Praca magisterska.

Metody wizualizacji przepływu krwi przez naczynia krwionośne.

Piotr Olkiewicz

1 marca 2015

www.uni.wroc.pl

Wrocław 2015

Podziękowania:

Dziękuje dr M. Matyce - promotorowi mojej pracy magisterskiej, który zawsze służył mi pomocą oraz cennymi wskazówkami. Wyrazy uznania kieruję również do dr W. Tarnawskiego, dzięki któremu praca ta została wykorzystana w przemyśle.

Streszczenie

Celem niniejszej pracy jest opracowanie modelu wizualizacji przepływu krwi przez naczynia krwionośne. Pole prędkości zostało wyliczone poprzez rozwiązanie równań Naviera–Stokesa metodą objętości skończonej z wykorzystaniem zewnętrznej biblioteki SpeedIt Flow. W pracy omówiono podstawy teoretyczne dynamiki płynów. W dalszej części pracy przedstawiono różne modele oświetlenia służące do rzeczywistego przedstawienia obiektów. Przedyskutowano ich podstawy fizyczne oraz wpływ na jakość generowanego obrazu. W kolejnej części opisano techniki przedstawiania pola prędkości takie jak linie prądu, cząstki i powierzchnie prądu. Przetestowano również wpływ doboru algorytmu całkującego i interpolującego pole prędkości na dokładność generowanych wizualizacji.

Visualization methods for flow of blood in human arteries.

Summary

In this thesis work various methods of visualization of velocity field of blood flow in human arteries have been discussed and analyzed. Velocity field was computed by solving Navier-Stokes equations by finite volume method for incompressible fluid with the help of the SpeedIt Flow library. Theoretical foundations of fluid dynamics have been presented and it was shown that appropriately chosen models of lighting can provide more realistic visualizations. Physical principles and their impact on quality of rendering pictures have been also discussed. In the last part methods for visualization of the velocity field such as streamlines, particles and stream surfaces have been described. Moreover, various algorithms for interpolation and integration of the velocity field and their impact on accuracy of generated visualization have been also tested.

Spis treści

1 Wstęp

XX wiek dał się poznać, jako okres w którym zaobserwowano bardzo wyraźny wzrost zachorowalności na choroby układu krążenia [13, 14]. Choroby te porównywane do epidemii współczesnej cywilizacji odpowiadają za prawie 50% zgonów w Polsce. Choroby występujące na tle miażdżycowym są główną przyczyną umieralności i niepełnosprawności na świecie, a także jedną z głównych przyczyn hospitalizacji. Z badań (WHO - World Health Organization) wynika, że choroba niedokrwienia serca oraz choroby naczyń mózgowych są największym zagrożeniem dla ludzkości. W Polsce rozpowszechnione są patologiczne zachowania sprzyjające występowaniu tych chorób, chociażby nałogowe palenie tytoniu, nadwaga, otyłość, mała aktywność fizyczna i przewlekły stres. Powodem dużej śmiertelności spowodowanej chorobami układu krążenia jest brak skutecznych metod diagnostycznych. Obraz naczyń krwionośnych możemy uzyskać chociażby ze zdjęć wykonanych za pomocą tomografu. Dzięki uzyskanym zdjęciom można zweryfikować, czy pacjent choruje na miażdżyce. Niestety nie można w oczywisty sposób uzyskać informacji dotyczących przebiegu choroby oraz czasu, kiedy może dojść do wylewu. Badając przepływ krwi możemy określić ilość odkładanego cholesterolu oraz to w jakim stopniu ona wpływa na ryzyko wylewu. Wykonanie takiego badania obarczone jest jednak wieloma trudnościami, tj.:

- dokładnym zbadaniem przepływu krwi, czyli cieczy transportującej różne substancje chemiczne,
- zmianą geometrii naczyń pod wpływem przepływu,
- trudnością z przeprowadzaniem eksperymentów w celu pomiaru,
- zaproponowaniem modelu który będzie opisywał to zjawisko.

Innowacyjnym podejściem do diagnostyki medycznej jest oprogramowanie Virtus firmy Vratis, które bazując na zdjęciach potrafi generować geometrię naczyń, a następnie uruchomić symulacje komputerową przepływu krwi w celu wyznaczenia różnych charakterystyk płynu, np. ciśnienia, prędkości czy naprężeń stycznych na powierzchni ścian naczyń.

Głównym celem tej pracy jest opracowanie modelu wizualizacji przepływu krwi przez naczynia krwionośne do aplikacji Virtus. Dane do wizualizacji zostaną policzone przy pomocy biblioteki SpeedIt Flow. Jest to produkt firmy Vratis, służący do rozwiązania równań Naviera-Stokesa metodą objętości skończonej. SpeedIt Flow jest w całości zaimplementowany w technologi CUDA, co znacznie skraca czas obliczeń. Poprawność uzyskanego rozwiązania została porównana z oprogramowaniem OpenFoam [16]. Na podstawie wyników symulacji komputerowej

przepływu krwi przez naczynia można określić naprężenie, ciśnienie działające na ściankę naczynia oraz prędkość krwi. Znajomość powyższych parametrów jest przydatna w ocenie stopnia zagrożenia pęknięcia naczynia krwionośnego w najbliższym czasie. Może wspomóc to podjęcie decyzji czy i w jaki sposób pacjent powinien być operowany. Kolejnym zastosowaniem tego modelu jest diagnostyka chorób układu krwionośnego, tj. miażdżyca lub zwyrodnienia naczyń krwionośnych. Temat ten został poruszony między innymi w [1].

W tej pracy opiszę podstawy teoretyczne dynamiki płynów i algorytmów zastosowanych do wyznaczenia pola prędkości płynu przepływającego przez dany ośrodek. Do opisu dynamiki płynów używam równań Naviera-Stokesa, opisujących zasadę zachowania masy i drugą zasadę dynamiki dla poruszającego się płynu, np. krwi.

W celu efektywnej wizualizacji rozważę kilka modeli oświetlenia, które mają za zadanie rzeczywiste przedstawienie danych uzyskanych z symulacji komputerowych. W celu uzyskania pola wektorowego oraz skalarnego z dyskretnych punktów siatki w których rozwiązano równania Naviera-Stokesa zastosuję algorytmy interpolujące. Sądzę, że wybór metody interpolującej ma istotny wpływ na czas generowania wizualizacji oraz realizm otrzymanych animacji komputerowych. W następnej części przedstawię algorytmy służące do całkowania pola prędkości po czasie, dzięki którym można tworzyć wizualizację przepływu przez naczynie. Przeprowadzę testy skuteczności omawianych metod na przykładzie generowania linii prądu.

2 Opis matematyczny dynamiki płynów

Z makroskopowego punktu widzenia przyjmuje się podział materii na ciała stałe i płyny (ciecze i gazy). Możemy rozróżnić kilka ogólnych parametrów charakteryzujących ruch (przepływ) płynów.

- Przepływ może być stacjonarny, bądź niestacjonarny, ruch płynu jest stacjonarny, jeśli w dowolnym punkcie przestrzeni prędkość płynu nie zależy od czasu.
- Przepływ może być ściśliwy lub nieściśliwy. Przepływem nieściśliwym nazwiemy przepływ w którym gęstość płynu jest stała. Można poglądowo uznać, iż nieściśliwym jest przepływ cieczy, natomiast ściśliwym - przepływ gazów.
- Przepływ może być lepki albo nielepki. Lepkość w płynach jest tarciem wewnętrznym. Wyjaśnieniem przyczyn lepkości są siły styczne występujące pomiędzy warstwami cieczy, które przesuwają się względem siebie. Lepkość jest odpowiedzialna za dyssypację, czyli stopniowe rozpraszanie energii mechanicznej w danym ośrodku.
- Przepływ, w którym nachodzą na siebie kolejne warstwy płynu nazywamy burzliwym(turbulentnym), wtedy możemy zaobserwować ruchy wirowe i chaotyczne. Przeciwieństwem jest przepływ laminarny, w którym strugi płynu nie mieszają się.

Teoretyczny model dla lepkości płynu zaproponował Isaac Newton, w którym naprężenie wynosi:

$$\bar{\tau} = \mu \nabla \mathbf{u} \tag{1}$$

gdzie:

- $\mathbf{u}$ - wektor prędkości,
- μ - współczynnik lepkości dynamicznej,
- $\bar{\tau}$ - tensor naprężenia.

Płyn opisywany powyższym modelem nazywamy idealnie lepkim lub niutonowskim.
W tej pracy badam przepływ laminarny i turbulentny dla cieczy jednofazowej, nieściśliwej i niutonowskiej dla naczyń z sztywnymi ściankami.

2.1 Równania Naviera-Stokesa

Równania Naviera-Stokesa stanowią zestaw równań opisujących zasadę zachowania masy drugą zasadę dynamiki dla poruszającego się płynu. Według nich zmiana pędu elementu płynu zależy

jedynie od zewnętrznego ciśnienia i wewnętrznych sił lepkości oraz sił zewnętrznych, na przykład grawitacji. Te klasyczne równania różniczkowe cząstkowe, intensywnie badane na początku XIX wieku, dały podstawy naszej wiedzy o hydromechanice, propagacji fal, przewodnictwie cieplnym i innych istotnych problemach fizycznych. Rozwiązaniu tego problemu poświęcało się wielu matematyków oraz fizyków, co zaowocowało powstaniem nowych metod badawczych w tych dziedzinach. W rezultacie w XX wieku nastąpił ogromny rozwój teorii równań różniczkowych cząstkowych. Niestety szereg fundamentalnych problemów związanych z istnieniem, jednoznacznością i asymptotycznym zachowaniem się rozwiązań pozostał nierozwiązany. Ruch turbulentny wody czy tworzenie się wirów są dwoma zjawiskami, których zrozumienie przysparza wiele trudności współczesnym naukowcom.

Rozwiązanie tego zagadnienia można znaleźć na drodze rachunku różniczkowego i całkowego. W taki sposób potrafimy rozwiązać jedynie najprostsze przypadki nieturbulentnego, stacjonarnego przepływu o regularnej geometrii, charakteryzujący się niską liczbą Reynoldsa. Liczbę Reynoldsa definiujemy:

$$R = \frac{\mathbf{u}l}{\nu} \tag{2}$$

gdzie:

- $\mathbf{u}$ - prędkość charakterystyczna płynu, np. średnia prędkość w badanym obszarze,
- l - wymiar charakterystyczny dla badanego zjawiska od którego zależy stateczność przepływu. (Na przykład dla rury jest to średnica.)
- ν - lepkość kinetyczna płynu.

Równania Naviera-Stokesa dla jednofazowej, lepkiej i nieściśliwej cieczy mają postać:

$$\nabla \cdot (\rho \mathbf{u}) = 0 \tag{3}$$

$$\rho \frac{D\mathbf{u}}{Dt} = -\nabla p + \mu \nabla^2 \mathbf{u} + \mathbf{F} \tag{4}$$

gdzie:

- $\mathbf{u}$ jest polem prędkości,
- ρ - gęstością,
- p - ciśnieniem,
- $\mathbf{F}$ - zewnętrzną siłą,

- μ - współczynnikiem lepkości dynamicznej.

Istnieją dwa sposoby opisu ruchu płynów:

- opis Lagrange'a,
- opis Eulera.

2.2 Opis Lagrange'a

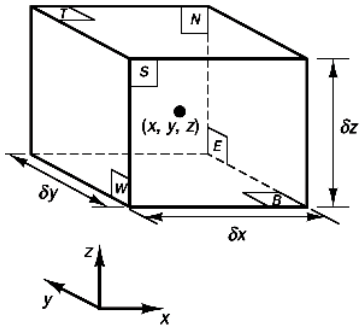
Podejście zaproponowane przez J. L. Lagrange'a zakłada podział płynu na infinitezymalne elementy objętości, zwane cząstkami płynu. Ta metoda służy do analizy zmienności prędkości, przyśpieszenia, ciśnienia, gęstości i temperatury indywidualnie dla każdego elementu płynu wzdłuż toru jego ruchu. W tym podejściu wielkości fizyczne takie jak gęstość ciśnienie itp. wyznaczamy poprzez obliczanie całek wzdłuż trajektorii cząstek płynu. Możemy sobie wyobrazić, że obserwator porusza się razem z każdą cząstką i śledzi zmiany wielkości fizycznych i na tej podstawie określa stan całego układu.
To podejście nie będzie jednak poruszane w dalszej części pracy. Dokładniejszy opis można znaleźć w [12].

2.3 Opis Eulera

W podejściu L. Eulera stan układu opisany jest przez pole prędkości, gęstość, i ciśnienie rozpięte w przestrzeni i zależne od czasu. Dla tej metody wyznaczamy obszar kontrolny, który posiada infinitezymalne rozmiary i badamy zmiany parametrów hydrodynamicznych dla tego obszaru. Obserwator w tej metodzie bada określony element przestrzeni, w przeciwieństwie do metody Lagrange'a gdzie poruszamy się za cząstkami płynu. W mojej pracy stosuję podejście Eulera i równania Navera-Stokesa do opisu dynamiki płynów.

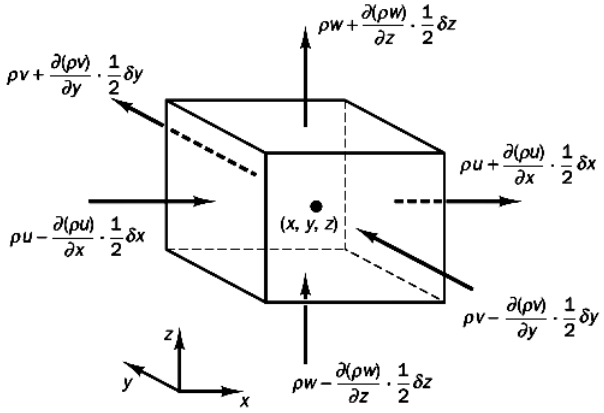
2.3.1 Równanie zachowania masy dla płynu

W celu lepszego zrozumienia z jakich rozważań wynikają równania Naviera-Stokesa i jak należy rozumieć opis Eulera dla dynamiki płynów przedstawię w jaki sposób można wyprowadzić równanie zachowania masy dla płynu. Na początek rozważmy mały element płynu o bokach δx,δy oraz δz, jak na poniższym rysunku:



Rysunek 1: Opis elementu płynu [2].

Środek elementu jest zlokalizowany w pozycji *(x,y,z)*. Masa płynu jest zachowana podczas przepływu, czyli zmiana gęstości w elemencie plus strumień przechodzący przez element musi być równe 0. Formuła przedstawiająca strumień przechodzący przez element wygląda następująco:



Rysunek 2: Przepływ masy przez element płynu [2].

$$
\begin{aligned}
Q = &\left(\rho u - \frac{\partial \rho u}{\partial x}\frac{1}{2}\delta x\right)\delta y\delta z - \left(\rho u + \frac{\partial \rho u}{\partial x}\frac{1}{2}\delta x\right)\delta y\delta z \\
&+ \left(\rho v - \frac{\partial \rho v}{\partial y}\frac{1}{2}\delta y\right)\delta x\delta z - \left(\rho v + \frac{\partial \rho v}{\partial y}\frac{1}{2}\delta y\right)\delta x\delta z \\
&+ \left(\rho w - \frac{\partial \rho w}{\partial z}\frac{1}{2}\delta z\right)\delta x\delta y - \left(\rho w + \frac{\partial \rho w}{\partial z}\frac{1}{2}\delta z\right)\delta x\delta y
\end{aligned} \tag{5}
$$

Z powyższych rozważań wynika równanie:

$$
\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} + \frac{\partial(\rho w)}{\partial z} = 0 \tag{6}
$$

Zapisując je w postaci wektorowej otrzymuję równanie ciągłości dla ściśliwego płynu:

$$
\nabla \cdot (\rho \mathbf{u}) + \frac{\partial \rho}{\partial t} = 0 \tag{7}
$$

Równanie ciągłości dla nieściśliwego płynu dla którego $\frac{\partial \rho}{\partial t} = 0$ upraszcza się do postaci:

$$\nabla \cdot \mathbf{u} = 0 \tag{8}$$

2.3.2 Druga zasada dynamiki

Drugim równaniem Naviera-Stokesa jest druga zasada dynamiki dla poruszającego się płynu, którą można wyprowadzić rozważając w jaki sposób siły lepkości, ciśnienia i zewnętrzne działają na element płynu. Ogólne równanie Naviera-Stokesa wygląda następująco:

$$\rho \frac{D\mathbf{u}}{Dt} = -\nabla p + \nabla \cdot (\mu \nabla \mathbf{u}) + \mathbf{F} \tag{9}$$

2.4 Przepływ turbulentny

Powyżej pewnej krytycznej wartości liczby Reynoldsa obserwujemy wyraźną zmianę w naturze przepływu. Strugi płynu przestają być równoległe i zaczynają się mieszać. Prowadzi to do zmiany charakteru przepływu, który staje się chaotyczny i jest określany jako turbulentny. Obserwujemy wtedy fluktuacje parametrów fizycznych tj. prędkość, ciśnienie itp. Dla takiego układu możemy zapisać:

$$\mathbf{u} = \mathbf{U} + \mathbf{u}' \tag{10}$$

$$p = P + p' \tag{11}$$

gdzie:

- **U**,P - są to odpowiednio średnia prędkość i ciśnienie.
- **u**', p' są to odpowiednio zaburzenia prędkości i ciśnienia.

W związku z wprowadzeniem fluktuacji wielkości fizycznych musimy zmodyfikować równania, które opisują przepływ. Dokładne wyprowadzanie można znaleźć w [2].

2.5 Metoda objętości skończonych

Równania różniczkowe cząstkowe opisane we wcześniejszym paragrafie można rozwiązać wieloma algorytmami numerycznymi wymagającymi dyskretyzacji w czasie i przestrzeni. Jednym z nich jest metoda objętości skończonej. Podstawy tej metody omówię na przykładzie równania adwekcji.

$$\frac{\partial \phi}{\partial t} + \nabla \cdot \mathbf{F} = 0 \tag{12}$$

gdzie:

- ϕ - pole skalarne, na przykład temperatura, którego zmianę chcemy badać,
- $\mathbf{F}$ - strumień (tu wynosi $\mathbf{u}\phi$.)

Powyższe równanie dla ogólnego przypadku jest trudno rozwiązać. Zagadnienie to można uprościć i przyjąć, że nasze rozwiązanie będzie spełniać równanie w sposób całkowy dla określonych komórek. Jest to podstawowe założenie na podstawie, którego została zbudowana metoda objętości skończonej. Teraz możemy szukać rozwiązania ograniczając się do wybranego elementu objętości i zapisać równanie adwekcji w formie całkowej.

$$\frac{\partial}{\partial t}\int_{\Omega}\phi d\Omega + \int_{\Omega}\nabla\cdot\mathbf{F}d\Omega = 0 \tag{13}$$

Zakładając, że ϕ jest stałe w elemencie objętości oraz przechodząc z całki objętościowej do powierzchniowej otrzymujemy:

$$\frac{\partial}{\partial t}\phi V + \int_{\partial}\mathbf{F}\cdot\mathbf{n}dS = 0 \tag{14}$$

gdzie:

- dS - element powierzchni,
- V - objętość komórki elementarnej,
- $\mathbf{n}$ wektor normalny,
- Ω obszar kontrolny,
- ∂ brzeg obszaru kontrolnego.

Teraz możemy z całką przejść do sumy po ściankach elementu kontrolnego oraz pochodną po czasie przybliżyć wzorem Eulera w przód.

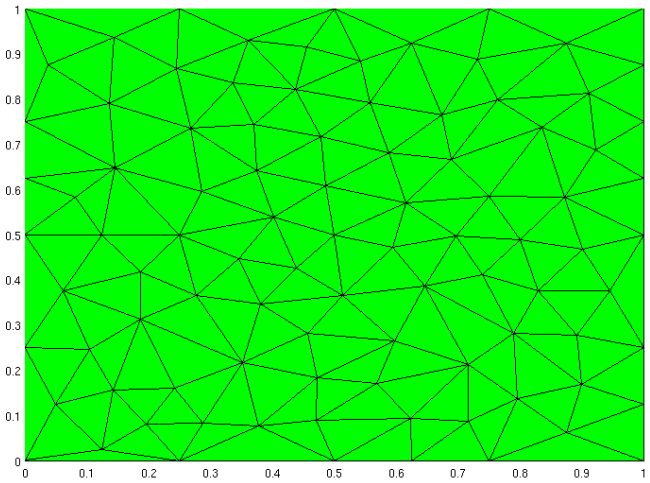
$$\frac{V}{\Delta t}(\phi^{n+1}-\phi^{n}) + \sum_{s}\mathbf{n}_s\cdot\mathbf{F}_s = 0 \tag{15}$$

Powyższy zapis oznacza że zmiana wartości pola w czasie w komórce jest równa strumieniowy przechodzącemu przez ten element. Jest to bardzo intuicyjne dla wszystkich wielkości fizycznych, które powinny być zachowane. Kolejnym krokiem jest przeprowadzenie dyskretyzacji obszaru w którym szukamy rozwiązania. Oznacza to, że musimy stworzyć siatkę dla której będziemy rozwiązywali równania. Następnym ważnym elementem jest sposób liczenia strumienia na ściance ($\mathbf{F}_s$), w zależności od wyboru algorytmu otrzymujemy różne schematy metody objętości skończonej. Najprostszym z nich jest upwind, w którym strumień przechodzący przez ściankę

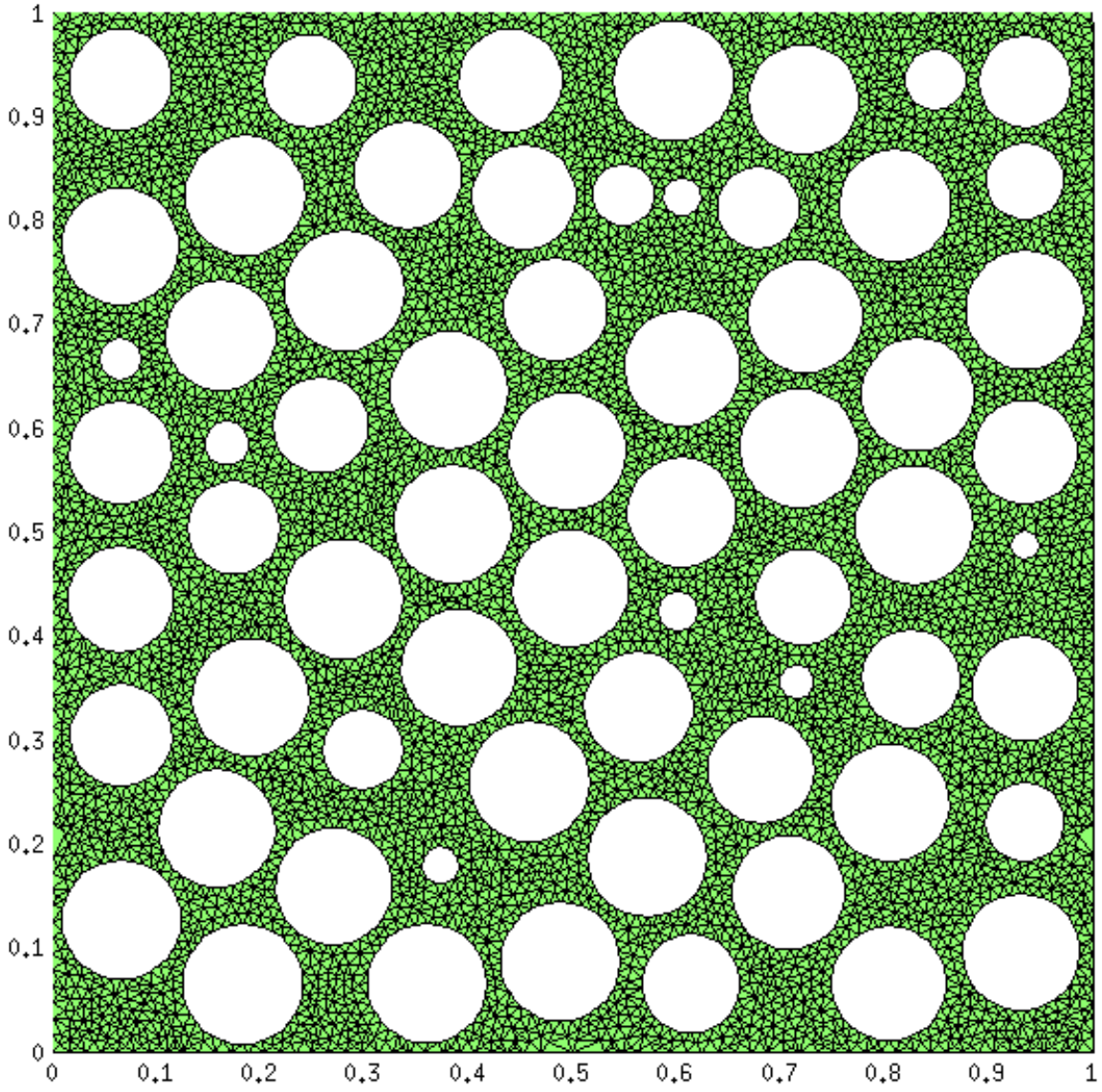
jest równy prędkości na ściance pomnożonej przez wartość pola komórki, z której wychodzi strumień. Innym podejściem jest obliczenie średniej arytmetycznej pola z sąsiednich komórek, wtedy otrzymamy schemat drugiego rzędu nazywany FTCS(forward time, center space), który jest niestabilny, dlatego należy go stosować z tzw. ogranicznikami strumienia. Wybierając sposób interpolacji strumienia i metodę całkowania w czasie tworzymy schemat metody objętości skończonej. Dokładniejsze rozważania na ten temat można znaleźć na przykład w [2, 3, 4, 5].

2.6 Rozwiązywanie równań N-S

Równania N-S w mojej pracy zostały rozwiązane przy użyciu biblioteki SpeedIt Flow, która udostępnia standardowe algorytmy umożliwiające rozwiązanie równania metodą objętości skończonej, popularnie z języka angielskiego nazwane solverami, np. ICO, PISO i SIMPLE, które są zaimplementowane na procesach graficznych (gpu). Interesującym pakietem zawierającym implementacje powyższych algorytmów na standardowych procesach (cpu) jest OpenFoam, ponieważ kody źródłowe dla tego pakietu są otwarte i każdy użytkownik może się z nimi zapoznać. W kolejnym rozdziale przedstawię w jaki sposób przygotowuje się symulacje komputerową przepływu.



Rysunek 3: Przykładowe obszary kontrolne dla siatki 2D w metodzie FVM wygenerowane przy pomocy programu triangle [21].



Rysunek 4: Przykładowe obszary kontrolne dla siatki 2D w metodzie FVM wygenerowane przy pomocy programu triangle [21].

3 Wykonanie symulacji komputerowej przepływu krwi.

Wykonanie symulacji przepływu krwi jest bardzo złożonym procesem. Najpierw trzeba uzgodnić model przepływu i uproszczenia na które się godzimy, aby uprościć badane przez nas zjawisko i skoncentrować się na tych problemach, które są dla nas istotne. Kolejnym etapem jest wybranie algorytmu, który zastosujemy do rozwiązania równań. W tym momencie należy pamiętać, że dla dowolnej geometrii nie da się w sposób dokładny wyznaczyć prędkości i ciśnienia. Metody numeryczne, których używamy, tj. FVM stosują kolejne przybliżenia. Ważnym elementem jest zrozumienie tego na jakie efekty pomijamy oraz jakim błędem są obarczone nasze obliczenia. Dopiero wtedy możemy analizować otrzymane przez nas rozwiązanie i na jego podstawie wyciągnąć wnioski dotyczące rzeczywistego przepływu krwi.

3.1 Wygenerowanie geometrii

Geometrię naczyń krwionośnych ciężko otrzymać z bezpośrednich metod. Zazwyczaj pomiar na naczyniu wykonuje się w ostateczności kiedy przygotowuje się stent, który musi być idealnie dopasowany do naczynia. Niestety taki zabieg jest inwazyjny, dlatego warto poszukać innych rozwiązań. Obraz naczyń można otrzymać dzięki użyciu tomografu. Na podstawie zdjęć wykonanych tomografem można określić czy istnieje zwyrodnienie naczynia takie jak tętniak lub miażdżyca, ale w celu uzyskania geometrii musimy przeprowadzić komputerową analizę obrazu. Poniżej przedstawiłem jedynie skrócony opis złożonych procesów które w tym czasie zachodzą.

- W pierwszej kolejności następuje wczytanie zdjęć z tomografu. Tomograf umożliwia robienie zdjęć w danej płaszczyźnie, a następnie przesuwa się prostopadle do płaszczyzny wzdłuż, której robi zdjęcie. W ten sposób powstaje seria zdjęć, którą możemy wczytać do komputera.

- Następnym etapem jest budowanie siatki powierzchniowej. Jest to najtrudniejsza część tego algorytmu, ponieważ na podstawie obrazu musimy budować trójkąty. Metoda zazwyczaj bazuje na znajdywaniu podobnych do siebie pixeli, tworzeniu na nich trójkątów i łączeniu w całość. Należy pamiętać, że siatka musi cały czas być spójna, nie może posiadać dodatkowych trójkątów w środku i innych artefaktów. Następnie musimy zadbać, aby powierzchnia siatki była gładka, gdyż w innym przypadku nie będziemy w stanie policzyć na niej symulacji przepływu.

- Budowanie siatki objętościowej. Na podstawie siatki powierzchniowej można stworzyć siatkę objętościową. Standardowe algorytmy opierają się na wstawianiu punktów do środka siatki a następnie na budowaniu czworościanów lub innych brył. Ostatnim etapem

jest wygładzanie, czyli poprawianie komórek, które nie spełniają naszych oczekiwań na przykład są za duże lub za płaskie. Stworzenie generatora, który działa szybko i generuje ładne siatki objętościowe jest sporym wyzwaniem. Na szczęście na rynku istnieje sporo darmowych generatorów, które obsługują format stl. Ja korzystam z netgena, tetgena i snappyHexMesh. Dla zainteresowanych więcej informacji w dodatkach, przy opisie zewnętrznego oprogramowania.

3.2 Zadanie warunków początkowych i brzegowych

Na tym etapie mamy już siatkę objętościową z zdefiniowanymi komórkami. Kolejnym etapem jest zadanie warunków początkowych i brzegowych dla symulacji przepływu krwi. Można rozróżnić trzy typy ścianek dla naczynia.

- wlot - w tym miejscu wpływa płyn,
- wylot - w tym miejscu wypływa płyn,
- brzeg - są to ścianki naczynia, które nie przepuszczają płynu.

Dla każdego typu ścianki powinniśmy zadać inne warunki. Warto wspomnieć, że rodzaj zadanego warunki ma bardzo istotny wpływ na wyniki symulacji i powinno to się robić bardzo dokładnie. Musimy określić jakie są warunki początkowe i brzegowe dla pól ciśnienia i prędkości. Warunki te są ustalane dla wyżej wspomnianych typów ścianek. Zazwyczaj dla prędkości przyjmuje się następujące wartości:

- wlot - warunek Dirichleta - prędkość jest stała lub zmienna w czasie,
- wylot - warunek Neumanna - pochodna prędkości $= 0$,
- brzeg - warunek Dirichleta - prędkość $= 0$.

Dla ciśnienia:

- wlot - warunek Neumanna - pochodna $= 0$,
- wylot - warunek Dirichleta - ciśnienie wynosi 0,
- brzeg - warunek Neumanna - pochodna $= 0$.

Kolejnym etapem jest wybranie jakich schematów dla metody objętości skończonej chcemy użyć. Jest to znowu obszerne zagadnienie, ponieważ dla każdego operatora różniczkowego możemy użyć innego. Tutaj ograniczę się do przedstawia przykładowych jakich ja użyłem:

- pochodna po czasie - metoda Eulera krok w przód,
- gradient - metoda interpolacji liniowa,
- dywergencja - metoda interpolacji liniowa,
- laplacian - metoda interpolacji liniowa.

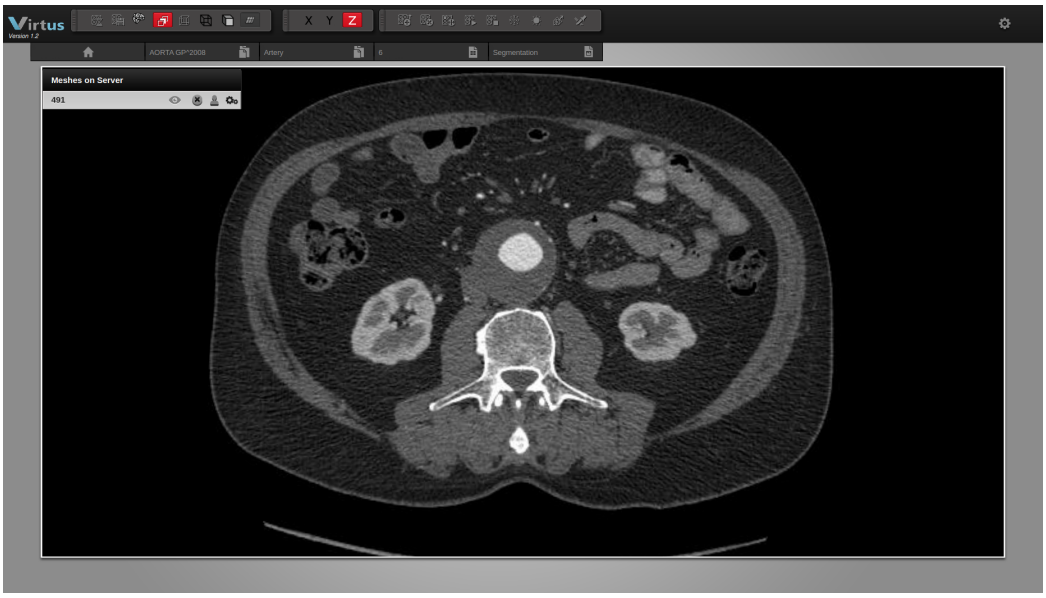
3.3 Uruchomienie symulacji

Ostatnim etapem jest uruchomienie symulacji przepływu. Sprowadza się to do uruchomienia programu, który na podstawie plików kontrolnych wylicza pole prędkości i ciśnienia w kolejnych momentach i zapisuje je do plików.

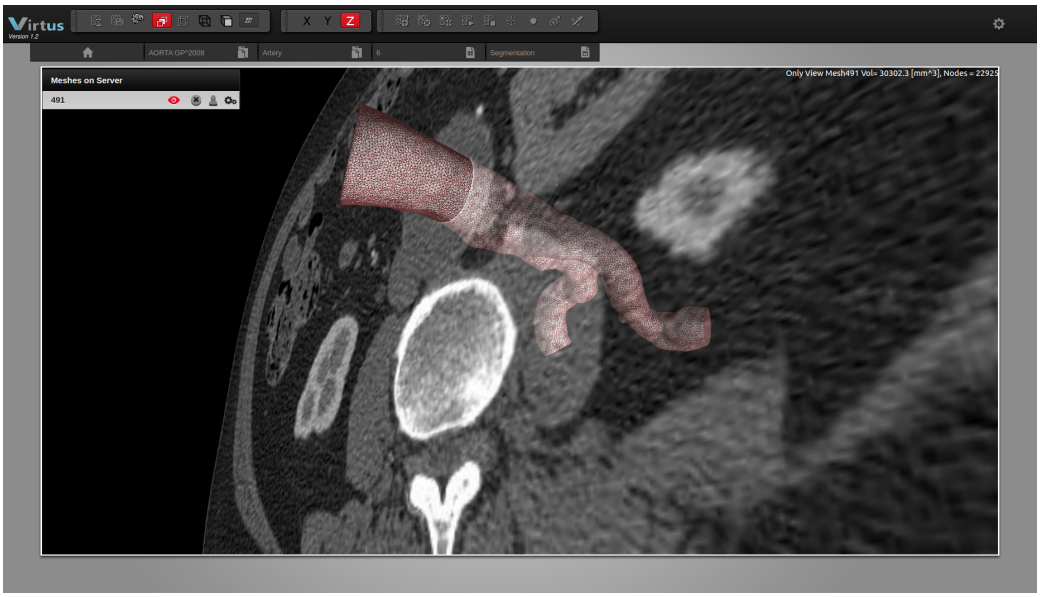
3.4 Virtus

Do opisanych w tym rozdziale kroków użyłem programu Virtus, który jest produktem firmy Vratis umożliwiający przeprowadzenie poszczególnych kroków opisanych w tym rozdziale. Jest to wtyczka do przeglądarki komunikująca się z serwerem działającym w architekturze rozproszonej. Główną ideą tego oprogramowania jest przeprowadzania analizy przepływu krwi przez naczynia. Virtus składa się z następujących modułów:

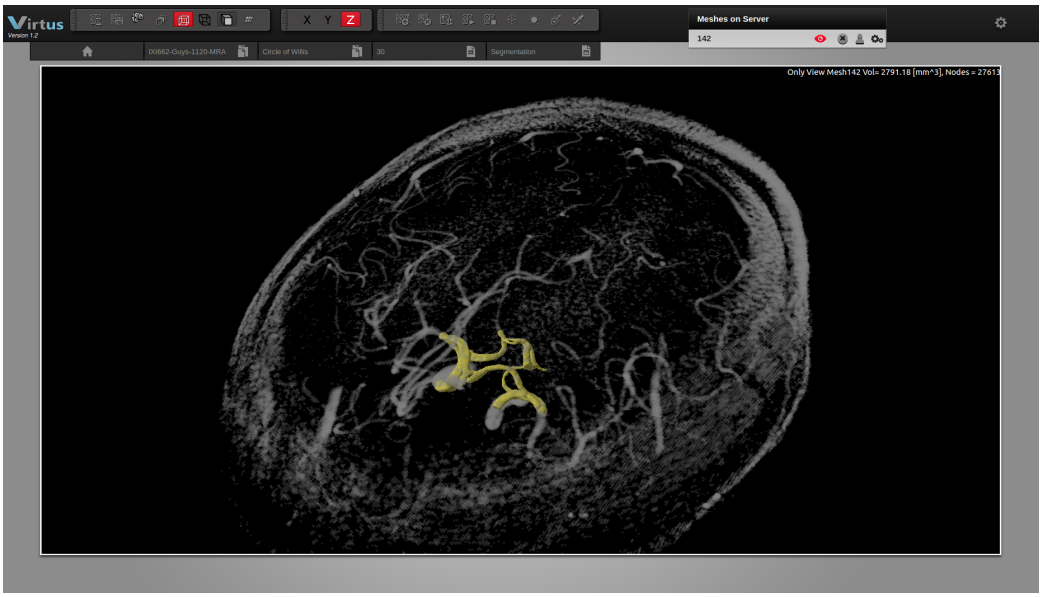
- pobieranie danych medycznych z serwerów szpitalnych,
- tworzenie wizualizacji 3D naczyń krwionośnych,
- segmentowanie geometrii na podstawie obrazów,
- generowanie siatki objętościowej i zadawanie warunków do symulacji,
- uruchomienie symulacji przepływu na klastrze komputerowym. Poniżej możemy zobaczyć zdjęcia z Virtusa przedstawiające powyższe funkcjonalności.



Rysunek 5: Obraz tomograficzny tętnicy brzusznej.



Rysunek 6: Obraz tomograficzny tętnicy brzusznej z siatką powierzchniową.



Rysunek 7: Wizualizacja 3D koła Willisa z siatką powierzchniową.

4 Wizualizacja

Z rozwiązania równań Naviera-Stokesa otrzymujemy wartości prędkości i ciśnienia w komórkach w kolejnych krokach czasowych. Dodatkowo w kolejnych etapach możemy policzyć wartości pochodne takie jak naprężenie powierzchniowe (WSS) i stężenie tłuszczy o niskiej gęstości (LDL). Jest to olbrzymi zbiór danych, który trzeba przeanalizować. Najprostszym podejściem jest narysowanie geometrii i pokolorowanie jej według wartości analizowanego pola. Wizualizacja składa się z kilku części. Najpierw powinniśmy przedstawić geometrię naczynia. W tym celu musimy zdefi niować model oświetlenia, czyli jakim światłem jest oświetlona geometria i w jakim stopniu światło jest odbijane od powierzchni naczynia.

4.1 Modele oświetlenia

Modele oświetlenia dają spory wkład do wyglądu wizualizacji naczyń krwionośnych. Jest to szczególnie pożądany efekt, gdyż, dzięki niemu wszelkie wizualizacje wyglądają bardziej realistycznie. W celu interpretacji wyników otrzymanych z symulacji komputerowych musimy je w jakiś sposób zaprezentować. Idealnym przypadkiem jest przedstawienie pola skalarnego np. ciśnienia w 2D. Wtedy punkt z przestrzeni odpowiada pikselowi, a kolor piksela określa wartość pola. Niestety ogólnie jest to dużo bardziej złożony problem, ponieważ punkt w przestrzeni 3D określamy poprzez trzy liczby x,y,z i aby odwzorować go na ekranie wprowadza się bufor głębokości. Następnie, aby otrzymać cienie, odbicia i inne zjawiska optyczne musimy uwzględnić światło podające na nasz obiekt. W tym celu wprowadza się różne modele oświetlenia.

4.1.1 Światło otoczenia

Za najprostszy rodzaj światła można uznać światło otoczenia. Za jego charakterystyczne cechy uznajemy równomierne rozpraszanie, które nie zależy od źródła światła na powierzchni przedmiotu oraz odbijanie się od przedmiotów znajdujących się w jego zasięgu, które powoduje rozjaśnianie przedmiotów znajdujących się dookoła.

Prostym przykładem światła otoczenia jest światło bezkierunkowe padające na nieoświetloną scenę na której znajdują się przedmioty jednakowo ze wszystkich stron oświetlone. Charakterystyczne dla światła otoczenia jest to, iż tworzy ono liczne odbicia od powierzchni znajdujących się w scenie nie oświetlając żadnego przedmiotu bezpośrednio. Równanie dla światła otoczenia przy założeniu, że światło to w każdym z kierunków pada w ten sam sposób, równomiernie oświetlając przedmioty wygląda następująco:

$$I = k_a I_a \tag{16}$$

przy czym I_a jest natężeniem światła otoczenia, o którym zakłada się, że jest stałe dla wszystkich obiektów. Natężenie tak odbitego od powierzchni światła wyznaczamy za pomocą współczynnika $k_a \epsilon [0, 1]$, który jest cechą materiału. W ten sposób opisałem najprostszy model oświetlenia, który często wchodzi w skład bardziej zaawansowanych modeli. Warto zauważyć, że taki opis światła nie ma podstaw fizycznych. Wprowadzamy go ze względu na trudność w policzeniu skomplikowanych i wielokrotnych trajektorii odbić promieni świetlnych.

4.1.2 Model Lamberta

Główną wadą światła otoczenia jest to, że nie uwzględnia geometrii przedmiotów, przez co nie możemy zaobserwować na przykład cieni. Każdy element bez względu na kształt świeci w ten sam sposób, co jest sprzeczne z doświadczeniem. Z optyki wiadomo, że natężenie światła odbitego od powierzchni jest proporcjonalne do kosinusa kąta padania. Powyższa zasada została sformułowana przez Lamberta stąd uwzględnienie tego zjawiska w grafice komputerowej zostało nazwane od nazwiska twórcy modelem Lamberta. W ten sposób jesteśmy w stanie opisać odbicie światła od matowych powierzchni, równanie światła można zapisać w postaci:

$$I = I_a k_a + I_d k_d cos(\beta) \tag{17}$$

gdzie:

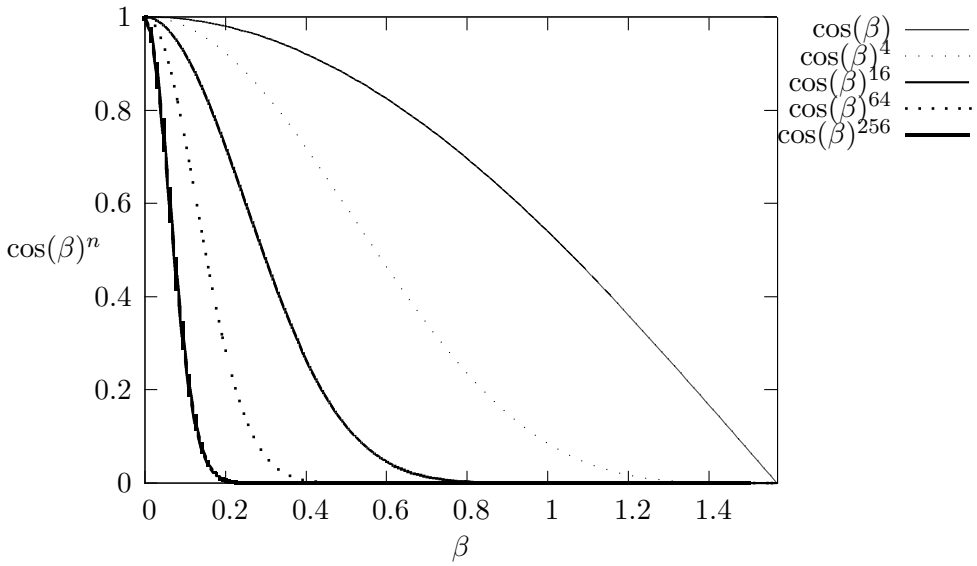
- I_d - natężenie światła rozproszonego,jest takie samo dla wszystkich obiektów,
- k_d - stała materiałowa określająca w jakim stopniu dany obiekt odbija światło rozproszone. Zawiera się w przedziale [0,1].

4.1.3 Model Phonga

W 1975 roku Phong Bui-Tuong opisał w rozprawie doktorskiej innowacyjny model oświetlenia, który opiera się na:

- maksimum odbicia zwierciadlanego występuje dla β równego zero,
- maksimum odbicia zwierciadlanego szybko spada ze wzrostem kata β.

Zanik natężenia światła z wzrostem β jest przybliżany funkcją $cos^n(\beta)$, gdzie wykładnik n zależy od materiału i może przyjmować wartości z zakresu 1 do kilkuset.



Rysunek 8: Przykładowe wartości funkcji $cos^n(\beta)$.

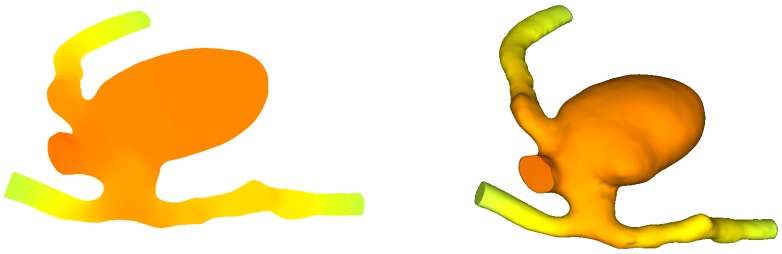
W tym modelu natężenie światła odbitego oblicza się ze wzoru:

$$I = I_a k_a + I_d k_d cos(\beta) + I_s k_s cos(\beta)^n$$

gdzie:

- I - całkowite natężenie światła,
- I_a - natężenie światła otoczenia,
- k_a - stała materiału odpowiedzialna za odbicie światła otoczenia,
- I_d - natężenie światła rozproszonego,
- k_d - stała materiału odpowiedzialna za odbicie światła rozproszonego,
- I_s - natężenie światła rozbłysku,
- k_s - stała materiału odpowiedzialna za intensywność rozbłysku,
- β - kąt między promieniem padającym na powierzchnię, a wektorem normalnym do powierzchni,
- n - stała dla materiału odpowiedzialna za siłę rozbłysku.

Obrazek z lewej strony jest oświetlony tylko przez światło otoczenia, z kolej z prawej strony został zastosowany model Phonga. Widzimy wyraźną różnicę w cieniach. Wynika ona z zastosowania światła rozproszonego.



Rysunek 9: Wizualizacja profilu ciśnienia dla tętniaka naczynia mózgowego. Lewy - tylko z światłem otoczenia, prawy - model Phonga.

4.2 Wizualizacje geometrii

Pierwszym etapem jest wizualizacja geometrii.Poniżej znajdują się charakterystyki siatek badanych w tej pracy.

4.2.1 Aorta podstawna

Jest to część koła Willisa, która doprowadza krew do mózgu. Przepływ dla tego przypadku jest turbulentny i zastosowano model k-omega [2]. Siatka składa się z 160745 punktów.

4.2.2 Tętniak naczynia mózgowego

Jest to tętniak naczynia w pobliżu koła Willisa. Dla tego przypadku przepływ jest turbulentny i zastosowano model k-omega [2]. Siatka składa się z 13891 punktów.

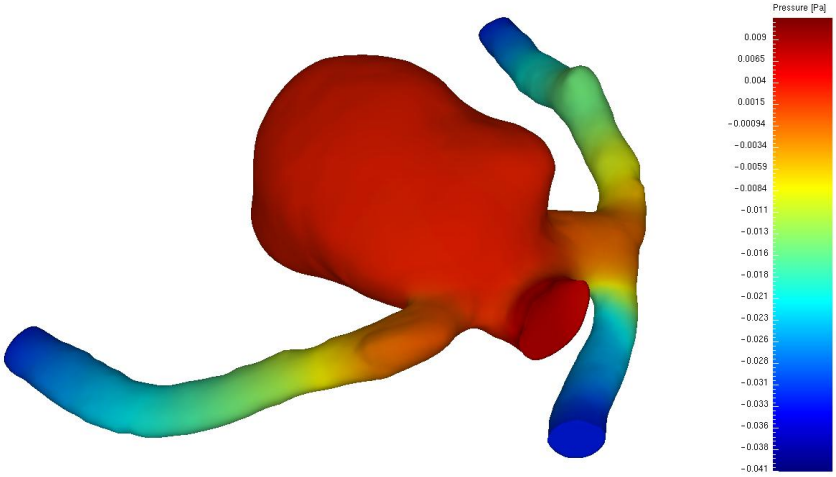
4.2.3 Naczynie wieńcowe

Siatka dla tego przypadku składa się z 52000 punktów. Przepływ jest laminarny. Zbiór rysunków dla poszczególnych geometrii znajduje się w dodatkach.

4.3 Wizualizacja wielkości fizycznych

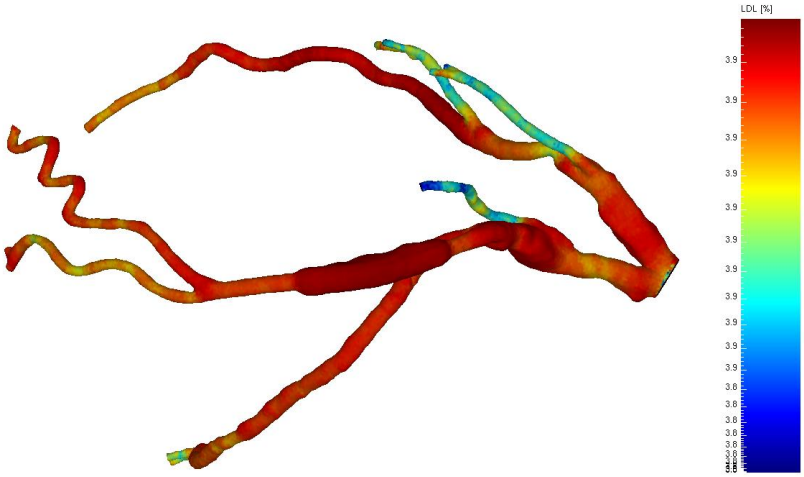
Z rozwiązania równań Naviera-Stokesa otrzymujemy wartości ciśnienia i prędkości. Następnie możemy policzyć naprężenie powierzchniowe (WSS) oraz stężenie tłuszczy o niskiej gęstości (LDL)[15]. Dzięki zaproponowanym modelom oświetlenia możemy prezentować pola skalarne w 3D, dla których wartość pola określa kolor pixela. Przedstawienie pól wektorowych dla których znamy kierunek wektorów (naprężenie powierzchniowe - kierunek prostopadły do powierzchni ścianki) jest analogiczne do pól skalarnych ponieważ w tym przypadku rysujemy tylko moduł wektora. Bardziej skomplikowanym przypadkiem jest wizualizacja pola prędkości, dla którego

nie potrafimy z góry przewidzieć kierunku. Dokładniejsza analiza tego problemu znajduje się w kolejnym podrozdziale. Poniżej przykład wizualizacji ciśnienia dla tętniaka naczynia mózgowego. Warto zauważyć, że największe ciśnienie jest na powierzchni worka w tylnej części. Jest to ważna



Rysunek 10: Profil ciśnienia dla tętniaka naczynia mózgowego.

informacja dla lekarzy podczas operacji ponieważ w tym miejscu może dojść do pęknięcia naczynia. Interesującą wielkością jest LDL ponieważ informuje nas w którym miejscu naczynia będzie powstawała blaszka miażdżycowa. Jest to ważna informacja dla lekarzy, ponieważ dzięki niej mogą podjąć decyzję czy wprowadzać stent do naczynia w celu udrożnienia naczynia. Idealnie to widać na obrazie naczynia wieńcowego. Zgrubienie widoczne z przodu naczynia jest to właśnie stent, który ma zapobiegać odkładaniu się w tym miejscu cholesterolu.



Rysunek 11: Profil LDL dla naczynia wieńcowego.

4.4 Wizualizacja pola prędkości

Prezentacja pola prędkości w 3D jest sporym wyzwaniem, ponieważ musimy przedstawić 3 zmienne przestrzenne oraz 3 składowe wektora. Dlatego wprowadza się metody pośrednie pozwalające analizować pole prędkości.

4.4.1 Cząsteczki

Metoda pozwala śledzić lokalne zmiany pola prędkości. Głównym zastosowaniem użycia cząstek jest generowanie animacji przepływu w czasie rzeczywistym. Generowanie cząstek sprowadza się do rozwiązania problemu adwekcji, czyli unoszenia substancji z przepływem, poprzez rozwiązanie równania ruchu dla poszczególnych cząstek.

$$\frac{d\mathbf{r}(t)}{dt} = \mathbf{v}(\mathbf{r}(t), t); \tag{18}$$

gdzie:

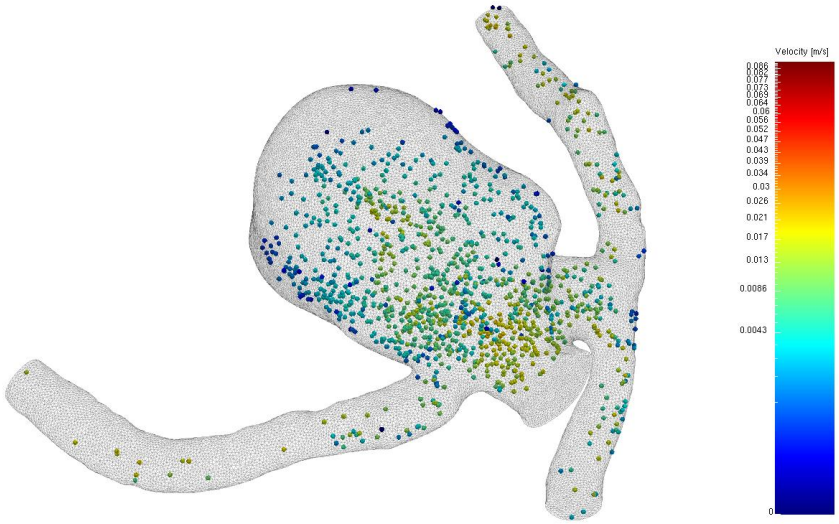
- **r** - położenie cząstki,
- **v** - prędkość dla cząstki, można ją otrzymać z interpolacji pola otrzymanego z rozwiązania równań Naviera-Stokesa.

Powyższe równanie rozwiązuje się poprzez całkowanie pola prędkości po czasie. Algorytm służący do wyznaczania cząstek składa się z kilku kroków.

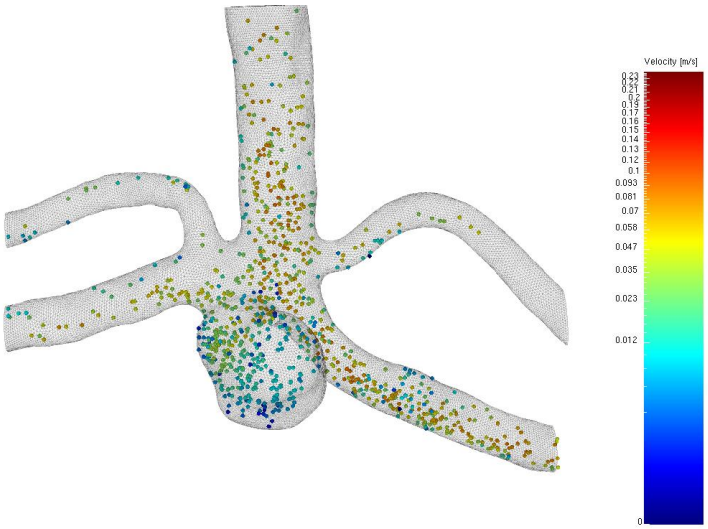
- zainicjuj początkowe cząstki (określ początkowe położenia i prędkości),
- przesuń cząstkę zgodnie z schematem całkującym,
- wyznacz nową prędkość cząstek, poprzez interpolację prędkości w objętości naczynia oraz pomiędzy kolejnymi momentami czasowymi symulacji,
- pokoloruj cząstki w zależności od wartości prędkości.

Problemem tej metody jest interpolowanie prędkości z objętości. Jeżeli chcemy otrzymać płynne zmiany prędkość podczas przechodzenia pomiędzy kolejnymi klatkami symulacji musimy interpolować prędkość dodatkowo z następnej klatki. Stosując metodę RK 2 dla 100 cząstek w każdym kroku dokonujemy 100*2*2=400 razy interpolacje, która zajmuje dużo czasu, ponieważ musimy przeszukiwać siatki, które składają się nawet z miliona punktów, dlatego tak ważny jest wybór metody interpolacji. Kolejnym problemem jest przeładowywanie dużej ilości danych. Jeżeli chcemy uzyskać płynne animacje musimy ograniczyć czas potrzebny na przejście do kolejnego punktu czasowego, dlatego nie możemy za każdym razem zmieniać wektora z danymi dla

których przeprowadzamy interpolacje. Rozwiązaniem jest wczytanie za pierwszym razem całej symulacji do pamięci RAM i w trakcie wizualizacji zmienianie tylko wskaźników na dane, ale to wiąże się z dużym zużyciem pamięci. Optymalny sposób polega na wczytywaniu kilku klatek do przodu i w trakcie generowania cząstek w osobnym wątku przeprowadzać przeładowywanie danych, ale to wiąże się z koniecznością napisania skomplikowanego menadżera pamięci. Zaletą generowania cząstek jest możliwość obserwacji dynamicznych zmian pola prędkości w czasie. Poniżej znajdują się przykładowe obrazy wizualizacji cząstek.



Rysunek 12: Cząstki dla tętniaka naczynia mózgowego.



Rysunek 13: Cząstki dla tętnicy podstawnej.

4.4.2 Linie prądu i tory cząsteczek

Linię prądu (streamline) definiujemy:

Jest to linia, która w każdym punkcie jest styczna do pola prędkości w tym punkcie.

Wyraża to tzw. równanie linii prądu:

$$\frac{dx}{u} = \frac{dy}{v} = \frac{dz}{w}, \tag{19}$$

gdzie:

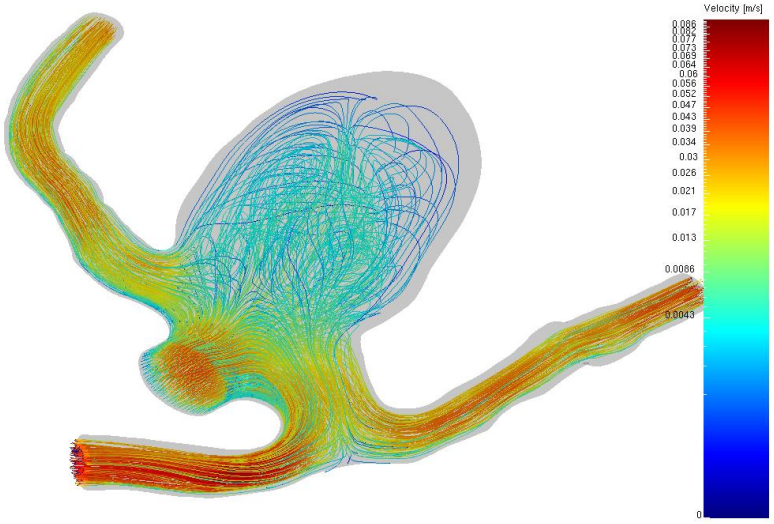
- x,y,z - składowe równania parametrycznego linii,
- u,v,w - składowe wektora prędkości w danym punkcie.

Algorytm generowania linii prądu polega na całkowaniu pola prędkości po czasie. Pole prędkości otrzymujemy z numerycznego rozwiązywania równań Naviera-Stokesa w wierzchołkach siatki objętościowej ze względu na to, iż w każdym kroku całkowania musimy interpolować prędkość do danego punktu. Ten problem jest dokładniej opisany w sekcji nr 6. Generowanie linii prądu składa się z następujących kroków:

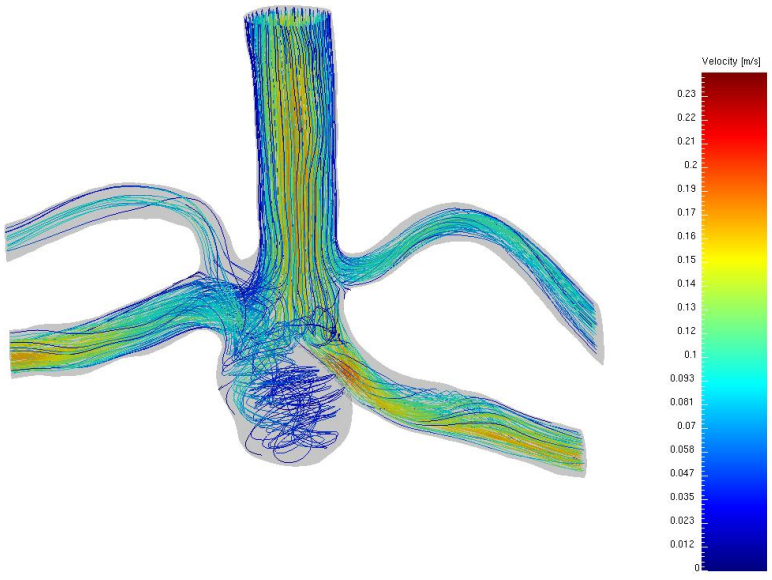
- Pierwszym krokiem jest wyznaczenie warunków początkowych, czyli musimy określić położenie początkowe x_0 i prędkość początkową v_0. Za punkt początkowy możemy wybrać dowolny punkt z wnętrza naczynia. Prędkość początkową wyznaczamy przy pomocy algorytmu interpolującego. Warto pamiętać, że linie powinno całkować się po czasie w przód(z dodatnim krokiem czasowym) i w tył (z ujemnym krokiem czasowym) wtedy powinniśmy dla każdego punktu znaleźć linię, która łączy wlot naczynia z wylotem.
- Całkowanie iteracyjnego równania ruchu dla schematu Eulera sprowadza się do równania: $x_{n+1} = x_n + v_n * dt$, dla którego znamy x_0 i v_0, więc możemy wyznaczyć x_1. Następnie poprzez interpolację możemy wyznaczyć v_1 i ponownie całkować równanie. Ważnym elementem w tym algorytmie jest wybór schematu do całkowania po czasie. Zastosowane schematy zostały opisane w następnym rozdziale.
- Pokolorowanie linii w zależności od wartości prędkości w danym punkcie. W tym celu musimy policzyć prędkość maksymalną (maxV) i minimalną (minV) w symulacji. Następnie musimy zdefiniować paletę kolorów, jako wektor barw(kolory). Dla każdego punktu k (v_k wartość prędkości) z linii obliczamy wartość koloru(kolor) z równania:

$$kolor[k] = kolory[(v_k - minV)/(maxV - minV) * liczbakolorow]$$

Poniżej znajdują się przykładowe obrazy dla linii prądu.



Rysunek 14: Linie prądu dla tętniaka naczynia mózgowego.

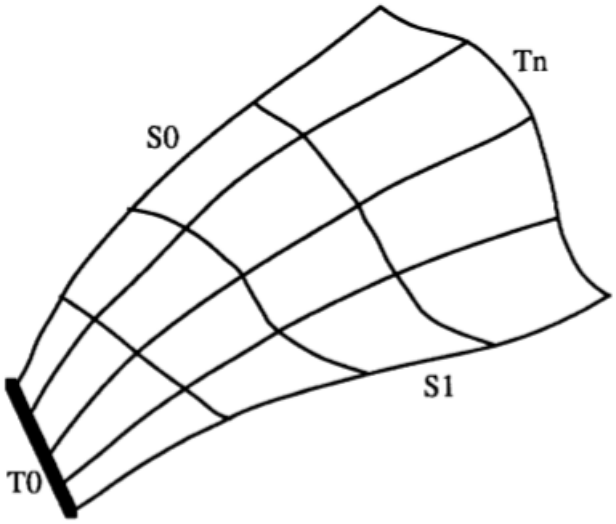


Rysunek 15: Linie prądu dla tętnicy podstawnej.

Interesujące wizualizacje można otrzymać generując linie prądu dla kolejnych chwil czasowych i składając otrzymane obrazki w animacje. Dzięki temu możemy obserwować jak pole prędkości zmienia się w całej symulacji. Jest to szczególne istotne dla zjawisk dynamicznych jak powstawanie wirów w naczyniach.

4.4.3 Powierzchnie prądu

Powierzchnie prądu w dobry sposób oddają wiry, ponieważ zawijają się wokół takich miejsc. Powierzchnią prądu nazywamy powierzchnię, która jest styczna do pola prędkości, jest wyznaczana poprzez linie prądu łączone w czworokąty, jak na rysunku poniżej.



Rysunek 16: Konstruowanie powierzchni przepływu [6].

gdzie:

T0 - linia startowa powierzchni;

S0 - lewa krawędź powierzchni;

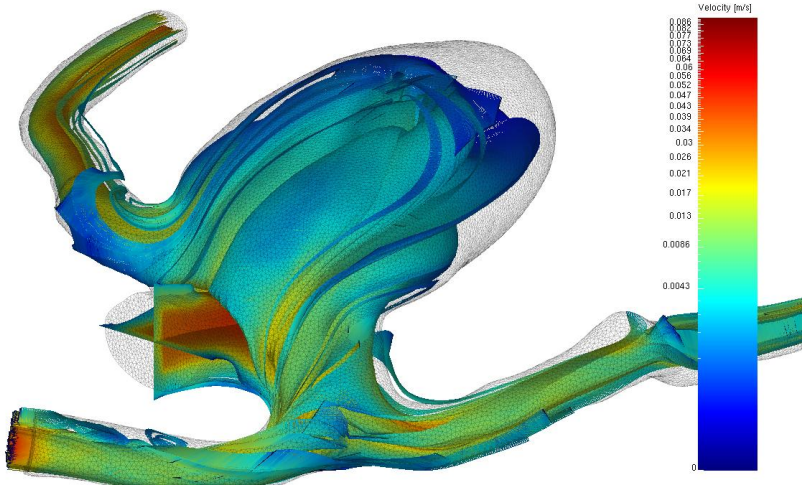
S1 - prawa krawędź powierzchni;

Tn - linia końcowa powierzchni;

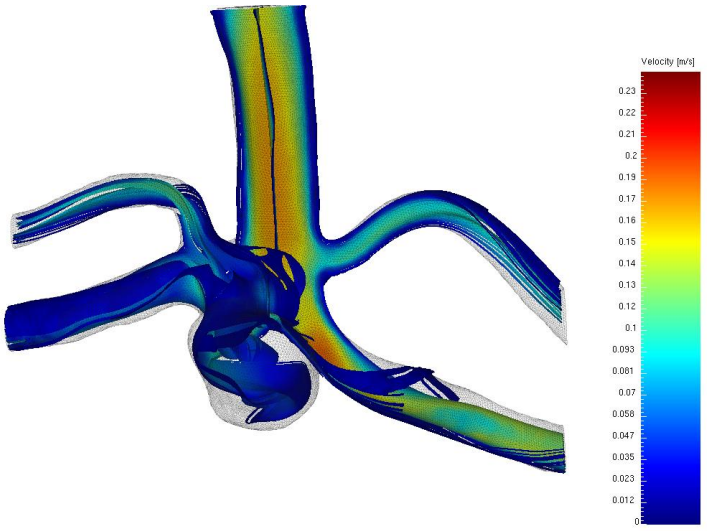
Algorytm generowania powierzchni prądu składa się z następujących kroków:

- wygeneruj linie prądu, na których będzie oparta powierzchnia;
- sąsiadujące linie prądu połącz w czworokąty, jeżeli są oddalone od siebie o odległość większą niż stała R to rozdziel powierzchnię;
- wyznacz kolor, jak dla linii prądu.

Dzięki tej technice otrzymamy cały profil przepływu krwi, co spowoduje zaobserwowanie znacznie większej liczby szczegółów. Poniżej znajdują się przykładowe obrazy dla powierzchni prądu.



Rysunek 17: Powierzchnia prądu dla tętniaka.



Rysunek 18: Powierzchnia prądu dla tętnicy podstawnej.

Szczególnie dobrze to widać na przykładzie tętniaka naczynia mózgowego, dla którego dzięki powierzchni prądu możemy obserwować jak złożony przepływ ma miejsce wewnątrz naczynia. Powyższe wizualizacje mają na celu pomóc lekarzom podczas operacji medycznych oraz wspomóc diagnostykę chorób układu krwionośnego. Dzięki analizie pola prędkości możemy w przybliżeniu stwierdzić w jakiej części naczynia będzie odkładała się blaszka miażdżycowa oraz gdzie może dojść do wylewu.

5 Algorytmy całkujące

Rozwiązywanie równań różniczkowych stanowi niezwykle ważną część nauk ścisłych gdy pojawiają się wymagające rozwiązania zmienne w czasie problemy natury dynamicznej, np. ruch trzech ciał czy przepływ płynu. Istnieje wiele typów równań różniczkowych oraz sposobów, dzięki którym możemy je rozwiązać. Nie wszystkie typy można obliczyć analitycznie, problem stwarza wiele równań nieliniowych. W takich przypadkach warto skorzystać z metod numerycznych, dzięki którym obliczymy krok po kroku wartości funkcji w kolejnych punktach dziedziny tej funkcji z zadanym krokiem całkowania.

Całkowanie numeryczne należy do metod przybliżonych rozwiązywania równań różniczkowych typu $y' = f(x, y)$. W moim przypadku równanie, które chcemy rozwiązać to:

$$\frac{d\mathbf{r}}{dt} = \mathbf{v}(\mathbf{r}, t)$$

gdzie:

- **v** - prędkość,
- **r** - położenie,
- t - czas,

Jest to zwyczajne równanie różniczkowe pierwszego rzędu. Możemy rozwiązać je poprzez całkowanie z krokiem h. Dla uproszczenia zapisu poniższe wzory są dla przypadku jednowymiarowego. Najprostszym schematem jaki możemy zastosować jest metoda Eulera:

$$x_{n+1} = x_n + v_n h$$

gdzie:

- x_{n+1} - położenie w kolejnym n+1 kroku,
- x_n - położenie w poprzednim n-tym kroku,
- v_n - prędkość w poprzednim n-tym kroku,
- h - krok czasowy.

Zaletą tej metody jest to, że jest bardzo prosta i szybka. Niestety błąd w tej metodzie jest rzędu h, dlatego stosujemy metody wyższych rzędów.

5.1 Metody Rungego–Kutty

Metoda Eulera, omówiona powyżej jest dość łatwa w implementacji, niestety w praktyce wraz ze wzrostem ilości iteracji obserwujemy znaczny wzrost błędów numerycznych. Przyczyną takiego zachowania jest to, iż zmiany wartości obliczanej funkcji różniczkowanej zależą jedynie od wartości pochodnej obliczonej na początku obliczeń dla zmiennej. Skutkuje to koniecznością użycia niewielkiej wartości kroku całkowania równania różniczkowego, co powoduje znaczny przyrost czasu obliczeń. Aby rozwiązać ten problem stworzono zbiór metod Rungego-Kutty. Podstawowa różnica pomiędzy metodą Eulera i metodami RK polega na tym, że oblicza się pochodną również wewnątrz przedziału zmiennej x, w którym różniczkujemy funkcję. Im wyższy stopień metody Rungego-Kutty tym charakteryzuje się ona wyższą dokładnością w których modyfikacji ulegają wzory na iterowanie kolejnych wartości różniczkowanych funkcji oraz wzory na współczynniki.

Metoda Rungego-Kutty drugiego rzędu jest używana w układach czasu rzeczywistego ze względu na swoją małą złożoność połączoną z większą dokładnością otrzymywanych przybliżeń w porównaniu z metodą Eulera.

- 2 rzędu:

$$x_{n+1} = x_n + \frac{1}{2}(k_1 + k_2) \tag{20}$$
$$k_1 = v(t_n, x_n)h$$
$$k_2 = v(t_n + \frac{1}{2}h, x_n + \frac{1}{2}k_1)h$$

- 3 rzędu:

$$x_{n+1} = x_n + \frac{1}{6}(k_1 + 4k_2 + k_3) \tag{21}$$
$$k_1 = v(t_n, y_n)h$$
$$k_2 = v(t_n + \frac{1}{2}h, x_n + \frac{1}{2}k_1)h$$
$$k_3 = v(t_n + h, x_n - k1 + 2k_2)h$$

- 4 rzędu:

$$x_{n+1} = x_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{22}$$
$$k_1 = v(t_n, y_n)h$$
$$k_2 = v(t_n + \frac{1}{2}h, x_n + \frac{1}{2}k_1)h$$
$$k_3 = v(t_n + \frac{1}{2}h, x_n + \frac{1}{2}k_2)h$$
$$k_4 = v(t_n + h, x_n + k_3)h$$

Metoda Rungego-Kutty czwartego rzędu jest szczególnie przydatna w badaniach symulacyjnych, ze względu na dużą dokładność. W układach czasu rzeczywistego jest ona praktycznie nie używana ze względu na swoja czasochłonność. Wymaga ona bowiem aż czterokrotnego określenia wartości prędkości.

- 5 rzędu:

$$x_{n+1} = x_n + \frac{1}{90}(7k_1 + 32k_3 + 12k_4 + 32k_5 + 7k_6) \tag{23}$$

$$k_1 = v(t_n, y_n)h$$

$$k_2 = v(t_n + \frac{1}{4}h, x_n + \frac{1}{4}k_1)h$$

$$k_3 = v(t_n + \frac{1}{4}h, x_n + \frac{1}{8}(k_1 + k_2))h$$

$$k_4 = v(t_n + \frac{1}{2}h, x_n + (-\frac{1}{2}k_2 + k_3))h$$

$$k_5 = v(t_n + \frac{3}{4}h, x_n + (\frac{3}{16}k1 + \frac{9}{16}k_4))h$$

$$k_6 = v(t_n + h, x_n + \frac{1}{7}(-3k_1 + 2k_2 + 12k_3 - 12k_4 + 8k_5))h$$

W tych metodach błąd jest proporcjonalny do h^r, gdzie r jest rzędem metody. Na przykład zmniejszając h o jeden rząd dla r=4 błąd zmaleje o cztery rzędy. Te metody dają dokładne wyniki. Niestety w każdym kroku musimy r razy liczyć prędkości, co powoduje wydłużenie obliczeń i spowolnienie wizualizacji. Metody te są zbyt wolne, by wykorzystywać ją w czasie rzeczywistym. Dokładniejsze rozważania na temat tych metod można znaleźć w [10].

5.2 Metody Adamsa–Bashfortha

Metody wielokrokowe są starsze i bardziej popularne od metod Rungego-Kutty. Metody te charakteryzują się prostotą w tym sensie, że nie wymagają wielokrotnego obliczania pochodnej w punktach pośrednich, korzystają wyłącznie z wartości obliczonych w punktach, w których chcemy uzyskać rozwiązanie. Jawne metody Adamsa wymagają tylko jednego obliczenia prędkości na krok. Możemy wyróżnić dwa typy metod Adamsa:

- jawne metody Adamsa-Bashfortha,
- niejawne metody Adamsa-Moultona.

W mojej pracy używam jawnych metod Adamsa-Bashforta ze względu na prostotę ich użycia do generowania wizualizacji. Na przykład metody Adamsa-Moultona, które są metodami niejawnymi nie nadają się do generowania cząstek, ponieważ w tym zagadnieniu chcemy znać położenia w kolejnych krokach czasowych. Klasyczne metody Adamsa polegają na iteracyjnym znajdowaniu kolejnych wartości funkcji według schematów:

- 2 rzędu:

$$x_{n+2} = x_{n+1} + h\left(\frac{3}{2}v(t_{n+1}, x_{n+1}) - \frac{1}{2}v(t_n, x_n)\right) \tag{24}$$

- 3 rzędu:

$$x_{n+3} = x_{n+2} + h\left(\frac{23}{12}v(t_{n+2}, x_{n+2}) - \frac{4}{3}v(t_{n+1}, x_{n+1}) + \frac{5}{12}v(t_n, x_n)\right) \tag{25}$$

- 4 rzędu:

$$x_{n+4} = x_{n+3} + h\left(\frac{55}{24}v(t_{n+3}, x_{n+3}) - \frac{59}{24}v(t_{n+2}, x_{n+2}) + \frac{37}{24}v(t_{n+1}, x_{n+1}) - \frac{3}{8}v(t_n, x_n)\right) \tag{26}$$

- 5 rzędu:

$$\begin{aligned} x_{n+5} = x_{n+4} &+ h\left(\frac{1901}{720}v(t_{n+4}, x_{n+4}) - \frac{1387}{360}v(t_{n+3}, x_{n+3})\right) \\ &+h\left(\frac{109}{30}v(t_{n+2}, x_{n+2}) - \frac{637}{360}v(t_{n+1}, x_{n+1}) + \frac{251}{720}v(t_n, x_n)\right) \end{aligned} \tag{27}$$

Metody te posiadają trochę mniejszą dokładność niż Rungego–Kutty. Zaletą ich jest możliwość jednorazowego liczenia prędkości w każdym kroku czasowym, gdyż wcześniejsze wartości prędkości są zapamiętywane w tablicy. Dzięki tak wybranym warunkom algorytm wykonuje się w podobnym czasie, jak metoda Eulera. Jest to najlepsza metoda dla tego typu problemów, gdyż zależy nam na krótkim czasie trwania oraz dokładności obliczeń. Dokładniejszy opis można znaleźć w [11].

Test poprawności zaimplementowanych metod można znaleźć w dodatkach.

6 Algorytmy interpolujące

W symulowaniu przepływu rozwiązujemy układ równań różniczkowych w środkach komórek tetraedrycznych. Jaśniej mówiąc, chcąc znać wartość prędkości w dowolnym punkcie przestrzeni (co jest potrzebne do generowania trajektorii cząstek) musimy stosować interpolacje. Należy pamiętać, że rozwiązane równania na przykład równanie ciągłości jest spełnione w sposób całkowy dla każdej komórki. Jest to ważne gdy rozważamy płyn nieściśliwy, ponieważ aby zachować tę własność musimy metodę interpolacji wybrać zgodną z sposobem dyskretyzacji. Alternatywnym podejściem jest interpolowanie wartości z środków komórek do wierzchołków siatki. Następnie przeprowadzanie obliczeń na nowych wartościach. W ten sposób na samym początku popełniamy pewien błąd, ale zyskujemy większą swobodę w operacji na danych, ponieważ wystarczy, że przechowujemy tylko wierzchołki i wartości, a nie musimy pamiętać jak są zbudowane poszczególne komórki. W tej pracy rozważam dwa sposoby wyznaczenia wartość pola w dowolnym punkcie r_0.

6.1 Wybór prędkości wprost z siatki obliczeniowej

W tym podejściu szukamy do której komórki należy punkt r_0. Algorytm znajdujący indeks komórki do której należy dowolny punkt można zaimplementować w oparciu o strukturę octree. W tej pracy wykorzystałem implementacje napisaną przez Marcina Krotkiewskiego, w ramach biblioteki mutils [17]. Interpolowanym wynikiem jest wartość w środku komórki. Problemem występującym w tym sposobie są punkty, które znajdują się blisko ścianek, ponieważ w tym przypadku taki wybór jest mało dokładny. Zaletą tej metody jest to, że wartość interpolacji otrzymujemy bezpośrednio z wyniku symulacji komputerowej, a nie bazując na wartościach w wierzchołkach siatki.

6.2 Interpolowanie z otoczenia węzła

Algorytm ten opiera się strukturze danych kd-tree [22], która umożliwia znalezienie wszystkich wierzchołków w odległości nie większej niż R od r_0. Oznacza to, że tworzymy kulę o promieniu R i wewnątrz niej szukamy wierzchołków. Następnie interpolujemy wartości pola z węzłów do r_0. W ramach tej pracy zbadano sposób interpolacji jak $\frac{1}{r^2}$. Na koniec wystarczy znormalizować otrzymaną wartość poprzez policzenie średniej ważonej z otrzymanych wyników. Problemem w tym podejściu jest dobór promienia R. Dla dużej wartości będziemy uśredniać pole, co zmienia w istotny sposób otrzymane rozwiązanie. Z kolei dla małych wartości możemy nie znaleźć żadnego wierzchołka. W tej pracy przyjęto R=2a, gdzie a oznacza średnią długość krawędzi komórki. Promień ten został dobrany empirycznie. Metoda ta daje dokładniejsze wyniki dla punktów

znajdujących się przy brzegach ścianek, gdyż uśrednia wartości z sąsiednich komórek, ale przez to otrzymane pole prędkości nie spełnia równania ciągłości, co może być przyczyną błędów podczas obliczania wielkości fizycznych zależnych od pola prędkości.

6.3 Interpolowanie wzdłuż linii

Poprawność zaimplementowanych metod można sprawdzić poprzez porównanie z wynikami teoretycznymi dla prostych przypadków. Dla cylindra ciśnienie w kierunku przepływu zmienia się w sposób liniowy, zaś w kierunku prostopadłym jest stałe. Prędkość w kierunku przepływu nie zmienia się, z kolei w kierunku prostopadłym zmienia się jak $-x^2$. Z miejscami zerowymi na ściankach i maksimum w środku. W celu wyznaczenia kierunku można narysować linie. Następnie wystarczy dla każdego punktu linii interpolować wartość i porównać z wynikiem teoretycznym. W ten sposób można sprawdzić poprawność interpolacji i sporządzić wykresy. Metoda ta jest dobrym sposobem na porównanie dwóch metod interpolacji stosowanych w tej pracy.

6.4 Porównanie metod interpolujących

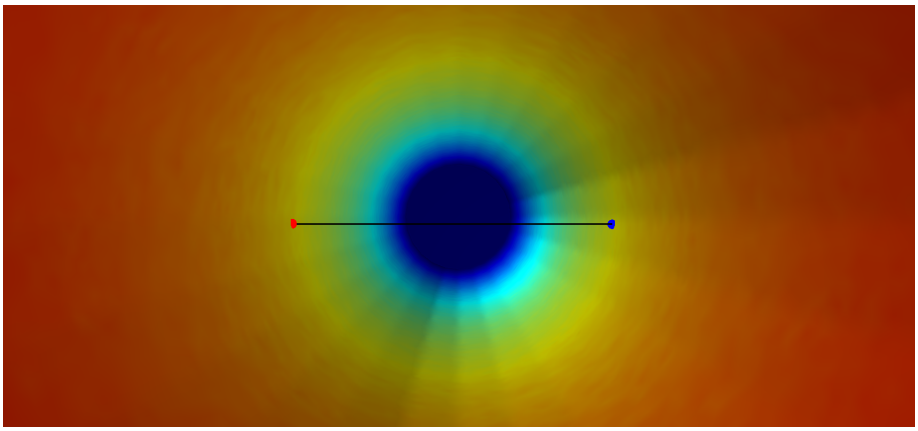
W celu porównania metod interpolacji można przeprowadzić następujący test:

- narysować linie w taki sposób, aby wiedzieć jak będzie zmieniało się pole,
- wyznaczyć kolejne punkty linii,
- przeprowadzić interpolację w tych punktach,
- porównać otrzymane wyniki dla dwóch rożnych interpolacji z wynikiem teoretycznym.

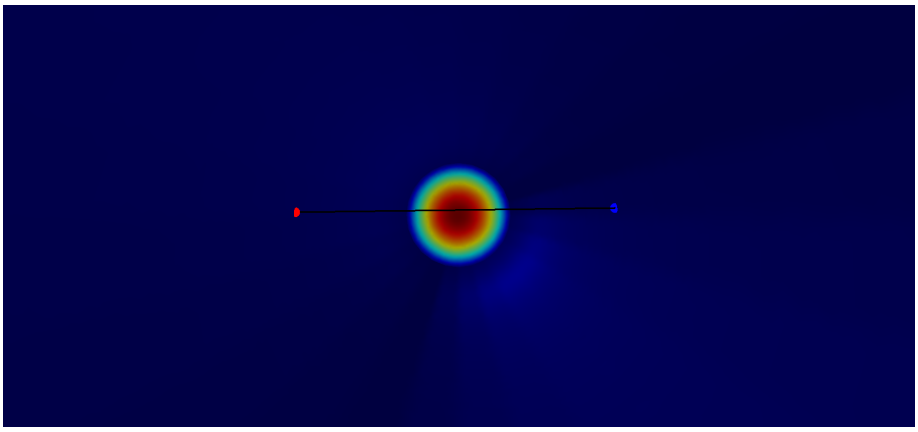
W ramach tej pracy przeprowadziłem powyższy test dla cylindra.

Na poniższych obrazach widzimy wizualizacje cylindra z narysowanymi liniami wzdłuż, których będziemy przeprowadzać interpolację. Ciśnienie powinno być stałe a prędkość powinna zmieniać się jak $-x^2$. Na wykresie interpolacji prędkości nie widać dużej różnicy pomiędzy dwoma badanymi metodami. Jedynym obszarem w którym można zaobserwować odchylenia są końce przedziałów. Są to miejsca przy brzegach siatki. kd-tree dla tych punktów zwraca wartości większe od zera. Wynika to z faktu, iż w tym algorytmie budujemy kulę, więc zawsze obejmiemy nią kilka punktów, które leża wewnątrz naczynia. Jest to cecha, która w negatywny sposób wpływa na generowanie na przykład cząstek, ponieważ może się zdarzyć, że przez to cząstka wypadnie poza naczynie, co jest oczywiście błędem numerycznym. Wykres dla ciśnienia jest jeszcze

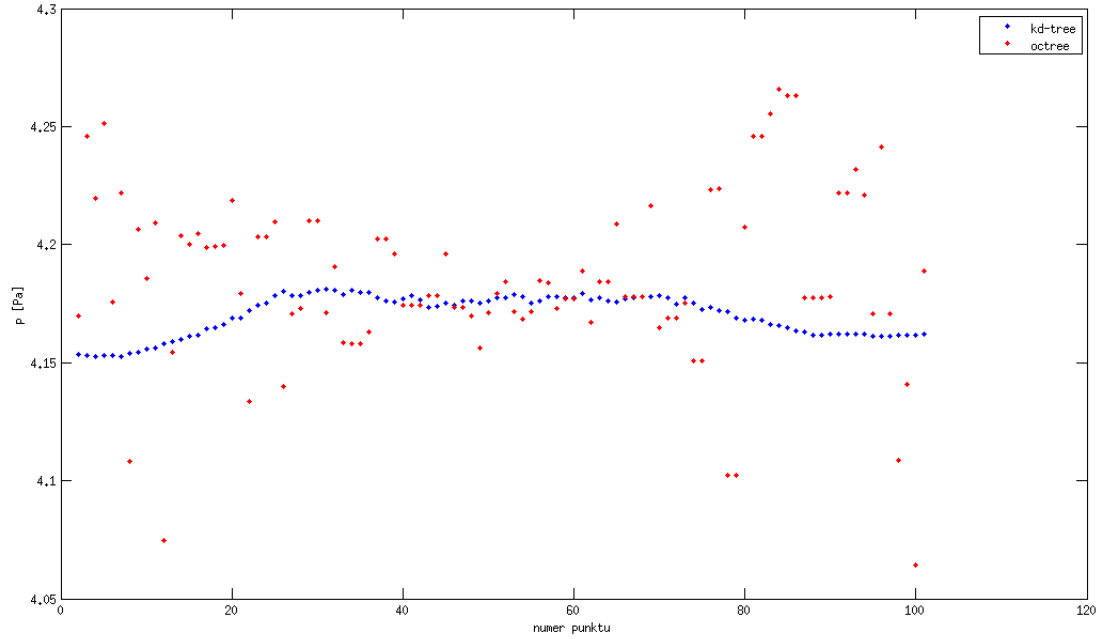
ciekawszy, ponieważ teoretycznie kd-tree w lepszy sposób oddaje spodziewany rezultat jakim jest linia prosta. Warto zauważyć, iż dla tego przypadku ciśnienie było przeskalowane pomiędzy 0-6.5 Pa, dlatego zakres zmienności na poziomie 0.2 Pa dla octree można uznać za nieistotny wynikający z nierównomiernego rozłożenia komórek. W ten sposób możemy zaobserwować jak dużych uśrednień dokonujemy stosując metodę budowania kuli. W skrajnym przypadku, jeżeli w jednej komórce będziemy mieli wartość 1, a w komórkach na około 0, to stosując kd-tree otrzymamy pole zanikające w sposób ciągły, co dla tego modelowego przypadku jest bardzo dużym błędem.



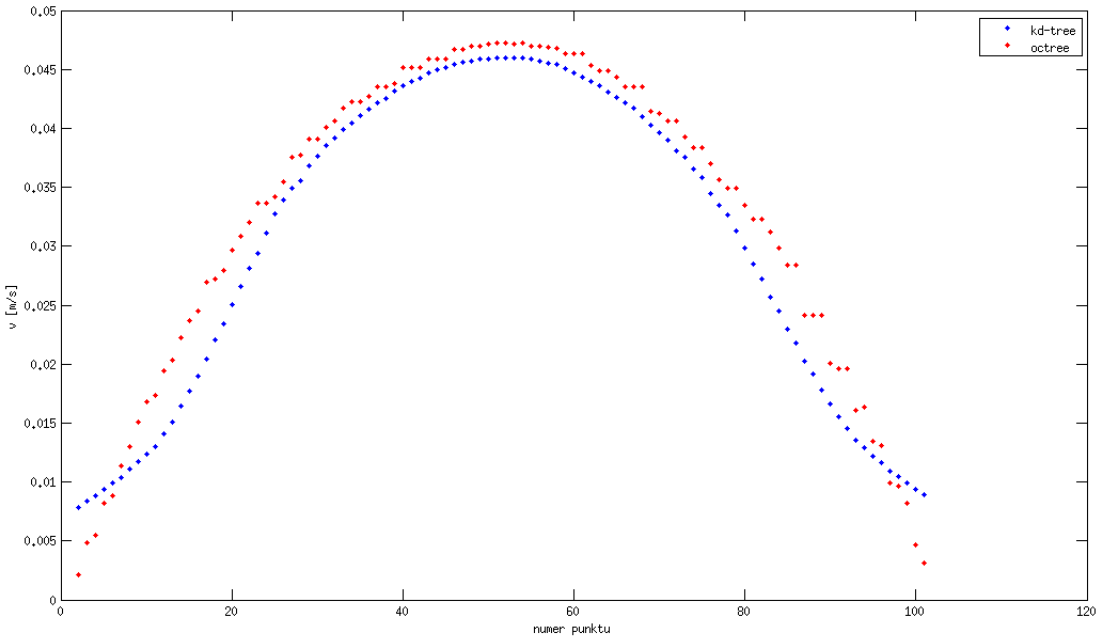
Rysunek 19: Profil ciśnienia dla cylindra, kamera wewnątrz obiektu.



Rysunek 20: Profil prędkości dla cylindra, kamera wewnątrz obiektu.



Rysunek 21: Wykres ciśnienia dla interpolacji wzdłuż linii.



Rysunek 22: Wykres prędkości dla interpolacji wzdłuż linii.

7 Test metod interpolacji i całkowania

W tym rozdziale przedstawię skuteczność zaimplementowanych metod interpolacyjnych i całkujących. Test polega na wyliczeniu linii prądu zaczynającej się z wlotu naczynia. Sukcesem nazwiemy moment, gdy linia przetnie powierzchnię wylotu naczynia, co w konsekwencji sprowadza się to do prostego warunku, czy prosta przechodzi przez powierzchnię. Porażkę obserwujemy z kolei wtedy, gdy linia przetnie ściankę naczynia. W teście generuje 100 linii, wartość na osi y na wykresach oznacza ilość linii, które skończyły się sukcesem. Testy, które wykonałem w pracy magisterskiej dotyczyły trzech przypadków dla których kroki czasowe h wynosiły odpowiednio:

- aorty podstawnej h = 0.0140,
- naczynia wieńcowego h = 0.0354,
- tętniaka naczynia mózgowego h = 0.0158.

Krok czasowy został obliczony z równania:

$$h = \frac{r_{min}}{maxVel} \tag{28}$$

gdzie:

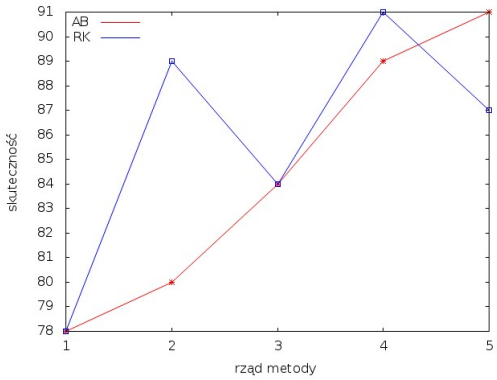
- r_{min} - długość najkrótszej krawędzi.
- maxVel - prędkość maksymalna dla symulacji.

W ten sposób jesteśmy pewni, że dla naszego schematu całkującego jest spełniony warunek Couranta-Friedrichsa-Lewy'ego mówiący, że w jednym kroku całkowania nie możemy pokonać więcej niż jedną komórkę.

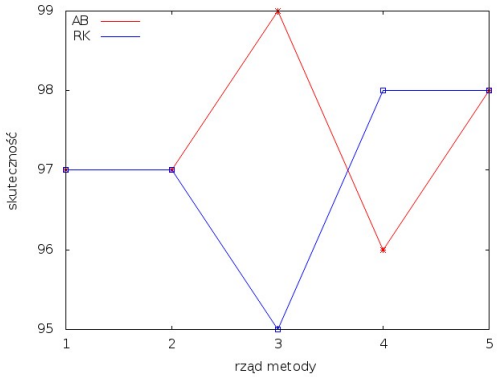
7.1 Metoda kdtree

Metoda Eulera jest szczególnym przypadkiem zarówno metod Rungego Kutty, jak i metody Adamsa Bashfortha. Dlatego pierwszy punkt na wykresach jest zawsze taki sam. Z wykresów widać, że ciężko stwierdzić, które metody są lepsze do całkowania. Obydwie grupy metod AB i RK dają bardzo dobre i porównywalne rezultaty. Widać natomiast, że najbardziej opłaca stosować się metody od trzeciego rzędu, ponieważ dają lepsze rezultaty. Wyjątkiem jest przypadek naczynia wieńcowego gdzie metoda Eulera miała skuteczność 97 co jest świetnym wynikiem. W pozostałych przypadkach metody niższych rzędów słabiej sobie radzą z całkowaniem pola prędkości. Z kolei metody wyższych rzędów stają się mocno nielokalne i mogą wprowadzać zaburzenie do otrzymanego przez nas rozwiązania, dlatego optymalne są metody drugiego i

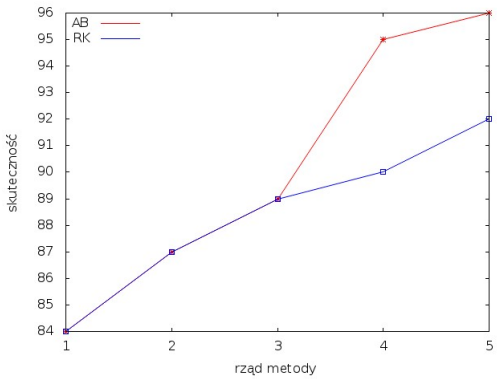
trzeciego rzędu, ponieważ mają dobrą skuteczność, a nie są tak złożone jak wyższych rzędów. Metody AB są dużo szybsze niż metody RK, a dla tych przypadków nie ma różnic w dokładności otrzymanych rozwiązań, dlatego powinniśmy stosować metody AB.



Rysunek 23: Wykres skuteczności metod całkowania dla tętnicy podstawnej.



Rysunek 24: Wykres skuteczności metod całkowania dla naczynia wieńcowego.



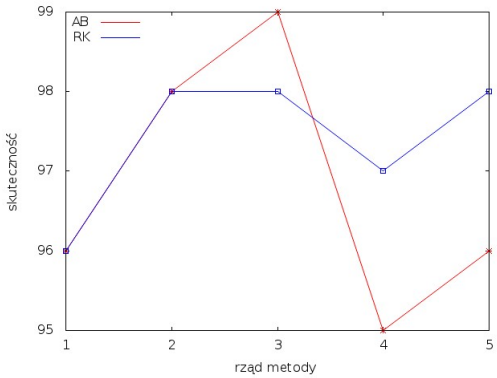
Rysunek 25: Wykres skuteczności metod całkowania dla tętniaka naczynia mózgowego.

7.2 Metoda octree

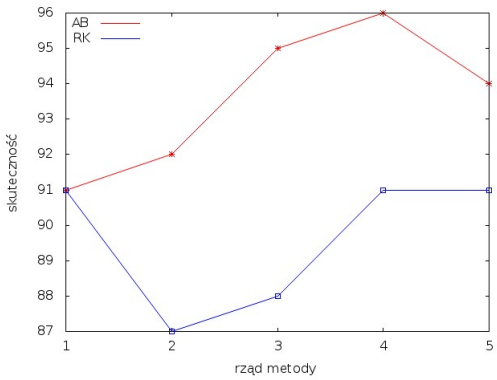
W aorcie podstawnej mamy przepływ z trudnym momentem do całkowania gdy cząstki odbijają się od podstawy naczynia. Ze względu na to metody niższych rzędów dla kd-tree gdzie interpolacja przy ściankach daje dość duże wartości prędkości nie radziły sobie z tym przypadkiem. Dla octree, gdzie interpolacja przy brzegu naczynia jest dużo dokładniejsza możemy zaobserwować znacznie lepsze rozwiązania. Istotnym rezultatem jest to, że nawet metoda Eulera daje bardzo dobre rozwiązanie.

W naczyniach wieńcowych w czasie przepływu występuje zjawisko cofania płynu. Skutkuje to powstawaniem drobnych lokalnych wirów. Ciekawe jest to, że w metodzie octree interpolacji cząstki są dużo dłużej uwięzione w wirze. Skutkuje to niewiele gorszym wynikiem skuteczności niż dla poprzedniej metody. Interesujące jest również to, że nie obserwujemy istotnych różnic pomiędzy wyborem metody całkowania a dokładnością procedury. Jest to spowodowane tym, że pole prędkości jakie otrzymujemy z interpolacji jest dużo lepszej jakości i wybór schematu całkującego ma mniejsze znaczenie.

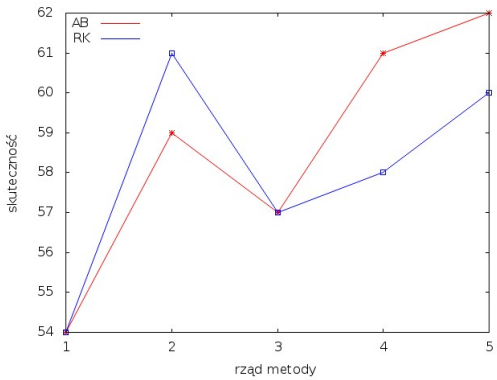
Dla tętniaka naczynia mózgowego jeżeli popatrzymy na skuteczność całkowania to możemy zaobserwować prawdziwą katastrofę. Maksymalny wynik wynosi tylko 61, co jest bardzo słabym rezultatem. Na szczęście jest to pozornie zły wynik wynikający z skonstruowania testu, który zakłada, że linia prądu musi przejść przez wylot naczynia. Dla tętniaka naczynia mózgowego, w tylnej części worka, istnieje olbrzymi wir. Dla tego przypadku porażka w teście wynika z faktu, że cząstki spędzają bardzo dużo czasu w wirze, a test numeryczny jest ograniczony przez 12000 tysięcy kroków. Skutkuje to tym, że cząstki nie zdążają dotrzeć do wylotu. Według mnie jest to ciekawym wynikiem, ponieważ pokazuje jak ważnym zagadnieniem jest dobór metody interpolacji. Dla metody kd-tree nie obserwujemy tego zjawiska. Podsumowując, Najbardziej optymalnym wyborem algorytmu całkującego jest metoda trzeciego rzędu AB, ponieważ daje dobrą dokładność otrzymywanych wyników, jest w miarę lokalna, oraz wymaga małego nakładu obliczeń. Do interpolacji powinniśmy stosować schemat zgodny z dyskretyzacją przeprowadzaną podczas rozwiązywania równań dynamiki płynu. Wynika to, z faktu, że w przeciwnym wypadku możemy otrzymać przepływ dla którego nie będzie spełniona nieściśliwość przepływu oraz możemy wprowadzać błędy na przykład przy brzegach naczynia, pomimo tych niedociągnięć algorytm bazujący na budowaniu kuli daje dobre wyniki. Należy jednak pamiętać, że w tym przypadku istotne staje się stosowanie metod wyższych rzędów do całkowania pola prędkości.



Rysunek 26: Wykres skuteczności metod całkowania dla tętnicy podstawnej.



Rysunek 27: Wykres skuteczności metod całkowania dla naczynia wieńcowego.



Rysunek 28: Wykres skuteczności metod całkowania dla tętniaka naczynia mózgowego.

8 Wnioski

Głównym zadaniem mojej pracy magisterskiej było opracowanie modelu wizualizacji przepływ krwi przez naczynia krwionośne do aplikacji Virtus. W pierwszej kolejności omówiłem teoretyczne aspekty dynamiki płynu w podejściu zaproponowanym przez Eulera i Lagrange. Następnie omówiłem równania Naviera-Stokesa i ich podstawy fizyczne w celu ustalenia rodzaju rozwiązywanego przepływu. W dalszej części przedstawiłem metodę objętości skończonej łącznie z zastosowanym sposobem dyskretyzacji, dzięki którym zaprezentowałem w jaki sposób otrzymuję wartości pola prędkości w określonych punktach. W kolejnym rozdziale opisałem kilka wybranych algorytmów oświetlenia i ich zastosowanie do generowania wizualizacji badanych naczyń krwionośnych i obiektów służących do przedstawiania pola prędkości. Pokazałem, że w celu uzyskania realistycznych wizualizacji powinniśmy stosować model Phonga. Następnie zaprezentowałem wizualizacje różnych wielkości fizycznych tj: ciśnienia, prędkości, naprężenia powierzchniowego i LDL dla wybranych naczyń krwionośnych, aby móc przeanalizować badany przepływ płynu. W dalszych rozdziałach omówiłem metody rozwiązywania zwykłych równań różniczkowych algorytmami Adamsa-Bashforda i Rungego-Kutty. Wyjaśniłem, dlaczego powinniśmy stosować metody jawne. Przedstawiłem algorytmy do 5-tego rzędu oraz przetestowałem ich poprawność i zbieżność na modelowym przykładzie. Wykazałem, że metody RK są dokładniejsze dla modelowego zagadnienia, ale wymagają większej ilości obliczeń niż metody AB. Przedstawiłem zagadnienie interpolacji i porównałem dwa różne algorytmy wyznaczania wartości pola w dowolnym punkcie przestrzeni z wynikiem teoretycznym, pierwszy zgodny z dyskretyzacją równań różniczkowych, co oznacza, że zachowuje, np. nieściśliwość płynu, natomiast drugi oparty na węzłach siatki bazujący na algorytmie budowania kuli. Wykazałem, że metoda zgodna z dyskretyzacją daje lepsze wyniki, ale wymaga przechowywania większej ilości danych. Następnie przeprowadziłem obszerny test sprawdzający jakość generowanych wizualizacji w zależności od wybranego algorytmu całkującego i metody interpolacji. Obydwie metody całkowania dały podobne rezultaty. Istnieje jednak znacząca różnica pomiędzy metodami niższych rzędów, a wyższych szczególnie dla interpolacji wewnątrz kuli, gdzie otrzymujemy trochę gorszej jakości pole prędkości. Metody AB są tyle razy szybsze od metod RK ile wynosi rząd metody, można to wytłumaczyć tym, iż najwięcej czasu w tych algorytmach zajmuje interpolacja, ze względu na duże rozmiary siatki. W AB interpolujemy zawsze jeden raz na każdy krok, zaś w RK tyle razy ile wynosi rząd metody, stąd powyższa różnica w złożoności tych algorytmów. Dlatego w tym zagadnieniu powinniśmy stosować metody AB. Najlepszym wyborem jest metoda trzeciego rzędu, ponieważ jest znacznie dokładniejsza od metod niższego rzędu, ale nie uśrednia jeszcze w znacznym stopniu pola prędkości. Metody czwartego rzędu i wyższych są mocno

nielokalne przez co w istotny sposób wprowadzają zmiany w otrzymanym rozwiązaniu. Kolejny test dotyczył metod interpolacji. Metoda zgodna z dyskretyzacją daje lepsze wyniki, a także jest bardziej poprawna z fizycznego punktu widzenia, gdyż spełnia równanie ciągłości. Wadą tej metody jest wymóg pamiętania złożonych struktur o budowie komórek oraz o sąsiedztwie. Metoda interpolacji wewnątrz kuli jest dokładniejsza przy ściankach komórek elementarnych, niestety nie zachowuje nieściśliwości płynu oraz może nie znaleźć punktu wewnątrz siatki przez co daje gorsze wyniki, ale wymaga dużo prostszych algorytmów i struktur. Zalecam używać metody zgodnej z dyskretyzacją, ponieważ generuje dokładne wyniki oraz zachowuje nieściśliwość płynu.

Podsumowując generowanie algorytmów do wizualizacji pola prędkości jest bardzo złożonym i skomplikowanym procesem. Wymaga znajomości metody użytej do rozwiązanie równań przepływu. Następnie zaprojektowanie całej sceny z modelem oświetlenia i pracą kamery. Kolejnym ważnym etapem jest wybór metody całkowania, która wpływa na jakość i czas generowanych wizualizacji. W mojej opinii metody AB 3-rzędu ze względu na optymalny czas generowania i jakość otrzymanych wyników są doskonałym narzędziem do przeprowadzenia procedury całkowania pola prędkości w czasie. Dobierając metodę stosowaną do interpolacji decyduję się na użycie różnych struktur danych umożliwiających szybkie wykonanie wymaganych kroków algorytmów. Po przeprowadzeniu wnikliwej analizy mogę stwierdzić, iż najważniejszym elementem przy interpolacji jest zachowanie zgodności z dyskretyzacją zastosowaną podczas rozwiązywania równań różniczkowych, ponieważ w przeciwnym wypadku możemy zaobserwować niefizyczne efekty takie jak zmiana masy układu. Reasumując powinniśmy zadbać o to, aby wyniki naszej interpolacji spełniały równania dynamiki płynu, które rozwiązujemy.

9 Bibliografia

Literatura

[1] N. Arslan, V. Tuzcu, S. Nas, A. Durukan: *CFD modeling of blood flow inside human left coronary artery bifurcation with aneurysms*;

[2] H.K. Versteeg, W. Malalasekera: *An Introduction to Computational Fluid Dynamics* Pearson Education 2007;

[3] H. Jasak, PhD 1996: *Error analysis and estimation in the Finite Volume method with applications to fluid flows*;

[4] F. Juretic, PhD 2004: *Error analysis in finite volume CFD*;

[5] R. Teman: *Navier-Stokes equations theory and numerical analysis* North-Holand 1977;

[6] J.P.M. Hultquist: *Constructing stream surfaces in steady 3D vector fields* Numerical Aerodynamic Simulation System;

[7] J. Blinn: *Models of Light Reflection for Computer Synthesized Pictures*, 1977;

[8] A.A.M. Kuijk, E.H. Blake: *Faster Phong Shading via Angular Interpolation*;

[9] V. Girault, P.A. Raviart: *Finite element approximation of the Navier-Stokes equations*, 1979;

[10] John Butcher: *Runge–Kutta methods for ordinary differential equations*, COE Workshop on Numerical Analysis, Kyushu University, May 2005;

[11] Manuela Utzinger: *Explicit Adams-Bashforth time integration with local time-stepping for damped wave equations*, February 2011;

[12] James F. Prince: *Lagrangian and Eulerian Representations of Fluid Flow*, June 2006;

[13] Alicja Marzec, Marta Muszalik: *Profilaktyka choroby niedokrwiennej serca*;

[14] Marianna Janion: *Profilaktyka pierwotna chorób układu krążenia*;

[15] Ning Yang, Kambiz Vafai: *Modeling of low-density lipoprotein (LDL) transport in the artery—effects of hypertension*,November 2005;

[16] Tadeusz Tomczak, Katarzyna Zadarnowska, Zbigniew Koza, Maciej Matyka, Łukasz Mirosław: *Acceleration of iterative Navier-Stokes solvers on graphics processing units*, May 2013;

Zewnętrzne Oprogramowanie:

[17] Marcin Krotkiewski: *Algorytm znajdujący do której komórki należy losowy punkt, octree.* http://milamin.sourceforge.net/downloads;

[18] Firma Vratis: *Oprogramowanie Virtus.* http://virtus.vratis.com/;

[19] Firma Vratis: *Oprogramowanie SpeedIt Flow.* http://vratis.com/;

[20] Open Foam Fundation: *Oprogramowanie do rozwiązywania równań różniczkowych metodą objętości skończonej.* http://www.openfoam.com/;

[21] Triangle: *Oprogramowanie do generowania siatek trójkątnych w 2D.* http://www.cs.cmu.edu/ quake/triangle.html;

[22] John Tsiombikas: *Algorytm kd-tree.* http://nuclear.mutantstargoat.com/;

Dodatki

A Zewnętrzne oprogramowanie

W tym rozdziale opiszę używane zewnętrzne oprogramowanie oraz podam linki do stron. Jeżeli ktoś jest zainteresowany większą ilością informacji oraz przykładami zachęcam do zapoznania się z oficjalną dokumentacją.

A.1 Virtus

Jest to produkt firmy Vratis. Jest to wtyczka do przeglądarki działająca na zasadzie klient-server w chmurze, pozwalająca na pobieranie obrazów medycznych z serwera. Następnie umożliwiające generowanie i zapisywanie siatek powierzchniowych w formacie stl. Virtus daje możliwość zlecania zadań na klastrze obliczeniowym firmy Vratis. Obecnie daje możliwość liczenia prostych filtrów na obrazach takich jak dyfuzja. Dodatkowo daje możliwość uruchomienia netgena i snapyHexMesh po stronie serwera i pobierania siatek wolumetrycznych. Następnie umożliwia zadanie warunków brzegowych i początkowych, oraz wykonanie symulacji przepływu krwi na serwerze. Ostatni moduł tego oprogramowania umożliwia wizualizacje pól skalarnych i wektorowych dla naczyń krwionośnych. Zostały w nim zaimplementowane metody opisane w tej pracy. Można pobrać bezpłatną wersję demo ze strony:
www.virtus.vratis.com
Filmiki z oprogramowania można obejrzeć na portalu youtube:
https://www.youtube.com/playlist?list=PLPPKZAixphkqaZTbide-A6ajqTNgCsSyu

A.2 SpeedIt Flow

Jest to biblioteka numeryczna do rozwiązywania równań przepływu na kartach graficznych rozwijana przez firmę Vratis. Biblioteka została zaimplementowana z użyciem technologii Cuda i OpenCL, dlatego umożliwia wykonywanie obliczeń zarówno na kartach Nvidia i AMD. Testować tą bibliotekę można poprzez wtyczkę Virtus, która umożliwia uruchamianie symulacji.
Link do strony www.vratis.com

A.3 OpenFoam

Jest to obszerny pakiet programistyczny umożliwiający rozwiązywanie równań różniczkowych metodą objętości skończonej. Jednym z zastosowań jest liczenie symulacji przepływu. Do-

datkowo umożliwia generowanie siatek wolumetrycznych programem snapyHexMesh. Dla zainteresowanych polecam przerobienie tutoriali, które można znaleźć na stronie:
www.openfoam.com

A.4 Netgen

Program do generowania siatek wolumetrycznych z powierzchniowych. Wspiera dużą ilość formatów. Dodatkowo zawiera przyjazny dla użytkownika interfejs graficzny. Dużą zaletą jest możliwość pobrania kodów źródłowych.

B Struktura katalogów OpenFoam

Przypadek do symulacji w pakiecie OpenFoam składa się z drzewa katalogów. W katalogu głównym (./) znajdują się katalogi z kolejnymi chwilami czasowymi, siatką i plikami kontrolnymi do metody objętości skończonej. W katalogu ./0 wpisujemy warunki początkowe, brzegowe do plików p i U. W ./system znajdują się pliki: fvSchemes i fvSolution w których wpisujemy z jakich schematów objętości skończonej chcemy korzystać. Dodatkowym plikiem jest controlDict w którym znajdują się między innymi informacje na temat symulacji tj. wielkość kroku czasowego i ilość kroków symulacji. Kolejnym katalogiem jest ./constant/polyMesh w którym przechowujemy pliki z zapisaną siatką w formacie OpenFoam. Dla tak przygotowanego przypadku w katalogu głównym możemy użyć jednego z solverów do rozwiązania równań przepływu wpisując w terminalu na przykład ico.

C Implementacja Modeli oświetlenia

Modele oświetlenia zostały zaimplementowane z użyciem biblioteki OpenGL. Jest to niskopoziomowa biblioteka napisana w języku C. Udostępnia one możliwość definiowania składowych światła oraz własności materiałów. Poniżej znajduje się funkcja określająca parametry do wizualizacji. Oznaczenia w kodzie takie same jak w opisach dotyczących modeli światła.

```
1 void MyWindow::DemoLight(void)
{
3     glEnable(GL_LIGHTING);
      glEnable(GL_LIGHT0);
5     glEnable(GL_NORMALIZE);

7     // Parametry modelu oswietlenia
      // ------------------------------------------
9
```

```
    // Natezenia swiatla
    GLfloat Ia[]  = {0.5f, 0.5f, 0.5f, 1.0f};
    GLfloat Id[]  = {0.5f, 0.5f, 0.5f, 1.0f};
    GLfloat Is[]  = {0.1f, 0.1f, 0.1f, 1.0f};

    glLightfv(GL_LIGHT0, GL_AMBIENT, Ia);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, Id);
    glLightfv(GL_LIGHT0, GL_SPECULAR, Is);

    // ------------------------------------------
    // Stale materialu:

    GLfloat material_Ka[] = {0.2f, 0.2f, 0.2f, 1.0f};
    GLfloat material_Kd[] = {0.5f, 0.5f, 0.5f, 0.5f};
    GLfloat material_Ks[] = {0.1f, 0.1f, 0.1f, 1.0f};

    glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, material_Ka);
    glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, material_Kd);
    glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, material_Ks);
}
```

Listing 1: DemoLight

Obliczanie wartości pixela na podstawie koloru i natężenia światła w danym punkcie można zostawić OpenGL włączając tylko odpowiednie flagi:

```
void initGL()
{
    //Wlaczenie swiatla
    glEnable(GL_LIGHTING);
    //Okreslenie z ktrorego swiatla chcemy korzystac
    glEnable(GL_LIGHT0);
    //wlaczamy uzycie wektorow normalnych
    glEnable(GL_NORMALIZE);
    //definiujemy swiatlo dla sceny
    glLightfv(GL_LIGHT0, GL_POSITION, m_Camera.lightPosition );
}
```

Listing 2: InitGL

Bardziej zaawansowaną metodą jest zrobienie tego ręcznie z wykorzystaniem języka GL Shader Language (GLSL). Jest to język umożliwiający bezpośrednie programowanie potoku graficznego.

Działa na podobnej zasadzie jak technologia CUDA i OpenCL. Zaletą GLSL jest to, że jest wspierany przez wszystkie układy graficzne. Oznacza to, że możemy go uruchomić na dowolnym urządzeniu wspierające standard OpenGL. W celu obliczenia koloru dla pixela musimy zdefiniować dwa pliki:

- vertexshader - plik zawiera definicje położeń wierzchołków,
- fragmentshader - w tym pliku definiujemy w jaki sposób obliczać kolor dla poszczególnych wierzchołków.

Poniżej znajdują się przykładowe kody.

```
//wektor normalny
varying vec3 N;
//wektor polozenia wierzcholka
varying vec3 v;
varying vec4 fcolor;

void main(void)
{

   //przypisanie wartosc
   v = vec3(gl_ModelViewMatrix * gl_Vertex);
   N = normalize(gl_NormalMatrix * gl_Normal);
   //okreslenie pozycji na scenie
   gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;
  //przekazanie koloru
   fcolor = gl_Color;

}
```

Listing 3: vertexshader

```
varying vec3 N;
varying vec3 v;
varying vec4 fcolor;

void main (void)
{
   //okreslenie kierunkow
   vec3 L = normalize(gl_LightSource[0].position.xyz - v);
   vec3 E = normalize(-v);
   vec3 R = normalize(-reflect(L,N));
```

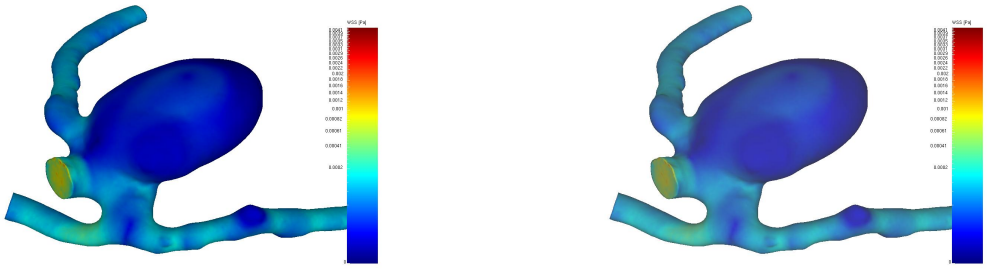
```
12    //oblicznie skladowej swiatla otoczenia:
      vec4 Iamb = gl_FrontLightProduct[0].ambient;
14
      //obliczenie skladowej swiatla rozproszonego:
16    vec4 Idiff = gl_FrontLightProduct[0].diffuse * max(dot(N,L), 0.0);

18    // oblicznie skladowej swiatla rozblysku
      vec4 Ispec = gl_FrontLightProduct[0].specular
20                 * pow(max(dot(R,E),0.0),0.3*gl_FrontMaterial.shininess);

22    // obliczenie calkowitego koloru

24    gl_FragColor = (gl_FrontLightModelProduct.sceneColor + Idiff + Ispec + Iamb)*
       fcolor;
  }
```

Listing 4: InitGL



(a) (b)

Rysunek 29: porównanie sposobów generowania obrazu a) GLSL b) zwykły OpenGl.

Obrazek wygenerowany przy użyciu GLSL posiada odrobinę głębsze cienie. Jest to spowodowane tym, iż programując shadery dokładnie określamy w jaki sposób mają być przetwarzane kolejne prymitywy. Warto, zauważyć, że domyśle parametry ustalone przez OpenGL dają w tym przypadku tylko odrobinę gorsze rezulaty. Do wizualizacji naczyń krwionośnych możemy z dobrym rezultatem stosować obydwie techniki.

D Test poprawności metod do Ode

Poprawność algorytmów całkujących można sprawdzić poprzez porównanie rozwiązania analitycznego z numerycznym. Załóżmy, iż chcemy rozwiązać następujące równanie różniczkowe:

$$\frac{dx}{dt} = v(x,t) = \frac{1}{tx} \tag{29}$$

z warunkiem początkowym x(1)=1, oraz x,t ϵ $\mathbf{R}_+$

- Metoda analityczna:

  zapiszmy równanie po rozdzieleniu zmiennych:

$$xdx = \frac{dt}{t} \tag{30}$$

$$\int xdx = \int \frac{dt}{t} \tag{31}$$

$$\frac{x^2}{2} = ln(t) + C \tag{32}$$

  C - stała całkowania, po uporządkowaniu wyrazów:

$$x(t) = \sqrt{2ln(t) + C} \tag{33}$$

  ponieważ x(1)=1 => C=1, więc

$$x(t) = \sqrt{2ln(t) + 1} \tag{34}$$

  W ten sposób otrzymaliśmy wartość funkcji x dla dowolnego t. Na przykład dla t=6 x(6) = 2.140915 i jest to wynik referencyjny do testu numerycznego.

- Metoda numeryczna:

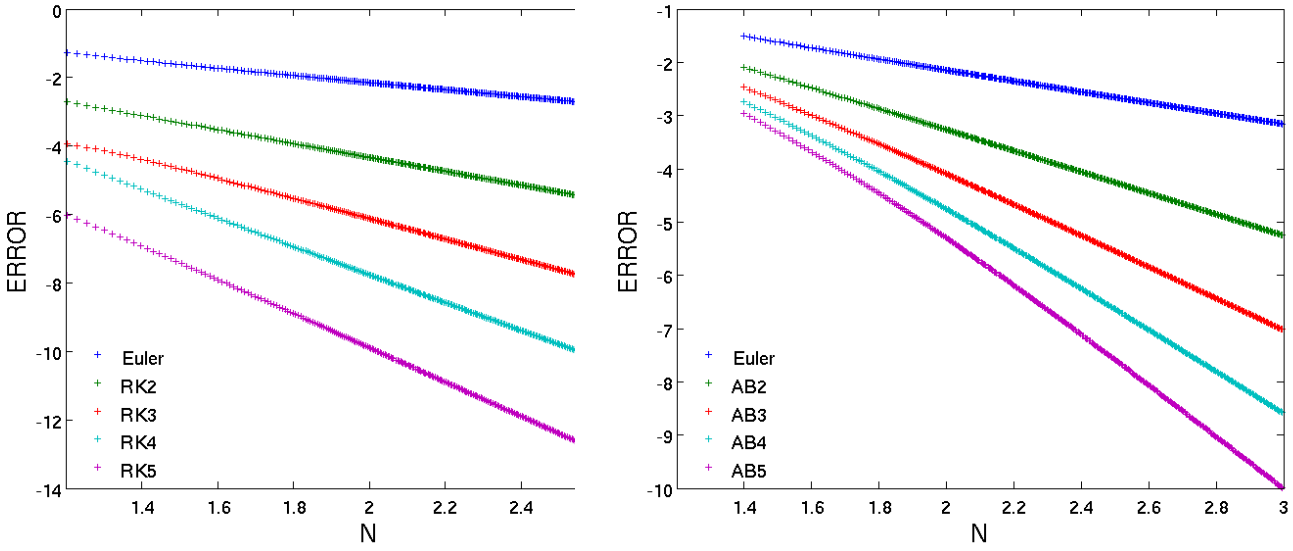
  omówione schematy numeryczne bazują na wyznaczaniu wartości funkcji x w kolejnych punktach t_i. Oznacza to, że musimy przeprowadzić dyskretyzację czasu, najprościej zrobić to tworząc wektor $t = [t_1, t_2, ..., t_n]$, gdzie t_1 jest to czas początkowy, a t_n czas końcowy, dla tego przypadku wynoszą one odpowiednio 1,6. Następnie wystarczy policzyć kojne x_i, x_n jest wynikiem numerycznym.

Błąd wyniku numerycznego można policzyć z równania

$$\delta = |\frac{x(t_n) - x_n}{x(t_n)}| \tag{35}$$

Następnie wyznaczamy zależności δ od liczby kroków - N, lub dt. Zwiększając N zmniejszamy dt czyli powinniśmy otrzymać dokładniejsze rozwiązanie. Poniżej znajdują się wykresy w skali

podwójnie logarytmicznej przedstawiające zależność błędu względnego od liczby kroków dla poszczególnych metod. Dla przykładu N = 2 oznacza 100 kroków, a ERROR = -2 oznacza $\delta = 0.01$. Rząd metody można przybliżyć współczynnikiem kierunkowym narysowanych prostych.



(a) porównanie metod Rungego-Kutty

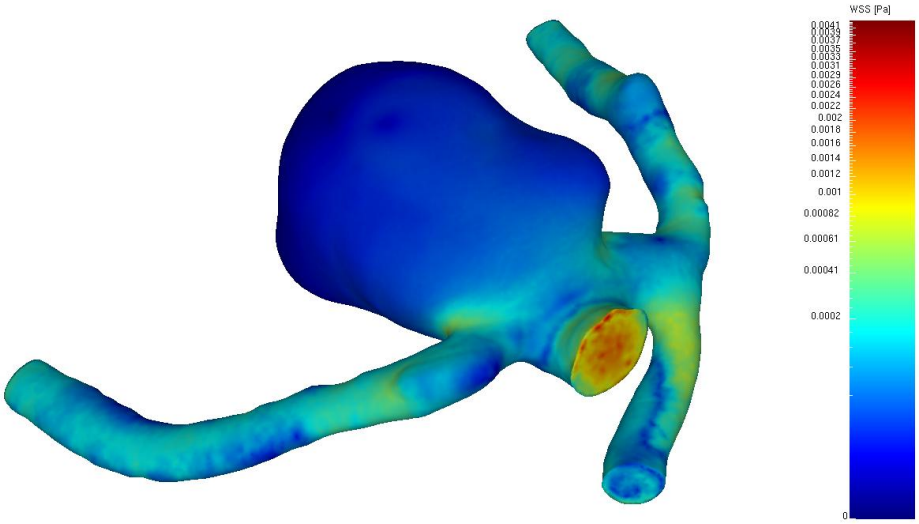
(b) porównanie metod Adamsa-Bashforda

Współczynniki kierunkowe prostych: Euler, -1.0057; RK2, -2.0057; RK3, -2.9979; RK4, -4.0212; RK5, -5.0478; AB2, -1.9942; AB3, -2.9639; AB4,-3.9109; AB5, -4.835;

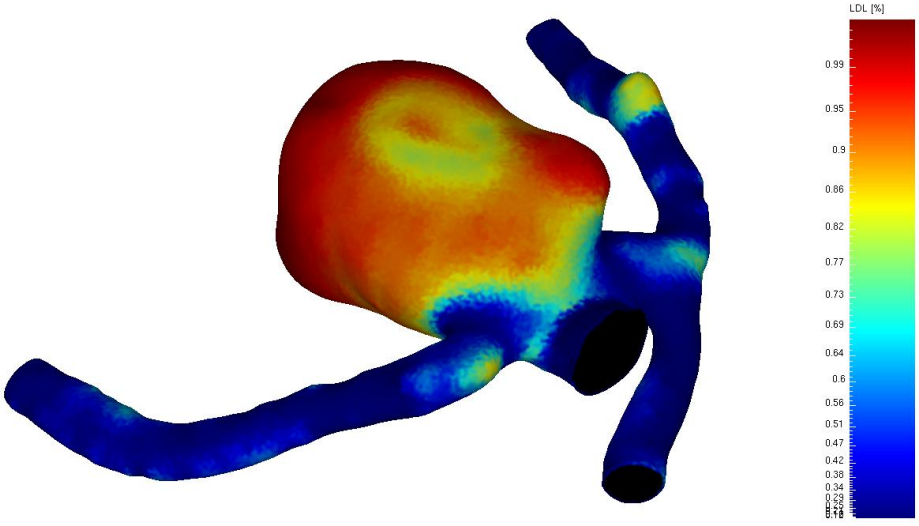
Widzimy, że metody Rungego-Kutty są dokładniejsze i szybciej zbieżne niż odpowiadające im metody Adamsa-Bashforda. Przeprowadzając powyższy test warto pamiętać o tym, że metody AB nie są samo startujące. W takim przypadku potrzebujemy innymi algorytmami obliczyć punkty startowe do tych metod. Najlepszym rozwiązaniem jest użycie do tego metod RK o takim samym rzędzie, gdyż dzięki temu zachowamy odpowiednią zbieżność algorytmu.

E Wizualizacja

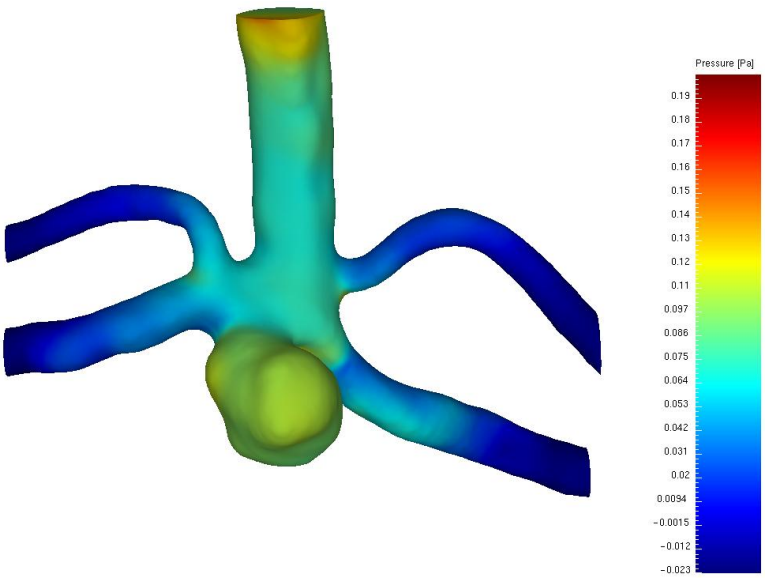
Dodatkowe obrazy dotyczące wizualizacji naczyń krwionośnych i przepływu.



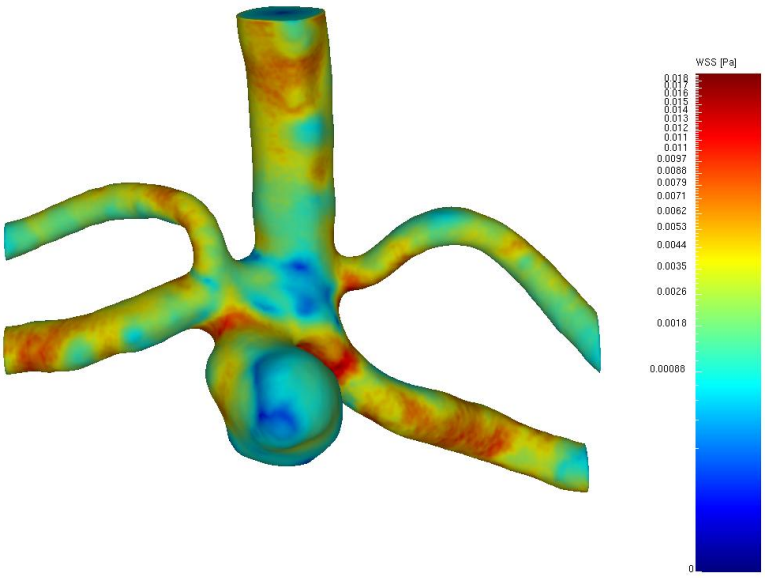
Rysunek 30: Profil WSS dla tętniaka naczynia mózgowego.



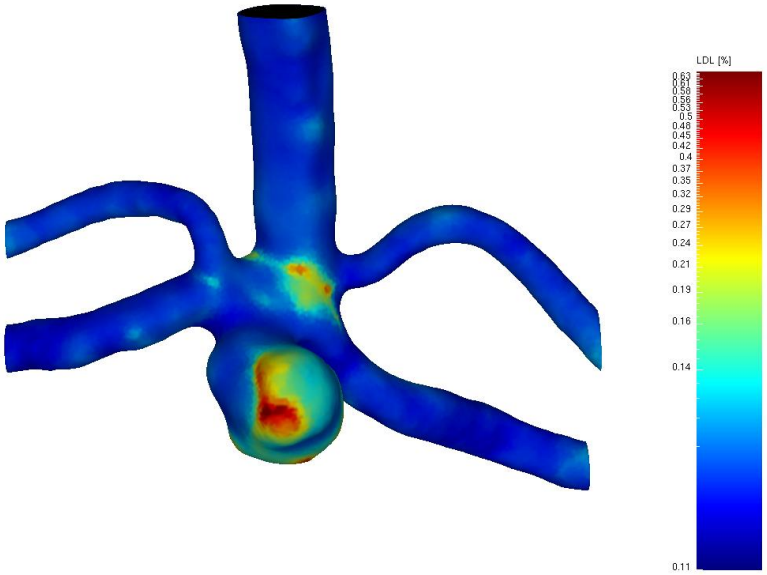
Rysunek 31: Profil LDL dla tętniaka naczynia mózgowego.



Rysunek 32: Profil ciśnienia dla tętnicy podstawnej.

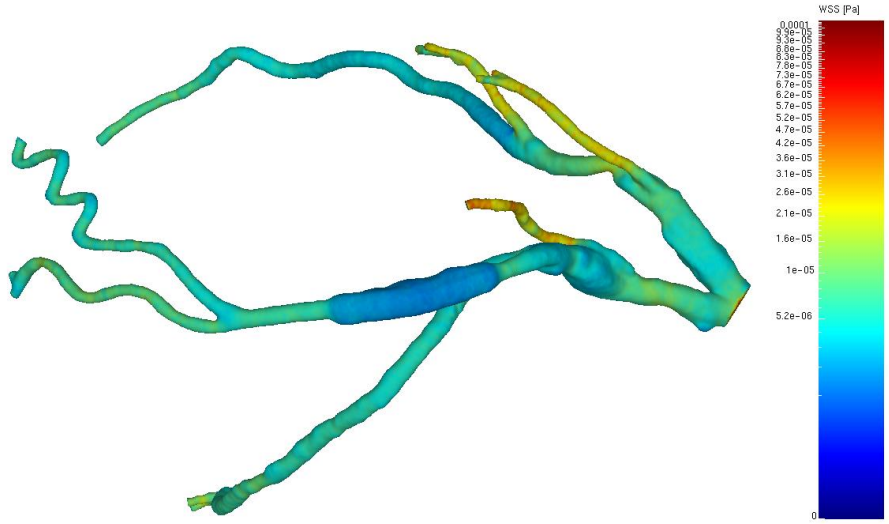


Rysunek 33: Profil WSS dla tętnicy podstawnej.

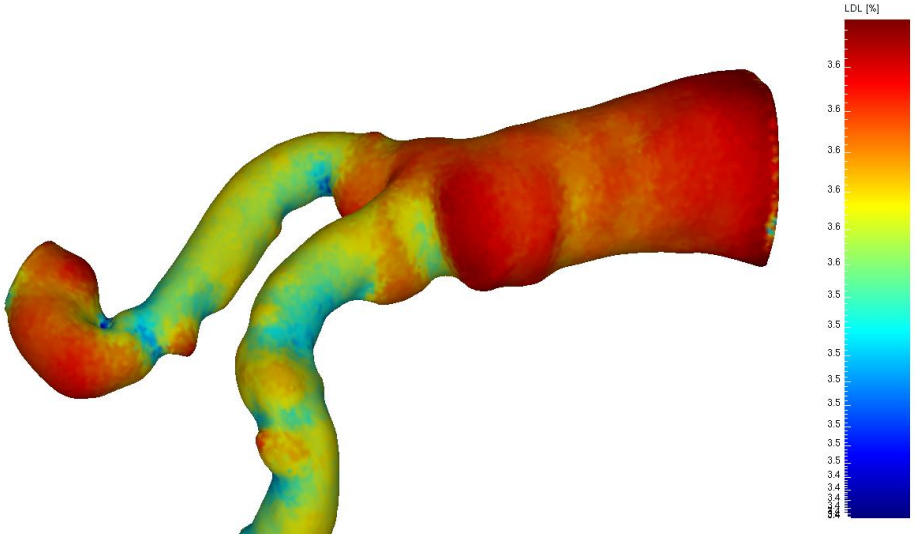


Rysunek 34: Profil LDL dla tętnicy podstawnej.

Rysunek 35: Profil ciśnienia dla naczynia wieńcowego.

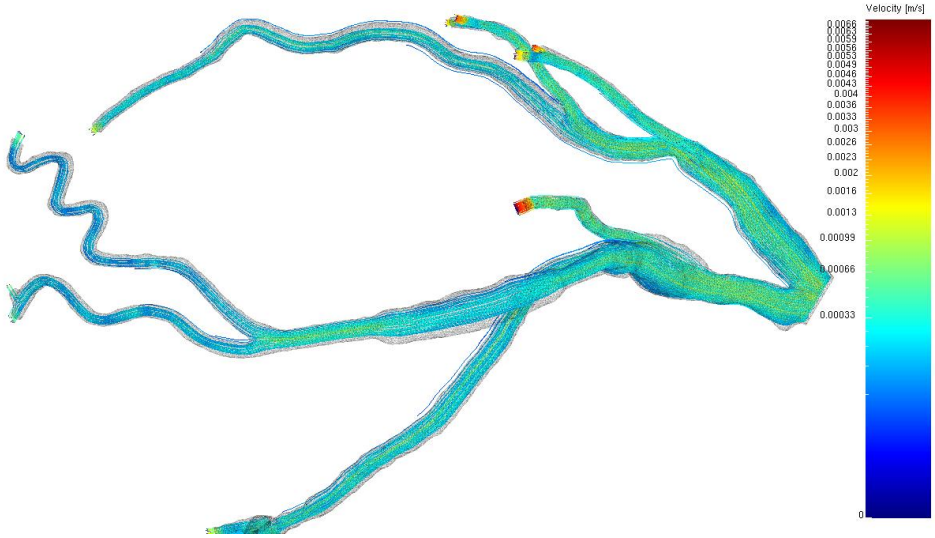


Rysunek 36: Profil WSS dla naczynia wieńcowego.

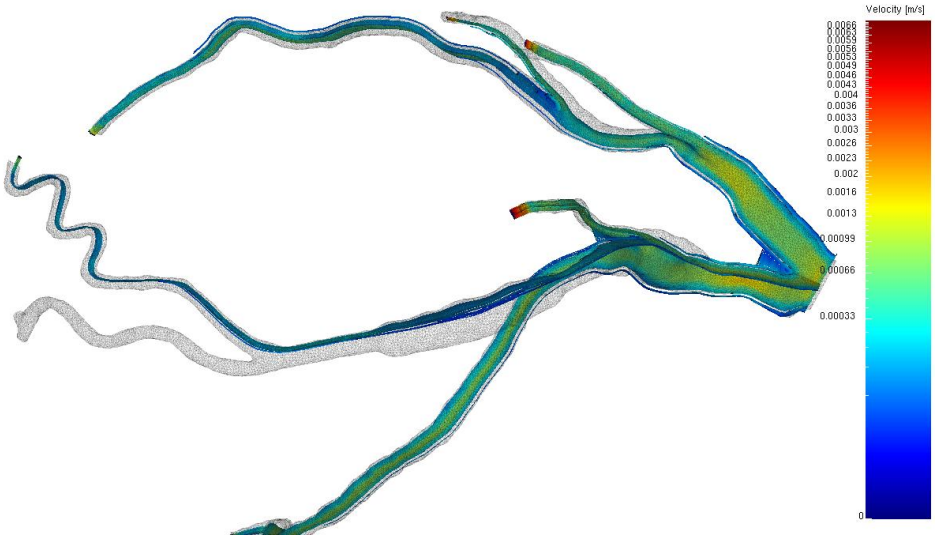


Rysunek 37: Profil LDL dla aorty brzusznej.

Rysunek 38: Cząstki dla naczynia wieńcowego.



Rysunek 39: Linie prądu dla naczynia wieńcowego.



Rysunek 40: Powierzchnia prądu dla naczynia wieńcowego.