МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

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

Кафедра Математического и компьютерного моделирования

Моделирование динамики твердых тел с применением алгоритма шарнирно-сочленённого тела

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 413 группы

направление 01.03.02 — Прикладная математика и информатика

механико-математического факультета

Желтова Евгения Алексеевича

Научный руководитель
Старший преподаватель В.С. Кожанов

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

Саратов 2019

Введение. Данная бакалаврская работа посвящена моделированию твердого тела с применением алгоритма шарнирно-сочлененного тела, рассматривается система абсолютно твердых тел, соединенных шарнирами. Для этого использована программа OpenFOAM 6.0.

Данная работа может предствлять ценность в различных областях механики, для решения задач динамики манипуляторов, в частности, динамики гидравлических кранов-манипуляторов.

В статье [1] представлен алгоритм определения напряжений в стержневых элементах конструкций гидравлических кранов-манипуляторов.

В этой статье манипулятор рассматривается как система абсолютно твердых тел (звеньев), соединенных шарнирами. В данном случае, в гидравлических кранах-манипуляторах в качестве звеньев выступают, как правило, тонкостенные стержни.

Для решения прямой задачи динамики в работе был предложен O(n)-сложный алгоритм шарнирно-сочлененного тела (ABA – Articulated Body Algorithm), также известный как алгоритм Фезерстоуна. В основе алгоритма лежит понятие об инерции шарнирно-сочлененного тела: i-м шарнирно-сочлененным телом называется кинематическая ветвь манипулятора, берущая начало в i-м звене манипулятора и рассматриваемая отдельно от остальной части конструкции. Его инерция характеризует связь между нагрузкой, приложенной к i-му звену манипулятора, и ускорением данного звена, когда оно рассматривается не как часть всего манипулятора, а лишь как часть i-го шарнирно-сочлененного тела. Первая версия алгоритма ABA могла работать только с шарнирами с одной степенью свободы. Преодолеть это ограничение и увеличить скорость работы алгоритма удалось в последующих модификациях алгоритма.

Актуальность данной бакалаврской работы можно продемонстрировать на примере задач, стоящих перед современной промышленностью. В статье [1] утверждается, что в настоящее время исследования в области динамики, прочности и оптимального проектирования гидравлических крано-манипуляторных установок мобильных транспортно-технологических машин специального и гражданского назначения являются весьма актуальными и в них заинтересовано как отечественные, так и зарубежные производители.

Общая структура работы. Для демонстрации работы алгоритма шарнирно-сочлененного тела в программе OpenFOAM 6.0 был смоделирован процесс, который заключается в запуске волны из одного угла куба в другой. Внутри куба находится жидкость. В ней плавают тела, на которые наложены определенные ограничения. В модели заданы различные параметры, такие как плотность жидкости, жесткость пружины и проч. В работе показано, как менялся процесс при различных значениях этих параметров.

Практическая значимость. Математическое моделирование нагрузки эксплуатирующегося оборудования дает возможность в максимальной степени учесть специфические особенности условий его работы и обслуживания. Это позволяет построить индивидуальные графики нагрузки и, таким образом, прогнозировать протекание процесса изнашивания оборудования. Способность математического моделирования факторов нагрузки представляет практический интерес для анализа работы грузоподъемных машин, относящихся к опасным техническим устройствам и требующим индивидуальной оценки их текущего функционального состояния.

Постановка задачи. Сочлененное тело. Представим тело, состоящее из n последовательно соединенных хорд. Отсечем все хорды, начиная с i-го элемента. Оставшаяся «связка» называется подцепью. Эта подцепь называется сочлененным телом изначального тела. Инерция этого тела называется инерцией сочлененного тела.

Хорда. Под хордой в данном случае понимается какое-либо тело.

Связка. Это соединение между двумя соседними хордами. Для решения задач динамики твердого тела используется алгоритм для сочлененного тела. В данном разделе этот алгоритм будет подробно рассмотрен. Данный алгоритм может быть использован для моделирования задач динамики твердого тела высокой степени сложности.

Алгоритм шарнирно-сочлененного тела позволяет решать задачу динамики для манипулятора, представленного кинематической цепью без ответвлений, или кинематическим деревом, и не поддерживает замкнутых контуров.

Для решения задач динамики твердого тела используется алгоритм для сочлененного тела. В данном разделе этот алгоритм будет подробно рассмотрен. Алгоритм может быть использован для моделирования задач динамики

твердого тела высокой степени сложности. Для лучшего понимания алгоритма ограничимся простым случаем, чтобы в дальнейшем рассмотреть более общий случай.

Уравнение движения системы твердых тел может быть записано следующим образом [2]: $H(q)\ddot{q} + C(q, \dot{q}) = \tau$.

Здесь q, $\dot{q}$, $\ddot{q}$ - это позиция, скорость и ускорение связок соответственно. H - матрица инерции. C - это сила, которая обеспечивает нулевое ускорение. Она учитывает гравитацию, силу Кориолиса и центростремительную силу. τ - это результирующая сила, действующая на тела. В данном уравнении значения могут быть вычислены в зависимости от того, какие начальные данные имеются. Можно выделить прямую и обратную задачи динамики.

Для прямой задачи имеем [1,3]:

$$\ddot{q} = FD(model, q, \dot{q}, \tau) \tag{1}$$

В данном случае известны начальные позиция, скорость и силы, которые используются для вычисления ускорения.

Для обратной задачи имеем [3,5]:

$$\tau = ID(model, q, \dot{q}, \ddot{q}) \tag{2}$$

Этот случай является обратным к предыдущему, то есть, дано ускорение, с помощью которого вычисляется сила, необходимая для данного ускорения. Работа будет вестись с прямой задачей динамики, то есть силы известны, необходимо вычислить ускорение. На первом шаге это ускорение будет проинтегрировано, чтобы получить новую позицию и скорость, которые будут служить входными данными на следующем шаге. Продолжаться это будет на протяжении всего процесса моделирования.

OpenFOAM. OpenFOAM — свободно распространяемый инструментарий вычислительной гидродинамики для операций с полями (скалярными, векторными и тензорными). На сегодня является одним из законченных и известных приложений, предназначенных для FVM-вычислений.

OpenFOAM, изначально разрабатываемый в Великобритании компанией OpenCFD, Limited, в настоящее время поддерживается и развивается усили-

ями некоммерческой организации The OpenFOAM Foundation, основателями которой являются Henry Weller (создатель исходного кода FOAM), Chris Greenshields и Cristel de Rouvray. Своё название и идеологию построения код берет от предшественника FOAM (Field Operation And Manipulation). Первоначально программа предназначалась для прочностных расчетов, но в результате многолетнего академического и промышленного развития на сегодняшний момент позволяет решать множество различных задач механики сплошных сред (не ограничиваясь ею).

ParaView. ParaView - это мультиплатформенное приложение для анализа и визуализации данных с открытым исходным кодом. Пользователи ParaView могут быстро создавать визуализации для анализа своих данных, используя качественные и количественные методы. Исследование данных может быть выполнено в интерактивном режиме в 3D или программно с использованием возможностей пакетной обработки ParaView. ParaView был разработан для анализа чрезвычайно больших наборов данных с использованием вычислительных ресурсов распределенной памяти. Он может быть запущен на суперкомпьютерах для анализа больших наборов данных, а также на ноутбуках для небольшого объема информации.

Кодовая база ParaView разработана таким образом, чтобы все ее компоненты можно было повторно использовать для быстрой разработки узконаправленных приложений. Такая гибкость позволяет разработчикам ParaView быстро разрабатывать приложения, которые имеют определенные функциональные возможности для конкретной области. ParaView работает на распределенной и совместно используемой памяти параллельных и однопроцессорных систем. Он был успешно развернут на Windows, Mac OS X, Linux, SGI, IBM Blue Gene, Cray и различных рабочих станциях Unix, кластерах и суперкомпьютерах.

Решатель InterFoam. Решатель InterFoam позволяет уравнения Навье-Стокса для двух несжимаемых изотермических несмешивающихся жидкостей. Это означает, что свойства материала постоянны в области, заполненной одной из двух текучих сред, за исключением межфазной границы.

Уравнение неразрывности с постоянной плотностью имеет вид:

$$\frac{\partial u_j}{\partial x_j} = 0 \tag{5}$$

Уравнение сохранения количества движения имеет вид:

$$\frac{\partial(\rho u_i)}{\partial t} + \frac{\partial}{\partial x_j}(\rho u_j u_i) = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}(\tau_{ij} + \tau_{t_{ij}}) + \rho g_i + f_{\sigma i} \tag{6}$$

где u - скорость, g_i - ускорение свободного падения, p - давление, τ_{ij} и $\tau_{t_{ij}}$ - вязкое и турбулентное напряжение соответственно. $f_{\sigma i}$ - поверхностное напряжение.

Плотность ρ определена следующим образом:

$$\rho = \alpha\rho_1 + (1 - \alpha)\rho_2 \tag{7}$$

α равно единице внутри первой жидкости с плотностью ρ_1 и равно нулю внутри второй жидкости с плотностью ρ_2. На границе между двумя жидкостями α варьируется в пределах от 0 до 1. Поверхностное напряжение $f_{\sigma i}$ рассчитывается следующим образом:

$$f_{\sigma i} = \sigma\kappa\frac{\partial\alpha}{\partial x_i} \tag{8}$$

σ - коэффициент поверхностного натяжения, κ - кривизна. Кривизна может быть приближенно рассчитана следующим образом:

$$\kappa = -\frac{\partial n_i}{\partial x_i} = -\frac{\partial}{\partial x_i}\left(\frac{\partial\alpha\backslash\partial x_i}{|\partial\alpha\backslash\partial x_i|})\right) \tag{9}$$

Чтобы узнать, где находится граница между двумя жидкостями, необходимо решить дополнительное уравнение для α:

$$\frac{\partial\alpha}{\partial t} + \frac{\partial(\alpha u_j)}{\partial x_j} = 0 \tag{10}$$

Исходный код решателя находится в файлах interFoam.C, UEqn.H, pEqn.H, alphaEqn.H.

Библиотека динамики твердого тела. Файловая структура библиотеки rigidBodyDynamics выглядит следующим образом: bodies, joints, restraints,

rigidBodyInertia, rigidBodyModel, rigidBodyModelState, rigidBodyMotion, rigidBodySolvers.

Библиотека твердого тела содержит в себе множество файлов, однако в данной работе рассматриваются три основных раздела, которые будут использоваться для постановки задачи. Три основных раздела - это: тела, связки и силы, действующие на тело.

В данном разделе будут рассмотрены типы тел, доступные в данной библиотеке. В ней содержатся следующие файлы: compositeBody, cuboid, jointBody, masslessBody, rigidBody, sphere, subBody.

Эти файлы используются, чтобы создавать такие тела как параллепипед, сфера, невесомое тело и проч.

Cuboid (Параллепипед) - простая и всем знакомая геометрическая фигура. Чтобы создать его, требуется знать длины трех его граней. Ввести их можно в dynamicMeshdict, как будет показано позднее.

Sphere (Сфера) - для ее создания требуется задать радиус.

Massless body (Невесомое тело) - тело, использующееся как соединение между двумя другими телами и обладающее нулевой массой.

rigidBody (Твердое тело) - применяется для создания различных геометрических фигур, формы которых не определены заранее. В руководстве под названием «floatingObject» в разделе «dynamicMeshDict» cuboid может быть заменен на rigidBody.

Joints (Связки). В системе, состоящей из множества тел, тела связаны друг с другом с помощью связок. Связки по большей части делятся на 3 простых типа [6]:

· Призматические

· Вращающиеся

· Сферические

Призматическая связка определяет ось переноса. В чисто призматической связке может осуществляться только скользящее движение.

Вращающаяся связка определяет ось вращения. Вокруг чисто вращающейся связки может осуществляться только вращательное движение.

Существует также еще несколько типов связок, такие как составная связка, подвижная связка, нулевая связка и т.д. Составная связка используется,

когда тело связано комбинацией связок, таких как смесь призматических и вращающихся связки. Подвижная связка обеспечивает 6 степеней свободы. Нулевая связка используется в конструировании связок в C++ файлах, и все тела вводятся, начиная с этой связки. Призматические связки с соответствующими степенями свободы записаны ниже [10]:

· Px-перенос вдоль оси OX

· Py-перенос вдоль оси OY

· Pz-перенос вдоль оси OZ

· Pxyz-перенос вдоль осей OX, OY и OZ

· Pa-перенос вдоль произвольной оси, которая вводится пользователем

Ось может быть введена пользователем в dynamicMeshDict. Аналогично для вращающихся связок доступны следующие опции:

· Rx-Вращение вокруг оси OX

· Ry-Вращение вокруг оси OY

· Rz-Вращение вокруг оси OZ

· Rxyz-сферическая связка для вращения вокруг осей OX, OY, OZ для x, y, z соответственно с помошью углов Эйлера

· Ryxz-сферическая связка для вращения вокруг осей OX, OY, OZ для y, x, z соответственно с помошью углов Эйлера

· Rzyx-сферическая связка для вращения вокруг осей OX, OY, OZ для z, y, x соответственно с помошью углов Эйлера

· Ra-вращение вокруг произвольной оси, которая вводится пользователем

Restraints (Ограничители). Существуют следующие виды ограничителей: linearAxialAngularSpring, linearDamper, linearSpring, sphericalAngularDamper.

linearAxialAngularSpring. linearAxialAngularSpring - это торсионная пружина, которая прикрепляется к телу. Строка «axis» определяет ось вращения тела, которая вводится как вектор. Строка referenceOrientation (установка в исходное положение) вводится как тензор. Строка Stiffenss (устойчивость) - это устойчивость торсионной пружины. Строка damping - это коэффициент амортизации торсионной пружины. Этот ограничитель вычисляет вращающий момент пропорционально углу поворота тела и всегда действует в сторону, обратную вращению.

linearDamper. linearDamper задает силу реакции, прямопропорциональную скорости тела. Коэффициент демпфера задается пользователем. Данный ограничитель реагирует только на прямолинейное движение. Все силы демпфера прикладываются к центру тяжести тела. Силы демпфера всегда прикладываются в направлении, противоположном вектору линейной скорости тела.

sphericalAngularDamper. sphericalAngularDamper задает силу реакции, прямо пропорциональную угловой скорости тела.

linearSpring. linearSpring предоставляет более полную модель для реакции связи. Строки stiffness и damping обозначают то же, что и в случае с linearAxialAngularSpring. Строка attachmentPoint обозначает точку, к которой будет приложена реакция связи. Строка restLength позволяет задавать начальные условия напряженности или сжатия пружины.

Примеры моделирования динамики некоторых тел. Каждая модель представляет собой тела, плавающие в жидкости внутри куба. В ходе моделирования запускается волна, которая тем или иным образом, в зависимости от значений параметров, влияет на поведение тел в жидкости. Каждое тело описывается в файле topoSetDict1 и topoSetDict2 соответственно (в случае, когда имеем лишь одно тело, создается один файл topoSetDict в той же папке), расположены эти файлы в папке system. Параметры тела и жидкости, которые будут меняться в ходе моделирования, заданы в файле dynamicMeshDict, который находится в папке constant.

Ограничитель и его параметры также задаются в файле dynamicMeshdict.

Два тела, связанные с дном. В первом случае имеются два тела, плавающих в жидкости, которые связаны с дном куба при помощи ограничителя linearSpring. Жидкость окрашена красным цветом, воздух – синим.

Два тела, связанные друг с другом. Во втором случае имеются те же два тела, которые соединены друг с другом с помощью того же ограничителя. В результате тела начали менять свое положение по оси OX, так как именно по этой оси действует linearSpring.

Первое тело прикреплено к боковой грани. В третьем случае первое тело тем же linearSpring прикреплено к боковой грани кубика, а второе, как и в первом случае, прикреплено ко дну. В результате первое тело колеблется

по оси OZ гораздо больше, чем второе, так как второе тело сдерживается пружиной.

Плотность жидкости равна 1500. Четвертая модель почти аналогична первой, с той разницей, что здесь плотность жидкости больше в полтора раза. Во всех предыдущих случаях она равнялась 998,2, что соответствует плотности воды. В этом случае плотность жидкости равна 1500. В результате тела больше выглядывают из поверхности жидкости.

Жесткость пружины увеличена в 2 раза. В пятой модели жесткость пружины, которая соединяет первое тело со дном, увеличена в 2 раза. В результате оно гораздо меньше смещается по оси OZ.

Замена кубика на сферическое тело. В шестом случае вместо двух кубиков в жидкости плавает сфера, так же прикрепленная ко дну с помощью linearSpring.

Заключение данной работе смоделирована динамика твердого тела с применением алгоритма шарнирно-сочлененного тела с помощью OpenFOAM. Рассмотрено поведение связанных твердых тел, плавающих в жидкости, при раличных значениях параметров. Продемонстрирован результат работы решателей rigidBodyMotion и InterFoam.