

CKVISION SDK 使用说明

Ver. 5.3.0.1

www.ckvision.net



CKVision SDK

1

Bin	
1.1	DLL 库文件8
	CKBase.dll 基础库8
	CKBlob.dll 斑点分析
	CKCalibration.dll 标定与校准8
	CKColor.dll 颜色识别8
	CKContour.dll 轮廓提取8
	CKGDI.dll 图形显示
	CKLocate.dll 形状匹配(识别定位)8
	CKMeasure.dll 测量(点、线、圆)8
	CKReader.dll 条码识别(一维码、二维码)8
1.2	功能实例 exe 执行文件9
	BarcodeDemo.exe 一维码检测9
	BlobToolDemo.exe 斑点分析11
	CalibrationDemo.exe 标定校准12
	CaliperDemo.exe 卡尺测量、间距检测15
	ColorMatchDemo.exe 色彩匹配17
	ColorThresholdDemo.exe 彩色二值化18
	ContourDemo.exe 轮廓提取20
	DataMatrixDemo.exe 二维码读取(DM 码)21
	EdgeToolDemo.exe 边缘点检测
	FitCircleDemo.exe 圆形测量(拟合圆)23
	FitLineDemo.exe 直线测量(拟合直线)24
	HistogramDemo.exe 灰度直方图、亮度检测、自动二值化阈值26
	ImageDemo.exe 图像预处理27

2 / 138



ImageWarpDemo.exe 环形展开裁剪图像28
ImgTransDemo.exe 图形变换(镜像、平移、旋转、缩放、仿射)29
InspectDemo.exe 图像对比缺陷检测33
ModelDemo.exe 模板轮廓匹配定位(老版本)
MultiModelDemo.exe 多轮廓匹配定位(新版本)
NCMatchDemo.exe 灰度匹配定位42
QRCodeDemo.exe 二维码检测(QR 码)44
ReadOcrDemo.exe 字符读取45
SearchDemo.exe 模板轮廓匹配定位(新版本)49
2 Bin_x64
2.1 等同 Bin 文件内功能。52
3 Document 文档
CKVision.chm53
CKVision 简介.pdf53
版本说明.doc53
CKVISION SDK 说明53
4 Include 开发库头文件
CKAcmeTool.h 顶点测量 (CAcmeTool) CKMeasure.dll54
CKBarcode.h 一维码读取 (CReadBarcode) CKReader.dll54
CKBase.h 基础模块 CKBase.dll54
CKBaseDef.h 导出/导入、数据结构定义54
CKBlob.h 斑点分析、图像对比 CKBlob.dll54
CKBlobAnalyzer.h 斑点分析 (CBlobAnalyze)54
CKBlobData.h Blob 数据 (CBlobData)54
CKBlobDef.h Blob 定义54
CKCalibration.h 标定功能 (CCalibration) CKCalibration.dll54



CKCaliper.h 卡尺	、间距测量 (CCaliper) CKMeasure.dll	54
CKCharset.h	字符集 (CCharset) CKReader.dll	54
CKColor.h 颜色	CKColor.dll	54
CKColorIdentify.h	颜色颜色识别 (CColorSamples 、CColorIdentify)	54
CKColorMonitor.h	颜色监测 (CColorMonitor)	54
CKColorSample.h	颜色样本 (CColorSample)	54
CKContour.h	轮廓检测、轮廓缺陷 CKContour.dll	54
CKContourDefect.	h 轮廓缺陷 (CContourDefect)	54
CKContourDetect.	h 轮廓检测 (CContourDetect)	54
CKDataMatrix.h	读取 DataMatrix 二维码(CDataMatrix)CKReader.dll	54
CKDotMatrix.h	圆形矩阵标定板 (CDotMatrix)	54
CKEdgeTool.h	边缘点检测 (CEdgeTool) CKMeasure.dll	54
CKFileStore.h	文件存储结构 (CFileStore) CKBase.dll	54
CKFindBarcode.h	读取一维码 (CFindBarcode) CKReader.dll	54
CKFindModel.h	形状模型搜索 (CFindModel) CKLocate.dll	54
CKFitCircle.h	圆拟合工具 (CFitCircle) CKMeasure.dll	54
CKFitLine.h 线抄	合工具 (CFitLine)	54
CKFrameTrans.h	坐标系变换 (CFrameTrans)CKBase.dll	54
CKGDI.h 图形	显示 CKGDI.dll	55
CKGdiBoxScan.h	旋转矩形框内扫描线 (CGdiBoxScan)	55
CKGdiCircle.h	圆形 (CGdiCircle)	55
CKGdiContour.h	轮廓图形 (CGdiContour)	55
CKGdiEllipse.h	椭圆图形 (CGdiEllipse)	55
CKGdiFigure.h	图形功能(基类) (CGdiFigure)	55
CKGdiFrame.h	坐标系显示 (CGdiFrame)	55
CKGdiHistogram.h	直方图 (CGdiHistogram)	55



CKGdiLine.h	线段图形 (CGdiLine)	55
CKGdiMask.h	掩摸显示 (CGdiMask)	55
CKGdiModel.h	模型轮廓显示 (CGdiModel)	55
CKGdiPoint.h	点、十字显示 (CGdiPoint)	55
CKGdiPolygon.h	多边形图形 (CGdiPolygon)	55
CKGdiProfile.h	投影曲线边缘位置(CGdiProfile)	55
CKGdiRect.h	矩形框 (CGdiRect)	55
CKGdiRing.h	圆环图形 (CGdiRing)	. 55
CKGdiRingScan.h	圆环内扫描线 (CGdiRingScan)	55
CKGdiRotBox.h	旋转矩形 (CGdiRotBox)	55
CKGdiText.h	文本显示 (CGdiText)	55
CKGdiType.h	模板类显示 (CGdiType)	55
CKGdiView.h	图形视图窗口 (CGdiView) CKGDI.dll	55
CKGeoMeas.h	基本几何测量 (CKVISION_API)CKBase.dll	55
CKHasp.h 校验	金锁	55
CKHistogram.h	直方图、分割阈值 (CHistogram)	55
CKHSIThreshold.h	HSI 颜色抽取 (CHSIThreshold) CKColor.dll	55
CKImage.h 图值	象基本功能 (CPrImage) CKBase.dll	. 56
CKImgConve.h	图像转换、高级调整(CKVISION_API)CKBase.dll	56
CKImgFilter.h	图像滤波	56
CKImgMorph.h	图像灰度形态学	56
CKImgOpera.h	图像算术和逻辑	56
CKImgTrans.h	图像变换(镜像、平移、旋转、缩放、等)	56
CKLocate.h 形状	代匹配、识别定位 CKLocate.dll	56
CKMask.h 图值	象掩摸 (CMask) CKBase.dll	56
CKMeasDef.h	测量定义 CKMeasure.dll	56



	CKMeasure.h	测量	L L	CKMeasure.	dll		56
	CKModel.h 천	莫型特征	E点模板	(CModel)	CKLocate.dll		56
	CKModelConto	our.h 模	型轮廓	(CModelCo	ontour)		56
	CKNCMatch.h	灰度	区域匹配	(CNCMat	ch)		56
	CKNCPat.h 力	灰度模板	ά (CNC	CPat)			56
	CKOverlay.h	覆盖	图功能	(COverlay) CKGDI.dll.		56
	CKPatInspect.h	□ 基于	图像对比	缺陷检测(0	PatInspect)	CKBlob.dll	56
	CKPixelStat.h	像素	统计功能	(CPixelSt	at) CKBase.	dll	56
	CKPointVector	.h	坐标点容	器 (CPoi	ntVector) CKI	Measure.dll	56
	CKProfile.h	图像截面	面投影曲线	(CProfile)	CKMeasure	.dll	56
	CKReadDXF.h	读取	ス DXF 文件	生成模板轮风	郭 (CReadDX	F) CKGDI.dll	56
	CKReader.h रे	卖取条码	ふ 、字符	CKRead	der.dll		56
	CKReadOcr.h	字符	F识别	(CReadOcr)	CKReader.d	II	56
	CKReadQRCod	e.h	读取 QR 和	马 (CRead	dQRCode)		56
	CKScanEdge.h	扫描	茚边缘	(CScanEdge)	CKMeasure	e.dll	56
	CKScanSpace.h	1 扫描	间距	(CScanSpace)		56
	CKShapeMatch	ı.h	边缘轮廓	形状匹配(新)	(CShapeM	atch) CKLoca	ate.dll56
	CKShapeMode	l.h	形状模板	(新) (CSh	apeModel) C	KLocate.dll	56
	CKSharpAssess	.h	图像清晰	度评估 (C	SharpAssess)	CKBase.dll	56
5 Install	运行库安装包.	•••••					57
6 Lib 开发	支库 Lib 文件	•••••					57
7 Lib_x64	开发库 64 位	版本文	件				57
8 Sample	s 功能 API 调月	目实例					58
	BarcodeDemo	一维码	检测				66
	BlobToolDemo) 斑点分	♪析				71
	CalibrationDer	no 标定	校准				77



CaliperDemo 卡尺测量、间距检测81
ColorMatchDemo 色彩匹配85
ColorThresholdDemo 彩色二值化87
ContourDemo 轮廓提取89
DataMatrixDemo 二维码读取(DM 码)
EdgeToolDemo 边缘点检测96
FitCircleDemo 圆形测量(拟合圆)98
FitLineDemo 直线测量(拟合直线)
HistogramDemo 灰度直方图、自动二值化阈值
ImageDemo 图像预处理108
ImageWarpDemo 环形展开裁剪图像109
ImgTransDemo 图形变换(镜像、平移、旋转、缩放、仿射)111
InspectDemo 基于图像对比缺陷检测113
ModelDemo 模板轮廓匹配定位(老版本)117
MultiModelDemo 多轮廓匹配定位(新版本)
NCMatchDemo 灰度匹配定位121
QRCodeDemo 二维码检测(QR 码)
ReadOcrDemo 字符读取127
SearchDemo 模板轮廓匹配定位(新版本)
9 附 1. CKVISION API 功能分类



1 Bin

1.1 DLL 库文件

CKBase.dll 基础库 CKBlob.dll 斑点分析 CKCalibration.dll 标定与校准 CKColor.dll 颜色识别 CKContour.dll 轮廓提取 CKGDI.dll 图形显示 CKLocate.dll 形状匹配(识别定位) CKMeasure.dll 测量(点、线、圆) CKReader.dll 条码识别(一维码、二维码)



1.2 功能实例 exe 执行文件

BarcodeDemo.exe 一维码检测

III BarcodeDemo	× 操作 打开图像
5的开发者, 氮结构的Intel® C4-FFM Guess 成程级并行。 图书分类:计算机>程序设计 ISBN 978-7-121-04005-4	1001 检测 参数 读取条码类 UFC A UFC A UFC A EAN 13 CODE_39 CODE_93 CODE_128
2 定价: 59.00元 数据 定位梯度 20 最大条宽度 25 ☑ 水平 □ 垂直 □ 45度 □ 135度	INTERLEAVED_2_5 滤波器半宽 1 ÷ 梯度阈值 20 ÷
9787121040054	消耗时间: 16.393 ms

打开图像:打开一张 8 位 bmp 的灰度图像。

ROI: 检测区域,指定区域内进行检测,可减少处理的时间。

检测: 自动定位识别条码所在的位置并读取出条码的值。

条码类型:选择当前需要读取的条码类型,可以同时选择多个(自动识别类型)。

滤波器半宽:用于增强边缘提取功能,去除图像上的噪音干扰;最小值为1,当边缘模 糊不清晰或有噪音干扰时可以增大滤波半宽值,这样可以使得检测结果更加稳定,但如 果边缘和边缘之间挨得太近(距离小于滤波半宽值)时反而会影响边缘位置的精度甚至 丢失边缘,所有要根据实际情况来设置。



梯度阈值:取值范围 0 到 255,只有梯度值大于该值的边缘点才被检测到,梯度值是度量图像边缘的清晰度或对比度。

II] BarcodeDemo	×
前开发者,	操作 打开图像 ROI 检测
图书分类:计算机>程序设计 IBBH 978-7-121-04005-4 	参数 读取条码类 UFC_A UFC_A EAX-8 EAX-8 EAX-13 CODE_39 CODE_393 CODE_128 INTERLEAVED_2_5
运价: 59.00元	滤波器半宽 1 흦 梯度阈值 20 🌲
数据 定位梯度 20 最大条宽度 25 ☑ 水平 □ 垂直 □ 45度 □ 135度	信息
9787121040054	;月秋山山曰: 7.122 ms

定位梯度:

梯度表示边缘的强度(清晰度),取值范围 0~255,当梯度大于该值的边缘点才会被识别。

最大调宽度:条码的黑条或白条可能出现的最大宽度值。

扫描方向:

水平,仅识别水平方向大约±30度范围内的条码;

垂直, 仅识别垂直方向大约±30 度范围内的条码;

45 度, 仅识别 45° 左右范围的条码;

135 度, 仅识别 135° 左右范围的条码。



BlobToolDemo.exe 斑点分析

检测图像中目标的数量和几何特征(面积、位置、方位、长轴和短轴),目标对象的定 义为二值图像中黑色或白色像素连通区域,该功能要求图像背景均匀并亮度和目标的亮 度区别比较明显。



分割阈值:设置二值图像的分割阈值,当像素灰度值大于等于该值为白色,否则为黑色。

自动计算阈值:软件将根据直方图分布自动计算出分割阈值。

二值图像预览:为了方便调整阈值,以二值化效果显示当前图像。

检测类型:可以设置当前需要检测的目标为黑色像素或白色像素区域。 **连通性:**算法中判定为连通的方法,4连通表示只在上、下、左和右4个方向上相邻才 被判定为互相连通,8连通则在左上、右上、左下和右下方向上也算是互相连通。

限定面积:当目标面积(像素数量)小于该值时会被删除。

主轴椭圆特征: 主轴椭圆为以区域重心为中心拟合的椭圆。



凸包特征:分析计算凸包面积。

最小外接矩形特征:计算连通区域的最小面积外接矩形。

🚴 BlobToolDemo									×
参数设置 其它功能 过速 特征: 面积 特征: 10,00 最小: 10,00 最大: 10000,00 执行 排序 特征: 面积 方式: 降序 执行 加行 道道: 1000,00 九行 方式: 降序 执行 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 10000									
打开图像	索引 0	面积 4219 3162	重心(X) 229.324 57.521	重心(Y) 317.790 415.733	灰度平均 230.980 201.088	孔洞数里 2 0	凸包面积 44788.0 3379.5	Feret中 315.214 57.500	
设置范围	2	2045 830	322.529 24.713 23.924	133.987 255.333 211.569	234.654 207.292 219.551	0	4962.5 1809.5	321.973 24.000 23.500	
检测	5 6 7	694 578	23, 924 53, 595 430, 413	150.280 73.055	219.661 217.756 218.675	1 1 1	1040.5 1070.5 1266.5	23.500 53.000 431.500	~

过滤:可以设置目标每种特征的最小值和最大值,当某个目标的特征不在该范围值只能则会被移除,列表中被勾选的项目表示使用该特征的过滤功能。

特征:按照指定特征将所有目标对象进行排序。

方式:排序方式,可以选择升序或降序。

激活排序:勾选表示使用排序功能。

CalibrationDemo.exe 标定校准

www.ckvision.net



通过拍摄圆点矩阵标定板,自动计算出图像矫正系数,并将倾斜拍摄的图像矫正成平面, 并输出矫正后的图像。



输入图像:链接需要进行校正的图像。

标记颜色:进行标定的圆点的类型为黑色或者白色。

设置区域:设置需要进行校正的图像区域。

最小面积:图像上圆点像素面积。

最大面积:图像上圆点像素面积。

标定方法:有透视与透视和畸变(径向畸变)两种校准类型。

两点之间的距离:圆点与圆点之间的距离,以mm为单位。

标定:获取标定结果。

校正图像:按标定方法校准图像。







CaliperDemo.exe 卡尺测量、间距检测



边缘极性:有亮到暗、暗到亮和任意 3 种模式; 亮到暗表示从亮度高过度到亮度低的边缘; 暗到亮表示从亮度低过度到亮度高的边缘; 任意模式则亮到暗和暗到亮的边缘都检测。

检测位置:有起始、最后、最宽和全部4种模式; **起始**表示只检测最靠近扫描起始位置的1个间距; **最后**表示只检测离扫描起始位置最远的1个间距; **最宽**表示只检测扫描范围内距离最远的1个间距, **全部**表示检测扫描范围所有间距。

梯度阈值:梯度表示边缘的强度(清晰度),取值范围 0~255,只有梯度值大于该值的边缘点才会被检测到。

滤波半宽:用于增强边缘和抑制噪音干扰,最小值为1,当边缘模糊不清晰或有噪音干扰时可以增大滤波半宽值,这样可以使得检测结果更加稳定,但如果边缘和边缘之间挨



得太近(距离小于滤波半宽值)时反而会影响边缘位置的精度甚至丢失边缘,所有要根据实际情况来设置。





ColorMatchDemo.exe 色彩匹配



预先学习颜色样本,识别出当前颜色属于哪一个样本,并返回匹配程度。





ColorThresholdDemo.exe 彩色二值化

				操作
				打开图像
				恢复图像
				保存图像
				执行
		n nin min ni		参数
				RGB ~
设置				信息
设置 lint	0	MaxH	255	信息 消耗时间: 11.121 ms
i设置 flinH flinS	0	MaxH MaxS	255	信息 消耗时间: 11.121 ms

恢复:恢复调节之前的图像。 保存图像:另存一张调节好的图像。 参数: RGB 、HSI 。 RGB 即是代表红、绿、蓝三个通道的颜色。

色调 H(Hue):与光波的波长有关,它表示人的感官对不同颜色的感受,如红色、绿色、 蓝色等,它也可表示一定范围的颜色,如暖色、冷色等。

饱和度 S(Saturation):表示颜色的纯度,纯光谱色是完全饱和的,加入白光会稀释饱和 度。饱和度越大,颜色看起来就会越鲜艳,反之亦然。

强度 I(Intensity): 对应成像亮度和图像灰度,是颜色的明亮程度。







ContourDemo.exe 轮廓提取





梯度阈值:提取边缘轮廓时使用的参数,当边缘对比度较差时需要降低梯度阈值,如果目标边缘清晰,则可以设置比较高,取值范围 0~255。

最短长度: 检测的最短轮廓数据,初始值默认为 20;

最长长度: 检测的最长轮廓数据, 初始值默认为 2000;

高精度: 使用插值功能可以提升定位精度。

滤波器:滤波器可以增强边缘效果,但也会丢失细节,有低、中和高3个选项,默认为中。



DataMatrixDemo.exe 二维码读取(DM 码)

读取 DataMatrix 二维码,可以自动定位二维码,并允许二维码图像旋转任意角度。



搜索数量:最多被允许搜索到的目标数量。

滤波级别:用于增强边缘和抑制噪音干扰,分为低、中、高三个级别。

梯度阈值:梯度表示边缘的强度(清晰度),取值范围 0~255,要求二维码的边缘强度 大于该值才有可能被定位和读取。

缺失长度:允许检测的二维码定位边缺失部分长度,默认值为 20。

二维码颜色:选择要读取的二维码是黑色、白色或者任意,其中任意包含了黑色和白色。 **二维码形状:**选择要读取的二维码是正方形、长方形或者任意。

检测结果:显示读取二维码数据结果。

预设尺寸:预先设置所需检测二维码的尺寸大小。

固定单元数:固定设置视野中所需检测二维码的行、列数量。



EdgeToolDemo.exe 边缘点检测



初始化 ROI:

初始 ROI 默认位置与大小, ROI 终点位置(在 ROI 边上中间有圆形旋转调节点的边)。

参数

- **边缘极性:** 有亮到暗、暗到亮和任意 3 种模式,从 ROI 起点边开始往结束边的方向上; **亮到暗**表示从亮度高过度到亮度低的边缘; **暗到亮**表示从亮度低过度到亮度高的边缘; **任意**模式则亮到暗和暗到亮的边缘都检测。
- 检测位置:有起始、最后、最宽和全部4种模式; 起始表示只检测最靠近扫描起始位置的1个间距; 最后表示只检测离扫描起始位置最远的1个间距; 最宽表示只检测扫描范围内距离最远的1个间距, 全部表示检测扫描范围所有间距。



梯度阈值:取值范围 0 到 255,只有梯度值大于该值的边缘点才被检测到,梯度值是度 量图像边缘的清晰度或对比度。

滤波半宽:用于增强边缘和抑制噪音干扰,最小值为1,当边缘模糊不清晰或有噪音干扰时可以增大滤波半宽值,这样可以使得检测结果更加稳定,但如果边缘和边缘之间挨得太近(距离小于滤波半宽值)时反而会影响边缘位置的精度甚至丢失边缘,所有要根据实际情况来设置。

FitCircleDemo.exe 圆形测量(拟合圆)

同时检测出多个边缘点并拟合成圆形,可用于测量圆的尺寸和定位。



重置 ROI:

初始 ROI 默认位置与大小。

参数

边缘极性: 有亮到暗、暗到亮和任意 3 种模式;

亮到暗表示从亮度高过度到亮度低的边缘;

www.ckvision.net



暗到亮表示从亮度低过度到亮度高的边缘; 任意模式则亮到暗和暗到亮的边缘都检测。

检测位置:有起始、最后、最宽和全部4种模式; **起始**表示只检测最靠近扫描起始位置的1个间距; **最后**表示只检测离扫描起始位置最远的1个间距; **最宽**表示只检测扫描范围内距离最远的1个间距, **全部**表示检测扫描范围所有间距。

梯度阈值:取值范围 0 到 255,只有梯度值大于该值的边缘点才被检测到,梯度值是度 量图像边缘的清晰度或对比度。

滤波半宽:用于增强边缘和抑制噪音干扰,最小值为1,当边缘模糊不清晰或有噪音干扰时可以增大滤波半宽值,这样可以使得检测结果更加稳定,但如果边缘和边缘之间挨得太近(距离小于滤波半宽值)时反而会影响边缘位置的精度甚至丢失边缘,所有要根据实际情况来设置。

扫描数量: 设置扫描边缘点 ROI (图中蓝色图形区域)的数量

扫描宽度:扫描边缘点 ROI(图中蓝色图形区域)的宽度,增大宽度可以在一定范围内 求平均,使得计算结果根据稳定。

容忍误差:容忍拟合的最小误差值, 拟合出来出来的圆形误差会小于该值, 误差大的点将会被排除。

FitLineDemo.exe 直线测量(拟合直线)





初始化 ROI:

初始 ROI 默认位置与大小。

参数

边缘极性:有亮到暗、暗到亮和任意 3 种模式; 亮到暗表示从亮度高过度到亮度低的边缘; 暗到亮表示从亮度低过度到亮度高的边缘; 任意模式则亮到暗和暗到亮的边缘都检测。

检测位置: 有起始、最后、最宽和全部4种模式;

起始表示只检测最靠近扫描起始位置的1个间距; 最后表示只检测离扫描起始位置最远的1个间距; 最宽表示只检测扫描范围内距离最远的1个间距, 全部表示检测扫描范围所有间距。

梯度阈值:取值范围 0 到 255,只有梯度值大于该值的边缘点才被检测到,梯度值是度 量图像边缘的清晰度或对比度。

滤波半宽:用于增强边缘和抑制噪音干扰,最小值为1,当边缘模糊不清晰或有噪音干扰时可以增大滤波半宽值,这样可以使得检测结果更加稳定,但如果边缘和边缘之间挨

25 / 138



得太近(距离小于滤波半宽值)时反而会影响边缘位置的精度甚至丢失边缘,所有要根据实际情况来设置。

扫描数量: 设置扫描边缘点 ROI (图中蓝色图形区域)的数量

扫描宽度:扫描边缘点 ROI(图中蓝色图形区域)的宽度,增大宽度可以在一定范围内 求平均,使得计算结果根据稳定。

容忍误差:容忍拟合的最小误差值, 拟合出来出来的圆形误差会小于该值, 误差大的点将会被排除。

HistogramDemo.exe 灰度直方图、亮度检测、自动二值化阈值

检测图像中指定区域的灰度平均值(亮度)和标准差。

www.ckvision.net





灰度信息:显示检测后的数据,最大值、最小值、平均值和标准差;标准差是一组数据 平均值分散程度的一种度量,一个较大的标准差,代表大部分数值和其平均值之间差异 较大;一个较小的标准差,代表这些数值较接近平均值。

ImageDemo.exe 图像预处理



深圳市创科自动化控制技术有限公司



对图像做一些预处理(平滑、锐化、腐蚀和膨胀等功能)后并输出处理后的图像。



ImageWarpDemo.exe 环形展开裁剪图像





按 ROI 区域裁剪展开。 裁剪展开之后效果如下图:



ImgTransDemo.exe 图形变换(镜像、平移、旋转、缩放、仿射)















旋转:

ImgTransDemo		>
	E76508115 DIGITAL CAMERO D3224 MADE IN CHINA D3224 OUYMPUS IMAGING CORP. MODEL NO. 4 840 E76508114 DIGITAL CAMERA 0327 MADE IN CHINA 0327 MADE IN CHINA 0327 E76508113 DIGITAL CAMERA 0327	操作 打开图像 保存图像 执行 显示图像 輸出图像 功能选项 ○ 貨像 ○ 平移 ③縮放
· 金典 14 1五		 ● 旋转 ○ 仿射
視隊还 坝 ☑ 水平镜像 □ 垂直镜像 □ 旋转	平移X: 0 比例X: 1 角度: 10 平移X: 0 比例X: 1 角度: 10 平移Y: 0 比例Y: 1 ☑ 双线性插值	信息 消耗时间: 24.121 ms



InspectDemo.exe 图像对比缺陷检测

操作 打开图像 设置学习区域 学习模板 编辑模板 执行 掩摸图操作 Half 1 🔮 生成 统计 检查参数 修正模式 Null v **SAE 5W-30** 差异类型 全部 🗸 ÎÎ ☑ 使用掩摸图像 Blob分析 阈值 60 面积 10 来源图像: 模板图像: **摍膜图像:** 差异图像: 信息 消耗时间: 11.455 ms Static

使用模板图像和当前图像进行对比,将差异部分检测出来,可用于一般的缺陷检测。 InspectDemo ×

设置学习区域:设置学习区域的 ROI。 **学习模板:**学习定位模板与图像对比模板。 **编辑模板:**编辑定位模板。



生成:按照(Half)半宽大小重新把学习的模板生成一个掩模图(屏蔽图)。 统计:把检测测试对比结果图添加到掩模图。

www.ckvision.net



修正模式:

亮度修正, 对检测结果由于对画面的明亮程度差异造成的影响可消除; 对比度修正, 对比度是图像中明暗区域最亮的白和最暗的黑之间不同亮度层级 的测量,差异范围越大代表对比越大,差异范围越小代表对比越小; 无修正, 不进行修正, 直接进行检测。

差异类型:选择哪一种不同的情况进行检测,亮度或对比度或者是全部(亮度和对比度 都检测)。

Blob 分析,分析对比之后的差异图像。

阈值: 设置二值图像的分割阈值,当像素灰度值大于等于该值为白色,否则为黑色。 **面积:** 限制最小的面积,只有缺陷大于限制的面积才会被检测出来。





InspectDemo	×
	操作 打开图像 设置学习区域 学习模板 编辑模板 执行 掩摸图操作 Half 1 ◆ 生成 统计 检查参数 修正模式 Null ◆ 差异类型 全部 ◆ 过使用掩摸图像 Blob分析 阈值 60 面积 10
	信息 消耗时间: 13.456 ms Static

TINSpectDemo				×
	SAE L	5W-30		操作
THE AND A			左杆凸版:	高思 消耗时间: 11.668 ms Static



ModelDemo.exe 模板轮廓匹配定位(老版本)

使用图像的边缘轮廓特征作为模板,在图像中搜索形状上相似的目标,可以设置角度和 比例范围,可用于定位、计数和判断有无等



设置学习范围:将会在画面中出现一个青蓝色的 ROI 矩形框,点击选取矩形框,再将该矩形移动到需要作为标准模板的图像区域并调整大小(如上图)。 学习模板:按学习 ROI 在图像上学模板轮廓。 编辑模板:学模板之后可查看编辑模板。



www.ckvision.net
深圳市创科自动化控制技术有限公司

精细级别:定义边缘轮廓的细腻程度,有 Fine、Noraml 和 Coarse3 个选项, Fine 模式精度最高,但边缘太模糊时将无法检测到边缘, Noraml 和 Coarse3 模式则会通 过压缩方式来提取模糊的边缘,同时会丢失细节部分,并会影响定位进精度。

滤波器:用于增强边缘提取功能,跟精细级别类似,但不会影响定位精度。

梯度阈值:取值范围0到255,只有梯度值大于该值的边缘点才被检测到,梯度值 是度量图像边缘的清晰度或对比度。

最短边缘:用于过滤长度小于该值的边缘轮廓。

重新学习:当修改参数后需要点击"重新学习"按钮来获得新的边缘轮廓。

编辑选项:手动编辑模板功能,"指示"可修改当前十字点位置,点击"擦除"或 "恢复"按钮后可以使用鼠标在画面中对着边缘轮廓进行擦除或者恢复,上图中蓝 色轮廓部分为被擦除,绿色部分为正在使用的边缘轮廓。

画笔大小:设置擦除或者恢复画笔的尺寸大小。

标记点: 画面中红色十字标为模板的标记点,在"指示"模式下可以使用鼠标点击 选取并拖动调整位置。

装载模板:从文件中加载模板数据。

保存模板:将模板数据保存到文件中。

设置自由度,

角度范围:匹配目标相对于模板可能存在的最小角度,值范围-180到180。

比例范围: 匹配目标相对于模板可能存在的最小比例,值范围80到120(原始比例值100)。

角度范围	-30	到	30	
比例范围	100	到	100	

设置搜索范围:在图像上画出一个红色的 ROI 矩形框,该矩形框为搜索目标是的搜索范围, 如果是全图搜索可以不用设置。





搜索数量:最多被允许搜索到的目标数量。

最小分数:分数表示目标和模板的相似程度,分数越高越相似,最大值100表示完全匹配,目标分数必须大于该值才会被搜索到,该参数值将会影响搜索速度。

梯度阈值:提取边缘轮廓时使用的参数,当边缘对比度较差时需要降低梯度阈值,如果目标 边缘清晰,则可以设置比较高,取值范围 0[~]255,一般设为 40 左右,该参数值将会影响搜 索速度。

匹配极性:可以设置正常和反转,正常表示目标和模板极性相同,反转则表示相反。

压缩级别:在搜索过程中对图像进行压缩处理可以提升搜索速度,但也会降低识别率(影响 程度跟模板和目标图像背景复杂度有关),一般采用"自动"设置。



分数:匹配目标与模板的相似度。 **位置:**匹配目标相对于当前图像的坐标位置。 **角度:**匹配目标相对于模板的旋转角度。 **比例:**匹配目标相对于模板的缩放比例。



MultiModelDemo.exe 多轮廓匹配定位(新版本)

以边缘轮廓特征作为模板,可学习多个目标作为模板,在图像中搜索形状相似的目标。



学习模板: 以图像上的 ROI 所在位置学习模板轮廓。 编辑模板:点击"编辑"按钮将会弹出编辑模板对话框,在编辑模板对话框下可以对模 板进行编辑。







特征数量:模板特征数量占学习区域特征数量的百分比。

轮廓长度:轮廓长度参数用于过滤,长度小于该值的轮廓将会被删除。

减少特征点:是否减少模版特征点数量。

金字塔级别:可预览当前图像模板层数,点击获取边缘轮廓按钮后会自动获取金字塔级别数值。

掩模编辑:点击"屏蔽"按钮可设置掩模图像屏蔽区域,设置完屏蔽区域后,需要点击 "获取边缘轮廓"才能成功屏蔽所选区域,如上图中红色区域部分为被屏蔽,绿色部分 为正在使用的边缘轮廓,点击"恢复"按钮可恢复图像中被屏蔽的区域。

画笔大小:设置擦除或者恢复画笔的尺寸大小。

学习:重新学习轮廓模板。



Search	Demo								
					-	1.000	操作		W. J. J. 1
					11			打开图像	Model_2
								清除模板	
								学习模板	编辑模板
					1	1		搜索区域	装载模板
					V			搜索	保存模板
					1		参数讨	置	
						. 1966	投索	微量 1 、	最小分数 70 最大角度 180
					2	6	最小	比例 100	最大比例 100
					1		重叠	距离 20	目标极性正常 ~
							搜索	速度	5
					1		定位	精度	5
	索引	分数	位置(X)	位置(Y)	角度	比例			
	0	99.261	442.005	195.996	0.040	1.000			
							Stati	c	
							Stati	c	

最小分数:分数表示目标和模板的相似程度,分数越高越相似,最大值100表示完全匹配, 目标分数必须大于该值才会被搜索到,该参数值将会影响搜索速度。

角度:可以设置被搜索目标可能存在的角度范围,角度为目标相对于模板的角度,取值范围-180[~]180。

比例:可以设置被搜索目标可能存在的比例范围,比例为目标相对于模板的比例,取值范围 80^{~120}。

重叠距离: 匹配多个目标时之间的最小距离不能小于重叠距离, 小于则忽略。

匹配极性:可以设置正常和反转,正常表示目标和模板极性相同,反转则表示相反。

搜索速度: 总共有 10 个等级,等级为 0[~]9,默认为 5,设置的搜素速度级别越高,识别度会 有所下降。

定位精度: 总共有 10 个等级,等级为 0[~]9,默认为 5,设置的定位精度级别越高,搜索速度 会有所下降。

搜索数量:最多被允许搜索到的目标数量。



🛃 Search	Demo							×
					я		操作	Model 1 Model 2
in .				1		,	清泳模板 学习模板 #安区##	编辑模板
					V		授素区域	保存模板
La Value					1	2	搜索数量 1 最小角度 ⁻¹⁸⁰ 最小比例 100	最小分数 70 最大角度 180 最大比例 100
and the second sec					7	5	重叠距离 20 搜索速度	
	충리	公志	(京署(王)	位要(双)	<u> </u>	比例	定位精度	5
1	1	99.391	447. 496	358.015	0.009	1.000	Static	
							 Static Static 」消耗时间: 4.542 ms	

NCMatchDemo.exe 灰度匹配定位

在图像指定区域中搜索跟模板图像相似的目标,使用灰度归一化互相关匹配方法,目标 图像和模板图像允之间许存在亮度和对比度变化,可用于定位、计数和判断有无等。





学习:点击"学习"按钮可进入学习模式,在学习模式下将画面中的 ROI 拖动到需要作为模板的图像区域,然后点击"确认"按钮完成学习。

设置标记点	>
A CONTRACTOR OF	X: <u>52.20</u>
100	Y: 49.06
+	
1000	确认
	取消

模板:显示模板图像数据和模板的标记点(红色十字标),可以使用鼠标点击选取标记 点,然后再拖动可调整标记点位置。 **装载模板:**从文件中加载模板数据。

保存模板:将模板数据保存到文件中。

搜索数量:最多被允许搜索到的目标数量。

最小分数:分数表示目标和模板的相似程度,分数越高越相似,最大值 100 表示完全匹配,目标分数必须大于该值才会被搜索到,该参数值将会影响搜索速度。

亚像素插值: 使用插值功能可以提升定位精度。





QRCodeDemo.exe 二维码检测(QR 码)

读取 QR 码,可以自动定位 QR 码,并允许 QR 图像旋转任意角度。

船 QRCod	leDemo								×
1			0 (198 86, 97	32)			打开图像	执行核	到
ý					1 (468. 79, 11	2.81)	分割设置 阈值		150
		3 0	1	3	: (449.00,372		参数设置 搜索二维码数量 检测二维码颜色 定位核最小面积	1 黑色 100	
编号 0	位置(X) 323.927	位置(Y) 237.969	角度 356.715	文本 qe3Ufh		5	定位核最大面积 消耗时间: 2.586	5000 ms	
						[确定		取消



分割设置

阈值:设置二值图像的分割阈值,当像素灰度值大于等于该值为白色,否则为黑色。
自动计算阈值:软件将根据直方图分布自动计算出分割阈值。
二值图像预览:为了方便调整阈值,以二值化效果显示当前图像。
参数设置
搜索二维码数量:最多被允许搜索到的二维码数量。
检测二维码颜色:选择要读取的二维码是黑色、白色。
定位核最小面积:定位标记最小面积。
定位核最大面积:定位标记最大面积。

ReadOcrDemo.exe 字符读取

读取标签上的字符文本,需要事先将标准字符录入字符集合中。

ReadOcrDemo	0					
		E765	08115	DIGIT	DC3.7V AL CAM	操作 打开图像 学习 字符集
	and a			1.4.5		装载字符集
		15 August	and the second	The Test		保存字符集
	OLY	MPUS	IMAGING	CORP.	ODEL N	阈值设置
	-		*3		DC3.7V:	执行
		E76	☆ろ 輸入字符串: E76508 确认	115	IN CHI	分割参数 字符 白底黒字 〜 限制面积 ¹⁰
				Same in	The seal	最小宽度 10
	OLY	MPUS	IMAGING	CORP.	DC3.7V	最大宽度 30 最小高度 20
		E76!	508113	DIGI	FAL CAN	
						□ 昌井町装字付 水平间隔 5
·符 分数 0	宽度 23	高度 35	分数阈值	60		垂直间隔 5
U 0 0	20 21 20 21 21 21	36 35 36 36 36	■ 返回串 ▼	\$35553555		信息

学习:先调整 ROI 后点"执行",然后在点"学习",出现输入学习字符框,填入对应 字符,确认完成字符学习。

字符集:学习字符后的字符集合。



字符	宽度	高度	and the second second
	26	42	
	26	42	and the second s
6	26	42	
5	26	42	and the second second
0	26	42	
8	26	42	
1	26	42	
1	26	42	
5	26	42	
			删除
			关闭

	serbenno		1911-17-1			1 48/5
		P	1111 0.0		DC3.7V	打开图像
	il s		E765	508115	MADE IN CHI	学习
		-				字符集
						装载字符集
			Carlo ant	and an area		保存字符集
		OL	MPUS	IMAGING (ORP. MODEL	阈值设置
		_			DC3.7V DIGITAL CAN	执行
			E76	508114	MADE IN CHI	分割参数
		the second	TANK TO			字符 白底黑字 ~
						限制面积 10
						最小宽度 10
		OU	MPUS	MAGING	CORP. MODEL	最大宽度 30
		-			DC3.7V	最小高度 20
		100			DIGITAL CAN	最大高度 45
			E76	508113	MADE IN CH	□ 合并断裂字符
	/\#6	***	古麻	, 识别参数和结	果	水平间隔 5
J	100	23	تھا 35	分数阈值	30	垂直间隔 5
	100 100 100	20 21 20	36 35 36	返回串 [信息
	100	21	36 36	~		

字符集:选择当前使用的字符集工具,点击右边的"..."按钮可以弹出字符集属性对话框。

阈值设置: 设置字符和背景之间亮度的分界线, 用于分割字符区域。



戓值	☑自动	169
	确认一	取消

自动阈值: 自动计算分割阈值功能。

字符极性:设置字符是黑色还是白色。

限制面积:设置噪声点面积,面积小于该值的区域将被过滤。

字符宽度:设置字符的最小宽度和最大宽度。

字符高度:设置字符的最小高度和最大宽度。

合并断裂字符:如果某个字符断裂成多个部分(点阵字符),可以使用该功能合并。

水平间隔:设置断裂字符水平方向间隔,水平间隔小于该值不能进行合并。

垂直间隔:设置断裂字符垂直方向间隔,垂直间隔小于该值不能进行合并。

分数阈值:设置识别字符的最小分数,小于该分数的字符将不能识别。

ReadOcrDemo			×
	E76508115	DC3.7V DIGITAL CAM MADE IN CHI	操作 打开图像 学习
T. B.	and the second s		字符集
- 575 (7.8%)			装载字符集
		CORP. MODEL N DC3.7V:	1 保存字付集 阈值设置 执行
	E76508114	MADE IN CHI	分割参数
	<u>よ。</u> 学习 輸入字符串: 4	×	字符 白底黒字 ~ 限制面积 10 最小宽度 10
	OLYMPUS Mil	□ 取消 DEL M C3.7V	最大宽度 30 最小高度 20
	E76508113	DIGITAL CAN MADE IN CHI	最大高度 45
■ 字符 分数 ? 0	党度 高度 识别参数和约 23 36 分数阈值	60	水平间隔 5 垂直间隔 5
	返回串	?	信息





🛃 ReadO	OcrDemo										×
		DE	E7(650	81	15	C N	DC3.7 DIGITAL CA	V: M HII	操作 打 	 开图像 学习 字符集
100										装	賊字符集
					322	No. C				保ィ	存字符集
100		OL	MPL	JS IN	IAG	ING	COR	P. MODE	LN	阈	值设置
		P-			84		[DC3.	7V:		执行
			E/	65U	81	14	P	MADE IN C	HI	分割参数	
		-	TARCE	and and	1		200	ALC: LANGE		字符	白底黑字 ~
100										限制面积	10
170										最小宽度	10
375		OU	MP	US IN	AAG	ING	COF	P. MODE	LI	最大宽度	30
903		-						DC3.	7V	最小高度	20
14		100			-	4.0	1	DIGITAL C	AN	最大高度	45
			E7	650	81	13	1	MADE IN C	HI		并断裂字符
ي بي ب	/\#6	**#	古麻		•	识别参数和	结果			水平间隔	5
-7-15 E	93	23	37			分粉调值	60			垂直间隔	5
7 6 5 0	97 95 98 96	21 21 21 21	37 37 37 37			返回串	E76508114			信息	
8	96	22	36		•						

返回串:显示识别结果字符串文本。



SearchDemo.exe 模板轮廓匹配定位(新版本)

以边缘轮廓特征作为模板,在图像中搜索形状相似的目标。

🚱 SearchDemo	×
	操作
	打开图像
	学习模板编辑模板
and the second	装载模板 保存模板
	搜索区域 搜索
	参数设置
	搜索数量 1 最小分数 70
	最小角度 -180 最大角度 180 I80
	最小比例 100 最大比例 100
	重叠距离 20 目标极性 正常 🗸
	投索速度 5
	定位精度 6
ID 分数 位置(X) 位置(X) 角度 比例	
	Static Static
	Static

学习模板: 以图像上的 ROI 所在位置学习模板轮廓。

编辑模板:点击"编辑"按钮将会弹出编辑模板对话框,在编辑模板对话框下可以对模 板进行编辑。



模板显示	×
	特征数量 5 轮廓长度 20
	減少特征点 否 ◇ 初始金字塔级别 3 ◇ 目标金字塔级别 0 ◇
*	 ● 指示 ○ 擦除
	 恢复 画笔 5 学习 确认
	取消

特征数量:模板特征数量占学习区域特征数量的百分比。

轮廓长度:轮廓长度参数用于过滤,长度小于该值的轮廓将会被删除。

减少特征点:是否减少模版特征点数量。

金字塔级别:可预览当前图像模板层数,点击获取边缘轮廓按钮后会自动获取金字塔级别数值。

掩模编辑:点击"屏蔽"按钮可设置掩模图像屏蔽区域,设置完屏蔽区域后,需要点击 "获取边缘轮廓"才能成功屏蔽所选区域,如上图中红色区域部分为被屏蔽,绿色部分 为正在使用的边缘轮廓,点击"恢复"按钮可恢复图像中被屏蔽的区域。

画笔大小:设置擦除或者恢复画笔的尺寸大小。

学习:重新学习轮廓模板。



SearchDe	emo									
						操作		71.		
				1.1	1 20000		「开图像			
				and a state of the			学习模板	编辑	漠板	
							裝載模板	保存	摸板	
				-	, 1000		數案区域	搜	索	
				V	1996	参数设置				
						搜索数重	1	最小分数	70	
						最小角度	-180	最大角度	180	
						最小比例	100	最大比例	100	
				24	4	重叠距离	20	目标极性	正常	~
				1	•	搜索速度			5	
						定位精度		•	5	
				1						
	分数	位置(X)	位置(Y)	角度	比例					
	99.391	445.995	359.516	0.009	1.000					
						Static				
						Static				

搜索数量:最多被允许搜索到的目标数量。

最小分数:分数表示目标和模板的相似程度,分数越高越相似,最大值100表示完全匹配, 目标分数必须大于该值才会被搜索到,该参数值将会影响搜索速度。

角度:可以设置被搜索目标可能存在的角度范围,角度为目标相对于模板的角度,取值范围 -180~180。

比例:可以设置被搜索目标可能存在的比例范围,比例为目标相对于模板的比例,取值范围 80[~]120。

重叠距离: 匹配多个目标时之间的最小距离不能小于重叠距离, 小于则忽略。

匹配极性:可以设置正常和反转,正常表示目标和模板极性相同,反转则表示相反。

搜索速度: 总共有 10 个等级,等级为 0[~]9,默认为 5,设置的搜素速度级别越高,识别度会 有所下降。

定位精度:总共有 10 个等级,等级为 0[~]9,默认为 5,设置的定位精度级别越高,搜索速度 会有所下降。



7		操作 打 学 英 授	开图像 之习模板 《载模板 《索区域	編辑 保存 投	模板
		打 学 装 機	开图像 之习模板 或模板 索区域	编辑 保存 投	模板
		学援	习模板 载模板 索区域	编辑 保存 搜	模板
		装援	载模板 索区域	保存	模板
		援参教设置	索区域	搜	*
전성에 실패했다. ~ 야간되다. 전성이		参教设罟			糸
		参数设置			
		搜索数里	1	最小分数	70
회원님께 방송되었는 🗢 🔫 그 화원님이 있었다.		最小角度	-180	最大角度	180
		最小比例	100		100
		重叠距离	20	目标极性	正常 ~
		柳壶油度			
		投杂团度			
Sand and a second second		定位精度			5
	States Zent				
分数 位置(X) 位置(Y) 角度	比例				
96.449 262.345 277.996 2.982	1.003				
		Static			
		Static			

- 分数: 匹配目标与模板的相似度。
- 位置: 匹配目标相对于当前图像的坐标位置。
- 角度: 匹配目标相对于模板的旋转角度。
- 比例: 匹配目标相对于模板的缩放比例。

2 Bin_x64

2.1 等同 Bin 文件内功能。



3 Document 文档

CKVision.chm

CKVision 简介.pdf

版本说明.doc

CKVISION SDK 说明



4 Include 开发库头文件

CKAcmeTool.h	顶点测量	(CAcmeTool)	CKMeasure.dll
CKBarcode.h	一维码读取	(CReadBarcode)	CKReader.dll
CKBase.h	基础模块		CKBase.dll
CKBaseDef.h	导出/导入、数据线	结构定义	
CKBlob.h	斑点分析、图像对	比	CKBlob.dll
CKBlobAnalyzer.h	斑点分析	(CBlobAnalyze)	
CKBlobData.h	Blob 数据	(CBlobData)	
CKBlobDef.h	Blob 定义		
CKCalibration.h	标定功能	(CCalibration)	CKCalibration.dll
CKCaliper.h	卡尺、间距测量(C	Caliper)	CKMeasure.dll
CKCharset.h	字符集	(CCharset)	CKReader.dll
CKColor.h	颜色		CKColor.dll
CKColorIdentify.h	颜色颜色识别 (CC	ColorSamples CCol	orldentify)
CKColorMonitor.h	颜色监测	(CColorMonitor)	
CKColorSample.h	颜色样本	(CColorSample)	
CKContour.h	轮廓检测、轮廓缺	·陷	CKContour.dll
CKContourDefect.h	轮廓缺陷	(CContourDef	ect)
CKContourDetect.h	轮廓检测	(CContourDet	ect)
CKDataMatrix.h	读取 DataMatrix 二	二维码 (CDataMatrix) CKReader.dll
CKDotMatrix.h	圆形矩阵标定板	(CDotMatrix)	
CKEdgeTool.h	边缘点检测	(CEdgeTool)	CKMeasure.dll
CKFileStore.h	文件存储结构	(CFileStore)	CKBase.dll
CKFindBarcode.h	读取一维码	(CFindBarcode) CKReader.dll
CKFindModel.h	形状模型搜索	(CFindModel)	CKLocate.dll
CKFitCircle.h	圆拟合工具	(CFitCircle)	CKMeasure.dll
CKFitLine.h	线拟合工具	(CFitLine)	
CKFrameTrans.h	坐标系变换	(CFrameTrans) CKBase.dll



深圳市创科自动化控制技术有限公司

CKGDI.h	图形显示		CKGDI.dll
CKGdiBoxScan.h	旋转矩形框内扫描线	(CGdiBoxScan)	
CKGdiCircle.h	圆形	(CGdiCircle)	
CKGdiContour.h	轮廓图形	(CGdiContour)	
CKGdiEllipse.h	椭圆图形	(CGdiEllipse)	
CKGdiFigure.h	图形功能(基类)	(CGdiFigure)	
CKGdiFrame.h	坐标系显示	(CGdiFrame)	
CKGdiHistogram.h	直方图	(CGdiHistogram)	
CKGdiLine.h	线段图形	(CGdiLine)	
CKGdiMask.h	掩摸显示	(CGdiMask)	
CKGdiModel.h	模型轮廓显示	(CGdiModel)	
CKGdiPoint.h	点、十字显示	(CGdiPoint)	
CKGdiPolygon.h	多边形图形 (C	GdiPolygon)	
CKGdiProfile.h	投影曲线边缘位置	(CGdiProfile)	
CKGdiRect.h	矩形框	(CGdiRect)	
CKGdiRing.h	圆环图形	(CGdiRing)	
CKGdiRingScan.h	圆环内扫描线	(CGdiRingScan)	
CKGdiRotBox.h	旋转矩形	(CGdiRotBox)	
CKGdiText.h	文本显示	(CGdiText)	
CKGdiType.h	模板类显示	(CGdiType)	
CKGdiView.h	图形视图窗口	(CGdiView)	CKGDI.dll
CKGeoMeas.h	基本儿何测重 1)	(CKVISION_API)	CKBase.dll
CKHasp.h	校验锁		
CKHistogram.h	直方图、分割阈值	(CHistogram)	
CKHSIThreshold.h	HSI 颜色抽取 (CH	ISIThreshold)	CKColor.dll



深圳市创科自动化控制技术有限公司

CKImage.h	图像基本功能	(CPrImage)	CKBase.dll
CKImgConve.h	图像转换、高级调整	(CKVISION_API)	CKBase.dll
CKImgFilter.h	图像滤波		
CKImgMorph.h	图像灰度形态学		
CKImgOpera.h	图像算术和逻辑		
CKImgTrans.h	图像变换(镜像、平移	多、旋转、缩放、等	È)
CKLocate.h	形状匹配、识别定位		CKLocate.dll
CKMask.h	图像掩摸	(CMask)	CKBase.dll
CKMeasDef.h	测量定义		CKMeasure.dll
CKMeasure.h	测量		CKMeasure.dll
CKModel.h	模型特征点模板	(CModel)	CKLocate.dll
CKModelContour.h	模型轮廓	(CModelCont	our)
CKNCMatch.h	灰度区域匹配	(CNCMatch)	
CKNCPat.h	灰度模板	(CNCPat)	
CKOverlay.h	覆盖图功能	(COverlay)	CKGDI.dll
CKPatInspect.h	基于图像对比缺陷检测	(CPatInspect) CKBlob.dll
CKPixelStat.h	像素统计功能	(CPixelStat)	CKBase.dll
CKPointVector.h	坐标点容器	(CPointVector) CKMeasure.dll
CKProfile.h	图像截面投影曲线	(CProfile)	CKMeasure.dll
CKReadDXF.h	读取 DXF 文件生成模板	反轮廓 (CReadD	XF) CKGDI.dll
CKReader.h	读取条码、字符		CKReader.dll
CKReadOcr.h	字符识别	(CReadOcr)	CKReader.dll
CKReadQRCode.h	读取 QR 码	(CReadQRCoc	de)
CKScanEdge.h	扫描边缘	(CScanEdge)	CKMeasure.dll
CKScanSpace.h	扫描间距	(CScanSpace))
CKShapeMatch.h	边缘轮廓形状匹配(新)	(CShapeMatch	n) CKLocate.dll
CKShapeModel.h	形状模板(新)	(CShapeModel)	CKLocate.dll
CKSharpAssess.h	图像清晰度评估	(CSharpAsses	s) CKBase.dll



5 Install 运行库安装包

5.1 VC2008 运行包

5.2 加密锁驱动

6 Lib 开发库 Lib 文件

CKBase.lib CKBlob.lib CKCalibration.lib CKColor.lib CKContour.lib CKGDI.lib CKLocate.lib CKMeasure.lib CKReader.lib

7 Lib_x64 开发库 64 位版本文件

CKBase.lib CKBlob.lib CKCalibration.lib CKColor.lib CKContour.lib CKGDI.lib CKLocate.lib CKMeasure.lib CKReader.lib



8 Samples 功能 API 调用实例

8.1)

以 vs 文件->新建->项目->创建 MFC 应用程式, 基于对话框模、在静态库中使用 MFC 的实例。

新建项目					? ×
项目举型(P);		模板(T):		.NET Framework 3.5	×
Visual C++ ATL CLR 單规 MFC 智能设备 Win32 分布式系统 其他项目类型		Visual Studio 已安装的模板 器 MFC ActiveX 拉件 器 MFC 应用程序 我的模板 ① 搜索联机模板	置 MFC DLL	t	
用于创建使用 Micro	osoft 基础类库的应用程	」 『序的项目			
名称(N):	MultiModelDemo				
位置(L):	J:\CKVISION\SDK\C	KVision SDK 5.3.0.1\Samples		~	浏览(B)
解决方案名称(M):	MultiModelDemo		创建解决方案的目录(I)))	
				确定	取消
MFC 应用程序向	回导 - MultiModel 应用程序类	Demo 注型		?	×
概述 应用程序类型 复合文档程序 支档模板专 相户界面功能 高级功能 生成的类	<u>)</u> 持 行 等	应用程序类型: ● 单个文档(<u>s</u>) ● 多个文档(<u>m</u>) ● 适项卡式文档(<u>b</u>) ● 基于对话框(<u>n</u>) ● 使用 HTML 对话框(● 多个顶级文档(<u>T</u>) ② 文档/视图结构支持(<u>y</u>) 资源语言(<u>L</u>): 中文(简体,中国) ☑ 使用 Unicode 库(<u>m</u>)	项目类型 ● MFC ● Wind ● Visu 0 Offi 初觉样式 Windows ■ 启用 MFC 的使 ● 在執 ● 在静	: 标准(点) lows 资源管理器(X al Studio(①) oe(乎) 和颜色(乎): 本机/默认 视觉样式切换(空) 用: 享 DIL 中使用 NFC(至) c())
		〈 上一步	下一步〉	完成	取消



项目属性

199 (0)				Total and		
;宣(C);	Release	~	千台(P):	活动(Win32)	~	配直百理請(O)
通用	属性	□ 常規				
配置	属性	輸出	目录		\$(SolutionDir)\$(Configuration	Name)
2	常规	中间	目录		\$(ConfigurationName)	
ij	周试	清除	时要删除的排	扩展名	*.obj;*.ilk;*.tlb;*.tli;*.tlh;*.tmp;*.rsp	;*.pgc;*.pgd;*.meta
C	C/C++	生成	日志文件		\$(IntDir)\BuildLog.htm	
ŧ	链接器 清单工具	继承	的项目属性制	表		
3		启用	托管增量生质	戓	是	
///→ 资源 XMI 文档生成器	□ 项目默;	默认值				
	8年成器 配置类型			应用程序(.exe)		
25	XIVIL 义恒主成語	MFC	的使用		在静态库中使用 MFC	~
U A		ATL	的使用		不使用 ATL	
-		字符	集		使用 Unicode 字符集	
F	日定义生成少操	生成步骤 公共语 全程序	共语言运行时支持 无公共语言运行时支持			
			序优化		使用链接时间代码生成	
	MFC	使用				

添加当前 CKVISION 开发功能对应的.h 与 Lib 文件。

选择到解决方案管理器->选择到当前的项目,展开列表,找到 Header Files->StdAfx.h 双 击打开文件,然后添加:

```
#include "..\\..\Include\\CKGDI.h"
#include "..\\..\Include\\CKBase.h"
#include "..\\..\Include\\CKLocate.h"
```

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKLocate.lib")
```

#else

```
#pragmacomment(lib, "..\\..\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\Lib\\CKLocate.lib")
#endif
```

usingnamespaceCKVision; // CKVISION 命名空间



在程序入口处增加 InitLibrary 函数,用于初始化 CKVision 库,只有调用初始函数:

CKVision::InitLibrary(); // 初始化CKVision库

初始化之后才能正常其它图像处理的功能。

在程序退出终止处增加 ExitLibrary 函数,用于释放 CKVision 库:

CKVision::ExitLibrary(); // 退出 CKVision 库 名称 BarcodeDemo BlobToolDemo CalibrationDemo CaliperDemo ColorMatchDemo ColorThresholdDemo ContourDemo DataMatrixDemo EdgeToolDemo FitCircleDemo FitLineDemo HistogramDemo ImageDemo ImageWarpDemo ImgTransDemo InspectDemo ModelDemo MultiModelDemo NCMatchDemo QRCodeDemo ReadOcrDemo SearchDemo AllDemos.ncb AllDemos.sln AllDemos.suo

以 vs 2008 以上版本打开 AllDemos.sln 加载功能实例文件。



深圳市创科自动化控制技术有限公司

打开工程文件后,可以选择类视图->选择当前需要查看的项目,右键鼠标->弹出菜单设置当前启动项目。

	1999	生成(U)	
奕倪图	_	●新生成(F)	
🚰 🗢 🔿 🛅 ·		主机土加(L)	
<搜索>		清埋(N)	
T Parcodo Don		仅用于项目(J)	•
		按配置优化(P)	•
		T石円/土井石(C)	
CalibrationD		坝日舣,颗坝(5)	
		项目生成顺序(I)	
		自定义生成规则(B)	
Color I hresh		工具生成顺序(L)	
ContourDem			
DataMatrixD		添加	•
EdgeToolDe		引用(F)	
FitCircleDem		添加 Web 引用(E)	
E FitLineDemo		沿为户动项目(A)	
HistogramD			
🗄 👔 ImageDemo		调试(G)	•
🗄 🚰 ImageWarpl		保存 SearchDemo (S)	
🗄 🚰 ImgTransDe	ssaren	按字母顺序排序(S)	
InspectDemo			
🗄 📅 ModelDemo	~	按对象类型排序(1)	
NCMatchDer		按对象访问排序(E)	
🕀 📅 QRCodeDen		按对象类型分组(U)	
🗄 📅 ReadOcrDer	Ph.	犀 性(R)	
🗄 📷 SearchDeme	-123	/mai (1 ()	

8.2)

->项目属性->配置属性->常规->MFC 的使用

->在静态库中使用 MFC



深圳市创科自动化控制技术有限公司

米加图		生成(U)	- 1
		重新生成(E)	
		清理(N)	
<搜索>		仅用于项目(J)	
🗄 📅 Barcodel		按配置优化(P)	-
🗄 🚰 BlobToo			-
🕀 📅 Calibratio		项目依赖项(S)	
🗄 🚰 CaliperD		项目生成顺序(I)	
🗄 🚰 ColorMa		自定义生成规则(B)	
🕀 🚰 ColorThr		工具生成顺序(L)	
Contourl			7
⊕ 🚰 DataMat			_
EdgeToc		与I用(F)	
		添加 Web 引用(E)	
FitLineDe		设为启动项目(A)	
Histogra		调试(G)	
ImageDe	-		-
🗄 🚰 ImageWi		保存 SearchDemo (S)	_
ImgTran		按字母顺序排序(S)	
🗄 🚰 InspectD		按对象举型排序(T)	
🕀 🚰 ModelDe			- 1
🗄 📅 NCMatch		按刘家功问俳序(E)	- 1
De To QRCodel		按对象类型分组(U)	
🕀 🚰 ReadOcr		属性(R)	
BearchDe	mo		_



	no 温住贝				? X
置(C):	活动(Release)	~ 平台(P):	活动(Win32)	~	配置管理器(0)
通配置 賞 谐 乙 推 消 论 X 沈 性 目		日 常规 输出目录 中间目录 清除时要删除的 生成日志文件 继承的项目属性非 同用影汰值 配置类型型 MFC的使用 存符集 公共语言运行时 全程序优化	扩展名 表 或 支持	·\Release ·\Release *.obj;*.ilk;*.tlb;*.tli;*.tlh;*.tmp;*.rsg \$(IntDir)\BuildLog.htm \$(VCInstallDir)VCProjectDefaults 是 应用程序(.exe) 在静态库中使用 MFC 不使用 ATL 使用多字节字符集 无公共语言运行时支持 无全程序优化	o;*.pgc;*.pgd;*.meta :\UpgradeFromVC6

8.3)->项目属性->查看编译生成输出的文件路径。

连接器->输出文件.

置(C):	活动(Release)	~	平台(P):	活动(Win32)	~	· 配置管理器(O)	
通用	属性	∧ 輸出	文件	337	\\Bin\SearchDemo.exe		
/ 配置	置属性	显示	进度		未设置		
	常规	版本					
	调试	启用	増量链接		否(/INCREMENTAL:NO)		
> C/C++		取消	取消显示启动版权标志		是(/NOLOGO)		
~	链接器	忽略	导入库		否		
	常规	注册	注册输出		否		
	输入		每个用户的重定向		否		
			库目录				
	调试	链接	库依赖项		是		
	~ 法	使用	库依赖项输入	λ	否		
	2557G	使用	UNICODE	响应文件	是		
	設入町」IDL						
5	中文行						
>	清甲 上具						
2		-					
>	XML 又档生成器	输出文	件				
>	浏覧信息	重写默	认的输出文件	‡名。 (/OUT:[fil	e])		
>	生成爭件	~					



8.4)

添加当前 CKVISION 开发功能对应的.h 与 Lib 文件。

选择到解决方案管理器->选择到当前的项目,展开列表,找到 Header Files->StdAfx.h 双 击打开文件,然后添加:

#include "..\\..\Include\\CKGDI.h"
#include "..\\..\Include\\CKBase.h"
#include "..\\..\Include\\CKLocate.h"

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKLocate.lib")
e
```

#else

```
#pragmacomment(lib, "..\\..\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\Lib\\CKLocate.lib")
#endif
```

usingnamespaceCKVision; // CKVISION 命名空间

```
解决方案资源管理器 - 解决方案'AllDemo... ▼ 및 X
                                                                                                                                                     StdAfx.h
   🗟 | 🗗 | 🗉 🍇
                                                                                                                                                   StdAfx.h
                                                                                                                                                                                                                               ✓ ↓ j:\CKVISION\SDK\CKVision SDK
   E ContourDemo
                                                                                                                                                   (全局范围)
   🗄 📲 DataMatrixDemo
   🗄 \overline 🔠 EdgeToolDemo
                                                                                                                                                      #if _MSC_VER > 1000
   🗄 📅 FitCircleDemo
                                                                                                                                                          #pragma once
#endif // _MSC_VER > 1000
   😥 📅 FitLineDemo
                                                                                                                                                         #define VC_EXTRALEAN
                                                                                                                                                                                                                                    // Exclude rarely-used stuff from Wind
   🗄 📅 HistogramDemo
                                                                                                                                                        #include (afreein h) // MPC core and standard components
#include (afreein h) // MPC extensions
#include (afredin h) // MPC extension classes
#include (afredin h) // MPC support for Internet Explorer 4
#include (afreen h) // MPC support for Windows Common Cont
#endif // _AFX_NO_AFXCDM_SUPPORT
   🗄 📅 ImageDemo
   🗄 📅 ImageWarpDemo
   🗄 📅 ImgTransDemo
   🗈 \overline InspectDemo
   🗄 📅 ModelDemo
   - MCMatchDemo
                                                                                                                                                         #include "...\\...\\Include\\CKGDI.h"
#include "...\\..\\Include\\CKBase.h"
#include "...\\..\\Include\\CKLocate.h"
   🗄 📲 QRCodeDemo
   🗄 📲 ReadOcrDemo
                                                                                                                                                      #ifdef WIN64
   🖃 📷 SearchDemo
                                                                                                                                                                     det #1864
#pragma comment(lib, "..\\.\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\.\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\.\Lib_x64\\CKLocate.lib")
                 - 🗁 Header Files
                              ModelDlg.h
                                                                                                                                                      = #else
                                                                                                                                                         ##15e
#pragma comment(lib, "..\\..\Lib\\CKBase lib")
#pragma comment(lib, "..\\..\Lib\\CKBUL.lib")
#pragma comment(lib, "..\\..\Lib\\CKLocate.lib")
#endif
                              h Resource.h
                             b SearchDemo.h
                             h SearchDemoDlg.h
                            h StdAfx.h
           Resource Files
                                                                                                                                                          using namespace CKVision;
                                                                                                                                                       White According to the second 
                             SearchDemo.ico
                              SearchDemo.rc2
             🗄 🗁 Source Files
                             🕶 ModelDlg.cpp
                                                                                                                                                     |#pragma comment(linker, /wanifestdependency:\"type="win32' name='#
|#mragma comment(linker,"/manifestdependency:\"type="win32' name='#
                              🐏 SearchDemo.cpp
                                                                                                                                                      = #pragma
                             SearchDemo.rc
                                                                                                                                                                           a comment(linker, "/manifestdependency:\"type='win32' name='W
                              SearchDemoDlg.cpp
                                                                                                                                                          #pragma c
#endif
//#endif
                              StdAfx.cpp
                     ReadMe.txt
                                                                                                                                                       //{{AFX INSERT LOCATION}}}
```



8.5)

在程序入口处增加 InitLibrary 函数,用于初始化 CKVision 库,只有调用初始函数:

CKVision::InitLibrary(); // 初始化CKVision库

之后才能正常其它图像处理的功能。

在程序退出终止处增加 ExitLibrary 函数,用于释放 CKVision 库: CKVision::ExitLibrary(); // 退出 CKVision 库





BOOL CQRCodeDemoApp::InitInstance() ↓ {



注:详细代码请打开对应的...Demo 功能实例。

BarcodeDemo 一维码检测

```
1.) 在 StdAfx.h 的头文件中添加读取条码相关的文件链接。
    //.h 头文件
#include "..\\..\\Include\\CKGDI.h"
#include "..\\..\\Include\\CKBase.h"
#include "..\\..\\Include\\CKReader.h"
//.lib 文件
#ifdef_WIN64
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKReader.lib")
#pragma comment(lib, "..\\..\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\Lib\\CKBase.lib")
```

usingnamespaceCKVision;



解决方案资源管理器 - BarcodeDemo ▼ 早	X CKReader, StdAfx.h BarcodeDemo.rcO_DIALOG - Dialog
陆 📴 Z	→ CKGDI.h → → h j:\CKVISION\SDK\CKVision !
😡 解决方案'AllDemos' (21 个项目)	(全局范围)
😑 📅 BarcodeDemo	
🖨 🗁 Header Files	If a staarx h : include file for standard system include files, // or project specific include files that are used frequently
- h BarcodeDemo.h	// are changed infrequently
- 🖻 BarcodeDemoDlg.h	
- 🖻 Resource.h	#if !defined(AFX_STDAFX_H_12E0F1E5_F928_484D_9874_5231C877724 ed.fi., AFX_STDAFX_H_12E0F1E5_F928_484D_9874_5231C877724 ed.fi., AFX_STDAFX_H_12E0F1E5_F928_484D_9874_5231C877724
🔤 🛅 StdAfx.h	Wdernie Wikjstowikjs_izzorizojrozojrozojrozojrozoicornizoo_zi
🐵 🗀 Resource Files	#if _MSC_VER > 1000
🖶 📴 Source Files	#endif // _MSC_VER > 1000
- 🔤 BarcodeDemo.cpp	#define W_ BYTPALBAN // Realade wavelenned shaff from
🛃 BarcodeDemo.rc	Wdefine Vo_Exhibitions // Exclude Fallery died staff from
🥶 BarcodeDemoDlg.cpp	#include (afravin.h) // MFC core and standard componen: #include (afravit.h) // MFC avianzions
- 🚰 StdAfx.cpp	#include (afxdisp.h) // MFC Automation classes
ReadMe.txt	<pre>#include (afxdtctl h) // MFC support for Internet Explos</pre>
🕢 🎁 BlobToolDemo	<pre>#include <afxcmn h=""> // MFC support for Windows Common</afxcmn></pre>
CalibrationDemo	#endif // _AFX_NO_AFXCMN_SUPPORT
💼 🔞 CaliperDemo	
ColorMatchDemo	<pre>#include ~</pre>
ColorThresholdDemo	#include " \\ \\Include \\CKReader. h"
ContourDemo	stifdef WIN64
DataMatrixDemo	<pre>#pragma comment(lib, "\\\\Lib_x64\\CKGDI.lib")</pre>
EdgeToolDemo	<pre>#pragma comment(lib,\\\\Lib_x64\\CKBase.lib) +#pragma comment(lib, "\\\\Lib x64\\CKReader.lib")</pre>
🗑 🎁 FitCircleDemo	e#else
FitLineDemo	<pre>#pragma comment(lib,</pre>
HistogramDemo	<pre>#pragma comment(lib, "\\\\Lib\\CKReader.lib")</pre>
ImageDemo	*enair
ImageWarpDemo	using namespace CKVision;
- ImgTransDemo	//#ifdef_UNICODE
InspectDemo	<pre>b#if defined _M_IX86</pre>
🗉 📷 ModelDemo	stelif defined _M_IA64
MCMatchDemo	+#pragma comment(Linker, "/manifestdependency:\"type= win32" nar
🕢 📅 QRCodeDemo	#pragma comment(linker, "/manifestdependency:\"type='win32' nar
🕢 🌆 ReadOcrDemo	<pre>##ise +#pragma comment(linker. "/manifestdependency:\"type='win32' nag</pre>
🖅 🛅 SearchDemo	#endif
Non to a manager and we have a second	//#endit

//初始化 CKVision 库





2.)	在资源视图	Dialog 中添加相应的界面操作。	,
-----	-------	--------------------	---

资源视图 - BarcodeDemo ▼ 平 X	CKReader.h StdAfx.h BarcodeDemo.rcO_DIALOG - Dialog BarcodeDemoDlg.h	
🖶 🚰 BarcodeDemo		
🚊 🛅 BarcodeDemo.rc	Line handered en der	
🖨 🦢 Dialog	E BarcodeDemo	83
- IDD_ABOUTBOX		
IDD_BARCODEDEMO_DIALOG		1¥1F
🗉 📜 Icon	- 确定	
🗄 🛅 String Table	取消	打开图像
🖲 🫅 Version		
BlobToolDemo		ROI
CalibrationDemo		
CaliperDemo		* 急301
ColorMatchDemo		
ColorThresholdDemo		
🗄 쫼 ContourDemo		参数
🔋 🚰 DataMatrixDemo		1 martine
🖶 📴 EdgeToolDemo		读取余码英
🖶 📴 FitCircleDemo		
🗄 🚰 FitLineDemo		
🖶 🚰 HistogramDemo		
🖶 📴 ImageDemo		
🗑 📴 ImageWarpDemo		
🗄 📴 ImgTransDemo		
🕮 🎦 InspectDemo		
🕮 🊰 ModelDemo		滤波器半宽 示例编辑框 🛟
		an ano an an an And an An
🐵 🚰 QRCodeDemo		特長順加盟 不均均用料性
🕫 📴 ReadOcrDemo		
🗄 📴 SearchDemo		
	:	信息
	定位梯度 示例编辑框 最大条宽度 示例编辑框 □水平 □垂直 □45度 □135度	
		Static

3.) 在对话框窗口的 .h 头文件中定义相应的图像处理功能:

	CPrImage	m_Image; // 基础图像类 m_FindBC;// 条码定位	
	CFindBarcode		
	CReadBarcode	m_Barcode;	// 读取条码
	CGdiRect	m_Rect;	// 矩形框显示
	COverlay	m_Overlay;	// 图像显示表面, 前显示的动态图形,主要用于ROI 显示。
显示	COverlay	m_Results;	// 图像显示表面, 前显示的静态图形,主要用于检测结果生成图形
	CGdiView	m_GdiView;	// 图形视图窗口

4.) 在对话框窗口的.cpp 实现文件中添加相应的功能实现。



深圳市创科自动化控制技术有限公司



// 执行条码读取

```
voidCBarcodeDemoDlg::OnExecute()
```

```
{
```

 $/\!/$ TODO: Add your control notification handler code here

```
m_Barcode.Release();
Overlay_DeleteAll(m_Results);
```

// 一维码类型

```
intnBarcodeType = 0;
```

```
if( m_List1.GetSel(0) )
    nBarcodeType |= BARCODE_UPC_A;
if( m_List1.GetSel(1) )
```

nBarcodeType |= BARCODE_UPC_E;

if(m_List1.GetSel(2))

nBarcodeType |= BARCODE_EAN_8;

if(m_List1.GetSel(3))

nBarcodeType |= BARCODE_EAN_13;

```
if( m_List1.GetSel(4) )
    nBarcodeType |= BARCODE_CODE_39;
```

```
if( m_List1.GetSel(5) )
    nBarcodeType |= BARCODE_CODE_93;
```



```
if( m_List1.GetSel(6) )
    nBarcodeType |= BARCODE_CODE_128;
```

```
if( m_List1.GetSel(7) )
    nBarcodeType |= BARCODE_INTERLEAVED_2_5;
```

m_Barcode.SetReadType(nBarcodeType);

- m_Barcode.SetFilterHalf(GetDlgItemInt(IDC_EDIT1));
- m_Barcode.SetThreshold(GetDlgItemInt(IDC_EDIT2));

intfx = 0;

- if(IsDlgButtonChecked(IDC_CHECK1)) $\label{eq:star} \begin{array}{l} f_x \mid = 0 \\ x \\ \end{array}$
- if(IsDlgButtonChecked(IDC_CHECK2))
 fx = 0x02;
- if(IsDlgButtonChecked(IDC_CHECK3)) $f_x = 0x04;$
- if(IsDlgButtonChecked(IDC_CHECK4)) $f_{\rm X} = 0x08;$
- m_FindBC.SetMaxCount(20);
- $m_FindBC.\,SetOrientation(fx);$

 $\mbox{m_FindBC.SetThreshold(GetDlgItemInt(IDC_EDIT4)});$

m_FindBC.SetMaxSpace(GetDlgItemInt(IDC_EDIT3));

// 开始计算时间

BeginTime();

```
// 执行条码定位
if( m_Rect.GetVisible() ) {
    m_FindBC.Execute( m_Image, m_Rect );
} else {
    m_FindBC.Execute( m_Image, MaxROI );
}
ROTRECTrc, *pRect;
for( inti=0; i<m_FindBC.GetNumResults(); i++ )
    Duct = m_FindBC.GetNumResults(); i++ )</pre>
```

```
pRect = m_FindBC.GetCodeBorder(i);
rc.center = pRect->center;
rc.angle = pRect->angle;
rc.width = pRect->width+10;
```

{



```
rc.height = min(max(pRect->height-50, 10), 80);
    // 指定位置读取条码
    if( m Barcode.Execute( m Image, rc ) ) {
         CGdiRotBox* p1 = newCGdiRotBox(*pRect);
         if( p1!=NULL ) {
              p1 \rightarrow 0 ff set(0.5, 0.5);
              p1->SetPenWidth(2);
              if( m_Barcode.GetCodeLen()>0 )
                  p1 \rightarrow SetPenColor(RGB(0, 255, 0));
              else
                  p1->SetPenColor( RGB(255,0,0) );
              m Results. AddItem(p1);// 添加显示图形到画面上显示
         }
         break:
    }
}
// 结束计算时间周期
EndTime();
SetDlgItemText( IDC_CODE_TEXT, m_Barcode.GetCodeText() );
m GdiView.Redraw();
                    // 视图刷新显示
```

BlobToolDemo 斑点分析

}

1.)在 StdAfx.h 的头文件中添加读取条码相关的文件链接。

```
#include "..\\..\\Include\\CKGDI.h"
#include "..\\..\\Include\\CKBase.h"
#include "..\\..\\Include\\CKBlobAnalyzer.h"
```

```
#ifdef _WIN64
#pragma comment(lib, "..\\..\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKBlob.lib")
#else
#pragma comment(lib, "..\\..\Lib\\CKGDI.lib")
#pragma comment(lib, "..\\..\Lib\\CKBase.lib")
```



#pragma comment(lib, "..\\..\Lib\\CKBlob.lib")
#endif

using namespace CKVision;

//在程式入口和退出的地方增加初始化和释放 CKVISION 库。 BOOLCBlobToolDemoApp::InitInstance()

CKVision::InitLibrary(); // 初始化 CKVision 库. //.....

CKVision::ExitLibrary(); // 退出 CKVision 库



2.)在资源视图 Dialog 中添加相应的界面操作。


资源视图 - BlobToolDemo 🔹 부 🗙	BlobToolDemo.rcO_DIALOG - Dialog BlobToolDer
🗉 🌇 BarcodeDemo	
🖶 🚰 BlobToolDemo	
BlobToolDemo.rc	Blob1oolDemo
🖮 🛅 Bitmap	Tab 1 Tab 2 Tab 3 Tab 4 1 1
🖨 🛅 Dialog	
-B IDD_PARAM1_DLG	
IDD_PARAM2_DLG	
🗈 🫅 Icon	
🗊 🧰 String Table	
🗄 🧰 Version	
🗈 📷 CalibrationDemo	
🗈 🌇 CaliperDemo	
🗄 🚰 ColorMatchDemo	
🗈 🚰 ColorThresholdDemo	
🗄 🚰 ContourDemo	
🗄 🚰 DataMatrixDemo	
🗈 🚰 EdgeToolDemo	
🗉 🚰 FitCircleDemo	
🗄 🚰 FitLineDemo	
🗉 🚰 HistogramDemo	
🗄 🚰 ImageDemo	
🗄 🌇 ImageWarpDemo	
🗉 🚰 ImgTransDemo	
🗈 🚰 InspectDemo	
🗉 🚰 ModelDemo	
🗄 🚰 NCMatchDemo	
🗄 🚰 QRCodeDemo	
🗄 🚰 ReadOcrDemo	- · · · · · · · · · · · · · · · · · · ·
🗄 🚰 SearchDemo	打开图像 第首色
	● ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

voidFilter(intnMeasure, doubledMin, doubledMax, BOOLbInvert);// 过滤 voidSort(intnMeasure, intnOrder);// 排序

voidThresholdImage(INTvalue);// 设置二值化阈值显示 voidThresholdDraw(BOOLdraw);// 显示二值化图像

// 数据结果插入到列表中显示

```
voidInsertBase( int&nColNum );
voidInsertBound( int&nColNum );
voidInsertMoment( int&nColNum );
voidInsertConvex( int&nColNum );
voidInsertMinBox( int&nColNum );
```

// 基础图像类

CPrImage m_Image; CPrImage m_TImage;

// 掩摸图 CMask m_Label; // ROI 检测区域



CGdiRect m_Rect; // Blob 分析工具类 CBlobAnalyzer m_blobTool;

// 图像显示表面,

COverlay	m_Overlay;	//前显示的动态图形,主要用于ROI	显示。
COverlay	m_Results;		
CGdiView	m_GdiView;	// 图形视图窗口	

// 参数设置

CParam1D1g	m_Param1D1g;
CParam2D1g	m_Param2D1g;

在对话框窗口的.cpp 实现文件中添加相应的功能实现。 // 对话框窗口初始化函数中 BOOLCBlobToolDemoDlg::OnInitDialog()

{

```
// TODO: Add extra initialization here
CRectdrc;
m_Tab1.GetWindowRect( &drc );
ScreenToClient( &drc );
drc.DeflateRect( 10, 25, 10, 10 );
m_Tab1.InsertItem( 0, "参数设置" );
m_Tabl.InsertItem(1, "其它功能");
m_Tab1.SetCurSel( 0 );
// 创建参数设置窗口
m_Param1Dlg.Create( IDD_PARAM1_DLG, this );
m_Param1Dlg.ShowWindow(SW_SHOW);
m_Param1Dlg.MoveWindow( &drc );
m_Param2Dlg.Create( IDD_PARAM2_DLG, this );
m_Param2Dlg.MoveWindow( &drc );
m_List1.SetExtendedStyle( 0x20 );
RECTrect;
GetDlgItem(IDC_GDI_RECT)->GetWindowRect( &rect );
ScreenToClient( &rect );
// 窗口图形显示视图
```



```
m_GdiView.Create( m_hWnd, rect );
    m GdiView.SetBackColor( RGB(0,0,64) );
    m GdiView. SetDisplayImage(&m Image);// 显示当前的图像
    m_GdiView.SetActiveOverlay(&m_Overlay);// ROI 显示
    m_GdiView.SetStaticOverlay(&m_Results);// 结果图形显示
    m Overlay. AddItem(&m Rect);//将需要显示的图形添加到覆盖图容器中。
    m Rect.left
                      = 100;
    m_Rect.top
                      = 100;
    m_Rect.right = 500;
    m Rect. bottom = 400;
    m_Rect.SetPenColor(RGB(255,0,0));
    m Rect.SetVisible( false );
    m_Displ.SetCheck(1);
}
// 在执行按钮中
voidCBlobToolDemoDlg::OnExecute()
{
    // TODO: Add your control notification handler code here
    m blobTool.SetBlobType( m Param1Dlg.m Combo1.GetCurSel() );
    m_blobTool.SetConnexity( m_Param1Dlg.m_Combo2.GetCurSel() );
    m_blobTool.SetThreshold( m_Param1Dlg.GetDlgItemInt( IDC_EDIT1 ) );
    m_blobTool.SetLimitArea( m_Param1Dlg.GetDlgItemInt( IDC_EDIT2 ) );
    m_blobTool.SetFeatures( m_Param1Dlg.GetFeature() );
    BeginTime();
    //直方图是否自动分割二值化阈值
    CHistogramhist;
    hist.SetAnalyse(Analyse_Threshold);// 设置分析分割阈值参数
    if( m_Rect.GetVisible() ) {
         if(m_Param1Dlg.m_bAutom.GetCheck()) {// 自动计算二值化阈值
             hist.Execute( m_Image, m_Rect );
             m_blobTool.SetThreshold( hist.GetThreshold() );
             m_Param1Dlg.UpdateThreshold( hist.GetThreshold() );
        }
        m_blobTool.Execute(m_Image, m_Rect);// 执行Blob 分析
```



```
} else {
    if( m_Param1Dlg.m_bAutom.GetCheck() ) {
        hist.Execute( m_Image, MaxROI );
        m_blobTool.SetThreshold( hist.GetThreshold() );
        m_Param1Dlg.UpdateThreshold( hist.GetThreshold() );
    }
    m_blobTool.Execute( m_Image, MaxROI );// 执行Blob 分析
}
EndTime();
```

```
intnColNum=0;
m_List1.DeleteAllItems();
Overlay_DeleteAll(m_Results);
while(m_List1.DeleteColumn(0));
```

// 添加基本特征

InsertBase(nColNum);

// 添加外接矩形特征

// 添加力主轴特征

// 添加凸包相关特征

// 添加最小外框特征

m_GdiView.Redraw();// 刷新视图

}



CalibrationDemo 标定校准

注意: CKVISION SDK 添加 .h 与 lib 文件、初始方法步骤与以上功能相同。

在 StdAfx.h 的头文件中添加 CKVISION 相关定义

#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKDotMatrix.h"
#include"..\\..\\Include\\CKCalibration.h"

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKGase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKCalibration.lib")
#else
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKCalibration.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKCalibration.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKCalibration.lib")
```

usingnamespaceCKVision;

// CalibrationDemo.cpp : Defines the class behaviors for the application. 在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。 B00LCCalibrationDemoApp::InitInstance()

CKVision::InitLibrary(); // 初始化 CKVision 库 // // 退出 CKVision::ExitLibrary(); // 退出 CKVision 库



深圳市创科自动化控制技术有限公司

映方案资源管理器・解決… ▼ ↓ 🗙	CalibrationDemo.cpp StdAfx.h CKCalibration.h CKDotMa
1 1 1 2 L	
灵 解决方案'AllDemos' (21 个项目)	CollibrationDomoApp
🗁 🛅 BarcodeDemo	
🗁 🚰 BlobToolDemo	// The one and only CCalibrationDemoApp object
- 📅 CalibrationDemo	CC-TibuttingBurnham Alabami
🚔 🗁 Header Files	CCalibrationDemoxpp theApp;
🖻 CalibrationDemo.h	//////////////////////////////////////
🖻 CalibrationDemoDlg.h	// ccaribrationemoxpp initialization
h Resource.h	BOOL CCalibrationDemoApp::InitInstance()
🖻 StdAfx.h	AfxEnableControlContainer();
🖅 🚞 Resource Files	CKVision::TnitLibrary(): // 初始化CKVision库
🚊 🗁 Source Files	
- 🚰 CalibrationDemo.cpp	// Standard initialization // If you are not using these features and wish to red
🛃 CalibrationDemo.rc	// of your final executable, you should remove from t
- 🔤 CalibrationDemoDlg.cpp	- // the specific initialization routines you do not ne
🔤 StdAfx.cpp	#ifdef _AFXDLL
ReadMe.txt	Enable3dControls(); // Call this when using MP
🗠 🚰 CaliperDemo	Enable3dControlsStatic(); // Call this when linking
🗁 📅 ColorMatchDemo	#endif
🗠 🚰 ColorThresholdDemo	CCalibrationDemoDlg dlg;
- 📅 ContourDemo	int nResponse = dlg. DoModal();
🗁 📴 DataMatrixDemo	if (nResponse = IDOK)
🗁 🚰 EdgeToolDemo	I // TODO: Place code here to handle when the dialog
- 📅 FitCircleDemo	- // dismissed with OK
🗠 🚰 FitLineDemo	else if (nResponse = IDCANCEL)
🗁 📴 HistogramDemo	[// TODO: Place code here to herdle when the dialog
- 🚰 ImageDemo	- // dismissed with Cancel
🗁 📴 ImageWarpDemo	
- 🚰 ImgTransDemo	CKVision::ExitLibrary(); // 退出CKVision库
🗁 📴 InspectDemo	// Since the dialog has been closed, return FALSE so t
- 🚰 ModelDemo	// application, rather than start the application's m
- 🚰 NCMatchDemo	return FALSE;
- 📴 QRCodeDemo	
🗁 📴 ReadOcrDemo	
SearchDemo	

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

```
// CalibrationDemoDlg.h : header file
在对话框窗口的 .h 头文件中定义相应的图像处理功能:
// CCalibrationDemoDlg dialog
   CPvImage
             m_ImageSrc;// 来源图像
   CPvImage
             m_ImageDst;// 校正后的图像
   COverlay
             m_Overlay;// 图像显示表面, 前显示的动态图形, 主要用于ROI 显示。
             m_Results;// 图像显示表面, 前显示的静态图形, 主要用于检测结果生成图形显示。
   COverlay
   CGdiView
             m_GdiView;// 图形显示视图窗口
   CGdiRect
              m_Roi;
                       // 矩形区域
                 m_DotMatrix; //获取圆形矩阵点位置数据
   CDotMatrix
   CCalibration m_Calibration; // 执行标定、图像畸变校正、九点标定。
```

//.....

// CalibrationDemoDlg.cpp : implementation file



```
在对话框窗口的.cpp 实现文件中添加相应的功能实现。
//CCalibrationDemoDlg message handlers
BOOLCCalibrationDemoDlg::OnInitDialog()
{
        // TODO: Add extra initialization here
    CRectrect;
    GetDlgItem(IDC_VIEW_RECT)->GetWindowRect( &rect );
    ScreenToClient( &rect );
    m GdiView.Create( m hWnd, rect );
                                            // 按自定义的区域创建图形显示视图窗口。
    m_GdiView.SetBackColor( RGB(0,0,64) );
                                             // 设置默认的背景颜色
    m GdiView.SetDisplayImage(&m ImageDst); // 设置当前显示的图像
    m_GdiView.SetActiveOverlay(&m_Overlay); // 设置当前显示的覆盖图ROI 图形显示。
    m_GdiView.SetStaticOverlay(&m_Results); // 设置当前显示的覆盖图结果图形显示。
    m_Overlay.AddItem( &m_Roi );
                                             //将需要显示的图形添加到覆盖图容器中。
}
// 获取图像上的标定点
voidCCalibrationDemoDlg::OnCheckDot()
{
    // TODO: Add your control notification handler code here
    BeginTime();
    m DotMatrix.SetDotType( m Combol.GetCurSel() );
    m_DotMatrix.SetMinArea(GetDlgItemInt(IDC_EDIT1));
    m_DotMatrix.SetMaxArea(GetDlgItemInt(IDC_EDIT2));
    CHistogramhistTool;
    histTool.SetAnalyse( Analyse_Threshold );
    if( m_Roi.GetVisible() ) {
        histTool.Execute( m_ImageSrc, m_Roi );
        m_DotMatrix.SetThreshold( histTool.GetThreshold() );
    if( !m_DotMatrix.Execute( m_ImageSrc, m_Roi ) ) {
            AfxMessageBox(_T("获取标定点失败!"));
        }
    } else {
        histTool.Execute( m_ImageSrc, MaxROI );
```



```
m_DotMatrix.SetThreshold( histTool.GetThreshold() );
         if( !m_DotMatrix.Execute( m_ImageSrc, MaxROI ) ) {
             AfxMessageBox(_T("获取标定点失败!"));
         }
    }
    Overlay_DeleteAll(m_Results);
    DPNT* points = m_DotMatrix.GetDotData();
    for( inti=0; i<m_DotMatrix.GetDotCount(); i++ ) {</pre>
         CGdiPoint* p1 = newCGdiPoint(points[i]);
         if( p1!=NULL ) {
             p1->SetStyle(1);
             p1->SetPenColor(RGB(0, 255, 0));
             m Results.AddItem( p1 );
         }
    }
    EndTime();
    SelectView( 0 );
}
// 执行标定
voidCCalibrationDemoDlg::OnExecute()
{
    // TODO: Add your control notification handler code here
    UpdateData();
    BeginTime();
    m_Calibration.SetActualSpace( m_dUnitSpaceX );
    m_Calibration.SetCalibrationType( m_Combo2.GetCurSel() );
    // 执行标定
    if( !m_Calibration.Execute( m_DotMatrix.GetDotData(), m_DotMatrix.GetDotCount() ) )
    {
         AfxMessageBox( "标定失败!");
         return;
    }
    // 执行多个点标定功能(九点标定)、坐标系变换
```



```
// pointSrc
             来源点,图像上的坐标点
             目标点,实际单位的坐标点
// pointDst
// nCount 点的数量
// bool Execute( const DPNT* pointSrc, const DPNT* pointDst, int nCount );
if ( !m Calibration.CreateLUTable ( m ImageSrc.GetWidth(), m ImageSrc.GetHeight() ) )
{
    AfxMessageBox( "创建图像校正查找表失败!");
    return;
}
EndTime();
CStringtext;
text.Format( _T("%0.9f"), m_Calibration.GetPixelScale() );
SetDlgItemText( IDC_PIXEL_SCALE_X, text );
SelectView( 0 );
```

```
//.....详情请打开实例参考。
```

}

CaliperDemo 卡尺测量、间距检测

// stdafx.h : include file for standard system include files,or project specific include
// files that are used frequently,butare changed infrequently
在 StdAfx.h 的头文件中添加 CKVISION 相关定义

#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKCaliper.h"

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
```



#endif

usingnamespaceCKVision;

```
在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。
// CaliperDemo.cpp : Defines the class behaviors for the
application.BOOLBOOLCCaliperDemoApp::InitInstance()
{
    CKVision::InitLibrary(); // 初始化 CKVision 库
    //...
    //...End
    CKVision::ExitLibrary(); // 退出 CKVision 库
}
```

```
在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。
```

```
在对话框窗口的 .h 头文件中定义相应的图像处理功能:
```

```
// CaliperDemoDlg.h : header file
```

```
// CCaliperDemoDlg dialog
```

	CPrImage	m_Image;// 基础图像
	CGdiRotBox	m_Rect;// ROI 检测区域
	CCaliper	m_Caliper;//卡尺测量、间距检测、
	COverlay	m_Overlay;// 图像上层显示表面, 前显示的动态图形,主要用于ROI 显示。
	COverlay	m_Results;// 图像上层显示表面, 前显示的静态图形,主要用于检测结果生成图形
显示	0	
	CGdiView	m_GdiView;// 图形显示视图窗口
	CcdiProfile	m Profile: // 坍影曲线边缘位置显示

```
CGdiProfilem_Profile;// 投影曲线边缘位置显示COverlaym_ProOver;// 曲线图形显示CGdiViewm_ProView;// 图形显示视图窗口
```

```
在对话框窗口的.cpp 实现文件中添加相应的功能实现。
```

```
// CalibrationDemoDlg.cpp : implementation file
```

```
// CCaliperDemoDlg message handlers
```

```
BOOLCCaliperDemoDlg::OnInitDialog()
```

```
{
```

```
//...
```



```
RECTrect;
   GetDlgItem(IDC GDI RECT)->GetWindowRect( &rect );
    ScreenToClient( &rect );
                                             // 按自定义的区域创建图形显示视图窗口。
    m GdiView.Create( m hWnd, rect );
    m_GdiView.SetBackColor( RGB(0,0,64) );
                                             // 设置默认的背景颜色
    m GdiView.SetDisplayImage( &m Image );
                                             // 设置当前显示的图像
    m_GdiView.SetActiveOverlay(&m_Overlay); // 设置当前显示的覆盖图ROI 图形显示。
    m_GdiView.SetStaticOverlay(&m_Results); // 设置当前显示的覆盖图结果图形显示。
                                              //将需要显示的图形添加到覆盖图容器中。
    m_Overlay.AddItem( &m_Rect );
}
// 执行测量
voidCCaliperDemoDlg::OnExecuct()
{
    // TODO: Add your control notification handler code here
    m_Caliper.SetPolarity1( m_Combol.GetCurSel() );
    m Caliper.SetPolarity2( m Combo2.GetCurSel() );
    m_Caliper.SetLocation( m_Combo3.GetCurSel() );
    m Caliper.SetThreshold(GetDlgItemInt(IDC EDIT1));
    m_Caliper.SetFilterHalf(GetDlgItemInt(IDC_EDIT2));
    BeginTime();
    // 执行测量
    m_Caliper.Execute( m_Image, m_Rect );
    EndTime();
    // 创建投影曲线
    m_Profile.Create( m_Caliper.GetLength() );
    m_Profile.SetCurve1( m_Caliper.GetProjection() );
    m_Profile.SetCurve2( m_Caliper.GetStrengths() );
    m_Profile.SetThreshold( double(GetDlgItemInt(IDC_EDIT1)) );
    // 建立坐标系变换
    CFrameTranstrans( m_Rect );
    intitem;
    CStringtext;
```

www.ckvision.net



```
CaliperData* data;
m List1.DeleteAllItems();
Overlay_DeleteAll(m_Results);// 清除已有的结果显示覆盖图形
for( inti=0; i<m_Caliper.GetCaliperCount(); i++ ) {</pre>
    data = m_Caliper.GetCaliperData( i );//获取数据
    text.Format( "%d", i+1 );
    item = m_List1.InsertItem( i, text );
    text.Format( "%0.2f", data->Width );
    m_List1.SetItemText( item, 1, text );
    text.Format( "%0.2f", data->Center.x );
    m List1.SetItemText( item, 2, text );
    text.Format( "%0.2f", data->Center.y );
    m List1.SetItemText( item, 3, text );
    m_Profile.AddEdge( data->Edge1.Distance );
    m Profile.AddEdge( data->Edge2.Distance );
    CGdiLine* p1 = newCGdiLine(
         data->Edge1.Distance, 0,
         data->Edge1.Distance,
         m_Rect.height );
    if( p1!=NULL ) {
         p1->Transform( &trans );
         p1->SetPenColor( RGB(0, 255, 0) );
         m_Results.AddItem( p1 );// 添加到结果显示图形表面
    }
    CGdiLine* p2 = newCGdiLine(
         data->Edge2.Distance, 0,
         data->Edge2.Distance,
         m_Rect.height );
    if (p2!=NULL) {
         p2->Transform( &trans );
         p2->SetPenColor( RGB(0,255,0) );
         m_Results.AddItem( p2 );
    }
    CGdiLine* p3 = newCGdiLine(
             data->Edge1.Position,
             data->Edge2.Position );
    if (p3!=NULL) {
         p3->SetPenColor( RGB(0, 255, 0) );
```



```
m_Results.AddItem(p3);
}
m_GdiView.Redraw();// 刷新显示视图
m_ProView.Redraw();
}
//.....详情请打开实例参考。
```

ColorMatchDemo 色彩匹配

```
在 StdAfx.h 的头文件中添加 CKVISION 相关定义
```

```
#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKColor.h"
```

```
//#include "..\\..\\Include\\CKColorIdentify.h" // 颜色识别
//#include "..\\..\\Include\\CKColorMonitor.h" // 颜色监测
```

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKColor.lib")
#else
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKColor.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKColor.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKColor.lib")
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。 // CColorMatchDemoApp initialization

BOOLCColorMatchDemoApp::InitInstance()

```
{
    CKVision::InitLibrary(); // 初始化 CKVision 库
    //...
```



```
//...End
CKVision::ExitLibrary(); // 退出 CKVision 库
```

```
}
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的.h 头文件中定义相应的图像处理功能: // ColorMatchDemoDlg.h : header file

CPrImage	m_Image; //	基础图像		
CGdiRect	m_Rect;	// 矩形区域ROI		
CGdiCircle	m_Roi;	// 圆形区域ROI		
COverlay	m_Overlay;	// 图像上层显示表面, 前显示的动态图形,主要用于ROI	显示。	
CGdiView	m_Display;	// 图形显示视图窗口		
CColorSamples m_Sample;// 颜色样本集合				
CColorIdentify m_Identify; // 颜色识别				

COverlay m_sOverlay; CGdiView m_sDisplay; CGdiHistogram m_sHistogram; //学习样本直方图显示

COverlay m_tOverlay; CGdiView m_tDisplay; CGdiHistogram m_tHistogram;//匹配样本直方图显示

在对话框窗口的.cpp 实现文件中添加相应的功能实现。 // ColorMatchDemoDlg.cpp : implementation file

// CColorMatchDemoDlg message handlers

BOOLCColorMatchDemoDlg::OnInitDialog()

// 颜色识别

voidCColorMatchDemoDlg::OnExecute()



```
{
    // TODO: Add your control notification handler code here
    doubledMatchScore = 0;
    m_Identify.SetSensitivity( m_Combol.GetCurSel() );
    m_Identify.SetSatThreshold( GetDlgItemInt(IDC_EDIT1) );
    BeginTime();
    // 颜色识别
    m_Identify.Execute( m_Image, m_Roi );
    // 识别当前颜色与颜色样本的区别
    m_Identify.Identify( m_Sample, dMatchScore );
    EndTime();
    CStringtext;
    text.Format( "%f", dMatchScore );
    SetDlgItemText( IDC_DATA1, text );
    m_tHistogram.SetValues(
        m Identify.GetColorValues(),
        m_Identify.GetColorCount() );
    m_tDisplay.Redraw();//识别样本直方图刷新显示
}
//.....详情请打开实例参考。
ColorThresholdDemo 彩色二值化
```

```
在 StdAfx.h 的头文件中添加 CKVISION 相关定义
#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKColor.h"
//#include "..\\..\\Include\\CKHSIThreshold.h" // 颜色抽取
```

```
#ifdef_WIN64
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
```



深圳市创科自动化控制技术有限公司

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKColor.lib")
#else
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKColor.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKColor.lib")
```

```
usingnamespaceCKVision;
```

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
// ColorThresholdDemo.cpp : Defines the class behaviors for the application.
BOOLCColorThresholdDemoApp::InitInstance()
```

```
{
    CKVision::InitLibrary(); // 初始化 CKVision 库
    //...
    //...End
```

```
CKVision::ExitLibrary();  // 退出 CKVision 库
```

}

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

```
// ColorThresholdDemoDlg.h : header file
    CPrImage m_Image; // 基础图像
    CPrImage m_Result; // 处理后的图像
    CGdiView m_GdiView; // 图形显示视图窗口
    CHSIThreshold m_hsiThre; // HSI 颜色抽取
```

在对话框窗口的.cpp 实现文件中添加相应的功能实现。

```
// ColorThresholdDemoDlg.cpp : implementation file
// CColorThresholdDemoDlg message handlers
B00LCColorThresholdDemoDlg::OnInitDialog()
{
```

// TODO: Add extra initialization here

RECTrect;



深圳市创科自动化控制技术有限公司

```
GetDlgItem(IDC_VIEW_RECT)->GetWindowRect( &rect );
    ScreenToClient( &rect );
    m_GdiView.Create( m_hWnd, rect );
                                             // 按自定义的区域创建图形显示视图窗口。
    m_GdiView.SetBackColor( RGB(0,0,64) );
                                             // 设置默认的背景颜色
    m_GdiView.SetDisplayImage( &m_Image );
                                             // 设置当前显示的图像
}
// 执行彩色二值化
voidCColorThresholdDemoDlg::OnExecute()
{
    /\!/ TODO: Add your control notification handler code here
    m hsiThre.SetHueRange( m Slider1.GetPos(), m Slider2.GetPos() );
    m_hsiThre.SetSaturationRange( m_Slider3.GetPos(), m_Slider4.GetPos() );
    m_hsiThre.SetIntensityRange( m_Slider5.GetPos(), m_Slider6.GetPos() );
    BeginTime();
    // 执行颜色抽取(彩色二值化)
    m_hsiThre.Execute( m_Image, m_Result, MaxROI );
    EndTime();
    m_GdiView.SetDisplayImage(&m_Result);// 设置当前显示的图像为处理后的图像
    m GdiView. Redraw();// 刷新显示视图
}
//.....详情请打开实例参考。
```

ContourDemo 轮廓提取

在 StdAfx.h 的头文件中添加 CKVISION 相关定义

```
// 添加自定义删除图形消息
#defineWM_DELETE_FIGURES WM_USER+123
```

```
#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKContour.h"
```

```
#ifdef_WIN64
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
```



深圳市创科自动化控制技术有限公司

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKContour.lib")
#else
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKContour.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKContour.lib")
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。 // CContourDemoApp initialization

```
BOOLCContourDemoApp::InitInstance()
```

```
{
    CKVision::InitLibrary(); // 初始化 CKVision 库
    //...
    //...End
    CKVision::ExitLibrary(); // 退出 CKVision 库
}
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

```
在对话框窗口的 .h 头文件中定义相应的图像处理功能:
             m Image; // 基础图像
   CPrImage
   CContourDetect
                 m_Contour; // 轮廓检测
   CGdiRect
             m_Rect;
                       // 矩形ROI 检测区域
                       // 图像显示表面, 前显示的动态图形, 主要用于ROI 显示。
   COverlay
             m_Overlay;
                        // 图像显示表面, 前显示的静态图形, 主要用于检测结果生成图形
   COverlay
             m_Results;
显示。
                       // 图形显示视图窗口
   CGdiView
             m GdiView;
   // 添加自定义消息删除图形
   afx_msgLRESULTOnDeleteFigures( WPARAMwParam, LPARAM1Param );
```

在对话框窗口的.cpp 实现文件中添加相应的功能实现。

```
//{{AFX_MSG_MAP(CContourDemoDlg)
```

```
ON_MESSAGE( WM_DELETE_FIGURES, OnDeleteFigures )
```



// 以消息的方式删除图形

```
LRESULTCContourDemoDlg::OnDeleteFigures( WPARAMwParam, LPARAM1Param )
{
    if( m_Results.GetCount() )
    {
         for( inti=0; i<m_Results.GetCount(); i++ )</pre>
         {
             deletem_Results[i];
         m_Results.RemoveAll();
    }
    return OL;
}
// 执行轮廓检测
voidCContourDemoDlg::OnExecute()
{
    /\!/ TODO: Add your control notification handler code here
    m Contour.SetThreshold( GetDlgItemInt(IDC EDIT1) );
    m_Contour.SetMinLength( GetDlgItemInt(IDC_EDIT2) );
    m_Contour.SetMaxLength( GetDlgItemInt(IDC_EDIT3) );
    m_Contour.SetFilterSize( m_Combol.GetCurSel() );
    m Contour.SetSubPixel( m SubPixel.GetCheck() );
    BeginTime();
    if( m_Rect.GetVisible() )
         m_Contour.Execute(m_Image, m_Rect);// 执行轮廓检测
    else
         m_Contour.Execute( m_Image, MaxROI );
    EndTime();
    CStringtext;
    MPNT* data=NULL;
    m_List1. DeleteAllItems();
    //Overlay_DeleteAll(m_Results);// 删除所有图形,请注意在线程中调用清除图形时,最好使用发
送消息的方式。
    SendMessage(WM_DELETE_FIGURES, 0, 0); // 发送消息删除图形
    if( m_Contour.GetPointCount()>0 ) {
```



```
intnum, n=0, i=0;
    while (i \ge 0) {
         data = m_Contour.GetPointData(i);
         i = m_Contour.GetContourId(i, num);
         text.Format( "%d", n );
         m_List1.InsertItem( n, text );
         text.Format( "%d", num );
         m_List1.SetItemText( n, 1, text );
         text = (data->m&MCP CLOSE) ? "是" : "否";
         m_List1.SetItemText( n, 2, text );
         n++;
    }
    // 根据结果数据添加到覆盖图显示
    CGdiContour* pp = newCGdiContour(
         m_Contour.GetPointData(0),
         m Contour.GetPointCount() );
    if( pp!=NULL ) {
         pp->Offset( 0.5, 0.5 );
         pp->SetPenWidth( 2 );
         pp->SetPenColor(RGB(0, 200, 0));
         m_Results.AddItem(pp);
    }
}
// 刷新视图显示
m_GdiView.Redraw();
```

```
//.....详情请打开实例参考。
```

}

DataMatrixDemo 二维码读取(DM 码)

在 StdAfx.h 的头文件中添加 CKVISION 相关定义



```
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKDataMatrix.h"
```

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKReader.lib")
#else
#pragmacomment(lib, "..\\..\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\Lib\\CKReader.lib")
#pragmacomment(lib, "..\\..\Lib\\CKReader.lib")
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库 //...
```

```
//...End
CKVision::ExitLibrary(); // 退出 CKVision 库
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

```
CPrImage
             m_Image; // 基础图像
   CGdiRect
             m_Rect; // 矩形ROI 检测区域
   COverlay
                        // 图像显示表面, 前显示的动态图形, 主要用于ROI 显示。
             m_Overlay;
   COverlay
             m Results; // 图像显示表面, 前显示的静态图形, 主要用于检测结果生成图形
显示。
   CGdiView
                        // 图形显示视图窗口
             m_GdiView;
   CDataMatrix
                           // 读取 DataMatrix 二维码
                 m_dm;
```

```
在对话框窗口的.cpp 实现文件中添加相应的功能实现。
```

```
// 执行读取DataMatrix 二维码
voidCDataMatrixDemoDlg::OnExecute()
```

```
{
```



// TODO: Add your control notification handler code here
Overlay_DeleteAll(m_Results);

```
intoption = 0;
```

```
if( IsDlgButtonChecked(IDC_CHECK2) )
        option |= 0x20;
```

if(IsDlgButtonChecked(IDC_CHECK3))
 option |= 0x01;

```
m_dm.SetFilterLevel(m_Combol.GetCurSel());
m_dm.SetPolarity(m_Combo2.GetCurSel());
m_dm.SetShapeType(m_Combo3.GetCurSel());
m_dm.SetOptions(option);
```

```
m_dm.SetMaxCount(GetDlgItemInt(IDC_EDIT5));
m_dm.SetThreshold(m_Combo4.GetCurSel());
m_dm.SetDefectLen(GetDlgItemInt(IDC_EDIT8));
```

```
m_dm.SetCodeWidth(GetDlgItemInt(IDC_EDIT3));
m_dm.SetCodeHeight(GetDlgItemInt(IDC_EDIT4));
```

```
m_dm.SetNumCellX(GetDlgItemInt(IDC_EDIT6));
m_dm.SetNumCellY(GetDlgItemInt(IDC_EDIT7));
```

```
BeginTime();
```

```
if( m_Rect.GetVisible() )
{
    m_dm.Execute( m_Image, m_Rect );// 执行读取DataMatrix 二维码
}
else
{
    m_dm.Execute( m_Image, MaxROI );
}
EndTime();
```

CStringtext;



```
DataMatrixResult* data;
m List1.DeleteAllItems();
for( inti=0; i<m_dm.GetResultCount(); i++ )</pre>
{
     data = m_dm.GetResultCode(i);
     text.Format( "%d", i+1 );
     m List1.InsertItem( i, text );
     text.Format( "%0.2f", data->border[0].x );
     m_List1.SetItemText( i, 1, text );
     text.Format( "%0.2f", data->border[0].y );
     m_List1.SetItemText( i, 2, text );
     m_List1.SetItemText( i, 3, data->codeText );
     CGdiPolygon* p1 = newCGdiPolygon;
     if( p1!=NULL )
     {
          p1 \rightarrow SetMax(4);
          p1->Add(data->border[0]);
          p1->Add(data->border[1]);
          p1 \rightarrow Add(data \rightarrow border[2]);
          p1->Add(data->border[3]);
          if( data->codeLen>0 )
          {
               p1->SetPenColor(RGB(0,255,0));
          }
          else
          {
               p1->SetPenColor(RGB(255,0,0));
          }
          p1->SetPenWidth(2);
          m_Results.AddItem(p1);// 添加显示图形
     }
}
```

m_GdiView.Redraw(); // 刷新视图,显示图形

}

```
//.....详情请打开实例参考。
```



EdgeToolDemo 边缘点检测

在 StdAfx.h 的头文件中添加 CKVISION 相关定义

```
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKEdgeTool.h"
```

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#else
#pragmacomment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKMeasure.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKMeasure.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKMeasure.lib")
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库
//...
```

//...End

```
CKVision::ExitLibrary(); // 退出 CKVision 库
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

CPrImage	m_Image; // 基	長础	图像	
CGdiRotBox	m_Rect;		// 旋转矩形R	10
CEdgeTool	m_EdgeTool;	//	边缘点检测	
COverlay	m_Overlay;	//	图像显示表面,	前显示的动态图形,主要用于ROI 显示。
COverlay	m_Results;	//	图像显示表面,	前显示的静态图形,主要用于检测结果生成图形



{

```
显示。
    CGdiView
                 m GdiView;
                              // 图形显示视图窗口
    CGdiProfile
                                  // 投影曲线边缘位置显示
                     m_Profile;
    COverlay
                              // 曲线图形显示
                 m ProOver;
    CGdiView
                 m_ProView;
                              // 图形显示视图窗口
在对话框窗口的.cpp 实现文件中添加相应的功能实现。
// 执行边缘点检测
voidCEdgeToolDemoDlg::OnExecute()
    // TODO: Add your control notification handler code here
    m_EdgeTool.SetPolarity( m_Combol.GetCurSel() );
    m EdgeTool.SetLocation( m Combo2.GetCurSel() );
    m_EdgeTool.SetThreshold(GetDlgItemInt(IDC_EDIT1));
    m_EdgeTool.SetFilterHalf(GetDlgItemInt(IDC_EDIT2));
    BeginTime();
    m_EdgeTool.Execute( m_Image, m_Rect ); // 执行区域内的边缘点检测
    EndTime():
    // 创建曲线显示视图窗口
    m Profile.Create( m EdgeTool.GetLength() );
    m_Profile.SetCurve1( m_EdgeTool.GetProjection() );
    m Profile.SetCurve2( m EdgeTool.GetStrengths() );
    m_Profile.SetThreshold( double(GetDlgItemInt(IDC_EDIT1)) );
    intitem;
    CStringtext;
    EdgeData* data;
    m_List1. DeleteAllItems();
    Overlay_DeleteAll(m_Results);
    for( inti=0; i<m_EdgeTool.GetEdgeCount(); i++ ) {</pre>
        data = m_EdgeTool.GetEdgeData( i );
        text.Format( "%d", i+1 );
        item = m_List1.InsertItem( i, text );
        text.Format( "%d", data->Polarity );
        m_List1.SetItemText( item, 1, text );
        text.Format( "%0.3f", data->Strength );
        m List1.SetItemText( item, 2, text );
        text.Format( "%0.3f", data->Position.x );
```

```
text.Format( "%0.3f", data->Position.y );
```



```
m_List1.SetItemText( item, 4, text );
    m Profile.AddEdge( data->Distance );
    // 把结果数据添加到覆盖图显示
    CGdiPoint* p = newCGdiPoint(
        data->Position.x+0.5,
        data->Position.y+0.5 );
    if( p!=NULL ) {
        p->SetSize( 3 );
        p->SetStyle( 1 );
        p->SetPenColor( RGB(0, 255, 0) );
        m Results.AddItem( p );
    }
}
m GdiView. Redraw(); // 刷新显示视图
m_ProView.Redraw();
                   // 刷新曲线视图
```

//.....详情请打开实例参考。

}

FitCircleDemo 圆形测量(拟合圆)

在 StdAfx.h 的头文件中添加 CKVISION 相关定义

#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKEdgeTool.h"
#include"..\\..\\Include\\CKFitCircle.h"

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKMeasure.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKMeasure.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKMeasure.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKMeasure.lib")
```



usingnamespaceCKVision;

```
在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。
```

```
CKVision::InitLibrary(); // 初始化 CKVision 库 //...
```

```
//...End
CKVision::ExitLibrary(); // 退出 CKVision 库
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

	CPrImage	m_Image; //	基础图像
	CEdgeTool	m_EdgeTool;	// 边缘点检测
	CFitCircle	m_Fit;	// 圆拟合工具
	CGdiRingScan	m_Ring;	// 圆环内扫描线ROI
	COverlay	m_Overlay;	// 图像显示表面, 前显示的动态图形,主要用于ROI 显示。
	COverlay	m_Results;	// 图像显示表面, 前显示的静态图形,主要用于检测结果生成图形
显示	0		
	CGdiView	m_GdiView;	// 图形显示视图窗口
	COverlay	m_ProOver;	// 曲线图形显示
	CGdiView	m_ProView;	// 图形显示视图窗口

在对话框窗口的.cpp 实现文件中添加相应的功能实现。

```
// 圆形测量
voidCFitCircleDemoDlg::OnExecute()
{
    // TODO: Add your control notification handler code here
    UpdateData( TRUE );
    Overlay_DeleteAll(m_Results);// 清除原来的显示
    Overlay_DeleteAll(m_ProOver);// 清除原来的显示
    // 边缘点检测设置参数
```

```
m_EdgeTool.SetPolarity( m_Combol.GetCurSel() );
m_EdgeTool.SetLocation( m_Combo2.GetCurSel() );
```



```
m_EdgeTool.SetThreshold(GetDlgItemInt(IDC_EDIT1));
m_EdgeTool.SetFilterHalf(GetDlgItemInt(IDC_EDIT2));
//拟合圆设置容忍误差
m_Fit.SetTolerance( m_dTolerate );
BeginTime();
intnum=0;
ROTRECTrc;
EdgeData* data;
// 圆环内扫描线设置扫描参数
m_Ring.SetScanCount(GetDlgItemInt(IDC_EDIT3));
m Ring.SetScanWidth( GetDlgItemInt(IDC EDIT4) );
// 坐标点容器
CPointVector(m_Ring.GetScanCount());
for( inti=0; i<m_Ring.GetScanCount(); i++ ) {</pre>
    m Ring.GetScanRoi( i, rc );
    m_EdgeTool.Execute( m_Image, rc );// 边缘点检测
    CGdiProfile* pProfile = new
         CGdiProfile(m EdgeTool.GetLength());
    data = m_EdgeTool.GetEdgeData(0);// 获取边缘点数据
    if( data != NULL ) {
         ptVector.Add(data->Position);// 把点添加到容器中
        CGdiPoint* pp = new
             CGdiPoint(data->Position);
         if( pp != NULL ) {
             pp->Offset( 0.5, 0.5 );
             pp->SetStyle( 1 );
             pp->SetPenColor( RGB(255, 255, 0) );
             m_Results.AddItem(pp);//显示扫描边缘点
         if( pProfile!=NULL ) {
             pProfile->AddEdge(data->Distance);
         }
    }
    if( pProfile!=NULL ) {
```



```
pProfile->SetCurve1( m_EdgeTool.GetProjection() );
         pProfile->SetCurve2( m_EdgeTool.GetStrengths() );
         pProfile->SetThreshold( m EdgeTool.GetThreshold() );
         pProfile->SetVisible( FALSE );
         m_ProOver.AddItem( pProfile );// 曲线图显示
    }
}
// 执行拟合圆形
if( m_Fit.Execute( ptVector ) ) {
    for( intn=0; n<m_Results.GetCount(); n++ ) {</pre>
         if( m_{Fit.GetUse(n) == false}) {
              m Results[n]->SetPenColor(RGB(255,0,0));
         }
    }
    CGdiCircle* p = newCGdiCircle;
    if (p != NULL) {
         p->radius = m_Fit.GetRadius();
         p->center.x = m_Fit.GetCenterX();
         p->center.y = m Fit.GetCenterY();
         p->0ffset(0.5, 0.5);
         p->SetPenColor(RGB(0, 255, 0));
         m_Results.AddItem( p );// 显示圆
    }
}
CStringtext;
text.Format( "半径(R): %0.2f", m Fit.GetRadius());
SetDlgItemText( IDC_DATA1, text );
text.Format( "中心(X): %0.2f", m_Fit.GetCenterX() );
SetDlgItemText( IDC_DATA2, text );
text.Format( "中心(Y): %0.2f", m_Fit.GetCenterY() );
SetDlgItemText( IDC_DATA3, text );
text.Format( "拟合误差: %0.2f", m_Fit.GetRMSError() );
SetDlgItemText( IDC_DATA4, text );
```

EndTime();



```
if( m_ProOver.GetCount()>0 )
    m_ProOver[0]->SetVisible( TRUE );
SetDlgItemInt( IDC_EDIT5, 0 );
m_GdiView.Redraw();// 刷新显示视图
```

```
m_ProView. Redraw();// 刷新显示曲线视图
```

```
}
```

```
//.....详情请打开实例参考。
```

FitLineDemo 直线测量(拟合直线)

```
在 StdAfx.h 的头文件中添加 CKVISION 相关定义
```

```
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKEdgeTool.h"
```

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKMeasure.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKMeasure.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKMeasure.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKMeasure.lib")
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库
//...
//...End
CKVision::ExitLibrary(); // 退出 CKVision 库
```



在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

```
CPrImage
                         // 基础图像
              m_Image;
   CEdgeTool
                            // 边缘点检测
              m EdgeTool;
   CFitLine
              m Fit;
                            // 直线拟合工具
   CGdiBoxScan
                 m Rect;
                                // 旋转矩形框内扫描线ROI
                            // 图像显示表面, 前显示的动态图形, 主要用于ROI 显示。
   COverlay
              m_Results;
                            // 图像显示表面, 前显示的静态图形,主要用于检测结果生成
   COverlay
              m_Overlay;
图形显示。
                            // 图形显示视图窗口
   CGdiView
              m_GdiView;
   COverlay
              m CurveOver; // 曲线图形显示
              m_CurveView; // 图形显示视图窗口
   CGdiView
```

在对话框窗口的.cpp 实现文件中添加相应的功能实现。

// 拟合直线 voidCFitLineDemoDlg::OnExecute() { // TODO: Add your control notification handler code here Overlay DeleteAll(m Results); // 清除原来的显示 Overlay_DeleteAll(m_CurveOver); // 清除原来的显示 // 边缘点检测设置参数 m_EdgeTool.SetPolarity(m_Combol.GetCurSel()); m_EdgeTool.SetLocation(m_Combo2.GetCurSel()); m EdgeTool.SetThreshold(GetDlgItemInt(IDC EDIT1)); m_EdgeTool.SetFilterHalf(GetDlgItemInt(IDC_EDIT2)); BeginTime(); intnum=0; ROTRECTrc: EdgeData* data; //旋转矩形框内扫描线设置

```
m_Rect.SetScanCount(GetDlgItemInt(IDC_EDIT3));
m_Rect.SetScanWidth(GetDlgItemInt(IDC_EDIT4));
// 坐标点容器
```

```
CPointVectorptVector(m_Rect.GetScanCount());
```



深圳市创科自动化控制技术有限公司

```
for( inti=0; i<m_Rect.GetScanCount(); i++ ) {</pre>
    m Rect.GetScanRoi( i, rc );
    m EdgeTool.Execute(m Image, rc);// 边缘点检测
    CGdiProfile* pProfile = new
        CGdiProfile(m EdgeTool.GetLength()); //创建投影曲线边缘位置
    data = m_EdgeTool.GetEdgeData(0);
    if( data != NULL ) {
        ptVector.Add( data->Position ); // 把点添加到容器中
        CGdiPoint* pp = new
             CGdiPoint(data->Position);
        if( pp != NULL ) {
             pp->SetStyle( 1 );
             pp->SetPenColor( RGB(0, 255, 255) );
             m Results.AddItem( pp );
        }
        if( pProfile!=NULL ) {
             pProfile->AddEdge( data->Distance );
        }
    }
    if( pProfile!=NULL ) {
        pProfile->SetCurve1(m EdgeTool.GetProjection()); // 设置曲线投影数据
        pProfile->SetCurve2(m_EdgeTool.GetStrengths()); // 设置曲线梯度数据
        pProfile->SetThreshold( m EdgeTool.GetThreshold() );
        pProfile->SetVisible( FALSE );
        m_CurveOver.AddItem( pProfile ); // 添加曲线到显示容器
    }
}
CStringtext;
GetDlgItemText( IDC_EDIT5, text );
// 拟合直线容忍误差设置
m_Fit.SetTolerance( atof(text) );
//执行拟合直线
if( m_Fit.Execute( ptVector ) ) {
    for( intn=0; n<m_Results.GetCount(); n++ ) {
         if(m_Fit.GetUse(n)==false) { // 不参与拟合的点
             m_Results[n]->SetPenColor(RGB(255,0,0)); // 设置红色显示
        }
    }
    DLINEline;
```

www.ckvision.net



}

}

```
m_Fit.GetLine(9999,line);
     CGdiLine* pp = newCGdiLine(line);
     if( pp != NULL ) {
         pp->Offset( 0.5, 0.5 );
         pp->SetPenColor( RGB(0, 255, 0) );
         m_Results.AddItem( pp );
    }
     text.Format( "%0.2f", m_Fit.GetRMSError() );
     SetDlgItemText( IDC_RMSE, text );
     text.Format( "%0.2f", m_Fit.GetSampling() );
     SetDlgItemText( IDC_SAMPL, text );
     text.Format( "%0.2f", m_Fit.GetAngle() );
     SetDlgItemText( IDC_ANGLE, text );
     text.Format( "%0.2f", m_Fit.GetCenterX() );
     SetDlgItemText( IDC_CENTER_X, text );
     text.Format( "%0.2f", m_Fit.GetCenterY() );
     SetDlgItemText( IDC_CENTER_Y, text );
} else {
     SetDlgItemText( IDC_RMSE, "" );
     SetDlgItemText( IDC_ANGLE, "" );
     SetDlgItemText( IDC LENGTH, "" );
     SetDlgItemText( IDC_CENTER_X, "" );
    SetDlgItemText( IDC_CENTER_Y, "" );
EndTime();
if( m_CurveOver.GetCount()>0 )
     m_CurveOver[0]->SetVisible( TRUE );
SetDlgItemInt( IDC_EDIT6, 0 );
//刷新显示
m_GdiView.Redraw();
m_CurveView.Redraw();
```



//.....详情请打开实例参考。

HistogramDemo 灰度直方图、自动二值化阈值

在 StdAfx.h 的头文件中添加 CKVISION 相关定义

#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKBase.h"

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#else
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKBase.lib")
#endif
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库 //...
```

```
CKVision::ExitLibrary(); // 退出 CKVision 库
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

CPrImage	m_Image; // 基	基础图像
CGdiRect	m_Rect;	// 矩形区域
CHistogram	m_Histogram;	// 直方图功能
COverlay	m_Overlay;	// 图像显示表面, 前显示的动态图形,主要用于ROI 显示。
CGdiView	m_GdiView;	// 图形显示视图窗口
COverlay	m_HistOver;	// 直方图显示图 覆盖图
CGdiView	m HistView;	// 图形显示视图窗口

106 / 138

www.ckvision.net



CGdiHistogram m_GdiHist; // 直方图显示数据

在对话框窗口的.cpp 实现文件中添加相应的功能实现。

// 执行直方图分析

{

voidCHistogramDemoDlg::OnExecute()

// TODO: Add your control notification handler code here

```
// 设置分析功能
```

m_Histogram.SetAnalyse(Analyse_Min_Max|Analyse_Mean_StdDev);

```
BeginTime();
```

```
if( m_Rect.GetVisible() )
```

m_Histogram.Execute(m_Image, m_Rect); // 执行直方图分析

```
else
```

}

m_Histogram.Execute(m_Image, MaxROI); EndTime();

```
CStringtext;
```

```
SetDlgItemInt( IDC_DATA1, m_Histogram.GetMin() );
SetDlgItemInt( IDC_DATA2, m_Histogram.GetMax() );
text.Format( "%0.2f", m_Histogram.GetMean() );
SetDlgItemText( IDC_DATA3, text );
text.Format( "%0.2f", m_Histogram.GetStdDev() );
SetDlgItemText( IDC_DATA4, text );
```

```
intitem;
intnLen = m_Histogram.GetLength();
int* pValues = m_Histogram.GetValues();
m_List1.DeleteAllItems();
for( inti=0; i<nLen; i++ ) {
    item = m_List1.InsertItem( i, "" );
    text.Format( "%d", i );
    m_List1.SetItemText( item, 0, text );
    text.Format( "%d", pValues[i] );
    m_List1.SetItemText( item, 1, text );
}
m_GdiHist.SetValues( pValues, nLen ); // 设置直方图数据
m_HistView.Redraw();// 直方图刷新显示
```



//.....详情请打开实例参考。

ImageDemo 图像预处理

在 StdAfx.h 的头文件中添加 CKVISION 相关定义

#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKGDI.h"

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
#else
#pragmacomment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#endif
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库
//...
```

```
CKVision::ExitLibrary(); // 退出 CKVision 库
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

CPrImage	m_Image; //	基础图像
CPrImage	<pre>m_Result; //</pre>	处理后的结果图像
CGdiView	m_GdiView;	// 图形显示视图窗□

在对话框窗口的.cpp 实现文件中添加相应的功能实现。


// 执行

{

```
voidCImageDemoDlg::OnExecute()
    /\!/ TODO: Add your control notification handler code here
    UpdateData( TRUE );
    BeginTime();
    switch( m_Combol.GetCurSel() )
    {
    case 0:
         ImgSmooth( m_Image, m_Result ); // 平滑
         break;
    case 1:
         ImgSharp( m_Image, m_Result ); // 锐化
         break;
    case 2:
         ImgSobel( m_Image, m_Result ); // Sobel边缘
         break;
    case 3:
         ImgErode( m_Image, m_Result );// 腐蚀
         break;
    case 4:
         ImgDilate( m_Image, m_Result );// 膨胀
         break;
    }
    EndTime();
    // 把处理结果图复制到当前图像
    m_Image.Copy( m_Result );
    m_GdiView.Redraw();
```

//.....详情请打开实例参考。

}

ImageWarpDemo 环形展开裁剪图像

```
在 StdAfx.h 的头文件中添加 CKVISION 相关定义
#include"..\\..\\Include\\CKBase.h"
```



#include"..\\..\\Include\\CKGDI.h"

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
#else
#pragmacomment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#endif
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库
//...
//...End
CKVision::ExitLibrary(); // 退出 CKVision 库
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

CPrImage	m_Image; //	基础图像
CPrImage	<pre>m_Result; // ;</pre>	处理后的结果图像
CGdiRing	m_Roi;	// 圆环图形ROI
COverlay	m_Overlay;	// 图像显示表面, 前显示的动态图形,主要用于ROI 显示。
CGdiView	m_GdiView1;	// 图形显示视图窗口
CGdiView	m_GdiView2;	// 图形显示视图窗口

在对话框窗口的.cpp 实现文件中添加相应的功能实现。

// 执行

{

voidCImageWarpDemoDlg::OnExecute()

```
// TODO: Add your control notification handler code here
BeginTime();
```

// 环形区域展开,把来源图像按ROI 裁剪,返回结果图像



```
ImgRingWarp(m_Image, m_Result, m_Roi, m_Check1.GetCheck());
EndTime();
m_GdiView2.FitSize();// 图像显示窗口自动适应显示。
m_GdiView2.Redraw(); // 刷新显示视图窗口。
}
//.....详情请打开实例参考。
```

ImgTransDemo 图形变换(镜像、平移、旋转、缩放、仿射)

在 StdAfx.h 的头文件中添加 CKVISION 相关定义

```
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKGDI.h"
```

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#else
#pragmacomment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#endif
```

```
usingnamespaceCKVision;
```

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库 //...
```

```
//...End
CKVision::ExitLibrary(); // 退出 CKVision 库
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:



```
m_Image; // 基础图像
    CPrImage
                 m_Result; // 处理结果图像
    CPrImage
    CGdiView
                 m_GdiView;
                             // 图形显示视图窗口
    int
                     m nFunSel;
                                  // 功能选择
                     m_nMirFlag;
                                  // 镜像方式
    int
在对话框窗口的.cpp 实现文件中添加相应的功能实现。
// 执行
voidCImgTransDemoDlg::OnExecute()
{
    // TODO: Add your control notification handler code here
    UpdateData(TRUE);
    BeginTime();
    switch( m nFunSel ) {
    case 0:
         {
             m_nMirFlag=0;
             if( m bMir1==TRUE )
                 m_nMirFlag |= MirrorHor;
             if( m_bMir2==TRUE )
                 m_nMirFlag |= MirrorVer;
             if( m_bMir3==TRUE )
                 m nMirFlag |= MirrorRot;
             ImgMirror(m_Image, m_Result, m_nMirFlag);// 镜像
        }
        break;
    case 1:
         {
             ImgShift( m_Image, m_Result, m_dShiftX, m_dShiftY, m_bInpl );// 平移
        }
        break;
    case 2:
         {
             ImgScale( m_Image, m_Result, m_dScaleX, m_dScaleY, m_bInpl );// 缩放
        }
        break;
```



```
case 3:
     {
         ImgRotate(m_Image, m_Result, m_dAngle, m_bInpl);// 旋转
    }
    break;
case 4:
     {
         doublematrix[6];
         doublerd = m_dAngle*PI/180;
         doubleca = \cos(rd);
         doublesa = sin(rd);
         matrix[0] = ca*m_dScaleX;
         matrix[1] = sa*m_dScaleY;
         matrix[2] = m_dShiftX*m_dScaleX;
         matrix[3] = -sa*m_dScaleX;
         matrix[4] = ca*m_dScaleY;
         matrix[5] = m_dShiftY*m_dScaleY;
         ImgAffine( m_Image, m_Result, matrix, m_bInpl );// 仿射
    }
    break;
}
EndTime();
m_Combol.SetCurSel( 1 );
m_GdiView.SetDisplayImage(&m_Result);// 显示结果图像
m_GdiView.FitSize();
                                  // 适应显示
                                         // 刷新显示
m_GdiView.Redraw();
```

//.....详情请打开实例参考。

}

InspectDemo 基于图像对比缺陷检测

```
在 StdAfx.h 的头文件中添加 CKVISION 相关定义
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKGDI.h"
```



```
#include"..\\..\\Include\\CKBlob.h"
#include"..\\..\\Include\\CKFindModel.h"
```

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKBlob.lib")
#pragma comment(lib, "..\\..\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\Lib\\CKBlob.lib")
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库 //...
```

```
//...End
CKVision::ExitLibrary(); // 退出 CKVision 库
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

CPrImage	m_Image; //	基础图像
CPrImage	m_ImgDst; //	目标图像
CModel	m_Model;	// 定位模板
CFindModel	m_Find;	//形状模型搜索
CPatInspect	m_Inspect;	// 基于图像对比缺陷检测
CBlobAnalyzer	m_Blob;	// 斑点分析
CGdiRect	m_Rect;	// 矩形检测区域
COverlay	m_Overlay;	// 图像显示表面, 前显示的动态图形,主要用于ROI 显示。

114 / 138

www.ckvision.net



COverlay	m_Results;	//	图像显示表面,	前显示的静态图形,主要用于检测结果生成	
图形显示。					
CGdiView	m_GdiView;	//	图形显示视图窗	a 🗆	
CGdiView	m SubView[4]: //	图形	显示视图窗口		

在对话框窗口的.cpp 实现文件中添加相应的功能实现。

// 执行

{

voidCInspectDemoDlg::OnExecute()

// TODO: Add your control notification handler code here
Overlay_DeleteAll(m_Results);

// 斑点分析参数设置

m_Blob.SetBlobType(BLOB_WHITE); m_Blob.SetConnexity(CONNEXITY_8); m_Blob.SetFeatures(BLOB_FEATURE_BASE|BLOB_FEATURE_AXIS); m_Blob.SetThreshold(GetDlgItemInt(IDC_THRE)); m_Blob.SetLimitArea(GetDlgItemInt(IDC_AREA));

// 对比参数设置

m_Inspect.SetMaskEnabled(m_bMask.GetCheck()); m_Inspect.SetEqualizeMode(m_Combol.GetCurSel()); m_Inspect.SetDefectType(m_Combo2.GetCurSel());

BeginTime();

// 模板定位参数设置

m_Find.SetMinScore(60); m_Find.SetThreshold(m_Model.GetThreshold()>>1); // 执行识别定位 m_Find.Execute(m_Image, m_Model, MaxROI);

FindResult* pData = m_Find.GetResultData(0);
if(pData!=NULL) {

ROTRECTrc; CPrImagetempl;

rc.angle = pData->Angle; rc.center = pData->Center; rc.width = pData->Width;



```
rc.height = pData->Height;
    templ.Cut( m_Image, rc );
                                           // 根据模板定位裁剪图像
    m_Inspect.Execute( templ, m_ImgDst ); // 进行图像对比
    m_Blob.Execute(m_ImgDst, MaxROI); // 分析对比后的图像
    CBlobData* data=NULL;
    CFrameTransframe( rc );
    for( inti=0; i<m_Blob.GetBlobCount(); i++ ) {</pre>
        data = m_Blob.GetBlobData(i);
        //椭圆图形功能
        CGdiEllipse* p1 = newCGdiEllipse(
             data->m dCenterMassX,
             data->m_dCenterMassY,
             data->m_dMajorAxis/2,
             data->m_dMinorAxis/2,
             data->m_dRotation );
        if( p1!=NULL ) {
             p1->Transform( &frame );
             p1 \rightarrow SetPenColor(RGB(255, 0, 0));
             m_Results.AddItem(p1); // 添加到显示覆盖图上
        }
    }
    /* 2D 旋转矩形功能*/
    CGdiRotBox* p2 = newCGdiRotBox(rc);
    if( p2!=NULL ) {
        p2->SetPenColor( RGB(0, 255, 0) );
        m_Results.AddItem( p2 );// 添加到显示覆盖图上
    }
} else {
    AfxMessageBox("没有搜索到目标!");
}
EndTime();
m_GdiView.Redraw();
                       //刷新显示
m_SubView[3].Redraw();
```

```
//.....详情请打开实例参考。
```

}



ModelDemo 模板轮廓匹配定位(老版本)

```
在 StdAfx.h 的头文件中添加 CKVISION 相关定义
#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKLocate.h"
```

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKLocate.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKLocate.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKLocate.lib")
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库
//...
//...End
CKVision::ExitLibrary(); // 退出 CKVision 库
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

CPrImage	m_Image;	// 基础图像
CModel	m_Model;	// 定位模板
CPrImage	m_ModImage;	// 模板图像
CFindModel	m_Find;	// 形状模型搜索
CGdiRect	m_lRoi;	// 学习ROI
CGdiRect	m sRoi;	// 搜索ROI



深圳市创科自动化控制技术有限公司

COverlay	m_Overlay;	// 图像显示表面, 前显示的动态图形,主要用于ROI 显示。	
COverlay	m_Results;	// 图像显示表面, 前显示的静态图形,主要用于检测结果生	成
图形显示。			
CGdiView	m_GdiView;	// 图形显示视图窗口	
CModelD1g	m_ModelDlg;	// 编辑定位模板	
CModVarD1g	m_ModVarDlg;	// 角度、比例设置	

在对话框窗口的.cpp 实现文件中添加相应的功能实现。

// 执行搜索

voidCModelDemoDlg::OnExecute()

{

// TODO: Add your control notification handler code here $\mbox{m_lRoi.SetVisible(false});$

// 删除所有图形,请注意在线程中调用清除图形时,最好使用发送消息的方式, // 可参考ContourDemo 中的CContourDemoDlg::OnExecute() 。

Overlay_DeleteAll(m_Results);

// 设置搜索参数

m_Find.SetMaxCount(GetDlgItemInt(IDC_EDIT1)); m_Find.SetMinScore(GetDlgItemInt(IDC_EDIT2)); m_Find.SetThreshold(GetDlgItemInt(IDC_EDIT3)); m_Find.SetPolarity(m_Combol.GetCurSel()); m_Find.SetCompressor(m_Combo2.GetCurSel()); m_Find.SetModResult(m_Check1.GetCheck());

BeginTime();

```
if(m_sRoi.GetVisible())
{
    m_Find.Execute(m_Image, m_Model, m_sRoi); // 执行搜索功能
}
else
{
    m_Find.Execute(m_Image, m_Model, MaxROI);
}
```

EndTime();



```
CStringtext;
FindResult* data=NULL;
m_List1.DeleteAllItems();
for( inti=0; i<m_Find.GetResultCount(); i++ )</pre>
{
     data = m_Find.GetResultData(i);// 获取匹配结果数据
     text.Format( _T("%d"), i+1 );
     m_List1.InsertItem( i, text );
     text.Format( _T(\%0.3f''), data->Score );
     m_List1.SetItemText( i, 1, text );
     text.Format( _T("%0.3f"), data->Center.x );
     m_List1.SetItemText( i, 2, text );
     text.Format( _T("%0.3f"), data->Center.y );
     m_List1.SetItemText( i, 3, text );
     text.Format( _T("\%0.3f"), data->Angle );
     m List1.SetItemText( i, 4, text );
     text.Format( _T("\%0.3f"), data->Scale );
     m_List1.SetItemText( i, 5, text );
     if( data->Model.count>0 ) {
         CGdiContour* p1 = new
              CGdiContour(data->Model);
         if( p1!=NULL ) {
              p1->0ffset(0.5,0.5);
              p1->SetPenWidth( 1 );
              p1->SetPenColor( RGB(0,255,0) );
              m_Results. AddItem(p1);// 添加模板轮廓显示
     }
     CGdiRotBox* p2 = newCGdiRotBox(
         data->Center.x,
         data->Center.y,
```



```
data->Width,
         data->Height,
         data->Angle );
    if( p2!=NULL ) {
         p2->0ffset(0.5,0.5);
         p2->SetPenColor( RGB(0,255,0) );
         m Results. AddItem(p2);// 添加旋转矩形ROI显示
    }
    CGdiPoint* p3 = new
         CGdiPoint(data->Position);
    if (p3!=NULL) {
         p3->0ffset(0.5,0.5);
         p3->SetSize(10);
         p3->SetStyle( 2 );
         p3->SetPenColor( RGB(0,255,0) );
         m_Results.AddItem(p3);// 添加旋转矩形ROI显示
    }
}
m GdiView. Redraw();// 刷新显示
```

}

//.....详情请打开实例参考。

MultiModelDemo 多轮廓匹配定位(新版本)

在 StdAfx.h 的头文件中添加 CKVISION 相关定义

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

在对话框窗口的.cpp 实现文件中添加相应的功能实现。



//.....详情请打开实例参考。

NCMatchDemo 灰度匹配定位

在 StdAfx.h 的头文件中添加 CKVISION 相关定义 #include"..\\..\\Include\\CKGDI.h" #include"..\\..\\Include\\CKBase.h" #include"..\\..\\Include\\CKLocate.h"

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKLocate.lib")
#else
#pragmacomment(lib, "..\\..\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\Lib\\CKLocate.lib")
#pragmacomment(lib, "..\\..\Lib\\CKLocate.lib")
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库
//...
//...End
CKVision::ExitLibrary(); // 退出 CKVision 库
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

CNCPat	m_Pat;	// 匹配模板
CNCMatch	m_Match;	// 基于灰度区域匹配功能
CPrImage	m_Image;	// 基础图像
CGdiRect	m_lRect;	// 学习ROI



深圳市创科自动化控制技术有限公司

	CGdiRect	m_sRect; /	// 摂	索	ROI	
	COverlay	m_Overlay;		//	图像显示表面,	前显示的动态图形,主要用于ROI 显示。
	COverlay	m_Results;		//	图像显示表面,	前显示的静态图形,主要用于检测结果生成
图形	显示。					
	CGdiView	m_GdiView;		//	图形显示视图窗	ÎП
	CPatternDlg	m_PatternD	lg;	//	编辑模板窗口	

在对话框窗口的.cpp 实现文件中添加相应的功能实现。

```
// 执行匹配
voidCNCMatchDemoDlg::OnSearch()
{
    // TODO: Add your control notification handler code here
    m_lRect.SetVisible( false );
                                   // 隐藏学习ROI
    // 设置匹配参数
    m_Match.SetMaxCount(GetDlgItemInt(IDC_EDIT1));
    m_Match.SetMinScore(GetDlgItemInt(IDC_EDIT2));
    m Match.SetSubPixel( m subPixel.GetCheck() );
    BeginTime();
    if( m_sRect.GetVisible() )
         m Match. Execute(m Image, m Pat, m sRect); //执行匹配功能
    else
         m_Match.Execute( m_Image, m_Pat, MaxROI );
    EndTime();
    intitem;
    CStringtext;
    constMatchData* data;
    m_List1. DeleteAllItems();
    for( intn=0; n<m_Results.GetCount(); n++ )</pre>
         deletem_Results[n];
    m_Results.RemoveAll();
    for( inti=0; i<m_Match.GetMatchCount(); i++ ) {</pre>
         data = m_Match.GetMatchData( i );
    // 添加数据到列表
         text.Format( "%d", i+1 );
         item = m_List1.InsertItem( i, text );
         text.Format( "%0.2f", data->Score );
```

m_List1.SetItemText(item, 1, text);



```
text.Format( "%0.2f", data->Position.x );
     m List1.SetItemText( item, 2, text );
     text.Format( "%0.2f", data->Position.y );
     m_List1.SetItemText( item, 3, text );
// 添加中心点
    CGdiPoint* p1 = new
         CGdiPoint(data->Position);
    if( p1!=NULL ) {
         p1 \rightarrow SetPenColor(RGB(0, 255, 0));
         p1->SetSize( 8 );
         p1->SetStyle( 2 );
         m Results.AddItem( p1 );
    }
// 添加矩形框
    CGdiRect* p2 = newCGdiRect;
     if (p2!=NULL) {
         p2->left = data->Center.x-data->Width/2+0.5;
         p2->top
                      = data->Center.y-data->Height/2+0.5;
         p2->right = data->Center.x+data->Width/2+0.5;
         p2->bottom = data->Center.y+data->Height/2+0.5;
         p2->SetPenColor( RGB(0, 255, 0) );
         m_Results.AddItem( p2 );
    }
// 添加标号
    CGdiText* p3 = newCGdiText;
     if( p3!=NULL ) {
         text.Format( "%d", i+1 );
         p3->SetText( text );
         p3->SetFontSize( 15 );
         p3->SetPenColor( RGB(0,255,0) );
         p3->SetPosition( p2->left+0.5, p2->top+0.5 );
         p3->SetSpace( 0, -15 );
         m_Results.AddItem( p3 );
    }
}
m_GdiView.Redraw();// 刷新显示
```

```
//.....详情请打开实例参考。
```

}



QRCodeDemo 二维码检测(QR 码)

在 StdAfx.h 的头文件中添加 CKVISION 相关定义

#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKReadQRCode.h"

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\Lib_x64\\CKReader.lib")
#pragmacomment(lib, "..\\..\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\Lib\\CKReader.lib")
#pragmacomment(lib, "..\\..\Lib\\CKReader.lib")
#pragmacomment(lib, "..\\..\Lib\\CKReader.lib")
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库
//...
```

CKVision::ExitLibrary(); // 退出 CKVision 库

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

```
在对话框窗口的 .h 头文件中定义相应的图像处理功能:
   CPrImage
             m_Image;
                      // 基础图像
                          // 搜索ROI
   CGdiRect
             m_Rect;
   COverlay
             m_Overlay;
                          // 图像显示表面, 前显示的动态图形, 主要用于ROI 显示。
                          // 图像显示表面, 前显示的静态图形, 主要用于检测结果生成
   COverlay
             m_Results;
图形显示。
   CGdiView
                          // 图形显示视图窗口
             m_GdiView;
   CReadQRCode
                m_QRCode;
                          // 二维码 QRCode 读取
```

www.ckvision.net



{

```
在对话框窗口的.cpp 实现文件中添加相应的功能实现。
// 执行
voidCQRCodeDemoDlg::OnBnClickedExecute()
   // TODO: 在此添加控件通知处理程序代码
   // 删除所有图形,请注意在线程中调用清除图形时,最好使用发送消息的方式,
    // 可参考ContourDemo 中的CContourDemoDlg::OnExecute() 。
   Overlay_DeleteAll(m_Results);
   BOOLisCheck1 = m_Check1.GetCheck();
    if(isCheck1) {
        AutoThreshold();
    }
    intnMaxCount = GetDlgItemInt(IDC_EDIT1);
    intnPos = m Slider1.GetPos();
    // 设置读取参数
    m_QRCode.SetMaxCount(nMaxCount);
    m_QRCode.SetThreshold(nPos);
    m_QRCode.SetPolarity(m_Combol.GetCurSel());
    m_QRCode.SetMinArea(GetDlgItemInt(IDC_EDIT2));
    m_QRCode.SetMaxArea(GetDlgItemInt(IDC_EDIT3));
   BeginTime();
    if (m_Rect.GetVisible()) {
        m_QRCode.Execute(m_Image, m_Rect); // // 执行读取二维码功能
    } else {
        m_QRCode.Execute(m_Image, MaxROI);
    }
    EndTime();
    m_List1.DeleteAllItems();
    for (inti=0; i<m_QRCode.GetResultCount(); i++)</pre>
    {
```



```
// 获取二维码数据
QRCodeResult* pResult = m_QRCode.GetResultItem(i);
if (pResult)
{
    intnItem = m_List1. InsertItem(i, _T(""));
    CStringstr;
    str.Format( _T("%d"), i);
    m_List1.SetItemText(nItem, 0, str);
    DPNTctPos;
    Center2P( pResult->border[0], pResult->border[2], ctPos );
    doubledAngle = 0;
    dAngle = Angle2P( pResult->border[0], pResult->border[1] );
    str.Format( _T("%0.3f"), ctPos.x);
    m_List1.SetItemText(nItem, 1, str);
    str.Format( _T("%0.3f"), ctPos.y);
    m_List1.SetItemText(nItem, 2, str);
    str.Format( _T("%0.3f"), dAngle);
    m_List1.SetItemText(nItem, 3, str);
    m_List1.SetItemText(nItem, 4, CString(pResult->codeText));
    // 多边形图形
    CGdiPolygon* p1 = newCGdiPolygon;
    if( p1!=NULL )
    {
         p1 \rightarrow SetMax(4);
         p1->Add(pResult->border[0]);
         p1->Add(pResult->border[1]);
         p1->Add(pResult->border[2]);
         p1->Add(pResult->border[3]);
         if( pResult->codeLen>0 )
         {
              p1->SetPenColor(RGB(0,255,0));
         }
         else
         {
              p1->SetPenColor(RGB(255,0,0));
         }
```

www.ckvision.net



```
p1->SetPenWidth(2);
                  m_Results.AddItem(p1); // 添加到显示
              }
              CGdiPoint* p2 = newCGdiPoint(ctPos);
              if (p2!=NULL)
              {
                  p2 \rightarrow SetSize(21);
                  p2->SetStyle(0);
                  p2->SetPenWidth(1);
                  p2->SetPenColor(RGB(0,255,0));
                  m_Results.AddItem(p2); // 添加到显示
              }
              for (intn=0; n<4; n++)</pre>
              {
                  CStringstr;
                  str.Format( _T("%d (%0.2f,%0.2f)"), n, pResult->border[n].x,
pResult->border[n].y);
                  CGdiText* pText = newCGdiText(CStringA(str));
                   if (pText!=NULL)
                   {
                       pText->SetPosition(pResult->border[n].x, pResult->border[n].y);
                       pText->SetPenColor(RGB(0, 0, 255));
                       m_Results.AddItem(pText);// 添加到显示
                  }
              }
         }
    }
    m_GdiView.Redraw(); // 刷新显示
}
```

```
//.....详情请打开实例参考。
```

ReadOcrDemo 字符读取

```
在 StdAfx.h 的头文件中添加 CKVISION 相关定义
#include<sup>"</sup>...\\..\\Include\\CKGDI.h<sup>"</sup>
```



```
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKReader.h"
```

#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
#pragma comment(lib, "..\\..\\Lib_x64\\CKReader.lib")
#else
#pragmacomment(lib, "..\\..\\Lib\\CKGDI.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKBase.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKReader.lib")
#pragmacomment(lib, "..\\..\\Lib\\CKReader.lib")
```

usingnamespaceCKVision;

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库
//...
```

```
CKVision::ExitLibrary(); // 退出 CKVision 库
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的 .h 头文件中定义相应的图像处理功能:

```
CPrImage
              m_Image;
                             // 基础图像
   CGdiRect
              m_Rect;
                             // 范围ROI
   COverlay
              m_Overlay;
                             // 图像显示表面, 前显示的动态图形, 主要用于ROI 显示。
   COverlay
              m_Results;
                             // 图像显示表面, 前显示的静态图形, 主要用于检测结果生成
图形显示。
   CGdiView
              m_GdiView;
                             // 图形显示视图窗口
   CCharset
              m_Charset;
                             // 字符集功
   CReadOcr
                             // 字符识别
              m_ReadOcr;
   BOOL
                             // 自动二值化
              m_bAutThers;
```



```
CLearnDlg
            m_LearnDlg;
                            // 学习字符窗口
COcrListDlg m OcrListDlg;
                            // 编辑字符集
CThresholdDlg m_ThresholdDlg;
                            // 二值化窗口
```

在对话框窗口的.cpp 实现文件中添加相应的功能实现。

// 执行字符读取

{

```
voidCReadOcrDemoDlg::OnExecute()
    // TODO: Add your control notification handler code here
    Overlay_DeleteAll(m_Results);
    if(m_bAutThers==TRUE) {// 自动二值化阈值
         CHistogramhist;
         hist.SetAnalyse( Analyse_Threshold ); // 设置参数分析二值化阈值
         hist.Execute( m_Image, m_Rect );
         m_ReadOcr.SetThreshold( hist.GetThreshold() ); // 设置分割阈值
    }
    // 字符识别参数设置
    m_ReadOcr.SetPolarity( m_Combol.GetCurSel() );
    m_ReadOcr.SetNoiseArea(GetDlgItemInt(IDC_EDIT2));
    m ReadOcr.SetMinWidth(GetDlgItemInt(IDC EDIT3));
    m_ReadOcr.SetMaxWidth(GetDlgItemInt(IDC_EDIT4));
    m_ReadOcr.SetMinHeight( GetDlgItemInt(IDC_EDIT5) );
    m_ReadOcr.SetMaxHeight( GetDlgItemInt(IDC_EDIT6) );
    m_ReadOcr.SetMinScore(GetDlgItemInt(IDC_EDIT7));
    m_ReadOcr.SetUniteSpaceX(GetDlgItemInt(IDC_EDIT9));
    m_ReadOcr.SetUniteSpaceY(GetDlgItemInt(IDC_EDIT10));
    m_ReadOcr.SetUniteEnabled( m_Check1.GetCheck() );
    m_ReadOcr.Execute( m_Image, m_Charset, m_Rect );// 执行字符识别
    m_List1. DeleteAllItems();
    OcrResult* data;
    CString str, text;
    for( inti=0; i<m_ReadOcr.GetResultCount(); i++ ) {</pre>
```



```
data = m_ReadOcr.GetResultData(i); // 返回单个字符数据
    str = data->Text;
    m_List1.InsertItem( i, str );
    text += str;
    str.Format( _T("%0.0f"), data->Score );
    m_List1.SetItemText( i, 1, str );
    str.Format( _T("%d"), RECT_WIDTH(data->Rect) );
    m_List1.SetItemText( i, 2, str );
    str.Format( _T("%d"), RECT_HEIGHT(data->Rect) );
    m_List1.SetItemText( i, 3, str );
    CGdiRect* p1 = new
         CGdiRect( data->Rect ); //矩形框
    if( p1!=NULL ) {
         p1->SetPenColor(RGB(0,255,0));
         m_Results.AddItem(p1); // 添加到显示
    }
}
SetDlgItemText( IDC_EDIT8, text );
m_GdiView.Redraw(); // 刷新显示
```

//.....详情请打开实例参考。

}

SearchDemo 模板轮廓匹配定位(新版本)

在 StdAfx.h 的头文件中添加 CKVISION 相关定义

```
#include"..\\..\\Include\\CKGDI.h"
#include"..\\..\\Include\\CKBase.h"
#include"..\\..\\Include\\CKLocate.h"
```



#ifdef_WIN64

```
#pragma comment(lib, "..\\..\\Lib_x64\\CKBase.lib")
    #pragma comment(lib, "..\\..\\Lib_x64\\CKGDI.lib")
    #pragma comment(lib, "..\\..\\Lib_x64\\CKLocate.lib")
#else
    #pragmacomment(lib, "...\\Lib\\CKBase.lib")
    #pragmacomment(lib, "...\\Lib\\CKGDI.lib")
    #pragmacomment(lib, "...\\Lib\\CKLocate.lib")
#endif
```

```
usingnamespaceCKVision;
```

在应用程序入口和退出的地方增加初始化和释放 CKVISION 库。

```
CKVision::InitLibrary(); // 初始化 CKVision 库
//...End
```

```
CKVision::ExitLibrary(); // 退出 CKVision 库
```

在资源视图 Dialog 中添加相应的界面操作 //.....详情请打开实例参考。

在对话框窗口的.h 头文件中定义相应的图像处理功能:

	CPrImage	m_Image;	// 基础图像
	CMask	m_Mask;	// 图像掩摸
	CPrImage	m_Pat;	// 模板图像
	CShapeMode1	m_Model;	// 形状模板
	CShapeMatch	m_Match;	// 基于边缘轮廓特征的形状匹配
	IRECT	m_lRect;	// 学习区域记录
	CGdiRect	m_lRoi;	// 学习ROI
	CGdiRect	m_sRoi;	// 搜索范围ROI
	COverlay	m_Overlay;	// 图像显示表面, 前显示的动态图形,主要用于ROI 显示。
	COverlay	m_Results;	// 图像显示表面, 前显示的静态图形,主要用于检测结果生成
图形	显示。		
	CGdiView	m_GdiView;	// 图形显示视图窗口



{

在对话框窗口的.cpp 实现文件中添加相应的功能实现。

```
// 执行匹配
voidCSearchDemoDlg::OnSearch()
    // TODO: Add your control notification handler code here
    UpdateData(TRUE);
    m_nSe1 = -1;
    // 设置搜索参数
    m_Match.SetMaxCount(GetDlgItemInt(IDC_EDIT1));
    m_Match.SetMinScore(GetDlgItemInt(IDC_EDIT2));
    m_Match.SetMinAngle(GetDlgItemInt(IDC_EDIT3));
    m_Match.SetMaxAngle(GetDlgItemInt(IDC_EDIT4));
    m_Match.SetMinScale(GetDlgItemInt(IDC_EDIT5));
    m_Match.SetMaxScale(GetDlgItemInt(IDC_EDIT6));
    m_Match.SetMaxOverlap(GetDlgItemInt(IDC_EDIT7));
    m_Match.SetMaxSpeed( m_Slider1.GetPos() );
    m Match.SetAccuracy( m Slider2.GetPos() );
    m_Match.SetPolarity( m_Combol.GetCurSel() );
    CStringtext;
    CModelContourtempl;
    m_List1. DeleteAllItems();
    // 删除所有图形,请注意在线程中调用清除图形时,最好使用发送消息的方式,
    // 可参考ContourDemo 中的CContourDemoDlg::OnExecute() 。
    Overlay_DeleteAll(m_Results);
    BeginTime();
    if( m_sRoi.GetVisible() )
    {
        m_Match.Execute(m_Image, m_Model, m_sRoi); // 执行搜索功能
    }
    else
    {
        m_Match.Execute( m_Image, m_Model, MaxROI );
    }
```



```
EndTime();
SMatchData* data;
for( inti=0; i<m_Match.GetNumMatchs(); i++ )</pre>
{
    data=m_Match.GetMatchData(i); // 获取匹配数据
    text.Format( "%d", i+1 );
    m_List1.InsertItem( i, text );
    text.Format( "%0.3f", data->score );
    m List1.SetItemText( i, 1, text );
    text.Format( "%0.3f", data->center.x );
    m_List1.SetItemText( i, 2, text );
    text.Format( "%0.3f", data->center.y );
    m_List1.SetItemText( i, 3, text );
    text.Format( "%0.3f", data->angle );
    m_List1.SetItemText( i, 4, text );
    text.Format( "%0.3f", data->scale );
    m_List1.SetItemText( i, 5, text );
/* CGdiRotBox* p1 = new CGdiRotBox(
              data->center.x,
              data->center.y,
              data->width,
              data->height,
              data->angle );
    if( p1!=NULL )
     {
         p1->0ffset(0.5,0.5);
         p1->SetPenColor( RGB(0,255,0) );
         m_Results.AddItem(p1); // 把旋转矩形添加到显示
    }*/
    CGdiContour* p2 = newCGdiContour(data->model);
    if( p2!=NULL )
     {
```



```
p2=>Offset(0.5,0.5);
p2=>SetPenWidth(1);
p2=>SetPenColor(RGB(0,255,0));
m_Results.AddItem(p2); // 把模板轮廓添加到显示
}
CGdiPoint* p3 = newCGdiPoint(data->frame.point);
if (p3 != NULL)
{
    p3=>SetSize(30);
    p3=>SetPenWidth(1);
    p3=>SetPenWidth(1);
    m_Results.AddItem(p3);// 把匹配点添加到显示
}
m_GdiView.Redraw(); // 刷新显示
```

}

//.....详情请打开实例参考。



9 附 1.CKVISION API 功能分类

其砷定			
查·仙/牛			
	CKImage.h	图像	
	CKMask.h	图像掩膜	
	/* 图像处理 */		
	CKImgOpera.h	算术和逻辑	
	CKImgFilter.h	滤波处理	
CKBase.dll	CKImgMorph.h	形态学	
CKBase.lib	CKImgTrans.h	几何变换	
CKBase.h	CKImgConve.h	转换功能	
	/* 检测功能 */		
	CKHistogram.h	直方图和灰度分析	
	CKPixelStat.h	像素统计	
	CKSharpAssess.h	清晰度评测	
	/* 其它功能 */		
	CKGeoMeas.h	几何测量	
	CKFrameTrans.h	坐标系变换	

斑占公垢			
<u><u></u> </u>			
	CKPatInspect.h		
	模板对比		
	CKBlobAnalyzer.h	CKMask.h	
CKBlob.dll	Blob 分析	图形掩摸	
CKBlob.lib			
CKBlob.h		CKBlobDef.h	
		Blob 定义	
		CKBlobData.h	
		Blob 数据	

www.ckvision.net



标完与校准		
小足马伐臣		
	CKDotMatrix.h	
CKCalibration.dll	圆形矩阵标定板	
CKCalibration.lib		
CKCalibration.h		
	CKCalibration.h	
	标定功能	

新布识别			
	CKColorMonitor.h	颜色监测	
CKColor.dll			
CKColor.lib	CKColorIdentify.h	颜色识别	
CKColor.h			
	CKHSIThreshold.h	颜色抽取	

因形見子			
国心亚小			
	CKGdiView.h	图形视图窗口功能	
	CKGdiType.h	GDI 模板类	
	CKGdiText.h	文本显示功能	
	CKGdiPoint.h	GDI 点显示类	
	CKGdiFrame.h	坐标系显示	
	CKGdiLine.h	线段图形功能	
	CKGdiRect.h	矩形框功能	
	CKGdiRotBox.h	旋转矩形功能	
CKGDI.dll	CKGdiBoxScan.h	旋转矩形框内扫描线	
CKGDI.1ib	CKGdiCircle.h	圆形功能	
CKGDI.h	CKGdiRing.h	圆环图形	
	CKGdiRingScan.h	圆环内扫描线	
	CKGdiEllipse.h	椭圆图形	

www.ckvision.net



CKGdiContour.h	轮廓图形显示	
CKGdiPolygon.h	多边形图形	
CKGdiProfile.h	投影曲线边缘位置显示	
CKGdiHistogram.h	直方图图形	
CKGdiMask.h	掩摸显示	
CKGdiModel.h	模型轮廓显示	

形状匹配			
	CKNCMatch.h	灰度匹配	
CKLocate.dll			
CKLocate.lib	CKFindModel.h	形状匹配	
CKLocate.h			
	CKShapeMatch.h	新形状匹配	
	CKModelContour.h	模型轮廓	

测量 (点、线、圆)			
	CKEdgeTool.h	边缘点检测	
	CKCaliper.h	卡尺工具	
CKMeasure.dll	CKAcmeTool.h	顶点测量工具	
CKMeasure.lib	CKFitCircle.h	圆拟合工具	
CKMeasure.h	CKFitLine.h	线拟合工具	
	CKScanEdge.h	扫描边缘工具	
	CKScanSpace.h	扫描间距工具	



条码识别(一维码、			
二维码)			
CKReader.dll	CKBarcode.h	条码读取	
CKReader.lib			
CKReader.h	CKReadOcr.h	读取字符	
	CKFindBarcode.h	条码定位	
	CKDataMatrix.h	DataMatrix二维码读取	
	CKReadQRCode.h	QRCode 二维码读取	