Zad. 5: Sterowanie dronem

1 Cel ćwiczenia

Wykształcenie umiejętności modelowania kluczowych dla danego problemu pojęć. Tworzenie diagramu klas, czynności oraz przypadków użycia. Wykorzystanie dziedziczenia do modelowanie kilku różnych pojęć będących uszczegółowieniem bardziej ogólnego pojęcia. Wykorzystanie pól i metod statycznych.

2 Program zajęć

- *Ocena realizacji zadania z poprzedniego laboratorium* – ocenie podlega poprawność realizacji zadania, styl pisania programu oraz opisy.
- *Modyfikacja programu wg wskazań osoby prowadzącej* – ocenie będzie podlegała poprawność realizacji modyfikacji. Pracę nad modyfikacją programu (wszystkie operacje należy wykonywać na kopii) należy rozpocząć już w trakcie pierwszej fazy laboratorium, gdyż prowadzący nie będzie w stanie ocenić wcześniejszego programu wszystkim jednocześnie.
- *Realizacja wstępnej fazy prac nad nowym zadaniem* – w ramach wstępnej realizacji zadania należy zapoznać się z przykładem, który wizualizuje ruch dronu (dostępny jest w katalogu `~bk/edu/kpo/zad/z5/przyklad-animacji-drona`). W tym przypadku chodzi tylko o efekt. Kod przykładu nie jest pisany obiektowo. Dlatego też nie należy się na nim wzorować. Oprócz zapoznania się przykładem należy przerobić program z poprzedniego zadania w taki sposób, aby nowe obliczone współrzędne były zapisywane do osobnej tablicy wierzchołków. Dopiero dane z tej tablicy były zapisywane do pliku. Tak więc w oryginalnej tablicy będą wciąż pierwotne współrzędne wierzchołków. Natomiast sumaryczny kąt obrotu będzie musiał być pamiętany w osobnym polu klasy `Prostopadloscian`. To sprawi, że błędy obliczeń nie będą się akumulowały.
- *Ocena realizacji wstępnej fazy zadania*

3 Opis zadania programowego

Należy napisać program, który zwizualizuje położenie drona w przestrzeni (w tym celu należy wykorzystać moduł `lacze_do_gnuplota`). Zakładamy, że dron wyposażony jest w cztery wirniki. Program powinien umożliwiać zadawanie przez użytkownika obrotu drona i jego wizualizację, jak też ruchu na wprost na zadaną odległość z zadanym kątem wznoszenia (wartość dodatnia kąta) lub opadania (wartość ujemna). W celu interakcji z użytkownikiem program powinien udostępniać proste menu, za pomocą którego użytkownik powinien móc wykonywać następujące czynności:

- zmianę orientacji drona,
- zadanie ruch na wprost na zadaną odległość z zadanym kątem wznoszenia się lub opadania,

- wyświetlenie menu,
- zakończenie działania programu.

Dodatkowo program powinien wyświetlać wraz z menu informację o liczbie aktualnie istniejących obiektów klasy `Wektor3D` oraz łączną liczbę wszystkich obiektów klasy `Wektor3D` utworzonej do tej pory.

4 Przygotowanie do zajęć

Należy zastanowić się i wypisać listę podstawowych pojęć, które trzeba będzie zamodelować w postaci klas. Należy pamiętać, że nie chodzi o to, aby lista ta była długa. Ważne jest, aby dała właściwą podstawę do dalszego etapu definiowania niezbędnych struktur danych. Tak więc pojęć tych nie będzie dużo.

5 Wymagania co do konstrukcji programu

Oprócz wymagań sformułowanych w opisie zadania należy uwzględnić uwarunkowania przedstawione poniżej.

- Należy zauważyć, że wszystkie elementy drona mają swoją reprezentację graficzną (w późniejszym zadaniu również przeszkody). W zaproponowanym wariancie kadłub drona jest reprezentowany przez prostopadłościan, zaś wirniki przez graniastosłupy foremne sześciokątne. Do wspomnianych brył (prostopadłościany i graniastosłupy foremne sześciokątne) odnosi się bardziej ogólne pojęcie, a mianowicie *graniastosłup prosty*. Jeszcze bardziej ogólnym pojęciem jest *obiekt geometryczny*. Należy zaprojektować odpowiednią hierarchię dziedziczenia, która pozwoli wydzielić model bardziej ogólnego pojęcia. Klasa, która będzie modelowała to pojęcie, będzie wspólną częścią klas modelujących pojęcie poszczególnych elementów drona. Będą one tym samym specjalizacją tego bardziej ogólnego pojęcia.

  Wykorzystanie w tym zadaniu mechanizmów dziedziczenia jest obligatoryjne.

- Zbiór wierzchołków brył może mieć różną wielkość dla różnych wariantów brył (np. prostopadłościan lub graniastosłup sześciokątny foremny). Dlatego też, aby móc to uwzględnić, należy wykorzystać szablon `vector<>`.

- Należy również zwrócić uwagę, że w przypadku rysunku poszczególnych elementów drona, aby nie dopuścić do akumulacji błędów obliczeń związanych z obrotami, należy przechowywać współrzędne obiektów *wzorcowych* sprzed transformacji.

- Przelot drona z punktu do punktu powinien być animowany wraz z animacją obrotów wirników. Dotyczy to również obrotu.

- Program musi zachować strukturę modułową i odpowiednią strukturę kartotek. O ile będzie to konieczne, należy zmodyfikować plik `Makefile` (np. gdy dodany zostanie nowy moduł).

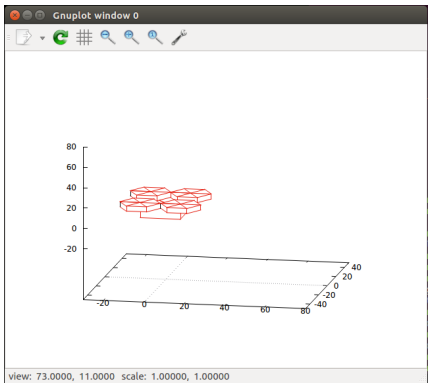
- Każda z klas powinna zostać zdefiniowana w oddzielnym pliku nagłówkowym. Metody tej klasy powinny być natomiast definiowane w osobnym module związanym z daną klasą. Proste metody można definiować bezpośrednio w ciele klasy.

- Dla poszczególnych klas należy przeciążyć niezbędne operatory działające na strumieniach. Nie wszystkie przeciążenia są w tym zadaniu potrzebne. Na pewno będą potrzebne przeciążenia operatorów wczytywania i zapisu dla klasy `Wektor3D`.
- Wszystkie metody, które nie zmieniają stanu obiektu, na którym działają, powinny być metodami typu **const**.
- Program powinien umożliwiać graficzną wizualizację wyników działania. Pozwala na to moduł łącza do programu `gnuplot`.
- Zarówno obrót jak też ruch robota powinien być animowany, tzn. obrót jak też ruch powinien być rozbity na sekwencję mniejszych obrotów lub ruchów w przód.
- Wszystkie klasy i metody oraz funkcje powinny zostać opisane. Opis ten powinien być zgodny z wymogami systemu `doxygen`. Ponadto należy wygenerować dokumentację w formacie HTML za pomocą programu `doxygen`.

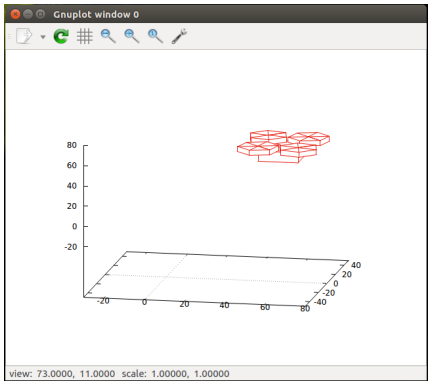
Oprócz tego pozostają w mocy wszystkie wcześniejsze wymagania dotyczące struktury katalogów, pliku Makefile, modułowej struktury programu, jak też opisów.

6 Przykład działania programu

```
jkowalsk@panamint> ./sterowanie-dronem

 Laczna ilosc stworzonych obiektow klasy Wektor3D: 184
       Ilosc istniejacych obiektow klasy Wektor3D: 26

  o - obrot drona
  j - lot na wprost
  w - wyswietl ponownie menu

  k - zakoncz dzialanie programu

 Twoj wybor (w - wyswietl menu)> j
```




```
 Podaj dlugosc drogi przelotu i kat wznoszenia (+)/ opadania (-).
            Dlugosc drogi: 85
 Kat wznoszenia [stopnie]: 45
```



```
Laczna ilosc stworzonych obiektow klasy Wektor3D: 372
      Ilosc istniejacych obiektow klasy Wektor3D: 26

Twoj wybor (w - wyswietl menu)> k

 Koniec dzialania programu.

jkowalsk@panamint> _
```

7 Materiały pomocnicze

W ramach materiałów pomocniczych został dostarczany program, który wizualizuje lot drona. Nie należy jednak wzorować się na jego kodzie, gdyż nie jest on pisany obiektowo, aby nie sugerować żadnych struktur danych. Źródła programu znajdują się w kartotece.

```
~bk/edu/kpo/zad/z5/przyklad-animacji-drona
```

8 Rozszerzenia

Możliwych jest kilka wariantów rozszerzenia tego zadania.

8.1 Zmiana szybkości lotu drona oraz rotacji wirników

Użytkownik powinien mieć możliwość zmiany szybkości lotu (długości elementarnego kroku w trakcie ruchu) oraz szybkości obrotu wirników.

8.2 Obrót drona z uwzględnieniem zmiany prędkości obrotu rotorów

Należy zwrócić uwagę, że wirniki po przekątnej mają ten sam kierunek obrotu, zaś sąsiednie przeciwny. Moment obrotowy drona, a więc jego obrót, realizowany jest poprzez zmniejszenie prędkości obrotów wirników po przekątnej, zaś obroty drugiej pary zostają w tej samej proporcji zwiększone. Jest to niezbędne, aby zachować ten sam ciąg. Rozszerzenie polega więc na zaimplementowaniu tego mechanizmu.

8.3 Skalowanie drona

Kształt drona powinno być można skalować. Tak więc jego obrys użytkownik powinien mieć możliwość powiększania lub pomniejszania.

9 Uproszenia

- Można zrezygnować z animacji ruch i obrotu robota. Wykorzystanie tego uproszczenia powoduje obniżenie oceny o $1,0$. Dobra struktura kodu oraz poprawnie stworzone opisy, jak też wygenerowana dokumentacja za pomocą `doxygen`, mogą podnieść ocenę. Jednak nie więcej niż o $0,5$.

- Można zrezygnować z wizualizacji wirników. To powoduje znaczne obniżenie oceny. Jednak jeżeli struktura programu jest poprawna, jak też dokumentacja oraz zastosowany jest mechanizm dziedziczenia, to możliwa jest ocena na poziomie $3,5$.

10 Wymagania i zarys programu zajęć w okresie realizacji zadania

10.1 Tydzień 0

Wymagania dotyczące tygodnia 0 zostały opisane w rozdziale 4.

10.2 Tydzień 1

Przed zajęciami muszą zostać przygotowane następujące elementy zadania. Wszystko co będzie ponad to będzie oceniane *in plus*.

- Pełny diagram klas w wersji elektronicznej. Można do tego wykorzystać program `dia`. Przykład jego użycia znajduje się w materiałach pomocniczych do zadania dostępnych na stronie.
- Przerobiona wcześniejsza wersja programu, która pozwoli zadawać przemieszczenie prostopadłościanu (ruch na wprost z zadanym kątem wznoszenia lub opadania) oraz jego obrót wokół własnej osi. Przemieszczenie i obrót nie muszą być animowane.
- Warunkiem koniecznym pozytywnej oceny jest poprawna kompilacja. W trakcie kompilacji nie powinny być generowane żadne ostrzeżenia.
- Dodatkowy kod powinien być opisany oraz powinna być wygenerowana dokumentacja za pomocą programu `doxygen`.

10.3 Tydzień 2

Rozliczenie się z gotowego programu i rozpoczęcie następnego zadania.