

# Dokumentacja i Opis Programu „Oblique-Throw-Sim”

## 1. Wykorzystane biblioteki

- 1.1. time – używa komendę sleep() do wyświetlania danych w symulacji co daną (stałą) jednostkę czasu, dzięki czemu możemy zauważyć czy ciało przyspiesza czy zwalnia w danym przedziale czasu.
- 1.2. turtle – pakiet graficzny, umożliwia wizualizacje okienkowe, dzięki czemu program posiada GUI użytkownika.
- 1.3. math – funkcje matematyczne, wykorzystane obliczenia to sin() i cos() konieczne do wyliczania parametrów pionowych i poziomych w zależności od kąta startu.

## 2. Klasa Obiekt

### 2.1. Elementy klasy

- list\_cox – lista przechowująca współrzędne X w czasie
- list\_cory – lista przechowująca współrzędne Y w czasie
- list\_velx – lista przechowująca prędkości poziome w czasie
- list\_vely – lista przechowująca prędkości pionowe w czasie
- list\_accx – lista przechowująca przyspieszenia poziome w czasie
- list\_accy – lista przechowująca przyspieszenia pionowe w czasie
- velocity – początkowa prędkość obiektu (domyślnie 0)
- acceleration – przyspieszenie (ciąg) generowane przez silnik
- timeofengine – czas od startu przez który silnik generuje ciąg
- startangle – kąt do powierzchni pod którym startuje rakiet
- k – współczynnik potrzebny do wyliczenia siły oporu powietrza (domyślnie 0.0005)
- cox – początkowe położenia X ciała (domyślnie jako 0)
- cory – początkowe położenie Y ciała (domyślnie jako 0)

### 2.2. Metody klasy

#### 2.2.1. def thrust(self):

Odpowiada za symulację lotu rakiety do momentu wyłączenia silników.

W oparciu o najświeższe dane z list przechowujących dane o obiekcie, funkcja oblicza aktualne wartości zmiennych (położenia, prędkości i przyspieszenia) i dodaje je na listę, po czym rozpoczyna kolejną pętlę już z nowymi danymi.

W pętli symulującej upływ czasu, każdy ruch jest obliczany jako ruch zależny od efektów poprzedniego. Zatem każdy ruch zaczyna się od parametrów na których poprzedni skończył. Konieczne było zrobienie tego w ten sposób ponieważ gdy w symulacji pojawia się opór powietrza mamy do czynienia z ruchem niejednostajnie przyspieszonym. I np. wzór na drogę w ruchu jednostajnie przyspieszonym  $S = vt + \frac{a \cdot t^2}{2}$  nie spełnia swojej roli licząc go dla czasu równego trwaniu lotu (ponieważ w trakcie zmienia się przyspieszenie układu).

#### 2.2.2. def freefall(self):

Odpowiada za symulację lotu rakiety od momentu wyłączenia silników do uderzenia z podłożem. Działa w sposób analogiczny do funkcji („thrust()”) tylko że uwzględnia dodatkowo czy ciało leci w górę czy w dół, oraz uwzględnia stałe przyspieszenie grawitacyjne  $g = 9,81$ . Przyspieszenie finalne jest różnicą między przyspieszeniem grawitacyjnym a tym wynikającym z oporu powietrza.

#### 2.2.3. `def save_datas_to_file(self):`

Odpowiada za zapis danych do pliku tekstowego o nazwie „datas.txt”. Dane zapisywane są bezpośrednio z list przechowawczych i są zaokrąglane do 4 cyfr po przecinku.

#### 2.2.4. `def simulate(self):`

Odpowiada za symulację. W pierwszym kroku program analizując dane położenia obiektu w czasie dostosuje skalę tak aby cały ruch po przeskalowaniu był widziany w okienku (aby uniknąć sytuacji w której punkt wychodzi poza przestrzeń widzianą przez użytkownika).

Następnie Tworzone są obiekty: scena, powierzchnia, punkt, z odpowiednimi parametrami co do położenia i rozmiaru.

Ostatnim krokiem jest pętla umieszczająca punkt zgodnie z danymi z list przechowujących dane o położeniu i robi to co daną jednostkę czasu. Aktualne położenie to kropka, a położenia poprzednie zapisywane są jako czarna linia.

Dodatkowo kropka zmienia kolor w zależności czy silniki są włączone. W chwili gdy silnik generuje ciąg kropka jest czerwona, po wyłączeniu silników kropka staje się zielona. Wszystkie parametry lotu wyświetlają się w dolnej części ekranu.

### 3. Używanie programu

#### 3.1. Proces uruchomienia

Program po uruchomieniu pyta użytkownika o parametry niezbędne do symulacji. Po wpisaniu parametrów program wykona niezbędne obliczenie i następnie wyświetli symulację trajektorii oraz zapisze jej dane do pliku.