# 曲线高斯平滑（GaussianSmoothXY）

# 算法概述

## 算法功能

对指定曲线curve，根据输入的smWindowSize（平滑窗口大小），对其进行高斯平滑操作，以达到消除曲线毛刺的作用。平滑结果保存在curve的成员变量gSmCordiXYCurve数组中。

## 算法思路

高斯曲线平滑使用一点附近相邻点坐标加权平均来更新现有点坐标。

本算法利用二项式系数在离散域可以很精确地代替Gaussian filter的特点，在程序外预先计算出特定的Gaussian filter系数直接代入使用，而不需在算法运行过程中计算权值以提升运行效率。

算法并行策略为每个Curve上的点对应一个thread，由于曲线为一维数据，所以采用多个block，每个block内有32\*1个thread的并行策略。

# 算法接口

算法所使用的接口函数以及函数参数说明如下：

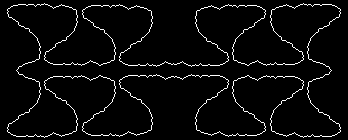
|  |
| --- |
| \_\_host\_\_ int gaussSmoothCurveXY( //返回值为算法是否成功执行，0表示成功，//否则表示出错  Curve \*curve, // 待平滑曲线  int smWindowSize // 平滑窗口大小,可以为3、5、7、9、11  // 窗口越大平滑程度越高  ) ; |

下面为实际使用代码示例：

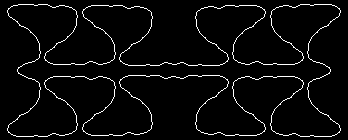
|  |
| --- |
| // Curve的创建、获取等代码省略。  // 创建高斯平滑对象  GaussSmoothXY gaussxy;  // 执行平滑窗口尺寸为9的高斯平滑。  gaussxy.gaussSmoothCurveXY(curve, 9); |

# 算法执行效果

原始图像：



平滑结果（平滑窗口大小：11）：



### 附件：算法执行时间

1. 平滑窗口大小对执行时间的影响。对上面实例中的curve分别以平滑窗口3、5、7、9、11行时间测试，计算100次运行时间，做出其概率密度累积图表，利用该表可直观的看出算法的性能。

可以看出不同平滑窗口大小对于执行速度基本没有影响

1. 不同curve长度对执行时间的影响。