# 一维测量

## 第一节 第一个实例（略）

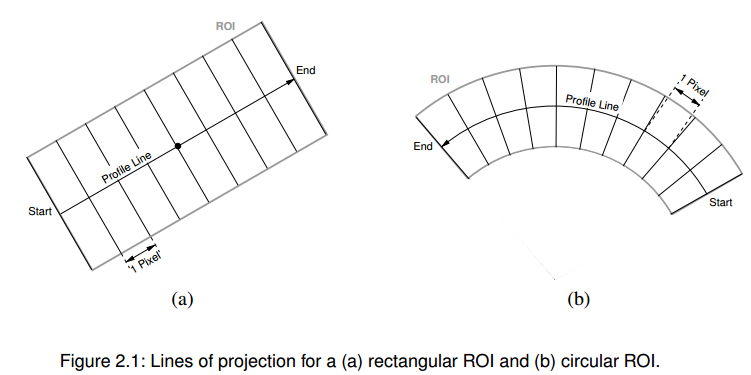
## 第二节 测量基础

### 2.1 HALCON的测量步骤：

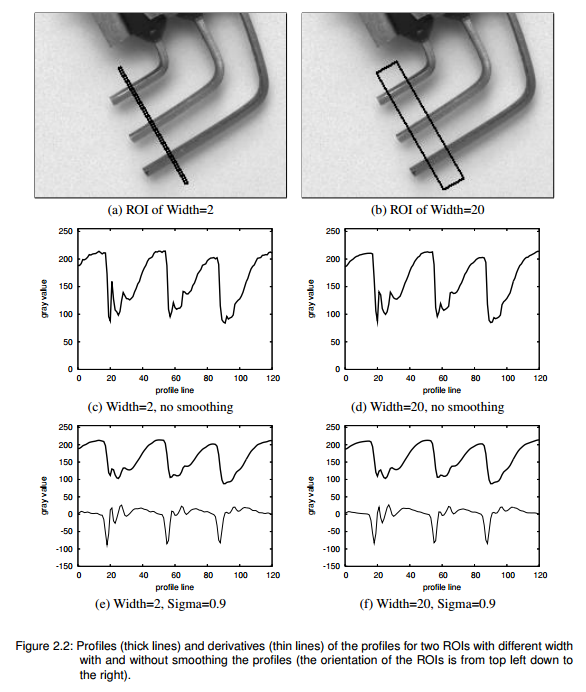
第一步，生成一系列与ROI等宽的，垂直于截面轮廓线的投影直线。

第二步，求取这些投影直线上平均的灰度值，这一平均灰度值的序列称为轮廓(profile)，如果投影线的方向不是水平或者垂直的，那么在投影上上的像素的灰度值由插值得到。用户可以在用gen\_measure\_rectangle2算子生成测量矩形的时候选择插值模式，例如：Interpolation = ’nearest\_neighbor’对应邻近插值，Interpolation = ’bilinear’对应线性插值，Interpolation = ’bicubic’对应双三次插值，虽然双三次插值可以得到最精确的结果，但它的速度是最慢的。

投影线的长度，即ROI的宽度，决定了截面轮廓线上平均灰度值。在图2.2中，粗线表示截面轮廓线上平均的灰度值，由图可知，较宽的ROI，所得到的平均灰度值噪声较少，所以，在截面轮廓线垂直于边界的情况下，ROI的宽度越宽越好。在截面轮廓不垂直于边界的情况下，ROI的宽度应越小越好。注意，在这种情况下，轮廓线会夹杂更多的噪声，因此，边缘检测的精度性会降低。

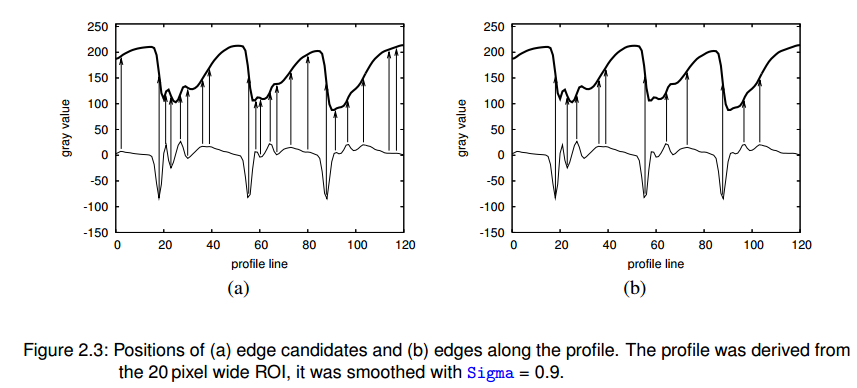


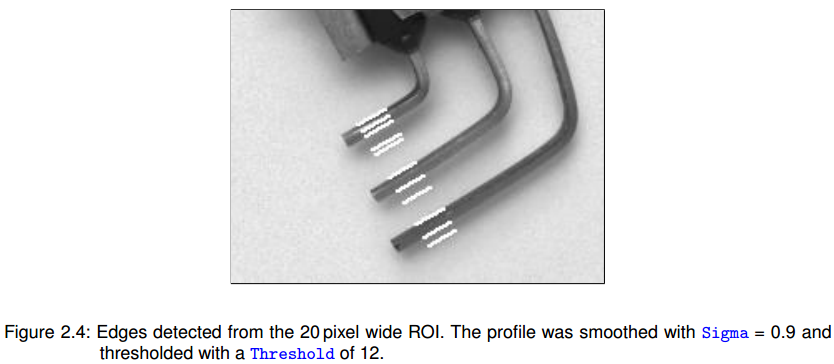
第三步，用高斯滤波器对轮廓线进行平滑，高斯滤波器的标准差为测量算子，如measure\_pairs，指定的参数Sigma。然后对平滑后的轮廓线求一阶导数，在图2.2中用细线表示。注意，这里导数的幅度值被放大了，亚像素精度的一阶导数极值，都是候选边缘。如图2.3所示。只有那些一阶导数绝对值大于一个给定阀值的候选边缘才能被认为是边缘。



对于每一条边缘，其与截面轮廓交点位置被返回。如图2.4。

注意，以上对一维边缘的提取方法与二维边缘的提取方法有很大不同，在一维边缘提取中，中有边缘与截面轮廓线的交点被返回，而二维边缘提取则将完整的边缘作为轮廓返回。如果边缘是直的，且与截面轮廓线大致垂直，其结果与二维边缘提取相似。如果边缘越弯曲，或越不垂直于截面轮廓线，那么结果越不相同。





### 2.2 编程关键步骤

1、创建Measure Object：gen\_measure\_rectangle2或gen\_measure\_arc

2、测量:measure\_pairs 或measure\_pos

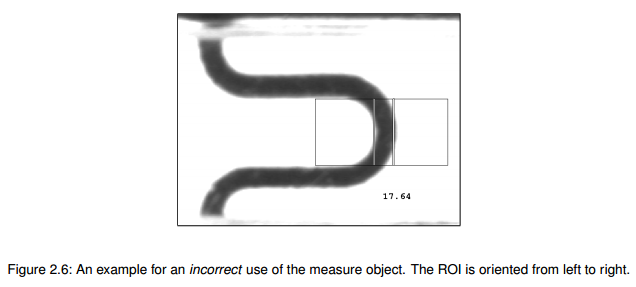
3、显示结果

4、销毁句柄

### 2.3 创建一个Measure Object

Halcon可以创建两种不同形状的Measure Object：旋转矩形和圆弧，分别对应两处算子：gen\_measure\_rectangle2和gen\_measure\_arc。

应记住，Measure Object是用来测量直线边缘的，而不能适用于测量曲线边缘，如图2.6所示的情况。



两种Measure Object创建方法如下：

矩形：

gen\_measure\_rectangle2 (Row, Column, Phi, Length1, Length2, Width, Height, \

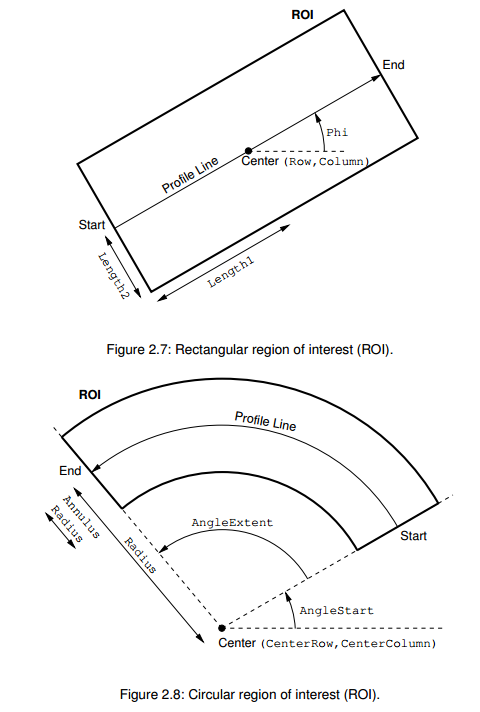
Interpolation, MeasureHandle)

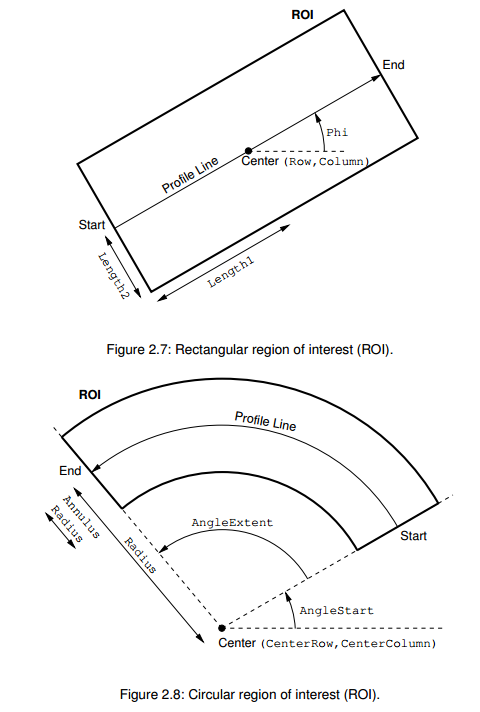
圆弧：

gen\_measure\_arc (CenterRow, CenterCol, Radius, AngleStart, AngleExtent, \

AnnulusRadius, Width, Height, Interpolation, MeasureHandle)

其参数的意义如图2.7和2.8所示。





## 第三节 使用Measure Object

### 3.1 边缘位置

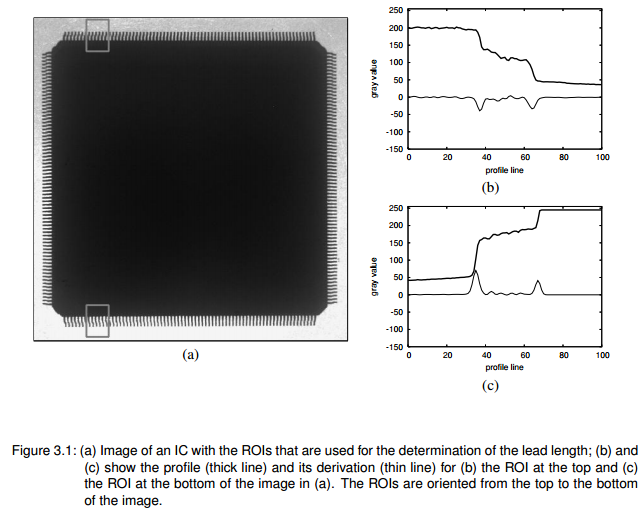
测量边缘的位置可用measure\_pos算子。

例程：solution\_guide/1d\_measuring/measure\_ic\_leads.hdev。该例程中的任务是测量IC芯片引脚的长度。由于引脚很线，两端没有明显的两条边缘，因而看起来该任务很难完成。在这里，我们可以设置一个较宽的ROI矩形，使其包括多根引脚，在图3.1a中，引脚的平均灰度值比背景暗，而比IC苾片主体亮，因此可以用如下的方法进行边缘提取，相应的代码如下。

在图像上方创建一个宽度包括几根引脚的测量矩形：

gen\_measure\_rectangle2 (Row, Column, Phi, Length1, Length2, Width, Height, \

Interpolation, MeasureHandle)



提取所有垂直于ROI的边缘：

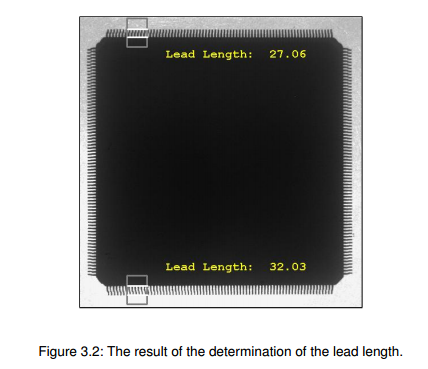
measure\_pos (Image, MeasureHandle, Sigma, Threshold, Transition, Select, \

RowEdge, ColumnEdge, Amplitude, Distance)

测量的结果返回到Distance参数中，由于这里只有两条边缘被提取出来，因此Distance参数在这里只有一个元素，这个距离即图像上方引脚的长度。

对于图像下方的引脚，同样可以采用这样的测量方法。在这里用到了translate\_measure算子。该算子用法是这样的：当多次测量区域的大小和方向相同时，可以在第一次用gen\_measure\_rectange2生成一个测量矩形，而在接下来的测量中可以用translate\_measure将测量矩形移动到要测量的区域，而不必在次创建一个新的测量矩形。

Sigama参数决定了轮廓线的平滑程度，Treshhold是边缘阀值，参数Transition表示极性，取’negative’时表示由白到灰，’positive’时表示由灰到白。在这里为了检测所有类型的边缘，这里的取值为’all‘。



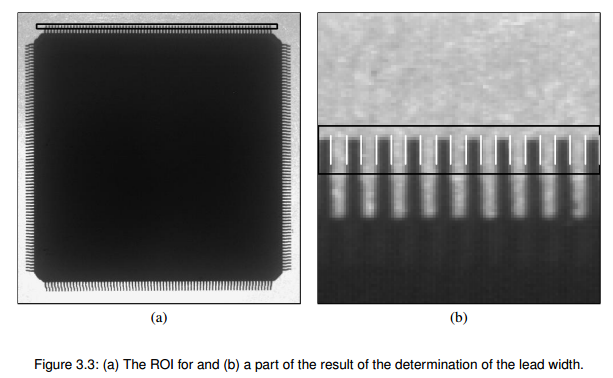
除了边缘距离之外，measure\_pos还返回边缘的位置，下面的代码使用其输出的位置参数，将边缘绘制出来。

disp\_line (WindowHandle, RowEdge, ColumnEdge-Length2, RowEdge, ColumnEdge+Length2)

### 3.2 边缘对的位置

例程：solution\_guide/1d\_measuring/measure\_ic\_leads.hdev。此次的任务是测量IC引脚的间隙宽度。

首先确定一个垂直于引脚的ROI矩形，这里方向为从左到右。按上一节的方法，创建一个Measure Object。使用measure\_pairs算子，提取边缘对。



measure\_pairs (Image, MeasureHandle, Sigma, Threshold, Transition, Select, \

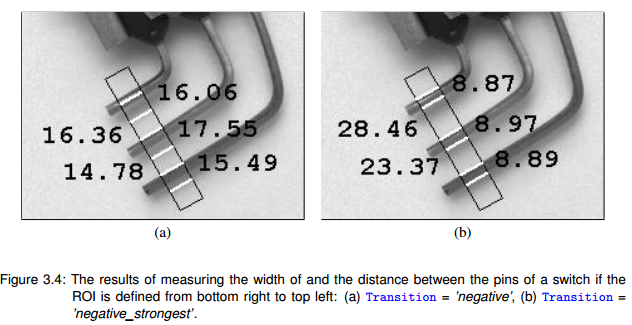
RowEdgeFirst, ColumnEdgeFirst, AmplitudeFirst, \

RowEdgeSecond, ColumnEdgeSecond, AmplitudeSecond, \

IntraDistance, InterDistance)

测量结果如图3.3b中，这里只显示了部分图像。measure\_pairs参数与measure\_pos类似，不在赘述。如果不止一条具有相同极性的边缘被查找到，则第一条边缘被作为边缘对元素。如下图中，检测到两条极性相同的边缘。





对于这种情况，可以将Transition参数选择为形如’\_strongest‘的参数值，如’negative\_strongest’。

### 3.3特定灰度值点的位置

Measure\_thresh算子可以轮廓线上给定灰度值得到亚像素精度位置，如同在二维亚像素阈值算子threshold\_sub\_pix。

该算子可以用于提取那些光照条件精确控制的应用中的边缘。注意，测量的结果严重依赖于光照的条件，如果光照条件不同，测量的结果也不一样。

### 3.4 灰度投影

在一些应用中，可以通过measure\_projection算子直接得到轮廓线上的灰度值序列。它返回的灰度值是没有经过高斯平滑，原始的灰度值，保存在GrayValues中。

例程：solution\_guide/1d\_measuring/measure\_caliper.hdev，该程序是测量游标卡尺上的刻度。

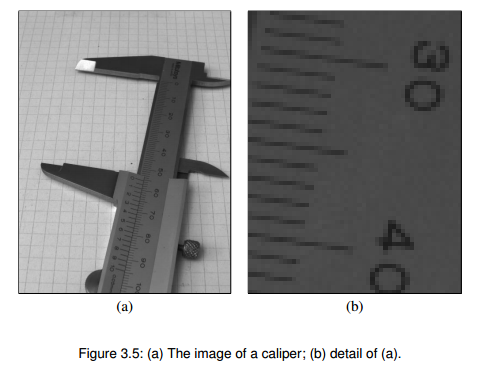
首先创建一个测量矩形：

gen\_measure\_rectangle2 (Row, Column, Phi, Length1, Length2, Width, Height, \

‘bilinear’, MeasureHandle)

然后，使用measure\_projection来提取截面轮廓线上的灰度值序列：

measure\_projection (Image, MeasureHandle, GrayValues)



为减小噪声，对灰度值序列进行平滑处理：

Sigma := 0.3

create\_funct\_1d\_array (GrayValues, Function)

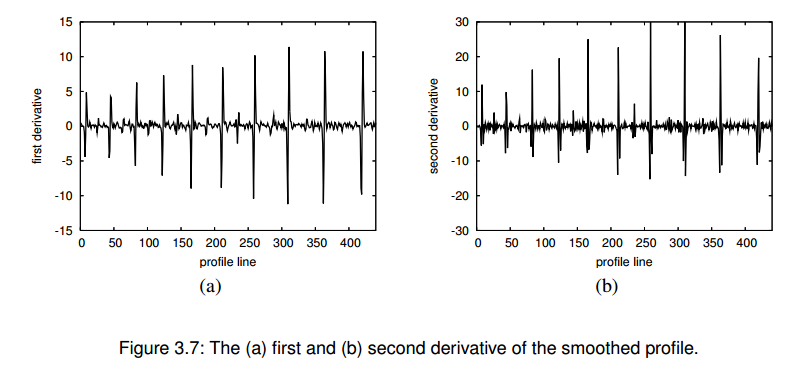
smooth\_funct\_1d\_gauss (Function, Sigma, SmoothedFunction)

其中，create\_funct\_1d\_array作用是根据GrayValues中的值，来创建一个函数。

对平滑后的函数求一阶导数，并求一阶导数为0的点位置：

derivate\_funct\_1d (SmoothedFunction, ✬first✬, FirstDerivative)

zero\_crossings\_funct\_1d (FirstDerivative, ZeroCrossings)



求二阶导数：

derivate\_funct\_1d (SmoothedFunction, ✬second✬, SecondDerivative)

在我们的应用中，刻度是在白色背景下的黑线，因此，我们必须选择那些二阶导数具有较大正值的直线位置。对于每一个直线位置即一阶导数为0的点，其二阶导数可以由get\_y\_value\_funct\_1d得到。如果这个值大于用户给定的阀值，那么它将会被保存在一个tuple数组中。

MinimumMagnitudeOfSecondDerivative := 8

PositionOfSalientLine := []

for i := 0 to |ZeroCrossings|-1 by 1

get\_y\_value\_funct\_1d (SecondDerivative, ZeroCrossings[i], ✬constant✬, Y)

if (Y > MinimumMagnitudeOfSecondDerivative)

PositionOfSalientLine := [PositionOfSalientLine,ZeroCrossings[i]]

endif

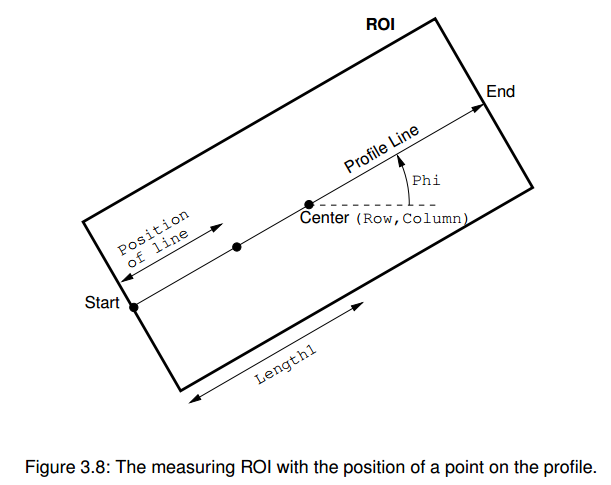
endfor

最后，所有的直线位置都应该被转变成像素坐标，因为measure\_projection只是返回轮廓线上的灰度。另外，必须对这些坐标进行圆整。

求于轮廓线的起始坐标：

RowStart := floor(Row+0.5)+floor(Length1)\*sin(Phi)

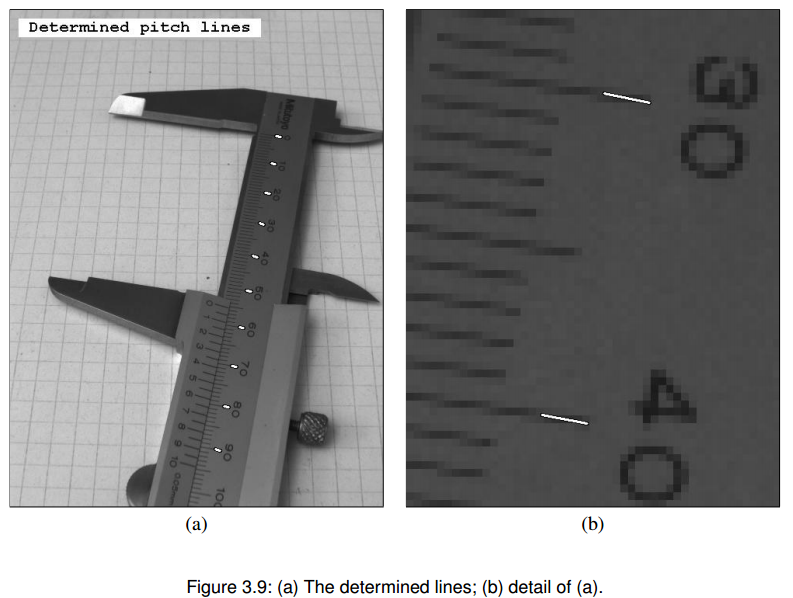
ColStart := floor(Column+0.5)-floor(Length1)\*cos(Phi)



求取直线的像素坐标：

RowLine := RowStart-PositionOfSalientLine\*sin(Phi)

ColLine := ColStart+PositionOfSalientLine\*cos(Phi)



### 3.5 在世界坐标中进行测量

在许多应用中，测量的结果应该在世界坐标中表示，例如µm。这可以由以下两种方法得到：将测量结果转换到世界坐标中，或先较正图像，再进行测量。作为先决条件，相机必须是被标定好的。

在本例中，用image\_points\_to\_world\_plane算子，实现坐标的转换。

image\_points\_to\_world\_plane (CamParam, WorldPose, RowLine, ColLine, ✬mm✬, X, Y)

其中参数CamParam包括了相机参数如相机的位置和方向，测量平面等。这些参数在相机标定过程中得到，在本例中，假设这些参数是已知的。

现在，可以用distance\_pp来计算世界坐标下，两点间的距离：

Num := |X|

distance\_pp (X[0:Num-2], Y[0:Num-2], X[1:Num-1], Y[1:Num-1], Distance)

### 3.6 镜头的畸变补偿

如果镜头畸变较大，有必要先较正图像。

### 3.7 改变Measure Object的测量的位置和方向

我们经常需要测量不同图像上同一类物体的位置，这种情况下，ROI区域必须能够自动适应，这个过程，我们称作对齐(alignment)。

物体的位置和方向可以由基于形状匹配来查找确定。如果ROI区域只有位置发生变化，可以使用translate\_measure来改变Measure Object的位置。如果其方向也发生变化，需要修改参数并以新的参数重新创建一个Measure Object。

### 3.8 其它事项及建议

#### 3.8.1 距离和圆弧ROI

测量弧形长度有三种度量：弧长、直线距离、角度。

gen\_measure\_arc (CenterRow, CenterCol, Radius, AngleStart, AngleExtent, \

AnnulusRadius, Width, Height, Interpolation, MeasureHandle)

measure\_pairs (Image, MeasureHandle, Sigma, Threshold, Transition, Select, \

RowEdgeFirst, ColumnEdgeFirst, AmplitudeFirst, RowEdgeSecond, \

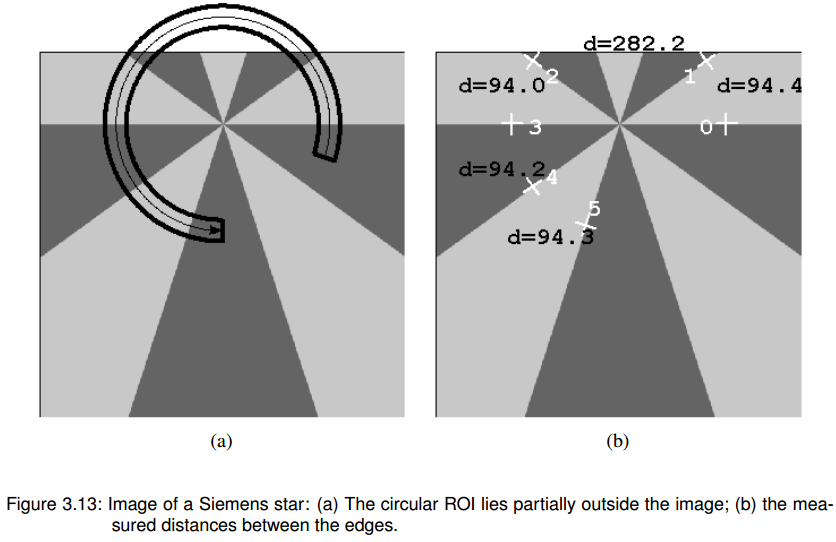
ColumnEdgeSecond, AmplitudeSecond, IntraDistance, InterDistance)

measure\_pairs返回的参数IntraDistance保存的是弧长，直线距离由distance\_pp得到，角度可以以下公式得到：

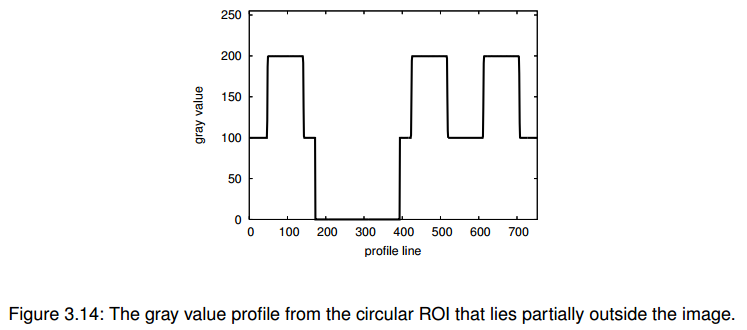
代码：AngularDistance := deg(IntraDistance/Radius)

#### 3.8.2 ROIs部分位于图像之外的情况

如果部分ROIs区域在图像之外，图像内部的边缘距离仍然能够正确得到。



注意，灰度为0的位置也能够得到，在图像之外，这些位置的灰度值为0。



### 3.9 销毁Measure Object

代码：

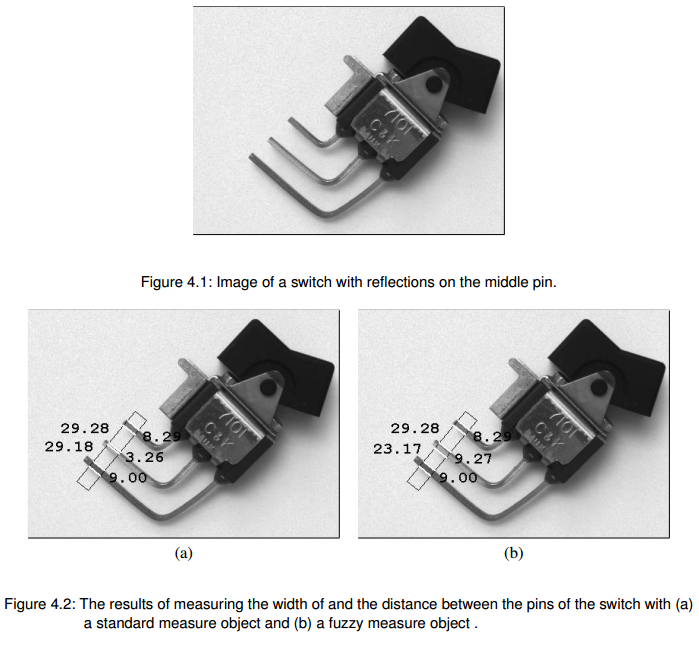
close\_measure (MeasureHandle)

## 第四节 高级测量：Fuzzy Measure Object

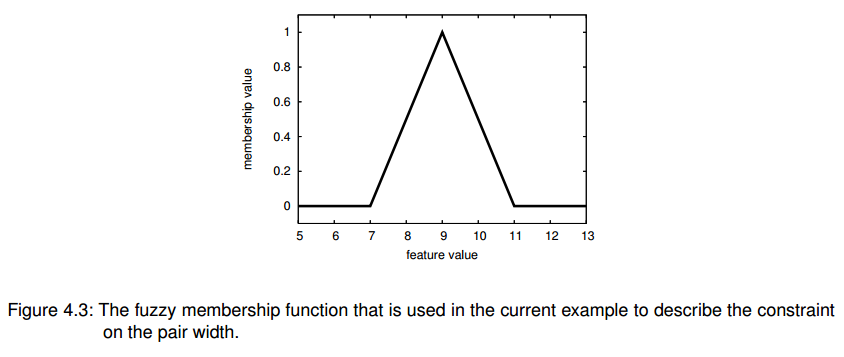
### 4.1 第一个实例

例程：solution\_guide/1d\_measuring/fuzzy\_measure\_switch.hdev

该实例对比了标准测量和模糊测量两种测量方法对图中开关引脚的测量结果，由于开关引脚的反光效果，使得标准测量中对中间引脚的测量不准确。



采用模糊测量。注意到引脚的宽度大致为9个像素，采用以下模糊关系函数：



该函数由create\_funct\_1d\_pairs算子获得：

create\_funct\_1d\_pairs ([7,9,11], [0.0,1.0,0.0], SizeFunction)

标准Measure Object 向Fuzzy Measure Object的转变可以用以下算子set\_fuzzy\_measure：

SetType := ✬size✬

set\_fuzzy\_measure (MeasureHandle, SetType, SizeFunction)

这样，Fuzzy measure Object可以提取出距离大概为9个像素宽的边缘对，这个“大概”程度由模糊关系函数来决定。

以下代码提出去想要的边缘对：

Sigma := 0.9

AmpThresh := 12

FuzzyThresh := 0.5

Transition := ✬negative✬

Select := ✬all✬

fuzzy\_measure\_pairs (Image, MeasureHandle, Sigma, AmpThresh, FuzzyThresh, \

Transition, RowEdgeFirst, ColumnEdgeFirst, AmplitudeFirst,RowEdgeSecond,ColumnEdgeSecond, \

AmplitudeSecond, RowEdgeCenter, ColumnEdgeCenter, FuzzyScore, IntraDistance, InterDistance)

指定的模糊关系函数用来决定各可能的边缘对的权重，边缘对的宽度被转换成关系值，如果其关系值大于给定的阀值，那么该边缘对被接受为边缘。

### 4.2 基本步骤：

1、创建测量矩形 gen\_measure\_rectangle2

2、创建模糊函数 create\_funct\_1d\_pairs

3、设置Fuzzy Measure Object，为其添加约束：set\_fuzzy\_measure或set\_fuzzy\_measure\_norm\_pair。

4、设置模糊关系阀值并测量：fuzzy\_measure\_pairs或fuzzy\_measure\_pos或fuzzy\_measure\_pairing。

5、显示测量结果

6、销毁测量句柄。

说明：

fuzzy\_measure\_pairing是fuzzy\_measure\_pairs的扩展，前者比后都多两个参数，Pairing和NumPairs，现在Pairing只能取'no\_restriction'，即所有可能的边缘对都会被返回，NumPairs参数由用户指定，得分最高的前NumPairs项边缘对被返回，如果为0，表示所有可以的边缘对会被返回。

用归一化的模糊关系函数的一个优势是，对于大小变化的物体，只要少许修改，便可轻易应用到不同的实例中，例如，测量较大的物体，或是使用较大分辨率的相机。

### 4.4 关于模糊关系函数

模糊关系函数是不确定程序的描述，在此不详细叙述。

### 4.4 模糊类型参数

详细说明及图例见手册。

### 4.5 指定多种模糊关系函数

用户可以多次调用set\_fuzzy\_measure来指定多种模糊关系函数，例如：

set\_fuzzy\_measure (MeasureHandle, ✬contrast✬, FuzzyMembershipFunction1)

set\_fuzzy\_measure (MeasureHandle, ✬position✬, FuzzyMembershipFunction2)

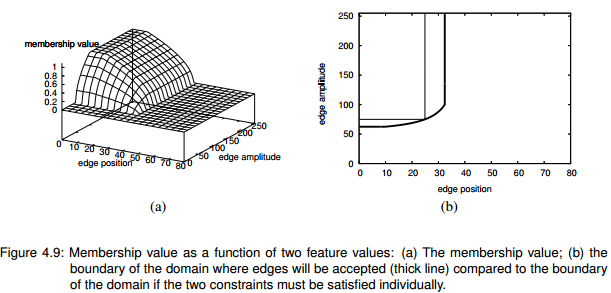
其综合的模糊关系值为各模糊关系值的几何平均，例如：

membership value\_position = 20 = 0.67

membership value\_contrast = 80 = 0.60

membership value\_20/80 = = 0.63

图4.9b中，粗线围成的面积是以0.5为阀值，综合模糊关系函数可接收的边缘参数所在范围，而细线区域是将特征分离来看，各特征参数可以接收的边缘参数范围，由图可见，前都的范围比后者大。



### 4.6 修改和重置Fuzzy Measure Object

要改变模糊关系中的类型，可以简单的为该类型指定一个新的模糊关系函数。重置可Fuzzy Measure Object可用reset\_fuzzy\_measure。