

涉密论文 ☐ 公开论文 ☐

浙江大學

本科生毕业论文



题目 三次样条表示的殷集的布尔代数

姓名与学号 黄瞭望 3140101938

指导教师 张庆海

年级与专业 信息与计算科学 1401

所在学院 数学科学学院

提交日期

致 谢

感谢我的导师张庆海老师在这几个学期对我的指导。认识张老师一年多来，他严肃的科学态度，严谨的治学精神，精益求精的工作作风，深深地感染和激励着我，我也一直视张老师为学术上的榜样。这一年来从张老师所授的课程和指导中收获到的不仅仅是知识，更重要的是对待知识的态度和学习的方法。

感谢我的父母给予我的无微不至的关怀和鼓励。

感谢黎至轩同学完成的相关工作。

感谢林港回、赵鑫安和我讨论写论文中出现的问题以及互相监督。

感谢数学科学学院的每一个老师和同学的帮助和支持。

最后非常感谢浙江大学，能在浙大学习对我来说是非常幸运的事，在浙大的四年是我最开心的四年，学到了很多知识，认识了很多的人。希望在离开浙大以后，能不忘初心，继续前行。

摘 要

用殷集来表示物理实体是一种新的对物体建立实体模型的方法。其有广泛的实用性以及严格的数学理论支撑，且设计到的数据结构较为简单易懂。多边形的布尔运算在许多领域上有广泛的应用。殷空间上的布尔代数毫无疑问能有许多应用。直线段的扫描线求交点算法是一种输出敏感的高效率算法，将其运用到殷空间的布尔代数上能提升程序的效率。本次的任务是基于直线段表示的殷集的布尔代数的C++代码完成MATLAB代码，同时将直线段升级为三次样条曲线段。

关键字：殷空间、布尔代数、扫描线算法、三次样条

Abstract

A novel method to establish the solid model of an object is to use the Yin sets to represent the object. The Yin set theory has a wide range of practicability and solid mathematical theory support, and the data structure is relatively easier to understand. The boolean operations of polygon is widely used in many fields, and the boolean operations of Yin sets is sure to have many chances to show its power. The sweep line algorithm is an efficient out-put sensitive method to compute the intersections among a set of segments, the use of which in boolean operations of Yin set is sure to improve the efficiency of the program. The task is to write the MATLAB routines based on the C++ code for boolean operations of Yin sets represented by line segments, and in the meantime, update the line segments into cubic spline.

Key words: Yin space, Boolean operations, sweep line algorithm, cubic spline

目 录

1 绪论	1
1.1 背景介绍.....	1
1.1.1 殷集的意义	1
1.1.2 殷空间上的布尔代数	2
1.2 研究意义和目的	3
2 算法分析	4
2.1 带有不确定性量化的扫描线算法	4
2.2 三次样条曲线段的扫描线算法	5
2.3 三次样条表示的殷集的布尔代数	8
3 MATLAB 程序	9
3.1 类的设计	10
3.2 输入与输出.....	11
4 测试结果	11
4.1 交运算.....	12
4.1.1 两个边界重合的殷集	12
4.1.2 边界分离的殷集	12
4.1.3 improper 交点.....	13
4.1.4 proper 交点.....	13
4.2 其它布尔运算	13
参考文献	15

第一部分

毕业论文（设计）

1 绪论

1.1 背景介绍

1.1.1 殷集的意义

在解决实际问题时经常会遇到对现实物理实体建立数学模型。计算机图形学、计算几何、计算机辅助设计、地理信息系统等学科都会有应用。而最开始使用线框模型来进行建模，线框模型 (Wireframe Model) 多用多边形或多面体 (点、线、面) 来表示实际物体。但是这样只能保存实体的几何特征，比如维数、线段长度、角度和面积。显然仅仅这样是不够的，我们还需要有实体的拓扑特征，比如连通性以及与邻近物体的关系。实体模型 (Solid Model) 则记录了物体的几何信息和拓扑信息，它能更清晰和完整地表示物体 [1]。实体模型要求任何实体是欧几里得空间中有界的正则半解析闭集。而表示由布尔运算产生的实体的正则点集称为 **r-sets** (正则集)。在建立了实体模型后经常会涉及到对模型的各种操作，比如布尔运算。现在有一种新的对现实物理实体建模的方法，即用殷集 [3] 来表示二维平面上的物理实体，并建立了相应的布尔运算。主要想把殷集运用于界面追踪等问题上。

殷集是二维欧式空间上边界有界的半解析正则开集。而所有的殷集形成殷空间。不同于 **r-sets** 中使用有界集，殷集放宽到边界有界。显然这样能描述更多的情况，而且实际中也不会出现边界无界的实体。使用正则集则是要捕捉物理实体的特征，在二维情况中免除孤立点和孤立曲线的影响，并且正则集在正则化的集合运算下能保持正则性和半解析性。至于选用半解析集，则是要排除无穷震荡的情况，因为在实际中是不会出现这种情况。另外任意集合我们都需要能在计算机上有限表示。例如，考虑

$$\begin{cases} \mathcal{A}_p := \{(x, y) \in \mathbb{R}^2 : -2 < y < \sin \frac{1}{x}, 0 < x < 1\}, \\ \mathcal{A}_s := \{(x, y) \in \mathbb{R}^2 : 0 < y < 1, -1 < x < 1\}. \end{cases}$$

虽然 \mathcal{A}_p 和 \mathcal{A}_s 是由两个不等式描述的，但是它们的交是无穷个不相交的正则集的并，并不能有限表示。 $f(x) = \sin(1/x)$ 的图像见图 1.1。使用半解析集合能将这样的情形排除在外。

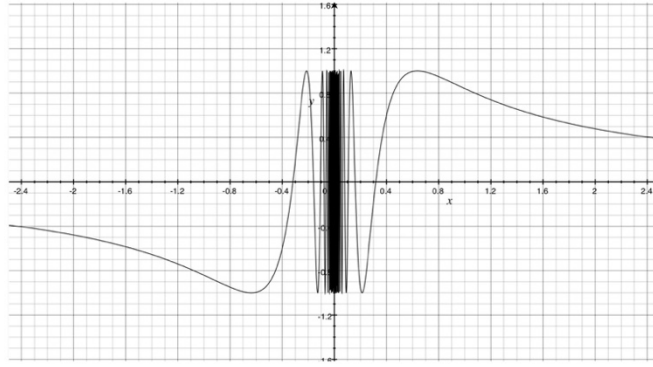


图 1.1 $f(x) = \sin(1/x)$ 的图像

而殷集与r-sets中所用的正则集最大的不同便是殷集使用了开集。而选用开集的原因如下。首先，在实体建模中，往往只对单一物体进行操作，而很多情况下，需要像操作单个实体一样操作许多实体的集合体。而使用闭集不能描述内部的重合边界，即在三维欧式空间中，r-sets会出现不是二维流形的边界[2]。所以更偏向于使用开集。在界面追踪问题中，流体的拓扑变化更为复杂，使用r-sets不能很好地建立实体模型。其次，在界面追踪问题中，要求实际物理区域是闭的对数值分析不友好。最后，在殷空间的理论中，我们用若当曲线来表示殷集的边界，并且会建立殷空间和一个的由特殊的若当曲线的集合构成的若当空间之间的一个同构关系，但如果用闭集来表示殷集，那么我们就得不到殷集的边界能唯一的分解成定向若当曲线这一良好的性质，从而同构会退化为同态。综合以上的原因，我们使用开集。

1.1.2 殷空间上的布尔代数

正如上面所说，在建立了殷空间和若当空间的同构关系后，殷空间上的布尔运算就可以由若当空间上相应的运算来实现。具体来说，一个很重要的定理是，对每个殷集 $\mathcal{Y} \neq \emptyset, \mathbb{R}^2$ 能唯一的表示成

$$\mathcal{Y} = \bigcup_j^{\perp\perp} \bigcap_i^{\perp\perp} \text{int}(\gamma_{i,j})$$

其中 j 是殷集中连通部分的索引， $\gamma_{i,j}$ 是几乎不相交的定向若当曲线。

由这个定理得到启示，我们定义 **spadjor** 为几乎不相交的定向若当曲线的集合。显然不是所有的 **spadjor** 对表示殷集都有帮助，我们将注意力集中到特殊的 **spadjor**。

我们定义 **atom spadjor** 为至多有一个正定向若当曲线 γ^+ 和有限个负定向若当曲线 $\gamma_1^-, \gamma_2^-, \dots, \gamma_{n_-}^-$ 的 **spadjor**, 使得,

- (a) γ_i^- 相对于包含运算是两两不可比较的,
- (b) $\gamma_l^- \prec \gamma^+$, 对每个 $l = 1, 2, \dots, n_-$,

其中包含关系指的是, 若 γ_k 包含 γ_l , 则 γ_l 的补集的有界部分是 γ_k 的补集的有界部分的子集。

从中我们可以 **atom spadjor** 为某个连通殷集的边界, 我们再顶一个边界-内部映射 ρ , 将对应的殷集分配给 **atom spadjor**, 再加上两个特殊的分别对应于空集和全平面的元素 $\hat{0}$ 和 $\hat{1}$ 。接着, 我们定义 **spadjor** 森林, 其为 **atom spadjor** 的集合, 并且一个森林中的 **atom spadjor** 在映射 ρ 下的像是分离的。很直观地, 我们能感觉到 **spadjor** 森林对应多个连通成分的殷集, 同时我们可以将边界-内部映射 ρ 延拓到 **spadjor** 森林上。**spadjor** 森林还有两个特殊的元素 $\hat{0}$ 和 $\hat{1}$ 就形成了若当空间。

要建立殷空间和若当空间之间的同构, 还需要在若当空间上建立对应于殷空间上正则交和正则补的 **meet** 和 **complementation** 运算, 具体的定义见[3]。而“交”运算可有 **meet** 和 **complementation** 运算得出。建立完同构后, 就能通过若当空间上的 **meet** 和 **complementation** 运算来实现殷空间上的布尔运算。

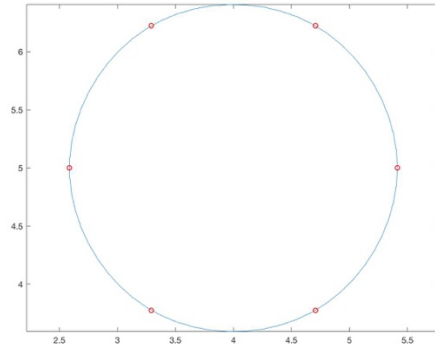


图 1.2 用六个点进行样条插值得到的图形

1.2 研究意义和目的

本研究旨在介绍一个殷空间上三次样条表示的殷集的布尔代数的理论框架和执

行一些布尔运算的算法。目的是将现有的基于直线段表示的殷集的布尔运算升级为三次样条函数。使用三次样条函数来逼近殷集边界时能比用直线段获得更高的精度和效率。例如，用圆的内接正六边形的顶点生成的三次样条函数就能很好的逼近圆，见图1.2。还有三次样条表示的殷集在界面追踪中的应用[9]。经典的扫描线算法能大幅提升求解直线段集交点问题的效率，将直线段升级为三次样条近似殷集后，将扫描线算法也引入求解样条曲线段的交点。

2 算法分析

有关殷空间上的布尔代数的具体执行算法，可见[3][7]。而涉及到的最重要的算法为直线段求交点的扫描线算法。而我的工作主要是将直线段升级为三次样条曲线段。

2.1 带有不确定性量化的扫描线算法

对于平面上一个线段的集合，要求集合中所有的线段交点，最自然，最简单的想法就是两两直线段都检查一遍，如果相交就求出交点。这个方法需要 $O(n^2)$ 的时间，但是在实际情况中，很少有所有线段都相交的情况，所以这个方法很费时。而直线段的交点扫描线算法，它的时间复杂度是与输出的结果的规模成正比，即跟交点个数成正比，那么在交点个数较少的情况下能大大提高效率。Bentley 和 Ottmann[5]给出了 $O(n \log n + l \log n)$ 时间的算法。具体细节和算法步骤可以见[4]。但是，以上两处都没有给出当有直线段重合时的处理情况，同时，在用计算机进行数值运算时不可避免会存在误差，所以，我们引入了一个容忍界(tol)作为允许的误差或相对误差的范围，即两个点的距小于设定的 tol 的值时就认为这两个点相等或者一个点与一条线段的距离小于 tol 时认为点在线段上(前提是点在线段的范围内)。黎至轩[6]进行了相关的改进被给出了相应算法。对于处理有线段重合时，想法是定义一个 **bundle**，它表示与当前扫描线都相交的重合线段的集合。则检验相邻线段是否相交则变为检验相邻的 **bundle**。同时，将一个新的直线段加入到一个 **bundle** 是求出这个新的线段与这个 **bundle** 里的其它线段的交点。这样就能处理有限段重合时的情况。

2.2 三次样条曲线段的扫描线算法

本次研究的各种算法都用 MATLAB 实现，所以先介绍一下用三次样条函数逼近殷集的方法。对于平面上一条简单曲线 γ ，选取 γ 上的 $n+1$ 个点 p_0, p_1, \dots, p_n （对于闭合曲线，取 $p_n = p_0$ ）。设 p_i 的坐标为 (x_i, y_i) 。用参数形式来表达横纵坐标的样条函数。令 $t_0 = 0, t_i = \sum_{j=1}^i \|p_j - p_{j-1}\|, i = 1, \dots, n$ ，其中 $\|p_j - p_{j-1}\|$ 表示两点间的距离。在 MATLAB 中分别对数组 $\{t_i\}$ 和 $\{x_i\}$ ， $\{t_i\}$ 和 $\{y_i\}$ 使用 **csape** 命令，便可得到横纵坐标的样条函数 $x(t)$ 和 $y(t)$ ，且它们都为分段三次函数，在 MATLAB 中的存储形式都为 **pp-form**。使用 **fnbrk** 命令可以很方便地提取出样条函数中的某一段，于是可将三次样条表示的曲面 γ 分解成 n 条曲线段。于是对应曲线段的扫描线算法的输入就为曲线段的集合。想要扫描线算法仍然适用，需要一些改动。根据 Benteley 和 Ottman[5]指出求交物体满足下面三条性质时，扫描线算法仍然适用：

- 1) 任何扫描线穿过物体时只与物体相交一次；
- 2) 任何一堆物体与扫描线相交时，能利用算法确定它们的相对顺序关系；
- 3) 任何两个物体，能用算法确定它们是否相交，如果相交的话能求出在某条固定的水平扫描线下最上面的一点（当有交点在扫描线上时，去当前事件点右边的点）。

根据以上三条，我所做的改进针对求交物体为三体样条曲线段时的改进为：

1) 将曲线段对 x 和 y 单调化，具体地，对 $x(t), y(t), t \in [0, t_n]$ ，分别求出在参数 t 的定义范围使得 $x'(t) = 0, y'(t) = 0$ 的 t 。根据所求的 t 再讲这一曲线段分成更多的曲线段，且得到的曲线段在 x 和 y 方向都是单调的，从而能保证曲线段与水平的扫描线相交时只有一个交点。同时，将曲线段单调化后，给定一条定向曲线和一个点，如果这个点的纵坐标在曲线段对纵轴投影的范围内，那么还能确定这个点是否在曲线的左侧（相对其方向），这一点在确定一个点是否在一个样条函数表示的曲线内时很有帮助。

2) 在直线段的扫描线算法中，在确定与当前的扫描线相交的 **bundle** 的偏序关系时。首先看 **bundle** 中任一条线段与扫描线交点的横坐标，横坐标小的 **bundle** 小。而横坐标相同时（在 **tol** 的误差内），与 x 轴正方向的夹角小的 **bundle** 小。而水平的关系定义为大于其它非水平的 **bundle**。而当 **bundle** 存储重合的样条曲线段时，首先仍可以用与扫描线交点的横坐标作为主要的判断标准。而当横坐标相同时，情况比直线段时复杂

很多。具体的判定方法见算法 1.

算法 1: $\text{bool} = \text{lt}(b_1, b_2)$

输入: b_1, b_2 是两个 bundle

输出: $\text{bool} = \text{true}$ 如果 $b_1 < b_2$

```

1: 全局变量  $q, \text{tol}$ , 此时扫描线为  $y = q(2)$  ( $q$  的纵坐标)。
2:  $c_1$  和  $c_2$  分别为  $b_1, b_2$  中的曲线段。
3: 求出扫描线与  $c_1$  和  $c_2$  的交点  $p_1$  和  $p_2$ 。
4: if  $|p_1(1) - p_2(1)| > \text{tol}$ 
5:      $\text{bool} = p_1(1) < p_2(1)$ 
6:     return;
7: end if
8:
9: 求出  $c_1$  和  $c_2$  在  $q_1$  处的斜率, 设为  $k_1$  和  $k_2$ 。
10: if  $k_1 \neq k_2 \ \&\& \ k_1 \neq 0 \ \&\& \ k_2 \neq 0$ 
11:      $\text{bool} = k_1 > k_2$ .
12:     return.
13: end if
14:
15: 令  $P$  为  $c_1$  和  $c_2$  的四个端点中垂直距离与  $q_1$  最小的那个点。
16: 令  $s_1$  和  $s_2$  为  $c_1$  和  $c_2$  与直线  $y = (P(2) + q_1(2))/2$  的交点。
17:  $\text{bool} = s_1(1) < s_2(1)$ 

```

3) 判断两条样条曲线段是否相交以及求出它们的交点较为复杂。首先, 在 MATLAB 中用相关命令可以求得曲线段在 x, y 方向上的最大最小值, 这样能求得一个覆盖这个曲线段的最小的矩形, 称为这个曲线段的边界盒。当两个曲线段的边界盒不相交时, 则两个曲线段也不会相交。这样能减少一些计算量。在我最开始求解两个曲线段的交点时, 所用的方法是直接用 MATLAB 的符号运算来求解二元三次方程组。即, 设两条样条曲线段的三次样条函数分别为 $x_1(t), y_1(t)$ 和 $x_2(s), y_2(s)$ 。定义符号变量 t, s 后, 设相应的三次函数为 $X_1(t), Y_1(t)$ 和 $X_2(s), Y_2(s)$ 。用 MATLAB 的命令 `vpasolve([X1(t) == X2(s), Y1(t) == Y2(s)])` 求解时, 因为符号运算对每一个数字都用精确的分数形式表示, 所以会产生有的数的分母多达几十位的情况, 在如此精确的表示下, 有些方程的求解速度会变的非常慢, 很长时间也不能完成求解的情况也出现过。这种情况的出现是因为符号运算的精确性, 而我们的整个布尔代数的运算过程中, 都允许一个误差 tol 的存在, 所以, 在计算过程使用近似计算得到的结果的误差未必会超出人为所规定的 tol 。我所用的相比符号运算牺牲一点精确性而提升计算速度的方

法有两种。

第一个方法是先找一个交点的估计值,以此为初值使用牛顿迭代法来进行数值计算。而初值往往选取两条曲线段的端点连成的线段的交点。而当这个交点不存在时,边采取第二种方法。

第二种方法是借助多远多项式理论来求解二元高次方程组[8]。设

$$X_1(t) = a_1 t^3 + a_2 t^2 + a_3 t + a_4, \quad Y_1(t) = b_1 t^3 + b_2 t^2 + b_3 t + b_4$$

$$X_2(s) = c_1 s^3 + c_2 s^2 + c_3 s + c_4, \quad Y_2(s) = d_1 s^3 + d_2 s^2 + d_3 s + d_4$$

令 $F_s(t) = X_1(t) - X_2(s)$, $G_s(t) = Y_1(t) - Y_2(s)$, 记

$$A = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 - X_2(s) & 0 & 0 \\ 0 & a_1 & a_2 & a_3 & a_4 - X_2(s) & 0 \\ 0 & 0 & a_1 & a_2 & a_3 & a_4 - X_2(s) \end{pmatrix}$$

$$B = \begin{pmatrix} c_1 & c_2 & c_3 & c_4 - Y_2(s) & 0 & 0 \\ 0 & c_1 & c_2 & c_3 & c_4 - Y_2(s) & 0 \\ 0 & 0 & c_1 & c_2 & c_3 & c_4 - Y_2(s) \end{pmatrix}$$

要求解方程组

$$\begin{cases} X_1(t) = X_2(s) \\ Y_1(t) = Y_2(s) \end{cases}$$

则先求解

$$R(F_s, G_s) = \left| \frac{A}{B} \right| \quad (1)$$

的根。 $R(F_s, G_s)$ 为一个关于 s 的多项式。设 s_0 为 $R(F_s, G_s)$ 的一个根,则将 s_0 带入 $F_s(t)$ 和 $G_s(t)$, 求解方程

$$\begin{cases} F_{s_0}(t) = 0 \\ G_{s_0}(t) = 0 \end{cases} \quad (2)$$

的公共根 t_0 , (t_0, s_0) 就是原方程的一组解。以此可以求得方程的所有解。然后取在所求范围内的解,就得到了两条曲线段的交点分别对应的参数值,从而能求得交点的坐标。

而在 MATLAB 中求解时,仍然借助符号运算求出 $R(F_s, G_s)$ 的表达式,下一步用 `sym2poly` 命令转化为以数组形式表示的多项式,再用 `roots` 命令就可求得 $R(F_s, G_s)$ 的所有的根。将求得的 s 带入公式(2)后,同样用 `roots` 命令求得方程的公共根。在这个方法中,使用 `roots` 而不是 `solve` 命令进行符号运算求解能提升计算的速度。

需要注意的是,如果有 $a_i = c_i$, $b_i = d_i$, 且 s, t 的取值范围相同,即两条求交的曲线段相同时, $R(F_s, G_s) \equiv 0$, 依照直线段求交点的设定,当出现重合时返回重合部分

的两个端点。在曲线段求交点时若出现 $R(F_s, G_s) \equiv 0$ 也应当返回端点的参数值。进一步地, 两条曲线段可能存在部分重合, 或者重合但是定向相反, 这些情况在计算时并不能反映出来, 所以, 更准确的做法是在两个曲线段求交点时, 先判断它们的次序关系, 若相等, 则返回重合部分的端点。

解决了以上三个问题后, 直线段的扫描线算法就能应用到三次样条曲线段上, 且算法不需要大的改动。

2.3 三次样条表示的殷集的布尔代数

如 2.2 中所说的, 选取若当曲线上的控制点并进行三次样条插值, 用生成的样条函数来表示若当曲线。当给定了所要表示的殷集的每一条若当曲线上的控制点后, 可以生成表示这个殷集的一个类对象 **SpadjorForest**, 在用输入的数据生成类对象的时候, 会根据每个若当曲线的位置关系来确定最终的 **Spadjor Forests**。因为一旦表示一个殷集的各条若当曲线确定后, 这个殷集就确定了, 因此输入的若当曲线的顺序并不影响最终生成的殷集。

在确定最终的殷集时, 很重要的一步是确定各个殷集的包含关系。因为一个殷集中的每个若当曲线都是不相交的, 所以只需要确定一个若当曲线内部 (若当曲线补集中有界的部分) 的一点相对另个若当曲线的位置关系 (在内部或在外部)。在用直线段表示殷集时, 每个若当曲线都可表示一个多边形。取出多边形内部一点的方法可以借鉴多边形三角划分的算法, 具体可见[4][6]。而到用三次样条表示殷集时, 样条多边形的边界变成了曲边, 取出其内部一点的情况边的更复杂。可以先取出其控制点形成的多边形的一个三角形, 如图 2。这是最理想的情况, 两条曲边相对其对应的线段都是凸的, 取三角形的中心即可得所求的曲边多边形内部的一点。当曲边相对线段是凹的时候, 可过三角形的内心做一条与不为曲边的线段平行的直线, 找出其与两条曲线段的两个交点, 这两个交点的终点即为所求点。

从上可见求一个曲边多边形内部一点要较多边形复杂的多。既然最终目的是要确定两个若当曲线的包含关系, 在曲线没有 **proper** 交点的情况下, 可以用一个若当曲线上不在另一个曲线上的一点来判断。详见算法 2。判断一个点是否在三次样条曲线上只需求解三元一次方程便可得到。

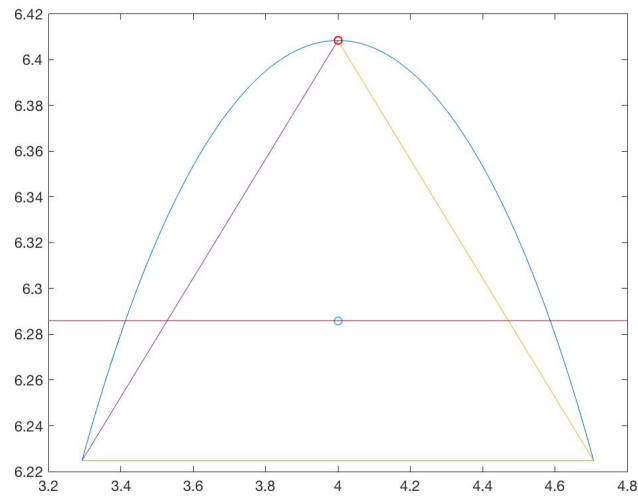


图 2.1

算法 2: $r = \text{include}(c_1, c_2)$

输入: c_1, c_2 是两个没有 proper 交点的若当曲线

输出: $r=1$ 如果 $c_1 > c_2$ (c_1 包含 c_2), $r=-1$ 如果 $c_1 < c_2$, $r=0$ 如果两者没有包含关系

- 1: v_1, v_2 分别存储为 c_1, c_2 的节点坐标
 - 2: 找出第一个不在 c_1 中不在 c_2 上的点 u 。
 - 3: 找出第一个不在 c_2 中不在 c_1 上的点 v 。
 - 4: 若不存在 u 或 v , 则两条曲线相等, 返回 $r=0$;
 - 5:
 - 6: if u 在 c_2 内, 且 v 在 c_1 外
 - 7: $r = -1$;
 - 8: else if u 在 c_2 外, v 在 c_1 内
 - 9: $r = 1$;
 - 10: else
 - 11: $r = 0$;
 - 12: end if
-

3 MATLAB 程序

现已有基于直线段表示的殷集的布尔代数的 C++ 代码。本次实验的目的是将基于 C++ 代码完成 MATLAB 代码, 并将直线段升级为三次样条曲线段。为了使得代码的结构更清晰合理, 我采用了 MATLAB 的面向对象编程。

3.1 类的设计

表 3.1 主要的类对象

类名称	表示对象	变量	主要函数
Point	点	coord	compare; area;
SimpleCubicCurve	三次样条表示的简单曲线	vertices; chLen; xsp; ysp; isClosed;	contain; extract; concat; collapse split;
CurveSegment	进行求交点的曲线段	p1; p2; xsp; ysp; domain;	substitute; contain; left; intersect; xymonotonize; lt;
SpdjorForest	殷集	loops; nodes; isBounded; type;	inclusionMatrix; buildHasse; buildDirectedMultigraph; setIntersection;

涉及到的重要的类以及其变量和主要函数见表 3.1。其中 SimpleCubicCurve 中的 isClosed 变量若为 true，则表示曲线是闭合的，这个类也用来表示殷集中的若当曲线。当对 SimpleCubicCurve 使用 collapse 后，会将这条曲线按照节点分解成许多的样条曲线段，即为 CurveSegment，这也是一条特殊的样条曲线，但是定义一个新的类是因为在许多算法中要对这些特殊的样条曲线进行项操作，比如进行单调化，求曲线段之间的交点等。

取一条曲线上若干个不同的节点，当用这些节点生成一个 SimpleCubicCurve 类对象时，还需要指定所要生成的样条函数的次数，4 次表示三次样条，2 次表示直线段。支持生成同时有以上两种次数的混合样条函数。输入中除了节点坐标意外，还需要一个量 ‘SplinePars’ 表示各模块的信息，更详细的信息可见 ‘generateSplineBdry.m’ 子程序。

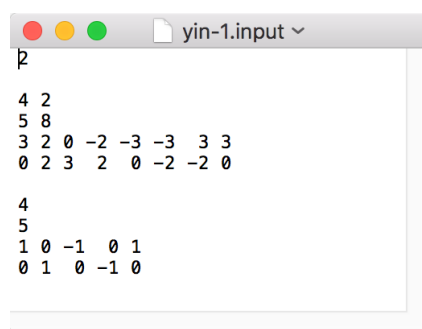
正如 2.3 节所说，一个 SpadjorForest 类对象表示一个殷集，它的的变量中，loops

表示其中所有的若当曲线，为 `SimpleCubicCurve` 类型变量的一个数组；`nodes` 为结构数组表示殷集的哈斯表。对普通的殷集其生成函数要求的输入是 `SimpleCubicCurve` 对象的数组，且对要求的输入的若当曲线的次序没有任何要求，因为在生成对象的过程中会根据输入确定殷集的结构。

3.2 输入与输出

当用户想要输入一个殷集时，可以将相关的数据放在一个文档里。该文档的第一行只有一个数字，表示这个殷集中若当曲线的数量。其后输入各个若当曲线的数据。在表示各个若当曲线的数据中，第一行表示点的个数，第二行和第三行表示 `SplinePars`，第四行为节点的横坐标，第五行为节点的纵坐标。图 3.1 表示了一个输入的例子。第一行的 2 表示这个殷集包含两个若当曲线。第一个若当曲线的 `SplinePars` 为 `[4 5;2 8]`，第二个的为 `[4 5]`。

函数 `'booleanOps(filename1,filename2,operation,tolerance)'` 的前两个输入即为存储要进行布尔运算的殷集的数据的文件名，`'operation'` 为相应的运算，`t` 目前支持 `'setIntersection'`，`'setUnion'`，`'setMinus'`，`'setSymmDiff'` 四种布尔运算，`tolerance` 为容忍的误差。函数的输出为一个 `'SpadjorForest'` 类。也可同样将输出的殷集中的各个若当曲线的节点的坐标输出到文本中。



```

2
4 2
5 8
3 2 0 -2 -3 -3 3 3
0 2 3 2 0 -2 -2 0

4
5
1 0 -1 0 1
0 1 0 -1 0

```

图 3.1 输入例子

4 测试结果

为了更直观的显示最终结果，在 `SpadjorForest` 类中加入了两个绘画图像的函数，`'plotBoundary'` 和 `'plot'`。前一个函数画出殷集中的所有若当曲线，并添加箭头以表示曲线定向；后一个函数填充殷集的内部，填充的颜色为黄色。在每次测试中，填充经过布尔运算得到的殷集的内部，并画出参与运算的两个殷集的边界，并以不

同的颜色加以区分。且由于其它集中布尔运算（并、差、对称差）都以交运算和求补运算为基础，所以主要对交运算进行测试。测试中选取一个圆周上的等距的六个点来生成样条函数。

4.1 交运算

4.1.1 两个边界重合的殷集

图 4.1 和图 4.2 中的两个参与运算的殷集的若当曲线都是重合的，但是后一个中的两个殷集方向相反。求交运算给出了正确的结果。

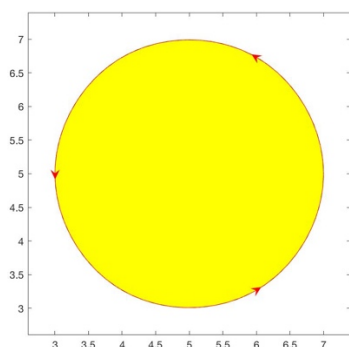


图4.1. 同方向的相同曲线

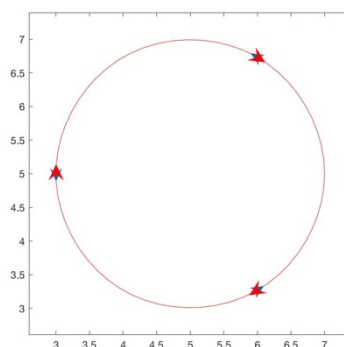


图4.2. 不同方向的相同曲线

4.1.2 边界分离的殷集

对于边界没有交点的两个殷集，其相交运算的结果见图 4.3 和图 4.4。

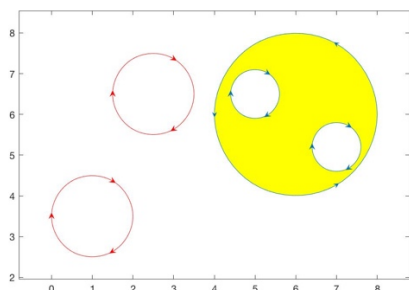


图4.3.

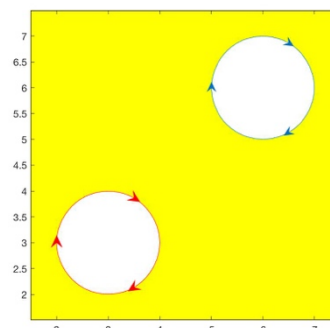


图4.4.

4.1.3 improper 交点

图 4.5 和图 4.6 是两个 improper 交点的情况。

4.1.4 proper 交点

图 4.7 和图 4.8 是两个交点个数较多的情况。

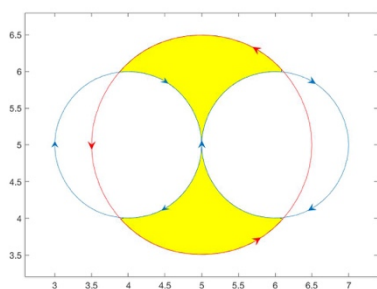


图4.5.

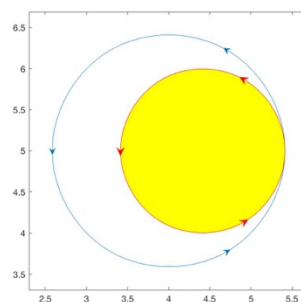


图4.6.

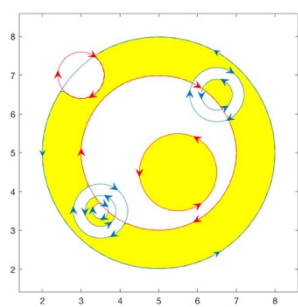


图4.7.

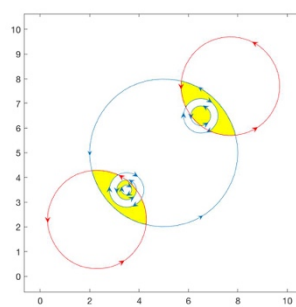


图4.8.

4.2 其它布尔运算

图 4.9, 图 4.10, 图 4.11 分别为两个股集的并、差和对称差运算的结果。

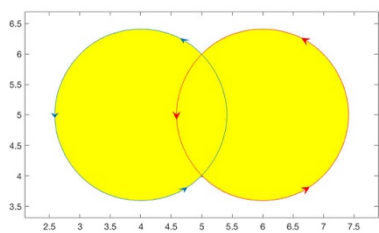


图4.9. 并

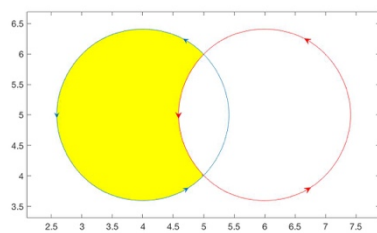


图4.10. 差

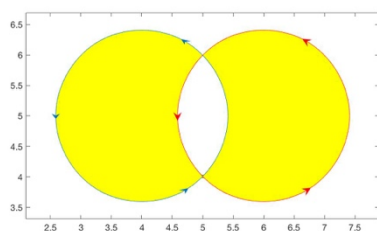


图4.11.对称差

参考文献

- [1] AML710 CAD, Lecture 30, *Solid Modeling*.
- [2] F. Arbab, *Set models and Boolean operations for solids and assemblies*, IEEE Computer Graphics Appl., 10(1990), pp. 76-86.
- [3] Q. Zhang, *Boolean algebra on physically meaningful regions in the plane*.
- [4] de Berg M, Cheong O, van Kreveld M, Overmars M. *Computational geometry: algorithms and applications*. 3rd ed. Berlin: Springer; 2008.
- [5] J. L. Bentley and T. A. Ottmann. Algorithms for reporting and counting geometric intersections. IEEE Trans. Comput., C-28:643-647, 1979.
- [6] 黎至轩, A Variant of the Line Sweep Algorithm with User-defined Uncertainty Quantification.
- [7] 黎至轩, An Implementation of the Boolean Algebra For Yin Sets.
- [8] 李方, 黄正达, 温道伟, 汪国军. 高等代数 下册 (第二版), 浙江大学出版社
- [9] Q.Zhang, A cubic Mars method for fourth-, sixth-, and eight-order interface tracking of an arbitrary number of materials in two dimensions.