1 Постановка задачи

a) Реализовать программу моделирования методом динамики частиц на C#.

- Использовать подход ООП
- Выводить результат в формат, пригодный для визуализации движения частиц в 3D.

b) Провести привязку констант программы, параметров потенциала и т. д. к физическим параметрам моделируемого материала.

- Реализовать потенциал Леннард-Джонса или потенциал Ми либо другой потенциал на выбор.
- Реализовать создание решетки FCC для задания твердых тел

c) Провести расчет какой-либо реальной физической задачи с использованием написанной программы

2 Математическая модель

2.1 Потенциал взаимодействия

В данной работе в качестве потенциала взаимодействия между частицами взят потенциал Леннард-Джонса Π_{lj}, имеющий вид:

$$\Pi_{lj} = D\left[\left(\frac{a_0}{r}\right)^{12} - 2\left(\frac{a_0}{r}\right)^{6}\right] \tag{1}$$

где D - энергия взаимодействия; a_0 - длина связи; r - расстояние между частицами. Сила взаимодействия, соответствующая потенциалу Леннарда-Джонса, вычисляется по формуле:

$$F_{lj} = -\Pi'_{lj} = \frac{12D}{a_0}\left[\left(\frac{a_0}{r}\right)^{13} - \left(\frac{a_0}{r}\right)^{7}\right] \tag{2}$$

2.2 Модифицированный алгоритм Верле (Leapfrog)

Для численного интегрирования уравнений движения в данной работе используется модифицированный алгоритм Верле

$$\ddot{\mathbf{r}}_i = \frac{d^2\mathbf{r}_i}{dt^2} = \mathbf{F}_i \tag{3}$$

или в равносильной форме

$$\dot{\mathbf{v}}_i = \frac{d\mathbf{v}_i}{dt} = \mathbf{F}_i, \;\; \dot{\mathbf{r}}_i = \frac{d\mathbf{r}_i}{dt} = \mathbf{v}_i \tag{4}$$

Плюс данного метода в высокой точности и стабильности, в сравнении с другими методами второго порядка, а так же то, что при интегрировании уравнений движения сохраняется энергия системы. В данной работе $dt = 0.004$.

3 Программная реализация

3.1 Начальная конфигурация

В данной работе рассматривается колебание стержня, один конец которого неподвижен.

За безразмерные величины радиуса и массы приняты $r = 1, m = 1$ соответственно. Каждая частица рассматриваемого тела имеет радиус $r_{FCC} = r$ и массу $m_{FCC} = m$. В начальный момент времени один из концов стержня имеет скорость $V_0 = v$, где $v = 1$ - безразмерная величина. Начальная конфигурация модели представлена на Рис. 1. Для удобства визуального восприятия используется color gradient по оси OX, который характеризует только позицию частицы по оси OX.

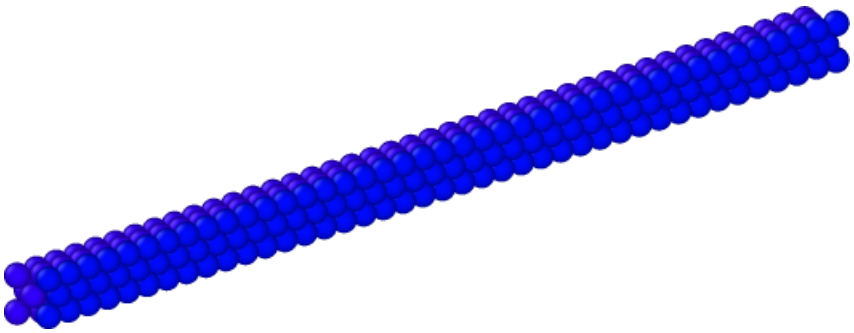

Рис. 1: Начальная конфигурация системы

3.2 Структура программы

Так как программа реализована с помощью языка программирования C#, то основой программы являются классы. В данной работе использовалось 8 различных классов:

- **Const.cs** - класс, содержащий в себе все константы, используемые в программе.

```
public static class Const
    {
        public static readonly int D = 1; //энергия связи
           в потенциале Л-Дж
        public static readonly int a0 = 1; //равновесное
           расстояние в потенциале Л-Дж
        public static readonly int a_cut = 4 * a0;
           //расстояние, отвечающее за определение
           соседних частиц
        public static readonly int n_wall_rows = 1;
           //число "строчек"  частиц в стене
        public static readonly int n_wall_col = 1; //число
           "столбцов"  частиц в стене
        public static readonly int id_standart = 0;
           //порядковый номер частицы
        public static readonly int type_standart = 1;
           //тип частицы(нужно для визуализации)
        public static readonly double MinMass = 0.001;
           //минимальная масса частицы (для отлова ошибок)
        public static readonly double MinDist = 0.001;
           //минимальное начальное расстояние между
           частицами (для отлова ошибок)
        public static readonly double DefaultMass = 1;
           //безразмерная массы
        public static readonly double DefaultDiameter = 1;
           //безразмерный диаметр частицы
        public static readonly double MaxTime = 80;
           //время интегрирования
        public static readonly double dt = 0.004; //шаг
           интегрирования
        public static readonly double BeginT = 0; //
           начальное время
    }
```

- **Vector.cs** - класс, позволяющий работать с векторами.
- **Particle.cs** - класс, позволяющий задать одну частицу.
- **Creator.cs** - класс, отвечающий за начальную конфигурацию системы.
- **WorkingModel.cs** - класс, позволяющий работать со всеми созданными частицами.
- **Integrator.cs** - класс, отвечающий за интегрирование уравнений движения частиц.
- **Optimization.cs** - класс, отвечающий за оптимизацию и ускорение вычислений.

- **OutputHelp.cs** - класс, позволяющий сохранять данные в пригодной для визуализации форме.

```
public class OutputHelp
    {
        public void SaveCurrentConfiguration(WorkingModel
           model)
        {
            string Text = "";
            Text = "ITEM: TIMESTEP" + "\r\n";
            Text += model.Time + "\r\n";
            Text += "ITEM: NUMBER OF ATOMS" + "\r\n";
            Text += model.size + "\r\n";
            Text += "ITEM: BOX BOUNDS ps ps ps" + "\r\n";
            Text += model._x + " " + model.x_ + "\r\n";
            Text += model._y + " " + model.y_ + "\r\n";
            Text += model._z + " " + model.z_ + "\r\n";
            Text += "ITEM: ATOMS id type xs ys zs" +
               "\r\n";

            foreach (var p in model.Particles)
            {
                Text += p.id + " " + p.type + " " + p.R.X
                   + " " + p.R.Y + " " + p.R.Z + "\r\n";
            }
           string NewText = Text.Replace(",",".");
           System.IO.File.AppendAllText(@"WriteLine.txt",
              NewText);
        }
    }
```

4 Результаты моделирования

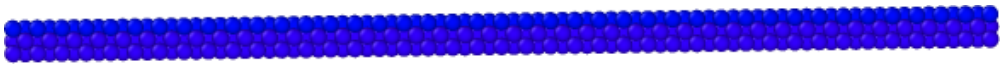

Рис. 2: кадр №0

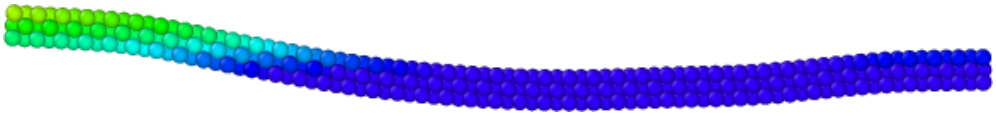

Рис. 3: кадр №1

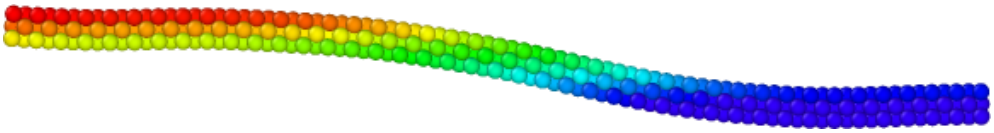

Рис. 4: кадр №2

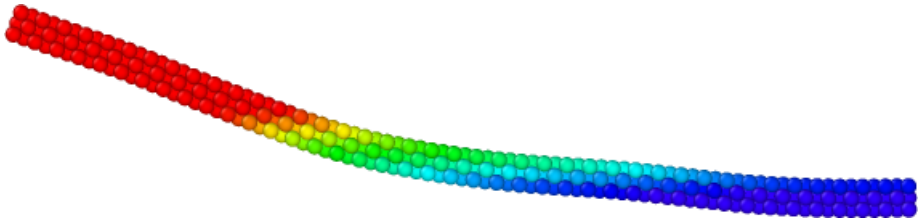

Рис. 5: кадр №3

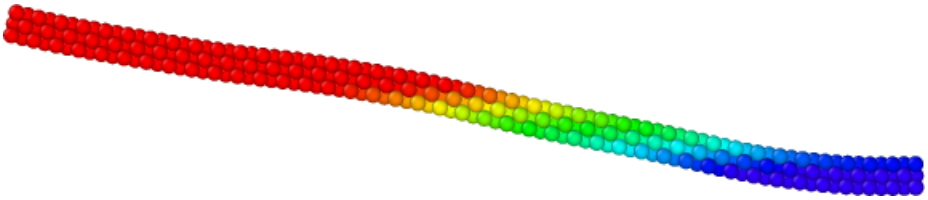

Рис. 6: кадр №4

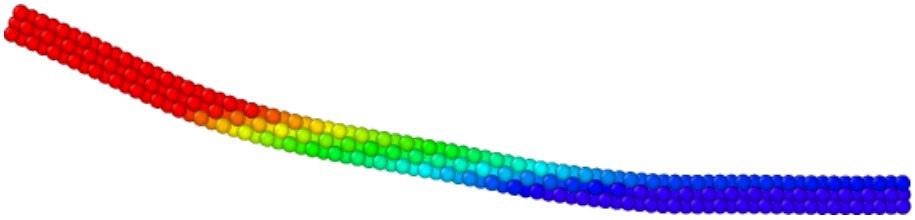

Рис. 7: кадр №5

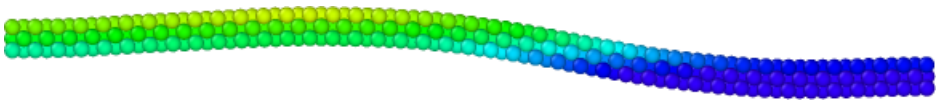

Рис. 8: кадр №6

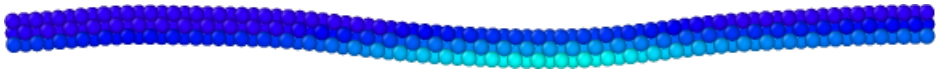

Рис. 9: кадр №7

5 Выводы

Была реализована программа моделирования с использованием идеи обьектно-ориентированного программирования методом динамики частиц на C#. Результаты были выведены в графическом виде. Константы программы, параметров потенциала и т. д. были привязаны к физическим параметрам моделируемого материала. В программе был реализован потенциал Леннард-Джонса и проведен расчет поперечных колебаний стержня. Полученные результаты совпадают с графическими формами колеблющегося стержня.

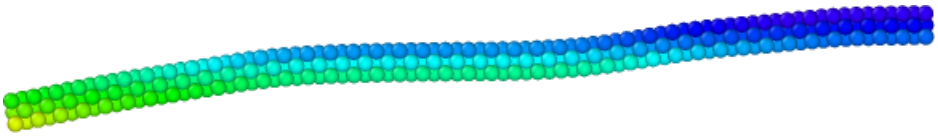

Рис. 10: кадр №8

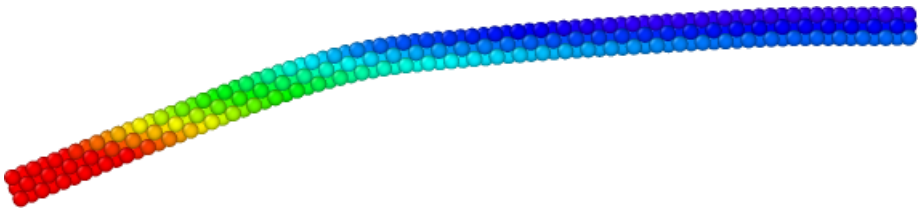

Рис. 11: кадр №9