Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie

Wydział Inżynierii Metali i Informatyki Przemysłowej

Sprawozdanie z Laboratorium: Kwadratura Gaussa 2D

Przedmiot: Metody Numeryczne

Kierunek: Inżynieria Obliczeniowa

Autor: Filip Rak

Prowadzący przedmiot: dr hab. inż. Marcin Hojny

Data: 12 kwietnia 2024

Numer lekcji: 6

Grupa laboratoryjna: 4

Wstęp teoretyczny

Kwadratura Gaussa jest metodą numeryczna stosowaną do przybliżonego obliczania wartości całek oznaczonych. Metoda polega na zastąpieniu całki z funkcji, sumą ważonych wartości tej funkcji w wybranych punktach. Może być ona również wykorzystywana w obliczeniu powierzchni. Metodę możemy opisać następującym wzorem:

$$\iint (x,y)dxdy = \int_{-1}^{1}\int_{-1}^{1} f\left(\xi_j, \tilde{\eta}_j\right) J_0 d\tilde{\eta} d\xi = \sum_{i=1}^{n}\sum_{j=1}^{n} \omega_i\, \omega_j f\left(\xi_j, \tilde{\eta}_j\right) J_0 \tag{1}$$

Gdzie:

- ξ_j i $\tilde{\eta}_j$ są węzłami kwadratury.
- ω_i i ω_j są odpowiadającymi tym węzłom wagami.
- J_0 – jakobian zawierający wszystkie pierwsze pochodne funkcji transformacji.

Bardzo istotnym krokiem metody jest przekształcenie figury na postać kwadratu o wymiarach 2 na 2. W tym celu obliczamy wyznacznik macierzy Jakobiego, zawierającej wszystkie pierwsze pochodne funkcji transformacji.

$$|J| = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \tilde{\eta}} - \frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \tilde{\eta}} \tag{2}$$

Przy czym pochodne cząstkowe funkcji transformacji jesteśmy w stanie obliczyć z poniższych wzorów

$$\frac{\partial x}{\partial \xi} = \sum_{i=1}^{4} \frac{\partial N_i}{\partial \xi} x_i \tag{3}$$

$$\frac{\partial x}{\partial \tilde{\eta}} = \sum_{i=1}^{4} \frac{\partial N_i}{\partial \tilde{\eta}} x_i \tag{4}$$

$$\frac{\partial y}{\partial \xi} = \sum_{i=1}^{4} \frac{\partial N_i}{\partial \xi} y_i \tag{5}$$

$$\frac{\partial y}{\partial \tilde{\eta}} = \sum_{i=1}^{4} \frac{\partial N_i}{\partial \tilde{\eta}} y_i \tag{6}$$

Funkcje transformacji są następujące:

$$N_1(\xi,\tilde{\eta}) = 0.25(1-\xi)(1-\tilde{\eta}) \tag{7}$$

$$N_2(\xi,\tilde{\eta}) = 0.25(1+\xi)(1-\tilde{\eta}) \tag{8}$$

$$N_3(\xi,\tilde{\eta}) = 0.25(1+\xi)(1+\tilde{\eta}) \tag{9}$$

$$N_4(\xi,\tilde{\eta}) = 0.25(1-\xi)(1+\tilde{\eta}) \tag{10}$$

Węzły oraz ich wagi są wartościami stablicowanymi. W przypadku dwuwymiarowej Kwadratury Gaussa metoda ta gwarantuje dokładność dla całek z funkcji wielomianowych, których suma stopni wielomianów po zmiennych ξ i η jest mniejsza lub równa $2n-1$ i $2m-1$ odpowiednio, gdzie n i m to liczby wykorzystanych węzłów wzdłuż każdego wymiaru.

W przypadku wykorzystania dwóch węzłów, ich wartości będą następujące.

$$\xi_0 = \tilde{\eta}_0 = \frac{\sqrt{3}}{3} = 0.57735026919 \tag{11}$$

$$\xi_1 = \tilde{\eta}_1 = -\frac{\sqrt{3}}{3} = -0.57735026919 \tag{12}$$

Wagi zaś wynoszą:

$$\omega_0 = \omega_1 = 1 \tag{13}$$

Implementacja

Implementacja metody została przeprowadzona w języku C++ w funkcji `gaussian_quadrature_2D`. Jako parametry, funkcja przyjmuje dwie tablice przechowujące zmienne typu `double`: `x[]` i `y[]`, będące wierzchołkami czworokąta. Typem zwracanym jest `double`.

Pełna definicja funkcji `gaussian_quadrature_2D`

```
double gaussian_quadrature_2D(double x[], double y[])
{
	// precomputed derivatives with respect to xi
	const double d_xi[2][4] = {
		{-0.394338, 0.394338, 0.105662, -0.105662},
		{-0.105662, 0.105662, 0.394338, -0.394338}
	};

	// precomputed derivatives with respect to eta
	const double d_eta[2][4] = {
		{-0.394338, -0.105662, 0.105662, 0.394338},
		{-0.105662, -0.394338, 0.394338, 0.105662}
	};

	double area = 0;
	for (int i = 0; i < 2; i++)
	{
		for (int j = 0; j < 2; j++)
		{
			// partial derivatives for jakobi matrix
			double dx_d_xi = 0;
			double dy_d_xi = 0;
			double dx_d_eta = 0;
			double dy_d_eta = 0;

			for (int k = 0; k < 4; k++)
			{
				dx_d_xi += d_xi[j][k] * x[k];
				dy_d_xi += d_xi[j][k] * y[k];
				dx_d_eta += d_eta[j][k] * x[k];
				dy_d_eta += d_eta[j][k] * y[k];
			}

			// area is the sum of all jakobians multiplied by the weights
			// weights are equal to 1, therefore we skip the multiplication
			area += fabs(dx_d_xi * dy_d_eta - dx_d_eta * dy_d_xi);
		}
	}

	return area;
}
```

Omówienie elementów funkcji `gaussian_quadrature_2D`

Celem zmaksymalizowania optymalizacji i ograniczenia niepotrzebnych obliczeń, pochodne po ξ i $\tilde{\eta}$, zostały wyliczone wcześniej i potraktowane jako wartości stałe. Wykorzystano do tego wzory (7), (8), (9), (10), (11) i (12).

```
// precomputed derivatives with respect to xi
const double d_xi[2][4] = {
     {-0.394338, 0.394338, 0.105662, -0.105662},
     {-0.105662, 0.105662, 0.394338, -0.394338}
};

// precomputed derivatives with respect to eta
const double d_eta[2][4] = {
     {-0.394338, -0.105662, 0.105662, 0.394338},
     {-0.105662, -0.394338, 0.394338, 0.105662}
};
```

Mając już wyliczone wartości pochodnych możemy przejść do obliczenia powierzchni figury. Wynik jest przechowywany w poniższej zmiennej. Nasze obliczenia będziemy dokonywać zgodnie ze wzorem (1).

```
double area = 0;
```

Pętle o iteratorach i oraz j są odpowiednikiem sum zawartych we wzorze (1). Ilość używanych przez nas węzłów to dwa, stąd nasze pętle będą pracować przez dwie iteracje. W celu poprawy czytelności, zawartość pętli została obliczona poniżej

```
for (int i = 0; i < 2; i++)
{
     for (int j = 0; j < 2; j++)
     {

     }
}
```

Wewnątrz pętli *j* najpierw dokonujemy obliczeń pochodnych cząstkowych funkcji transformacji. Wykorzystujemy wzory (3), (4), (5) i (6). W związku z tym, że równania są bardzo zbliżone, jesteśmy w stanie policzyć je w pojedynczej pętli.

```
double dx_d_xi = 0;
double dy_d_xi = 0;
double dx_d_eta = 0;
double dy_d_eta = 0;

for (int k = 0; k < 4; k++)
{
     dx_d_xi += d_xi[j][k] * x[k];
     dy_d_xi += d_xi[j][k] * y[k];
     dx_d_eta += d_eta[j][k] * x[k];
     dy_d_eta += d_eta[j][k] * y[k];
}
```

Nadal będąc wewnątrz pętli *j*, wyliczamy wyznacznik macierzy Jakobiego, zwany również Jakobianem. Ponieważ wagi są równe kolejno 1 oraz 1, pomijamy zbędny krok mnożenia i bezpośrednio dodajemy wartość bezwzględną wyznacznika do sumy. Warto podkreślić, że nie moglibyśmy zastosować takiego skrótu w przypadku, w którym ilość węzłów jest inna od dwóch.

```
area += fabs(dx_d_xi * dy_d_eta - dx_d_eta * dy_d_xi);
```

Po zakończeniu prac pętli *j* oraz *i* funkcja zwraca wynik

```
return area;
```

Testy na wybranych przykładach

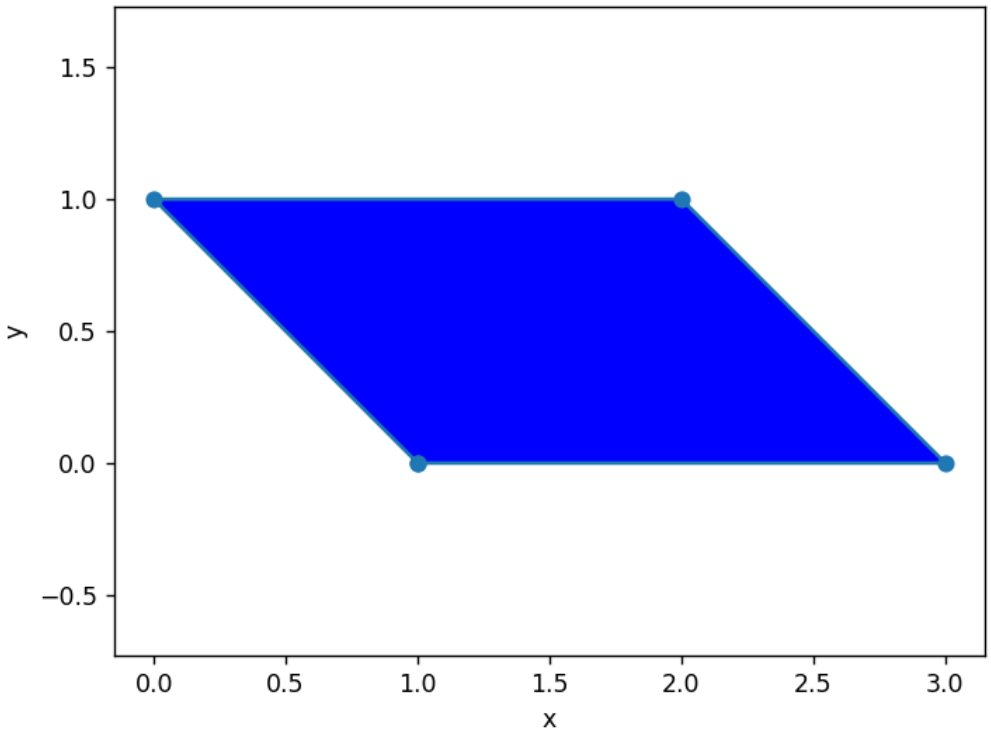
Skuteczność zaimplementowanej metody została przetestowana na tle starannie wyselekcjonowanych przykładów. Metoda zostały przetestowane dla różnych powierzchni, zaś jej wyniki zostały porównane z wynikami Irregular Polygon Area Calculator od omnicalculator.com

Test 1: Trapez o podstawie równoległej do osi X

Koordynaty:

x = [1, 3, 2, 0]

y = [0, 0, 1, 1]



Wykres 1

Powierzchnia:

- wynik Irregular Polygon Area Calculator: 2
- wynik funkcji `gaussian_quadrature_2D`: 2

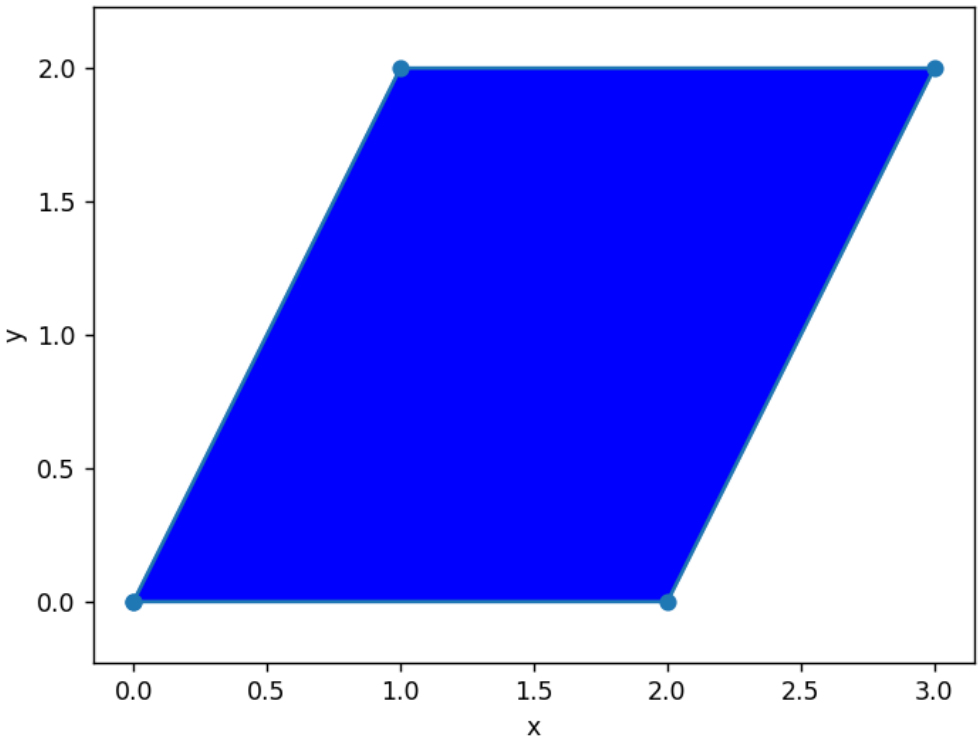
Wyniki są ze sobą zgodne

Test 2: Trapez skośny

Koordynaty:

x = [0, 2, 3, 1]

y = [0, 0, 2, 2]



Wykres 2

Powierzchnia:

- wynik Irregular Polygon Area Calculator: 4
- wynik funkcji `gaussian_quadrature_2D`: 4

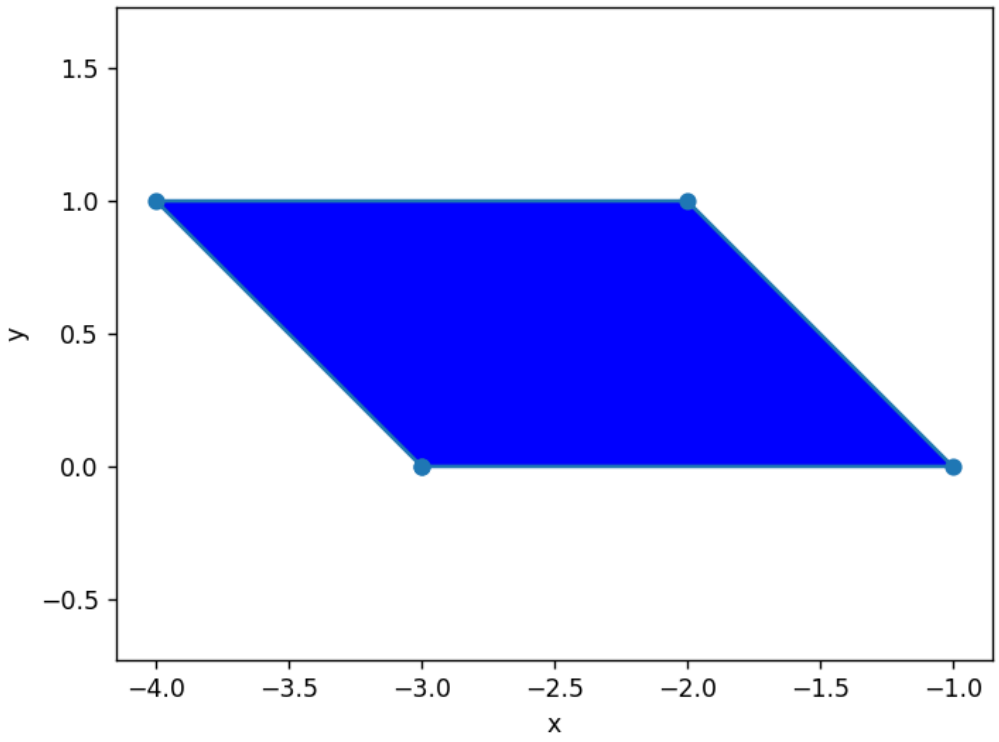
Wyniki są ze sobą zgodne

Test 3: Trapez z wierzchołkiem w negatywnej przestrzeni współrzędnych

Koordynaty:

x = [-3, -1, -2, -4]

y = [0, 0, 1, 1]



Wykres 3

Powierzchnia:

- wynik Irregular Polygon Area Calculator: 2
- wynik funkcji `gaussian_quadrature_2D`: 2

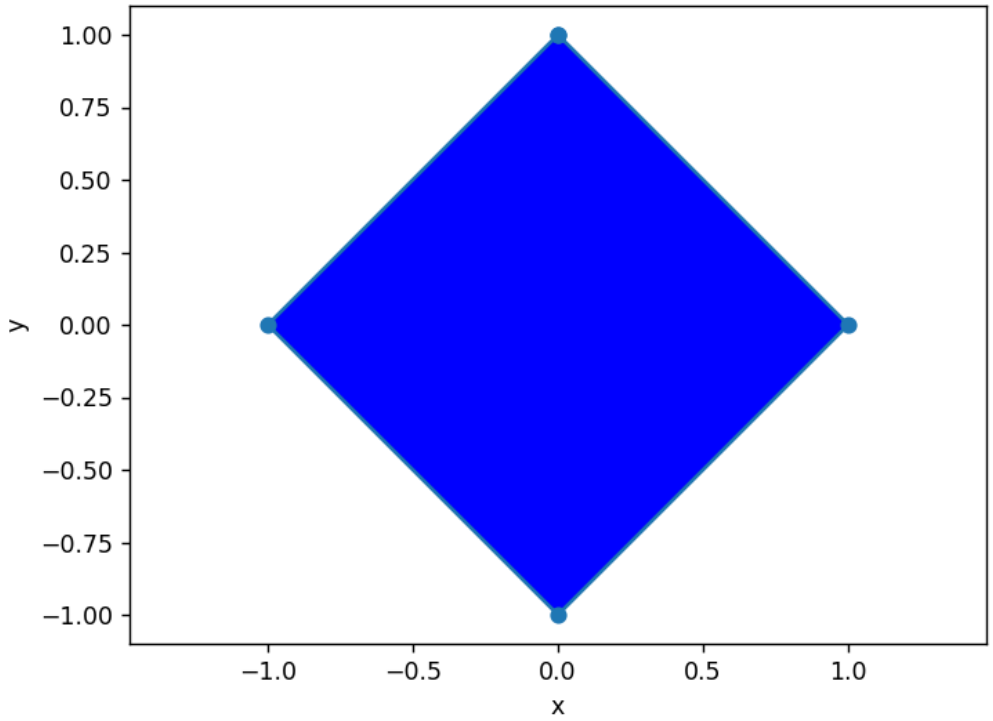
Wyniki są ze sobą zgodne

Test 4: Symetryczny romb (obrót kwadratu)

Koordynaty:

x = [0, 1, 0, -1]

y = [1, 0, -1, 0]



Wykres 4

Powierzchnia:

- wynik Irregular Polygon Area Calculator: 2
- wynik funkcji `gaussian_quadrature_2D`: 2

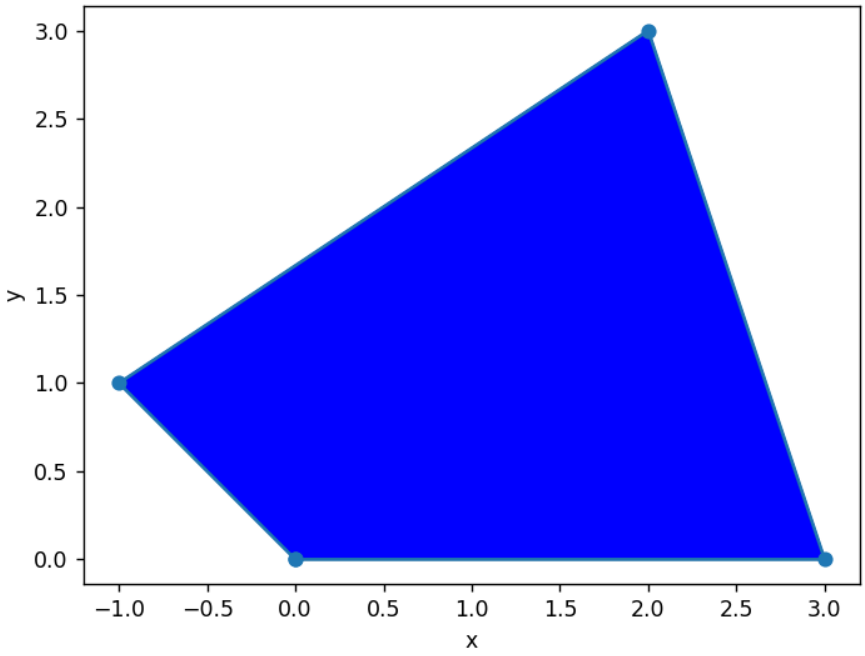
Wyniki są ze sobą zgodne

Test 5: Trapez z nietypowymi kątami

Koordynaty:

x = [0, 3, 2, -1]

y = [0, 0, 3, 1]



Wykres 5

Powierzchnia:

- wynik Irregular Polygon Area Calculator: 7
- wynik funkcji `gaussian_quadrature_2D`: 7

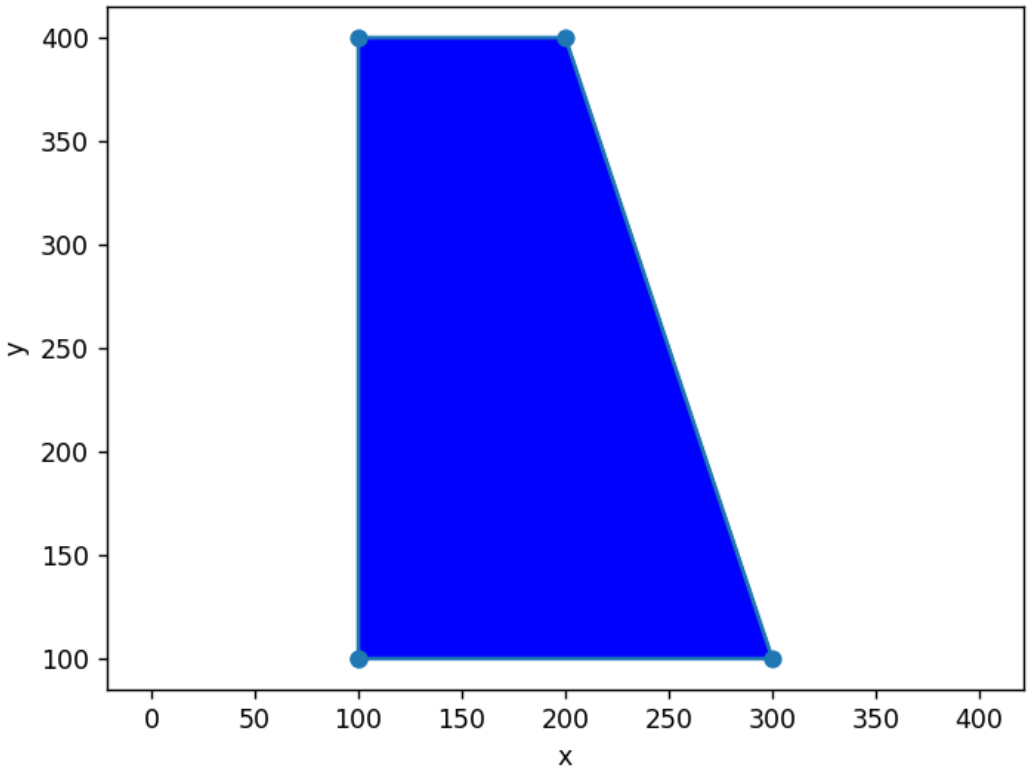
Wyniki są ze sobą zgodne

Test 6: Trapez o dużych wartościach współrzędnych

Koordynaty:

x = [100, 300, 200, 100]

y = [100, 100, 400, 400]



Wykres 6

Powierzchnia:

- wynik Irregular Polygon Area Calculator: 45000
- wynik funkcji `gaussian_quadrature_2D`: 45000

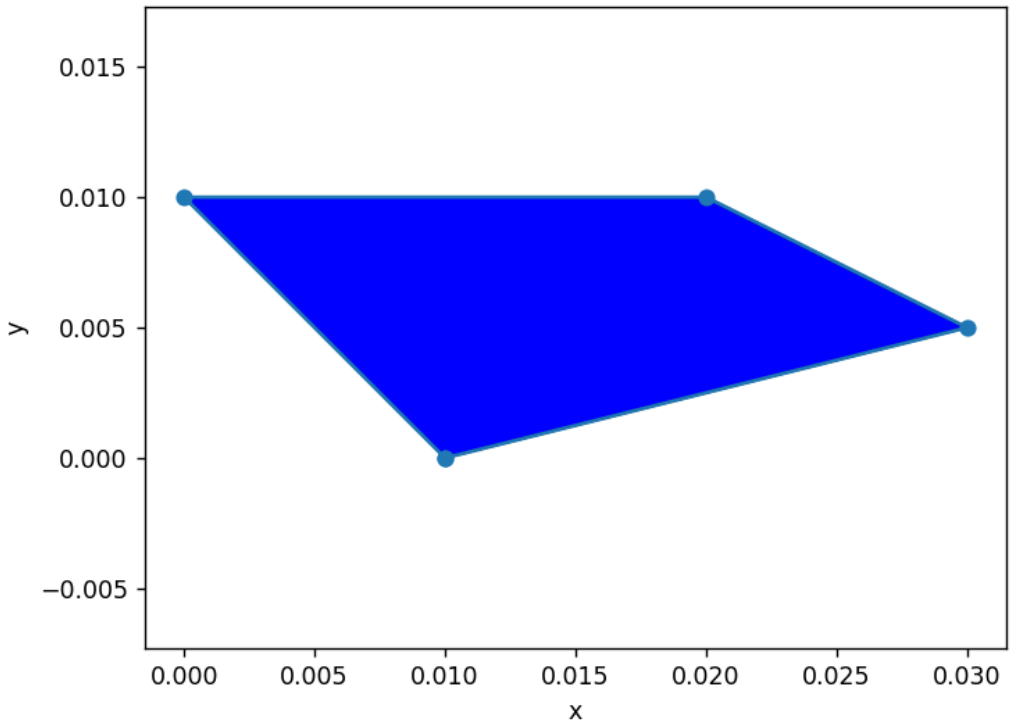
Wyniki są ze sobą zgodne

Test 7: Trapez o małych wartościach współrzędnych

Koordynaty:

x = [0.01, 0.03, 0.02, 0]

y = [0, 0.005, 0.01, 0.01]



Wykres 7

Powierzchnia:

- wynik Irregular Polygon Area Calculator: 0.000175
- wynik funkcji `gaussian_quadrature_2D`: 0.000175

Wyniki są ze sobą zgodne

Opracowanie wyników

Metoda	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7
Irregular Polygon Area Calculator	2	4	2	2	7	45000	0.000175
`gaussian_quadrature_2D`	2	4	2	2	7	45000	0.000175

Tabela 1

Analiza wyników wskazuje na pełną zgodność wartości powierzchni obliczonych za pomocą kwadratury Gaussa z wartościami uzyskanymi z kalkulatora online. Zarówno w przypadku standardowych kształtów, takich jak trapez o podstawie równoległej do osi X (Test 1), jak i dla bardziej skomplikowanych konfiguracji, takich jak trapez z nietypowymi kątami (Test 5) czy trapez o dużych wartościach współrzędnych (Test 6), metoda kwadratury Gaussa wykazała wysoką dokładność.

Dla trapezu skośnego (Test 2) i symetrycznego rombu (Test 4), wyniki ponownie potwierdziły wiarygodność metody. Warto zauważyć, że skuteczność metody została również zachowana dla figur umieszczonych w różnych ćwiartkach układu współrzędnych, co demonstrowane jest przez Test 3 z wierzchołkami znajdującymi się w negatywnej przestrzeni współrzędnych.

Dodatkowo, test zastosowania metody dla bardzo małych wartości współrzędnych (Test 7) wykazał, że metoda kwadratury Gaussa jest wydajna również w skali mikro, zachowując odpowiedni poziom precyzji.

Podsumowanie

Przeprowadzone testy metody kwadratury Gaussa dla obliczania powierzchni różnych geometrii wykazały, że zaimplementowana metoda jest niezawodna i precyzyjna.

Implementacja okazała się skuteczna zarówno dla prostych figur, jak i dla bardziej złożonych kształtów, w tym trapezów o nietypowych kątach oraz figur o małych i dużych wymiarach. Zgodność wyników obliczeń dla figur umieszczonych w różnych ćwiartkach układu współrzędnych dodatkowo podkreśla wszechstronność i metody.