GraphicsEditor

1. Описание на проекта

Целта на проекта е да реализира графичен редактор с основни функционалности:

- рисуване
- чертане на отсечки
- рисуване на криви по 2 начина
- чертане на фигури
- чертане на красиви фигури
- палитра с цветове
- запаметяване на изображение
- зареждане на изображение
- прилагане на ефекти върху нарисуваните или заредени изображения

Проектът е резизиран на C++ с помощта на библиотеката SDL2, като е използвано Visual Studio 2015. Единственият използван примитив е точка, с който се реализират почти всички функционалности.

2. Изчертаване на фигури

2.1. Основен алгоритъм

При рисуването на фигурите сме използвали библиотеката SDL2 за обработка на събития. При всички фигури чертането започва от натискането на мишката до нейното пускане. Взима се точката от първоначалния клик и тя се използва за първа точка от фигурата (в рисуването на окръжност първата точка е центърът). При пускане на мишката се взима точката, на която е бил курсурът в този момент и тя се използва за крайна.

2.2. Растеризиране на отсечка

Използван е алгоритъма на Брезенхам. Като входни параметри за алгоритъма имаме две точки в равнината, които представляват крайните точки на отсечката. Резултатът от изпълнението на алгоритъма е отсечката, съединяваща двете точки.
Нека $P_0(x_0, y_0), P_1(x_1, y_1)$ са съответно началната и крайната точка.

За по-добро разбиране ще разгледаме случая, когато $x_0 < x_1$ и $y_0 < y_1$. Всеки друг случай се обработва, като се свежда до дадения.

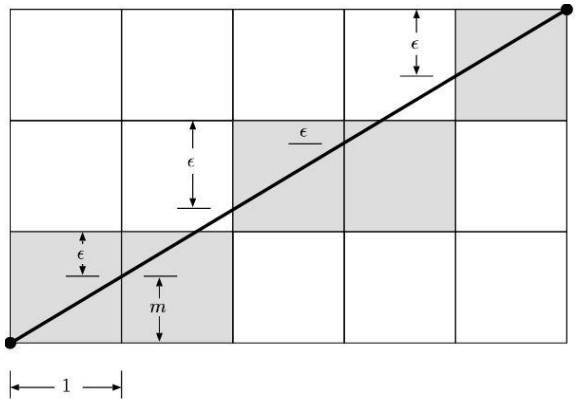
Алгоритъмът е итеративен, като на всяка стъпка се изчисляват координатите на следващия пиксел и той се запълва.

Правата през двете точки P_0 и P_1 може да бъде определена чрез уравнението

$$y = m\,(x - x_0) + y_0, \qquad m := \frac{y_1 - y_0}{x_1 - x_0}$$

Алгоритъмът използва променливата ε, наречена грешка, за изчисляване на стойността на y. Тази променлива представлява разстоянието между точката, в която правата "излиза" от пиксела, и горния му връх, взето с отрицателен знак (Фиг. 1). В началото, стойността на ε се задава да бъде m – 1 .

На всяка стъпка тя се увеличава с m. Ако ε стане по-голямо от 0, следва че правата се е "изкачила" нагоре и трябва да увеличин стойността на y. Стойността на ε трябва да бъде преизчислена, за да отразява вярно стойността на y координатата. Това става като от грешката се извади 1.



Фиг. 1

Процедура:

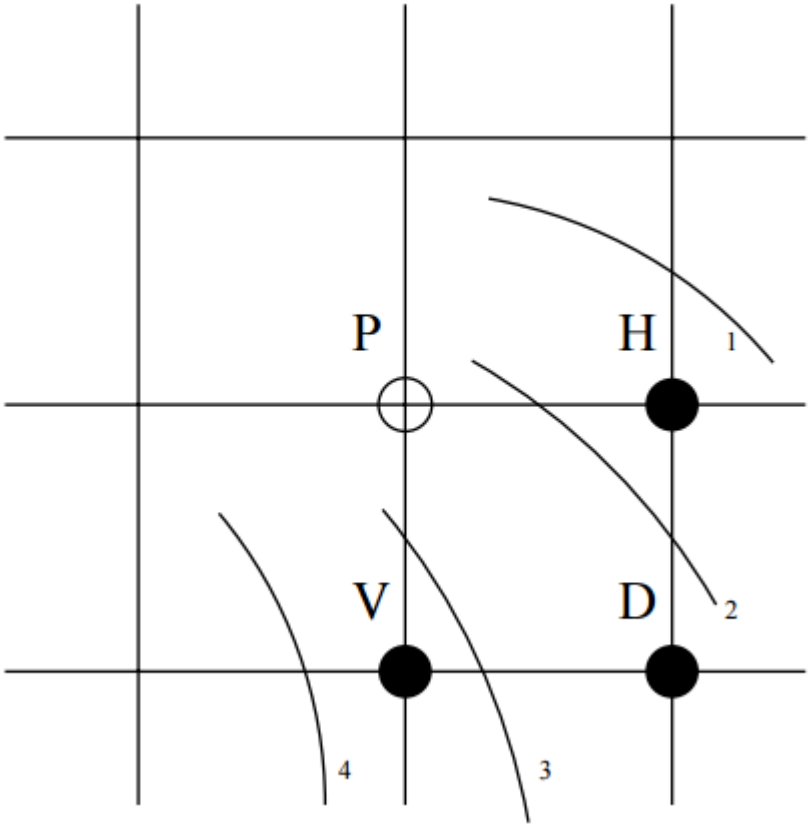
1. Задаваме $\Delta y = y_1 - y_0$, $\Delta x = x_1 - x_0$, $i = x_0$, $j = y_0$
2. Задаваме начална стойност на грешката $\varepsilon = m - 1$, $m = \frac{\Delta y}{\Delta x}$
3. Запълваме пиксела (i, j). Ако $i = x_1 - 1$, алгоритъмът приключва.
4. Ако $\varepsilon \geq 0$, то $j = j + 1$ и $\varepsilon = \varepsilon - 1$
5. $i = i + 1$, $\varepsilon = \varepsilon + m$. Отиваме на стъпка 3.

2.3. Окръжност

За чертането на окръжност използваме алгоритъма на Брезенхам за растеризиране на окръжност. С цел по-лесно обяснение на алгоритъма ще опишем как той действа когато центъра на окръжността съвпада с центъра на координатната ситема. Ясно е, че общия случай се получава аналогично след тривиално преобразуване на координатите.

За окръжност с радиус R и център началото на осите, точката (0,R) е част от нея. Ще започнем чертането от тази точка и ще се движим по часовниковата стрелка до правата y = x. В настоящия алгоритъм ще разгледаме само втори октант от координатната система (заключен между правите x = 0 и y = x). При чертането на всяка точка ще чертаем и нейните симетрични в другите 7 октанта. Например за точката (x, y) от втори октант ще начертаем и точките (y, x), (−y, x), (x, −y), (−x, −y), (−y, −x), (−x, y) в другите 6 октанта.

Нека $f(x,y) = x^2 + y^2 - R^2$ е уравнението на окръжността. Функцията f е отрицателна за всички точки вътре в кръга, положителна за всички външни точки и се нулира за точките от окръжността. $|f(x,y)|$ ни дава разстоянието между сегашната точка и окръжността.



Фиг. 2

Ако на текущата стъпка се намираме в точка P с координати (x, y), възможностите за следваща точка са H(x + 1, y), D(x + 1, y - 1) и V(x, y – 1).

Алгоритъмът на Брезенхам за растеризиране на окръжност е следният:

Входни данни: Координати на текущата точка, радиус

Изходни данни: Окръжност

Процедура:

1. Ако координатите на точката съвпадат, значи сме достигнали правата y = x и приключваме алгоритъма. В противен случай преминаваме към стъпка 2.
2. Изчисляваме грешката за текущата точка $\Delta p = f(D) = (x+1)^2 + (y-1)^2 - R^2$
3. Ако $\Delta p < 0$ се преминава към стъпка 4, в противен случай се преминава към стъпка 8
4. Изчисляваме величината $\delta_p = |f(H)| - |f(D)|$
5. Ако $\delta p < 0$ се преминава към стъпка 5, в противен случай стъпка 6

6. Чертаем точка H (и седемте съответстващи и точки), взимаме я за текуща точка и преминаваме към стъпка 1
7. Чертаем точка D (и седемте съответстващи и точки), взимаме я за текуща точка и преминаваме към стъпка 1
8. Изчисляваме $\varepsilon_p = |f(D)| - |f(H)|$
9. Ако $\varepsilon_p < 0$ се преминава към стъпка 10, в противен случай стъпка 11
10. Чертае се точка D (и седемте съответстващи и точки), взимаме я за текуща точка и преминаваме към стъпка 1
11. Чертае се точка V (и седемте съответстващи и точки), взимаме я за текуща точка и преминаваме към стъпка 1

2.4. Правоъгълник

Чертането на правоъгълник става чрез неколкократно прилагане на алгоритъма на Брезенхам за отсечка. Започва се от точката A(x1, y1) и се стига до точка C(x2, y2). Другите 2 точки се изчисляват като те имат координати B(x2, y1) и D(x1, y2). С алгоритъма на Брезенхам се свързват точките, за да се получи правоъгълника ABCD.

2.5. Триъгълник

В проекта е възможно чертането само на правоъгълен триъгълник. Няма специален алгоритъм за това, вместо това се използва модификация на алгоритъма за чертане на правоъгълник. При начална точка (x1, y1) и крайна точка (x2, y2) се чертае отсечка между тях (чрез алгоритъма на Брезенхам за чертане на отсечка). След това подобно на правоъгълника, но се взима само една точка C (x2, y1) и тя се свързва с първите 2.

3. Обработка на изображение

3.1. Запаметяване

Запаметяването ще става във формат BMP, като файлът ще се записва в директорията на проекта. Самото запаметяване ще се извършва чрез функцията SDL_SaveBMP(“file name”) от библиотеката SDL2.

3.2. Ефекти

Всеки BMP файл се състои от така наречения Bitmap. Т.е за всеки пиксел на екрана има 3 стойности за цвят – RGB (red, green, blue). Тези стойности са числа от 0 до 255 представени в двоичен вид(8 бита). Всяка от тези стойности отговаря за съдържанието на съответния цвят в текущия пиксел. Повечето ефекти се състоят в промяна на тези стойности за всеки пиксел по дадено правило

3.2.1. Неатив

Негатив ефектът е също известен като обръщане на цветовете. При него за всеки пиксел стойностите за цветовете се заменят с „обратните“ им. Т.е. от 255 се вади текущата им стойност. Например ако един пиксел е син неговите стойности са R = 0, G = 0, B = 255. Ако приложим негатив върху него ще се получи R’ = 255 – R, G’ = 255 – G, B’ = 255 – B, което означава R’ = 255, G’ = 255, B’ = 0, което е жълт цвят.

3.2.2. Сепия

За постигането на Сепия ефект се използват специални коефициенти, различни в зависимост от тона, който се цели. За всеки пиксел се изчисляват новите стойности на цветовете според коефициентите. Нашата реализация е класическа Сепия във кафяви тонове. Нека старите цветове са redOld, greenOld, blueOld, а търсение цветове са red, green, bue. Новите цветове се получават по следния начин:

red = redOld * 0.393 + greenOld * 0.769 + blueOld * 0.189

green = redOld * 0.349 + greenOld * 0.686 + blueOld * 0.168

blue = redOld * 0.272 + greenOld * 0.534 + blueOld * 0.131

3.2.3. Grayscale

В Grayscale изображението всеки пиксел съдържа информация за яркостта на цветовете от преди промяната.

3.2.4. Blurr

Blurr или „замъгляване“ се постига чрез осредняване на стойностите на пикселите. За всеки пиксел новият цвят се получава като средно аритметично на стария цвят + цветовете на всички околни пиксели.

4. Сплайн функции

Сплайн кривите са основен инструмент в системите за компютърен дизайн (CAD) и за компютърна графика. Основно се използват като апроксимиращи криви, но е възможна и интерполация на множество от контролни точки. Широкото им използване започва през втората половина на 20-ти век, при моделирането на автомобили в заводите на френските производители. Името се дължи на факта, че основен градивен елемент са сплайн функциите, които предоставят необходимите качества.

Дефиниция: Нека е дадена редицата $\boldsymbol{t}=\{t_i\}_{i=1}^{n}$, като $t_1 < t_2 < \dots < t_n$ Функцията s(x) се нарича **сплайн функция** от ред r с възли $\boldsymbol{t}$, ако:

1. $s(x) \in \Pi_r$ във всеки интервал (t_i, t_{i+1}), където i = 0, 1, …… , n, като считаме, че $t_0 = -\infty$ и $t_{n+1} = +\infty$.
2. Функциите s(x), s’(x), …., $s^{(r-1)}(x)$ са непрекъснати в интервала $(-\infty, +\infty)$ или по-точно - непрекъснати в точките $t_1, t_2, \dots, t_n$.

Името на функциите е взето от прибор, който е използван за моделирането на корпусите на корабите. Уредът представлява линия, направена от пластичен материал, която може да заема формата на различни гладки криви.

Можем да проверим, че множеството S_r от сплайн функциите от ред r върху възлите $\boldsymbol{t}(t_1, t_2, \dots, t_n)$ е линейно пространство. Това означава, че можем да намерим базис, по който

можем да представим всяка една функция от $S_r(\mathbf{t})$. За тази цел ще дефинираме отсечената степенна функция:

Дефиниция: *Функцията*

$$(x-c)_+^k = \begin{cases} (x-c)^k, & x \geq c \\ 0 & \text{, иначе} \end{cases}$$

наричаме отсечена степенна функция.

От дефиницията получаваме, че степенната функция е сплайн функция с един възел в точката ***c***. Може да бъде доказана следната теорема:

Теорема: *Всяка сплайн функция $s \in S_r(t_1, t_2, \dots, t_n)$, може да бъде представена по единствен начин във вида :*

(1) $$s(x) = p(x) + \sum_{i=1}^{n} c_k (x-t_k)_+^r \text{ , където}$$

$c_k = \frac{\lim_{h\to 0, h>0} s^{(r)}(x_k+h) - \lim_{h\to 0, h<0} s^{(r)}(x_k+h)}{r!}$, k = 1, 2, … , n и p(x) ∈ π_r.

Всеки израз от вида (1) е сплайн функция от ред r с възли $\boldsymbol{t}$ (правим проверка дали са изпълнени двете условия, за сплайн функциите). Но и всяка сплайн функция може да се представи вида (1). Следователно множеството на сплайн функциите от ред r и с възли $\boldsymbol{t}$ и множеството от функции представени чрез израз (1) съвпадат.

Вече имаме базис за пространството $S_r(\boldsymbol{t})$. Ясно е, че размерността на $S_r(\boldsymbol{t})$ е **n+r+1**. От друга страна, локално в интервала (t_i, t_{i+1}) $s \in S_r(t_1, t_2, \dots, t_n)$ е полином от степен **r** и би трябвало да се представи като линейна комбинация на **r+1** елемента. Оказва се, че такова представяне съществува, като за базис се използват така наречените ***B-сплайни***.

Дефиниция: *Нека са дадени възлите $\Delta = \{t_0 \leq t_1 \leq t_2 \dots \leq t_{n+r}\}$. Редицата от* ***B-сплайни*** *от ред r-1 $\{\boldsymbol{N_{i,r-1}}\}_{\boldsymbol{i=1}}^{\boldsymbol{n}}$ се дефинира така:*

$$N_{i,r-1}(t) = (t_{i+r} - t_i)(\cdot - t)_+^{r-1}[t_i, \dots, t_{i+r}] .$$

В случая степенната функция е функция на една променливи - t. Точковата нотация означава, че при пресмятането на разделените разлики заместваме на мястото на първия аргумент.

Знаем, че разделената разлика може да бъде представена във вида:

$$f[x_0, \dots, x_n] = \sum_{i=0}^{n} \frac{f(x_i)}{\prod_{k=0, k\neq i}^{n} (x_i - x_k)}.$$

Следователно всеки сплайн от ред r с тези възли може да бъде представен във вида $\sum_{k=i}^{i+r} c_k (t_k - t)_+^{r-1}$. От (1), но приложено за (t_i, t_{i+r}) получаваме, че B-сплайните от ред r-1, чиито възли са различни, са сплайн функции от ред r-1 и следователно $N_{i,r-1}(t) \in C^{r-2}[t_i, t_{i+r}]$. Оказва се, че ако $t_j \in [t_i, t_{i+r}]$ се повтаря m на брой пъти, то $N_{i,r-1}(t) \in C^{r-m-1}[t_i, t_{i+r-1}]$. Това

следва от свойствата на разделените разлики. В този случай B-сплайн функцията вече не е част от пространството $S_{r-1}(t)$, тъй като не изпълнява условието за непрекъснатост.

Можем да изведем няколко полезни свойства на B-сплайните, които показват защо те са много подходящи за целите на компютърната графика и моделирането чрез гладки криви.

1. B-сплайните имат краен носител:

$$supp\ N_{i,r-1} = [\ t_i, t_{i+r}]$$

Доказателство: *Когато* $t > t_{i+r}$ *за* $x = t_i, t_{i+1}, \dots, t_{i+r}$, $(x\ -\ t\ )_+^{r-1} = 0$ *и така и* $N_{i,\,r-1} = 0$. *Когато* $t < t_i$ *за* $x = t_i, \dots, t_{i+r}$ $(x - t\ )_+^{r-1} = (x-t)^{r-1}$. *Но е от степен r-1. Следователно разделената разлика в r произволни възела ще бъде 0. От където следва, че и* $N_{i,r-1} = 0$. □

2. Рекурентна зависимост (de Boor [1972]) :

$$N_{i,r-1}(t) = \frac{t-t_i}{t_{i+r-2}-t_i} N_{i,r-2} + \frac{t_{i+r-1}-t}{t_{i+r-1}-t_{i+1}} N_{i+1,r-2}$$

$$N_{i,1}(t) = \begin{cases} 1, & t \in [t_i, t_{i+1}) \\ 0, & \text{иначе} \end{cases}$$

3. $\sum_{i\,=\,0}^{n} N_{i,r-1}(t) \equiv 1$, за $t \in [t_r, t_{n+1}]$

Доказателство: $\sum_{i=0}^{n} N_{i,r-1}(t) = (\cdot - t)_+^{r-1}([t_{i+1} \dots t_{i+r}] - [t_i \dots t_{i+r-1}])$
Ако $t \geq t_k$, следва $(x-t)_+^{r-1} = 0$ за $x = t_0, \dots, t_r$ и $(x-t)_+^{r-1}[t_0, \dots, t_r] = 0$
Ако $t \leq t_{n+1}$, следва $(x-t)_+^{r-1} = (x-t)^{r-1}$ за $x = t_{n+1}, \dots, t_{n+r}$ и
$(x-t)_+^{r-1}[t_0, \dots, t_r] = 1$. □

B-сплайн криви

Дефиниция: *За последователност от точки в равнината сплайн крива* $c(t)$ *се задава като комбинация от точките* P_i *(наречени контролни точки) и сплайн функциите* $N_{i,r}(t)$ - $c(t) = \sum_{i=0}^{n} P_i\, N_{i,r}(t)$, $t \in [t_0, t_{n+r}]$, *и B-сплайн функциите са дефинирани чрез възлите* $\boldsymbol{t} = \{t_0, \dots, t_{n+r}\}$.

По-горе показахме основни свойства на сплайн функциите, които се пренасят и върху сплайн кривите и определят поведението им.

Установихме, че когато възлите са различни B-сплайн функциите от ред r имат непрекъснати производни до ред r-1. Следователно, дефинираните по този начин криви са гладки. Когато обаче имаме повтарящи се възли, гладкостта на кривата "намалява". Този факт може да се използва при моделирането на криви, които пресичат крайните две контролни точки или при интерполация на множество точки.

Знаем също така всяка сплайн функция има краен носител. Това означава, че за всеки интервал $[t_i, t_{i+1}]$, сплайн кривата е определена само от част от контролните точки – тези, за които съответстващите им базисни функции се ненулеви в интервала. Това свойство е познато като

локалност, защото при промяна на дадена точка само част от кривата се променя. По-точно, ако променим точка P_i, то ще се промени частта от кривата, дефинирана в $[t_i, t_{i+r})$.

Знаем, че за сплайните имаме свойството $\sum_{i=0}^{n} N_{i,r-1}(t) \equiv 1$, за $t \in [t_r, t_{n+1}]$. Но във всеки отделен интервал само една част от всички базисни функции са ненулеви и сумата им е тъждествено равна на 1. Това означава, че за всеки един интервал $[t_i, t_{i+1})$, кривата е афинна комбинация на част от точките и се съдържа в изпъкналия четириъгълник, образуван от точките. Това свойство определя и добрите апроксимиращи качества на B-сплайн кривите.

Алгоритъм за намиране на точка от B-сплайн крива.

Изчисляването на координатите на точка от сплайн крива може да се извърши чрез пресмятане по формулата за разделените разлики. Този метод не е подходящ за използване в CAD системи, защото се губи точност.

По-добър алгоритъм е открит от Carl de Boor и е основан на рекурентната формула, която също е открита от него.

$$P_{j,k} = \begin{cases} P_j\ , & k = 0 \\ (1\ -\ \alpha_{j,k})P_{j-1,k-1} + \alpha_{j,k}P_{j,k-1}, & 1 \le k \le r, i-d+k \le j \le i \end{cases}$$

$$\alpha_{j,k} = \frac{u - u_j}{u_{j+r+1-k} - u_j},$$

$$i \in \{r,\ r+1,\ \ \ldots, n-1\}$$

5. Криви на Безие

Кривата на Безие представлява параметрична полиномиална крива P(t), т.е. координатните функции на радиус-вектора на произволна точка от кривата са полиноми на един реален аргумент.

През 60-те години на миналия век френските математици и дизайнери Пол дьо Кастелжо (Ситроен) и Пиер Безие (Рено), независимо един от друг, достигнали до идеята за използване на параметрични криви, зададени чрез контролни полигони от точки. Подобни идеи имало още през 20-те години в класическата диференциална геометрия, но по онова време не намерили приложение. Сега кубичните криви на Безие са стандартен инструмент в редица приложения за графичен дизайн и предпечатна подготовка: PhotoShop, InDesign, GIMP, Scribus и др. Квадратните криви на Безие намират приложение и в дизайна на шрифтове - за изчертаване на контурите на символите в True Type шрифтовете и кубични в Post Script Type 1 и TeX шрифтовете на Доналд Кнут.

За задаване на крива на Безие от степен n са необходими:

n + 1 на брой контролни точки P_0, P_1, ..., P_n, където n ∈ N. n е и степента на полиномиалната функция. Кривата не минава през контролните точки (с изключение на първата и последната). Всяка точка влияе на посоката на кривата, като я „дърпа“ към себе си. Това влияние е най-силно, когато кривата е най-близо до точката.

Формулата, с която се задава кривата на Безие, е следната:

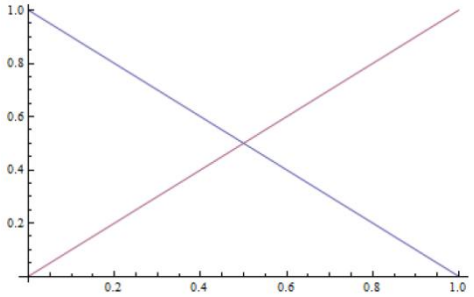
$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t), \qquad 0 \le t \le 1$, където $B_{n,i}(t)$) са полиномите на Сергей Бернщайн (базисните функции на Безие):

$$B_{n,i}(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}, \ i = 0,1,...,n .$$

За всяко естествено n, $B_{n,i}(t)$ са n + 1 на брой полинома на t от n-та степен. Освен това по определение t ∈ [0, 1].

Полиномите на Бернщайн от първа степен (при n=1) са 2 линейни функции на t (Фиг. 3):

$$B_{1,0}(t) = \binom{1}{0} t^0 (1-t)^{1-0} = 1-t \text{ и } B_{1,1}(t) = \binom{1}{1} t^1 (1-t)^{1-1} = t .$$



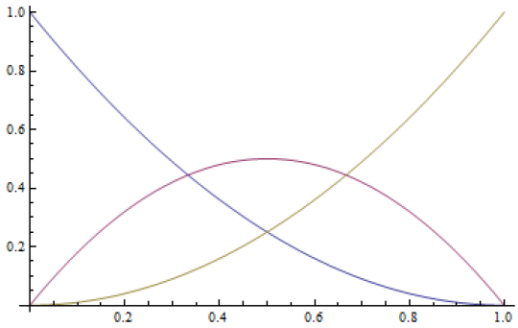
Фиг. 3

Полиномите на Бернщайн от втора степен (при n=2) имат вида:

$$B_{2,0}(t) = \binom{2}{0} t^0 (1-t)^{2-0} = (1-t)^2 ,$$

$$B_{2,1}(t) = \binom{2}{1} t^1 (1-t)^{2-1} = 2t(1-t) ,$$

$$B_{2,2}(t) = \binom{2}{2} t^2 (1-t)^{2-2} = t^2 .$$

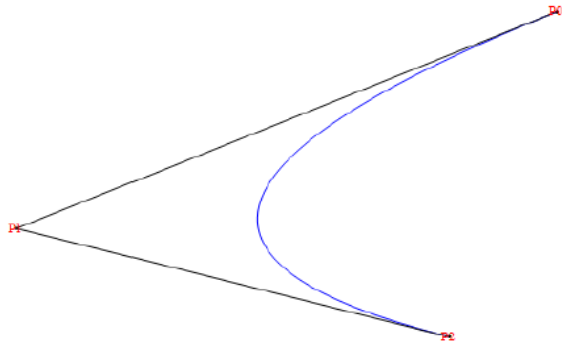


Фиг. 4

За кривата на Безие при n=2 получаваме:

$$P(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2 = ((1-t)^2, 2t(1-t), t^2)(P_0, P_1, P_2)^T =$$

$$= (t^2, t, 1)\begin{pmatrix} 1 & -2 & 1 \\ -2 & 2 & 0 \\ 1 & 0 & 0 \end{pmatrix}\begin{pmatrix} P_0 \\ P_1 \\ P_2 \end{pmatrix}.$$

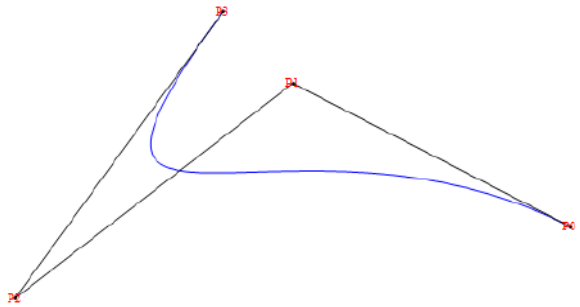


На Фиг. 5 е показана квадратната крива за контролните точки: $P_0(2, 3)$, $P_1(-3, 1)$, $P_2(1, 0)$.

Фиг. 5

Полиномите на Бернщайн от трета степен (при n=3) имат вида:

$$B_{3,0}(t) = (1-t)^3,$$
$$B_{3,1}(t) = 3t(1-t)^2,$$
$$B_{3,2}(t) = 3t^2(1-t),$$
$$B_{3,3}(t) = t^3.$$



На Фиг. 6 е показана квадратната крива за контролните точки: $P_0(4, 2)$, $P_1(0, 4)$, $P_2(-4, 1)$, $P_3(-1, 5)$.

Фиг. 6

Лесно се установява,че сумата на трите полинома на Бернщайн от 2-ра степен е 1. Същото е изпълнено и за четирите полинома от 3-та степен. Това свойство ("разделяне на единицата") важи за полиномите на Бернщайн от произволна n-та степен. Изпълнено е, че:

$$1 = (t + (1-t))^n = \sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} = \sum_{t=0}^{n} B_{n,i}(t) .$$

Полиномите на Бернщайн са линейно независими помежду си и тъй като са n + 1 на брой, образуват база на π_n – векторното пространство на полиномите на t с реални коефициенти от степен по-малка или равна на n.

Така кривата на Безие C(t) от n-та степен, определена от контролните точки P_0, P_1,…, P_n, има вида:

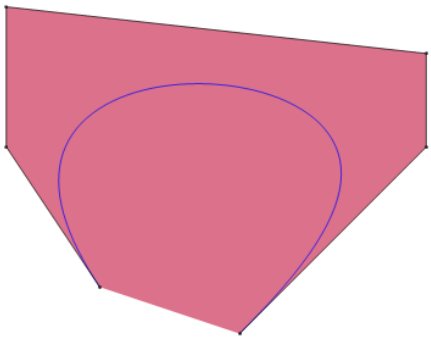
$C(t) = \sum_{i=0}^{n} B_{n,i}(t) P_i$, където $B_{n,i}(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}$ и t ∈ [0, 1]. Първото уравнение се нарича уравнение на Безие на кривата C(t) или още уравнение във форма на Бернщайн.

Някои от свойствата на кривата на Безие:

- Всяка крива на Безие минава през първата и последната си контролни точки, като двете крайни точки от контролния полигон на кривата се получават съответно при t = 0 и t = 1.
- Кривата на Безие е симетрична и ако първоначално точките са номерирани P_0, P_1,…, P_n и разменим номерацията по следния начин: P_n, P_{n-1},…, P_0, ще получим същата крива. Това свойство е в сила, тъй като за полиномите на Бернщайн е вярно, че $B_{n,j}(t) = B_{n,n-j}(1-t)$, за $0 \le j \le t$.
- Преместването на една от контролните точки променя цялата крива. Това свойство се дължи на факта, че тегловите функции $B_{n,i}(t)$ са ненулеви за всички стойности на t освен t = 0 и t = 1.
- Кривата на Безие не изменя формата си при промяна на координатната система (афинна инвариантност).
- Поради свойството „разделяне на единацата“ и неотрицателността на полиномите на Бернщайн,

  $\sum_{i=0}^{n} B_{n,i}(t) = 1, \quad 0 \le B_{n,i}(t) \le 1$ за t ∈ [0, 1],

  всяка крива на Безие лежи изцяло в изпъкналата обвивка на контролните си точки. Това е най-малкият изпъкнал многоъгълник, който съдържа във вътрешността си или върху контурите си всички контролни точки.



Фиг. 7

На Фиг. 7 е показана крива на Безие от 5-та степен за контролните точки: (-3, -2), (-5, 1), (-5, 4), (4, 3), (4, 1), (0, -3) и изпъкналата обвивка на тези точки.

Кривите на Безие се допират до първото (P_0, P_1) и последното си контролно рамо (P_{n-1} P_n), както се вижда и от горните фигури.

Доказателство:

$$p'(t) = \left(\sum_{i=0}^{n} p_i \binom{n}{i} t^i (1-t)^{n-i}\right)' =$$

$$= \left(p_0 \binom{n}{0} t^0 (1-t)^n + \sum_{i=1}^{n-1} p_i \binom{n}{i} t^i (1-t)^{n-i} + p_n \binom{n}{n} t^n (1-t)^0\right)' =$$

$$= (p_0.1.1.(1-t)^n)' + \left(\sum_{i=1}^{n-1} p_i \binom{n}{i} t^i (1-t)^{n-i}\right)' + (p_n.1.t^n.1)' =$$

$$= -np_0(1-t)^{n-1} + \sum_{i=1}^{n-1} p_i \binom{n}{i} [(t^i(1-t)^{n-i})'] + np_n t^{n-1} =$$

$$= -np_0(1-t)^{n-1} + \sum_{i=1}^{n-1} p_i \binom{n}{i} [it^{i-1}(1-t)^{n-i} - (n-i)t^i(1-t)^{n-i-1}] + np_n t^{n-1}.$$

Заместваме с t = 0 и t = 1:

$$p'(0) = -np_0 + np_1 = n\overrightarrow{p_0p_1},$$

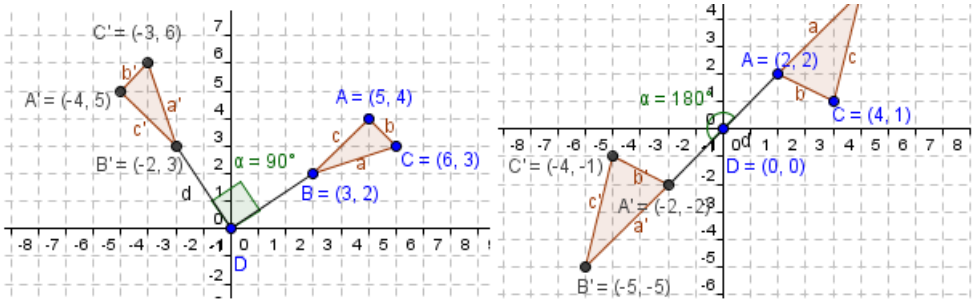
$$p'(1) = np_n - np_{n-1} = n\overrightarrow{p_{n-1}p_n}.$$

- Кривата е непрекъсната. Фукцията е непрекъсната по всяка от координатите, следователно е непрекъсната самата крива.

6. Ротация

Определение:

Дадени са точка O и насочен ъгъл. Преобразуване в равнината, при което на произволна точка X, различна от O, се съпоставя точка X', за която OX'=OX и ъгълът, сключен между OX и OX' е равен на дадения насочен ъгъл, се нарича ***ротация*** *(въртене). Точка O се нарича* ***център на ротацията****, а насоченият ъгъл се нарича* ***ъгъл на ротацията****.*



Фиг. 8

Означения (Фиг. 8):

A`(x`,y`) – точката преди преместването

A(x,y) – точката след преместването

Параметри:

$P_R(x_R,y_R)$ – точка на ротацията

α – ъгъл на ротация

$$\begin{cases} x` = x_R + ((x - x_R).\cos(\alpha)) - ((y - y_R).\sin(\alpha)) \\ y` = y_R + ((y - y_R).\cos(\alpha)) + ((x - x_R).\sin(\alpha)). \end{cases}$$

Друг запис на уравнението на ротацията е:

$\begin{bmatrix} x` \\ y` \end{bmatrix} = \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix} \left(\begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} x_R \\ y_R \end{bmatrix} \right)$, където

$\begin{bmatrix} x` \\ y` \end{bmatrix}$ - нова точка; $\begin{bmatrix} x \\ y \end{bmatrix}$ - точка за преобразуване; $\begin{bmatrix} x_R \\ y_R \end{bmatrix}$ - център на ротацията;

φ - ъгъл на ротация; R = $\begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix}$ - матрица на ротацията.

Всяка ротация е еднаквост. Следователно:

- Ротацията има единствена двойна точка - нейният център.
- Ротация с ъгъл "+" или "-" 180^0 няма двойни прави.
- Двойни окъжности на ротация са всички окръжности с център - центърът на ротацията.

Реализирани са следните ротации:

- Завъртане наляво на 90° на целия прозорец с център на ротацията - пресечната точка на диагоналите на прозореца. Координатите на новите точки се получават по формулите:

$$\begin{cases} x` = x_R + ((x - x_R).\cos(90^\circ)) - ((y - y_R).\sin(90^\circ)) \\ y` = y_R + ((y - y_R).\cos(90^\circ)) + ((x - x_R).\sin(90^\circ)). \end{cases}$$

- Завъртане на целия прозорец надясно на 90° с център на ротацията - пресечната точка на диагоналите на прозореца:

$$\begin{cases} x` = x_R + ((x - x_R).\cos(-90^\circ)) - ((y - y_R).\sin(-90^\circ)) \\ y` = y_R + ((y - y_R).\cos(-90^\circ)) + ((x - x_R).\sin(-90^\circ)). \end{cases}$$

- Завъртане на целия прозорец хоризонтално:

$$\begin{cases} x` = x_R + ((x - x_R).\cos(-90^\circ)) - ((y - y_R).\sin(-90^\circ)) \\ y` = y_R + ((y - y_R).\cos(-90^\circ)) + ((x - x_R).\sin(-90^\circ)). \end{cases}$$

- Завъртане на целия прозорец вертикално:

$$\begin{cases} x` = x_R + ((x - x_R).\cos(90^\circ)) - ((y - y_R).\sin(90^\circ)) \\ y` = y_R + ((y - y_R).\cos(90^\circ)) + ((x - x_R).\sin(90^\circ)). \end{cases}$$

Източници:

BRESENHAM'S ALGORITHM, Kenneth I. Joy
Bresenham's line algorithm, Wikipedia

Лекции по Числени методи, Борислав Боянов
Lectures on Approximation Theory, Applied and Computational Analysis group at the Department of Applied Mathematics and Theoretical Physics, Cambridge
A Practical Guide to Splines, Carl de Boor

Лекции - ОКГ ФМИ
http://matematika.martinmarinov.info/index.php?no=25

Лекция 3: Криви на Безие, доц. д-р Марта Теофилова
Curves and Surfaces for Computer Graphics – David Salomon
Лекции - ОКГ на ФМИ