**Halcon机器视觉视频教程**

##### 第一章：halcon机器视觉视频教程介绍

1、本课程适用对象及目标介绍

◆适用对象：零基础者、halcon需求者、图像处理、工业视觉入门者等。特别是适合想短时间内了解机器视觉的使用者。

◆目标：本教程将从实用角度出发，零基础、循序渐进进行讲解；课程将通过实际halcon编程操作与视觉算法原理两个方面同时进行讲解，使观看者能够更好的在操作的基础上理解实际的算法原理；课程并且会安排halcon软件与c#软件的混合编程介绍，更好的理解如何在实际使用中去使用halcon。本课程最终的目标是希望大家在观看视频后，能够使初学者简单的入门，学会如何使用halcon以及理解部分算子，并且利用halcon去解决实际的问题，比如宽度测量、二维码识别、字符识别等；了解机器视觉的常用算法，并且如何在实际中根据实际情况去选择算法。

◆课程安排：

第一章 halcon机器视觉视频教程介绍

机器视觉内容介绍、halcon功能介绍等。

第二章 学习halcon的准备工作

包括Halcon配置、软件介绍以及图像、视频读取、常用快捷键等。

第三章 halcon基本编程介绍

介绍基本的编程方法以及编程逻辑、数组运算等，以便于更好的为算法服务。

第四章 halcon中图像预处理

介绍图像常用变化以及相关的使用案例等，特别是将结合实际halcon算子进行讲解。

第五章 图像区域有关操作介绍

介绍图像常见的区域分割、形态学处理等。

第六章 图像边缘及线提取

介绍边缘的提取、边缘拟合等，同样是原理与操作并行讲解。

第七章 图像模板匹配

介绍模板匹配的原理，以及如何创建模板、如何进行模板匹配等。

第八章 ocr字符识别

介绍如何实现标准字符识别、如何延申至非标准字符识别等。

第九章 二维码及条形码识别算法介绍

介绍二维码识别的原理以及实现方法。

第十章 halcon算法设计思路介绍

结合个人的经验，介绍如何进行Halcon的设计思路实现、如何根据需求去设计算法。

第十一章 halcon与c#混合编程介绍

与其他软件c#进行混合编程，如何将算法应用到实际项目中。

2、机器视觉介绍

◆基本介绍：机器视觉是使得机器具有与人一样的视觉，实现检测、测量以及判别等功能。一个正常的机器视觉系统应该包括图像采集系统（光源、相机、镜头等部分）、图像处理系统（图像分析、图像特征提取等）以及机械执行单元。图像采集系统：（1）光源：用于给采集物体进行大光，凸显物体本身。包括同轴光、环形光以及其他光源。（2）相机：工业相机种类较多，主要实现对图像的采集，其内部包括图像采集卡以及CCD或者CMOS芯片等，其中工业使用的CCD相机较多，具有高灵敏度以及低噪声。（3）镜头：镜头与相机配套使用，当然有的相机也会与镜头做成一体。图像处理系统：图像处理系统主要是指利用采集到的图像，进行提取有用信息的过程。该过程包括图像处理、模式识别等知识，也是机器视觉系统中的核心本质内容，是机器的眼睛。机械执行单元：该部分一般由PLC、机械手等组成，属于运动执行部分，实现根据图像处理结果进行实际运动的执行。

◆机器视觉的用途：机器视觉具有广泛的用途，主要应用包括图像检测、视觉定位、物体测量以及物体识别等。通俗讲，就是给机器装上眼睛，比如想进行物体抓取，那首先应该利用视觉获得物体的世界坐标；如果想进行尺寸测量，可以利用视觉获取目标区域以及目标的分离界限。

◆halcon功能介绍： HALCON是德国MVtec公司开发的一套完善的标准的机器视觉算法包，拥有应用广泛的机器视觉集成开发环境。它节约了产品成本，缩短了软件开发周期--HALCON灵活的架构便于机器视觉，医学图像和图像分析应用的快速开发。在欧洲以及日本的工业界已经是公认具有最佳效能的Machine Vision软件。它实现了大部分基本的图像处理算法，是一个丰富的视觉库，可以进行直接调用算法；同时软件又具有自己的操作界面，可以进行自我编程实现视觉算法；还支持语言的转换，可以将halcon编程直接导出为c++、c#等语言。

教程中将主要介绍图像分析、图像处理部分，将从算法原理、算子实现以及效果演示几个部分进行介绍。教程中使用的软件为halcon18，主要原因是低版本官方不再更新license。



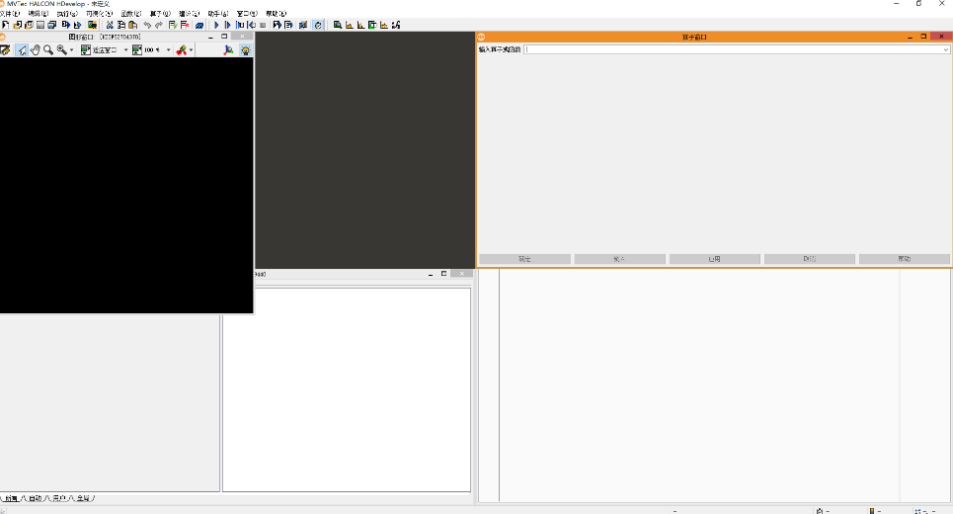


图 halcon图标以及界面

##### **第二章：学习Halcon的准备工作**

1、halcon的安装配置以及界面介绍

◆halcon的安装以及配置：安装过程、如何进行license放置

◆halcon的界面介绍：由四个窗口组成，分别是图像窗口、算子窗口、变量窗口、程序窗口。

2、图片及文件夹的读取

◆图片的读取：图片的读取有两种方法，第一种是直接将图片拖动到halcon界面中，既可以完成图片的读取；第二种是通过read\_image算子进行实现，然后通过更改路径，实现读取图片。两个本质都是利用read\_image，只是前者可以自动识别路径。

read\_image(A,B):读取图片，将路径B中的图片，读取后放到变量A中，也就是A是变量名称，B为路径位置。

绝对路径设置：将程序与图片放置同一个文件夹下，然后将该文件夹以前的路径名称删除即可。

◆文件夹的读取：很多时候，需要对图像中的文件进行读取，可以通过list\_files算子进行实现，实现后再通过算法的遍历，来实现文件夹内图片的读取。

list\_files (A, 'files', Files):读取文件夹A下的图片路径，然后存入Files中。

3、图片动态选择以及视频的读取

◆图片动态选择读取：图片的动态选择是指每一次运行程序时，可以通过显示文件，然后通过文件路径去选择新的图片，不需要将路径进行固定。

dev\_open\_file\_dialog ('read\_image', 'default', 'default', Selection)：读取文件夹下的图片，并将图片路径放置进Selection中进行存储，其他三个参数保持默认值即可。

◆视频的读取：仅限于avi格式的视频，视频的本质是一帧帧的图片，因此读取视频就是将视频转换为图片。

open\_framegrabber ('DirectFile', 1, 1, 0, 0, 0, 0, 'default', 8, 'rgb', -1, 'false', '2-3视频素材.avi', 'default', -1, -1, AcqHandle) 打开视频，并创建一个句柄，句柄是用于操作使用。

4、halcon中的常用快捷键介绍

ctrl+E 打开浏览例程

ctrl+F 查找

F1 查看帮助

F2 重置程序

F3 取消程序注释

F4 程序注释

F5 程序运行，从第一行程序运行到最后一行

F6 程序单步运行

F7 单步跳入函数

F8 单步跳出函数

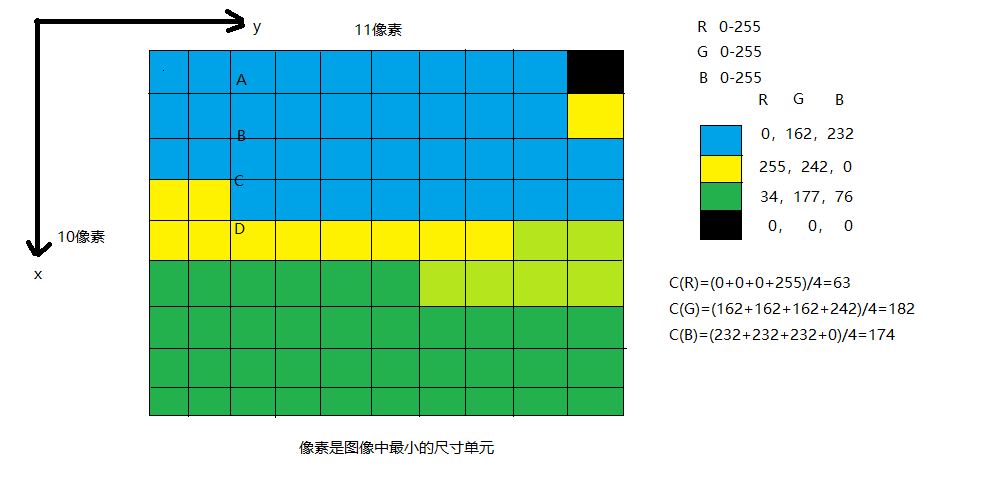
5、如何使用帮助

Halcon中帮助快捷键是F1，打开帮助后会进入每一个算子的详细介绍，指定哪一个算子就可以查看哪一个的介绍，介绍的组成包括几个部分，包括定义、原理介绍以及相关的使用等，如果能够学会正确的使用帮助，将会对学习事半功倍，下边进行详细的介绍。

##### **第三章：halcon中的基本编程介绍**

1、图片的通道转换

◆图片的定义：图片是存储信息的载体。在图像处理（机器视觉）中图片并不是连续的，而是离散的数据点，本质其实就是一个网格，而网格上的每一个点对应着一个或者多个值，该值的范围一般是0-255（跟实际存储位数有关，该处只考虑0-255存储）。



任何一幅彩色图像都可以由RGB三种颜色搭配出来，因此任何图像都可以用数组进行表示，并且该数组是三维数组，f(x,y)=(r,g,b)。但是图像处理中，往往某一个颜色就能够反应图像的特征，没有必要进行三个颜色的分析，因此需要用到通道分离。

◆通道的定义：通道是描述图像颜色层数的名词，比如RGB三个颜色，就称为图像具有三个通道，那分别是R、G、B通道。对于只具有一个通道的图像我们称为单通道图，而灰度图是特殊的单通道图，是将彩色图按照v=aR+bG+cB的关系式将彩色图转换后的结果，单通道图像是图像分析中的主要对象，其中灰度图使用频率最高。

decompose3（image,R,G,B） 通道分离

rgb1\_to\_gray（image,gray\_image）彩色图转换为灰度图

◆颜色空间通道变化：对于有的图像，我们需要进行颜色分析，往往使用HSV通道转换，其中H色相、S饱和度、V明亮度，是按照一定的关系进行转换，具有固定的关系模型，不需要关注。

trans\_from\_rgb（R,G,B,H,S,V, 'hsv'） 颜色通道转换

2、halcon中算子结构介绍

◆算子名称：halcon算子名称是按照英文进行命名的，一般英文就代表真实的含义。但是实际上halcon有一千多个算子，我们怎么去记住呢？如果去一个个记，那就没有必要了，有些算子使用频率非常低，我们只需要记住一个分类方法，然后结合本身的软件提示即可。

draw\_ 表示界面互动的算子，需要人去画区域

gen\_ 开头的，表示生成类的算子，比如生成圆、矩形等

get\_ 开头的，是获取某些属性的算子

tuple\_ 开头的代表与数组有关的算子

select\_ 开通的代表筛选有关的算子

带有 contour 的表示轮廓xld有关的算子

带有 gray 表示灰度有关的算子

Halcon 中算子要学会进行归类

◆算子参数：halcon中算子主要分为两类，参数运算类以及对象运算类，对象包括区域、轮廓xld、图像等。参数运算类一般是数组等，一般是输入输出都为数组，参数多数只有两个值；对于对象运算类算子，一般是第一参数是输入对象，第二个为输出对象，也就是运算结果，而第三行以外的为参数选项。

3、数组的创建及基本运算

◆数组的定义：halcon中的数组跟其他编程中的数组一样，是一组组数据的集合。

在halcon中数组的英文是tuple

◆数组的表示：A=[] A=[1,2] A=[A,1]

注意：先声明再赋值

A=[] A=[A,1] A=[A,1,2] A=[A,[1,2]]

◆数组的基本运算：加减乘除与其他编程类似，其他运算需要使用halcon中的tuple有关的算子

◆如何利用数组修改图像像素值：程序演示

4、矩阵的创建及基本运算

◆矩阵也是一组组的数据，但是与数组不同的地方是，矩阵不仅仅是一个集合，并可以进行求逆、求范数等操作。矩阵的运算只能够通过算子来实现，无法像数组一样直接运算，要想直接操作，只能够先转换为数组，再进行运算。

5、基本编程结构介绍

◆编程结构种类：halcon中编程结构包括顺序结构、分支结构、循环结构、三种，不存在跳转的结构，类似于switch这种语句是不存在的。

◆顺序结构：从上到下，一直运行完成

◆分支结构：if…endif or、and

◆循环结构种类：for…endfor while…endwhile

6.halcon机器视觉实现步骤介绍

◆基本步骤：图像预处理——>区域分割——>区域筛选——>特征分析、有用信息提取等

◆图像预处理：图像去噪、图像基本变化、图像增强、图像位置变化等

◆区域分割：将图像感兴趣的区域从图像中分割出来。包括基于阈值的分割、基于聚类的分割等等。

◆区域筛选：从分割出的所有区域中，根据区域的某些特征，筛选出我们需要的区域。这些特征包括长度、长宽比例、凸性、圆度等等。

◆特征分析：根据筛选的区域，对区域进行定量分析，包括区域的大小、外接圆以及其他特征分析等，来判断区域是否满足某些条件；如果是测量类，需要对区域之间的关系进行分析等等。

##### **第四章：halcon中图像预处理**

1.图像灰度值变化(图像增强)

◆灰度变换目的：图像灰度值变换主要为了提高图像的对比度。对比度就是图像的清晰程度，不同的物体与背景是否对比清晰。包括灰度变换以及直方图变换等。

◆灰度线性变换：将图像的像素点的灰度值按照线性变换函数进行变换。

g(x,y)=af(x,y)+b

还可以进行分段变换，比如

g(x,y)=v1 0<f(x,y)<m1

g(x,y)=af(x,y)+b m1<=f(x,y)<m2

g(x,y)=v3 m2<=f(x,y)<=255

图像取反：g(x,y)=255-f(x,y)

add\_image() 图像相加

invert\_image() 图像取反

◆灰度的非线性变换：图像灰度值采用非线性函数进行变换，常用的为对数函数与指数函数。

g(x,y)=a+bln[f(x,y)+1] 一般是提高低范围的像素，压缩高范围的像素

g(x,y)=a[f(x,y)]+b 相反，提高高范围，压缩低范围像素

log\_image (inputImg,outputImg, 'e')

exp\_image (inputImg,outputImg, 'e')

◆灰度直方图变换：对图像中的像素灰度做映射变换，但是这种映射是基于灰度直方图的。这里介绍直方图均衡化。

equ\_histo\_image（inputImg,outputImg） 直方图均衡化

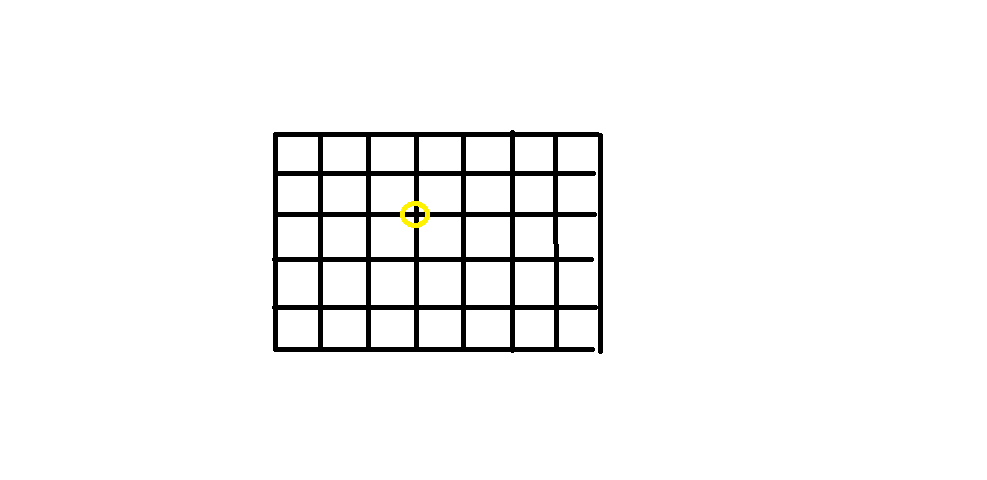
2.图像滤波

◆作用：去除图像中的干扰（噪声）。噪声一般是由外界干扰产生的，比如灰尘、热噪声等。

◆均值滤波：对于模糊图像以及图像内部的噪声具有很好的作用。

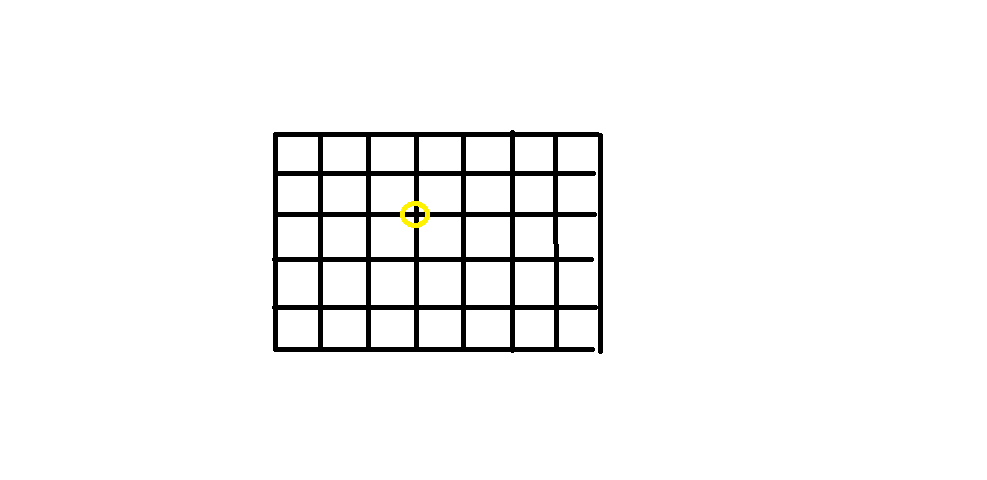
滤波模板是描述滤波器的大小，里边存储着具体的数值。滤波器模板大小是指滤波矩阵的维度。

2\*2 [1 1;1 1]



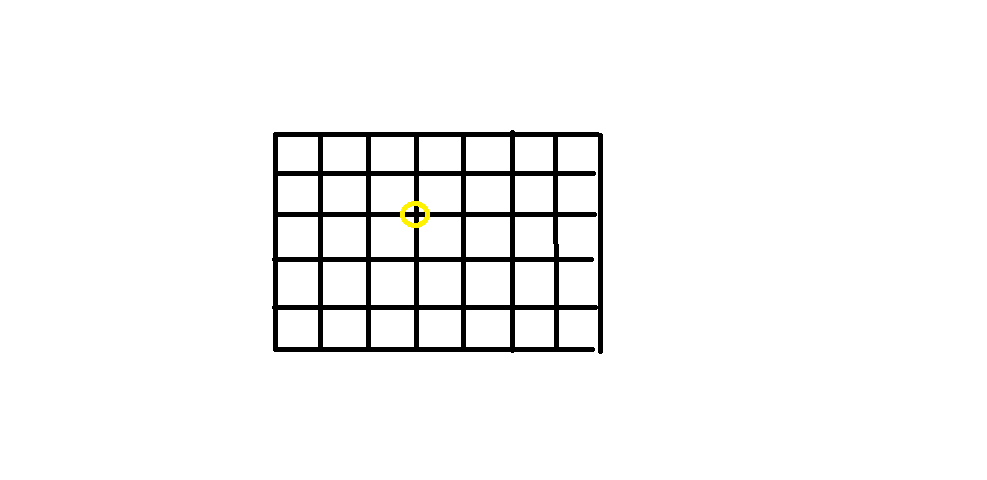
mean\_image()

◆中值滤波：对单个噪声点具有很好的平滑作用，特别是椒盐噪声。



median\_image()

◆高斯滤波：根据高斯函数，考虑权重的滤波。



gauss\_filter ()

◆导向滤波：对于边缘具有很好的保持作用，同时能够对其他地方去除噪声，是很受欢迎并且常用的去噪方法。

guided\_filter ()

3.图像仿射变换

◆解释：图像的变换包括仿射变换与透视变换，一般多数情况下只需要仿射变换。图像的仿射变换是指图像的平移与旋转。

◆映射矩阵：在平移与旋转中，是由一个含有6个参数的矩阵组成，用来描述图像的旋转平移关系。所谓平移关系，是指移动前图像的位置与移动后图像的位置之间的关系，只有有了这关系，才能进行图像的移动。

比如：点（1，2） 与点（2，3）之间的关系为[1,0,1,0,1,1]

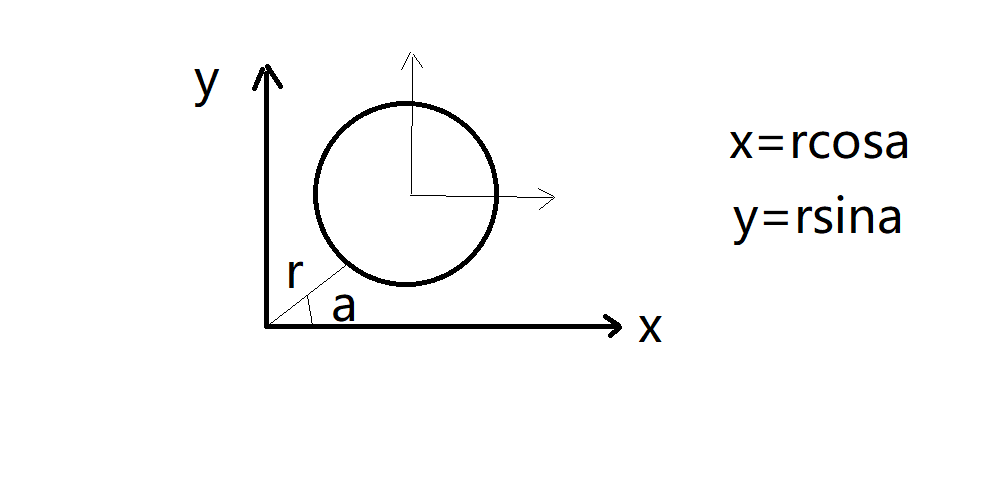
vector\_angle\_to\_rigid（） 计算移动前后的矩阵关系

◆旋转图像：根据矩阵进行图像的旋转。

affine\_trans\_image（） 旋转图像

4.图像的极坐标变换

◆极坐标系的定义：



在平面内取一个定点O， 叫极点，引一条射线，叫做极轴，再选定一个长度位和角度的正方向。对于平面内任何一点，用r表示线段的长度，a表示角度，r叫做点的极径，a叫做点的极角，有序数对 (r,a)就叫点的极坐标，这样建立的坐标系叫做极坐标系。

◆极坐标系的变换：选取极坐标原点，并将原坐标系变换为极坐标系的过程称为极坐标系的变换。关键点在于极坐标系原点的选取以及起始角度的设置。

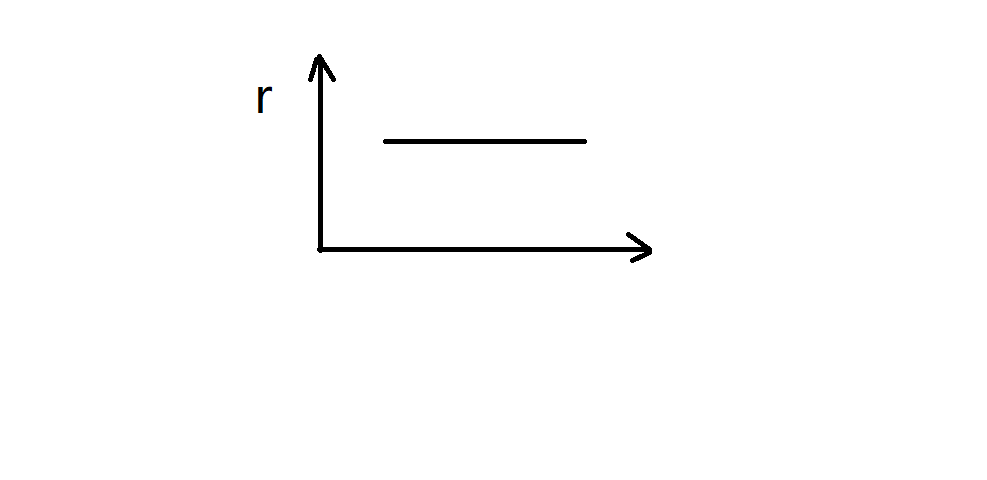
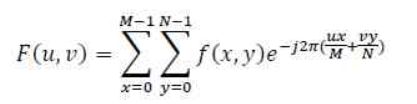


示意图 极坐标展开

5.图像的傅里叶变换

◆傅里叶变换定义：傅里叶变换是时域到频域的变换方法，通俗讲是将现在的空间变换到一个能够反映某些事物出现频率的空间。

◆图像傅里叶变换：



◆用途：一般用于对出现频率高的像素点的分析以及噪声的去除。

◆频率图特点：图像中心为频率为0的原点，由内到外频率越来越高。

其中灰度变换激烈的地方对应高频成分，如边缘；灰度变换不大的地方对应低频。

6.案例-人脸去斑

◆原理：人脸去斑是利用滤波进行去除斑点，一般用的滤波包括双边滤波、各向异性滤波等， 这些滤波与其他滤波原理相似，只是这些滤波考虑了灰度这一信息，不单纯的只是考虑距离。

##### **第五章：图像区域有关操作介绍**

1.全局阈值分割介绍

◆阈值分割：指按照给定的阈值范围，从图像中将像素点灰度值介于阈值范围内的那些点分割出来。（图像处理中又称为二值化）

◆全局阈值分割：全局阈值分割是指针对于整张图像，给定数值，将像素点从图像中分割出来。

◆全局阈值分割关键点：在于阈值的选取，一般是根据图像的特征或者图像的灰度直方图进行阈值的设置。

2.动态阈值分割介绍

◆动态阈值分割：动态阈值分割是指在图像分割的过程中，不用人为的去设置阈值，而是根据图像中存在的特征，进行分割。

◆原理：一般是将原图像与处理后的图像作差，然后去计算差值图像中的亮色区域或者暗色区域。其本质相当于对图像灰度直方图的平滑，进而求取图像中的波谷或者波峰。

◆特点：动态阈值分割具有抗干扰性强，稳定性强的特点，对光照变化不敏感。

◆主要算子：

dyn\_threshold（） 动态阈值分割（需要与滤波进行结合）

local\_threshold（） 动态阈值分割（一步到位）

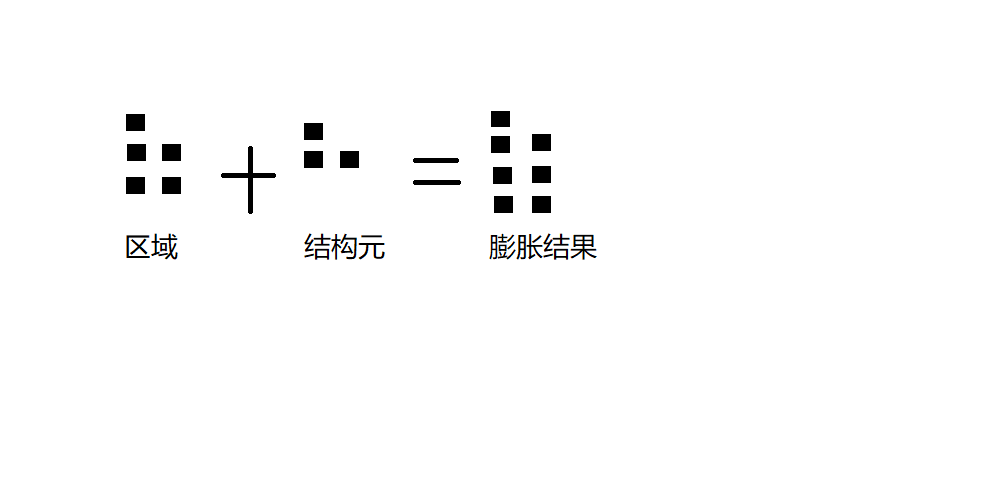
3.其他阈值分割介绍

◆介绍：除了全局阈值分割、动态阈值分割，halcon中还提供了很多其他的阈值分割，比如自动阈值分割、二值化阈值分割、分水岭阈值分割等等。

4.形态学膨胀介绍

◆形态学介绍：数学形态学是用集合论方法描述目标集合结构的学科，在集合的基础上通过物体和结构元素的相互作用运算来得到物体本质的形态。

◆形态学膨胀：为什么叫形态学膨胀呢？是因为图像处理中膨胀是借鉴了形态学的方法。



◆膨胀特点：对于填补图像分割后物体的空洞以及区域边界的凹陷区域具有很好的效果。

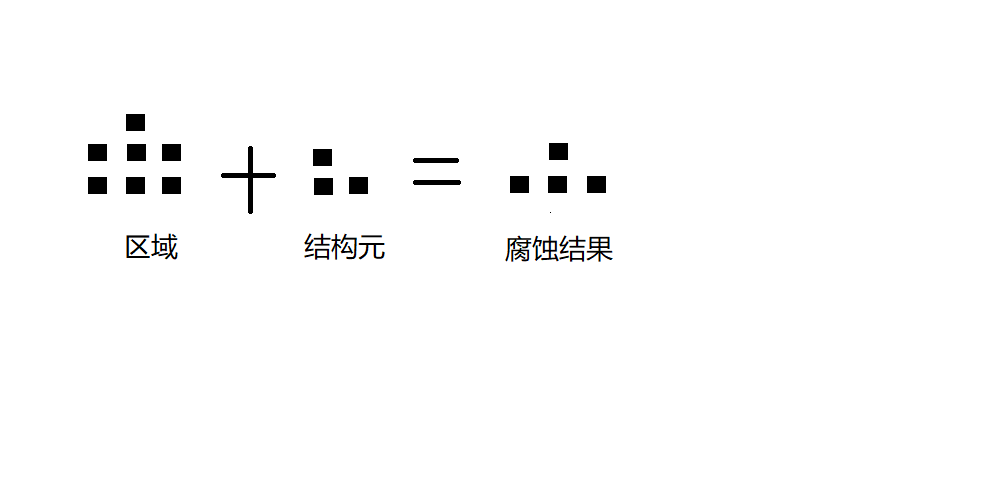
◆膨胀分类：根据膨胀结构元的形状，在halcon中分为圆形膨胀以及矩形膨胀等。

dilation\_circle（） 圆形膨胀

dilation\_rectangle1（） 矩形膨胀

5.形态学腐蚀介绍

◆形态学腐蚀：



◆腐蚀特点：对于去除图像分割后物体的突出区域以及区域边界的凸区域具有很好的效果。

◆腐蚀分类：根据腐蚀结构元的形状，在halcon中分为圆形腐蚀以及矩形腐蚀等。

erosion\_circle（） 圆形腐蚀

erosion\_rectangle1（） 矩形腐蚀

◆灰度图像的腐蚀：扩大暗色背景，腐蚀亮色区域

gray\_erosion\_rect()

◆灰度图像的膨胀：扩大亮色背景，腐蚀暗色区域

gray\_dilation\_rect()

6.形态学开闭运算

◆开运算：对区域先进行腐蚀操作，然后对腐蚀的结果再进行膨胀操作，称为开运算。

◆开运算特点：具有与腐蚀相似的特点，但是能够防止区域腐蚀过度，能够很好的保持区域的形状。

opening\_circle（）

opening\_rectangle1（）

◆闭运算：对区域先进行膨胀操作，然后对膨胀的结果再进行腐蚀操作，称为闭运算。

◆闭运算特点：具有与膨胀相似的特点，但是能够防止区域膨胀过度，能够很好的保持区域的形状。

closing\_circle（）

closing\_rectangle1（）

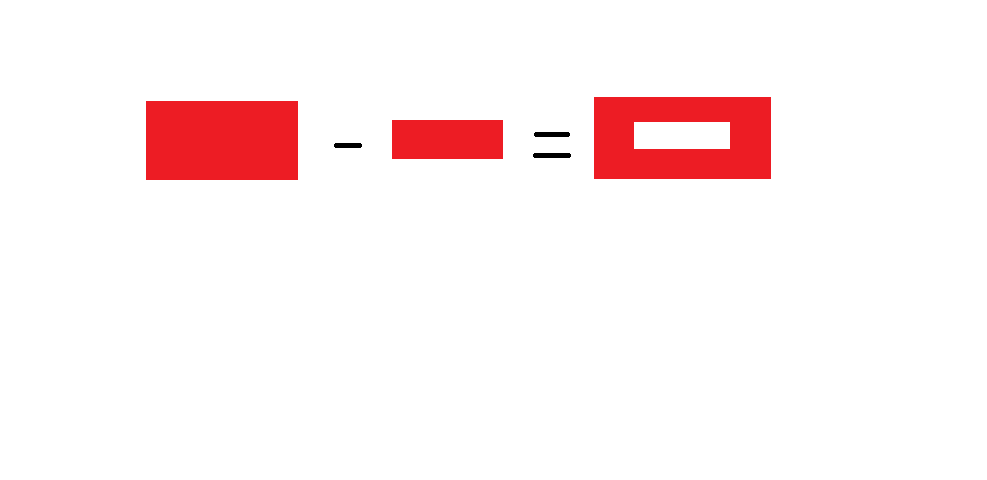
◆灰度图像的开闭运算：

开运算：gray\_opening\_rect（）

闭运算：gray\_closing\_rect（）

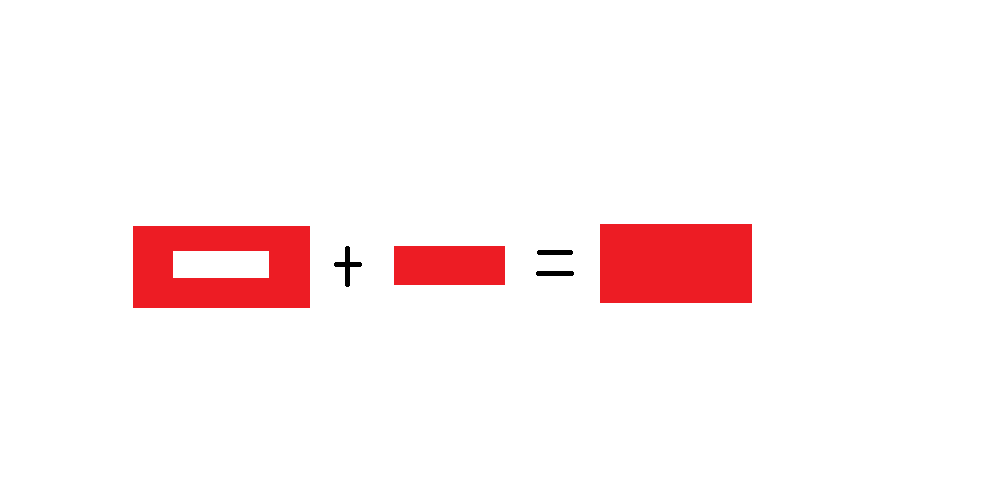
7.区域的运算

◆区域作差：将一个区域减去另一个区域，便得到区域的差值。



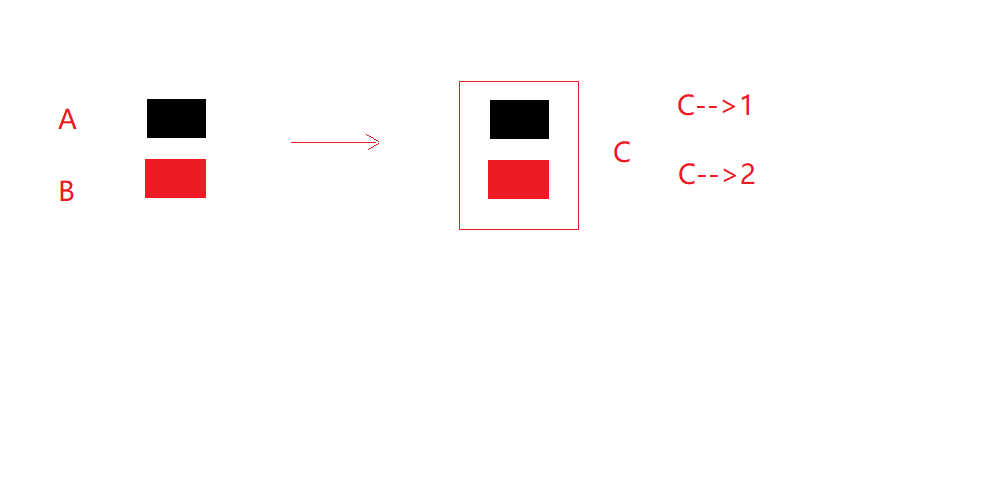
算子：difference（）

◆区域合并（求和）：将一个区域与另一个区域求和，便可以将两个区域合并为一个区域。



算子：union1() union2()

◆区域集合：将两个或者两个以上的区域放入一个集合中，这个集合就成为区域的集合。特别注意的是，在集合中，每一个区域都是孤立的，并没有进行合并，要与区域的和进行区分。每一个区域称为一个obj。

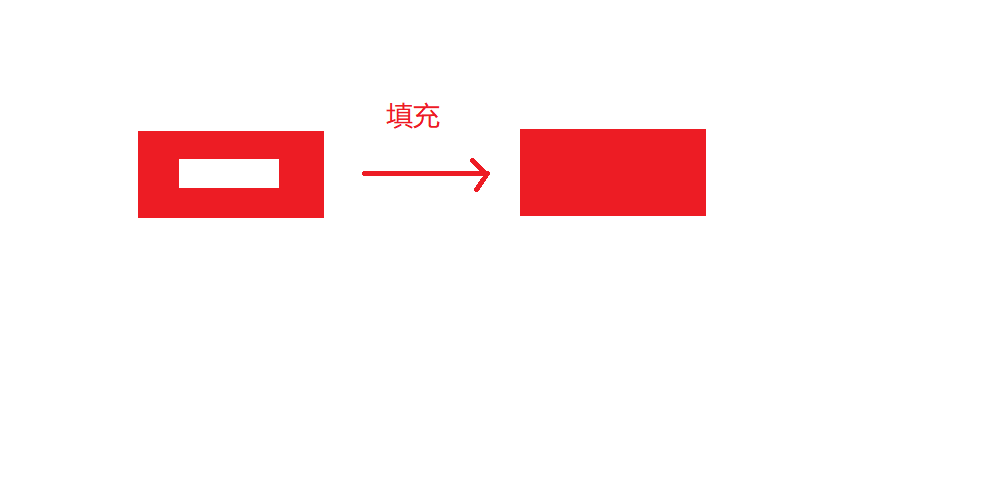


算子：concat\_obj()

◆区域的选择：从集合中进行区域的选择。

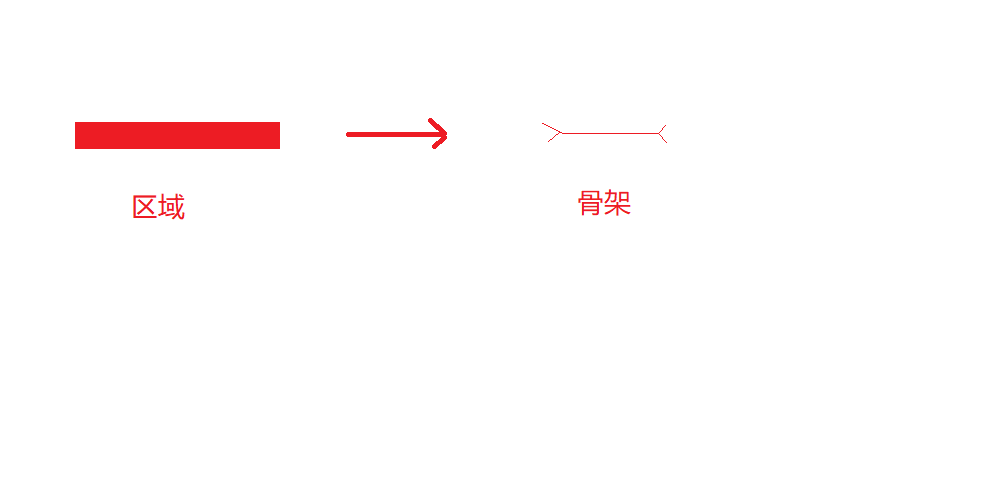
算子：select\_obj()

◆区域的填充：对区域中的孔洞进行封闭，称为孔洞的填充。



算子：fill\_up() fill\_up\_shape()

◆区域骨架求取：骨架可以理解为图像的中轴，骨架求取就是求取区域中轴的过程。



算子：skeleton（）

8.区域的特征分析（1）

◆为什么要学习区域特征：根据前边的讲解，我们已经学会区域的提取，但是如果我们只想从区域中选择其中的一个区域，那该如何选择呢？这就需要用到区域的特征，要根据每一个区域的特征，进行选择出我们需要的区域。

◆常用的区域特征：区域的面积、区域的最小外接矩形的长宽以及角度、区域的角度、区域的凸性、区域的圆度、区域的最小外接圆大小等。

◆区域的面积：区域的像素点的数量总和称为区域的面积。

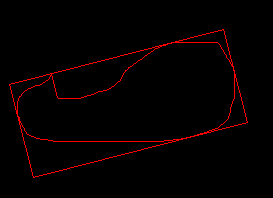
9.区域的特征分析（2）

◆代码演示

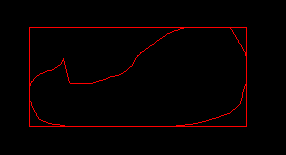


算子：area\_center()

◆区域的最小外接矩形：区域的最小外接矩形，外接矩形的长宽的一半称为长宽半轴，对应的矩形长轴的方向称为区域的方向。



外接矩形rect2



外接矩形rect1

smallest\_rectangle2（） 最小外接矩形2

smallest\_rectangle1（） 最小外接矩形1

◆区域的角度：区域的角度就是外接矩形的角度。

◆区域的凸性：每个区域都存在一个最小的凸性区域，这个区域中任意中两点连线都在该区域内，这样的区域就称为区域的凸性区域。其中原区域与凸性区域的比值，称为区域的凸性，其范围是0-1.

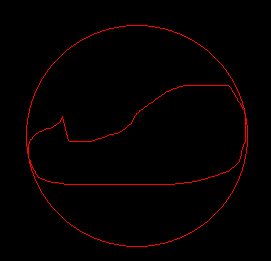
shape\_trans（） 将区域转换为凸性区域（凸包）

convexity() 获取区域的凸性值

◆区域的圆度：区域的圆度，是指区域边界是否接近圆形的程度，范围是介于0-1.

circularity（）

◆区域的外接圆：能够将区域完全包含的圆区域，称为区域的最小外接圆。



区域的外接圆

smallest\_circle（） 获取区域的最小外接圆

region\_features（） 可以获得区域的任意特征

10.案例-提取遥控器字符特征（1）

◆需求：提取遥控器上的字符区域，并求取每一个区域的面积大小以及外接矩形的长宽，并存储到txt文件中。

◆需求分析：可以先转换为灰度图或者颜色空间图，将遥控器区域选择出来，选择出来以后，再去求取字符区域，完成后，再对每一个区域进行面积计算

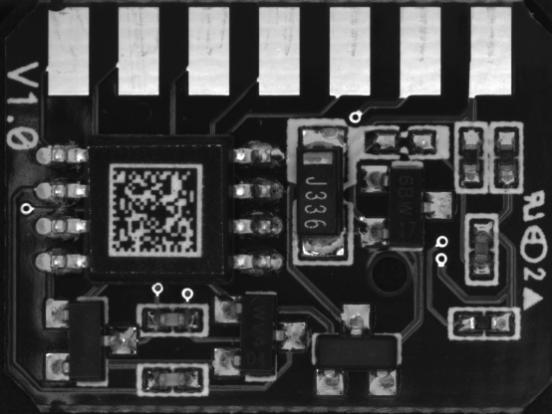


11.案例-提取遥控器字符特征（2）

**第六章：图像边缘及线提取**

1.图像边缘提取原理

◆边缘的定义：边缘是图像中灰度发生明显变化的地方，是不同灰度区域之间的界限。



◆边缘的矩阵形式：

1 1 1 10 10 10

1 1 1 10 11 12

1 1 1 10 12 15

◆如何进行求取边缘：采用掩模的方式，在图像矩阵上进行移动，然后利用掩模内的掩模值进行运算，进而实现边缘的求取。

1 1 1 10 10 10

1 1 1 10 11 12

1 1 1 10 12 15

举例:3\*3掩模

-1 0 1

-1 0 1

-1 0 1

-1\*1+0\*1+1\*1+（-1）\*1+0\*1+1\*1+（-1）\*1+0\*1+1\*1=0

-1\*1+0\*1+1\*10+（-1）\*1+0\*1+1\*10+（-1）\*1+0\*1+1\*10=27

◆常用的掩模：

Robert边缘：边缘定位准，但是对噪声敏感;适用于边缘明显且噪声较少的图像分割;边缘定位精度低。

模板1：

1 0

0 -1

模板2：

0 1

-1 0

Prewitt边缘：对噪声可以适当抑制，但是会起到边缘平滑，对边缘的定位不如Roberts算子。

x方向：

-1 0 1

-1 0 1

-1 0 1

y方向：

1 1 1

0 0 0

-1 -1 -1

Sobel边缘：Sobel算子对于像素的位置的影响做了加权，可以降低边缘模糊程度。

x方向：

-1 0 1

-2 0 2

-1 0 1

y方向：

1 2 1

0 0 0

-1 -2 -1

其他边缘：拉普拉斯边缘等

◆Canny边缘：非极大值抑制与双阈值的思想，边缘的提取效果在目前边缘提取中是相对效果最好的。

2.边缘提取算子介绍

◆内容：本节将对上一节讲解的边缘提取方法进行halcon算子的实验介绍。

3.图像的亚像素边缘提取

◆亚像素边缘：所谓亚像素边缘，就是精度比像素要高。由原先的像素单位为1，变为比1小的精度。

◆亚像素边缘原理：采用插值的方式，精确定位边缘的位置。

1 5 最邻近插值： 1 3 5

1 6 1 3.5 5

双线性插值、三次样条插值

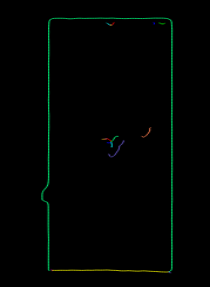
◆亚像素边缘：常用的方法是canny边缘方法，算子是edges\_sub\_pix（），可以直接得到亚像素边缘。

区域 region obj 亚像素轮廓：xld contours

感兴趣区域 ROI

4.亚像素轮廓的特征分析

◆为什么需要特征分析：如图所示，这是上一节的边缘情况，真实的边缘是周围线条，而现在引入了中间的干扰，我们需要根据特征进行筛选，而这个特征依据，就是该节的亚像素特征。



◆亚像素轮廓特征：与之前的区域特征一样，轮廓同样具有自己的特征，在亚像素轮廓中使用较多的特征包括长度、角度、外接矩形的长宽等。对于其他的使用较少的特征讲不做讲解。

◆亚像素轮廓的长度：轮廓所占的像素数。

◆亚像素轮廓的角度：轮廓最小外接矩形中，长边所对应的方向。

◆亚像素轮廓外接矩形：能包括整个轮廓的最小矩形就是最小外接矩形。

◆注意：在亚像素的外接矩形求取时，需要加上\_xld

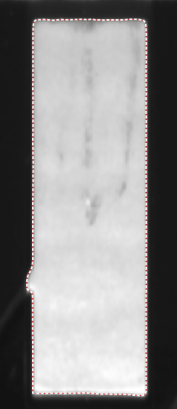
smallest\_rectangle1\_xld（） 求取xld的最小外接矩形1

smallest\_rectangle2\_xld（） 求取xld的最小外接矩形2

select\_shape\_xld() 筛选xld

5.xld的分割及直线拟合

◆xld的分割：根据xld之间各个节点的连接情况以及设置条件，将一条或多条线分割成更多条线的情况，称为xld的分割。一般线的折点的地方为分割点。



算子：segment\_contours\_xld（）

◆xld的拟合：根据线条的预先模型，对线条进行重新生成。包括直线的拟合、圆的拟合以及椭圆拟合等。

算子：

fit\_line\_contour\_xld（） 拟合直线

fit\_circle\_contour\_xld（） 拟合圆

fit\_ellipse\_contour\_xld（） 拟合椭圆

◆xld的拟合原理：基于最小二乘法拟合、考虑权重的拟合方法



6.圆以及椭圆拟合

◆介绍：上一节讲了直线的拟合，这一节针对圆的拟合进行讲解。圆的拟合与直线的拟合，也包括线条的分割、筛选以及拟合。

◆案例分析：

7.中心线的提取

◆介绍：中心线是处于某区域中心的线,比如图中的道路中心，很多情况下需要求解中心。



◆求解方法：

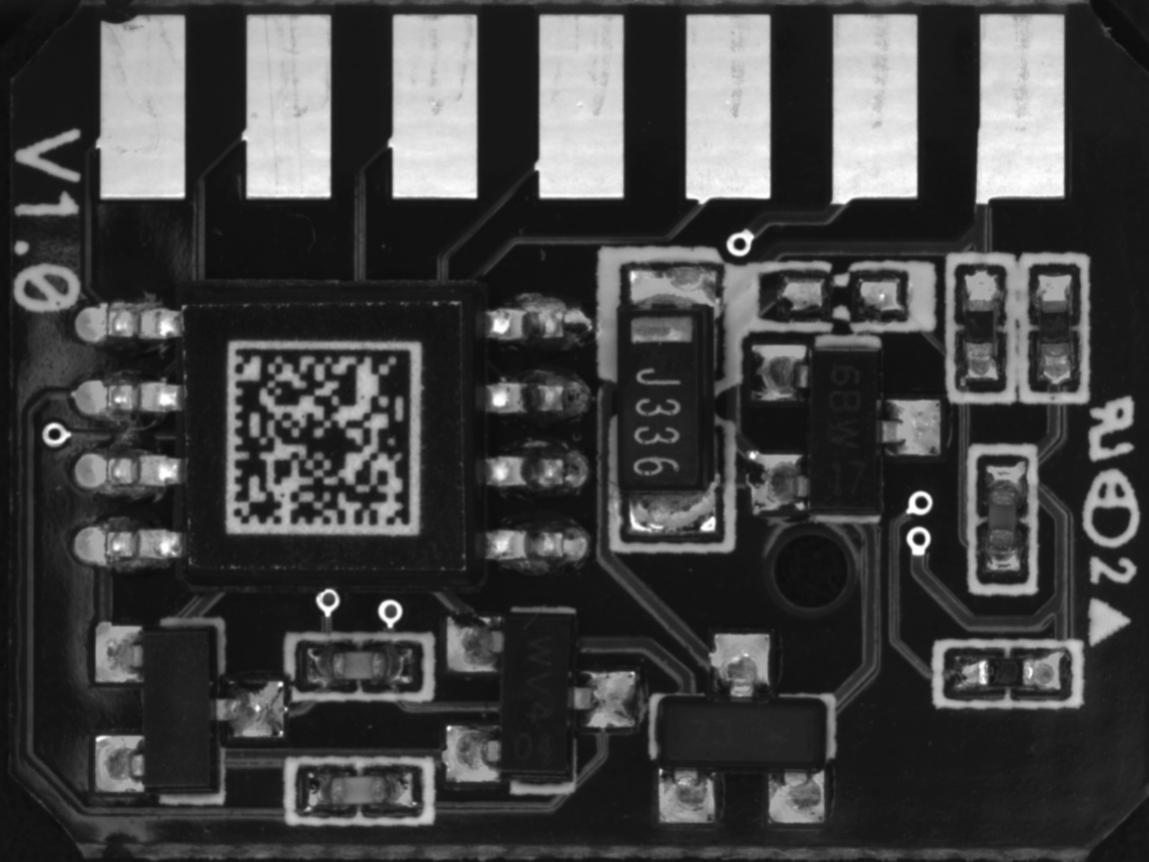
方法1：求解区域的边缘，然后进而计算中心。

方法2：先求解区域，然后利用区域骨架求解中心。

方法3：利用lines-gauss算子求解

8.案例-宽度测量（1）

◆需求：计算白色矩形的平均宽度。



9.案例-宽度测量（2）

10.案例-宽度测量（3）

11.案例-宽度测量（4）

##### **第七章：图像模板匹配**

1.模板匹配基本原理

◆什么是模板匹配：模板匹配通俗讲，就是拿着模板去匹配，意思是，我们事先制作一个模板，然后利用这个模板，去图像中寻找与其相似的的部分，并记录出寻找到的位置。

举例：比如我们想找到下面这幅图片中的“开”字的位置，按照我们以前的方法，可以阈值分割，并进行特征提取、特征筛选。但是并不一定会求得准确。如果采用模板的方法，我们可以将开始进行裁剪，裁剪后制作成模板，然后去图中进行匹配。

模板：



◆模板匹配分类：按照有无变形，分为刚性模板匹配与变形模板匹配，由于变形匹配的复杂性以及工业使用较少，将不做详细讲解，主要讲解刚性模板匹配。所谓刚性模板匹配，是模板与搜索的区域大小几乎相同，当然相差几个像素是允许的。

◆模板匹配原理：模板匹配是通过搜索的方式进行匹配，比如最简单的匹配。

举例：

假设模板为：

2 5

1 6

搜索对象：

1 5 9 8

2 5 3 9

7 2 2 5

5 7 1 6

当搜索完全重合时：

|2-2|+|5-5|+|1-1|+|6-6|=0

◆模板匹配分数：在模板匹配中，我们会通过某些方法，来计算模板匹配的相似程度，这个就是匹配分数，当相似程度越高，匹配分数越接近1，否则越接近0.我们会记录所有的搜索结果，最终选择匹配值最高的作为结果，就是搜索的实际位置。

常用的计算方法：包括上面利用差值的方法、利用区域内部灰度方差的方法等。

◆模板匹配搜索方式：上面知识最基本的搜索原则，而实际使用中，只是利用差值等，对于图像光照等变化，匹配效果会十分不好，而实际halcon中常用的搜索为ncc模板匹配以及基于形状的搜索方法等等。

2.ncc模板匹配（1）

◆ncc匹配原理：模板图像减去模板图像的平均值后，再与图像减去图像的平均值后，求两个差值的乘积，再处以灰度方差。相当于零均值，对光照变化的敏感度变低。



◆模板匹配的过程：创建模板以及模板匹配两个过程，具体程序分析。

模板：



3.ncc模板匹配（2）

4.形状模板匹配

◆形状匹配原理：模板创建后，会根据模板中的边缘等关系去图像中进行搜索，可以理解成提取模板中的灰度发生变化的那些点，然后再去图像中匹配，这样的效果是对干扰比较稳定，匹配速度快，并且即使边缘发生部分遮挡，匹配效果依旧较好。

模板：



5.xld模板匹配

◆匹配原理：xld是亚像素轮廓，所谓xld模板匹配，是直接利用灰度变化明显的地方进行xld的提取，然后去图像中进行匹配。

6.模板保存及匹配加速

◆为什么要保存模板：我们前边程序创建模板与模板匹配都是在一个程序中，而实际使用中模板与匹配是程序分开的，因此我们需要对模板文件进行保存，然后另一个程序进行读取，进而实现模板创建与模板匹配分离。

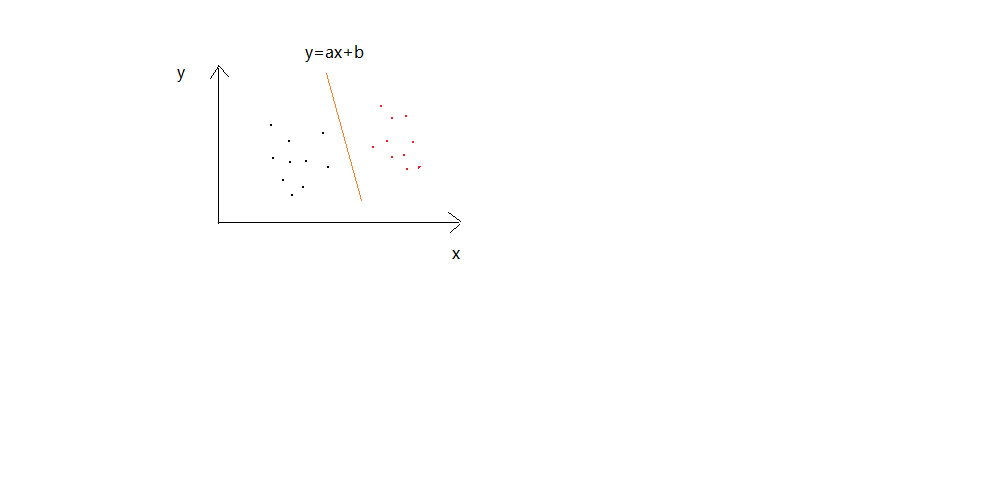
具体看程序分析。

##### **第八章：分类器及ocr字符识别**

1、分类器原理(1)

◆分类器：分类器属于模式识别的范畴，是通过给定的数据，根据事先的标签结果，来寻找合适的分界线以及分界规律，同时要使得这些规律对于类似的数据能够更大程度的适用。

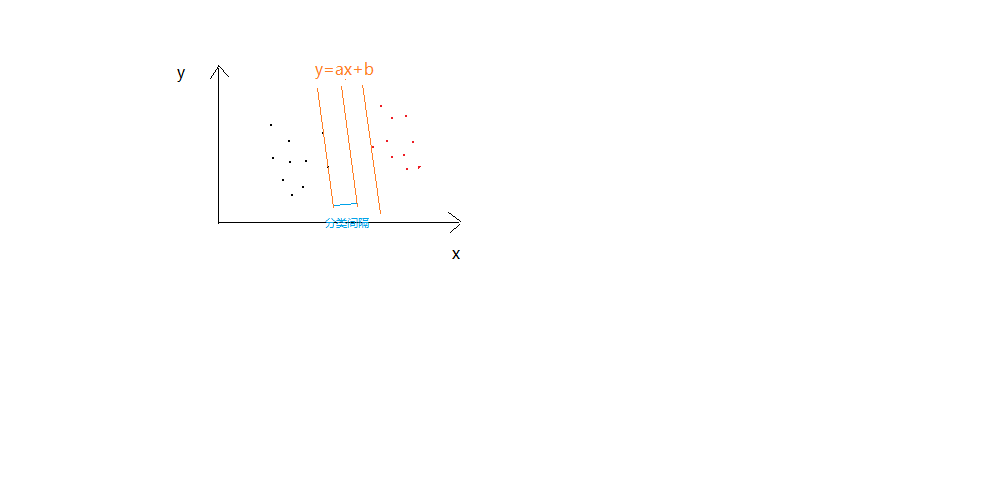
◆举例：左侧代表类型1，右边代表类型2，那么y=ax+b就是他的分界线。



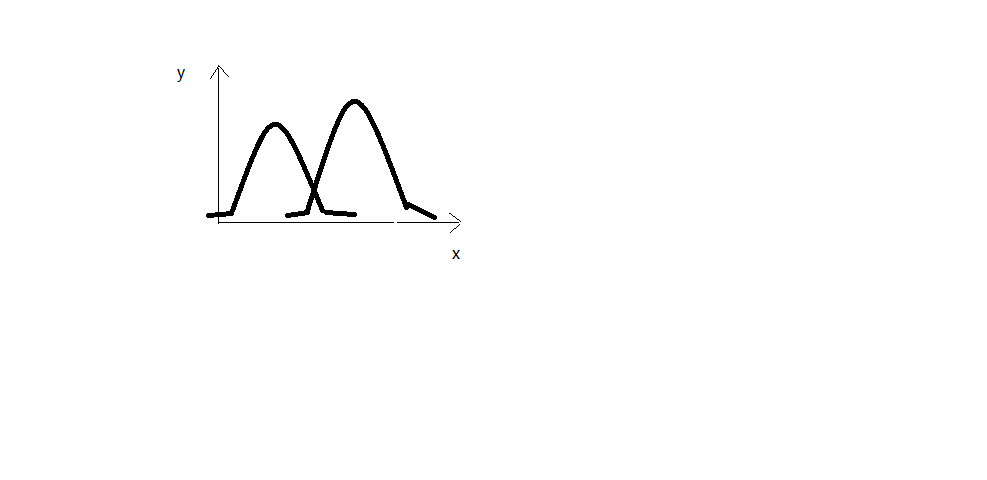
◆分类器的优化：分界线的位置以及分界线的形状、数据点的输入都是优化的主要方面。

（1）比如y=ax+b，其实它的位置取多个位置都能够满足要求，但是实际上，最优的只有一条。

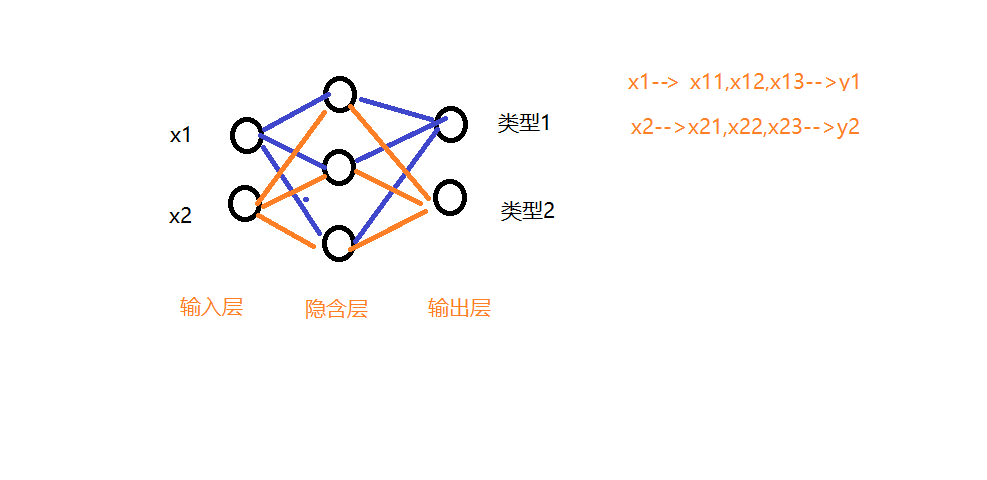
在分类的原则中，常用的一种方法就是保证类间方差最大，而类内方差最小。典型的代表就是是svm分类器。svm分类器称为支撑向量机。



1. 分界线的形状：高斯混合分类器GMM，就是利用数据中出现的概率进行分类，理论上可以生成任何形状的线条。

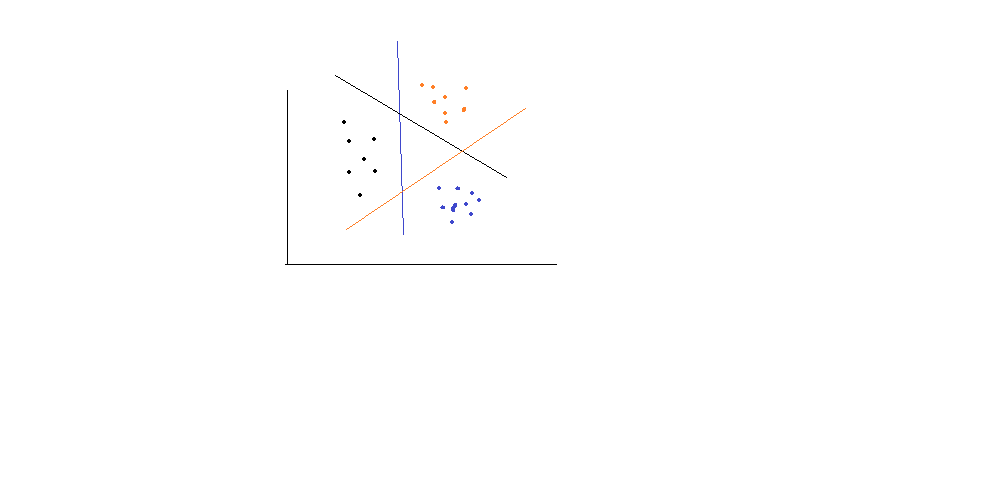


1. 感知器mlp（类似bp神经网络）



2、分类器原理(2)

◆多类型分类：两种方法，一对多训练或者每一个都训练比较概率。

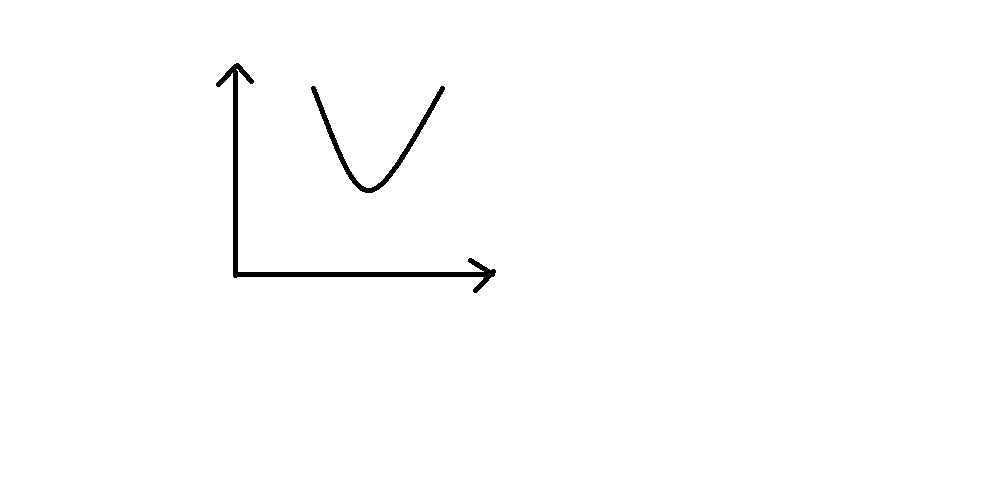


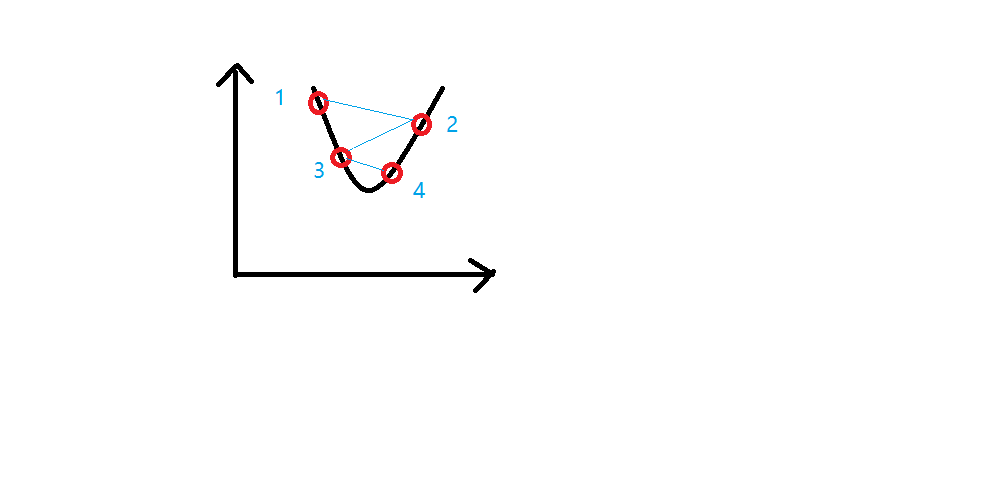
实际上，这些步骤，在halcon只是两个算子，创建分类器以及分类器训练，原理主要是帮助大家理解。

◆特征训练：分类器本质是对数据的训练，所以我们需要提取图像的特征，最简单的就是直接使用图像的所有灰度，但是这样相对耗时，使用的较多的是图像的灰度特征，比如平均灰度、灰度方差等等。

◆什么叫训练：其实训练就是数据迭代的过程。比如上面的分类器中，我们讲解的主要是其原理，而在实际中，既然存在一个最佳的优化结果，那么就可以根据数学理论得到优化目标函数，我们称其为损失函数，要想得到最佳的结果，那么就需要使得损失函数最小。

比如说：d=ax2+bx+c是一个损失函数，如果按照我们的思维，可以直接求得最小值，但是在计算机中，实际并不是这样，往往采用迭代的方式进行求解。





3、分类器实现图像分割

◆介绍：比如下面这幅图像，我们可以通过分类器实现白色、黄色以及蓝色、黑色区域的一个阈值分割。首先要通过选定这些区域中的代表区域，进行训练，然后识别。



创建svm分类器：create\_class\_svm（）

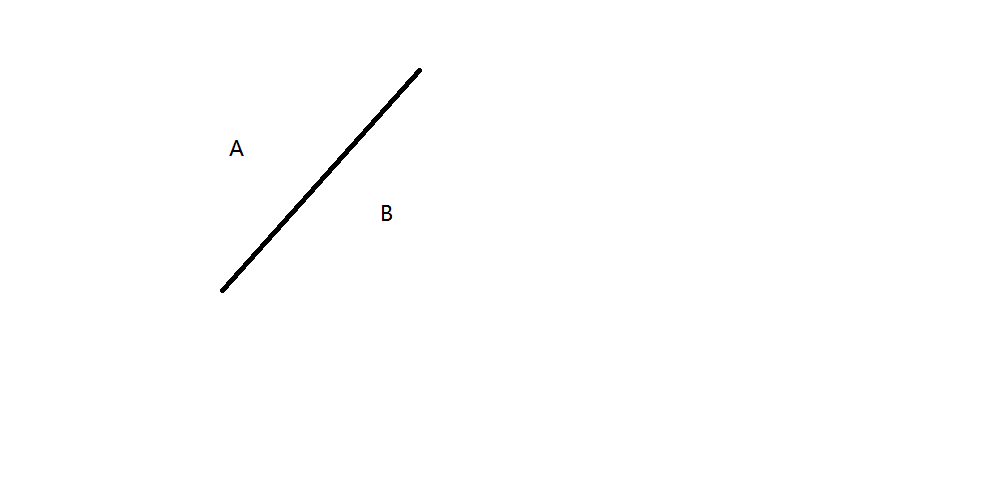
增加样本：add\_samples\_image\_class\_svm（）

分类器训练：train\_class\_svm（）

分类识别算子：classify\_image\_class\_svm（）

4、字符识别介绍

◆原理：其实字符识别的本质是利用分类器实现的，比如如何区别A与B，那么可以通过建立两者的分类器模型。然后对其进行分类。在halcon中其实已经训练好了一些标准的字符，并且可以直接调用。



字符区域的提取——》读取分类器训练结果——》字符识别

5、案例-车牌识别

◆需求：请识别出图中车牌号码，并保存至txt文档。

◆思路：利用颜色信息，提取车牌号，然后将图像旋转至水平，再去提取字符，并利用halcon中的ocr库，进行识别。

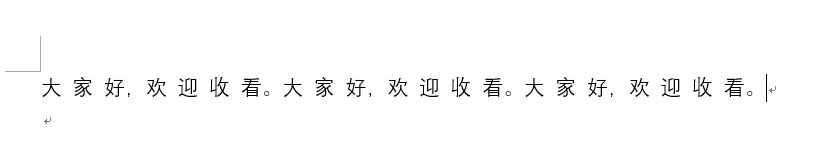
注意：要把字符旋转至水平，识别率会更好。



◆延伸：如果说字符出现连接等，可以通过水平投影等方法进行准确分割。

6、案例-汉字识别（1）

◆需求：请识别出图中汉字，并保存至txt文档。



◆汉字与字符识别区别：原理上是一样的，区别在于无法使用halcon内置的分类结果，需要自己取训练，然后分类。

7、案例-汉字识别（2）

8、案例-汉字识别（3）

##### **第九章：条码识别及助手介绍**

1、条码原理及算子介绍

◆条形码原理：一维条形码由黑白两种颜色组成，具有宽高两个尺寸，其中宽度方向代表着数据信息。条形码上是一般标注着字符、数字、符号等。比如下面的Code 128-A、Code 128-B、Code 128-C.条形码一般是由前置符、中止符、数据符、中间分隔符、校验符等组成。条形码是利用条纹和间隔或宽窄条纹构成二进制的”0“和”1“，反映实际的信息。



Code 128-A



Code 128-B



Code 128-C

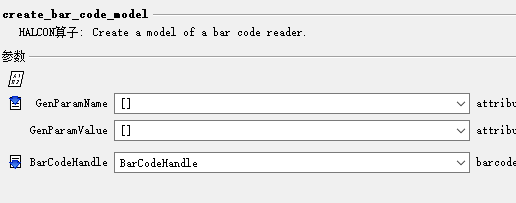
◆二维条码原理：二维条码是在一维条码的基础上扩展出来的条码，使用黑白矩形图案表示二进制数据。一维条码的宽度代表着某些数据，而长度没有记载数据。二维码的长度、宽度都载着数据。二维条码并且有定位矩形以及容错机制，即使二维码有部分污染时，也能够尽可能的还原信息。同样二维码也具有不同的标准，具有多种格式。但halcon仅支持utf8和locale两种字符集。UTF-8一种字符集，是一种编码规则.



◆算子介绍

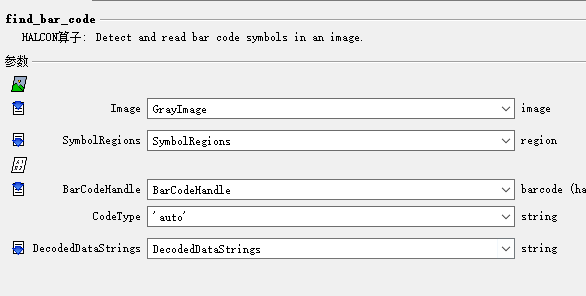
create\_bar\_code\_model（） 创建条形码模型算子

条形码名称、条形码参数、条形码模型



find\_bar\_code（） 识别条形码算子

输入图片、识别出的条形码区域、条形码模型、条形码类型、识别结果



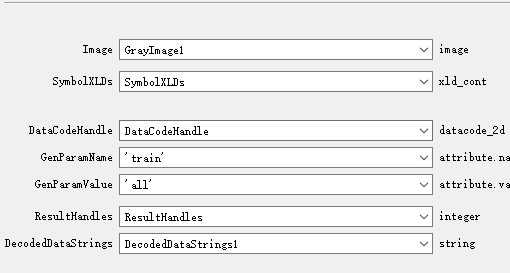
create\_data\_code\_2d\_model（） 创建二维码模型

二维条码类型、模型名称、条形码参数、条形码模型



find\_data\_code\_2d（） 识别二维码

输入图片、识别出的二维码区域、模型、类型、代表是否成功、识别结果



任意方向、裁剪

2、条码识别算法介绍

◆介绍：本节将根据上一节的分析，针对上一节的条形码及二维码进行识别。

3、采集助手介绍

◆介绍：在halcon中提供了助手这个菜单，包含采集助手、标定助手、匹配助手、测量助手以及ocr助手，一共五个助手。在我们实际使用中，使用的最多的是采集助手以及测量助手。



4、匹配助手介绍

◆介绍：匹配助手中，我们可以直接勾选模板区域，并输入图片，然后自动去进行模板的匹配。



5、测量助手介绍

◆介绍：测量助手中，我们可以先选定图片，然后直接勾选测量的宽度区域，然后参数设置后，自动去进行宽度的测量。



6、ocr助手介绍

◆介绍：识别助手中，我们可以先选定图片中的字符，然后直接将预先的字符放入，必要时候需要进行训练，可以实现字符的识别，但是需要是英文字符。



7、函数封装介绍

◆函数定义：把某些算法语句，可以封装在一起，实现某些特定的功能，封装得到的就是函数。

◆封装优点：函数封装的优点在于简化程序以及增加程序的可读性，特别是可以减少重复出现语句的书写工作量。

◆如何进行封装：通过软件中的函数——》创建新函数，来实现封装。

具体的根据编程操作说明。

8、函数封装注意事项

◆如何访问函数：F7可以单步进入函数，F8可以直接跳出函数，F6运行时会直接把函数当成一个算子运行。

◆如何更改接口：当函数完成后，如果想进行接口的增加，则需要进入函数进行操作。

◆如何调用函数：当需要调用哪一个文件夹的函数时，需要将该文件夹加载进程序中。



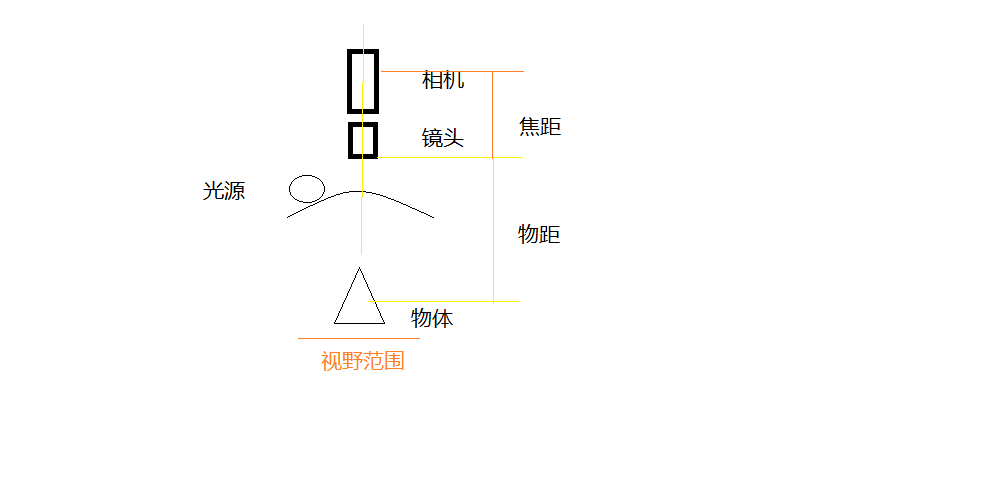
##### **第十章：halcon算法设计思路介绍**

1、视觉系统介绍

◆本节介绍：通过对整个系统的简单讲解，希望大家树立整个系统的逻辑关系。

◆视觉系统组成：完整的视觉系统是包括相机、镜头、光源、位置标定、图像算法几个部分，包括图像采集到结果反馈的整个过程，特别注意的是，图像的采集也十分重要。

◆位置关系介绍：



相机CCD芯片：相机里边实际成像的是CCD芯片，可以理解成是一块平面板，上边布满存储

点。

光源 算法： 七分靠光 三分算法

2、如何选择硬件

◆相机的选择：首先根据要采集物体的尺寸大小，比如面积为S，然后根据我们需要的分辨

率，比如我们要求精度为X 米/像素。那么图像的分辨率就可以据此进行计算，得到的

就是相机的长与宽的乘积。

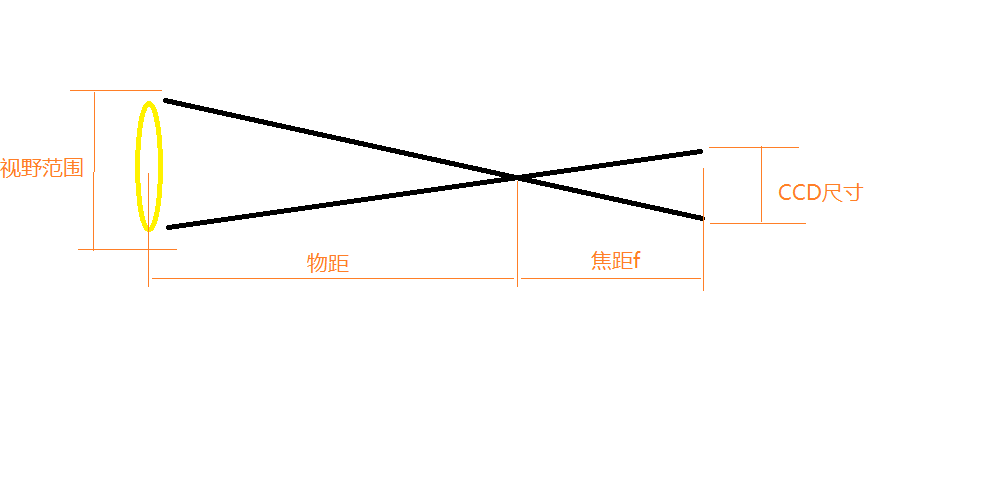
图像像素面积=S/X

据此可以进行像素面积的粗略估计，估计后便可以进行市场上相机的选择，根据分辨率而

定。同时考虑黑白还是彩色相机。

◆镜头的选择：镜头分为普通镜头与远心镜头，远心镜头没有畸变。选择的关键点是焦距计

算。



物距/视野范围=焦距f/CCD尺寸

34焦距 =35

镜头是标准

◆光源的选择：光源主要从光源颜色、尺寸大小以及位置设定几个进行考虑。光源的原则是

要尽可能的保证我的需求目标清楚，比如边缘、对比度等要保证。

光源选择根据实验制定，没有统一的规则。

3、算法设计的原则

◆算法设计原则：算法的设计要充分考虑效率（运行时间）、程序稳定性、程序适应性、程序的通用性、程序的精度等。

◆算法效率：算法要尽可能的运行速度快，才能够满足实际的使用情况。

（1）算法的运行区域ROI要尽可能的小 （多次定位）

（2）算法中尽量避免for循环的存在

（3）重复性的运算要减少

◆算法稳定性：算法在各种情况下要稳定，不能够崩溃。

（1）算法要尽可能测试实际存在的图像可能情况

（2）算法要能够接受用户操作带来的各种情况

◆算法适应性：算法要尽可能的适应多种情况。

（1）算法要考虑光照等环境的变化

（2）算法要能够接受相机等震动带来的精度损失

◆算法通用性：

（1）算法要考虑不同人的操作习惯

（2）算法要尽可能的通用，场景改变时算法依旧可以使用

◆算法精度：

（1）算法精度要能够适应实际的要求

（2）算法的精度应该能够稳定，在一个范围内波动

算子参数名称：英文名字

算子的注释

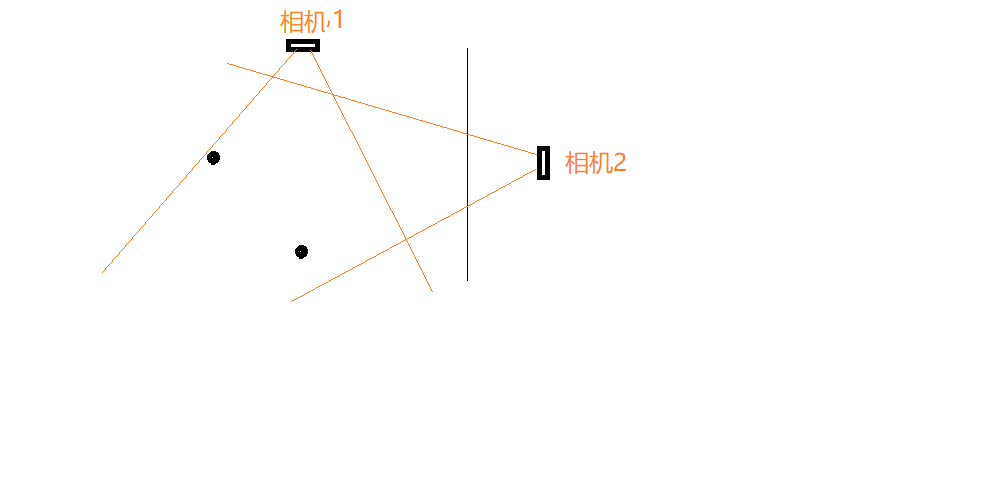
函数的封装：模块

4、定位抓取理论简单介绍

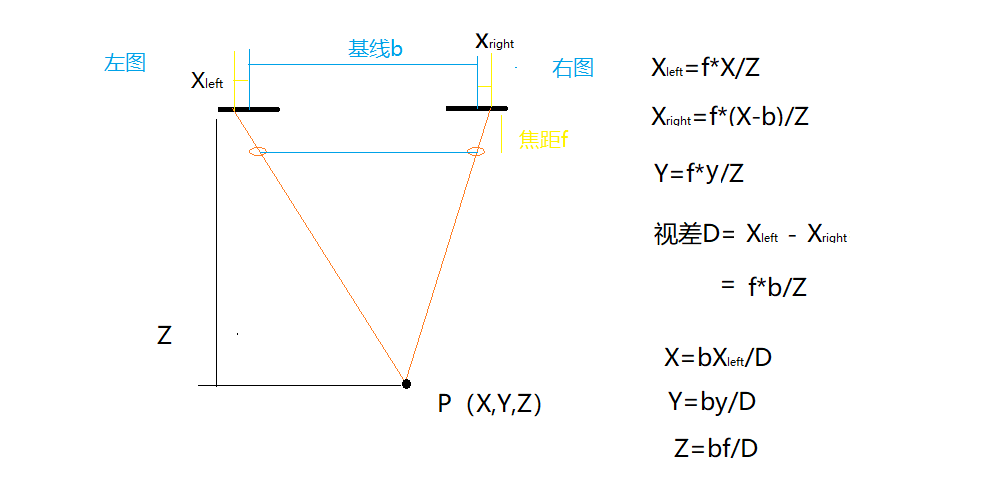
◆平面定位抓取：平面抓取是指只有两个方向坐标的移动，即x、y，水平方向上的移动，这种没有竖直方向移动的定位，可以通过阈值分割或者模板匹配的方法进行定位，获取物体移动的中心坐标，再与实际的位置进行换算即可。

◆空间定位抓取：空间定位抓取一般有两种方法进行定位，分别是通过不同位置相机采集图像定位以及双目立体视觉两种方案。

◆不同位置相机采集图像定位：



◆双目立体视觉：



5、图像算法总结

◆介绍：整个halcon算法部分的介绍到此已经结束，后边课时将介绍如何与c#的混合编程，因为最终的程序还是要实现工程化，而c#具有很好的工程应用。为了更好的去融会贯通以及检验自己的理解情况，安排了该节课，将对以前的课程进行总结。

◆图像预处理：图像预处理包括图像去噪、图像灰度变化、图像增强、图像滤波去噪等。图像预处理是为了使得图像满足我们的实际需求而已，有些时候需要去噪、增强都使用，有些时候需要进行图像滤波，而有些时候为了保证精度又不会使用，图像预处理往往不是一成不变的，不要盲目的去遵循这些步骤。比如白纸黑字，如果直接就能够提取，那就没有必要进行预处理增强了。

◆图像基本编程：图像中数组、矩阵、循环结构等，这些都是为编程所服务的，没有必要去刻意的记忆，当你编程多的时候，自然就记住了其中的规则。

◆图像分割：用于提取图像中的感兴趣区域。采用的方法有全局阈值分割、局部阈值分割、动态阈值分割，以及没有讲过的区域生长等方法。对于图像阈值的选择，自己完全可以有很大的改善空间，任何有依据的选择方法都可以进行尝试，只要能够满足效果以及稳定性即可。

◆图像形态学：图像形态学处理包括区域膨胀、区域腐蚀、区域开闭操作，以及没有讲过的顶帽、底帽变换，本质上就是一种根据区域形状制定的一种运算规则，更多的情况下我们是去追求一种效果、一种结果，要大体明白其运算方式。

◆图像边缘以及线：图像边缘以及线，都是图像中灰度发生明显变换的地方，通过制定的一些规则去寻找这些区域，往往都是通过模板运算的方式进行求取。要清楚这种运算方式，不仅对于自己的实际编程，对于以后解决问题也是一种帮助。

◆模板匹配：模板匹配在很多中情况下，是一种不错的解决问题的方法，包括ncc、形状模板匹配等。而在实际使用中，由于速度的问题，往往在考虑阈值分割无法实现的情况下，才会进行模板匹配。

◆分类器以及字符识别：无论分类器还是字符识别，都是通过一系列的特征去寻找规律，然后再利用这种规律去判别新的类型。在使用中，要尽可能的选择明显的规律特征，不然很容易寻找到的分界线并不明显，出现分类错误的情况。

以往视频中的算法，主要是挑取了常用的一些方法进行讲解，在实际的视觉算法中，图像

处理、机器学习以及深度学习，都有自己独特的特点以及丰富的理论内容，由于本人水平限

制，目前只能够提供这么多借鉴视频，谢谢。

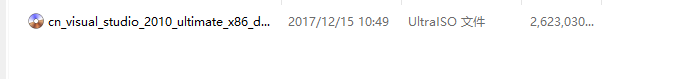
##### **第十一章：halcon与C#混合开发**

1. C#编程介绍及VS2010安装

◆C#介绍：C#是C sharp的称呼。C#是一种面向对象、运行在.Net框架上的高级程序设计语言，其安全、稳定、简单，属于c和c++的衍生，具有简单的可视化以及高运行效率。并且编程简单，工业应用较多。视频将主要介绍开发的流程，使用较多的是Visual Studio，课程将使用VS2010旗舰版版本，C#编程部分将主要针对界面编程进行讲解。

◆VS2010软件安装介绍：

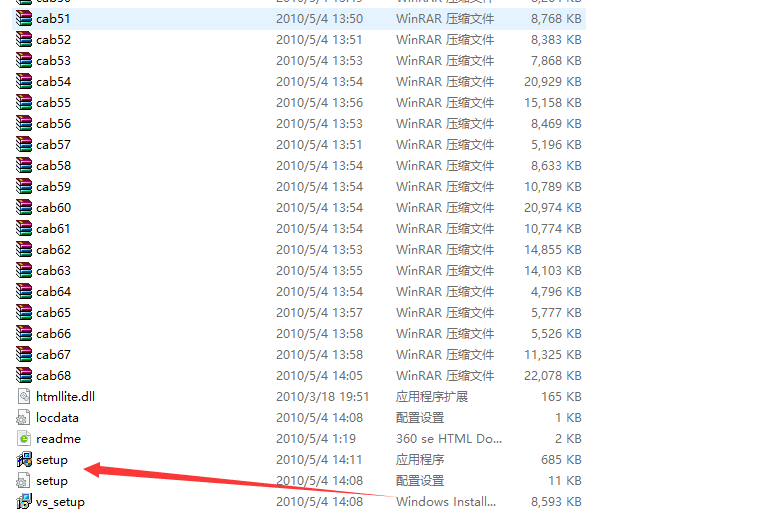
软件安装包：

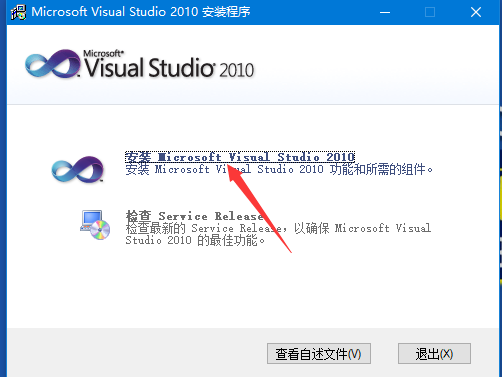


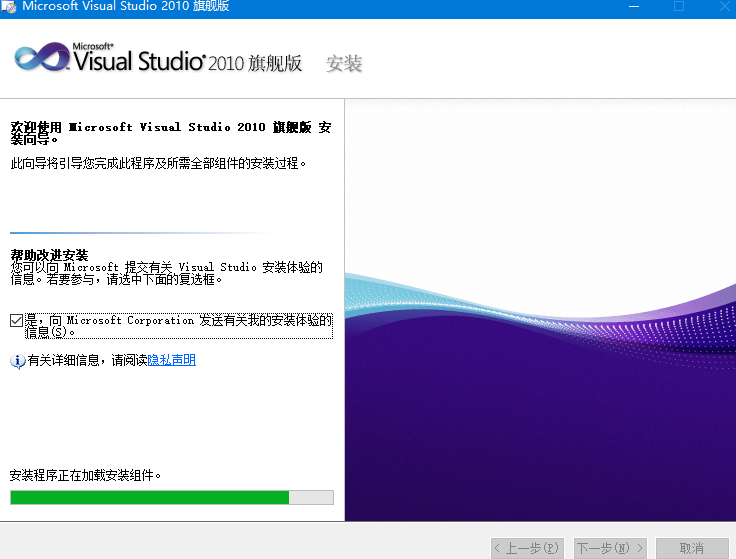
鼠标右键单击——>解压

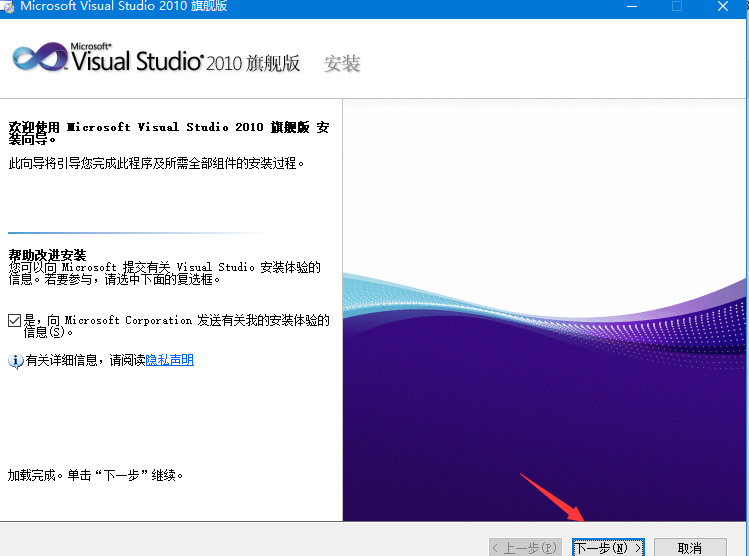


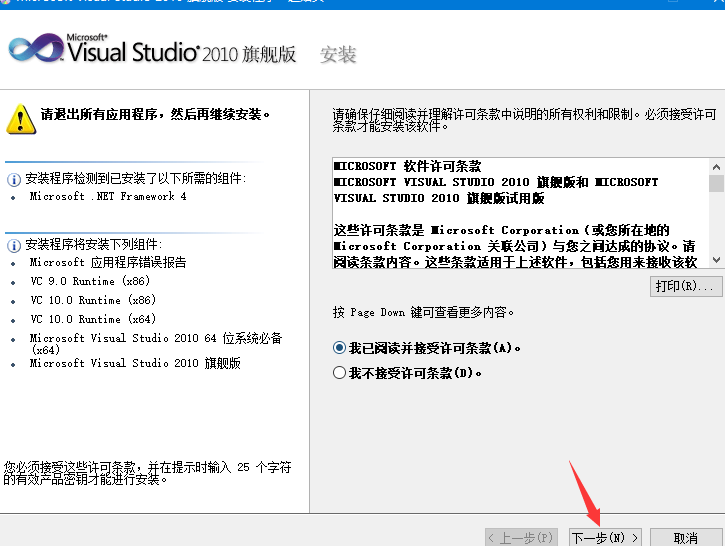
解压后的软件包：

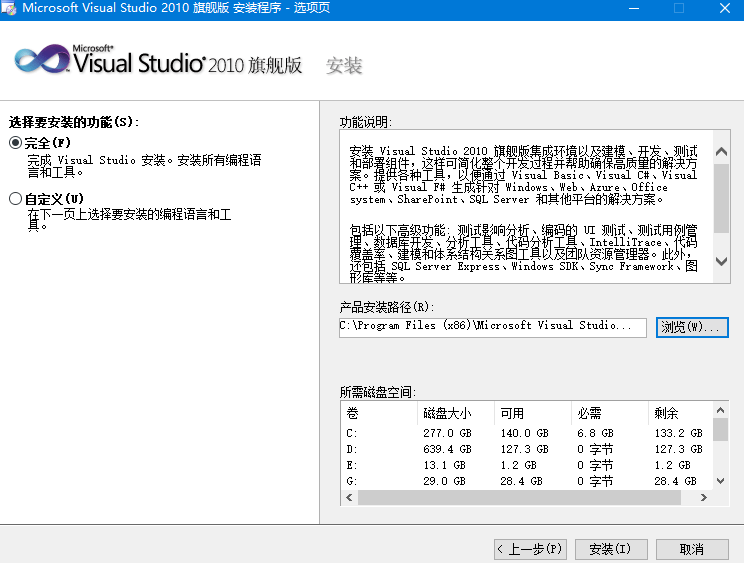


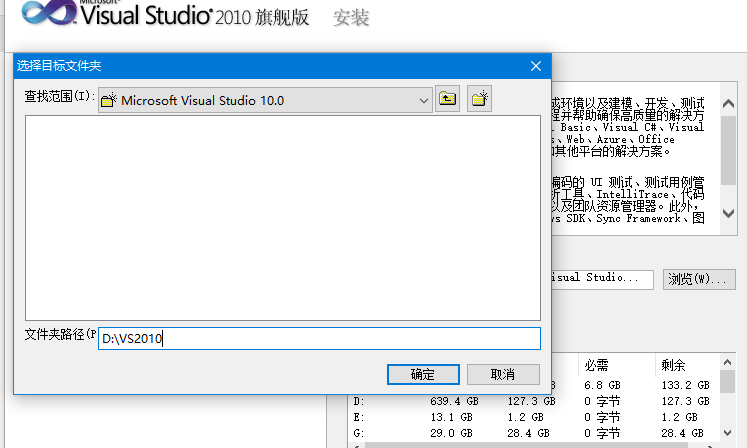


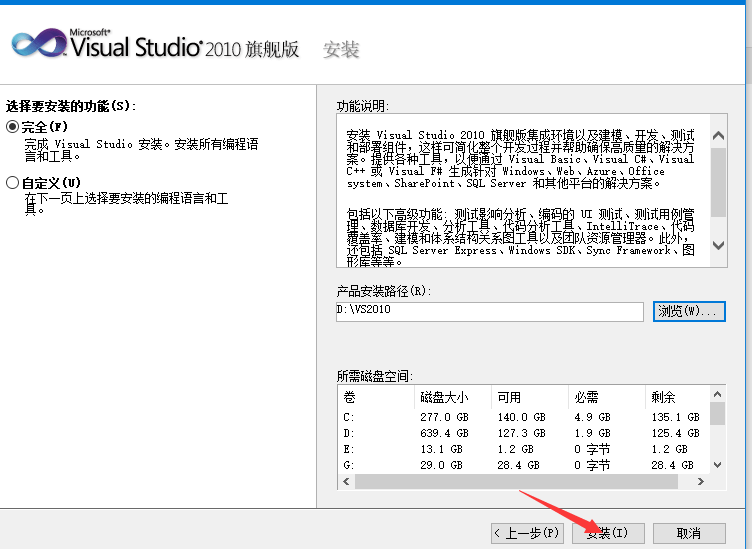




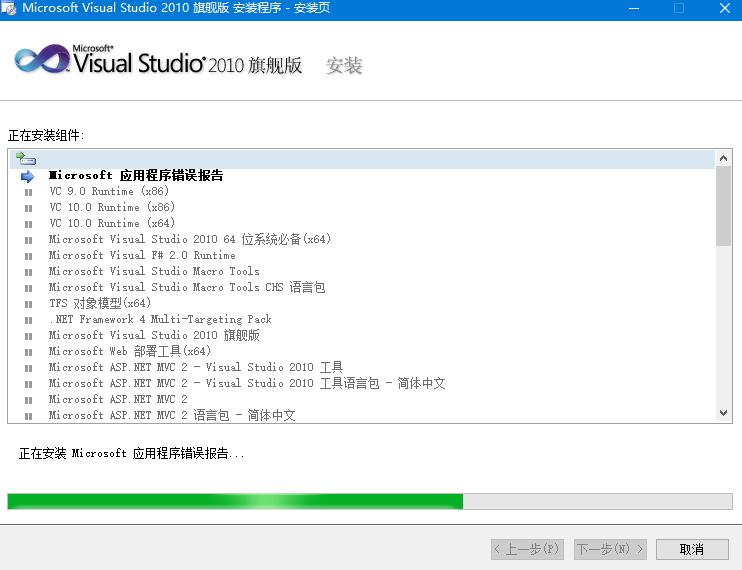






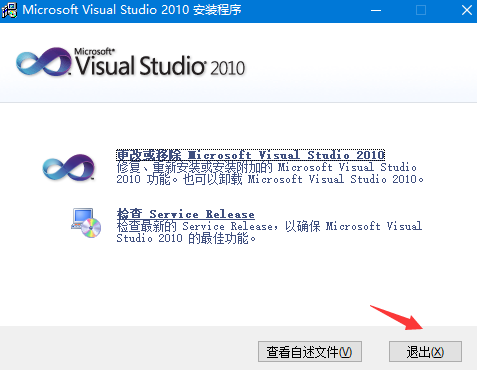


安装等待，直到其安装完成，即可。

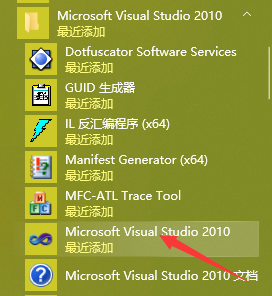


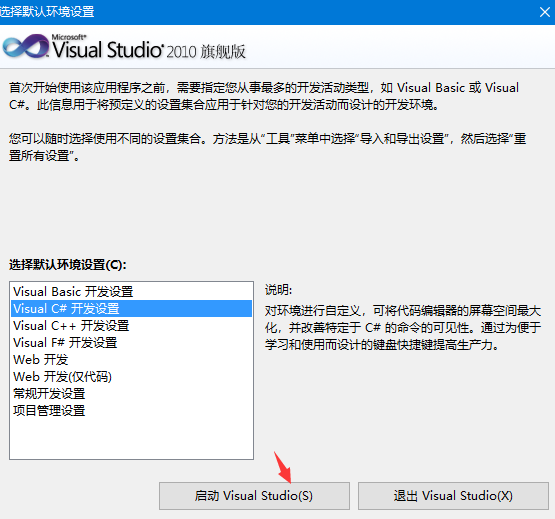
大概30分钟后，安装完成后会出现下面的界面，点击完成即可。

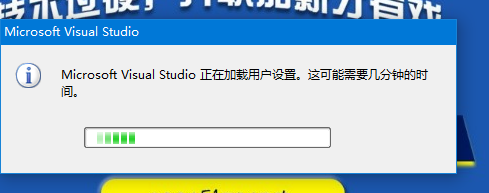




完成后，如果没有显示桌面快捷键，在所有程序中进行寻找，以M开始。







1. Winform应用程序介绍（1）

◆介绍：Winform应用程序就是基于界面编程，也是相对简单的编程方式。

我们将通过创建应用程序来详细的介绍

1. Winform应用程序介绍（2）
2. halcon程序导出及c#环境配置（1）

◆介绍：前边三节属于混合编程的必备储备知识，但是并没有讲明两者之间如何混合开发，本节以后的内容将正式介绍如何去进行环境的配置等。

5、halcon程序导出及C#环境配置(2)

1. 读取文件及显示设置

7、混合案例-车牌识别(1)

◆介绍：本节将利用8-5中的车牌识别算法导出为c#程序进行编程，然后不断深化，实现车牌的识别、显示等。

8、混合案例-车牌识别(2)

9、混合案例-车牌识别(3)

10、混合案例-车牌识别(4）