

# Моделирование деформаций упругих объектов в виртуальных окружениях в реальном времени, используя метод конечных элементов и предвычисленные функции Грина

Никитина Л.Д., Никитин И.Н., Фролов П.В., Гёббельс Г., Гёбель М.  
*Фраунгоферовский Институт Медиакоммуникаций, Санкт Августин, Германия*  
Клименко С.В.  
*Институт Физики Высоких Энергий, Протвино, Россия*  
Нильсон Г.  
*Аризонский Университет, Тэмпе, Аризона, США*

## Аннотация

Моделирование упругих деформаций объектов чрезвычайно важно для приложений виртуального окружения, в которых требуется реалистично представлять поведение трехмерных объектов. В случае взаимодействия пользователя с объектом особенно желательно вычислять эти деформации так быстро, как возможно. В данной работе мы представляем новый подход для моделирования упругих объектов в виртуальных окружениях в условиях реального времени. Наш подход использует метод конечных элементов и предвычисленные функции Грина. Деформации интерактивно визуализируются в виртуальных окружениях обратной проекции с полным погружением, таких как CyberStage и в системах с частичным погружением, таких как Responsive Workbench. Взаимодействуя с объектом посредством виртуального луча, пользователь может интерактивно прикладывать силы к объекту, производя его деформации.

**Ключевые слова:** упругие деформации, виртуальные окружения, метод конечных элементов, функции Грина.

## 1 Введение

В большинстве приложений моделирования, таких как инженерные, медицинские и т.д. имеется большая необходимость в подходах, позволяющих моделирование материальных свойств объекта и его поведения под действием приложенных к нему сил. В медицинских приложениях требуется моделирование мягких тканей; в этой области основоположниками являются Бро-Нильсен и Котин [1], которые использовали *метод конечных элементов* для моделирования упругих деформаций. Технические приложения нуждаются, в основном, в моделировании упругих деформаций объектов, компоненты которых состоят из изотроп-

ных материалов, в условиях применимости закона Гука [2]. Обзорная работа [3] описывает большую часть предшествующих работ по моделированию деформируемых объектов. Недавними достижениями являются применение метода граничных элементов и формулы Шермана-Моррисона-Вудбари для моделирования линейно-упругих однородных объектов [4], и использование итеративных динамических схем [5, 6] для моделирования нелинейной упругости. Для нелинейно-упругих объектов достаточно высокой детализации до сих пор не удается моделировать деформации в реальном времени. Для упрощенной, линейной формулировки теории упругости существует возможность решать системы уравнений достаточно большого размера в реальном времени синхронно с графической обработкой, или, для ускорения метода, часть этих вычислений может быть выполнена предварительно. Также в зависимости от внутреннего строения объекта можно использовать метод конечных элементов или метод граничных элементов. В данной работе мы коротко обсудим эти различные подходы и затем опишем методы и техники, использованные в нашей системе, которые концентрируются на применении линейной теории упругости, метода конечных элементов и предвычислении полного базиса решений (функций Грина). Система разработана в программной среде Аванго и используется для моделирования в реальном времени упругих деформаций трехмерных объектов в виртуальном окружении, позволяя интерактивное, определенное пользователем приложение силы.

## 2 Системы виртуального окружения

Предназначение виртуальных окружений заключается в том, чтобы обеспечить отдельных пользователей, или группы инженеров, дизайнеров, медиков виртуальным рабочим пространством, в котором они могут наблюдать, исследовать и создавать в реальном времени необходимые им виртуальные данные. Большинство таких виртуальных окружений имеют сходную аппаратную конфигурацию. Центральное место занимает графический обработчик: SGI суперкомпьютер или кластер Linux PC, который обрабатывает данные и отображает их на экран, а также устройство слежения, которое измеряет положение и ориентацию головы пользователя. Данные, произведенные устройством слежения, читаются графическим обработчиком, чтобы определить перспективно правильное изображение для любой точки зрения пользователя. Аппаратурная конфигурация, используемая для интерактивной визуализации деформируемых объектов в подходе данной работы включает устройства ввода для взаимодействия и стереоскопические системы обратной проекции в качестве устройств вывода. Далее мы описываем наиболее часто используемые проекционные системы Responsive Workbench и CyberStage, а также систему разработки виртуальных окружений Аванго, используемую для визуализации.



Рис.1. Двусторонний Responsive Workbench.

**Responsive Workbench™** – это стереоскопическая проекционная система высокого разрешения, разработанная в отделе виртуальных окружений Немецкого Национального Исследовательского Центра Информационных Технологий [7]. Responsive Workbench использует CRT-проектор, который отображает стерео-изображения через зеркало на плоскость экрана. Для каждого глаза вычисляется соответствующее изображение, графический обработчик быстро чередует эти

два изображения на экране. Пользователи носят жидкокристаллические шаттр-очки, у которых левое стекло становится непрозрачным, в то время как изображение для правого глаза отображается на экране, и наоборот. Тем самым производится стереоскопический эффект. Электромагнитный датчик фирмы Polhemus, у которого измеряются все 6 степеней свободы, т.е. положение и ориентация, прикреплен к шаттр-очкам. Это позволяет системе вычислить перспективно правильное изображение для любого положения пользователя. Пользователи взаимодействуют напрямую с 3-мерными виртуальными объектами, находящимися в поле зрения, используя устройства ввода, у которых аналогичным образом измеряются положение и ориентация. Типичными областями приложений Responsive Workbench являются инженерная и медицинская визуализация.

Новый двусторонний Responsive Workbench имеет горизонтальный и вертикальный экраны, которые гладко примыкают друг к другу (рис.1). Дополнительный вертикальный экран значительно увеличивает поле зрения и позволяет наблюдать виртуальные объекты на уровне глаз пользователя, что ранее не было возможно. Используя современные персональные компьютеры и графические рабочие станции, двусторонний Responsive Workbench производит стерео-изображение с разрешением 1280x1024x96Гц. Благодаря расширенному полю зрения двусторонний Responsive Workbench используется в более широком диапазоне приложений, чем односторонний.

**CyberStage™** – это проекционная система объемного изображения и звука, в которой достигается эффект полного погружения [8, 9]. Она имеет размеры небольшой комнаты (3м×3м×2.4м) и включает 4-стороннюю проекцию стерео-изображения и 8-канальную проекцию пространственного звука, контролируемые положением головы пользователя. Суперкомпьютер SGI Onyx 2 с четырьмя графическими картами Infinite Reality 2 и двенадцатью процессорами MIPS R12000 производит четыре изображения с разрешением 1280x1024x120Гц. Каждая карта в пиковом режиме генерирует 11 млн. треугольников в секунду.

Для создания иллюзии присутствия в виртуальных пространствах, система CyberStage предоставляет различные интерфейсы и метафоры взаимодействия, визуально и акустически откликаясь на действия пользователя. Эти интерфейсы используются для навигации в виртуальных пространствах и для взаимодействия с виртуальными объектами. Программное обеспечение для CyberStage и Responsive Workbench – это система разработки виртуальных окружений Аванго [10].

**Аванго™** – среда программирования, предназначенная для создания распределенных взаимодействующих

щих приложений виртуального окружения. Аванго использует язык программирования C++ для определения двух категорий объектов – узлы и датчики. Узлы составляют объектно-ориентированный *граф сцены*, который осуществляет представление изображаемой геометрической модели. *Датчики* обеспечивают связь с реальным миром и используются для ввода данных с внешних устройств в приложение. Все объекты Аванго – полевые контейнеры, представляющие информацию о своем состоянии в виде совокупности *полей*. Объекты Аванго обладают универсальным потоковым интерфейсом, который позволяет записывать в поток сами объекты и информацию об их состоянии и впоследствии реконструировать эти данные из потока. Аванго использует язык сценариев *Scheme* [11] для объединения объектов в граф сцены, а также вводит понятие графа потоков данных, концептуально ортогонального графу сцены и служащего для определения взаимодействия между объектами и для ввода в сцену реальных данных, необходимых для моделирования интерактивного поведения. Scheme – универсальный язык программирования, происходящий от Алгола и Лиспа. Это язык высокого уровня, который умеет оперировать со структурными данными, такими как строки, списки и векторы. Все объекты Аванго могут быть созданы и управляемы посредством Scheme. Аванго основан на графической системе Irix/OpenGL Performer, которая обеспечивает наиболее качественное исполнение приложения, и удовлетворяет специальным требованиям, возникающим при разработке приложений для виртуального окружения. Расширенные графические возможности, такие как стирание невидимых граней, переключение степени детализации и взаимодействие с графическим оборудованием, целиком выполняются системой Performer. Если аппаратное обеспечение позволяет, Performer использует множественные процессоры и множественные графические конвейеры. Мы используем реализацию Аванго на графических станциях SGI и Linux-PC.

Наша цель – представить деформируемые модели в виртуальных окружениях. В современном математическом моделировании это обычно делается с помощью дискретных методов, таких как *метод конечных элементов*. При этом тело разбивается на конечное множество элементов, и определяются условия физического равновесия для каждого из элементов. Эти условия записываются в виде очень большой системы уравнений с десятками тысяч неизвестных. Такая система обычно является вырожденной, но после наложения соответствующих граничных условий становится хорошо определенной и имеет однозначное решение. Для нахождения решения необходимо чрезвычайно интенсивное вычисление, требующее большого объема оперативной памяти и времени исполнения. Для приложений вирту-

альной реальности желательно оптимизировать алгоритм решения, в особенности по отношению к его скорости исполнения. В следующих разделах мы приводим необходимые детали о методах, используемых для моделирования деформируемых объектов в реальном времени.

### 3 Математические основы метода

**Теория упругости** [2] характеризует деформации упругих объектов с помощью смещений  $\vec{u}(\vec{x})$ , преобразующих точки объекта следующим образом:  $\vec{x} \rightarrow \vec{x} + \vec{u}(\vec{x})$ . Тензор деформации определяется частными производными:  $\epsilon_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} + \frac{\partial u_k}{\partial x_i} \frac{\partial u_k}{\partial x_j} \right)$ , где предполагается суммирование по повторяющимся индексам. Тензор деформации характеризует изменение расстояний в деформированном объекте, так что длина малого элемента  $d\vec{x}$  преобразуется следующим образом:  $|d\vec{x}|^2 \rightarrow |d\vec{x}|^2 + 2\epsilon_{ij}dx_i dx_j$ . В случае малых смещений можно сохранить в тензоре деформации только линейные члены, записывая просто  $\epsilon_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)$ , результирующее приближение называется теорией малых деформаций. Удобно поместить компоненты тензора деформации в столбец  $\epsilon = (\epsilon_{xx}, \epsilon_{yy}, \epsilon_{zz}, 2\epsilon_{xy}, 2\epsilon_{xz}, 2\epsilon_{yz})^T$ , и записать его определение через смещения как  $\epsilon = \Delta u$ , где  $\Delta$  – символическая матрица дифференцирования:

$$\begin{bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \epsilon_{zz} \\ 2\epsilon_{xy} \\ 2\epsilon_{xz} \\ 2\epsilon_{yz} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} \quad (1)$$

Распределение сил в упругом теле характеризуется следующим образом. Рассмотрим воображаемый срез тела и малый элемент  $d\vec{S}$  на нем (этот вектор направлен вдоль нормали к срезу, модуль  $|d\vec{S}|$  равен площади элемента). Сила  $d\vec{f}$ , действующая сквозь этот элемент между частями тела, разделенными срезом, определяется как  $d\vec{f}_i = \sigma_{ij}dS_j$ . Коэффициент  $\sigma_{ij}$  называется тензором напряжения. Он симметричен:  $\sigma_{ij} = \sigma_{ji}$ , также как тензор деформации, и может быть записан аналогично:  $\sigma = (\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{xz}, \sigma_{yz})^T$ .

Тензоры напряжения и деформации связаны законом Гука:  $\sigma = D\epsilon$ , где матрица  $D$  не зависит от деформаций и характеризует свойства материала. Это линейное соотношение справедливо для малых деформаций при дополнительном предположении, что недеформированное тело обладает нулевым тензором напряжения, т.е. материал не является преднапряжен-

ным. Для однородного изотропного материала матрица  $D$  представляется в виде линейной комбинации  $D = \lambda D_1 + \mu D_2$ , где  $D_{1,2}$  – постоянные 6x6 матрицы:

$$D_1 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad D_2 = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

(пустые места заполнены нулями) и  $\lambda, \mu$  – постоянные Ламе, например

	$\lambda$ , ГПа	$\mu$ , ГПа
керамика	124.12	124.12
сталь	119.42	79.23
резина	1.12	0.047

Энергия деформированного тела задается интегралом  $E = \int dV (\frac{1}{2} \epsilon^T \sigma + U)$ , где первое слагаемое представляет чисто упругую энергию, а второе ассоциируется с потенциальным полем внешних сил  $\vec{f} = -\nabla U$  (например, поле тяжести). Условия равновесия, соответствующие минимуму энергии, имеют вид  $\frac{\partial \sigma_{ij}}{\partial x_j} = f_i$ . В терминах неизвестных смещений  $\vec{u}$  это дает систему линейных дифференциальных уравнений в частных производных 2-го порядка.

При подстановке в правую часть этой системы сил вида  $f_i(\vec{x}) = \delta_{ik} \delta(\vec{x} - \vec{x}_0)$ , где  $\delta_{ik}$  – символ Кронекера и  $\delta()$  – функция Дирака, решение системы для данных  $\vec{x}_0, k$ :  $u_i(\vec{x}; \vec{x}_0, k) = G_{ik}(\vec{x}, \vec{x}_0)$  называется функцией Грина системы [12]. Эта функция описывает, как воздействие силы, приложенной локально в точке  $\vec{x}_0$ , распространяется в точку  $\vec{x}$ . Если эта функция известна, решение системы для произвольных правых частей  $\vec{f}(\vec{x})$  можно найти как  $u_i(\vec{x}) = \int d^3x_0 G_{ik}(\vec{x}, \vec{x}_0) f_k(\vec{x}_0)$ .

**Метод конечных элементов [13]** вводит дискретизацию тела, используя, например, разбиение тела на тетраэдры, и представляет непрерывное поле смещений  $\vec{u}(\vec{x})$  с помощью линейной интерполяции величин, заданных в узлах:

$$\vec{u}(\vec{x}) = \sum_{k=1}^4 N_k(\vec{x}) \vec{u}_k, \quad N_k(\vec{x}) = 1 - \frac{1}{3V} \vec{S}_k(\vec{x} - \vec{x}_k),$$

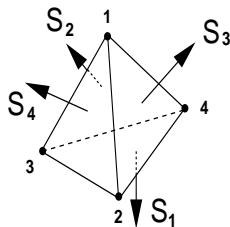


Рис.2. Тетраэдральный элемент.

где  $V$  – объем тетраэдра и  $\vec{S}_k$  – площади его граней, направленные по внешним нормальям (рис.2). Явные формулы, выражающие  $V, \vec{S}_k$  в терминах координат узлов  $\vec{x}_k$ :

$$\begin{aligned} \vec{S}_1 &= +\frac{1}{2}(d_{234}^x, d_{234}^y, d_{234}^z), \\ \vec{S}_2 &= -\frac{1}{2}(d_{341}^x, d_{341}^y, d_{341}^z), \\ \vec{S}_3 &= +\frac{1}{2}(d_{412}^x, d_{412}^y, d_{412}^z), \\ \vec{S}_4 &= -\frac{1}{2}(d_{123}^x, d_{123}^y, d_{123}^z), \end{aligned} \quad V = \frac{1}{6} \begin{vmatrix} 1 & 1 & 1 & 1 \\ x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \end{vmatrix},$$

$$d_{ijk}^x = \begin{vmatrix} 1 & 1 & 1 \\ y_i & y_j & y_k \\ z_i & z_j & z_k \end{vmatrix}, \quad d_{ijk}^y = \begin{vmatrix} 1 & 1 & 1 \\ z_i & z_j & z_k \\ x_i & x_j & x_k \end{vmatrix}, \quad d_{ijk}^z = \begin{vmatrix} 1 & 1 & 1 \\ x_i & x_j & x_k \\ y_i & y_j & y_k \end{vmatrix},$$

(ср. с [13, 1]). Частные производные записываются следующим образом:  $\frac{\partial u_i}{\partial x_j} = -\frac{1}{3V} \sum_{k=1}^4 S_k^j u_k^i$  и

$$\begin{bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \epsilon_{zz} \\ 2\epsilon_{xy} \\ 2\epsilon_{xz} \\ 2\epsilon_{yz} \end{bmatrix} = \underbrace{\begin{bmatrix} \vec{S}_1 & \vec{S}_2 & \vec{S}_3 & \vec{S}_4 \\ \hat{S}_1 & \hat{S}_2 & \hat{S}_3 & \hat{S}_4 \end{bmatrix}}_{\Delta} \begin{bmatrix} \vec{u}_1 \\ \vec{u}_2 \\ \vec{u}_3 \\ \vec{u}_4 \end{bmatrix}$$

где  $\vec{u}_k$  – столбцы  $(u_{kx}, u_{ky}, u_{kz})^T$  и  $\vec{S}_k, \hat{S}_k$  обозначают 3x3 блоки

$$\vec{S}_k = \begin{bmatrix} S_{kx} & 0 & 0 \\ 0 & S_{ky} & 0 \\ 0 & 0 & S_{kz} \end{bmatrix}, \quad \hat{S}_k = \begin{bmatrix} S_{ky} S_{kx} & 0 \\ S_{kz} & 0 & S_{kx} \\ 0 & S_{kz} & S_{ky} \end{bmatrix}$$

В этом соотношении  $\Delta$  – дискретизованный аналог матрицы дифференцирования из уравнения (1), ассоциированный с тетраэдрическим разбиением.

*Замечание:* для проверки трансляционной инвариантности системы, мы подставляем в это соотношение постоянное смещение  $\vec{u}_k = \vec{c}$  и немедленно получаем  $\epsilon = 0$  в силу тождества:  $\vec{S}_1 + \vec{S}_2 + \vec{S}_3 + \vec{S}_4 = 0$  (геометрическая теорема “о еж”). Аналогичным образом можно показать, что тензор деформации обращается в нуль при инфинитезимальных вращениях, т.е. при линейных преобразованиях вида  $\vec{u}_k = \alpha \vec{n} \times \vec{x}_k$ , где  $\alpha$  – угол и  $\vec{n}$  – ось вращения.

Упругая энергия для одного тетраэдра записывается как  $E_{tet} = \frac{1}{2} u^T K_{tet} u$ ,  $K_{tet} = \Delta^T D \Delta V$ , после суммирования по всем тетраэдрам и добавлении потенциальной энергии это дает  $E = \frac{1}{2} u^T K u + U$ , где  $K = \sum_{tet} \text{transfer}(K_{tet})$  называется *матрицей жесткости*. Функция “transfer” [1] выполняет переход от локальных индексов (1234), помечающих вершины одного тетраэдра, к глобальным индексам, помечающим узлы модели. Матрица  $K$  симметрична и положительно

полуопределена:  $K^T = K$ ,  $u^T K u \geq 0$ , она имеет нулевые направления  $Ku = 0$ , соответствующие описанным выше степеням свободы твердого тела (трансляциям и инфинитезимальным вращениям). Чтобы исключить эти степени свободы, должны быть зафиксированы граничные условия, для чего достаточно положить  $u = 0$  в  $n \geq 3$  узлах, не лежащих на одной прямой. Устранение строк и столбцов, соответствующих этим узлам, из матрицы  $K$  делает ее положительно определенной:  $u^T K u > 0$ ,  $u \neq 0$ , что соответствует строгому минимуму энергии. Условие минимума имеет вид системы линейных уравнений  $Ku = f$ , которая должна быть решена:  $u = K^{-1}f$ . Функция Грина в этих обозначениях равна обратной матрице  $K^{-1}$ , в которой  $(ij)$ -элемент описывает, как воздействие единичной силы, приложенной к  $j$ -тому узлу, распространяется в  $i$ -тый узел.

В более общем виде можно рассмотреть границу тела  $\Gamma = \partial\Omega$  и область границы  $\Gamma_b \subset \Gamma$ , где заданы ненулевые смещения:  $u = b$ . Ставится задача о нахождении смещения  $u$  на  $\Gamma \setminus \Gamma_b$  (и, если требуется, во внутренних областях  $\Omega$ ). Такие граничные задачи сводятся к задачам с заданной силой, используя следующее разбиение матрицы  $K$ :

$$\begin{array}{|c|c|} \hline A & B \\ \hline B^T & C \\ \hline \end{array} \begin{array}{|c|} \hline u \\ \hline b \\ \hline \end{array} = \begin{array}{|c|} \hline 0 \\ \hline f_b \\ \hline \end{array}$$

где  $f_b$  – сила, действующая на  $\Gamma_b$ , в то время как для других узлов предполагается отсутствие внешней силы. Отсюда мы получаем следующие две системы:  $B^T u + Cb = f_b$ , определение силы  $f_b$ , которую можно использовать, например, для генерации тактильных ощущений [14];  $Au = -Bb$ , система на  $u$ , непосредственно используемая при визуализации деформаций объекта. Дальнейшее упрощение [1] возможно, если изображается только граница (не внутренние области) тела. В последней системе участвуют смещения  $u$  как на границе, так и для внутренних узлов, однако, когда эта система решена:  $u = -A^{-1}Bb$ , необходимо сохранить в матрице  $A^{-1}B$  только тот блок, который соответствует граничным узлам, т.е. функцию Грина, описывающую распространение воздействия из граничных точек в граничные точки.

*Замечание:* к аналогичной формулировке приводит использование метода граничных элементов [4]. Этот метод использует *аналитическую* переформулировку исходных дифференциальных уравнений в частных производных к интегральному виду, который включает только поверхностные переменные (смещения и распределения сил по поверхности). Такая переформулировка возможна только для случая однородных изо-

тропных материалов. Полученные интегральные уравнения являются сингулярными, и для их численного решения требуются специальные регуляризации. В методе граничных элементов необходимо вводить разбиение только поверхности объекта, практически можно использовать ту же триангуляцию, что и для графической обработки. В методе конечных элементов разбиение необходимо вводить также во внутренних областях объекта, даже в том случае, если не требуется проводить визуализацию этих областей. Применение метода граничных элементов ограничено объектами, внутренность которых однородна и изотропна. Для объектов со сложной внутренней структурой необходимо использовать метод конечных элементов. Метод конечных элементов порождает большие и разреженные линейные системы, в то время как эквивалентные им системы в методе граничных элементов имеют меньший размер и являются плотными. Оба метода могут использоваться в режиме предварительных вычислений. В области совместной применимости они производят один и тот же результат – функцию Грина на поверхности. Этот результат можно затем использовать для ускорения вычислений в реальном времени. В дальнейшем мы опишем такую оптимизированную схему для метода конечных элементов.

## 4 Оптимизированная схема вычислений

### Хранение матрицы жесткости.

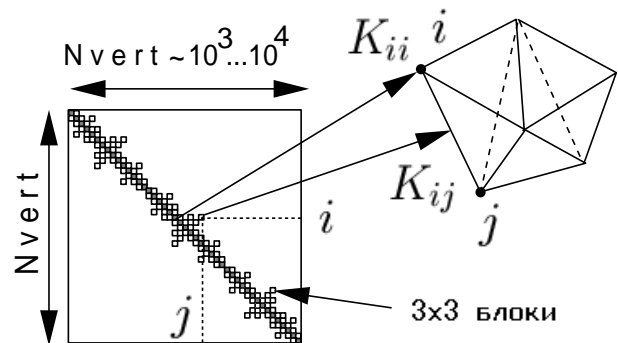


Рис.3. Хранение матрицы жесткости.

Матрица  $K$  обычно имеет очень большой размер:  $Ndof \times Ndof$ ,  $Ndof = 3Nverts$ ,  $Nverts \sim 10^3 \dots 10^4$ , однако она является весьма разреженной, содержащей обычно более 99% нулей. Эффективное хранение этой матрицы возможно в силу следующего наблюдения: элемент  $K_{ij}$  с  $i \neq j$  можно представить как отрезок, соединяющий  $i, j$ -тые узлы, и если этот отрезок не совпадает с ребром одного из тетраэдров в разбиении, соответствующий элемент обращается в нуль. В результате этого,

недиагональные элементы  $K_{ij}$  можно хранить на ребрах разбиения, в то время как диагональные элементы  $K_{ii}$  могут храниться в узлах. Такой способ хранения требует объема памяти  $M_K$  для матрицы  $K$ , а также дополнительного объема  $M_0$  для хранения данных о разбиении (координаты узлов, индексирующие массивы) и необходимых рабочих массивов:

$$M_K = (6N_{\text{verts}} + 9N_{\text{edges}}) \cdot \text{sizeof}(\text{double}) \sim 600\text{Kb} \dots 6\text{Mb},$$

$$M_0 = 18(N_{\text{verts}}) \cdot \text{sizeof}(\text{double}) + (2N_{\text{edges}} + 10N_{\text{tets}}) \cdot \text{sizeof}(\text{long int}) \sim 440\text{Kb} \dots 4.4\text{Mb}.$$

Произведение  $g = Ku$ , необходимое в дальнейших вычислениях, можно найти, используя следующий алгоритм:

$$\text{для каждого узла } i \{ \vec{g}_i = K_{ii} \vec{u}_i \}$$

$$\text{для каждого ребра } ij \{ \vec{g}_i + = K_{ij} \vec{u}_j; \vec{g}_j + = K_{ij}^T \vec{u}_i \}$$

**Замечание:** В узлах хранятся симметричные 3x3 матрицы  $K_{ii}$ ; ребра рассматриваются как *ориентированные* отрезки, в которых хранится матрица  $K_{ij}$ , если  $ij$  совпадает с направлением ребра, и транспонированная матрица  $K_{ij}^T$ , если  $ij$  противоположно этому направлению.

**Решение линейной системы  $Ku = f$**  может находиться, используя либо явные, либо итеративные методы [14]. Явные методы, такие как метод Гаусса, LU-разложение и др. требуют значительного времени для вычислений и большого объема оперативной памяти. В частности, один шаг метода Гаусса в терминах нашей модели хранения данных можно представить как соединение между собой всех узлов, связанных с  $i$ -тым узлом, и устранение  $i$ -того узла. На промежуточных стадиях этого процесса почти все оставшиеся узлы будут связаны между собой, приводя чрезвычайно большому потреблению памяти.

Среди итеративных методов наиболее часто используется метод сопряженных градиентов (СГ). Этот метод, по существу, является последовательной минимизацией энергии вдоль направлений  $\{p_i\}$ , обладающих свойством  $K$ -ортогональности:  $p_i^T K p_j = 0$ ,  $i \neq j$ . Теоретически, для квадратичных энергий, через конечное число шагов  $N_{\text{steps}} = N_{\text{dof}}$  эта последовательность исчерпывает пространство решений, и алгоритм останавливается на точном решении. На практике  $N_{\text{steps}} \ll N_{\text{dof}}$  дает достаточную точность. Обычно используется предварительно обусловленная (preconditioned) версия метода сопряженных градиентов (ПСГ), обладающая лучшей сходимостью [14].

**Сравнение разных схем вычисления.** Описанные методы могут применяться в реальном времени для вычисления решения, одновременно с графической обработкой, или часть этих вычислений может быть

выполнена предварительно. Обе схемы обсуждались в [1] и мы также опробовали обе. Схема вычислений в реальном времени больше подходит для малых моделей. В частности,  $N_{\text{verts}} \sim 200$  могут быть обработаны в реальном времени (20 кадров в секунду),  $N_{\text{verts}} \sim 3000$  требуют порядка 1 сек релаксационного времени для полной визуальной сходимости итераций, в то время как для  $N_{\text{verts}} \sim 10^4$  это время составляет около 1 мин (для машин типа SGI/Onyx2 300MHz MIPS R12000). Метод сходится быстрее для тел, размеры которых во всех трех измерениях приблизительно равны (причины этого обсуждаются в приложении), и для материалов, обладающих  $\lambda \leq \mu$ , по свойствам сходных с керамикой.

В работе [5] использовалась итеративная динамическая схема для моделирования деформаций объектов в рамках линейной и нелинейной теории упругости. В этой схеме вычисляются упругие силы, действующие на узлы модели, и дальнейшая эволюция системы просчитывается в реальном времени с помощью стандартных методов интегрирования по законам Ньютона, моделируя должным образом распределения масс и сил трения. При этом нелинейные члены в упругой силе, отвечающие квадратичному члену в тензоре деформаций (см. выше), значительно увеличивают объем вычислений. В работе [5] для объекта, содержащего 1,400 узлов, была достигнута скорость вычислений 45 итераций в секунду для линейной модели и 8 итераций в секунду для нелинейной модели, на компьютере типа PC Pentium III 500 MHz. В то же время, по нашим данным, для сходимости итераций до относительных погрешностей  $\epsilon \sim 10^{-3}$  (достаточных для визуализации) требуется от десяти итераций до нескольких тысяч итераций, в зависимости от плавности взаимодействия, формы и материальных свойств объекта. Это приводит к релаксационным эффектам той же длительности, как и полученные нами, которые внешне сходны с физическими процессами затухания колебаний, но протекают медленнее, чем в реальности, из-за недостаточной мощности вычислителя.

Другое свойство, которое обычно рассматривается как преимущество вычислений в реальном времени, это возможность выполнять интерактивные изменения системной матрицы  $K$ , включая те, которые вызваны вариацией граничных условий и изменением топологии модели (разрезы). Однако, в недавних работах [4, 15] эти возможности были также реализованы для предвычисленных моделей, используя ускоренный алгоритм обращения матрицы  $K$  при ее локальных изменениях.

В нашей работе мы производим предварительное обращение матрицы  $K$  и представление решений в виде  $u = K^{-1}f$ , в результате получаются следующие характеристики: объект из 10,000 узлов деформируется при скорости графики 85 кадров в секунду на Linux PC

Athlon 1.3 GHz. Метод является квазистационарным, т.е. на каждый кадр находится точное состояние равновесия объекта, отвечающее данному взаимодействию. Основная проблема этого подхода состоит в том, что матрица  $K^{-1}$  больше не является разреженной и требует довольно много памяти для хранения. К счастью, не всю матрицу необходимо хранить, а только те строки, которые отвечают видимым узлам, и только те столбцы, которые отвечают узлам, в которых может быть приложена внешняя сила:

$$\begin{bmatrix} u_S \\ * \end{bmatrix} = \begin{bmatrix} (K^{-1})_S & * \\ * & * \end{bmatrix} \begin{bmatrix} f_S \\ 0 \end{bmatrix}$$

Для большинства приложений достаточно выделить из  $K^{-1}$  блок  $(K^{-1})_S$  размера  $(3N_{\text{vert}}) \times (3N_{\text{vert}})$ , соответствующий поверхностным узлам. Для этой цели мы решаем систему  $Ku = f$  с  $f_i = \delta_{ik}$ , где  $k$  ограничено на поверхностные узлы. Решение вычисляется используя метод ПСГ. Заметим, что внутренние узлы дают вклад в элементы  $K^{-1}$  во время предварительной стадии вычислений, и затем могут быть опущены. Полученные решения  $u_i^{(k)}$  (столбцы  $K^{-1}$ ) пишутся в файл, для  $i \in$  поверхностным узлам. Далее, используя симметрию  $(K^{-1})_S$  и уменьшая точность матричных элементов до float (двойная точность необходима для устойчивости предварительного вычисления, в то время как одинарная точность достаточна для визуализации), мы уменьшаем объем памяти, необходимой для хранения  $(K^{-1})_S$  до приемлемой величины:

$$M_S = 4.5(N_{\text{verts}})^2 \cdot \text{sizeof(float)} \sim 6\text{Mb} \dots 450\text{Mb},$$

для числа поверхностных узлов в интервале  $N_{\text{verts}} \sim 600 \dots 5000$ .

Вычисление произведения  $u_S = (K^{-1})_S f_S$ , которое необходимо выполнять в реальном времени, в общем случае все еще является интенсивным. Дальнейшая оптимизация возможна, если использовать дополнительные предположения о характере силовых распределений. В частности, если сила локализована в нескольких узлах поверхности, можно использовать ускоренный алгоритм умножения [1]. Для силы, локализованной в единственном узле, этот алгоритм сводится к извлечению соответствующего столбца из  $(K^{-1})_S$ . В другом случае сила может быть распределена в широких областях поверхности, но вводится небольшое число типов силовых распределений, которые представляют различные типы взаимодействия (растяжение, сдвиг, кручение и т.п.). В этом случае имеется небольшое число линейных комбинаций столбцов  $(K^{-1})_S$ , которые можно вычислить предварительно, составить из них

новую матрицу  $\Lambda$ , сохранить ее, и затем комбинировать ее столбцы в реальном времени, используя короткое вычисление (в простейшем случае сводящееся к извлечению необходимого столбца из матрицы  $\Lambda$ ). Еще один пример, появляющийся в инженерных приложениях, это деформация упругого тела, определяемая по нескольким контрольным элементам, прикрепленным к границе. Мы рассмотрим эту модель более детально.

### Упругие деформации с помощью контрольных элементов

Рассмотрим граничную задачу  $Au = -Bb$ , где смещение  $b$  на участке границы  $\Gamma_b$  задаются аффинными преобразованиями  $\vec{b} = \vec{T} + M\vec{b}_0$ . Эти преобразования описываются 12 параметрами, а именно, 3-мя переносами  $\vec{T}$  и 3x3 матрицей общих линейных преобразований  $M$

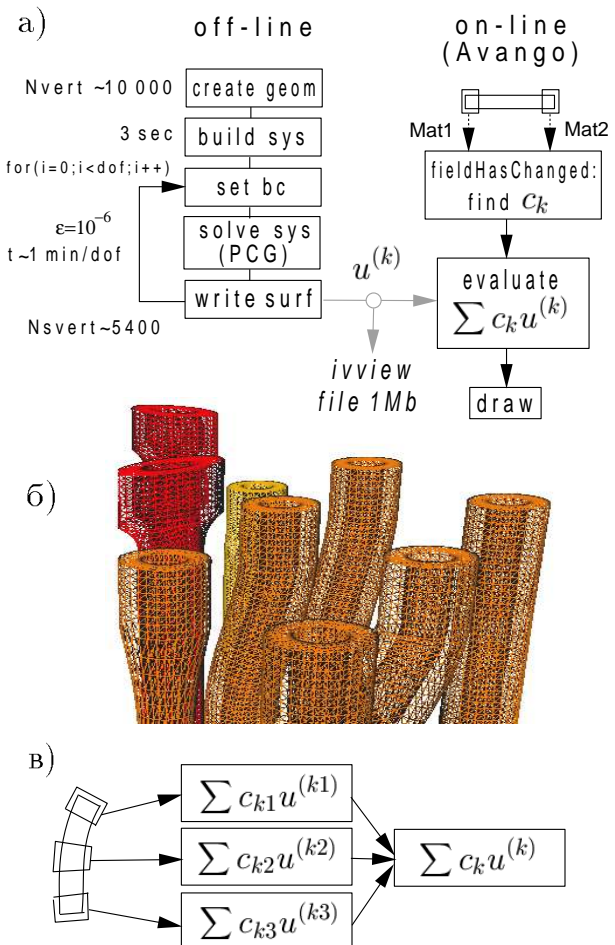
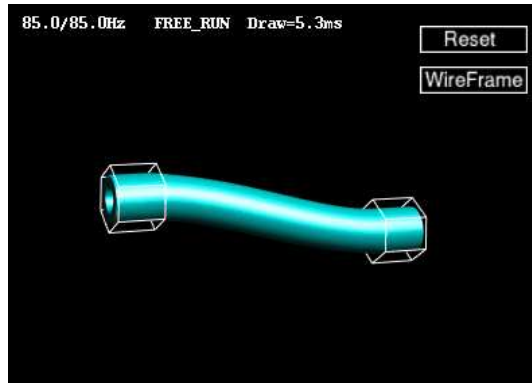


Рис.4. а) схема вычислений; б) базис решения (iv-файл); в) оптимизация вычислений, выполняемых в реальном времени.



**Рис.5.** интерактивно деформируемая модель трубки в демонстрационном приложении Аванго.

(включающей, в частности, 3 вращения), определяя 12-мерное пространство на контрольный элемент. Пусть  $b^{(k)}$ ,  $k = 1..12$  – базис в этом пространстве, соответствующий единице, помещаемой последовательно в ячейки  $\vec{T}$  и  $M$ , в то время как остальные ячейки заполнены нулями. Пусть  $u^{(k)}$  – соответствующие решения системы, т.е.  $u^{(k)} = -A^{-1}Bb^{(k)}$ . Тогда, в силу линейности системы, форма упругого тела для данного взаимодействия  $b = \sum_k c_k b^{(k)}$  будет задаваться выражением  $u = \sum_k c_k u^{(k)}$ . Фактически, нам необходимо вычислить решения для элементарных взаимодействий, соответствующих базисным векторам  $b^{(k)}$ , в предварительном режиме, и затем комбинировать эти решения, чтобы найти форму тела для любого заданного взаимодействия, используя короткое вычисление в реальном времени. Объем памяти на контрольный элемент, требуемый для хранения решений  $u^{(k)}$ , мал:

$$M_u = 36N_{\text{verts}} \cdot \text{sizeof}(\text{float}) \sim 100\text{Kb} \dots 700\text{Kb}.$$

Данная схема вычислений показана на рис.4а. Для модели с числом узлов  $N_{\text{verts}} \sim 10,000$  инициализация системы занимает порядка 3 сек, затем система решается в цикле для различных базисных векторов  $b^{(k)}$ , с относительной погрешностью  $\epsilon = 10^{-6}$ , требуя около 1 мин на решение  $u^{(k)}$ , и около 20 мин на предвычисление системы с двумя контрольными элементами. Время вычисления зависит от формы модели и упругих свойств материала. Полученные решения для поверхностных узлов  $N_{\text{verts}} \sim 5400$  пишутся в файл, который в сжатом виде занимает 1Mb. Мы используем графический формат Open Inventor, чтобы иметь возможность контролировать решения визуально (рис.4б) и чтобы упростить перенос данных в виртуальные окружения. Визуализация выполняется в реальном времени посредством программного модуля, разработанного с помощью системы виртуального окружения Аванго.

**Реализация в Аванго:** Взаимодействие с виртуальными объектами в крупномасштабных системах виртуального окружения выполняется с помощью спе-

циального электромагнитного устройства (трехмерной указки), визуально представленной в виртуальном мире, например, как зеленый луч. Аванго предоставляет удобный интерфейс, называемый *MatrixDragger*, который на нижнем уровне определяет пересечение луча с объектом и копирует матрицу устройства взаимодействия в матрицу объекта, что позволяет с легкостью перемещать виртуальные объекты. В нашем приложении *MatrixDragger* прикреплены к контрольным элементам и их матрицы присоединены к нашему программному модулю посредством полевых связей. Всякий раз, когда матрицы изменяются, активируется метод *fieldHasChanged*, который извлекает коэффициенты  $c_k$  из элементов матриц. Вычисление суммы  $\sum_k c_k u^{(k)}$  откладывается до вызова метода *evaluate*, который активируется только один раз на кадр, если какое-либо поле изменилось. Различные методы могут использоваться для ускорения этого вычисления. В частности, если используется только одно устройство взаимодействия, тогда только один контрольный элемент может переместиться за один акт взаимодействия, при этом изменяются только соответствующие ему коэффициенты  $c_k$ , и только соответствующая часть суммы должна быть перевычислена, см рис.4в. Дополнительные возможности для ускорения предоставляют использование арифметики указателей, помещение данных в область памяти быстрого доступа (cache) и параллелизация вычислений. Описанная модель интерактивно деформируется в виртуальном окружении при скорости графики 20 кадров в секунду, при этом все вычисления, необходимые для деформации, требуют около 30% времени одного кадра, для компьютеров типа SGI/Onyx2 300MHz MIPS R12000. Характеристики исполнения на Linux PC Athlon 1.3MHz лучше: 85 кадров в секунду.

## Заключение

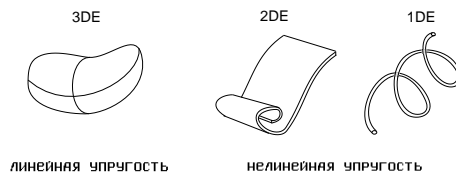
В данной работе мы описали подход, применимый для моделирования упругих деформаций объектов в реальном времени. Этот подход использует метод конечных элементов для представления уравнений теории упругости. Большая часть интенсивных вычислений выполняется заранее. Полученные данные, сохраненные в файле, позволяют ускорить в тысячи раз оставшуюся часть вычислений, которые выполняются в реальном времени. Это дает возможность моделировать упругие деформации объектов высокой сложности в виртуальных окружениях.



## Список литературы

- [1] M. Bro-Nielsen, S. Cotin, Real-Time Volumetric Deformable Models for Surgery Simulation using Finite Elements and Condensation, Eurographics'96, Vol.15 (1996) No.3, C57-C66.
- [2] Л.Д.Ландау, Е.М.Лифшиц, Теория упругости, Теоретическая физика, Т.7, Москва, Наука, 1970.
- [3] Gibson S.F., Mirtich B., A survey of deformable models in computer graphics, Technical Report TR-97-18, Mitsubishi Electric Research Laboratories, Cambridge, MA, November 1997.
- [4] James D., Pai D., Artdefo Accurate Real Time Deformable Objects, Computer Graphics, vol.33, pp.65-72, 1999.
- [5] Picinbono G., Delingette H., Nicholas H.-A., Non-Linear Anisotropic Elasticity for Real-Time Surgery Simulation, INRIA Research Report 4028, October 2000.
- [6] Y.Zhuang, J.F.Canny, Real-time global deformations, In Algorithmic and Computational Robotics, pp.97-107, Natick MA, 2001. A.K.Peters.
- [7] Krüger W., Bohn C., Fröhlich B., Schüth H., Strauss W., Wesche G., The Responsive Workbench: A Virtual Work Environment, IEEE Computer, pp. 12-15, 1994.
- [8] Eckel G., Göbel M., Hasenbrink F., Heiden W., Lechner U., Tramberend H., Wesche G., Wind J. Benches and Caves. In: Bullinger H.J., Riedel O. (eds.) Proc. 1st Int. Immersive Projection Technology Workshop. Springer-Verlag, London, 1997.
- [9] Cruz-Neira C. Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE. Computer Graphics Proc., Annual Conference Series, 1993, pp.135-142.
- [10] Tramberend H., Avocado: A Distributed Virtual Reality Framework, *Proc. of the IEEE Virtual Reality, 1999.*
- [11] R. Kent Dybvig. The Scheme programming language: ANSI Scheme. P T R Prentice-Hall, Englewood Cliffs, NJ 07632, USA, Edition 2, 1996.
- [12] В.С.Владимиров, Уравнения математической физики, Издание 5, Москва, Наука, 1988.
- [13] O. C. Zienkiewicz, R. L. Taylor, The Finite Element Method, Vol.1, Edition 5, Butterworth-Heinemann, Oxford 2000.
- [14] A. O. Frank, I. A. Twombly, J. D. Smith, Finite Element Methods for real-time Haptic Feedback of Soft-Tissue Models in Virtual Reality Simulators, Proceedings of VR2001.
- [15] U. Meier et al, Real-Time Simulation of Minimally-Invasive Surgery with Cutting Based on Boundary Element Methods, Lecture Notes in Computer Science (2001) V.2208, p.1263.
- [16] Hirota, G., Fisher, S. and Lin, M., Simulation of Non-penetrating Elastic Bodies Using Distance Fields. UNC Technical Report TR00-018, 2000, <http://www.cs.unc.edu/~sfisher/Research/research.html>

## Приложение: За пределами теории малых деформаций



**Рис.6.** Различные типы упругих объектов.

Имеются два типа нелинейностей, которые могут усложнить моделирование: материальные и геометрические. Материальные нелинейности, вызванные нарушением линейного соотношения между тензорами напряжения и деформации (закон Гука), обычно, по крайней мере в инженерных приложениях, возникают только в экстремальных условиях сильных напряжений, когда материал начинает терять свои упругие свойства. Геометрические нелинейности проявляются при больших смещениях точек тела друг относительно друга и связаны с квадратичным членом в тензоре деформаций, см. раздел 3. Для тел, размеры которых примерно одинаковы во всех трех измерениях и которые мы будем называть 3DE-телами, см. рис.6, геометрические нелинейности обычно пренебрежимо малы, поскольку большие относительные смещения для этого типа тел могут возникать только под действием очень больших напряжений. В противоположность этому, при деформациях длинных стержней (1DE-тел) и цилиндрическом изгибе тонких пластин (2DE-тел) большие смещения возможны без создания большого напряжения. Существенная часть теории упругости посвящена этим специальным случаям.

Большинство разработанных в настоящее время численных методов, пригодных для моделирования деформаций упругих объектов в реальном времени, основаны на линейной теории малых деформаций. Итеративные методы обладают худшей сходимостью в случае 1DE или 2DE тел, поскольку для таких тел существуют направления, вдоль которых энергия изменяется намного медленнее, чем вдоль других. Именно эти направления, отвечающие большим смещениям без создания большого напряжения, являются предпочтительными при деформации тел, в то время как шаг итерации, определяемый другими направлениями, оказывается малым, что замедляет сходимость метода. Также, для 1DE или 2DE объектов решения линейной теории часто проявляют нефизические свойства. Например, если один конец длинного стержня зафиксирован, в то время как другой вращается вокруг своей оси, выполняя один полный оборот, граничное условие  $b$  возвращается к недеформированному состоянию, при этом решение теории малых деформаций  $u = -A^{-1}Bb$  также непрерывно возвращается в недеформированное состояние,

сложным образом проникая сквозь себя. Эта проблема вызвана опущенным квадратичным членом в определении тензора деформаций, который не является малым для данных деформаций. Этот член также отвечает за полную вращательную инвариантность в теории, так что энергия линейной теории оказывается инвариантной только относительно инфинитезимальных, но не относительно полных вращений. Неправильное поведение линейной системы по отношению ко вращениям объекта является индикатором проблем, связанных с нелинейными эффектами [5].

Ниже мы приводим несколько подходов, которые способны учесть геометрически нелинейные эффекты для моделирования упругих деформаций. Мы подразделяем подходы на те, которые должны целиком использоваться в реальном времени, те, которые основаны на предварительных вычислениях, и смешанные.

#### Методы, работающие в реальном времени:

1а) использование тензора деформаций с квадратичным членом, как было сделано в [5];

1б) введение системы локальных координат [2], в которых квадратичные члены в тензоре деформаций малы и ими можно пренебречь.

Методы, использующие предварительные вычисления (применимые для не очень больших углов вращения, например, при более точном моделировании 3DE-тел):

2а) в схеме, использующей контрольные элементы, выделение средней матрицы вращения всех контрольных элементов и перенесение ее на вращение модели как целого. Это уменьшит углы вращения контрольных элементов по отношению к среднему положению, значительно уменьшая нелинейные члены (пропорциональные квадратам углов).

2б) решение нелинейной системы, используя теорию возмущений по нелинейному члену и предвычисляя достаточное количество коэффициентов разложения.

#### Смешанные методы:

3а) подразбиение 1DE и 2DE тел на множество 3DE частей, для которых применима теория малых деформаций и поведение которых может быть описано в терминах предвычисленных функций Грина, состыковка частей вдоль границ, выражение энергии системы в терминах меньшего набора граничных координат и ее минимизация в реальном времени, используя подходящий нелинейный минимизатор [16].

3б) рассматривая нелинейное поведение длинных стержней и тонких пластин, можно использовать методы, основанные на частично аналитическом разрешении дифференциальных уравнений в частных производных, проведенном в нелинейной теории упругости [2].