Projekt nr: 18 – Przebiegi czasowe

Tytuł projektu i autorzy

Celem projektu „Przebiegi czasowe” jest napisanie programu mogącego być pomocnym w późniejszych badaniach nad danymi zapisanych przez sprzęt laboratoryjny .

Program pozwala na zwizualizowanie pojedynczego parametru w funkcji, oraz na wykonywanie podstawowych pomocnych operacji dla wygodniejszej ich analizy. Dane odczytywane są z pliku dane.dat . Na potrzeby projektu plik jest generowany przez program „generator.exe ”.

Projekt zrealizowany został przez:

1. **Karolina Mizera**

   Obsługa pliku wejściowego oraz wczytywanie danych.

2. **Konrad Pasik**

   Budowa GUI, skalowanie oraz wyświetlanie przebiegów na panelu.

3. **Krystian Molenda**

   Schemat budowy programu, budowa GUI, bieżące wyświetlanie na przebiegów na panelu.

Opis projektu

Zakładane jest stworzenie programu, mającego generować na ekranie wykres funkcji przebiegu danego parametru w czasie na podstawie pliku o rozszerzeniu .dat.

W programie dostępne są dwa mody pracy:

mod 1 – Wyświetlanie aktualnych 10s przebiegu czasowego w kolorze czerwonym.

Mod 2 – Wyswietlanie aktualnych 10s przebiegu czasowego w kolorze czerwonym oraz poprzednich 10s w kolorze zielonym. Po wybraniu tego trybu w pierwszych 10s zielony wykres nie będzie widoczny. Tryb w takim przypadku uruchomi się automatycznie po pierwszych 10s.

Dodatkowo możliwe jest nałożenie pomocniczej siatki na panel, na którym rysowany jest przebieg.

Założenia wstępne przyjęte w realizacji projektu

1. Program przeznaczony jest na platformę Microsoft Windows.
2. Program wczytuje dane z pliku o rozszerzeniu .dat. Dane są ułożone w dwóch kolumnach, gdzie jedna to czas podany w sekundach a druga to parametr. Do pliku regularnie dopisywane są kolejne dane co 0.1s. W pliku maksymalnie znajduje się ostatnie 10s par – czas, parametr.

3. Po wciśnięciu przycisku START dane wyświetlane są na panelu w postaci wykresu ciągłej funkcji czasu (połączone punkty odczytane z pliku o rozszerzeniu .dat) . Przycisk STOP zatrzymuje akcje i zostawia widoczny na panelu ostatni przebieg.
4. Program działa zarówno w trybie okienkowym jak i pełnoekranowym. Powiększenie okna powoduje automatyczne wyskalowanie i powiększenie panelu wraz z wykresem przebiegu czasowego.
5. Możliwość działania w dwóch trybach pracy, które można dowolnie przełączać w czasie trwania programu. Wyświetlane jest tylko aktualne 10s przebiegu czasowego, lub jednocześnie aktualne oraz poprzednie 10s.
6. Opcja nałożenia pomocniczej siatki na panel z wykresem(wykresami) poprzez zaznaczenie odpowiedniej opcji w menu programu.
7. Dodatkowa opcja zresetowania danych wczytanych do pamięci przez program.
8. Wykres po dojściu do krawędzi panelu, na którym jest rysowany, automatycznie przesuwa się wraz z podpisami w lewą stronę robiąc miejsce dla nowych danych. Zasada działania zbliżona do tej znanej z oscyloskopów.

Analiza projektu

- ## Specyfikacja danych wejściowych

Dane wejściowe są generowane przez odpowiedni program „generator.exe” .

Program tworzy plik „dane.dat” , który docelowo zawiera dwie kolumny danych. Pierwsza kolumna oznacza czas a druga parametr przyjęty w tym czasie.
Początkowo plik jest pusty. Dane dodawane są stopniowo parami co 0.1s. Symuluje to pobieranie danych z generatora. W pliku znajduje się jednocześnie maksymalnie 100s danych. Następnie plik jest kasowany a w jego miejsce umieszczany nowy – pusty, w którym dane są zapisywane od nowa.

- ## Opis oczekiwanych dane wyjściowe

Program nie generuje nowych danych. Efektem jego działania jest wizualizacja bieżąco wczytywanych danych w postaci wykresu funkcji ciągłej w sposób podobny do tego znanego z oscyloskopów.

- ## Zdefiniowanie struktur danych

Dane są reprezentowane przez obiekt klasy Data. Klasa ta zawiera dwie zmienne typu double reprezentujące odpowiednio czas oraz wartość funkcji w danej chwili czasowej. Konstruktor domyślnie ustawia te zmienne jako zera. Jeśli w konstruktorze zadane zostaną parametry wejściowe to pierwszy parametr zostanie przypisany do zmiennej czasu, a drugi do zmiennej wartości funkcji. Klasa posiada metody dostępowe do zmiennych. Klasa zawiera metodę ustawiającą nowe wartości zmiennych. Klasa posiada metodę pobierającą wartości z pliku zadanego jako parametr wejściowy. Pierwsza pobrana z pliku wartość jest interpretowana jako czas, a druga wartość odpowiada wartości funkcji. Obiekt tej klasy jest zawierany przez klasę kontenerową DataContainer. Jako, że dane z obiektu klasy Data są dynamicznie zmieniane przy każdym odczycie, to w celu ich zachowania są przepisywane w do odpowiednich tablic punktów w klasie kontenerowej. Obiekt klasy kontenerowej przechowuje do 100 sekund danych, po czym tworzony jest nowy obiekt. Pozwala to na rozbudowę programu w przyszłości do zakresu rozszerzonego i zapamiętanie w partiach całego przebiegu funkcji. Klasa DataContainer posiada niezbędne metody dostępowe oraz metodę pozwalającą określić ostatni indeks tablicy, pod którym znajduje się dana. Odpowiada również za pracę na pliku z danymi.

- ## Specyfikacja interfejsu użytkownika

Zarówno program przebiegi_czasowe.exe jak i generator.exe jest przeznaczony na platformę Microsoft Windows.

W przypadku użycia systemu Microsoft Windows 10 należy wyłączyć antywirusa, Firewall oraz filtr Windows SmartScreen. Aplikacja nie jest podpisana cyfrowo co powoduje problemy i informacje o niechcianym oprogramowaniu.

Przed uruchomieniem programu przebiegi_czasowe.exe, w katalogu z programem, musi znajdować się plik „dane.dat". W przeciwnym wypadku program zakończy swoje działanie. Zalecanym jest uruchomienie generatora danych zapisującego je w opisanej w punkcie „Specyfikacja danych wejściowy" formie. Następnie można uruchomić program przebiegi_czasowe.exe.

Po uruchomieniu programu ukazuje się przejrzysty i prosty w obsłudze interfejs użytkownika.

Po lewej stronie znajduje się panel (obiekt klasy wxPanel), na którym wyświetlana będzie wizualizacja wczytywanych danych. W pierwszym momencie wyświetlone jest jedynie tło oraz osie. Służą one jako punkt odniesienia, więc podczas działania programu nie zmieniają swojego położenia.

Prawa strona programu składa się z elementów aktywnych, za pomocą których możliwa jest obsługa programu.

Od góry znajdują się trzy przyciski (obiekty klasy wxButton) :
START – rozpoczyna wczytywane danych z pliku. W przypadku braku pliku wykres nie wyświetla się.

STOP – zatrzymuje wczytywanie wykresu zostawiając widoczny przebieg zmienności wczytany dotychczas. Po ponownym wciśnięciu przycisku START, nie wczytane w czasie pauzy dane zostają trwale utracone.

RESET – Resetuje pamięć do, której wczytane zostały dane z pliku/plików.

Następnie znajdują się dwa guziki odpowiadające za mod pracy programu (mod1 oraz mod2, obiekty klasy wxRadioButton) .

MOD1 – Domyślnie wybrany mod pracy programu. Rysowane jest wyłącznie ostatnie 10s przebiegu czasowego na jednym ekranie.

MOD2 – Po wyborze uruchamia się drugi mod pracy programu. Rysowane jest ostatnie 10s w kolorze czerwonym oraz przedostatnie 10s przebiegu czasowego w kolorze zielonym na jednym ekranie.

Na samym dole znajduje się **kratka** (obiekt klasy wxCheckBox) po której zaznaczeniu na panel nanoszona jest pomocnicza siatka, rysowana cienką przerywaną linią. Tak jak osie służy jako dodatkowe punkty odniesienia.

- ## Wyodrębnienie i zdefiniowanie zadań

1. Stworzenie graficznego interfejsu użytkownika.
2. Stworzenie klasy kontenerowej, która posiadała tymczasową możliwość generowania prostych danych testowych.
3. Rysowanie testowych danych co 0.1s na obiekcie klasy wxPanel nie zaburzając możliwości używania menu. Nadanie podstawowych funkcjonalności przyciskom START, STOP, RESET, mod1, mod2, oraz rysowanie siatki pomocniczej.

4. Uzupełnienie klasy kontenerowej o jej docelowe funkcjonalności. Podmienienie metody generującej na wczytującą dane z pliku „dane.dat” oraz rysowanie rzeczywistych, docelowych danych wczytywanych na bieżąco z pliku.
5. Skalowanie rysowanego wykresu zależnie od wielkości okna oraz zaimplementowanie przesuwania wykresu po dojściu do krawędzi panelu.

- ### Decyzja o wyborze narzędzi programistycznych

Program napisany został w środowisku Microsoft Visual Studio 2017 oraz 2015. Zależnie od wersji używanego środowiska używany był kompilator Visual Studio 2015 - Visual C++ 14.0 lub Visual Studio 2017 - Visual C++ 14.1 . Ponadto do utworzenia GUI użyliśmy programu wxFormBuilder. Zdecydowaliśmy się na użycie wymienionych środowisk ze względu na ich znajomość oraz perspektywiczność, ze względu na używanie ich w wielu firmach i korporacjach.

Do budowy aplikacji wykorzystaliśmy bibliotekę wxWidgets. Pozwoliła ona nam na stworzenie użytecznego, prostego, i lekkiego interfejsu graficznego, jednocześnie nie ograniczając możliwości przy wykonywaniu zadania. Ostatecznie zarówno środowisko jak i biblioteka jest nam dobrze znana z zajęć.

Podział pracy i analiza czasowa

Krystian Molenda

Po otrzymaniu plików pozwalających na wyświetlenie interfejsu użytkownika rozbudowałem program o kolejne klasy, z której każda miała docelowo mieć swoją funkcjonalność. Stworzyłem niezbędne połączenia między klasami. Klasą główną całego programu, łączącą wszystkie komponenty jest MyInterface dziedzicząca po klasie MyFrame. W klasie MyInterface znajduje się implementacja wszystkich eventów dziedziczonych z klasy MyFrame, wskaźnik na obiekt klasy kontenerowej DataContainer oraz na obiekt klasy odpowiadającą za rysowanie wykresu – DrawingClass.

Następujący podział aplikacji na kilka klas, z czego każda jest niezależna i posiada wyodrębnioną funkcjonalność, pozwala na łatwiejsze rozwijanie programu w przyszłości. Jednocześnie struktura budowy programu staje się znacznie bardziej przejrzysta.

Kolejnym moim zadaniem było zrealizowanie wstępnego rysowania wykresu „na żywo” czyli jednocześnie z generowaniem/wczytywaniem danych. Do tego celu użyłem odpowiednio skonfigurowanego timera (obiekt klasy wxTimer). Po odpowiednim skonfigurowaniu pozwolił on na wczytywanie danych w sposób cykliczny co 0.1s, jednocześnie nie blokując możliwości używania menu programu. Jako, że dane są podawane w postaci dyskretnej, a przebieg funkcji czasowej w oscyloskopie jest w postaci ciągłej, zdecydowałem na łączenie kolejnych punktów odcinkami. Odległość między kolejnymi punktami jest niewielka, więc nie jest to praktycznie zauważalne.

Przybliżony czasz poświęcony na realizację zadań: 3 dni

Karolina Mizera

Stworzyłam klasę Data odpowiadającą za przechowywanie danych. Korzystając z biblioteki fstream opracowałam pracę z dynamicznie zmieniającym się plikiem czyli ich zapis i gospodarowanie pobranymi danymi w klasie Data oraz poprawny odczyt z pliku i konwersję danych w klasie DataContainer.

Przybliżony czas na realizację zadania: 1.5 dnia

Konrad Pasik

Stworzyłeś interfejs. Opisz swoje dokonania

Opracowanie i opis niezbędnych algorytmów

Kodowanie

Dane wczytane z pliku umieszczone są w tablicach typu double w obiekcie klasy DataContainer. Dodatkowym elementem tej klasy jest obiekt klasy Data, który dynamicznie przechowuje aktualnie wczytane dane z pliku. Dane te są dopiero przepisywane do tablic w klasie kontenerowej DataContainer.

Klasa MyInterface jest głównym elementem programu. Łączy ona wszystkie warstwy aplikacji. Dziedziczy po klasie MyFrame, definiuje obsługę eventów klasy MyFrame i zawiera wskaźniki na obiekty klasy DataContainer oraz DrawingClass. Klasa DrawingClass odpowiada za rysowanie na obiekcie klasy wxPanel. Odpowiada za rysowanie osi i podziałek, pozwala na rysowanie wykresów funkcji w dwóch modach oraz nakładanie siatki. Również w tej klasie znajdują się metody odpowiadające za skalowanie wykresów względem wielkości okna oraz ich przesuwanie po dojściu wykresu do krawędzi panelu.

Pełna dokumentacja techniczna znajduje się w załączonym pliku HTML. Została wygenerowana za pomocą Doxygen. Zawiera ona grafy wizualizujące zależności pomiędzy poszczególnymi elementami programu, oraz ich opisy.

Testowanie

Testowanie programu wykonywaliśmy na bieżąco w miarę jego budowy. Staraliśmy się każdy powstający element sprawdzać, początkowo na generowanych danych w samym programie, a później na rzeczywistych wczytywanych danych z pliku.

Ostateczne sprawdzenie programu przeprowadzaliśmy dla zmieniając funkcje czasu w generatorze. Dla wszystkich otrzymaliśmy poprawne wykresy.

Dla otrzymanego generatora nie zostały znalezione błędy w działaniu programu.

Sprawdziliśmy program pod kątem długotrwałego działania (3h). Nie stwierdziliśmy błędów w działaniu programu, lub utraty jego funkcjonalności. Dane rysowane były w sposób poprawny, taki sam jak na początku działania programu.

Podczas działania programu nie stwierdziliśmy utraty nad nim panowania, t.j. braku możliwości wyboru którejkolwiek opcji w menu. Wyjątkiem był brak możliwości zamknięcia okna programu. Problem został bardzo szybko rozwiązany. Test został wykonany ponownie. Tym razem nie stwierdziliśmy żadnych problemów.

Wdrożenie, raport i wnioski

Po uruchomieniu programu z niezależnymi danymi otrzymaliśmy zadowalającą wizualizację przebiegu zmienności funkcji w czasie. Nie nastąpiły żadne problemy z wyświetlaniem, funkcjami, które

program oferuje, czy błędami po długim czasie działania programu. Udało się osiągnąć wszystkie w pełni wszystkie wymagania określone jako podstawowe. W przyszłości możliwe jest rozbudowanie programu do wersji rozszerzonej, tj. obserwacja po czasie dowolnego fragmentu przebiegu oraz przybliżanie danych.