

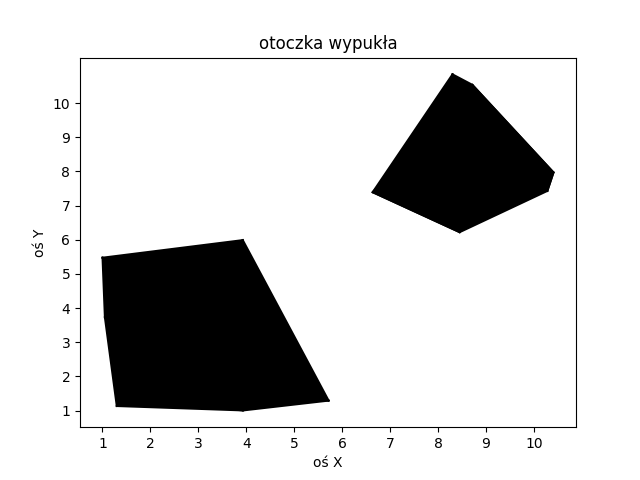
|  |
| --- |
| QUADTREE  PROJEKT |
|  |
| 24 maja  IO  Autor: Bartłomiej Filipowicz |

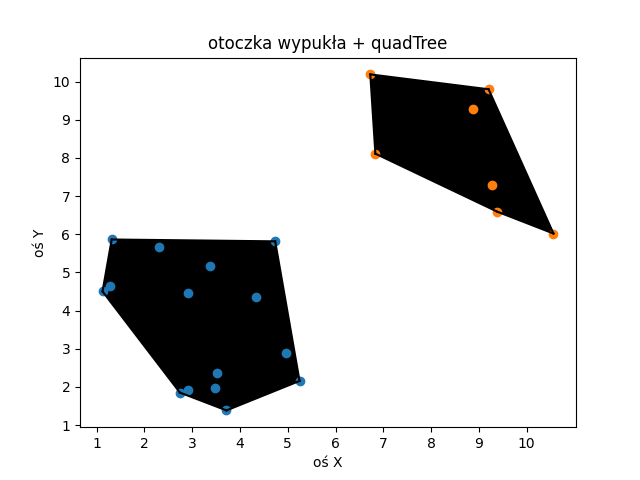
# Algorytm otoczki wypukłej + quadTree

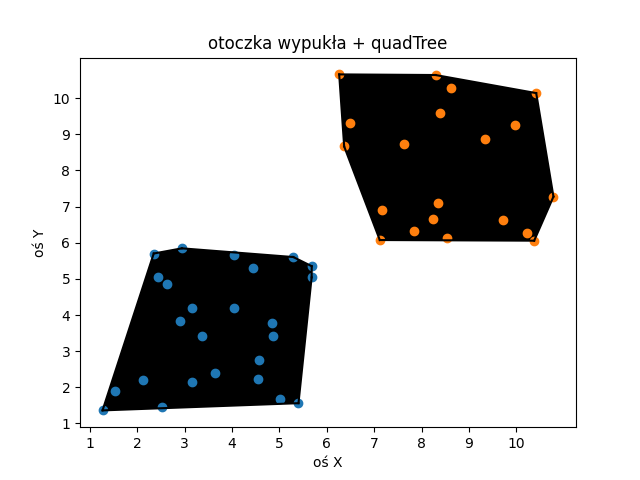
|  |
| --- |
| Wykonany w języku Python Do stworzenia otoczki wypukłej wykorzystałem bibliotekę scipy oraz numpy, a żeby narysować otrzymany efekt wykorzystałem bibliotekę matplotlib. Tak się prezentuje kod tworzenia samej otoczki:    Tak natomiast prezentuje się fragment odpowiedzialny za narysowanie otoczki:    Wypełnienie otoczki załatwiłem jedną linijką:    Omówię teraz sam algorytm quadTree.  Algorytm ten zrealizowałem przy użyciu stworzonej przeze mnie klasy QuadTree oraz biblioteki PIL, dzięki której uzyskałem strukturę danych z dostępem do każdego pojedynczego piksela z obrazka wejściowego.  Tak wygląda konstruktor tej klasy(w obiekcie pix są przechowywane wszystkie piksele): |
| *„To był świetny projekt do zrealizowania. Doskonale się przy nim bawiłem!”* |
| W destruktorze jest zapisywany wynik działania algorytmu jako plik graficzny:    Nie wkleiłem kodu metody divide(), ponieważ nie zmieściłby się on na stronie(odsyłam do kodu źródłowego). Jest to najważniejsza funkcja w całym programie. Działa w ten sposób, że na początku sprawdza czy mam do czynienia z jednym pikselem. Jeśli tak, funkcja kończy działanie. Jeśli nie, iteruję po wszystkich pikselach danego podkwadratu, dopóki nie trafię na co najmniej jeden piksel biały i czarny. Robię tak, ponieważ quadTree ma działać tylko na brzegach figur, a nie na całych figurach. Jeśli powyższy warunek jest spełniony, to wywołuję funkcję draw(), a następnie rekurencyjnie wywołuję cztery metody divide() dla odpowiednich kwadratów(pamiętając, że linie siatki zajmują jeden piksel). Po wywołaniu tych czterech metod kończę działanie funkcji instrukcją return. Jest to bardzo ważne, ponieważ na początku zapomniałem dodać tę instrukcję i program bardzo długo się wykonywał(jest wtedy wywoływanych bardzo wiele funkcji divide()).  Tak wygląda metoda draw():    Sam program główny jest bardzo schludny i przejrzysty: |

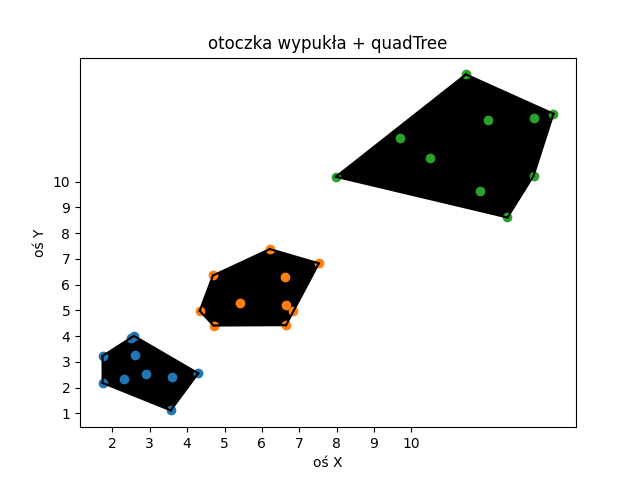
A oto wyniki działania programu.

Najpierw otoczka wypukła:

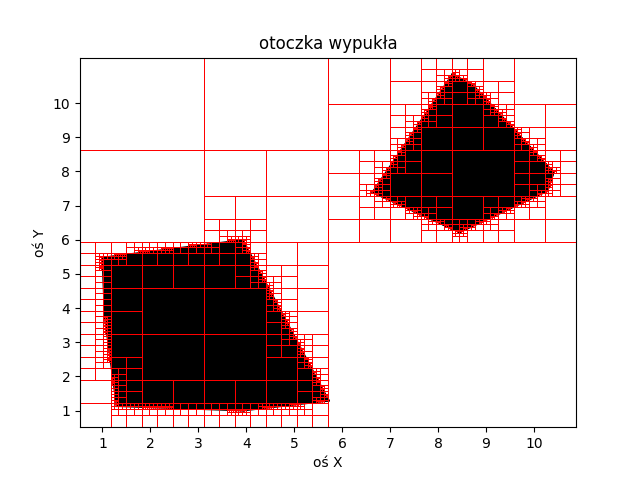


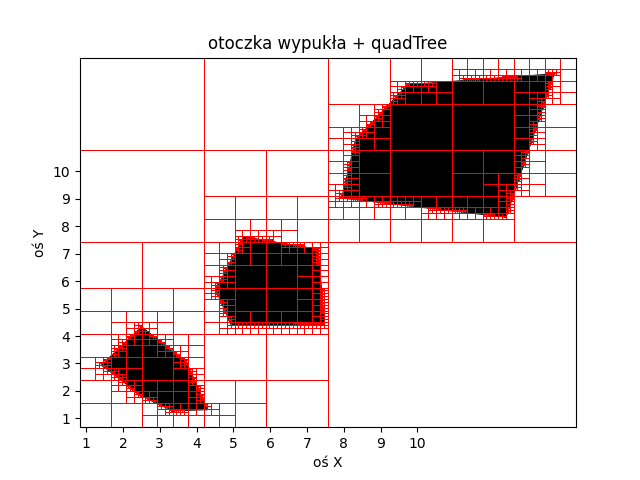






Teraz quadTree:





Wnioski

Projekt quadTree jest świetnym projektem, który można śmiało dodać do swojego githuba. Ten projekt wciągnął mnie całkowicie – cięzko było oderwać się od komputera, a samo kodzenie było czystą przyjemnością. Jedynie trzeba było się skupić podczas podawania argumentów dla rekurencyjnych wywołań metody divide().