на правах рукописи

Зырянов Александр Владимирович

ПРОГРАММНЫЙ КОМПЛЕКС ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА НА БАЗЕ МАНИПУЛЯЦИОННЫХ УСТРОЙСТВ ВВОДА

05.13.18 – математическое моделирование, численные методы и комплексы программ

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата физико-математических наук

Екатеринбург — 2010

Работа выполнена в Государственном образовательном учреждении высшего профессионального образования «Уральский государственный университет им. А.М. Горького» на кафедре информатики и процессов управления.

Научный руководитель:	кандидат технических наук, доцент Авербух Владимир Лазаревич
Официальные оппоненты:	доктор физико-математических наук, профессор Сесекин Александр Николаевич
	доктор физико-математических наук, профессор Мазуров Владимир Данилович
Ведущая организация:	Государственное образовательное учреждение высшего профессионального образования "Южно-уральский государственный университет"

Защита диссертации состоится 20 октября 2010 года в 15 часов на заседании диссертационного совета Д 212.286.10 по защите докторских и кандидатских диссертаций при ГОУ ВПО «Уральский государственный университет им. А.М. Горького» по адресу: 620000, г. Екатеринбург, пр. Ленина, 51, комн. 248.

С диссертацией можно ознакомиться в научной библиотеке ГОУ ВПО "Уральский государственный университет им. А.М. Горького".

Автореферат разослан 15 сентября 2010 года.

Ученый секретарь диссертационного совета,
доктор физико-математических наук,
профессор В.Г. Пименов

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы. Вычислительные возможности компьютеров стремительно растут, а вместе с ними увеличивается и объём вычисляемых данных. Объём результатов процесса моделирования, размеры сложных связанных структур (например, графов) настолько велики, что данные становится практически невозможно отобразить на плоскости. Однако трёхмерное отображение порождает проблему пользовательского ввода, ведь практически всегда пользователю необходимо не просто взглянуть на сгенерированную картинку, но и повлиять на неё (хотя бы переместиться внутри сцены).

При использовании обычных экранов, пользователь может работать при помощи привычных устройств ввода (клавиатура, мышь). Однако при усложнении моделей возникает необходимость в расширении возможностей визуализации, а значит в использовании сред виртуальной реальности и «больших» экранов (т.е. экранов, диагональ которых измеряется метрами, а количество пикселей – десятками миллионов). Для таких средств отображения привычные средства ввода уже не подходят (к примеру, потому, что они привязывают пользователя к рабочему месту, не позволяя подойти к той или иной части экрана).

За последние десять лет было создано (и продолжают создаваться) большое количество новых средств ввода. Исследования и разработки активно ведутся в таких научных центрах, как Массачусетский технологический институт, Мэрилендский университет, Оксфордский университет, университет города Осака, Стэнфордский университет, Торонтский университет, а также исследовательские центры таких компаний как Apple, Microsoft, Nintendo, Sony. В России подобные работы выполняются, к примеру, в МГТУ им. Н.Э. Баумана.

Новые средства ввода (и связанные с ними новые пользовательские интерфейсы) гораздо лучше подходят для взаимодействия с «большими» экранами, однако и они не лишены тех или иных недостатков. Одни решения являются громоздкими, дорогими, требующими большого количества времени на установку и настройку. Другие решения требуют серьёзных модификаций (вплоть до замены средства ввода) при расширении набора взаимодействий, и, как следствие, требуют повторного обучения пользователей. Искусственность некоторых средств ввода заставляет пользователя концентрироваться не на взаи-

модействии при помощи интерфейса, а на взаимодействии с самим интерфейсом.

В силу сказанного выше проблема манипулирования объектами в трёхмерных визуальных средах по-прежнему является актуальной и не до конца решённой задачей. Необходимо обеспечить естественный, надежный и точный интерфейс, что проще всего сделать при помощи интерфейса, основанного на жестах.

Отметим также сложность установки и настройки многих трёхмерных интерфейсов, связанную с использованием нескольких оптических камер и, как следствие, с необходимостью калибровки. Существующие методы калибровки требуют больших усилий со стороны пользователей, больших вычислительных ресурсов, они могут приводить к ошибкам в силу вычислительной неустойчивости самой задачи калибровки. В настоящей работе приводится метод калибровки оптических камер, способный работать в фоновом режиме (т.е. не требующий от пользователя выполнения каких-либо специальных действий) в реальном времени, и доказывается, что при использовании данного метода задача калибровки является вычислительно устойчивой.

Цель работы. Целью работы является разработка легко расширяемого пользовательского интерфейса для современных средств графического вывода, в том числе «больших» экранов и виртуальных сред. Манипуляции с устройством ввода должны соответствовать воздействию на виртуальный объект. Разрабатываемый интерфейс должен быть легко и быстро настраиваемым. Необходимо, чтобы обеспечивающие работу пользовательского интерфейса алгоритмы работали в реальном времени вне зависимости от характера входных данных, обеспечивая при этом требуемую точность.

Методика исследования. При создании интерфейса использовались методы распознавания жестов. При разработке алгоритмов использовались элементы теории устойчивости, линейной алгебры и пространственной геометрии.

Научная новизна.

1. Разработана новая методика определения положения объекта в пространстве, на основе распознавания в кадре источника света, закреплённого на объекте. Для эффективной работы данного метода достаточно использовать простые аппаратные средства (веб-камера и фонарик в качестве источника света).

2. Создан алгоритм распознавания трёхмерных манипулятивных жестов в реальном времени, способный выделять жест в непрерывном пользовательском вводе, т.е. не требующий обозначения начала и конца жеста. Доказано, что алгоритм распознавания жестов обладает наименьшей вычислительной сложностью.

3. Разработан новый метод калибровки оптических камер, который не накладывает никаких ограничений на используемые камеры, работает на порядок быстрее существующих методов, и способен корректировать результаты калибровки в процессе работы. Доказано, что при использовании данного метода задача калибровки является вычислительно устойчивой.

4. Разработана программная библиотека реализующая все созданные алгоритмы. Данная библиотека позволяет создавать простой, удобный и легко расширяемый пользовательский интерфейс, который может успешно применяться как при работе на персональном компьютере, так и при работе с «большими» экранами.

Теоретическая и практическая значимость. Разработанная методика человеко-компьютерного взаимодействия на базе жестов использует распознавание источника света, а не сложное и приближённое распознавание конечностей человека. Помимо простоты и надёжности, это обеспечивает естественность ввода, и лёгкость реализации за счёт самого дешёвого оборудования. Ввод на основе распознавания источника света позволяет расширить область использования, распознавая не только движения человека, но и перемещения произвольного объекта, на котором закреплён источник света.

Разработанный метод калибровки оптических камер обеспечивает вычислительную устойчивость решения задачи калибровки. Для осуществления калибровки не требуется выполнять какие-то специальные действия или использовать специальный предмет. Данный метод может использоваться для калибровки произвольной системы оптических камер. Можно использовать устройства разных производителей, с разными линзами и фокусным расстоянием, обладающие различным разрешением и различной частотой кадров. Разработанный метод работает на порядки быстрее традиционных решений.

Разработанные программная библиотека и алгоритмы могут быть использованы при создании пакетов математического моделирования.

Апробация работы. Результаты диссертации были представлены на X и XI международном семинаре по супервычислениям и

математическому моделированию (Саров, 2008, 2009), третьей международной конференции «Информационно-математические технологии в экономике, технике и образовании» (Екатеринбург, 2008) и международной научной конференции «Параллельные вычислительные технологии (ПаВТ'2009)» (Нижний Новгород, 2009).

Публикации. Основные результаты диссертации изложены в 8 работах, список которых приведён в конце автореферата. Из них 2 опубликованы в ведущих рецензируемых научных журналах, определеных ВАК. Результаты работ получены диссертантом самостоятельно.

Структура и объём диссертации. Диссертация состоит из введения, 5 глав, заключения и списка литературы. Объём диссертации составляет 119 страниц, включая библиографический список из 43 наименований.

СОДЕРЖАНИЕ ДИССЕРТАЦИИ

ВО ВВЕДЕНИИ рассматриваются достоинства и недостатки различных методов человеко-машинного взаимодействия. Описывается пробема выбора интерфейса для программ математического моделирования, обосновывается актуальность диссертационной работы и формулируется её цель.

В ПЕРВОЙ ГЛАВЕ обосновывается выбор жестов в качестве метода взаимодействия, приводится обзор существующих технологий ввода жестов, и производится выбор технологии, в наибольшей степени подходящей для поставленной задачи.

В начале главы обосновывается связь между интерфейсом и физическим устройством ввода (манипулятором). Далее формулируются требования, которым должно удовлетворять устройство ввода:

1. Устройство ввода должно быть универсальным, т.е. для выполнения различных действий не должны требоваться различные устройства ввода.

2. Устройство ввода должно позволять взаимодействовать виртуальным объектом как с его реальным аналогом, причём как для уже существующих манипуляций, так и для любых манипуляций, которые могут возникнуть в будущем.

3. Устройство ввода должно быть трёхмерным, т.е. обеспечивать ввод трёх координат положения в пространстве.

4. Устройство ввода должно быть простым с технической точки зрения (простота установки устройства ввода и лёгкость его повседневного использования) и, по возможности, недорогим.

Выполнение второго требования невозможно в случае использования манипуляторов в качестве устройств ввода, поскольку любой манипулятор обеспечивает лишь ограниченный набор взаимодействий. Следовательно, необходимо отказаться от физического манипулятора и осуществлять ввод при помощи трёхмерных жестов (что автоматически обеспечивает выполнение первых трёх требований).

Формулируется задача исследования: создать или выбрать простую и недорогую технологию ввода трёхмерных жестов, а также разработать основанный на ней расширяемый интерфейс, позволяющий осуществлять манипуляции с трёхмерным виртуальным объектом точно так же, как и если бы реальная копия этого объекта находилась у пользователя в руках.

Далее приводится обзор существующих технологий захвата движений (Motion Capture), а также перечисляются их достоинства и недостатки применительно к поставленной задаче. Отдельно описываются методы калибровки оптических камер (калибровка необходима для оптического захвата движений), и приводятся проблемы, связанные с этими методами.

В конце главы производится выбор технологии ввода трёхмерных жестов в наибольшей степени подходящей для поставленной задачи. В качестве основных критериев отбора выступают точность определения положения в пространстве и удобство для пользователя. Этим критериям лучше всего соответствует оптический захват движений с активными маркерами, который и берётся за основу.

ВО ВТОРОЙ ГЛАВЕ описывается разработанный метод ввода трёхмерных жестов, приводится алгоритм вычисления положения маркеров в пространстве и доказывается теорема о вычислительной сложности данного алгоритма. Также описываются результаты исследования качества работы алгоритма, как при помощи тестовых замеров, так и с участием потенциальных пользователей.

В качестве недорогого манипулятора для ввода трёхмерных жестов был выбран обыкновенный карманный фонарик. В качестве датчика света используется стандартная веб-камера, которая крепится к мони-

тору. Фонарик, помимо низкой стоимости и лёгкости использования обладает одним немаловажным преимуществом: он является не точечным, а протяжённым источником света. Иными словами, камера видит источник света не как точку, а как круг или эллипс. Эта особенность позволяет нам вычислять расстояние до объекта на основе анализа изображения всего одной камеры. В силу этого, при установке системы не требуется производить калибровку, а сама камера может быть размещена совершенно произвольно. Для облегчения выделения светового пятна в кадре, на источник света одевается цветная бумага (что также предотвращает ослепление веб-камеры и других людей).

Поскольку источники света гораздо ярче, чем прочие объекты, можно настроить яркость камеры таким образом, чтобы в кадре были видны только источники света, а всё остальное сливалось в чёрный фон. В результате, без всякой программной обработке, мы получим изображение, в котором присутствуют только источники света (либо их отражения от зеркальных поверхностей). Проанализировать такой кадр и выделить в нём световые пятна фонариков очень легко - достаточно проверить, что форма светового пятна близка к эллипсу, а его цвет – заданному.

Зная эллипс светового пятна, можно вычислить координаты источника света по формулам:

$$x = \frac{c_2\left(\frac{w}{2} - x_0\right)}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}$$

$$y = \frac{c_2\left(\frac{h}{2} - y_0\right)}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}$$

$$z = \frac{c_1 w}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}$$

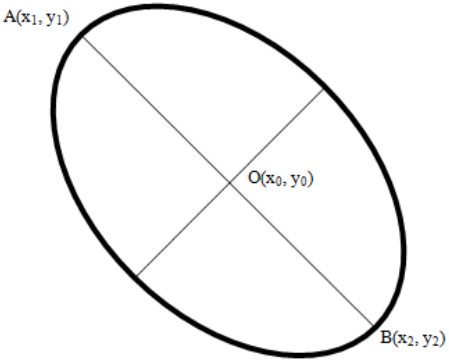

Рис. 1 – Эллипс светового пятна

где AB – самая длинная диагональ эллипса, O – центр эллипса, w и h – высота и ширина кадра в пикселях соответственно, c_1 и c_2 – константы, задающие чувствительность ввода (рис. 1).

Теорема 1: *Если количество используемых источников света не превышает* $\sqrt[3]{MN}$*, то вычислительная сложность алгоритма определения положения источников света в пространстве составляет O(MN), где MxN – разрешение камеры.*

Поскольку в реальных системах количество источников света не превышает $\sqrt[3]{MN}$, определение положения источников света в пространстве работает за время, линейно зависящее от разрешения камеры.

Проведённые исследования показали, что разработанный алгоритм безошибочно выделяет источник света в кадре при любых условиях внешнего освещения, причём окружающие условия практически не влияют на загрузку процессора. Исследования, проведённые с участием пользователей, показали, что время обучения интерфейсу занимает, в среднем, не более трёх минут, после чего пользователи могли успешно задавать своё положение в трёхмерном пространстве, а также выполнять перемещение объектов. Всё это позволяет говорить о том, что разработанный метод ввода является удобным для пользователей и обладает достаточной скоростью и точностью.

В ТРЕТЬЕЙ ГЛАВЕ рассматривается проблема создания пользовательского интерфейса, который бы одновременно обладал богатыми возможностями и обеспечивал бы непосредственное взаимодействие. В главе приводится алгоритм распознавания трёхмерных жестов и доказывается, что данный алгоритм обладает наименьшей вычислительной сложностью.

Существует различные виды и классификации жестов. В нашем случае, жест G – это траектория перемещения рук пользователя в пространстве. Т.к. мы снимаем кадры с какой-то частотой, данная траектория дискретизирована, т.е. представлена в виде последовательности трёхмерных точек: (x_1, y_1, z_1), (x_2, y_2, z_2), … , (x_n, y_n, z_n). Распознавание жеста состоит в вычислении расстояния между жестом G и каждым из базовых шаблонов $B_1, B_2, \dots, B_m$. (которые также представляют собой последовательность трёхмерных точек). Допустимые преобразования, метрики расстояния и алгоритмы, находящие эти метрики, могут быть самыми различными. Критериями того, какой метод лучше, как правило, являются скорость работы и процент ошибок распознавания.

В нашем случае задача распознавания жестов состоит из двух частей: выделение жеста в непрерывном пользовательском вводе и, собственно, вычислении расстояния. Необходимость выделять жест связана с тем, что на источниках света отсутствуют какие-либо кнопки, которые могли бы задавать режим работы (что выполняется в данный момент: жест или перемещение).

Выделение жестов в непрерывном пользовательском вводе осуществляется следующим образом. Каждый раз, когда появляются новые данные (т.е. каждый кадр), мы предполагаем, что пользователь только что завершил ввод жеста, и пытаемся его распознать. Таким образом, конец траектории – это последняя введённая точка. Начало траектории – это какая-то точка в прошлом, которая может быть определена лишь по косвенным признакам. Эксперимент показал, что равномерное разбиение (в качестве начал берутся точки, расположенные с шагом 1/8 секунды) позволяет уверенно распознавать вводимые пользователем траектории. Такой подход также гарантирует, что если мы ограничим продолжительность жеста, то число запусков алгоритма распознавания будет ограничено константой.

В качестве основы алгоритма распознавания жестов используется "$1 Gesture Recognizer Algorithm", предложенный Jacob O. Wobbrock, Andrew D. Wilson, Yang Li. Данный алгоритм был выбран в качестве основы, поскольку он обладает очень высокой точностью (более 97% при использовании единственного шаблона для каждого жеста и более 99% при использовании трёх шаблонов для каждого жеста), высокой скоростью работы и модульной структурой, позволяющей производить модификации.

Поскольку "$1 Gesture Recognizer Algorithm" умеет распознавать лишь двумерные жесты, была выполнена модификация, позволившая работать данному алгоритму и в трёхмерном пространстве. Тот факт, что человеческие жесты хорошо укладываются на плоскость либо вообще выполняются вдоль одной прямой, позволяет существенно упростить модифицированный алгоритм.

Алгоритм распознавания жестов состоит из 7 шагов:

1. На основе траектории жеста G, представленной в виде последовательности точек (x_1, y_1, z_1), (x_2, y_2, z_2), … , (x_n, y_n, z_n), построим новую траекторию P состоящую ровно из 64 точек: (x'_1, y'_1, z'_1), (x'_2, y'_2, z'_2), … , $(x'_{64}, y'_{64}, z'_{64})$ таким образом, чтобы:

a. $(x'_1, y'_1, z'_1) = (x_1, y_1, z_1)$

b. $(x'_{64}, y'_{64}, z'_{64}) = (z_n, y_n, z_n)$

c. $L(i,i+1) = L(j,j+1)\ \forall\ i,j = 1,\dots,63$, где $L(i,i+1)$ – расстояние между точками i и $i+1$ траектории P вдоль ломанной, образуемой последовательностью точек из G.

2. Найдём центр траектории $C(x_0, y_0, z_0) = \left(\frac{\sum_{i=1}^{64} x'_i}{n}, \frac{\sum_{i=1}^{64} y'_i}{n}, \frac{\sum_{i=1}^{64} z'_i}{n}\right)$.

3. Построим набор векторов $\vec{V}_i = (x'_i - x_0, y'_i - y_0, z'_i - z_0)$ и упорядочим их по убыванию длины. Обозначим $\overrightarrow{W} = (x'_1 - x_0, y'_1 - y_0, z'_1 - z_0)$.

4. Проверим, является жест линейным (жест называется линейным, если все его точки практически лежат на одной прямой, т.е. отклонение не превышает допустимых пределов).

4.1. Найдём вектор $\vec{S} = \sum_{i=1}^{24} PM(\overrightarrow{V_i})$, где $PM(\vec{V})$ возвращает $\vec{V}$, если сумма координат $\vec{V}$ положительна, и $-\vec{V}$ в противном случае.

4.2. Если $\forall\, i = 1, \dots, 64$ угол между векторами $\overrightarrow{V_i}$ и $\vec{S}$ отличается от 0^o или 180^o не более чем на 10^o, то жест G считается линейным. Иначе жест G считается двумерным.

5. Если жест двумерный, то вычисляется плоскость наилучшего приближения при помощи метода наименьших квадратов, и находится её нормальный вектор $\vec{N}$. Если жест линейный, то данный шаг пропускается.

6. Для каждой базовой траектории соответствующего типа (линейная/двумерная) производится следующее:

6.1. Для линейного жеста вычисляется угол между векторами $\vec{S}$ (базовой и пользовательской траекторий). Для двумерного вычисляется угол между векторами $\vec{N}$. Если найденный угол превышает 45^o, то дальнейшие сравнения с данной базовой траекторией не производятся.

6.2. В случае двумерного жеста построим ортонормированные базисы для пользовательской и базовой траекторий из векторов $\vec{N}$, $\frac{\vec{T}}{|\vec{T}|}$, $\left[\vec{N}, \frac{\vec{T}}{|\vec{T}|}\right]$, где $\vec{T} = \overrightarrow{W} - \frac{|\overrightarrow{W}| \cos \angle(\vec{N}, \overrightarrow{W})}{|\vec{N}|} \vec{N}$, $\angle(\vec{N}, \overrightarrow{W})$ - угол между векторами, $[\vec{a}, \vec{b}]$ – векторное произведение $\vec{a}$ и $\vec{b}$. Далее, найдём матрицу перехода M из базиса пользовательской траектории в базис базовой траектории. Вычислим $\vec{Q}_i = M\overrightarrow{V_i}$.

В случае линейного жеста найдём матрицу поворота M вокруг вектора $\left[\overrightarrow{S_{\text{польз.}}}, \overrightarrow{S_{\text{баз.}}}\right]$ на угол $\angle(\overrightarrow{S_{\text{польз.}}}, \overrightarrow{S_{\text{баз.}}})$. Вычислим $\vec{Q}_i = M\overrightarrow{V_i}$.

6.3. Найдём k – значение максимальной по модулю координаты среди всех $\vec{Q}_i$. Если $k < C_{min}$, где C_{min} – некоторая константа, то сравнение не производится. Иначе вычислим $\vec{A}_i = \frac{C}{k}\overrightarrow{Q_i}$, где C – константа.

6.4. Выполним поиск минимального расстояния l_{min} между пользовательской траекторией и базовой траекторией с использованием «по-

иска золотого сечения». Поиск оптимального угла выполняется в диапазоне [-45°, 45°] с шагом 2°. Вращение выполняется вокруг $\overrightarrow{S_{баз}}$ (линейный случай) или $\overrightarrow{N_{баз}}$ (двумерный случай). Расстояние между траекториями вычисляется по формуле:

$$l = \sum_{i=1}^{64} \left|\overrightarrow{A_i} - \overrightarrow{B_i}\right|$$

где $\overrightarrow{B_i}$ – i-ая точка траектории, получаемой из базовой траектории, путём применения к ней преобразований, описанных в пунктах 1, 2, 3, 4, 5 и 6.3 данного алгоритма.

6.5. Вычисляется степень похожести в процентах по формуле:

$$r = 100\left(1 - \frac{2l_{min}}{\sqrt{3}C}\right)$$

Если r < 0, то r = 0.

7. Среди всех r выбирается максимальная r_{max}. Если это значение превышает минимальный порог, то введенная пользователем траектория считается распознанной. Значение r_{max} и номер траектории, которому данное значение соответствует, являются результатами работы данного алгоритма.

Теорема 2: *Вычислительная сложность алгоритма распознавания жестов составляет O(B), где B – количество базовых траекторий. Данная вычислительная сложность является наилучшей из всех возможных.*

Далее рассматривается вопрос создания легко расширяемого интерфейса, который позволял бы взаимодействовать с виртуальным объектом так, как если бы перед нами находился его реальный аналог.

Поскольку в реальной руке пользователя находится предмет – источник света, при помощи которого осуществляется ввод, в виртуальной руке пользователя тоже должен находиться некий предмет – инструмент, при помощи которого осуществляется воздействие на виртуальную среду. Иными словами, взаимодействия с виртуальным объектом должны выполняться при помощи виртуальных инструментов.

Данный подход имеет массу преимуществ. Во-первых, данный интерфейс легко расширяется: достаточно добавить новый виртуальный инструмент, что делается исключительно программным образом. Во-вторых, если в качестве виртуальных инструментов использовать хорошо известные объекты, то пользователь сможет проецировать опыт

реальной жизни на виртуальную среду. Просто взглянув на предмет, пользователь будет знать, какой жест необходимо осуществить, чтобы воспользоваться данным инструментом, и каких результатов при взаимодействии с объектом следует ожидать. В-третьих, жест взаимодействия связан с виртуальным инструментом и потому не зависит от национальной культуры пользователя (в отличие, скажем, от жестов согласия, которые в разных странах различны). Это делает интерфейс универсальным и доступным к использованию в любой точке мира без каких-либо модификаций. В-четвёртых, виртуальные инструменты задают контекст, ограничивая набор возможных взаимодействий. Это повышает точность распознавания жестов и уменьшает количество ошибок ввода.

В конце главы описывается пример интерфейса для сторонней системы, которая наглядно представляет большие связные графы в трёхмерном пространстве. Данный пример также использовался для исследования качества интерфейса с участием потенциальных пользователей. Результаты исследования позволяют говорить о высоком качестве распознавания жестов и о широкой доступности данного интерфейса (распознавание жестов успешно выполнялось без подстройки систему под каждого пользователя). Также необходимо отметить, что некоторые пользователи заинтересовалась системой настолько, что пожелали использовать её в своей повседневной работе.

В ЧЕТВЁРТОЙ ГЛАВЕ рассматривается проблема калибровки оптических камер. Приводится алгоритм автоматической калибровки камер, не требующий от пользователя выполнения каких-либо специальных действий. Данный метод не накладывает никаких ограничений на камеры, умеет корректировать результаты калибровки в процессе работы, и выполняется на три порядка быстрее традиционных методов. Показывается, что при использовании данного метода задача калибровки является вычислительно устойчивой. Также доказывается теорема о вычислительной сложности алгоритма.

Описанный во второй главе метод ввода трёхмерных жестов использует всего одну камеру, а потому обладает небольшой активной зоной. Для работы на персональном компьютере данное ограничение не является существенным, однако при работе с «большими» экранами необходимы большая активная зона, и сохранение работоспособности в случае возникновения препятствий между источником света и

камерой (обзор могут загораживать другие пользователи или, скажем, колонны). Следовательно, при реализации трёхмерного интерфейса для «больших» экранов нам необходимо вернуться к одновременному использованию нескольких камер и к процедуре калибровки.

В традиционных системах оптического захвата движений трёхмерные координаты P маркера восстанавливаются по формуле:

$$P_{1\times3} = tM_{3\times3}\begin{bmatrix} x \\ y \\ f \end{bmatrix} + C_{1\times3}$$

где M – матрица поворота камеры, C – координаты центра камеры, f – фокусное расстояние камеры, t – числовой коэффициент. Параметры M, C и f определяются в процессе калибровки, параметр t вычисляется в процессе работы путём сопоставления кадров с разных камер (выполняется триангуляция). Что же касается, собственно, калибровки, то различных методов и алгоритмов существует достаточно много. В общем случае калибровка сводятся к решению следующей задачи:

Имеется множество координат $(x_i^j(k), y_i^j(k))$, где i – индекс точки на изображении камеры номер j во время k-ой демонстрации эталонного объекта. Необходимо найти такие M^j, C^j, f^j и $t_i^j(k)$, чтобы:

1. $t_q^j(k)M^j\begin{bmatrix} x_q^j(k) \\ y_q^j(k) \\ f_q^j \end{bmatrix} \neq t_r^j(k)M^j\begin{bmatrix} x_r^j(k) \\ y_r^j(k) \\ f_r^j \end{bmatrix}$ для всех j, k и q≠r. Иными словами, разные точки на изображении одной камеры должны соответствовать разным точкам в трёхмерном пространстве.

2. Количество различных значений во множестве $\left\{ t_i^j(k)M^j\begin{bmatrix} x_i^j(k) \\ y_i^j(k) \\ f_i^j \end{bmatrix} + C^j \;\middle|\; \forall\, i,j \right\}$ не превышает количества используемых при калибровке маркеров для каждого значения k.

В общем случае, данная задача решается при помощи последовательных приближений, на что уходит порядка 60-90 минут для калибровки 16 камер. При таком количестве вычислений, учитывая зашумлённость исходных данных, учитывая возможное отсутствие синхронизации камер (из-за чего движущиеся маркеры снимаются в разных точках пространства), учитывая возможную неоднозначность триан-

гуляции, очень легко получить совершенно неверные результаты. Причём увеличение количества демонстраций эталонного объекта может лишь ухудшить точность калибровки.

Описанный во второй главе диссертации метод ввода трёхмерных жестов позволяет определять трёхмерное положение источника света относительно камеры без использования данных от других камер. В этом случае абсолютное положение источника света в пространстве может быть вычислено по следующей формуле:

$$P_{3\times1} = fM_{3\times3}\begin{bmatrix}x\\y\\z\end{bmatrix} + C_{3\times1}$$

В отличие от традиционных методов, здесь отсутствует параметр t. Это позволяет предложить следующий метод калибровки.

Пусть имеются две камеры: камера 1 откалибрована (т.е. известны M(1), C(1) и f(1)), а камера 2 не откалибрована (т.е. M(2), C(2) и f(2) не известны). Пусть обе камеры наблюдают за перемещение маркера в течение какого-то времени. Мы получим два упорядоченных по времени множества координат $\{(x_i(1), y_i(1), z_i(1))\}$ и $\{(x_i(2), y_i(2), z_i(2))\}$. Выберем q, r, s, t таким образом, чтобы матрицы $N(i) = \begin{bmatrix} x_q(i) & x_r(i) & x_s(i) & x_t(i) \\ y_q(i) & y_r(i) & y_s(i) & y_t(i) \\ z_q(i) & z_r(i) & z_s(i) & z_t(i) \\ 1 & 1 & 1 & 1 \end{bmatrix}$, где i=1, 2, имели ранг 4.

Пусть $A = fM$, и матрица D получается из матрицы A путём присоединения к ней справа матрицы C, т.е. D_{3x4}=[A C]. Тогда:

$$P = fM\begin{bmatrix}x\\y\\z\end{bmatrix} + C = A\begin{bmatrix}x\\y\\z\end{bmatrix} + C = D\begin{bmatrix}x\\y\\z\\1\end{bmatrix}$$

Следовательно

$$D(1)N(1) = D(2)N(2)$$

Откуда находим:

$$D(2) = D(1)N(1)N(2)^{-1}$$

Зная D(2) мы получаем A(2) и C(2). Зная A(2), мы, из условия det(M)=1, находим M(2) и f(2). Таким образом, мы нашли все неизвестные параметры, выполнив калибровку за единственную итерацию.

Необходимость найти четыре точки такие, чтобы ранг матрицы N равнялся 4, не является существенной проблемой, поскольку этому

условию удовлетворяют любые четыре точки, через которые не проводится плоскость. Иными словами, любая траектория, которая не укладывается в плоскость, подходит для осуществления калибровки.

Поскольку в данном методе каждая камера работает независимо, и в качестве калибрующих перемещений может выступать практически любая активность пользователя, процедуру калибровки можно выполнять в фоновом режиме. Т.е. при запуске системы активна одна камера (которой достаточно для начала работы), а затем, без специальных действие со стороны пользователя, выполняется калибровка всех остальных камер и включение их в работу. И, в силу простоты вычислений, всё это может выполняться в реальном времени.

Калибровка последней камеры не означает прекращение процесса. Система постоянно проверяет себя для того чтобы выявить возможные ошибки в калибровке, или среагировать на добавление камер или их перестановку. Этот самоконтроль и перекалибровка при необходимости позволяют не ограничивать количество используемых в период калибровки источников света – даже если система обознается и примет два разных источника за один, то потом сама же исправит свою ошибку.

Приведённые выше формулы прекрасно работают в том случае, когда все координаты известны точно. Однако в реальной системе любое значение вычисляется с некоторой погрешностью, как в силу погрешности вычисления, так и в силу оптического шума, дискретности пиксельной сетки, и несинхронности камер. Кроме того, существенная активность пользователя состоит в выполнении жестов, которые, как указывалось ранее, хорошо укладываются в плоскость. Поэтому приведённый ниже алгоритм отличается от предложенного выше метода.

Будем описывать каждую камеру следующим набором параметров:

* Положением в абсолютной системе координат $C_{3\times1}$.

* Матрицей перехода из системы координат камеры в абсолютную систему координат A_{3x3}.

* Набором историй перемещения источника света за последние пять секунд. История перемещения – это множество пар $\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}, t\right)$, где

t – момент времени, в который были зафиксированы координаты $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$.

Каждая история находится в состоянии «активна» или «не активна».

* Состоянием «откалибрована»/«не откалибрована».

* Степенью недоверия к камере b (неотрицательное целое число).

Первую подключенную камеру примем в качестве точки отсчёта:

$$b=0,\ C=\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},\ A=\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Каждый источник света, обозреваемый откалиброванной камерой, обладает следующими параметрами:

* Положение в абсолютной системе координат P.

* Состояние «свободен»/«не свободен».

* Группа, с которой связан источник света (если он находится в состоянии «не свободен»).

Источники света, обозреваемые разными откалиброванными камерами, объединяются в группы. Каждая группа означает один реальный источник, а элементы группы – это его ракурсы, полученные с разных камер. Каждая группа обладает следующими характеристиками:

* Положение в абсолютной системе координат G. Это значение означает положение реального источника света в абсолютной системе координат безотносительно используемых камер.

* Состояние «активна»/«пассивна».

Алгоритм калибровки, верификации и вычисления абсолютных положений источников света в пространстве состоит из трёх процедур:

Процедура 1. Цикл работы системы.

Всякий раз, когда новые данные поступают с одной из камер (назовём её Cam), выполняются следующие действия:

1. Новые положения $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$ и текущий момент времени t обозреваемых камерой Cam источников света записываются в соответствующие истории перемещения. Изменённые истории переходят в состояние «активна»; все остальные истории этой камеры – в «не активна».

2. Из всех историй перемещения («активных» и «не активных») камеры Cam удаляются устаревшие данные – элементы, t которых отличается от текущего момента времени более чем на 5 секунд.

3. Если камера находится в состоянии «не откалибрована», то:

3.1. Строится список S историй перемещения, в который включается все «активные» истории перемещения камеры Cam.

3.2. Строится список L историй перемещения, в который включается все «активные» истории перемещения всех откалиброванных камер.

3.3. Для каждой пары значений из списков S и L выполняется процедура калибровки (см. далее).

4. Если камера Cam находится в состоянии «откалибрована», то выполняется процедура вычисления абсолютного положения источников света в пространстве (см. далее).

5. Для всех откалиброванных камер, у которых b>0, происходит уменьшение степени недоверия на единицу. Степень недоверия первой камеры уменьшается на пять единиц. Отрицательное значение заменяется на ноль. Если в системе осталась лишь одна откалиброванная камера, то её степень недоверия выставляется на ноль.

6. Среди всех откалиброванных камер, у которых b>15, случайно выбирается одна. Выбранная камера переходит в состояние «не откалибрована».

Процедура 2. Процедура калибровки.

На вход подаются две истории: для откалиброванной камеры C и не откалиброванной камеры U. Истории представляют собой набор пар ($\vec{P_i}$, t_i), где $\vec{P_i}$ – положение источника света в пространстве в момент времени t_i. Необходимо выяснить, соответствуют ли эти наборы одному и тому же источнику света, и, если возможно, произвести калибровку камеры. Делается это следующим образом:

1. Если в истории есть пропущенные кадры (камера не видела источник света), то история усекается до последнего пропущенного кадра. Кадр считается пропущенным, если промежуток времени между двумя соседними элементами истории перемещения в 1.5 раза превышает частоту кадров данной камеры. Например, если история состоит из кадров 1, 4, 7, 8 и 9, то в результате усечения мы получим историю из кадров 7, 8 и 9. Также из истории камеры U удаляются все пары ($\vec{P_i}$, t_i) у которых t_i меньше минимального значения t в истории камеры C.

2. Производится синхронизация историй путём вычисления положений источника света для камеры U в моменты времени из истории камеры C на основе линейной интерполяции. В результате мы получаем два множества положений источника света в пространстве относи-

тельно каждой из камер. Эти множества будут синхронизированы по времени. Обозначим их как PU=$\{\overrightarrow{pu_i}\}$ и PC=$\{\overrightarrow{pc_i}\}$.

3. Найдём $\overrightarrow{sc} = \frac{\sum_1^n \overrightarrow{pc_i}}{n}$, $\overrightarrow{su} = \frac{\sum_1^n \overrightarrow{pu_i}}{n}$. Затем построим множества векторов VU=$\{\overrightarrow{vu_i}\}$ и VC=$\{\overrightarrow{vc_i}\}$, где $\overrightarrow{vu_i} = \overrightarrow{pu_i} - \overrightarrow{su}$, $\overrightarrow{vc_i} = \overrightarrow{pc_i} - \overrightarrow{sc}$.

4. Среди всех возможных пар индексов (j, k), где $1 \le j, k \le n$, выбираются те, которые удовлетворяют следующим условиям:

* Длины векторов $\overrightarrow{vu_j}$, $\overrightarrow{vu_k}$, $\overrightarrow{vc_j}$ и $\overrightarrow{vc_k}$ превышают 30 единиц.

* Угол между векторами $\overrightarrow{vu_j}$ и $\overrightarrow{vu_k}$ лежит в пределах [45^0, 135^0].

* Угол между векторами $\overrightarrow{vc_j}$ и $\overrightarrow{vc_k}$ лежит в пределах [45^0, 135^0].

Для каждой найденной пары индексов строятся базисы BU=($\overrightarrow{vu_j}$, $\overrightarrow{vu_k}$, [$\overrightarrow{vu_j}$, $\overrightarrow{vu_k}$]) и BC=($\overrightarrow{vc_j}$, $\overrightarrow{vc_k}$, [$\overrightarrow{vc_j}$, $\overrightarrow{vc_k}$]), где [$\vec{a}$,$\vec{b}$] – векторное произведение $\vec{a}$ и $\vec{b}$. Затем вычисляется матрица перехода M из базиса BU в базис BC. И наконец, вычисляется значение $D = \frac{\sum_1^n |\mathrm{A(C)}M\overrightarrow{pu_i} - \mathrm{A(C)}\overrightarrow{pc_i}|}{n}$, где $|\vec{a}|$ – длина вектора $\vec{a}$, A(C) – матрица перехода откалиброванной камеры C.

Из всех найденных D выбирается минимальное D_{min} (этому значению соответствует матрица перехода M_{min}). Если D_{min} – превышает максимальный порог, или подходящих пар индексов (j, k) не было найдено, то калибровка не производится (камеры наблюдают разные фонарики или перемещения фонарика слишком малы). Если D_{min} – не превышает максимальный порог, и для всех $D \le D_{min} + C_1$, где C_1 – некоторая константа, выполняется

$$\left|m_{min}^{ij} - m^{ij}\right| \le C_2(D - D_{min}),$$

где m^{ij} - элементы матрицы M, C_2 – некоторая константа, то камера U переходит в состояние «откалибрована», а её параметры выставляются следующим образом:

* Матрица перехода из системы координат камеры в абсолютную систему координат $\mathrm{A(U)=A(C)}M_{min}$

* Абсолютное положение камеры $\mathrm{C(U) = C(C) + A(U)}\overrightarrow{su} - \mathrm{A(C)}\overrightarrow{sc}$

* Степень недоверия камере b(U)=0.

Процедура 3. Процедура вычисление абсолютных положений источников света в пространстве.

Поскольку камеры вычисляют абсолютное положение источников света независимо друг от друга, необходимо объединить эти данные,

предварительно выяснив, какие камеры обозревают один и тот же источник света, а какие обозревают разные. Алгоритм следующий:

1. Для каждого обозреваемого камерой Cam источника света производится вычисление абсолютного положения $P = A\begin{bmatrix} x \\ y \\ z \end{bmatrix} + C$.

2. Если камера Cam увидела новый источник света, то он помечается как «свободный». Если старый источник в результате перемещения удалился от центра группы больше чем на заданную величину, то он также помечается как «свободный». Прочая разметка сохраняется.

3. Для всех источников света, помеченных как «свободные» *принадлежащих камере Cam*, выполним сопоставление с группами при помощи Венгерского алгоритма. Если в результате сопоставления расстояние между «свободным» источником света и центром группы, с которой его сопоставили, оказывается больше допустимого порога, то создаётся новая группа, состоящая из одного элемента – этого источника света. Все «свободные» источники света камеры Cam помечаются как «не свободные».

4. Для каждой группы, содержащей, или содержавшей источники света камеры Cam, пересчитываются координаты центра G как среднее арифметическое абсолютных положений всех входящих в группу источников света. Если в результате выполнения предыдущего шага у нас получилось больше групп, чем максимальное число используемых источников света, то пометим лишние группы как «пассивные». В качестве лишних групп выберем те, что содержат наименьшее число элементов. Количество лишних групп выберем таким, чтобы осталось групп ровно столько, сколько источников света используется в системе. Оставшиеся группы пометим как «активные». Найденные центры G «активных» групп объявляются положениями источников света и передаются в программу, использующую данный интерфейс.

5. Выполняется пересчёт степеней недоверия. Во-первых, для камеры, сообщившей об элементе из «пассивной» группы, степень недоверия b увеличивается на единицу за каждый элемент. Во-вторых, для каждой откалиброванной камеры можно вычислить зону видимости, т.е. пространственную область при нахождении в которой источник света будет виден камере. Пусть камера A видит источник света, который находится в зоне видимости камеры B. Если при этом камера B

не видит источник света, то степень недоверия к обеим камерам увеличивается на три единицы.

6. Все «пассивные» группы удаляются. Входившие в эти группы источники света помечаются как «свободные».

Теорема 3: *Процедура калибровки камер является вычислительно устойчивой.*

Теорема 4: *Вычислительная сложность алгоритма составляет* $O(F^2(C)+F))$*, где C – количество используемых камер, F – количество используемых источников света. Если все камеры откалиброваны, то вычислительная сложность составляет* $O(F*(C+F^2))$.

Для тестирования качества работы алгоритмов был создан виртуальный испытательный стенд. Виртуальными на стенде являлись камеры – вместо них, использовалась специальная процедура, которая генерировала пользовательских ввод, симулируя данные, которые бы поступали от реальной камеры, при осуществлении пользователем подобного ввода. Серия виртуальных экспериментов показала, что добавление случайного шума порядка 5% от размера калибрующей траектории позволяет успешно произвести калибровку, причём расстояние между вычисленным и реальным положениями не превышает величины шума. Это экспериментально подтверждает то, что алгоритм калибровки камер является вычислительно устойчивым.

Время, необходимое на калибровку 16 камер составляет примерно 0.1 секунды, что на три порядка быстрее, чем при использовании стандартных методов. Также эксперименты показали, что система из 16 камер и двух источников света способна работать на обычном ноутбуке в реальном времени (в том числе и выполнять калибровку) с частотой порядка 200 кадров в секунду. На перекалибровку уходит столько же времени, сколько и на первоначальную калибровку, каких-либо провалов в производительности перекалибровка не вызывает.

Некоторые виртуальные испытания были повторены с использованием реальных камер. Результаты получились примерно такими же.

В конце главы отмечается, что отсутствие синхронизации камер может использоваться для увеличения частоты кадров и, как следствие, точности вводимых данных. Также данный метод позволяет динамически отключать лишние камеры без ущерба для работы системы и включать их, когда они потребуются вновь (когда они вновь смогут увидеть источники света).

В ПЯТОЙ ГЛАВЕ производится сравнение разработанных методов с существующими решениями и приводится описание программной библиотеки, реализующей все описанные в диссертации алгоритмы.

По сравнению с существующими методами калибровки оптических камер, предлагаемый подход не требует от пользователя выполнения каких-либо специальных действий со специальными объектами, не накладывает никаких ограничений на камеры и работает на три порядка быстрее. Также при использовании разработанного метода задача калибровки является вычислительно устойчивой. Кроме того, данный метод может корректировать результаты калибровки в процессе работы - качество, которого традиционные подходы лишены.

По сравнению с существующими методами ввода жестов, анализирующими изображения с оптических камер, предлагаемый подход не накладывает жёстких ограничений на фон изображения, обладая при этом высокой точностью. Кроме того, стоимость разработанного решения оказалась ниже, чем стоимость уже существующих методов.

Для сравнения разработанного интерфейса с интерфейсом, использующим мышь, было проведено специальное исследование с участием пользователей. Результаты показывают, что разработанный интерфейс является более удобным, чем интерфейс на основе мыши. Всё это позволяет говорить о преимуществе разработанного интерфейса над традиционными решениями применительно к задаче взаимодействия с трёхмерной средой.

Все предложенные в диссертации алгоритмы реализованы в виде dll библиотеки на Microsoft Visual Studio 2008 .NET. Данная библиотека содержит 2 публичных класса, 13 публичных процедур, 6 публичных событий и 13 публичных свойств. Также было разработано приложение для создания шаблонов жестов, позволяющее задавать шаблон как путём ввода координат точек, так и при помощи источника света. Общее количество строк кода, написанное в рамках работы над данным проектом, превышает десять тысяч.

В ЗАКЛЮЧЕНИИ формулируются основные результаты работы, перечисляются возможности разработанных алгоритмов и интерфейса и рассматривается вопрос применимости созданных методов и алгоритмов помимо программ математического моделирования.

ПУБЛИКАЦИИ АВТОРА ПО ТЕМЕ ДИССЕРТАЦИИ

СТАТЬИ, ОПУБЛИКОВАННЫЕ В ВЕДУЩИХ РЕЦЕНЗИРУЕМЫХ НАУЧНЫХ ЖУРНАЛАХ, ОПРЕДЕЛЁННЫХ ВАК.

1. В.Л. Авербух, А.В. Зырянов. Методы манипуляций объектами в трёхмерных визуальных средах. // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2009 г., Вып. 3, стр. 58-69.

2. Зырянов А.В. Самокалибрующаяся масштабируемая система ввода трёхмерных жестов // Системы управления и информационные технологии, 1.1(39), 2010 г. стр. 135-139.

ДРУГИЕ ПУБЛИКАЦИИ

3. Зырянов А. В., Авербух В. Л. Интерфейс на основе жестов для взаимодействия с виртуальными средами. // X международный семинар по супервычислениям и математическому моделированию. 29 сентября - 3 октября 2008 г. Тезисы докладов. стр. 73-74.

4. Зырянов, А.В. Методы ввода и распознавания жестов для взаимодействия с виртуальными средами. // Третья международная конференция "Информационно-математические технологии в экономике, технике и образовании". 20-22 ноября 2008 г., стр. 282-283.

5. Зырянов, А.В. Использование языка жестов для манипуляций с трёхмерными объектами в системах научной визуализации. // Параллельные вычислительные технологии 2009: Труды международной научной конференции. 30 марта - 3 апреля 2009 г., стр. 485-489.

6. Зырянов, А.В. Интерфейс на основе жестов для манипулирования трёхмерными виртуальными объектами и его применение в системах научной визуализации. // XI международный семинар по супервычислениям и математическому моделированию. 5 - 9 октября 2009 г. Тезисы докладов. стр. 70-71.

7. Зырянов А.В. Самокалибрующаяся масштабируемая система ввода трёхмерных жестов // Информационные технологии моделирования и управления, №1, 2010 г. стр. 42-49.

8. Авербух В.Л. Байдалин А.Ю., Бахтерев М.О., Васёв П.А., Зырянов А.В., Казанцев А.Ю., Манаков Д.В. К обоснованию проекта визуализационной компоненты виртуального испытательного стенда

// Параллельные вычислительные технологии (ПаВТ'2010): Труды международной научной конференции (Уфа 29 марта - 2 апреля 2010 г.). — Челябинск. Издательский центр ЮУрГУ. стр. 378-386

Подписано в печать 07.09.2010 г. Формат 60 × 84 × 16.
Бумага офсетная. Усл. печ. л. 1,5.
Заказ № 132. Тираж 100.

Отпечатано в типографии ИПЦ
«Издательство УрГУ».
г. Екатеринбург, ул. Тургенева, 4.