Algorytmy Geometryczne

Laboratorium 3

Triangulacja wielokątów monotonicznych

Imię i Nazwisko: Wermiński Gabriel
Grupa: 6

Data wykonania: 18.11.2025

1 Środowisko obliczeniowe

Eksperymenty zostały przeprowadzone w następującej konfiguracji:

Procesor:	CPU 12th Gen Intel(R) Core(TM) i5-1235U, 1.30 GHz
Pamięć RAM:	16.0 GB, 3200 MT/s
System operacyjny:	Microsoft Windows 11 Home
Środowisko programistyczne:	Jupyter Notebook, Pycharm
Język programowania:	Python 3.13.5
Biblioteki:	numpy, pandas, matplotlib, os

2 Cel ćwiczenia

Zapoznanie się z zagadnieniami dotyczącymi monotoniczności wielokątów oraz implementacja algorytmów sprawdzania y-monotoniczności wielokąta, klasyfikacji wierzchołków w dowolnym wielokącie oraz triangulacji wielokąta y-monotonicznego. Celem było także wykonanie poszczególnych wizualizacji oraz analiza danych.

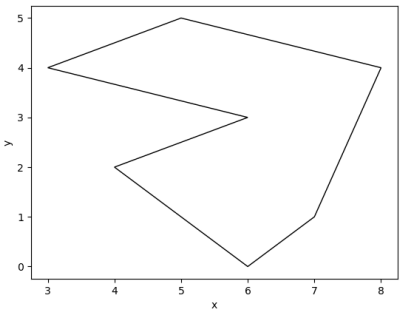
3 Wstęp teoretyczny

3.1 Monotoniczność wielokąta

Wielokąt prosty jest ściśle monotoniczny względem prostej l (wyznaczającej kierunek monotoniczności), kiedy jego brzeg można przedstawić w postaci dwóch spójnych łańcuchów, z których każdy przecina się z dowolną prostą l' prostopadłą do l w nie więcej niż jednym punkcie.

Przecięcie wielokąta z l' jest spójne, co oznacza że jest ono odcinkiem, punktem lub jest puste.

Wielokątem y-monotonicznym (Rysunek 1) nazywamy wielokąt, który jest monotoniczny względem osi y, czyli w przypadku przejścia z najwyższego wierzchołka do najniższego, wzdłuż prawego lub lewego łańcucha, zawsze poruszamy się w dół lub poziomo.

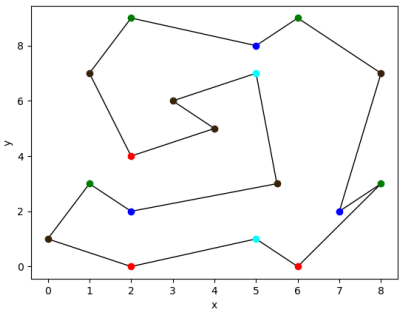


Rysunek 1: wielokąt y-monotoniczny

3.2 Klasyfikacja wierzchołków wielokąta

Wierzchołki zostały podzielone na 5 rodzajów:

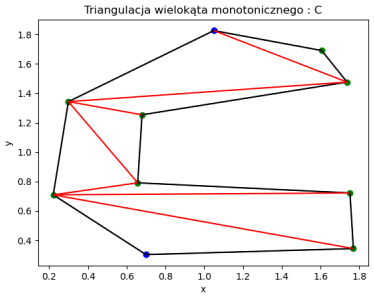
- **początkowy**, gdy obaj jego sąsiedzi leżą poniżej i kąt wewnętrzny $< \pi$,
- **końcowy**, gdy obaj jego sąsiedzi leżą powyżej i kąt wewnętrzny $< \pi$,
- **łączący**, gdy obaj jego sąsiedzi leżą powyżej i kąt wewnętrzny $> \pi$,
- **dzielący**, gdy obaj jego sąsiedzi leżą poniżej i kąt wewnętrzny $> \pi$,
- **prawidłowy**, w pozostałych przypadkach (ma jednego sąsiada powyżej, drugiego – poniżej).



Rysunek 2: wielokąt niemonotoniczny z wierzchołkami pokolorowanymi zgodnie z ich klasyfikacją

3.3 Triangulacja wielokąta

Triangulacja wielokąta to podział wielokąta na rozłączne trójkąty poprzez dodanie nieprzecinających się przekątnych łączących jego wierzchołki. W procesie triangulacji wszystkie dodawane przekątne muszą leżeć wewnątrz wielokąta, a ich końce muszą być jego wierzchołkami. Dla wielokąta o n wierzchołkach triangulacja zawsze składa się z $n-2$ trójkątów połączonych $n-3$ przekątnymi. Przykład takiej triangulacji można zauważyć na Rysunku 3.



Rysunek 3: Przykładowa triangulacja wielokąta y-monotonicznego

4 Realizacja ćwiczenia

4.1 Szczegóły implementacyjne

4.1.1 Opis Algorytmu Weryfikacji Monotoniczności Względem Osi y

Metoda ma na celu sprawdzenie, czy wielokąt, którego wierzchołki zadane są w kolejności przeciwnej do ruchu wskazówek zegara (CCW), jest y-monotoniczny. Wielokąt spełnia ten warunek, jeśli jego obwód dzieli się na dwa łańcuchy monotoniczne względem osi y: Lewy(nierosnący) i Prawy(niemalejący).

1. **Identyfikacja Ekstremów** - Wyznaczane są indeksy wierzchołków : $\boldsymbol{P_{\max}}$(punkt najwyższy) i $\boldsymbol{P_{\min}}$(punkt najniższy). Te punkty stanowią granice podziału obwodu na dwa łańcuchy.
2. **Weryfikacja Łańcuchów Monotonicznych.**
   - Łańcuch Lewy (CCW od $\boldsymbol{P_{\max}}$ do $\boldsymbol{P_{\min}}$) - Wartości y muszą być nierosnące $\left(y_{\text{poprzedni}} \geq y_{\text{obecny}}\right)$. Jeśli y wzrośnie, wielokąt nie jest monotoniczny.
   - Łańcuch Prawy (CCW od $\boldsymbol{P_{\min}}$ do $\boldsymbol{P_{\max}}$) - Wartości y muszą być niemalejące $\left(y_{\text{poprzedni}} \geq y_{\text{obecny}}\right)$. Jeśli y zmaleje, wielokąt nie jest monotoniczny.
3. **Wynik** - Jeśli jakikolwiek warunek monotoniczności zostanie naruszony podczas przechodzenia któregokolwiek łańcucha, funkcja natychmiast zwraca False. W przeciwnym razie zwracana jest wartość True.

4.1.2 Funkcja obliczająca wartość wyznacznika

W rozważanych algorytmach kluczową rolę odgrywa funkcja do obliczania wyznacznika macierzy 3×3. Ta funkcja, operująca na współrzędnych trzech punktów płaszczyzny a, b i c, jest zdefiniowana jako:

$$\det(a,b,c) = \begin{vmatrix} a_x & a_y & 1 \\ b_x & b_y & 1 \\ c_x & c_y & 1 \end{vmatrix} = (b_x - a_x)(c_y - b_y) - (b_y - a_y)(c_x - b_x)$$

W przyjętej implementacji, próg precyzji ustalono na: $\boldsymbol{\varepsilon = 10^{-24}}$.

4.1.3 Algorytm klasyfikacji wierzchołków wielokąta

Algorytm przydziela każdemu wierzchołkowi P_i jedną z pięciu kategorii (0-4), zgodnie z warunkami szczegółowo opisanymi w Sekcji 3.2 . Proces ten bazuje na cyklicznej analizie wierzchołków wielokąta zdefiniowanego w porządku CCW.

1. Dla każdego wierzchołka P_i następuje dwuetapowa weryfikacja:
   - Analiza Pionowa (y): Porównanie współrzędnej y wierzchołka P_i z poprzednikiem P_{i-1} i następnikiem P_{i+1}. Określa to, czy wierzchołek leży na łuku wznoszącym, czy opadającym, co zawęża potencjalne typy (Początkowy/Dzielący vs. Końcowy/Łączący).
   - Analiza Kąta (Wyznacznik): Obliczenie wartości $\det(P_{i-1},P_i,P_{i+1})$ ustala charakter kąta w P_i jako wypukły, wklęsły lub współliniowy.
2. Klasyfikacja i Przypisanie Kodu :
   - Kombinacja wyników tych analiz determinuje finalną kategorię, zgodnie z którą kod (0, 1, 2, 3 lub 4) zostaje przypisany do wierzchołka. Wierzchołki, które są współliniowe lub leżą wewnątrz pojedynczego monotonicznego łuku, są klasyfikowane jako Prawidłowe.

4.1.4 Implementacja algorytmu triangulacji wielokąta monotonicznego

1. **Identyfikacja punktów ekstremalnych** - Algorytm rozpoczyna od zlokalizowania wierzchołków o maksymalnej i minimalnej współrzędnej y, które stanowią naturalne granice podziału wielokąta.
2. **Klasyfikacja wierzchołków na ścieżki** - Punkty są przypisywane do lewej lub prawej ścieżki poprzez dwukierunkowe przejście: od najwyższego do najniższego punktu (tworząc lewą ścieżkę) oraz w kierunku przeciwnym (prawą ścieżkę). Punkty ekstremalne otrzymują osobną etykietę.
3. **Scalanie i sortowanie** - Obie ścieżki są łączone w jedną sekwencję uporządkowaną malejąco względem współrzędnej y przy użyciu procedury merge, z punktami ekstremalnymi na pozycjach brzegowych.
4. **Inicjalizacja stosu** - Dwa pierwsze elementy z posortowanej sekwencji są umieszczane na stosie jako kandydaci do tworzenia przekątnych.
5. **Obsługa różnych ścieżek** - Gdy aktualny wierzchołek należy do innej ścieżki niż szczyt stosu, tworzone są przekątne z wszystkimi elementami stosu (pomijając pierwszy). Specjalny warunek dla najniższego punktu zapobiega duplikacji ostatniej krawędzi wielokąta. Stos jest resetowany do ostatniego elementu plus aktualny punkt.
6. **Obsługa tej samej ścieżki** - Gdy punkty należą do jednej ścieżki, algorytm iteracyjnie sprawdza możliwość utworzenia przekątnych ze zdejmowanymi ze stosu wierzchołkami, zachowując minimum dwa elementy na stosie.
7. **Weryfikacja orientacji** - Poprawność triangulacji sprawdzana jest przez wyznacznik macierzy 3×3 - dla lewej ścieżki wymagana jest orientacja lewoskrętna ($\det > \varepsilon$), dla prawej - prawoskrętna ($\det < -\varepsilon$).
8. **Tolerancja numeryczna** - Użycie epsilon w porównaniach wyznaczników eliminuje problemy z błędami zaokrągleń arytmetyki zmiennoprzecinkowej.

9. **Normalizacja wyników** - Każda para indeksów przekątnej jest sortowana, co ułatwia późniejszą eliminację potencjalnych duplikatów i standaryzuje reprezentację.
10. **Gwarancja złożoności liniowej** - Każdy wierzchołek jest dodawany na stos dokładnie raz i może być usunięty maksymalnie raz, co zapewnia złożoność $O(n)$.

4.2 Zestawy danych testowych

Zbiór wielokątów geometrycznych został stworzony interaktywnie za pomocą narzędzia wykorzystującego bibliotekę **matplotlib**. Ta graficzna aplikacja umożliwiła precyzyjne definiowanie wierzchołków na płaszczyźnie, co stanowiło bazę dla dalszych obliczeń algorytmicznych. Przykłady utworzonych wielokątów oraz rezultaty ich triangulacji zostaną zaprezentowane w dalszej części pracy.

5 Analiza wyników

5.1 Rezultaty Weryfikacji Monotoniczności

Przeprowadzona analiza geometryczna, bazująca na zaimplementowanym algorytmie sprawdzania y-monotoniczności, potwierdziła przewidywane charakterystyki figur.

Konkretnie, testy wykazały, że:

- Wielokąty A, B, C oraz E spełniają warunek y-monotoniczności.
- Wielokąt D nie spełnia warunku y-monotoniczności.

Uzyskane wyniki są w pełni zgodne z wstępną hipotezą dotyczącą wybranych figur testowych.

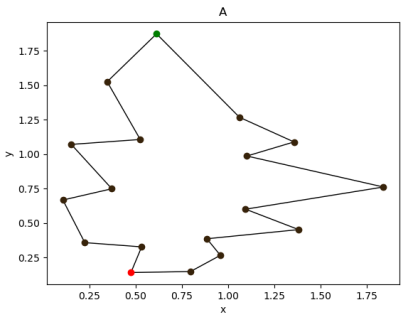
5.2 Algorytm klasyfikacji wierzchołków wielokątów

Wyniki kategoryzacji, zgodnie z konwencją barwną z Sekcji Sekcji 3.2, wyraźnie ilustrują związek między typem wierzchołków a właściwością y-monotoniczności:

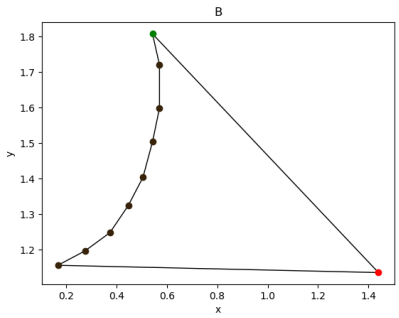
Wielokąty y-Monotoniczne: Figury te (A, B, C, E, F) konsekwentnie wykazują obecność tylko jednego wierzchołka początkowego (start) i jednego końcowego. Wszystkie pozostałe wierzchołki są poprawnie identyfikowane jako prawidłowe. Stanowi to bezpośrednie potwierdzenie twierdzenia, że wielokąty y-monotoniczne są wolne od wierzchołków łączących i dzielących.

Wielokąt Nie-y-Monotoniczny: Z kolei w figurach nie-y-monotonicznych, figura(D) obserwuje się występowanie wierzchołków łączących lub dzielących, które są geometrycznym powodem utraty monotoniczności.

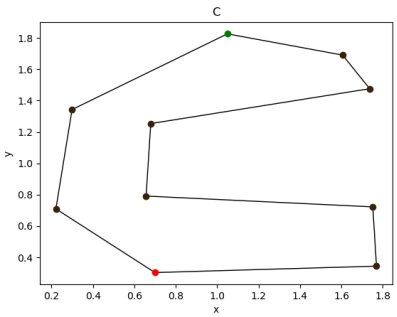
Na poniższych rysunkach zamieszczone zostały wizualizacje wielokątów testowych zwrócone przez algorytm, wraz z wierzchołkami pokolorowanymi zgodnie z przyjętą w Sekcji 3.2 konwencją.



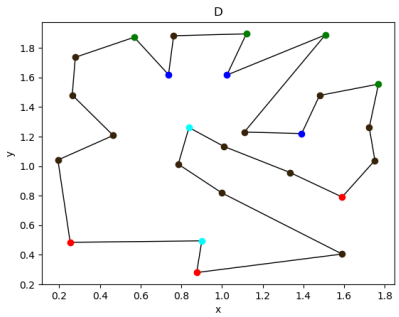
Rysunek 4: Klasyfikacji figury A



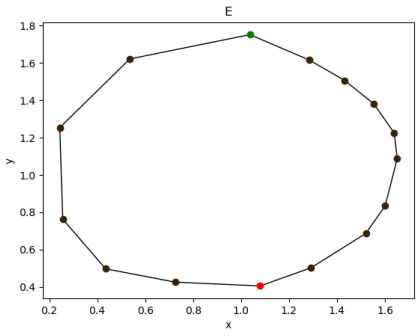
Rysunek 5: Klasyfikacji figury B



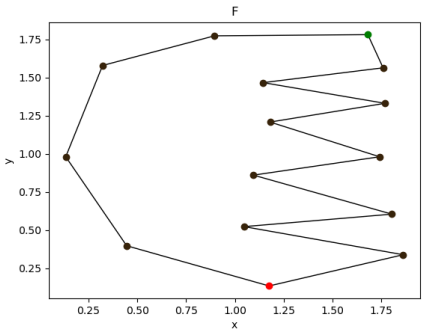
Rysunek 6: Klasyfikacji figury C



Rysunek 7: Klasyfikacji figury D



Rysunek 8: Klasyfikacji figury E



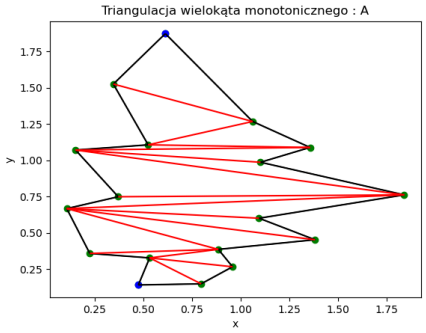
Rysunek 9: Klasyfikacji figury F

Dane wizualne, zgodnie z oczekiwaniami, demonstrują, że zaimplementowany algorytm z sukcesem identyfikuje i kategoryzuje wierzchołki w kluczowych punktach geometrycznych, co stanowi niezbędny etap do dalszych procesów, takich jak triangulacja.

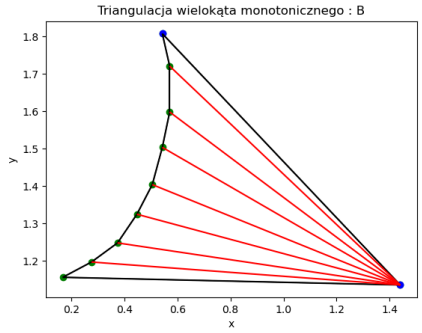
5.3 Algorytm triangulacji wielokątów y-monotonicznych

W ramach eksperymentów uruchomiono zaimplementowany algorytm triangulacji dla przygotowanych wcześniej zestawów danych testowych. Przeprowadzono również testy na wielokątach niespełniających warunku y-monotoniczności, aby zademonstrować, że algorytm nie działa poprawnie bez zachowania tego wymagania. Zastosowanie go w takich sytuacjach wymagałoby wcześniejszej dekompozycji wielokątów na części spełniające warunek y-monotoniczności.

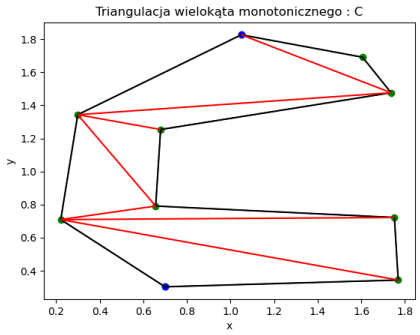
W przedstawionych wizualizacjach zastosowano następującą konwencję kolorystyczną: krawędzie stanowiące brzeg wielokąta oznaczono kolorem czarnym, niebieskim kolorem wyróżniono wierzchołki ekstremalne ($P_{\max}$, $P_{\min}$), zielony kolor przypisano pozostałym wierzchołkom, natomiast kolorem czerwonym oznaczono przekątne wprowadzone w procesie triangulacji.



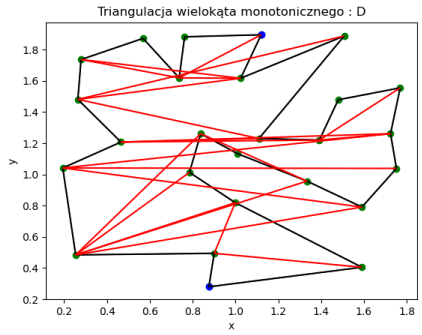
Rysunek 10: Triangulacja figury A



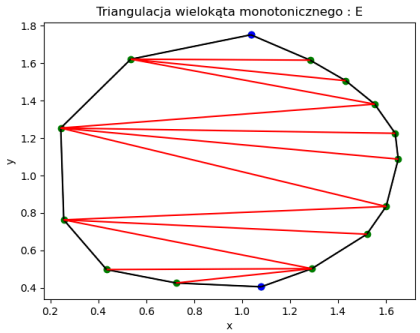
Rysunek 11: Triangulacja figury B



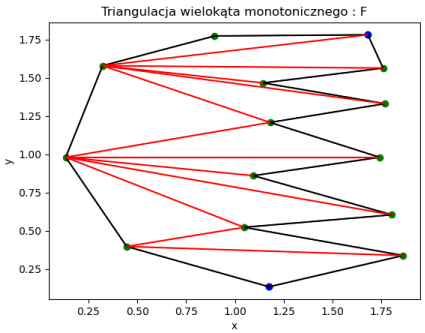
Rysunek 12: Triangulacja figury C



Rysunek 13: Triangulacja figury D



Rysunek 14: Triangulacja figury E



Rysunek 15: Triangulacja figury F

Zgodnie z przewidywaniami, zastosowanie przedstawionego algorytmu do wielokątów niespełniających warunku y-monotoniczności nie przynosi oczekiwanych rezultatów. Na wizualizacjach wyraźnie widać, że dla przypadku D wprowadzone przekątne nie tylko generują trójkąty leżące poza obszarem figury, ale także przecinają jej krawędzie brzegowe. Przyczyna tkwi w niewłaściwym działaniu mechanizmu rozdzielającego wierzchołki na oba łańcuchy w sytuacji braku monotoniczności, co następnie powoduje błędy w procedurze scalającej. Sam algorytm triangulacji również nie jest przystosowany do obsługi takich struktur – konieczna byłaby uprzednia dekompozycja na komponenty y-monotoniczne.

Wielokąt A stanowi przykład figury wypukłej, w której punkty rozłożone są „symetrycznie” wzdłuż obu łańcuchów. Podczas wykonywania triangulacji, każdy kolejno analizowany wierzchołek przynależy do łańcucha przeciwnego względem dwóch elementów przechowywanych na szczycie stosu. Taka konfiguracja eliminuje ryzyko tworzenia trójkątów zewnętrznych. Rezultat triangulacji zaprezentowano na Rysunku 10.

Wielokąt B zawiera pojedynczy kąt wklęsły umieszczony na lewym łańcuchu. Test ten weryfikuje, czy mechanizm triangulacji prawidłowo identyfikuje sytuacje, w których krawędź brzegowa figury nie powinna być błędnie dodawana do zbioru przekątnych.

Wielokąt C został zaprojektowany w celu weryfikacji zachowania algorytmu w obecności wielu kątów wklęsłych skoncentrowanych na jednym łańcuchu, przy jednoczesnej wypukłości pozostałych fragmentów. Przypadek ten bada efektywność zarządzania stosem oraz poprawność wykrywania trójkątów wykraczających poza granice wielokąta lub odcinków współliniowych z jego bokami.

Wielokąt D – jak wspomniano wcześniej – reprezentuje przypadek niemonotoniczny, dla którego algorytm nie funkcjonuje poprawnie zgodnie z przewidywaniami teoretycznymi.

Wielokąt E ma kształt elipsy z wieloma wierzchołkami rozłożonymi równomiernie wzdłuż obwodu. Przypadek ten testuje sytuację, w której punkty należące do obu łańcuchów występują naprzemiennie. Triangulacja weryfikuje zdolność algorytmu do prawidłowego łączenia wierzchołków z przeciwległych łańcuchów, gdy struktura wielokąta jest wypukła i symetryczna.

Wielokąt F w którym niemal wszystkie wierzchołki (z wyjątkiem kilku skrajnych) są zgrupowane wzdłuż prawego łańcucha i tworzą charakterystyczny układ przypominający „grzebień". Ten scenariusz sprawdza, jak algorytm radzi sobie z sytuacją, gdy jeden z łańcuchów jest zdominowany przez liczne punkty, podczas gdy drugi jest minimalny. W procesie triangulacji wszystkie wierzchołki z prawego łańcucha są systematycznie łączone przekątnymi, tworząc strukturę trójkątów rozpiętych między gęsto rozmieszczonymi punktami a prostą krawędzią lewego łańcucha.

Figura	A	B	C	D	E	F
Liczba wierzchołków	18	10	10	26	15	15
Liczba dodanych przekątnych	15	7	7	23	12	12

Tabela 1: Zestawienie liczby wierzchołków i przekątnych dla badanych wielokątów

Z przedstawionej tabeli wynika, że ilość wprowadzonych przekątnych jest zgodna z zależnością zaprezentowaną w Sekcji 3.3 i odpowiada wartości $n - 3$, przy czym n oznacza liczbę wierzchołków danego wielokąta. Dla wymienionych figur algorytm wygenerował prawidłowe triangulacje.

6 Wnioski

Przeprowadzone eksperymenty pozwoliły na weryfikację poprawności zaimplementowanych algorytmów oraz potwierdzenie ich zgodności z założeniami teoretycznymi dotyczącymi monotoniczności wielokątów i ich triangulacji.

Algorytm weryfikacji y-monotoniczności wielokąta wykazał pełną zgodność z przewidywaniami teoretycznymi. Poprawnie zidentyfikowano wielokąty spełniające warunek monotoniczności (A, B, C, E, F) oraz wielokąt niemonotoniczny (D). Metoda oparta na analizie dwóch łańcuchów monotoniczności okazała się skuteczna i efektywna obliczeniowo. Kluczowym elementem imple-

mentacji było precyzyjne wyznaczenie punktów ekstremalnych ($P_{\max}$ i $P_{\min}$) oraz weryfikacja nierosnącego charakteru łańcucha lewego i niemalejącego charakteru łańcucha prawego.

Algorytm klasyfikacji wierzchołków wielokąta prawidłowo przypisał każdy wierzchołek do odpowiedniej kategorii (początkowy, końcowy, łączący, dzielący, prawidłowy). Wizualizacje potwierdziły fundamentalną właściwość wielokątów y-monotonicznych: obecność dokładnie jednego wierzchołka początkowego i jednego końcowego, przy braku wierzchołków łączących i dzielących. W przypadku wielokąta niemonotonicznego (D) algorytm poprawnie wykrył wierzchołki łączące lub dzielące, które stanowią geometryczną przyczynę utraty monotoniczności. Zastosowanie funkcji wyznacznika macierzy 3×3 z progiem precyzji $\varepsilon = 10^{-24}$ okazało się wystarczające do rozróżnienia kątów wypukłych od wklęsłych, unikając problemów numerycznych związanych z arytmetyką zmiennoprzecinkową.

Algorytm triangulacji wielokątów y-monotonicznych zrealizował poprawną dekompozycję wszystkich wielokątów monotoniczych na trójkąty nieprzecinające się. Dla każdego wielokąta o n wierzchołkach wygenerowano dokładnie $n - 3$ przekątnych, co potwierdza teoretyczną zależność. Mechanizm stosowy zapewnił liniową złożoność czasową $O(n)$ algorytmu, a weryfikacja orientacji przez wyznacznik skutecznie eliminowała próby utworzenia przekątnych wykraczających poza obszar wielokąta lub przecinających jego brzeg. Test na wielokącie niemonotoniczny (D) zademonstrował krytyczne znaczenie spełnienia warunku y-monotoniczności – bez niego algorytm generuje nieprawidłowe przekątne przecinające się z krawędziami brzegowymi oraz trójkąty leżące poza obszarem wielokąta.

Różnorodność testowanych przypadków (wielokąty wypukłe, wklęsłe, symetryczne, asymetryczne, typu „grzebień") pozwoliła na kompleksową walidację implementacji. Szczególnie istotne okazały się testy graniczne: wielokąt B z pojedynczym kątem wklęsłym weryfikował poprawność wykrywania krawędzi brzegowych, wielokąt C z wieloma kątami wklęsłymi testował efektywność zarządzania stosem, wielokąt E o kształcie elipsy sprawdzał naprzemienne łączenie wierzchołków z obu łańcuchów, a wielokąt F typu „grzebień" badał zachowanie algorytmu przy silnej asymetrii rozkładu punktów między łańcuchami.