INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Bahia

Campus
Camaçari

INSTITUTO FEDERAL DA BAHIA – IFBA
CAMPUS CAMAÇARI-BA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

GUSTAVO ALVES COELHO

DESENVOLVIMENTO DE FERRAMENTA PERSONALIZADA PARA O PÓS-PROCESSAMENTO EM SIMULAÇÕES DE FLUIDO

Camaçari - BA
2023

GUSTAVO ALVES COELHO

DESENVOLVIMENTO DE FERRAMENTA PERSONALIZADA PARA O PÓS-PROCESSAMENTO EM SIMULAÇÕES DE FLUIDO

Projeto de pesquisa apresentado ao Curso de Bacharelado em Ciência da Computação como requisito parcial para aprovação da disciplina TCC sob orientação do Prof. Fábio Marques.

Camaçari
2023

C672 Coelho, Gustavo Alves
Desenvolvimento de ferramenta personalizada para o pós-processamento em simulações de fluido / Gustavo Alves Coelho . – .
53 f.

TCC (Graduação em Bacharelado em Ciência da Computação) – Instituto Federal de Educação, Ciência e Tecnologia da Bahia, Camaçari, 2023.
Orientação: Prof. Dr. Fábio Marques da Cruz.

1. Simulações de fluido. 2. OpenFOAM - software. II. Título

GUSTAVO ALVES COELHO

DESENVOLVIMENTO DE FERRAMENTA PERSONALIZADA PARA O PÓS-PROCESSAMENTO EM SIMULAÇÕES DE FLUIDO

Trabalho de Conclusão de Curso apresentado ao Instituto Federal de Educação, Ciência e Tecnologia da Bahia, Campus Camaçari, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Aprovado em 18 de agosto de 2023.

BANCA EXAMINADORA

Documento assinado digitalmente
gov.br FABIO MARQUES DA CRUZ
Data: 19/07/2024 08:38:25-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Fábio Marques da Cruz - Orientador
Instituto Federal de Educacão Ciência e Tecnologia da Bahia (Campus Camaçari)

Documento assinado digitalmente
gov.br LUCIO MARCOS SILVA DOS SANTOS
Data: 02/08/2024 19:00:21-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Lucio Marcos Silva dos Santos
Instituto Federal de Educação Ciência e Tecnologia da Bahia (Campus Camaçari)

Larissa N. V. Carneiro

Documento assinado digitalmente
gov.br LARISSA NATALIA DAS VIRGENS CARNEIRO
Data: 19/07/2024 10:33:07-0300
Verifique em https://validar.iti.gov.br

Profa. Ma. Larissa Natalia das Virgens Carneiro
Instituto Federal de Educação Ciência e Tecnologia da Bahia (Campus Camaçari)

Documento assinado digitalmente
gov.br FERNANDA REGEBE CASTRO
Data: 29/07/2024 20:57:33-0300
Verifique em https://validar.iti.gov.br

Prof. Dra. Fernanda Regebe Castro
Instituto Federal de Educação Ciência e Tecnologia da Bahia (Campus Camaçari)

Camaçari - BA
2023

RESUMO

Este projeto tem como objetivo fundamental o desenvolvimento de uma ferramenta personalizada para o pós-processamento de simulações de fluido, preenchendo uma lacuna identificada na pesquisa interdisciplinar que requer conhecimento em física, matemática e computação. A motivação para esta iniciativa surgiu da falta de ferramentas adequadas para lidar com as complexidades das simulações de fluido. A ferramenta em desenvolvimento será capaz de analisar arquivos do *software OpenFOAM*, contendo informações cruciais para as simulações de fluido. Isso simplificará substancialmente o processo de análise, eliminando tarefas que anteriormente eram realizadas manualmente ou com vários *softwares* diferentes. Resumidamente, este projeto representa um avanço significativo na pesquisa de simulações de fluido, oferecendo uma solução personalizada que economiza tempo e aumenta a precisão das análises.

Palavras-chave: Simulação de fluidos; ferramenta personalizada; pós-processamento; OpenFOAM; eficiência.

ABSTRACT

This project aims at developing a customized tool for post-processing fluid simulations, addressing an identified gap in interdisciplinary research that requires expertise in physics, mathematics, and computer science. The motivation for this initiative stemmed from the lack of suitable tools to handle the complexities of fluid simulations. The tool in development will be capable of analyzing files from the OpenFOAM software, containing vital information for fluid simulations. This will significantly streamline the analysis process, eliminating tasks that were previously done manually or with various different software programs. In summary, this project represents a significant advancement in fluid simulation research by offering a customized solution that saves time and enhances the accuracy of analyses. It is expected that this tool will stimulate new discoveries and insights, benefiting researchers and professionals and contributing to notable progress in the understanding of fluid simulations.

Keywords: Fluid simulations; customized tool; post-processing; OpenFOAM; efficiency.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

O objetivo deste trabalho é discutir o desenvolvimento de uma ferramenta personalizada para o pós-processamento de simulações de fluidos realizadas no OpenFOAM, incluindo capacidade de realizar operações matemáticas, estatísticas, gráficos e exportação de dados.

Para o desenvolvimento desta pesquisa é necessário compreender o que é um fluido, como ocorre sua movimentação, seu comportamento e todas as três etapas envolvidas durante uma simulação, sendo elas, pré-processamento, processamento e o pós-processamento. O foco desta pesquisa é o estudo do pós-processamento, etapa em que se é feita uma análise analítica do que é simulado, utilizando de técnicas computacionais, com isso, o desenvolvimento de um *software* se fez essencial, desde que, com a capacidade de ler e compreender os arquivos de dados disponibilizados pela etapa de processamento se torna possível compreender os dados estatísticos, como desvio padrão, velocidade média, entre outras funções de uma simulação computacional de fluido.

As medidas estatísticas, matemáticas e todos os conceitos utilizados estarão disponíveis nos respectivos tópicos da revisão literária. Realizar o estudo do pós-processamento de dados exige a compreensão de cada parte que está presente no arquivo de saída e saber como operar com o mesmo, algo que será minimizado pelo desenvolvimento do *software* que será capaz de formatar e entregar arquivos com as informações desejadas, como as coordenadas dos pontos, séries temporais, cálculo das velocidades médias instantâneas, entre outras, agilizando e contribuindo, assim, para o avanço do estudo na área.

O estudo da mecânica de fluidos teve seus primeiros experimentos executados de maneira manual, devido ao fato da impossibilidade desse estudo na época, assim, dando origem ao estudo da hidráulica, que é responsável por compreender o movimento de líquidos dentro de um duto, canais ou outros dispositivos. Já a hidrodinâmica é a ciência que estuda e relaciona os movimentos de fluidos e as forças que causam esse movimento. (FORTUNA, 2000 p.19).

A realização de experimentos em fluidos é algo relativamente recente. O estudo dos fluidos deu seus primeiros sinais em 283 e 133 AC na cidade de Pergamon, atual Turquia, porém, os estudos e materiais desses construtores foram perdidos com o tempo, e o primeiro registro a ser considerado para o estudo da mecânica dos fluidos foi feita pelo matemático grego Arquimedes em 285 a 215 AC (ÇENGEL; CIMBALA, 2015, p. 6). Com isso é

perceptível que já havia motivação, contudo, ainda lhes faltava algo para poder realizar as operações matemáticas envolvendo o assunto.

2 JUSTIFICATIVA

O estudo da dinâmica de fluidos tem se mostrado relevante para diversas áreas, estando presente em diversas aplicações do dia a dia, desde as mais simples até as mais complexas, como no aspirador de pó, numa aeronave supersônica, na tubulação das casas, entre outras. Também está relacionado com projeto de aeronaves, embarcações, submarinos, foguetes, motores a jato, turbinas eólicas, dispositivos biomédicos, refrigeração de componentes eletrônicos, transporte de combustíveis, entre tantas outras funcionalidades (ÇENGEL; CIMBALA 2015. p 5).

O pós-processamento exige o uso de ferramentas ou algum método capaz de compreender os dados informados, com isso foi feita uma pesquisa de *softwares* com essa capacidade e foi encontrada a ferramenta ParaView, que é recomendada pelo OpenFOAM. Sendo ela capaz de gerar cálculos, visualizar dados, gerar gráficos da simulação em tempo real, entre outras funcionalidades. Contudo, no ParaView é feita uma análise a partir de arquivos que compõe toda a simulação, enquanto este estudo irá fazer uma análise do arquivo de velocidades em determinado local, agindo de maneira distinta e possibilitando o estudo de dados mais precisos para determinado momento (GREENSHIELDS, 2023).

Assim, este projeto busca desenvolver uma ferramenta capaz de calcular a partir dos dados momentâneos a velocidade média, o desvio padrão, gerar arquivos com informações relevantes, como: coordenadas, as suas respectivas velocidades e série temporal de maneira simples, com uma interface que será capaz de unir as operações necessárias para um estudo simplificado, direto e eficaz.

Sendo, então, primordial colaborar com tal tema a partir do desenvolvimento deste projeto que se faz tão importante pelo alto grau de complexidade do arquivo que contêm os dados que devem ser interpretados, que faz com que se torne inviável o estudo sem o uso de algum processo computacional e sendo esse capaz de unir técnicas capazes de realizar operações matemáticas, estatísticas e gerar arquivos formatados de acordo com dados desejados ideal para o estudo na área.

3 PROBLEMA OU QUESTÃO DE PESQUISA

Como obter dados ou arquivos específicos do pós-processamento em uma simulação de fluidos que usa o OpenFOAM como processador, a partir dos arquivos de dados contidos no diretório *PostProcessing* referente à realização de uma simulação de fluido.

4 OBJETIVOS

4.1 OBJETIVO GERAL

Desenvolver ferramenta personalizada para a obtenção de dados ou arquivos específicos a partir do arquivo de pós-processamento em simulação realizada no OpenFOAM.

4.2 OBJETIVOS ESPECÍFICOS

a) Compreender conceitos relacionados com os fluidos e sua simulação.

b) Compreender o arquivo de saída oferecido pelo OpenFOAM.

c) Elaborar rotinas em Linguagem C para calcular velocidade média e desvio padrão e desenvolver técnicas para realizar a formatação de arquivos de saída.

d) Desenvolver interface gráfica utilizando Python.

5 REVISÃO DE LITERATURA

5.1 HISTÓRIA DA MECÂNICA DE FLUIDOS E PRINCIPAIS AUTORES

O uso da água sempre foi essencial para o ser humano, assim como sua manipulação. O estudo dos fluidos surgiu a partir da necessidade de gerir o uso da água para consumo, uso doméstico e irrigação de plantas. A primeira contribuição reconhecida na área de mecânica dos fluidos data da época do matemático grego Arquimedes (285 a 212 a.C.) (ÇENGEL; CIMBALA 2015, p. 8).

Uma das maiores e mais importante construções já encontradas data entre 283 e 133 a.C, localizada na cidade Helenística Pergamon, hoje conhecida como Turquia, que consiste em uma série de tubulações de chumbo e argila pressurizadas, com até 45 quilômetros de comprimento e que operava com pressão maior que 1,7MPa (180 metros de altura de carga). Contudo, o nome da maioria dos construtores envolvidos se perdeu com o tempo (ÇENGEL; CIMBALA 2015 p. 6).

Durante a Idade Média houve a expansão do maquinário hidráulico de forma vagarosa. Foram desenvolvidas bombas de pistão com fim de realizar a remoção de águas das minas e moinhos movidos a água e vento foram aprimorados para moer grãos, forjar metais e outras tarefas, substituindo o esforço humano por tais dispositivos, possibilitando, inclusive, a posterior Revolução Industrial (ÇENGEL; CIMBALA 2015, p. 6).

Entre os séculos XIV e XVI houve um contínuo desenvolvimento dos sistemas e máquinas de fluidos, contudo, o fato que foi mais importante foi o aperfeiçoamento do método científico e sua implementação em toda a Europa. Os autores Simon Stevin (1548-1617), Galileo Galilei (1564-1642), Edme

Mariotte (1620-1684) e Evangelista Torricelli (1608-1647) foram os primeiros a aplicar métodos científicos aos fluidos quando investigaram distribuições de pressão hidrostática e vácuo (ÇENGEL; CIMBALA 2015 p. 8).

O médico Jean Poiseuille (1799-1869) conseguiu medir com precisão o escoamento de fluidos múltiplos em tubos capilares. Já na Alemanha, o cientista Gotthilf Hagen (1797-1884) conseguiu definir a diferença entre escoamento laminar e turbulento em tubulações. Contribuindo e seguindo este estudo, Lord Osborne Reynolds (1842-1912) desenvolveu o número adimensional, que leva seu nome. Seguindo este novo estudo e em paralelo ao de

Navier, George Stokes (1819-1903) completou a equação geral do movimento dos fluidos com atrito (ÇENGEL; CIMBALA 2015, p. 8).

Já no fim do século XX, com a chegada dos computadores digitais, se tornou possível a resolução de operações grandes e complexas, como a modelagem do clima global, ou realização da otimização do projeto de uma pá de turbina. Com isso, tornou-se possível a utilização desses mesmos computadores para simulações de fluidos, trazendo uma gama de possibilidades e benefícios para a sociedade (ÇENGEL; CIMBALA, 2015 p. 8).

Pode-se considerar o meado do século XX como a época de ouro das aplicações da mecânica de fluidos, contribuindo, assim, para a expansão da aeronáutica e dos setores químico, industrial e de recursos hidráulicos, e levando a mecânica de fluidos a novas áreas (ÇENGEL; CIMBALA, 2015, p. 8).

5.2 CONCEITOS DE COMPUTAÇÃO E FERRAMENTAS UTILIZADAS

Para realizar o desenvolvimento da ferramenta personalizada para o pós-processamento de simulações de fluido é necessário compreender conceitos de computação, como: programação estruturada, programação orientada a objeto, interface gráfica, linguagem de programação C, linguagem de programação Python e comandos de sistema utilizados no sistema operacional Windows 11, entre outros.

5.2.1 Software e Linguagem de Programação

Os programas que são instalados no computador são todos *Softwares*, ele que é o meio virtual do computador. Sendo esses programas formados por linhas de comando que controlam o processamento do computador, fazendo com que ele siga uma lógica de execução a partir de uma linguagem de programação, esta que é semelhante a um idioma que faz o contato entre o programador e o computador, sendo através desta ferramenta que o criador de *software* desenvolve os programas de uso geral ou específico (NETTO 2005, p. 11).

Existem dois tipos de software, os de sistema (programas que tem como fim o funcionamento adequado da máquina, a partir de instruções operacionais, sempre seguindo uma lógica de programação) e os de aplicativo (programas com determinadas aplicações definidas pelo desenvolvedor, atuando como meio da interação humano computador).

As linguagens de programação são definidas em três níveis: baixo, médio e alto. O nível é definido a partir do grau de proximidade entre a linguagem e o idioma comum, em geral inglês. Quanto menor o nível mais distante é a linguagem de programação do entendimento comum, sendo de difícil compreensão, um exemplo é a linguagem Assembly. Já um exemplo de alto nível pode ser a linguagem de programação Java, em que se é possível ler o código e visualizar palavras que carregam consigo seu próprio significado e facilitando seu uso e implementação.

5.2.2 Técnicas de computação utilizadas

O desenvolvimento deste software conta com conhecimentos de programação e da linguagem de programação C que serão descritos aqui, a partir da leitura do livro Algoritmos e Programação em linguagem C (SOFFNER 2013). Para compreender o funcionamento deste produto, devemos ter ciência de alguns conceitos básicos de programação, como o que é uma variável, o que é o tipo de um dado, o que são tipos primitivos e alguns conceitos e técnicas mais específicas das ferramentas que serão utilizadas, sendo elas a linguagem de programação C e Python.

Variável é o termo dado ao componente que guarda os dados trabalhados pelo programa, logo é um nome dado a um dado, exemplo seria, a informação x significa *hello world,* logo, se eu consultar x encontrarei o valor dos dados contido nele, que é *hello world* e eu poderia transferir essa informação de uma variável para outra, então se eu informar ao programa que y é igual a x, ambos terão os dados semelhantes.

Ao se definir uma variável é necessário que se defina o tipo desta variável, havendo os tipos primitivos, eles que tipificam os dados de maneira geral em números, valores booleanos, palavras ou letras. Se o tipo de variável x que contém a informação *hello world* fosse consultado seria encontrado o tipo *string*, ela que define que a variável possui palavras como informação. Se criarmos uma nova variável chamada de primeira, que contenha apenas a primeira letra da palavra *hello*, que seria h, seria perceptível que seu conteúdo seria tal letra e seria caracterizado como *char*, que define que aquela variável carrega com si apenas um caracter, podendo ser um número ou uma letra.

Já se o intuito da variável é realizar cálculos ou guardar um dado numérico é possível utilizar dos tipos: inteiro ou int, *float* ou Real e *double* ou Real de precisão dupla. O tipo inteiro permite números, eles que tem algumas regras, como: ser um número inteiro, sem

casas decimais, sendo que o inteiro varia de -32768 e 32767. Já o *float* ou ponto flutuante varia de 3.4e-38 a 3.4e+38 e trata de números reais. Enfim o *double, que* é um tipo de dado com grande precisão, variando de -1.7E-308 a 1.7E+308.

Na programação ao se deparar com situação que podem se repetir se aplica uma técnica de repetição, sendo esta técnica de suma importância para o desenvolvimento da maioria dos *softwares,* é a partir desta que se tornou possível alcançar grandes poderes computacionais. Ela consiste em permitir a repetição de certo trecho de código atrelado à uma condição, que caso seja alcançada irá encerrar a execução desse ciclo. Logo, é a partir de uma estrutura de repetição que podemos trabalhar com muitos dados que possuem padrão, permitindo assim, realizar operações com muitos dados e com eficiência.

Avançando temos o conceito de Lista, que é utilizada em grande parte do desenvolvimento deste *software*. Ela que é uma estrutura de dados que permite armazenar um conjunto ordenado de elementos sob um único nome, possibilitando a manipulação desses elementos de forma eficiente. Exemplo seria se eu quisesse guardar cinco notas e um bilhete na minha carteira, a carteira sendo a lista e as notas os valores que preencherão a lista juntamente com o bilhete, logo, temos uma carteira que contém dentro de si notas e um bilhete, e sua estrutura seria: carteira = [nota1, nota2, nota3, nota4, nota5, bilhete], sendo possível manipular toda esta estrutura e permitindo que a partir disso trabalhemos com os dados presentes no arquivo que será interpretado.

5.2.3 Técnicas de Programação associadas à linguagem C

A partir de três tipos de estrutura de controle do fluxo de programação, sendo elas: sequencial, de decisão (ou seleção) e de repetição é possível se desenvolver um programa. A utilização dos mesmos deve seguir algumas regras, com fim de evitar um código confuso e sem controle sistemático. Sendo este conjunto de regras chamada de programação estruturada (SOFFNER, 2013).

Com isso, foi feito o desenvolvimento utilizando a linguagem de programação C com as regras da programação estruturada, utilizando a IDE (*Integrated Development Environment*, Ambiente de desenvolvimento integrado, que é um local preparado para o desenvolvimento de software) Dev-C++ versão 5.11, (LAPLACE, 2023). Resultando em um arquivo executável, este que tem como fim atuar como a parte de processamento do projeto, etapa em que é realizado os cálculos e operações necessárias para seu funcionamento.

A linguagem C consiste em uma linguagem veloz e mais compacta, sendo mais eficiente que outras linguagens, compartilha de características tanto de alto nível, quanto baixo nível, tornando seu uso bastante eficaz (COCIAN 2004 p. 93). Os programas em C geram arquivos executáveis que são o resultado da compilação, sendo um compilador um programa de software que tem como objetivo converter programas escritos em uma linguagem de programação em linguagem de máquina. Permitindo assim que o *hardware* compreenda o que está sendo requisitado, eliminando o processo tedioso de trabalhar com um computador na linguagem binária (COCIAN, 2004 p.44). Sendo então, este executável gerado a partir da compilação uma das partes essenciais do *software* que é responsável por realizar as operações necessárias no código.

A programação em C permite a criação de estruturas ou *structs*, a manipulação de arquivos, conceitos esses que foram aplicados no projeto. A aplicação do *struct* permite a criação de um tipo de dado personalizado, logo, uma estrutura é um tipo de dado definido pelo autor, que pode conter mais do que um tipo de informação na mesma variável que assume esse novo tipo. Assim, permitindo que sejam salvos dados do tipo inteiro, junto com dados do tipo *char*, ou qualquer outro e tornando-os acessíveis ao programador, sendo possível a sua definição a partir do comando *typedef*.

Figura 1: Criação do Struct em C

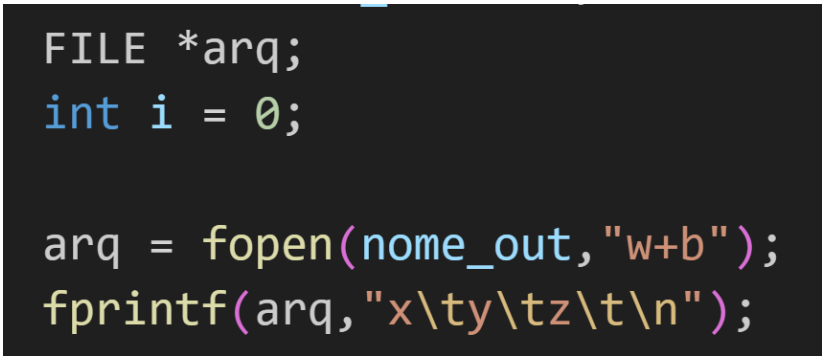
```c
struct _DADOS_
{
    int coluna;
    double soma_x, soma_y, soma_z;
    float x, y, z;
    struct _DADOS_ * proxima;
    struct _DADOS_ * anterior;
};
typedef struct _DADOS_ * dados;
```

Fonte: Software desenvolvido

Na figura 1 é possível observar como é feita a criação de um tipo de dado em C, este que é composto por alguns elementos, sendo eles: posição da coluna que está sendo lida, as velocidades somadas em cada e os endereços dos dados que estão ligados ao mesmo.

A utilização e a escrita de arquivos são essenciais nesta obra, visto que é a partir dela que podemos interagir com os dados e gerar os arquivos formatados em geral. Isso é possível devido a manipulação de arquivos, sendo seus principais comandos: *fopen*, *fscanf*, *fprintf* e *fclose*. O comando *fopen* é o responsável pela abertura do arquivo, sendo possível operar com o mesmo após sua chamada, com o arquivo aberto podemos realizar operações de leitura ou escrita, sendo possível ler os arquivos a partir do comando *fscanf*, enquanto para escrever se utiliza o comando *fprintf,* ao finalizar as operações necessárias se utiliza o *fclose,* que tem como intuito fechar o arquivo em um momento aberto.

Figura 2: Comandos de manipulação de arquivos



Fonte: *Software desenvolvido*

Na figura 2 é perceptível a maneira como é feita a declaração de um dado do tipo *FILE*, a abertura do arquivo com o comando *fopen,* que recebe como parâmetro o nome do arquivo que será aberto e temos como segundo parâmetro "w+b", que define que será escrito um arquivo binário. Já o comando *fprintf* serve para escrever em arquivo, arquivo este que será um arquivo de saída que terá escrito após realização dessa operação: "x y z".

5.2.4 Técnicas de Programação associada à linguagem Python

Programação orientada a objeto, ao se definir um objeto é possível manipulá-lo de diversas maneiras, objeto este é que uma instância de classe (RICARTE, 2001). Utilizando desse conceito foi utilizada a linguagem de programação Python, em que o desenvolvimento da parte gráfica com o objetivo de realizar a interação com o usuário e integrar suas ações (ROCHA, BARANAUSKAS 2003 p.7) com a parte do processamento feita em C (PYTHON 2023).

As bibliotecas e comandos utilizados a partir da tecnologia Python foram: *Pyside6*, *sys, os* e *matplotlib.* A biblioteca *Pyside6* foi utilizada para o desenvolvimento da interface gráfica, logo sendo o responsável por gerar a tela que é utilizada pelo usuário (QT, 2023). A biblioteca *sys* (PYTHON SYS, 2023) é responsável por permitir a execução do mesmo, o encerramento e operações do sistema. Já a biblioteca *os,* é referente ao sistema operacional, em que permite utilizar comandos do mesmo, sendo usado na chamada da parte do processamento e criação dos diretórios (PYTHON OS, 2023). E o *matplotlib* é utilizado para realizar a geração de gráficos (MATPLOTLIB, 2023).

5.2.5 OpenFOAM e Simulação de Fluidos

Este tópico foi desenvolvido a partir do conteúdo disponibilizado no User Guide do OpenFOAM (GREENSHIELDS, 2023)

Para realização das simulações de fluido é utilizado um software específico que é o OpenFOAM, um software de uso livre, gratuito e open source destinado a simulações de fluido computacional. Seu uso é totalmente livre e para poder utilizar o mesmo da maneira adequada foi feito o estudo a partir do user guide disponível no site do OpenFOAM. O *user guide* oferece informações tanto sobre si, quanto sobre simulações em geral, colaborando assim, com o desenvolvimento de toda simulação.

5.2.6 Pré-Processamento

Para compreender o funcionamento do pré-processamento de uma simulação de fluido é necessário ter o conhecimento do que é uma malha ou *mesh*, entender o que é um fluido, algumas características e o que será esperado como resultado, no caso, da simulação de fluidos (GREENSHIELDS, 2023).

Fluido e Tensão serão definidos a partir da leitura do livro Mecânica dos Fluidos Fundamentos e Aplicações. Vol 3 (ÇENGEL; CIMBALA, 2015 p. 2-4). Uma substância que se encontra no estado líquido ou gasoso é um fluido. A diferença entre o sólido e o fluido é seu comportamento perante alterações na tensão de cisalhamento aplicada, logo, sua capacidade de mudar de forma, ou deformar-se. O sólido tem a capacidade de resistir à tensão de cisalhamento, enquanto o fluido não possui tal capacidade, deformando-se continuamente sob a influência da tensão de cisalhamento.

Tensão é definida como força por unidade de área e é determinada ao se dividir a força pela área de atuação. Temos a tensão normal que é a força que atua na superfície por unidade de área com força direcionada para baixo, já a tensão de cisalhamento ocorre ao aplicar uma força tangente à área de superfície. Um fluido ao estar em repouso está sob atuação da tensão normal, que é denominada de pressão, está também com o estado de cisalhamento nulo visto que não há forças aplicadas tangentes à sua área de superfície. Ao imaginar um recipiente quadrado com paredes nos seus lados, na parte inferior e uma abertura superior, a água que se encontra neste recipiente está em repouso, assumindo o formato do próprio recipiente e

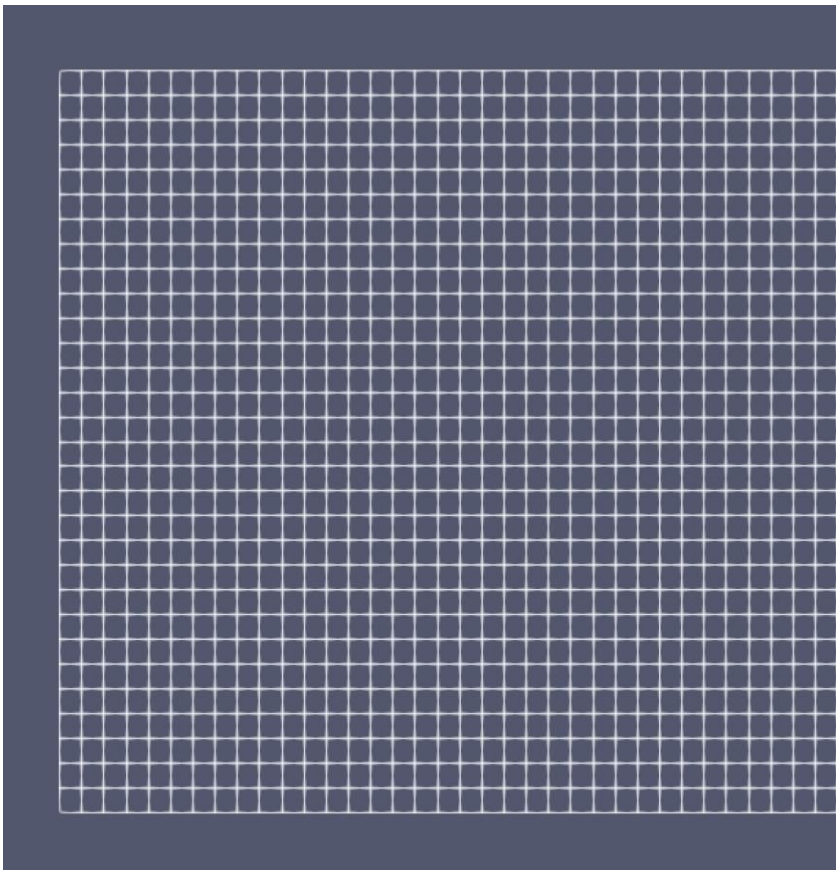
sofrendo a chamada tensão normal, já se o copo for inclinada ou algumas das suas paredes for removida a força de cisalhamento entrará em ação, por conta do líquido se esparramar ou mover-se para manter a superfície livre na sua horizontal, e assim o fluido começará a se deformar livremente enquanto houver alguma força de cisalhamento, podendo assumir formas aleatórias diversas.

A diferença entre os estados líquido e gasoso consiste na sua capacidade de manter suas moléculas unidas e a maneira como sua deformação ocorre. É de sabedoria comum que os líquidos assumem a forma dos seus recipientes, e isso ocorre por conta de possuírem grupos de moléculas que podem mover-se entre si, mudando de posição constantemente. Apesar da variação de formato, o volume é constante por conta das suas fortes forças de coesão entre as moléculas, fazendo com que o líquido não se separe e mantenha a mesma massa desde o início. Já o gás por conta do seu baixo teor de interação molecular, possuindo moléculas soltas pelo recipiente, assim não sendo capaz de gerar superfícies livres. Ao pensar no mesmo recipiente quadrado que já foi utilizado, ao preencher o mesmo com gás seria observável que ele iria se mover de maneira livre, tentando sempre se expandir, assim saindo do recipiente quando possível e ocupando todo espaço oferecido. Enquanto, em um recipiente fechado, teríamos o gás com seu comportamento expansivo que iria preencher todo o espaço gerando uma pressão interna no recipiente.

Para definição dos arquivos de configuração é necessário compreender como é feito esse estudo e algumas das suas características, como células e malha.

A malha é como uma rede tridimensional formada por pontos que se conectam para criar formas. Cada um desses pontos é chamado de célula. Uma célula é como um pequeno elemento que compõe algo.

Figura 3: Malha de um duto



Fonte: *Software ParaView*

Na figura 3 é possível ver uma malha de duto, esta que foi definida com uma geometria retangular que define bem o contexto do duto.

Essas células são fundamentais para simular o comportamento de fluidos ou outras coisas que se movem. Através delas, é possível dividir o espaço e entender como tudo se encaixa. Semelhante a um local controlado em que se pode manipular o fluido de maneira arbitrária.

É possível configurar as simulações no OpenFOAM a partir da edição dos arquivos de entrada de dados. Sendo necessário escolher um ambiente de desenvolvimento adequado para abrir os arquivos, alguns recomendados são: gedit, vi, nedit, entre outros.

5.2.7 Processamento

Nesta etapa os cálculos desenvolvidos pelos autores do tema são realizados a partir do software *OpenFOAM*. Assim, colocando em ação a teoria da hidrodinâmica e cálculos como o

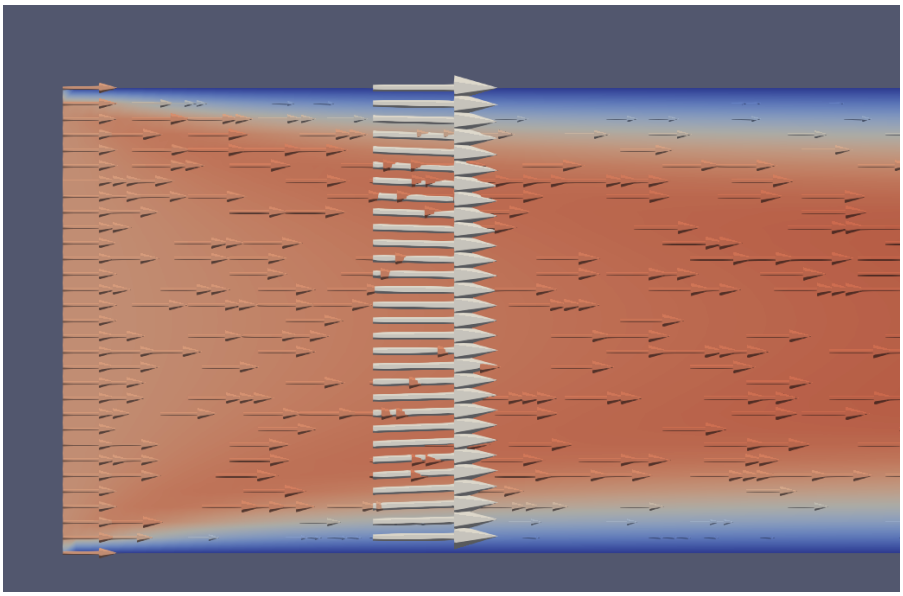
de George Stokes, Lord Osborne Reynolds, entre outros, com fim de realizar as operações matemáticas referentes ao estudo da mecânica de fluidos.

5.2.8 Pós-Processamento

O estudo do pós-processamento é feito após definidas as características do cenário, etapa respectiva ao pré-processamento. Em seguida, seus cálculos são realizados na etapa de processamento, e com isso o resultado nos arquivos de saída possibilitam o estudo da simulação, a partir do uso de softwares como o *ParaView*, *Ensight*, ou outros softwares com o objetivo de realizar a análise do pós-processamento. Sendo esta a proposta neste trabalho. Com fim de entender como será feito esse estudo é necessário compreender o que é um vetor e o que é um recorte, também chamado de *slice* ou *probe.*

O OpenFOAM, responsável pelo processamento, lida com os dados a partir da utilização de uma estrutura multidimensional, sendo este o vetor, que é composto por três posições espaciais, sendo elas: x, y e z. A partir disto em qualquer ponto dentro de uma simulação de fluido será encontrado um vetor, que é composto por três velocidades, uma na direção horizontal, outra na direção vertical e a última na direção perpendicular. Conclui-se, então, que um ponto dentro de uma simulação é um vetor que possui três velocidades e direções distintas.

Figura 4: *Probe* em uma simulação de fluidos

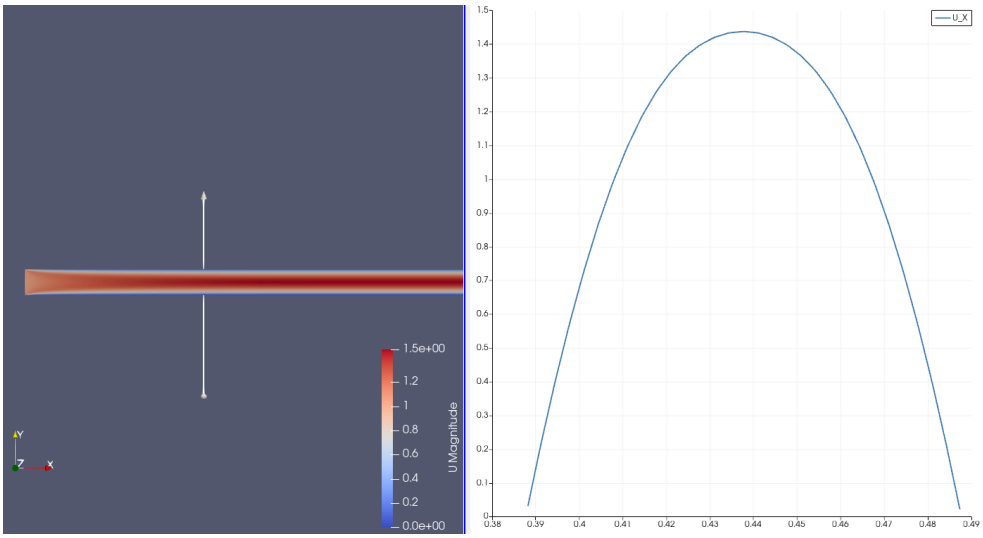


Fonte: *Software ParaView*

Na figura 4 é possível observar a presença de setas apontando para a direita, estas representam a velocidade e direção do fluido. As setas brancas representam a velocidade em um recorte, cobrindo toda uma posição do duto, logo, verificando sua velocidade naquele momento durante toda sua seção.

Para compreender melhor o que é um recorte ou *probe* é preciso imaginar uma torneira com uma mangueira conectada. A torneira ao ser aberta jorra água, logo a água percorre a mangueira, e com isso, se quisermos saber a velocidade do fluido que por lá percorre será necessário definir um local para examinar essa velocidade, visto que a mesma varia de acordo com sua posição relativa a entrada de água, outro ponto é que os vetores mais próximo as paredes tendem a sofrer mais dificuldade, por ter o atrito como uma força contrária que faz com que sua velocidade seja diminuída.

Figura 5: Gráfico de uma simulação de fluidos



Fonte: *Software Paraview*

A figura 5 demonstra uma simulação de fluidos e um recorte na mesma, seguida de um gráfico referente ao recorte. É possível observar que o perfil da velocidade formou uma parábola.

Logo, suas velocidades variam de acordo com sua posição relativa dentro do objeto em que o fluido percorre, logo sua velocidade varia em função da sua posição, sendo assim, necessário realizar o estudo da velocidade a partir de recortes pelo duto, estes que terão posições estratégicas para permitir realizar o estudo de maneira eficaz. Um recorte, então, é a definição dos pontos a partir de uma posição fixa, percorrendo os pontos dentro de um duto de

maneira que seja capaz definir a velocidade em qualquer posição por todo o diâmetro da mangueira, naquele *probe*, assim, definindo a velocidade em dada posição do duto.

O resultado esperado após o estudo da velocidade de um fluido a partir de um recorte feito em um simulação de fluidos é de um gráfico no formato de parabolóide, como pode ser visto na figura 5, que consiste em um gráfico que informa que a velocidade no meio do duto é a mais veloz, visto que ali é o local em que se tem menos atrito, já que se encontra longe das paredes, enquanto as posições que vão se aproximando das paredes já vão sofrendo com o atrito, e logo, a redução de velocidade, justificando assim, o gráfico neste formato.

6 METODOLOGIA

O objetivo deste projeto é desenvolver uma ferramenta personalizada de pós-processamento em simulação de fluidos, com o objetivo de realizar operações matemáticas, estatísticas, analíticas e gerar gráficos de maneira simplificada, assim, contribuindo com o estudo na área. O desenvolvimento desse projeto é feito em etapas, sendo elas referentes a seleção das tecnologias, compreender os dados do arquivo de saída, entender como é feita a manipulação destes dados e a partir dela gerar o cálculo da velocidade média, do desvio padrão, gerações de gráficos e de arquivos formatados.

6.1 SOFTWARE:

Será feito o desenvolvimento do *software*, e com isso deve-se escolher as tecnologias que serão implementadas e como será feito seu uso.

Para compreender melhor o que será feito é necessário verificar o que é essencial para o desenvolvimento do mesmo, logo se observa que é primordial o desenvolvimento de método para realização cálculos e de uma interface gráfica para oferecer ao usuário uma navegação simples e direta. Assim deve-se escolher uma tecnologia veloz, eficiente e capaz de realizar um bom gerenciamento de memória, desde que há uma grande quantidade de dados presentes no arquivo de pós-processamento, dados esses que serão manipulados com diversas operações e para o desenvolvimento da interface gráfica é necessário escolher uma tecnologia de fácil implementação e com uma documentação extensa.

Com isso foi feita a seleção das tecnologias, e para trabalhar com muitos dados de maneira eficaz é necessário escolher uma tecnologia veloz e que seja boa para trabalhar com muitos dados, logo a linguagem C foi escolhida. Sendo esta capaz de realizar operações tanto de alto quanto baixo nível, permite o gerenciamento de memória, escrita e leitura de arquivos binários, sendo uma ferramenta que apresenta grande vantagem em relação a outras (SOFFNER 2013).

Já para o desenvolvimento das telas e gráficos é necessária uma tecnologia veloz e de fácil implementação, que seja capaz de criar uma interface gráfica, gráficos e permitir a execução de comandos de sistema de maneira simples e eficiente. Com isso, Python que é uma linguagem de código aberto, possui uma larga comunidade, sendo também esta uma

tecnologia recente e que possui muitas bibliotecas e extensões, sendo ela escolhida juntamente com algumas tecnologias.Sendo elas *PySide6, Matplotlib, os.*

PySide6 é uma biblioteca para criação de interfaces gráficas, sendo esta biblioteca uma das mais recentes, atualizadas e de fácil implementação (PYPI 2023). Para o desenvolvimento de gráficos foi utilizada a biblioteca Matplotlib, que oferece tecnologias para a geração de gráficos de maneira facilitada (MATPLOTLIB 2023). Já para realizar os comandos de sistema a biblioteca O*s* se faz essencial, desde que é ela que permite realizar as operações de chamada do executável responsável pelo processamento dos dados (PYTHON OS, 2023).

6.2 ARQUIVO DE SAÍDA:

Com fim de compreender o que foi simulado é preciso entender o arquivo de saída, este que é gerado na etapa de processamento. Sendo este arquivo composto por uma formatação padrão. Sendo esta: as posições de recorte ou *probe*, descrição de quantos pontos tem e suas respectivas velocidades nos seus pontos x, y e z em tal momento da simulação, esse que pode ser considerado também como iteração, sendo visível na figura 6.

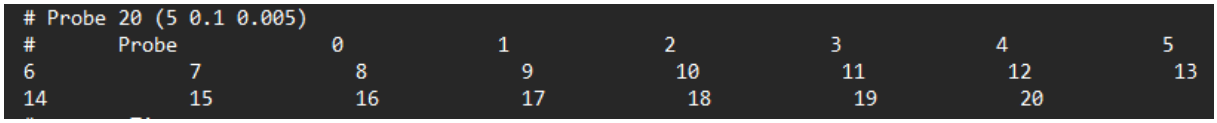
Figura 6: Coordenadas de cada ponto no arquivo

```
# Probe 0 (5 0 0.005)
# Probe 1 (5 0.005 0.005)
# Probe 2 (5 0.01 0.005)
# Probe 3 (5 0.015 0.005)
# Probe 4 (5 0.02 0.005)
# Probe 5 (5 0.025 0.005)
# Probe 6 (5 0.03 0.005)
# Probe 7 (5 0.035 0.005)
# Probe 8 (5 0.04 0.005)
# Probe 9 (5 0.045 0.005)
# Probe 10 (5 0.05 0.005)
# Probe 11 (5 0.055 0.005)
# Probe 12 (5 0.06 0.005)
# Probe 13 (5 0.065 0.005)
# Probe 14 (5 0.07 0.005)
# Probe 15 (5 0.075 0.005)
# Probe 16 (5 0.08 0.005)
# Probe 17 (5 0.085 0.005)
# Probe 18 (5 0.09 0.005)
# Probe 19 (5 0.095 0.005)
# Probe 20 (5 0.1 0.005)
```

Fonte: Arquivo de saída gerado pelo OpenFOAM

Na figura 6 é possível observar que neste arquivo há vinte e um pontos selecionados nas suas respectivas posições percorrendo o que seria o *probe* ou recorte que está definido em uma posição horizontal fixa, sendo aqui a posição referente a x o valor cinco. Ao percorrer seus pontos presentes na posição cinco variando sua posição apenas na vertical é possível compreender a velocidade de tal posição por todo o diâmetro do duto naquele recorte. Após as coordenadas do recorte vem a informação referente aos pontos presentes.

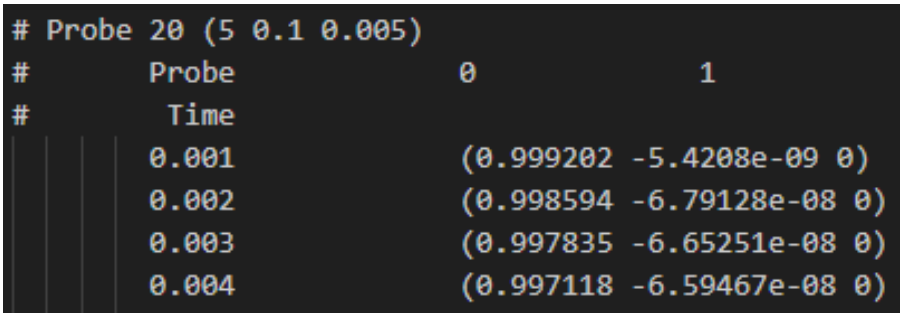
Figura 7: Pontos presentes no arquivo



Fonte: Arquivo de saída gerado pelo *OpenFOAM*

Na figura 7 está sendo descrita a quantidade e ordenação dos pontos no recorte (*probe*), estando eles organizados na mesma ordem que as colunas de velocidade estarão. Sendo perceptível que será ordenado da esquerda para direita as colunas de velocidade que aparecerão em ordem, logo após essa parte no arquivo.

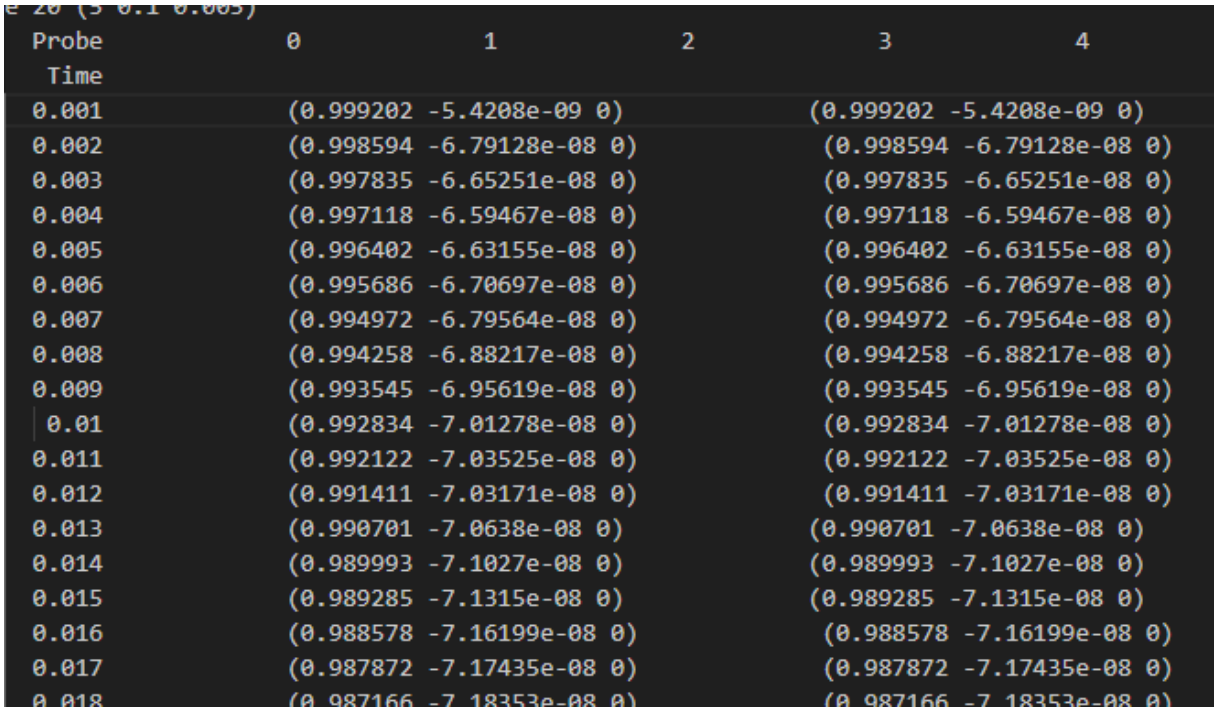
Figura 8: Velocidade em relação ao tempo



Fonte: Arquivo de saída gerado pelo *OpenFOAM*

Na figura 8 estão presentes as informações anteriores e é possível ver os dados referentes aos tempos da simulação e sua velocidade em tal posição do recorte. Cada estrutura presente após o *time* representa sua velocidade nas posições x, y e z, respectivamente. Logo, ao visualizar podemos concluir que na primeira iteração ou no *time* 0.001 na posição do primeiro recorte sua velocidade era de 0.999202 metros por segundo na direção x, horizontal, na posição y, vertical, a velocidade é de -0.0000000054208 ou -5.4208e-09 metros por segundo, já na posição z, na perpendicular, sua velocidade é zero.

Figura 9: Velocidades em relação ao tempo



Probe	0	1	2	3	4
Time					
0.001	(0.999202 -5.4208e-09 0)			(0.999202 -5.4208e-09 0)	
0.002	(0.998594 -6.79128e-08 0)			(0.998594 -6.79128e-08 0)	
0.003	(0.997835 -6.65251e-08 0)			(0.997835 -6.65251e-08 0)	
0.004	(0.997118 -6.59467e-08 0)			(0.997118 -6.59467e-08 0)	
0.005	(0.996402 -6.63155e-08 0)			(0.996402 -6.63155e-08 0)	
0.006	(0.995686 -6.70697e-08 0)			(0.995686 -6.70697e-08 0)	
0.007	(0.994972 -6.79564e-08 0)			(0.994972 -6.79564e-08 0)	
0.008	(0.994258 -6.88217e-08 0)			(0.994258 -6.88217e-08 0)	
0.009	(0.993545 -6.95619e-08 0)			(0.993545 -6.95619e-08 0)	
0.01	(0.992834 -7.01278e-08 0)			(0.992834 -7.01278e-08 0)	
0.011	(0.992122 -7.03525e-08 0)			(0.992122 -7.03525e-08 0)	
0.012	(0.991411 -7.03171e-08 0)			(0.991411 -7.03171e-08 0)	
0.013	(0.990701 -7.0638e-08 0)			(0.990701 -7.0638e-08 0)	
0.014	(0.989993 -7.1027e-08 0)			(0.989993 -7.1027e-08 0)	
0.015	(0.989285 -7.1315e-08 0)			(0.989285 -7.1315e-08 0)	
0.016	(0.988578 -7.16199e-08 0)			(0.988578 -7.16199e-08 0)	
0.017	(0.987872 -7.17435e-08 0)			(0.987872 -7.17435e-08 0)	
0.018	(0.987166 -7.18353e-08 0)			(0.987166 -7.18353e-08 0)	

Fonte: Arquivo de saída gerado pelo *OpenFOAM*

Ao analisar a figura 9 é possível compreender melhor o funcionamento desta interpretação, visto que o *probe*, é referente a quantidade de recortes em cada destas colunas. Logo, se há vinte e um pontos no recorte há vinte e uma colunas com as velocidades de cada ponto em cada tempo. Com a figura 9 é possível descrever o quão complexo é para realizar esses cálculos, visto que, nesse arquivo que será interpretado e usado de modelo há vinte e um pontos de recorte do duto, tendo então cada iteração de *time* vinte e uma colunas, havendo neste arquivo quinze mil tempos, o que resulta em trezentos e quinze mil vetores, sendo que cada um é composto por três velocidades, logo tendo novecentos e quarenta e cinco mil velocidades no arquivo que devem ser analisados para poder realizar o estudo do pós-processamento.

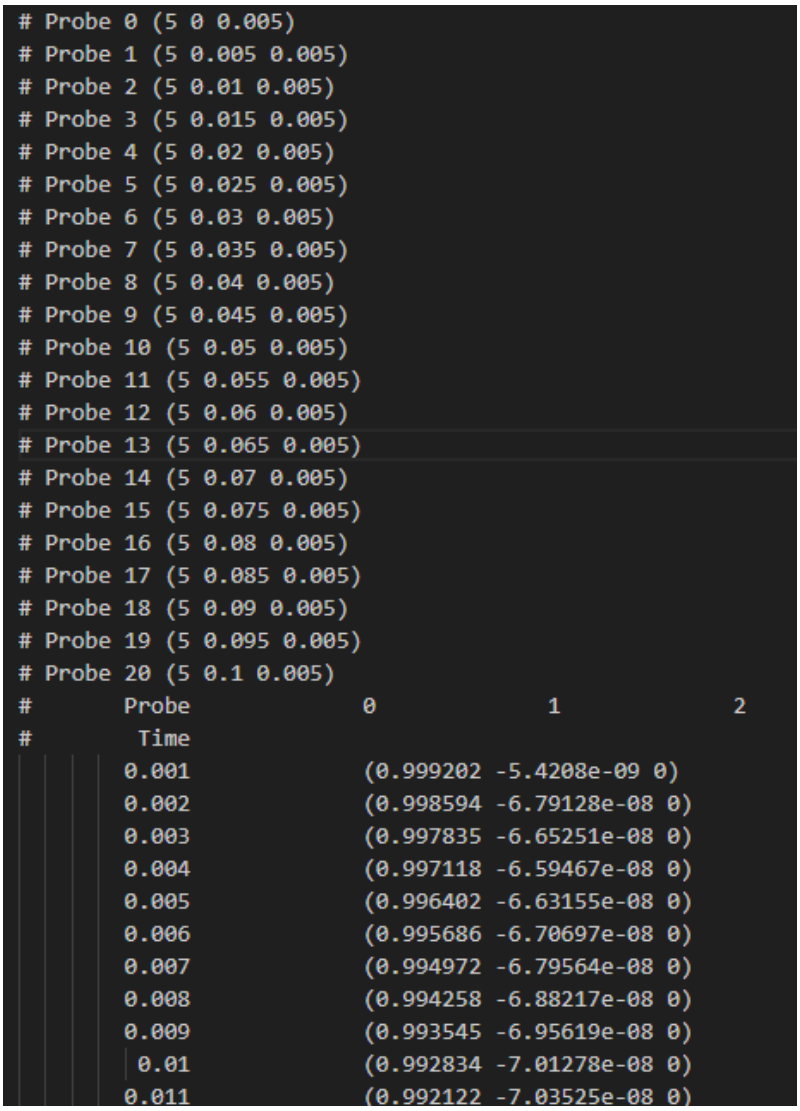
6.3 LEITURA DOS DADOS

Com a compreensão de como o arquivo de saída é escrito percebe-se que há uma grande quantidade de dados. Logo, é essencial que seja feito o desenvolvimento de uma etapa de leitura ao executarmos o *software*, este que tornará possível manipular os dados da maneira desejada. Para realizar a leitura dos dados é feita a estratégia de ler todos os caracteres

presentes no texto a partir do comando *fscanf*, e com isso, carregar os mesmo em uma lista, e com isso poder realizar a manipulação dos mesmos. Logo, uma fila por permitir estrutura ordenada, nós temos do começo do arquivo até o final, sendo a leitura dividida na ordem: coordenadas e os dados (Soffner 2013).

Contudo, essa operação ainda não está muito viável devido ao fato do arquivo de pós-processamento possuir caracteres que tornam o arquivo mais pesado, e consequentemente dificultando o seu estudo. Para realizar o estudo do pós-processamento é necessário considerar apenas os dados referentes à velocidade, posição ou tempo contidos no arquivo de saída, logo os caracteres especiais, ou palavras que não possuem valor na pesquisa podem ser retirados do arquivo. Sendo esta uma parte essencial para o funcionamento adequado do *software*, surgindo a necessidade do desenvolvimento de uma etapa de exclusão e formatação do novo arquivo sem os seguintes caracteres que estão presentes no arquivo: ‘(‘, ‘)’ e ‘#’, além das palavras *Probe, Time,* e as linhas referentes aos recortes.

Figura 10: Arquivo sem a formatação adequada



```
# Probe 0 (5 0 0.005)
# Probe 1 (5 0.005 0.005)
# Probe 2 (5 0.01 0.005)
# Probe 3 (5 0.015 0.005)
# Probe 4 (5 0.02 0.005)
# Probe 5 (5 0.025 0.005)
# Probe 6 (5 0.03 0.005)
# Probe 7 (5 0.035 0.005)
# Probe 8 (5 0.04 0.005)
# Probe 9 (5 0.045 0.005)
# Probe 10 (5 0.05 0.005)
# Probe 11 (5 0.055 0.005)
# Probe 12 (5 0.06 0.005)
# Probe 13 (5 0.065 0.005)
# Probe 14 (5 0.07 0.005)
# Probe 15 (5 0.075 0.005)
# Probe 16 (5 0.08 0.005)
# Probe 17 (5 0.085 0.005)
# Probe 18 (5 0.09 0.005)
# Probe 19 (5 0.095 0.005)
# Probe 20 (5 0.1 0.005)
#        Probe             0             1             2
#         Time
          0.001             (0.999202 -5.4208e-09 0)
          0.002             (0.998594 -6.79128e-08 0)
          0.003             (0.997835 -6.65251e-08 0)
          0.004             (0.997118 -6.59467e-08 0)
          0.005             (0.996402 -6.63155e-08 0)
          0.006             (0.995686 -6.70697e-08 0)
          0.007             (0.994972 -6.79564e-08 0)
          0.008             (0.994258 -6.88217e-08 0)
          0.009             (0.993545 -6.95619e-08 0)
           0.01             (0.992834 -7.01278e-08 0)
          0.011             (0.992122 -7.03525e-08 0)
```

Fonte: Arquivo de saída gerado pelo *OpenFOAM*

Para o desenvolvimento desta etapa é realizada a leitura do arquivo de saída que contém os dados e os caracteres indesejados e a reescrita do mesmo ignorando os caracteres parênteses, *hashtag*, as palavras *Time, Probe* e os números que seguem após a palavra *Probe* que não são coordenadas (GREENSHIELDS, 2023).

Com o método de formatação definido é possível ter todos os dados lidos, então pode-se começar o desenvolvimento de uma ideia de como deve ser feito para conseguir organizar estes dados de maneira eficaz, e para isso será necessário utilizar de estruturas de repetição juntamente com o *fscanf* para realizar essa operação. Ao organizar os dados lidos em estruturas de fila é possível separar os dados da maneira desejada, com isso, foi organizado

que primeiro deveria se ler as coordenadas e logo após os dados, estes que terão seus valores organizados por colunas (Soffner 2013).

6.4 MANIPULAÇÃO DOS DADOS

Após ter a capacidade de ler o texto e organizar o mesmo, torna-se possível operar com os dados presentes no arquivo, assim possibilitando a realização de cálculos, geração de arquivos formatados e os gráficos.

6.5 CÁLCULOS

No estudo do pós-processamento dos fluidos se tem algumas medidas que são importantes para a compreensão do seu resultado, sendo estas a velocidade média e o desvio padrão.

A velocidade média é uma importante medida, desde que é a partir dela que se torna possível compreender como o fluido se comporta, em que direção está indo permitindo avaliar como está o comportamento do mesmo, possibilitando a geração de gráficos e um estudo detalhado do mesmo. Para realização do cálculo da velocidade média é necessário saber sobre a velocidade, a distância ou o tempo. Visto que a velocidade média é calculada a consiste na taxa com a qual o objeto se desloca no momento da sua representação (Knight 2009 p. 39).

Com fim de compreender melhor o cálculo da velocidade média é necessário conhecer o conceito de escoamento em regime estacionário e em regime turbulento, já que inicialmente o sistema encontra-se em repouso e ao iniciar a simulação o fluido é inserido no sistema e seu movimento vem acelerado, sendo este momento indesejado para estudo, desde que o movimento que é esperado é o constante (LAGO, 2010). Sendo este o valor mínimo de iterações ou tempos a serem ignorados, logo sendo subtraído do tempo total e tendo suas velocidades em tais momentos sendo ignoradas. Uma maneira de ilustrar isso seria a partir de um cano que não possui fluido dentro, inserir de maneira constante, assim, inicialmente o fluido dentro do duto entrará de maneira desordenada até que irá manter um padrão, sendo este o momento desejado para a análise, o momento em que o fluido está estacionário.

Sendo a fórmula de: velocidade média é igual a distância dividida pelo tempo, com isso, deve-se somar as velocidades e dividir pela quantidade de tempo subtraindo a quantidade

de iterações a serem ignoradas, então se uma simulação tem quinze mil tempos simulados deve-se somar as velocidades dos pontos desejados e dividir pela quantidade de tempo útil.

Utilizado como medida de dispersão temos o desvio padrão utilizado juntamente com a média. Tendo o desvio padrão como valor mínimo 0, o que informa que não há variabilidade, com isso, pode se afirmar que todos os valores são iguais à média. (CASTRO; FERNANDES; ALMEIDA, 2015). Logo sua obtenção permite verificar a precisão das medições dos resultados obtidos de uma simulação, em que se houver baixo desvio padrão os dados são consistentes e respeitam uma média, se houver um resultado alto também é possível concluir que há alguma inconsistência nos dados, visto que seu valor sai da média padrão, entre outras possibilidades.

O cálculo do desvio padrão se torna possível desde que os dados sejam bem manipuláveis e utilizáveis. A implementação desse cálculo exige que sejam feitas as somas das velocidades menos a média dos valores ao quadrado dividido pela quantidade de pontos (CASTRO; FERNANDES; ALMEIDA, 2015).

6.6 GERAÇÃO DE ARQUIVOS FORMATADOS

A geração de arquivos formatados é essencial para o estudo dos fluidos, visto que é a partir destes arquivos que serão disponibilizados os resultados do processamento dos dados. Os dados que podem ser obtidos a partir de uma simulação são as coordenadas, a velocidade média, a série temporal e o desvio padrão entre os pontos.

Para emissão do arquivo das coordenadas é necessária a realização da leitura dos dados referentes às coordenadas e a sua reescrita em outro arquivo. Funcionando de maneira semelhante aos outros arquivos de saída, a partir da leitura dos dados realizando operações matemáticas em conjunto ou apenas escrevendo no arquivo de destino os valores desejados.

Para a escrita do arquivo de velocidade média é realizada a leitura do arquivo a partir do ponto inicial definido, ignorando as iterações que se deseja evitar, e seus valores são somados, logo em seguida seu valor somado é dividido pela quantidade de tempo útil, logo sem os valores referentes aos momentos ignorados, assim tornando possível a obtenção dos dados requisitados.

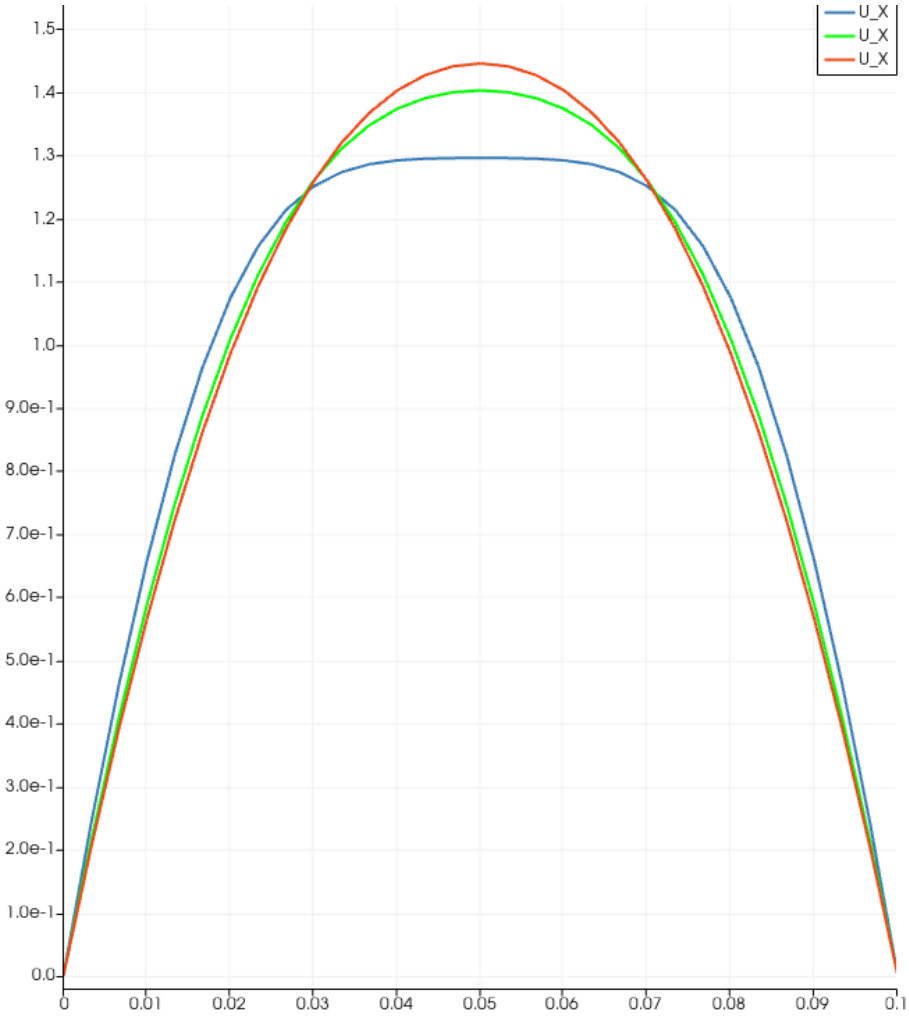
Com fim de obter todas as velocidades de determinado ponto em relação ao tempo de maneira simples será possível utilizar a série temporal. Para essa opção funcionar é feita uma

seleção do ponto que se deseja obter as velocidades em relação ao tempo, sendo este ponto o que será descrito pela série.

6.7 GRÁFICOS

Uma das maneiras de verificar os resultados de uma simulação de maneira gráfica é a partir da utilização de gráficos, estes que servirão para verificar a consistência dos dados, se houve o resultado esperado e possibilita uma melhor compreensão da simulação (GREENSHIELDS, 2023). Para obtenção dos gráficos é necessário informar os dados das velocidades e seus pontos, logo, foi feito o desenvolvimento de um método que permite a escrita de um arquivo com os dados necessários e o *software* será capaz de interpretar o mesmo e gerar gráficos a partir da biblioteca *Matplotlib* (Matplotlib, 2023).

Figura 11: Gráfico de três simulações de fluido



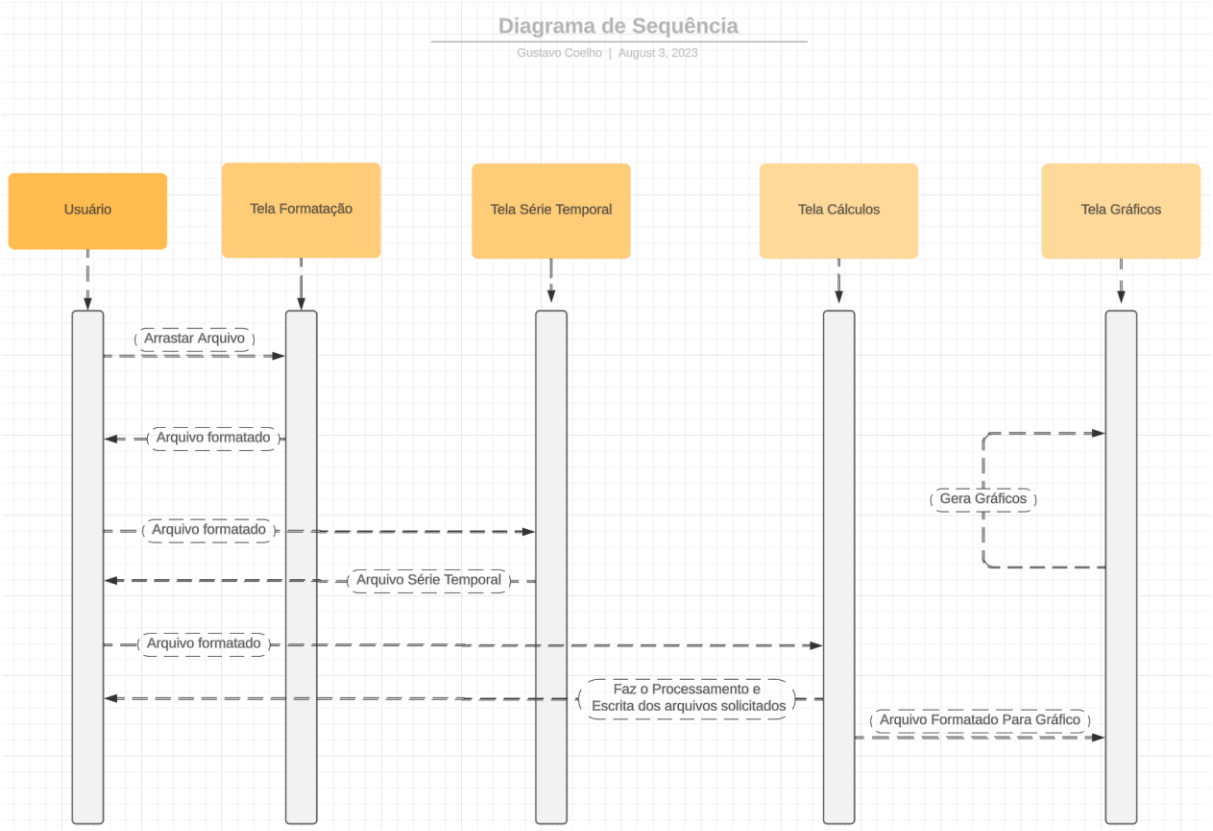
Fonte: Software Paraview

Na imagem 11 é possível ver como é o resultado esperado de uma simulação de fluido, que geral deve apresentar um perfil parabólico.

6.8 DIAGRAMA DE SEQUÊNCIA

Será apresentado agora o diagrama de sequência do *softwar*e.

Figura 12: Diagrama de Sequência



Fonte: Diagrama gerado no *Lucidchart*

No diagrama da imagem 12 é possível observar como é o fluxo do funcionamento do produto. Funcionando a partir da inserção de arquivos nas telas, iniciando a partir da formatação do arquivo de saída do processamento, gerado a partir do OpenFOAM, isto tornará possível utilizar as outras funcionalidades do *software*. Para realizar as operações

desejadas o usuário deve arrastar o arquivo formatado para seu respectivo local e com isso poder realizar o processo para obtenção dos dados necessários.

7 RESULTADOS E DISCUSSÕES

Após desenvolvida a metodologia, o desenvolvimento das telas e da parte responsável pelo processamento foi feito, resultando em uma ferramenta personalizada de pós-processamento em simulações de fluido com fim de colaborar em pesquisas referentes ao pós-processamento, oferecendo a disponibilidade de realizar operações matemáticas, estatísticas, analíticas e gerar gráficos de maneira simplificada.

Será descrito também o processo do estudo do pós-processamento sem a utilização do projeto desenvolvido, com fim de comparação.

Seu desenvolvimento foi feito utilizando as tecnologias já citadas anteriormente, que foram: C, Python e as bibliotecas *Pyside6, os* e *Matplotlib*. Sendo C utilizado para realizar as operações de leitura e escrita dos arquivos e operações matemáticas, tendo suas funcionalidades disponíveis nas telas de formatação, série temporal e cálculos e estatísticas. Já Python foi utilizado para a produção das telas juntamente com a biblioteca *Pyside6* (QT, 2023), telas essas que fazem a integração com a parte de processamento de dados feita em C, a partir da biblioteca *os* (PYTHON OS, 2023), que permite a realização de comandos de sistema. Já os gráficos são gerados a partir da linguagem Python em conjunto com a biblioteca *Matplotlib* (MATPLOTLIB, 2023).

Apresentando então as quatro telas e um botão de ajuda, sendo elas compostas por um componente que permite o arrastar de arquivos e um botão de confirmação, sendo possível formatar o arquivo para uso, gerar arquivos de série temporal, fazer cálculo da velocidade média, desvio padrão e geração dos arquivos para gráficos, de coordenadas, de velocidade média, e a exibição de gráficos correspondentes a simulação, em suas respectivas telas.

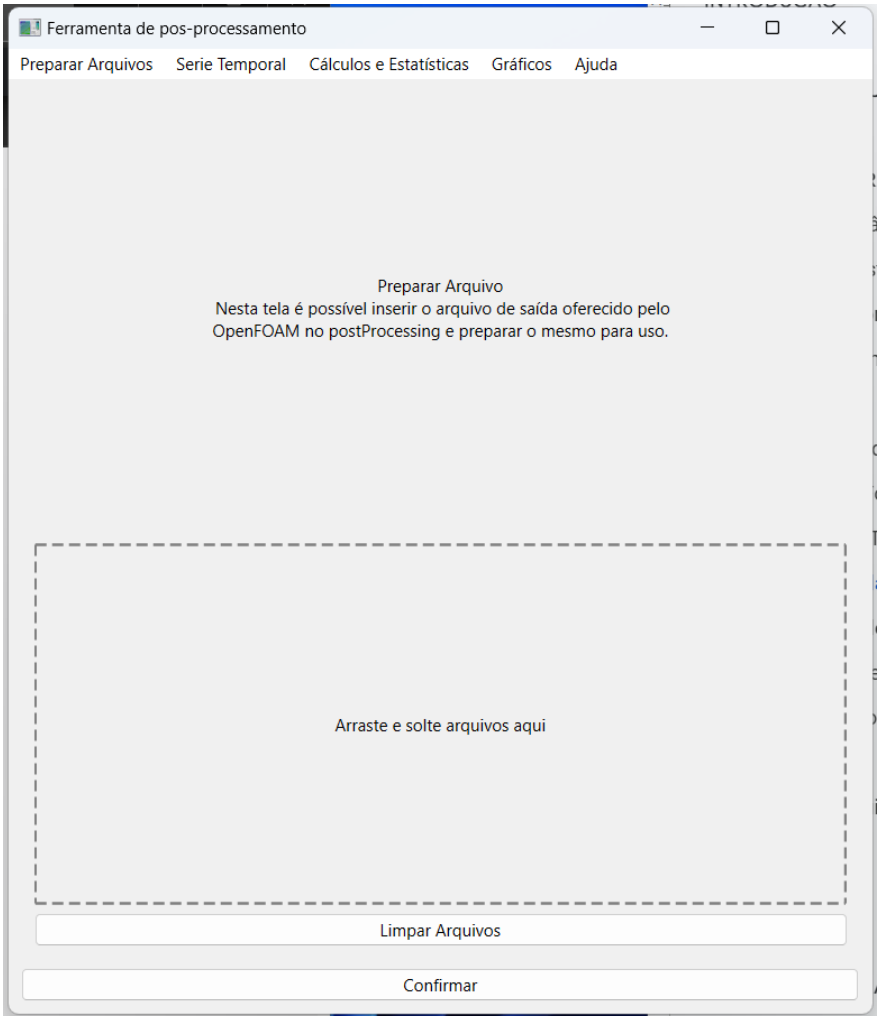
7.1 TELAS

Com fim de tornar possível o uso do *software* mais simples possível para o usuário final foi feito o desenvolvimento de quatro telas, estas que serão responsáveis por oferecer ao pesquisador as opções e disponibilidades do programa. Seu desenvolvimento foi feito utilizando Python juntamente com a biblioteca *PySide6*, esta que é uma tecnologia mais recente e oferece mais opções de uso e de fácil implementação (QT, 2023).

7.1.1 Tela de Formatação de Arquivos

Esta tela tem como objetivo preparar os arquivos que são entregues após o processamento, sendo então, a próxima etapa necessária para a realização do pós-processamento. Esta tela surgiu da necessidade de oferecer ao usuário uma maneira simples e eficaz de realizar a formatação do arquivo. Sendo ela responsável por realizar o processo de formatação dos arquivos, desde que é possível inserir diversos arquivos com os caracteres inválidos e selecionar um novo diretório de saída e lá encontrar os arquivos com nome semelhante ao de entrada.

Figura 13: Tela de Formatação de Arquivo

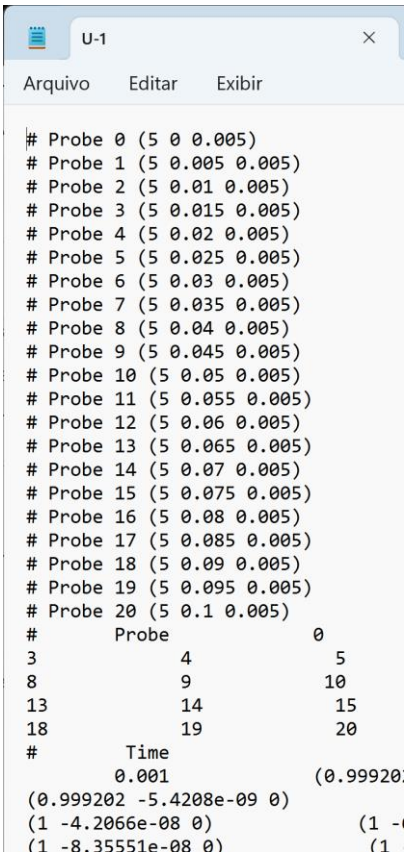


Fonte: *Software* desenvolvido

Nesta tela apresentada na imagem 13 é possível inserir arquivos, remover os arquivos inseridos e confirmar, o que irá fazer a operação de remoção de caracteres, será feita também a seleção de diretório de destino, onde será escrito o novo arquivo.

Logo, será feito um teste com os arquivos U-1.txt e U-2.txt, que estão com os caracteres indesejados e são idênticos, sendo possível ao arrastar os documentos até a área de destino.

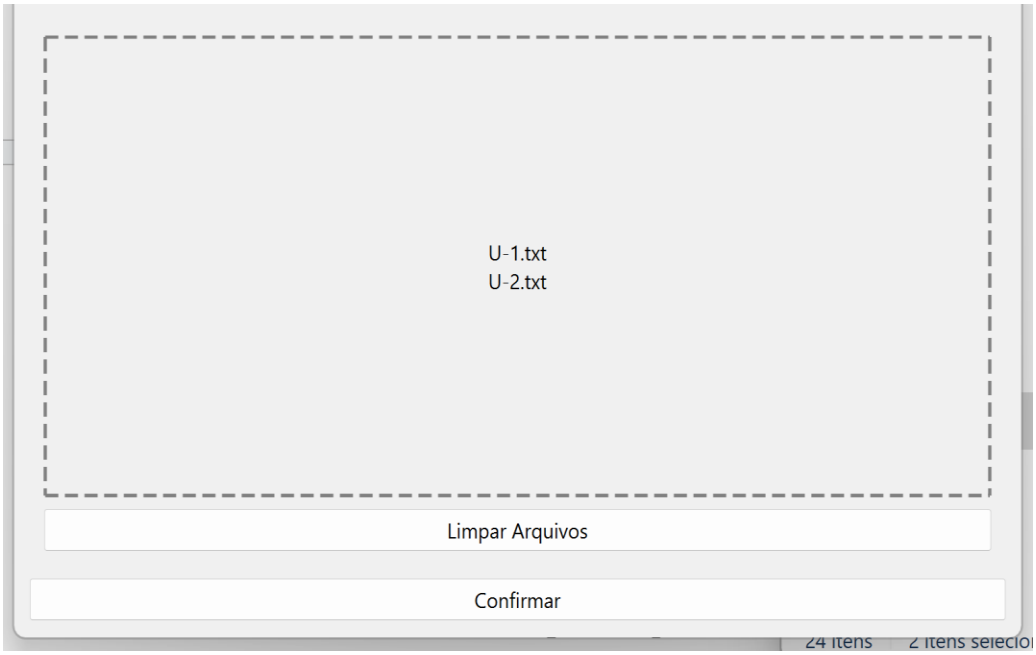
Figura 14: Arquivo inserido para formatação



Fonte: Arquivo de saída gerado pelo *OpenFOAM*

Na figura 14 é possível observar que o arquivo U-1.txt e o arquivo U-2.txt estão com caracteres inválidos, logo, precisando da formatação adequada para seguimento do uso do *software.*

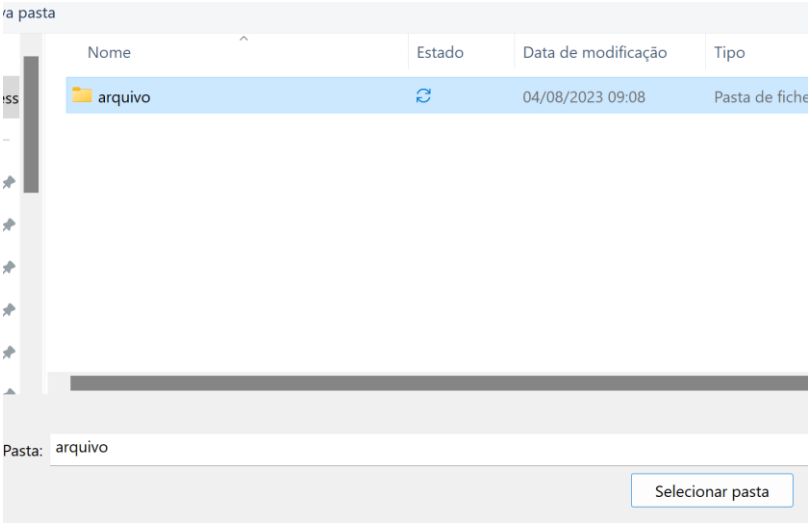
Figura 15: Tela de Preparação com dois arquivos



Fonte: *Software* desenvolvido

Os dois arquivos são então arrastados para a área designada e ao confirmar poderemos selecionar o diretório de destino que os arquivos serão escritos, como pode ser visto na imagem 15.

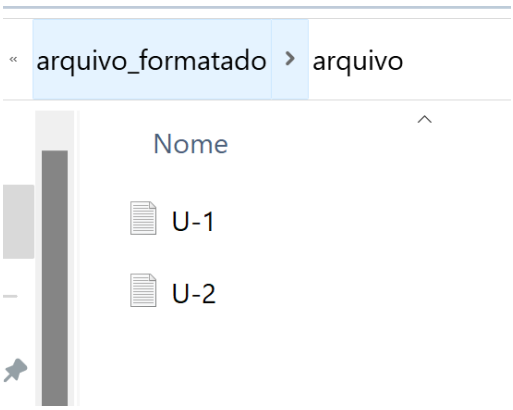
Figura 16: Seleção de Diretório



Fonte: *Software* desenvolvido

Na figura 16, ao selecionar a pasta os arquivos inseridos serão processados e entregues ao usuário no diretório que foi escolhido com os nomes originais, logo serão encontrados os arquivos com o mesmo nome, porém formatados.

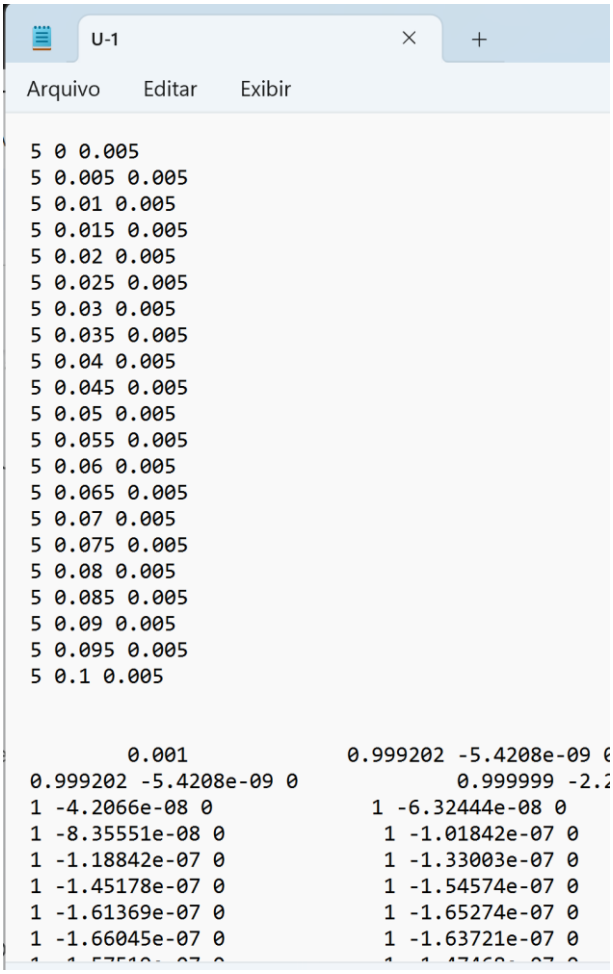
Figura 17: Saída dos Arquivos Formatados



Fonte: Arquivo de saída gerado pelo *Software* desenvolvido

Na figura 17 é possível verificar a saída dos dois arquivos formatados e prontos para uso no diretório de destino selecionado.

Figura 18: Arquivo U-1.txt após formatação



Fonte: Arquivo de saída gerado pelo Software desenvolvido

E com a obtenção desse novo arquivo o usuário pode realizar todas as operações disponíveis. Já no método manual de operação seria necessário fazer a retirada dos caracteres via trabalho manual, em que cada arquivo seria aberto e o usuário ou pesquisador teria que fazer a retirada de um a um para cada arquivo de saída que será estudado. Permitindo então que o estudo se torne mais eficiente ao realizar o mesmo.

7.1.2 Tela de Série Temporal

Figura 19: Tela da geração da Série Temporal

Fonte: *Software* desenvolvido

Para o funcionamento adequado desta tela presente na figura 19 é necessário realizar previamente a formatação do arquivo que se deseja estudar. Possuindo o arquivo formatado, agora pode-se arrastar o mesmo para a área designada e escolher quais pontos se deseja obter a série temporal. Podendo utilizar mais que um arquivo.

A série temporal de um ponto, é um dos dados necessários para o estudo e compreensão de uma simulação de fluido, a série temporal é em um arquivo que contém todas as velocidades relativas ao tempo de certo ponto, em que se pode selecionar até dez pontos. Tendo como resultado então todas as velocidades de tal ponto, em arquivos distintos que serão escritos com seu nome original e a junção da sua posição na definição de pontos, logo se o arquivo U-1.txt e o arquivo U-2.txt, forem arrastados para a área designada e definidos

dois pontos para geração da série temporal o resultado seria U-1.txt_st1, U-2.txt_st1, U-1.txt_st2 e U-2.txt_st2.

Figura 20: Inserção dos arquivos e definição de dois pontos

Fonte: *Software* desenvolvido

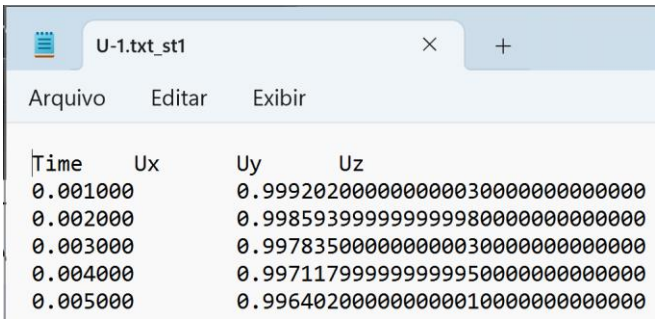
Após confirmação é selecionado o diretório, e para esse resultado foi gerado o diretório serie_temporal para escrita dos arquivos. Como visto na imagem 20.

Figura 21: Arquivos de saída da série temporal

Fonte: Arquivos de saída gerados pelo *Software* desenvolvido

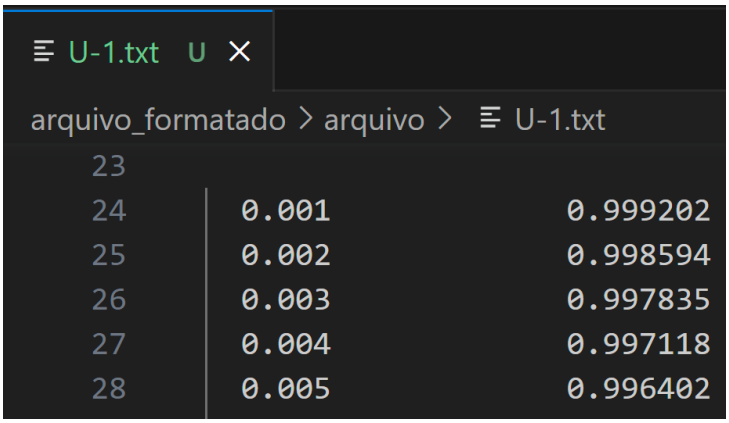
Agora com os arquivos gerados pode ser feita a validação, para verificar se o arquivo foi escrito da maneira adequada.

Figura 22: Arquivo de saída U-1.txt_st1



Fonte: Arquivo de saída gerado pelo *Software* desenvolvido

Figura 23: Arquivo U-1.txt Formatado



Fonte: Arquivo de saída gerado pelo *Software* desenvolvido

Com fim da verificação aqui são comparadas as velocidades das cinco primeiras iterações e com isso é possível concluir que o arquivo de saída referente a série temporal está adequado.

A série temporal é um dado em que sua obtenção deveria ser feita manualmente, visto que essa aplicabilidade foi desenvolvida a partir de uma necessidade encontrada pelos orientadores no projeto de pesquisa. Logo, a implementação do *software* com esta funcionalidade permite um estudo bem mais completo e eficaz.

7.1.3 Tela de Cálculos

Para a geração dos cálculos de maneira simples e organizada foi desenvolvida uma interface gráfica com fim de disponibilizar ao usuário as opções disponíveis pelo *software*.

Figura 24: Tela de realização de Cálculos e obtenção de arquivos de saída

Fonte: *Software* desenvolvido

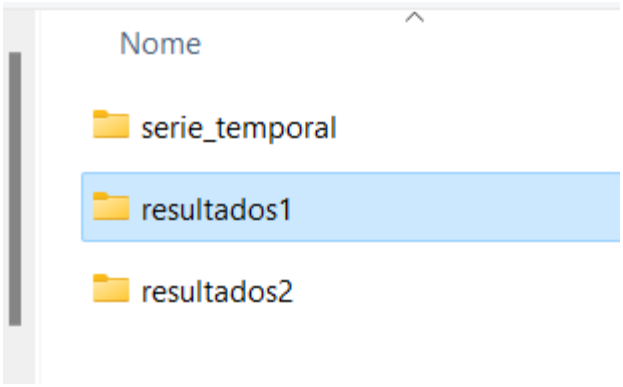
A utilização dos arquivos de saída formatados é essencial para a realização dos cálculos. Sendo os cálculos disponibilizados para a realização do cálculo de velocidade média e desvio padrão de X, de Y ou de Z, e a obtenção de dados como coordenadas, e a formatação de dados para geração de gráficos.

Figura 25: Inserção de arquivos na tela de cálculos e estatísticas

Fonte: *Software* desenvolvido

Sendo possível inserir o arquivo no espaço designado, escolher a quantidade de iterações a se ignorar e selecionar as operações desejadas disponibilizadas. Ao inserir o arquivo U-1.txt e U-2.txt é possível realizar as operações a partir da leitura dos mesmos, a escolha da quantidade de iterações mínimas é feita a partir de uma entrada de texto, em que se decide esse valor, pode se escolher as operações a partir da caixa de seleção e ao confirmar se espera a seleção de um diretório de destino, este que conterá os arquivos resultantes das seleções feitas, sendo feita a seleção para a saída de cada arquivo.

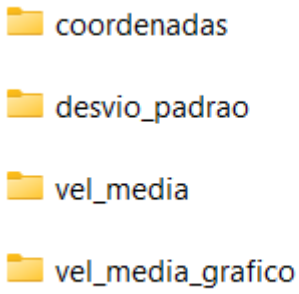
Figura 26: Seleção do diretório de saída após confirmação



Fonte: Pastas no explorador de arquivos

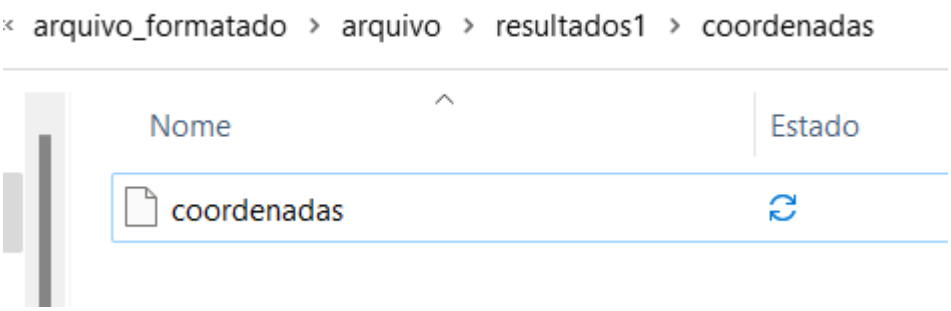
O diretório resultados1 ao ser selecionado conterá os resultados do processamento dos dados referentes aos dados selecionados nas caixas de seleção, utilizando o arquivo U-1.txt para leitura. Para o resultado do arquivo U-2.txt será selecionado o diretório resultados2, este que conterá uma estrutura semelhante ao resultados1.

Figura 27: Diretórios de saída do processamento



Fonte: Diretórios de saída gerados pelo *Software* desenvolvido

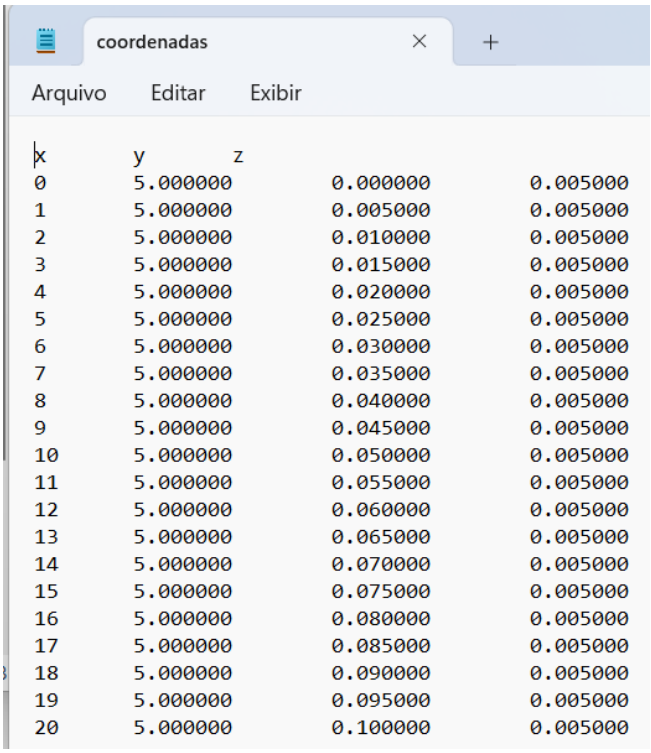
Figura 28: Arquivo resultado na pasta coordenadas



Fonte: Arquivo de saída gerado pelo *Software* desenvolvido

Aqui se encontra o arquivo referente a caixa de seleção “Gerar arquivos com coordenadas”.

Figura 29: Arquivo de saída das coordenadas

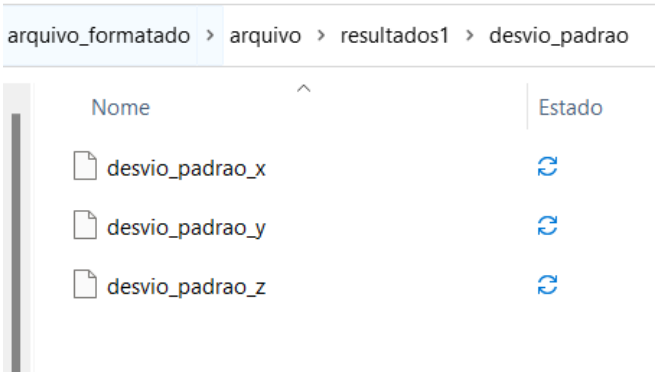


x	y	z
0	5.000000	0.000000	0.005000
1	5.000000	0.005000	0.005000
2	5.000000	0.010000	0.005000
3	5.000000	0.015000	0.005000
4	5.000000	0.020000	0.005000
5	5.000000	0.025000	0.005000
6	5.000000	0.030000	0.005000
7	5.000000	0.035000	0.005000
8	5.000000	0.040000	0.005000
9	5.000000	0.045000	0.005000
10	5.000000	0.050000	0.005000
11	5.000000	0.055000	0.005000
12	5.000000	0.060000	0.005000
13	5.000000	0.065000	0.005000
14	5.000000	0.070000	0.005000
15	5.000000	0.075000	0.005000
16	5.000000	0.080000	0.005000
17	5.000000	0.085000	0.005000
18	5.000000	0.090000	0.005000
19	5.000000	0.095000	0.005000
20	5.000000	0.100000	0.005000

Fonte: Arquivo de saída gerado pelo *Software* desenvolvido

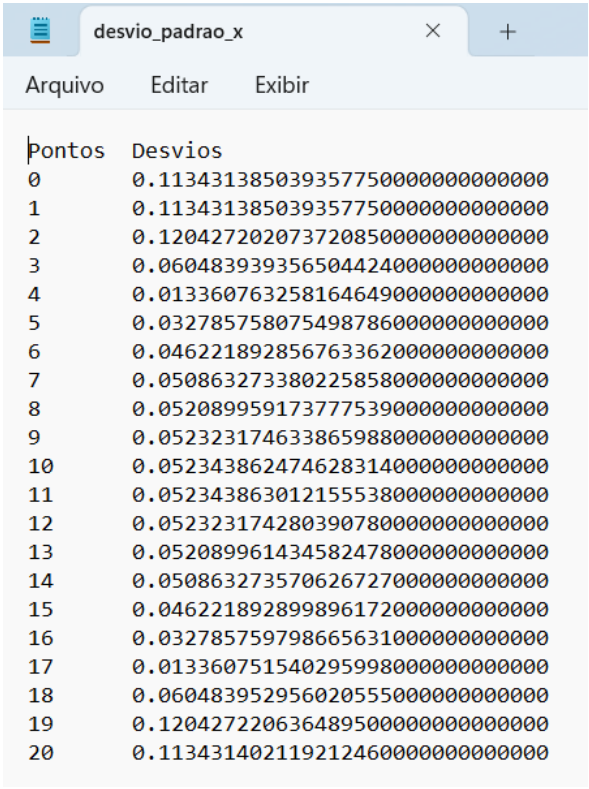
No arquivo presente na imagem 29 se encontram os dados referentes a coordenadas do arquivo U-1.txt, havendo um semelhante na pasta resultados2 referentes ao arquivo U-2.txt.

Figura 30: Arquivos referentes ao desvio padrão



Fonte: Arquivos de saída gerados pelo *Software* desenvolvido

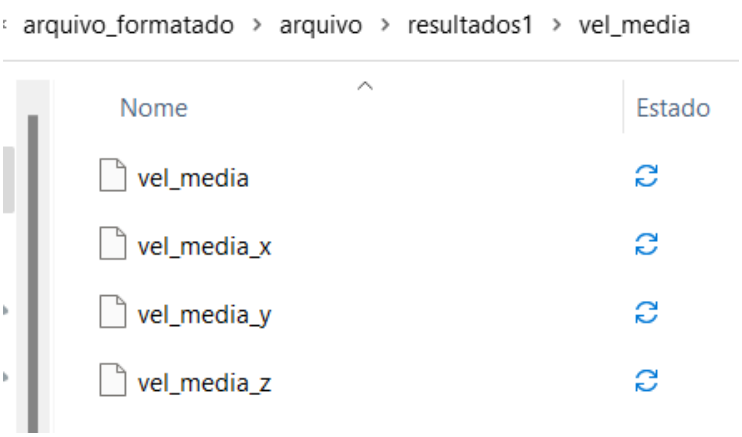
Figura 31: Arquivo de saída referente aos desvios padrões de cada ponto de x



Fonte: Arquivo de saída gerado pelo *Software* desenvolvido

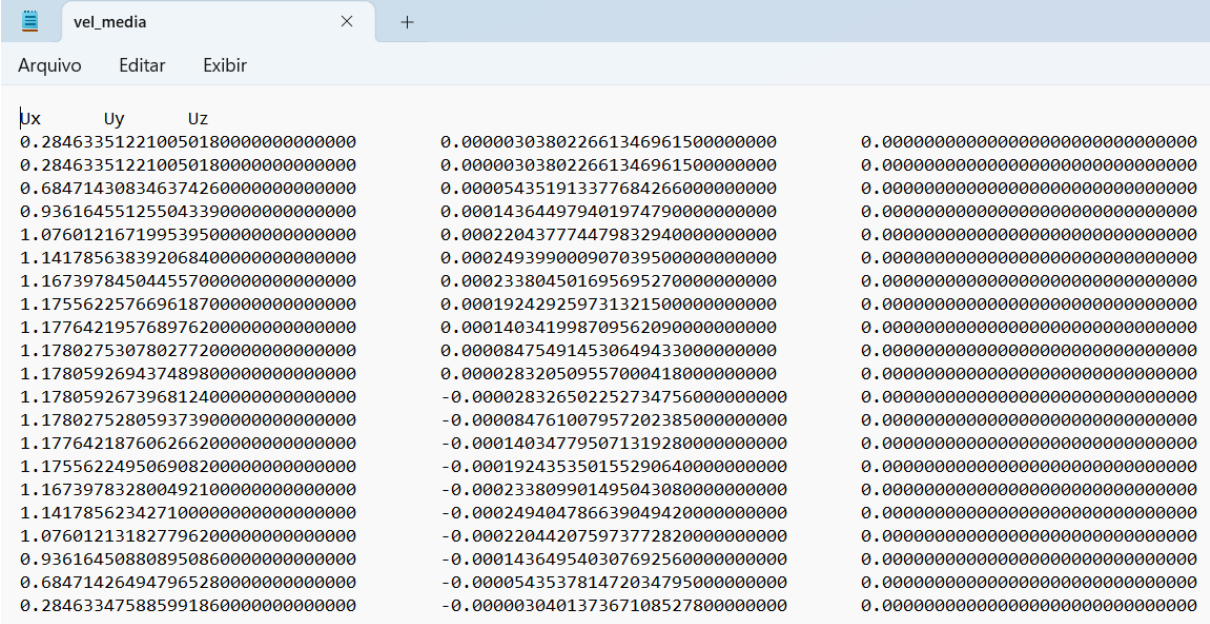
Sendo possível observar os arquivos de saída resultantes do processamento dos dados, na figura 31, em que a informação disposta no arquivo desvio_padrao_x segue a regra de informar o ponto e seu respectivo valor resultante do cálculo do desvio padrão. O diretório resultados2 conterá o mesmo arquivo com os dados referentes ao arquivo U-2.txt.

Figura 32: Arquivos da velocidade média



Fonte: Arquivo de saída gerado pelo *Software* desenvolvido

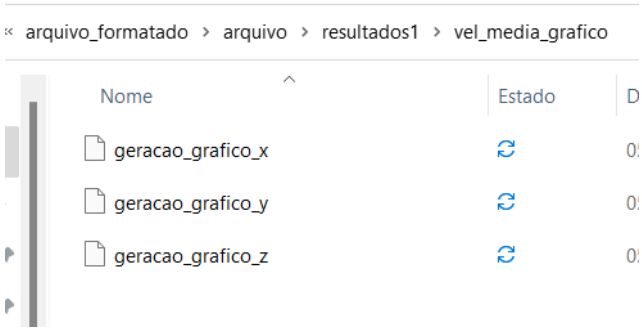
Figura 33: Arquivo de saída com todas as velocidades



Fonte: Arquivo de saída gerado pelo *Software* desenvolvido

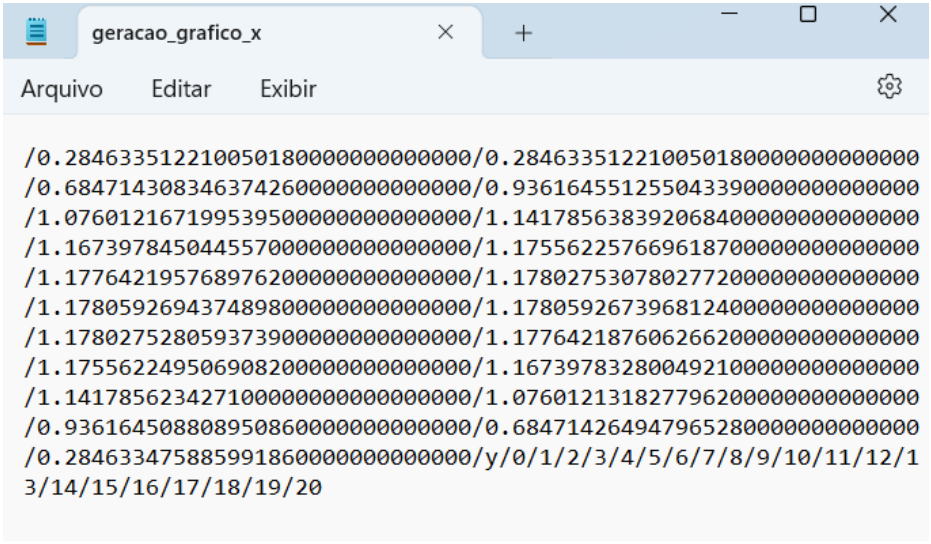
Nestas figuras é possível perceber que a saída dos arquivos da velocidade média é feita no diretório vel_media e com o nome de acordo com sua velocidade, variando entre x, y, z e todas, sendo o arquivo vel_media o que contém todas as velocidades. Disposto a partir de três colunas, que definem as resultantes do cálculo da velocidade média de todas as velocidades, o diretório resultados2 é composto pelos mesmos elementos e estrutura.

Figura 34: Diretório de saída das velocidades para geração de gráfico



Fonte: Arquivo de saída gerado pelo *Software* desenvolvido

Figura 35: Arquivo de geração gráfico de x, geracao_grafico_x



Fonte: Arquivo de saída gerado pelo *Software* desenvolvido

Neste diretório é possível observar a estrutura do diretório que contém os arquivos de saída, estes que são nomeados a partir da sua velocidade. O arquivo gerado possui as velocidades médias formatadas a partir de '/', do carácter 'y' e os seus respectivos pontos separados por '/'. O carácter especial '/' será utilizado para realizar a leitura das velocidades no momento de gerar os gráficos, assim como o 'y' especifica que agora será feita a leitura dos dados referentes aos pontos. O diretório resultados2 contém arquivos semelhantes ao resultados1.

Após obtenção e verificação de todos os dados é possível realizar estudos que irão contribuir com o processamento de entendimento da simulação de fluidos, logo sendo uma das principais telas e colaborando com o pós-processamento de simulações de fluido.

Esta etapa de geração de arquivos foi desenvolvida inicialmente em FORTRAN, linguagem esta que apresentava algumas limitações, como a inserção de dados a partir do usuário, o que fazia com que o uso do *software* utilizado na época fosse bom, mas não o melhor, sendo neste *software* a etapa de definição de valores automática, fazendo com que o próprio programa seja capaz de identificar as propriedades do arquivo.

7.1.4 Tela de Gráficos

Após a etapa de gerar os arquivos de gráficos se torna possível a geração dos mesmos, utilizando de referência a velocidade e seus pontos, como visto anteriormente.

Figura 36: Tela de geração de gráficos

Fonte: Arquivo de saída gerado pelo *Software* desenvolvido

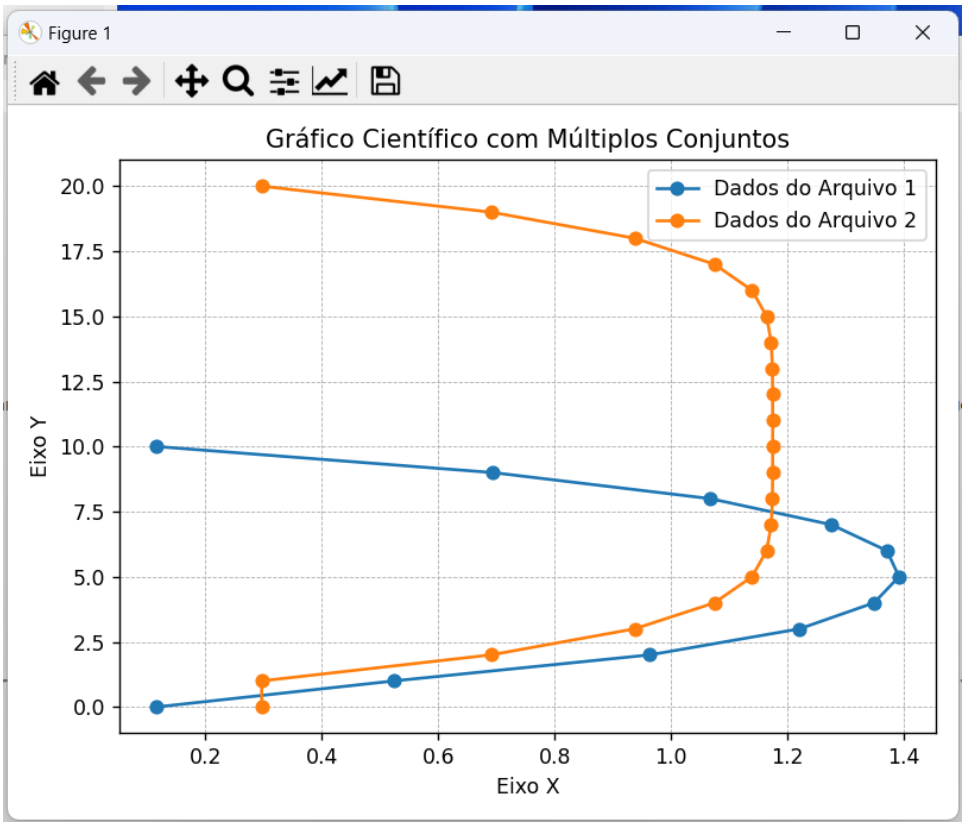
Nesta tela é possível utilizar a área designada para inserção de arquivos, estes que estão no diretório vel_media_grafico e podem ser utilizados para para realizar a formação de gráficos comparativos de velocidade.

Figura 37: Geração gráfico com arquivos

Fonte: Arquivo de saída gerado pelo *Software* desenvolvido

Os arquivos referentes a geração do gráfico de x presentes no diretório resultados1 e resultados2 foram inseridos na tela de gráficos e será possível observar o mesmo após confirmação.

Figura 38: Gráfico gerado a partir de dois arquivos



Fonte: Gráfico gerado pelo *Software* desenvolvido

Após leitura dos dados presentes no arquivo se compreende as velocidades e seus respectivos pontos, tornando então possível observar a maneira como a velocidade se comporta de acordo com o seu respectivo ponto e comparar um ou mais arquivos no mesmo gráfico. O gráfico é feito a partir da biblioteca *Matplotlib*, que recebe os dados a partir do arquivo de entrada que possui as velocidades e os pontos, com isso, tornando possível realizar a organização e manipulação dos dados. (MATPLOTLIB, 2023). Colaborando assim, com o estudo na área.

A geração de gráficos foi uma implementação que surgiu com a necessidade, já que uma das grandes ferramentas de estudo é a mesma e deve ser feita com grande frequência. A realização destes gráficos era a partir de um *software* externo, em que se levava os arquivos formatados para o mesmo e lá se obtinha os gráficos, já neste produto desenvolvido esta parte é feita no próprio *software*.

7.1.5 Botão de ajuda

O botão de ajuda foi desenvolvido para que quando houver um clique ir ao repositório do projeto disponibilizado no *GitHub*. Estando neste local um manual com algumas instruções de uso do mesmo, tornando o uso do *software* mais facilitado.

8 CONSIDERAÇÕES FINAIS

Conclui-se que o desenvolvimento desta ferramenta é capaz de oferecer ao pesquisador a possibilidade de obter dados (velocidade média e desvio padrão, dados esses que já foram validados perante o projeto de pesquisa já realizado) e arquivos específicos a partir do uso da ferramenta desenvolvida.

É observável que este projeto entrega um *software* que será capaz de colaborar com a obtenção da informação desejada de maneira simples e eficaz, trazendo então mais uma ferramenta à sociedade científica. Podendo ser utilizada para realizar o estudo do pós-processamento de qualquer simulação de fluido realizada com o OpenFOAM. Como visto anteriormente, diversas etapas realizadas manualmente ou que exigiam a utilização de uma ferramenta externa foram suprimidas.

Este projeto tem a habilidade de algumas atividades que eram feitas de maneira manuais, assim, automatizando etapas do estudos, sendo estas atividades: limpeza dos arquivos, geração das séries temporais ou qualquer outro arquivo, visto que essa ferramenta foi desenvolvida a partir da necessidade, visto que não havia algum *software* dedicado a essas operações, tendo a capacidade de unir e integrar diversas funcionalidades referentes ao estudo do arquivo de saída do pós-processamento em simulações de fluido.

Verifica-se que as tecnologias implantadas tiveram boa eficiência e justificam seu uso, permitindo realizar operações que tornam o uso do *software* viável e eficaz. O fato de o desenvolvimento da parte responsável pela compilação dos dados ter sido realizado em C permitiu uma boa gestão da memória, e a utilização da tecnologia Python permitiu que fosse entregue ao cliente ou usuário uma maneira simples de utilizar o produto, deixando de lado a utilização de terminal para realização das operações, como é feita em geral ao utilizar a Linguagem C com entrada de dados.

Uma das dificuldades durante o desenvolvimento deste *software* foi encontrada na leitura do arquivo de saída gerado pelo OpenFOAM, na leitura dos caracteres, que inicialmente utilizava o método de gerenciamento de arquivos, tornando a execução lenta e ineficaz. Para superar esse obstáculo, foi necessário desenvolver uma nova habilidade: o gerenciamento da memória utilizando a linguagem de programação C. A gestão da memória exige atenção para garantir uma liberação adequada da mesma após o uso, evitando, deste modo, problemas no funcionamento do computador.

Este projeto apresenta uma ampla gama de possibilidades tanto para uso imediato quanto para mudanças futuras. Há diversas opções disponíveis para serem exploradas por pesquisadores e expandidas com o tempo, podendo se tornar a ferramenta oficial para o pós-processamento de simulações de fluidos geradas pelo processador OpenFOAM, uma que a manipulação dos dados ocorre de maneira sucinta, permitindo diversas outras operações.

REFERÊNCIAS

NETTO, Eduardo Bráulio Wanderley. **Arquitetura de Computadores**: A visão do software. Natal: CEFET-RN, 2005.

KNIGHT, Randall D.. **Uma Abordagem Estratégica**: Mecânica Newtoniana, Gravitação, Oscilações e Ondas. Porto Alegre: Bookman, 2009. v. 1.

SOFFNER, Renato. **Algoritmos e Programação em linguagem C**. 1. ed. São Paulo: Saraiva, 2013.

LAPLACE, Colin . Dev-C++. **Dev-C++ Official Website,** 2023. Disponível em: https://www.bloodshed.net/. Acesso em: 30 jul. 2023.

QT, Qt For Python Team. PySide6 6.5.2. **PySide6,** 2023. Disponível em: https://pypi.org/project/PySide6/. Acesso em: 30 jul. 2023.

PYTHON SYS, Software Foundation. Sys. **sys — Parâmetros e funções específicas do sistema,** 2023. Disponível em: https://docs.python.org/pt-br/3/library/sys.html. Acesso em: 30 jul. 2023.

PYTHON OS, Software Foundation. Os. **os — Miscellaneous operating system interfaces,** 2023. Disponível em: https://docs.python.org/3/library/os.html. Acesso em: 30 jul. 2023.

MATPLOTLIB, The Matplotlib Development Team. Matplotlib. **Matplotlib 3.7.2 documentation,** 2023. Disponível em: https://matplotlib.org/stable/index.html. Acesso em: 30 jul. 2023.

RICARTE, Ivan Luiz Marques. **Programação Orientada a Objetos: Uma Abordagem com Java**. Campinas: Unicamp, 2001.

ROCHA, Heloísa; BARANAUSKAS, Maria Cecília. **Design e Avaliação de interfaces humano-computador**. Campinas: Unicamp, 2003. ISBN 85-88833-04-2.

CASTRO E SILVA, Jorge Luiz De; FERNANDES, Maria Wilda; ROSA LÍVIA, Freitas De Almeida. **Estatística e Probabilidade**. 3. ed. Fortaleza: EdUECE, 2015. ISBN 979-85-89427-32-5.

COCIAN, Luis. **Manual da Linguagem C**. 1. ed. Canoas: Ulbra, 2004.

PYTHON, Software Foundation. Python 3.11.4 documentation. **Python,** 2023. Disponível em: https://docs.python.org/3.11/. Acesso em: 30 jul. 2023.

ÇENGEL, Yunus; CIMBALA, John. **Mecânica dos Fluidos Fundamentos e Aplicações.**. Porto Alegre: AMGH, 2000. v. 3.

FORTUNA, Armando. **Técnicas Computacionais para Dinâmica dos Fluidos Conceitos Básicos e Aplicações**. São Paulo: Editora da Universidade de São Paulo, 2000.

LAGO, Jonatas. **AULAS DE HIDROSTÁTICA E HIDRODINÂMICA**. Rio de Janeiro: UFRJ, 2010. Disponível em: https://pantheon.ufrj.br/bitstream/11422/4076/3/JHPLago.pdf. Acesso em: 30 jul. 2023.

GREENSHIELDS, Chris Greenshields. CFD Direct: The Architects of OpenFOAM. **OpenFOAM v11 User Guide,** 2023. Disponível em: https://doc.cfd.direct/openfoam/user-guide-v11/index. Acesso em: 30 jul. 2023.