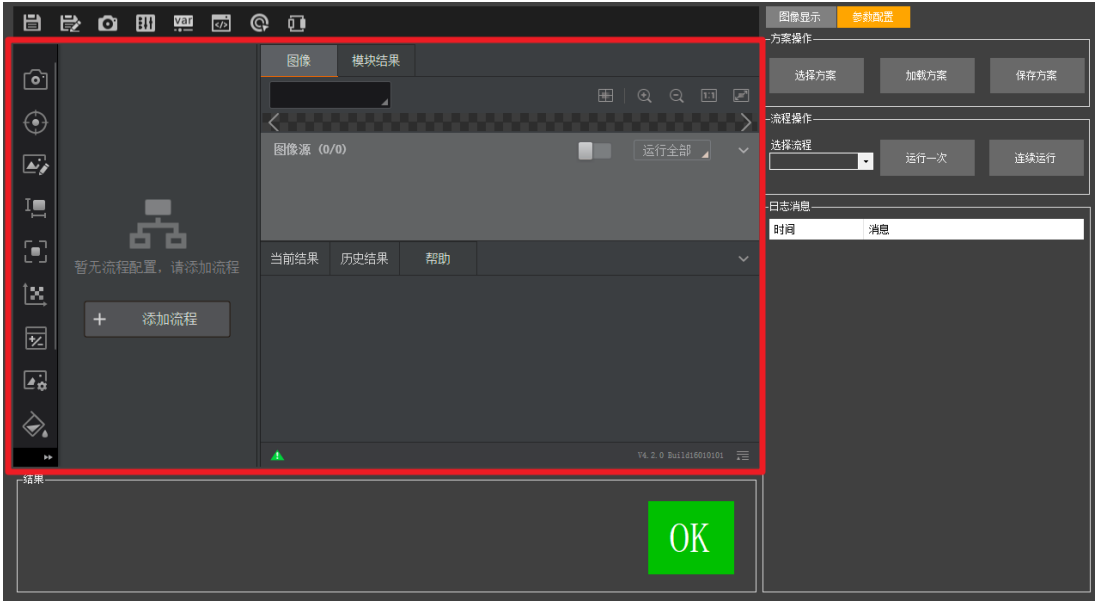


图3-5 VmMainViewConfigControl 控件

- VmGlobalToolControl 控件：用于显示全局配置工具。

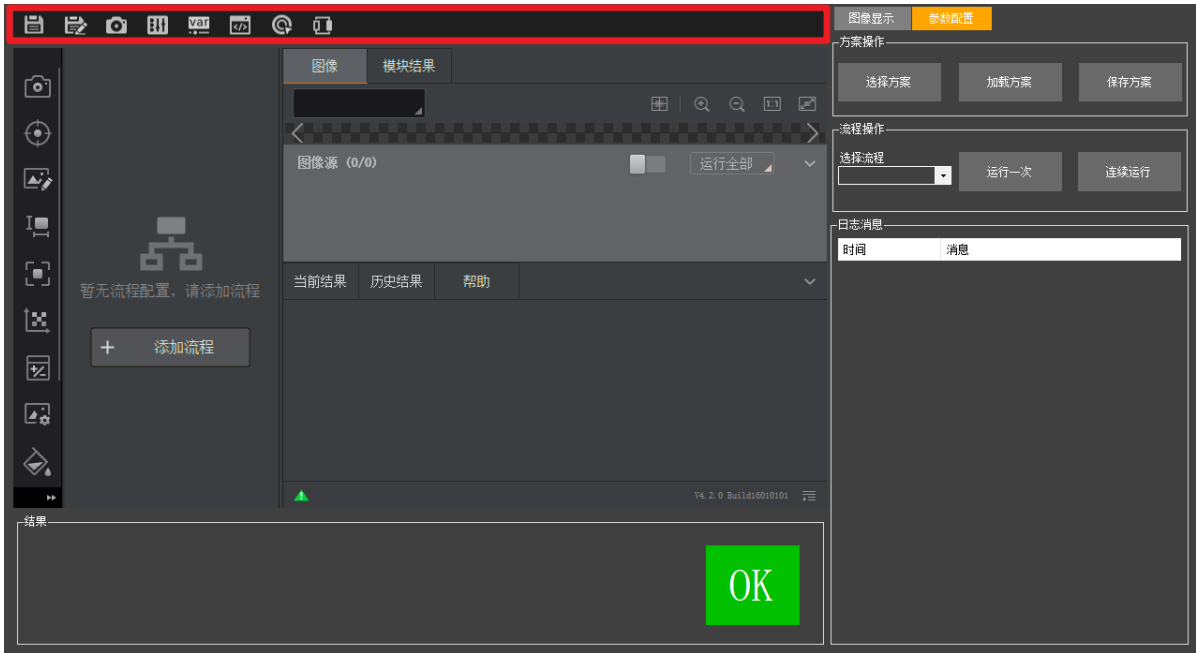


图3-6 VmGlobalToolControl 控件

