Dokumentacja projektu

”Zagraj w hokeja z ładunkami elektrycznymi”

Ruciński Mateusz

Serweta Jakub

Nocoń Przemysław

Grupa: 5b

Kierunek: informatyka

June 2019

Spis treści

1 Cel projektu

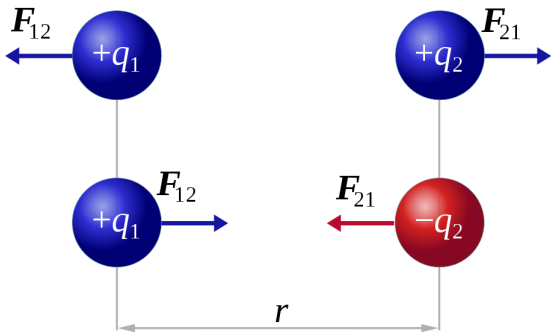
Celem jest stworzenie symulowanej komputerowo gry z naładowanym krążkiem (lub „piłką") na poziomej powierzchni, bez tarcia. Celem symulacji, podobnie jak w tradycyjnej grze w hokeja na trawie, jest zdobycie bramki przez wbicie krążka w sieć. Jednak tutaj krążek jest naładowany dodatnio (lub ujemnie) i porusza się tylko w wyniku wpływu innych naładowanych cząstek, które po umieszczeniu ich tam, gdzie chcesz, są „przyklejone" na powierzchni gry. Gra może być utrudniona przez dodanie przeszkód, które będą musiały zostać ominięte przez krążek. Trafienie w przeszkodę zatrzymuje grę z niepowodzeniem, natomiast trafienie krążkiem do bramki kończy symulację.

2 Model fizyczny

2.1 Prawo Coulomba

Do przeprowadzenia symulacji siły ładunków elektrycznych, wykorzystamy znane prawo fizyczne, prawo Coulomba.

Prawo Coulomba – mówi, że siła wzajemnego oddziaływania dwóch punktowych ładunków elektrycznych jest wprost proporcjonalna do iloczynu tych ładunków i odwrotnie proporcjonalna do kwadratu odległości między nimi. Siła oddziaływania ładunków jest siłą centralną i zachowawczą.



Rysunek 1: Prawo Coulomba

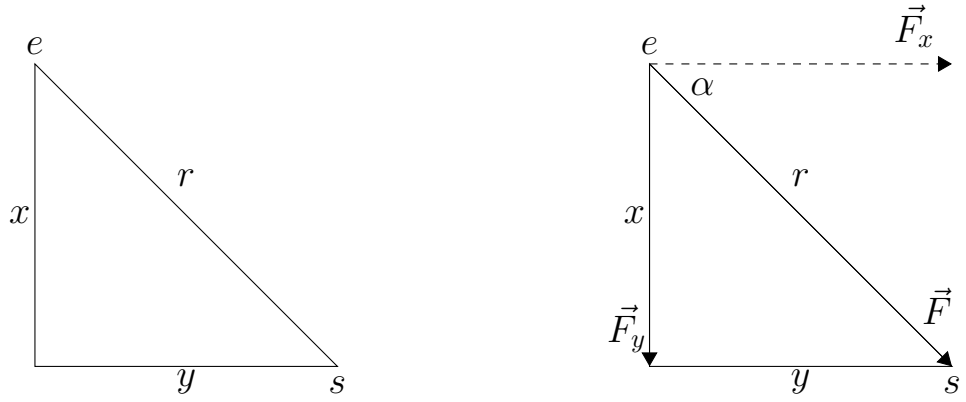
$$|F_{12}| = |F_{21}| = k\frac{|q_1 \times q_2|}{r^2} \tag{1}$$

gdzie
$k = 8,9875 * 10^9 Nm^2C^{-2}$ – stała oddziaływań ładunków elektrycznych w próżni
q_1, q_2 – ładunki ciał oddziałujących na siebie
r – odległość między ciałami

2.2 Uproszczony model fizyczny

Oparcie modelu fizycznego, naszego programu, o Prawo Coulomba było by bardzo trudne do wykonania. Jest to spowodowane tym, że potrzebujemy dynamicznie symulować siły, jakie działają na naładowany krążek. Dlatego postanowiliśmy zastosować uproszczony model, który ułatwi implementację symulacji.

Nasz model jest oparty na trójkącie prostokątnym, gdzie środek elektronu to – **e**, natomiast środek źródła to – **s**.



Rysunek 2: Rozkład sił oraz oznaczenia

Ładunki elektronu q_1 oraz źródła q_2 są stałe, dlatego jedyną zmienną, od której zależy siła Coulomba $\vec{F}$, jest **r** – odległość pomiędzy ładunkami. Odległość ta jest liczona ze wzoru na odległość między dwoma punktami.

$$r = \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2} \tag{2}$$

gdzie
x_a, x_b – współrzędne punktu **e**
y_a, y_b – współrzędne punktu **s**

Siłę $\vec{F}$ rozkładamy na dwie składowe $\vec{F_x}$ prostopadłe do osi układu współrzędnych oraz $\vec{F_y}$ równoległe do osi układu współrzędnych.

Do obliczenia wartości składowych siły $\vec{F}$ wykorzystujemy funkcje trygonometryczne.

$$\sin\alpha = \frac{\vec{F_x}}{r} = \frac{x}{r} \qquad\qquad \cos\alpha = \frac{\vec{F_y}}{r} = \frac{y}{r} \tag{3}$$

Zakładając, że wartości ładunków q_1, q_2 są takie same możemy je pominąć. Łącząc wzór na Prawo Coulomba (1) oraz wyprowadzone wzory (3), możemy obliczyć składowe $\vec{F_x}, \vec{F_y}$.

$$\vec{F_x} = \frac{\sin\alpha}{r^2} \qquad\qquad \vec{F_y} = \frac{\cos\alpha}{r^2} \tag{4}$$

Wiemy, że wzór na siłę to: $F = m * a \Rightarrow a = \frac{F}{m}$ obliczamy potrzebne nam przyśpieszenia a_x oraz a_y dla sił składowych.

$$a_x = k * \frac{\sin \alpha}{m * r^2} \qquad a_y = k * \frac{\cos \alpha}{m * r^2} \tag{5}$$

3 Implementacja

Nasz projekt wykonaliśmy przy pomocy języka programowania jakim jest Java. Interfejs graficzny programu oraz zamodelowanie ruchu zostało wykonane przy pomocy JavaFX.

W tej sekcji dokumentacji przedstawiona jest tylko ogólna budowa programu oraz krótkie opisy poszczególnych klas i metod. Bardziej szczegółowe opisy znajdują się w dokumencie generowanym za pomocą programu JavaDoc.

3.1 Klasy programu

Program składa się z 3 głównych klas:

1. Main – odpowiada za stworzenie interfejsu graficznego, a także wystartowanie algorytmu.
2. Electron – definiuje zmienne określające właściwości fizyczne:
   - Electron – elektronu
   - Source – źródeł pola elektrycznego
3. Controller – w tej klasie znajduje się implementacja całego algorytmu, metody odpowiadające za symulowanie pola elektrycznego oraz ruchu elektronu w tym polu.
4. oraz pliku sample.fxml – szczegółowo definiuje wygląd interfejsu graficznego

3.2 Najważniejsze metody klasy Controller

- **AnimationTimer timer = new AnimationTimer(){}** – główna funkcja odpowiadająca za wykonanie symulacji w czasie
- **DetermineForce** – definiuje rozkład sił pola elektrycznego

  private double[] DetermineForce(ArrayList<Source> sources, AnimationTimer timer)

  Dynamicznie oblicza:
  - odległość **r** pomiędzy elektronem, a źródłem (2)
  - $\sin \alpha$ oraz $\cos \alpha$ potrzebne do policzenia sił $\vec{F_x}$, $\vec{F_y}$ ze wzorów (3) wyprowadzonych wcześniej
  - przyśpieszenia składowe a_x oraz a_y ze wzorów (5)

  Wyniki zapisywane są na bieżąco w dwuelementowej liście ***ArrayList<Source> sources***.

  Metoda ta sprawdza również, czy nie doszło do kolizji elektronu z ładunkiem elektrycznym, co powoduje, że gra kończy się porażką.

- **addRectangles** – tworzy przeszkody na planszy dla poszczególnych poziomów trudności.

  private void addRectangles()

- **collision_checker** oraz **boundary_checker**

  private void collision_checker(AnimationTimer timer)

  private void boundary_checker(AnimationTimer timer)

  Metody odpowaidające za sprawdzanie, czy nie doszło do kolizji elektronu z przeszkodą utrudniającą grę lub granicami pola gry. Jeżeli tak gra kończy się porażką.

  Do wykrywania kolizji, wykorzystywana jest metoda *intersect* klasy *Shape*, po której dziedziczą wszystkie obiekty.

- **goal_check** oraz **win**

  private void goal_check(Rectangle post1, Rectangle post2, Rectangle backpost, AnimationTimer timer)

  Odpowiada za sprawdzenie, czy elektron trafił do bramki. Sprawdzane jest to poprzez wykrycie kolizji z tylą ścianą bramki ***backpost***. Wykorzystywana jest do tego metoda *intersect* klasy *Shape*. Jeżeli natomiast elektron uderzy w boczne ściany ***post1, post2*** zostanie odbity.

  private void win(AnimationTimer timer)

  Jeżeli poprzednia metoda wykryje kolizję z ***backpost*** gra kończy się sukcesem.

Literatura

[1] Halliday David, Resnick Robert, and Walker Jearl. *Podstawy fizyki*. Wydawnictwo Naukowe PWN, Warszawa, 2015.

[2] Dubson Michael. Electric field hockey simulation. PhET Interactive Simulations https://phet.colorado.edu/en/simulation/legacy/electric-hockey.

[3] Z.P. Piątek and P. Jabłoński. *Podstawy teorii pola elektromagnetycznego*. Wydawnictwa Naukowo-Techniczne, 2010.