

PODSTAWY SZTUCZNEJ INTELIGENCJI

DOKUMENTACJA KOŃCOWA

---

## Interaktywny projektant soczewek

---

*Autorzy:*

Arkadiusz Szlachetka

Maciej Jagiełło

Maciej Kucharski

Semestr 13Z

# 1 Kluczowe decyzje projektowe

## 1. Interakcja z użytkownikiem

Podczas uruchomienia programu, użytkownik jest pytany o parametry konieczne do przeprowadzenia symulacji, tj. ilość symulowanych promieni, ilość segmentów, z których składa się soczewka, wartość współczynnika załamania dla materiału, z którego jest wykonana soczewka, tolerancję zerowania skupienia elementów oraz minimalną wartość zaburzenia losowego. Następnie symulacja rozpoczyna się i jest to zarazem koniec interakcji z użytkownikiem.

## 2. Zastosowany algorytm ewolucyjny

W programie zaimplementowany algorytm ewolucyjny 1+1. W każdej iteracji utrzymywana jest jednoelementowa populacja. Następnie przeprowadzane jest krzyżowanie, w wyniku którego jest tworzona nowa soczewka. Jeśli wartość jej funkcji przynależności jest korzystniejsza, niż dla starej, to jest ona przekazywana do następnej generacji, w przeciwnym wypadku przekazywana jest stara. W przypadku znalezienia soczewki o zadowalającym wyniku, bądź wykrycia, że aktualnie używana wartość elementu losowego jest mniejsza od założonej minimalnej, symulacja jest przerywana.

## 3. Wybrany punkt skupiania

Założony punkt skupiania nie podlega wyborowi przez użytkownika. Został arbitralnie wybrany jako jeden ze znajdujących się na osi układu współrzędnych.

## 4. Przyjęta funkcja przystosowania

Za wartość funkcji przystosowania uznawana jest suma kwadratów odległości załamanych promieni od założonego ogniska soczewki. W przypadku wystąpienia zjawiska całkowitego wewnętrznego odbicia, dany promień nie jest przetwarzany i do wyniku soczewki zostaje dodana bardzo duża liczba. Wartość funkcji przystosowania jest minimalizowana.

# 2 Opis struktury programu

## 2.1 Architektura

Program został przygotowany zgodnie z wzorcem projektowym Model-View-Controller. W pakiecie `mma.psz.model` znajdują się klasy odpowiedzialne za odzorowanie soczewek i promieni.

Pakiet `mma.psz.view` zawiera klasy zajmujące się wizualizacją danych.

Pakiet `utils` to głównie klasy realizujące istotne, jednak poboczne z punktu widzenia zadania, zagadnienia.

Klasa `mma.psz.controller.Controller` zawiera główną pętlę programu, pośredniczy w wymianie danych między warstwą modelu i widoku oraz interpretuje otrzymywane wyniki. Główną klasą programu jest klasa `mma.psz.LensDesigner`. Tworzy ona kontroler i uruchamia pętlę programu.

## 2.2 Narzędzia poboczne

W programie są wykorzystywane zewnętrzne biblioteki, stąd celem łatwiejszego zarządzania zależnościami wykorzystany został program Apache Maven. Aby zbudować uruchamialną wersję programu (bez IDE) należy, znajdując się w katalogu programu, wpisać w terminalu:

```
mvn clean package -Dmaven.test.skip=true
```

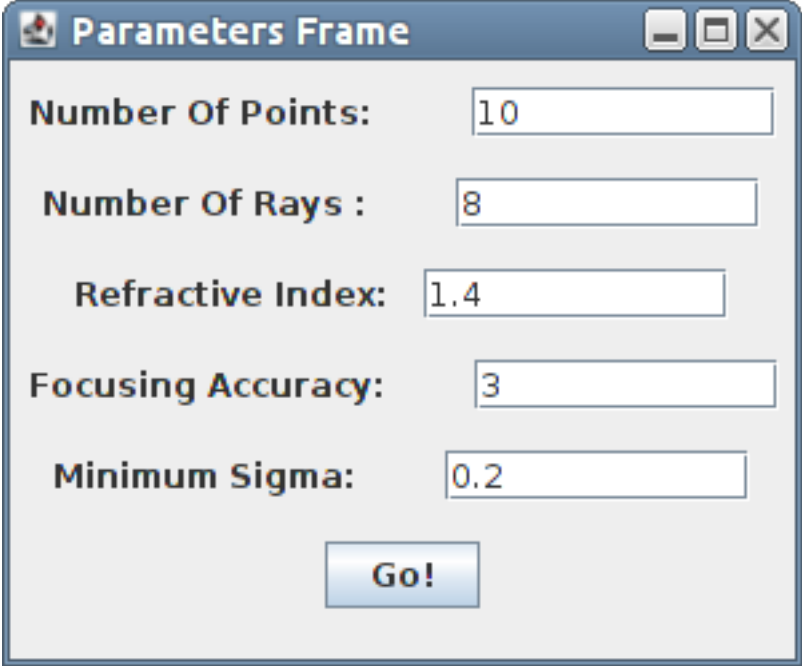
Po zbudowaniu aplikację można uruchomić poprzez wpisanie polecenia:

```
java -jar target/LensDesigner-1.0-SNAPSHOT.jar
```

Uruchomi się wtedy okno programu, w którym należy podać dane niezbędne do przeprowadzenia symulacji. Po tym okienko wprowadzania danych zniknie i pojawi się okno prezentujące przebieg symulacji.

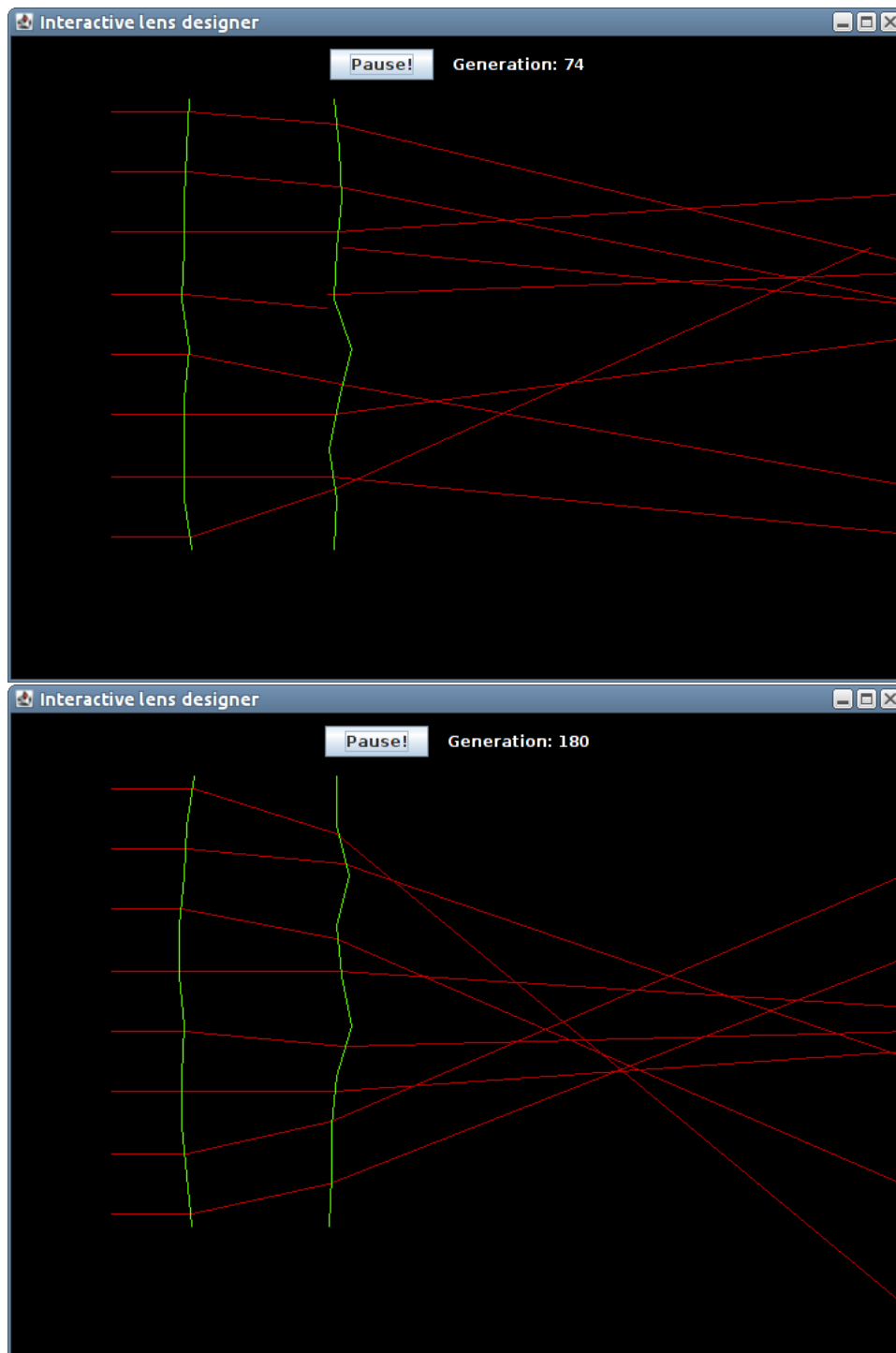
## 3 Wyniki testowania

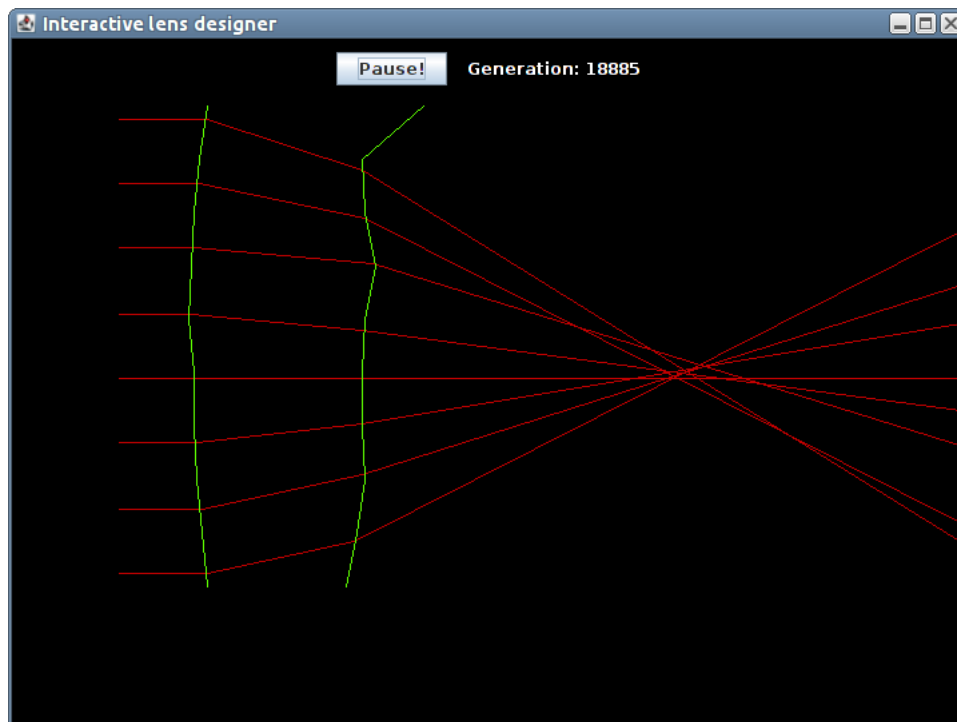
Dla parametrów wejściowych:



The image shows a Java Swing window titled "Parameters Frame". It contains five labeled text input fields arranged vertically. The labels and their corresponding values are: "Number Of Points:" with "10", "Number Of Rays :" with "8", "Refractive Index:" with "1.4", "Focusing Accuracy:" with "3", and "Minimum Sigma:" with "0.2". Below these fields is a single button labeled "Go!". The window has standard OS window controls (minimize, maximize, close) in the top right corner.

uzyskane zostały następujące wyniki:





## 4 Wnioski

Algorytm ewolucyjny okazuje się być dobrym rozwiązaniem problemu w sytuacji, kiedy ciężko jest bezpośrednio zminimalizować wartość funkcji przystosowania. Możliwe jest jednak nie uzyskanie rozwiązania ze względu na istnienie wielu minimów lokalnych. Pomysłem na obejście tego problemu byłoby np. zastosowanie algorytmu niezależnie, w kilku wątkach dla różnych wartości początkowego zaburzenia losowego.