

Sprawozdanie

Spis treści:

1. Treść zadania.....	2
2. Struktury danych.....	2
3. Wykorzystane wzory i metody.....	2
4. Obsługa programu.....	3
5. Format danych wejściowych.....	5
6. Ważniejsze zmienne.....	6
7. Opisy funkcji.....	6

1. Treść zadania

Zad 1.

Zaimplementować program zawierający GUI w środowisku WinAPI, który wczytuje, przetwarza i wizualizuje sygnał z akcelerometru robota (sygnał z osi X). Program ma usuwać składową stałą z sygnału, wyznaczyć i wyświetlać (w GUI) przyspieszenie, prędkość, drogę. W GUI należy dodać przyciski odpowiedzialne za wyświetlanie tych sygnałów na wykresie (należy umożliwić wyświetlanie wszystkich sygnałów jednocześnie).
Zad 1a: outputRobotForwardB01.log
Zad 1b: outputRobotForwardB02.log
Zad 1c: outputRobotForwardB03.log

2. Struktury danych

Program wykorzystuje tablice typu *double* zadeklarowane globalnie o rozmiarze 2500. Liczba ta jest dopasowana do wielkości układu współrzędnych.

`double tab[2500]` - przechowuje informacje wczytane z pliku

`double przysp[2500]` - przechowuje wartości przyspieszenia

`double predkosc[2500]` - przechowuje wartości prędkości

`double droga[2500]` - przechowuje wartości drogi

`double srednia[2500]` - przechowuje średnie wartości przyspieszenia z ostatnich 25 pomiarów

3. Wykorzystane wzory i metody

a) Wzory

$$1\text{ G} = 9,81\text{ m/s}^2$$

Wzór na prędkość:

$$v = v_0 + \int_{t_0}^{t_1} a(t) dt$$

Wzór na drogę:

$$s = s_0 + \int_{t_0}^{t_1} |v(t)| dt$$

Metoda lewych prostokątów do przybliżenia wartości całek:

$$\int_{x_0}^{x_n} f(x) dx = h \sum_{i=0}^{n-1} f(x_i)$$

$$h = \frac{x_n - x_0}{n} = 0,04s$$

n- liczba prostokątów

Prosta średnia ruchoma z ostatnich 25 prób:

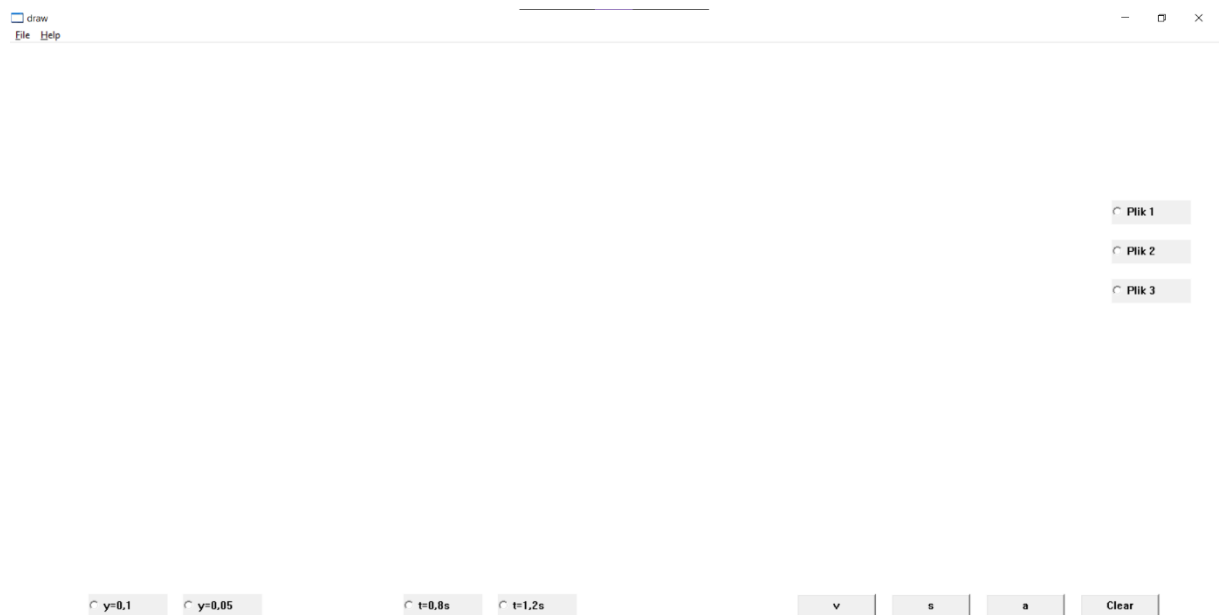
$$SMA = \frac{a_0 + a_1 + \dots + a_{24}}{25}$$

b) Rysowanie

- Najpierw decydujemy o liczbie rysowanych punktów, dzieląc liczbę danych przez t . Wartość podziałki na osi X można obliczyć ze wzoru $x_0 = 0,04 * t * 10$ (10 to liczba pikseli w podziałce; 0,04 to okres pomiędzy pomiarami).
- Następnie, rysujemy co t punkt pomnożony przez y (współczynnik amplitudy) i odjęty od 385 (to wysokość osi X). Wartość podziałki na osi Y można obliczyć ze wzoru $y_0 = \frac{10}{y}$ (10 to liczba pikseli w podziałce).

4. Obsługa programu

Wygląd po uruchomieniu:



Rysunek 1



Rysunek 2

Program posiada 4 kategorie przycisków:

a) Wybór amplitudy

- $y=0,1$ -> oznacza, że jedna podziałka na pionowej osi wynosi 0,1 danej jednostki (m/s dla prędkości, m dla drogi, m/s^2 dla przyspieszenia)
- $y=0,2$ -> oznacza, że jedna podziałka na pionowej osi wynosi 0,2 danej jednostki (m/s dla prędkości, m dla drogi, m/s^2 dla przyspieszenia)

b) Wybór przedziału czasowego

- $t=0,8s$ -> oznacza, że jedna podziałka na poziomej osi wynosi 0,8 sekundy
- $t=1,2s$ -> oznacza, że jedna podziałka na poziomej osi wynosi 1,2 sekundy

c) Wybór pliku

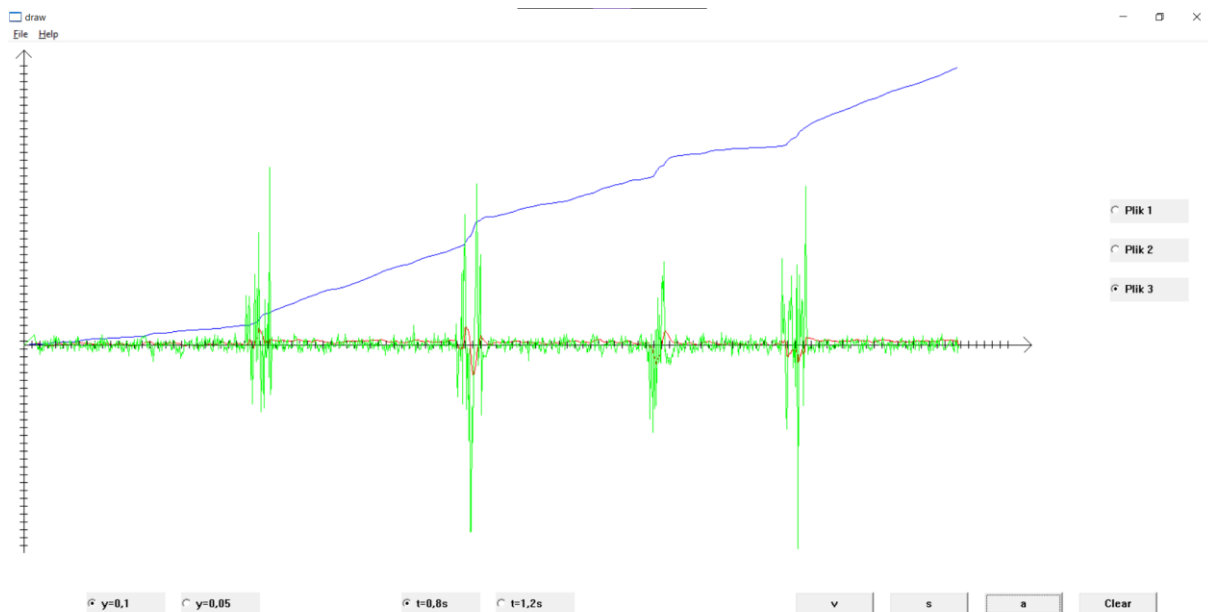
- Plik 1 -> otwiera plik outputRobotForwardB01.log
- Plik 2 -> otwiera plik outputRobotForwardB02.log
- Plik 3 -> otwiera plik outputRobotForwardB03.log

d) Wybór rysowania funkcji

- v -> rysuje wykres prędkości kolorem czerwonym
- s -> rysuje wykres drogi kolorem niebieskim
- a -> rysuje wykres przyspieszenia kolorem zielonym
- Clear -> czyści ekran, pozostawia sam układ współrzędnych

Na początku należy wybrać amplitudę, czas i plik, by móc narysować wykres. Każda zmiana amplitudy, czasu lub pliku (a także użycie przycisku Clear) zmienia wygląd okna na ten z rysunku 2. Zaleca się używanie programu w trybie pełnoekranowym.

Wykresy prędkości, drogi i czasu mogą się na siebie nakładać.



Rysunek 3

5. Format danych wejściowych

Program wczytuje dane z pliku tekstowego, który zawiera 12 kolumn liczb typu *double*. Program ten używa tylko kolumny 4, która musi zawierać przyspieszenie sensora wzdłuż osi X w G odczytywane co 0,04 sekundy.

outputRobotForwardA01.log — Notatnik											
Plik	Edycja	Format	Widok	Pomoc							
-0.000	0.000	-0.000	-0.024	-0.032	-1.008	0.150	-1.162	0.951	0.006	-0.002	-0.003
0.045	0.005	2.280	-0.016	-0.024	-1.000	0.150	-1.162	0.951	-0.004	-0.002	0.003
0.094	0.006	4.558	-0.016	-0.020	-0.992	0.145	-1.159	0.964	0.005	-0.005	0.011
0.147	0.022	6.809	-0.012	-0.028	-0.988	0.145	-1.159	0.964	0.004	-0.006	0.002
0.195	0.055	9.043	-0.012	-0.032	-0.996	0.145	-1.159	0.964	-0.002	0.004	0.003
0.255	0.073	11.253	-0.020	-0.032	-0.988	0.152	-1.153	0.953	-0.001	-0.002	0.002
0.308	0.084	13.435	-0.020	-0.024	-0.996	0.152	-1.153	0.953	0.003	-0.001	0.000
0.361	0.125	15.585	-0.016	-0.036	-0.984	0.152	-1.153	0.953	-0.002	0.005	-0.002
0.416	0.155	17.712	-0.016	-0.032	-0.992	0.152	-1.153	0.953	0.004	-0.002	0.005
0.457	0.189	19.813	-0.016	-0.028	-0.992	0.140	-1.162	0.951	0.000	0.006	0.003
0.507	0.223	21.876	-0.016	-0.032	-0.996	0.140	-1.162	0.951	0.004	-0.001	0.000
0.564	0.247	23.906	-0.024	-0.032	-0.988	0.140	-1.162	0.951	-0.001	-0.001	-0.002
0.603	0.278	25.882	-0.016	-0.028	-0.992	0.150	-1.171	0.964	0.003	0.000	-0.002
0.646	0.302	27.825	-0.020	-0.032	-0.992	0.150	-1.171	0.964	-0.006	-0.005	0.003
0.694	0.318	29.719	-0.024	-0.024	-0.996	0.150	-1.171	0.964	0.005	-0.002	-0.007
0.753	0.333	31.583	-0.032	-0.028	-0.992	0.155	-1.165	0.955	0.000	0.000	0.003
0.809	0.356	33.397	-0.032	-0.032	-1.000	0.155	-1.165	0.955	-0.004	0.003	-0.001
0.874	0.379	35.176	-0.036	-0.036	-0.996	0.155	-1.165	0.955	-0.006	-0.001	0.003
0.930	0.421	36.900	-0.028	-0.036	-0.996	0.155	-1.165	0.955	0.005	0.003	0.003
0.960	0.455	38.589	-0.020	-0.028	-0.984	0.137	-1.162	0.955	0.003	0.002	0.002
0.991	0.471	40.232	-0.024	-0.024	-0.992	0.137	-1.162	0.955	-0.001	-0.003	-0.002
1.016	0.499	41.834	-0.020	-0.024	-0.988	0.137	-1.162	0.955	0.005	0.000	0.008
1.022	0.510	42.360	-0.016	-0.020	-0.988	0.147	-1.153	0.963	0.000	0.003	0.000

Rysunek 4

Program jest w stanie zwizualizować plik zawierający do 2500 wierszy. W przypadku większej ilości danych wczytane zostanie jedynie 2500 początkowych wierszy.

Pliki z danymi znajdują się w folderze *draw*.

6. Ważniejsze zmienne

a) Zmienne globalne

`fstream plik`- służy do otwarcia i odczytania danych z wybranego pliku

`int ile`- przechowuje liczbę danych

b) Zmienne lokalne

`int odrz`- wykorzystywana w funkcji `wpis2()`, jest licznikiem w pętli wypełniającą tablicę `srednia[]`

`double pred, drog`- suma poprzednich policzonych wartości- odpowiednio prędkości i drogi, wykorzystywana w funkcji `wpis2()`

`double suma`- suma poprzednich 25 wartości przyspieszenia, wykorzystywana w funkcji `wpis2()`

`int ilep`- określa ile punktów zostanie narysowanych, wykorzystywana w funkcjach `RysPrzysp()`, `RysDroga()`, `RysPredkosc()`

`int pocz`- określa, od którego punktu ma zacząć się rysowanie, wykorzystywana w funkcjach `RysPrzysp()`, `RysDroga()`, `RysPredkosc()`

`UINT a, b, c, d`- służą do sprawdzania, który przycisk radiowy jest wciśnięty, wykorzystywane w funkcji `WndProc()`

7. Opisy funkcji

<code>void rysuj(HDC hdc)</code>
Funkcja, która rysuje układ współrzędnych. <u>Parametry:</u> <code>hdc</code> - uchwyt do kontekstu urządzenia
<code>void wpis(int ktory)</code>
Odczytuje dane z pliku i zapisuje je do tablicy <code>tab</code> i liczy ilość wierszy w pliku, jeżeli liczba wierszy przekracza 2500 to pomija resztę danych. <u>Parametry:</u> <code>ktory</code> - określa, który plik należy wczytać
<code>void wpis2()</code>
Liczy średnią z ostatnich 25 danych przyspieszeń, zamienia jednostkę z G na m/s^2 , a następnie zapisuje wyniki w tablicy <code>srednia</code> . Następnie liczy wartości przyspieszenia, prędkości i drogi bez wartości średniej i zapisuje je kolejno do tablic <code>przysp</code> , <code>predkosc</code> , <code>droga</code> . <u>Parametry:</u> brak

<code>void RysPredkosc(HDC hdc, int y, int t)</code>
Rysuje wykres prędkości od czasu. <u>Parametry:</u> hdc- uchwyt do kontekstu urządzenia y- odpowiada za skalę amplitudy wykresu t- odpowiada za zmianę wartości podziałki czasowej

<code>void RysDroga(HDC hdc, int y, int t)</code>
Rysuje wykres drogi od czasu. <u>Parametry:</u> hdc- uchwyt do kontekstu urządzenia y- odpowiada za skalę amplitudy wykresu t- odpowiada za zmianę wartości podziałki czasowej

<code>void RysPrzysp(HDC hdc, int y, int t)</code>
Rysuje wykres przyspieszenia od czasu. <u>Parametry:</u> hdc- uchwyt do kontekstu urządzenia y- odpowiada za skalę amplitudy wykresu t- odpowiada za zmianę wartości podziałki czasowej

<code>void czysc(HWND hWnd, HDC& hdc, PAINTSTRUCT& ps, RECT* drawArea)</code>
Czyści ekran, a następnie rysuje układ współrzędnych. <u>Parametry:</u> hWnd- uchwyt okna hdc- uchwyt do kontekstu urządzenia ps- struktura używana przez aplikację do rysowania w programie drawArea- wskaźnik na współrzędne lewego górnego i prawego dolnego narożnika prostokąta

<code>void repaintWindow(HWND hWnd, HDC &hdc, PAINTSTRUCT &ps, RECT *drawArea, int y, int t, int ktorywykres)</code>
hWnd- uchwyt okna hdc- uchwyt do kontekstu urządzenia ps- struktura używana przez aplikację do rysowania w programie drawArea- wskaźnik na współrzędne lewego górnego i prawego dolnego narożnika prostokąta y- odpowiada za skalę amplitudy wykresu t- odpowiada za zmianę wartości podziałki czasowej ktorywykres- decyduje, który wykres ma być narysowany

<code>int APIENTRY _tWinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPTSTR lpCmdLine, int nCmdShow)</code>
Funkcja główna. <u>Parametry:</u> hInstance- uchwyt aplikacji

hPrevInstance- uchwyt do poprzedniego wystąpienia aplikacji lpCmdLine- zawiera linię poleceń, z jakiej został uruchomiony program nCmdShow- określa stan okna programu
--

ATOM MyRegisterClass(HINSTANCE hInstance)

Zawiera informacje o klasie okna.

<u>Parametry:</u>

hInstance- uchwyt aplikacji

BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)
--

Tworzy przyciski.

<u>Parametry:</u>

hInstance- uchwyt aplikacji

nCmdShow- określa stan okna programu

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

Definiuje, co program ma wykonać w przypadku naciśnięcia przycisków.
--

<u>Parametry:</u>

hWnd- uchwyt okna

message- kod komunikatu
