МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»**

Кафедра Математического и компьютерного моделирования

Моделирование вращательного движения средствами OpenFOAM

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 411 группы

направление 01.03.02 - Прикладная математика и информатика

механико-математического факультета

Нуриева Тимура Тагировича

Научный руководитель
доцент, к.т.н., доцент И.А. Панкратов

Зав. кафедрой
зав. каф., д.ф.-м.н., доцент Ю.А. Блинков

Саратов 2022

Введение. В бакалаврской работе рассмотрена математическая модель движения жидкости в цилиндре с вращающейся крестовиной.

Для моделирования движения жидкости, в случае решения уравнений гидродинамики в результате получается лишь уравнение движения, которое остается только визуализировать в виде графика или же 3D-модели протекания явления. Существует множество способов сделать это, но такая утилита, как OpenFoam, обладает возможностью решить уравнения и создать файлы для дальнейшей визуализации движения. Так как уравнения гидродинамики при моделировании движения жидкости сводятся к уравнению движения, то остается только построить график или 3D модель. Для построения 3D модели в OpenFOAM существует множество алгоритмов:

- icoFoam — это нестационарный решатель для несжимаемого ламинарного течения ньютоновских жидкостей.
- pisoFoam — это переходный решатель для несжимаемого потока. Моделирование турбулентности является общим.
- simpleFoam — стационарный решатель несжимаемого турбулентного потока.
- pimpleFoam - это решатель переходных процессов с большим временным шагом для несжимаемого потока с использованием алгоритма PIMPLE (объединенный PISO-SIMPLE).

В бакалаврской работе на примере задачи mixerVesselAMI2D с помощью алгоритма pimple будут посчитаны промежуточные данные и построена 3D модель.

Платформа OpenFoam является не единственной программой для моделирования физических явлений, но самой популярной при решении задач гидродинамики (CFD - computational fluid dynamics). Поэтому для реализации моделирования была выбрана именно эта программа.

Целью бакалаврской работы является исследование изменения скорости и давления частиц в потоке вязкой несжимаемой жидкости внутри ротора. Основная часть работы посвящена реализации модели вращения ротора в утилите OpenFOAM.

Структура бакалаврской работы. В данной бакалаврской работе содержится введение, 4 раздела и заключение:

- **Во введении** рассматривается актуальность утилиты OpenFOAM при решении объемных и плоских задач гидродинамики.
- **В первом разделе** рассматриваются уравнения с помощью которых можно задать движение индивидуальных частиц вязкой несжимаемой жидкости в потоке этой жидкости.
- **Второй раздел** посвящен разбору примера и определению начальных условий по которым строится модель в примере mixerVesselAMI2D.
- **Третий раздел** посвящен способам построения моделей жидкостей с использованием словаря DynamicMeshDict.
- **В четвертом разделе** проводится изучение алгоритма PIMPLE с последующим исследованием зависимости модели от начальных значений переменных.

Основное содержание работы. Движение сетки в примере mixerVesselAMI2D достигается путем применения словаря DynamicMeshDict, который контролирует деформацию и морфинг сетки во время моделирования. Этот словарь необходим только для решателей, которые вызывают движение сетки. Как правило, все эти решатели будут иметь термин DyM, включенный в название базового решателя. Например, стандартным базовым решателем может быть pimpleFoam. С движением сетки, решатель будет pimpleDyMFoam и потребует dynamicMeshDict.

Словарь позволяет указать четыре типа движения сетки, которые задаются следующими шаблонами:

1. staticFvMesh: не обеспечивает движения сетки. Сетка статична. Полезно для отладки симуляции, включающей движение сетки.

```
dynamicFvMesh staticFvMesh;
staticFvMeshCoeffs
{
}
```

2. dynamicRefineFvMesh: уточняет сетку моделирования и обеспечивает изменения топологии на основе полей моделирования.

```
dynamicFvMesh dynamicRefineFvMesh;
dynamicRefineFvMeshCoeffs
{
                //Parameter definitions
```

```
}
```

3. dynamicMotionSolverFvMesh: движение сетки, основанное на решенном движении сетки для движения твердого тела.

```
dynamicFvMesh dynamicMotionSolverFvMesh;
motionSolverLibs ("libsixDoFRigidBodyMotion.so");
solver sixDoFRigidBodyMotion;
sixDoFRigidBodyMotionCoeffs
{
                //Parameter definitions
}
```

4. solidBodyMotionFvMesh: заданное движение сетки. Нет изменения топологии в сетке.

```
dynamicFvMesh solidBodyMotionFvMesh;
solidBodyMotionFvMeshCoeffs
{
                //Parameter definitions
}
```

В задаче mixerVesselAMI2D используется тип движения сетки solidBodyMotionFvMesh. В этом случае движение сетки определено и задано перемещением, известным до начала моделирования. Пользователь указывает тип движения, и сетка перемещается в соответствии с этим заданным движением. Этот тип движения полезен в случаях, когда топология сетки не меняется, но сетка по-прежнему включает в себя некоторый тип движения. Примеры этого могут включать вращающиеся подобласти для моделирования вентиляторов, пропеллеров и т. д. Также может включать некоторые типы колебательного движения для моделирования линейных движений поршня. Рассмотрим подробней этот тип движения

```
dynamicMeshDict
{
  dynamicFvMesh solidBodyMotionFvMesh;
  solidBodyMotionFvMeshCoeffs
  {
      //Parameter Definitions
  }
  motionSolverLibs ( "libfvMotionSolvers.so" );
```

}

Можно указать разные движения для нескольких тел. В таком случае будет определено несколько подобластей, и каждая подобласть является телом. Затем можно указать различные заранее полученные перемещения для каждого тела. Нельзя указывать разные решатели движения для разных тел. Движение внутри каждого тела определяются так же, как определяется движение для единственного тела. В дальнейшем будут рассматриваться определения движения одного тела, но всё написанное далее можно адаптировать и для использования в движении нескольких тел. При использовании solidBodyMotionFvMesh поддерживается несколько функций движения. Они определяют характеристику заданного движения. Каждая функция движения имеет ряд параметров, связанных с характеристикой движения. Некоторые из этих движений можно использовать в случаях полустационарного потока. Это может относиться к таким случаям, как замороженный ротор, когда ротор решается для нескольких репрезентативных повернутых положений. Остальные движения предназначены для реального движения, зависящего от времени.

1. axisRotationMotion: вращательное движение с постоянной скоростью.

```
dynamicFvMesh solidBodyMotionFvMesh;
solidBodyMotionFvMeshCoeffs
{
    cellZone             rotating;
    solidBodyMotionFunction axisRotationMotion;
    axisRotationMotionCoeffs
    {
        origin          ( 0 0.5 5.0 );
        radialVelocity  ( 0 0 360);          //Vector rotation
            direction, in deg/s
    }
}
```

Функция axisRotationMotion аналогична rotateMotion которая будет рассмотрено ниже в работе. Только вместо указания оси и скорости вращения они объединяются в одно определение трехмерного вектора. Одно очень важное отличие: функция axisRotationMotion ожидает ско-

рость вращения в $*deg/s*$, тогда как функция rotateMotion ожидает скорость вращения в $*rad/s*$. При работе с использованием этих двух функций необходимо всегда проверять, соответствует ли ваша скорость вращения ожидаемой.

2. linearMotion: линейное перемещение с постоянной скоростью.

```
dynamicFvMesh solidBodyMotionFvMesh;
 solidBodyMotionFvMeshCoeffs
 {
        cellZone              Motion;
        solidBodyMotionFunction linearMotion;
        linearMotionCoeffs
        {
            velocity          (1.0 0.0 0.25);
        }
 }
```

   Линейное движение определяет линейное перемещение с постоянной скоростью. Его можно применить ко всей области или только к подобласти.

3. oscillatingLinearMotion: колебательное линейное перемещение по синусоидальному закону.

```
dynamicFvMesh solidBodyMotionFvMesh;
solidBodyMotionFvMeshCoeffs
{
        cellZone              moving;
        solidBodyMotionFunction oscillatingLinearMotion;
        oscillatingLinearMotionCoeffs
        {
            amplitude         (3.0 0.5 2.0);
            omega             5.25;
        }
}
```

   Функция oscillatingLinearMotion обеспечивает колебательное движение с определенной частотой. Это полезно для моделирования таких сценариев, как колеблющиеся поршни. Это создает неравномерную скорость смещения сетки, которая изменяется синусоидально.

4. oscillatingRotatingMotion: колебательное вращательное движение по синусоидальной схеме.

```
dynamicFvMesh solidBodyMotionFvMesh;
 solidBodyMotionFvMeshCoeffs
 {
        cellZone             rotating;
        solidBodyMotionFunction oscillatingRotatingMotion;
        oscillatingRotatingMotionCoeffs
        {
            origin           (0 0.5 1.5);
            amplitude        (3.0 0.5 2.0);  //units of rad
            omega            5.25;       //units of rad/s
        }
 }
```

   Функция oscillatingRotatingMotion создает неравномерное вращательное движение. Движение изменяется синусоидально, создавая колебательный узор. Это полезно для моделирования колеблющихся колес, маятников или других вращающихся объектов, совершающих периодические движения.

5. SDA: Движение, характерное для движения морского корабля. Три степени свободы движения корабля. SDA - это движение, характерное для моделирования морского перемещения. SDA создает заданное движение в трех направлениях: крен, подъем и раскачивание. Это было предназначено для имитации движения судна по морскому пути. Предполагаемым применением было выплескивание большого резервуара для хранения жидкости на судне. В этом сценарии резервуар будет моделироваться как область CFD(contact for differences), а движение SDA будет применятся для моделирования движений корабля и управления выплескиванием резервуара.

6. tabulated6DoFMotion: набор табличных значений, определяющих шесть степеней свободы движения тела.

```
dynamicFvMesh solidBodyMotionFvMesh;
solidBodyMotionFvMeshCoeffs
{
        solidBodyMotionFunction tabulated6DoFMotion;
```

```
        tabulated6DoFMotionCoeffs
        {
            CofG                ( 1.0  3.0 2.5 );
            timeDataFileName    "Absolute Path to data file/
                datafile.dat";
        }
    }
```

Функция Tabulated6DoFMotion позволяет пользователю указать набор сложных взаимосвязанных движений с помощью внешней таблицы, которая дает движение в несколько моментов времени.

7. multiMotion: комбинация других определенных движений.

```
dynamicFvMesh solidBodyMotionFvMesh;
 solidBodyMotionFvMeshCoeffs
 {
        solidBodyMotionFunction multiMotion;
        multiMotionCoeffs
        {
            NameofMotion1
            {
                solidBodyMotionFunction typeOfMotionFunction
                    ;
                typeOfMotionFunctionCoeffs
                {
                    //Parameters
                }
            }

            NameofMotion2
            {
                solidBodyMotionFunction typeOfMotionFunction
                    ;
                typeOfMotionFunctionCoeffs
                {
                    //Parameters
                }
            }
        }
    }
```

Multimotion сочетает в себе несколько функций движения. Примером может служить область, движущаяся с колебательным линейным движением, но с разной частотой колебаний в каждом направлении. Это было бы невозможно с одним определением «колебательного линейного движения». Однако, можно использовать multiMotion для предоставления трех разных определений «колебательного линейного движения», где каждое движение имеет разную частоту колебаний. Это позволяет создавать сложные движения, комбинируя несколько простых функций движения.

8. rotatingMotion: постоянное вращательное смещение.

```
dynamicFvMesh solidBodyMotionFvMesh;
 solidBodyMotionFvMeshCoeffs
 {
        cellZone             rotating;
        solidBodyMotionFunction rotatingMotion;
        rotatingMotionCoeffs
        {
            origin           ( 0 0.5 5.0 );
            axis             ( 0 0 1);
            omega            5.52
        }
 }
```

Эта Функция обеспечивает простое определение постоянного вращения сетки. Вы можете вращать всю область или только одну зону ячеек внутри области. Это очень полезно для временного моделирования турбомашин. Рассмотрим данную функцию более детально.

Первое, на что стоит обратить внимание это ключевое слово cellZone:

```
cellZone             rotating;
```

Праметр cellZone это имя ячейки, назначенной во время создания сетки. Он используется для того, чтобы указать подобласть для вращающихся турбомашин или пропеллеров. Если cellZone отсутствует, то вращать можно всю область. Чтобы выполнить вращение все области, неоходимо указать значение «none» для ключевого слова cellZone, либо не указывать эту запись вовсе. После того как вращающаяся область была определена, перейдем к ко-

эффициентам вращения rotatingMotionCoeffs. Они представлены тремя ключевыми словами: origin, axis, omega. Начало вращения задается с помощью ключевого слова origin:

```
origin          ( 0 0.5 5.0 );
```

Задается начало вращения вектор из точки начала симуляции. Векторные единицы обычно задаются в метрах. Направление вращения задается ключевым словом axis:

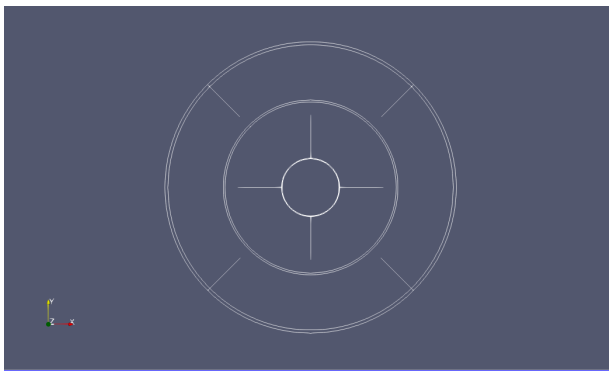
```
axis            ( 0 0 1);
```

Область или подобласть будут вращаться вокруг этой оси. Ось задается как трехмерный вектор. Как и в случае origin векторные единится обычно задаются в метрах. OpenFOAM автоматически преобразует заданный вектор в единичный. Последний параметр который осталось разобрать это omega:

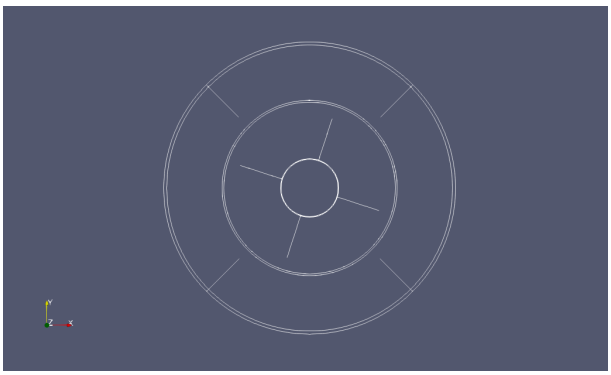
```
omega           5.52
```

Это скорость вращения для заданной области или подобласти. Указывается в rad/s. Можно указать как положительную, так и отрицательную скорость вращения. Направление вращения следует правилу правой руки вокруг заданной ранее оси вращения.

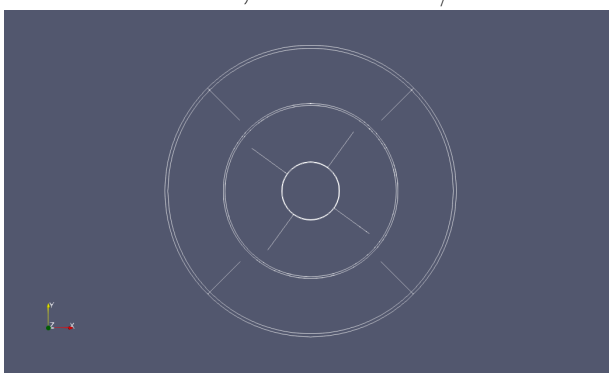
В задаче mixerVesselAMI2D вращается подобласть внутри области. Область и подобласть заданы в файле, «mixerVesselAMI2D/system/blockMeshDict». Вращающаяся подобласть имеет название rotor. Ниже, в соответсвии с рисунком 1, представлены графики положения сетки областей в разные моменты времени, при фиксированной угловой скорости вращения $\omega = 2\pi$ rad/s.



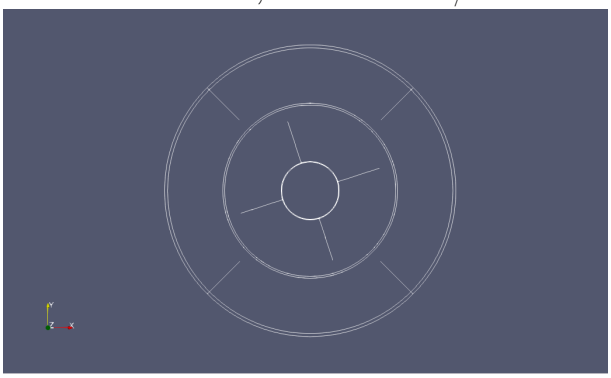
$t = 0s, \omega = 2\pi$ rad/s



$t = 0.2s, \omega = 2\pi$ rad/s



$t = 0.4s, \omega = 2\pi$ rad/s



$t = 0.8s, \omega = 2\pi$ rad/s

Рисунок 1 — Положение сетки в разные моменты времени

Вращение осуществляется против часовой стрелки.

Заключение. В данной работе был рассмотрен процесс моделирования вращательного движения в OpenFOAM. В процессе рассмотрения были изучены фунции моделирования движения и их параметры. В примере mixerVesselAMI2D было разобранно поведение жидкости при ее перемешивании. Получены графики положения подобласти с лопастями внутри области.