PROJEKT PROGRAMOWANIA KOMPUTERÓW, INFORMATYKA PRZEMYSŁOWA PROFIL PRAKTYCZNY, SEMESTR 3

UKŁAD AUTOMATYCZNEJ REGULACJI

Skład sekcji:

1) Patryk Kowal

2) Jakub Guzik

Podział obowiązków:

1. Patryk Kowal – funkcjonalność backend
2. Jakub Guzik – funkcjonalność frontend

Historia zmian projektu w stosunku do pierwotnego projektu:

1. Usunięcie klasy zbiorczej „Kolejki” oraz klasy „EI”.
2. Dodanie klas „SprzezenieZwrotne” i „GenWartZadana”.
3. Zmiana relacji z dziedziczenia na kompozycje między klasami w backend oraz zmiana sposobu komunikacji między nimi.
4. Zmiana relacji Frontend z backend na asocjację.
5. Dodanie klasy „Manager”, odpowiedzialnej za komunikacje Frontend i Backend.
6. Dodanie metod do zmieniania/resetowania ustawień obiektów w istniejących klasach.
7. Zmiana typów niektórych zmiennych.
8. Dodanie metod do pobierania wartości wyjściowych/wejściowych obiektów w istniejących klasach.
9. Dokonanie refaktoryzacji Backendu (Zmiana nazw zmiennych, metod, klas na bardziej zrozumiałe, itd.)
10. Zmiana wstępnego wyglądu GUI – dodanie wykresów do GUI (ich ilość zmieniała się wraz z postępem projektu. Ostatecznie są trzy). Znaczne zwiększenie ilości podawanych przez użytkownika zmiennych
11. Zmiana QtCharts na QCustomPlot.
12. Dodanie metod zapisu i odczytu z pliku w klasie „Manager”.
13. Dokonanie refaktoryzacji GUI (usunięcie niepotrzebnych linijek, zmiana nazw zmiennych i serii na odpowiadające sygnałom itd.)

Napotkane trudności i sposób ich rozwiązania:

1. Dynamiczna zmiana parametrów modelu ARX – trudnością była aktualizacja parametrów wymagała ponownej inicjalizacji kolejek. Rozwiązaniem było dodanie w metodzie setARX automatycznego dostosowania rozmiarów kolejek i ich inicjalizacja zerami.
2. Obsługa opóźnień w modelu ARX – trudnością było uwzględnię historii sygnału wejścia i wyjścia. Rozwiązaniem było użycie kolejek deque oraz przesunięcie kolejki Queue_U o opóźnienie modelu ARX.
3. Dynamiczne aktualizowanie wykresów - wykresy muszą być przewijane i automatycznie dopasowywać swoje osie do zmieniających się wartości

sygnałów. Rozwiązaniem była implementacja funkcji updateChart() z logiką przewijania i dopasowania zakresów

4. Generowanie i resetowanie wykresów - Przy resetowaniu symulacji konieczne jest odświeżenie wykresów oraz ich danych. Rozwiązaniem była implementacja w metodzie on_Reset_Button_clicked() logiki odpowiednio resetującej zarówno wykresy jak i symulację.
5. Napotkaliśmy problemy z przesuwaniem wykresów w QtCharts (odkryliśmy że przesuwają się one zależnie od rozdzielczości, co przy skalowaniu okna powodowało problemy). Rozwiązaniem było napisanie wykresów na nowo z użyciem QCustomPlot.

Czego nauczył się każdy członek sekcji:

Patryk Kowal – Jak implementować metody zdolne do dynamicznej zmiany parametrów obiektu - zarządzanie dynamiczną aktualizacją parametrów modelu ARX, regulatora PID oraz generatora wartości zadanej. Zarządzania kolejkami i tablicami dynamicznymi. Zarządzania komunikacją między różnymi komponentami aplikacji. Doboru odpowiednich typów zmiennych. Poprawy czytelności i modularności kodu (Refaktoryzacja). Przedewszystkim nauczyłem się projektowania backendu aplikacji.

Jakub Guzik - współpracy napisanego kodu z kodem przygotowanym przez osobę odpowiedzialną za inną część funkcjonalności programu. Korzystania z narzędzi wbudowanych w Qt z którymi wcześniej nie miałem styczności – QtCharts, a także rozwiązywania problemów z nimi napotkanych. W późniejszej fazie projektu nauczyłem się że QCustomPlot jest od nich lepszy i prostszy (dlatego też ostatecznie go wybraliśmy). Odpowiedniego nazywania zmiennych – przy wielu seriach na jednym wykresie odpowiednie nazwanie ich znacząco ułatwia pracę. Projektowania frontendu aplikacji.

Obaj nauczyliśmy się współpracy, refaktoryzacji kodu, oraz podstaw działania układu automatycznej regulacji.