16.06.2024, Kraków

**WYKRESY 3D**

PROJEKT NR 007 - dokumentacja

Anna Nowak

Bartosz Fryska

Aleksander Kopyto

1. **Tytuł projektu i autorzy projektu**

W ramach kursu „Podstawy Grafiki Komputerowej” mieliśmy za zadanie wykonać **projekt o numerze 007 – „Wykresy 3D”** w grupie trzyosobowej.

Członkowie zespołu i podział pracy:

1. Anna Nowak
   1. zaprojektowanie i przygotowanie GUI projektu
   2. funkcjonalności związane z interakcją pomiędzy użytkownikiem,  
      a programem
   3. możliwość druku oraz zapisu wykresu w postaci zdjęcia  
      w formacie .png
   4. algorytm mapy konturowej
   5. przygotowanie dokumentacji
2. Bartosz Fryska
   1. algorytm rzutu perspektywicznego
   2. algorytm mapy konturowej
   3. obrót wykresu w postaci rzutu perspektywicznego
   4. generowanie wykresów w postaci rzutu perspektywicznego i mapy konturowej w oparciu o dane pobrane od użytkownika
   5. pomoc przy tworzeniu dokumentacji
3. Aleksander Kopyto
   1. ?
4. **Opis projektu**

Celem projektu było stworzenie programu generującego wykresy funkcji   
w postaci mapy konturowej lub rzutu perspektywicznego w zależności od wyboru użytkownika.

1. **Założenia wstępne przyjęte w realizacji projektu**
   1. **wymagania podstawowe**

Użytkownik ma możliwość wprowadzenia funkcji w postaci tekstowej, ustalenia jej przedziałów względem zmiennej x, y oraz wartości minimalnej i maksymalnej funkcji (zmin i zmax). W zależności od wyboru użytkownika możliwe jest generowanie wykresu w postaci rzutu perspektywicznego oraz mapy konturowej. Istnieje możliwość druku wykresu oraz zapisania go w postaci pliku graficznego w formacie .png.

* 1. **wymagania rozszerzone**

Wykres w postaci rzutu perspektywicznego można obracać za pomocą ruchu myszy po panelu, gdzie generowany jest wykres, przy jednoczesnym przytrzymaniu lewego klawisza myszy.

Interfejs użytkownika jest intuicyjny i przejrzysty, a nad polem tekstowym służącym do wprowadzania funkcji został umieszczony przycisk, którego naciśnięcie powoduje wyświetlenie informacji dotyczącej wskazówek co do poprawności formatu wprowadzanej przez użytkownika funkcji.

Poprawność wszystkich danych wprowadzanych przez użytkownika przed wygenerowaniem wykresu jest sprawdzana i w przypadku wystąpienia błędów pola tekstowe, gdzie znajdują się błędne dane, są czyszczone.

1. **Analiza projektu**
   1. **specyfikacja danych wejściowych**

Program przyjmuje od użytkownika funkcję, obszar zmienności dla argumentów x i y oraz wartość minimalną i maksymalną funkcji widoczną na wykresie w formie tekstowej (std::string). Następnie funkcja analizowana jest przy pomocy biblioteki TinyExpr, dzięki czemu może zostać wykorzystana do wygenerowania tablicy wartości badanej funkcji. Zakres argumentów x i y oraz zmin i zmax konwertowany jest do wartości typu double. Oprócz tego program umożliwia ustalenie formy wyświetlanego wykresu przy pomocy naciśnięcia przez użytkownika odpowiedniego wxRadioButton. Wszystkie z powyższych ustawień są możliwe do zmiany przez użytkownika dowolną ilość razy przez cały czas działania programu.

Na panelu dostępne są przyciski „generate” – generuje wykres, „print” – otwiera okno dialogowe służące do drukowania, „save” – otwiera okno dialogowe umożliwiające zapis pliku w formacie .png. Ponadto program pobiera dane na temat tego, czy mysz użytkownika znajduje się w polu gdzie generowany jest wykres i czy lewy przycisk myszy jest naciśnięty. Na tej podstawie możliwa jest manipulacja położeniem wykresu w postaci rzutu perspektywicznego.

* 1. **opis oczekiwanych danych wyjściowych**

Aplikacja generuje dowolny, wybrany przez użytkownika wykres w postaci rzutu perspektywicznego bądź mapy konturowej na panelu do tego przeznaczonym. Pozwala również na zapisanie wykresu w formacie .png oraz druk wykresu.

* 1. **zdefiniowanie struktur danych**

W celu przechowywania danych takich jak wartości funkcji uzyskane przy pomocy biblioteki TinyExpr (*GUIMyFrame1.h*), wykorzystano klasę C++ STL – vector. Jest to rodzaj kontenera, który ma zdolność do automatycznego powiększania swojej pojemności co czyni go bardzo elastycznym. Klasy vector użyto również do przechowywania odcinków służących do konstrukcji rzutu perspektywicznego (*perspectivic.h*) czy wartości punktów niezbędnych do generowania mapy (*map.h*).

W sytuacjach kiedy przez cały okres działania programu ilość przechowywanych danych była niezmienna w celu oszczędzenia używanej pamięci i optymalizacji działania programu wykorzystano tablice jedno- bądź dwuwymiarowe o stałej zdefiniowanej wielkości. Miało to miejsce przykładowo przy przechowywaniu punktów (x, y, z) dla rzutu perspektywicznego i mapy (*perspectivic.h*, *map.h*).

Do przechowywania danych odnośnie odcinków budujących rzut perspektywiczny wykorzystano struktury zawierające parametry punktów tworzących dane odcinki (*perspectivic.h**– struct Segment*). Natomiast chcąc uzyskać obrót generowanego wykresu skorzystano z techniki rzutu perspektywicznego, co wiązało się z użyciem macierzy transformacji zbudowanych w oparciu o tablice dwuwymiarowe.

* 1. **specyfikacja interfejsu użytkownika**

Interfejs użytkownika został zaprojektowany tak żeby był jak najbardziej intuicyjny  
i poruszanie się po nim nie sprawiało trudności użytkownikowi.

A screenshot of a computer

Description automatically generated

**2.**

**1.**

Zdjęcie 1. Okno aplikacji - start.

Istnieje możliwość zmiany wymiarów okna – wyświetlane wykresy regenerują się przy każdej zmianie wielkości panelu rysowania. Lewy panel (1.) dopasowuje się do każdego ustawionego przez użytkownika rozmiaru okna natomiast prawy panel (2.) ma ustaloną minimalną wysokość oraz szerokość co zabezpiecza przed złym rozmieszczeniem i zachodzeniem na siebie poszczególnych jego komponentów  
w przypadku zbytniego zmniejszenia okna przez użytkownika.

Po włączeniu aplikacji (zdjęcie. 1)

* 1. **wyodrębnienie i zdefiniowanie zadań**

Kolejne działania podjęte w celu realizacji projektu:

1. Analiza tematu projektu, dyskusja nad możliwymi rozwiązaniami w oparciu o wiedzę zdobytą podczas kursu „Podstawy Grafiki Komputerowej”.
   1. **decyzja o wyborze narzędzi programistycznych**
2. **Podział pracy i analiza czasowa**
3. **Opracowanie i opis niezbędnych algorytmów**
4. **Kodowanie**
   1. **testy niezależnych bloków**
   2. **testy powiązań bloków**
   3. **testy całościowe**
   4. **określenie niezmienników**
5. **Wdrożenie, raport i wnioski**