UNIWERSYTET ŚLĄSKI WYDZIAŁ NAUK ŚCISŁYCH I TECHNICZNYCH INSTYTUT INFORMATYKI

Mateusz Liber

Bezier Levy C Curves

(projekt zaliczeniowy z przedmiotu Elementy Grafiki i Animacji 3D, temat nr 24.)

Sosnowiec 2021

1. Wstęp

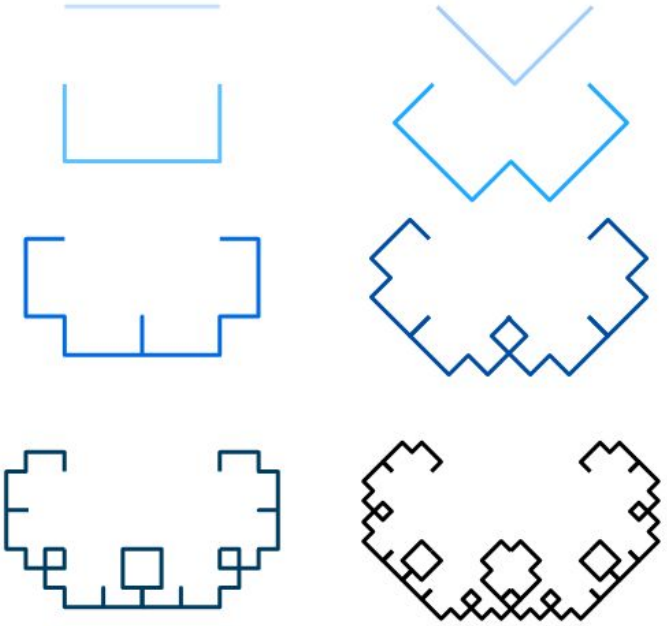
Program ma na celu wizualizację matematycznych **krzywych fraktalnych Lévy'ego (Lévy C curves)** za pomocą **kwadratowych krzywych Bézier'a (Quadratic Bézier curves)**. Program napisany jest z myślą o urządzeniach z systemem Android w wersjach od 6.0 wzwyż. Niniejsza aplikacja ma charakter naukowy pozwalający na zaobserwowanie i zrozumienie generowania powyższych krzywych, a także może zostać użyta w celach artystycznych do wizualizacji krzywych.

2. Teoria i Algorytmy

2.1. Krzywe Lévy'ego

C Krzywa Levy'ego to nieregularna, samopodobna struktura fraktalna zbudowana z trójkątów równoramiennych. Zaczyna się jako linia prosta, która jest podstawą trójkąta równoramiennego, a z każdą iteracją linia ta jest zastępowana przez dwa pozostałe boki trójkąta. Kolejne iteracje tworzące C Krzywą Levy'ego można zaobserwować na Rysunku 2.1. Standardowa krzywa Levy'ego zbudowana jest z trójkątów równoramiennych 45°. Krzywe te mogą być konstruowane przy użyciu trójkątów równoramiennych o kątach innych niż 45°, ale kąt musi być mniejszy niż 60°. Nowe linie wprowadzane na każdym etapie są krótsze od linii którą zastępują, więc proces budowy krzywej ma tendencję do osiągnięcia limitu granicznego. Zmiana kątów pozwala na tworzenie ciekawych wariacji tych fraktali.

W programie użyto rekurencyjny algorytm wyznaczający punkty kontrolne krzywej Levy'ego. Pseudokod opisujący działanie powyższej funkcji zaprezentowano w punkcie Algorytm 1. Należy zwrócić uwagę na konieczność dodania punktu startu przed rozpoczęciem działania funkcji. Punkty przechowywane są za pomocą struktury Punkt(x, y) przechowującej wartości współrzędnych kartezjańskich.



Rys 2.1. Wizualizacja C Krzywych Lévy'ego dla 8 powtórzeń.

Algorytm 1: Wyznaczanie punktów kontrolnych zadanej C Krzywej Levy'ego.

Wejście: Lista CP przechowująca punkty kontrolne krzywej w formacie Punkt(x, y)
Wejście: Ilość iteracji krzywej N
Wejście: Szerokość krzywej L
Wejście: Kąt krzywej R
Wejście: Kąt trójkąta składowego krzywej T
Wejście: Punkt startowy X
Wejście: Punkt startowy Y

```
do CP dodaj Punkt(X, Y)
funkcja get_control_points (N, L, R, X, Y):
        i = N

        Jeżeli (i > 0):
                L = L / sqrt(2)
                get_control_points(i - 1, L, R + T, X, Y)

                X = X + (L*cos(R + T))
                Y = Y + (L*sin(R + T))

                get_control_points(i - 1, L, R - T, X, Y)
        Inaczej:
                newX = X + L*toRadians(cos(R))
                newY = Y + L*toRadians(sin(R))
                do CP dodaj Punkt(newX, newY)
```

2.2. Krzywe Kwadratowe Bézier'a

Krzywa Béziera – parametryczna krzywa powszechnie stosowana w programach do projektowania inżynierskiego CAD, tworzenia grafiki wektorowej, reprezentowania kształtów znaków w czcionkach komputerowych i systemach przetwarzania grafiki oraz w grafice wektorowej. Kwadratowe Krzywe Béziera są łatwą w implementacji krzywą wyznaczoną przez odnajdywanie punktu, pomiędzy dwoma punktami podziału prostych utworzonych z kolejnych punktów kontrolnych krzywej.

Kwadratową krzywą Béziera otrzymamy kreśląc funkcję $B(t)$, mając punkty P_0, P_1 i P_2.

$$B(t) = (1-t)[(1-t)P_0 + tP_1] + t[(1-t)P_1 + tP_2]$$

Dla

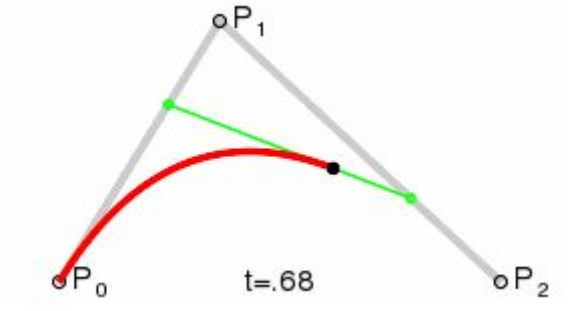
$$t \in [0,1].$$

Może być interpretowane jako interpolacja liniowa odpowiadających punktów na krzywej liniowej z P_0 do P_1 oraz z P_1 do P_2. Upraszczając wyrażenie otrzymamy:

$$B(t) = (1-t)2P_0 + 2(1-t)tP_1 + t2P_2$$

Dla

$$t \in [0,1].$$



Rys 2.2. Wizualizacja generowania kwadratowej krzywej Beziera.

Poniżej (Algorytm 2.) przedstawiono algorytm wyznaczający punkty kontrolne krzywej beziera utworzonej przy użyciu 3 podanych punktów i wartości podziału (rozdzielczości). Aby zachować ciągłość rysowania tak, aby cała krzywa sprawiała wrażenie narysowanej "bez odrywania długopisu", jako punkt kontrolny końca rysowania, do funkcji przekazywany jest wyznaczony punkt, leżący w połowie drogi do punktu końcowego. Poprzedni punkt końcowy staje się punktem środkowym następnej krzywej beziera.

Algorytm 2: Wyznaczanie punktów kontrolnych kwadratowej krzywej Beziera.

Wejście: Punkty kontrolne C krzywej P1, P2, P3, gdzie P3 jest punktem środkowym
Wejście: Podział krzywych t
Wejście: Lista przechowująca wyznaczone Punkty kontrolne CP

funkcja get_bezier_curve_points(P1, P2, P3, t):

```
for (i=0; i<=1; i+t):
        newX = pow(1 - t, 2)*P1.x + 2*(1 - t)*P3.x + pow(t, 2)*P2.x
        newY = pow(1 - t, 2)*y + 2*(1 - t)*P3.y + pow(t, 2)*P2.y
        do CP dodaj Punkt(newX, newY)
```

3. Opis Programu

Aplikacja napisana jest w języku Java. Platformą docelową są urządzenia z systemem Android w wersji co najmniej 6.0 wzwyż. Program napisany jest w języku angielskim. Instalacja aplikacji odbywa się przy pomocy wygenerowanego pliku w formacie apk, co wymaga zezwolenia urządzeniu na instalowanie aplikacji z nieznanych źródeł.

Użyto zewnętrznej biblioteki służącej do wyświetlania okna interakcji, umożliwiającej wybór kolorystyki elementów:

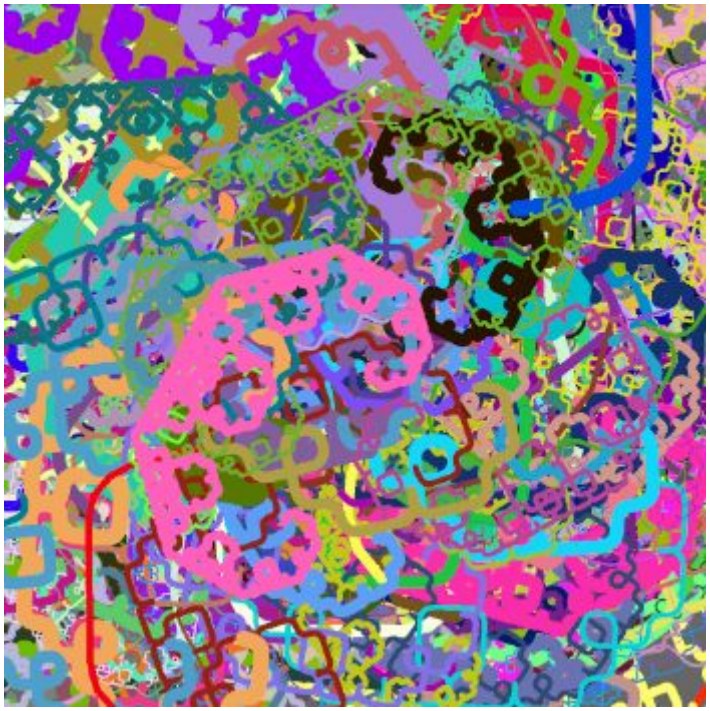
https://github.com/yukuku/ambilwarna

3.1. Możliwości Programu

Program umożliwia generowanie jednej lub wielu krzywych i dostosowywaniu ich parametrów. Dla każdej krzywej można określić ilość iteracji dla fraktalu Lévy'ego, kąt nachylenia krzywej, koordynaty pozycji od której zostanie zaczęte rysowanie krzywej, jej długość, szerokość linii tworzącej, a także kolor. Dodana została funkcja pozwalająca na wygenerowanie ustalonej ilości losowych krzywych. Krzywe przechowywane są w lokalnej bazie danych na konkretnym urządzeniu. Możliwe jest dodawanie, edycja i usunięcie krzywej. Ponadto program pozwala na zmianę ustawień takich jak kolor tła, określić wyświetlenie punktów kontrolnych i ich kolor, a także rozdzielczość krzywych tworzących.

Program nie zawiera bogatego systemu walidacji wprowadzonych ustawień. Od użytkownika wymagane jest wybranie takich ustawień, aby wykonanie programu było możliwe. Użytkownik może ustawić graniczne wartości i przetestować zachowanie programu. Zaleca się zwrócić uwagę na ustawioną ilość iteracji dla krzywych Levy'ego. Ilość

iteracji większa niż 11 znacząco wpływa na czas generowania krzywych, a więc na czas wykonania programu. W ramach testu przeprowadzono jednorazową generację nawet 1000 krzywych (Rys. 3.1), jednakże czas wykonania (który zależy także od mocy obliczeniowej używanego urządzenia) znacząco wzrósł. Parametrem który także ma znaczący wpływ na czasy wykonania jest rozdzielczość krzywych beziera tworzących krzywe.



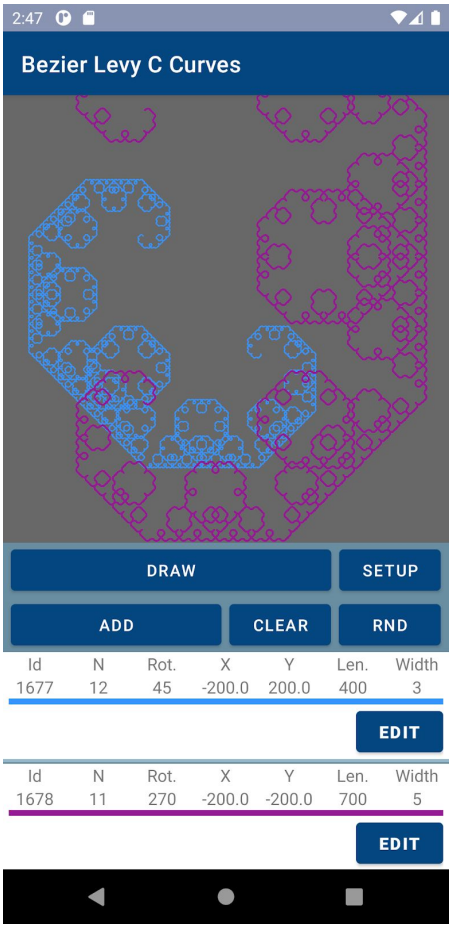
Rysunek 3.1: Wygenerowane 1000 losowych krzywych - przykład zastosowania artystycznego.

3.2. Opis Programu

Po uruchomieniu aplikacji przed użytkownikiem wyświetlony zostanie główny panel programu (Rys. 3.2). Przy pierwszym uruchomieniu, przed dodaniem krzywej do bazy danych, przyciski Draw (Rysuj), Clear (Wyczyść) będą zablokowane, a lista krzywych będzie pusta. Przyciski te zostaną odblokowane, a krzywe pojawią się na liście po dodaniu ich do bazy danych. Przyciski dostępne w menu głównym:

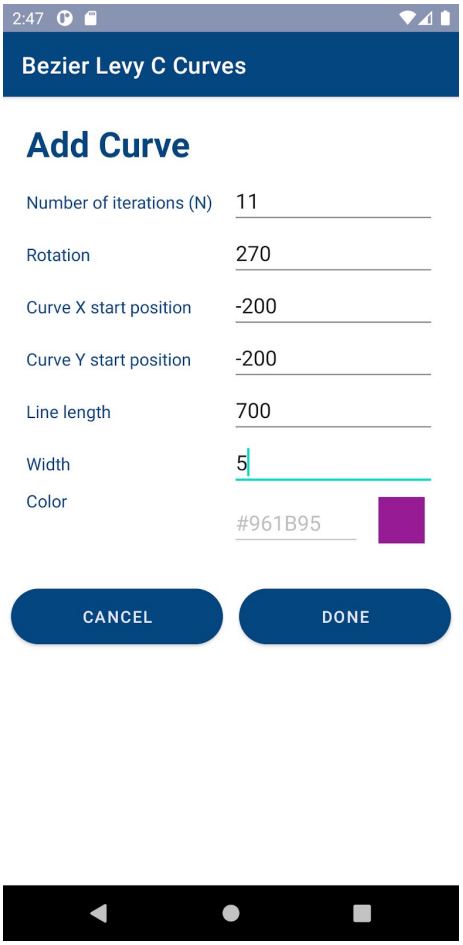
- **Draw (Rysuj):** Uruchamia główny cykl programu generujący pośrednie punkty kontrolne, a następnie przesyła je do funkcji rysującej krzywe.
- **Setup (Ustaw):** Uruchamia aktywność ustawień programu.
- **Add (Dodaj):** Uruchamia aktywność dodawania nowej krzywej do bazy danych programu.
- **Clear (Wyczyść):** Usuwa wszystkie krzywe z bazy danych i czyści obszar rysowania.
- **Rnd (Losuj):** Generuje losowe krzywe zgodnie z ustawieniami generatora.

Poniżej panelu z przyciskami na ekranie głównym aplikacji znajduje się lista krzywych. Po dodaniu krzywych wyświetlone zostaną podstawowe informacje o każdej z nich, a także przycisk umożliwiający wejście w aktywność edycji i usunięcia danej krzywej (przycisk Edit).



Rysunek 3.2. Widok na ekran główny programu.

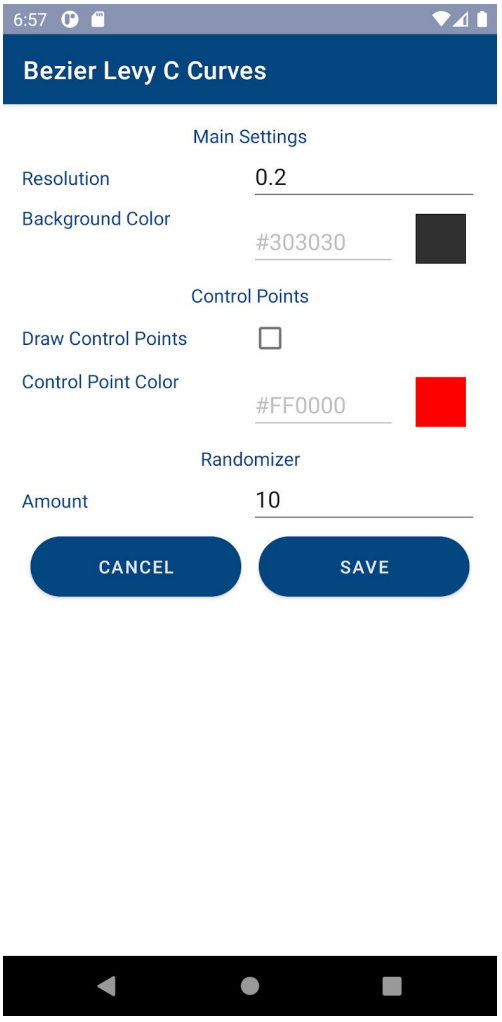
Aktywność dodawania nowej krzywej (patrz Rys 3.3) i edycji krzywej różni się od siebie tym, że w aktywności edycji znajduje się dodatkowy przycisk służący do usunięcia krzywej, a także po wejściu w aktywność edycji, pola uzupełniane są informacjami o wybranej, już istniejącej krzywej.



Rysunek 3.3. Ekran aktywności tworzenie nowej krzywej.

Aktywność ustawień aplikacji (Rys. 3.4) pozwala na dostosowanie kilku funkcjonalności podzielonych na sekcje:

- Main Settings (Ustawienia Główne):
  - Resolution (Rozdzielczość) - Wartość określająca podział krzywej beziera. Wartość powinna należeć do przedziału (0, 1). Mała wartość (np. 0.001) może mieć znaczący wpływ na wydajność wyświetlania krzywych i drastycznie zwiększyć ilość punktów kontrolnych tworzących krzywe.
  - Background Color (Kolor tła) - Określa kolor tła głównego rysunku.
- Control Points (Punkty Kontrolne):
  - Draw Control Points (Rysowanie Punktów Kontrolnych) - pole wyboru określające czy punkt kontrolne powinny być rysowane.
  - Control Point Color (Kolor Punktu Kontrolnego) - określa kolor wszystkich punktów kontrolnych dla każdej z wyświetlanych krzywych.
- Randomizer (Losowanie):
  - Amount (Ilość) - określa ile losowych krzywych zostanie utworzonych przy użyciu przycisku Rnd w panelu głównym.



Rysunek 3.4. Ekran aktywności ustawień aplikacji.

Bibliografia

[1] Quadratic Bézier Curve, Kuang-Hua Chang, in e-Design, 2015
https://www.sciencedirect.com/topics/engineering/quadratic-bezier-curve

[2] L´evy Curves, Kelsey Bates, June 8, 2015
https://sites.math.washington.edu/~morrow/336_15/papers/kelsey.pdf