3 раза, а с температурой более 11°С – сократилась в 20 раз. Расчётами установлено, что за сутки средняя температура поверхности Южного Каспия снизилась с 11,10°С до 9,58°С.

На суше, как это было показано, активизация геотермических процессов приводит к увеличению температуры. Нами были изучены данные гидрометстанции «Chagy» в Таджикистане, расположенной к востоку от залива Кара Богаз Гол (№ 38511 40°47'N 55°20'E). По данным этой станции в дату охлаждения поверхности Каспия ветер отсутствовал, а максимальная температура приземного воздуха увеличилась с 16,6°С до 26,1°С. На другой день минимальная (ночная) температура выросла с 6,8°С до 10,4°С. Рост температуры ночью при отсутствии адвекции свидетельствует об активизации геотермических потоков тепла. Данные наземных наблюдений полностью подтверждают факт активизации геотермических процессов, которые привели к резкому понижению температуры поверхности Южного Каспия в течении суток в апреле 2012г.

Выводы. На основе спутникового мониторинга в ИК диапазоне на конкретных примерах показано, что геотермические потоки тепла следует изучать как важный климатообразующий фактор. Максимальный вклад эти процессы оказывают в зимний сезон в условиях низких температур и отрицательного радиационного баланса. Спутниковые технологии позволяют оценивать и эффект антиподального влияния на климат Земли проявления геотермических потоков в районах континентов и акваторий. Показано, что под влиянием тектонических процессов потоки геотермического тепла могут за короткий срок изменить температуру акватории и суши на несколько градусов, что особенно важно в зимний сезон в условиях минимальной амплитуды суточного хода температуры.

Литература

1. Гладких І.І. Формування погодних умов в морських та прибережних районах.Гладких І.І., Капочкін Б.Б, Лісоводський В.В.,, Одеса, Астропринт, 2007 р., 152 с.

Надійшла до редакції
21.1.2013 р.

УДК 621

С.И. ВЯТКИН
Институт автоматики и электрометрии СО РАН, Россия
А.Н. РОМАНЮК, М.П. ПОДДУБЕЦКАЯ
Винницкий национальный технический университет, Украина

АНИМАЦИЯ ТРЕХМЕРНЫХ ОБЪЕКТОВ

В данной работе рассматриваются методы инверсной кинематики, а также рассматривается модифицированный итерационно-численный метод вычисления новых положений цепей инверсной кинематики для каждого кадра.

Ключевые слова: инверсная кинематика, итерационно-численный метод, DOF, BFGS-алгоритм.

In this article the methods of inverse kinematics are discusses. A modified iteration-numerical method for calculating the new provisions of inverse kinematics chains for each frame is developed.

Keywords: inverse kinematics, Iterative-numerical method, DOF, BFGS-algorithm.

Введение

В компьютерной графике реального времени при анимации персонажей трехмерной сцены, возникает задача инверсной кинематики (inverse kinematics).

Инверсная кинематика – широко используемый метод анимации моделей. Он используется при создании движения, как в простейших, так и в сложных иерархических моделях. Этот инструмент призван существенно облегчить труд дизайнера. При использовании инверсной кинематики не требуется анимировать каждый отдельный узел иерархически связанной цепи, чтобы получить ее движение в целом. Для этого стоит лишь задать необходимые параметры, а расчет движения цепи с учётом связности будет выполняться автоматически на каждом кадре.

Цепь инверсной кинематики – это иерархия, где взаимодействие между объектами осуществляется «снизу вверх», от дочернего объекта к родительскому. Для примера возьмем классическую модель человека - бипод. Если перемещать в пространстве туловище (родительский объект), вместе с ним, как жестко закрепленные, будут перемещаться руки, ноги и голова (дочерние объекты) – это цепь прямой кинематики, где воздействие на родительский объект влияет на его дочерние объекты. Если же в этом биподе реализована цепь обратной кинематики, то перемещение в пространстве дочернего объекта, например, кисти руки, приведёт к перемещению родительских объектов: предплечья, плеча, туловища.

Разработка новых и усовершенствование существующих методов вычисления новых положений цепей инверсной кинематики для каждого кадра является очень важной задачей компютерной графики:

данные процедуры работают в режиме реального времени и требуют высокой производительности разрабатываемых методов.

Анализ существующих методов

В современных 3D редакторах задача инверсной кинематики реализована довольно успешно, что позволило более гибко осуществлять анимацию в 3D сцене, делая ее реалистичней. Существуют следующие методы инверсной кинематики: алгебраические, геометрические, итерационные, численные и смешанные. Для алгебраического решения [1], [2] задачи инверсной кинематики, требуется решить уравнение для 2N независимых переменных (N объектов, 2N степеней свободы). Алгебраический метод дает решения для манипуляторов не более чем с шестью степенями свободы. В геометрическом методе решение в аналитическом виде получается с использованием геометрии цепочки. В работе [3] использованы координатные методы, описанные в [4], для получения аналитического решения манипулятора с семью степенями свободы. Этот метод применим к любому манипулятору с заранее известной геометрией. Недостатком метода является то, что для его функционирования необходимо знать аналитическое решение для первых трех объектов цепочки, а также геометрический подход применим только для заранее известной геометрии манипулятора [5]. Суть итерационных методов заключается в том, что решение достигается в ходе итерационного приближения. Основные проблемы, возникающие при этом – это сходимость таких методов. Большая часть итерационных методов основаны на алгоритмах численной минимизации нелинейной функции, но есть и алгоритмы, где используется геометрический подход. Численные алгоритмы является универсальными в плане способности решить задачу инверсной кинематики для любого количества степеней свободы. В задаче инверсной кинематики также важным моментом является переход от задачи поиска решения без ограничений на переменные (т.е. на углы вращения) к задаче поиска решения с ограничениями на переменные. Ограничения степеней свободы существенны, так как моделируемые объекты, для которых и появилась задача инверсной кинематики, обычно в природе имеют физические ограничения на возможность вращения. Например, алгебраический метод эти ограничения не учитывает, и реализовать ограничения, используя алгебраическое решение затруднительно. Итерационный метод, в силу итерационного приближения к решению, естественным образом дает решение с ограничениями на переменные [6].

В данной работе рассматривается модифицированный итерационно-численный метод вычисления новых положений цепей инверсной кинематики для каждого кадра, в режиме реального времени, откуда и появляются жесткие требования к производительности данного метода.

Постановка задачи

Для понимания структуры и объектной модели инверсной кинематики необходимо ввести еще несколько понятий:

Опорная точка (pivot) – точка, определяющая координаты и ориентацию объекта в пространстве. Это точка, вокруг которой будет осуществляться вращение объекта (например, вращая куб, мы получим разные положения в пространстве, если будем вращать его относительно геометрического центра и относительно угла).

Сочленение (joint) – проводя аналогию с конечностью человека, можно сказать, что joint – этот сустав. Сочленения могут быть двух типов движения – вращательное (rotational), и поступательное (sliding).

Терминатор (terminator) – сочленение, на котором прекращаются вычисления движения цепи инверсной кинематики. Терминатором можно назначить любое сочленение в цепи, если вы не хотите, чтобы сочленения, расположенные выше терминатора по иерархии, участвовали в цепи инверсной кинематики. Терминаторы обычно используются в сложных иерархиях, в которых реализовано несколько цепей инверсной кинематики.

Конечный эффектор (end effector) – Цепь кинематики – это одиночная ветвь иерархии, используемая для анимации методом инверсной кинематики. Цепь начинается с выбранного дочернего объекта и идет вверх по восходящей, пока не достигает базы цепи. Выбранный дочерний объект в цепи кинематики и называется конечным эффектором.

Цель (target) – объект, за которым «тянется» вся цепочка, начиная с конечного эффектора. Обычно для каждой цепочки инверсной кинематики задается такой объект, который и определяет ее движение.

Детали реализации виртуального пространства могут отличаться в разных продуктах, тем не менее, суть везде одна и та же. Каждый объект имеет свою локальную систему координат и характеризуется шестью степенями свободы: углы поворота вокруг осей координат и сами координаты. Все это можно описать одной матрицей размером 3x4 элемента: первые три столбца характеризуют ориентацию (поворот) объекта, четвертый столбец – положение объекта. Вложив, локальные системы координат объектов друг в друга, т.е. связав их в иерархическую цепочку, получаем возможность вычислять глобальные координаты объектов (т.е. их расположение в некоторой мировой системе координат первого объекта в цепочке).

Заметим, что каждая локальная матрица объекта цепочки зависит от трех углов вращения вокруг каждой оси, т.е. имеет три степени свободы (DOF – degree of freedom), однако достаточно задействовать два угла, т.е. две DOF, чтобы уже можно было произвольно вращать объект с этой матрицей. Таким образом, матрица зависит от 2N DOF, т.е. имеем 2N переменных. В задаче инверсной кинематики необходимо иметь

ограничения на углы вращения локальных матриц объектов цепочки, т.е. надо задавать диапазон изменения каждой переменной. Такие ограничения в итоге играют важную роль при задании реалистичности движения получаемых цепочек. Примером может быть анимация двуногого персонажа, часто используемая в сценах. Ноги такого персонажа, его колени, очевидно, не должны сгибаться при ходьбе более чем на 180 градусов, что является реалистичным физическим ограничением моделируемого персонажа. Таким образом, каждый шарнир в цепочке объектов имеет как минимум шесть ограничительных параметров: выбор активных осей вращения и ограничения на углы каждой оси вращения (нижняя, верхняя границы). Все это входит в параметры цепочки инверсной кинематики объектов.

Предлагается модифицированный итерационно-численный метод вычисления новых положений цепей инверсной кинематики, с высокой точностью решения задачи и эффективностью работы для режима реального времени. При этом учитывалась универсальность метода, как с точки зрения количества степеней свободы, так и возможности реализации всего спектра параметров, требуемых для настройки цепочки инверсной кинематики.

Описание метода

Необходимо найти точку локального минимума X непрерывно дифференцируемой функции f(x): $R^n \to R$, начиная из X_0, где X, $X_0 \in R^n$. Используется алгоритм линейного поиска, конечно-разностные градиенты, и аппроксимации Гессиана.

Алгоритм состоит из следующих основных шагов.

На **k**-той итерации:

1. Вычислить $f(x)$, если этого не было сделано раньше, и решить - остановиться или продолжать.

2. Вычислить градиент функции - $g(x)$ и матрицу Гесса - H_k, (причем $H_0 = E$) используя формулу *BFGS* ($H_{k+1} = H_k - \frac{H_k * s * s^T * H_k}{s^T * H_k * s} + \frac{y * y^T}{y^T * s}$, где $s = x - x_0$, $y = g(x) - g(x_0)$)

3. Запустить алгоритм *линейного поиска* вдоль шага $d = -H_k * g(x)$ и найти λ, при котором $f(x)$ достигает минимального значения на векторе $x_0 + \lambda * d$ (линейный поиск)

4. Присвоить $x = x_0 + \lambda * d$

5. Вернуться к шагу 1.

Линейный поиск

Для заданных $g^p(\nabla f(x_0)) * p < 0$ и $\alpha < 1/2$ (используется $\alpha = 10^{-4}$), используя линейный поиск с дроблением шага, найти точку $x = x_0 + \lambda * p$, $\alpha \in (0,1]$, такую, что $f(x) \le f(x_0) + \alpha * \lambda * g^p * p$. Пусть $x_+(\lambda) = x_0 + \lambda * p$.

1. Положить $\lambda = 1$

2. Установить, удовлетворительна ли точка $x_+(\lambda)$. Если да, то завершить работу.

3. Установить, не слишком ли мала длина шага. Если да, то завершить работу.

4. Уменьшить λ с помощью множителя, лежащего между 0.1 и 0.5 , следующим образом: (обозначим через $f'_p(x_0)$ и обозначим производную $f(x)$ в точке x_c по направлению ***p***)

- При первом дроблении: выбрать новое значение λ таким, чтобы $x_+(\lambda)$ была точкой минимума квадратичной функции одной переменной, интерполирующей $f_p(x_0)$, $f'_p(x_0)$, $f(x_0 + p)$. Но ограничить снизу новое λ числом 0.1. (Гарантируется, что новое $\lambda < 1/(2*(1-\alpha)) = 1/1.9998$)

- При всех последующих дроблении: выбрать новое λ таким, чтобы $x_+(\lambda)$ была точкой локального минимума кубической функции одной переменной, интерполирующей $f(x_0)$, $f'_p(x_0)$, $f(x_+(\lambda))$, $f(x_+(\lambda_{prev}))$, но ограничить новое λ отрезком $[0.1*(старое\ \lambda),\ 0.5*(старое\ \lambda)]$.

5. Вернуться к шагу 2.

Представленный выше алгоритм является исходным видом BFGS-алгоритма, его простой формой. Алгоритм выполняет итерации, проверяя критерии остановки и в случае выполнения одного из критериев, завершает работу. В настоящее время приведенный алгоритм уже существенно модифицирован, в частности

разработаны формулы более быстрого пересчета матрицы Гесса для случаев большого количества переменных, а так же модифицирован линейный поиск, где предлагается использовать дополнительные условия на производную по направлению, чтобы ускорить сходимость. Необходимость использования этих изменений для случая функции инверсной кинематики требует дополнительной проверки.

В данной работе предлагается применить решение задачи инверсной кинематики для режима реального времени. Ключевым моментом для инверсной кинематики является быстродействие и точность решения. Теоретическая основа алгоритма не дает гарантий глобальной сходимости, так как BFGS относится к алгоритмам поиска локального экстремума, но не глобального. Условием глобальной сходимости является монотонность взятой функции на всей области определения. В случае инверсной кинематики, вид функции неочевиден, особенно при большом количестве переменных. Вторая проблема – это недостаточная точность. В отличии от реального мира, виртуальный мир имеет конечную точность, что неизбежно ведет к погрешностям вычисления значений функций так и их производных.

Частично эта проблема была снята выделением алгоритма в независимый модуль. Реализация алгоритма была выполнена в виде динамической библиотеки. Далее, в отдельном приложении для теста брались специальные функции: расширенная функция Розенброка, расширенная обобщенная функция Пауэла, функция типа спиралевидного желоба и функция Вуда. Алгоритм успешно справляется с минимизацией функции Вуда, однако это лишь отдельно взятая функция и нет гарантий, что на другой тестовой функции алгоритм найдет решение. В связи с этим, единственный путь, хоть сколько-нибудь гарантирующий корректную работу алгоритма на произвольной функции – это испытание его на как можно большем множестве нелинейных функций с глобальным минимумом. Кроме успешной работы алгоритма, необходимо качественно реализовать вычисление самой функции инверсной кинематики и ее производных. В отличии от тестовых функций, имеющих аналитический вид, функция инверсной кинематики такого вида не имеет, что существенно уменьшает точность вычисления производной по конечным разностям. Таким образом, не менее существенной проблемой, чем реализация алгоритма, стала проблема как можно более точного вычисления производных, в условиях значительной потери точности при вычислении функции. В данном методе был реализован линейный поиск с условием на производную и сделан переход к многопроходному алгоритму. В работе [1] предлагался однопроходный вариант алгоритма, где он завершает свою работу по одному из вышеуказанных терминальных кодов. На практике это не дает желаемого результата, так как алгоритм завершает работу, не достигнув заданной точности. Анализ тестов показал, что так происходит потому, что обновляемая на каждом шаге матрица Гесса, которая призвана сохранять в себе информацию о предыдущей кривизне функции, ухудшает выбор направления спуска. Это вызвано тем, что всегда есть масса погрешностей в вычислении производных функций, которые после нескольких итераций могут ухудшить приближение матрицы Гесса. Поэтому было принято решение периодически приравнивать единице матрицу Гесса, что, по сути, равносильно новому запуску алгоритма. Так родилась идея многопроходного алгоритма. Однако заметим, что при принятой идеологии, мы оставляем то же число максимального количества итераций, а это значит, что мы не запускам алгоритм заново, увеличивая общее время работы, а делим данное нам время на несколько «небольших», но более эффективных проходов алгоритма. Осталось определить, когда стоит делать такое обновление: пересчет градиента, установка матрицы Гесса в единицу, перезапуск алгоритма с текущего вектора. Анализ тестов показал, что хороший результат получается, если проводить такое обновление в случае, когда значения функции, получаемые на нескольких последовательных итераций, слабо отличаются друг от друга. На практике, такая стратегия привела к усложнению программного модуля и увеличению количества аварийных ситуаций, которые необходимо обрабатывать для устойчивой работы алгоритма. В итоге потребовалась возможность анализировать результат каждого прохода алгоритма завершаемый каким-либо терминальным кодом. После каждого прохода алгоритма требуется принимать решение, что делать дальше. Поэтому, в качестве общей схемы программного модуля была выбрана схема из двух компонент: сам алгоритм и анализатор его работы. Несмотря на то, что алгоритм является главной частью программы, как показывает практика, от анализатора может зависеть до 40% затрачиваемых ресурсов. В частности, на отдельных примерах минимизации функции многих переменных, удавалось достигать 30% повышения производительности. Сам анализатор по сути своей не является самостоятельным алгоритмом. Он содержит эмпирически определяемую для конкретно взятой функции информацию о том, как поступать при выработке тех или иных кодов завершения алгоритма BFGS.

Заключение

В работе представлен модифицированный высокопроизводительный итерационно-численный метод вычисления новых проложений цепей инверсной кинематики для каждого кадра, в режиме реального времени. Тестирование предложенного метода показало улучшение скорости сходимости, которое в частных случаях достигало 30%. Метод надежно работает на множестве тестовых примеров.

Литература

1. Kwan W. Chin, B.R. von Konsky, and A. Marriott. Closed-form and generalized inverse kinematics solutions for the analysis of human motion // Engineering in Medicine and Biology society, 1997. Proceedings of

the 19th Annual International Conference of the IEEE, Vol. 5 (1997), pp. 1911-1914.

2. Paolo Baerlocher, Inverse Kinematics Techniques for the interactive posture control of articulated figures. These No 2383, 2001. (156).

3. Lee, G.C.S. (1982). Robot arm kinematics, dynamics and control. Computer, 15(12), 2-79.

4. Anon. EA Mathematical Formulae and Statistical Tables Book. Secondary Education Authority. 1992.

5. Craig, J.J. Introduction to Robotics: Mechanics and Control. Addison-Wesley. 1989.

6. Richard H. Byrd, P. Lu, J.Nocedal and C. Zhu. A limited memory algorithm for bound constrained optimization, SIAM Journal on Scientific Programming Computing, 16, 5 pp. 1190-1208. 1994.

Надійшла до редакції
13.2.2013 р.

УДК 65.011.56:004(043.2)

В.В. ЛЮБЧЕНКО, В.П. КВАСНІКОВ
Національний авіаційний університет

АНАЛІЗ АЛГОРИТМІВ РОЗРАХУНКУ НОРМАЛІЗУЮЧОЇ КОНСТАНТИ

В роботі проведений аналіз найбільш відомих алгоритмів знаходження нормалізуючої константи. Застосування алгоритму у формі дерева дозволить скоротити затрати часу і витрати пам'яті для знаходження вихідних даних в комп'ютерних мережах.

Ключові слова: алгоритм, мережа.

In this paper, an analysis of the best known algorithms for finding the normalizing constants. Application of the algorithm in the form of trees will reduce the cost of time and memory costs for finding initial data in computer networks.

Keywords: algorithm, network.

Вступ

У зв'язку із зростанням вартості проектування і самої проектованої системи пред'являють підвищені вимоги до якості проектних рішень. А особливо: до точності визначення завантаженості каналів, часу затримки пакетів, обсягів пам'яті буферів, аналізу продуктивності локальних мереж, розрахунок втрат і загрузки цифрових ліній зв'язку.

Стохастичний характер надходження даних і детермінована обробка їх у вузлах комутації визначає можливість використання моделей теорії систем масового обслуговування (СМО) для аналізу і проектування комп'ютерних мереж. Але дослідження комп'ютерних мереж за допомогою простих однофазових, двохфазових моделей масового обслуговування, дозволяє отримати лише якісне представлення о характері проходження інформаційних процесів. Це зумовлено складною взаємодією пристроїв і процесів комп'ютерної системи. Для вирішення цих питань проведемо аналіз і визначимо найбільш оптимальний алгоритм визначення нормалізуючої константи.

Аналіз останніх досліджень та публікацій. Алгоритми розрахунку нормалізуючої константи розглянуто в багатьох книгах та публікаціях. Це наприклад Скрипко Е.В., Гришин О.М., Вишневський В.М.[1] , Simon S. [2] який розробив алгоритм у формі дерева, та інші.

Ціль роботи. Провести аналіз основних алгоритмів розрахунку нормалізуючої константи.

Результати дослідження

Розглянемо основні алгоритми розрахунку нормалізуючої константи:

- *рекурентний метод Бузена (алгоритм згортки).* Головним недоліком є те, що знайдені нормалізуючі константи не є однозначними, так як вони залежать від конкретного вибору величини e_j, визначених із системи рівнянь $e_j = \sum_{i=1}^{M} e_f P_{ij}, j = \overline{1, M}$, де *P*- загрузка мережі, *M* - центри.

- *метод середніх значень.* Дозволяє використовувати евристичні рішення для розв'язку мереж масового обслуговування (МО) великої розмірності із випадковим розподіленням часу обслуговування в центрах. Але нажаль алгоритм потребує $2RM - R$ операцій додавання і $2RM + R$ операцій множення (ділення) на крок основного циклу. При цьому для загальної кількості кроків алгоритму $\prod_{r=1}^{R} N_r$ необхідно оперативної пам'яті порядку $M \prod_{r=1}^{R} N_r$,де *R* - класи повідомлень, *N* - повідомлення в мережі, що приблизно дорівнює затратам пам'яті для рекурентного методу Бузена.

- *алгоритм згортки у формі дерева(або просто алгоритм дерева).* Алгоритм дерева використовує інформацію про маршрут повідомлень кожного класу. Використання інформації значно скорочує затрати часу і пам'яті при розрахунку мереж МО з великим числом центрів обслуговування і класів повідомлень, в якому повідомлення кожного класу в середньому відвідують лише невелику частку центрів. Даний алгоритм дозволяє групувати в певних центрах обробки повідомлення одного класу, що скорочує час на обробку цих повідомлень.

Так як розрахунок нормалізуючої константи нам буде потрібний для комп'ютерних мереж, тож