Projekt Indywidualny

Implementacja (w wybranym języku) generatora linii prostej w przestrzeni 3W wg zadanego algorytmu i przestrzeni barw RGB

Artur Szajdecki

Warszawa, 30 Stycznia 2015

1 Wstęp

Dokumentacja projekt zawiera opis problemu, instrukcje, procedury, przykłady i bibliografie. Do projektu jest dołączony kod programistyczny i program wykonywalny napisany w C++ który generuje linie prostą w przestrzeni barw RGB na obrazku dzięki zaawansowanemu algorytmowi. Kod zawiera dokumentacje programistyczną.

2 Krótki Opis Problemu

Mamy układ 3W w przestrzeni RGB. Każdy punkt w tej przestrzeni odpowiednia jakiemuś kolorowi w RGB P(r, g, b) na przykład punkt P(255,0,0) odpowiada kolor czerwony. Współrzędne w tym układzie 3W również można wyrażać jako, P(x, y, z), oś x jest dla red, oś y jest dla green, a oś z jest dla blue. Na układzie 3W wybieramy dwa punkty (x_1, y_1, z_1) oraz (x_2, y_2, z_2) i chcemy aby odcinek (x_1, y_1, z_1)-(x_2, y_2, z_2) tworzył linie składający się z punktów/pikseli, z dużą precyzją. Więc problem sprowadza się do pokolorowania kolejnych pikseli, które leżą najbliżej punktu na odcinku o tej samej współrzędnej, co piksel. Nasz Algorytm mówi nam, kiedy należy wykonać ruch w kierunku wolnym(wektorów kierunków jest dużo $V^1 - V^7$, zależy one od wielkości zmiennych dx, dy, dz, i dzięki nim można ustalić jakie wektory się zawierają w pyramidach $O^1 - O^6$). Rozważmy równania różnic:

$$dx = (x_2 - x_1)$$

$$dy = (y_2 - y_1)$$

$$dz = (z_2 - z_1)$$

Przepuśćmy że mamy: $(x_2 - x_1) > (y_2 - y_1) > (z_2 - z_1)$ Niech L_{xy} i L_{xz} będą rzutami L na płaszczyźnie xy i xz. Będziemy budować linię od przyłożonego punktu środkowego leżących na liniach L_{xy} i L_{xz}. Teraz linia L_{xy} idzie od punktu $(x_1, y_1, 0)$ do $(x_1, y_1, 0)$ oraz linia L_{xz} idzie od $(X_1, 0, z_1)$ do $(x_2, 0, z_2)$. Niech d_{xy} będzie zmiennej decyzja dla linii L_{xy}, $posInc_{xy}$, $negInc_{xy}$ i dzięki niej będziemy wiedzieli kiedy inkrementować, dekrementować punkty. Podobnie jest z xz, niech d_{xz}, $posInc_{xz}$ i $negInc_{xz}$ dla linii L_{xz}. Kiedy metryka jest 26-połączeniowa, rozmiary zmiennych dx, dy, dz mają duże znaczenie, szuka się wtedy największej różnicy która "będzie się inkrementowała za każdym razem, kiedy będzie określany nowy punkt oraz można zauważyć że w trakcie zmiany x też się zmieniają y oraz z dzięki punktów środowych linii L_{xy} i L_{xz}.

3 Instrukcja Obsługi Programu

Pierw program wymaga żeby użytkownik podał współrzędne początkowe linii 3D, współrzędne końcowe linii 3D oraz podał kod RGB, współrzedne muszą być jako wartość typu integer z przedziału od 0 do 255, poniewaz będzie on generowany w przestrzeni RGB, później pogram w terminalu wyświetla po kolej zmienne punktów rgb które są odpowiedzialne za tworzenie linii, na koniec program wygeneruje, obrazek o nazwie 'Graph.png', na której jest układ 3W z linia w przestrzeni barw RGB.

4 Procedury

void RE PYRAMID(int x1, int y1, int z1, const int x2, const int y2, const int z2); Jest to procedura która generuje linie prostą w przestrzeni barw RGB. Algorytm tworzy 26-connected line w 3 wymiarach. Pierwsze 3 argumenty, prezentują punkt początkowy przestrzeni barw, a trzy pozostałe punkty odpowiadają punktowi końcowemu.

void RE LO1(int point[], int d xy, int d xz, int posInc xy, int posInc xz, int negInc xy, int negInc xz, int x2, int y2, int z2);Procedura zawiera pętle generującą po kolej punkty w przestrzeni 3W. Funkcja się wywoła kiedy warunek dx > dy > dz będzie się zgadzał.

void RE LO2(int point[], int d xy, int d xz, int posInc xy, int posInc xz, int negInc xy, int negInc xz, int x2, int y2, int z2);Procedura zawiera pętle generującą po kolej punkty w przestrzeni 3W. Funkcja się wywoła kiedy warunek dx > dz > dy będzie się zgadzał.

void RE LO3(int point[], int d yx, int d yz, int posInc yx, int posInc yz, int negInc yx, int negInc yz, int x2, int y2, int z2);Procedura zawiera pętle generującą po kolej punkty w przestrzeni 3W. Funkcja się wywoła kiedy warunek dy > dx > dz będzie się zgadzał.

void RE LO4(int point[], int d yx, int d yz, int posInc yx, int posInc yz, int negInc yx, int negInc yz, int x2, int y2, int z2);Procedura zawiera pętle generującą po kolej punkty w przestrzeni 3W. Funkcja się wywoła kiedy warunek dy > dz > dx będzie się zgadzał.

void RE LO5(int point[], int d zx, int d zy, int posInc zx, int po-

sInc zy, int negInc zx, int negInc zy, int x2, int y2, int z2);Procedura zawiera pętle generującą po kolej punkty w przestrzeni 3W. Funkcja się wywoła kiedy warunek dz > dx > dy będzie się zgadzał.

void RE LO6(int point[], int d zx, int d zy, int posInc zx, int posInc zy, int negInc zx, int negInc zy, int x2, int y2, int z2);Procedura zawiera pętle generującą po kolej punkty w przestrzeni 3W. Funkcja się wywoła kiedy warunek dz > dy > dx będzie się zgadzał.

void InitMGL(const int x1, const int y1, const int z1, const int x2, const int y2, const int z2); Jest to procedura inicjalizująca układ 3W, posługująca się metodami z biblioteki MathGL, argumenty ten funkcji prezentują wielkość układu w 3W, czyli 3 pierwsze parametry prezentują początek układu, a 3 pozostałe argumenty prezentują koniec układu.

void Zaznacz Na Przestrzeni Barw (int point[3]); Jest to procedura służąca do zaznaczania punktów na przestrzeni barw RGB, argumentem ten funkcji jest tablica point[], prezentująca punkty osi na przestrzeni.

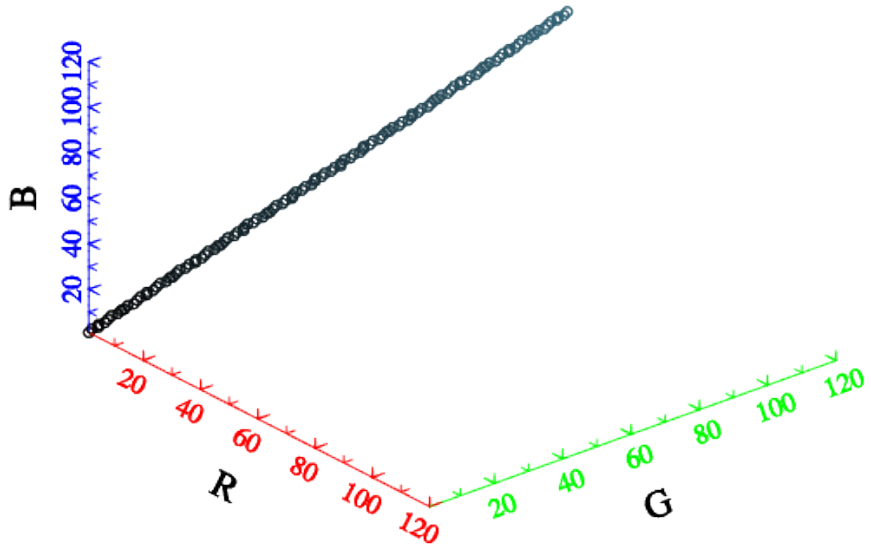
listing:

// Poniżej instrukcje już się wykonały w procedurze
// RE PYRAMID() i są przekazywane do procedury RE LO1:
//$dx := x_2 - x_1; dy := y_2 - y_1; dz := z_2 - z_1;$
//$d_x y := 2 * dy - dx; posInc_x y = 2 * dy; negInc_x y = 2 * (dy - dx);$
//$d_x z := 2 * dz - dx; posInc_x z = w * dz; negInc_x z = 2 * (dz - dx);$
//Ten listing prezentuje RE LO w przypadku kiedy warunek dx > dy > dz
//się spełnia.

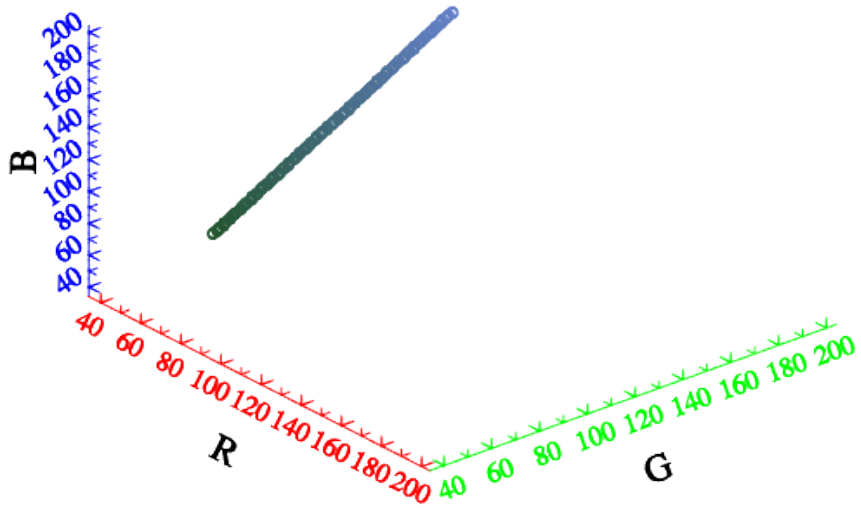
```
procedure RE LO1(int point[], int d_xy, int d_xz, int posInc_xy,
int posInc_xz, int negInc_xy, int negInc_xz, int x_2, int y_2, int z_2)
begin
      while x < x_1 do
            begin
                  if d_xy <= 0
                        then
                              begin
                                    d_xy := d_xy + posInc_xy;
                                    if d_xy <= 0
                                          then d_xz := d_xz + posInc_xz;
                                          else
                                                begin
                                                      d_xz := d_xz + negInc_xz;
                                                      z := z + 1;
                                                end
                              end
                        else
                              begin
                                    d_xy := d_xy + negInc_xy;
                                    if d_xz <= 0
                                          then d_xz := d_xz + posInc_xz;
                                          else
                                                begin
                                                      d_xz := d_xz + negInc_xz;
                                                      z := z + 1;
                                                end
                                    y := y + 1;
                              end
                        x := x + 1;
                        Zaznacz_Na_Przestrzeni_Barw(x,y,z);
            end
end;
```

5 Przykładowe Wygenerowane Linie

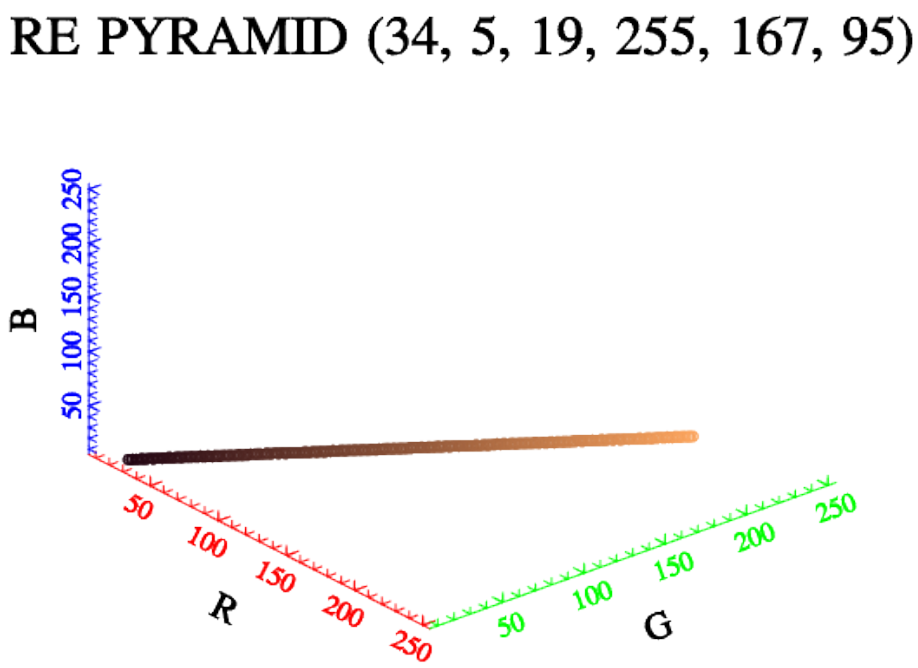
RE PYRAMID (1, 1, 1, 50, 100, 120)



RE PYRAMID (34, 86, 45, 101, 130, 204)



Powyżej są wygenerowane układy 3W w przestrzeni barw RGB. Jedna linia jest od punktu (1,1,1) do punktu (50,100,120), a druga od punktu (34, 86, 45) do punktu (101, 130, 204) w przestrzeni RGB. A tutaj kolejny przykład:



6 Bibliografie

J. E. Bresenham. Algorithm for computer control of a digital plotter. „IBM Systems Journal". 4 (1) (ang.). [zarchiwizowane z adresu 2008-05-28].

Michał Jankowski: Elementy grafiki komputerowej. Warszawa: Wydawnictwa Naukowo-Techniczne, 1990, s. 27-34. ISBN 83-204-1326-5.

Mokrzycki W.: A new algorithm for 3D line generation. Institute of Computer Science, Polish Academy of Sciences.