Ewolucyjny malarz

Rafał Kwiatkowski, Franciszek Sioma

24 kwietnia 2020

1 Opis projektu

1.1 Cel projektu

Celem projektu *Ewolucyjny Malarz* było stworzenie aplikacji wykorzystującej algorytm ewolucyjny. Aplikacja przyjmuje na wejściu kolorowy obrazek, a na wyjściu zwraca możliwe jak najdokładniejsze odwzorowanie obrazu wejściowego przy użyciu prostokątów RGBA.

1.2 Przyjęte założenia

W algorytmie ewolucyjnym istotne jest to, aby zdefiniować czym będzie *Osobnik*. W naszym przypadku osobnikiem jest obraz stworzony z n prostokątów RGBA. Każdy prostokąt składa się z 8 wartości:

R kanał czerwony

G kanał zielony

B kanał niebieski

A kanał alpha (mówi o przezroczystości danego piksela)

X pozycja względem osi poziomej

Y pozycja względem osi pionowej

W szerokość

H wysokość

Kolejnym niejasnym punktem może być sposób obliczania funkcji przystosowania (J). W naszym przypadku funkcja ta przyjmuje wartości od 0 do 1 i będzie obliczana na podstawie porównywania piksel po pikselu obrazu stworzonego przez osobnika z obrazem oryginalnym.

1.3 Wkład autorów

- Obsługa obrazów - Rafał Kwiatkowski
- Algorytm Ewolucyjny - Franciszek Sioma i Rafał Kwiatkowski
- Testy i eksperymenty - Rafał Kwiatkowski i Franciszek Sioma
- Dokumentacja - Franciszek Sioma

1.4 Decyzje projektowe

Algorytm zakłada dwa rodzaje krzyżowania osobników:

- uśrednianie
- interpolacja

W naszym programie postanowiliśmy umieścić oba te sposoby, by móc przebadać ich wpływ na końcowy wynik algorytmu. Taki sam zabieg dotyczy metod wybierania kolejnej generacji populacji. Zaimplementowane metody to:

- metoda μ najlepszych
- metoda koła ruletki
- metoda selekcji rankingowej

Algorytm zwraca najlepszego osobnika wraz z jego wartością funkcji dopasowania. Program wyświetla obraz reprezentowany przez tego osobnika.

W celu przyśpieszenia algorytmu każdy z osobników przechowuje swoją wartość funkcji dopasowania. Dzięki temu jest ona liczona tylko raz dla każdego osobnika.

1.5 Wykorzystane narzędzia i biblioteki

Do napisania aplikacji użyliśmy języka Python w wersji: 3.8, dokumentacja została stworzona przy użyciu języka Latex, a IDE z którego korzystaliśmy to Visual Studio Code. Użyliśmy również systemu kontroli wersji Git.

Spis użytych bibliotek znajduje się w pliku *requirements.txt*. Do najważniejszych z nich należy biblioteka Pillow wykorzystana do obsługi obrazów.

2 Uruchamianie aplikacji i odtworzenie wyników testów

2.1 Uruchomienie aplikacji

W celu uruchomienia aplikacji należy wykonać komendę:

```
python app.py "simple_img.jpg" 30 20 45 1000 0.95 "inter" "best"
```

Wówczas aplikacja zostanie wykonana dla wielkości populacji równej 30, w każdym z osobników będzie 20 prostokątów, podpopulacja wybierana z populacji do krzyżowania będzie składać się z 45 osobników, maksymalna liczba iteracji wyniesie 1000, a warunkiem stopu będzie dopasowanie na poziomie 95%. Algorytm użyje interpolacji przy krzyżowaniu, oraz użyje metody μ najlepszych do wybrania kolejnej generacji populacji. Pozostałe dostępne wartości to *mean* - uśrednianie - dla krzyżowania oraz *roulette* i *ranking* (odpowiednio metoda koła ruletki i selekcji rankingowej) dla wybierania.

2.2 Odtworzenie testów

W celu odtworzenia przeprowadzonych testów należy wykonać komendę:

```
python examination.py
```

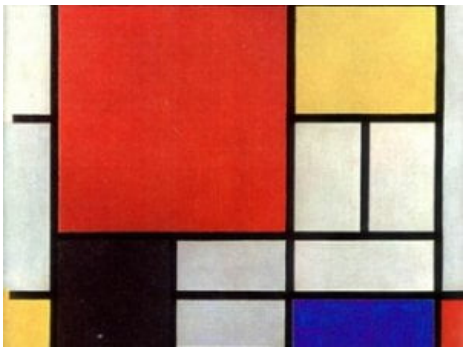
3 Eksperymenty

Wykonaliśmy szereg testów, sprawdzających jak działa aplikacja w zależności od następujących parametrów:

- liczebność populacji
- liczebność populacji λ
- liczba prostokątów w osobniku
- strategia krzyżowania
- strategia wybierania kolejnej generacji populacji

3.1 Eksperymenty i wyniki

W celu uzyskania możliwie jak najbardziej prawdziwych wyników dla każdego przypadku testowego, każdy eksperyment z danymi parametrami był przeprowadzony 5 razy. Wszystkie testy zostały wykonane dla obrazu na Rysunku 1 oraz z warunkami stopu 1000 iteracji lub dokładność 99%.



Rysunek 1: Obrazek użyty w eksperymentach

Gdy nie było to przedmiotem eksperymentu to metodą krzyżowania było uśrednianie, a metodą wybierania - wybór μ najlepszych. Wykresy są tworzonę na podstawie punktów w których zmieniała się wartość funkcji dopasowania dla najlepszego osobnika, dlatego niektóre przebiegi kończą się wcześniej. Oznacza to, że wynik się nie zmienił już do końca eksperymentu.

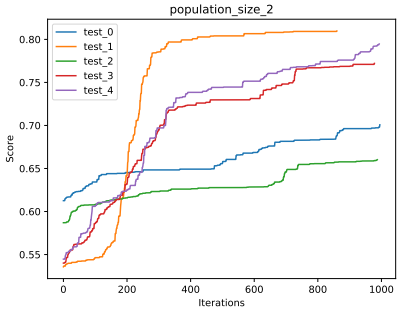
Wpływ liczebności populacji

Test wpływu liczebności populacji został przeprowadzony dla populacji o wielkościach: 2, 10, 15, 30, 40, 50, 100, liczbie prostokątów w osobniku równej 20, wielkości populacji μ będącej 1,5 raza większej od wielkości testowanej populacji.

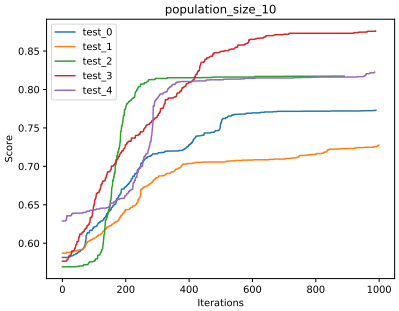
Liczebność populacji	Najlepszy osobnik	Średnio najlepszy osobnik
2	81%	75%
10	88%	80%
15	87%	80%
30	86%	78%
40	89%	84%
50	90%	89%
100	90%	86%

Tablica 1: Wyniki testów liczebności populacji

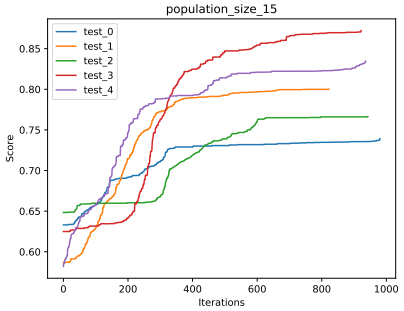
Z wyników tego testu można zauważyć, że zwiększanie wielkości populacji daje lepsze wyniki. Dzieje się tak z tego powodu, że większa populacja oznacza większą różnorodność w populacji i większe prawdopodobieństwo pojawienie się osobnika optymalnego.



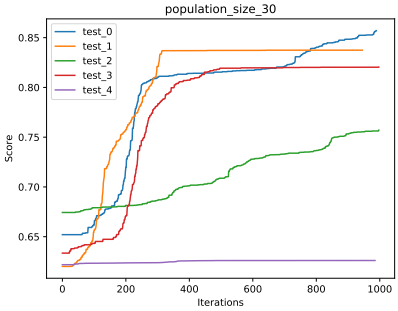
(a) $\mu = 2$



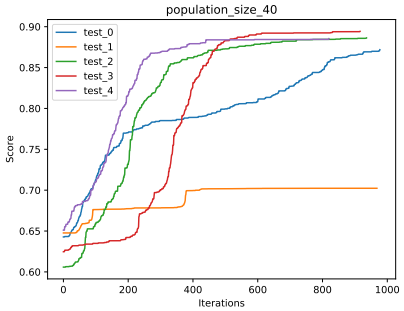
(b) $\mu = 10$



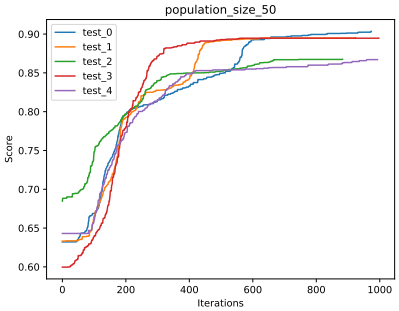
(c) $\mu = 15$



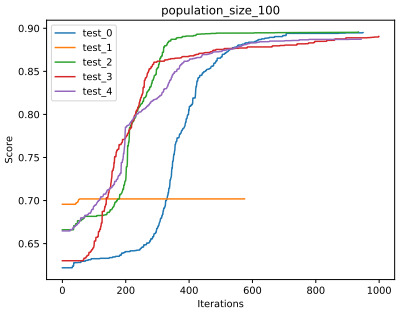
(d) $\mu = 30$



(e) $\mu = 40$



(f) $\mu = 50$

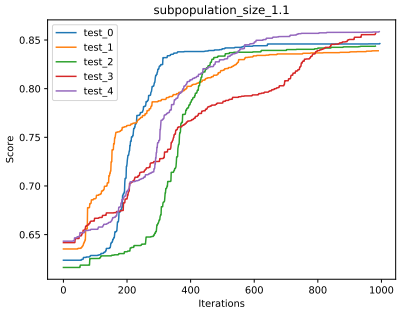


(g) $\mu = 100$

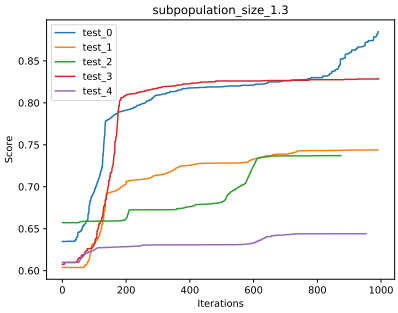
Rysunek 2: Wielkość populacji

Wpływ liczebności populacji λ

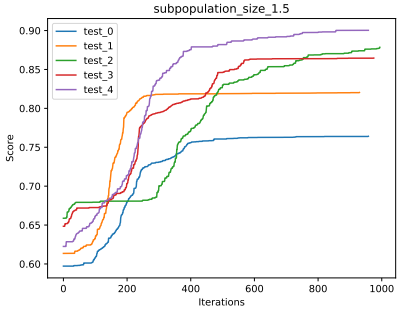
Test wpływu liczebności populacji λ został przeprowadzony dla stałej wielkości populacji μ równej 40 na podstawie której kolejne przypadki testowe był obliczane za pomocą wartości procentowych: 110%(44), 130%(52), 150%(60), 200%(80), 250%(100), 300%(120). Liczba prostokątów w osobniku była równa 20.



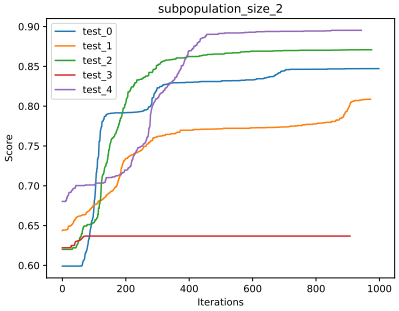
(a) $\lambda = 110\%$



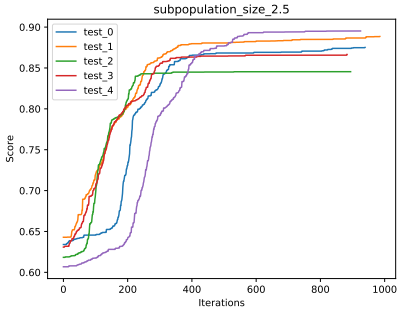
(b) $\lambda = 130\%$



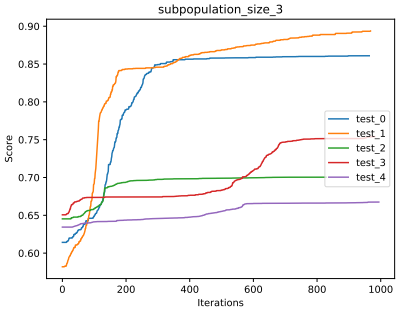
(c) $\lambda = 150\%$



(d) $\lambda = 200\%$



(e) $\lambda = 250\%$



(f) $\lambda = 300\%$

Rysunek 3: Wielkość populacji λ

Liczebność populacji λ	Najlepszy osobnik	Średnio najlepszy osobnik
110%	86%	85%
130%	88%	77%
150%	90%	84%
200%	90%	81%
250%	90%	88%
300%	89%	77%

Tablica 2: Wyniki testów liczebności populacji λ

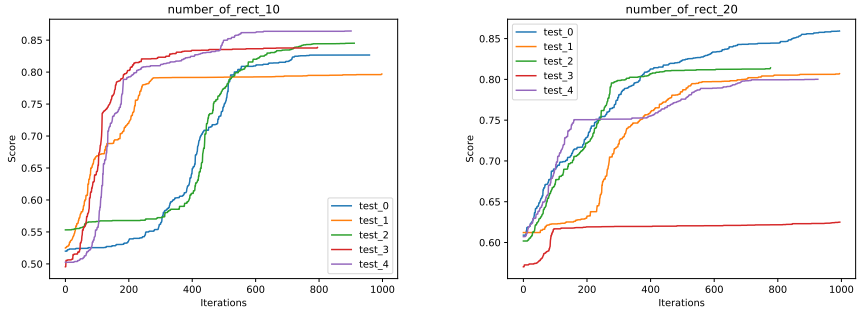
W przypadku wyników testu wielkości populacji λ można powiedzieć, że zwiększenie wartości λ do 250% wartości μ daje najlepsze efekty. Dalsze zwiększanie wartości λ powoduje spadek efektów końcowych.

Wpływ liczby prostokątów w osobniku

Następny test został przeprowadzony dla następujących liczb prostokątów w osobniku: 10, 20, 50, 100, 200, 300. Wielkość populacji wynosiła 10, podpopulacji 15. Pozostałe parametry zostały niezmienione względem poprzedniego testu.

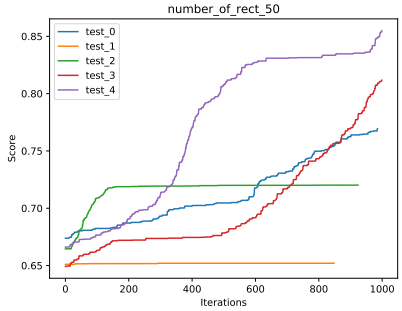
Liczba prostokątów	Najlepszy osobnik	Średnio najlepszy osobnik
10	86%	84%
20	86%	78%
50	85%	76%
100	74%	72%
200	75%	74%
300	75%	74%

Tablica 3: Wyniki testów liczby prostokątów w osobniku

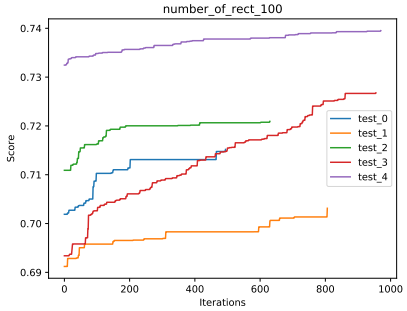


(a) 10

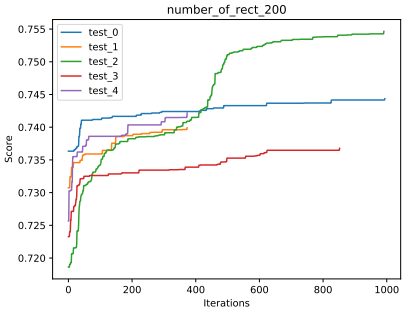
(b) 20



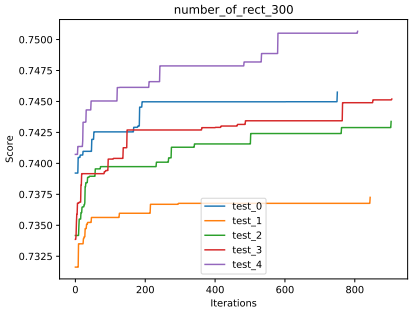
(c) 50



(d) 100



(e) 200



(f) 300

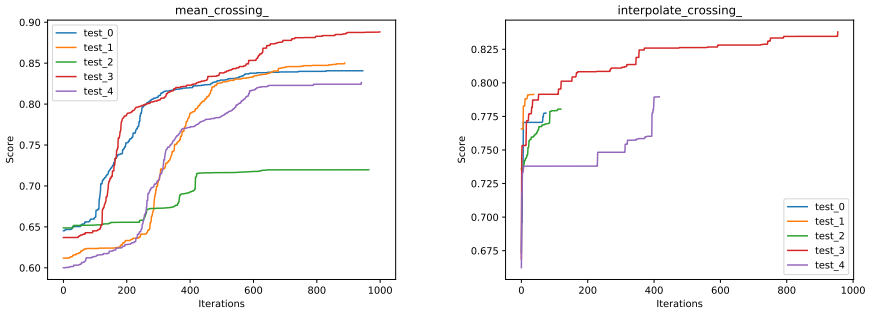
Rysunek 4: Liczba prostokątów w osobniku

W przypadku liczby prostokątów w osobniku można zauważyć, że w przeprowadzonym teście zwiększanie tej liczby zmniejsza dokładność odtworzenia zadanego obrazu. Wynika to z faktu, że zadany obraz składa się z 14 prostokątów o prawie jednolitych kolorach, więc algorytmowi łatwiej zbliżyć się do żądanej dokładności gdy używa mniejszej liczby prostokątów. Wynika to jedynie ze szegółowości obrazu wejściowego. By uzyskać najlepsze wyniki liczba prostokątów w osobniku powinna być zbliżona do liczby względnie jednolitych

pól na obrazie.

Wpływ użytej strategii krzyżowania

Dla populacji λ - 20, populacji μ - 30 i liczbie 20 prostokątów w osobniku testy zostały przeprowadzone dla uśredniania i interpolacji.



(a) Uśrednianie (b) Interpolacja

Rysunek 5: Krzyżowanie

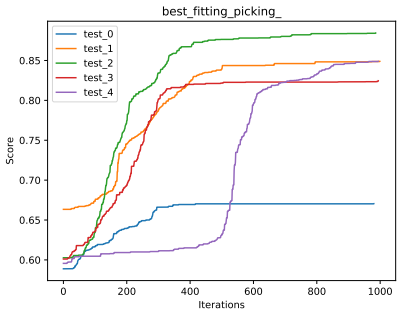
Metoda	Najlepszy osobnik	Średnio najlepszy osobnik
Uśrednianie	89%	83%
Interpolacja	83%	79%

Tablica 4: Wyniki testów krzyżowania

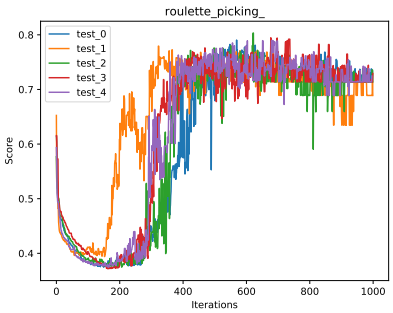
Dość dobrze widać, że w przypadku naszego problemu lepiej się sprawdza uśrednianie. Jeśli chodzi o interpolacje to warto zauważyć że wartości rosną bardzo szybko na początku, a potem zwykle zostają stałe lub niewiele się zmieniają. Może to być spowodowane, że algorytm dość szybko znajduje w naszym przypdaku osobnika z jednym lub kilkoma dużymi kwadrawtami, które potem ciężko rozmnożyć na lepsze osbniki. W przypadku uśredniania osbniki zmieniają się znacznie wolniej i taka sytuacja zachodzi rzadziej.

Wpływ użytej strategii wybierania kolejnej populacji

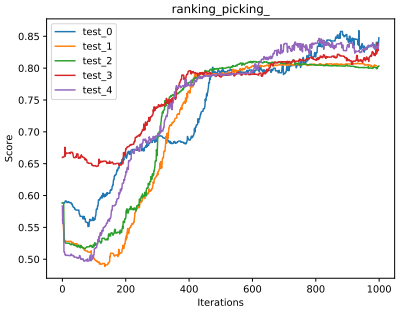
Dla populacji λ - 20, populacji μ - 30 i liczbie 20 prostokątów w osobniku testy zostały przeprowadzone dla wyboru najlepszych, ruletki i metody rankingu.



(a) Wybór najlepszych



(b) Ruletka



(c) Ranking

Rysunek 6: Wybieranie

Metoda	Najlepszy osobnik	Średnio najlepszy osobnik
Wybór najlepszych	88%	82%
Ruletka	73%	72%
Ranking	85%	82%

Tablica 5: Wyniki testów wybierania

Ciężko stwierdzić, która metoda wybierania jest najlepsza, ale skłaniamy się do preferencji wyboru najlepszych lub przez ranking. Metoda ruletki wydaje się produkować najsłabsze rezultaty. Metoda wyboru najlepszych w późniejszych fazach traci mocno na efektywności. Metoda rankingu również w późniejszej fazie nieco zwalnia ale nie tak bardzo i wydaje się być dobrym rozwiązaniem dla naszego problemu.