Politechnika Śląska
Wydział Budownictwa

PRACA DYPLOMOWA

Systemy generacji strukturalnych siatek elementów skończonych

Bartłomiej BŁACHOWSKI

promotor dr hab. inż. J.Skrzypczyk
profesor Politechniki Śląskiej

Gliwice 2000

Spis treści

Wstęp **3**

1 Zastosowania w budownictwie **6**
1.1 Wprowadzenie 6
1.2 Sklepienia 8
1.2.1 Sklepienie krzyżowe 8
1.3 Łuki 9
1.3.1 Łuk kołowy 9
1.4 Dachy 12
1.4.1 Dach konoidalny 12
1.4.2 Dach paraboloidalny 14
1.5 Obiekty budownictwa przemysłowego 17
1.5.1 Chłodnia kominowa 17

2 Modelowanie geometryczne wspomagane komputerowo (CAGD) **19**
2.1 Wprowadzenie 19
2.2 Krzywe i powierzchnie Beziera 20
2.2.1 Krzywe Beziera 20
2.2.2 Algorytm de Casteljau 24
2.2.3 Podwyższanie stopnia krzywej Beziera 24
2.2.4 Łączenie segmentów krzywych Beziera 26
2.2.5 Powierzchnie Beziera 28
2.3 Krzywe i powierzchnie NURBS 31
2.3.1 Krzywe NURBS 31
2.3.2 Powierzchnie NURBS 33
2.3.3 Produkt sferyczny 33

3 Algebraiczne systemy generacji siatek (TFI) **37**
3.1 Wprowadzenie 37
3.2 Podstawy matematyczne 37
3.3 Interpolacja algebraiczna (TFI) 39

3.3.1 Sformułowanie metody przy użyciu sum logicznych 39
3.3.2 Sformułowanie rekurencyjne 40
3.4 Praktyczne zastosowania TFI 40
3.4.1 Interpolacja liniowa 40
3.4.2 Interpolacja Lagrange'a 45
3.4.3 Interpolacja Hermite'a 46
3.5 Kontrola gęstości siatki 48
3.5.1 Pojedyncza funkcja wykładnicza 48
3.5.2 Podwójna funkcja wykładnicza 50

4 Eliptyczne systemy generacji siatek 52
4.1 Wprowadzenie 52
4.2 Podstawy matematyczne 52
4.3 Iteracyjne sformułowanie metody 56
4.4 Implementacja numeryczna 56
4.5 Podstawowe konstrukcje siatek eliptycznych 60
4.5.1 Siatki harmoniczne uzyskane metodą Picarda 60
4.5.2 Siatki harmoniczne uzyskane poprzez minimalizację funkcjonału harmonicznego 60
4.5.3 Siatki bazujące na długości łuku 65
4.5.4 Siatki brzegowo ortogonalne 66

Literatura 68

Wstęp

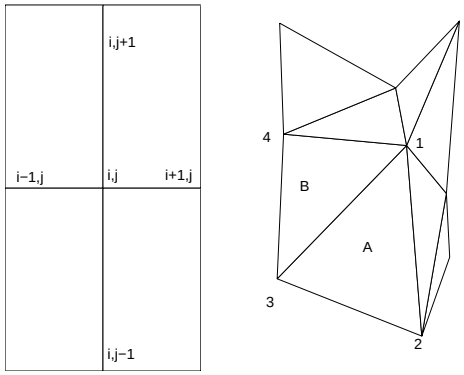
Generacja siatki jest podstawowym narzędziem wykorzystywanym do symulacji komputerowych wielu zjawisk i procesów fizycznych.

Z technicznego punktu widzenia, generacja siatki jest, wciąż czymś bardziej w rodzaju sztuki niż nauki. Pomimo iż aparat matematyczny dostarcza solidnych podstaw w celu przeniesienia ciężaru generacji siatki z użytkownika na programy komputerowe, w dalszym ciągu zarówno nauka jak i sztuka są wykorzystywane przy tworzeniu konkretnych algorytmów obliczeniowych. Dzieje się tak dlatego, iż nie istnieje jedna uniwersalna metoda pozwalająca tworzyć siatki w każdym, dowolnie wybranym przypadku. Dlatego też, do użytkownika należy ostateczna decyzja co do wyboru najwłaściwszego algorytmu, a ułatwić ten wybór mają kryteria jakie spełniać powinna otrzymana siatka.

I chociaż ostatnio pojawił się na rynku oprogramowania cały szereg generatorów siatek gotowych do zastosowania, znajomość pewnych podstaw teoretycznych zapewnia nie tylko poprawne, ale również efektywne i co najważniejsze świadome korzystanie z wyżej wymienionych programów.

Od momentu powstania metody elementu skończonego obserwujemy bezustanny rozwój technik generacji siatek i wciąż dowiadujemy się o pojawiających się modyfikacjach już istniejących. Na dzień dzisiejszy możemy jednak podzielić systemy generacji siatek na dwie grupy. Do pierwszej grupy zaliczamy systemy strukturalne, w których główna idea polega na przekształceniu dowolnego krzywoliniowego konturu w prostokątny regularny obszar, następnie wygenerowaniu siatki w tym obszarze, a na koniec poprzez transformację odwrotną uzyskanie ostatecznej siatki. Drugą grupę stanowią systemy generacji siatek niestrukturalnych, w których wypełniamy interesujący nas obszar stopniowo tworząc w ten sposób nieuporządkowany zbiór trójkątów bądź czworokątów. Kolejną zasadniczą różnicą pomiędzy strukturalnymi a niestrukturalnymi algorytmami jest sposób reprezentacji danych.

W przypadku algorytmów strukturalnych, w pamięci komputera przechowujemy jedynie współrzędne węzłów siatki zorganizowane odpowiednio w macierzach X_{ij}, Y_{ij} oraz Z_{ij} (rys 1). Natomiast dla siatek niestrukturalnych musimy dodatkowo zapamiętać połączenia między węzłami siatki i tak na rysunku 1 mamy element A utworzony przez węzły 1,2,3 ; element B przez węzły 1,3,4 itd.



Rysunek 1: Połączenie węzłów dla siatek strukturalnych i niestrukturalnych

Celem niniejszej pracy jest przedstawienie i usystematyzowanie istniejącej wiedzy na temat strukturalnych siatek elementów skończonych. Oprócz podstawowych zagadnień teoretycznych zamieszczono również szereg przykładów, które wydają się być najbardziej interesujące z punktu widzenia inżyniera budownictwa. Należy jednak zaznaczyć, że do pełnego opisu konstrukcji budowlanej, którą następnie moglibyśmy obliczyć metodą elementów skończonych, oprócz samej siatki elementów konieczny jest jeszcze wybór odpowiedniego typu elementu. Musimy ocenić czy charakter pracy konstukcji jest sprężysty, sprężysto-plastyczny, czy też należy zastosować element skończony właściwy konstrukcjom wykonanym z materiałów lepkosprężystych. Ten problem stanowi jednak odrębne zagadnienie, którego w tej pracy nie poruszano.

Praktyczne wykorzystanie opisanych w dalszej części pracy systemów generacji siatek zawiera rozdział 1. Znajdują się w nim siatki uzyskane dla konstrukcji mających szczególne znaczenie w problematyce projektowania budowlanego. Przedstawiono następujące przykłady:

1. sklepienie krzyżowe
2. łuk kołowy
3. dach konoidalny
4. dach paraboloidalny
5. hiperboloidalna chłodnia kominowa

Procedura generowania siatki rozpoczyna się zazwyczaj od zdefiniowania geometrii brzegu rozpatrywanego obszaru. Z tego powodu, w rozdziale 2, zdecydowano

się umieścić niezbędne informacje na temat modelowania geometrycznego wspomaganego komputerem. Zawarto w nim podstawowe zagadnienia CAD takie jak: krzywe i powierzchnie Beziera oraz krzywe i powierzchnie NURBS. Kolejnym krokiem podczas generacji siatki jest odpowiednia dyskretyzacja zadania, co jest równoważne z właściwym rozmieszczeniem punktów na ograniczeniach brzegowych. Ostatecznie, w przypadku generatorów strukturalnych, należy wybrać metodę obliczenia współrzędnych węzłów siatki. Do dyspozycji mamy dwa rodzaje metod:

- metoda algebraiczna (TFI) - siatka uzyskiwana jest poprzez jednokierunkową interpolację
- metoda eliptyczna (PDE) - siatka otrzymywana jest poprzez rozwiązanie układu równań różniczkowych cząstkowych rzędu drugiego.

Algebraiczna generacja siatki, opisana w rozdziale 3, jest najszybszą techniką spośród systemów generacji strukturalnych siatek i w związku z tym najczęściej wykorzystywana do utworzenia początkowego rozmieszczenia węzłów w systemach eliptycznych. Pomimo ogromnej zalety jaką niewątpliwie jest szybkość, systemy bazujące na TFI mogą, w przypadku dużych wypukłości fragmentów brzegu, generować siatki wychodzące poza kontur brzegowy. Dodatkową wadą jest nie najlepsza gładkość uzyskanej siatki co ostatecznie może wpłynąć na znaczne błędy obliczeń, a w skrajnym przypadku zdyskwalifikować taką siatkę.

Dla systemów bazujących na metodzie eliptycznej uzyskujemy znacznie lepszą gładkość siatki oraz całkowitą pewność uzyskania siatki znajdującej się wewnętrz konturu. Oczywiste jest, że odbija się to wszystko na ilości obliczeń koniecznych do wykonania. Opis eliptycznej metody generacji strukturalnych siatek zamieszczono w rozdziale 4.

Na zakończenie warto jeszcze raz zaznaczyć, że tematyka dotycząca generowania siatek, zarówno strukturalnych jak i niestrukturalnych, jest stosunkowo młoda i w niedalekiej przyszłości należy spodziewać się jej dalszego dynamicznego rozwoju.

Rozdział 1

Zastosowania w budownictwie

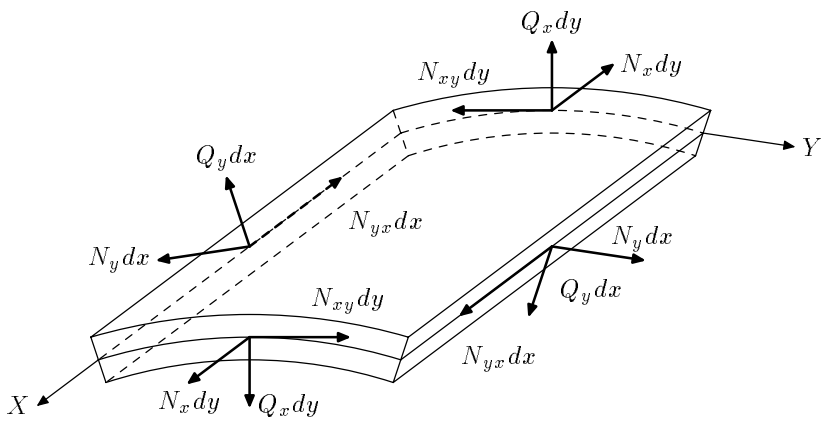
1.1 Wprowadzenie

Obserwowany w ostatnich latach rozwój metod komputerowych i technik obliczeniowych, w szczególności metody elementu skończonego, sprawił, iż wiele nauk inżynierskich zostało ukierunkowanych na obliczenia numeryczne.

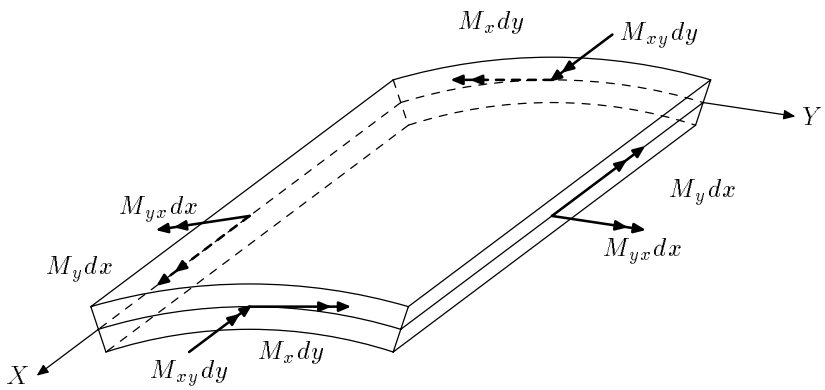
Przy pomocy komputera stało się możliwe wykonywanie obliczeń, których do tej pory nie można było wykonywać ze względu na ogromny nakład pracy. Obecnie uzyskanie wielkości naprężeń, panujących w nawet najbardziej skomplikowanych konstrukcjach budowlanych, jest możliwe bez konieczności używania metod uproszczonych dających bardzo często jedynie wartości szacunkowe. Projektowanie na podstawie wyników uzyskanych metodami uproszczonymi prowadzi do znacznego przewymiarowania konstrukcji a co za tym idzie nieekonomicznego wykorzystania materiału. Kolejnym atutem, przemawiającym za używaniem komputera w pracy projektanta, stała się możliwość optymalizacji konstrukcji według wybranego kryterium np. minimalizacja ciężaru, kosztów produkcji itp.

Jednak pomimo wspomnianych zalet w wyniku komputeryzacji procesu projektowania pojawiły się również i nowe problemy. Jednym z tych problemów jest oczywiście generacja siatki dla dowolnie ukształtowanej konstrukcji. W przypadku konstrukcji budowlanych niejednokrotnie mamy do czynienia z konstrukcjami będącymi powierzchniami prostokreślnymi. W niniejszym rozdziale przedstawiono przykłady siatek wygenerowanych dla takich właśnie konstukcji. Dodatkowo oprócz rysunków ilustrujących omawiane zagadnienia dołączono fragmenty skryptów z opisem rozwiązania problemu w środowisku Matlab.

Ze względu na małe grubości powłokowych konstrukcji budowlanych w stosunku do ich pozostałych wymiarów zadanie poszukiwania rozkładu naprężeń możemy sprowadzić do poszukiwania wielkości sił i momentów. Interpretację graficzną wielkości wewnętrznych panujących w konstrukcji tego typu przedstawiono na ry-



Rysunek 1.1: Siły dla elementarnego wycinka powłoki walcowej



Rysunek 1.2: Momenty dla elementarnego wycinka powłoki walcowej

sunkach 1.1 oraz 1.2.

Do konstrukcji, które spełniają powyższe założenie możemy zaliczyć:

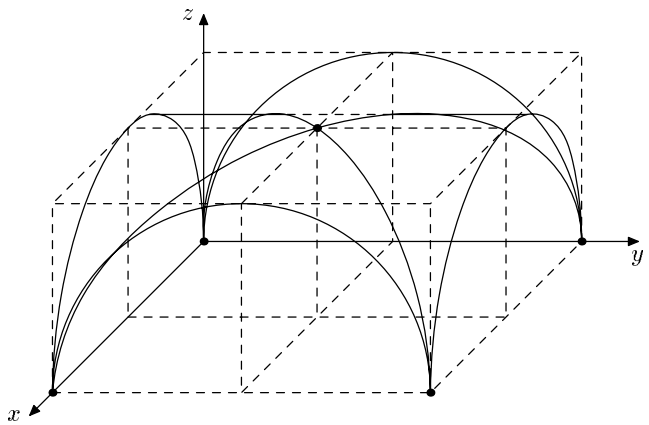
- chłodnie kominowe
- przekrycia konoidalne
- tarczownice
- powłoki cylindryczne

Więcej informacji na temat klasycznej teorii powłok można znaleźć w książce [2], natomiast metodzie elementu skończonego poświęcono pracę [14].

1.2 Sklepienia

1.2.1 Sklepienie krzyżowe

W tym punkcie przedstawiono jedno z możliwych rozwiązań zadania generacji siatki dla sklepienia krzyżowego (rys 1.3).



Rysunek 1.3: Sklepienie krzyżowe

Dla tego typu konstrukcji rozsądny wydaje się wybór metody algebraicznej, a wynika to z łatwości wyznaczenia czterech krzywych ograniczających. Dodatkowo jak łatwo zauważyć zadanie to jest symetryczne, więc rozpatrujemy tylko jeden ćwierćwalec.

Przekład powyższej procedury postępowania na język zrozumiały dla komputera wygląda następująco:

```
%
% Przykład siatki TFI dla sklepienia krzyżowego
%

n = 10 ;
fi = 0 :pi/(2*(n-1)):pi/2 ;
x = zeros(n) ; y = x ; z = x ;

x(1,:) = -cos(fi)       ; x(n,:) = -cos(fi) ;
x(:,1) = -ones(n,1)     ; x(:,n) = zeros(n,1) ;

y(1,:) = cos(fi)        ; y(n,:) = 3*ones(1,n);
y(:,1) = [1:2/(n-1):3]' ; y(:,n) = [0:3/(n-1):3]' ;
```

```
z(1,:) = sin(fi)        ; z(n,:) = sin(fi) ;
z(:,1) = zeros(n,1)     ; z(:,n) = ones(n,1) ;

[x1,y1,z1] = linearTFI(x,y,z)  ;  [x2,y2,z2] = linearTFI(-x,y,z)  ;
[x3,y3,z3] = linearTFI(x,-y,z) ;  [x4,y4,z4] = linearTFI(-x,-y,z) ;
[x5,y5,z5] = linearTFI(y,x,z)  ;  [x6,y6,z6] = linearTFI(-y,x,z)  ;
[x7,y7,z7] = linearTFI(y,-x,z) ;  [x8,y8,z8] = linearTFI(-y,-x,z) ;

% Operacje graficzne

figure, hold on, axis off, colormap gray, rotate3d, view(-70,78)
surf(x1,y1,z1) ; surf(x2,y2,z2) ; surf(x3,y3,z3) ; surf(x4,y4,z4) ;
surf(x5,y5,z5) ; surf(x6,y6,z6) ; surf(x7,y7,z7) ; surf(x8,y8,z8) ;
```

W wyniku działania powyższego skryptu uzyskujemy siatkę przedstawioną na rysunku 1.4. Zawartość odpowiadajcą za operacje graficzne stanowią standardowe procedury programu Matlab, natomiast opis dodatkowej funkcji `linearTFI(x,y,z)` wykorzystanej do obliczenia współrzędnych węzłów siatki zamieszczono w rozdziale 3.

1.3 Łuki

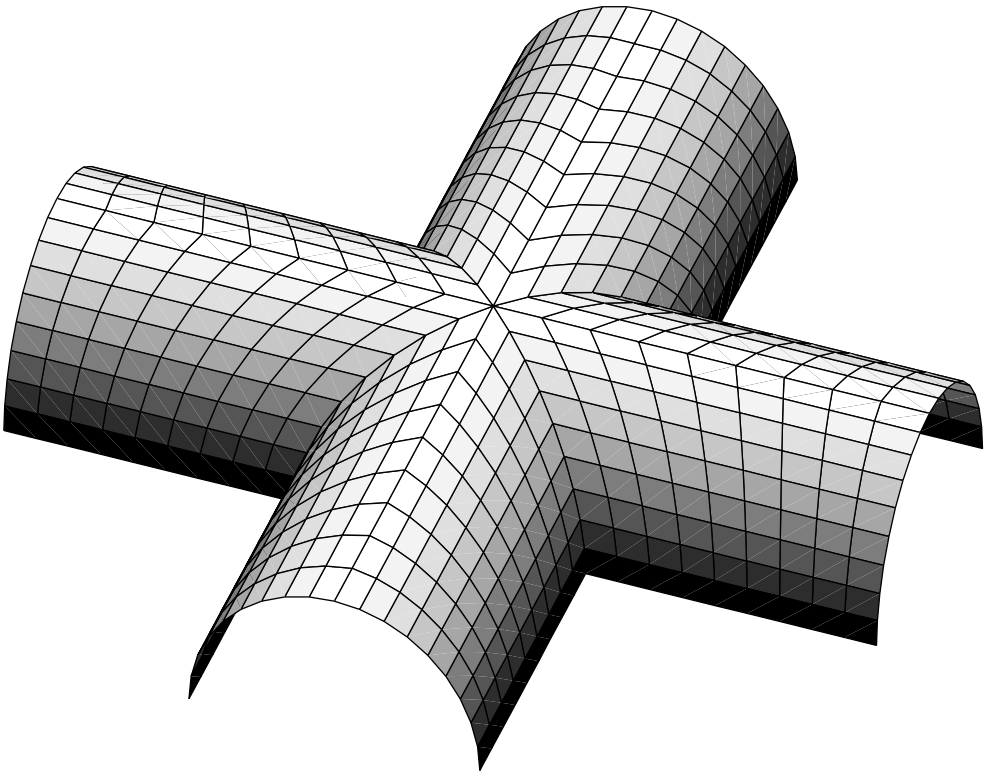
1.3.1 Łuk kołowy

Znaczną grupę, spośród ciekawych architektonicznie, konstrukcji budowlanych stanowią konstrukcje łukowe. W tym punkcie zajmiemy się łukiem kołowym (rys 1.5).

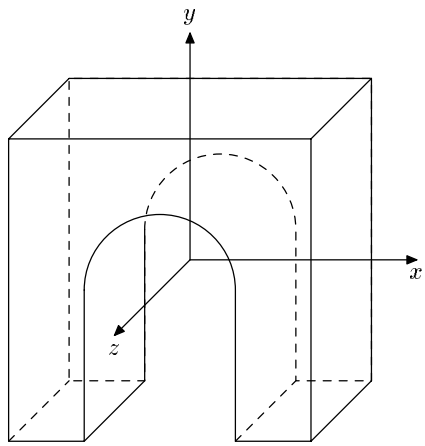
Do tego typu konstrukcji zastosujemy siatkę przestrzenną, którą zrealizujemy przy użyciu algorytmu interpolacji algebraicznej, a następnie wygenerujemy inną siatkę przez zastosowanie metody eliptycznej w celu porównania zalet i wad tych metod. Aby zaakcentować podstawowe różnice pomiędzy systemami algebraicznymi i różniczkowymi zastosujemy różne podziały na opisane w rozdziale 4 bloki. Trzy charakterystyczne fragmenty konstrukcji dla metody TFI przedstawiono na rysunku 1.6, natomiast podział blokowy dla systemów eliptycznych obrazuje rysunek 1.7.

Po takim podziale konstrukcji na łatwiejsze do zamodelowania fragmenty możemy przystąpić do generowania siatki dla poszczególnych jej części oddzielnie.

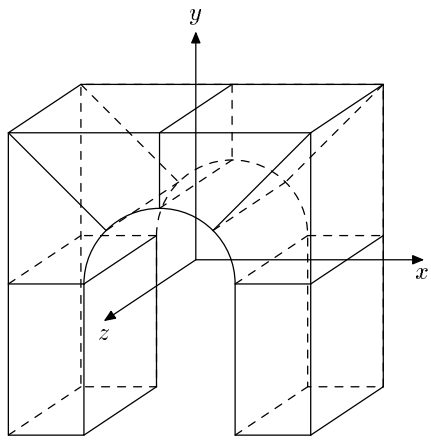
Aby jednak ostatecznie otrzymana siatka dla całego łuku nadawała się do zastosowania metody elementów skończonych, należy pamiętać o równym podziale



Rysunek 1.4: Siatka elementów skończonych dla sklepienia krzyżowego



Rysunek 1.5: Łuk kołowy

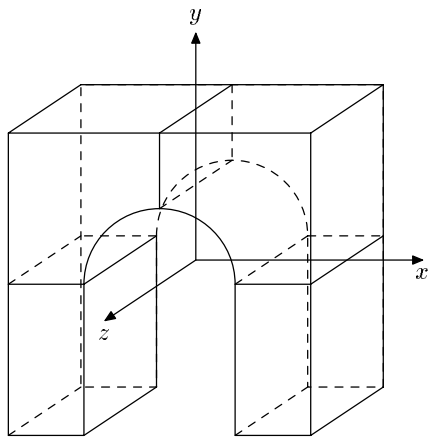


Rysunek 1.6: Podział konstrukcji na bloki dla metody algebraicznej

odpowiadających sobie krawędzi. Oznacza to, że bloki posiadające wspólną krawędz po złączeniu ze sobą powinny posiadać na wspólnej krawędzi węzły, które pokrywają się nawzajem. W przeciwnym przypadku siatka końcowa nie będzie mogła być wykorzystana do dalszych obliczeń.

Uwzględniając powyższe uwagi oraz stosując algorytm generacji siatki opisany w rozdziale 3 otrzymamy siatkę elementów pokazaną na rysunku 1.8.

Jak widać siatka uzyskana metodą algebraiczną nie zapewnia gładkiego przejścia linii pomiędzy sąsiednimi elementami. W celu osiągnięcia lepszej gładkości oraz większej kontroli nad kształtem generowanej siatki zastosujmy metodę eliptyczną. Jak już wcześniej wspomniano dokonamy również nowego wyboru bloków,



Rysunek 1.7: Podział blokowy dla metody eliptycznej

na których przeprowadzimy stosowne obliczenia (rys 1.7).

Zgodnie z zaleceniem autorów pracy [12] ustalamy liczbę iteracji Picarda na 10. Po wykonaniu obliczeń otrzymujemy siatkę pokazaną na rysunku 1.8.

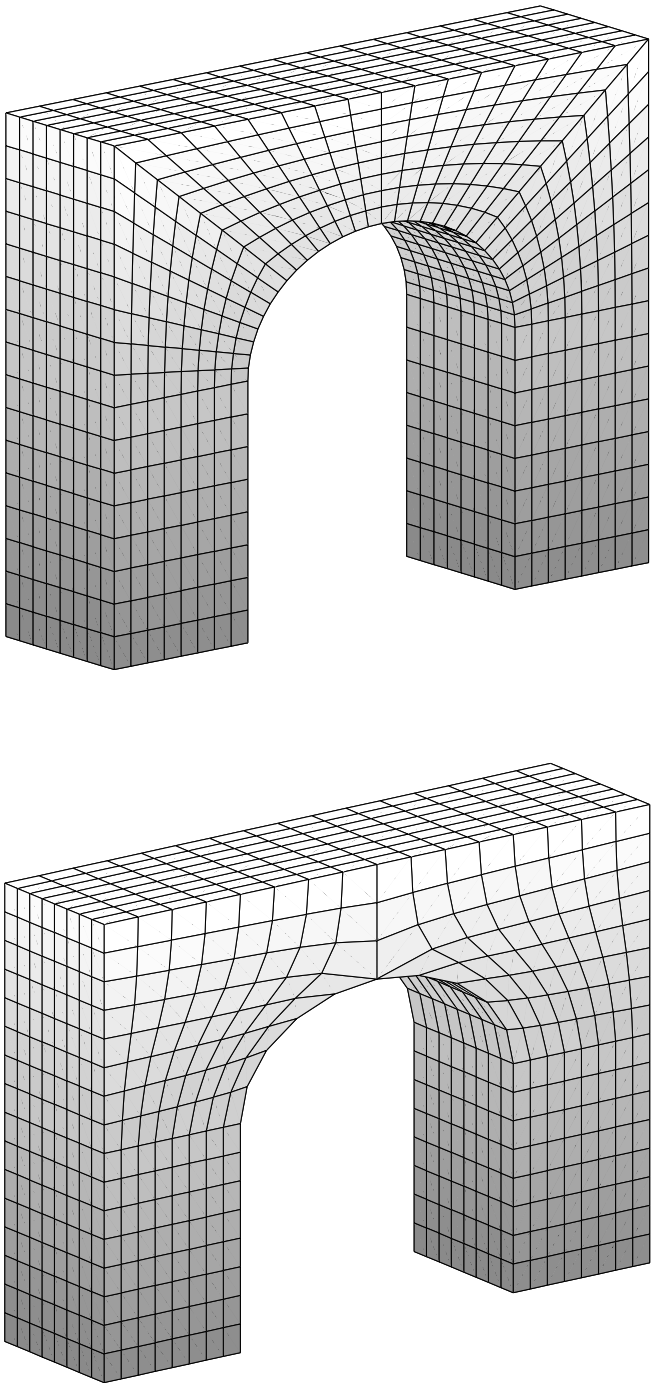
1.4 Dachy

1.4.1 Dach konoidalny

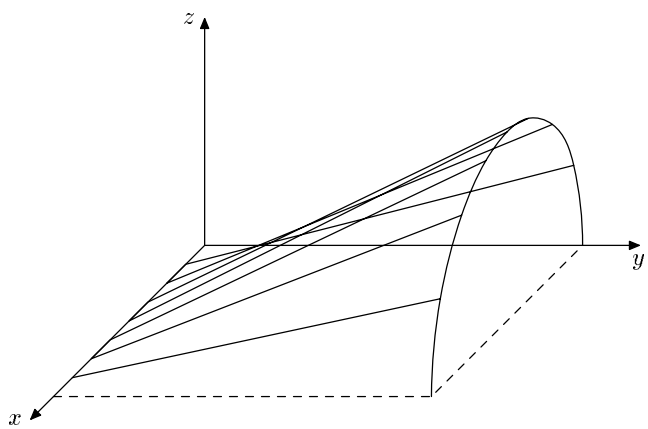
Jako następny przykład zostanie zaprezentowany dach składający się z trzech fragmentów powierzchni konoidalnej. Wygląd konoidy bazującej na łuku kołowym pokazano na rysunku 1.9.

Ponieważ w tym zadaniu również nie napotkamy trudności z określeniem czterech krzywych brzegowych (3 proste i półokrąg) wybieramy ponownie metodę liniowej interpolacji TFI. Zapis komputerowy problemu ukazuje poniższy listing:

```
%
% Przekrycie konoidalne
%

n = 20 ;
x = zeros(n) ; y = x ; z = x ;
fi = 0 :pi/(n-1):pi ;

x(1,:) = -cos(fi) ;     x(n,:) = x(1,:) ;
x(:,1) = -1*ones(n,1) ; x(:,n) =    ones(n,1) ;
```



Rysunek 1.8: Porównanie siatki otrzymanej metodą TFI(na górze) oraz poprzez rozwiązania układu Poissona (na dole)



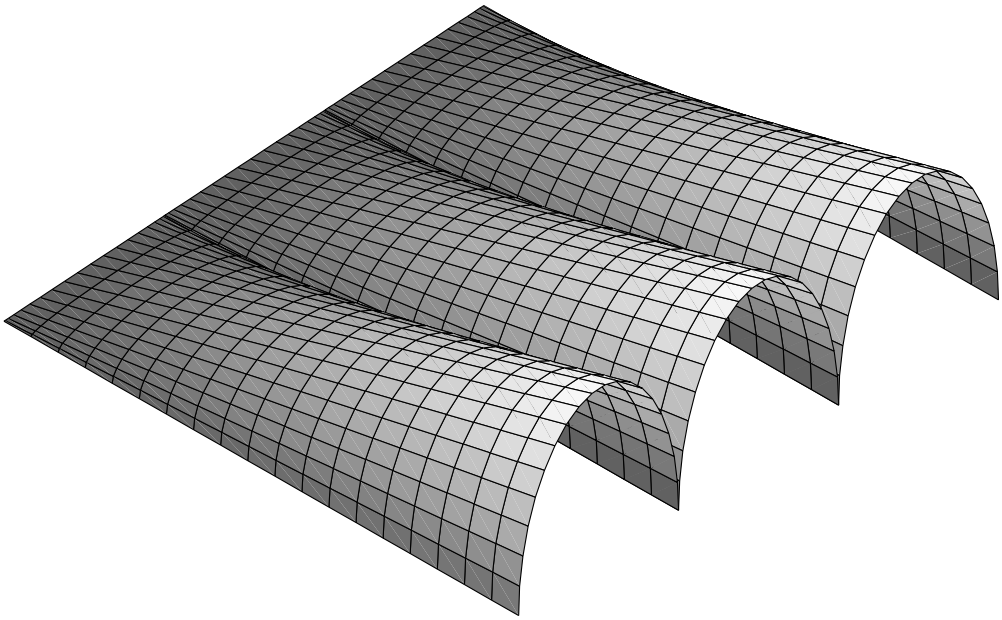
Rysunek 1.9: Konoida

```
y(1,:) = zeros(1,n) ;   y(n,:) = ones(1,n) ;
y(:,1) = [0:1/(n-1):1]';y(:,n) = [0:1/(n-1):1]' ;

z(1,:) = zeros(1,n) ;   z(n,:) = sin(fi) ;
z(:,1) = zeros(n,1) ;   z(:,n) = zeros(n,1) ;

[x1,y1,z1]=linearTFI(x,y,z) ;
[x2,y2,z2]=linearTFI(x+2,y,z) ;
[x3,y3,z3]=linearTFI(x+4,y,z) ;

% Operacja graficzne

figure,hold on,rotate3d, view(133,68) ;
surf(x1,y1,z1) ; surf(x2,y2,z2) ; surf(x3,y3,z3); axis off
kolor = gray ; colormap(kolor(25:64,:));
```

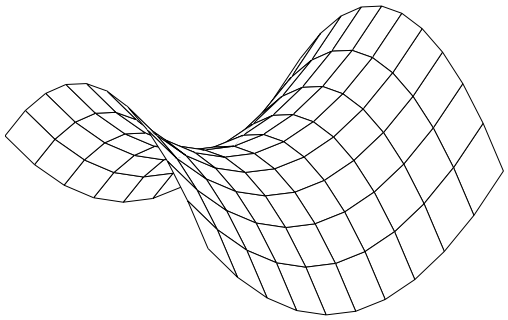
Ostatecznie po przetworzeniu takiego skryptu w środowisku Matlab otrzymamy siatkę pokazaną na rysunku 1.10.

1.4.2 Dach paraboloidalny

Kolejną powierzchnią prostokreślną wykorzystywaną bardzo chętnie przez architektów jest paraboloida hiperboliczna(rys 1.11).



Rysunek 1.10: Siatka wygenerowana dla dachu konoidalnego

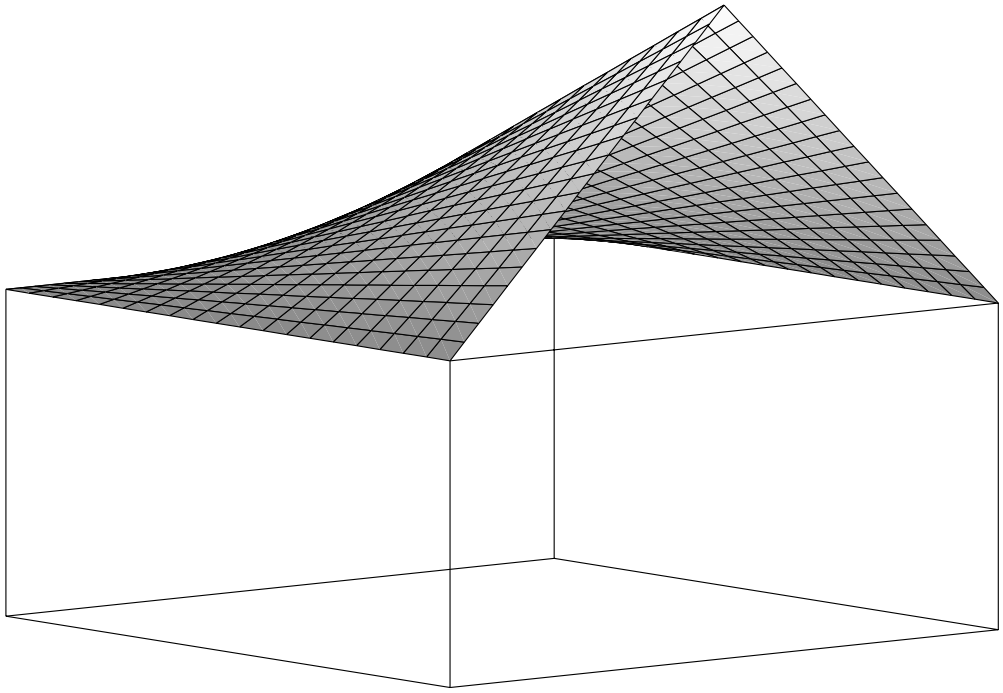


Rysunek 1.11: Paraboloida hiperboliczna

Równanie paraboloidy hiperbolicznej ma postać

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 2z$$

Rozpatrzmy zatem konstrukcję dachową nad prostopadłościenną halą złożoną z dwóch takich płatów. Okazuje się, iż ten przykład jest jeszcze prostszy od poprzedniego ponieważ cztery krzywe ograniczające stanowią linie proste. Po odpowiedniej modyfikacji listingu z poprzedniego punktu uzyskamy siatkę elementów skończonych przedstawioną na rysunku 1.12.

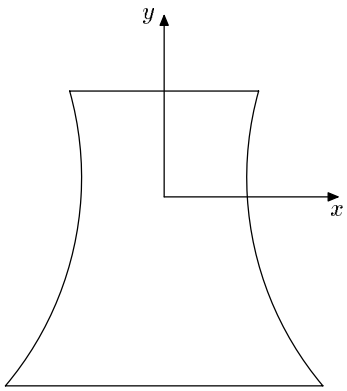


Rysunek 1.12: Siatka wygenerowana dla dachu złożonego z dwóch płatów paraboloidy hiperbolicznej

1.5 Obiekty budownictwa przemysłowego

1.5.1 Chłodnia kominowa

Klasycznym przykładem powłokowej konstrukcji budowlanej jest chłodnia kominowa o kształcie hiperboloidy jednopowłokowej (rys. 1.13).



Rysunek 1.13: Hiperboloida jednopowłokowa

W celu wygenerowania siatki elementów skończonych dla tej powierzchni stosujemy metodę produktu sferycznego (rozdział 2). Jako równoleżnik wskażemy oczywiście okrąg o środku w początku układu współrzędnych, natomiast południkiem będzie hiperbola (rys. 1.13) o równaniu

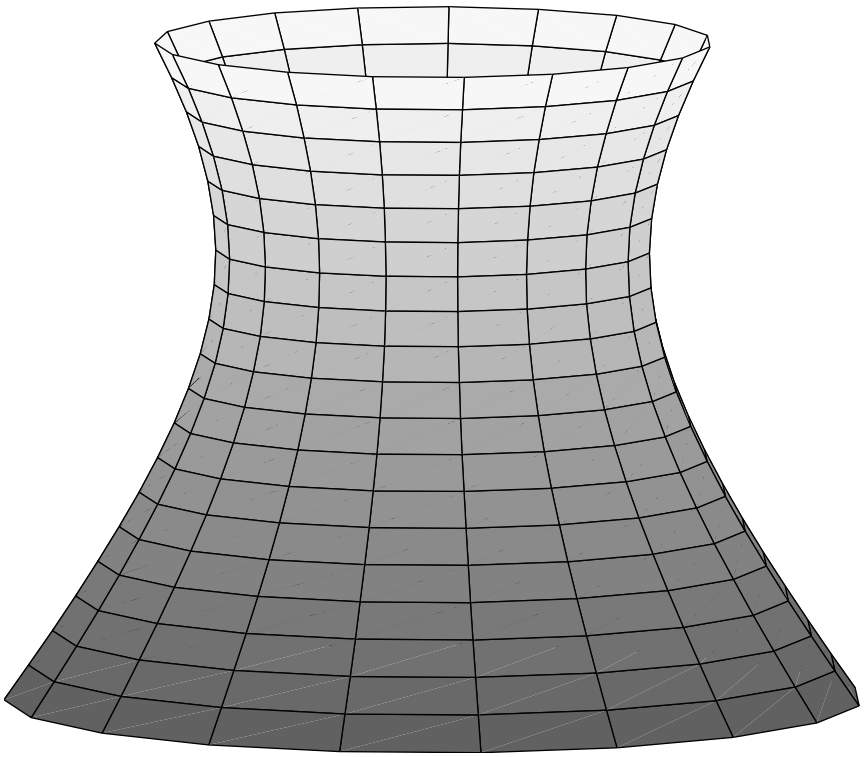
$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$$

Po określeniu południka i równoleżnika możemy powyższe zadanie zapisać w postaci następującego skryptu programu Matlab:

```
%
% Chłodnia kominowa
%

n   = 20                  ;
fi  = 0:2*pi/(n-1):2*pi   ;
ksi = -1.7:2.5/(n-1):0.8  ;

xr = cos(fi)              ;  yr = sin(fi) ;
xq = sqrt(1 + ksi.*ksi)   ;  yq = ksi     ;
```

```
[x1,y1,z1] = prodsfer(xr,yr,xq,yq) ;

% Operacja graficzne

figure , rotate3d ,surf(x1,y1,z1) , axis off ,view(-30,8) ;
kolor = gray ; colormap(kolor(25:64,:)) ;
```

a na wyjściu otrzymamy regularną siatkę południków i równoleżników pokazaną na rysunku 1.14.



Rysunek 1.14: Siatka elementów skończonych dla chłodni kominowej

Rozdział 2

Modelowanie geometryczne wspomagane komputerowo (CAGD)

2.1 Wprowadzenie

Jednym z podstawowych zadań projektantów przemysłu lotniczego, samochodowego, czy też maszynowego jest konstruowanie gładkich krzywych, które interpolują zbiór zadanych punktów. Historycznie najstarsze rozwiązanie tego problemu było dziełem Lagrange'a. Skonstruował on wielomian interpolacyjny postaci

$$\mathbf{p}(t) = \sum_{i=0}^{n} \mathbf{p}_i \left(\prod_{\substack{j=0,\ldots,n \\ j \neq i}} \frac{t - u_j}{u_i - u_j} \right) \quad ,$$

który spełniał powyższy warunek. Jednak jak się okazuje przy dużej liczbie punktów, a co za tym idzie przy konstruowaniu wielomianu wysokiego stopnia, dochodzi do niekontrolowanych oscylacji takiej krzywej co w dużej mierze utrudnia proces projektowania.

Z tego powodu konieczne było wprowadzenie nowej metody, która polegać miała na łączeniu fragmentów krzywych stopnia trzeciego tzn. krzywych sklejanych. Takie krzywe bardzo szybko przyjęły się w projektowaniu i już w XIX wieku w przemyśle stoczniowym wykorzystywano giętką sprężystą listwę (ang. *spline*), którą podpierając w wybranych punktach można było otrzymać krzywą interpolacyjną trzeciego stopnia.

Potrzeba projektowania złożonych kształtów przy użyciu komputera doprowadziła do powstania nowej gałęzi grafiki komputerowej zwanej skrótowo CAGD

(Computer Aided Geometric Design - Komputerowe Wspomaganie Geometrycznego Projektowania). To właśnie za sprawą aplikacji typu CAD/CAM obserwujemy jak odchodzą w zapomnienie tradycyjne biura projektowe wypełnione linijkami, cyrklami, czy też deskami kreślarskimi. Ich miejsce coraz częściej na biurku inżyniera zajmuje komputer.

W związku ze stale rosnącą popularnością systemów CAD i ich ścisłym związkiem z metodami generacji siatek w tym rozdziale przedstawiono przegląd podstawowych technik modelowania geometrycznego.

2.2 Krzywe i powierzchnie Beziera

2.2.1 Krzywe Beziera

Dowolną krzywą w przestrzeni trójwymiarowej możemy wyrazić za pomocą następujących trzech równań parametrycznych

$$\begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix} = \mathbf{x}(t) = \sum_{i=0}^{n} \mathbf{c}_i F_i(t)$$

gdzie

$F_i(t)$- funkcje bazowe
$\mathbf{c}_i$- wektor współczynników liczbowych

Najpowszechniej znane jest podejście, w którym jako funkcje bazowe wybierane są jednomiany

$$F_i(t) = t^i \quad .$$

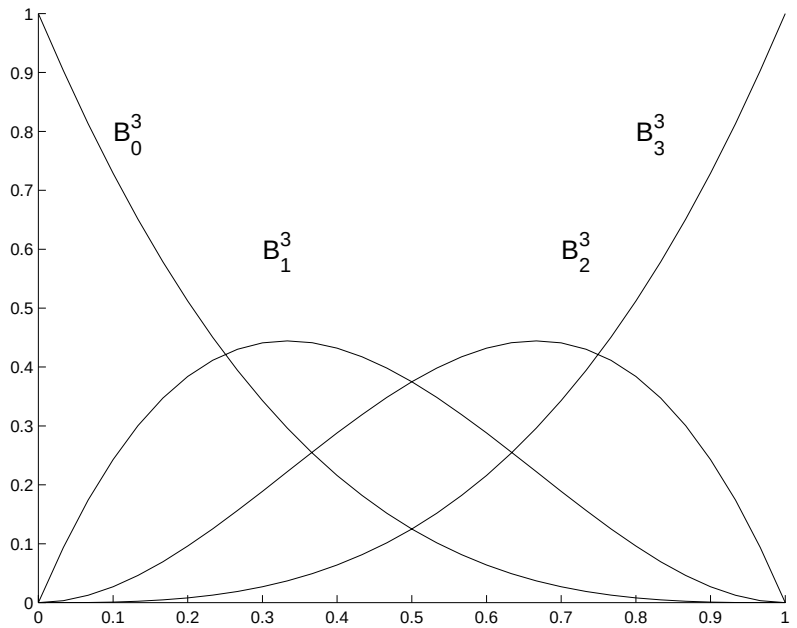
Jednak takie podejście nie jest poręczne z punktu widzenia zastosowań komputerowych a związane jest to z określaniem współczynników liczbowych.

Jako pierwszy zajął się tym problemem francuski matematyk, zatrudniony przy projektowaniu nadwozi samochodowych w firmie Renault, Pierre Bezier. Użył on jako funkcji bazowych wielomianów Bernsteina określonych równaniem

$$B_i^n(t) = \binom{n}{k} t^i (1-t)^{n-i} \qquad i \in [0, n]$$

W praktyce stosowane funkcje Bernsteina ograniczone są do $n = 3$. Przebieg wielomianów Bernsteina dla $i = 0, 1, 2, 3$ przedstawiono na rysunku 2.1. Po zastosowaniu jako funkcji bazowych wielomianów Bernsteina krzywe Beziera możemy zapisać jako

$$\mathbf{p}(t) = \sum_{i=0}^{n} \mathbf{p}_i B_i^n(t) = \mathbf{p}_0 B_0^3(t) + \mathbf{p}_1 B_1^3(t) + \mathbf{p}_2 B_2^3(t) + \mathbf{p}_3 B_3^3(t) =$$
$$= \mathbf{p}_0 (1-t)^3 + \mathbf{p}_1 3t(1-t)^2 + \mathbf{p}_2 3t^2(1-t) + \mathbf{p}_3 t^3$$



Rysunek 2.1: Wielomiany Bernsteina

Wówczas współczynniki liczbowe $\mathbf{p}_i$ określone są bezpośrednio przez punkty kontrolne. Jak łatwo zauważyć taka krzywa interpoluje dwa końcowe punkty kontrolne ($\mathbf{p}_0$ oraz $\mathbf{p}_3$) i aproksymuje dwa pozostałe. Przykład krzywej Beziera przedstawia rysunek 2.2. Ze względu na ważną rolę jaką pełnią krzywe Beziera warto wymienić ich podstawowe własności:

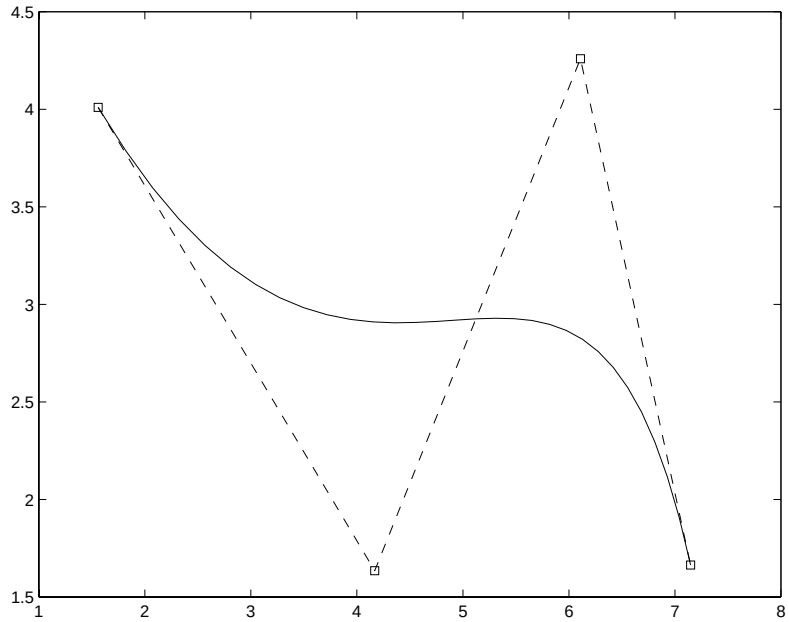
- niezmienniczość względem afinicznej transformacji parametru. Własność powyższą można wyrazić za pomocą wzoru

$$\sum_{i=0}^{n} \mathbf{p}_i B_i^n(t) = \sum_{i=0}^{n} \mathbf{p}_i B_i^n \left(\frac{u-a}{b-a} \right)$$

  Oznacza on, że krzywą możemy zdefiniować na przedziale $[a, b]$ równie dobrze jak na przedziale [0,1].

- własność wielokąta wypukłego. Dowolny punkt krzywej Beziera znajduje się wewnątrz wielokąta rozpiętego na punktach kontrolnych, jeśli tylko wartość parametru zawiera się pomiędzy zerem a jednością. Własność powyższa wynika z faktu nieujemności wielomianów Bernsteina oraz z poniższej zależności

$$\sum_{i=0}^{n} B_i^n(t) = 1$$



Rysunek 2.2: Przykładowa krzywa Beziera

Interpretacja geometryczna tej własności została przedstawiona na rysunku 2.3.

- liniowa precyzja. Tożsamość

$$\sum_{j=0}^{n} \frac{j}{n} B_j^n(t) = t$$

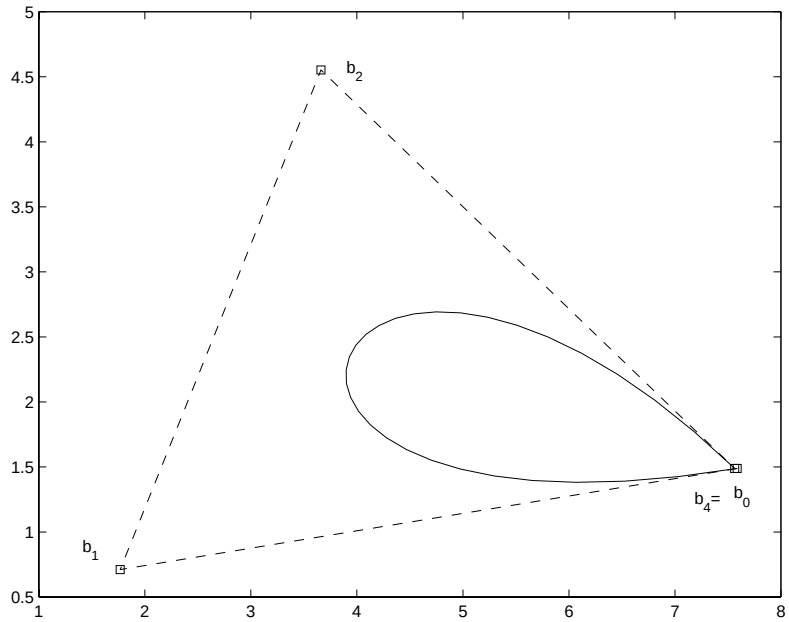
posiada następujące zastosowania: jeżeli wierzchołki wielokąta kontrolnego $\mathbf{p}_i$ są równomiernie rozmieszczone na linii prostej pomiędzy punktami $\mathbf{q}$ i $\mathbf{r}$ to wówczas zachodzi równość

$$\mathbf{p}_j = \left(1 - \frac{j}{n}\right)\mathbf{q} + \frac{j}{n}\mathbf{r} \qquad j = 0, .., n$$

i krzywa wygenerowana dla takiego wielokąta jest odcinkiem pomiędzy punktami $\mathbf{q}$ i $\mathbf{r}$.

- różniczkowanie. Pochodną krzywej Beziera stopnia n-tego obliczamy z następującego równania

$$\frac{d}{dt}\mathbf{p}^n(t) = n \sum_{j=0}^{n-1} \Delta\mathbf{p}_j B_j^{n-1}(t) \qquad \Delta\mathbf{p}_j = \mathbf{p}_{j+1} - \mathbf{p}_j$$



Rysunek 2.3: Własność wielokąta wypukłego

- całkowanie. Całkę z wielomianu Bernsteina stopnia n-tego możemy łatwo obliczyć z

$$\int_0^t B_i^n(u)du = \frac{1}{n+1}\sum_{j=i+1}^{n+1} B_j^{n+1}(t)$$

- przejście między bazą potęgową i bazą Bernsteina.

$$B_i^n(t) = \sum_{j=i}^{n}(-1)^{i+j}\binom{n}{j}\binom{j}{i}t^i$$

- iloczyn wielomianów Bernsteina.

$$B_i^n(t)B_j^m(t) = \frac{\binom{n}{i}\binom{m}{j}}{\binom{n+m}{i+j}}B_{i+j}^{n+m}(t)$$

Dodatkowo okazuje się, że wielomiany Bernsteina spełniają następującą zależność rekurencyjną

$$B_i^n(t) = (1-t)B_i^{n-1}(t) + tB_{i-1}^{n-1}(t)$$

co łatwo sprawdźić

$$
\begin{aligned}
(1 - t)B_i^{n-1}(t) + tB_{i-1}^{n-1}(t) = \\
(1-t)\binom{n-1}{i} t^i (1-t)^{n-1-i} + t\binom{n-1}{i-1} t^{i-1}(1-t)^{n-i} = \\
\left\{\binom{n-1}{i} + \binom{n-1}{i-1}\right\} t^i (1-t)^{n-i} = \binom{n}{i} t^i (1-t)^{n-i} = B_i^n(t)
\end{aligned}
$$

Powyższa zależność prowadzi do **algorytmu de Casteljau**. Algorytm ten to jedna z najczęściej spotykanych definicji krzywej Beziera. Ma on jednak większe znaczenie teoretyczne niż praktyczne, ponieważ do obliczania punktów krzywej Beziera wydajniejszy jest zmodyfikowany algorytm Hornera. Koszt algorytmu Hornera w bazie wielomianów Bernsteina wynosi $O(n)$, natomiast dla algorytmu de Casteljau wynosi on $O(n^2)$. Przez pojęcie kosztu danego algorytmu rozumiemy tutaj czas wykonywania działań arytmetycznych przewidzianych w ramach tego algorytmu. Szczegółowo całe zagadnienie omówiono w pracy [8].

2.2.2 Algorytm de Casteljau

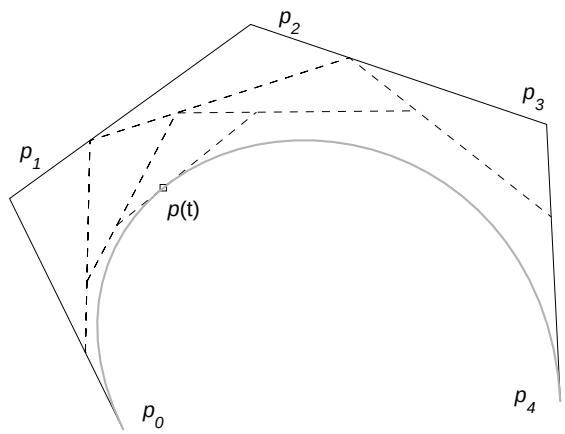
Algorytm de Casteljau przebiega następująco:

1. Wybieramy $n+1$ punktów, które oznaczymy $\mathbf{p}_0, \ldots, \mathbf{p}_n$.
2. Tworzymy łamaną, której kolejnymi wierzchołkami są te punkty.
3. Dokonujemy podziału wszystkich n odcinków tej łamanej w pewnej ustalonej proporcji $t : 1-t$.
4. W wyniku tego działania otrzymujemy n punktów, które uznajemy za wierzchołki kolejnej łamanej, złożonej z $n-1$ odcinków.
5. Proces powtarzamy aż do chwili, gdy otrzymamy jeden punkt.

Można pokazać, że powstałe w wyniku tego algorytmu punkty tworzą krzywą Beziera. I podobnie jak poprzednio zbiór punktów $\mathbf{p}_0, \ldots, \mathbf{p}_n$ nazywamy punktami kontrolnymi a łamana przechodząca przez te punkty to łamana kontrolna. Interpretację graficzną algorytmu de Casteljau przedstawia rys. 2.4.

2.2.3 Podwyższanie stopnia krzywej Beziera

Kolejnym zagadnieniem dotyczącym krzywych Beziera jest możliwość podwyższenia stopnia krzywej. Zabieg taki może być przydatny dla uzyskania większej swobody modyfikowania kształtu krzywej (poprzez większą liczbę punktów kontrolnych).



Rysunek 2.4: Algorytm de Casteljau

Matematycznie podwyższenie stopnia zapiszemy jako

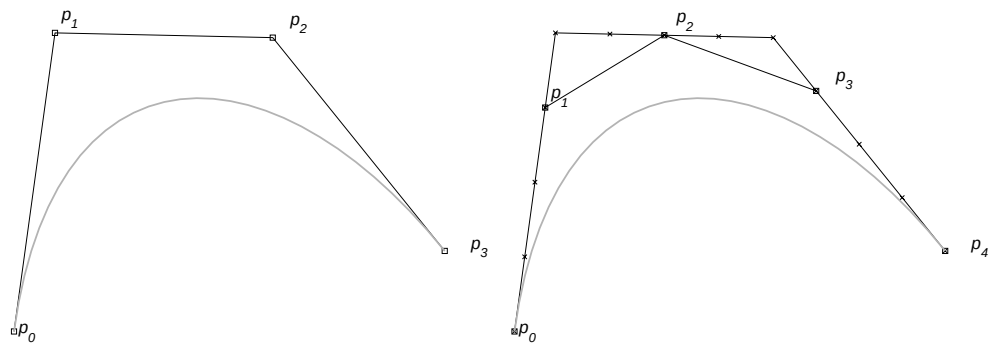
$$\begin{aligned}
\mathbf{p}(t) &= \sum_{i=0}^{n} \mathbf{p}_i B_i^n(t) = t\sum_{i=0}^{n} \mathbf{p}_i B_i^n(t) + (1-t)\sum_{i=0}^{n} \mathbf{p}_i B_i^n(t) = \\
&= \mathbf{p}_0 B_0^{n+1}(t) + \sum_{i=1}^{n} \left(\mathbf{p}_i \frac{n+1-i}{n+1} B_i^{n+1}(t) + \mathbf{p}_{i-1} \frac{i}{n+1} B_i^{n+1}(t) \right) + \\
&+ \mathbf{p}_n B_{n+1}^{n+1}(t) = \sum_{i=0}^{n+1} \widehat{\mathbf{p}}_i B_i^{n+1}(t)
\end{aligned}$$

otrzymujemy w ten sposób nowe punkty

$$\widehat{\mathbf{p}}_i = \frac{n+1-i}{n+1}\mathbf{p}_i + \frac{i}{n+1}\mathbf{p}_{i-1}$$

W wyrażeniach dla $\widehat{\mathbf{p}}_0$ i $\widehat{\mathbf{p}}_{n+1}$ nieokreślone punkty $\mathbf{p}_{-1}, \mathbf{p}_{n+1}$ są mnożone przez 0, zatem otrzymany wzór jest poprawnie określony. W taki sposób otrzymaliśmy punkty kontrolne krzywej wyjściowej odpowiadające bazie wielomianowej Bernsteina stopnia $n+1$.

Wynika stąd, iż istnieje nieskończenie wiele reprezentacji Beziera dowolnej krzywej wielomianowej. Z tego powodu określenie *stopień krzywej* ma dwa różne znaczenia. Pierwszym jest **stopień reprezentacji**, czyli stopień wielomianów, w bazie któych krzywa jest reprezentowana. Drugie znaczenie to najniższy stopień reprezentacji, której można użyć do przedstawienia krzywej. Oczywiste jest, że ze względów praktycznych pożądane jest osiągnięcia jak najniższego stopnia. Przykład podwyższenia dla krzywej Beziera 4-stopnia przedstawiono na rys. 2.5.



Rysunek 2.5: Podwyższenie stopnia dla krzywej Beziera 4-stopnia, po prawej stronie widzimy większą liczbę punktów kontrolnych.

2.2.4 Łączenie segmentów krzywych Beziera

Wektory pochodnych do rzędu k w punkcie początkowym lub końcowym krzywej są określone przez $k+1$ pierwszych albo ostatnich punktów kontrolnych.

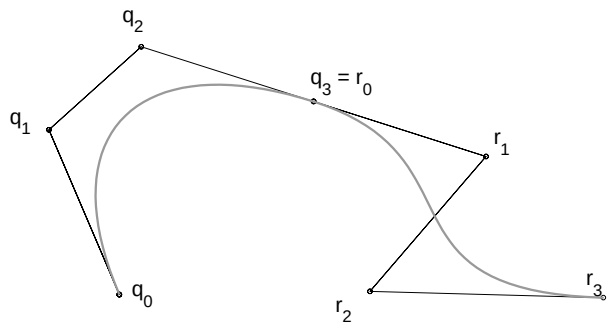
$$\begin{aligned}\frac{d^k}{dt^k}\mathbf{p}(t) &= \sum_{j=0}^{n-k}\frac{n!}{(n-k)!}\Delta^k\mathbf{p}_j B_j^{n-k}(t) = \frac{n!}{(n-k)!}\Delta^k(\sum_{j=0}^{n-k}\mathbf{p}_{j+1}B_j^{n-k}(t)) = \\ &\frac{n!}{(n-k)!}\Delta^k\mathbf{p}_0^{(n-k)}\end{aligned}$$

gdzie

$$\begin{aligned}\Delta^0\mathbf{p}_i &= \mathbf{p}_i \\ \Delta^k\mathbf{p}_i &= \Delta^{k-1}\mathbf{p}_{i+1} - \Delta^{k-1}\mathbf{p}_i\end{aligned}$$

Możemy stąd uzyskać warunki ciągłości pochodnych rzędu $1,\ldots,k$ krzywej złożonej z dwóch połączonych krzywych Beziera stopnia $n \geqslant k$ co oznaczymy symbolem C^k. Rozważmy krzywą $\mathbf{p}$ stopnia n i algorytm de Casteljau w celu podziału tej krzywej na dwa łuki $\mathbf{q}$ i $\mathbf{r}$. Przemieszczania punktów $\mathbf{q}_0,\ldots,\mathbf{q}_{n-k-1}$ oraz $\mathbf{r}_{k+1},\ldots,\mathbf{r}_n$ nie zmienia pochodnych rzędu $1,\ldots,k$ we wspólnym punkcie. Po odrzuceniu tych punktów zostają dwie łamane kontrolne powstałe z podziału krzywej stopnia k, a wyznaczonej przez punkty kontrolne $\mathbf{p}_0^{(n-k)},\ldots,\mathbf{p}_k^{(n-k)}$, otrzymane w $n-k$-tej iteracji algorytmu de Casteljau.

Widzimy zatem, że jeśli dwie krzywe $\mathbf{q}$ i $\mathbf{r}$ są połączone z ciągłością C^k, to punkty pośrednie w algorytmie de Casteljau, stanowiące dane dla ostatnich $k+1$ iteracji, powinny być identyczne, niezależnie od tego na podstawie której krzywej je odtwarzamy.



Rysunek 2.6: Krzywe Beziera połączone z zachowaniem ciągłości klasy C^1

Rozważmy teraz, najchętniej używane w projektowaniu geometrycznym, kubiczne krzywe Beziera. Weźmy dwie krzywe $\mathbf{q}$ i $\mathbf{r}$ określone na przedziale $[0, 1]$ i określmy warunki ciągłości kolejno C^0, C^1 oraz C^2.

1. Pierwsze równanie jest chyba oczywiste i nie wymaga żadnych komentarzy. Ciągłość krzywej powstałej z połączenia $\mathbf{q}$ i $\mathbf{r}$ uzyskamy gdy

$$\mathbf{q}_3 = \mathbf{r}_0$$

2. Drugie równanie również nie stanowi większego problemu, zatem obliczając pochodne $\mathbf{q}(1)$ oraz $\mathbf{r}(0)$ otrzymujemy

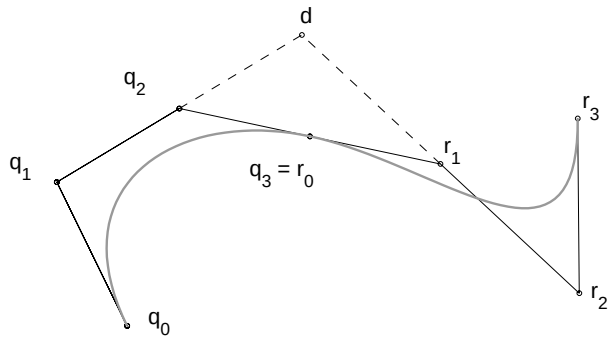
$$\mathbf{q}_3 - \mathbf{q}_2 = \mathbf{r}_1 - \mathbf{r}_0$$

   Co pokazano na rysunku 2.6.

3. I wreszcie warunek ciągłości krzywizny obliczymy przyrównując drugie pochodne krzywych $\mathbf{q}$ i $\mathbf{r}$, który ostatecznie przyjmie następującą postać

$$2\mathbf{q}_2 - \mathbf{q}_1 = 2\mathbf{r}_1 - \mathbf{r}_2$$

   Przykładowe krzywe Beziera spełniające ten warunek przedstawiono na rysunku 2.7. W celu zapewnienia ciągłości klasy C^2 należy oczywiście spełnić również warunki dla ciągłości C^0 oraz C^1.



Rysunek 2.7: Krzywe Beziera połączone z zachowaniem ciągłości klasy C^2

2.2.5 Powierzchnie Beziera

Rozróżniamy dwa rodzaje powierzchni Beziera: prostokątną oraz trójkątną. Prostokątna powierzchnia Beziera jest zdefiniowana poprzez macierz punktów kontrolnych (rys. 2.8).

$$\mathbf{p}_{ij} = \begin{pmatrix} \mathbf{p}_{00} & \mathbf{p}_{01} & \mathbf{p}_{02} & \mathbf{p}_{03} \\ \mathbf{p}_{10} & \mathbf{p}_{11} & \mathbf{p}_{12} & \mathbf{p}_{13} \\ \mathbf{p}_{20} & \mathbf{p}_{21} & \mathbf{p}_{22} & \mathbf{p}_{23} \\ \mathbf{p}_{30} & \mathbf{p}_{31} & \mathbf{p}_{32} & \mathbf{p}_{33} \end{pmatrix}$$

Przykładową powierzchnię Beziera przedstawiono na rysunku 2.9. Dowolny punkt leżący na powierzchni wyznaczymy z następującej zależności

$$\mathbf{p}^{mn}(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} \mathbf{p}_{ij} B_i^m(u) B_j^n(v) \qquad 0 \leqslant u, v \leqslant 1$$

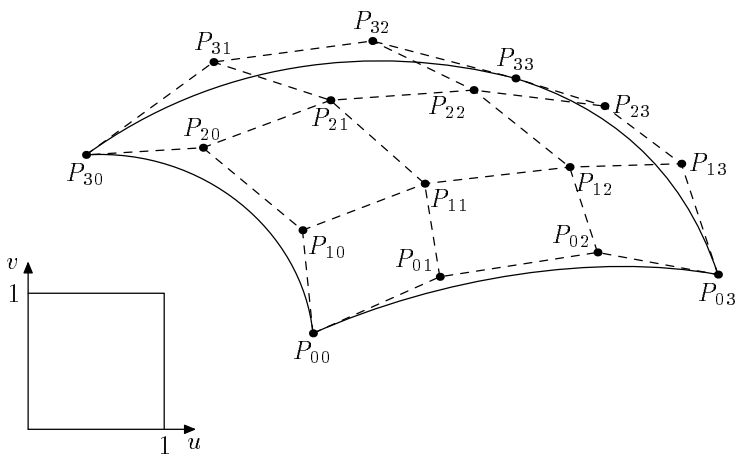
gdzie

$\mathbf{p}_{ij}$- macierz punktów kontrolnych
$B_i^m(u)\, B_j^n(v)$- wielomiany Bernsteina

Powierzchnia otrzymana dla $m = n = 3$ często nazywana jest bikubiczną powierzchnią Beziera.

Trójkątna powierzchnia Beziera jest określona przez następującą tablicę punktów kontrolnych ($n = 4$)

$$\begin{array}{ccccc} & & \mathbf{p}_{040} & & \\ & \mathbf{p}_{031} & & \mathbf{p}_{130} & \\ \mathbf{p}_{022} & & \mathbf{p}_{121} & & \mathbf{p}_{220} \\ \mathbf{p}_{013} & \mathbf{p}_{112} & & \mathbf{p}_{211} & \mathbf{p}_{310} \\ \mathbf{p}_{004} & \mathbf{p}_{103} & \mathbf{p}_{202} & \mathbf{p}_{301} & \mathbf{p}_{400} \end{array}$$



Rysunek 2.8: Znaczenie tablicy punktów kontrolnych

Punkt $\mathbf{p}(\mathbf{u})$ leżący na trójkątnym płacie Beziera wyliczamy z poniższej zależności:

$$\mathbf{p}^n(\mathbf{u}) = \sum_{|\mathbf{j}|=n} \mathbf{p}_j B_j^n(\mathbf{u})$$

gdzie

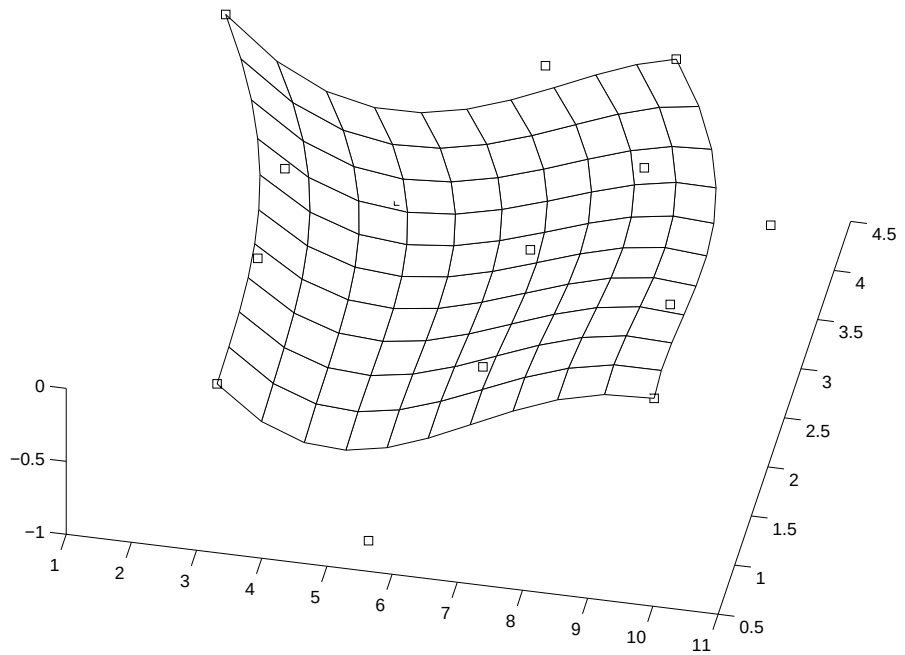
$$B_j^n(\mathbf{u}) = \frac{n!}{i!j!k!} u^i v^j w^k$$
$$\mathbf{u} = (u, v, w)$$
$$|\mathbf{j}| = i + j + k \qquad i, j, k \geqslant 0$$

u, v, w są to współrzędne barycentryczne, sposród których jedna współrzędna jest zależna od pozostałych. Współrzędne powyższe spełniają następującą zależność $u + v + w = 1$.

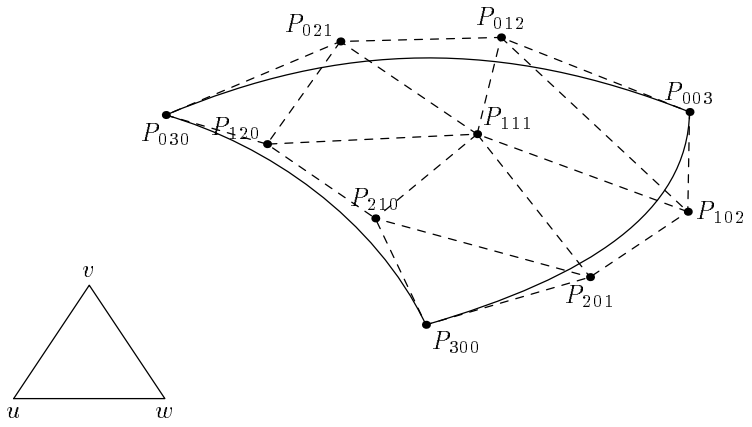
Dla najchętniej stosowanego przypadku $n = 3$ (rys. 2.11) uzyskamy wzór na trójkątną powierzchnię Beziera postaci

$$\begin{aligned} \mathbf{p}^3(\mathbf{u}) &= \mathbf{p}_{030}v^3 + 3\mathbf{p}_{120}uv^2 + 3\mathbf{p}_{021}v^2w + 3\mathbf{p}_{210}u^2v + 6\mathbf{p}_{111}uvw + \\ &+ 3\mathbf{p}_{012}vw^2 + \mathbf{p}_{300}u^3 + 3\mathbf{p}_{201}u^2w + 3\mathbf{p}_{102}uw^2 + \mathbf{p}_{003}w^3 \end{aligned}$$

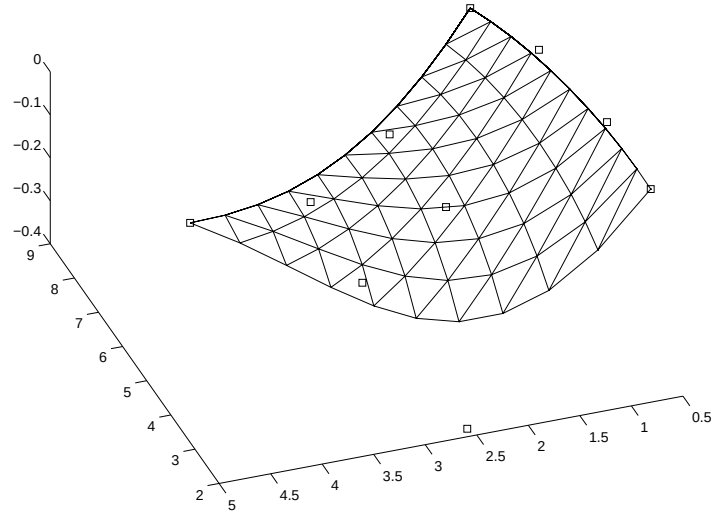
Więcej informacji dotyczących krzywych i powierzchni Beziera wraz z omówieniem pojęć ciągłości parametrycznej i geometrycznej można znaleźć w pracy [3].



Rysunek 2.9: Powierzchnia Beziera i opisujące ją punkty kontrolne



Rysunek 2.10: Znaczenie punktów kontrolnych dla powierzchni trójkątnej



Rysunek 2.11: Trójkątna powierzchnia Beziera

2.3 Krzywe i powierzchnie NURBS

2.3.1 Krzywe NURBS

Określenie NURBS pochodzi od angielskiego zwrotu "NonUniform Rational B-Splines" (niejednorodne ułamkowe krzywe sklejane) a nie oznacza ono nic więcej jak tylko najbardziej ogólne sformułowanie znanych w matematyce krzywych sklejanych. Ten rodzaj krzywych został wprowadzony do grafiki komputerowej w celu uzyskania ciągłości wyższej klasy niż C^1, ponieważ łączenie krzywych Beziera tak, aby spełniały ten warunek jest wysoce nieefektywne. Krzywa B-Spline zawiera kilka segmentów wielomianowych połączonych z zachowaniem określonej gładkości. Typowa krzywa sklejana stopnia n posiada gładkość C^{n-1}. W przypadku krzywych NURBS wygodniej czasami zamiast stopniem krzywej operować terminem rząd krzywej. Rząd krzywej określa liczba współczynników liczbowych w równaniu krzywej, natomiast stopień to jak wiadomo wartość największej potęgi funkcji bazowej użytej do opisu krzywej.

Uwzględniając powyższe krzywą NURBS rzędu k definiujemy jako

$$\mathbf{C}(t) = \frac{\sum_{i=0}^{n} W_i \mathbf{p}_i N_i^k(t)}{\sum_{i=0}^{n} W_i N_i^k(t)}$$

gdzie

$\mathbf{p}_i, i = 0, ..., n$	- punkty kontrolne
W_i	- wagi przypisane dla wszystkich punktów kontrolnych
$N_i^k(t)$	- znormalizowana funkcja bazowa

Funkcja bazowa rzędu k określona jest poprzez wektor węzłowy $T = T_i$, $i = 0, ..., n + k$ oraz poprzez rekurencyjną zależność

$$N_i^k(t) = \frac{(t - T_i)N_i^{k-1}(t)}{T_{i+k-1} - T_i} + \frac{(T_{i+k} - t)N_{i+1}^{k-1}(t)}{T_{i+k} - T_{i+1}}$$

$$N_i^1(t) = \begin{cases} 1 & \text{jeżeli } T_i \leqslant t < T_{i+1} \\ 0 & \text{w przeciwnym razie} \end{cases}$$

Zwykle wektor węzłowy zakłada się postaci

$$T = \{0, ..., 0, T_k, ..., T_n, 1, ..., 1\}$$

z wielokrotnością k dla wartości węzłów na obu końcach wektora.

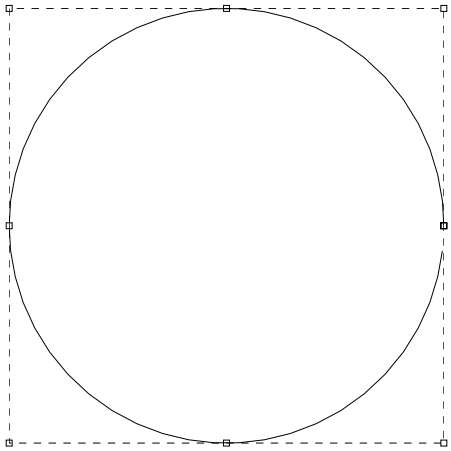
Zwiększenie wielokrotności węzła wewnętrznego wpływa na redukcję ciągłości parametrycznej sąsiednich segmentów. I tak dla krzywej rzędu czwartego (trzeciego stopnia) wielokrotność 2 dla węzła powoduje redukcję ciągłości z C^2 na C^1, wielokrotność 3 to redukcja ciągłości z C^1 na C^0, natomiast dla trzech dodatkowych węzłów (wielokrotność 4) uzyskujemy brak ciągłości między segmentami krzywej.

Oprócz łatwości w dobieraniu gładkości krzywe ułamkowe posiadają również dwie inne zalety, a mianowicie:

- są niezmiennicze względem przekształceń obrotu, skalowania, przesunięcia i perspektywy (krzywe nieułamkowe są niezmiennicze tylko względem obrotu, skalowania i przesunięcia). Oznacza to, że przekształcenie perspektywiczne obliczamy tylko dla punktów kontrolnych a następnie za ich pomocą generujemy ostateczną krzywą

- w przeciwieństwie do krzywych nieułamkowych dokładnie definiują krzywe stożkowe, co jest istotne zwłaszcza przy modelowaniu CAD.

W pracy [11] można znaleźć przykłady krzywych stożkowych utworzonych z wykorzystaniem techniki NURBS.



Rysunek 2.12: Okrąg zamodelowany za pomocą czterech segmentów NURBS, jako funkcje bazowe użyto wielomianów 2-ego stopnia (kwadratami wyróżniono punkty kontrolne)

2.3.2 Powierzchnie NURBS

Powierzchnia NURBS jest zdefiniowana jako iloczyn tensorowy rozszerzający reprezentację krzywej NURBS do dwóch kierunków, który sformułowano jako

$$\mathbf{S}(s,t) = \frac{\sum_{i=0}^{m}\sum_{j=0}^{n} W_{ij}\mathbf{p}_{ij}N_i^{k1}(s)N_j^{k2}(t)}{\sum_{i=0}^{m}\sum_{j=0}^{n} W_{ij}N_i^{k1}(s)N_j^{k2}(t)}$$

gdzie

$\mathbf{p}_{ij}, i = 0,\ldots,m \quad j = 0,\ldots,n$	- punkty kontrolne
W_{ij}	- wagi przypisane punktom kontrolnym
$N_i^{k1}(s)N_j^{k2}(t)$	- funkcje bazowe odpowiednio rzędu $k1$ i $k2$

Wektory węzłowe dla powierzchni NURBS to

$$T_1 = T_i, i = 0, ..., m + k1$$
$$T_2 = T_i, i = 0, ..., n + k2$$

Definicja funkcji bazowych ma postać jak dla krzywych NURBS.

2.3.3 Produkt sferyczny

Pewnym szczególnym, ale godnym przytoczenia, przypadkiem powierzchni NURBS jest powierzchnia określona za pomocą dwóch krzywych płaskich:

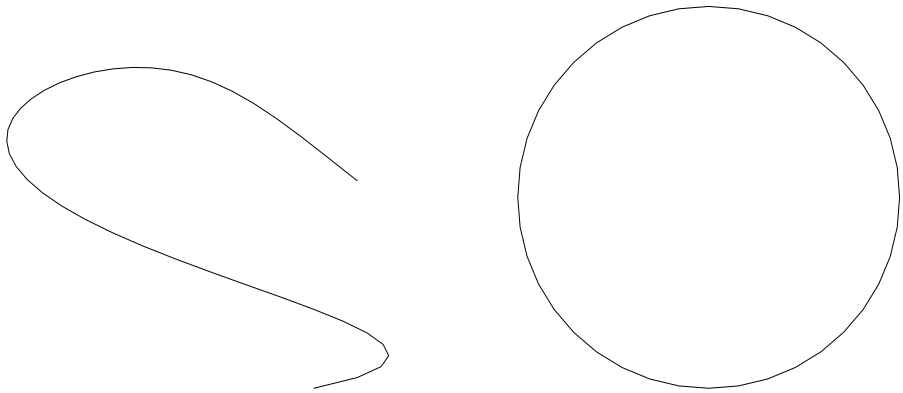
- równoleżnika $\mathbf{r}(u) = [x_r(u), y_r(u)]^T$
- południka $\mathbf{q}(v) = [x_q(v), y_q(v)]^T$

Wzór określający taką powierzchnię zapiszemy jako

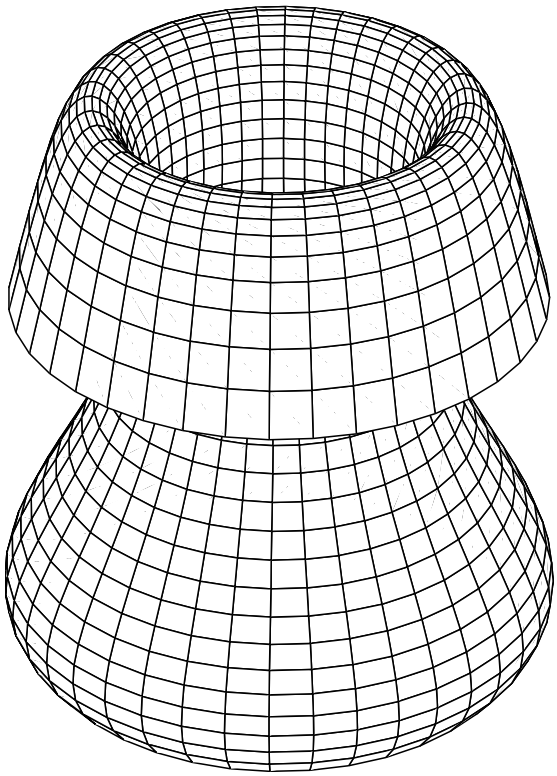
$$\mathbf{s}(u, v) = [x_r(u)x_q(v), y_r(u)x_q(v), y_q(v)]^T$$

Nazwa produkt sferyczny pochodzi stąd, że jeśli równoleżnik jest okręgiem, a południk półokręgiem, oraz dodatkowo środek obu krzywych leży w początku układu współrzędnych, to powierzchnią jaką otrzymamy z powyższego wzoru będzie sfera. Dodatkową zaletą produktu sferycznego jest fakt, iż jest on uogólnieniem wszystkich powierzchni obrotowych jeśli tylko równoleżnik jest okręgiem o środku w punkcie [0,0]. Na rysunkach 2.14 i 2.16 przedstawiono przykłady powierzchni produktu sferycznego dla różnych przypadków równoleżnika.

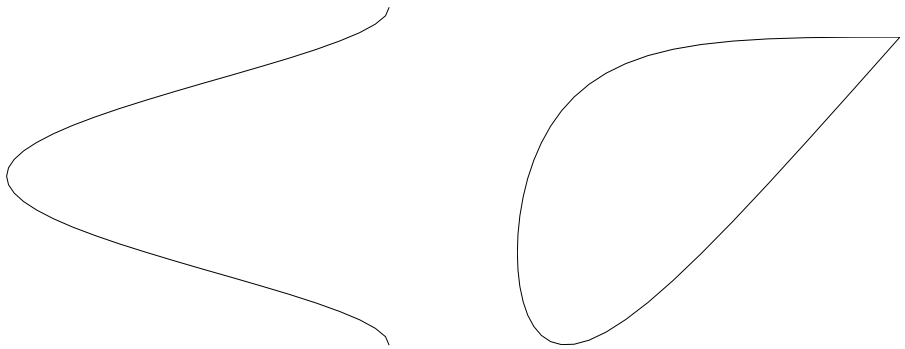
Zarówno powierzchnie Beziera jak i powierzchnie NURBS doczekały się swioch implementacji w wielu bibliotekach graficznych, warto choćby wspomnieć o OpenGL standardowej bibliotece firmy Silicon Graphics. W pracy [10] znaleźć można praktyczne wykorzystanie powierzchni NURBS do modelowania złożonych geomertycznie kształtów.



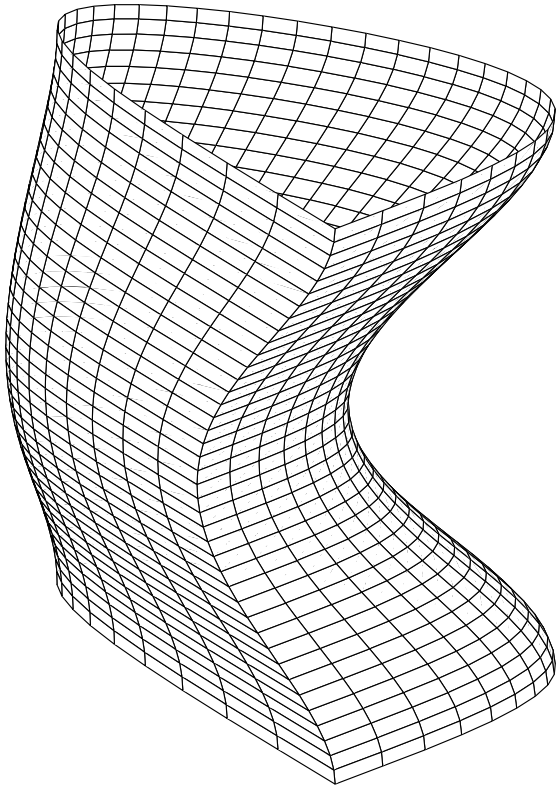
Rysunek 2.13: Krzywe określające południk(z lewej) i równoleżnik(z prawej) dla powierzchni obrotowej



Rysunek 2.14: Powierzchnia obrotowa dla powyższych krzywych



Rysunek 2.15: Krzywe określające południk(z lewej) i równoleżnik(z prawej) dla dowolnej powierzchni



Rysunek 2.16: Powierzchnia odpowiadająca powyższym krzywym

Rozdział 3

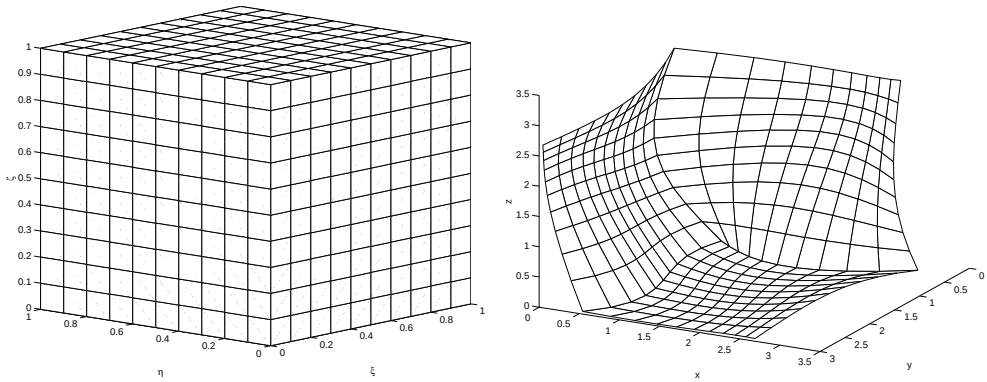
Algebraiczne systemy generacji siatek (TFI)

3.1 Wprowadzenie

Generatory siatek dostarczają pomocnego narzędzia do numerycznego rozwiązywania równań (różniczkowych lub całkowych) jakie rządzą wieloma zjawiskami występującymi w fizyce. Numeryczne rozwiązanie jest zazwyczaj rozwiązaniem przybliżonym w związku z czym obarczone jest pewnymi błedami. W większości prac poświęconych takim właśnie metodom analiza błędu ogranicza się do błędu samej metody, gdy tym czasem przy rozwiązywaniu określonego zagadnienia należy zwrócić uwagę również na jakość uzyskanej siatki, która może wpływać w istotny sposób na dokładność obliczeń. W tym rozdziale przedstawiono opis algebraicznych systemów generacji siatki w skrócie zwanych TFI (TransFinite Interpolation). Metoda ta, mimo iż może stanowić samodzielny algorytm generacji siatki, wykorzystywana jest głównie jako procedura startowa do generacji strukturalynch siatek bazujących na rozwiązywaniu równań różniczkowych cząstkowych (patrz rozdział 4). Korzyścią płynącą z używania tej metody jest możliwość wygenerowania siatek dopasowanych do wskazanych brzegów oraz bezpośrednia kontrola gęstości siatki. Ponadto TFI jest łatwe do zaprogramowania i bardzo efektywne jeśli chodzi o czas obliczeń.

3.2 Podstawy matematyczne

Jak wcześniej wspomniano algebraiczne techniki generacji siatki należą do najbardziej efektywnych a bazują one na wzorach interpolacyjnych. W technice zwanej TFI oprócz przestrzeni fizycznej (XYZ), w której znajduje się rozpatrywane zagadnienie, wprowadza się dodatkowo pewną abstrakcyjną przestrzeń obliczeniową



Rysunek 3.1: Układ współrzędnych obliczeniowych a układ współrzędnych fizycznych

(ξ, η, ζ). Matematycznie przestrzeń obliczeniową rozpartujemy na obszarze jednostkowego sześcianu (w przypadku 3-wymiarowym) oraz jednostkowego kwadratu (w przypadku 2-wymiarowym). Algebraiczna technika generacji siatki dokonuje następnie transformacji z prostokątnej przestrzeni obliczeniowej do dowolnie ukształtowanej przestrzeni fizycznej. Całe zagadnienie zilustrowano schematycznie na rysunku 3.1 oraz w postaci ogólnych wzorów przedstawiono poniżej

$$\mathbf{X}(\xi, \eta, \zeta) = \begin{bmatrix} x(\xi, \eta, \zeta) \\ y(\xi, \eta, \zeta) \\ z(\xi, \eta, \zeta) \end{bmatrix}$$

$$0 \leqslant \xi \leqslant 1 \qquad 0 \leqslant \eta \leqslant 1 \qquad 0 \leqslant \zeta \leqslant 1$$

Dyskretny podzbiór $\mathbf{X}(\xi_I, \eta_J, \zeta_K)$ jest strukturalną siatką dla

$$0 \leqslant \xi_I = \frac{I-1}{\widehat{I}-1} \leqslant 1 \qquad 0 \leqslant \eta_J = \frac{J-1}{\widehat{J}-1} \leqslant 1 \qquad 0 \leqslant \zeta_K = \frac{K-1}{\widehat{K}-1} \leqslant 1$$

gdzie

$$I = 1, 2, ..., \widehat{I} \qquad J = 1, 2, ..., \widehat{J} \qquad K = 1, 2, ..., \widehat{K}$$

Związki pomiędzy I, J oraz K a obliczeniowymi współrzędnymi (ξ, η, ζ) równomiernie zdyskretyzowanymi w przestrzeni obliczeniowej implikują relacje pomiędzy zagęszczeniem siatki w odpowiednich kierunkach. Następnie tak zdyskretyzowany zbiór punktów jest przekształcony do przestrzeni fizycznej a zadaniem TFI jest dostarczenie niezbędnego aparatu matematycznego do uzysaknia wartości $\mathbf{X}(\xi, \eta, \zeta)$.

3.3 Interpolacja algebraiczna (TFI)

Interpolację algebraiczną (Transfinite interpolation) jako pierwszy opisał w swojej pracy [5] William Gordon w roku 1973. Korzyścią przemawiającą za TFI była generacja siatki całkowicie dopasowanej do brzegów w przestrzeni fizycznej. Na początku lat 80-tych użyteczność tej metody do rozwiązywania zagadnień z mechaniki płynów wykazał Lars Eriksson. Do dnia dzisiejszego metoda interpolacji algebraicznej doczekała się wielu modyfikacji i jest powszechnie stosowana we wszystkich szanujących się aplikacjach metody elementu skończonego.

3.3.1 Sformułowanie metody przy użyciu sum logicznych

Istotą TFI jest wyznaczenie jednokierunkowej interpolacji, w każdym kierunku obliczeniowego układu współrzędnych, poprzez iloczyny tensorowe i ostatecznie sumy logiczne (Boolean sum). Funkcje jednokierunkowej interpolacji są kombinacją liniową znanych funkcji mieszanych oraz pochodnych z funkcji $\mathbf{X}(\xi,\eta,\zeta)$ obliczonej w zadanych wartościach współrzędnych obliczeniowych. Ogólne sformułowanie jednokierunkowej interpolacji w trzech wymiarach jest następujące

$$\mathbf{U}(\xi,\eta,\zeta) = \sum_{i=1}^{L}\sum_{n=0}^{P} \alpha_i^n(\xi)\frac{\partial^n \mathbf{X}(\xi_i,\eta,\zeta)}{\partial \xi^n}$$

$$\mathbf{V}(\xi,\eta,\zeta) = \sum_{j=1}^{M}\sum_{m=0}^{Q} \beta_j^m(\eta)\frac{\partial^m \mathbf{X}(\xi,\eta_j,\zeta)}{\partial \eta^m}$$

$$\mathbf{W}(\xi,\eta,\zeta) = \sum_{k=1}^{N}\sum_{l=0}^{R} \gamma_k^l(\zeta)\frac{\partial^l \mathbf{X}(\xi,\eta,\zeta_k)}{\partial \zeta^l}$$

Warunki jakie muszą spełniać funkcje mieszane

$$\frac{\partial^{\bar{n}} \alpha_{\bar{i}}^n(\xi_i)}{\partial \xi^{\bar{n}}} = \delta_{i\bar{i}}\delta_{n\bar{n}}$$

$$\frac{\partial^{\bar{m}} \beta_{\bar{j}}^m(\eta_j)}{\partial \eta^{\bar{m}}} = \delta_{j\bar{j}}\delta_{m\bar{m}}$$

$$\frac{\partial^{\bar{l}} \gamma_{\bar{k}}^l(\zeta_k)}{\partial \zeta^{\bar{l}}} = \delta_{k\bar{k}}\delta_{l\bar{l}}$$

$$\begin{array}{lll} \bar{i} = 1,2,...,L & \bar{j} = 1,2,...,M & \bar{k} = 1,2,...N \\ \bar{n} = 1,2,...,P & \bar{m} = 1,2,...,Q & \bar{l} = 1,2,...R \end{array}$$

Iloczyny tensorowy funkcji $\mathbf{U}(\xi,\eta,\zeta)$, $\mathbf{V}(\xi,\eta,\zeta)$ oraz $\mathbf{W}(\xi,\eta,\zeta)$ mają następującą postać

$$\mathbf{UW} = \mathbf{WU} = \sum_{i=1}^{L}\sum_{k=1}^{N}\sum_{l=0}^{R}\sum_{n=0}^{P} \alpha_i^n(\xi)\gamma_k^l(\zeta)\frac{\partial^{ln}\mathbf{X}(\xi_i,\eta,\zeta_k)}{\partial\zeta^l\partial\xi^n}$$

$$\mathbf{UV} = \mathbf{VU} = \sum_{i=1}^{L}\sum_{j=1}^{M}\sum_{m=0}^{Q}\sum_{n=0}^{P} \alpha_i^n(\xi)\beta_j^m(\eta)\frac{\partial^{nm}\mathbf{X}(\xi_i,\eta_j,\zeta)}{\partial\eta^m\partial\xi^n}$$

$$\mathbf{VW} = \mathbf{WV} = \sum_{j=1}^{M}\sum_{k=1}^{N}\sum_{m=0}^{Q}\sum_{l=0}^{R} \beta_j^m(\eta)\gamma_k^l(\zeta)\frac{\partial^{lm}\mathbf{X}(\xi,\eta_j,\zeta_k)}{\partial\zeta^l\partial\eta^m}$$

$$\mathbf{UVW} = \sum_{i=1}^{L}\sum_{j=1}^{M}\sum_{k=1}^{N}\sum_{l=0}^{R}\sum_{m=0}^{Q}\sum_{n=0}^{P} \alpha_i^n(\xi)\beta_j^m(\eta)\gamma_k^l(\zeta)\frac{\partial^{lnm}\mathbf{X}(\xi_i,\eta_j,\zeta_k)}{\partial\zeta^l\partial\eta^m\partial\xi^n}$$

Ostatecznie sumą Boole'a trzech interpolacji jest

$$\mathbf{X}(\xi,\eta,\zeta) = \mathbf{U}\oplus\mathbf{V}\oplus\mathbf{W} = \mathbf{U}+\mathbf{V}+\mathbf{W}-\mathbf{UV}-\mathbf{UW}-\mathbf{VW}+\mathbf{UVW}$$

3.3.2 Sformułowanie rekurencyjne

Alternatywą dla sformułowania TFI poprzez użycie sum Boole'a może być trójkrokowy algorytm rekurencyjny. Pierwszy krok wyraża jednorodną interpolację w jednym kierunku

$$\mathbf{X}_1(\xi,\eta,\zeta) = \sum_{i=1}^{L}\sum_{n=0}^{P} \alpha_i^n(\xi)\frac{\partial^n\mathbf{X}(\xi_i,\eta,\zeta)}{\partial\xi^n}$$

Drugi i trzeci krok są następujące

$$\mathbf{X}_2(\xi,\eta,\zeta) = \mathbf{X}_1(\xi,\eta,\zeta) + \sum_{j=1}^{M}\sum_{m=0}^{Q} \beta_j^m(\eta)\left[\frac{\partial^m\mathbf{X}(\xi,\eta_j,\zeta)}{\partial\eta^m} - \frac{\partial^m\mathbf{X}_1(\xi,\eta_j,\zeta)}{\partial\eta^m}\right]$$

$$\mathbf{X}(\xi,\eta,\zeta) = \mathbf{X}_2(\xi,\eta,\zeta) + \sum_{k=1}^{N}\sum_{l=0}^{R} \gamma_k^l(\zeta)\left[\frac{\partial^l\mathbf{X}(\xi,\eta,\zeta_k)}{\partial\zeta^l} - \frac{\partial^l\mathbf{X}_2(\xi,\eta,\zeta_k)}{\partial\zeta^l}\right]$$

3.4 Praktyczne zastosowania TFI

3.4.1 Interpolacja liniowa

Najprostszym przykładem TFI jest użycie do interpolacji funkcji liniowych oraz określenie współrzędnych zbioru punktów brzegowych na sześciu powierzchniach

ograniczających (w zadaniu 3D) lub czterech krzywych (w przypadku 2D). Współczynniki P, Q, R, L, M, N występujące przy znakach sumy wynoszą

$$P = Q = R = 0 \qquad L = M = N = 2$$

Liniowe funkcje spełniające warunki stawiane funkcjom mieszanym mają postać

$$\begin{aligned}
\alpha_1^0(\xi) &= 1 - \xi \\
\alpha_2^0(\xi) &= \xi \\
\beta_1^0(\eta) &= 1 - \eta \\
\beta_2^0(\eta) &= \eta \\
\gamma_1^0(\zeta) &= 1 - \zeta \\
\gamma_2^0(\zeta) &= \zeta
\end{aligned}$$

Uwzględniając powyższe otrzymujemy następujące wzory na funkcje $\mathbf{U}, \mathbf{V}, \mathbf{W}$ oraz ich iloczyny tensorowe

$$\begin{aligned}
\mathbf{U}(\xi_I, \eta_J, \zeta_K) &= (1 - \xi_I)\mathbf{X}(0, \eta_J, \zeta_K) + \xi_I \mathbf{X}(1, \eta_J, \zeta_K) \\
\mathbf{V}(\xi_I, \eta_J, \zeta_K) &= (1 - \eta_J)\mathbf{X}(\xi_I, 0, \zeta_K) + \eta_J \mathbf{X}(\xi_i, 1, \zeta_K) \\
\mathbf{W}(\xi_I, \eta_J, \zeta_K) &= (1 - \zeta_K)\mathbf{X}(\xi_I, \eta_J, 0) + \zeta_K \mathbf{X}(\xi_i, \eta_J, 1) \\
\mathbf{UW}(\xi_I, \eta_J, \zeta_K) &= (1 - \xi_I)(1 - \zeta_K)\mathbf{X}(0, \eta_J, 0) + (1 - \xi_I)\zeta_K \mathbf{X}(0, \eta_J, 1) + \\
&\quad + \xi_I(1 - \zeta_J)\mathbf{X}(1, \eta_J, 0) + \xi_I \zeta_K \mathbf{X}(1, \eta_J, 1) \\
\mathbf{UV}(\xi_I, \eta_J, \zeta_K) &= (1 - \xi_I)(1 - \eta_J)\mathbf{X}(0, 0, \zeta_K) + (1 - \xi_I)\eta_J \mathbf{X}(0, 1, \zeta_K) + \\
&\quad + \xi_I(1 - \eta_J)\mathbf{X}(1, 0, \zeta_K) + \xi_I \eta_J \mathbf{X}(1, 1, \zeta_K) \\
\mathbf{VW}(\xi_I, \eta_J, \zeta_K) &= (1 - \eta_J)(1 - \zeta_K)\mathbf{X}(\xi_I, 0, 0) + (1 - \eta_J)\zeta_K \mathbf{X}(\xi_I, 1, 0) + \\
&\quad + \eta_J(1 - \zeta_K)\mathbf{X}(\xi_I, 0, 1) + \eta_J \zeta_K \mathbf{X}(\xi_I, 1, 1) \\
\mathbf{UVW}(\xi_I, \eta_J, \zeta_K) &= (1 - \xi_I)(1 - \eta_J)(1 - \zeta_K)\mathbf{X}(0, 0, 0) + (1 - \xi_I)(1 - \eta_J)\zeta_K \mathbf{X}(0, 0, 1) + \\
&\quad + (1 - \xi_I)\eta_J(1 - \zeta_K)\mathbf{X}(0, 1, 0) + (1 - \xi)\eta_J \zeta_K \mathbf{X}(0, 1, 1) + \\
&\quad + \xi_I(1 - \eta_J)(1 - \zeta_K)\mathbf{X}(1, 0, 0) + \xi_I(1 - \eta_J)\zeta_K \mathbf{X}(1, 0, 1) + \\
&\quad + \xi_I \eta_J(1 - \zeta_K)\mathbf{X}(1, 1, 0) + \xi_I \eta_J \zeta_K \mathbf{X}(1, 1, 1)
\end{aligned}$$

gdzie

$$I = 1, 2, ..., \widehat{I} \qquad J = 1, 2, ..., \widehat{J} \qquad K = 1, 2, ..., \widehat{K}$$

Ostatecznie współrzędne węzłów siatki wyznaczamy z

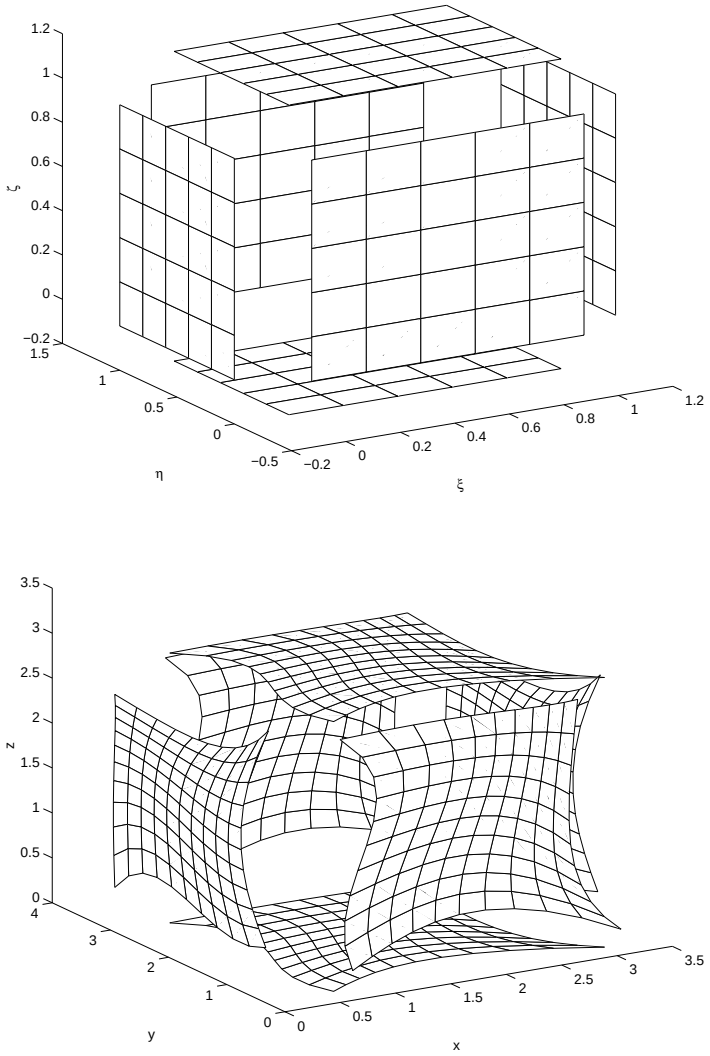
$$\begin{aligned}
\mathbf{X}(\xi_I, \eta_J, \zeta_K) &= \mathbf{U}(\xi_I, \eta_J, \zeta_K) + \mathbf{V}(\xi_I, \eta_J, \zeta_K) + \mathbf{W}(\xi_I, \eta_J, \zeta_K) - \mathbf{UW}(\xi_I, \eta_J, \zeta_K) \\
&\quad - \mathbf{UV}(\xi_I, \eta_J, \zeta_K) - \mathbf{VW}(\xi_I, \eta_J, \zeta_K) + \mathbf{UVW}(\xi_I, \eta_J, \zeta_K)
\end{aligned}$$

Powyższy wzór, pomimo swojej złożonej struktury, posiada bardzo prostą interpretację geometryczną. Dla przypadku 3D interpretację taką przedstawiono na

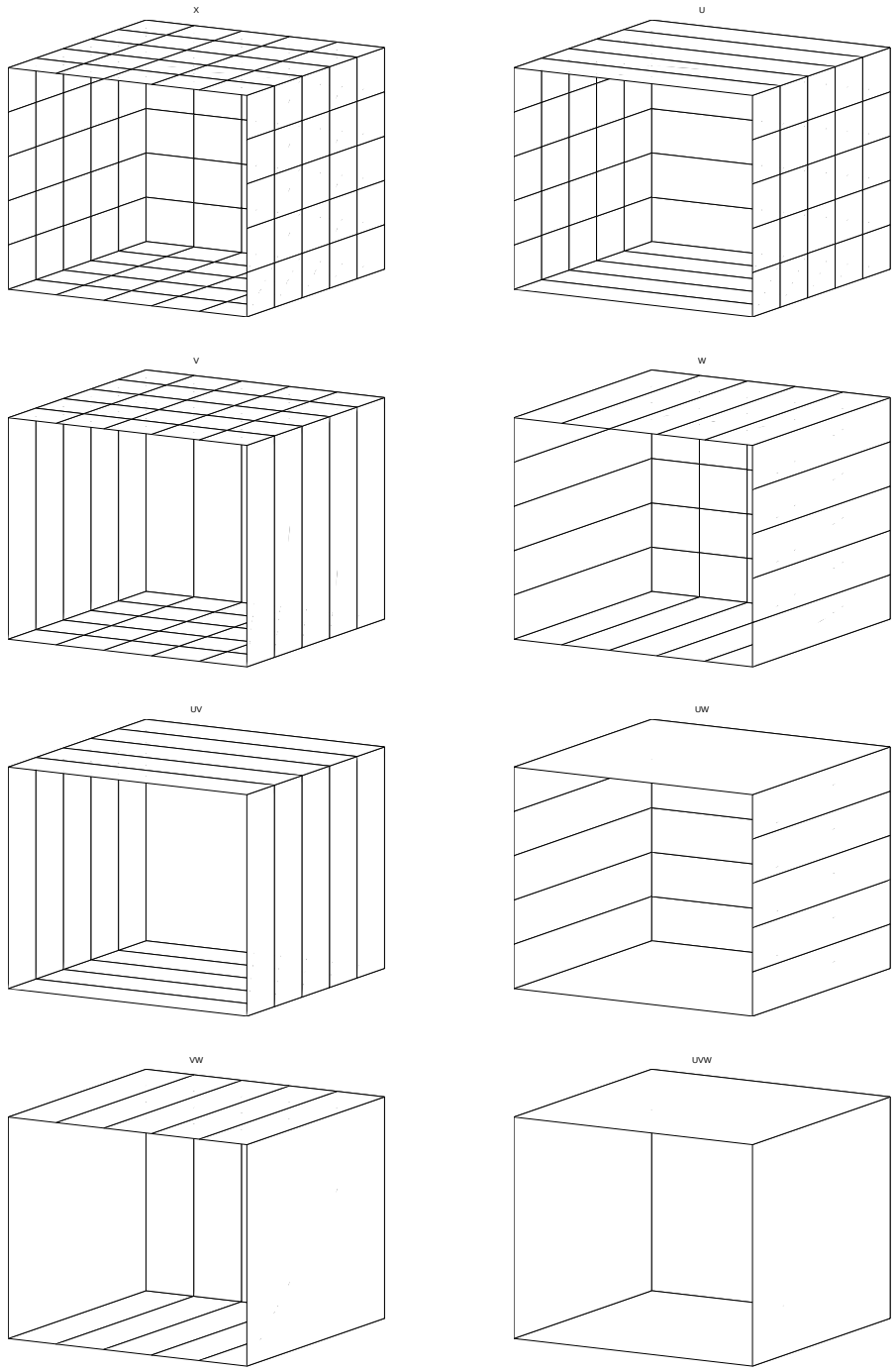
rysunku 3.3. Rozpatrując zadanie płaskie obserwujemy dalsze uproszczenie w wyniku którego macierze $\mathbf{W}, \mathbf{UW}, \mathbf{VW}$ oraz $\mathbf{UVW}$ redukują się do zera. Graficzny obraz tego zjawiska przedstawiono na rysunku 3.4.

Jedna z możliwych implementacji (środowisko MATLAB) metody TFI dla powierzchni została przedstawiona poniżej

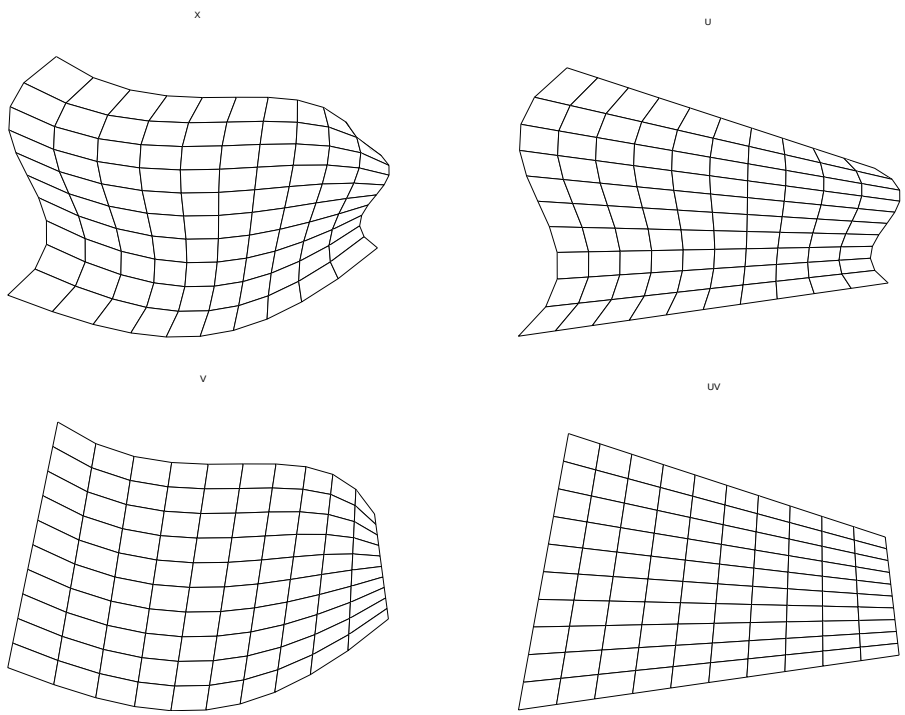
```
%
%    Generacja siatki przy użyciu interpolacji algebraicznej (TFI)
%      jako wielomianów interpolacyjnych użyto funkcji liniowych
%
%    opracował   W.Gordon
%    programował B.Błachowski
%

function [x,y,z] = f(x,y,z)

n = length(x)-1  ;

ksi = 0:1/n:1    ;  eta = ksi    ;

ux=zeros(n+1)    ;    uy = ux    ;    uvx = ux   ;
uvy = ux         ;    vx = ux    ;    vy = ux    ;

ux  = x(:,1)*(1-ksi) +  x(:,n+1)*ksi ;
vx  = (1-eta)'*x(1,:) +  eta'*x(n+1,:) ;
uvx = x(1,1)*(1-ksi)'*(1-eta) + x(1,n+1)*(1-ksi)'*eta + ...
      x(n+1,1)*ksi'*(1-eta) + x(n+1,n+1)*ksi'*eta;

uy  = y(:,1)*(1-ksi) + y(:,n+1)*ksi ;
vy  = (1-eta)'*y(1,:) +  eta'*y(n+1,:) ;
uvy = y(1,1)*(1-ksi)'*(1-eta) + y(1,n+1)*(1-ksi)'*eta + ...
      y(n+1,1)*ksi'*(1-eta) + y(n+1,n+1)*ksi'*eta;

uz  = z(:,1)*(1-ksi) + z(:,n+1)*ksi ;
vz  = (1-eta)'*z(1,:) +  eta'*z(n+1,:) ;
uvz = z(1,1)*(1-ksi)'*(1-eta) + z(1,n+1)*(1-ksi)'*eta + ...
      z(n+1,1)*ksi'*(1-eta) + z(n+1,n+1)*ksi'*eta;

x = ux + vx - uvx ;
y = uy + vy - uvy ;
z = uz + vz - uvz ;
```



Rysunek 3.2: Liniowa TFI układ współrzędnych obliczeniowych(na górze) i układ współrzędnych fizycznych(na dole)



Rysunek 3.3: Interpretacja geometryczna sum Boole'a



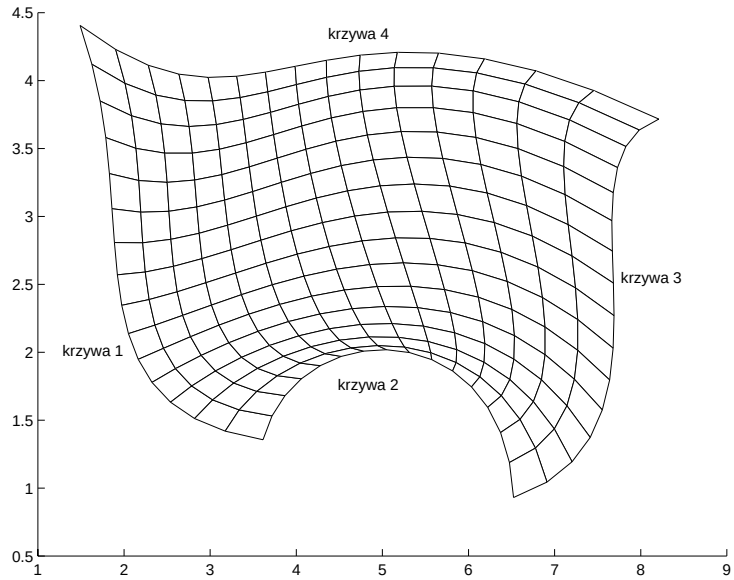
Rysunek 3.4: Płaska interpretacja sum Boole'a

Realizacja algorytmu liniowej TFI przy użyciu powyższego programu jest następująca: wprowadzamy na wejściu cztery krzywe, które reprezentowane są przez dyskretny zbiór punktów, w wyniku działania funkcji uzyskujemy współrzędne węzłów siatki (znajdujące się w macierzach x, y oraz z). Następnie w celu wyświetlenia wyników należy wywołać funkcję `mesh(x,y,z)`.

3.4.2 Interpolacja Lagrange'a

Gdy istnieje potrzeba uwzględnienia w procesie generacji siatki dodatkowych powierzchni (krzywych) znajdujących się wewnątrz rozpatrywanego obszaru, należy jako funkcje mieszane zastosować wielomiany Lagrange'a. W ogólnym przypadku dla L dostarczonych powierzchni (krzywych) w kierunku ξ otrzymujemy

$$\alpha_i^0(\xi) = \frac{\prod_{\substack{\tilde{i}=1 \\ \tilde{i}\neq i}}^{L}(\xi - \xi_i)}{\prod_{\substack{\tilde{i}=1 \\ \tilde{i}\neq i}}^{L}(\xi_{\tilde{i}} - \xi_i)}$$



Rysunek 3.5: Przykładowa siatka uzyskana za pomocą TFI

Wówczas jednorodna interpolacja w obliczeniowym kierunku ξ jest następująca

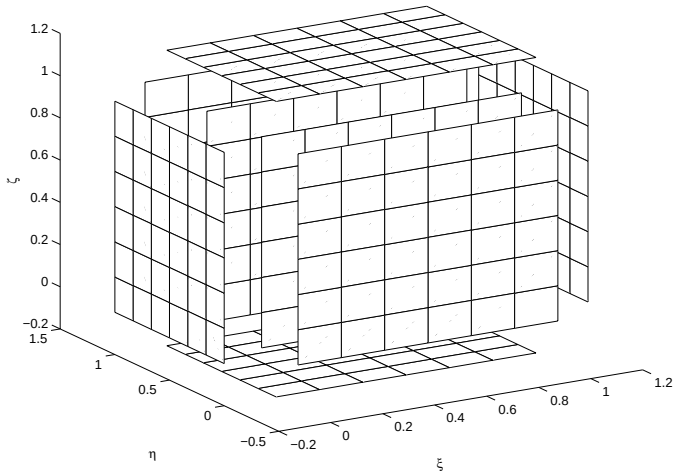
$$\mathbf{U}(\xi,\eta,\zeta) = \sum_{i=1}^{L} \alpha_i^0(\xi) \frac{\partial^0 \mathbf{X}(\xi,\eta,\zeta)}{\partial \xi^0} = \sum_{i=1}^{L} \alpha_i^0(\xi) \mathbf{X}(\xi,\eta,\zeta)$$

Funkcja Lagrange'a tworzy wielomian stopnia $L-1$ przechodzący przez L wskazanych punktów i spełnia warunek $\alpha_i^0(\xi_i) = \delta_{i\bar{i}}$. Oczywiste jest, że jeśli $L = 2$ to wówczas interpolacja Lagrange'a przechodzi w interpolację liniową.

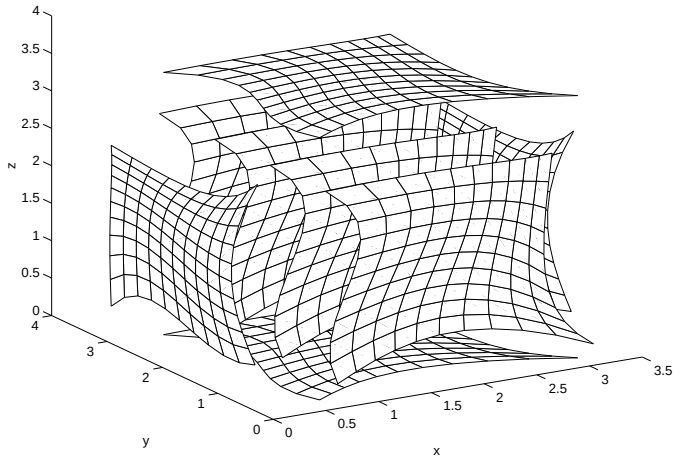
3.4.3 Interpolacja Hermite'a

Bardzo często przy generowaniu siatki możemy dostarczyć informacji na temat normalnej do brzegu otaczającego dany obszar. W takiej sytuacji należy zastosować interpolację Hermite'a. Dla zilustrowania zagadnienia przedstawiono wzór na funkcję $\mathbf{U}(\xi,\eta,\zeta)$ $(L = 2, P = 1)$.

$$\begin{aligned}
\mathbf{U}(\xi,\eta,\zeta) &= \sum_{i=1}^{2}\sum_{n=0}^{1} \alpha_i^n(\xi) \frac{\partial^n \mathbf{X}(\xi,\eta,\zeta)}{\partial \xi^n} = \\
&= \alpha_1^0(\xi)\mathbf{X}(\xi_1,\eta,\zeta) + \alpha_1^1(\xi)\frac{\partial \mathbf{X}(\xi_1,\eta,\zeta)}{\partial \xi} + \\
&+ \alpha_2^0(\xi)\mathbf{X}(\xi_2,\eta,\zeta) + \alpha_2^1(\xi)\frac{\partial \mathbf{X}(\xi_2,\eta,\zeta)}{\partial \xi}
\end{aligned}$$



Rysunek 3.6: Przestrzeń obliczeniowa dla interpolacji Lagrange'a z dodatkowymi powierzchniami



Rysunek 3.7: Przestrzeń fizyczna dla TFI Lagrange'a

gdzie

$$\begin{aligned}
\alpha_1^0(\xi) &= 2\xi^3 - 3\xi^2 + 1 \\
\alpha_1^1(\xi) &= \xi^3 - 2\xi^2 + \xi \\
\alpha_2^0(\xi) &= -2\xi^3 + 3\xi^2 \\
\alpha_2^1(\xi) &= \xi^3 - \xi^2
\end{aligned}$$

3.5 Kontrola gęstości siatki

TFI przekształca prostokątny sześcian obliczeniowy w fizyczną bryłę z nieregularnymi ograniczeniami. Równomierna siatka w przestrzeni obliczeniowej jest uzyskiwana poprzez podział, każdego kierunku na równe części. W czasie transformacji dyskretny zbiór punktów jest następnie odpowiednio rozmieszczany w przestrzeni fizycznej. Odstępy między węzłami siatki mogą być kontrolowane poprzez funkcje mieszane. Jednak funkcje mieszane produkujące siatkę dopasowaną do brzegów mogą generować niewłaściwe odstępy pomiędzy węzłami siatki. Z tego powodu należy dostarczyć dodatkowych informacji dotyczących obszarów zagęszczenia siatki. W tym celu wprowadzamy dodatkową przestrzeń (u, v, w) pośredniczącą pomiędzy przestrzenią obliczeniową (ξ, η, ζ), a przestrzenią fizyczną (X, Y, Z).

$$u = f(\xi, \eta, \zeta) \qquad v = g(\xi, \eta, \zeta) \qquad w = h(\xi, \eta, \zeta)$$

Po zastosowaniu takich funkcji uzyskujemy, z równomiernej siatki na jednostkowym sześcianie, nierównomierną siatkę określoną również na jednostkowym sześcianie, lecz w układzie kontrolnym. Wówczas funkcje mieszane stają się funkcjami zmiennej zależnej

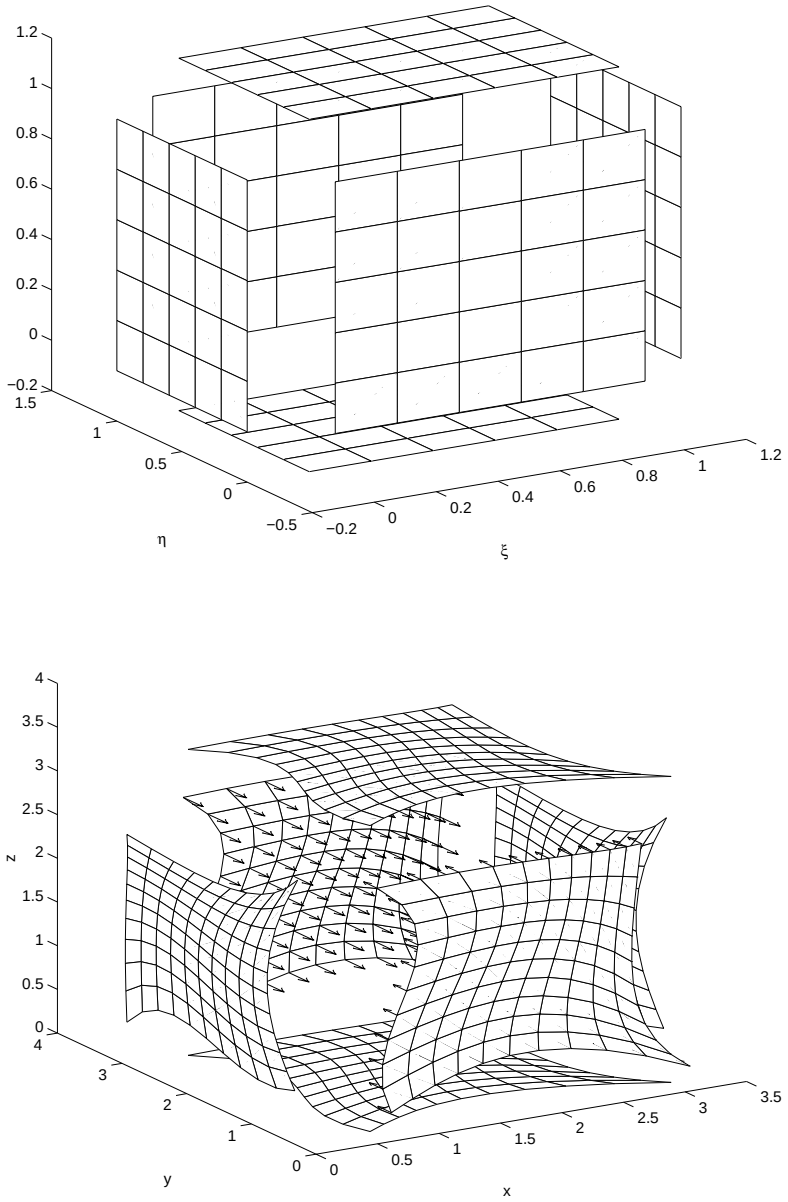
$$\alpha_i^n(u) \qquad \beta_j^m(v) \qquad \gamma_k^l(w)$$

3.5.1 Pojedyncza funkcja wykładnicza

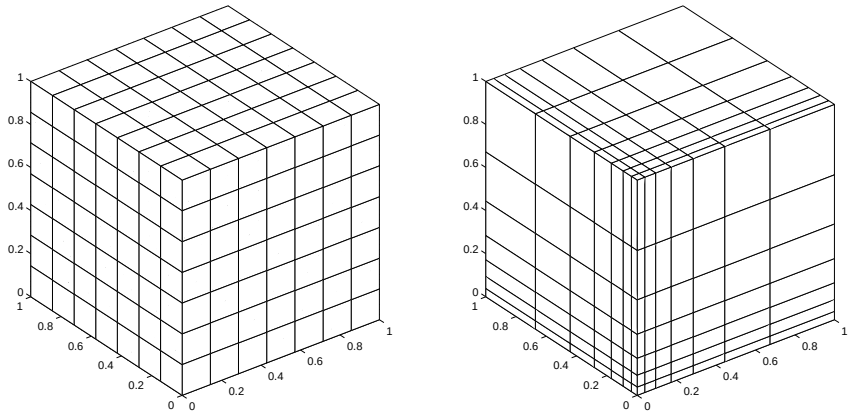
Pojedyncza funkcja wykładnicza ma postać

$$r = \frac{e^{A\rho} - 1}{e^A - 1} \qquad 0 \leqslant \rho \leqslant 1$$

Funkcja ta została dobrana tak, aby jej wartości również były z zakresu [0,1]. Jedynym ograniczeniem narzuconym na tę funkcję jest to, aby parametr $A \neq 0$. Na rysunku 3.11 przedstawiono przebieg pojedynczej funkcji wykładniczej.



Rysunek 3.9: Przestrzeń fizyczna interpolacji Hermite'a i wektory normalne do powierzchni brzegowych



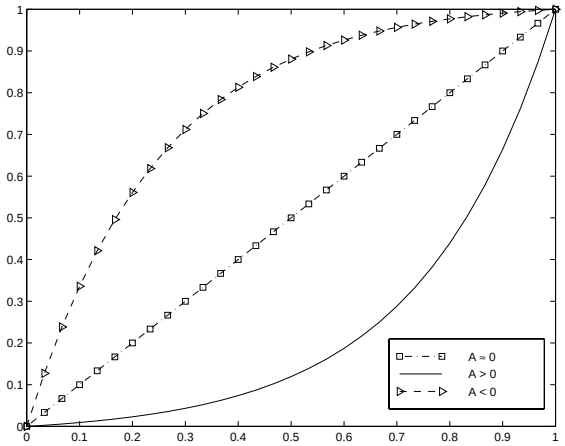
Rysunek 3.10: Przestrzeń obliczeniowa i odpowiadająca jej przestrzeń kontrolna

3.5.2 Podwójna funkcja wykładnicza

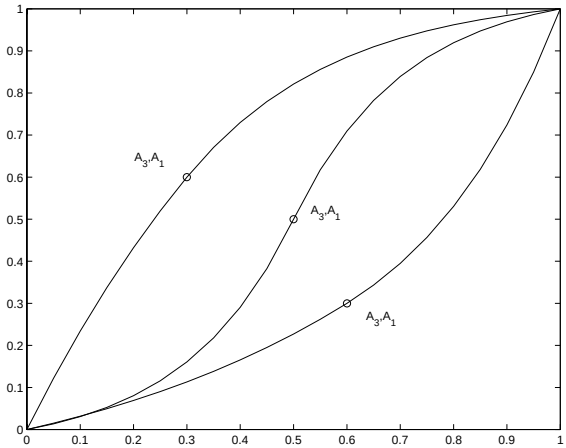
Następna funkcja, która odwzorowuje zmienną niezależną ρ, $0 \leqslant \rho \leqslant 1$, w monotonicznie rosnącą zmienną zależną r, $0 \leqslant r \leqslant 1$, jest bardziej elastyczna niż pojedyncza funkcja wykładnicza

$$
\begin{aligned}
& r = A_1 \frac{e^{\frac{A_2}{A_3}} - 1}{e^{A_2} - 1} \\
& 0 \leqslant \rho \leqslant A_3 \qquad 0 \leqslant r \leqslant A_1 \\
& r = A_1 + (1 - A_1) \frac{e^{A_4 \frac{\rho - A_3}{1 - A_3}} - 1}{e^{A_4} - 1} \\
& A_3 \leqslant \rho \leqslant 1 \qquad A_1 \leqslant r \leqslant 1
\end{aligned}
$$

Parametr A_4 obliczamy z warunku równości pochodnych w punkcie A_3. Jak można zauważyć z powyższych zależności podwójna funkcja wykładnicza to sklejone dwie pojedyncze funkcje wykładnicze. W celu zdefiniowania tej funkcji należy podać paramerty A_1, A_2, A_3 a ostatni parametr A_4 należy obliczyć. Podwójna funkcja wykładnicza, w przeciwieństwie do pojedynczej funkcji wykładniczej, umożliwia zagęszczenie siatki również wewnątrz rozpatrywanego obszaru.



Rysunek 3.11: Wykresy funkcji wykładniczej dla różnych wartości parametru A



Rysunek 3.12: Wykres podwójnej funkcji wykładniczej

Rozdział 4

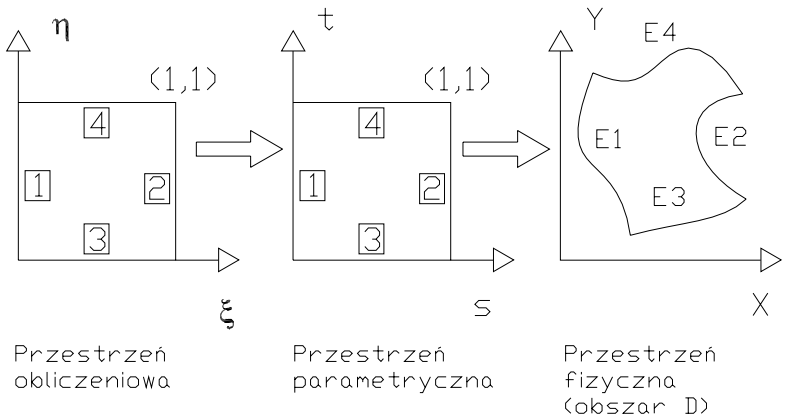
Eliptyczne systemy generacji siatek

4.1 Wprowadzenie

Od momentu ukazania się pionierskich prac Thompsona na temat eliptycznej metody generacji siatek, wiadomo jest, że układy eliptycznych równań różniczkowych cząstkowych drugiego rzędu produkują najlepsze siatki zarówno pod względem gładkości jak i rozmieszczenia węzłów siatki. Eliptyczny system generacji siatek stanowią quasi-liniowe równania różniczkowe cząstkowe, znane jako równania Poissona, oraz funkcje kontrolne. Początkowo Thompson i Warsi [12] zalecali używanie systemu Poissona z określonymi bezpośrednio funkcjami kontrolnymi, jednak obecnie dużym powodzeniem cieszy się metoda wyznaczania funkcji kontrolnych przy transformacji z jednego układu współrzędnych do drugiego. Transformacja ta, zwana również siatką kontrolną, jest odwzorowaniem węzłów z przestrzeni obliczeniowej do przestrzeni parametrycznej. Zmienne niezależne w przestrzeni parametrycznej stają się następnie funkcjami harmonicznymi w przestrzeni fizycznej. Odwzorowanie z przestrzeni fizycznej do przestrzeni parametrycznej jest nazywane odwzorowaniem harmonicznym. Odwzorowanie odwrotne do odwzorowania harmonicznego uzyskuje się poprzez rozwiązanie układu równań Poissona z funkcjami kontrolnymi określonymi na siatce kontrolnej. Numeryczne rozwiązanie systemu Poissona otrzymujemy w procesie iteracyjnym po zastosowaniu metody Picarda rozwiązywnia równań różniczkowych cząstkowych.

4.2 Podstawy matematyczne

Rozważamy dowolnie zakrzywiony obszar D w dwuwymiarowej przestrzeni we współrzędnych kartezjańskich (x, y). Zakładamy, że obszar D jest ograniczony



Rysunek 4.1: Całkowite odwzorowanie z przestrzeni obliczeniowej do przestrzeni fizycznej

przez cztery krzywe E1,E2,E3,E4. Niech (E1,E2) oraz (E3,E4) będą dwoma parami przeciwległych krawędzi (rys. 4.1). Odwzorowanie harmoniczne definiujemy jako jednoznaczne odwzorowanie z obszaru D na jednostkowy kwadrat w taki sposób, że:

- krawędzie brzegowe D są odwzorowane na krawędzie jednostkowego kwadratu
- wierzchołki narożne z obszaru D przestrzeni fizycznej są odwzorowane w naroża kwadratu w przestrzeni parametrycznej
- obie współrzędne odwzorowania są funkcjami harmonicznymi we wnętrzu D.

Niech $\vec{s}$: $D \rightarrow P$ będzie odwzorowaniem harmonicznym, gdzie parametryczna przestrzeń P jest jednostkowym kwadratem w dwuwymiarowej przestrzeni kartezjańskiej $\vec{s} = (s,t)^T$. Zakładamy, że

- $s \equiv 0$ na brzegu E1 i $s \equiv 1$ na brzegu E2
- $t \equiv 0$ na brzegu E3 i $t \equiv 1$ na brzegu E4

Następnie problem wygenerowania siatki w przestrzeni fizycznej D może być zredukowany do zagadnienia generacji siatki w przestrzeni parametrycznej P, która następnie może być odwzorowana na D, poprzez użycie odwrotnego odwzorowania harmonicznego $\vec{x}$: $P \rightarrow D$ gdzie $\vec{x} = (x,y)^T$. Dodatkowo, podobnie jak przy metodzie algebraicznej TFI, definiujemy przestrzeń obliczeniową C jako jednostkowy kwadrat w kartezjańskim układzie współrzędnych $\vec{\xi} = (\xi,\eta)^T$. Siatka kontrolna

jest zdefiniowana jako odwzorowanie jednorodnej siatki z przestrzeni obliczeniowej C na niejednorodną siatkę w przestrzeni parametrycznej P. Konstrukcja siatki kontrolnej $\vec{s}\colon C \to P$ oraz odwrotnego odwzorowania harmonicznego $\vec{x}\colon P \to D$ definiuje odwzorowanie

$$\vec{x}\colon C \to D$$

które przeprowadza równomierną siatkę C na dowolnie zakrzywioną w obszarze D. Końcowe odwzorowanie spełnia quasi-liniowy układ równań Poissona z funkcjami kontrolnymi. Sekretem każdego dobrego algorytmu eliptycznej generacji siatki jest właściwe dobranie funkcji kontrolnych co jest równoważne z konstrukcją stosownej siatki kontrolnej. Poniżej przedstawiono wyprowadzenie quasi-liniowego układu równań Poissona służącego do generacji siatki.

Cytując za autorami pracy [12] wprowadzamy dwa kowariantne wektory bazowe

$$\vec{a}_1 = \frac{\partial \vec{x}}{\partial \xi} = \vec{x}_\xi \quad , \quad \vec{a}_2 = \frac{\partial \vec{x}}{\partial \eta} = \vec{x}_\eta$$

i obliczamy kowariantny tensor metryczny jako iloczyn skalarowy wektorów kowariantnych

$$a_{ij} = \langle \vec{a}_i, \vec{a}_j \rangle \qquad i,j = 1,2$$

Następnie określamy dwa wektory kontrawariantne

$$\vec{a}^1 = \nabla \xi = (\xi_x, \xi_y)^T \qquad \vec{a}^2 = \nabla \eta = (\eta_x, \eta_y)^T$$

i kontrawariantny tensor metryczny

$$a^{ij} = \langle \vec{a}^i, \vec{a}^j \rangle \qquad i,j = 1,2$$

$$a_{ij} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \qquad a^{ij} = \begin{pmatrix} a^{11} & a^{12} \\ a^{21} & a^{22} \end{pmatrix}$$

Następnie obliczamy wyznacznik kowariantnego tensora metrycznego

$$J^2 = a_{11}a_{22} - a_{12}^2$$

Rozważmy teraz dowolną funkcję $\phi = \phi(\xi, \eta)$ zdefiniowaną w obszarze D i obliczamy laplasjan tej funkcji

$$\Delta\phi = \phi_{xx} + \phi_{yy} = \frac{1}{J}\left\{(Ja^{11}\phi_\xi + Ja^{12}\phi_\eta)_\xi + (Ja^{12}\phi_\xi + Ja^{22}\phi_\eta)_\eta\right\}$$

Rozważając szczególne przypadki $\phi \equiv \xi, \phi \equiv \eta$ otrzymujemy

$$\Delta\xi = \frac{1}{J}\left\{(Ja^{11})_\xi + (Ja^{12})_\eta\right\}$$
$$\Delta\eta = \frac{1}{J}\left\{(Ja^{12})_\xi + (Ja^{22})_\eta\right\}$$

stąd otrzymamy wyrażenie na laplasjan ϕ

$$\Delta\phi = a^{11}\phi_{\xi\xi} + 2a^{12}\phi_{\xi\eta} + a^{22}\phi_{\eta\eta} + \Delta\xi\phi_\xi + \Delta\eta\phi_\eta$$

Podstawiając odpowiednio s i t za ϕ otrzymujemy

$$\begin{aligned}\Delta s &= a^{11}s_{\xi\xi} + 2a^{12}s_{\xi\eta} + a^{22}s_{\eta\eta} + \Delta\xi s_\xi + \Delta\eta s_\eta \\ \Delta t &= a^{11}t_{\xi\xi} + 2a^{12}t_{\xi\eta} + a^{22}t_{\eta\eta} + \Delta\xi t_\xi + \Delta\eta t_\eta\end{aligned}$$

Używając tych równań i wykorzystując fakt, że s oraz t są harmoniczne w obszarze D otrzymamy $\Delta s = 0$ i $\Delta t = 0$ a stąd znajdujemy wyrażenia na laplasjan dla ξ oraz η

$$\begin{pmatrix} \Delta\xi \\ \Delta\eta \end{pmatrix} = a^{11}\vec{P}_{11} + 2a^{12}\vec{P}_{12} + a^{22}\vec{P}_{22}$$

gdzie

$$\vec{P}_{11} = -T^{-1}\begin{pmatrix} s_{\xi\xi} \\ t_{\xi\xi} \end{pmatrix} \qquad \vec{P}_{12} = -T^{-1}\begin{pmatrix} s_{\xi\eta} \\ t_{\xi\eta} \end{pmatrix} \qquad \vec{P}_{22} = -T^{-1}\begin{pmatrix} s_{\eta\eta} \\ t_{\eta\eta} \end{pmatrix}$$

a macierz T zdefiniowano jako

$$T = \begin{pmatrix} s_\xi & s_\eta \\ t_\xi & t_\eta \end{pmatrix}$$

Sześć współczynników występujących w wektorach $\vec{P}_{11} = (P^1_{11}, P^2_{11})^T$, $\vec{P}_{12} = (P^1_{12}, P^2_{12})^T$ oraz $\vec{P}_{22} = (P^1_{22}, P^2_{22})^T$ to wspomniane wcześniej funkcje kontrolne. Ostatecznie podstawiając za ϕ wektor (x, y) otrzymamy

$$\Delta\vec{x} = a^{11}\vec{x}_{\xi\xi} + 2a^{12}\vec{x}_{\xi\eta} + a^{22}\vec{x}_{\eta\eta} + \Delta\xi\vec{x}_\xi + \Delta\eta\vec{x}_\eta$$

Uwzględniając równanie na $\Delta\xi$ i $\Delta\eta$ oraz warunek $\Delta x \equiv 0$ i $\Delta y \equiv 0$ otrzymamy układ równań Poissona

$$\begin{aligned} a^{11}\vec{x}_{\xi\xi} + 2a^{12}\vec{x}_{\xi\eta} + a^{22}\vec{x}_{\eta\eta} &+ (a^{11}P^1_{11} + 2a^{12}P^1_{12} + a^{22}P^1_{22})\vec{x}_\xi \\ &+ (a^{11}P^2_{11} + 2a^{12}P^2_{12} + a^{22}P^2_{22})\vec{x}_\eta = 0 \end{aligned}$$

Korzystając z zależności pomiedzy wektorami ko- i kontrawariantnym

$$\begin{aligned} \vec{a}^1 &= a^{11}\vec{a}_1 + a^{12}\vec{a}_2 \qquad & \vec{a}^2 &= a^{12}\vec{a}_1 + a^{22}\vec{a}_2 \\ \vec{a}_1 &= a_{11}\vec{a}^1 + a_{12}\vec{a}^2 \qquad & \vec{a}_2 &= a_{12}\vec{a}^1 + a_{22}\vec{a}^2 \end{aligned}$$

zapiszemy

$$J^2a^{11} = a_{22} = \langle\vec{x}_\eta, \vec{x}_\eta\rangle \qquad J^2a^{12} = -a_{12} = \langle\vec{x}_\xi, \vec{x}_\eta\rangle \qquad J^2a^{22} = a_{11} = \langle\vec{x}_\xi, \vec{x}_\xi\rangle$$

Następnie mnożąc układ Poissona przez jakobian podniesiony do kwadratu uzyskujemy ostateczną postać równań dla eliptycznej metody generacji siatek

$$\begin{aligned} a_{22}\vec{x}_{\xi\xi} - 2a_{12}\vec{x}_{\xi\eta} + a_{11}\vec{x}_{\eta\eta} &+ (a_{22}P^1_{11} - 2a^{12}P^1_{12} + a_{11}P^1_{22})\vec{x}_\xi \\ &+ (a_{22}P^2_{11} - 2a^{12}P^2_{12} + a_{11}P^2_{22})\vec{x}_\eta = 0 \end{aligned}$$

4.3 Iteracyjne sformułowanie metody

Wprowadzając podział w przestrzeni obliczeniowej na $(N+1)\times(M+1)$ punktów, dyskretyzujemy zmienne ξ oraz η jak następuje

$$\xi_i = \frac{i}{N} \qquad \eta_j = \frac{j}{M} \qquad i = 0, ..., N \quad j = 0, ..., M$$

Następnie dyskretyzujemy krawędzie E1,E2,E3 oraz E4 umieszczając informację o nich w macierzach x(i,j) i y(i,j). Kolejno konstruujemy siatkę kontrolną s(i,j) i t(i,j) oraz w każdym punkcie wewnętrzym siatki obliczamy sześć funkcji kontrolnych korzystając z wzorów rachunku różnicowego. Następnie w celu uniknięcia nieliniowości formułujemy rozwiązanie układu równań Poissona przez metodę iteracyjną Picarda

$$P^{k-1}\vec{x}^k_{\xi\xi} - 2Q^{k-1}\vec{x}^k_{\xi\eta} + R^{k-1}\vec{x}^k_{\eta\eta} + S^{k-1}\vec{x}^k_{\xi} + T^{k-1}\vec{x}^k_{\eta} = 0$$

gdzie

$$\begin{aligned}
&P^{k-1} = \langle \vec{x}^{k-1}_{\eta}, \vec{x}^{k-1}_{\eta} \rangle \qquad Q^{k-1} = \langle \vec{x}^{k-1}_{\xi}, \vec{x}^{k-1}_{\eta} \rangle \qquad R^{k-1} = \langle \vec{x}^{k-1}_{\xi}, \vec{x}^{k-1}_{\xi} \rangle \\
&S^{k-1} = P^{k-1}P^1_{11} - 2Q^{k-1}P^1_{12} + R^{k-1}P^1_{22} \\
&T^{k-1} = P^{k-1}P^2_{11} - 2Q^{k-1}P^2_{12} + R^{k-1}P^2_{22}
\end{aligned}$$

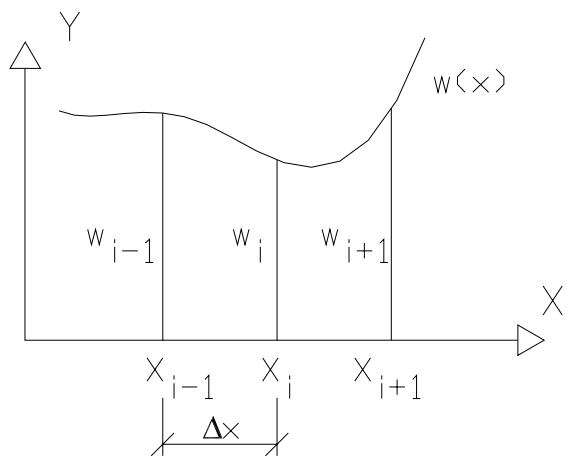
Dowód twierdzenia Picarda znajduje się w książce [9].

Poniżej przedstawiono algorytm obliczania węzłów siatki metodą eliptyczną:

- Krok 1. Obliczamy sześć funkcji kontrolnych poprzez różnice centralne położenia węzłów siatki kontrolnej.
- Krok 2. Obliczamy początkową siatkę przez zastosowanie jednej z metod algebraicznych. Wystarczy tu w zupełności liniowa interpolacja TFI, ponieważ jakość początkowej siatki nie jest istotna. Końcowa siatka wygenerowana przez rozwiązanie równań Poissona nie zależy od położenia węzłów siatki startowej.
- Krok 3. Rozwiązanie quasi-liniowego układu Poissona przy użyciu iteracji Picarda. W ogólności zalecane jest wykonanie około 10 iteracji.

4.4 Implementacja numeryczna

Obecnie zajmiemy się dyskretyzacją i implementacją uzyskanego wcześniej układu równań różniczkowych Poissona. Najpierw jednak wprowadzimy niezbędne podstawy rachunku różnicowego, a następnie wyprowadzimy iteracyjne wzory na obliczenie współrzędnych węzłów siatki typu eliptycznego.



Rysunek 4.2: Funkcja w w postaci ciągłej $w(x)$ i dyskretnej $w(x_i)$

Jak powszechnie wiadomo, podstawową ideą metody różnic skończonych jest zastąpienie równań różniczkowych przez odpowiednie równania algebraiczne zwane równaniami różnicowymi. W tym celu wprowadza się następujące pojęcia:

- różnica lewostronna
- różnica prawostronna
- różnica centralna

I tak, różnicę lewostronną w punkcie i obliczamy jako

$$\left(\frac{\Delta w}{\Delta x}\right)_i^L = \frac{w_i - w_{i-1}}{\Delta x}$$

różnica prawostronna to

$$\left(\frac{\Delta w}{\Delta x}\right)_i^P = \frac{w_{i+1} - w_i}{\Delta x}$$

oraz różnica centralna

$$\left(\frac{\Delta w}{\Delta x}\right)_i = \frac{w_{i+1} - w_{i-1}}{2\Delta x}$$

Interpretację geometryczną różnic skończonych przedstawiono na rysunku 4.2. Różnicę skończoną dowolnego rzędu lewostronną, prawostronną jak i centralną

obliczymy z następujących zależności rekurencyjnych

$$\left(\frac{\Delta^n w}{\Delta x^n}\right)_i^L = \frac{1}{\Delta x}\left[\left(\frac{\Delta^{n-1} w}{\Delta x^{n-1}}\right)_i - \left(\frac{\Delta^{n-1} w}{\Delta x^{n-1}}\right)_{i-1}\right]$$
$$\left(\frac{\Delta^n w}{\Delta x^n}\right)_i^P = \frac{1}{\Delta x}\left[\left(\frac{\Delta^{n-1} w}{\Delta x^{n-1}}\right)_{i+1} - \left(\frac{\Delta^{n-1} w}{\Delta x^{n-1}}\right)_i\right]$$

dla $n = 2K$

$$\left(\frac{\Delta^n w}{\Delta x^n}\right)_i = \frac{1}{\Delta x}\left[\left(\frac{\Delta^{n-1} w}{\Delta x^{n-1}}\right)_i^P - \left(\frac{\Delta^{n-1} w}{\Delta x^{n-1}}\right)_i^L\right]$$

dla $n = 2K + 1$

$$\left(\frac{\Delta^n w}{\Delta x^n}\right)_i = \frac{1}{2}\left[\left(\frac{\Delta^n w}{\Delta x^n}\right)_i^P + \left(\frac{\Delta^n w}{\Delta x^n}\right)_i^L\right]$$

Mając tak przygotowane wzory określamy pochodne funkcji $w(x)$ jako

$$\frac{dw(x_i)}{dx} \approx \left(\frac{\Delta w}{\Delta x}\right)_i$$
$$\frac{d^2 w(x_i)}{dx^2} \approx \left(\frac{\Delta^2 w}{\Delta x^2}\right)_i \qquad \text{itd.}$$

Stosując powyższe założenia otrzymujemy wzór na pierwszą pochodną dla funkcji $x(\xi, \eta)$

$$\frac{\partial x}{\partial \xi} \approx \frac{x_{i+1,j} - x_{i-1,j}}{2\Delta\xi}$$

oraz

$$\frac{\partial x}{\partial \eta} \approx \frac{x_{i,j+1} - x_{i,j-1}}{2\Delta\eta}$$

są to wprowadzone wcześniej różnice centralne, gdzie $\Delta\xi$ oraz $\Delta\eta$ to odpowiednio odległości pomiędzy liniami siatki w przestrzeni obliczeniowej ξ, η. Kolejno korzystamy ze wzoru na drugą pochodną cząstkową w kierunku ξ oraz η

$$\frac{\partial^2 x(\xi,\eta)}{\partial \xi^2} \approx \frac{x_{i+1,j} - 2x_{i,j} + x_{i-1,j}}{\Delta\xi^2}$$
$$\frac{\partial^2 x(\xi,\eta)}{\partial \eta^2} \approx \frac{x_{i,j+1} - 2x_{i,j} + x_{i,j-1}}{\Delta\eta^2}$$

I wreszcie na drugą pochodną mieszaną

$$\frac{\partial^2 x(\xi,\eta)}{\partial \xi\eta} \approx \frac{x_{i+1,j+1} - x_{i+1,j-1} - x_{i-1,j+1} + x_{i-1,j-1}}{4\Delta\xi\Delta\eta}$$

Następnie równania Poissona

$$P\vec{x}_{\xi\xi} - 2Q\vec{x}_{\xi\eta} + R\vec{x}_{\eta\eta} + S\vec{x}_{\xi} + T\vec{x}_{\eta} = 0$$

dyskretyzujemy i przyjmując rozstaw siatki w przestrzeni obliczeniowej równy jedności ($\Delta\xi = \Delta\eta = 1$) otrzymujemy

$$P(x_{i+1,j} - 2x_{i,j} + x_{i-1,j}) - 2Q\left(\frac{x_{i+1,j+1} - x_{i+1,j-1} - x_{i-1,j+1} + x_{i-1,j-1}}{4}\right) +$$
$$+R(x_{i,j+1} - 2x_{i,j} + x_{i,j-1}) + S\left(\frac{x_{i+1,j} - x_{i-1,j}}{2}\right) + T\left(\frac{x_{i,j+1} - x_{i,j-1}}{2}\right) = 0$$

Przenosząc $x_{i,j}$ na drugą stronę otrzymujemy gotowe do zastosowania wzory iteracyjne dla strukturalnych siatek typu eliptycznego.

$$\begin{aligned} x_{i,j} &= \frac{1}{4(P+R)}[2P(x_{i+1,j} + x_{i-1,j}) - Q(x_{i+1,j+1} - x_{i+1,j-1} - x_{i-1,j+1} + x_{i-1,j-1}) \\ &+ 2R(x_{i,j+1} + x_{i,j-1}) + S(x_{i+1,j} - x_{i-1,j}) + T(x_{i,j+1} - x_{i,j-1})] \end{aligned}$$

Szczegółowy opis numerycznego rozwiązywania równań różniczkowych cząstkowych typu eliptycznego można znaleźć w klasycznym już podręczniku do metod numerycznych [4].

Poniżej zamieszczono funkcję realizującą algorytm Picarda w środowisku MATLAB.

```
%
%       Iteracyjne rozwiazanie równań Poissona
%        z zastosowaniem funkcji kontrolnych
%
%       opracował   J.F.Thompson
%       programował B.Błachowski
%

function [x,y] = f(x,y,iteracja)

for Picard = 1:iteracja
 for i = 2: length(x)-1
  for j = 2: length(x)-1

   xk = 0.5*(x(i+1,j)-x(i-1,j)) ;  xn = 0.5*(x(i,j+1)-x(i,j-1)) ;
   yk = 0.5*(y(i+1,j)-y(i-1,j)) ;  yn = 0.5*(y(i,j+1)-y(i,j-1)) ;

   g11 = xk^2+yk^2   ;
```

```
   g12 = xk*xn+yk*yn ;
   g22 = xn^2+yn^2   ;

   x(i,j) = 1/(4*( g11 + g22 )) *(                                ...
      2*g22*( x(i+1,j) + x(i-1,j) )    +                          ...
      2*g11*( x(i,j+1) + x(i,j-1) )    -                          ...
      g12* ( x(i+1,j+1) - x(i+1,j-1) + x(i-1,j-1) - x(i-1,j+1) )...
      ) ;

   y(i,j) = 1/(4*( g11 + g22 )) *(                                ...
      2*g22*( y(i+1,j) + y(i-1,j) )   +                           ...
      2*g11*( y(i,j+1) + y(i,j-1) )   -                           ...
      g12* ( y(i+1,j+1) - y(i+1,j-1) + y(i-1,j-1) - y(i-1,j+1) )...
      ) ;

  end
 end
end
```

4.5 Podstawowe konstrukcje siatek eliptycznych

4.5.1 Siatki harmoniczne uzyskane metodą Picarda

Najprostszym przykładem siatki kontrolnej jest odwzorowanie tożsamościowe $s = \xi$, $t = \eta$. Wówczas wszystkie funkcje kontrolne są tożsamościowo równe zero i system generacji siatek Poissona redukuje się do

$$a_{22}\vec{x}_{\xi\xi} - 2a_{12}\vec{x}_{\xi\eta} + a_{11}\vec{x}_{\eta\eta} = 0$$

co jest równoważne z $\Delta\xi = 0$ i $\Delta\eta = 0$. Siatki uzyskiwane poprzez powyższe zależności zwane są siatkami Laplace'a lub harmonicznymi i były początkowo wprowadzone przez Winslowa [13]. Operator Laplace'a pojawiający się w tych równaniach generuje siatki równomiernie rozmieszczone wewnątrz obszaru. Z tego powodu siatka Laplace'a jest do zaakceptowania tylko wówczas, gdy punkty na brzegu są również rozmieszczone równomiernie.

4.5.2 Siatki harmoniczne uzyskane poprzez minimalizację funkcjonału harmonicznego

Alternatywną metodę generacji siatki bazującej na odwzorowaniach harmonicznych zaprezentował w swoich pracach [6],[7] Sergey Ivanenko. Podstawowym za-

gadnieniem jakie rozważał było znalezienie warunków dyskretnego odwzorowania w celu uzyskania siatki niezdegenerowanej. Wykazał on, że w celu zapewnienia wypukłości wszystkim pojedynczym komórkom płaskiej siatki jakobian przekształcenia musi być większy od zera. Spełnienie tego warunku jest gwarantem uzyskania siatek niezdegenerownaych. Oznacza to, że jeżeli wszystkie komórki siatki są wypukłe to żaden węzeł siatki nie opuści wnętrza konturu oraz węzły takiej siatki nie będą się wzajmnie na siebie nakładać.

Do rozwiązania tego problemu Ivanenko użył metody polegającej na bezpośredniej minimalizacji dyskretnego odpowiednika funkcjonału harmonicznego.

Płaskie siatki niezdegenerowane

Węzły strukturalnej siatki łączą się pomiędzy sobą na zasadzie sąsiednich indeksów. Typowym przykładem strukturalnej siatki jest krzywoliniowa siatka współrzędnych np.biegunowych, utworzona przez odwzorowanie jednostkowego kwadratu w krzywoliniowe współrzędne fizyczne. Warunkiem niezdegenerowania siatki jest to, aby jakobian przekształcenia był większy od zera.

Twierdzenie 1 *Jeżeli gładkie odwzorowanie jednego obszaru w drugi, jednoznaczne pomiędzy brzegami tego obszaru, posiada dodatni jakobian nie tylko wewnątrz tego obszaru, ale również na brzegach, wówczas takie odwzorowanie jest jednoznaczne na całym obszarze.*

Dowód tego twierdzenia zamieszczono w pracy [1]. Krzywoliniowy układ współrzędnych nie będzie zdegenerowany jeżeli jakobian odwzorowania $x(\xi,\eta) \quad y(\xi,\eta)$ jest dodatni:

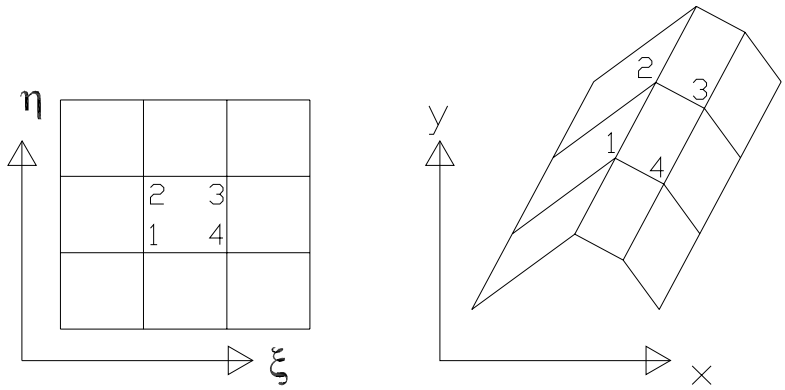
$$J = x_\xi y_\eta - x_\eta y_\xi \qquad 0 \leqslant \xi \leqslant 1 \quad 0 \leqslant \eta \leqslant 1$$

Tym samym widzimy, że problem skonstruowania krzywoliniowych współrzędnych w pewnym obszarze jest problemem znalezienia gładkiego odwzorowania jednostkowego kwadratu do wnętrza tego obszaru, tak aby jakobian takiego przekształcenia przyjmował wartości dodatnie.

Dyskretna interpretacja jakobianu

Niech $(x,y)_{i,j}$ oznaczają współrzędne węzłów siatki. Następnie konstruujemy odwzorowanie kwadratu (rys.4.3) 1,2,3,4 na płaszczyźnie ξ,η w czworobok na płaszczyźnie x,y utworzony przez węzły $(x,y)_{i,j} \quad (x,y)_{i+1,j} \quad (x,y)_{i,j+1} \quad (x,y)_{i+1,j+1}$. Każdy wierzchołek jest przypisany do odpowiedniego trójkąta: wierzchołek 1 do Δ_{412}, wierzchołek 2 do Δ_{123} itd. Podwojone pole tych trójkątów $J_k \quad ,k = 1,2,3,4$ wygląda następująco

$$\begin{aligned} J_1 &= (x_4 - x_1)(y_2 - y_1) - (y_4 - y_1)(x_2 - x_1) = \\ &= (x_{i+1,j} - x_{i,j})(y_{i,j+1} - y_{i,j}) - (y_{i+1,j} - y_{i,j})(x_{i,j+1} - x_{i,j}) \end{aligned}$$



Rysunek 4.3: Interpretacja geometryczna dyskretnej postaci jakobianu

Oznaczmy teraz odwzorowanie $x^h(\xi,\eta), y^h(\xi,\eta)$ jako następujące funkcje określone wewnątrz kwadratu 1,2,3,4 ($i \leqslant \xi \leqslant i+1$, $j \leqslant \eta \leqslant j+1$).

$$\begin{aligned} x^h(\xi,\eta) &= x_1 + (x_4 - x_1)(\xi - i) + (x_2 - x_1)(\eta - j) + \\ &+ (x_3 - x_4 - x_2 + x_1)(\xi - i)(\eta - j) \\ y^h(\xi,\eta) &= y_1 + (y_4 - y_1)(\xi - i) + (y_2 - y_1)(\eta - j) + \\ &+ (y_3 - y_4 - y_2 + y_1)(\xi - i)(\eta - j) \end{aligned}$$

Widać stąd, że kwadrat 1,2,3,4 jest liniowo przekształcany w odpowiedni czworobok. Ponieważ transformacja $x^h(\xi,\eta), y^h(\xi,\eta)$ jest ciągła na brzegach sprawdzamy warunek jednoznaczności przekształcenia

$$J^h = x^h_\xi y^h_\eta - x^h_\eta y^h_\xi = det \begin{pmatrix} x_4 - x_1 + A(\eta - j) & x_2 - x_1 + A(\xi - i) \\ y_4 - y_1 + B(\eta - j) & y_2 - y_1 + B(\xi - i) \end{pmatrix}$$

gdzie $A = x_3 - x_4 - x_2 + x_1, B = y_3 - y_4 - y_2 + y_1$. Widać stąd, że jakobian jest liniowy, ponieważ współczynniki przed członem $\xi\eta$ są równe zeru. W konsekwencji jeżeli $J^h > 0$ we wszystkich wierzchołkach kwadratu, to żaden z nich nie znika wewnątrz kwadratu. W wierzchołku 1 ($\xi = i, \eta = j$) jakobian wynosi

$$J^h = (x_4 - x_1)(y_2 - y_1) - (y_4 - y_1)(x_2 - x_1)$$

i jest to podwojone pole trójkąta Δ_{412}. Z powyższego wynikają następujące warunki dodatności jakobianu

$$[J_k]_{i+1/2,j+1/2} > 0 \qquad k = 1,2,3,4 \quad i = 1,...,i^* - 1 \quad j = 1,...,j^* - 1$$

gdzie $J_k = (x_{k-1} - x_k)(y_{k+1} - y_k) - (y_{k-1} - y_k)(x_{k+1} - x_k)$ i jeżeli $k = 1$ to za $k-1$ wstawiamy 4, natomiast jeżeli $k = 4$ to $k+1 = 1$. Jeżeli powyższe warunki są spełnione, to wówczas wszystkie komórki otrzymanej siatki są czworobokami wypukłymi.

Adaptacyjne siatki harmoniczne

Jak wspomniano wcześniej problem znalezienia odpowiedniej siatki sprowadza się do minimalizacji funkcjonału harmonicznego, który możemy zapisać jako

$$I = \int \frac{x_\xi^2 + y_\xi^2 + x_\eta^2 + y_\eta^2}{\sqrt{(x_\xi^2 + y_\xi^2)(x_\eta^2 + y_\eta^2) - (x_\xi x_\eta + y_\xi y_\eta)^2}} d\xi d\eta$$

Następnie należy określić równanie Eulera dla tego funkcjonału [7]. Ostatecznie po dyskretyzacji równania Eulera uzyskujemy następujące zależności iteracyjne

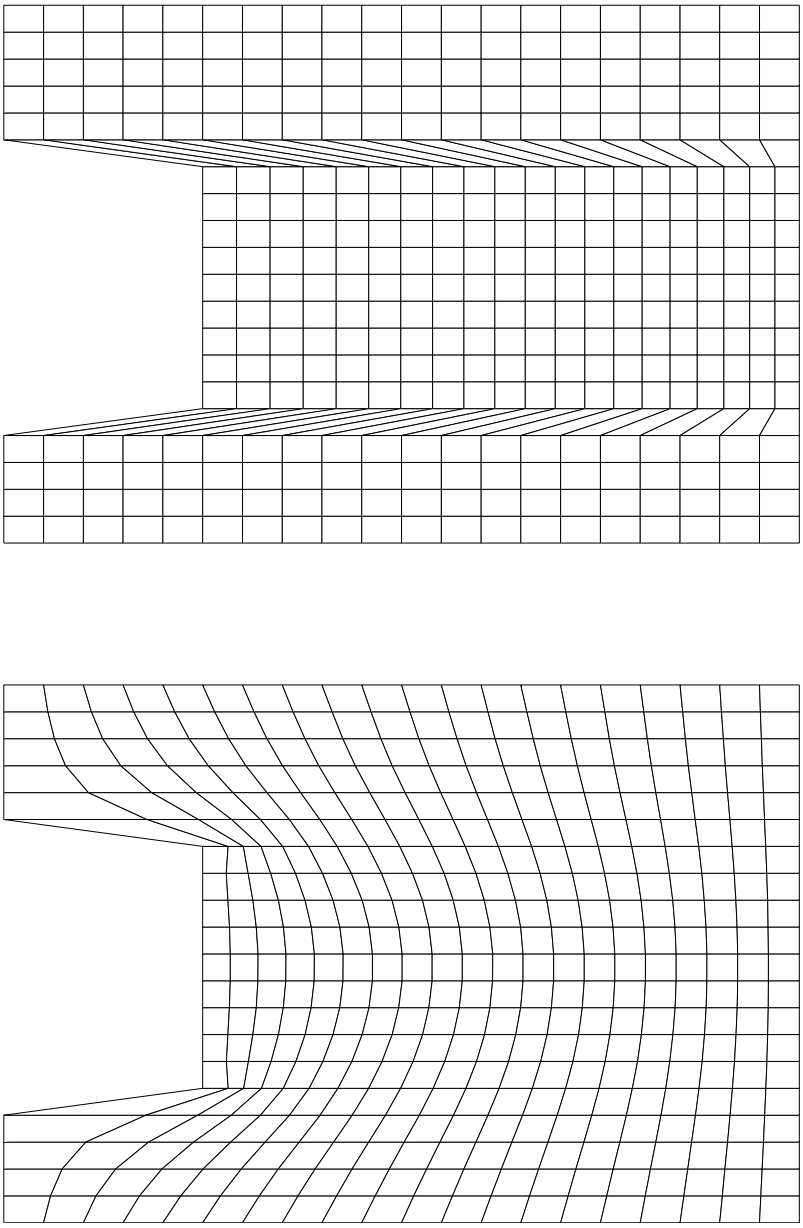
$$x_{ij}^{l+1} = x_{ij}^l + \tau \frac{[L(x)]_{ij}}{2[\alpha]_{ij} + 2[\beta]_{ij}} \qquad y_{ij}^{l+1} = y_{ij}^l + \tau \frac{[L(y)]_{ij}}{2[\alpha]_{ij} + 2[\beta]_{ij}}$$

gdzie

$$\begin{aligned}
&\alpha = x_\eta^2 + y_\eta^2 \qquad \beta = x_\xi x_\eta + y_\xi y_\eta \qquad \gamma = x_\xi^2 + y_\xi^2 \\
&L(x) = \alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta} = 0 \\
&L(y) = \alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta} = 0 \\
&0 < \tau < 1 \qquad \tau\text{- najczęściej } 0.5
\end{aligned}$$

Numeryczna implementacja generacji adaptacyjnej siatki harmonicznej może wyglądać następująco

```
%
%      Iteracyjne  rozwiazanie  równań  Laplace'a
%      przez dyskretyzację funkcjonału Dirichleta
%
%      opracował   S.Invanenko
%      programował B.Błachowski
%

function [x,y] = f(x,y,iteracja)

for Picard = 1:iteracja
 for i = 2: length(x)-1
  for j = 2: length(x)-1

xk = 0.5*(x(i+1,j)-x(i-1,j))   ;  xn=0.5*(x(i,j+1)-x(i,j-1))   ;
yk = 0.5*(y(i+1,j)-y(i-1,j))   ;  yn=0.5*(y(i,j+1)-y(i,j-1))   ;

xkk = x(i+1,j)-2*x(i,j)+x(i-1,j) ;
```



Rysunek 4.4: Porównanie metod TFI(siatka na górze) i PDF(na dole)

```
xkn = 0.25*(x(i+1,j+1)-x(i+1,j-1)-x(i-1,j+1)+x(i-1,j-1)) ;
xnn = x(i,j+1)-2*x(i,j)+x(i,j-1) ;

ykk = y(i+1,j)-2*y(i,j)+y(i-1,j) ;
ykn = 0.25*(y(i+1,j+1)-y(i+1,j-1)-y(i-1,j+1)+y(i-1,j-1)) ;
ynn = y(i,j+1)-2*y(i,j)+y(i,j-1) ;

alpha = xn^2+yn^2 ;
beta  = xk*xn+yk*yn ;
gamma = xk^2+yk^2 ;

Lx = alpha*xkk - 2*beta*xkn + gamma*xnn ;
Ly = alpha*ykk - 2*beta*ykn + gamma*ynn ;

x(i,j) = x(i,j) + 0.5*Lx/(2*alpha+2*gamma) ;
y(i,j) = y(i,j) + 0.5*Ly/(2*alpha+2*gamma) ;

  end
 end
end
```

4.5.3 Siatki bazujące na długości łuku

Kolejnym przykładem siatek kontrolnych są siatki bazujące na długości łuku. Za ich pomocą uzyskujemy siatki zarówno dobrze dopasowane do brzegu jak i z właściwym rozmieszczeniem punktów wewnętrzych. Odwzorowanie kontrolne bazujące na długości łuku definiujemy jako układ dwóch równań algebraicznych

$$\begin{aligned} s &= s^a_{E_3}(\xi)(1-t) + s^a_{E_4}(\xi)t \\ t &= t^a_{E_1}(\eta)(1-s) + t^a_{E_2}(\eta)s \end{aligned}$$

gdzie

$$\begin{array}{llll} s(0,\eta)=0 & s(1,\eta)=1 & s(\xi,0)=s^a_{E_3}(\xi) & s(\xi,1)=s^a_{E_4}(\xi) \\ t(\xi,0)=0 & t(\xi,1)=1 & t(0,\eta)=t^a_{E_1}(\eta) & t(1,\eta)=t^a_{E_2}(\eta) \end{array}$$

czyli

- $s \equiv 0$ na brzegu E1 i $s \equiv 1$ na brzegu E2
- s jest znormalizowaną długością łuku na brzegach E3 i E4
- $t \equiv 0$ na brzegu E3 i $t \equiv 1$ na brzegu E4

- t jest znormalizowaną długością łuku na brzegach E1 i E2

Dyskretyzując zagadnienie wprowadzamy miarę odległości między punktami na brzegu

$$\begin{array}{lll} \bar{d}_{0,j} = \|\vec{x}_{0,j} - \vec{x}_{0,j-1}\| & \bar{d}_{N,j} = \|\vec{x}_{N,j} - \vec{x}_{N,j-1}\| & j = 1, \ldots, M \\ \bar{d}_{i,0} = \|\vec{x}_{i,0} - \vec{x}_{i-1,0}\| & \bar{d}_{i,M} = \|\vec{x}_{i,M} - \vec{x}_{i-1,M}\| & i = 1, \ldots, N \end{array}$$

Sumując odpowiednie długości odcinków otrzymamy długość całego brzegu

$$L_{E_1} = \sum_{j=1}^{M} \bar{d}_{0,j} \qquad L_{E_2} = \sum_{j=1}^{M} \bar{d}_{N,j} \qquad L_{E_3} = \sum_{i=1}^{N} \bar{d}_{i,0} \qquad L_{E_4} = \sum_{i=1}^{N} \bar{d}_{i,M}$$

A następnie znormalizowane długości łuku

$$\begin{array}{lll} d_{0,j} = \bar{d}_{0,j}/L_{E_1} & d_{N,j} = \bar{d}_{N,j}/L_{E_2} & j = 1, \ldots, M \\ d_{i,0} = \bar{d}_{i,0}/L_{E_3} & d_{i,M} = \bar{d}_{i,M}/L_{E_4} & i = 1, \ldots, N \end{array}$$

Ostatecznie kolejno wyznaczamy s oraz t na brzegach przestrzeni parametrycznej

$$\begin{array}{lll} s_{0,j} = 0 & s_{N,j} = 1 & j = 1, \ldots, M \\ t_{i,0} = 0 & t_{i,M} = 1 & i = 1, \ldots, N \end{array}$$

I punkty wewnętrzne jako rozwiązanie układu równań

$$\begin{array}{ll} s_{i,j} = s_{i,0}(1 - t_{i,j}) + s_{i,M} t_{i,j} & \\ t_{i,j} = t_{0,j}(1 - s_{i,j}) + t_{N,j} s_{i,j} & (i,j) \in (1 \ldots N-1, 1 \ldots M-1) \end{array}$$

4.5.4 Siatki brzegowo ortogonalne

Ostatnim z rozważanych tutaj rodzajem siatek kontrolnych są siatki ortogonalne na brzegach. W przypadku takich siatek nie wystarcza już podanie tylko warunku brzegowego typu Dirichleta (czyli określenie wartości współrzędnych węzłów brzegowych), musimy dodatkowo dostarczyć informacji o wektorach normalnej do brzegu (warunek brzegowy Neumanna). Poniższe zależności :

- $s \equiv 0$ na brzegu E1 i $s \equiv 1$ na brzegu E2
- $\partial s / \partial n$ wzdłuż krawędzi E3 i E4
- $t \equiv 0$ na brzegu E3 i $t \equiv 1$ na brzegu E4
- $\partial t / \partial n$ wzdłuż E1 i E2

oraz dwa równania Laplace'a $\Delta s = 0$ i $\Delta t = 0$ określają problem brzegowy dla siatek ortogonalnych na brzegach.

Podobnie jak poprzednio wprowadzamy oznaczenia

$$\begin{array}{llll} s(0,\eta) = 0 & s(1,\eta) = 1 & s(\xi,0) = s^o_{E_3}(\xi) & s(\xi,1) = s^o_{E_4}(\xi) \\ t(\xi,0) = 0 & t(\xi,1) = 1 & t(0,\eta) = t^o_{E_1}(\eta) & t(1,\eta) = t^o_{E_2}(\eta) \end{array}$$

a następnie odwzorowanie kontrolne zdefniujemy jako

$$\begin{aligned} s &= s^o_{E_3}(\xi)H_0(t) + s^o_{E_4}(\xi)H_1(t) \\ t &= t^o_{E_1}(\eta)H_0(s) + t^o_{E_2}(\eta)H_1(s) \end{aligned}$$

gdzie

$$H_0(s) = (1-2s)(1-s)^2 \qquad H_1(s) = (3-2s)s^2 \qquad 0 \leqslant s \leqslant 1$$

Na koniec warto odnotować jak ważną rolę pełnią siatki brzegowo ortogonalne zwłaszcza w metodach generacji siatek typu wieloblokowego. Jak łatwo zauważyć, obiekty o złożonej geometrii nie zawsze można opisać przy użyciu czterech krzywych, wówczas należy podzielić rozpatrywane zadanie na kilka tzn. bloków i generować siatkę dla każdego bloku oddzielnie. Siatki ortogonalne na brzegach zapewniają wtedy gładkie przejście, uzyskanej w ten sposób kompozytowej siatki, pomiędzy sąsiednimi blokami.

Spis literatury

[1] Bobilev,N.A.,Ivanenko,S.A.,Ismailov,I.G, *Some remarks on homeomorphysms*, Russian Mathematical Notes, Vol.60(4), pp.593-596, 1996.

[2] Flugge,W., *Powłoki. Obliczenia statyczne.* Arkady, Warszawa 1972.

[3] Foley,J.,A. Van Dam,S.Feiner, J.Hughes, *Computer Graphics : Principles and Practice*, Second Edition, Addison-Wessley, Reading,MA,1990.

[4] Fortuna,Z.,Macukow,B.,Wąsowski,J., *Metody numeryczne*, Wydawnictwa Naukowo-Techniczne, Warszawa 1998.

[5] Gordon,W.N,Hall,C.A, *Construction of curvilinear coordinate systems and application to mesh generation*, International J.Num.Methods in Eng., Vol.7, pp 461-477, 1973.

[6] Ivanenko,S.A., *Adaptive grids and grids on surfaces*, Comput. Maths. Math.Phys., 33(9), pp.1179-1193, 1993.

[7] Ivanenko,S.A, *Adaptive-harmonic grid generation and its application for numerical solution of the problems with boundary and interior layer*, Comput.Maths.Math.Phys., 35(10), pp.1203-1220, 1995.

[8] Kiciak,P., *Podstawy modelowania krzywych i powierzchni - zastosowania w grafice komputerowej*, Wydawnictwa Naukowo-Techniczne, Warszawa 2000.

[9] Leitner,R., *Zarys matematyki wyższej dla inżynierów Część III: Rachunek całkowy, Równania różniczkowe zwyczajne*, Wydawnictwa Naukowo-Techniczne, Warszawa 1968.

[10] OpenGL Architecture Review Board, *OpenGL Reference Manual*, Addison-Wesley.

[11] Soni,B,K,Yu,T,Y, *The application of NURBS in numerical grid generation*, I.of Computer-Aided Design, Feb.1995, Vol.27, No.2, pp 147-157.

[12] Thompson,J,F,Warsi,Z,U,A,Mastin,C,W, *Numerical Grid Generation : Formulations and Applications*, Elsevier, New York, 1985.

[13] Winslow,A.M., *Numerical solution of quasilinear Poisson equation in nonuniform triangle mesh*, J.Comput.Phys.,1(2), pp.149-172, 1966.

[14] Zienkiewicz,O.C., *Metoda elementów skończonych*, Arkady, Warszawa 1972.