第4-4章B样条曲线

课程目标

- B样条曲线的定义
- B样条曲线的数学表达式
- 二次B样条曲线
- 三次B样条曲线
- B样条曲线生成算法

4.4.1 B样条曲线

- **Bezier**曲线的不足之处：
  - 多边形的顶点数（m个），决定**Bezier**曲线的阶次（m-**1**次），不灵活
  - 当顶点数（m）较大时，曲线的阶次将较高。多边形对曲线形状的控制将明显减弱。
  - 调和函数的值在开区间（0， 1）内均不为零。曲线在（$0<t<1$）的区间内的任何一点均要受到全部顶点的影响。即改变其中任一个顶点的位置，将会对整条曲线产生影响，因而对曲线进行局部修改将成为不可能。

4.4.1 B样条曲线

- 拓展**Bezier**曲线
  - 用n次B样条基函数替换了伯恩斯坦基函数，构造B样条曲线。
  - B样条曲线除了保持了原**Bezier**曲线所具有的优点外，增加了可以对曲线进行局部修改的优点。
  - 具有对特征多边形更逼近，多项式阶次较低等优点。

4.4.2 B样条曲线的数学表达式

给定m+n+1个顶点P_i（i=0，1，…，m+n），可定义

m+1段n次的参数曲线：

$$P_{k,n}(t)=\sum_{i=0}^{n}P_{i+k}F_{i,n}(t)\qquad (0\leqslant t\leqslant 1),\ i=0,1,\ldots,n$$

$P_{k,n}(t)$为第k段n次B样条曲线段（k=0，1，…，m），

$F_{i,n}(t)$为n次B样条基函数，其形式为：

$$F_{i,n}(t)=\frac{1}{n!}\sum_{j=0}^{n-i}(-1)^j C_{n+1}^{j}(t+n-i-j)^n\qquad (0\le t\le 1, i=0,1,\cdots,n)$$

4.4.2 B样条曲线的数学表达式

- 连接全部曲线段所组成的整条曲线称为**n次B样条曲线**。
  - n次B样条曲线可达到n-1阶连续。
  - 三次B样条曲线和二次B样条曲线应用得较为广泛。
  - 高于三次的B样条曲线，计算过于复杂

4.4.3 二次B样条曲线

- 对于二次B样条曲线，n=2，i=0，1，2。

$$F_{0,2}(t)=\frac{1}{2!}\sum_{j=0}^{2}(-1)^{j}C_{3}^{j}(t+2-0-j)^{2}$$

$$=\frac{1}{2!}\left[(-1)^{0}C_{3}^{0}(t+2)^{2}+(-1)^{1}C_{3}^{1}(t+1)^{2}+(-1)^{2}C_{3}^{2}(t)^{2}\right]$$

$$=\frac{1}{2}\left[\frac{3!}{3!}(t+2)^{2}-\frac{3!}{2!}(t+1)^{2}+\frac{3!}{2!}t^{2}\right]$$

$$=\frac{1}{2}\left[t^{2}+4t+4-3t^{2}-6t-3+3t^{2}\right]=\frac{1}{2}(t-1)^{2}$$

4.4.3 二次B样条曲线

$$F_{1,2}(t)=\frac{1}{2!}\sum_{j=0}^{1}(-1)^{j}C_{3}^{j}(t+2-1-j)^{2}=\frac{1}{2}\left[(-1)^{0}C_{3}^{0}(t+1)^{2}+(-1)^{1}C_{3}^{1}(t)^{2}\right]$$

$$=\frac{1}{2}\left[(t+1)^{2}-3t^{2}\right]=\frac{1}{2}\left(-2t^{2}+2t+1\right)$$

$$F_{2,2}(t)=\frac{1}{2!}\sum_{j=0}^{0}(-1)^{j}C_{3}^{j}(t+2-2-j)^{2}=\frac{1}{2}\left[(-1)^{0}C_{3}^{0}(t)^{2}\right]=\frac{1}{2}t^{2}$$

4.4.3 二次B样条曲线

- 二次B样条曲线的分段表达式：

$$P_i(t)= F_{0,2}(t)P_i + F_{1,2}(t)P_{i+1} + F_{2,2}(t)P_{i+2} \quad (i= 0,1,2，\ldots.\ m)$$

- 即

$$P_i(t)=\frac{1}{2}(t-1)^2\cdot P_i+\frac{1}{2}\left(-2t^2+2t+1\right)P_{i+1}+\frac{1}{2}t^2P_{i+2}$$

4.4.3 二次B样条曲线

- 二次B样条曲线更一般化的形式：

$$\mathbf{P}(t)=\sum_{k=0}^{2}P_k F_{k,2}(t)=\begin{bmatrix}t^2 & t & 1\end{bmatrix}\cdot\frac{1}{2}\cdot\begin{bmatrix}1 & -2 & 1\\ -2 & 2 & 0\\ 1 & 1 & 0\end{bmatrix}\cdot\begin{bmatrix}P_0\\ P_1\\ P_2\end{bmatrix}\qquad(0\le t\le 1)$$

- P_k为分段曲线的B特征多边形的顶点：P_0，P_1，P_2
- 第i段曲线的P_k为：P_i，P_{i+1}，P_{i+2}连续三个顶点

4.4.3 二次B样条曲线

- 对P(t)求导，可得：

$$\mathbf{P}'(t)=\begin{bmatrix} t & 1 \end{bmatrix}\cdot\begin{bmatrix} 1 & -2 & 1 \\ -1 & 1 & 0 \end{bmatrix}\cdot\begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix} \quad (0 \le t \le 1)$$

4.4.3 二次B样条曲线

- 总结二次B样条曲线在端点处的性质

$$\because\ P(t)=\frac{1}{2}(t-1)^2\cdot P_0+\frac{1}{2}\left(-2t^2+2t+1\right)P_1+\frac{1}{2}t^2P_2$$

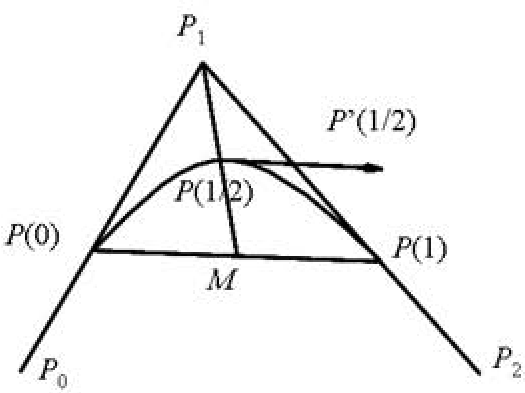
$$P'(t)=(t-1)P_0+\left(-2t+1\right)P_1+tP_2$$

$$\therefore\begin{cases}P(0)=\dfrac{1}{2}(P_0+P_1)\\ P(1)=\dfrac{1}{2}(P_1+P_2)\end{cases}\qquad\begin{cases}P'(0)=P_1-P_0\\ P'(1)=P_2-P_1\end{cases}$$

$$\begin{cases}P(\frac{1}{2})=\dfrac{1}{8}P_0+\dfrac{3}{4}P_1+\dfrac{1}{8}P_2=\dfrac{1}{2}\left\{\dfrac{1}{2}\left[P(0)+P(1)\right]+P_1\right\}\\ P'(\frac{1}{2})=\dfrac{1}{2}(P_2-P_0)=P(1)-P(0)\end{cases}$$

4.4.3 二次B样条曲线

- 由以上的三对式子说明：
  - 二次B样条曲线段的起点**P(0)**在第一条边的**中点**处，且其切向量P_1-P_0
  - 终点**P(1)**在第二条边的**中点**处，且其切向量P_2-P_1
  - **P(1/2)**正是△P(0)P_1P(1)的**中线P_1M的中点**，且在P(1/2)处的**切线**平行于**P(0)P(1)**。



4.4.3 二次B样条曲线

- 由n个顶点定义的二次B样条曲线
  - 是n-2段抛物线（相邻三点定义）的连接
  - 并在连接处达到一阶连续

4.4.4 三次B样条曲线

- 对于三次B样条曲线，**n=3,k=0,1,2,3**。

$$
\begin{aligned}
F_{0,3}(t) &= \frac{1}{3!}\sum_{j=0}^{3}(-1)^j C_4^j (t+3-0-j)^3 \\
&= \frac{1}{6}\left[C_4^0(t+3)^3 - C_4^1(t+2)^3 + C_4^2(t+1)^3 - C_4^3(t)^3\right] \\
&= \frac{1}{6}\left[t^3+9t^2+27t+27-4t^3-24t^2-48t-32+6t^3+18t^2+18t+6-4t^3\right] \\
&= \frac{1}{6}\left[(t+3)^3-4(t+2)^3+6(t+1)^3-4(t)^3\right] \\
&= \frac{1}{6}(-t^3+3t^2-3t+1)
\end{aligned}
$$

4.4.4 三次B样条曲线

$$F_{1,3}(t)=\frac{1}{3!}\sum_{j=0}^{2}(-1)^{j}C_4^{j}(t+3-1-j)^3$$

$$=\frac{1}{6}\left[C_4^0(t+2)^3-C_4^1(t+1)^3+C_4^2(t)^3\right]$$

$$=\frac{1}{6}\left[t^3+6t^2+12t+8-4t^3-12t^2-12t-4+6t^3\right]$$

$$=\frac{1}{6}\ (3t^3-6t^2+4)$$

4.4.4 三次B样条曲线

$$F_{2,3}(t)=\frac{1}{3!}\sum_{j=0}^{1}(-1)^{j}C_4^{j}(t+3-2-j)^3$$

$$=\frac{1}{6}\left[C_4^{0}(t+1)^3-C_4^{1}(t)^3\right]$$

$$=\frac{1}{6}\left[t^3+3t^2+3t+1-4t^3\right]$$

$$=\frac{1}{6}\;(-3t^3+3t^2+3t+1)$$

4.4.4 三次B样条曲线

$$F_{3,3}(t)=\frac{1}{3!}\sum_{j=0}^{0}(-1)^{j}C_{4}^{j}(t+3-3-j)^{3}$$

$$=\frac{1}{6}\left[C_{4}^{0}(t)^{3}\right]$$

$$=\frac{1}{6}t^{3}$$

4.4.4 三次B样条曲线

- 三次B样条曲线的表达式：

$$P(t) = F_{0,3}(t)P_0 + F_{1,3}(t)P_1 + F_{2,3}(t)P_2 + F_{3,3}(t)P_3$$

$$= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \cdot \frac{1}{6} \cdot \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix} \qquad (0 \le t \le 1)$$

4.4.4 三次B样条曲线

- 讨论三次B样条曲线的端点性质

$$P'(t)=\begin{bmatrix}t^2 & t & 1\end{bmatrix}\cdot\frac{1}{2}\cdot\begin{bmatrix}-1 & 0 & -3 & 1\\ 2 & -4 & 2 & 0\\ -1 & 3 & 1 & 0\end{bmatrix}\cdot\begin{bmatrix}P_0\\ P_1\\ P_2\\ P_3\end{bmatrix}\qquad(0\le t\le 1)$$

$$P(t)=\frac{1}{6}[(1-t)^3P_0+(3t^3-6t^2+4)P_1+(-3t^3+3t^2+3t+1)P_2+t^3P_3]$$

$$P'(t)=\frac{1}{6}[-3(1-t)^2P_0+(9t^2-12t)P_1+(-9t^2+6t+3)P_2+3t^2P_3]$$

$$P''(t)=\frac{1}{6}[6(1-t)P_0+(18t-12)P_1+(-18t+6)P_2+6tP_3]$$

4.4.4 三次B样条曲线

∴ 以t的端点值代入，得：

$$\begin{cases} P(0) = \dfrac{1}{6}(P_0 + 4P_1 + P_2) = \dfrac{1}{3}\left(\dfrac{P_0 + P_2}{2}\right) + \dfrac{2}{3}P_1 \\ P(1) = \dfrac{1}{6}(P_1 + 4P_2 + P_3) = \dfrac{1}{3}\left(\dfrac{P_1 + P_3}{2}\right) + \dfrac{2}{3}P_2 \end{cases}$$

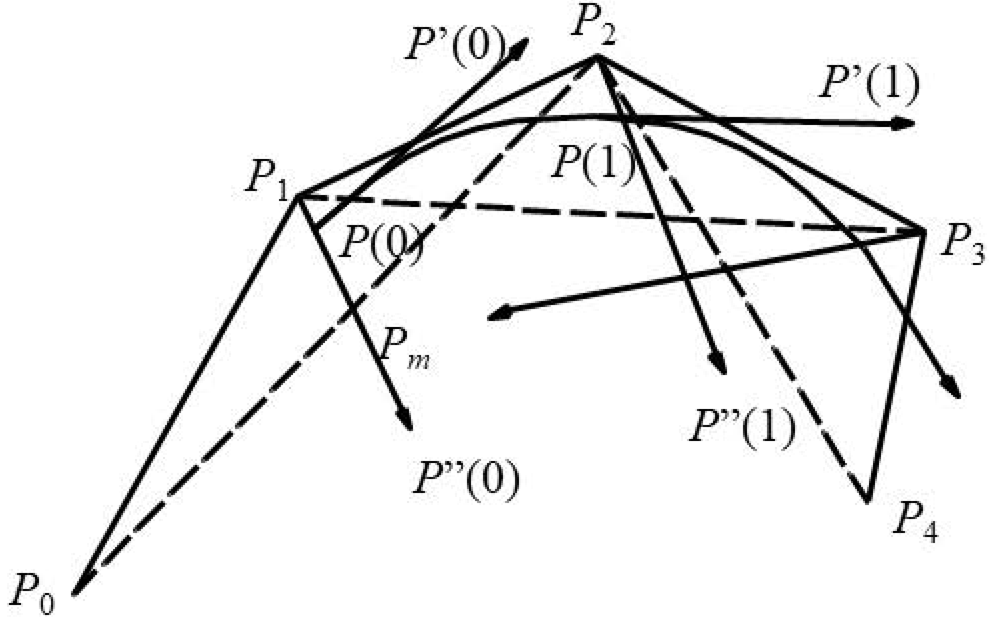
$$\begin{cases} P'(0) = \dfrac{1}{2}(P_2 - P_0) \\ P'(1) = \dfrac{1}{2}(P_3 - P_1) \end{cases}$$

$$\begin{cases} P''(0) = P_0 - 2P_1 + P_2) = (P_2 - P_1) + (P_0 - P_1) \\ P''(1) = P_1 - 2P_2 + P_3) = (P_3 - P_2) + (P_1 - P_2) \end{cases}$$

4.4.4 三次B样条曲线

- 从以上的端点结果可以看到：
  - 曲线段的起点P(0)位于△$P_0P_1P_2$底边P_0P2的中线P_1P_m上，且距P_1点的1/3处。
  - 起点的切矢P'(0)平行于△$P_0P_1P_2$的底边P_0P_2，且长度为其1/2。
  - 起点的二阶导数P''(0)等于中线矢量P_1P_m的二倍.
  - 同理，对于终点P(1)处的情形与此相应。

4.4.4 三次B样条曲线



4.4.4 三次B样条曲线

- 若增加顶点P_4，$P_1P_2P_3P_4$可定义一段新的三次B样条曲线。
- 因为新曲线段起点的有关数据和上一段曲线的终点的有关数据都只和P_1、P_2、P_3三点有关，所以该二段曲线在连接处的位置矢量，二阶切矢应相等

$$P''_1(1) = P''_2(0)$$

- 三次B样条曲线可以达到二阶连续。

三次B样条曲线的算法源程序

```
void BSpLine(POINT *p, int n){
    int x,y,i,j,k=1000;
    double t,t1,t2,t3,a,b,c,d;
    t=1.0/k;
    p[n].x=2*p[n-1].x-p[n-2].x;
    p[n].y=2*p[n-1].y-p[n-2].y;
    moveto(p[1].x,p[1].y);
    for(i=1;i<n-1;i++){
      for(j=1;j<=k;j++){
        t1=j*t;   t2=t1*t1;    t3=t2*t1;
        a=(3*t2-t3-3*t1+1)/6;
        b=(3*t3-6*t2+4)/6;
        c=(3*t2-3*t3+3*t1+1)/6;
        d=t3/6;
         x=(int)(a*p[i-1].x+b*p[i].x+c*p[i+1].x+d*p[i+2].x);
         y=(int)(a*p[i-1].y+b*p[i].y+c*p[i+1].y+d*p[i+2].y);
            lineto(x,y);
       }
    }
}
```

BSpline.c

人物漫画

绘制过程

首先需要一幅绘制好的画作，通过观察寻找绘制的规律，然后分析绘制的层次。

在用程序绘制比较复杂的画作时，一定要耐心，将画作的层次分析清楚，可以大大节省很多代码量。一般都是将复杂的图形用简单的图形进行拼接，覆盖等方式进行绘制。

代码分析

```c
#include<graphics.h>
#include<conio.h>
#include<math.h>

#define PI acos(-1.0)

void Background();      // 背景
void Hat();             // 帽子
void Hair();            // 头发
void Face();            // 脸
void Food();            // 食物
void Clothes();         // 衣服
void Redface();         // 红色脸蛋
void Eyes();            // 眼睛
void Logo();            // 标志

int main()
{
    initgraph(640, 640);
    Background();
    Clothes();
    Hat();
    Face();
    Food();
    Hair();
    Logo();
    _getch();
    return 0;
}

void Background()
{
    setbkcolor(RGB(237, 202, 160));
    cleardevice();
    setfillcolor(RGB(249, 211, 172));
    for (int i = 0; i < 10; i++)
    {
        bar(0, i * 65, 640, 25 + i * 65);
        bar(i * 65, 0, 25 + i * 65, 640);
    }
}
```

```c
void Hat()
{
    setlinecolor(RGB(154, 97, 78));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 2);
    arc(111, 31, 575, 400, 318.0 / 180 * PI, 191.5 / 180 * PI);
    arc(97, 107, 557, 425, 352.5 / 180 * PI, 160.0 / 180 * PI);
    setfillcolor(RGB(199, 62, 44));
    floodfill(390, 80, RGB(154, 97, 78));
    arc(115, 122, 522, 423, 354.0 / 180 * PI, 190.0 / 180 * PI);
    arc(376, 206, 521, 383, 335.0 / 180 * PI, 45.0 / 180 * PI);
    arc(461, 307, 517, 397, 275.0 / 180 * PI, 55.0 / 180 * PI);
    setfillcolor(RGB(184, 46, 27));
    floodfill(393, 128, RGB(154, 97, 78));
}

void Face()
{
    setlinecolor(RGB(154, 97, 78));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 2);
    arc(157, 166, 500, 426, 189.0 / 180 * PI, 346.0 / 180 * PI);
    arc(118, 272, 167, 334, 58.0 / 180 * PI, 317.0 / 180 * PI);
    arc(155, 172, 267, 360, 156.0 / 180 * PI, 237.0 / 180 * PI);
    arc(167, 177, 289, 389, 120.0 / 180 * PI, 212.0 / 180 * PI);
    line(171, 257, 184, 222);
    arc(160, 136, 605, 487, 106.0 / 180 * PI, 165.0 / 180 * PI);
    arc(316, -74, 622, 274, 194.0 / 180 * PI, 259.0 / 180 * PI);
    arc(417, 91, 610, 312, 193.0 / 180 * PI, 254.0 / 180 * PI);
    arc(450, 298, 494, 359, 355.0 / 180 * PI, 54.0 / 180 * PI);
    setfillcolor(RGB(254, 228, 215));
    floodfill(315, 267, RGB(154, 97, 78));
    setlinecolor(RGB(255, 214, 198));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 10);
    arc(165, 142, 605, 493, 108.0 / 180 * PI, 165.0 / 180 * PI);
    arc(312, -62, 622, 280, 193.0 / 180 * PI, 259.0 / 180 * PI);
    Redface();
    Eyes();
    // 耳朵
    setlinecolor(RGB(154, 97, 78));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 3);
    line(136, 297, 150, 292);
    line(150, 292, 142, 312);
    line(142, 312, 151, 316);
    // 脖子
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 2);
    line(287, 423, 287, 446);
    line(287, 446, 321, 467);
    line(321, 467, 360, 441);
    line(360, 441, 359, 423);
    setfillcolor(RGB(254, 228, 215));
    floodfill(320, 445, RGB(154, 97, 78));

    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 20);
    setlinecolor(RGB(249, 210, 193));
    arc(147, 183, 508, 433, 259.0 / 180 * PI, 280.0 / 180 * PI);
}
```

代码分析

```cpp
void Redface()
{
    setlinestyle(PS_SOLID, 2);
    float H = 11.f;
    float V = 1.f;
    float S;
    for (int i = 0; i < 32; i++)
    {
        S = (50 - i) / 100.f;
        setlinecolor(HSVtoRGB(H, S, V));
        circle(206, 336, i);
        circle(443, 333, i);
    }
}

void Eyes()
{
    setfillcolor(RGB(113, 102, 97));
    solidellipse(220, 254, 248, 309);
    setlinestyle(PS_SOLID, 8);
    setlinecolor(RGB(113, 102, 97));
    line(383, 289, 429, 277);
    line(383, 289, 421, 291);
    line(421, 291, 436, 298);
    setlinestyle(PS_SOLID, 2);
    // 嘴巴
    setlinecolor(RGB(154, 97, 78));
    setfillcolor(RGB(216, 107, 74));
    fillellipse(310, 307, 331, 337);
}

void Food()
{
    setlinecolor(RGB(154, 97, 78));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 2);
    setfillcolor(RGB(250, 171, 114));
    fillellipse(243, 426, 319, 617);
    setfillcolor(RGB(252, 164, 93));
    fillellipse(168, 384, 243, 627);
    setlinecolor(RGB(221, 138, 84));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 2);
    for (int i = 265; i < 288; i++)
    {
        setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, i - 265)
        line(i, 481, 288, 481);
    }
    for (i = 261; i < 283; i++)
    {
        setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, i - 261)
        line(i, 515, 283, 515);
    }
    setlinecolor(RGB(154, 97, 78));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 3);
    line(285, 470, 264, 481);
    line(264, 481, 290, 488);
    line(285, 508, 259, 515);
    line(259, 515, 283, 523);
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 4);
    setlinecolor(RGB(255, 212, 176));
    line(190, 426, 190, 522);
    line(210, 401, 210, 524);
    line(225, 434, 225, 530);
    // 食物袋子
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 2);
    setlinecolor(RGB(156, 103, 71));
    setfillcolor(RGB(232, 187, 156));
    line(329, 640, 325, 547);
    line(325, 547, 164, 528);
    line(164, 528, 155, 535);
    line(155, 535, 156, 569);
    line(156, 569, 312, 579);
    line(312, 579, 325, 551);
    line(308, 589, 306, 640);
    line(160, 569, 154, 640);
    floodfill(214, 554, RGB(156, 103, 71));
    floodfill(220, 600, RGB(156, 103, 71));
}

void Hair()
{
    setlinecolor(RGB(154, 97, 78));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 2);
    arc(132, 283, 195, 391, 172.0 / 180 * PI, 266.0 / 180 * PI);
    arc(131, 351, 197, 445, 138.0 / 180 * PI, 257.0 / 180 * PI);
    arc(136, 421, 183, 496, 136.0 / 180 * PI, 262.0 / 180 * PI);
    arc(143, 461, 192, 543, 156.0 / 180 * PI, 246.0 / 180 * PI);
    // 右边辫子
    arc(446, 333, 491, 418, 200.0 / 180 * PI, 264.0 / 180 * PI);
    arc(441, 382, 515, 474, 138.0 / 180 * PI, 210.0 / 180 * PI);
    arc(430, 424, 490, 503, 125.0 / 180 * PI, 251.0 / 180 * PI);
    arc(420, 469, 473, 539, 122.0 / 180 * PI, 250.0 / 180 * PI);
    arc(406, 363, 503, 446, 271.0 / 180 * PI, 15.0 / 180 * PI);
    arc(419, 416, 489, 475, 280.0 / 180 * PI, 18.0 / 180 * PI);
    arc(410, 439, 476, 513, 262.0 / 180 * PI, 16.0 / 180 * PI);
    arc(422, 479, 462, 536, 278.0 / 180 * PI, 5.0 / 180 * PI);
    arc(406, 552, 452, 684, 300.0 / 180 * PI, 250.0 / 180 * PI);
    // 蝴蝶结
    setfillcolor(RGB(204, 62, 48));
    fillellipse(396, 518, 431, 558);
    fillellipse(436, 535, 479, 566);
    fillcircle(433, 545, 9);
    line(169, 527, 154, 540);
    // 染头发
    setfillcolor(RGB(254, 253, 251));
    floodfill(167, 380, RGB(154, 97, 78));
    floodfill(394, 187, RGB(154, 97, 78));
    floodfill(429, 585, RGB(154, 97, 78));
    setlinecolor(RGB(253, 247, 231));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 14);
    arc(120, 130, 515, 430, 20.0 / 180 * PI, 170.0 / 180 * PI);
    arc(425, 87, 610, 305, 215.0 / 180 * PI, 254.0 / 180 * PI);
    arc(155, 180, 560, 446, 189.0 / 180 * PI, 215.0 / 180 * PI);
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 2);
```

代码分析

```
    // 染头发
    setfillcolor(RGB(254, 253, 251));
    floodfill(167, 380, RGB(154, 97, 78));
    floodfill(394, 187, RGB(154, 97, 78));
    floodfill(429, 585, RGB(154, 97, 78));
    setlinecolor(RGB(253, 247, 231));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 14);
    arc(120, 130, 515, 430, 20.0 / 180 * PI, 170.0 / 180 * PI);
    arc(425, 87, 610, 305, 215.0 / 180 * PI, 254.0 / 180 * PI);
    arc(155, 180, 560, 446, 189.0 / 180 * PI, 215.0 / 180 * PI);
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 2);
    setlinecolor(RGB(154, 97, 78));
    setfillcolor(RGB(200, 82, 68));
    fillcircle(144, 325, 4);
    fillellipse(139, 332, 151, 373);
    fillellipse(485, 341, 497, 385);
    line(421, 577, 423, 607);
    line(430, 589, 433, 629);
}

void Clothes()
{
    setlinecolor(RGB(140, 106, 96));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 2);
    setfillcolor(RGB(204, 155, 148));
    fillellipse(122, 427, 509, 907);
    setfillcolor(RGB(255, 220, 216));
    fillellipse(280, 410, 380, 758);
    setlinecolor(RGB(191, 141, 130));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 7);
    ellipse(285, 412, 385, 760);
    setlinecolor(RGB(140, 106, 96));
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 2);
    setfillcolor(RGB(249, 211, 172));
    fillellipse(280, 409, 365, 530);
    setfillcolor(RGB(202, 72, 48));
    fillellipse(311, 475, 342, 534);
    fillcircle(326, 475, 10);
    line(337, 482, 346, 495);
    line(346, 495, 365, 477);
    line(463, 575, 466, 640);
    line(399, 451, 434, 501);
    line(434, 501, 400, 640);
}

void Logo()
{
    setlinecolor(WHITE);
    setlinestyle(PS_SOLID | PS_ENDCAP_FLAT, 2);
    rectangle(151, 546, 201, 593);
    settextcolor(WHITE);
    settextstyle(25, 0, _T("微软雅黑"),0, 0, 0, false, false, false, DEFAULT_CHARSET, OUT_DEFAULT_PRECIS, CLIP_DEFAULT_PRECIS, ANTIALIASED_QUALITY, DEFAULT_PITCH);
    setbkmode(TRANSPARENT);
    outtextxy(154, 546, _T("@ 令"));
    outtextxy(154, 568, _T("也 小"));
}
```

carton.cpp