

Ушаков Д. М.

**Введение
в математические основы САПР
(курс лекций)**



Москва, 2011

УДК 32.973.26-018.2

ББК 004.438

У93

Ушаков Д. М.

У93 Введение в математические основы САПР: курс лекций.– М.: ДМК Пресс, 2011. – 208 с. : ил.

ISBN 978-5-94074-500-6

Книга представляет собой краткое изложение курса лекций «Введение в математические основы САПР», организованного Новосибирским государственным университетом при поддержке компании ЛЕДАС. Лекции рассчитаны на студентов старших курсов, специализирующихся в области прикладной математики, информатики и информационных технологий. Излагаемый материал может быть полезен разработчикам САПР, ученым, инженерам, а также всем интересующимся современными тенденциями в области автоматизации промышленных процессов.

УДК 32.973.26-018.2

ББК 004.438

Все права защищены. Любая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

Материал, изложенный в данной книге, многократно проверен. Но поскольку вероятность технических ошибок все равно существует, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. В связи с этим издательство не несет ответственности за возможные ошибки, связанные с использованием книги.

© Ушаков Д. М., 2011

ISBN 978-5-94074-500-6

© Оформление, ДМК Пресс, 2011

Содержание

Введение	10
Организатор курса	10
Целевая группа	10
Цели курса	11
Организация занятий	11
Структура курса	11
Благодарности	12
Лекций 1. Введение в САПР	13
Классы САПР	14
Автоматизация современного машиностроительного предприятия	14
Исторический обзор развития систем автоматизации проектирования	16
Функциональность CAD-систем	22
Современные CAD-системы и их классификация	25
Системы инженерного анализа (CAE)	26
Системы технологической подготовки производства (CAPP)	27
Системы автоматизации производства (CAM)	28
Системы управления данными об изделии (PDM)	28
Интегрированные пакеты управления жизненным циклом изделия	30
Вопросы для самоконтроля	31
Дополнительная литература	31
Лекция 2. Геометрическое моделирование	33
Автоматизация черчения и геометрическое моделирование ...	34
Виды геометрического моделирования	35
Функции твердотельного моделирования	37
Декомпозиционные модели	38
Конструктивные модели	39
Границные модели	40
Корректность граничных моделей	42

Пакеты геометрического моделирования	
и их функциональность	43
Вопросы для самоконтроля	44
Дополнительная литература	44

Лекция 3. Базовые геометрические объекты 45

Аффинное пространство и соглашение о нотации	46
Способы задания аналитических кривых и поверхностей	46
Изометрии аффинного пространства	48
Матричное представление трансформации в аффинном пространстве	49
Однородные координаты	50
Углы Эйлера	51
Экспоненциальное представление трансформации	52
Вопросы для самоконтроля	53
Дополнительная литература	54

Лекция 4. Инженерные кривые и поверхности ... 55

Кусочные кривые и их гладкость	56
Билинейный лоскут	56
Поверхности сдвига и вращения	56
Линейчатая поверхность	57
Лоскут Кунса	57
Эрмитова кривая, бикубическая поверхность и лоскут Фергюсона	58
Кривые и поверхности Безье	61
Алгоритм де Кастельжо	62
В-сплайны и В-сплайновые поверхности	63
Рациональные кривые и поверхности	64
Интерполяционные кривые и поверхности	65
Вопросы для самоконтроля	65
Дополнительная литература	66

Лекция 5. Обмен геометрическими данными 67

Стандарты обмена геометрическими данными	68
Формат IGES	68
Формат DXF	70

Формат STEP	70
Мозаичные модели	71
Формат STL	72
Формат VRML	73
Поверхности подразделения	73
Вопросы для самоконтроля	79
Дополнительная литература	79

Лекция 6. Вариационное моделирование: алгебраический подход 81

Параметры, ограничения и вариационные модели	82
Создание эскизов и проектирование сборок	82
Задача размещения геометрических объектов и ее характеристики	83
Вариационный геометрический решатель	84
Способы алгебраического моделирования геометрической задачи	85
Метрический тензор геометрической задачи	86
Методы символьного упрощения систем алгебраических уравнений	87
Декомпозиция Далмеджа–Мендельсона	88
Метод Ньютона–Рафсона	89
Решение систем линейных уравнений	91
Методы координатного и градиентного спуска	92
Вопросы для самоконтроля	93
Дополнительная литература	93

Лекция 7. Вариационное моделирование: диагностика и декомпозиция задачи 95

Диагностика геометрических задач	96
Методы упрощения геометрических задач	96
Определение и классификация методов декомпозиции	97
Граф ограничений	97
Методы рекурсивного деления	98
Методы рекурсивной сборки	99
Формирование кластеров с помощью анализа графа ограничений	100

Формирование кластеров на основе шаблонов	102
Эвристическое формирование псевдокластеров	103
Распространение степеней свободы	103
Вопросы для самоконтроля	103
Дополнительная литература	104
Лекция 8. Инженерия знаний в САПР	105
Параметрическое проектирование на основе конструктивных элементов	106
Инженерные параметры	108
Отношения базы знаний	109
Параметрическая оптимизация	110
Экспертные знания и продукционные системы	112
Вопросы для самоконтроля	113
Дополнительная литература	114
Лекция 9. Методы поиска и оптимизации решения	115
Задачи удовлетворения ограничениям и оптимизации в ограничениях в общей постановке, их связь	116
Классификация методов поиска и оптимизации решения	117
Метод координатного спуска	118
Метод градиентного спуска	118
Жадный алгоритм	119
Метод Ньютона	119
Методы перебора	120
Методы редукции областей	121
Метод ветвей и границ	123
Алгоритм модельной закалки	124
Генетические алгоритмы	125
Вопросы для самоконтроля	126
Дополнительная литература	126
Лекция 10. Инженерный анализ кинематики	127
Прямая и обратная задачи кинематики механизмов	128
Виды кинематических пар	128

Моделирование механизмов	131
Геометрические измерения	131
Моделирование задачи кинематики	132
Дифференциальное уравнение движения	133
Натуральный градиент уравнения	134
Алгоритмы численного решения дифференциальных уравнений	135
Планирование движения	136
Вопросы для самоконтроля	137
Дополнительная литература	138
Лекция 11. Инженерный анализ динамики	139
Задача анализа динамики механизмов	140
Движение абсолютно твердого тела в трехмерном пространстве	140
Моделирование контакта тел	142
Альтернативный подход: уравнения Лагранжа	143
Методы определения столкновений	145
Алгоритмы широкой фазы	145
Алгоритмы фазы сужения	147
Коммерческое программное обеспечение для симуляции движения	148
Вопросы для самоконтроля	148
Дополнительная литература	149
Лекция 12. Инженерный анализ методом конечных элементов	151
Конечно-элементный анализ	152
Введение в метод конечных элементов	152
Анализ упругости тела	152
Тензор деформаций	153
Тензор напряжений	154
Обобщенный закон Гука, матрицы жесткости и упругости	155
Уравнение равновесия тела под нагрузкой	157
Применение МКЭ для расчета малых напряжений тела под нагрузкой	157
Другие приложения МКЭ	159
Типы конечных элементов	159

Разбиения для МКЭ	160
Общая схема конечно-элементного анализа в САЕ-системах	161
Коммерческие пакеты конечно-элементного анализа	162
Вопросы для самоконтроля	162
Дополнительная литература	164
Лекция 13. Автоматизация производства	165
Архитектура станков с ЧПУ	166
Принципы программирования для станков с ЧПУ	167
Языки программирования высокого уровня для станков с ЧПУ ..	168
Генерация программ для станков с ЧПУ по CAD-моделям	170
Быстрое прототипирование и изготовление	171
Виртуальная инженерия	173
Вопросы для самоконтроля	173
Дополнительная литература	174
Лекция 14. Технологическая подготовка производства	175
Интеграция CAD и CAM	176
Задачи инженера-технолога	176
Модифицированный подход к технологической подготовке	177
Групповая технология	178
Классификация и кодирование деталей	178
Генеративный подход к технологической подготовке	180
Конструкторско-технологические элементы	181
Методы автоматического распознавания конструктивных элементов	182
Пример автоматического распознавания КТЭ	185
Вопросы для самоконтроля	185
Дополнительная литература	186
Лекция 15. Управление данными на протяжении жизненного цикла изделия	187
Системы управления данными об изделии	188
Цифровой макет изделия (DMU) и спецификация материалов (BOM)	188

Примеры PDM-систем	189
Программное обеспечение для организации бизнес-процессов	189
Из чего состоит PLM?	191
Интеграция PLM с системами управления отношениями с заказчиками	193
Интеграция PLM с системами управления цепочками поставок	194
Интеграция PLM с системами управления ресурсами предприятия	195
Практические подходы к интеграции систем PLM с CRM, SCM и ERP	197
Преимущества внедрения систем PLM	199
Вопросы для самоконтроля	200
Дополнительная литература	201
Краткий англо-русский словарь аббревиатур в области автоматизации проектирования и производства	202
Список литературы	205

Введение

Настоящая книга представляет собой второе издание оригинального одноименного курса лекций, который раз разработан автором для обучения студентов профильных специальностей математическим и информационным основам разработки систем автоматизации проектных работ и смежного программного обеспечения. Данный курс читается автором с 2005 года в Новосибирском государственном университете для студентов-магистрантов двух факультетов: механико-математического и информационных технологий.

Организатор курса

Курс лекций был подготовлен автором при поддержке компании ЛЕДАС, являющейся независимым производителем вычислительных программных компонентов для систем автоматизации проектирования и планирования. Компанией разработаны оригинальные технологии, основанные на программировании в ограничениях (научная область на стыке вычислительной математики и комбинаторики), которые широко применяются при производстве собственных и заказных программных продуктов, а также при оказании консультационных услуг. Среди клиентов ЛЕДАС – ведущие мировые и российские производители систем автоматизированного проектирования (CAD), систем подготовки производства (CAM), инженерного анализа (CAE), управления жизненным циклом изделия (PLM), а также проектного и ресурсного планирования.

Целевая группа

Курс ориентирован на студентов старших курсов университетов, специализирующихся в области прикладной математики, информатики и информационных технологий. От слушателей требуются базовые знания линейной алгебры, аналитической геометрии, теории графов, программирования. Для глубокого понимания излагаемого материала полезно знакомство с вычислительными методами решения систем алгебраических и дифференциальных уравнений, а также с алгоритмами из области исследования операций.

Цели курса

При составлении программы курса были приняты во внимание следующие цели:

- познакомить студентов с математическими основами современных САПР (систем автоматизации проектных работ, подготовки производства, инженерного анализа, управления жизненным циклом изделия);
- научить алгоритмам и методам, применяемым при решении типичных задач автоматизации проектирования;
- ввести в проблематику создания современных вычислительных компонентов для САПР;
- подготовить студентов к участию в промышленной разработке вычислительных модулей для САПР.

Организация занятий

Курс организован в виде еженедельных лекций (по два академических часа каждая), читаемых на протяжении одного семестра (всего 15 лекций). Возможна организация практикума на персональном компьютере с использованием образовательных лицензий на один из современных САПР-пакетов. Студенты, успешно прослушавшие настоящий курс и подтвердившие свои знания на экзамене, могут расчитывать на прохождение преддипломной практики и последующее трудоустройство в софтверных компаниях, занимающихся производством научноемкого программного обеспечения для автоматизации проектирования и планирования производства.

Структура курса

Курс состоит из пятнадцати лекций. Первая лекция посвящена обзору современного состояния программных систем, традиционно относимых к классу САПР. Следующие четыре лекции описывают геометрические основы систем автоматизированного проектирования. Материал шестой и седьмой лекций содержит описание различных алгоритмов, используемых при решении задач вариационного проектирования. Инженерные инструменты САПР и алгоритмы решения соответствующих задач рассматриваются в восьмой и девятой лекци-.

ях. Десятая, одиннадцатая и двенадцатая лекции представляют математический аппарат систем инженерного анализа. Тринадцатая описывает математический аппарат, используемый для работы со станками ЧПУ, а четырнадцатая посвящена математическим основам систем технологической подготовки производства. Последняя, пятнадцатая лекция посвящена системам интеграции данных об изделии, используемых на протяжении его жизненного цикла. В конце каждой лекции приводится список вопросов для самоконтроля, а также рекомендации по дополнительному чтению.

Благодарности

Идея издания настоящей книги принадлежит генеральному директору ЗАО ЛЕДАС Давиду Яковлевичу Левину, который также взял на себя руководство процессом ее издания и поддерживал автора на всех этапах подготовки текста. Корректура текста, осуществленная Людмилой Александровной Каревой, позволила избежать многих ошибок. Автор выражает благодарность своим коллегам, которые прочитали предварительные версии настоящего курса лекций и любезно указали на пробелы и недостатки в его структуре и содержании. Автор также признателен всем читателям настоящего текста за возможные замечания, исправления, пожелания по излагаемому материалу, которые он с благодарностью примет по e-mail ushakov@ledas.com.

Лекция 1

Введение в САПР

Классы САПР	14
Автоматизация современного машиностроительного предприятия	14
Исторический обзор развития систем автоматизации проектирования	16
Функциональность CAD-систем	22
Современные CAD-системы и их классификация	25
Системы инженерного анализа (CAE)	26
Системы технологической подготовки производства (CAPP)	27
Системы автоматизации производства (CAM)	28
Системы управления данными об изделии (PDM)	28
Интегрированные пакеты управления жизненным циклом изделия	30
Вопросы для самоконтроля	31
Дополнительная литература	31

Классы САПР

За русским термином САПР (Система Автоматизации Проектных Работ) скрывается несколько классов программных систем, имеющих отношение к автоматизации труда инженеров, конструкторов и технологов. Каждый из классов имеет устоявшуюся трехбуквенную английскую аббревиатуру:

- двумерное черчение и трехмерное геометрическое проектирование (CAD);
- инженерный анализ (CAE);
- технологическая подготовка производства (CAPP);
- автоматизация производства (CAM);
- управление данными об изделии (PDM);
- управление жизненным циклом изделия (PLM).

Кроме того, к САПР относятся программы для автоматизации труда архитекторов и строителей, топографов и геологов, которые, однако, остаются за рамками данного курса. В фокусе нашего внимания будут «механические» САПР (MCAD), используемые машиностроительными предприятиями и конструкторскими бюро. Механические САПР являются одними из исторически первых программ для ЭВМ, занимая в настоящее время около 3% мирового рынка программного обеспечения. Без систем САПР невозможно представить себе ни одно современное производственное предприятие аэрокосмической, автомобильной, судостроительной, электронной и других отраслей промышленности, включая производство потребительских товаров.

Автоматизация современного машиностроительного предприятия

Для четкого понимания излагаемых в рамках данного курса концепций автоматизации различных процессов, связанных с жизненным циклом изделия, рассмотрим сначала типичную схему организации производства современного машиностроительного предприятия. Как правило, любое предприятие специализируется на производстве конкретных типов изделий. У предприятия обязательно имеется отдел маркетинга, который проводит рыночные исследования, ведет работу с потенциальными и реальными клиентами и добивается заключения контрактов на производство и поставку партии изделий. Получив конкретный заказ, отдел маркетинга передает его главному

инженеру предприятия, который должен оценить, можно ли в принципе выполнить этот заказ на технологической базе предприятия. Для этого он поручает конструкторскому отделу подготовить проект изделия. В большинстве случаев инженер-конструктор имеет дело с проектированием изделия, которое конструктивно похоже на выполненные ранее работы (так как обычное предприятие специализируется на каком-то одном типе изделий). Поэтому он сначала находит похожий проект среди работ, выполненных в конструкторском отделе ранее, копирует его и вносит требуемые изменения, а затем передает обратно главному инженеру. Далее проект попадает в технологический отдел, где осуществляется составление проектного плана – последовательности операций, которые необходимо выполнить в цехах предприятия для производства изделия (обработки деталей и их сборки). Третий этап – проверка наличия на складе всех необходимых комплектующих (заготовок или готовых деталей от предприятий-смежников) и заказ недостающих частей при необходимости. Четвертый – собственно производство партии изделий, как правило, связанное с изготовлением отдельных деталей на станках и прессах (при использовании станков с числовым программным управлением требуется их соответствующим образом перепрограммировать), а также сборка конечного изделия (при использовании конвейерных линий с роботами-сборщиками требуется переналадить их на сборку модифицированного изделия). Пятый этап – контроль качества, шестой – упаковка изделий; а завершается все поставкой изделий заказчику, организацией их послепродажного обслуживания и – при необходимости – утилизацией.

Таким образом, любое современное производство, даже весьма скромных объемов, требует наличия квалифицированного персонала, выполняющего большое количество самых разных интеллектуальных операций, сопровождающихся значительным документооборотом. Излишне говорить, что трудоемкость создания каждого документа (как правило, это чертежи изделий и техническая документация к ним) традиционными методами (с использованием чертежной доски – кульмана) невероятно велика. В условиях усиливающейся конкуренции (связанной с постоянным ростом экономики, с открытием рынков, с глобализацией в масштабах всей планеты) выживают только те предприятия, которые способны быстро и адекватно отвечать на постоянное изменение рыночных условий, то есть минимизировать время выполнения любого заказа. Для помощи таким предприятиям и были разработаны интегрированные программные системы

автоматизации, ускоряющие каждую область деятельности по отдельности и в то же время связывающие их между собой в рамках одной информационной системы предприятия.

Исторический обзор развития систем автоматизации проектирования

Развитие систем автоматизации проектирования происходило в тесном сотрудничестве научных лабораторий, военных ведомств и промышленных предприятий. История этого развития включала в себя несколько ключевых событий, которые можно объединить по десятилетиям:

1950-е годы. Создание станков с числовым программным управлением (ЧПУ)

1952. В Массачусетском технологическом институте (Massachusetts Institute of Technology, MIT) создан первый фрезерный станок с ЧПУ.

1957. Система PRONTO – первое коммерческое ПО для управления станками с ЧПУ.

1960-е годы. Системы компьютерной графики и системы автоматизации черчения

1963. Айван Сазерленд (Ivan Sutherland) из MIT создал программу SKETCHPAD, которая намного опередила свое время и теперь считается первой системой автоматизации черчения (рис. 1).



Рис. 1

1964. Американские математики Фергюсон (J.C. Ferguson) из Boeing и Кунс (Steven A. Coons) из МИТ предлагают различные способы задания параметрических поверхностей, обладающих определенными геометрическими свойствами. Сотрудник General Motors де Бур (C. de Boor) впервые использует для инженерных целей понятие B-сплайна, предложенное еще в 1948 г. Пару лет спустя французские математики Безье (Pierre Bézier) и де Кастельжо (Paul de Casteljau), работающие на конкурирующие компании Renault и Citroën, независимо изобрели аппарат для построения инженерных кривых и поверхностей по контрольным точкам, который лег в основу современного поверхностного моделирования.

1965. В Computer Laboratory Кембриджского университета создается CAD Group. Эта команда ученых, возглавляемая Чарльзом Лангом (Charles Lang), проводит исследования в области создания программных средств, лежащих в основе MCAD/CAM. Вскоре к CAD Group присоединяется Ян Брэйд (Ian Braid), который разрабатывает экспериментальную систему BUILD, систему геометрического моделирования на основе революционной для того времени технологии – граничного представления (BRep).

1965. Ведущие машиностроительные корпорации (Lockheed и McDonnell) создают первые коммерческие CAD/CAM-системы, а также системы анализа методом конечных элементов.

1967, 1969. Создание первых софтверных компаний, производящих САПР: американских SDRC и Computervision. Их продукты – I-DEAS и CADSS (а позднее и Windchill) на долгие годы становятся стандартом САПР. Позднее обе компании были поглощены новыми лидерами рынка – с 1998 г. Computervision принадлежит Parametric Technology Corporation, а в 2002 г. SDRC была куплена EDS и объединена с Unigraphics.

1970-е. Первые 3D-системы

1974. Ведущие участники CAD Group в Кембридже образуют компанию Shape Data Ltd., которая начинает разработку коммерческого геометрического ядра ROMULUS (на языке Fortran), основанного на идеях, обкатанных в экспериментальном ядре BUILD. Вскоре продается первая коммерческая лицензия на ROMULUS. Покупатель – компания НР – использует ее для создания своей CAD-системы ME30 (наследником которой является CoCreate One-Space Modeler).

1974. Выступление американского художника-дизайнера Чайкина (G. Chaikin) на конференции CAGD в университете Юты с пред-

ствленным им алгоритмом быстрой генерации кривой заданной формы породило новую область исследований – теорию поверхностей подразделения; в настоящее время все известные способы компактного представления трехмерных моделей основаны на этой теории.

1977. Французская авиастроительная компания Avions Marcel Dassault (ныне Dassault Aviation) создает систему трехмерного проектирования CATIA (Computer-Aided Three-dimensional Interactive Application; позднее для разработки следующих версий системы будет создана специальная софтверная компания – Dassault Systèmes), остающуюся до сих пор непревзойденной по своим уникальным возможностям поверхностного моделирования, широко используемым ведущими предприятиями авиакосмической и автомобильной промышленности во всем мире.

1979. Boeing, General Electric и другие компании разрабатывают первый стандарт для обмена инженерными геометрическими данными – формат IGES (Initial Graphic Exchange Standard), который включает в себя спецификацию NURBS кривых и поверхностей (конические сечения, кривые Безье и В-сплайны являются частными случаями NURBS).

1980-е. Первые системы твердотельного моделирования для UNIX, первые программы автоматизации черчения для PC

1980. Подразделение американского авиастроительного концерна McDonnell Douglas выпускает первую в мире коммерческую систему твердотельного моделирования – Unigraphics. В 1991 г. этот бизнес покупает компания EDS. Спустя еще 10 лет EDS приобретает компанию SDRC, объединяет ее линейку продуктов с Unigraphics и продает объединенную компанию частным инвесторам под именем UGS. Наконец, в 2007 г. компанию UGS поглощает немецкий концерн Siemens.

1980. Французская компания Matra Datavision выпускает САПР EUCLID, который по мере развития превращается из CAD-системы в интегрированный CAD/CAM/CAE-пакет. В 1998 г. Matra продает этот бизнес Dassault Systèmes.

1982. Создание компании Autodesk и выпуск ее первого продукта AutoCAD – первой системы автоматизации черчения для персональных компьютеров, которая долгие годы оставалась (а по некоторым данным остается до сих пор) самой популярной САПР в мире.

1983. Начало работы над международным стандартом обмена CAD-данными – STEP, призванного заменить IGES.

1985. Компания Shape Data начинает разработку пакета Parasolid – прямого наследника геометрического ядра ROMULUS. В этом же году ведущие сотрудники Shape Data оставляют компанию и создают собственную под названием Three-Space Ltd., которая начинает разработку принципиально нового геометрического ядра – ACIS – совместно с американской компанией Spatial Technology (спустя 15 лет поглощенной Dassault Systèmes). Компания Shape Data через три года поглощается Unigraphics вместе с ее разработкой Parasolid.

1985. Эмигрировавший из СССР в США профессор Ленинградского университета Семен Петрович Гейзберг основывает компанию PTC (Parametric Technology Corp.). Выпущенный два года спустя продукт Pro/ENGINEER становится первой в мире системой параметрического проектирования на основе конструктивных элементов.

1987. Компания 3D Systems выпускает первые станки для быстрого прототипирования изделий путем изготовления их копий из пластика методом стереолитографии.

1989. Компания Deneb Robotics (ныне поглощенная Dassault Systèmes) выпускает первую в мире программу, моделирующую движения человека при работе за станком, положившую начало эргономическому анализу в САПР.

1989. Джон Оуэн (John Owen) создает в Кембридже компанию D-Cubed Limited, которая занимается разработкой геометрических программных библиотек, в частности, вычислительных модулей для параметризации двумерных и трехмерных геометрических моделей. Модули D-Cubed используют практически все разработчики САПР. В 2004 г. компанию поглощает UGS (ныне Siemens PLM Software).

1989. Создается первая российская софтверная компания, разрабатывающая САПР – АСКОН (Санкт-Петербург) с продуктом КОСМОС. В настоящее время АСКОН является ведущим отечественным поставщиком решений для конструкторско-технологической подготовки производства и управления жизненным циклом изделия.

1990-е. Полноценные САПР на платформе Windows

1991. Компания Autodesk лицензирует геометрическое ядро ACIS у Spatial Technologies для реализации элементарных функций твердотельного моделирования в AutoCAD (а затем – также в пакетах Mechanical и Inventor).

1992. Выпускниками МГТУ «СТАНКИН» создана компания Топ Системы, занимающаяся разработкой линейки САПР-решений Т-FLEX (на основе ядре Parasolid).

1993. Джон Хирштик (John Hirschtick) из компании Computervision вместе с Майклом Пэйном (Michael Payne) из компании PTC основывают собственную компанию – SolidWorks, вскоре поглощенную Dassault Systèmes; ныне одноименная САПР SolidWorks (основанная на геометрическом ядре Parasolid) является одной из самых популярных в мире систем трехмерного проектирования.

1996. Компания Intergraph выпускает Solid Edge – трехмерную САПР для платформы Windows NT на геометрическом ядре ACIS. Два года спустя права на Solid Edge перекупила компания UGS, которая перевела систему на собственное ядро Parasolid.

1998. PTC поглощает компанию Computervision и выпускает Windchill, систему управления жизненным циклом изделия в среде Интернет.

1999. На основе успеха программы AutoCAD в области автоматизации черчения компания Autodesk создала трехмерную САПР Inventor для платформы Windows на основе лицензированного геометрического ядра ACIS, которая в настоящее время составляет серьезную конкуренцию другим популярным САПР среднего уровня – SolidWorks и Solid Edge.

2000-е. Системы для управления жизненным циклом изделия (PLM)

2000. Бурно растущий рынок средств управления жизненным циклом изделия (включая средства конструирования, инженерного анализа, подготовки производства, управления данными и организации совместной работы) привлекает внимание мирового лидера в области корпоративных программных решений – немецкую компанию SAP, которая выпускает специальный модуль в рамках своего ERP-портфеля.

2000. После продажи своего бизнеса по разработке САПР EUCLID в Dassault Systèmes, компания Matra Datavision решает открыть исходный код геометрического ядра, использовавшегося при разработке этой системы, и предоставить его в свободное использование всем желающим. Коммерческое обслуживание компаний, занимающихся разработкой САПР на основе этого ядра, осуществляется специально созданная компания Open CASCADE. S.A.S.

2002. После слияния компаний Unigraphics и SDRC объединенная линейка САПР-продуктов получает название NX.

2003. PTC выпускает новое поколение своей САПР Pro/ENGINEER под названием Wildfire. Система включает в себя значительно переработанный пользовательский интерфейс и является полностью интегрированной в среду для управления жизненным циклом изде-

лия на основе web-сервисов. Гибкая ценовая политика позволяет Wildfire конкурировать не только с системами верхнего (CATIA, NX), но и среднего (SolidWorks, Inventor, Solid Edge) уровня.

2004. Число установленных по всему рабочих мест MCAD-систем достигает 5 миллионов. Самыми популярными семействами продуктов являются Pro/ENGINEER, CATIA, NX, Mechanical Desktop, Inventor, SolidWorks и Solid Edge.

2004. Отечественная компания ЛЕДАС выпускает геометрический решатель LGS, который используется для реализации параметрической функциональности в САПР. Первыми клиентами становятся компании Proficiency (Израиль) и ADEM (Россия).

2007. Майкл Пэйн (основатель PTC и SolidWorks) создает новую компанию SpaceClaim. Одноименный продукт позиционируется не как конкурент существующим системам механического проектирования, а как полезное дополнение к ним, основанное на возможности прямого редактирования геометрии модели без истории построения (информации о конструктивных элементах).

2007. PTC поглощает компанию CoCreate, бывшее подразделение Hewlett Packard, разрабатывающее одноименную САПР на основе методов прямого моделирования.

2007. Немецкий концерн Siemens поглощает компанию UGS и объявляет о своем намерении выйти на рынок решений для управления жизненным циклом изделия.

2007. Oracle поглощает компанию Agile, известного поставщика решений для управления жизненным циклом изделий, и начинает конкурировать на этом рынке с SAP, альянсом IBM/Dassault, PTC и Siemens.

2008. После десятилетней серии крупных поглощений (SolidWorks, Deneb, Smart Solutions, Spatial, ABAQUS, MatrixOne) Dassault Systèmes объявляет о запуске принципиально новой платформы PLM V6, в рамках которой компания собирается реализовать концепцию PLM 2.0, означающую, что все услуги по разработке изделий и управления их жизненным циклом будут доступны в сети для совместной работы с удаленным доступом в режиме реального времени.

2008. Siemens PLM Software (бывшая UGS) объявляет о разработке нового поколения средств трехмерного моделирования на основе синхронной технологии, которая в рамках которой конструктор может одновременно работать как с конструктивными элементами, так и напрямую с ее граничными элементами (методом прямого редактирования).

Функциональность САД-систем

Современные системы проектирования предлагают следующую базовую функциональность:

- проектирование деталей (part design);
- проектирование сборок деталей и механизмов (assembly design);
- специальное проектирование (пресс-формы для изделий из листового металла, формы для литья для изделий из пластмасс, прокладка трубопроводов, расчет электрических схем и пр.);
- генерация чертежей (drafting);
- создание трехмерной модели по чертежу;
- расчеты инженерных параметров и их оптимизация.

Разберем каждый из этих наборов функций подробнее. Детальное проектирование используется при разработке геометрических моделей трехмерных деталей. Основным концептуальным подходом к детальному проектированию в современных САД-системах является *параметрическое моделирование на основе конструктивных элементов* (parametric feature-based design). Этот метод позволяет максимально упростить как собственно процесс проектирования новой детали, так и внесение изменений в существующую. Как правило, создание любого трехмерного *конструктивного элемента* (feature), такого как отверстие, полость, скругление, включает в себя рисование эскиза плоского профиля, поэтому подпрограмма двумерного эскизного черчения (называемая sketcher) является составной частью модуля детального проектирования. При создании эскиза конструктивного элемента важно правильно задать *геометрические ограничения* (constraints), чтобы изменение размеров приводило к предсказуемым изменениями геометрии (например, прямоугольник при изменении длины одной из сторон должен оставаться прямоугольником, то есть длина противоположной стороны тоже должна автоматически измениться). Важной функцией систем эскизного черчения является помочь пользователю в наложении необходимых ограничений на геометрию, а также выделение разными цветами недо- и переопределенных частей эскиза. При создании конструктивного элемента по эскизу указываются размеры элемента: расстояние от плоскости эскиза, углы и т. п. Конфигурации (библиотеки) типовых деталей, возможность работы с которыми предоставляют современные САД-системы, значительно сокращают время проектирования.

Альтернативой параметрическому моделированию на основе конструктивных элементов является метод прямого, или динамического, моделирования. В рамках этого подхода объем создается и вычитается с помощью операции вытягивания (push-and-pull) замкнутого плоского профиля, а также ряда похожих операций. Ключевым моментом является отсутствие информации об истории построения формы, что подразумевает прямое управление ее граничными элементами (гранями, ребрами, вершинами) с помощью перемещения их в пространстве или задания геометрических ограничений между ними. При таком походе становится возможной параметрическая модификация деталей без истории построения (обычно история построения – дерево конструктивных элементов – теряется при импорте модели из одного САПР-пакета в другой), но снижается уровень заложенных в модели знаний (*намерений проектировщика*). В последнее время на рынке стали появляться системы, в которых параметрическое моделирование на основе конструктивных элементов можно сочетать с прямым моделированием – даже в рамках работы с одной деталью. Соответствующая концепция называется *синхронной технологией*.

Говоря о проектировании сборок деталей и механизмов, различают два подхода – нисходящий и восходящий. *Нисходящий подход* подразумевает проектирование механизма с нуля. Когда в механизм необходимо добавить очередную деталь, вызывается модуль детального проектирования, причем размеры создаваемой детали согласуются с размерами уже созданных частей механизма. При *восходящем проектировании* механизм собирается из ранее спроектированных деталей, которые позиционируются по отношению друг к другу с помощью *ограничений сборки* (соосность, инцидентность и пр.). Важной функцией модуля проектирования сборок является возможность расчета степеней свободы деталей в механизме и их динамического перемещения в соответствии с наложенными ограничениями. Это позволяет оценить кинематику будущего изделия еще на этапе проектирования. Библиотеки стандартных деталей (крепежи, трубы, шестерни, подшипники) позволяют пользоваться готовыми элементами при проектировании механизма. Важной характеристикой модуля проектирования сборок является его производительность при больших сборках (состоящих из десятков тысяч деталей).

Модули для специального проектирования предоставляют инструменты, характерные для конкретной предметной области. Например, при проектировании электрических схем инженер имеет дело

с набором примитивов, моделирующих печатные платы, микросхемы и пр. Алгоритмы, которые используются для автоматизации подобного проектирования, тоже являются предметно-ориентированными. Типичными модулями специального проектирования являются средства для проектирования сварочных конструкций и моделирования разводки.

Генерация чертежей трехмерной детали по сей день остается востребованной функциональностью в CAD-системах. Отметим, что данная функция выполняется в автоматическом режиме – система сама определяет ключевые размеры детали, исходя из ее геометрической модели, и помещает их на чертеж. Файлы с чертежными данными являются ассоциативными по отношению к файлу трехмерной модели. Это значит, что при изменении модели (например, в модуле детального проектирования) и при последующей загрузке созданного ранее файла чертежа, последний автоматически обновится в соответствии с трехмерной моделью.

На машиностроительных предприятиях обычно имеется обширная база спроектированных ранее изделий. Если они проектировались на кульмане или с помощью систем автоматизации черчения, встает задача получения трехмерной геометрической модели изделия (например, для внесения в нее продиктованных временем изменений или интеграции с другими деталями в общий механизм). Соответствующий модуль современных CAD-систем позволяет удобно ввести эти чертежи в систему (импортируя файлы чертежей или их отсканированные изображения), внести в них необходимые изменения и автоматически построить трехмерную геометрическую модель.

При проектировании изделия необходимо учитывать не только его геометрические параметры (форму и характерные размеры), но и физические характеристики (такие как площадь поверхности, объем, масса, центр тяжести). Кроме того, конструктивно похожие части одного или разных изделий удобно не проектировать каждый раз заново, а вставлять в модель, используя параметры. Все эти функции (создание параметров и связывание их друг с другом посредством различных отношений) выполняет инженерный модуль CAD (knowledgeware). К нему же примыкает модуль оптимизации, позволяющий, например, минимизировать массу изделия, расход материалов, улучшить его динамические характеристики и т. п.

Современные CAD-системы и их классификация

Современные MCAD-системы по набору предлагаемой функциональности и стоимости лицензий традиционно разделяются на три уровня. Верхний уровень образуют пакеты CATIA (производства Dassault Systèmes), NX (Siemens PLM Software), Pro/ENGINEER (PTC). Все эти системы (или их идеальные предшественники) появились еще в 1980-х годах и были ориентированы прежде всего на рабочие станции в среде UNIX (Sun Solaris, HP-UX, IBM AIX, Silicon Graphics IRIX). С появлением в конце 1990х годов мощных персональных компьютеров были созданы Windows-версии САПР верхнего уровня. Каждый из больших САПРов представляет собой конфигурируемый программный пакет, в который входят сотни различных наборов инструментальных средств (не только CAD, но и CAE, CAM, PDM). Эти пакеты, как правило, ориентированы на совместную работу нескольких пользователей и требуют предварительного обучения персонала предприятия. Типичная цена лицензии на одно рабочее место составляет десять–двадцать тысяч долларов США, не считая стоимости оборудования. Внедрять большие САПР лучше всего в комплексе с другими программами управления жизненным циклом изделия от того же производителя.

Средний уровень САПР для машиностроения – это такие популярные системы, как SolidWorks (Dassault Systèmes), Solid Edge (Siemens PLM Software), Autodesk Inventor. Менее известны такие системы как CoCreate (PTC), Keycreator (Kubotek), SpaceClaim. Российские разработки, такие как T-FLEX (Топ-системы), ADEM (ADEM Technologies), КОМПАС (АСКОН) вполне в состоянии конкурировать по своей функциональности с популярными системами среднего уровня. Все эти системы работают только под управлением Microsoft Windows, отличаются более скромной функциональностью (по сравнению с системами верхнего уровня), но являются значительно более доступными по цене (для типичной конфигурации три–пять тысяч долларов США за рабочее место). Тем не менее, САПР среднего уровня – вполне достойные программы, которые в некоторых аспектах реализации базовой функциональности порой даже превосходят САПР верхнего уровня. Объясняется это тем фактом,

что САПР среднего уровня, как правило, строятся из готовых блоков – геометрических ядер, расчетных пакетов, средств визуализации – от третьих производителей. Производители таких блоков-полуфабрикатов накопили значительный опыт и обширную клиентскую базу, что объясняет достаточно высокое качество соответствующего программного обеспечения. Все это обуславливает высокую популярность САПР среднего уровня (по количеству проданных лицензий лидируют именно они). Основные потребители данных программ – небольшие производственные предприятия и опытно-конструкторские бюро.

Нижний уровень САПР представляют системы AutoCAD (Autodesk), bCAD (разработка новосибирской компании ПроПро) и др. Как правило, это системы двумерного черчения (как AutoCAD) либо трехмерного моделирования с очень ограниченной функциональностью, ориентированной скорее на графическую визуализацию, чем на реальное проектирование (как bCAD). Тем не менее, у этих классов систем есть свой немаленький рынок, который, впрочем, уступает рынку систем верхнего и среднего уровня.

К пакетам САПР примыкают также коммерческие программы для геометрического моделирования и обмена геометрическими данными. Мы их разберем ниже в соответствующих разделах нашего курса.

Системы инженерного анализа (САЕ)

Системы инженерного анализа (другое название – системы автоматического конструирования) предназначены для изучения поведения продукта с использованием его виртуального (хранящегося только в памяти компьютера, но не воплощенного ни в каком материале) макета. Именно благодаря развитым САЕ-системам, первый же собранный в реальном цехе самолет не только взлетает и демонстрирует все заложенные его проектировщиками характеристики, но является настолько безупречным, что тут же поставляется заказчику. Типичными видами инженерного анализа являются:

- анализ кинематики изделия – расчет траекторий движущихся частей и их визуализация на компьютере;
- анализ динамики изделия – расчет поведения изделия в реальном времени с учетом действующих на него физических сил, взаимодействия механизмов и пр.;
- расчет статических напряжений, магнитного поля, температур, определение критических нагрузок;
- имитация работы электронных цепей.

Для статического анализа, а также для анализа динамики, связанного с деформацией изделия, широко используется *метод конечных элементов*. Системы САЕ не могут работать без геометрической модели изделия – как правило, такая модель создается в системе CAD, а затем импортируется в САЕ. Следует также отметить, что многие CAD-системы верхнего уровня (такие как CATIA) уже содержат в себе базовые средства инженерного анализа, поэтому использование специализированной САЕ-системы для пользователей таких CAD-систем необязательно.

Для моделирования кинематических и динамических аспектов изделия, не требующих расчетов деформаций и напряжений, могут использоваться такие коммерческие пакеты, как ADAMS (MSC.Software) и DADS (LMS). Для расчетов методом конечных элементов широко применяются системы ANSYS (ANSYS, Inc.), NASTRAN (MSC.Software) и ABAQUS (приобретя эту компанию, Dassault Systèmes объявила о создании нового САЕ-бренда SIMULIA). Примером автономной САЕ-системы отечественного производства служит АРМ WinMachine (НТЦ АПМ).

Системы технологической подготовки производства (САПР)

Системы технологической подготовки производства в традиционном понимании – это прежде всего программы для работы с базой данных технологических планов предприятия. Каждый такой план связывает с определенным изделием цепочку технологических процессов, применяемых при производстве на данном предприятии (например, последовательность обработки детали на разных станках, сборка механизма на конвейере и т. п.). При необходимости организации на предприятии производства нового изделия прежде всего с помощью САПР-системы осуществляется поиск технологического плана подобного изделия, выпускавшегося на предприятии раньше. Затем этот план корректируется для нужд конкретного нового изделия и сохраняется в базе данных как новый план. Всю эту функциональность и обеспечивают САПР-системы, основанные на модифицированном подходе. Генеративный подход к технологической подготовке производства заключается в автоматическом распознавании в геометрической модели детали типовых конструкторско-технологических элементов и ассоциированием с ними типовых техпроцессов.

В ряду современных коммерческих САПР-систем прежде всего стоит отметить CAM-I CAPP, MIPLAN, MetCAPP, ICEM-PART. Из отечественных разработок заслуживают внимания системы ТехноПро и TechnologiCS.

Системы автоматизации производства (САМ)

Системы САМ предназначены для создания программ обработки деталей на станках с числовым программным управлением (ЧПУ), а также программ управления роботизированными сборочными линиями. Конечно, создание управляющих программ происходит в таких системах по большей части автоматически – с использованием информации о геометрической модели детали и о положении деталей в сборке. Важной особенностью САМ-систем являются встроенные средства проверки корректности сгенерированных программ, для чего используются два основных подхода. Первый подход состоит в визуализации процесса работы станка на экране компьютера (пользователь может видеть работу металлорежущего инструмента или сборочных роботов и оценить корректность траекторий их движения). Второй способ – это моделирование процесса получения детали из заготовки и сравнение геометрии полученных в результате обработки поверхностей с данными, хранящимися в геометрической модели.

Как правило, соответствующая функциональность существует во многих интегрированных CAD/CAM пакетах, включая CATIA, SolidWorks, T-FLEX и др. Из независимых решений третьих производителей (не поставляющих собственно CAD программы) отметим Mastercam (CNC Software), SURFCAM (Surfware), EdgeCAM (Pathtrace), CimatronE (Cimatron) и продукты компании Delcam. Среди отечественных разработок заслуживает упоминания ГеММа-3Д (НТЦ ГеММа).

Системы управления данными об изделии (PDM)

Данный подкласс САПР образуют системы, интегрирующие в себе доступ к самым разнотиповым данным, необходимым для работы с изделием на всех этапах его жизненного цикла: во время маркетин-

говых исследований, планирования, проектирования, производства, контроля качества, упаковки, доставки, послепродажного обслуживания и утилизации. Как правило, все эти действия выполняются на предприятии сотрудниками разных отделов с помощью различных систем автоматизации. Поэтому системы PDM в первую очередь упрощают передачу данных между отделами и доступ к информации, необходимой для работы в разных системах. Их использование на предприятии:

- улучшает взаимодействие;
- уменьшает бумажный документооборот;
- повышает эффективность управления.

Как правило, одна PDM-система поддерживает работу с моделями, созданными в разных системах проектирования. Стандартом де-факто для таких систем стало наличие у них web-интерфейса, что делает их аппаратно независимыми. Еще одним важным моментом PDM-систем является наличие у них программных интерфейсов для подключения к системам ERP, SCM и CRM, что позволяет выполнять полный спектр операций планирования и доступа к различным данным предприятия непосредственно из системы PDM.

Из коммерческих PDM-пакетов отметим прежде всего три системы, объединенные под брендом ENOVIA (VPLM, SmarTeam и MatrixOne). Эти системы производятся Dassault Systèmes, но позиционируются на рынке по-разному. SmarTeam – это универсальная PDM-система, которая может использоваться на предприятии совместно с CAD-системами CATIA или SolidWorks, а также любыми решениями сторонних производителей. Система имеет web-интерфейс и может быть интегрирована с произвольным пакетом ERP. ENOVIA VPLM позиционируется как глубоко интегрированная с CATIA система PDM, предназначенная для управления данными об изделиях, состоящих из большого количества деталей (таких как самолеты или автомобили). Пользовательский интерфейс ENOVIA выполнен в том же графическом стиле, что и у систем CATIA и DELMIA, что облегчает обучение работе с системой пользователям, уже знакомым с другими продуктами Dassault Systèmes. Наконец, ENOVIA MatrixOne ориентирована на совместную работу огромного числа пользователей по всему миру. В настоящее время Dassault Systèmes занимается интеграцией этих систем на основе одной платформы. Из PDM-решений других производителей отметим Teamcenter (Siemens PLM Software) и Winchill (PTC).

Интегрированные пакеты управления жизненным циклом изделия

Автоматизация различных областей деятельности производственно-го предприятия, осуществленная с помощью CAD/CAE/CAPP/CAM-систем, а также необходимость организации хранения проектных данных в общей базе (осуществленная с помощью PDM-систем) привели ведущих разработчиков САПР к мысли, что все эти системы могут быть связаны в единый комплекс программных решений от одного поставщика. В принципе, работать с системами от одного производителя, которые имеют одинаковый пользовательский интерфейс, покупаются и обслуживаются в одном месте, выгодно и предприятиям-потребителям. Поэтому неудивительно, что в начале 2000-х годов появилась PLM (Product Lifecycle Management) – концепция управления жизненным циклом изделия, которая тут же была воплощена в линейке продуктов ведущих поставщиков CAD/CAE/CAPP/CAM/PDM-решений, прежде всего – Dassault Systèmes, Siemens PLM Software и PTC. Тесная интеграция программных решений в рамках PLM-линейки вынуждает компании, работающие на этом рынке, проводить агрессивную политику по покупке более мелких специализированных компаний. Например, французская компания Dassault Systèmes за последние десять лет произвела ряд громких поглощений таких известных компаний, как SolidWorks, Deneb Robotics, Smart Solutions, Spatial, ABAQUS, MatrixOne и др.

Говоря о современных системах управления жизненным циклом изделия, прежде всего стоит упомянуть интегрированные пакеты от ведущих производителей. Альянс Dassault Systèmes/IBM продвигает программный комплекс V5 PLM Solutions (недавно было объявлено о выходе версии V6), состоящий из систем CATIA, DELMIA, ENOVIA, SIMULIA. Недавно было продекларировано расширение этого набора новой системой 3DVIA для массового использования трехмерных данных в среде Интернет. Компания Siemens PLM Software строит свою PLM-линейку продуктов вокруг PDM-системы Teamcenter, с которой взаимодействуют решения для разработки семейства NX. PLM-решения от Parametric Technology Corp. конфигурируются с помощью системы Product Development System (PDS) и включают в себя продукты семейств Pro/ENGINEER Wildfire и Windchill.

Вопросы для самоконтроля

1. Опишите программное обеспечение, относящееся к классу САПР. Какова его доля на современном рынке ПО?
2. Опишите типичную схему автоматизации современного машиностроительного предприятия.
3. Укажите ключевые этапы в истории развития САПР.
4. Какова базовая функциональность систем механического проектирования?
5. Опишите концепцию параметрического проектирования на основе конструктивных элементов.
6. В чем отличия восходящего и нисходящего методов проектирования механизмов?
7. Как классифицируются современные CAD-системы? Назовите примеры в каждом классе.
8. Опишите функциональность систем инженерного анализа и приведите примеры таких систем.
9. Их чего складывается функциональность систем технологической подготовки производства? Приведите примеры систем САПР.
10. Для чего предназначены системы автоматизации производства? Приведите примеры САМ-систем.
11. Какие задачи решают системы управления данными об изделии? Приведите примеры коммерческих систем PDM.

Дополнительная литература

Первые две главы книги [8] являются хорошим дополнением к материалу настоящей лекции. Полезно ознакомиться также с главой 4 монографии [2].

Лекция 2

Геометрическое моделирование

Автоматизация черчения и геометрическое моделирование ...	34
Виды геометрического моделирования	35
Функции твердотельного моделирования	37
Декомпозиционные модели	38
Конструктивные модели	39
Граничные модели	40
Корректность граничных моделей	42
Пакеты геометрического моделирования и их функциональность	43
Вопросы для самоконтроля	44
Дополнительная литература	44

Автоматизация черчения и геометрическое моделирование

Инженеры и конструкторы имеют дело с математической (прежде всего – геометрической) моделью разрабатываемого изделия. Исторически первым и главным языком их общения (то есть языком описания инженерных моделей) был язык чертежей. Чертеж (и другие подобные графические схемы) широко использовался (и используется до сих пор) не только для описания механического изделия и его частей, но также и для описания электрических схем, архитектурных конструкций, карт местности и т. п. Четкие стандарты (как национальные, так и международные) гарантируют однозначное понимание языка чертежей всеми «читателями» – от инженера-конструктора до токаря, слесаря и фрезеровщика. Однако создание чертежей вручную – чрезвычайно дорогостоящая процедура, доступная только подготовленным специалистам и требующая использования специальной чертежной доски с линейкой – кульмана, а также разных вспомогательных средств (например, лекал для рисования кривых). Неудивительно, что первые системы автоматизации в этой области были предназначены именно для упрощения и ускорения создания чертежей (подобно другой эпохальной концепции автоматизации с помощью компьютера – текстовым процессорам, предназначенным для упрощения создания текстовых документов и легкого внесения изменений в них). Системы класса computer-aided drafting существуют и поныне, самый известных их представитель – AutoCAD. Типичная функциональность таких систем включает в себя средства, необходимые для создания и редактирования чертежей, а процесс работы концептуально не отличается от работы в графическом редакторе. (Основными понятиями графических моделей таких систем являются графические *примитивы с атрибутами*, отображаемые *уровни*, а также различные *способы конструирования*.) В нашем курсе мы не будем рассматривать системы этого класса, так как в настоящее время они стремительно уступают свои позиции системам трехмерного моделирования (даже Autodesk делает основную ставку не на AutoCAD, а на Inventor Series).

Дело в том, что с изобретением трехмерной компьютерной графики (возможности реалистического изображения трехмерной сцены на двумерном дисплее компьютера и ее вращения с помощью манипуляторов «мышь» или «спейсбол» в воображаемом трехмерном пространстве) у инженеров появилась возможность работать напря-

мую с трехмерной геометрической моделью проектируемого изделия, а не с его двумерными чертежами. Геометрическое моделирование оказалось настоящим прорывом в конструировании и производстве изделий. Оно не только значительно упрощает процесс проектирования (теперь инженер-конструктор не обязан обладать развитым пространственным мышлением или использовать подручные материалы типа пластилина – он видит проектируемое изделие непосредственно на экране), но и снимает многие коммуникативные проблемы – языка чертежей на наших глазах становится мертвым.

За последние сорок лет было разработано множество способов геометрического моделирования, которые мы разберем детально ниже.

Виды геометрического моделирования

Хронологически различают следующие подходы к геометрическому моделированию:

- каркасное моделирование;
- поверхностное моделирование;
- твердотельное моделирование;
- немногообразное моделирование.

Каркасное моделирование представляет собой прямой перенос векторного подхода к двумерной геометрии на трехмерный случай. При таком моделировании геометрическая модель строится из ограниченного набора *графических примитивов* – отрезки, дуги, конические кривые. Однако каркасная модель содержит лишь скелет (каркас) изделия, по которому в общем случае невозможно восстановить само изделие, так как могут существовать несколько топологически неэквивалентных трехмерных тел с одинаковым каркасом, как это видно на приведенном рис. 2.

Поверхностное моделирование является развитием каркасного – с его помощью можно точно описывать поверхности геометрического тела, формирующие его оболочку. Поверхностное моделирование играет важную роль при проектировании изделий из листового металла (sheet metal parts), таких как капоты и крылья автомобилей, где форма поверхности важна как для дизайна, так и для аэродинамики изделия. Более подробно средства поверхностного моделирования (различные виды задания кривых и поверхностей) рассматриваются в материале следующих лекций.

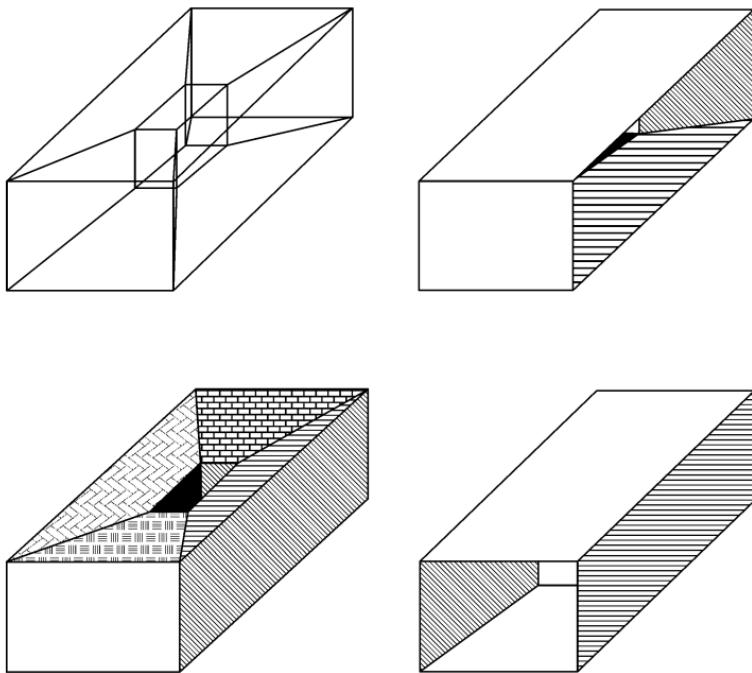


Рис. 2

Твердотельное (объемное) моделирование – логическое развитие каркасного и поверхностного. Основной объект моделирования – трехмерное объемное тело, которое может описываться разными способами: декомпозиционным, конструктивным или граничным. Мы разберем их подробнее ниже. Главным преимуществом твердотельного моделирования перед каркасным и поверхностным является свойство физической корректности – все твердотельные модели имеют аналоги в реальном мире (чего не скажешь о каркасных и поверхностных моделях).

Немногообразное (non-manifold) моделирование снимает ограничения, присущие классическому твердотельному моделированию – с его помощью можно описывать геометрические модели, которые локально могут быть не только многообразиями размерности три (объемными телами), но и размерности два (поверхностями), один (кривыми), нуль (точками), а также участками сопряжения многообразий разной размерности.

Функции твердотельного моделирования

Функции твердотельного моделирования подразделяются на следующие группы:

- функции создания примитивов,
- перенос и поворот тела,
- булевы операции,
- функции заметания и скиннинга,
- конструктивные элементы,
- расчет объемных параметров тела (объема, массы, моментов инерции).

Все эти базовые функции, как правило, напрямую доступны пользователям программ, построенных на основе твердотельного моделирования.

Типичные *твердотельные примитивы* – брус (прямоугольный параллелепипед), цилиндр, конус, шар, клин, тор. При создании примитива пользователь должен определить его размеры и положение в пространстве.

Булевы операции включают в себя операции объединения, пересечения и разности двух тел. Результаты применения булевых операций к двум твердым телам (куб и сфера) можно видеть на рис. 3.

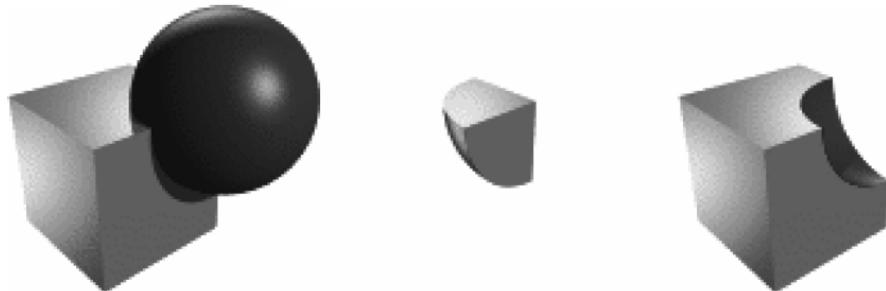


Рис. 3

Функции заметания создают объемное тело поступательным или вращательным движением замкнутого двумерного контура в трехмерном пространстве. Функция скиннинга «натягивает» трехмерное тело на его плоские срезы, заданные в виде замкнутых двумерных контуров.

Типичные конструктивные элементы включают в себя скругление, поднятие, проделывание отверстия. При создании конструктивного элемента задаются его размеры.

Важным свойством систем твердотельного моделирования является возможность расчета объемных параметров тела – объема, центра масс, тензора инерции и пр. Все такие параметры выражаются объемным интегралом по телу.

В большинстве современных CAD-систем пользователь может создать свой набор конструктивных элементов. Отметим, что в одной конкретной системе геометрического моделирования могут поддерживаться не все функции твердотельного моделирования, а только их часть.

Декомпозиционные модели

Декомпозиционные модели являются простейшим подходом к твердотельному моделированию, представляя трехмерное тело композицией некоторых простых элементов. Различают следующие декомпозиционные модели:

- воксельное (voxel) представление;
- октантное дерево;
- ячеичное представление.

Воксельное представление – полный трехмерный аналог растрового одноцветного изображения. Тело представляется трехмерным булевым массивом, каждый элемент которого является пространственным кубиком одинакового размера со своими уникальными координатами. Такой кубик называется вокселем (voxel – от VOlume riXEL). Воксели равномерно покрывают всю область (прямоугольный параллелепипед), в которой содержится моделируемое тело. Соответственно, те воксели, которые имеют непустое пересечение с моделируемым телом, представляются в массиве значением ИСТИНА, прочие – значение ЛОЖЬ. Отметим удобство воксельного представления для реализации на его основе булевых операций твердотельного моделирования. Для этого необходимо построить согласованные воксельные представления двух тел и применить соответствующую операцию к булевым значениям ячеек массива. Отметим, что сложность такого алгоритма будет прямо зависеть от числа вокселов. На воксельном представлении несложно вычислять объемные параметры тела – достаточно лишь вычислить их аналитически для каждого вокселя и просуммировать.

Октаантное дерево является развитием воксельного представления. Каждый узел октаантного дерева соответствует некоторому кубу в трехмерном пространстве, который является либо:

- полностью (с заданной точностью) принадлежащим описываемому телу;
- полностью непринадлежащим описываемому телу;
- частично пересекающимся с описываемым телом.

Первые два типа узлов – терминальные (листья в октаантном дереве), а каждый узел третьего типа обязательно имеет 8 дочерних узлов, соответствующих геометрическому разбиению его куба на 8 частей (октаантов) – рис. 4.

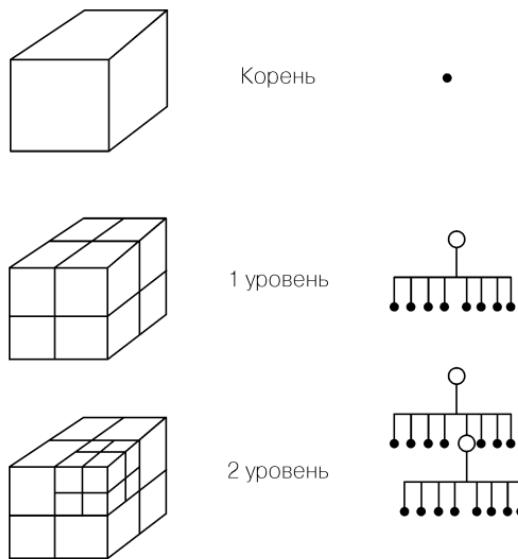


Рис. 4

Ячеичное представление соответствует разбиению моделируемого тела на произвольные непересекающиеся выпуклые многогранники, полностью (в соответствии с заданной погрешностью) заполняющие его объем.

Конструктивные модели

Конструктивные модели основаны на журнале применения операций твердотельного моделирования. Для создания копии модели необходимо просто заново выполнить все операции из журнала, то есть

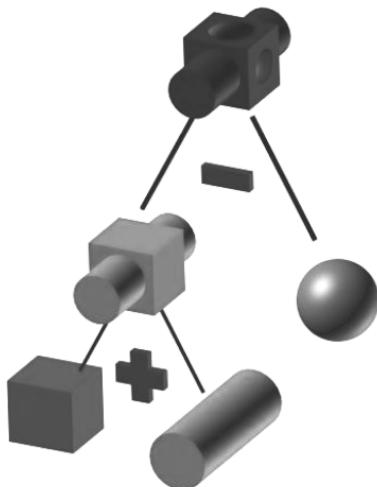


Рис. 5

заново сконструировать изделие, поэтому подобный подход к моделированию называется конструктивным.

CSG (constructive solid geometry)- модели реализуют конструктивный подход в терминах булевых операций над параметрическими твердотельными примитивами (прямоугольный параллелепипед, цилиндр и пр.). CSG-модели обычно представляются в виде дерева, у которого все листья являются примитивами с указанными размерами, а каждая нетерминальная вершина является либо аффинной трансформацией (переносом или вращением), либо булевым оператором. И трансформация, и опе-

ратор применяются к вершинам-потомкам. Пример CSG-дерева приведен на рис. 5.

Реализация булевых операций для конструктивных моделей, основанных на дереве CSG, достаточно прямолинейна – ведь для этого всего лишь нужно соединить (сконкатенировать) два дерева в одно, используя новый узел, символизирующий конкретную операцию. Никаких вычислений такой алгоритм не подразумевает.

Нетрудно составить алгоритм перевода CSG-дерева в октантное дерево, алгоритм трассировки луча (для отрисовки CSG-дерева на дисплее), а также алгоритм упрощения дерева (удаления из него лишних узлов).

Расчет объемных параметров выполняется простым обходом дерева. Для терминальных узлов (представляющих собой примитивные тела) расчет объемных параметров выполняется аналитически, для нетерминальных (трансформаций и булевых операций) требуется лишь выполнить арифметические действия над вычисленными параметрами дочерних узлов.

Границные модели

Границные модели хранят информацию о границах тела (границах, ребрах и вершинах). Для простоты манипулирования эта информация подразделяется на геометрические и топологические данные. Геометрические данные для каждой границной сущности свои:

- для вершины – ее координаты;
- для ребра – параметрическое уравнение кривой (прямой);
- для грани – параметрическое уравнение поверхности либо тип и набор параметров в случае канонической поверхности (плоскости, сферы, цилиндра, конуса, тора).

Топологические данные – это информация о смежности вершин и ребер, ребер и граней, а также о внутренних и внешних границах грани. Для удобного манипулирования топологической информацией было предложено несколько структур данных, называемых *BRep*:

- многогранные (фасетные) модели;
- вершинные модели;
- полуреберные модели;
- крыльевые реберные модели.

Разберем одну из самых популярных структур – *полуреберную*, основанную на том простом факте, что каждое ребро границы твердого тела принадлежит ровно двум граням (рис. 6).

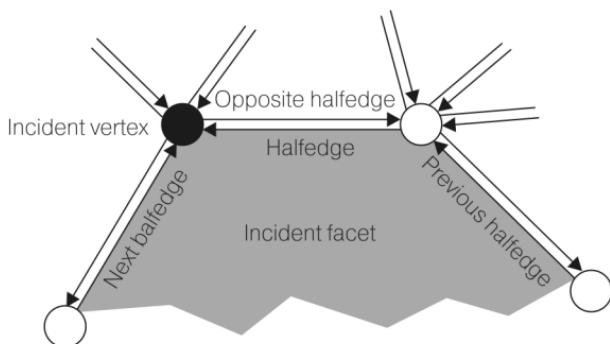


Рис. 6

Структуры данного представления таковы:

- список *тел* (solid), каждое тело состоит из списка его *граней* (faces), *ребер* (edges) и *вершин* (vertices);
- грань состоит из *колов* (loop), представляющих собой внешнюю границу грани, а также ее optionalные внутренние границы;
- кольцо состоит из списка *полуребер* (halfedges);
- полуребро указывает на начальную вершину и следующее полуребро, а также на свое ребро;
- ребро хранит указатели на два своих полуребра.

Корректность граничных моделей

Важным свойством при работе с граничными моделями является обеспечение их корректности. В отличие от декомпозиционных и конструктивных моделей граничная может не соответствовать никакому твердому телу. Для автоматического обеспечения корректности граничных моделей используют операторы Эйлера, основанные на формуле Эйлера–Пуанкаре.

К топологическим элементам граничной модели применимо следующее равенство, называемое *формулой Эйлера–Пуанкаре*:

$$v - e + f = 2(s - h) + r,$$

где v – число вершин, e – число ребер, f – число граней, s – число тел, h – число сквозных отверстий в телах, r – количество внутренних граней (кольец) в гранях.

При модификации топологии граничной модели используют операторы Эйлера, которые добавляют в модель (или удаляют из нее) топологические элементы, сохраняя в силе формулу Эйлера–Пуанкаре. Ниже приведен популярный набор операторов Эйлера (табл. 1).

Таблица 1

Оператор	V	e	f	h	s	r	Описание
mvfs	+1	0	+1	–	+1	0	создать вершину, грань, тело
mev	+1	+1	0	0	0	0	создать ребро, вершину
mef	0	+1	+1	0	0	0	создать ребро, грань
kemr	0	–1	0	0	0	+1	удалить ребро, создать кольцо
kfmrh	0	+1	–1	+1	0	0	удалить грань, создать кольцо, отверстие
kvfs	–1	0	–1	0	–1	0	удалить вершину, грань, тело
kev	–1	–1	0	0	0	0	удалить ребро, вершину
kef	0	–1	–1	0	0	0	удалить ребро, грань
mekr	0	+1	0	0	0	–1	создать ребро, удалить кольцо
mflrh	0	–1	+1	–1	0	0	создать грань, удалить кольцо, отверстие

Об операторах Эйлера справедливы следующие утверждения:

- любая топологически корректная граничная структура данных может быть как создана, так и полностью удалена применением конечного числа операторов Эйлера;
- операторы Эйлера не могут создать топологически некорректную граничную структуру данных.

Одним из достоинств граничной модели является удобный способ расчета ее объемных параметров. Напомним, что объемные параметры выражаются объемным интегралом по телу. Для BRep-модели объемный интеграл преобразуется в поверхностный (в соответствии с теоремой Остроградского–Гаусса), который в свою очередь расписывается на сумму поверхностных интегралов для каждой грани. Поверхностный интеграл по грани вычисляется либо как двойной интеграл (если грань отображается на прямоугольную область в пространстве параметров), либо расписывается по теореме Грина как криволинейный интеграл, который вычисляется аналитически или численно по каждому ребру грани, а затем суммируется.

Пакеты геометрического моделирования и их функциональность

Пакет геометрического моделирования (называемый также *геометрическим ядром*) – набор библиотек с программным интерфейсом (API), с помощью которого можно пользоваться функциями геометрического (например, твердотельного) моделирования. Многие ведущие CAD-системы (такие как CATIA, Pro/Engineer, NX) построены на основе собственных геометрических ядер (CGM, GRANITE и Parasolid соответственно), тогда как другие (SolidWorks, T-FLEX, ADEM и пр.) построены на основе лицензированных геометрических ядер. Самыми популярными ядрами (используемыми в наибольшем количестве САПР) являются Parasolid (от компании Siemens PLM Software), ACIS (выпускаемый Spatial Corp. – дочерней компании Dassault Systèmes) и GRANITE (PTC). Отметим также свободно распространяемый в открытом коде пакет Open CASCADE (выпускаемый одноименной компанией).

Типичной функциональностью пакета геометрического моделирования является предоставление набора программных интерфейсов (структур данных, функций и классов) для создания приложения каркасного, поверхностного, твердотельного или немногообразного моделирования. Обычно родственные интерфейсы группируются в модули, среди которых выделяют:

- базовые типы и операции;
- моделирование топологии;
- геометрические объекты и операции над ними;

- булевы операции и операции редактирования поверхностей;
- удаление невидимых линий и рендеринг;
- модули для чтения и записи геометрических файлов популярных форматов.

Вопросы для самоконтроля

1. Опишите разницу между автоматизацией черчения и геометрическим моделированием.
2. Назовите и опишите виды геометрического моделирования.
3. Каковы основные функции твердотельного (объемного) моделирования?
4. Опишите три вида декомпозиционных моделей.
5. Что такое CSG-дерево? Опишите алгоритм перевода CSG-дерева в октантное дерево.
6. В чем разница между геометрией и топологией граничной модели? Опишите структуры данных BRep.
7. Приведите формулу Эйлера–Пуанкаре и опишите операторы Эйлера. Какими свойствами они обладают?
8. Что такое объемные параметры и как они рассчитываются по граничной модели?
9. Какова базовая функциональность пакетов геометрического моделирования? Приведите примеры таких пакетов.

Дополнительная литература

Подробную информацию (со ссылками на первоисточники) по различным видам геометрического моделирования можно получить из главы 5 книги [8] и главы 2 в [37].

Базовые геометрические объекты

Аффинное пространство и соглашение о нотации	46
Способы задания аналитических кривых и поверхностей	46
Изометрии аффинного пространства	48
Матричное представление трансформации в аффинном пространстве	49
Однородные координаты	50
Углы Эйлера	51
Экспоненциальное представление трансформации	52
Вопросы для самоконтроля	53
Дополнительная литература	54

Аффинное пространство и соглашение о нотации

Напомним, что *аффинное пространство* задается двумя непересекающимися множествами – точек и векторов, а также операцией откладывания точки от другой точки с помощью вектора и обратной к ней операции вычисления вектора, соединяющего две точки. Множество векторов должно образовывать евклидово пространство (линейное пространство со скалярным произведением). Мы будем иметь дело только с трехмерным аффинным пространством, в котором также определено векторное произведение. Точки и векторы в этом пространстве могут задаваться тройками вещественных чисел.

В дальнейшем будем придерживаться следующего соглашения о нотации: точки будем обозначать прописными жирными латинскими и греческими буквами: \mathbf{P}, \mathbf{Q} , векторы – строчными жирными буквами: $\mathbf{e}, \mathbf{\Theta}$, скалярные величины – обычным шрифтом: x, α . Оставшийся способ обозначений – прописные нежирные буквы – будем использовать для обозначения матриц. Скалярное произведение векторов \mathbf{u} и \mathbf{v} обозначим (\mathbf{u}, \mathbf{v}) , векторное – $\mathbf{u} \wedge \mathbf{v}$. При работе с формулами, содержащими векторное произведение, часто бывает удобно представлять его в виде произведения 3×3 -матрицы и вектора. Делается это путем определения операции \wedge : $\mathbf{R}^3 \rightarrow \mathbf{R}^{3 \times 3}$, отображающей произвольный трехмерный вектор в матрицу, называемую его *кососимметрическим тензором*:

$$\hat{\mathbf{e}} = \begin{pmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{pmatrix} \text{ для вектора } \mathbf{e} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}.$$

Нетрудно видеть, что $\mathbf{u} \wedge \mathbf{v} = \hat{\mathbf{u}} \mathbf{v}$. Нормой вектора будем называть корень из его скалярного произведения с самим собой (которое всегда положительно): $\|\mathbf{v}\| = \sqrt{(\mathbf{v}, \mathbf{v})}$. При записи векторно-матричных операций будем пользоваться операцией транспонирования, обозначая ее R^T .

Способы задания аналитических кривых и поверхностей

Задавать множество точек в трехмерном аффинном пространстве можно несколькими способами. Первый – описать условия на координаты точек множества в алгебраическом виде. При этом речь может

идти о явном ($y = ax^2, z = 0$) или неявном ($x^2 + y^2 + z^2 = 1$) задании. Другим способом спецификации множества точек является его параметрическое описание (например, уравнение спирали $x(t) = \sin t, y(t) = \cos t, z(t) = a(t)$). Заметим, что 1-многообразия (кривые) параметризуются одной переменной, тогда как 2-многообразия (поверхности) требуют двух переменных при параметрическом описании. В силу ряда причин в CAD-системах удобно комбинировать неявное координатное и параметрическое задание многообразий (в частности, при нахождении пересечения двух множеств удобно подставить в координатное уравнение одного параметрическое описание другого). Ниже мы разберем, каким образом задаются простейшие многообразные формы в трехмерном пространстве, указывая оба способа задания.

Одним из способов задания прямой является спецификация какой-либо точки на ней, а также указание единичного вектора, задающего направление прямой:

$$\mathbf{L} = (\mathbf{P}, \mathbf{e}) = \{\mathbf{Q} | (\mathbf{Q} - \mathbf{P})^\wedge \mathbf{e} = 0\}.$$

Параметрическое уравнение прямой в этом случае имеет вид $\mathbf{L}(t) = \mathbf{P} + t\mathbf{e}$. Популярным способом представления плоскости является спецификация какой-либо точки на ней и указание единичного вектора нормали:

$$\mathbf{F} = (\mathbf{P}, \mathbf{e}) = \{\mathbf{Q} | ((\mathbf{Q} - \mathbf{P}), \mathbf{e}) = 0\}.$$

Однако для параметрического задания плоскости приходится строить два дополнительных вектора, ортогональных нормали плоскости и неколлинеарных между собой. Проще всего это сделать с помощью следующей процедуры: взять три единичных координатных вектора $(1,0,0)^T, (0,1,0)^T, (0,0,1)^T$ и вычислить их векторные произведения с \mathbf{e} . Нетрудно видеть, что среди трех полученных векторов всегда найдутся два неколлинеарных, обозначим их \mathbf{f} и \mathbf{g} . Тогда параметрическое уравнение плоскости выглядит как

$$\mathbf{F}(u, v) = \mathbf{P} + u\mathbf{f} + v\mathbf{g}.$$

Сфера задается центром и радиусом:

$$\mathbf{S} = (\mathbf{P}, r) = \{\mathbf{Q} | \|\mathbf{Q} - \mathbf{P}\| = r\}$$

и параметризуется сферическими координатами:

$$\mathbf{S}(u, v) = \mathbf{P} + (r \cos u \cos v, r \sin u \cos v, r \sin v).$$

Цилиндр задается точкой на оси, направлением оси и радиусом:

$$\mathbf{C} = (\mathbf{P}, \mathbf{e}, r) = \{\mathbf{Q} | \|(\mathbf{Q} - \mathbf{P})^\wedge \mathbf{e}\| = r\}.$$

Окружность задается своим центром, направлением оси и радиусом:

$$\mathbf{O} = (\mathbf{P}, \mathbf{e}, r) = \{ \mathbf{Q} \mid ((\mathbf{Q} - \mathbf{P}) \cdot \mathbf{e}) = 0 \text{ и } ((\mathbf{Q} - \mathbf{P}) \wedge \mathbf{e}) = r \}.$$

Ее параметризацию удобно выполнять для случая $\mathbf{e} = (0, 0, 1)^T$. Тогда

$$\mathbf{O}(t) = \mathbf{P} + (r \cos t, r \sin t, 0).$$

Общая параметризация получается преобразованием систем координат, которое мы разберем в следующих разделах. Несложно сформулировать аналогичные уравнения для других кривых и поверхностей второго порядка, однако они часто представляются в системах геометрического моделирования специальным образом, который рассматривается в следующей лекции.

Изометрии аффинного пространства

Как известно, изометрия (от греч. *isos* и *metron* – равное измерение) – это трансформация метрического пространства, сохраняющая расстояния между любыми его точками. В аффинном пространстве изометрии сохраняют углы между векторами и расстояния между точками. Отметим, что множество всех изометрий образует группу. С каждой изометрией аффинного пространства связано соответствующее преобразование ассоциированного евклидова пространства, которое называется *ортогональным*. Ортогональные преобразования трехмерного евклидова пространства – это *вращения* (ортогональные преобразования, сохраняющие векторное произведение) и *отражения* (не сохраняющие знак векторного произведения) векторов. Изометрии аффинного пространства, сохраняющие знак векторного произведения, называются *трансформациями*; их можно исчерпывающим образом разделить на три класса:

- параллельный перенос вдоль заданного вектора;
- вращение вокруг заданной оси;
- винтовое движение (комбинация вращения вокруг заданной оси со смещением вдоль нее).

Таким образом, любую трехмерную трансформацию можно охарактеризовать следующими *геометрическими параметрами*:

- \mathbf{e} – единичный вектор (задающий направление оси вращения или направление параллельного переноса);
- $\mathbf{\Omega}$ – точка опоры (вместе с вектором \mathbf{e} она задает ось вращения);
- α – угол вращения (зависит от направления);
- β – величина смещения вдоль вектора \mathbf{e} .

Свойства этих параметров в зависимости от класса трансформации определяются с помощью табл. 2.

Таблица 2

Класс трансформации	\mathbf{e}	$\mathbf{\Omega}$	α	β
Параллельный перенос	Направление переноса	0	0	Величина переноса
Вращение	Направление оси вращения	Точка на оси вращения	Угол вращения	0
Винтовое движение	Направление оси винта	Точка на оси винта	Угол вращения	Смещение точки $\mathbf{\Omega}$ вдоль оси винта

Отметим, что если $(\mathbf{e}, \mathbf{\Omega}, \alpha, \beta)$ – произвольная трансформация, то обратная к ней трансформация может быть задана как $(\mathbf{e}, \mathbf{\Omega}, -\alpha, -\beta)$. Однако комбинацию двух трансформаций в данном представлении вычислить уже сложнее (как это сделать, мы разберем ниже). Отметим также, что вместо семи вещественных параметров, задающих аффинную трансформацию (трехмерный единичный вектор можно задать двумя сферическими координатами), на практике используются иные способы ее задания – шесть, двенадцать или шестнадцать параметров. Мы их также разберем ниже.

Матричное представление трансформации в аффинном пространстве

Как известно, любое линейное преобразование векторов в трехмерном евклидовом пространстве можно задать 3×3 -матрицей с вещественными коэффициентами. В этом случае новые координаты вектора можно вычислить простым перемножением этой матрицы и вектора старых координат. Однако не все матрицы задают трансформации, сохраняющие углы и длины. Этим свойством обладают только ортогональные матрицы ($R^T R = I$). Определитель ортогональной матрицы может быть равен +1 или -1, и трансформациям соответствует первая группа, называемая *матрицами вращения* (ортогональные матрицы с определителем -1 геометрически соответствуют зеркальным отражениям). Как известно, для получения обратного вращения достаточно транспонировать матрицу (в силу свойства ортогональ-

ности), а для комбинации двух вращений – перемножить соответствующие матрицы.

При задании трансформаций точек одной матрицы вращения оказывается недостаточно, так как она задает только вращения вокруг осей, проходящих через начало координат. Поэтому приходится указывать еще три числа – координаты вектора, называемого трансляционной (поступательной) компонентой трансформации. Геометрический смысл трансляционной компоненты состоит в том, что после вращения точки вокруг оси вращения, проходящей через начало координат, ее необходимо сместить в направлении, заданном трансляционным вектором:

$$\mathbf{P}' = R\mathbf{P} + \mathbf{t}.$$

Отсюда нетрудно вывести формулы обратного преобразования (R', \mathbf{t}') для трансформации (R, \mathbf{t}) :

$$\begin{aligned} R' &= R^T, \\ \mathbf{t}' &= -R^T \mathbf{t}, \end{aligned}$$

а также комбинации двух трансформаций – (R_1, \mathbf{t}_1) и (R_2, \mathbf{t}_2) :

$$\begin{aligned} R &= R_2 R_1, \\ \mathbf{t} &= R_2 \mathbf{t}_1 + \mathbf{t}_2, \end{aligned}$$

Связь геометрических параметров трансформации с ее матричным представлением

Формула Эйлера–Родригеса позволяет описать коэффициенты матрицы вращения в трехмерном пространстве на заданный угол α вокруг произвольного направления, задаваемого единичным вектором \mathbf{e} :

$$R = \text{Rot}(\mathbf{e}, \alpha) = \cos \alpha I + \sin \alpha \hat{\mathbf{e}} + (1 - \cos \alpha) \mathbf{e}^T \mathbf{e}.$$

Трансляционная компонента соответствующей трансформации зависит не только от α и \mathbf{e} , но и от Ω с β :

$$\mathbf{t} = (I - R)\Omega + \beta \mathbf{e}.$$

Из этих формул нетрудно вывести алгоритм для вычисления геометрических параметров трансформации по матрице вращения и вектору смещения.

Однородные координаты

Чтобы преодолеть неудобства, вызванные разными способами применения трансформаций к точкам и векторам (каждая из этих сущностей задается тремя скалярными величинами), часто используют по-

нятие *однородных координат*. Однородные координаты задаются четверкой чисел $(x, y, z, w)^T$. При этом точка $\mathbf{P} = (a, b, c)^T$ представляется в однородных координатах как $(a, b, c, 1)^T$, а вектор $\mathbf{v} = (d, e, f)^T$ как $(d, e, f, 0)^T$. Впрочем, одну и ту же точку можно представить и по-другому – с использованием произвольной четвертой координаты, достаточно лишь удовлетворить равенства:

$$a = xw,$$

$$b = yw,$$

$$c = zw,$$

$$w \neq 0.$$

Главное удобство однородных координат состоит в том, что с их помощью трансформация для точек и векторов задается одним и тем же способом – с помощью матрицы 4×4 , имеющей следующую структуру:

$$\begin{pmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ t_1 & t_2 & t_3 & 1 \end{pmatrix}.$$

Отметим, что умножение такой матрицы на однородный вектор с нулевой четвертой координатой (который, напомним, задает вектор в трехмерном аффинном пространстве) будет эквивалентно повороту этого трехмерного вектора в соответствии с матрицей вращения, задаваемой коэффициентами r_{11}, \dots, r_{33} . Аналогично умножение такой матрицы на четырехмерный вектор с ненулевой четвертой координатой даст в точности тот же эффект, что вращение соответствующей аффинной точки, задаваемое коэффициентами r_{11}, \dots, r_{33} , и последующее смещение на вектор $(t_1, t_2, t_3)^T$. Таким образом, можно не различать точки и векторы на уровне реализации аффинных трансформаций в однородном представлении. Отметим также, что в однородном представлении чрезвычайно удобно вычислять комбинацию аффинных трансформаций – достаточно лишь перемножить соответствующие 4×4 -матрицы.

Углы Эйлера

Параметризации трансформаций трехмерного аффинного пространства с помощью семи и тем более двенадцати параметров не являются минимальными, ведь твердое тело имеет всего шесть степеней свободы.

ды в трехмерном пространстве. Если представление трансляционной части трансформации с помощью трех компонент выглядит минимальным, то описание вращения с помощью вектора и угла или 3×3 -матрицы можно сократить. Одним из известных способов является декомпозиция произвольного вращения вокруг оси, проходящей через начало координат, на три вращения вокруг координатных осей. Во-первых, нетрудно вывести формулы для матриц вращения вокруг координатных осей (аналитическими рассуждениями или простым использованием вышеприведенной формулы Эйлера–Родригеса для вращения вокруг произвольной оси):

$$R_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & \sin \alpha \\ 0 & -\sin \alpha & \cos \alpha \end{pmatrix}, R_y(\alpha) = \begin{pmatrix} \cos \alpha & 0 & -\sin \alpha \\ 0 & 1 & 0 \\ \sin \alpha & 0 & \cos \alpha \end{pmatrix}, R_z(\alpha) = \begin{pmatrix} \cos \alpha & \sin \alpha & 0 \\ -\sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Далее, можно показать, что произвольная матрица вращения R может быть представлена в виде произведения матриц поворота вокруг координатных осей: $R = R_x(\theta)R_y(\phi)R_z(\psi)$. Параметры θ, ϕ и ψ , задающие углы поворота вокруг координатных осей, называются *углами Эйлера*. Из этой формулы нетрудно получить алгоритм расчета этих углов по произвольной матрице вращения, заданной девятью коэффициентами. Главным недостатком параметризации Эйлера является тот факт, что траектория, описываемая точкой $\mathbf{P}(t) = R_x(t\theta)R_y(t\phi)R_z(t\psi)\mathbf{P}_0$, является сложной кривой, которая не совпадает с вращательным движением точки в случае непрерывного движения. Например, вращая глобус, мы видим, что точки земного шара описывают окружности вокруг оси вращения, а в случае параметризации Эйлера этот процесс можно смоделировать, если только вращение идет строго вокруг одной из координатных осей. Поэтому такая параметризация непригодна для приложений, моделирующих непрерывные движения тел.

Экспоненциальное представление трансформации

Напомним, что экспонента произвольной квадратной матрицы вычисляется по формуле

$$\exp(M) = \sum_{n=0}^{\infty} \frac{1}{n!} M^n.$$

Для матрицы вращения вокруг оси \mathbf{e} на угол α выполняется замечательное свойство:

$$\text{Rot}(\mathbf{e}, \alpha) = \exp(\hat{\mathbf{u}}) \leftrightarrow \mathbf{u} = \alpha \mathbf{e}.$$

Данное свойство дает нам еще один способ (помимо рассмотренных выше углов Эйлера) представить вращение тремя вещественными числами. Замечательные свойства такого представления очевидны: траектория, описываемая точкой $\mathbf{P}(t) = \exp(t\hat{\mathbf{e}})\mathbf{P}_0$, является геодезической (дугой окружности). Этот факт позволяет использовать экспоненциальную параметризацию (в отличие от эйлеровой) для решения задач, связанных с непрерывным движением. Экспоненциальная параметризация матриц вращения естественно расширяется на случай произвольных трансформаций (с учетом параметров Ω и β). Для этого мы пользуемся приведенным выше вектором-параметром \mathbf{u} и дополняем его еще одним вектором-параметром \mathbf{v} :

$$\mathbf{v} = \Omega^\wedge \mathbf{u} + \beta \mathbf{e}.$$

В матричных терминах трансляционная компонента определяется как

$$\mathbf{t} = (I - \exp(\hat{\mathbf{u}})) \frac{\mathbf{u}^\wedge \mathbf{v}}{(\mathbf{u}, \mathbf{u})} + \frac{(\mathbf{u}, \mathbf{v})}{(\mathbf{u}, \mathbf{u})} \mathbf{u}.$$

Вопросы для самоконтроля

1. Назовите основные способы задания кривых и поверхностей в трехмерном аффинном пространстве. Приведите примеры.
2. Назовите основные классы трансформаций в трехмерном аффинном пространстве. Какими геометрическими параметрами они характеризуются?
3. Опишите матричное представление трансформации в трехмерном аффинном пространстве и назовите его свойства.
4. Приведите алгоритмы вычисления матричного представления трехмерной трансформации по ее геометрическим параметрам и наоборот.
5. Что такое однородные координаты? В чем преимущества их использования для представления трансформаций в трехмерном аффинном пространстве?
6. Дайте определение углов Эйлера. Приведите алгоритмы вычисления трансформации с заданными углами Эйлера и вычисления углов Эйлера по трансформации, заданной в матричном виде.

7. Какие параметры задают экспоненциальное представление трансформации? Приведите алгоритм их расчета по матричному представлению.

Дополнительная литература

Про матрицы преобразования можно почитать в третьей главе книги [8], а также в многочисленных источниках по компьютерной графике, например в [32].

Лекция 4

Инженерные кривые и поверхности

Кусочные кривые и их гладкость	56
Билинейный лоскут	56
Поверхности сдвига и вращения	56
Линейчатая поверхность	57
Лоскут Кунса	57
Эрмитова кривая, бикубическая поверхность и лоскут Фергюсона	58
Кривые и поверхности Безье	61
Алгоритм де Кастельжо	62
В-сплайны и В-сплайновые поверхности	63
Рациональные кривые и поверхности	64
Интерполяционные кривые и поверхности	65
Вопросы для самоконтроля	65
Дополнительная литература	66

Кусочные кривые и их гладкость

Говоря о C^n -непрерывности (гладкости) кривых и поверхностей, подразумевают непрерывность k -х производных их параметрических уравнений для всех $0 \leq k \leq n$. Зачастую кривые составляются из криволинейных сегментов (а поверхности – из лоскутов) разной параметризации. В этом случае говорят о G^n -непрерывности, которая подразумевает непрерывность направления (единичного вектора) k -х производной параметрического уравнения для всех $0 \leq k \leq n$. Для сходимости итерационных методов второго порядка (например, метода Ньютона–Рафсона) необходимо, чтобы рассматриваемые кривые и поверхности имели G^2 -непрерывность.

Уравнения кривых и поверхностей обычно записываются в некой удобной системе координат, а для преобразования в глобальную систему координат используются аффинные трансформации, описанные выше.

Билинейный лоскут

Куски поверхностей удобно представлять в виде области отображения прямоугольника в параметрическом пространстве $\mathbf{P}(u, v)$, $u_0 \leq u \leq u_1$, $v_0 \leq v \leq v_1$ (зачастую $u_0 = v_0 = 0$, $u_1 = v_1 = 1$). Такая конечная поверхность называется *лоскутом*. Лоскуты удобно «сшивать» друг с другом, образуя непрерывную поверхность нужной степени гладкости. Простейшим видом лоскута является билинейная поверхность, задаваемая четырьмя граничными вершинами:

$$\begin{aligned}\mathbf{P}(0, 0) &= \mathbf{P}_{00}, \\ \mathbf{P}(0, 1) &= \mathbf{P}_{01}, \\ \mathbf{P}(1, 0) &= \mathbf{P}_{10}, \\ \mathbf{P}(1, 1) &= \mathbf{P}_{11}.\end{aligned}$$

Оставшиеся точки поверхности образуются линейной аппроксимацией заданных. Нетрудно видеть, что уравнение билинейного лоскута имеет следующий вид:

$$\mathbf{P}(u, v) = (1-u)(1-v)\mathbf{P}_{00} + (1-u)v\mathbf{P}_{01} + u(1-v)\mathbf{P}_{10} + uv\mathbf{P}_{11}, \\ 0 \leq u \leq 1, 0 \leq v \leq 1.$$

Поверхности сдвига и вращения

Поверхность *сдвига* (swept surface) задается точками заданной кривой $\mathbf{P}(t) = (x(t), y(t), z(t))$, $0 \leq t \leq 1$, при ее движении в заданном направлении $\mathbf{e} = (\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z)$. При этом получается следующая параметри-

зация поверхности сдвига: $\mathbf{P}(u, v) = \mathbf{P}(u) + v\mathbf{e}$. Аналогичным образом задается поверхность вращения. Общий случай описывается движением заданной кривой ($\mathbf{P}_1(t)$) вдоль направляющей кривой ($\mathbf{P}_2(t)$) (рис. 7).

Уравнение обобщенной поверхности сдвига записывается как

$$\mathbf{P}(u, v) = \mathbf{P}_1(u) + \mathbf{P}_2(v) - \mathbf{P}_2(0), \\ 0 \leq u \leq 1, 0 \leq v \leq 1.$$

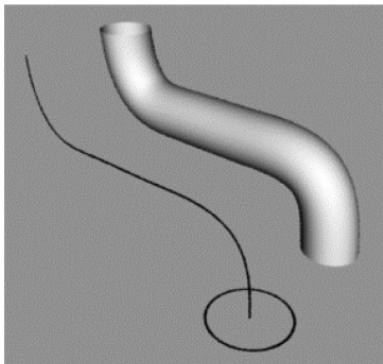


Рис. 7

Линейчатая поверхность

Линейчатая поверхность (ruled surface) является еще одним способом задания поверхности по двум кривым – $\mathbf{P}_1(t)$ и $\mathbf{P}_2(t)$, $0 \leq t \leq 1$. Поверхность образуется прямыми линиями, соединяющими точки двух кривых с одинаковой параметризацией (рис. 8).

Например, если две кривые представляют собой окружности одинакового радиуса с общей осью, то соответствующая линейчатая поверхность будет цилиндром. Точка и окружность образуют конус. В общем случае параметрическое уравнение линейчатой поверхности имеет вид

$$\mathbf{P}(u, v) = u\mathbf{P}_1(v) + (1-u)\mathbf{P}_2(v), 0 \leq u \leq 1, 0 \leq v \leq 1.$$

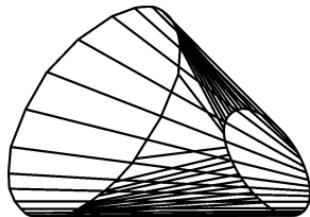


Рис. 8

Лоскут Кунса

Лоскут Кунса (Coons' patch) является обобщением поверхности сдвига и линейчатой поверхности и задается не двумя, а четырьмя *граничными кривыми* $\mathbf{P}_0(t)$, $\mathbf{P}_1(t)$, $\mathbf{Q}_0(t)$, $\mathbf{Q}_1(t)$, образующими замкнутый контур в трехмерном пространстве (рис. 9):

$$\mathbf{P}_0(0) = \mathbf{Q}_0(0) = \mathbf{P}_{0,0},$$

$$\mathbf{P}_0(1) = \mathbf{Q}_1(0) = \mathbf{P}_{0,1},$$

$$\mathbf{P}_1(0) = \mathbf{Q}_0(1) = \mathbf{P}_{1,0},$$

$$\mathbf{P}_1(1) = \mathbf{Q}_1(1) = \mathbf{P}_{1,1}.$$

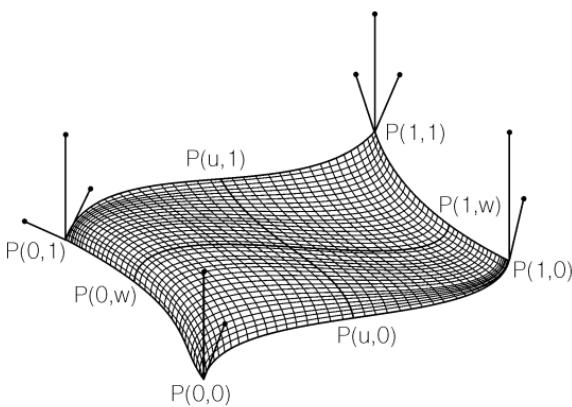


Рис. 9

с помощью следующей параметризации $\mathbf{P}(u, v)$:

$$(1-u)\mathbf{P}_0(v) + u\mathbf{P}_1(v) + (1-v)\mathbf{Q}_0(u) + v\mathbf{Q}_1(u) - (1-u)(1-v)\mathbf{P}_{0,0} - u(1-v)\mathbf{P}_{1,0} - (1-u)v\mathbf{P}_{0,1} - uv\mathbf{P}_{1,1}.$$

В принципе, такая параметризация может быть продолжена и за пределы граничных кривых, но обычно рассматривается только для $0 \leq u \leq 1, 0 \leq v \leq 1$. Отметим, что рассмотренная выше билинейная поверхность представляет собой частный случай лоскута Кунса, в котором граничные кривые не задаются, а считаются прямыми, проходящими через граничные точки.

Эрмитова кривая, бикубическая поверхность и лоскут Фергюсона

Кубическая кривая – основной примитив при работе в САПР. Сегменты разных кубических кривых легко сопрягать друг с другом, обеспечивая для комбинированной кривой G^2 -непрерывность в любой точке. Однако алгебраические a_{ij} коэффициенты уравнения кубической кривой

$$\mathbf{P}(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix} = (t^3 \ t^2 \ t \ 1) \begin{pmatrix} a_{31} & a_{32} & a_{33} \\ a_{21} & a_{22} & a_{23} \\ a_{11} & a_{12} & a_{13} \\ a_{01} & a_{02} & a_{03} \end{pmatrix}$$

не имеют явного геометрического смысла. Поэтому в САПР часто рассматривается специальная форма кубических кривых, называемая

мая Эрмитовой кривой (Hermite curve). Вместо двенадцати скалярных коэффициентов (по четыре для задания многочлена третьей степени для каждой координаты) указываются четыре трехмерных вектора, задающих *граничные условия*. Эти векторы определяют поведение кривой в граничных точках – они описывают координаты начальной (\mathbf{P}_0) и конечной (\mathbf{P}_1) точек сегмента кривой, а также касательных векторов в этих точках (\mathbf{P}'_0 и \mathbf{P}'_1). Примеры Эрмитовых кривых с различными граничными условиями изображены на рис. 10.

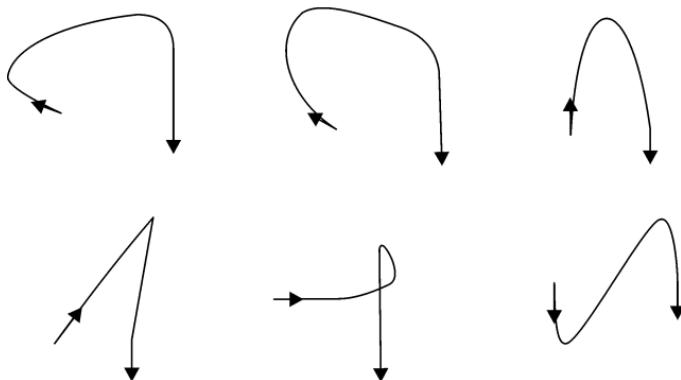


Рис. 10

Нетрудно вывести уравнение Эрмитовой кривой:

$$\mathbf{P}(t) = (1 - 3t^2 + 2t^3)\mathbf{P}_0 + (3t^2 - 2t^3)\mathbf{P}_1 + (t - 2t^2 + t^3)\mathbf{P}'_0 + (-t^2 + t^3)\mathbf{P}'_1,$$

при этом

$$\mathbf{P}(0) = \mathbf{P}_0,$$

$$\mathbf{P}(1) = \mathbf{P}_1,$$

$$\mathbf{P}'(0) = \mathbf{P}'_0,$$

$$\mathbf{P}'(1) = \mathbf{P}'_1.$$

Это уравнение можно записать в матричном виде:

$$\mathbf{P}(t) = (F_1(t) \ F_2(t) \ F_3(t) \ F_4(t)) \begin{pmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}'_0 \\ \mathbf{P}'_1 \end{pmatrix},$$

где многочлены F_1, \dots, F_4 называются *функциями сопряжения* (blending functions):

$$F_1(t) = 2t^3 - 3t^2 + 1,$$

$$F_2(t) = -2t^3 + 3t^2,$$

$$F_3(t) = t^3 - 2t^2 + t,$$

$$F_4(t) = t^3 - t^2.$$

Сегменты Эрмитовых кривых легко сопрягать друг с другом, обеспечивая гладкость C^2 – для этого достаточно задать одинаковые точки и касательные для смежных сегментов.

Бикубическая поверхность (bicubic surface) по аналогии с кубической кривой задается алгебраически с помощью 48 коэффициентов, что, однако, неприемлемо для задания ее геометрической формы. Но если мы определим четыре граничные точки $\mathbf{P}_{00}, \mathbf{P}_{01}, \mathbf{P}_{10}, \mathbf{P}_{11}$, восемь касательных векторов в этих точках $\mathbf{P}_{00}^u, \mathbf{P}_{00}^v, \dots, \mathbf{P}_{11}^u, \mathbf{P}_{11}^v$ и четыре вектора кручения $\mathbf{P}_{00}^{uv}, \dots, \mathbf{P}_{11}^{uv}$, то соответствующую бикубическую кривую можно задать следующим уравнением в матричном виде (заметим, что коэффициенты матрицы сами являются трехмерными векторами, то есть матрица представляет собой тензор третьего порядка):

$$\mathbf{P}(u, v) = (F_1(u) \ F_2(u) \ F_3(u) \ F_4(u)) \begin{pmatrix} \mathbf{P}_{00} & \mathbf{P}_{01} & \mathbf{P}_{00}^v & \mathbf{P}_{01}^v \\ \mathbf{P}_{10} & \mathbf{P}_{11} & \mathbf{P}_{10}^v & \mathbf{P}_{11}^v \\ \mathbf{P}_{00}^u & \mathbf{P}_{01}^u & \mathbf{P}_{00}^{uv} & \mathbf{P}_{01}^{uv} \\ \mathbf{P}_{10}^u & \mathbf{P}_{11}^u & \mathbf{P}_{10}^{uv} & \mathbf{P}_{11}^{uv} \end{pmatrix} \begin{pmatrix} F_1(v) \\ F_2(v) \\ F_3(v) \\ F_4(v) \end{pmatrix},$$

в котором используются определенные выше многочлены F_1, \dots, F_4 . Отметим, что для $\delta, \sigma \in \{0, 1\}$ справедливы следующие равенства:

$$\mathbf{P}(\delta, \sigma) = \mathbf{P}_{\delta\sigma},$$

$$\frac{\partial \mathbf{P}}{\partial u}(\delta, \sigma) = \mathbf{P}_{\delta\sigma}^u,$$

$$\frac{\partial \mathbf{P}}{\partial v}(\delta, \sigma) = \mathbf{P}_{\delta\sigma}^v,$$

$$\frac{\partial^2 \mathbf{P}}{\partial u \partial v}(\delta, \sigma) = \mathbf{P}_{\delta\sigma}^{uv}.$$

Лоскут Фергюсона (Fergusson's patch) является частным случаем бикубической поверхности, в которой векторы кручения в четырех граничных точках считаются нулевыми.

Кривые и поверхности Безье

В 1960-х годах французский инженер Пьер Безье (Pierre Bézier) предложил специальный вид гладкой кривой, названной впоследствии его именем. Независимо его открытие было повторено другим французским инженером – Полем де Кастельжо (Paul de Faget de Casteljau). Именем последнего назван алгоритм линейной аппроксимации кривых Безье. Кривые Безье могут рассматриваться как более интеллектуальные конструкции по отношению к Эрмитовым кривым. Действительно, задавая Эрмитову кривую, мы указываем только ее поведение в концевых точках, но не можем влиять явным образом на форму кривой между этими точками (в частности, кривая может удаляться сколь угодно далеко от отрезка, соединяющего ее концевые точки). В отличие от Эрмитовой кривой кривая Безье задается ломаной линией (называемой *характеристическим многоугольником*), форму которой она повторяет (проходя через две концевые точки и оставаясь полностью внутри характеристического многоугольника). Кривая Безье может иметь произвольную степень, определяемую количеством задающих точек – кривая степени n задается $n+1$ точкой (таким образом, кубические кривые Безье задаются четырьмя точками). Уравнение кривой Безье степени n имеет следующий вид:

$$\mathbf{P}(t) = \sum_{i=0}^n B_{i,n}(t) \mathbf{P}_i, \quad 0 \leq t \leq 1,$$

где \mathbf{P}_i – задающие кривую точки, а $B_{i,n}(t)$ – многочлены Бернштейна, определяемые как слагаемые в биноминальном разложении $(t + (1-t))^n$:

$$B_{i,n}(t) = C_n^i t^i (1-t)^{n-i}.$$

В частности, уравнение кривой Безье третьей степени (рис. 11) имеет вид

$$\mathbf{P}(t) = (1-t)^3 \mathbf{P}_0 + 3t(1-t)^2 \mathbf{P}_1 + 3t^2(1-t) \mathbf{P}_2 + t^3 \mathbf{P}_3.$$

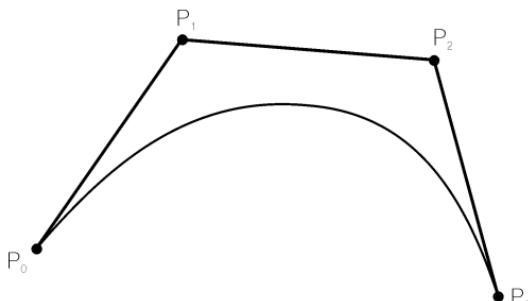


Рис. 11

Поверхности Безье степени nm определяются аналогичным способом по набору из $(n+1)(m+1)$ точек \mathbf{P}_{ij} :

$$\mathbf{P}(u,v) = \sum_{i=0}^n \sum_{j=0}^m B_{i,n}(u) B_{j,m}(v) \mathbf{P}_{ij}, \quad 0 \leq u \leq 1, \quad 0 \leq v \leq 1.$$

Отметим, что граничные кривые для лоскута Безье – $\mathbf{P}(u,0)$, $\mathbf{P}(u,1)$, $\mathbf{P}(0,v)$ и $\mathbf{P}(1,v)$ – являются кривыми Безье (так как $B_{0,k}(0) = B_{0,k}(1) = 0$, $B_{i,k}(0) = B_{i,k}(1) = 1$ для $i > 0$) – рис. 12.

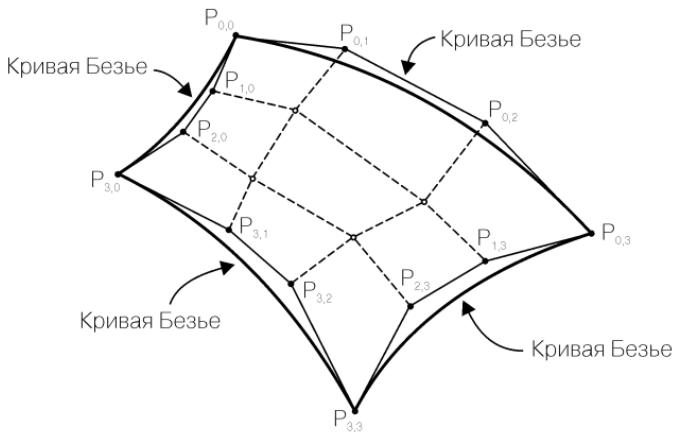


Рис. 12

Алгоритм де Кастельжо

Алгоритм де Кастельжо основан на следующем утверждении: координаты точки кривой Безье $\mathbf{P}(t)$ равны значению \mathbf{P}_0^n , вычисляемому за n шагов с помощью следующей рекуррентной формулы:

$$\mathbf{P}_i^k = (1-t)\mathbf{P}_i^{k-1} + t\mathbf{P}_{i+1}^{k-1},$$

где $\mathbf{P}_i^0 = \mathbf{P}_i$ (точки характеристического многоугольника). Геометрический смысл алгоритма де Кастельжо состоит в соединении отрезком двух соседних точек характеристического многоугольника и разбиении этого отрезка в пропорции $t : (1-t)$. Полученные таким образом точки второго ряда будут более точно аппроксимировать кривую Безье в окрестности точки $\mathbf{P}(t)$; применяя этот алгоритм n раз, мы получим в точности точку $\mathbf{P}(t)$ (см. рис. 13).

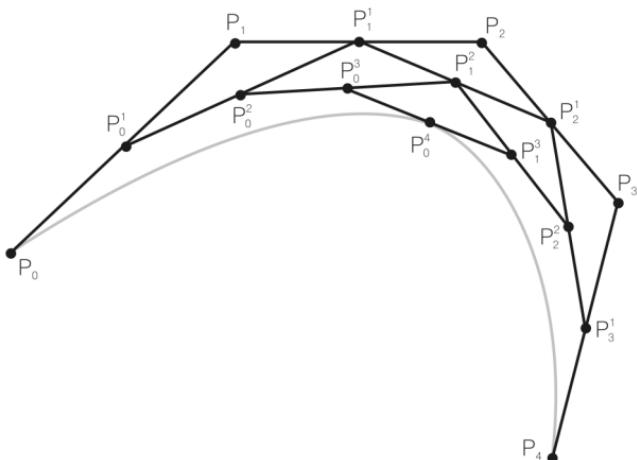


Рис. 13

В-сплайны и В-сплайновые поверхности

Главным недостатком кривой Безье является ее высокая степень, которая растет одновременно с увеличением числа вершин характеристического многоугольника. Однородные В-сплайны (B-spline – сокращение от Basic spline) являются обобщением кривых Безье и позволяют аналогично контролировать форму кривой, одновременно ограничивая ее степень. В-сплайны всегда проходят через первую и последнюю точки характеристического многоугольника и касаются его первого и последнего отрезков. Уравнение В-сплайна степени $k - 1$, определяемого $n + 1$ точками, имеет вид, аналогичный кривой Безье:

$$\mathbf{P}(t) = \sum_{i=0}^n N_{i,k}(t) \mathbf{P}_i, 0 \leq t \leq n - k + 2,$$

где сопрягающие функции $N_{i,k}(t)$ не являются многочленами Бернштейна, а определяются следующим рекурсивным образом:

$$N_{i,k}(t) = \frac{(t - t_i)N_{i,k-1}}{t_{i+k-1} - t_i} + \frac{(t_{i+k} - t)N_{i+1,k-1}}{t_{i+k} - t_{i+1}},$$

$$N_{i,1}(t) = \begin{cases} 1, & \text{если } t_i \leq t \leq t_{i+1}, \\ 0, & \text{в противном случае.} \end{cases}$$

Значения t_i , $0 \leq i \leq n + k$, называются *узловыми* и определяются следующим образом:

$$\begin{aligned} t_i &= 0, & \text{если } i < k, \\ t_i &= i - k + 1, & \text{если } k \leq i \leq n, \\ t_i &= n - k + 2, & \text{если } i > n. \end{aligned}$$

Отметим, что в случае $k = 1$ вместо кривой получаем набор несвязанных точек, в случае $k = 2$ получаем C^0 -кривую, которая в точности повторяет характеристический многоугольник (то есть состоит из прямолинейных сегментов), в общем случае мы имеем дело с кривой гладкости C^{k-2} (рис. 14).

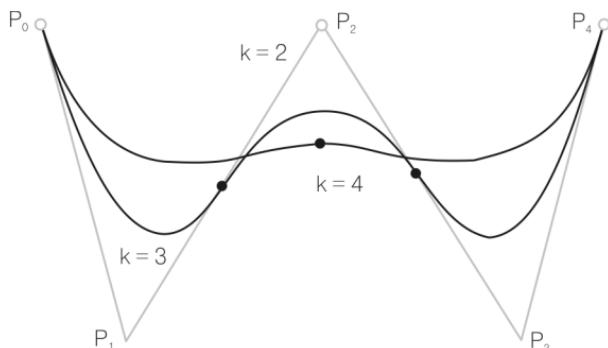


Рис. 14

В-сплайновая поверхность степени $(k - 1)(l - 1)$ строится по $(m + 1)(n + 1)$ точкам:

$$\mathbf{P}(u, v) = \sum_{i=0}^n \sum_{j=0}^m N_{i,k}(u) N_{j,l}(v) \mathbf{P}_{i,j}, \quad 0 \leq u \leq n - k + 2, \quad 0 \leq v \leq m - l + 2.$$

Рациональные кривые и поверхности

Недостатком кривых Безье и В-сплайнов является тот факт, что с их помощью нельзя точно аппроксимировать такие популярные кривые, как конические сечения. Для преодоления этого недостатка было предложено понятие рациональных кривых (Безье и В-сплайнов). *Рациональная кривая Безье* задается следующим образом:

$$\mathbf{P}(t) = \frac{\sum_{i=0}^n h_i B_{i,n}(t) \mathbf{P}_i}{\sum_{i=0}^n h_i B_{i,n}(t)}, \quad 0 \leq t \leq 1.$$

При $h_i = 1$ ($0 \leq i \leq n$) получаем обычную поверхность Безье. С помощью параметров $h_0 = h_1 = 1$ и $h_2 = \cos \theta$ получим дугу окружности, у которой \mathbf{P}_0 и \mathbf{P}_2 являются концевыми точками дуги, а \mathbf{P}_1 – пересечение касательных к окружности в этих точках. Аналогичным образом определяется *рациональный B-сплайн*, получивший в технической САПР-литературе название *NURBS* (non-uniform rational B-spline). Оба понятия без труда расширяются на рациональные поверхности Безье и NURBS.

Интерполяционные кривые и поверхности

Инженеры, работающие с САПР, предпочитают иметь дело с гладкими кривыми и поверхностями, проходящими через задающие их точки. Такие кривые традиционно называются *сплайнами*, а поверхности – *сплайновыми*. Сплайны представляют собой сегменты Эрмитовых кривых или В-сплайновых кривых третьей степени, гладко совмещенных друг с другом (G^2) и проходящих через задающие их точки. Аналогично сплайновые поверхности состоят из гладко спрягающихся бикубических лоскутов или В-сплайновых поверхностей.

Вопросы для самоконтроля

1. Дайте определение C^n и G^n гладкости кривых и поверхностей. Какой класс гладкости является предпочтительным на практике и почему?
2. Что такое билинейный лоскут и лоскут Кунса? Каковы их геометрические свойства?
3. Какие существуют способы задания поверхности по двум кривым?
4. В чем разница между Эрмитовой и кубической кривыми? Выведите формулу задания Эрмитовой кривой.
5. Как задается бикубическая поверхность? Что такое лоскут Фергюсона?
6. Дайте определение кривой Безье. Каковы ее геометрические свойства?
7. Опишите алгоритм де Кастельжо и объясните, как с его помощью можно построить кривую Безье шестой степени.
8. Как задаются однородные В-сплайновые кривые и поверхности?
9. Что такое NURBS? Какие классы кривых и поверхностей описываются с помощью NURBS?

Дополнительная литература

Представлению кривых и поверхностей посвящены главы 6 и 7 книги [8], а также глава 2 в [37]. Интересно ознакомиться также с переведенной на русский язык работой одного из пионеров в этой области (четвертая часть книги [5]). Весьма наглядно представлена информация на Интернет-ресурсе [33]. Подробный исторический обзор можно найти в первой главе книги [25].

Лекция 5

Обмен геометрическими данными

Стандарты обмена	
геометрическими данными	68
Формат IGES	68
Формат DXF	70
Формат STEP	70
Мозаичные модели	71
Формат STL	72
Формат VRML	73
Поверхности подразделения	73
Вопросы для самоконтроля	79
Дополнительная литература	79

Стандарты обмена геометрическими данными

Каждое геометрическое ядро по-своему реализует одни и те же функции твердотельного моделирования. К тому же набор этих функций от ядра к ядру различается. Поэтому важной становится проблема переноса геометрической модели, созданной с помощью одного геометрического ядра, в систему, основанную на другом ядре. Существует множество трансляторов данных из одного представления в другое, но реально независимым от третьих поставщиков способом обмена данными является поддержка каждой CAD-системой *нейтральных форматов* данных. В рамках системы такая поддержка состоит в реализации двух типов конверторов – преобразовать внутренние данные в нейтральный формат (такой конвертор называется *препроцессором*) и преобразовать данные в нейтральном формате во внутренние структуры конкретной системы (*постпроцессор*). Если каждая система реализует пре- и постпроцессинг своих данных в нейтральном формате, то никакие дополнительные конверторы не требуются. Исторически первыми нейтральными форматами геометрических данных стали IGES и DXF.

Формат IGES

Первая спецификация формата появилась в 1980 г. в результате усилий компаний Boeing и General Electric, а в 1981 он был принят в качестве стандарта ANSI. Первая версия была ориентирована в основном на обмен чертежами между системами автоматизации черчения. В версии 2.0 появилась поддержка данных для метода конечных элементов и специфических элементов печатных плат. В версии 3.0 были поддержаны пользовательские макрокоманды, в 4.0 – твердые тела в виде деревьев CSG, в 5.0 – структура BRep. В рамках стандарта поддерживаются три содержательно и структурно эквивалентных формата IGES-файлов – текстовый (ASCII) со строками фиксированной длины, сжатый ASCII и бинарный. Отметим богатую номенклатуру поддерживаемых форматом типов данных:

- твердые тела (начиная с версии 4.0):
 - параллелепипед;
 - прямоугольный клин;
 - прямой круглый цилиндр;
 - прямой круглый конус;

- сфера;
- топ;
- тело вращения;
- тело линейного перехода;
- эллипсоид;
- булево дерево;
- объемный агрегат;
- экземпляр твердотельного объекта;
- поверхности:
 - плоскость;
 - поверхность параметрического сплайна;
 - линейчатая поверхность;
 - поверхность вращения;
 - табулированный цилиндр;
 - поверхность рационального В-сплайна;
 - поверхность смещения;
 - усеченная параметрическая поверхность;
- кривые:
 - дуга окружности;
 - составная кривая;
 - коническая дуга;
 - прямая линия;
 - кривая параметрического сплайна;
 - точка;
 - кривая рационального В-сплайна;
 - кривая смещения;
 - кривая на параметрической поверхности;
- прочие объекты:
 - множественные данные;
 - матрица преобразования;
 - отражение;
 - узел;
 - конечный элемент;
 - точка соединения;
 - узловое отображение и вращение;
 - узловые результаты;
 - элементные результаты.

IGES-файл имеет следующую структуру:

- флаг (для идентификации сжатого ASCII и бинарного формата);

- начало (информация на английском языке о происхождении файла);
- глобальные данные (информация об использованном препроцессоре, символы-разделители, имя файла, количество значащих цифр в записи целых чисел и чисел с плавающей точкой, единицы измерения, дата и время создания файла и пр.);
- запись в каталоге (перечисляются все объекты и их типы);
- параметрические данные (задаются координаты и размеры для перечисленных выше объектов);
- конец (контрольные числа – общее количество записей в каждой секции файла).

Формат DXF

Изначально родной формат системы автоматизации черчения AutoCAD стал де-факто стандартом для обмена чертежами. Этот формат поддерживается практически всеми современными CAD-системами. Текстовый файл в этом формате содержит пять основных разделов, содержимое которых изменяется с каждой новой версией AutoCAD:

- заголовок (номер версии AutoCAD, в которой был приготовлен файл DXF);
- таблица (типы линий и слоев, стили текста и виды чертежа);
- блок (список графических элементов, определенных как группа);
- элемент (главный раздел файла, в котором собственно и описываются все графические элементы – точки, линии, дуги, надписи и пр.);
- конец.

Формат STEP

Стандарт STEP (STandard for Exchange of Product model data) разрабатывается непосредственно международной организацией стандартов (ISO) с 1983 г. В основе его разработки лежат следующие принципы:

- ориентация на данные о продукте, включающие информацию обо всем жизненном цикле (проектирование, производство, контроль качества, испытания и поддержка);
- информация, специфичная для конкретного приложения, хранится в отдельной секции;
- для описания данных используется специальный язык, называемый Express.

Формат одновременно разрабатывается несколькими комитетами и рабочими группами, отвечающими за каждую конкретную часть стандарта. Текущая структура стандарта такова:

- методы описания:
 - обзор и фундаментальные принципы;
 - справочное руководство по языку Express;
- интегрированные информационные ресурсы:
 - интегрированные прикладные ресурсы (черчение, конечно-элементный анализ, кинематика и пр.);
 - интегрированные обобщенные ресурсы (геометрическое и топологическое представление продукта, структурная конфигурация продукта, материалы, структура и свойство процесса и пр.);
 - конструкции, интерпретируемые приложением (реберный каркас, геометрические ограничения, надписи на чертеже и пр.);
- прикладные протоколы и абстрактные испытательные пакеты (черчение, проектирование с использованием конкретного геометрического представления, планирование процесса обработки на станках с ЧПУ и пр.);
- методы реализации (текстовый обменный файл, интерфейсы для доступа к данным на языках C++, Fortran, IDL);
- методологические основы аттестационных испытаний.

Мозаичные модели

Мозаичные, или *фасетные*, модели моделируют поверхности с помощью набора смежных треугольников. Триангуляция произвольной поверхности используется для:

- реалистичной визуализации поверхности (тела) на дисплее компьютера (с учетом перспективы, удаления невидимых граней, отражающих свойств материалов, наложения теней от источников света);
- представления трехмерных данных в нейтральном формате для использования в любом приложении (включая машины для быстрого прототипирования и изготовления);
- расчета других сеточных представлений модели (сетки для метода конечных элементов, четырехугольной базовой сетки для поверхностей подразделения);
- быстрого определения столкновений (collisions) твердых тел.

Мозаичная модель состоит из набора треугольных граней, каждая из которых характеризуется своей нормалью (для различия внешней и внутренней сторон поверхности) и координатами вершин. Для компактного представления мозаичных моделей наборы смежных треугольников организуются в *ленты* и *звезды* (тогда для описания следующего треугольника достаточно указать одну вершину – две другие являются двумя последними вершинами предыдущего треугольника). Отметим, что мозаичная модель является *поверхностной*: она не содержит информации об объемах и твердых телах.

Отметим типичные недостатки мозаичных моделей, созданных различными алгоритмами:

- зазоры – пропуск какой-либо (даже очень тонкой) грани может привести к катастрофическим последствиям (например, в приложениях для быстрого prototyping и изготовления);
- несогласованные нормали соседних ячеек – получаемое из такой поверхностной модели твердое тело оказывается топологически некорректным;
- неправильные нормали – заданная нормаль треугольника может не совпадать с нормалью, рассчитанной по координатам его вершин;
- неправильные пересечения – две треугольные грани могут пересекаться не только по ребрам и вершинам, но и внутри треугольной поверхности;
- внутренние грани – в модели могут присутствовать лишние грани внутри замкнутой поверхности;
- вырождение граней – плоские треугольники (которые могут быть двух видов: совпадение двух вершин или три различные точки на одной прямой) либо треугольники с тремя совпадающими вершинами.

Как правило, все эти ошибки можно обнаружить в STL-файлах, созданных различными системами твердотельного моделирования. Для их исправления разработан ряд алгоритмов, которые доступны в том числе и в виде специализированных коммерческих пакетов программ.

Формат STL

Формат STL (от STereoLithography – название одного из популярных методов быстрого prototyping) является стандартом для хранения мозаичных моделей. В рамках стандарта поддерживаются как текстовая (ASCII), так и бинарная версии файлов. В обоих случаях модель состоит из последовательных записей о треугольниках,

каждый из которых определяется своей нормалью (три числа с плавающей точкой двойной точности) и координатами вершин (девять чисел с плавающей точкой двойной точности). Из плюсов такого представления можно отметить возможность разбиения STL-модели между любыми двумя треугольниками для независимой работы с ними, из недостатков – явную избыточность данных.

Формат VRML

VRML (Virtual Reality Modeling Language) – язык моделирования виртуальной реальности, уже довольно давно применяемый в сети Интернет для описания интерактивной трехмерной графики и мультимедийных приложений. VRML-документ представляет собой обычный текстовый файл, который содержит описания трехмерных фигур и свойств их поверхностей (цвет, текстура материала, освещение и т. п.).

Язык VRML впервые предложен Марком Песке (Mark Pesce) в 1993 г., а его первая спецификация (VRML 1.0) подготовлена на основе формата Open Inventor фирмы SGI и представлена на второй конференции WWW в октябре 1994 г. Дальнейшее развитие проходило уже не только на основе разработок фирмы SGI; к созданию формата подключились такие фирмы, как Sony Research, Mitra и многие другие. Во втором выпуске формата (VRML 2.0) его интерактивные возможности были значительно расширены. Стандарт VRML 2.0 поддерживает анимацию и звуковые эффекты; для него существует поддержка на уровне языков Java и JavaScript. VRML 2.0 был рассмотрен открытой дискуссионной группой и одобрен многими компаниями, а в августе 1996 г. был принят его стандарт. В декабре 1997 г. VRML 2.0 официально заменен на VRML 97. Новый стандарт практически идентичен спецификациям VRML 2.0 с учетом редакционных поправок и некоторых незначительных функциональных различий. Таким образом, текущим VRML-стандартом сегодня является VRML 97, а в работе находится новый формат – VRML 200x. Однако средства и методы представления 3D-графики в Интернет продолжают постоянно развиваться и уже не ограничиваются только языком VRML.

Поверхности подразделения

Очевидным недостатком мозаичных моделей является значительный объем данных, необходимый для представления поверхностей с требуемой точностью. Несмотря на ряд разработанных способов

компактного хранения данных в мозаичной модели, объем ее данных все еще является непреодолимым препятствием для их передачи по сети Интернет. Прорыв в этой области произошел лишь с разработкой теории *поверхностей подразделения* (subdivision surfaces).

Поверхности подразделения представляют собой мозаичные модели, которые итеративно строятся по *базовой сетке* (base mesh), с каждой итерацией приближаясь к форме моделируемой поверхности. Таким образом, две составные части поверхности подразделения – это базовая сетка и алгоритм ее сглаживания. Исторически теория поверхностей подразделения началась с работы американского художника-дизайнера Чайкина (Chaikin), который предложил способ итеративного построения кривой по контрольным точкам. Аналогично Безье, построение кривой Чайкин начинает с характеристического многоугольника, задаваемого набором точек $\{\mathbf{P}_0, \mathbf{P}_1, \dots, \mathbf{P}_n\}$ (рис. 15).

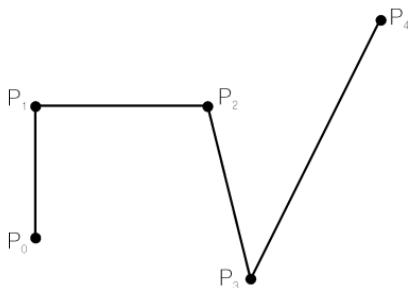


Рис. 15

На следующем этапе образуется новая последовательность контрольных точек $\{\mathbf{Q}_0, \mathbf{R}_0, \mathbf{Q}_1, \mathbf{R}_1, \dots, \mathbf{Q}_n, \mathbf{R}_n\}$ такая, что

$$\mathbf{Q}_i = \frac{3}{4}\mathbf{P}_i + \frac{1}{4}\mathbf{P}_{i+1},$$

$$\mathbf{R}_i = \frac{1}{4}\mathbf{P}_i + \frac{3}{4}\mathbf{P}_{i+1}.$$

Таким образом, точки \mathbf{Q}_i и \mathbf{R}_i делят отрезок $\mathbf{P}_i\mathbf{P}_{i+1}$ в соотношении 1:2:1 (рис. 16).

Процесс продолжается итеративно (рис. 17) до тех пор, пока не будет получена кривая требуемой степени детализации (рис. 18).

Вскоре после изобретения Чайкина было доказано, что генерируемая его алгоритмом кривая есть не что иное, как квадратичный однородный B-сплайн.

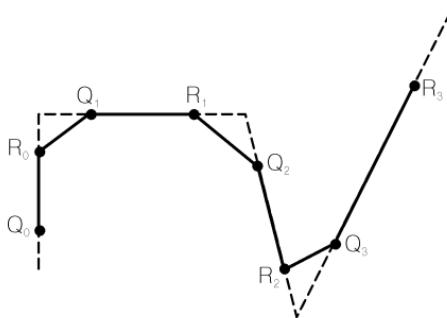


Рис. 16

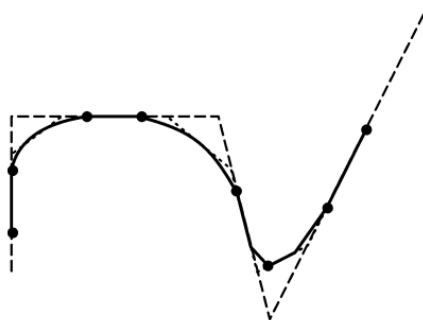


Рис. 17

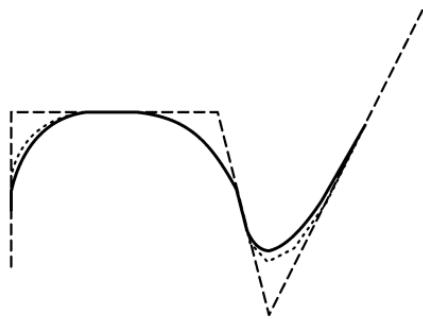


Рис. 18

Метод Чайкина получил название *обрезание углов* (corner cutting) и лег в основу целого семейства алгоритмов, предложенных его последователями. Первым таким алгоритмом стал предложенный Ду (Doo) и Сабиным (Sabin) метод построения квадратичной однородной В-сплайновой поверхности по базовой четырехугольной сетке

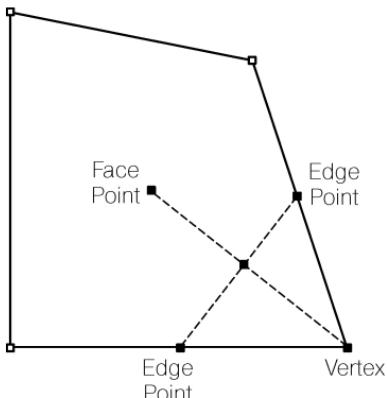


Рис. 19

(каждая грань в такой сетке является выпуклым четырехугольником). Вскоре они смогли распространить свой метод на любые базовые сетки, в которых каждая грань может иметь произвольное число вершин – 3, 4, 5... Полученная поверхность при этом локально (за исключением конечного числа точек) является квадратичным однородным В-сплайном. Метод Ду–Сабина состоит в том, что на очередном шаге каждая грань заменяется гранью меньшего размера с тем же количеством вершин. При

этом каждая вершина уменьшенной грани есть среднее арифметическое исходной вершины (Vertex), центров двух смежных ребер (Edge Points) и центра самой грани (Face Point) – рис. 19.

В результате получается несвязная сетка (рис. 20), в которой затем каждая новая вершина соединяется со всеми другими вершинами, полученными из одной и той же старой вершины, образуя новые грани (рис. 21).

Полученный связный многогранник представляет собой основу для следующего шага алгоритма. Нетрудно видеть, почему метод получил название обрезание углов (см. рис. 22).

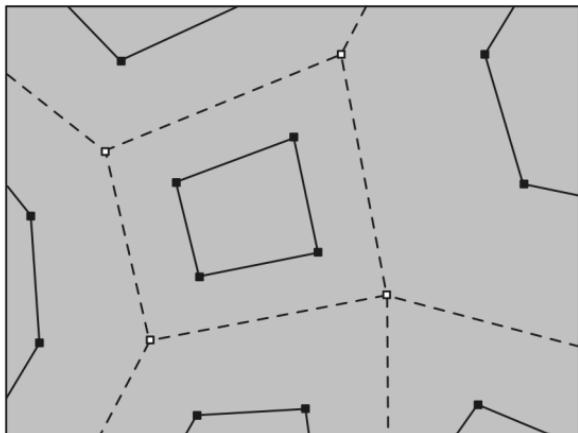


Рис. 20

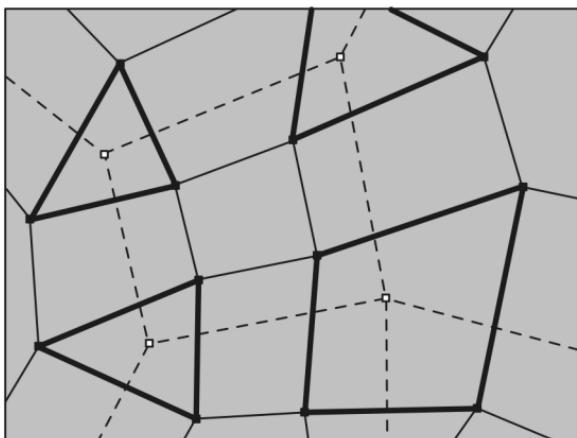


Рис. 21

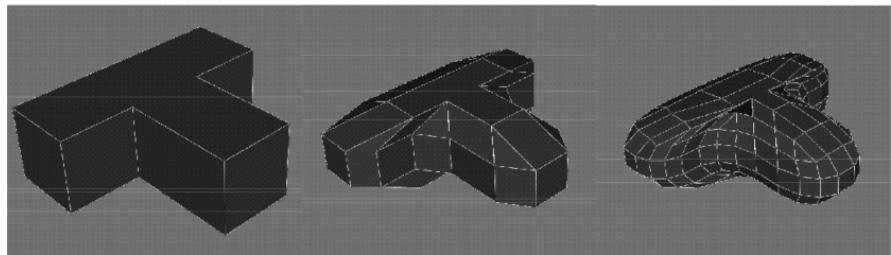


Рис. 22

Аспиранты университета Юты Катмул (Catmull) и Кларк (Clark) смогли расширить метод обрезания углов для построения однородных кубических В-сплайнов. Предложенный ими метод, как и метод Ду–Сабина, может работать на базовых сетках произвольной топологии (получаемая поверхность является локально подобной кубическому В-сплайну). Алгоритм сглаживания состоит в итеративном построении новой сетки, вершинами которой являются

- вершины-грани – центры граней исходной сетки;
- вершины-ребра – центры ребер исходной сетки;
- вершины-вершины – n точек, полученные из вершины каждой n -угольной грани по правилу $\mathbf{P} = \mathbf{Q} + 2\mathbf{R} + \frac{n-3}{n}\mathbf{S}$, где \mathbf{Q} – среднее арифметическое центров граней, смежных с данной гранью,

R – среднее арифметическое центров ребер, смежных с данной вершиной, **S** – исходная вершина.

При этом ребра новой сетки строятся по правилу:

- вершина-грань соединяется со всеми вершинами-ребрами, прообразы которых были смежными исходной грани в исходной сетке;
- вершина-вершина соединяется со всеми вершинами-ребрами, прообразам которых была инцидентна исходная вершина.

Работу метода иллюстрирует рис. 23.

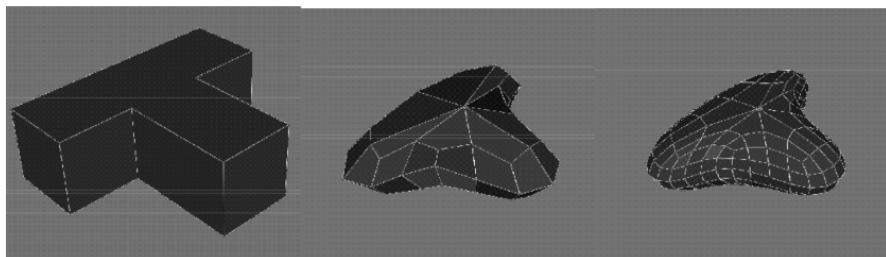


Рис. 23

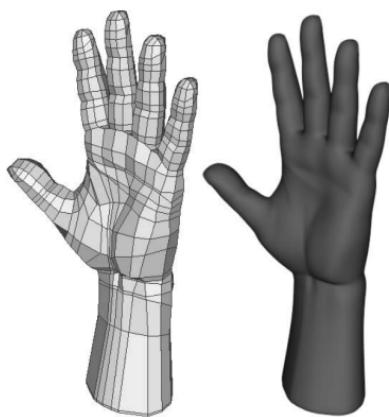


Рис. 24

Поверхности подразделения – удобный способ представления гладких поверхностей компактным образом. Это свойство широко используется для представления различных объектов живой природы, а поэтому хорошо подходит также для описания сложных поверхностей в системах поверхностного моделирования (рис. 24). (Для поддержки негладких сопряжений – острых ребер – используются специальные атрибуты, которые ограничивают область действия алгоритмов подразделения.)

В настоящее время разрабатывается несколько промышленных стандартов обмена геометрическими данными на основе поверхностей подразделения.

Вопросы для самоконтроля

1. Опишите типичные схемы обмена геометрическими данными между CAD-системами.
2. Опишите формат IGES.
3. Опишите формат DXF. Какова область его применения?
4. Опишите формат STEP. В чем его преимущества перед IGES?
5. Что такое мозаичные модели, и каковы области их применения?
6. Опишите формат STL. Каковы его недостатки?
7. Приведите алгоритм Чайкина. Какие кривые строятся с его помощью?
8. Опишите метод Ду–Сабина. Какие поверхности подразделения строятся этим методом?
9. Опишите метод Катмула–Кларка и приведите его отличия от метода Ду–Сабина.

Дополнительная литература

Описанию стандартов обмена данными между САПР-системами посвящена глава 14 книги [8]. Структура STL-файла описана в той же книге в главе 12. Кривые и поверхности подразделения достаточно подробно описаны на многих Интернет-ресурсах, например на [33]. Первоисточники по той же теме представлены в [19], [20] и [17].

Вариационное моделирование: алгебраический подход

Параметры, ограничения и вариационные модели	82
Создание эскизов и проектирование сборок	82
Задача размещения геометрических объектов и ее характеристики	83
Вариационный геометрический решатель	84
Способы алгебраического моделирования геометрической задачи	85
Метрический тензор геометрической задачи	86
Методы символьного упрощения систем алгебраических уравнений	87
Декомпозиция Далмеджа–Мендельсона	88
Метод Ньютона–Рафсона	89
Решение систем линейных уравнений	91
Методы координатного и градиентного спуска	92
Вопросы для самоконтроля	93
Дополнительная литература	93

Параметры, ограничения и вариационные модели

Параметры геометрической модели – это координаты и размеры ее элементов. В параметрических геометрических моделях размеры и положение каждого примитива или конструктивного элемента могут быть изменены, что позволяет быстро получать по существующей модели изделия его модификации. В твердотельных моделях, основанных на CSG-дереве, такая модификация параметров может быть легко реализована путем полного или частичного повторения операций, хранящихся в дереве построения, с новыми значениями параметров.

Геометрическое *ограничение* – это связывание точек, ребер и граней геометрической модели логическим или параметрическим отношением. Примерами ограничений служат инцидентность точки и кривой, касание кривой и поверхности, параллельность двух прямых, расстояние между двумя точками, угол между плоскостями и пр. Ограничение является декларативной (а не конструктивной) конструкцией – оно не задает никакой процедуры расположения одного геометрического элемента относительно другого. Более того, ограничения могут образовывать циклы (например, n попарных расстояний расстояния между n точками), для удовлетворения которым приходится одновременно модифицировать все элементы. Декларативная параметрическая модель с геометрическими ограничениями называется *вариационной*. Параметры ограничений (величины длин и углов) расширяют традиционный набор параметров геометрической модели (размеры и координаты конструктивных элементов). Для удовлетворения ограничениям вариационной модели используются специальные символьные и численные алгоритмы, которые мы разберем ниже.

Создание эскизов и проектирование сборок

Двумя традиционными областями CAD, где в настоящий момент широко используется вариационное моделирование, являются создание плоских эскизов и трехмерных сборок. *Эскиз* (sketch) служит основой для создания большинства конструктивных элементов в системах твердотельного моделирования. Например, чтобы сделать от-

верстие или выемку в детали с помощью продукта CATIA V5 Part Design, пользователь задает плоскость, в которой рисует профиль этого отверстия в виде замкнутого плоского контура, а затем указывает размер выемки/выреза в направлении нормали этой плоскости. Аналогично при создании трехмерного тела путем заметания сначала необходимо сделать двумерный контур. При создании таких контуров (с помощью инструментального средства Sketcher, входящего в Part Design) широко используются двумерные геометрические примитивы (точки, отрезки, дуги, кривые), на взаимное расположение и характерные размеры которых накладываются геометрические ограничения.

При проектировании механизмов, состоящих из нескольких сот или тысяч деталей, пользователь CAD задает ограничения на их взаимное расположение, называемые *ограничениями сборки* (например, с помощью продукта CATIA V5 Assembly Design). Как правило, ограничения выражают свойства сопряжения плоских граней, соосности цилиндрических элементов и т. д.

Задача размещения геометрических объектов и ее характеристики

Задача размещения геометрических объектов (другое ее название – *задача удовлетворения геометрическим ограничениям*) на плоскости (2D) или в пространстве (3D) задается

- набором объектов (каждый объект характеризуется своим типом и начальными значениями параметров);
- набором логических и параметрических ограничений (для параметрических ограничений задаются требуемые значения параметров).

Набор объектов, как правило, включает в себя точки, прямые, окружности, эллипсы и параметрические кривые, а для трехмерных задач – еще и плоскости, аналитические и параметрические поверхности. Параметры объектов – это их координаты и размеры, например: параметрами двумерного эллипса являются координаты его центра, направление главной полуоси и радиусы полуосей (для трехмерного эллипса необходимо также задать направление нормали плоскости эллипса). Логическое ограничение инцидентности и параметрическое ограничение расстояния могут задаваться между двумя любыми объектами (однотипными или разнотипными). Ограничения парал-

лельности, касания и заданного угла могут задаваться только между *направленными* объектами. К таковым традиционно относят все объекты, кроме точки, двумерной окружности и сферы. Два специальных вида ограничения – абсолютная и относительная фиксация. Абсолютная фиксация запрещает изменение положения или ориентации объекта в пространстве задачи, относительная фиксация группирует несколько объектов между собой, запрещая им менять относительные расстояния и углы (такие наборы объектов называются *жесткими множествами*).

Решением геометрической задачи является такое означивание параметров ее объектов, которое удовлетворяет всем заданным ограничениям. Любая геометрическая задача или ее часть может иметь конечное или бесконечное число решений либо не иметь их вообще. Задача без решений называется *переопределенной*, задача с конечным множеством решений называется *хорошо определенной*, а с бесконечным – *недоопределенной*. (Заметим, что на практике ограничения удовлетворяются с некоторыми заданными допусками, поэтому понятия недо-, пере- и хорошей определенности имеют лишь теоретический смысл.) Часто свойство хорошей определенности полезно рассматривать в контексте группы изометрий двумерной плоскости или трехмерного пространства. В частности, задача, которая имеет конечное количество решений по модулю группы изометрий, называется *жесткой* (*rigid*). Двумя важными свойствами геометрической задачи являются *избыточность* и *сингулярность*. Понятие избыточности относится к ограничению – если удаление ограничения не приводит к появлению новых решений задачи, такое ограничение называется избыточным. Сингулярность – свойство не структурное (синтаксическое), но численное, характеризующее такую задачу, бесконечно малое изменение параметра (или группы параметров) которой ведет к изменению структуры ее решений.

Вариационный геометрический решатель

Программная компонента для решения геометрических задач, возникающих при вариационном моделировании, называется *геометрическим решателем*. Как правило, решатель реализует несколько функций, относящихся к геометрической задаче, а именно:

- собственно решение задачи (то есть размещение геометрических объектов в соответствии с заданными ограничениями);

- диагностика пере-, недо- и хорошо определенных частей задачи, а также расчет степеней свободы геометрических объектов;
- динамическое перемещение геометрических объектов в соответствии с наложенными ограничениями;
- автоматическое наложение минимального набора ограничений, делающих задачу хорошо определенной.

Некоторые системы CAD (например, Pro/Engineer, CATIA, T-FLEX) имеют в своем составе собственные геометрические решатели, но большинство коммерческих систем пользуются разработкой английской компании D-Cubed (ныне – дочерняя компания Siemens PLM Software), называемой DCM, Dimensional Constraint Manager. Этот полнофункциональный геометрический решатель представляет собой библиотеку функций для прямого и обратного (callback) вызовов, которые могут быть встроены в любое ядро параметрического моделирования. Решатель существует в двух вариантах – двумерный и трехмерный. Другим доступным вычислительным модулем для разработчиков САПР является решатель LGS (LEDAS Geometric Solver) производства российской компании ЛЕДАС, имеющий не только две версии (2D и 3D), но также различные конфигурации, оптимизированные по набору функций и их стоимости для использования в разных приложениях.

Способы алгебраического моделирования геометрической задачи

Простейшим способом решения геометрической задачи размещения объектов является ее *декартово моделирование*. Каждому объекту сопоставляется набор вещественных координат, которые полностью описывают его положение на плоскости или в пространстве (например, двумерной точке **P** можно сопоставить две координаты – x и y). Каждое ограничение представляется одним или несколькими уравнениями. Например, ограничение расстояния между точками $\mathbf{P}_1 = (x_1, y_1)$ и $\mathbf{P}_2 = (x_2, y_2)$ представляется алгебраическим уравнением $(x_1 - x_2)^2 + (y_1 - y_2)^2 - d^2 = 0$,

где d – параметр ограничения расстояния. Таким образом, по геометрической задаче нетрудно сгенерировать систему алгебраических уравнений с количеством неизвестных, прямо пропорциональным

числу геометрических объектов, и с количеством уравнений, прямо пропорциональным числу ограничений.

Относительное моделирование состоит в связывании с каждым объектом не абсолютных, а относительных координат. В простейшем случае это могут быть трансформации, которые переводят объект из его исходного положения. Однако в ряде случаев количество относительных координат можно существенно сократить. Например, положение точки, инцидентной некоторой прямой, можно описать единственным вещественным параметром, задающим позицию точки в системе координат прямой. В этом случае мы не только сэкономим две лишние переменные (так как иначе точку пришлось бы описывать тремя вещественными параметрами), но и избежим генерации двух лишних уравнений для ограничений инцидентности точки и прямой (так как в данном относительном моделировании это ограничение не может быть нарушено).

Декартово моделирование, несмотря на свою простоту, обладает рядом недостатков. Самый главный из них состоит в том, что для одной и той же задачи в разных системах координат могут быть получены разные решения (которые не эквивалентны друг другу даже в смысле преобразования систем координат). Поэтому рядом исследователей были предложены различные способы *недекартова моделирования*, один из которых мы разберем в следующем разделе.

Метрический тензор геометрической задачи

Все базовые геометрические объекты (точки, прямые, плоскости, сферы, цилиндры, конусы и т. п.), а также все геометрические ограничения над ними могут быть представлены элементами трехмерного аффинного пространства – точками и векторами с наложенными на них метрическими ограничениями – длины и углы. Напомним, что метрическим тензором набора векторов $\{\mathbf{v}_1, \dots, \mathbf{v}_n\}$ называется квадратная симметрическая матрица, элементами которой являются скалярные произведения $(\mathbf{v}_i, \mathbf{v}_j)$. Отметим следующие важные свойства метрического тензора:

- симметричность;
- неотрицательность диагональных элементов (они равны квадратам длин векторов);
- ранг, не превосходящий размерность пространства;

- если сумма некоторых векторов равна нулю, то сумма соответствующих им элементов в любой строке (столбце) метрического тензора тоже равна нулю.

Эти свойства метрического тензора могут быть использованы при моделировании геометрической задачи в терминах точек и векторов. Прежде всего, каждый вектор с неизвестной нормой представляется в виде произведения его длины (она будет переменной алгебраической задачи) и единичного вектора. Из всего набора единичных векторов выбираются три (два для двумерного пространства) базовых, углы между которыми зафиксированы. Все остальные векторы выражаются через выбранный базис ($\mathbf{v} = v_1\mathbf{e}_1 + v_2\mathbf{e}_2 + v_3\mathbf{e}_3$). Для этого в алгебраическую формулировку исходной геометрической задачи добавляется три (два для 2D) неизвестных коэффициента, связанных уравнением

$$v_1^2 + v_2^2 + v_3^2 = 1.$$

Далее, в наборе векторов ищется независимый набор циклов – векторов, сумма которых (некоторые из слагаемых, возможно, взяты с обратным знаком) равна нулю. Для каждого цикла генерируются три (два в 2D) уравнения – сумма коэффициентов соответствующих векторов в разложении по базисному вектору равна нулю. Наконец, осталось учесть заданные углы между векторами. Предположим, между единичными векторами \mathbf{u} и \mathbf{v} задан угол α . Векторы эти имеют следующие разложения по базису $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$ с использованием переменных $u_1, u_2, u_3, v_1, v_2, v_3$:

$$\mathbf{u} = u_1\mathbf{e}_1 + u_2\mathbf{e}_2 + u_3\mathbf{e}_3,$$

$$\mathbf{v} = v_1\mathbf{e}_1 + v_2\mathbf{e}_2 + v_3\mathbf{e}_3.$$

Тогда в алгебраической постановке задачи генерируется следующее уравнение:

$$u_1v_1 + u_2v_2 + u_3v_3 = \cos \alpha.$$

Методы символьного упрощения систем алгебраических уравнений

Так как методы численного решения системы уравнений чрезвычайно дороги (как правило, их трудоемкость растет кубически с ростом размера задачи), активное внимание уделяется символьным методам упрощения систем уравнений. В этом разделе мы разберем методы подстановки, а в следующем – методы декомпозиции.

Простейшим методом символьной подстановки является линейная подстановка переменной. Именно, если в системе существует уравнение, в которое одна из переменных входит только один раз в виде линейного терма, то эту переменную можно выразить из данного уравнения и заменить полученным выражением все ее вхождения в другие уравнения системы. После чего рассматриваемое уравнение и выраженную переменную можно удалить, снизив тем самым размер системы по обоим измерениям. После решения полученной подсистемы значение исключенной переменной получается прямым вычислением ее выражения из исключенного уравнения.

Другие методы символьной подстановки основаны на поиске в системе уравнений специального вида. Конечно, шаблоны для такого поиска существенно зависят от способа алгебраического моделирования геометрических ограничений. Например, при тензорном моделировании все уравнения системы являются линейными или квадратичными, а значит, велика вероятность найти среди них уравнение вида $axy = 0$. Информацию, содержащуюся в таком уравнении, можно использовать двумя способами. Первый состоит в том, что все вхождения термов bxy в другие уравнения системы можно опустить (независимо от значения b). Второй способ состоит в анализе двух возможностей, вытекающих из данного уравнения – подстановки вместо переменной x значения 0, либо той же подстановки для переменной y . Если какая-то из этих подстановок ведет к противоречию (получению уравнения вида $a = 0$ для ненулевой константы a), значит единственной возможностью остается вторая, и изо всех уравнений системы можно удалить термы, содержащие эту переменную.

Наконец, еще одним важным частным случаем символьной редукции является нахождение в системе уравнения вида $\sum a_i x^2 = 0$, где знаки всех a_i одинаковы, и замена всюду в системе входящих в него переменных на нули.

Декомпозиция Далмеджа–Мендельсона

Для описания этого метода нам понадобится ввести понятие *графа уравнений* как двудольного графа, одна доля которого состоит из вершин, соответствующих переменным задачи, а вторая – из вершин, соответствующих уравнениям. При этом вершина-переменная связана ребром с вершиной-уравнением тогда и только тогда, когда соответствующая переменная входит в соответствующее уравнение. На-

помним, что *паросочетанием* (matching) в двудольном графе называется любой набор его ребер, среди которых нет двух инцидентных. Говорят, что вершина входит в паросочетание, если она инцидентна одному из его ребер. *Максимальным* называется паросочетание, которое не содержит ни в каком другом. Далмеж (Dulmage) и Мендельсон (Mendelsohn) показали, что максимальное паросочетание графа уравнений обладает следующими свойствами:

- все уравнения, не входящие в максимальное паросочетание, являются переопределеными (алгебраически лишними) в системе;
- уравнения и переменные, входящие в максимальное паросочетание, образуют максимальную хорошо определенную подсистему;
- переменные, не входящие в максимальное паросочетание, являются недоопределенными (алгебраически лишними).

Максимальное паросочетание определяет также ориентированный граф (все ребра, входящие в паросочетание, ориентируются в обоих направлениях, тогда как ребра, не входящие в паросочетание, ориентируются только от переменных к уравнениям). Компоненты сильной связности в таком ориентированном графе и являются подсистемами, на которые может быть декомпозирована исходная система уравнений (рис. 25).

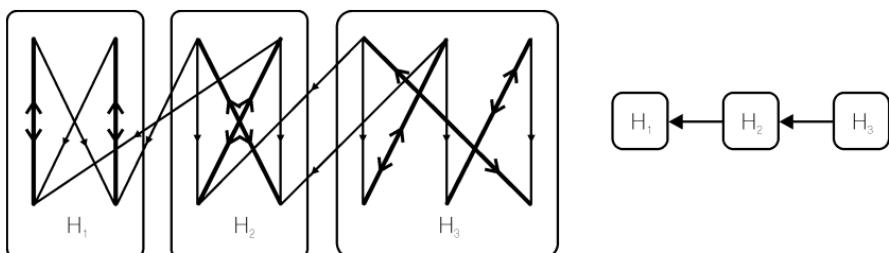


Рис. 25

Метод Ньютона–Рафсона

Традиционным методом решения систем нелинейных уравнений является итеративный *метод Ньютона–Рафсона*, который основан на линейной аппроксимации гладкой функции $F: \mathbf{R}^n \rightarrow \mathbf{R}^m$ в окрестности текущей точки $\mathbf{x}^{(k)}$:

$$F(\mathbf{x}) \approx F(\mathbf{x}^{(k)}) + J_F(\mathbf{x}^{(k)})(\mathbf{x} - \mathbf{x}^{(k)}),$$

где $J_F(\mathbf{x}^{(k)})$ – матрица Якоби (первых частных производных) функции F , вычисленная в точке $\mathbf{x}^{(k)}$ (приближении к решению, достигнутом на шаге k). Согласно этой формуле, если мы хотим найти нуль функции ($F(\mathbf{x}) = 0$), находясь в его окрестности, то от точки $\mathbf{x}^{(k)}$ надо перейти к точке $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \Delta\mathbf{x}$, решив систему линейных уравнений с неизвестными $\Delta\mathbf{x}$:

$$J_F(\mathbf{x}^{(k)})\Delta\mathbf{x} = -F(\mathbf{x}^{(k)}).$$

Несмотря на то, что линейная аппроксимация функции, на которой основан метод Ньютона–Рафсона, справедлива только для достаточно малой окрестности ее нуля, вышеописанный метод стартует из произвольной точки. При этом, как правило, после выполнения небольшого числа итераций (не более пятидесяти) метод сходится к одному из решений уравнения $F(\mathbf{x}) = 0$. Бассейны притяжения каждого нуля в комплексной плоскости образуют области с бесконечно сложными границами, называемые *фракталами* (рис. 26).

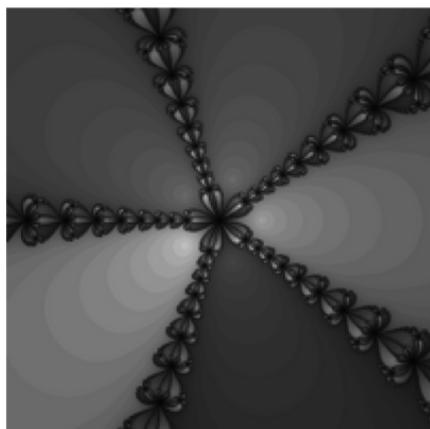


Рис. 26

Из-за непредсказуемости своего поведения (невозможно предугадать, к какому из решений метод сойдется, и как – если решение не подходит – найти другое) метод Ньютона–Рафсона часто используется на практике в комбинации с другими численными методами, которые осуществляют приближение к окрестности желаемого решения.

Решение систем линейных уравнений

Напомним, что на каждом шаге метода Ньютона–Рафсона необходимо решать систему линейных уравнений $Ax = b$. В общем случае мы имеем дело с прямоугольной матрицей A неизвестного ранга. Такая система линейных уравнений может не иметь решений, иметь единственное решение или же бесконечно много решений. Какое же решение подходит для метода Ньютона? Прежде всего среди всех приближений к решению необходимо выбрать то, которое минимизирует норму вектора невязки: $\delta = \|Ax - b\|$. Понятно, что для $\delta = 0$ мы имеем точное решение, в противном случае – приближение к нему. Независимо от конкретного значения δ , в общем случае может существовать бесконечно много векторов x , удовлетворяющих условию $\|Ax - b\| = \delta$. Как известно, данное множество является аффинным пространством: $\Sigma = \{x^* + y \mid y \in \text{Ker } A\}$, где x^* – произвольный вектор, удовлетворяющий $\|Ax^* - b\| = \delta$, а $\text{Ker } A$ – линейное пространство решений однородной системы уравнений $Ax = 0$. Среди всех $x \in \Sigma$ с точки зрения метода Ньютона–Рафсона лучше всего подходит элемент с минимальной нормой ($\|x\| \rightarrow \min$). Таким образом, метод решения систем линейных уравнений, возникающих на каждом шаге метода Ньютона–Рафсона, должен находить решение:

- минимальное по норме вектора невязки,
- минимальное по собственной норме.

Именно такое решение выдает *метод SVD* (singular value decomposition), основанный на представлении произвольной $m \times n$ матрицы A в виде произведения $A = UDV^T$, где U и V – ортогональные $m \times m$ и $n \times n$ матрицы соответственно, а D – диагональная матрица, содержащая сингулярные значения матрицы A .

Метод SVD имеет сравнительно высокую временную сложность. Поэтому на практике вместо него используют другие методы, позволяющие найти какое-либо решение x^* с минимальной невязкой, а также базис пространства $\text{Ker } A$, после чего вычислить ортогональную проекцию x^* на $\text{Ker } A$ и отнять ее из x^* . Очевидно, полученный вектор будет минимальным по норме. В качестве метода нахождения решения x^* и базиса $\text{Ker } A$ можно использовать *исключение Гаусса–Жордана*, с помощью которого вычисляется *RREF-форма* (reduced row echelon form) матрицы A .

Методы координатного и градиентного спуска

Метод Ньютона–Рафсона быстро сходится в окрестности решения, но он имеет два существенных недостатка:

- если начальное приближение далеко от решения, метод может делать много итераций, не приближающихся к решению;
- стоимость одной итерации чрезвычайно высока (кубически зависит от размера системы уравнений).

Чтобы преодолеть оба этих недостатка, метод Ньютона–Рафсона зачастую используют в комбинации с более простыми методами, не требующими решения системы линейных уравнений на каждом шаге. Такими методами являются *метод координатного спуска* (другое его название – *метод релаксации*) и *градиентный метод*. Оба алгоритма являются итеративными методами последовательного приближения и имеют общую схему. На шаге k задача решения системы из m уравнений над n переменными $F(\mathbf{x}) = 0$ сводится к минимизации вещественной функции одного неизвестного $\sum F_i(\mathbf{x}^{(k)}(t))^2 = 0$, где вектор переменных \mathbf{x} заменяется (линейной) вектор-функцией $\mathbf{x}^{(k)}(t)$ от одной переменной t . Решив эту задачу минимизации относительно t (то есть найдя t^* – значение, на котором достигается минимум), полагаем $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)}(t^*)$. Заметим, что вектор-функция $\mathbf{x}^{(k)}(t)$ меняется на каждом шаге алгоритма. Для метода координатного спуска на шаге k она имеет вид:

$$x_i(t) = \begin{cases} x_i^{(k-1)} + t & \text{для } i = k \bmod n, \\ x_i^{(k-1)} & \text{в противном случае,} \end{cases}$$

где $\mathbf{x}^{(k-1)}$ – значение на предыдущем шаге. Таким образом, на каждом шаге фиксируем значения всех переменных, кроме одной (очередной). Для градиентного метода используется линейная функция в направлении градиента:

$$\mathbf{x}(t) = \mathbf{x}^{(k-1)} + t \nabla \mathbf{x}^{(k-1)}.$$

Минимизация неотрицательной функции одной переменной может выполняться численными или аналитическими методами. При решении геометрических задач зачастую приходится иметь дело с системами квадратичных уравнений, для которых $\sum F_i(\mathbf{x}^{(k)}(t))^2$ будет многочленом четвертой степени, минимумы которого находятся с помощью метода Кардано.

Методы координатного и градиентного спуска имеют линейную скорость сходимости (то есть норма вектора невязки убывает линейно с ростом числа шагов), поэтому применять их собственно для поиска решения с высокой точностью нецелесообразно. Однако комбинация одного из этих методов с методом Ньютона–Рафсона (имеющим квадратичную скорость сходимости в окрестности решения) оказывается достаточно эффективной для решения задач с произвольным начальным приближением к решению.

Вопросы для самоконтроля

1. Дайте определение вариационной параметрической модели.
2. Опишите области приложения вариационного моделирования.
3. Дайте определение задаче размещения геометрических объектов и объясните смысл понятий недоопределенность, переопределенность, хорошая определенность, жесткость, избыточность и сингулярность.
4. Что такое вариационный геометрический решатель? Приведите примеры.
5. В чем разница между декартовым и относительным моделированием задачи удовлетворения геометрическим ограничениям?
6. Опишите тензорное моделирование задачи размещения геометрических объектов. В чем состоят его преимущества?
7. Опишите методы символьного упрощения систем алгебраических уравнений.
8. Опишите методы декомпозиции систем алгебраических уравнений.
9. Опишите метод Ньютона–Рафсона. Какие методы решения систем линейных алгебраических уравнений могут использоваться совместно с ним?
10. Опишите методы релаксации и градиентного спуска. В чем их преимущества и недостатки по сравнению с методом Ньютона–Рафсона?

Дополнительная литература

С различными методами вариационного моделирования можно ознакомиться в главе 8 книги [37]. Про некоторые методы, используемые в геометрическом решателе DCM, можно получить представление из работы [34], а для ознакомления с LGS рекомендуется прочитать [22]. Обзорное описание на русском языке методов решения геометрических задач в ограничениях содержится в [16].

Вариационное моделирование: диагностика и декомпозиция задачи

Диагностика геометрических задач ...	96
Методы упрощения	
геометрических задач	96
Определение и классификация	
методов декомпозиции	97
Граф ограничений	97
Методы рекурсивного деления	98
Методы рекурсивной сборки	99
Формирование кластеров	
с помощью анализа графа	
ограничений	100
Формирование кластеров	
на основе шаблонов	102
Эвристическое формирование	
псевдокластеров	103
Распространение степеней	
свободы	103
Вопросы для самоконтроля	103
Дополнительная литература	104

Диагностика геометрических задач

Как мы уже знаем, каждая геометрическая задача (или любая ее подзадача) является переопределенной, хорошо определенной или недопределенной. Однако наряду с этими очевидными характеристиками задачи на практике полезно рассматривать и другие: избыточные ограничения, группы зависимостей и оставшиеся степени свободы геометрических объектов. Напомним, что ограничение называется *избыточным* (redundant), если его удаление не приводит к появлению новых решений задачи. Группа параметрических ограничений называется *зависимой*, если изменение параметра одного из них создает переопределенную задачу, у которой, однако, снова появляются решения, если мы изменим параметр другого ограничения из группы. (Таким образом, параметры ограничений в группе зависят друг от друга.) *Мгновенной степенью свободы* геометрического объекта называется его бесконечно малая трансформация, сохраняющая удовлетворенными все наложенные на объект ограничения.

Для нахождения избыточных ограничений, зависимых групп и объектов со степенями свободы применяются специальные методы *диагностики* геометрической задачи. Все они основаны на *линейной аппроксимации* задачи и сводятся к поиску линейных зависимостей в матрице Якоби (каждая строка которой соответствует какому-то ограничению, а каждый столбец – степени свободы какого-то объекта).

Методы упрощения геометрических задач

Работая с системой CAD, пользователь ожидает линейного увеличения времени отклика системы с ростом размера геометрической модели. Чистый алгебраический подход, однако, не обеспечивает линейную сложность решения задачи – обычно требуемое время кубически зависит от размера входных данных (так как решение системы линейных уравнений, выполняемое на каждой итерации метода Ньютона–Рафсона, имеет кубическую сложность). Такая времененная сложность алгоритмов становится особенно критичной при решении больших задач, содержащих сотни и тысячи ограничений. Чтобы ускорить время решения задачи, было предложено много разных методов, которые упрощают исходную геометрическую задачу, разбивая ее на совокупность более простых. Отметим, что зачастую такая декомпозиция может состоять в выделении одной нетривиальной

подзадачи и нескольких тривиальных, решения которых получаются из решения нетривиальной подзадачи путем использования несложных формул. С реализационной точки зрения такую группу методов декомпозиции можно рассматривать отдельно как методы упрощения, однако с математической точки зрения это является именно декомпозицией, то есть разбиением на подзадачи.

Определение и классификация методов декомпозиции

Декомпозиция задачи удовлетворения геометрическим ограничениям – это ее разбиение на несколько подзадач с частичным порядком между ними и оператором комбинации решений этих подзадач в решение целой задачи. Частичный порядок на подзадачах задает последовательность их решения, например, пустой порядок означает, что все подзадачи могут быть решены одновременно и независимо друг от друга. По принципу своей работы все методы декомпозиции можно разделить на три группы:

- методы рекурсивного деления;
- методы рекурсивной сборки (подразделяются на методы нахождения структурно жестких подзадач и методы, основанные на шаблонах);
- распространение степеней свободы (или методы отсечения).

Отметим также, что рассмотренные в рамках предыдущей лекции методы декомпозиции системы уравнений, сгенерированных по геометрической постановке задачи, тоже можно формально отнести к методам геометрической декомпозиции. Однако мы не будем их рассматривать повторно в данном разделе.

Граф ограничений

Для объяснения принципа работы методов декомпозиции нам понадобится понятие *графа ограничений*. Напомним, что неориентированный мультиграф G – это структура, состоящая из двух множеств: множества вершин V и множества ребер E . Между вершинами и ребрами задано отношение инцидентности. Каждое ребро инцидентно одной, двум или большему количеству вершин. Два ребра называются инцидентными, если они инцидентны общей вершине. Говорят, что между двумя вершинами v_1 и v_2 существует путь, если можно задать последовательность ребер, каждое из которых инцидентно пре-

дидущему ребру в последовательности, первое ребро инцидентно v_1 , а последнее – v_2 . Граф ограничений (constraint graph) задачи удовлетворения ограничениям получается ассоциированием множества вершин с множеством объектов задачи, а множества ребер – с множеством ограничений (ребро инцидентно вершинам, являющимся аргументами соответствующего ограничения). Ниже приведен чертеж геометрической задачи и соответствующий граф ограничений (рис. 27).

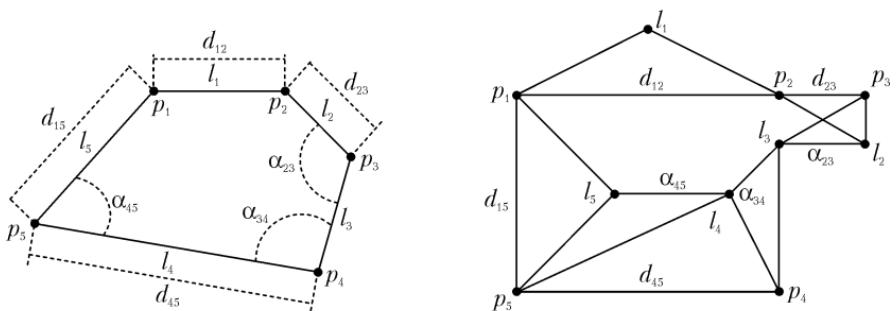


Рис. 27

Методы рекурсивного деления

Методы рекурсивного деления, первоначально предложенные в работах английского ученого Джона Оуэна (John Owen), последовательно разбивают задачу на компоненты одинарной, двойной и тройной связности. Напомним, что неориентированный мультиграф называется *связным*, если между любыми двумя его вершинами существует путь. Несвязный граф распадается на компоненты (одинарной) связности, каждая из которых образует отдельную задачу удовлетворения ограничениям. Понятно, что решения этих задач не зависят друг от друга. Таким образом, простейшим методом декомпозиции является анализ связности графа и разделение его на компоненты связности.

Произвольная вершина связного графа называется *точкой сочленения* (articulation point), если ее удаление из графа (вместе со всеми инцидентными ей ребрами) делает этот граф несвязным. Связный граф, не имеющий точек сочленения, называется *двусвязным графом*. Максимальный двусвязный подграф называется *компонентой двойной связности*. Разделив задачу на подзадачи, соответствующие компонентам двойной связности ее графа ограничений, и решив их по

отдельности, можно затем скомбинировать их решения в решение исходной задачи. Для этого достаточно совместить друг с другом геометрические объекты, соответствующие точкам сочленения графа ограничений.

Произвольная пара вершин в двусвязном графе называется *парой сочленений* (articulation pair), если их удаление (вместе со всеми инцидентными им ребрами) делает граф несвязным. Двусвязный граф, не имеющий пар сочленения, называется *трисвязным графом*. Максимальный трисвязный подграф называется *компонентой тройной связности*. Для декомпозиции геометрической задачи на подзадачи, соответствующие компонентам тройной связности ее графа ограничений, необходимо зафиксировать относительные позиции объектов, соответствующих парам сочленения (для того, чтобы потом «склеить» решенные фрагменты в общее решение, используя пары сочленения).

Методы рекурсивной сборки

Если методы рекурсивного деления декомпозируют задачу «сверху вниз», то методы рекурсивной сборки действуют «снизу вверх», выделяя в задаче хорошо определенные (с точностью до фиксации в пространстве) компоненты. Например, две точки, связанные ограничением расстояния, являются хорошо определенной компонентой. Решение соответствующей подзадачи состоит в произвольном расположении этих точек на заданном расстоянии. Если существует третья точка, связанная с каждой из этих двух ограничениями расстояния, то ее всегда можно однозначно разместить после размещения двух предыдущих (с точностью до двух возможных положений относительно прямой, соединяющей две первые точки). В общем случае методы рекурсивной сборки описываются с помощью следующей метасхемы.

1. Фаза идентификации – найти жесткую (rigid) подзадачу (не имеющую внутренних степеней свободы), называемую *кластером*:
 - шаг слияния: два кластера, связанные достаточным количеством ограничений для формирования жесткой подзадачи, заменяются одним кластером;
 - шаг расширения: геометрический объект, связанный с неким кластером достаточным количеством ограничений для жесткого размещения, добавляется в этот кластер.

2. Фаза сокращения:

- каждый найденный кластер образует отдельную подзадачу; вхождения кластеров друг в друга образуют частичный порядок решения подзадач;
- каждый кластер заменяется в подзадаче, куда он входит, определенным количеством переменных, соответствующим числу его степеней свободы.

Два вышеупомянутых шага повторяются итеративно до тех пор, пока с их помощью не удастся обнаружить новых кластеров. Для нахождения кластеров в исходной задаче используются два разных подхода:

- анализ графа ограничений в соответствии с абстрактными степенями свободы;
- семантический анализ путем поиска соответствия заданным шаблонам.

На рис. 28 ниже приведен пример декомпозиции геометрической задачи на кластеры.

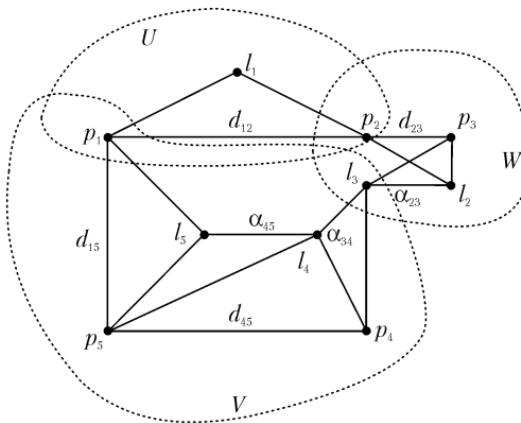


Рис. 28

Формирование кластеров с помощью анализа графа ограничений

Данная группа полиномиальных методов, основанная на понятии структурной жесткости, была впервые предложена и затем эволюционировала в работах группы американских математиков под руко-

водством Кристофа Хоффманна (Christoph M. Hoffmann). *Абстрактной степенью свободы* геометрического объекта является минимально возможное количество параметров, задающее однозначное положение данного объекта в рассматриваемом пространстве. Например, точка имеет две степени свободы в двумерном пространстве и три в трехмерном, прямая имеет две степени свободы в трехмерном пространстве и четыре в трехмерном и т. п. Абстрактной степенью свободы ограничения является число степеней свободы, которое обычно удаляет из задачи данное ограничение. Например, ограничение расстояния между двумя точками удаляет одну степень свободы из задачи, но если параметр расстояния равен нулю, то количество удаляемых степеней свободы составляет два для двумерных задач и три для трехмерных. Задача (или ее подзадача) называется *структурно жесткой*, если суммарное количество степеней свободы всех объектов не превосходит суммы абстрактных степеней свободы ограничений плюс число степеней свободы абстрактного полноразмерного жесткого множества ($D = 3$ в двумерном пространстве, $D = 6$ в трехмерном). Все методы анализа структурной жесткости основаны на вычислении максимального потока в сети, порожденной по графу ограничений:

- сеть состоит из четырех уровней – фиктивная вершина-источник, вершины-ограничения, вершины-объекты и фиктивная вершина-сток;
- фиктивная вершина-источник соединена в такой сети дугами с вершинами, представляющими ограничения; пропускная способность каждой такой дуги равна числу абстрактных степеней свободы соответствующего ограничения;
- каждая вершина-ограничение соединена дугами со всеми вершинами, представляющими объекты-аргументы данного ограничения; пропускная способность каждой из таких дуг не ограничена;
- каждая вершина-объект соединена дугой с фиктивной вершиной-стоком; пропускная способность такой дуги равна числу абстрактных степеней свободы соответствующего геометрического объекта.

Максимальный поток по такой сети демонстрирует оптимальное распределение степеней свободы ограничений по степеням свободы объектов. Для нахождения максимального жесткого кластера на вход каждого ограничения подается дополнительный поток величины D .

Хоффманн предложил три алгоритма декомпозиции, основанных на анализе максимального потока в сети ограничений:

- конденсирующий алгоритм;
- пограничный алгоритм;
- модифицированный пограничный алгоритм.

Все они отличаются друг от друга только фазой сокращения. Конденсирующий алгоритм заменяет найденный кластер в задаче одним жестким множеством (описываемым D степенями свободы), тогда как два других разделяют множество объектов кластера на пограничные и внутренние, не удаляя явно из задачи первое подмножество. Применяя алгоритмы этой группы на практике, следует быть осторожным – структурная жесткость далеко не всегда означает отсутствие внутренних степеней свободы в подзадаче. Самый известный контрпример на эту тему называется «два банана» (рис. 29).

Рядом исследователей были разработаны модификации алгоритмов, более аккуратные в смысле учета жесткости. В частности, жесткость подзадачи можно проверить численно, анализируя ранг матрицы Якоби сгенерированной по соответствующей системе уравнений в разных точках.

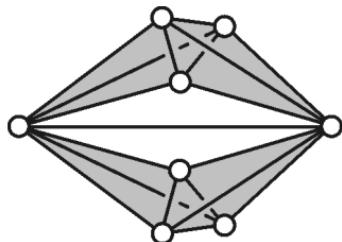


Рис. 29

Формирование кластеров на основе шаблонов

Алгоритмы этой группы основаны на применении правил (шаблонов). Например, две точки и расстояние, равно как две прямые и угол, образуют кластер. Далее, если точка связана двумя ограничениями расстояния с точками из некоторого кластера, то она формирует с кластером жесткую подзадачу. Таким образом, последовательно находя заданные шаблоны в задаче и выделяя их в кластеры, можно идентифицировать многие структурно жесткие подзадачи и использовать эту информацию для декомпозиции.

Эвристическое формирование псевдокластеров

Вышеописанную группу методов можно расширить, если допустить, что кластер может быть не структурно жесткой подзадачей (назовем такие кластеры псевдокластерами). Типичным примером метода является помещение в такой псевдокластер двух жестких множеств, объекты которых обладают следующими свойствами одновременно:

- не являются аргументами неудовлетворенных в начальной позиции ограничений;
- существует как минимум одно ограничение между объектами.

Распространение степеней свободы

Исторически первая группа методов декомпозиции (впервые примененная еще в 1963 г. в первой системе геометрического моделирования Sketchpad) состоит в декомпозиции задачи в порядке, обратном ее решению. Например, через две точки всегда можно провести прямую, а значит, если в задаче присутствует прямая, связанная с другими объектами только двумя ограничениями инцидентности с точками, эту прямую и эти ограничения из задачи можно удалить, сформировав тем самым подзадачу меньшего размера. После решения данной подзадачи прямая всегда может быть размещена относительно двух уже размещенных точек. Существует множество других примеров такого рода *отсечений*, например, склеивание инцидентных однотипных объектов и пр.

Вопросы для самоконтроля

1. Какие понятия и методы используются для диагностики геометрических задач с ограничениями?
2. Что такое декомпозиция геометрической задачи с ограничениями? Для чего она применяется?
3. Классифицируйте известные методы декомпозиции геометрической задачи с ограничениями.
4. Что такое граф ограничений?

5. Что такое точка и пара сочленения? Опишите метод декомпозиции задачи с геометрическими ограничениями, основанный на рекурсивном делении графа ограничений.
6. Опишите общую схему декомпозиции задачи с геометрическими ограничениями методом «снизу вверх».
7. Что такое абстрактная степень свободы и структурно жесткая геометрическая подзадача? Приведите пример структурно жесткой задачи, не являющейся жесткой.
8. Опишите методы, используемые для поиска структурно жестких геометрических подзадач.
9. В чем состоит декомпозиция задачи с геометрическими ограничениями с помощью отсечений?

Дополнительная литература

Обзорная работа [30] по методам декомпозиции геометрических задач предлагает стройную классификацию и исчерпывающее описание известных методов (со ссылками на первоисточники). Метод структурной декомпозиции графа ограничений на компоненты двойной связности описан в [34]. Основополагающая работа о декомпозиции на основе выделения структурно жестких подзадач – [29]. О некоторых методах декомпозиции, реализованных в решателе LGS, можно узнать из [3].

Лекция 8

Инженерия знаний в САПР

Параметрическое проектирование на основе конструктивных элементов	106
Инженерные параметры	108
Отношения базы знаний	109
Параметрическая оптимизация	110
Экспертные знания и продукционные системы	112
Вопросы для самоконтроля	113
Дополнительная литература	114

Параметрическое проектирование на основе конструктивных элементов

Моделирование на основе конструктивных элементов (feature-based modeling) – базовый инструмент проектирования многих современных САПР. Конструктивные элементы (КЭ, features) концептуально представляет собой надстройку над геометрической (чаще всего граничной) моделью. Многие КЭ определяют локальные свойства формы проектируемого изделия (такие как объем, заметаемый протягиванием плоского замкнутого профиля, круглое отверстие, скругление острого ребра) и соответственно имеют непосредственный образ в геометрической модели. Например, в рамках граничного представления каждому КЭ соответствует набор топологических элементов (граней, ребер, вершин), генерируемых данным КЭ (например, круглому отверстию соответствуют порождаемые им цилиндрические грани со своими границами). Такая связь может быть процедурной либо декларативной. В последнем случае каждый КЭ описывается набором геометрических ограничений, которые связывают генерируемые им топологические элементы (задавая их геометрические свойства). Для совместного решения систем таких ограничений требуется производительный вариационный геометрический решатель. В современных САПР, однако, получил широкое распространение процедурный подход – прежде всего из-за простоты реализации и высокой производительности (он не требует одновременного разрешения набора геометрических ограничений). Этот подход состоит в том, что с каждым классом КЭ связываются процедуры, которые вносят соответствующие изменения в геометрическую модель: порождение нового экземпляра, изменение параметров, копирование, удаление. Заметим, что процедурный подход предполагает отсутствие циклических зависимостей между элементами – иначе цикл обновления модели при изменении параметров (состоящий в последовательном применении процедур перестройки КЭ) окажется бесконечным. Этого ограничения можно избежать применением декларативного подхода.

В рамках процедурного подхода КЭ моделируются внутри CAD-системы (например, в системе CATIA V5 за это моделирование отвечает специальный модуль – Feature Modeler) как объекты с заданным *поведением и структурой данных*. Поведение описывается декларативным

цией (и реализацией) набора операций, которые можно применять к данному конструктивному элементу. Структура данных описывается набором атрибутов. Каждый атрибут имеет имя, тип, значение и качество (специальную пометку из множества «входной», «выходной», «нейтральный»). Тип атрибута может быть как простым (например, булевым, целым, вещественным, строковым), так и составным, то есть атрибут сам может быть конструктивным элементом или же ссылкой на таковой. Конструктивные элементы образуют схему прототип–экземпляр. Это значит, что к любому конструктивному элементу можно применять две операции – порождение копии и наследование. Копия конструктивного элемента (называемая его *экземпляром*) полностью наследует поведение и структуру данных исходного объекта (называемого *прототипом*). Однако для созданного экземпляра можно не только изменять значения существующих атрибутов, но также добавлять новые (то есть расширять структуру данных). Экземпляр имеет тот же тип, что и его прототип. К объектам-экземплярам можно рекурсивно применять операции порождения новых экземпляров. Наследование имеет несколько иную природу. При наследовании создается новый тип конструктивных элементов, который полностью наследует поведение и структуру данных своего *супертипа*. При наследовании можно не только добавлять новые атрибуты, но и задавать новые операции (то есть расширять поведение). Обычно операции наследования конструктивных элементов выполняются разработчиками систем CAD, результаты записываются в специальный каталог конструктивных элементов, объекты которого служат прототипами для экземпляров, создаваемых пользователями CAD.

Важной особенностью моделирования конструктивных элементов является наличие встроенного механизма их обновления, управляемого изменением данных (значений атрибутов). Все конструктивные элементы имеют общий базовый супертип, который декларирует две операции – *построения* и *обновления*. Как правило, операция построения реализуется для производных типов конструктивных элементов по-своему, а операция обновления наследуется из супертипа. Операция построения обычно выполняет некоторые действия, связанные с представлением конструктивного элемента в геометрической модели или его отображением на экране. Например, отрезок должен строиться по известным координатам его концевых точек. Задачей операции обновления является пересчет значений выходных атрибутов по значениям входных. Реализация операции обнов-

ления, выполненная в базовом супертипе, вызывает операцию построения для входных атрибутов, а затем – для всего конструктивного элемента. Тем самым обеспечивается совместность структур данных всех конструктивных элементов.

Все конструктивные элементы по умолчанию включены в цикл обновления модели (при необходимости отдельные элементы можно исключить из этого цикла, сделав их неактивными). Это значит, что при любом изменении значения произвольного атрибута (выполненного как непосредственно пользователем, так и любым расчетным алгоритмом) выполняется полный цикл обновления модели в соответствии с операциями построения и обновления для каждого конструктивного элемента.

Инженерные параметры

Параметрические спецификации позволяют пользователям САПР специфицировать и связывать между собой самые разные параметры проектируемого изделия, отражающие его геометрические, физические, экономические свойства. В предыдущих лекциях мы узнали, что параметры в CAD-системах используются при создании примитивов и конструктивных элементов (в этом случае они являются *геометрическими* параметрами – длинами и углами). Кроме того, в системах параметрического проектирования существует возможность задания произвольных параметров, называемых *инженерными*. При создании параметра определяется его имя и тип. Опционально можно задавать область допустимых значений – в виде интервала либо в виде явного перечисления конечного числа допустимых значений. Если в процессе работы системы параметр означен значением, не входящим в область допустимых значений, пользователь получит сообщение об этом. Тип параметра может быть как скалярным (булевым, целым, вещественным, строковым), геометрическим (углом или длиной), так и *размерным*. Размерными типами являются, например, площадь, объем, масса, скорость, сила, температура и т. п. С каждым размерным типом связана своя система измерений, и пользователь может выбрать удобные ему единицы – метры или миллиметры, тонны или граммы. Внутри системы значения всех размерных параметров приводятся к единой шкале (как правило, это миллиметр-грамм-секунда), но пользователь видит их в выбранном им самим масштабе. Система инженерных спецификаций САПР, называемая также *базой знаний* (knowledgeware), позволяет связывать инженерные и геомет-

рические параметры друг с другом посредством инженерных отношений, которые мы разберем ниже. При этом в каждой формуле контролируется соблюдение размерности (например, если пользователь свяжет параметр *площадь* с произведением *длины* и *массы*, он получит системное сообщение о нарушении размерности).

Отношения базы знаний

В CAD-системах для связи параметров между собой используются разные инструменты. Простейшим из них является *функциональное связывание*, которое подразумевает, что значение одного параметра вычисляется по значениям других параметров с помощью заданной формулы. В общем случае в формулу могут входить не только аналитические функции (сумма, произведение, степень, синус и т. п.), но и инженерные, вычисляемые численно по геометрической модели – площадь плоского контура, объем твердого тела, координаты его центра масс и пр. Формула тоже рассматривается как конструктивный элемент, у которого имеются входные атрибуты (параметры, входящие в формулу) и один выходной (параметр, который пересчитывается в соответствии с данной формулой). Операция построения, реализованная для формулы как для конструктивного элемента, выполняет непосредственный численный пересчет значения выходного параметра по значениям входных. Тем самым формула включается в общий цикл обновления модели конструктивных элементов – при изменении значения любого входного параметра значение выходного будет автоматически обновлено. Если в параметрической модели присутствует несколько формул, они не должны образовывать циклических зависимостей (система моделирования конструктивных элементов следит за этим и при необходимости делает часть формул неактивными, сообщая об этом пользователю).

Расширением понятия формулы является *правило* – линейная программа, состоящая из условных операторов и операторов присваивания, которая позволяет менять одновременно значения нескольких выходных параметров. Правила и формулы вместе называются *отношениями базы знаний* (knowledge relations). Другим примером отношения является *проверка*, которая по сути представляет собой формулу с булевым выходным параметром. Изменение значения этого параметра с ИСТИНА на ЛОЖЬ подразумевает инициацию некоторого действия в системе CAD – выдачи окна диалога с сообщением типа: «Силовой кабель проходит слишком близко к нагреваемо-

му элементу» или выполнения некоторого пользовательского сценария, записанного на макроязыке. Еще одним примером отношения базы знаний служит *расчетная таблица* (design table), которая позволяет вычислять значения своих параметров по таблице (например, созданной в Microsoft Excel), указывая номер строки, в которой записаны нужные значения. Возможность работы с формулами, проверками, правилами, расчетными таблицами и другими отношениями базы знаний в полной мере реализована в продукте CATIA V5 Knowledge Advisor.

Недостатком формул, правил и расчетных таблиц является их процедурная направленность – они позволяют вычислять значения выходных параметров (например, напряжения) по значениям входных (силе тока и сопротивлению – в соответствии с законом Ома). С помощью той же самой формулы невозможно проделать обратные операции (например, вычислить силу тока по заданному напряжению и сопротивлению) – для этого придется создать отдельные формулы. Чтобы преодолеть этот недостаток, в системе CATIA V5 был реализован новый тип отношения базы знаний, называемый *множеством уравнений*. Множество уравнений специфицирует отношения между параметрами декларативно, а не процедурно. Пользователь просто задает набор уравнений и неравенств, связывающих параметры, после чего указывает, какие из параметров являются выходными. Система автоматически (с использование специальных итеративных алгоритмов, которые мы разберем в следующей лекции) будет вычислять значения выходных параметров по значениям входных. Для перехода от прямой задачи к обратной достаточно отредактировать список выходных параметров, не меняя при этом никакие ранее введенные уравнения.

Параметрическая оптимизация

Средства оптимизации, встроенные в САПР, как правило, не являются интерактивными (и в этом смысле не похожи на отношения базы знаний, описанные выше). Запрос на оптимизацию инициируется пользователем при необходимости. Однако сама задача оптимизации представляется в виде инженерного конструктивного элемента, который является частью геометрической модели, поэтому ее не приходится каждый раз формулировать заново. Задача оптимизации задается с помощью:

- целевого параметра (это может быть произвольный инженерный или геометрический параметр);

- указания направления оптимизации (минимизировать целевой параметр, максимизировать его, или приблизить к заданному целевому значению);
- выбора среди всех параметров модели тех, значения которых можно изменить для оптимизации цели;
- опционального указания для каждого параметра оптимизации диапазона, в рамках которого можно менять его значение;
- опционального задания списка дополнительных ограничений оптимизации;
- выбора алгоритма оптимизации и настройки его параметров.

Запуск оптимизации осуществляется по команде пользователя. Оптимизация может быть прервана в любой момент, при этом пользователь увидит лучшее из найденных за время работы алгоритма решений. Как правило, существует возможность изучить графики работы алгоритма – как менялось значение целевой функции во время поиска оптимума. Для этих целей можно записать историю работы оптимизации в таблицу в формате Microsoft Excel. После окончания оптимизации и получения результатов (которые состоят из набора новых значений для выбранных параметров оптимизации) пользователь может принять решение или отказаться от него. В последнем случае модель вернется к состоянию, в котором она находилась до запуска оптимизации.

Важно отметить, что в такой постановке задача оптимизации не содержит никаких существенных данных о предметной области – все они берутся из геометрической модели. Например, целевой параметр оптимизации может быть связан инженерными отношениями с другими параметрами (в частности, выражен формулой, связывающей его с массой некоторого твердого тела), в этом случае эти отношения будут автоматически учитываться во время работы алгоритма оптимизации. Сам алгоритм (виды которого мы разберем в следующей лекции) состоит в последовательном изменении значений параметров оптимизации, за которыми следует цикл обновления модели конструктивных элементов. В результате обновления модели устанавливается новое значение для целевого параметра, которое в свою очередь анализируется алгоритмом оптимизации. Заметим, что цикл обновления модели может включать в себя сколь угодно сложные расчеты, например, при изменении значения геометрического параметра профиля конструктивного элемента требуется:

- 1) пересчитать форму этого профиля в соответствии с заданными геометрическими ограничениями (для этого требуется вызвать вариационный геометрический решатель);

- 2) на основе изменившегося профиля выполнить новое построение твердотельного конструктивного элемента;
- 3) вычислить форму других конструктивных элементов, зависящих от изменившегося элемента;
- 4) численно вычислить массу изменившегося твердого тела, и, возможно, учесть иные инженерные отношения, связывающие целевой параметр с параметрами оптимизации.

В результате одна итерация алгоритма оптимизации может занимать значительное время (до нескольких минут и даже часов для моделей со сложной геометрией), что накладывает особые условия на выбор подходящего оптимизационного алгоритма, от которого требуется производить как можно меньше изменений параметров оптимизации. Типичным примером оптимизационного сценария является задача проектирования пластиковой бутылки, которая должна

- иметь строго заданный внутренний объем (0.33, 0.5, 1, 1.5, 2 л);
- иметь достаточно жесткую конструкцию (материал и толщину стенок), позволяющую сохранять форму с налитой жидкостью;
- иметь узнаваемую фирменную форму, разработанную художником-дизайнером;
- иметь определенные внешние размеры и формы для удобного складирования, а также отвечающие соображениям эргономичности (независимо от своего объема бутылка должна удобно размещаться в руке взрослого человека);
- отвечать требованиям удобства утилизации (то есть в случае отсутствия жидкости внутри сминаться до минимального объема).

Все эти требования задаются в виде геометрических конструктивных элементов, определяемых по набору ключевых параметров. Оптимизация в данном случае изменяет набор ключевых параметров для получения бутылки заданного объема по одной и той же геометрической модели.

Более подробно ознакомиться с возможностями параметрической оптимизации в САПР можно на примере продукта CATIA V5 Product Engineering Optimizer.

Экспертные знания и производственные системы

Экспертные правила и проверки отличаются от других отношений базы знаний возможностью их связывания не с конкретными параметрами и конструктивными элементами, а со всей моделью. Более

того, экспертные правила можно группировать в базы правил, которые затем можно применять к любой модели. Типичным примером экспертной проверки является: «Для всех отверстий в модели проверить, что их диаметр не превосходит 20 мм». Экспертное правило может выглядеть так: «Для всех отверстий в модели: если диаметр отверстия равен 50 мм, изменить его на 10 мм, иначе – изменить на 20 мм». Будучи примененным к конкретной геометрической модели, данное правило изменит диаметры всех отверстий (конструктивных элементов соответствующего типа) в ней.

Рассмотрим подробнее возможности работы с экспертными правилами и проверками на примере продукта CATIA V5 Knowledge Expert. С помощью этого продукта инженер может создавать экспертные правила и проверки (подобные описанным выше), группировать их в множества, а множества – в базы правил. Базу правил можно сохранить в специальном каталоге для дальнейшего использования с другими моделями. Кроме того, можно непосредственно применить базу правил, созданную в одной модели, к другой. Для описания экспертных правил и проверок можно пользоваться как специальным языком KWE Language, так и Visual Basic. Важной возможностью системы является генерация проверочного отчета (check report) в виде XML- или HTML-файла, содержащего информацию о всех выполненных проверках.

Заметим, что набор экспертных правил и проверок образует *производственную систему* (rule-based system), поэтому его реализация основана на тех же принципах, что и реализация любой экспертной системы. В частности, при этом используются алгоритмы прямого и обратного выводов, Rete-алгоритм.

Вопросы для самоконтроля

1. Опишите схему «прототип–экземпляр» для моделирования конструктивных элементов.
2. Что такое цикл обновления конструктивного элемента?
3. Для чего используются инженерные параметры?
4. Опишите типичные отношения базы знаний.
5. Что такое параметрическая оптимизация в САПР? Приведите примеры.
6. Что такое «черный ящик» в контексте параметрической оптимизации? Приведите пример цикла обновления модели при оптимизации.
7. Как в САПР задаются экспертные знания?

Дополнительная литература

Параметрическому проектированию на основе конструктивных элементов посвящена пятая глава книги [37]. Использование инженерных ограничений в системе CATIA V5 описано в работе [1].

Методы поиска и оптимизации решения

Задачи удовлетворения ограничениям и оптимизации в ограничениях в общей постановке, их связь	116
Классификация методов поиска и оптимизации решения	117
Метод координатного спуска	118
Метод градиентного спуска	118
Жадный алгоритм	119
Метод Ньютона	119
Методы перебора	120
Методы редукции областей	121
Метод ветвей и границ	123
Алгоритм модельной закалки	124
Генетические алгоритмы	125
Вопросы для самоконтроля	126
Дополнительная литература	126

Задачи удовлетворения ограничениям и оптимизации в ограничениях в общей постановке, их связь

Задача удовлетворения ограничениям представляет собой обобщение понятия системы алгебраических уравнений (или геометрических ограничений) на случай произвольных ограничений над произвольными областями значений. *Многосортная сигнатура* определяет набор *сортов* (имен областей или типов, например *int*, *real*, *bool*, *point*, *line* и т. п.), а также типизированные (по сортам) наборы константных, функциональных и предикатных символов (например, *0*, *3.14*, *true*, *origin*, *sin*, *+*, ***, *distance*, *=*, *<*, *coincidence* и т. п.). Набор константных символов расширяется типизированными символами переменных из множества X . На основе константных символов и символов переменных с помощью функциональных символов можно строить термы (например, $2.5*x + \sin(3.6 + y)$, *distance(origin, p1)*), а на основе термов с помощью предикатных символов – атомарные формулы, называемые также *ограничениями* (например, $2*distance(origin, p1) < 10*x$). *Моделью* заданной сигнатуры Σ является алгебраическая структура, состоящая из множеств-носителей значений каждого сорта (например, \mathbf{R} , \mathbf{Z} , \mathbf{E}^3), а также функций и предикатов, определенных на этих множествах. Тем самым, в модели каждый константный, функциональный и предикатный символ получает определенную математическую интерпретацию. Соответственно каждое ограничение в заданной модели либо удовлетворяется, либо нет. Удовлетворение ограничения означает наличие такого означивания переменных (отображения символов переменных в значения из носителей их сортов), после подстановки которого в формулу, задающую ограничение, и вычисления всех термов получается набор совместных значений для предиката. *Задача удовлетворения ограничениям* (ЗУО, Constraint Satisfaction Problem, CSP) определяется как набор ограничений заданной сигнатуры и множества переменных, для которых требуется проверить одновременную выполнимость в заданной модели. Решение ЗУО – это означивание переменных, которое удовлетворяет всем ограничениям. В общем случае множество решений ЗУО может быть пустым или содержать больше одного элемента (означивания переменных). *Задача оптимизации в ограничениях* (ЗОО, Constraint Optimization Problem, COP) задается в сигнатуре, в которых для выде-

ленного сорта s существует предикатный символ $<$, который в моделях данной сигнатуры трактуется как предикат полного порядка (транзитивное упорядочивание множества значений в носителе сорта s). Для задания ЗОО необходимо задать ЗУО и указать *целевую переменную* выделенного сорта s . Решением ЗОО является такое решение соответствующей ЗУО, которому соответствует минимальное (в смысле интерпретации предикатного символа $<$) значение целевой переменной. Предикат упорядочивания на носителе сорта s позволяет также упорядочить все решения ЗУО (назовем соответствующий предикат на множестве всех означиваний «лучше»). Тем самым имеет смысл говорить о суб-оптимальном решении ЗОО, которое не является оптимальным (то есть решением ЗОО в смысле данного выше определения), но будет *лучше* некоторого известного решения ЗУО. *Задача безусловной оптимизации* (ЗБО) задается одним ограничением вида $x = f(t_1, \dots, t_n)$, где t_1, \dots, t_n – произвольные термы, а x – целевая переменная.

Во многих сигнтурах предикатные символы над любыми сортами могут быть преобразованы в функции над сортом *real*, которые достигают значения 0 только при выполнении предиката. Такие функции называются *невязками*, или *штрафными функциями*, соответствующих ограничений. Например, ограничение $t_1 = t_2$ эквивалентно $t_1 - t_2 = 0$, $\text{coincidence}(p, l)$ эквивалентно $\text{distance}(p, l) = 0$ и т. п. В таком случае задача удовлетворения ограничений может быть сведена к задаче безусловной оптимизации, в которой роль единственного ограничения вида $x = f(t_1, \dots, t_n)$ играет равенство целевой переменной сумме квадратов невязок всех ограничений.

Классификация методов поиска и оптимизации решения

Итак, зачастую задача поиска может быть сведена к задаче оптимизации путем введения штрафных функций. Поэтому целесообразно все алгоритмы разделить на два больших класса: алгоритмы редукции области поиска и алгоритмы оптимизации внутри заданной области. Кроме того, многие методы могут применяться только к задачам с непрерывными областями (представляемыми отрезками вещественной прямой) и гладко дифференцируемыми функциями, в то время как другие применяются только к задачам в дискретной постановке (где важным понятием является конечный набор соседей у любой точки в многомерном пространстве). В принципе, первые алго-

ритмы не являются препятствием для дискретных задач (если у задачи есть обобщение на непрерывные области; после получения оптимального решения в непрерывных областях можно выбрать ближайшую к нему дискретную точку). И тем более дискретные алгоритмы могут без особых проблем применяться к непрерывным задачам – достаточно лишь должным образом дискретизировать область поиска. Тем не менее, начнем мы с классических методов для непрерывных областей – методов координатного и градиентного спуска, а также метода Ньютона, а продолжим описанием методов, традиционно относимых к дискретным областям – перебора, редукции областей, ветвей и границ, модельной закалки и генетическими алгоритмами.

Метод координатного спуска

Идея метода координатного спуска проста – зафиксировав текущие значения всех переменных, кроме одной, свести задачу минимизации функции нескольких переменных к задаче минимизации функции одной переменной. Последняя задача хорошо изучена и имеет много классических методов решения – метод Фибоначчи, метод золотого сечения, метод Ньютона и другие. Более того, если после замены всех переменных, кроме одной, на константные значения и приведения подобных слагаемых целевая функция приобретает вид многочлена пятой или меньшей степени, то задача ее минимизации и вовсе имеет аналитическое решение. Метод координатного спуска (называемый также *релаксацией*) состоит в итеративном применении вышеописанных фиксаций и однопеременных оптимизаций для каждой переменной по очереди. После завершения цикла по всем переменным процесс начинается снова с текущей точки. Метод имеет линейную скорость сходимости в окрестности решения, поэтому не может быть использован для эффективного поиска решения с высокой точностью – для этого необходимо применять методы второго порядка.

Метод градиентного спуска

Градиентный метод, как и метод координатного спуска, является методом первого порядка с линейной скоростью сходимости к решению (при условии попадания в достаточно малую его окрестность). Он основан на вычислении градиента целевой функции и движении в направлении, обратном градиенту. Заметим, что градиент может вычисляться как аналитически, так и численно – в зависимости от

вида целевой функции. Величина смещения в направлении антиградиента может быть задана как эвристически, так и вычислена аналитически или численно путем минимизации целевой функции вдоль градиента. В последнем случае мы опять имеем дело с задачей минимизации функции одной переменной (характеризующей смещение от текущей точки вдоль градиента), как и в методе координатного спуска.

Метод градиентного спуска, пожалуй, – самый популярный оптимизационный алгоритм и является, например, основным инструментом оптимизационного пакета системы CATIA V5 – Product Engineering Optimizer.

Жадный алгоритм

Жадный алгоритм является разновидностью координатного и градиентного спуска для дискретных задач, в которых само понятие направления движения в пространстве параметров может отсутствовать. В этом случае используется понятие *соседних конфигураций* – множества означиваний переменных, в том или ином смысле похожих на текущее означивание. Жадный алгоритм вычисляет значения целевой функции в текущей конфигурации, а также в каждой из соседних и выбирает ту, в которой это значение оказывается меньше. Как и любой метод локальной оптимизации (включая координатный и градиентный спуски), жадный алгоритм не может обойти ловушки локальных минимумов.

Метод Ньютона

Методы второго порядка используют информацию о значениях вторых производных, выражаемую для скалярной функции нескольких переменных в виде матрицы Гессе. Данные методы имеют квадратичную скорость сходимости в окрестности решения и могут применяться для точного вычисления оптимальной точки. Классическим примером метода второго порядка является многопеременный метод Ньютона. При высокой скорости сходимости главный недостаток метода Ньютона состоит в том, что время выполнения одной итерации кубически растет с ростом размерности задачи (из-за сложности обращения матрицы Гессе), становясь неприемлемо большим (на пользовательских сценариях, требующих интерактивной реакции системы) даже для задач относительно скромной размерности – от сотни до тысячи переменных. Чтобы избежать высокой стоимости

итерации, используются так называемые *квазиньютоновские методы*, в которых обратная матрица Гессе, найденная на предыдущем шаге, аппроксимируется в соответствии с новыми значениями.

Методы перебора

Простейшим методом поиска решения является *полный перебор*. На практике такой метод никогда не используется из-за своей высокой сложности – решить этим методом даже простейшие задачи за ограниченное время невозможно.

Метод *хронологического перебора с откатами* (backtracking) основан на понятии частичного означивания переменных. Частичное означивание подразумевает, что мы уже выбрали значения для ряда переменных из их областей определения, но другие переменные остались пока неозначенными. Важной функцией предметной области, позволяющей реализовать схему перебора с откатами, является возможность проверки валидности частичного означивания (то есть проверки, находятся ли уже означенные переменные в противоречии с каким-либо ограничением задачи). Хронологический перебор с откатами устроен следующим образом: алгоритм по порядку означивает *очередную* переменную *очередным* значением из ее области определения (тем самым и набор переменных, и область значений каждой из них предполагаются линейно упорядоченными) и проверяет валидность текущего частичного означивания. Если оно оказалось невалидным, то алгоритм пытается означить последнюю означенную переменную следующим значением из ее области определения. Если других (следующих) значений в области нет, алгоритм рекурсивно выполняет откат к предыдущим переменным до тех пор, пока не найдет переменную, у которой есть еще неиспытанные значения из ее области определения. При отсутствии таких переменных поиск считается законченным. Если переменная найдена и ей присвоено новое значение, то все следующие за ней (в смысле линейного порядка) переменные считаются неозначенными, то есть необходим перебор всех значений из их областей определения.

Хронологический перебор с откатами может быть *эвристическим*. Это значит, что линейное упорядочивание множества переменных и множества значений каждой из них не является статическим, а может меняться на каждом шаге алгоритма. Например, если перебор с откатами интегрирован с каким-либо методом редукции областей (см. ниже), то практически полезной эвристикой оказывается выбор пе-

переменной с наименьшей (в смысле количества значений) областью определения. Свои эвристики существуют и при выборе значений. Например, можно запоминать, сколько раз отвергалось то или иное значение для каждой переменной, и в следующий раз в первую очередь выбирать значение, отвергавшееся наименьшее количество раз.

Вышеописанная схема потому называется хронологическим перебором, что каждый раз при обнаружении противоречия на текущем частичном означивании мы возвращаемся к *последней* рассмотренной конфигурации. В нехронологических схемах точки возврата могут быть иными – например, после нескольких безуспешных попыток продолжить текущее частичное означивание, мы можем решить, что причина противоречия кроется где-то в ранее сделанных выборах, и вернуться к самой первой переменной, выбрав для нее другое значение.

Методы редукции областей

В 1960-х гг. были изобретены методы редукции областей при переборе. Заметим для начала, что означивание переменной можно рассматривать как временное сужение области ее определения до единственного значения. В этом смысле частичное означивание ничем не отличается от начальной постановки задачи – у нас есть переменные, области их значений (некоторые из них, возможно, содержат единственный элемент) и набор ограничений. Требуется проверить, нет ли в каких-то из областей заведомо лишних значений. Предположим, что решаемая задача содержит в числе прочих три переменные x, y , и z с областями значений $D_x = \{0, 1, 2\}, D_y = \{3, 4, 5\}, D_z = \{6, 7, 8\}$, связанные ограничением $x + y = z$. Понятно, что значение 0 для переменной x несовместно ни с какими значениями из указанных областей для переменных y и z . Аналогично, переменная y не может быть означена значением 3, а переменная z – значением 8. Таким образом, без всякого перебора (рассмотрев только одно ограничение задачи) мы сократили области значений переменных x, y и z до $D_x = \{1, 2\}, D_y = \{4, 5\}, D_z = \{6, 7\}$. Подобный метод называется *достижением совместности областей с ограничением*. Сложность метода определяется арностью ограничения (количеством переменных, которые оно связывает) и размером областей значений. В большинстве реальных приложений ограничения бывают унарными, бинарными и тернарными, поэтому на практике сложность можно считать линейно, квадратично или кубически зависимой от размера максимальной области. Алгоритм дос-

тижения совместности по дугам (arc consistency) состоит в нахождении областей, совместных с каждым ограничением модели. В простейшем случае такие области могут быть найдены последовательным рассмотрением каждого ограничения, а затем возвращением к первому ограничению и повторением процесса сначала (так как редукция областей с одним ограничением может сделать их несовместными с другим, даже рассмотренным ранее). Этот итеративный алгоритм можно ускорить, если сделать его управляемым данными, а именно организовать очередь из *активных* ограничений. В начале работы алгоритма все ограничения считаются активными. На каждом шаге алгоритма из очереди извлекается ограничение и выполняется редукция областей его переменных-аргументов, чтобы сделать их совместными друг с другом в смысле данного ограничения. Если при этом какая-то область редуцировалась (то есть количество элементов в ней сократилось), то активируются (становятся в очередь) все ограничения, аргументами которых является соответствующая переменная. Процесс прекращается, когда очередь становится пуста, или когда пустой становится область какой-либо переменной (что означает отсутствие решений в соответствующей задаче удовлетворения ограничениям).

Настоящий алгоритм сформулирован впервые Маквортом (Macworth) в 1977 г. для случая дискретных областей и бинарных ограничений, под названием AC-3. Однако уже в 1981 г. отечественный математик А. Нариньяни, не знакомый в то время с работой Маквorta, предложил значительно более общую концепцию, названную им *недоопределенными моделями*. Суть концепции в том, что область переменной трактуется как ее недоопределенное значение, которое может *доопределиться* во время работы алгоритма. Области могут быть как дискретными, так и непрерывными. Последние представляются пайрой вещественных чисел – нижней и верхней границей соответствующего отрезка вещественной прямой. Для редукции таких *интервальных областей* в соответствии с алгебраическими ограничениями можно использовать правила *интервальной арифметики*. Открытие Нариньяни оказалось незамеченным на Западе и было «переоткрыто» в начале 1990-х гг. под названием «интервальные ограничения». (Интервальные ограничения с тех пор ушли вперед – был предложен ряд специализированных методов, позволяющих более эффективно сужать интервалы.) Подход Нариньяни, однако, предполагает использование произвольных областей (дискретных, непрерывных, составных – таких как структуры, массивы и множества) в рамках од-

ного алгоритма. В 1995 г. Д. Ушаков показал, при каких условиях аппарат недоопределенных моделей Наринъяни является корректным, то есть редуцированные области определяются однозначно и не зависят от порядка рассмотрения ограничений и переменных во время работы алгоритма.

Недоопределенные модели в комбинации с методами перебора, основанными на делении областей переменных (например, на бисекции вещественного интервала), представляют собой мощный аппарат для решения смешанных задач (содержащих как целочисленные, так и вещественные переменные), который по своей простоте и эффективности превосходит иные известные подходы. Кроме того, на основе той же технологии можно проводить оптимизацию (например, используя описанную ниже схему метода ветвей и границ). Конечно, как и любая универсальная технология, метод недоопределенных моделей проигрывает большинству специализированных алгоритмов (в частности, решение систем алгебраических уравнений над полем вещественных чисел лучше выполнять методом Ньютона). Однако в тех случаях, когда необходимо не только найти одно из решений задачи, но и эффективно оценить область, в которой лежат все решения, альтернативы недоопределенным моделям нет. Следует также отметить, что на основе этого аппарата в системе CATIA V5 было реализовано отношение базы знаний Set Of Equations, а также новый метод оптимизации (названный Constraint Satisfaction) в рамках продукта CATIA V5 Product Engineering Optimizer.

Метод ветвей и границ

Метод ветвей и границ является метаметодом в том смысле, что может быть встроен в любую переборную схему с частичным означиванием переменных. Кроме того, алгоритм должен уметь быстро вычислять нижнюю оценку значений целевой функции для каждого частичного означивания. Нижняя оценка частичного означивания – это любое число, меньшее либо равное значению целевой функции на любом полном означивании переменных, полученным продолжением текущего означивания. Предполагается, что вычисление такой нижней оценки (она не обязательно должна быть точной) выполняется достаточно быстро и не требует перебора всех полных означиваний, получаемых продолжением текущего неполного означивания. Одним из методов получения нижней оценки является редукция областей значений переменных, описанная выше. Метод ветвей и гра-

ниц оперирует понятием «текущего рекорда» – наилучшим найденным значением целевой функции. В качестве начального значения текущего рекорда можно взять значение целевой функции на случайной конфигурации. Каждый раз, когда выбранная переборная схема означивает очередную переменную, производится оценка нижней границы целевой функции на данном частичном означивании. Если эта оценка больше либо равна значению текущего рекорда, продолжать текущее означивание нецелесообразно – нет никаких шансов получить решение, более близкое к оптимальному, чем найденное ранее. В этом случае осуществляется обычный откат в соответствии с выбранной схемой перебора. Иначе означивание продолжается.

Алгоритм модельной закалки

Алгоритм модельный закалки (simulated annealing) является типичным алгоритмом *стохастического локального поиска*. Исследуя все соседние конфигурации (то есть означивания переменных, близкие к текущему), в данном случае мы выбираем не ту, которая дает наилучшее значение целевой функции (как это делает жадный алгоритм), а в зависимости от датчика псевдослучайных чисел можем выбрать любую соседнюю конфигурацию. Однако вероятность выбора конфигурации, дающей лучшее значение целевой функции, больше, чем какой-либо другой конфигурации. Более того, с увеличением числа итераций вероятность выбрать конфигурацию, не являющуюся лучшей среди всех соседей, уменьшается. Более точно, эта вероят-

ность для конфигурации S' , соседней с S , равна $e^{\frac{F(S') - F(S)}{T}}$, где T является настраиваемым параметром алгоритма, называемым «температурой». Этот параметр вначале устанавливается в некоторое большое значение и понижается каждый раз после серии итераций, когда падает темп снижения целевой функции при переходе от текущей к следующей конфигурации. Понятно, что при уменьшении температуры вероятность выбрать соседнюю конфигурацию, ухудшающую текущую, падает. Данный алгоритм является прямым аналогом математического моделирования физического процесса закалки (отжига) твердого тела, при котором изделие сначала нагревается до некоторой высокой температуры, которая затем скачкообразно снижается (позволяя системе достигать термодинамического равновесия на каждом шаге).

Алгоритм модельной закалки реализован во многих системах САПР, включающих средства оптимизации, например, в продукте

CATIA V5 Product Engineering Optimizer. Кроме того, имеются успешные примеры использования алгоритма модельной закалки для решения таких задач, как оптимальная компоновка деталей на плоском листе заготовок, распознавание образов, оптимизация маршрутов, проектирование СБИС и др.

Генетические алгоритмы

Генетические алгоритмы, относящиеся к группе алгоритмов стохастического поиска, моделируют на виртуальном уровне процесс эволюции биологической жизни. В таких алгоритмах каждое означивание переменных трактуется как живой организм (особь) со своей уникальной структурой хромосом. «Структура хромосом» в данном случае представляется в виде набора генов – числа фиксированной длины, записанного в двоичном виде. Таким образом, для работы с генетическими алгоритмами задачи с непрерывными областями должны быть дискретизированы, причем отображение множества дискретных означиваний в «хромосомы» должно быть биекцией. Набор конкретных особей образует популяцию (число особей в популяции является настраиваемым параметром). Работа генетического алгоритма состоит в том, что на каждом шаге по текущей популяции порождается следующая. Процесс останавливается, когда большинство особей в популяции становятся похожими друг на друга. Новая популяция порождается предыдущей с помощью процесса, называемого *кроссовером*. (Из школьного курса общей биологии известно, что в процессе мейоза при конъюгации гомологичных хромосом происходит обмен перекрестными участками гамет, который именуют английским словом crossover.) Виртуальный кроссовер в данном случае состоит в том, что две родительские хромосомы разбиваются в произвольном месте на части и обмениваются своими «хвостами», в результате чего получаются две новые особи. Согласно законам эволюции, более приспособленные организмы получают больше шансов произвести потомство. В терминах генетического алгоритма «более приспособленный организм» означает «означивание переменных, дающее лучшее значение целевой функции». Закон эволюции моделируется датчиком псевдослучайных чисел, при этом вероятность быть выбранным прямо зависит от значения целевой функции на данном означивании. Чтобы бороться с традиционной для алгоритмов локальной оптимизации проблемы сваливания в локальный минимум, генетические алгоритмы реализуют еще один

прием, подсмотренный у природы, – возможность мутации. С некоторой небольшой вероятностью каждый организм может мутировать, то есть изменить свои гены. Реализуется это опять же с помощью датчика псевдослучайных чисел и инвертирования некоторых разрядов в двоичной записи числа, кодирующего означивание переменных.

Вопросы для самоконтроля

1. Дайте общее определение задаче удовлетворения ограничениям.
2. Дайте общее определение задачам условной и безусловной оптимизации.
3. Как можно классифицировать методы поиска и оптимизации решения?
4. Опишите метод координатного и градиентного спуска в применении к непрерывным и дискретным областям.
5. Опишите метод Ньютона для решения оптимизационных задач. Что такое квазиньютоновские методы?
6. В чем отличие хронологического и нехронологических методов перебора? Приведите примеры.
7. Для чего используются методы редукции областей? Опишите их.
8. Опишите метод ветвей и границ. Каковы его достоинства и недостатки?
9. Опишите метод модельной закалки. Каковы области его применения?
10. Опишите общую схему работы генетического алгоритма.

Дополнительная литература

Обзор методов оптимизации, применяемый в САПР, можно найти в девятой главе книги [8]. Методы редукции областей и перебора классифицируются в [15]. Использование интервальных ограничений при работе с «черным ящиком» описано в работах [13] и [9].

Инженерный анализ кинематики

Прямая и обратная задачи	
кинематики механизмов	128
Виды кинематических пар	128
Моделирование механизмов	131
Геометрические измерения	131
Моделирование задачи	
кинематики	132
Дифференциальное уравнение	
движения	133
Натуральный градиент уравнения	134
Алгоритмы численного решения	
дифференциальных уравнений	135
Планирование движения	136
Вопросы для самоконтроля	137
Дополнительная литература	138

Прямая и обратная задачи кинематики механизмов

Кинематика – область механики, изучающая движения недеформируемых твердых тел в геометрическом смысле – без учета массы тел и действующих на них сил. Анализ кинематики механизмов является важной частью систем автоматизации проектирования. Здесь *механизмом* (манипулятором, роботом) мы будем называть систему тел, предназначенную для преобразования движения одного или нескольких тел в требуемое движение других твердых тел. Если в преобразовании движения участвуют жидкые или газообразные тела, то механизм называется гидравлическим или пневматическим (такие механизмы мы рассматривать не будем).

Группа неподвижно соединенных твердых тел или отдельное твердое тело называется *звеном*. Подвижное соединение звеньев называется *кинематической парой* (kinematics joint). Каждая кинематическая пара моделирует контакт звеньев и, следовательно, накладывает ряд ограничений на их взаимное положение (то есть оставляя только некоторые из шести степеней свободы твердого тела в трехмерном пространстве).

Управлением механизмом называется задание закона движения одного или нескольких звеньев (такие звенья называются *ведущими*), чтобы движения остальных звеньев были целесообразными (с точки зрения предназначения механизма) относительно неподвижных звеньев (стоеч). Количество ведущих звеньев определяется степенью подвижности механизма.

Прямая задача кинематики состоит в определении положения каждого звена в каждый момент времени в зависимости от заданного управления механизмом. *Обратная задача кинематики* состоит в вычислении положения ведущих звеньев для достижения заданного положения ведомых.

Виды кинематических пар

Рассмотрим типичные кинематические пары и задаваемые ими степени свободы на примере продукта CATIA V5 Digital Mock-Up Kinematics Simulator (табл. 3).

Таблица 3

Кинематическая пара	Иллюстрация	Степени свободы	Управление
Вращательная пара		Одно вращение	Угол
Призматическая пара		Одно смещение	Длина
Цилиндрическая пара		Одно вращение и одно смещение	Угол и длина
Сферическая пара		Три вращения	–
Планарная пара		Два смещения и одно вращение	–
Жесткая пара		Нет степеней свободы	–
Прокатная кривая пара		Одно вращение и одно смещение	Длина

Таблица 3 (окончание)

Кинематическая пара	Иллюстрация	Степени свободы	Управление
Скользящая кривая пара		Два вращения и одно смещение	–
Пара типа «точка – кривая»		Три вращения и одно смещение	Длина
Пара типа «точка – поверхность»		Два смещения и три вращения	–
Универсальная пара		Одно вращение	–
Зубчатая пара		Одно вращение	Один из углов
Реечная пара		Одно вращение или одно смещение	Длина или угол
Кабельная пара		Одно смещение	Одна из длин
Винтовая пара		Одно вращение или одно смещение	Угол или длина
Пара равных угловых скоростей		–	–

Моделирование механизмов

Каждая кинематическая пара может быть выражена с помощью инженерно-геометрических ограничений. Разберем это на примере винтовой пары. Если два звена связаны такой парой, это значит, во-первых, что их оси должны совпадать, а смещение вдоль этой общей оси должно быть согласовано с вращением вокруг нее в соответствии с шагом резьбы. В геометрическом смысле первый факт (соосность) задается ограничением инцидентности между двумя прямыми, каждая из которых принадлежит тому же жесткому множеству, в которое входят все остальные геометрические элементы данной детали. Второй факт (шаг резьбы) моделируется ограничением переменного угла, переменного расстояния и их инженерной связи: расстояние равно углу, разделенному на 360° , и умноженному на величину шага резьбы. (Ограничения с переменными параметрами, называемые также *измерениями*, рассматриваются в следующем параграфе.) Угол задается между двумя прямыми, перпендикулярными осям резьбы и принадлежащими тем же жестким множествам, что и сами детали, а расстояние задается между двумя точками, образованными пересечением перпендикулярных прямых в каждом жестком множестве. Подобным образом моделируются и другие кинематические соединения. (Отметим, что более сложные кинематические связи сначала могут быть представлены в виде нескольких простых – например, шарнир равных угловых скоростей можно представить композицией двух карданных передач.)

Геометрические измерения

Зачастую в геометрических задачах необходимо использовать инженерную информацию, не имеющую прямого геометрического смысла, например: заданное соотношение длин двух отрезков можно смоделировать в общем случае только для некоторых рациональных значений, задача трисекции угла не решается геометрически и т. д. Поэтому для моделирования подобных соотношений проще всего пользоваться понятиями геометрического измерения и алгебраического (или инженерного) уравнения. Под измерением имеется в виду возможность сослаться на любое расстояние или угол в модели, не задавая его значение. Проще всего такая связь может быть организована с использованием понятия алгебраической вещественной переменной. Если в уже известном нам понятии геометрического ограничения с параметром (то есть ограничении расстояния или угла)

заменить постоянный параметр алгебраической переменной, то получим синтаксическую конструкцию, с помощью которой можно определить геометрическое измерение в терминах вариационной модели. Переменные могут быть связаны между собой алгебраическими уравнениями, которые наряду с геометрическими ограничениями становятся частью вариационной модели. Никаких сложностей в поддержке новых понятий в рамках определенных ранее способах решения задачи нет. Большинство видов геометрической декомпозиции переносятся без какого-либо изменения на расширенную формулировку, а декомпозиция с помощью абстрактных степеней свободы может быть естественным образом расширена на алгебраические переменные и уравнения. Тем самым уже описанные методы решения задачи позиционирования геометрических объектов и диагностики их степеней свободы без всяких изменений переносятся и на случай расширенной постановки. Поэтому ниже мы разберем решение задачи кинематической симуляции.

Моделирование задачи кинематики

Положение каждого звена (жесткого множества геометрических элементов) в трехмерном пространстве описывается шестью или более вещественными параметрами (в зависимости от способа параметризации трансформации – углы Эйлера, экспоненциальная параметризация, кватернионы), дополнительные переменные используются для моделирования управления кинематическими парами. Как правило, одни переменные могут принимать значения внутри некоторого замкнутого интервала (например, от 0 до p), а другие – пробегать всю вещественную прямую. Таким образом, положение механизма полностью описывается вектором из n значений. Соответствующее подпространство \mathbf{R}^n в этом случае называется *конфигурационным пространством*. Для моделирования кинематических аспектов данный вектор заменяется вектор-функцией $\mathbf{p}(t)$, $\mathbf{p}: \mathbf{R}^n \rightarrow \mathbf{R}$, где t пробегает значения от 0 до 1. При этом положение звеньев должно удовлетворять наложенным кинематическим связям, описываемым системой из m уравнений $C: \mathbf{R}^n \rightarrow \mathbf{R}^m$. Положение, описываемое в конфигурационном пространстве точкой $\mathbf{p}(0)$, соответствует начальному положению звеньев механизма. Положение $\mathbf{p}(1)$ – целевая конфигурация, которую необходимо найти для решения прямой или

обратной задачи кинематики. В каждый момент времени должно выполняться тождество

$$C(\mathbf{p}(t)) = 0.$$

Целевая конфигурация задается указанием желательных значений для некоторых координат в конфигурационном пространстве: для прямой задачи – требуемых значений переменных управления кинематическими параметрами, для обратной задачи – требуемых значений переменных, задающих положение нужных звеньев в пространстве. В целях получения натурального решения логично задать целевые значения и для всех остальных переменных – равные их начальным значениям. Тем самым будет гарантировано нахождение минимального решения (самого короткого движения механизма). В этих условиях положение целевой конфигурации в конфигурационном пространстве задается точкой \mathbf{p}^* . Заметим, что в общем случае $C(\mathbf{p}^*) \neq 0$. Но даже если целевая конфигурация удовлетворяет всем кинематическим связям, для решения задачи кинематики требуется найти траекторию движения механизма $\mathbf{p}(t)$.

Дифференциальное уравнение движения

Наиболее натуральным способом моделирования движения механизма является представление его в виде решения дифференциального уравнения. В каждой точке траектории осуществляется линеаризация системы уравнений $C(\mathbf{p}(t))$:

$$C(\mathbf{p}(t + dt)) = J_C(\mathbf{p})\mathbf{p}dt,$$

где $J_C(\mathbf{p})$ – матрица Якоби (частных производных) системы уравнений C , вычисленная в точке \mathbf{p} . В предположении $C(\mathbf{p}(t)) \equiv 0$ для всех t получаем $J_C(\mathbf{p})\mathbf{p} = 0$, то есть $\dot{\mathbf{p}} \in \text{Ker } J_C(\mathbf{p})$. Предположим, M – ортонормированный базис $\text{Ker } J_C(\mathbf{p})$. Тогда линейный оператор проектирования на это подпространство будет иметь вид MM^T , а дифференциальное уравнение кинематики запишется как

$$\begin{cases} \dot{\mathbf{p}} = MM^T\mathbf{p}^*, \\ \mathbf{p}(0) = \mathbf{p}_0, \end{cases}$$

то есть вектор скорости движения совпадает с проекцией целевой конфигурации на ядро матрицы Якоби. Тем самым обеспечивается движение, локально не нарушающее кинематические ограничения и в то же время направленное на достижение целевой конфигурации.

Натуральный градиент уравнения

Для решения дифференциального уравнения, описанного в предыдущем параграфе, необходимо научиться дифференцировать уравнения кинематических пар в произвольной точке \mathbf{p} . Эти уравнения связывают координаты ключевых точек, прямых и плоскостей звеньев с помощью геометрических и инженерных ограничений. Геометрические координаты объектов вычисляются по их начальным координатам путем применения трансформаций, задающих положение жестких множеств (звеньев) в трехмерном пространстве. Множество трансформаций образуют группу (с операцией композиции), называемую специальной евклидовой группой $SE(\mathbf{E}^3)$. На этой группе определена метрика (в виде евклидовой нормы разности матриц трансформаций), а значит, она является группой Ли. Таким образом, нам удобней заменить введенную в предыдущем параграфе функцию уравнения $C_i: \mathbf{R}^n \rightarrow \mathbf{R}$ на сложную функцию $C_i(D_k(\mathbf{p}), D_l(\mathbf{p}))$, где $D_j: \mathbf{R}^n \rightarrow SE(\mathbf{E}^3)$, а $C_i: SE(\mathbf{E}^3) \times SE(\mathbf{E}^3) \rightarrow \mathbf{R}$. Таким образом, вектор-функция $\mathbf{D}: \mathbf{R}^n \rightarrow (SE(\mathbf{E}^3))^k$ описывает положения всех k звеньев с помощью трансформаций, а каждое уравнение задает связь между двумя звеньями, используя эти трансформации. *Натуральным градиентом* уравнения $C_i: SE(\mathbf{E}^3) \times SE(\mathbf{E}^3) \rightarrow \mathbf{R}$ назовем вектор $\nabla C(T_1, T_2) = (\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}) \in \mathbf{R}^{12}$, где

$$a_i = \frac{d}{dt} \bigg|_{t=0} C(\exp(t\mathbf{e}_i, 0)T_1, T_2),$$

$$b_i = \frac{d}{dt} \bigg|_{t=0} C(\exp(0, t\mathbf{e}_i)T_1, T_2),$$

$$c_i = \frac{d}{dt} \bigg|_{t=0} C(T_1, \exp(t\mathbf{e}_i, 0)T_2),$$

$$d_i = \frac{d}{dt} \bigg|_{t=0} C(T_1, \exp(0, t\mathbf{e}_i)T_2).$$

Функция \exp задает экспоненциальную параметризацию однопараметрической подгруппы группы Ли $SE(\mathbf{E}^3)$. Натуральный градиент позволяет вычислить частные производные функций C_i по переменным конфигурационного пространства с использованием операций алгебры Ли.

Алгоритмы численного решения дифференциальных уравнений

На практике используют следующие основные итерационные методы решения дифференциальных уравнений движения:

- метод Эйлера;
- классический метод Рунге–Кутта;
- метод Дорманда–Принса (DOPRI5).

Все эти методы являются частными случаями общей схемы Рунге–Кутта для решения дифференциального уравнения с начальным условием:

$$\begin{cases} \frac{dy}{dt} = f(t, y), \\ y(0) = y_0, \end{cases}$$

где $y: \mathbf{R} \rightarrow \mathbf{R}^n$ – искомая, а $f: \mathbf{R}^{n+1} \rightarrow \mathbf{R}^n$ – известная функции. Общая схема Рунге–Кутта определяется так:

$$\begin{aligned} y_{n+1} &= y_n + h \sum_{i=1}^s b_i k_i, \\ k_1 &= f(t_n, y_n), \\ k_2 &= f(t_n + c_2 h, y_n + a_{21} h k_1), \\ &\dots \\ k_s &= f(t_n + c_s h, y_n + \sum_{i=1}^{s-1} a_{si} h k_i). \end{aligned}$$

Здесь s задает порядок схемы, h – величину шага, а константы a_{ij} определяют конкретный метод. При $s = 1$ используется метод Эйлера, решающий дифференциальное уравнение с точностью третьего порядка (от величины шага), при $s = 2$ работает классический метод Рунге–Кутта (обладающий точностью четвертого порядка), а при $s = 7$ применяется метод Дорманда–Принса с точностью пятого порядка. На практике предпочтительней применять последний метод, несмотря на то, что он требует вычисления на каждом шаге большего количества значений.

Планирование движения

Задача планирования движения задается описанием:

- окружения;
- устройства (манипулятора, робота);
- начальной и целевой конфигураций устройства.

Решением задачи является свободный от столкновений допустимый путь. Окружение описывается геометрической моделью, задающей форму и положение всех препятствий. Устройство также описывается геометрической моделью – множеством перемещаемых тел, связанных кинематическими парами. Ключевым в планировании движения является уже известное нам понятие *конфигурационного пространства*, которое представляет собой подмножество пространства параметров, описывающих положение перемещаемого устройства. Например, для описания положения твердого тела в трехмерном пространстве достаточно шести параметров. Число параметров, задающих положение манипулятора, равно числу его степеней свободы, задаваемых кинематическими парами. Препятствия в реальном пространстве описываются областями в конфигурационном пространстве. Таким образом, задача планирования движения сводится к поиску в конфигурационном пространстве траектории точки (представляющей устройство), которая огибает все препятствия (не пересекает соответствующие им области).

Задача поиска траектории в конфигурационном пространстве в общем случае разрешима, но точные алгоритмы ее решения весьма трудоемки. Поэтому были разработаны различные методы, которые представляют препятствия в конфигурационном пространстве лишь приближенно, либо не представляют их вовсе (в этом случае в текущей точке конфигурационного пространства вычисляется минимальное расстояние до препятствия и формируется потенциал, который действует на тело, «отталкивая» его от препятствия). Современный подход к решению задачи планирования движения состоит в использовании понятия случайного поиска *дорожной карты* (roadmap). Дорожная карта – это граф, каждая вершина которого представляет собой точку в конфигурационном пространстве, описывающую положение устройства, не находящегося в контакте с препятствиями, а ребро представляет собой свободный от столкновений путь. Вершины графа находятся случайным образом путем расстановки заданного количества точек в конфигурационном пространстве и проверки их на столкновения с препятствиями. Ребрами соединяются сосед-

ние вершины, если между ними существует тривиальный свободный от столкновений путь (рис. 30).

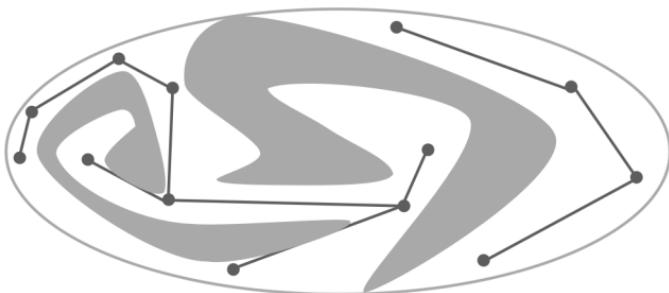


Рис. 30

Таким образом, для построения дорожной карты необходимо уметь выполнять три действия:

- проверить заданную конфигурацию на отсутствие пересечений частей устройства и препятствий;
- вычислить набор простейших манипуляций, необходимых для перемещения устройства из одной точки пространства в другую без учета препятствий;
- проверить что объем, заметаемый устройством при своем движении вдоль пути, не пересекается с препятствиями.

С готовым графом дорожной карты задача планирования движения становится намного проще. Прежде всего, в граф добавляются точки, которые соответствуют начальной и целевой позиции устройства, и ребра, соединяющие их с ближайшими вершинами дорожной карты при наличии тривиального свободного от столкновений пути между ними. Затем задача поиска свободного от столкновений пути сводится к задаче поиска минимального пути на графе. Как правило, после нахождения в конфигурационном пространстве дискретной траектории требуется ее сгладить, чтобы минимизировать движение устройства.

Вопросы для самоконтроля

1. Дайте определение прямой и обратной задачам кинематики.
2. Опишите основные кинематические пары.
3. Как моделируются механизмы в терминах задач удовлетворения ограничениям?

4. Что такое конфигурационное пространство механизма?
5. Каким дифференциальным уравнением описывается движение механизма?
6. Что такое натуральный градиент уравнения кинематической пары, и для чего он нужен?
7. Опишите общую схему численного решения системы обыкновенных дифференциальных уравнений.
8. Как осуществляется планирование движения с помощью дорожной карты?

Дополнительная литература

Хорошее введение в предметную область кинематики механизмов можно получить из учебников по технической механике, например [12]. Представление о методах параметризации группы $SE(\mathbf{E}^3)$ и дифференцирования ее однопараметрических подгрупп можно получить из работы [38]. Использование групп Ли для расчета движения механизма в конфигурационном пространстве описано в [17]. Обзор известных методов планирования движения дан в [31].

Лекция 11

Инженерный анализ динамики

Задача анализа динамики механизмов	140
Движение абсолютно твердого тела в трехмерном пространстве ...	140
Моделирование контакта тел	142
Альтернативный подход:	
уравнения Лагранжа	143
Методы определения столкновений	145
Алгоритмы широкой фазы	145
Алгоритмы фазы сужения	147
Коммерческое программное обеспечение для симуляции движения	148
Вопросы для самоконтроля	148
Дополнительная литература	149

Задача анализа динамики механизмов

Динамика – раздел механики, изучающий общие свойства механического движения системы тел под действием сил. Основной задачей динамики является определение геометрических свойств движения в связи со свойствами заданных физических факторов. Все свойства динамики выводятся из четырех основных аксиом: принципа инерции, основного закона динамики (второй закон Ньютона), закона независимости действия сил и закона равенства действия и противодействия. В данной лекции мы будем рассматривать динамику твердых тел, связанных между собой геометрическими ограничениями, движущихся в поле действия сил, с учетом сил трения и непроникающего неразрушающего столкновения между телами.

Движение абсолютно твердого тела в трехмерном пространстве

Как известно, движение частицы с массой m в поле действия сил $\{f_i\}_{i=1,\dots,n}$ описывается вторым законом Ньютона, постулирующим равенство изменения импульса частицы (определенного произведением ее массы и вектора скорости) действующим на нее силам. В предположении неизменной массы частицы, второй закон Ньютона выглядит так:

$$\frac{d}{dt}mv = m\frac{d^2}{dt^2}\mathbf{r} = \sum_i \mathbf{f}_i,$$

где $\mathbf{r}(t)$ – радиус-вектор, задающий положение частицы в выбранной системе координат в момент времени t , а $\mathbf{v} = \frac{d}{dt}\mathbf{r}$ – ее скорость относительно этой системы координат. (Напомним, что вторая производная радиус-вектора частицы, фигурирующая во втором законе Ньютона, называется ее ускорением.) Непосредственно из второго закона Ньютона следует уравнение, связывающее изменение момента импульса частицы ($\mathbf{r} \times m\mathbf{v}$) с моментами действующих на нее сил ($\mathbf{r} \times \mathbf{f}_i$):

$$\frac{d}{dt}(\mathbf{r} \times (m\mathbf{v})) = \underbrace{\left(\frac{d}{dt}\mathbf{r}\right)}_0 \times (m\mathbf{v}) + \mathbf{r} \times \frac{d}{dt}m\mathbf{v} = \mathbf{r} \times \sum_i \mathbf{f}_i = \sum_i (\mathbf{r} \times \mathbf{f}_i).$$

Уравнение поступательного (без вращений) движения твердого тела получается интегрированием второго закона Ньютона, так как

тело состоит из точечных масс, распределенных по его объему V в соответствии с заданной плотностью ρ . Пусть \mathbf{r}_c – радиус-вектор центра масс тела:

$$\mathbf{r}_c = \frac{1}{M} \iiint_V \mathbf{r} \rho dV,$$

где общая масса тела M определяется интегралом плотности по объему:

$$M = \iiint_V \rho dV.$$

Обозначив скорость центра масс \mathbf{v}_c , получаем

$$m \frac{d}{dt} \mathbf{v}_c = m \frac{d^2}{dt^2} \mathbf{r}_c = \sum_i \mathbf{f}_i.$$

Уравнение вращения тела вокруг его центра масс получается интегрированием уравнения изменения момента импульса для точечных масс:

$$\iiint_V (\mathbf{r} \times \mathbf{v}) \rho dV = \iiint_V (\mathbf{r} \times (\boldsymbol{\omega} \times \mathbf{r})) \rho dV = \iiint_V -\hat{\mathbf{r}}^2 \boldsymbol{\omega} \rho dV = \mathbf{I} \boldsymbol{\omega},$$

где $\boldsymbol{\omega}$ – угловая скорость, а \mathbf{I} – тензор инерции тела:

$$\mathbf{I} = \frac{1}{M} \iiint_V -\hat{\mathbf{r}}^2 \rho dV = \frac{1}{M} \iiint_V \begin{pmatrix} y^2 + z^2 & -xy & -xz \\ -xy & x^2 + z^2 & -yz \\ -xz & -yz & x^2 + y^2 \end{pmatrix} \rho dV.$$

Таким образом, из уравнения изменения момента импульса следует

$$\frac{d}{dt} \mathbf{I} \boldsymbol{\omega} = \mathbf{I} \boldsymbol{\omega} + \left(\frac{d}{dt} \mathbf{I} \right) \boldsymbol{\omega} = \mathbf{I} \boldsymbol{\omega} + \boldsymbol{\omega} \times \mathbf{I} \boldsymbol{\omega} = \sum_i \mathbf{r}_i \times \mathbf{f}_i,$$

где \mathbf{r}_i – точка приложения к телу силы \mathbf{f}_i . Введя обозначения

$$\dot{\mathbf{r}}_c = \frac{d}{dt} \mathbf{r}_c = \mathbf{r}_c \times \boldsymbol{\omega},$$

$$\ddot{\mathbf{r}}_c = \frac{d^2}{dt^2} \mathbf{r}_c = \dot{\mathbf{r}}_c \times \boldsymbol{\omega} + \mathbf{r}_c \times \ddot{\boldsymbol{\omega}},$$

можно выразить закон движения твердого тела в поле действия сил, известный как *система обыкновенных дифференциальных уравнений Ньютона–Эйлера*:

$$\begin{cases} m \ddot{\mathbf{r}}_c = \sum_i \mathbf{f}_i, \\ \mathbf{I}_c \ddot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \mathbf{I}_c \boldsymbol{\omega} = \sum_i \mathbf{r}_c^i \times \mathbf{f}_i. \end{cases}$$

Моделирование контакта тел

Уравнения Ньютона–Эйлера описывают движение изолированного тела. В системе (механизме) тела могут взаимодействовать посредством столкновений друг с другом, а также путем постоянных контактов в виде заданных кинематических пар. Простейший способ моделирования таких контактов состоит в вычислении (тем или иным способом) неизвестных сил, действующих на тела в процессе контакта, и учете их в уравнениях Ньютона–Эйлера. Выделяют три основных подхода к вычислению неизвестных сил в точке контакта:

- пружинно-амортизаторные пары (метод штрафов);
- двусторонние контактные силы;
- односторонние контактные силы;
- повторный импульс.

Метод штрафов состоит в помещении пружинно-амортизаторной связки между контактирующими телами в случае, если одно из тел частично проникает в другое. Такая связка задает силы, действующие на оба тела и отталкивающие их друг от друга. Таким образом, на взаимопроникновение тел накладывается штраф. Когда тела перестают контактировать, пружинно-амортизаторная пара удаляется. Данный метод прост в реализации и зачастую обеспечивает достаточно реалистичное моделирование контакта, но численно нестабилен.

Методы расчета одно- и двусторонних контактных сил строятся путем накладывания временного ограничения фиксации точки контакта, из которого – в предположении нулевого ускорения между контактирующими телами – можно вывести скорости и ускорения, решая соответствующую обратную задачу кинематики. Вычисленные скорости и ускорения позволяют определить силы, действующие на тела в точке контакта.

Метод вычисления повторного импульса основан на вычислении импульса из уравнения момента импульса. Для учета продолжительного контакта используются повторяющиеся импульсы. Тела в точке контакта имеют общую нормаль \mathbf{n} , относительная скорость тел в направлении этой нормали определяет вид контакта – столкновение (ненулевая скорость) или скольжение (нулевая относительная скорость в направлении нормали). Для учета скольжений тел используют метод введения временной кинематической пары (задающей скольжение заданных участков поверхности тел друг по другу). Данная пара действует, пока тела находятся в контакте.

Альтернативный подход: уравнения Лагранжа

С использованием уравнений движения Ньютона–Эйлера моделирование динамики системы из n тел состоит в решении системы из $6n$ обыкновенных дифференциальных уравнений второго порядка (например, методом Эйлера). В данной системе участвуют $6n$ переменных, задающих положение в пространстве каждого из n тел. Кроме того, в зависимости от выбранного метода разрешения контактов требуется решать ту или иную систему линейных уравнений или задачу линейного программирования. В случае движения механизма с небольшим количеством степеней свободы (например, робота-манипулятора) такой подход получается неоправданно дорогим – ведь используя конфигурационное пространство манипулятора (см. материал предыдущей лекции), можно обойтись меньшим количеством переменных и уравнений. Однако уравнения Ньютона–Эйлера в этом случае не подойдут – ведь они описывают движение тела в трехмерном евклидовом пространстве.

Наиболее общая формулировка закона движения механических систем дается так называемым *принципом наименьшего действия* (или *принципом Гамильтона*). Согласно этому принципу каждая механическая система характеризуется определенной функцией

$$L(q_1, q_2, \dots, q_s, \dot{q}_1, \dot{q}_2, \dots, \dot{q}_s, t)$$

или, в краткой записи, $L(\mathbf{q}, \dot{\mathbf{q}}, t)$, причем движение системы удовлетворяет следующему условию. Пусть в определенные моменты времени $t = t_1$ и $t = t_2$ система занимает определенные положения, характеризуемые двумя наборами координат $\mathbf{q}^{(1)}$ и $\mathbf{q}^{(2)}$. Тогда между этими положениями система движется таким образом, чтобы интеграл

$$S = \int_{t_1}^{t_2} L(\mathbf{q}, \dot{\mathbf{q}}, t) dt$$

имел наименьшее возможное значение. Функция L называется *функцией Лагранжа* данной системы, а вышеприведенный интеграл – *действием*. Нетрудно вывести, что минимизация этого интеграла эквивалентна решению системы дифференциальных уравнений, называемых *уравнениями Лагранжа*:

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = 0 \quad (i = 1, 2, \dots, s)$$

или в векторном виде:

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{\mathbf{q}}} - \frac{\partial L}{\partial \mathbf{q}} = 0.$$

Координаты \mathbf{q} , которые задают положение механической системы, называются *обобщенными координатами*. В случае моделирования движения механизма в качестве \mathbf{q} могут быть взяты его степени свободы, например: для двух тел, связанных вращательной парой, положение второго относительно первого может быть задано одним параметром (углом). При таком представлении мы экономим не только на количестве переменных системы, но и на количестве уравнений – ведь саму вращательную пару в таком представлении брать в расчет уже не надо. Производные по времени от обобщенных координат называются *обобщенными скоростями*.

Функция Лагранжа твердого тела равна разности его кинетической и потенциальной энергии:

$$L = \frac{m\mathbf{v}^2}{2} + \frac{I\omega^2}{2} - U.$$

Заметим, что

$$\frac{\partial U}{\partial \mathbf{r}_i} = - \sum_i \mathbf{f}_i.$$

Для нахождения условного экстремума (согласно принципу наименьшего действия) можно воспользоваться *методом множителей Лагранжа*, добавив к функции Лагранжа производные уравнений кинематических связей, умноженные на неизвестный множитель:

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{\mathbf{q}}} - \frac{\partial L}{\partial \mathbf{q}} = J\lambda,$$

где J – матрица Якоби системы уравнений, задающей кинематические связи в терминах обобщенных координат, а λ – вектор неизвестных множителей Лагранжа. Получаемая система содержит меньше уравнений, чем при использовании уравнений Ньютона–Эйлера. Однако для ее решения мы должны также учесть собственно уравнения кинематических связей, то есть решить систему дифференциально-алгебраических уравнений, что является более сложной вычислительной задачей, чем решение системы обыкновенных дифференциальных уравнений.

Методы определения столкновений

Важным моментом при моделировании динамики системы твердых тел является определение их столкновений. На каждом шаге решения обыкновенного дифференциального (или дифференциального алгебраического) уравнения необходимо проверять, произошло ли столкновение каких-либо тел, и вычислять неизвестные силы, возникающие при этом столкновении (одним из разобранных выше методов). В случае, когда система состоит из большого количества объектов, попарное тестирование на пересечение ведет к квадратичной сложности алгоритма, поэтому для систем, состоящих из большого количества тел, применяют двухфазные алгоритмы. Первая фаза (называемая *широкой*) позволяет быстро найти пары потенциально пересекающихся тел, а вторая фаза (*сужения*) состоит в проверке каждой пары на пересечение.

Заметим, что методы определения столкновений используются в разных приложениях САПР и в общем случае отвечают на один из трех вопросов:

- есть ли столкновение в системе;
- каковы точки контакта и нормали в них;
- чему равны пересекающиеся объемы (глубина пересечения).

Первый вопрос полезен в контексте кинематических приложений, где нет нужды обрабатывать геометрическую информацию о контакте, а достаточно проверить сам факт контакта. В приложениях динамики необходимо получать также ту или иную геометрическую информацию, которая позволит вычислить неизвестные силы, действующие на тела в момент контакта.

Алгоритмы широкой фазы

Все известные алгоритмы широкой фазы в той или иной степени основаны на использовании *деревьев ограничивающих объемов* (Bounding Volume Trees, BV-trees), представляющих собой аппроксимацию объемного тела набором непересекающихся канонических объемов (таких как параллелепипеды или сферы).

Простейшим примером BV-дерева является известное нам понятие *октантного дерева*, аппроксимирующего объем с помощью пра-

вильных кубов со сторонами, параллельными осям координат. Построив октантное дерево для каждого тела, мы можем с помощью простого алгоритма искать пересечение. Недостатком данного подхода является тот факт, что при вращательном движении тел октантное дерево приходится перестраивать.

Деревья сфер, напротив, позволяют получить инвариантное относительно вращения представление. Окружив каждое тело сферой, центр которой совпадает с центром масс тела, мы можем проверять пересечения простым вычислением расстояний между центрами и сравнением их с радиусами. Для эффективного представления требуется дробить тело на части и окружать каждую из них своей сферой.

С-деревья представляют собой смесь выпуклых многогранников и сфер, что позволяет более точно окружать объем тела. Однако строить такие деревья автоматически достаточно трудоемко.

AABB-деревья окружают каждое тело параллелепипедом со сторонами, параллельными координатным осям (AABB – Axis Aligned Bounded Boxes). Для поиска пересечения параллелепипедов используются их проекции на координатные оси (представляющие собой интервалы). Для каждой оси эти интервалы сортируются, что позволяет обнаружить пересечения. При обнаружении пересечения между двумя параллелепипедами по всем трем координатным осям алгоритм может разделить параллелепипед на четыре части и продолжить сравнение. Обычно используют два разных типа AABB-деревьев: параллелепипеды фиксированных и динамических размеров. Первые окружают тело вокруг его центра масс с учетом всех возможных вращений. Такие параллелепипеды могут быть построены заранее, а затем использованы на каждом шаге алгоритма динамической симуляции движения. Динамический размер подразумевает построение наименьшего параллелепипеда для данной ориентации тела, что требует дополнительных вычислений на каждом шаге симуляции.

OBB-деревья отличаются от AABB тем, что параллелепипеды допускают произвольную ориентацию (OBB – Oriented Bounded Boxes), тем самым каждое тело окружается наименьшим параллелепипедом, который затем вращается вместе с телом. Проверка на пересечение для OBB-деревьев выполняется сложнее, чем для AABB.

K-DOPs (от DOP – Discrete Oriented Polytopes) определяют семейства выпуклых многогранников, грани которых имеют только конечное число (K) ориентаций (нормалей). Например, AABB-деревья являются 6-DOP. В данном подходе подразумевается статическое построение многогранников до начала динамической симуляции

движения, а на каждом шаге проверки столкновений вычисляется пересечение заранее построенных K -DOPs вокруг текущих центров масс тел.

Заметаемые сферой объемы (SSV – Swept Sphere Volumes) определяют семейство таких канонических объемов, как сфера, конечный цилиндр со скругленными окончаниями, параллелепипед со скругленными гранями. Сложность их обработки сопоставима со сложностью обработки ОВВ-деревьев, но данное представление во многих случаях позволяет провести более аккуратную проверку на столкновение.

Еще одним известным методом аппроксимации тел во время широкой фазы является окружение их *эллипсоидами минимального объема*. Данный способ также является улучшением представления в виде ОВВ-деревьев.

Алгоритмы фазы сужения

Напомним, что на фазе сужения вычисляется точная проверка на пересечения двух заданных тел (точнее, фрагментов тел, так как после широкой фазы доступна информация о локализации потенциальных областей пересечения). Самым известным алгоритмом фазы сужения является линейное программирование. Представив фрагменты тел выпуклыми многогранниками (с заданной погрешностью), задачу проверки пересечения тел можно свести к задаче нахождения решения системы линейных неравенств. Например, мы можем минимизировать искусственную целевую функцию относительно системы неравенств, применяя симплекс-метод.

Метод срединной оси основан на использовании каркаса тела. Каркас тела является геометрическим местом центров максимальных вписанных в тело сфер. При этом подходе каждое тело представляется набором пересекающихся вписанных сфер максимального объема, нанизанных на каркас с заданным шагом. Проверка пересечения двух тел в этом случае сводится к попарной проверке сфер на пересечение.

Алгоритм Лина–Канни (его авторы – Ming C. Lin и John F. Canny) основан на свойстве *когерентности* – использовании информации, вычисленной на предыдущем шаге симуляции движения. Алгоритм выполняет локальный поиск точек пересечения, начиная с точки, найденной на предыдущем шаге. Представив каждое тело выпуклым многогранником, можно осуществить данный поиск, переходя от

вершины к вершине по ребрам и граням тела. Для корректности алгоритма Лина–Канни необходимо, чтобы многогранники были выпуклыми и непересекающимися. Алгоритм *V-Clip* является расширением алгоритма Лина–Канни на случай невыпуклых и пересекающихся многогранников.

Коммерческое программное обеспечение для симуляции движения

Обычно динамическая симуляция механизмов осуществляется с помощью автономных пакетов программ, относящихся к классу САЕ – систем инженерного анализа. Два самых известных пакета динамической симуляции – это ADAMS (производства MSC.Software) и DADS (производства LMS; ныне DADS – часть пакета Virtual.Lab Motion). Основная функциональность таких систем позволяет:

- импортировать геометрические модели из CAD-системы;
- собрать механизм из деталей, связав их кинематическими параметрами;
- задать действующие силы (контакта, пружинных соединений, трения);
- вычислить массы деталей и их тензоры инерции;
- определить столкновения в процессе симуляции работы механизма;
- визуализировать процесс решения прямой и обратной задач кинематики и динамики.

Вопросы для самоконтроля

1. Какими уравнениями описывается движение системы твердых тел в поле действия сил?
2. Как моделируются контакты тел при описании динамической системы с помощью уравнений Ньютона–Эйлера?
3. Что такое уравнения Лагранжа и в чем их преимущества перед уравнениями Ньютона–Эйлера при решении задач динамики системы твердых тел?
4. Опишите общую схему методов определения столкновений. Для чего они используются в САПР?
5. Опишите алгоритмы широкой фазы при определении столкновений.

6. Опишите алгоритмы фазы сужения при определении столкновений.
7. Какова основная функциональность пакетов программ для динамической симуляции механизмов? Приведите примеры таких пакетов.

Дополнительная литература

Освежить теоретические знания о динамике движения твердого тела поможет классический учебник [6]. Обзор известных подходов к численному моделированию динамики системы твердых тел и механизмов дан в [27]. Методы определения столкновений в системе твердых тел исчерпывающим образом охарактеризованы в [24]. Работа [4], описывающая оригинальный подход к моделированию динамики системы твердых тел, является едва ли не единственной, доступной на русском языке.

Лекция 12

Инженерный анализ методом конечных элементов

Конечно-элементный анализ	152
Введение в метод конечных	
элементов	152
Анализ упругости тела	152
Тензор деформаций	153
Тензор напряжений	154
Обобщенный закон Гука, матрицы	
жесткости и упругости	155
Уравнение равновесия тела	
под нагрузкой	157
Применение МКЭ для расчета	
малых напряжений тела	
под нагрузкой	157
Другие приложения МКЭ	159
Типы конечных элементов	159
Разбиения для МКЭ	160
Общая схема	
конечно-элементного анализа	
в САЕ-системах	161
Коммерческие пакеты	
конечно-элементного анализа	162
Вопросы для самоконтроля	162
Дополнительная литература	164

Конечно-элементный анализ

Конечно-элементный анализ (в английской терминологии FEA – Finite Elements Analysis) широко применяется при решении задач механики деформируемого твердого тела, теплообмена, гидро- и газодинамики, электро- и магнитостатики и других областей физики. Потребность в решении данных задач возникает в системах инженерного анализа (САЕ) для моделирования поведения изделия в цифровом виде (не изготавливая само изделие или его макет). Типичными примерами процессов, моделирование которых на компьютере позволяет значительно сократить расходы на испытания, являются продувка в аэродинамической трубе и аварийные испытания (к्रэш-тесты).

Конечно-элементный анализ основан на использовании математического метода конечных элементов (МКЭ).

Введение в метод конечных элементов

Метод конечных элементов (МКЭ, в английской терминологии FEM – Finite Elements Method) используется для приближенного решения дифференциальных и интегральных уравнений. Метод основан на аппроксимации непрерывной функции дискретной моделью, которая строится на множестве кусочно-непрерывных функций, определенных на конечном числе подобластей, называемых *конечными элементами*. Исследуемая геометрическая область разбивается на элементы таким образом, чтобы на каждом из них неизвестная функция аппроксимировалась пробной функцией (как правило, полиномом). Причем эти пробные функции должны удовлетворять граничным условиям непрерывности, совпадающим с граничными условиями, налагаемыми самой задачей. Выбор для каждого элемента аппроксимирующей функции будет определять соответствующий тип элемента. Мы рассмотрим применение МКЭ для анализа упругости твердого тела сложной формы, а затем опишем вкратце другие области применения.

Анализ упругости тела

Теория упругости – раздел физики сплошных сред, изучающий равновесие твердых тел, их поведение при статических и динамических нагрузках. Главная задача теории упругости – выяснить, каковы будут деформации тела и как они будут меняться со временем при заданных внешних воздействиях. Основной системой уравнений для решения

этой задачи является обобщенный закон Гука, связывающий тензор деформаций и тензор напряжений тела. Важным свойством этого уравнение является то, что при достаточно больших напряжениях устойчивых решений не существует. Физически это означает, что при достаточно больших нагрузках система разрушается. Определение этой границы устойчивости и поведение тел в околокритическом состоянии – также одна из главных задач теории упругости.

Рассмотрим трехмерный объект, который находится в равновесном состоянии под воздействием некоторой нагрузки, задаваемой:

- объемными (массовыми) силами \mathbf{f}^V , действующими на все точки объекта $\mathbf{x} \in V$;
- поверхностными силами (трения) \mathbf{f}^S , действующими на части поверхности объекта S ;
- сосредоточенными внешними силами $\mathbf{f}_i, i = 1, \dots, n$, действующими на конкретные точки приложения \mathbf{x}_i .

Смещение точек деформируемого тела под нагрузкой (по сравнению с состоянием без нагрузки) описывается неизвестной функцией $\mathbf{u}: \mathbf{R}^3 \rightarrow \mathbf{R}^3$, определенной для всех точек тела:

$$\mathbf{u}(\mathbf{x}) = \mathbf{u}(x, y, z) = \begin{pmatrix} u(x, y, z) \\ v(x, y, z) \\ w(x, y, z) \end{pmatrix} \in \mathbf{R}^3.$$

Смещения точек тела, описываемые функцией \mathbf{u} , ведут к появлению в теле деформаций и напряжений.

Тензор деформаций

Деформацией называется изменение относительных размеров и формы тела под нагрузкой. Рассмотрим две близкие точки тела \mathbf{x} и $\mathbf{x} + \Delta\mathbf{x}$. В состоянии под нагрузкой они смещаются в точки $\mathbf{x} + \mathbf{u}(\mathbf{x})$ и $\mathbf{x} + \Delta\mathbf{x} + \mathbf{u}(\mathbf{x} + \Delta\mathbf{x})$ соответственно. Обозначив разницу между точками под нагрузкой $\Delta\mathbf{x}'$, получаем $\Delta\mathbf{x}' = \Delta\mathbf{x} + \mathbf{u}(\mathbf{x} + \Delta\mathbf{x}) - \mathbf{u}(\mathbf{x})$. В пределе (при $\Delta\mathbf{x} \rightarrow 0$) направление $\Delta\mathbf{x}'$ (то есть $d\mathbf{x}'$) задается формулой $d\mathbf{x}' = d\mathbf{x} + \nabla\mathbf{u}(\mathbf{x})d\mathbf{x} = (I + \nabla\mathbf{u})d\mathbf{x}$, где

$$\nabla\mathbf{u}(x, y, z) = \begin{pmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} & \frac{\partial v}{\partial z} \\ \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} & \frac{\partial w}{\partial z} \end{pmatrix}.$$

Понятно, что в случае $\nabla \mathbf{u} \equiv 0$ все направления и расстояния между точками в теле сохраняются, что означает поступательное движение без вращений и деформаций. Однако при $\nabla \mathbf{u} \neq 0$ необходимо уметь отличать вращения от деформаций, что ведет к необходимости рассматривать сохранение (вращение) или изменение (деформацию) величины скалярного произведения векторов, задающих направления между бесконечно близкими точками тела. Нетрудно видеть, что $(d\mathbf{x}'_1, d\mathbf{x}'_2) = ((I + \nabla \mathbf{u})d\mathbf{x}_1, (I + \nabla \mathbf{u})d\mathbf{x}_2) = (d\mathbf{x}_1, (I + \nabla \mathbf{u})^T(I + \nabla \mathbf{u})d\mathbf{x}_2) = (d\mathbf{x}_1, d\mathbf{x}_2) + (d\mathbf{x}_1, (\nabla \mathbf{u}^T + \nabla \mathbf{u} + \nabla \mathbf{u}^T \nabla \mathbf{u})d\mathbf{x}_2)$. Симметрическая матрица

$$E^* = \frac{1}{2}(\nabla \mathbf{u}^T + \nabla \mathbf{u} + \nabla \mathbf{u}^T \nabla \mathbf{u}),$$

описывающая изменение скалярного произведения между векторами, задающими направления близких точек тела, называется *лагранжевым тензором деформаций*. Заметим, что этот тензор, как и вектор смещений, вообще говоря, различен в каждой точке тела. В случае $E^* \equiv 0$ при $\nabla \mathbf{u} \neq 0$ имеем дело с бесконечно малым вращением без деформаций. При малых деформациях квадратичные члены матрицы деформаций обычно опускают, получая классический *тензор деформаций*:

$$\boldsymbol{\varepsilon} = \frac{1}{2}(\nabla \mathbf{u}^T + \nabla \mathbf{u}) = \begin{pmatrix} \frac{\partial u}{\partial x} & \frac{1}{2}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) & \frac{1}{2}\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right) \\ \frac{1}{2}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) & \frac{\partial v}{\partial y} & \frac{1}{2}\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) \\ \frac{1}{2}\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right) & \frac{1}{2}\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) & \frac{\partial w}{\partial z} \end{pmatrix}.$$

Физический смысл коэффициентов тензора $\boldsymbol{\varepsilon}$ следующий. Коэффициенты ε_{ii} задают линейную деформацию вдоль трех координатных осей (то есть предел отношения изменения расстояния между точками к исходному расстоянию), а коэффициенты $2\varepsilon_{ji}$ ($i \neq j$) задают угловую деформацию между координатными осями.

Тензор напряжений

Тензор деформаций описывает деформацию тела с кинематической точки зрения, то есть безотносительно причин, породивших ее. Для рассмотрения этих причин (действующих на тело сил) необходимо определить понятие *напряжения* как силы, действующей на единицу площади сечения детали. Рассмотрим плоский срез деформируемого

тела, проходящий через точку \mathbf{P} с нормалью \mathbf{n} . Пусть $\Delta\mathbf{f}$ – сила, действующая на маленький участок плоскости ΔA , содержащий точку \mathbf{P} . Тогда предел

$$\mathbf{t}_n = \lim_{\Delta A \rightarrow 0} \frac{\Delta\mathbf{f}}{\Delta A}$$

существует и называется *напряжением* в точке \mathbf{P} вдоль вектора \mathbf{n} . Для определения напряжения в произвольном направлении используется *тензор напряжений* $\boldsymbol{\sigma}$, который задает напряжение в произвольном направлении \mathbf{n} как $\mathbf{t}_n = \boldsymbol{\sigma}\mathbf{n}$. Для большинства материалов тензор $\boldsymbol{\sigma}$ задается симметрической матрицей. Физический смысл тензора напряжений иллюстрируется на примере срезов, параллельных координатным плоскостям (рис. 31).

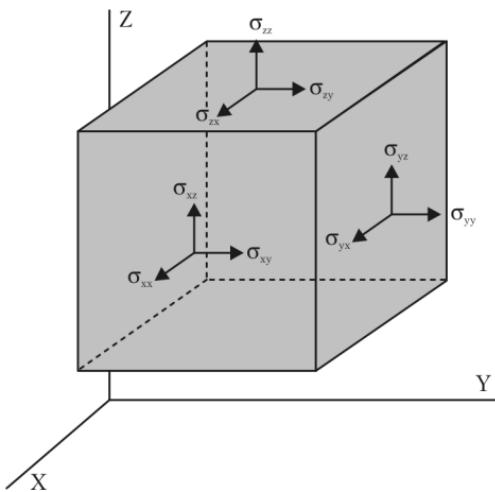


Рис. 31

Обобщенный закон Гука, матрицы жесткости и упругости

В предыдущей лекции мы рассматривали динамику движения недеформируемого твердого тела. Как известно, она описывается уравнениями Ньютона–Эйлера (основанными на втором законе Ньютона), которые связывают линейное и угловое ускорение тела с действующими на него силами посредством массы и тензора инерции. Аналогичную роль в задачах теории упругости играет *обобщенный закон*

Гука. Открытый в XVII веке английским математиком Гуком (Hooke) закон растяжения для тонкого стержня имеет вид $F = k\Delta x$, где F – действующая на стержень сила, Δx – величина растяжения, а k – коэффициент упругости. Величину коэффициента упругости можно связать с физическими размерами стержня следующим образом: $k = ES/L$, где S – площадь поперечного сечения стержня, L – его длина, а E – модуль Юнга. С введением понятий относительного удлинения $\epsilon = \Delta x/L$ и нормального напряжения в поперечном сечении $\sigma = F/S$ закон Гука принимает вид $\sigma = E\epsilon$. Как мы уже знаем, в общем случае напряжение σ и деформация ϵ определяются симметрическими тензорами размера 3×3 . Тем не менее, между этими тензорными сущностями действует то же линейное соотношение, что и между скалярными, называемое обобщенным законом Гука:

$$\sigma_{ij} = \sum_{kl} C_{ijkl} \epsilon_{kl}.$$

Тензор четвертого порядка \mathbf{C} в данной формуле задается 81 коэффициентом (3^4), но так как он связывает симметрические 3×3 -матрицы, каждая из которых определяется шестью скалярными величинами, то может быть представлен в виде 6×6 -матрицы жесткости \mathbf{D} , которая связывает вектор деформаций $\boldsymbol{\epsilon}' = (\epsilon_{11}, \epsilon_{22}, \epsilon_{33}, 2\epsilon_{12}, 2\epsilon_{23}, 2\epsilon_{31})$ с вектором напряжений $\boldsymbol{\sigma}' = (\sigma_{11}, \sigma_{22}, \sigma_{33}, \sigma_{12}, \sigma_{23}, \sigma_{31})$:

$$\boldsymbol{\sigma}' = \mathbf{D}\boldsymbol{\epsilon}'.$$

Это линейное соотношение справедливо для малых деформаций. Обратная матрица к матрице жесткости (\mathbf{D}^{-1}) называется *матрицей упругости*. Заметим, что матрица жесткости (упругости) полностью определяется свойствами материала и не зависит от конкретных нагрузок на тело. Упругие свойства материала могут быть описаны с помощью двух параметров, измеряемых в заданных направлениях: модуля Юнга E , который определяет отношение напряжения к внутренней деформации

$$E = \frac{\sigma_{11}}{\epsilon_{11}},$$

и коэффициента Пуассона ν , который характеризует отношение относительного поперечного сужения к относительному продольному удлинению

$$\nu = -\frac{\epsilon_{11}}{\epsilon_{22}}.$$

Для линейно-эластичных изотропных материалов (таких как металлы, стекло, полипропилен и полиэтилен, резина – в случае малых

деформаций) модуль Юнга и коэффициент Пуассона являются константами, не зависящими от направления и точки измерения, а матрица жесткости имеет следующий вид:

$$\mathbf{D} = \frac{E}{(1+v)(1-2v)} \begin{pmatrix} 1-v & v & v & 0 & 0 & 0 \\ v & 1-v & v & 0 & 0 & 0 \\ v & v & 1-v & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2}-v & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2}-v & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2}-v \end{pmatrix}.$$

Конкретные значения модуля Юнга и коэффициента Пуассона для разных материалов можно найти в инженерных справочниках.

Уравнение равновесия тела под нагрузкой

Внутренняя энергия dU бесконечно малого объема dv рассматриваемого тела под нагрузкой определяется по его напряжениям и деформациям с помощью следующей формулы:

$$dU = \frac{1}{2}(\sigma', \epsilon')dv = \frac{1}{2}(\mathbf{D}\epsilon', \epsilon')dv.$$

Полная потенциальная энергия тела $P = U - W$ равна разности внутренней потенциальной энергии тела и виртуальной работы внешних сил:

$$P = \frac{1}{2} \iiint_V (\mathbf{D}\epsilon', \epsilon')dv - \iiint_V (\mathbf{f}^V, \mathbf{u})dv - \iint_{dV} (\mathbf{f}^S, \mathbf{u})ds - \sum_i (\mathbf{f}_i, \mathbf{u}(\mathbf{x}_i)).$$

Применение МКЭ для расчета малых напряжений тела под нагрузкой

Представим рассматриваемое тело в виде совокупности конечного числа узловых точек $\mathbf{x}_1, \dots, \mathbf{x}_n$, каждой из которых сопоставим неизвестный вектор \mathbf{u}_i , определяющий значение искомой функции (смещения \mathbf{u})

в точке \mathbf{x}_i : $\mathbf{u}(\mathbf{x}_i) = \mathbf{u}_i$. Для представления значения \mathbf{u} в других точках разобьем тело на *конечные элементы* с вершинами в узловых точках. Внутри каждого конечного элемента значения функции \mathbf{u} будем интерполировать по значениям в узловых точках. Более подробно виды конечных элементов и функции интерполяции разберем ниже, а пока ограничимся простейшими тетраэдральными элементами с линейной интерполяцией. Тетраэдр $T_i = (\mathbf{x}_{i1}, \mathbf{x}_{i2}, \mathbf{x}_{i3}, \mathbf{x}_{i4})$ задается четырьмя узловыми точками и представляет собой трехмерный симплекс:

$$T_i = \left\{ \sum_{k=1}^4 t_{ik} \mathbf{x}_{ik} \mid \sum_{k=1}^4 t_{ik} = 1, t_{ik} \geq 0 \right\}.$$

Коэффициенты t_{ik} называются *барицентрическими координатами* точки

$$\mathbf{x} = t_{i1} \mathbf{x}_{i1} + t_{i2} \mathbf{x}_{i2} + t_{i3} \mathbf{x}_{i3} + t_{i4} \mathbf{x}_{i4}$$

в тетраэдре T_i . Линейная интерполяция предполагает следующее выражение для искомой функции \mathbf{u} в точке \mathbf{x} , принадлежащей тетраэдру T_i :

$$\mathbf{u}(\mathbf{x}) \approx \sum_{k=1}^4 t_{ik} \mathbf{u}_{ik}.$$

Понятно, что такая интерполяция искомой функции \mathbf{u} тем точнее, чем больше узловых точек выбрано и чем ближе друг к другу они расположены.

В общем случае можно положить, что конечные элементы задаются произвольным числом узловых точек, а функции интерполяции – произвольные полиномы вида $\sum N_i(\mathbf{x}) \mathbf{u}_i$. При вычислении виртуальной работы W^V массовой силы \mathbf{f}^V интеграл по объему можно представить суммой интегралов по каждому конечному элементу:

$$\iiint_V (\mathbf{f}^V, \mathbf{u}) dv \approx \sum_{i=1}^m \iiint_{V_i} (\mathbf{f}^V, \mathbf{u}) dv \approx \sum_{i=1}^m \iiint_{V_i} (\mathbf{f}^V, \sum_{j=1}^{k_i} N_j(\mathbf{x}) \mathbf{u}_j) dv.$$

Аналогично обстоит дело с поверхностным интегралом, который надо вычислять по граням конечных элементов, приходящихся на выделенную часть границы тела. Работа точечных сил вообще не требует интегрирования и вычисляется напрямую. В результате в качестве выражения для полной виртуальной работы W всех внешних сил получим линейное выражение от неизвестных векторов $\mathbf{u}_1, \dots, \mathbf{u}_n$. Используя вектор $\tilde{\mathbf{u}} = (\mathbf{u}_1, \dots, \mathbf{u}_n)$, виртуальную работу можно задать равенством $W = (\mathbf{f}, \tilde{\mathbf{u}}) + \text{const}$.

Интегрируя приведенное выше выражение для внутренней энергии в виде суммы интегралов по конечным элементам, получаем

квадратичную форму $(\mathbf{K}\tilde{\mathbf{u}}, \tilde{\mathbf{u}}) + \text{const}$. Согласно вариационному принципу Лагранжа вариация разности потенциальной энергии и виртуальной работы внешних сил должна быть нулевой, то есть $(\mathbf{K}\tilde{\mathbf{u}}, \tilde{\mathbf{u}}) = -(\mathbf{f}, \tilde{\mathbf{u}}) + \text{const}$ для всех допустимых перемещений $\tilde{\mathbf{u}}$. Это значит, что вектор $\tilde{\mathbf{u}}$ должен удовлетворять системе линейных уравнений $\mathbf{K}\tilde{\mathbf{u}} = \mathbf{f}$. Таким образом, для расчета малых смещений тела под нагрузкой методом конечных элементов достаточно решить систему линейных уравнений. Решение этой системы задает смещение узловых точек сетки конечных элементов. Смещение внутренних точек сетки вычисляется затем с помощью интерполяционных полиномов. По смещениям можно вычислить тензоры деформаций и напряжений в каждой точке тела.

Другие приложения МКЭ

Так как вариационный принцип Лагранжа справедлив не только для механических систем, то метод конечных элементов находит широкое применение при моделировании самых разных физических процессов. Физическая интерпретация векторов $\tilde{\mathbf{u}}$ и \mathbf{f} дана в табл. 5. Матрицу \mathbf{K} во всех приложениях МКЭ традиционно называют матрицей жесткости.

Таблица 5

Область приложения	Вектор состояния ($\tilde{\mathbf{u}}$)	Сопряженный вектор (\mathbf{f})
Механика твердых тел	Перемещение	Сила
Теплообмен	Теплопроводность	Тепловой поток
Течение	Скорость	Поток
Электростатика	Электрический потенциал	Плотность заряда
Магнитостатика	Магнитный потенциал	Интенсивность магнитного поля

Типы конечных элементов

На практике при использовании МКЭ применяют несколько простых типов конечных элементов. Тип конечного элемента определяется размерностью пространства задачи, базовой геометрией и степенью интерполяции. Зачастую с помощью МКЭ лучше решать двух- и даже одномерные задачи, представляющие собой идеализацию трехмерных тел в случае, если один или два линейных размера незна-

чительны по сравнению с другими. Базовая геометрия элемента, как правило, задается либо *симплексом* (отрезок, треугольник, тетраэдр), либо четырехугольником/шестигранником. Интерполяция в свою очередь бывает линейной, квадратичной или кубической. Все разнообразие типов конечных элементов показано на рис. 32.

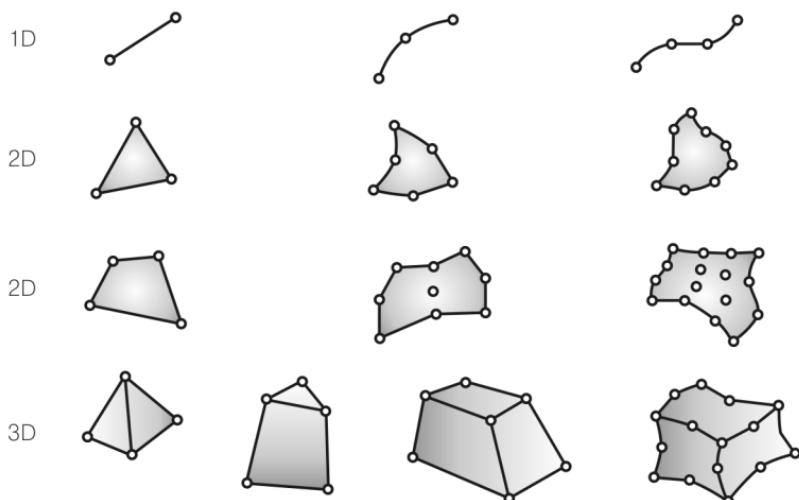


Рис. 32

Разбиения для МКЭ

Для автоматического построения разбиения тела на конечные элементы чаще всего применяются следующие методы:

- соединения узлов;
- топологического разбиения;
- геометрического разбиения;
- решеточные;
- отображаемых элементов.

Метод соединения узлов состоит в равномерном случайном размещении узловых точек, которые затем соединяются ребрами, образуя конечные элементы. Такое соединение можно осуществлять с помощью диаграмм Вороного/триангуляции Делоне, которые дают разбиение двумерного облака точек на треугольники, а трехмерного – на тетраэдры, максимизируя минимальный угол между ребрами.

Топологическое разбиение состоит в рекурсивном разделении тела на треугольники (тетраэдры), а *геометрическое* – в разбиении тела на выпуклые многоугольники (многогранники), которые затем рекурсивно делятся пополам.

Решеточные методы состоят в наложении на тело правильной решетки (тетраэдальной, призматической или гексаэдальной), выделении элементов, которые полностью содержатся внутри тела, и построении дополнительного разбиения вблизи границ (используя, например, октантные деревья).

Наконец, *метод отображаемых элементов* состоит в построении отображения правильной прямоугольной решетки (заданной в пространстве параметров) на криволинейную поверхность (в 3D) и последующее построение трехмерных элементов из двумерных.

Общая схема конечно-элементного анализа в САЕ-системах

Большинство систем автоматического конструирования и инженерного анализа (САЕ) позволяют проводить анализ различных характеристик изделия с помощью метода конечных элементов. На входе таких систем задается геометрическая модель изделия, подготовленная в CAD-системе. В случае интегрированных CAD/САЕ-систем нет никаких сложностей с передачей модели из CAD-части в САЕ, однако для автономных САЕ-систем требуется перевод модели в формат, воспринимаемый данной САЕ-системой. К счастью, большинство коммерческих САЕ-систем понимают форматы большинства коммерческих CAD-систем, в крайнем случае можно воспользоваться нейтральным форматом (IGES или STEP).

Перед импортом геометрической модели в САЕ ее полезно упростить, убрав из нее все незначительные геометрические элементы (маленькие отверстия, карманы, пазы), которые не играют роли для расчетов требуемых физических параметров изделия. Чем проще геометрия, тем проще будет САЕ-системе построить для нее сетку конечных элементов, тем меньше элементов будет содержать эта сетка и тем быстрее будет выполнен расчет физических параметров.

Импортировав упрощенную модель, пользователь САЕ-системы задает материал детали, указывает нагрузки (силы и точки их приложения) и ограничения сборки и крепежа. После чего выполняется автоматическое построение сетки конечных элементов. Построенная с первого раза сетка может не удовлетворить инженера, поэтому

у него всегда есть возможность выполнить новое построение всей детали или ее части, указав системе конкретные параметры метода построения (характеризующие величину и равномерность ячеек).

На следующем этапе данные (сетка и информация о нагрузках и ограничениях) передаются в решатель, который выполняет расчеты с помощью МКЭ. Полученные численные результаты необходимо визуализировать, для чего используются специальные постпроцессоры. Например, для задачи расчета деформаций тела под нагрузкой постпроцессор должен вычислить величину деформации в каждой точке тела и сопоставить ей определенный цвет по заданной шкале (зеленый – деформации незначительны, желтый – деформации велики, красный – критические деформации). После чего трехмерная деталь визуализируется в цвете, предоставляя инженеру возможность увидеть все места критических деформаций и при необходимости внести изменения в конструкцию изделия.

Коммерческие пакеты конечно-элементного анализа

Несмотря на то, что большинство интегрированных CAD/CAM/CAE-систем имеют встроенные модули для расчета деформаций и некоторых других характеристик изделия методом конечных элементов, по-настоящему полная функциональность для конечно-элементного анализа предлагается только в рамках автономных CAE-систем. Самыми известными из них являются:

- ANSYS (производства одноименной компании);
- NASTRAN (производитель – MSC.Software);
- ABAQUS (до недавнего времени независимая компания, ныне принадлежит Dassault Systèmes);
- LS-DYNA (Livermore Software Technology Corporation).

На рис. 33 ниже приведен пример расчета деформаций с помощью МКЭ в системе CATIA V5.

Вопросы для самоконтроля

1. Что такое конечно-элементный анализ? На каком математическом аппарате он основан? Каковы области его применения?

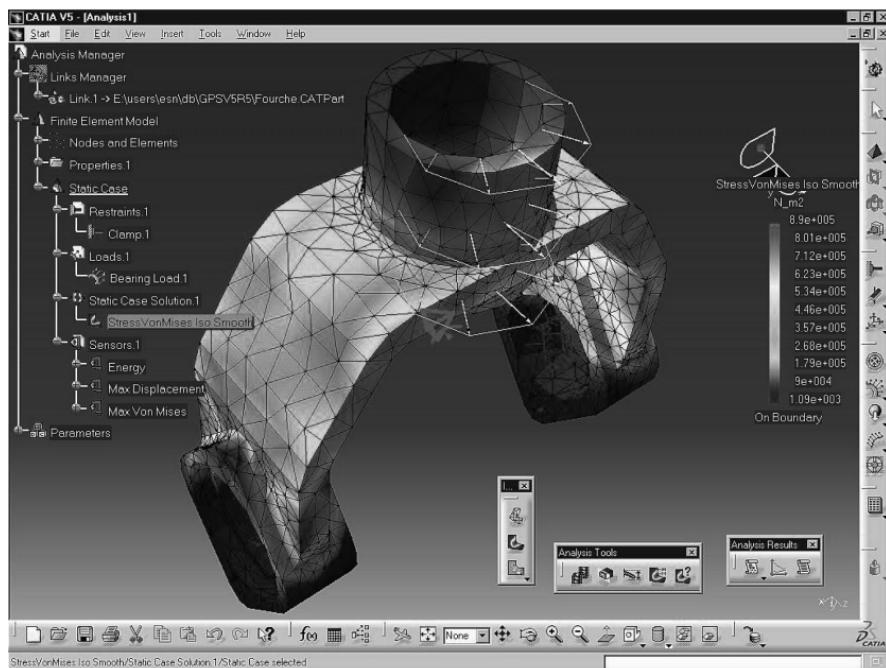


Рис. 33

2. Опишите постановку задачи расчета деформации тела под нагрузкой.
3. Что такое тензоры деформаций и напряжений? Охарактеризуйте их физически и математически.
4. Опишите обобщенный закон Гука. Какие свойства материала определяются модулем Юнга и коэффициентом Пуассона?
5. Опишите и объясните уравнение деформации тела под нагрузкой.
6. Как применяется метод конечных элементов для решения задачи расчета деформаций тела под нагрузкой?
7. Что такое обобщенная матрица жесткости? Опишите применение метода конечных элементов для различных классов физических задач.
8. Опишите основные типы конечных элементов.
9. Какие существуют способы построения сеток для метода конечных элементов?

10. Опишите общую схему конечно-элементного анализа в САЕ-системах и приведите примеры пакетов программ.

Дополнительная литература

Краткое введение в конечно-элементный анализ можно найти в восьмой главе книги [8], а также в третьей части [5]. Более подробное изложение содержится в монографии [6] и в лекциях [9].

Лекция 13

Автоматизация производства

Архитектура станков с ЧПУ	166
Принципы программирования	
для станков с ЧПУ	167
Языки программирования	
высокого уровня для станков	
с ЧПУ	168
Генерация программ для станков	
с ЧПУ по CAD-моделям	170
Быстрое прототипирование	
и изготовление	171
Виртуальная инженерия	173
Вопросы для самоконтроля	173
Дополнительная литература	174

Архитектура станков с ЧПУ

Традиционно станки с числовым программным управлением (ЧПУ) применяются для механической обработки деталей из металлов и сплавов. Обычные металлообрабатывающие станки функционально состоят из механизмов для закрепления заготовки обрабатываемых деталей и режущих инструментов (резцов, фрез, сверл), вращательного привода (шпинделя) с управлением его скоростью, устройства для динамического позиционирования резца относительно детали во время обработки и системы охлаждения. Самыми популярными видами станков являются токарный (позволяет обрабатывать поверхности вращения), сверлильный (позволяет создавать полости и сквозные отверстия в деталях) и фрезерный (который в зависимости от числа степеней свободы может обрабатывать достаточно сложные поверхности). Число степеней свободы металлообрабатывающего станка определяет сложность поверхностей, с которыми может работать станок. Одна степень свободы подразумевает, что во время обработки детали резец может двигаться только вдоль одной прямой, две степени свободы позволяют резцу динамически повторять любую плоскую кривую, три – пространственную кривую, а четыре и большее количество – еще и управлять нормалью оси резца по отношению к обрабатываемой поверхности.

Для работы с непрограммируемыми станками требуется квалифицированный специалист, прошедший соответствующее обучение (токарь, фрезеровщик). Сложные детали может обрабатывать только специалист высокого разряда, обладающий большим опытом работы. Поэтому еще в конце 1940-х гг. начались исследования в области автоматизированного управления металлорежущими станками. Первый фрезерный станок с ЧПУ был создан в 1952 году в Массачусетском технологическом институте.

Станок с ЧПУ (numerical control, NC) отличается от обычного наличием блока управления станком (machine control unit, MCU), функционально состоящего из модуля обработки данных (data processing unit, DCU) и замкнутой системы управления (control loop unit, CLU). Модуль MCU считывает данные с входного источника данных (сначала это были перфокарты и перфоленты, потом их заменили магнитные носители), а DCU преобразует их в сигналы управления станком. С появлением микросхем и микропроцессоров эта архитектура эволюционировала в привычную современную схему управления любым внешним устройством с помощью интегрирован-

ного в устройство *контроллера* и внешнего компьютера. С введением компьютера в схему управления станком последний стал называться станком с *компьютеризированным числовым управлением* (computer numerical control, CNC). В современных производственных цехах все компьютеры, контролирующие станки с ЧПУ, соединены в сеть под командой центрального компьютера, с которого и происходит непосредственное управление всем цехом, включая загрузку данных на конкретный станок. Подобная схема называется *распределенным числовым управлением* (distributed numerical control, DNC).

Принципы программирования для станков с ЧПУ

Основным способом общения с контроллером станка с ЧПУ является язык *программ обработки* (называемый также *G-кодом*), который состоит из небольшого количества элементарных команд, разбитых на слова. (Можно провести прямую аналогию между языками программ обработки для станков с ЧПУ и языками ассемблера для ЭВМ.) Команды языка обработки группируются в блоки, каждый из которых имеет следующий фиксированный формат:

- N – идентификатор блока;
- G – предварительные команды (позиционирование, остановка, ускорение и пр.; все доступные команды закодированы двузначными числами);
- X, Y, Z, I, J, K – координаты инструмента в системе координат станка (X, Y, Z – координаты конечной точки, I, J, K – координаты центра окружности, задаются только при движении по дуге);
- F – скорость подачи (в дюймах в минуту);
- S – скорость шпинделя (в оборотах в минуту);
- T – номер инструмента (резца, фрезы, сверла – зависит от станка);
- M – вспомогательные команды (включить охлаждение и пр.; все доступные команды закодированы двузначными числами).

Ненужные слова в блоке могут пропускаться. Если какие-то координаты не заданы, подразумевается, что они не изменяются относительно текущего положения. Вот пример типичной команды:

N009 G02 X-28.28 Y0.0 I14.14 J5.0

Семантика команды такова: из текущей точки резец должен описать дугу окружности с центром в точке (14.14, 5.0) в направлении движения часовой стрелки и остановиться в точке (-28.28, 0.0). Если после этого надо с повышенной скоростью (не касаясь обрабатываемой детали) перевести резец в точку (-75.0, 0.0, 40.0), после чего выключить шпиндель и охлаждение, то соответствующая команда будет выглядеть так:

```
N010 G01 X-75.0 Y0.0 Z40.0 F950 M30
```

Система координат целиком определяется типом станка. Существует общее соглашение, что ось z совпадает с осью вращения шпинделя (на котором закреплена деталь или инструмент), а ось x всегда горизонтальна. Причем движение в положительном направлении по обеим осям удаляет инструмент от заготовки.

Языки программирования высокого уровня для станков с ЧПУ

Понятно, что программировать вручную коды обработки – задача не из легких: необходимо правильно рассчитать траекторию резца относительно поверхности детали, определить ключевые точки на этой траектории и т. п. Для облегчения ручного составления программ для ЧПУ были разработаны специальные языки высокого уровня (ADAPT, AUTOSPOT, EXAPT, COMPACT, SPLIT, MAPT). Все они в той или иной степени базируются на языке *APT* (Automatically Programmed Tool – автоматически программируемый инструмент), разработанном в Массачусетском технологическом институте в 1956 г. и с тех пор непрерывно совершенствующемся. Разберем этот язык подробнее.

Компилятор программ языка APT генерирует машинно-независимый бинарный файл ISO-формата *CL* (Cutter Location – положение резца), по которому уже специальный постпроцессор генерирует программу обработки для конкретного станка. Основной принцип языка APT состоит в возможности описать и поименовать стандартную геометрическую форму (точку, отрезок, окружность, плоскость, сферу, цилиндр, конус и пр.), а потом ссылаться на нее при описании операций перемещения резца. Вот пример программы на языке APT, предназначенный для сверления двух отверстий в заготовке:

```

P0 = POINT/0.0, 3.0, 0.1
P1 = POINT/1.0, 1.0, 0.1
P2 = POINT/2.0, 1.0, 0.1
FROM/P0
GOTO/P1
GODLTA/0, 0, -0.7
GODLTA/0, 0, 0.7
GOTO/P2
GODLTA/0, 0, -0.7
GODLTA/0, 0, 0.7
GOTO/P0

```

Важно, что кроме позиционного управления движением инструмента (абсолютного или относительного, как в вышеприведенном примере) АРТ поддерживает режим *контурного регулирования*, при котором управление движением инструмента в пространстве осуществляется указанием трех поверхностей: *поверхности детали*, *поверхности движения* (по-другому называемой направляющей поверхностью) и *контрольной поверхности*. Инструмент обрабатывает своей режущей частью поверхность детали, сам следуя поверхности движения (как правило, эти две поверхности перпендикулярны друг другу), до тех пор, пока не соприкоснется с контрольной поверхностью. Для перемещения инструмента к начальной точке обработки используется команда GO, например:

GO/PAST, L1, TO, PS, TANTO, C1

задает движение в точку, где резец (фреза) касается поверхности детали (L1) со стороны, противоположной начальному положению (для касания с другой стороны используется команда GO/TO, для позиционирования оси инструмента строго на поверхности пересечения – команда GO/ON), касается направляющей поверхности (PS) со стороны начальной точки и касается контрольной поверхности (C1). После начального позиционирования инструмента его движение можно задать командами GOLF, GORGT, GOUP, GODOWN, GOFWD, GOBACK, которые описывают движение в одном из возможных направлений относительно заданных поверхностей.

Дополнительные операторы языка АРТ осуществляют выбор конкретного станка и инструмента, регулируют скорость движения режущего инструмента и вращения шпинделя, включают и выключают подачу охлаждающей жидкости. Кроме того, АРТ подобно другим языкам высокого уровня позволяет организовывать циклы и вызывать подпрограммы.

Генерация программ для станков с ЧПУ по САД-моделям

Если деталь, подвергаемая обработке на станке с ЧПУ, была спроектирована в системе CAD, то ручное написание программы на языке АРТ для ее обработки выглядит в значительной степени избыточным – ведь описываемые геометрические поверхности уже содержатся в геометрической модели детали. Поэтому основной функциональностью САМ-систем (как автономных, так и интегрированных CAD/CAM-пакетов) является автоматическая генерация управляющих программ. Конечно, программы генерируются сразу в машинно-независимом бинарном виде (упомянутых выше CL-файлов), а потом преобразуются в набор команд обработки для конкретного станка. Полностью автоматизировать процесс не удается, поэтому инженер должен произвести следующие действия в САМ-системе:

- выделить в общей геометрической модели детали те элементы, которые важны при машинной обработке;
- определить геометрию режущего инструмента (обычно из широкого набора, встроенного в САМ-систему);
- определить последовательность операций металлообработки и выбрать траекторию для каждой из них (пользователь выбирает только тип построения траектории из нескольких встроенных в систему, а конкретная траектория строится автоматически);
- визуализировать траекторию на экране компьютера в виде анимации, изображающей движение резца;
- проверить (встроенными средствами САМ-системы), совпадает ли (в пределах заданных допусков) форма каждой поверхности заготовки после виртуальной металлообработки с поверхностями детали, заданными в ее геометрической модели;
- сохранить CL (машинно-независимый) или MCD (машинно-зависимый) файл с командами обработки для контроллера станка с ЧПУ.

Отметим важность шага, связанного с моделированием процесса металлообработки в виде анимационного ролика и расчета формы детали. До появления подобной функциональности в САПР программисты станков с ЧПУ вынуждены были использовать деревянные или пластиковые заготовки для того, чтобы проверить корректность составленной ими программы обработки.

Быстрое прототипирование и изготовление

Очень часто до начала производства детали из металла необходимо изготовить ее копию из пластика или композитного материала (называемую *прототипом*) для того, чтобы:

- визуально и тактильно оценить дизайн будущего изделия;
- проверить процессы сборки-разборки механизма из составляющих его деталей;
- протестировать кинематику механизма;
- вычислить реальные аэродинамические характеристики детали;
- использовать прототип для физического изготовления формы с последующим литьем в ней металлической детали.

Важно, что процесс изготовления детали из композитного материала не требует использования металлорежущих станков, а значит, отпадает необходимость в подготовке технологического плана изготовления прототипа и программирования станков с ЧПУ. Все данные для производства прототипа содержатся непосредственно в его твердотельной виртуальной модели. Процесс изготовления физической детали непосредственно по ее геометрической CAD-модели называется *быстрым прототипированием*. В настоящее время существуют следующие процессы быстрого прототипирования:

- стереолитография;
- отверждение на твердом основании;
- избирательное лазерное спекание;
- трехмерная печать;
- ламинарирование;
- моделирование методом наплавления.

Существует несколько промышленных станков для производства деталей вышеперечисленными методами. Разберем подробнее самый популярный процесс быстрого прототипирования – *стереолитографию*. Данный процесс основан на избирательном отверждении фоточувствительного полимера (пластика). Исследования в этом направлении начались еще в конце 1970-х гг., а в 1987 г. фирма 3D Systems продемонстрировала первую промышленную установку для стереолитографии, называемую SLA-1. Входные геометрические данные для всех процессов быстрого прототипирования готовятся в мозаичном представлении, независимом от любой системы твердотельного моделирования (чаще всего в виде STL-файлов, структуру которых

мы разобрали в одной из предыдущих лекций). Процесс протекает следующим образом:

- 1) определяется оптимальная ориентация детали в трехмерном пространстве и рассчитываются ее плоские сечения (называемые *слоями*), перпендикулярные оси *z* с заданным шагом (называемым *толщиной слоя*) в соответствии с выбранной ориентацией;
- 2) моделируются *поддерживающие структуры* в виде дополнительных частей изготавливаемой трехмерной детали (они также представляются в виде STL-файлов) и рассчитываются их поперечные сечения;
- 3) затвердевающий на свету фоточувствительный полимер, поддерживаемый в жидком состоянии, наливается в специальную форму, внутри которой установлена платформа, способная перемещаться в вертикальном направлении; начальное положение платформы соответствует растеканию над ней полимера по толщине одного слоя;
- 4) ультрафиолетовый лазер сканирует слой полимера по форме вычисленного заранее профиля поперечного сечения, обеспечивая его отверждение;
- 5) платформа вместе с деталью опускается на заданную толщину слоя вниз;
- 6) шаги (4) и (5) повторяются, пока не будут обработаны все вычисленные поперечные сечения;
- 7) из контейнера сливается оставшаяся жидкость и извлекается затвердевшая деталь;
- 8) деталь подвергается окончательному отверждению с помощью ультразвукового излучения в специальном аппарате;
- 9) производится окончательная доводка детали (удаляются поддерживающие структуры, проводится ручная шлифовка поверхности).

Важным моментом подготовки процесса стереолитографии является выбор наилучшей ориентации детали и проектирование для нее поддерживающих структур. Последние представляют собой подпорки – новые части детали, удаляемые на шаге (9). Эти структуры обеспечивают, чтобы деталь, у которой площадь следующего слоя значительно превосходит площадь предыдущего, не деформировалась в процессе изготовления. Кроме того, они нужны для поддержки *островков* (islands) – топологически изолированных частей слоя.

Ориентация детали влияет на количество и сложность поддерживающих структур для нее, а также на точность изготовления детали. Последний параметр также управляет варьированием толщиной слоя. Наконец, в процессе стереолитографии важно учитывать уменьшение плотности материала при его затвердевании, а значит, увеличивать объем изготавливаемой детали для компенсации ее усадки.

Виртуальная инженерия

Область САПР, называемая *виртуальной инженерией*, включает программные продукты для виртуального проектирования, цифровой имитации, виртуального прототипирования и построения виртуальных заводов. В то время как виртуальное проектирование по большому счету остается технологией будущего (оно подразумевает совершенно новый подход к объекту проектирования с помощью методов виртуальной реальности), методы цифровой имитации и виртуального прототипирования уже реализованы в ряде систем инженерного анализа, включая интегрированные CAD/CAM/CAE-системы, такие как CATIA V5. Однако еще один продукт Dassault Systèmes, называемый DELMIA, уже сейчас предлагает своим пользователям возможность реалистичной визуализации множества производственных процессов, включая работу целого цеха. Данный продукт был создан после поглощения Deneb Robotics компанией Dassault Systèmes. Унифицированная по своему пользовательскому интерфейсу с системами CATIA и ENOVIA, система DELMIA предоставляет возможности по проектированию движений роботов, по анализу эргономики и человеческого фактора, по имитации работы станков с ЧПУ и сборочных линий.

Вопросы для самоконтроля

1. Что такое станок с ЧПУ? Опишите архитектуру станка с ЧПУ.
2. Что такое степени свободы станка с ЧПУ? Как строится система координат станка?
3. Что такое G-код? Приведите примеры блоков команд.
4. Что такое CL-данные? Для чего нужны постпроцессоры при программировании станков с ЧПУ?
5. Какие существуют языки высокого уровня для программирования станков с ЧПУ?
6. Как осуществляется генерация программ для станков с ЧПУ по CAD-моделям?

7. Охарактеризуйте известные методы быстрого прототипирования и изготовления.
8. Что такое виртуальная инженерия и цифровое производство? Приведите примеры.

Дополнительная литература

Программированию станков с ЧПУ посвящена одиннадцатая глава книги [8]. Двенадцатая глава той же книги содержит описание популярных методов быстрого прототипирования, а тринадцатая может служить введением в виртуальную инженерию. Более подробное изложение принципов числового программного управления можно найти в третьей части книги [2].

Лекция 14

Технологическая подготовка производства

Интеграция CAD и CAM	176
Задачи инженера-технолога	176
Модифицированный подход к технологической подготовке	177
Групповая технология	178
Классификация и кодирование деталей	178
Генеративный подход к технологической подготовке	180
Конструкторско-технологические элементы	181
Методы автоматического распознавания конструктивных элементов	182
Пример автоматического распознавания КТЭ	185
Вопросы для самоконтроля	185
Дополнительная литература	186

Интеграция CAD и CAM

Описанные в предыдущих разделах методы проектирования изделий и программирования для их производства станков с ЧПУ нуждаются в тесной интеграции друг с другом, а также в большей автоматизации. Программировать последовательность операций станка с ЧПУ для каждой новой детали вручную (даже с использованием интерактивного программирования по геометрической модели CAD) достаточно трудоемко. Будучи достаточно интеллектуальным процессом, составление плана изготовления детали, тем не менее, может быть автоматизировано. Такая автоматизация осуществляется с помощью *систем автоматизированного планирования процессов* (Computer-Aided Process Planning, CAPP), которые также называются *системами технологической подготовки производства*. Данные системы и их математические основы служат предметом настоящей лекции. Но сначала мы разберем постановку задачи технологической подготовки производства, которую на промышленных предприятиях решают инженеры-технологи.

Задачи инженера-технолога

С помощью систем CAD инженеры-проектировщики разрабатывают геометрическую и физическую модели детали или механизма. Иная задача стоит перед инженером-технологом – составить *технологический план* производства изделия. Глядя на проект изделия, инженер-технолог должен прежде всего решить, какие технологические процессы следует применить для производства изделия и в какой последовательности. Количество процессов определяется количеством поверхностей у детали. Как правило поверхности вращения обрабатываются на токарном станке, цилиндрические отверстия – на сверлильном, прочие поверхности – на фрезерном, тонкая обработка – на шлифовальном. Кроме того, деталь или ее заготовка может подвергаться термической обработке, сварке, резке, прессовке и т. п. Описывая каждый технологический процесс, инженер-технолог определяет:

- используемый материал;
- форму заготовки будущей детали (с учетом требования минимальных отходов);
- станок, на котором она будет обработана данным процессом (токарный, сверлильный, фрезерный, шлифовальный и т. п.);

- способ крепежа детали в станке;
- используемый инструмент (вид резца, диаметр сверла и т. п.);
- требуемую точность изготовления;
- время, необходимое на обработку детали.

Несколько технологических процессов могут быть выполнены в одной конфигурации (на одном станке без смены способа крепления детали), и это тоже надо учитывать при планировании. В конце этапа технологической подготовки производства создается план, описывающий последовательность технологических процессов или сборочных операций и называемый *операционной, или маршрутной, картой*.

Понятно, что выполнить подобное планирование под силу только весьма квалифицированному специалисту. Тем не менее, последовательность его действий достаточно стандартна, а разнообразие видов изготавливаемых деталей не так велико. Все это служит основанием для осуществления автоматизации работы инженера-технолога. Как мы уже упоминали, программные системы такого рода относят к классу САРР. Исторически первые САРР-системы, появившиеся в 1960-е гг. были основаны на *модифицированном подходе* к технологической подготовке производства. В конце 1970-х появились системы, основанные на *генеративном подходе*. Ниже мы разберем оба этих подхода детально.

Модифицированный подход к технологической подготовке

Модифицированный подход (variant approach) к технологической подготовке производства основан на том, что геометрически похожие изделия имеют похожие операционные планы. Как правило, каждая новая деталь оказывается подобной другим деталям, которые уже выпускались на данном предприятии, следовательно, для них уже были подготовлены планы технологических процессов производства, и задача инженера-технолога состоит прежде всего в поиске этих технологических планов и их модификации для новой детали. Итак, простейшим способом автоматизации труда инженера-технолога является организация на предприятии базы данных технологических планов с удобным поиском в ней. Для удобной организации поиска применяют так называемую *групповую технологию*, разработанную С. П. Митрофановым. Суть этой технологии состоит в кодировании всех основных типов обрабатываемых деталей в зависимости от их формы и размера.

Групповая технология

Основополагающим понятием в групповой технологии и теории классификации и кодирования является понятие *семейства деталей*, представляющее собой совокупность таких объектов, которые подобны друг другу либо по геометрической форме и размерам, либо по последовательности технологических операций, необходимых для их изготовления. Понятие семейства деталей является центральным в информационно-поисковых проектных системах и для большинства автоматизированных систем планирования производства. Еще одно производственное преимущество введения этого понятия состоит в его применении для компоновки оборудования в производственном цехе. Классическая схема компоновки состоит в группировке станков по видам обработки (токарный, фрезерный, сверлильный, шлифовальный и сборочный участки). Однако данная схема имеет очевидный недостаток в случае, если предприятие производит небольшое количество семейств деталей (по сравнению с объемами производства) – в этом случае накладные расходы на передачу заготовок с участка на участок очень высоки. Альтернативная схема компоновки станков в таком случае состоит в размещении их по участкам в соответствии с семействами деталей. В этом случае каждый участок отвечает за изготовление деталей своего семейства.

Можно выделить три основных метода группировки деталей в семейства:

- визуальный контроль;
- анализ технологических маршрутов;
- классификация и кодирование деталей.

Наиболее практическим и распространенным следует признать последний метод, который мы разберем отдельно.

Классификация и кодирование деталей

Данный метод группировки деталей в семейства состоит в идентификации конкретных проектных и/или технологических характеристик каждой детали. Среди проектных характеристик обычно выделяют:

- опорные внешние очертания детали;
- опорные внутренние очертания детали;
- отношение длины к диаметру;
- тип материала;
- функциональное назначение;
- главные размеры;
- вспомогательные размеры;
- допуски;
- чистоту поверхности.

Технологические характеристики включают в себя некоторые из вышеперечисленных (например, размеры), а также:

- основную обработку;
- вспомогательные операции;
- последовательность операций;
- продолжительность цикла изготовления;
- объем партии;
- ежегодный выпуск;
- необходимые приспособления;
- станочное оборудование;
- режущий инструмент.

Кодирование детали сводится к использованию последовательности символов, отождествляемых с теми или иными проектными и/или технологическими характеристиками. На практике в системах кодирования используют три основные кодовые структуры:

- иерархическую;
- цепную;
- гибридную.

В рамках иерархической структуры интерпретация каждого следующего символа зависит от значения предыдущего, в рамках цепной структуры интерпретация каждого символа не зависит от других. На практике удобно пользоваться комбинацией кодовых структур, то есть гибридной структурой.

Три самые известные системы классификации и кодирования деталей – это система Опица (Университет г. Аахена, Германия), MICLASS (компания TNO, Голландия) и CODE (фирма MDSI, США). На рис. 34 приведен пример кодирования детали в системе MICLASS.

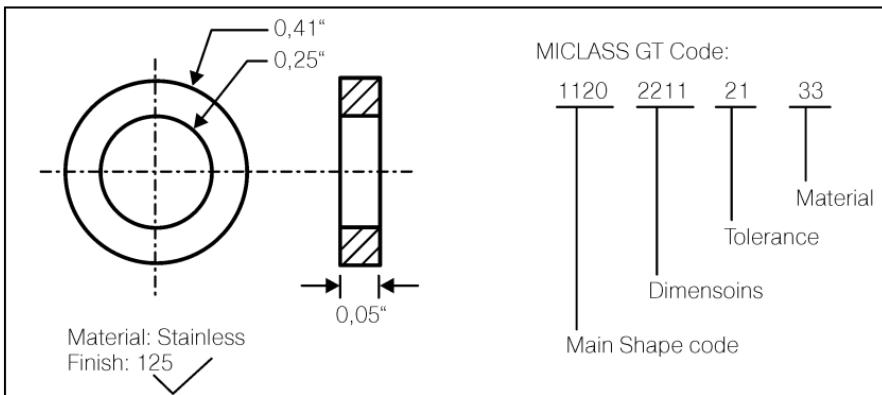


Рис. 34

Генеративный подход к технологической подготовке

Генеративный подход состоит в автоматической генерации технологического плана производства изделия по заданным техническим требованиям. Технические требования при этом извлекаются непосредственно из CAD-данных. Проще всего это сделать, если деталь была спроектирована на основе *конструктивных элементов* (design features). Конечно, далеко не все конструктивные элементы (КЭ), используемые при проектировании, соответствуют элементам, которые можно изготовить машинной обработкой (manufacturing/machining features; чтобы отличить их от конструктивных элементов проектирования, будем называть последние *конструкторско-технологическими элементами*, КТЭ), поэтому в общем случае эти элементы должны быть распознаны и преобразованы. Немаловажным является и тот факт, что сведения о допусках и материалах зачастую отсутствуют в CAD-моделях, поэтому инженер-технолог должен задавать их вручную.

Таким образом, генеративный подход к технологической подготовке состоит в использовании обширной базы данных конструкторско-технологических элементов, а также системы вывода, которая осуществляет распознавание КТЭ по геометрии изделия или их получение из КЭ.

Конструкторско-технологические элементы

Итак, конструкторско-технологический элемент – ключевое понятие систем, основанных на генеративном подходе к технологической подготовке. Разберем это понятие подробнее. Типичными конструкторско-технологическими элементами детали из цельного металла являются отверстия, пазы, карманы. Для изделий из листового металла примерами КТЭ служат профиль, изгиб, сварной шов; при планировании сборок – механические соединения, такие как подшипники. С каждым классом таких конструкторско-технологических элементов связан соответствующий технологический процесс фрезерования, прессовки, сварки или сборки. При описании техпроцесса используется специальная модель, задаваемая:

- типом операции (сверление, фрезерование, шлифование, токарная обработка);
- требуемыми ресурсами (станки, инструменты, крепежи);
- параметрами процесса (скорости вращения и подачи);
- дополнительными атрибутами (такими как время и стоимость).

Эти параметры вычисляются автоматически (с помощью специальных функций, связанных с каждым типом КТЭ) по размерам соответствующих геометрических элементов. Зачастую один и тот же КТЭ может быть реализован с помощью нескольких альтернативных техпроцессов (скажем, круглое отверстие можно сделать как фрезерованием, так и сверлением), поэтому в общем случае с одним типом КТЭ связывается коллекция техпроцессов, из которых пользователь (инженер-технолог) может самостоятельно выбрать наиболее подходящий.

В некоторых CAD/CAM-системах (таких как Pro/ENGINEER, I-DEAS, MicroStation) понятия КЭ и КТЭ совпадают. Конечно, такой подход значительно облегчает задачу САПР-подсистемы, так как план производства детали, по сути, закладывается при ее проектировании. Однако этот подход одновременно ограничивает возможности инженера-проектировщика, вынуждая последнего думать в терминах машинной обработки. Например, изображенную на рис. 35 простую деталь необходимо проектировать с использованием большего количества КТЭ, чем КЭ.

Следует также вспомнить, что многие системы предоставляют проектировщику возможность изменять геометрию детали без ис-

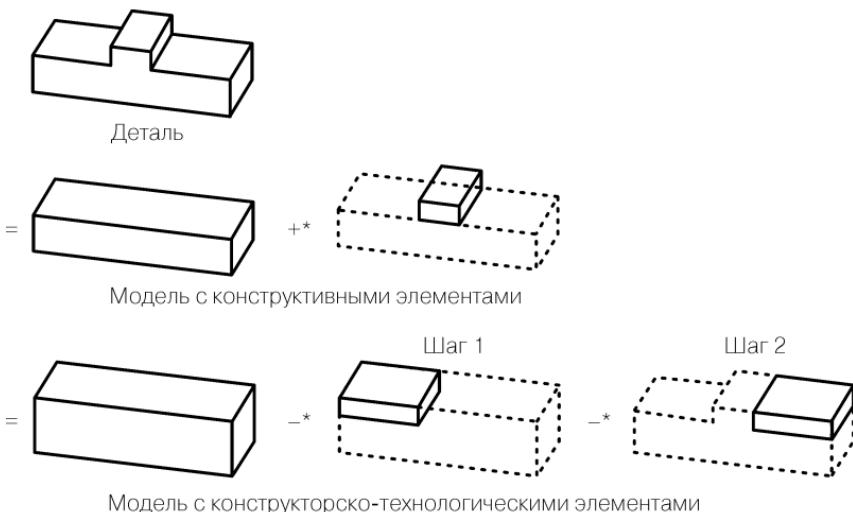


Рис. 35

пользования КЭ (например, с помощью булевых операций). Таким образом, автоматическое распознавание конструктивно-технологических элементов остается актуальной задачей. В контексте такого распознавания наиболее практическая роль принадлежит методам распознавания так называемых 2,5-мерных элементов, которые полностью характеризуются плоским замкнутым контуром и глубиной каждой точки на границе и внутри контура. Такие элементы могут быть легко изготовлены на трехосевых сверлильных и фрезерных станках с ЧПУ.

Методы автоматического распознавания конструктивных элементов

Потребность в распознавании конструктивных элементов (в частности, конструкторско-технологических элементов) существует не только в САПР-системах. Другим важным примером применения таких методов служит собственно проектирование. При обмене данными между CAD-системами, как правило, дело ограничивается передачей топологии и геометрии (то есть информации о границной структуре модели – BRep), поэтому импортированная модель перестает быть

параметрической, что резко ограничивает возможности ее редактирования в новой системе. Для проведения инженерного анализа методом конечных элементов требуется построить равномерную сетку тела, что намного проще сделать для тел правильной формы (см. соответствующую лекцию настоящего курса). Для этого из исходной геометрии полезно удалить все незначительные отверстия, выемки и т. п. Проще всего это сделать после распознавания соответствующих конструктивных элементов в исходной геометрической модели.

В контексте настоящей лекции мы разберем три самых популярных подхода к распознаванию конструкторско-технологических элементов:

- анализ графов;
- объемную декомпозицию;
- с помощью подсказок.

Одним из исторически первых методов распознавания КТЭ на основе графов был метод графовых шаблонов, примененный в известной САПР-системе PART фирмы Technomatics (ныне принадлежит Siemens PLM Software). *Граф смежности граней* (Face Adjacency Graph, FAG) строится на основе топологической информации ВRep. Вершины этого графа представляют собой грани модели, а ребра – отношения смежности между гранями. Заметим, что ребро графа в точности соответствует одному или нескольким ребрам граничной модели, так как две смежные грани всегда имеют как минимум одно общее ребро. Отметим также, что вершины граничной модели в таком графе никак не отражаются. Для эффективного распознавания КЭ собственно топологической информации недостаточно, поэтому граф смежности граней снабжается геометрическими атрибутами. Джоши (Joshi) и Чанг (Chang) предложили с каждым ребром связать единственный атрибут – информацию о выпуклости/вогнутости угла между разделяемыми гранями по отношению в телу. На рис. 36 ниже изображена деталь и соответствующий ей граф смежности граней.

После построения графа смежности граней применяется метод его декомпозиции на несвязные подграфы с использованием простой эвристики: вершина, у которой все смежные ребра выпуклы, не может образовывать 2.5-размерный КТЭ, поэтому ее следует удалить из графа. Например, для графа смежности граней детали на вышеприведенном рисунке будут последовательно удалены вершины $f_1, f_2, f_3, f_4, f_5, f_6$ и f_{10} . После упрощения графа анализируется его форма. В нашем случае граф будет состоять из трех вершин и двух вогнутых ребер, то есть соответствовать КТЭ «паз», причем средняя вершина

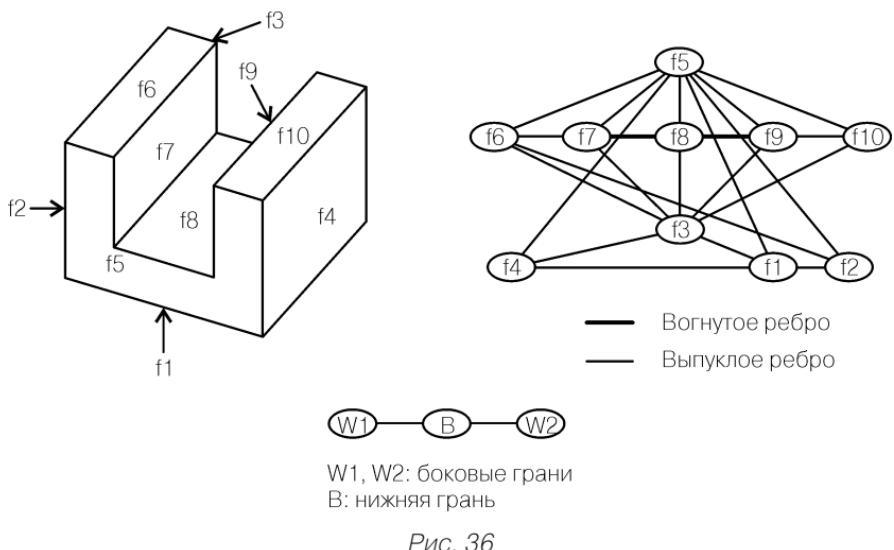


Рис. 36

такого графа будет гранью-дном, а две крайние – гранями-стенами. Подобные шаблоны существуют и для других КТЭ, поэтому задача алгоритма состоит лишь в их распознавании. Понятно, что таким способом можно довольно успешно распознавать изолированные КТЭ, но как быть с теми, которые пересекаются друг с другом, как в случае на рис. 37 (где для простоты приведена только двумерная проекция детали)?

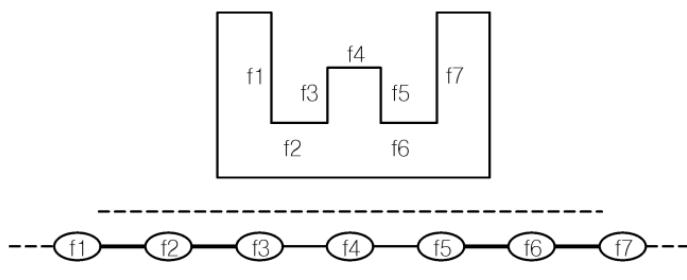


Рис. 37

В данном случае алгоритм обнаружит два паза, но этого будет недостаточно для изготовления детали данной формы. Было предложено несколько алгоритмов для восстановления некоторых ребер в графе смежности после его упрощения, что позволяет идентифици-

ровать все КТЭ, но ни один из этих алгоритмов не является достаточно общим, работая только над некоторыми конфигурациями. Другой проблемой данного подхода является возможность идентификации формально корректных КТЭ, которые, тем не менее, невозможно изготовить на практике. Пример тому система вложенных пазов (рис. 38).

Наконец, еще один недостаток подхода – экспоненциальная сложность алгоритма проверки изоморфизма двух графов. На практике же все шаблоны графов КТЭ имеют ограниченный размер, поэтому можно считать алгоритм полиномиальным.

Пример автоматического распознавания КТЭ

В качестве примера ПО для автоматического распознавания конструкторско-технологических элементов рассмотрим продукт Prismatic Machining Preparation Assistant, входящий в систему CATIA V5 (рис. 39). Данный продукт помогает автоматически подготовить программу для обработки деталей на сверлильных и фрезерных станках с ЧПУ. Для этого пользователю предоставляется набор функций для генерации всех призматических КТЭ. Благодаря встроенной технологии для распознавания конструктивных элементов, данный продукт может генерировать КТЭ (machining features) по любой геометрической модели CATIA, даже той, в которой нет исходных геометрических конструктивных элементов (design features). Данный продукт тесно интегрирован с продуктом Prismatic Machining, который представляет собой САМ-систему для программирования трехосевых сверлильных и фрезерных станков с ЧПУ для работы с 2.5-мерными элементами.

Лицензия на технологию автоматического распознавания КТЭ, используемую в продукте Prismatic Machining Preparation Assistant, была приобретена компанией Dassault Systèmes у индийской компании Geometric Software Solutions – одного из лидеров в этой области, запатентовавшей соответствующую технологию.

Вопросы для самоконтроля

1. Какие задачи решает инженер-технолог?
2. В чем состоит модифицированный подход к технологической подготовке производства?

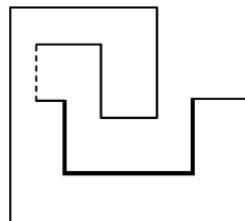


Рис. 38

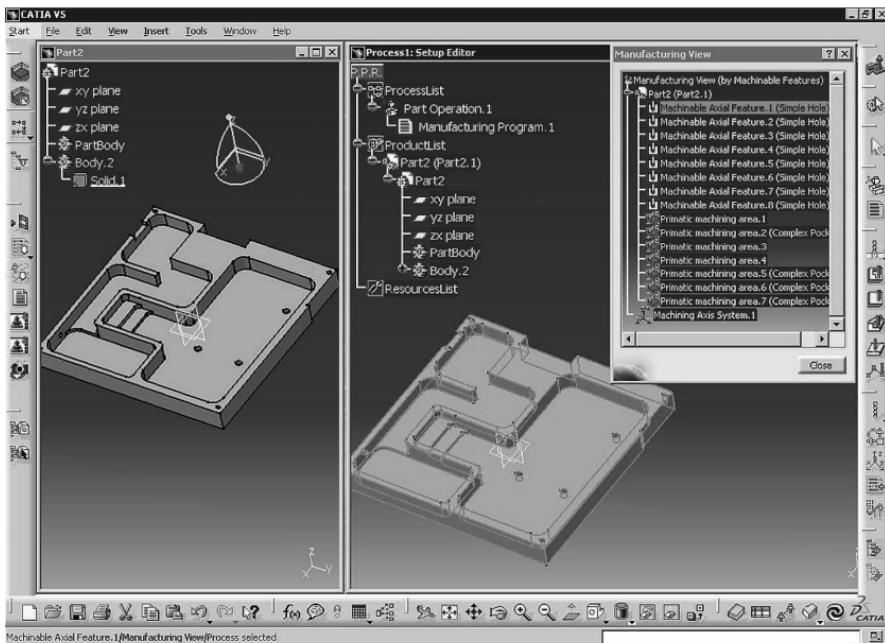


Рис. 39

3. Охарактеризуйте базовые принципы групповой технологии.
4. Опишите известные системы классификации и кодирования деталей.
5. Опишите генеративный подход к технологической подготовке производства.
6. Что такое конструкторско-технологичный элемент? В чем состоят его отличия от конструктивных элементов?
7. Опишите известные методы распознавания конструкторско-технологических элементов.

Дополнительная литература

Технологической подготовке производства посвящена десятая глава книги [8]. Подробнее о групповой технологии можно узнать из пятой части [2] или обратившись к первоисточнику [10]. Обзор известных методов распознавания конструкторско-технологических элементов можно найти в девятой главе монографии [37], а также в работе [28].

Лекция 15

Управление данными на протяжении жизненного цикла изделия

Системы управления данными об изделии	188
Цифровой макет изделия (DMU) и спецификация материалов (BOM)	188
Примеры PDM-систем	189
Программное обеспечение для организации бизнес-процессов	189
Из чего состоит PLM?	191
Интеграция PLM с системами управления отношениями с заказчиками	193
Интеграция PLM с системами управления цепочками поставок	194
Интеграция PLM с системами управления ресурсами предприятия	195
Практические подходы к интеграции систем PLM с CRM, SCM и ERP	197
Преимущества внедрения систем PLM	199
Вопросы для самоконтроля	200
Дополнительная литература	201

Системы управления данными об изделии

На протяжении жизненного цикла изделия (от разработки его концепции – через проектирование, изготовление, организацию продаж и послепродажного обслуживания – до утилизации) существует потребность работать с данными об изделии, к которым прежде всего относятся инженерные данные, такие как CAD-модели, чертежи, технологические карты, программы для станков с ЧПУ, результаты тестов и анализов и многое другое. Кроме собственно данных об изделии имеются еще и *метаданные* – такие как информация о создателе и текущем статусе документа. Разные отделы одного и того же предприятия (маркетинга, проектирования, производства, поддержки, финансов) должны постоянно работать с этими данными, передавая их друг другу. Для того чтобы упростить и автоматизировать передачу таких данных, были разработаны PDM (Product Data Management systems) – *системы управления данными об изделии*.

Система PDM позволяет организовать совместный доступ ко всем данным об изделии, гарантируя их постоянную целостность, обеспечивает внесение необходимых изменений во все версии изделия, позволяет модифицировать и конфигурировать варианты изделия, отслеживать историю изменений. Большинство PDM-систем позволяют одновременно работать с инженерными данными, полученными от разных CAD-систем, что делает возможным их эффективное использование в рамках *расширенного предприятия* (понятие, которое включает в себя как собственно производящую компанию, так и совокупность ее поставщиков и заказчиков). Практически все PDM-системы имеют веб-интерфейс, что позволяет использовать их для обмена данными в сетях Интернет и интранет. Однако самым важным преимуществом системы PDM является ее использование на протяжении всего жизненного цикла изделия в рамках концепции управления этим циклом.

Цифровой макет изделия (DMU) и спецификация материалов (BOM)

Ключевым видом данных, которые хранятся в системах PDM, является *цифровой макет изделия* (Digital Mock-Up, DMU), представляющий собой виртуальную технологию определения модели реально-

го продукта. Такая модель обычно состоит из коллекции трехмерных геометрических моделей (взятых из базы данных), размещенных в пространстве в соответствии с представлением о форме продукта, с каждой из которых связана *спецификация материалов* (Bill Of Materials, BOM). Последнее понятие включает в себя данные о составе изделия и нормах расхода сырья, материалов и компонентов на единицу измерения. Обычно эти данные организованы в иерархическом виде – в соответствии со структурой изделия. Спецификация материалов используется для представления цифрового макета изделия, а также для планирования потребности в материалах в рамках соответствующей функциональности систем управления ресурсами предприятия (ERP). Визуализация трехмерного цифрового макета позволяет инженерам анализировать большие сложные изделия на предмет удобства их сборки из компонентов и последующего технического обслуживания.

Примеры PDM-систем

PDM-системы предлагают практически все крупные производители интегрированных CAD/CAE/CAPP/CAM-решений: Teamcenter от Siemens PLM Software, Windchill от PTC, ENOVIA SmarTeam, MatrixOne и VPLM от Dassault Systèmes. Последняя система тесно интегрирована с CAD/CAM/CAE-системой CATIA V5 и позволяет работать с цифровыми макетами таких продуктов, как автомобили, самолеты и суда. Специфика данной предметной области такова, что цифровой макет таких сложных изделий состоит из десятков и сотен тысяч деталей. Это накладывает серьезные требования на производительность системы. Система ENOVIA SmarTeam не рассчитана на работу со столь внушительными сборками, зато может быть легко интегрирована в самые разные системы классов CAD и ERP.

Программное обеспечение для организации бизнес-процессов

Говоря о жизненном цикле изделия, выделяют следующие этапы:

- исследования (маркетинговые, НИОКР и пр.);
- разработки;
- подготовки производства;
- производства и активных продаж;
- снятия с производства.

Рассмотрим организацию управления компании, производящей некую продукцию и реализующую ее потребителям с целью получения прибыли. В деятельности такого предприятия можно выделить четыре взаимосвязанных контура управления бизнес-процессами (рис. 40):

- управления взаимодействием с заказчиком (потребителем) продукции, который формирует требования к продукции; реализуется в системах класса CRM (Customer Relationship Management, *управление взаимоотношениями с заказчиками*);
- управления ресурсами компании, который планирует и контролирует расход финансовых, материальных и трудовых ресурсов в процессе производства продукции; реализуется в системах класса ERP (enterprise resource planning, *планирование ресурсов предприятия*);
- взаимодействия с поставщиками, который планирует и контролирует поставки комплектующих и выполнение работ внешними контрагентами; реализуется в системах класса SCM (Supply Chain Management, *управление цепочками поставок*);
- управления продукцией, который контролирует, хранит и предоставляет всю конструкторскую, технологическую и эксплуата-

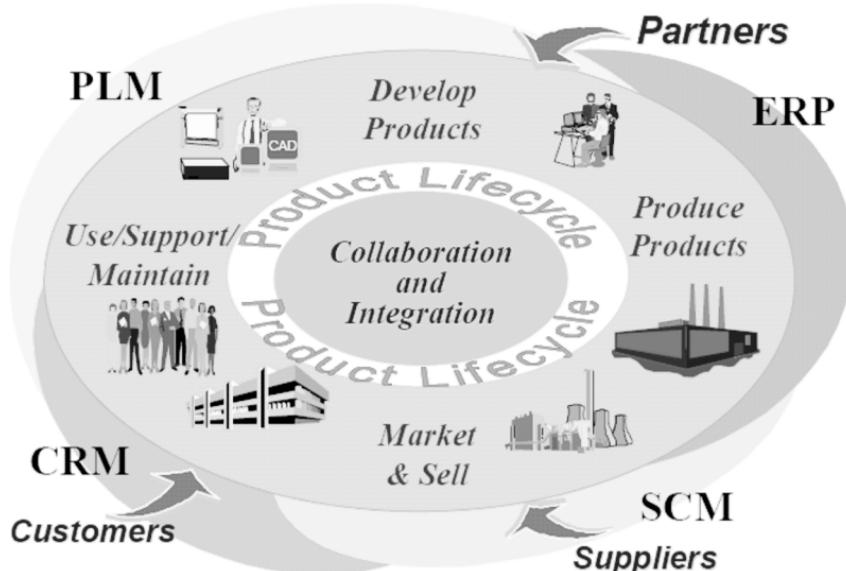


Рис. 40

тационную информацию об изделии; реализуется в системах класса PLM (Product Lifecycle Management, управление жизненным циклом изделия).

На рис. 41 приведено место каждого класса ПО на этапах жизненного цикла изделия. Отметим главенствующую роль систем PLM.

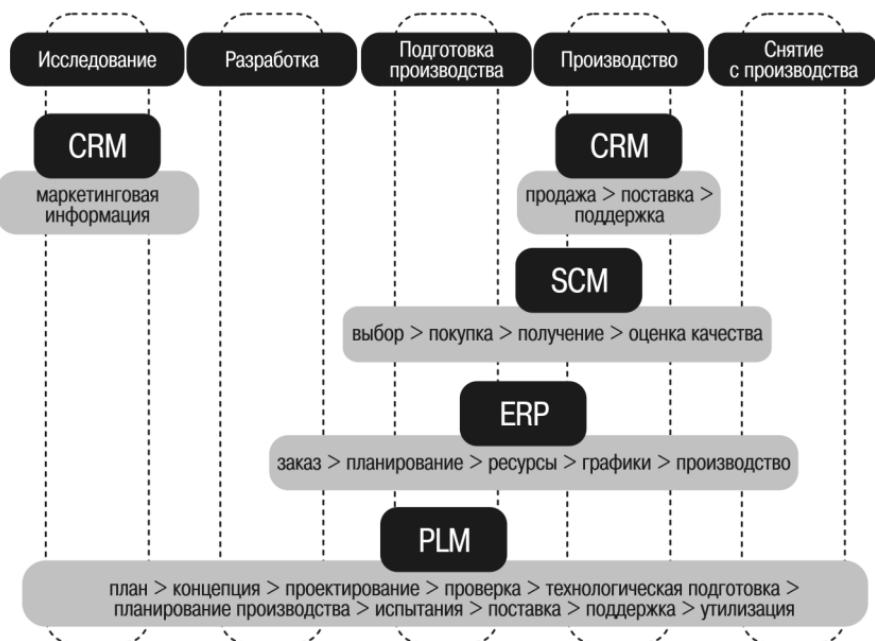


Рис. 41

Из чего состоит PLM?

Аналитическая компания CIMdata определяет PLM как стратегический бизнес-подход, состоящий в применении на предприятии совместимых решений для поддержки совместного создания, управления, распространения и использования определяющей изделие информации и охватывающий все этапы жизненного цикла, интегрирующий сотрудников, процессы, системы и информацию. Выделяют три фундаментальных концепции PLM:

- возможность универсального, безопасного и управляемого доступа к определяющей изделие информацию и ее использование;

- поддержание целостности информации, определяющей изделие, на протяжении всего его жизненного цикла;
- управление и поддержка бизнес-процессов, применяемых при создании, распределении и использовании подобной информации.

Согласно этому определению ПО для PLM включает в себя набор средств для проектирования (куда попадают системы классов CAD, CAE, CAPP и CAM) и организации совместной работы, визуализации, управления и разделения информации об изделии (PDM).

Более детально, PLM включает в себя

- управление:
 - данными об изделии (их содержимое и контекст);
 - цепочкой проектирования;
 - документами и их ассоциированным содержимым (все типы, форматы и носители);
 - требованиями (функциональными, производительности, качества, стоимости, физических факторов, взаимозаменяемости, времени и т. п.);
 - портфелями изделий и проектов, семействами изделий;
 - имуществом, то есть станками и оснасткой, оборудованием сборочных линий;
 - сервисной информацией, включая поддержку послепродажного обслуживания;
- руководство программами и проектами;
- визуализацию и совместную работу;
- управление поставками компонентов;
- цифровое производство;
- определение изделия;
- анализ, тестирование и симуляцию изделия;
- технические публикации, такие как
 - сервисные мануалы;
 - руководства пользователя;
 - инструкции по сборке.

Важно понимать, что для внедрения PLM на предприятии недостаточно набора вышеупомянутых средств. Во-первых, все эти средства должны быть полностью интегрированы для возможности совместной работы в рамках единой информационной системы предприятия. Во-вторых, внедрение PLM подразумевает полную перестройку бизнес-модели, приведение ее в соответствие с этапами жизненного цикла.

Интеграция PLM с системами управления отношениями с заказчиками

Важно понимать, что системы PLM не заменяют собой другие важные подсистемы управления предприятием, а напротив интегрированы в них. В области отношений с заказчиками автоматизация осуществляется с помощью CRM-систем. CRM – это корпоративная информационная система, предназначенная для улучшения обслуживания клиентов, сохраняя информацию об истории взаимоотношений с клиентами, установления и улучшения бизнес-процедур (на основе сохраненной информации) с последующей оценкой их эффективности. Основные принципы системы таковы:

- наличие единого хранилища информации, откуда в любой момент можно получить все сведения обо всех случаях взаимодействия с клиентами;
- синхронизация управления множественными каналами взаимодействия (то есть существуют организационные процедуры, которые регламентируют использование этой системы и информации в каждом подразделении компании);
- постоянный анализ собранной информации о клиентах и принятии соответствующих организационных решений – например, о ранжировании клиентов по их значимости для компании.

Таким образом, CRM-подход подразумевает, что при любом взаимодействии с клиентом по любому каналу сотруднику организации доступна полная информация обо всех взаимоотношениях с клиентами и решение принимается на ее основе; информация о решении, в свою очередь, тоже сохраняется и доступна при всех последующих взаимодействиях.

Выделяют следующие стратегические бизнес-процессы, которые можно эффективно осуществлять путем разделения интеграции информации об изделии и о потребностях заказчика:

- управление требованиями (Requirements Management);
- конфигурирование изделия (Product Configuration);
- обслуживание и улучшение изделия (Product Service and Improvement).

При управлении требованиями из системы CRM в систему PLM важно перенести информацию об ожиданиях заказчика и потребно-

стях рынка относительно будущего изделия. Данная информация может быть использована при проектировании изделия, а также при последующей его верификации на соответствие ожиданиям заказчика. При конфигурировании изделия, интеграция CRM и PLM позволяет ответить на вопросы заказчика относительно внешнего вида изделия, управления изделием, ограничения на условия его использования. Наконец, при послепродажном обслуживании изделия PLM-системы позволяют быстрее найти источник неполадки, отследить точный жизненный цикл изделия, предсказать влияние обнаруженной неполадки на эксплуатацию других экземпляров изделия, внести необходимые изменения во всю партию и в следующие версии изделия.

Интеграция PLM с системами управления цепочками поставок

Системы управления цепочками поставок (SCM) предназначены для автоматизации процесса планирования, осуществления операций и контроля над ними в цепи поставок (в логистической сети), основная цель которого – удовлетворить требования заказчика максимально эффективно. Данная деятельность состоит в управлении всеми перемещениями и складированиями сырья, полуфабрикатов и готовых изделий от пункта отправления до пункта потребления товара. Следующие проблемы успешно решаются в рамках SCM:

- конфигурация распределенной сети – количество и местоположение поставщиков, производственных мощностей, оптовых баз, складов и заказчиков;
- стратегия распространения товара – централизованная или децентрализованная, прямые поставки или стыковки, маркетинговая стратегия вытягивания или вталкивания товаров на рынок (pull or push strategy), логистические услуги третьей стороны;
- информация – интеграция систем и процессов во всей цепочке поставок для выделения ценной информации, такой как сигналы о запросах, прогнозы, инвентаризация и транспортировка;
- управление инвентаризацией – количество и местоположение инвентаря, включая сырье, полуфабрикаты и готовые изделия.

Понятно, что построение и оптимизация логистической цепи невозможно без информации о структуре продукта. В обрабатывающей

промышленности планирование цепочки поставок не зависит от структуры продукта («соль есть соль»), но, например, в авиастроении болт, которым двигатель крепится к крылу, имеет определенную геометрию, допуски, сопротивление материала и многие другие свойства, которые должны браться в расчет в случае замены в цепи поставок одного поставщика болтов на другого.

Интеграция PLM с системами управления ресурсами предприятия

ERP-системы – это информационные управляющие системы, которые объединяют множество бизнес-процессов, связанных с операционными или производственными аспектами предприятия:

- производство;
- логистику;
- дистрибуцию;
- складирование;
- погрузку;
- выставление счетов;
- бухучет.

Термин ERP появился в результате развития концепции *планирования производственных ресурсов* – MRP II, которая в свою очередь представляет собой эволюцию понятия *планирования потребности в материалах* (MRP). MRP-системы автоматизируют производственное планирование и инвентаризацию, необходимые для эффективного управления процессами производства изделия. MRP-системы помогают достичь следующих целей одновременно:

- проверить доступность материалов и изделий для производства или доставки заказчикам;
- управлять наименьшим возможным уровнем инвентаризации;
- планировать производственные процессы, поставки и закупки.

Ключевым понятием MRP-систем является *основной производственный план* (Master Production Schedule, MPS). На входе таких систем задаются:

- основной производственный план (Master Production Schedule, MPS), представляющий собой комбинацию всех известных и ожидаемых потребностей в определенном продукте на обозримый период планирования (несколько месяцев или лет в будущее);

- данные о запасах (информация о доступности сырья и полуфабрикатов);
- спецификация материалов (BOM);
- данные о планировании (маршрутные, трудовые и машинные стандарты).

На выходе получается рекомендованный производственный план (с детальной информацией о времени начала и окончания каждой операции в терминах компонентов изделия) и рекомендованный план закупок сырья и полуфабрикатов. Возникшая в конце 1960-х гг. технология MRP затем была расширена до более общей технологии планирования производственных ресурсов (MRP II). Последняя позволяет осуществлять операционное планирование (в единицах продукции), финансовое планирование (в долларах) и моделировать различные ситуации, отвечая на вопросы «что если». MRP II состоит из набора взаимосвязанных функций, основными из которых являются:

- бизнес-планирование;
- планирование производства и продаж;
- планирование выпуска продукции;
- составление основного производственного плана (MSP);
- планирование потребности в материалах (MRP);
- планирование потребности в производственных мощностях (CRP);
- системы поддержки управления производственными мощностями и материалами.

Интегрированные финансовые отчеты, получаемые с помощью систем класса MRP II, содержат:

- бизнес-план;
- отчет обязательств по заказам;
- экспедиторский бюджет;
- цену запасов в долларах.

Интеграция систем PLM и MRP II необходима во многих областях. Например, загрузка рабочих мест при планировании потребности в производственных мощностях (CRP) рассчитывается на основе технологического маршрута изготовления изделия – набора шагов (операций), которые необходимо совершить для изготовления изделия или его части. Технологический маршрут в свою очередь разрабатывается в рамках СAPP-систем (или, вообще говоря, MPM-систем, см. материал предыдущей лекции). Поэтому необходимо уметь передавать данные из MPM-системы в систему CRP.

Практические подходы к интеграции систем PLM с CRM, SCM и ERP

Выделяют три основных подхода к осуществлению интеграции PLM и ERP (что применимо также к CRM и SCM) – рис. 42:

- инкапсуляцию;
- интерфейс;
- интеграцию.

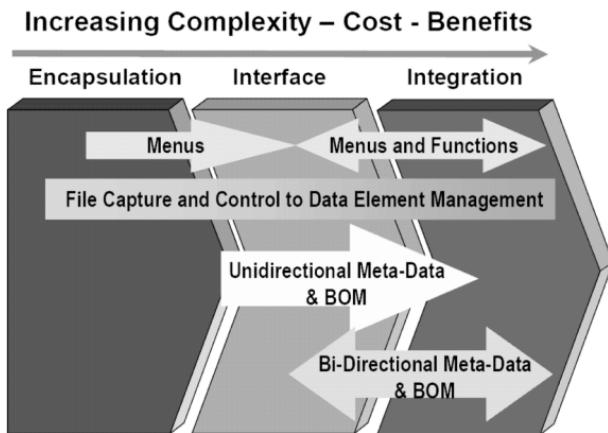


Рис. 42

Каждый подход обеспечивает различные уровни и сложность интеграции, функциональность, масштаб, а также предполагает различные цены на реализацию и поддержку. Инкапсуляция означает создание оболочки вокруг CAD-файлов и передачу их в ERP-систему, где они могут храниться. Дальнейший доступ к таким файлам (включая поиск и сортировку) может осуществляться из ERP-системы, но для использования этих данных необходимо загружать специальное приложение. Реализация такого подхода к интеграции требует нескольких человеко-дней.

При интерфейсном подходе PLM и ERP могут обмениваться файлами и метаданными автоматически. Доступ к PLM-функциям при этом обеспечивается через меню приложений ERP. Обычно информация о цифровом макете изделия и спецификации материалов передается из PLM в ERP. Интеграция такого рода требует знания внутренней структуры PLM- и ERP-приложений и может быть осуществлена за несколько человеко-недель.

Разработка полной интеграции дает возможность полного автоматического обмена всеми типами данных и метаданных между обеими сферами. Большинство PLM-функций при этом становятся доступными в рамках ERP, и определенные ERP-функции будут доступны в PLM. В этом сценарии пользователи работают в согласованном окружении, получая доступ к нужной информации и процессам непосредственно из их основного приложения, независимо от того, чем управляется информация – PLM или ERP. Разработчики интеграции должны иметь глубокие знания о структурах PLM и ERP. Интеграция предъявляет также требования к пользовательскому интерфейсу: он должен работать в обеих сферах. Критическим фактором для осуществления успешной интеграции является сотрудничество поставщиков PLM и ERP-решений. Стоимость осуществления такой интеграции составляет от нескольких человеко-месяцев до нескольких человеко-лет.

Независимо от того, какой метод и какая отправная точка выбраны, существует множество подходов для реализации этих методов, включающих:

- информационные порталы;
- точечную интеграцию;
- сервер интеграции приложений, EAI (от Enterprise Application Integration);
- поставляемые готовые интеграции;
- индивидуальные решения.

Информационный портал обеспечивает только частичную интеграцию, но за меньшую цену и за меньшее время. Web-браузер используется для представления информации из нескольких корпоративных систем и хранилищ данных, например, складской информации из ERP или чертежей из PLM. Такой подход не обеспечивает (или обеспечивает в малой степени) целостность данных или интеграцию процессов. Однако потенциально это недорогая отправная точка для интеграции.

Точечный подход основан на создании индивидуальных интерфейсов между отдельными приложениями с использованием программных интерфейсов приложений (API). Такие интеграции могут быть получены от поставщиков приложений или разработаны самостоятельно. Точечные интеграции могут обеспечить очень полное, уникально скроенное решение, хотя и за более высокую цену, по сравнению с другими подходами.

Для уменьшения общей стоимости и сложности реализации и сопровождения интеграции PLM и ERP, многие поставщики и компании используют EAI. EAI-интеграции могут быть получены от производителей или разработаны самостоятельно. Инструменты EAI разработаны для интеграции приложений, процессов и данных на уровне предприятия, и включают в интеграцию бизнес-правила. Что еще важно, EAI существенно сокращает требования к реализации и поддержке частных интеграционных решений. Такой подход является «неразрушающим» и не требует модификации интегрируемых систем.

Поставляемые готовые интеграции могут служить прочным фундаментом для построения окончательного решения. В некоторых случаях требуются только небольшие дополнительные усилия (в основном административного характера); в других случаях необходима некоторая минимальная настройка, однако встречаются случаи, когда нужна серьезная переделка. Вот некоторые аспекты, которые необходимо учесть при оценке поставляемого решения по интеграции PLM и ERP: функциональность, полнота и документированность API, гибкость интеграции, степень покрытия команд и функций приложений, доступ к структурам данных приложений, наличие консультационной поддержки.

Индивидуальные решения могут быть разработаны с комбинированным использованием API, технологии EAI и ручного программирования. Индивидуальные решения спроектированы под специфические нужды компаний, но они настолько хороши, насколько хорошо собрана информация о потребностях предприятия, лежащая в основе такого решения. Более того, индивидуальные решения могут требовать большего времени на разработку и, как правило, используют более дорогие методы интеграции PLM и ERP (как в начальной стадии, так и на этапе сопровождения).

Преимущества внедрения систем PLM

Рисунок 43 демонстрирует типичный график возврата инвестиций, привязанный к жизненному циклу изделия, с применением систем PLM и без них (положительное направление вертикальной оси соответствует возврату инвестиций, то есть получению прибыли).

Главное преимущество использования систем PLM состоит в сокращении времени вывода нового изделия на рынок (time-to-market) и в продлении периода активных продаж, что обусловлено:

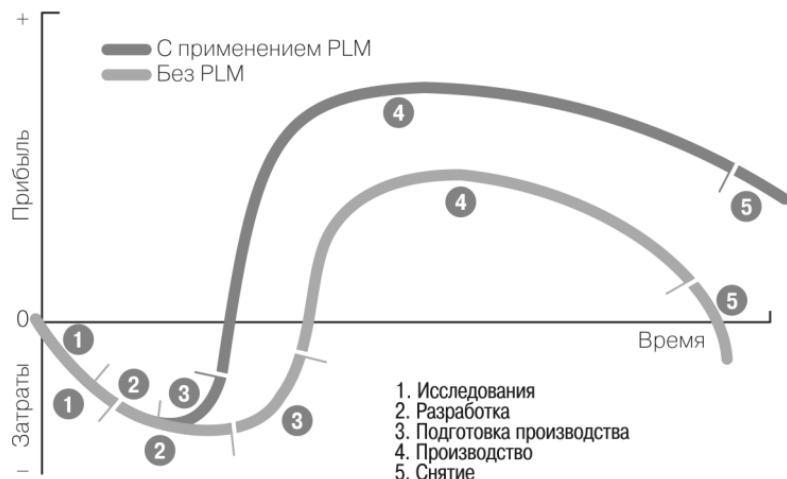


Рис. 43

- повышением эффективности взаимодействия (благодаря возможности обмена данными) при проведении исследований и совместной разработке изделия;
- сокращением времени разработки благодаря повышению количества повторно используемых деталей и заимствованных решений, сокращению издержек на устранение изначально заложенных в изделие ошибочных решений;
- сокращением времени на подготовку производства благодаря наличию на этом этапе полной и достоверной спецификации на выпускаемое изделие; наличие цифрового макета изделия позволяет смоделировать производственные процессы и заранее предотвратить возможные проблемы.

Вопросы для самоконтроля

1. Что такое станок цифровой макет изделия и спецификация материалов? Каковы типичные свойства систем управления данными об изделии?
2. Опишите жизненный цикл изделия. Какие задачи приходится решать на каждом из этапов?
3. Что такое управление жизненным циклом изделия? Опишите три фундаментальных концепции PLM. Охарактеризуйте основные компоненты соответствующего программного обеспечения.

4. Что такое системы управления отношениями с заказчиками? Каковы потоки информации между системами CRM и PLM?
5. Что такое системы управления цепочками поставок? Каковы потоки информации между системами SCM и PLM?
6. Опишите основные блоки систем планирования ресурсов предприятия? Каковы потоки информации между системами ERP и PLM?
7. Опишите практические подходы к интеграции систем PLM с системами CRM, SCM и ERP.
8. Охарактеризуйте преимущества внедрения PLM на предприятии.

Дополнительная литература

Подробнее о концепции управления жизненным циклом изделия можно прочитать в работах [14] и [36]. Различные аспекты интеграции информационных систем предприятия освещаются в [23], [26] и [35].

Краткий англо-русский словарь аббревиатур в области автоматизации проектирования и производства

AEC CAD, Architecture Engineering and Construction CAD – САПР архитектуры и строительства.

BOM, Bill of Material – спецификация материалов.

BRep, Boundary Representation – граничное представление.

CAD, Computer-Aided Design – автоматизированное проектирование (САПР).

CADD, Computer-Aided Design and Drafting – автоматизированное проектирование и черчение.

CAGD, Computer-Aided Geometric Design – автоматизированное геометрическое проектирование.

CAE, Computer-Aided Engineering – автоматизированный инженерный анализ, автоматизированное конструирование.

CAM, Computer-Aided Manufacturing – автоматизированное производство, автоматизированная подготовка производства.

CAPP, Computer-Aided Process Planning – автоматизированная технологическая подготовка производства.

CIM, Computer-Integrated Manufacturing – комплексно-автоматизированное производство

CL, Cutter Location – положение инструмента.

CNC, Computerized Numerical Control – компьютеризированное числовое программное управление.

CPD, Collaborative Product Development – коллективная разработка изделия.

CRM, Customer Relationship Management – управление взаимодействием с клиентом.

CRP, Capacity Requirements Planning – планирование потребности в производственных мощностях.

CSG, Constructive Solid Geometry – конструктивное представление объемной геометрии.

CSP, Constraint Satisfaction Problem – задача удовлетворения ограничениям.

DMU, Digital Mock-Up – цифровой макет.

DNC, Distributed Numerical Control – распределенное числовое программное управление.

DOF, Degree Of Freedom – степень свободы.

EAI, Enterprise Application Integration – интеграция приложения предприятия.

ECAD, Electronic CAD – САПР электроники.

EDA, Electronic Design Automation – автоматизированное проектирование электронных приборов и устройств.

ERP, Enterprise Resource Planning – планирование (управление) ресурсами предприятия.

FBS, Feature-Based System – система, основанная на конструктивных элементах.

FEA, Finite Element Analysis – конечно-элементный анализ.

FEM, Finite Element Method/Modeling – моделирование методом конечных элементов.

IGES, Initial Graphic Exchange Standard – исходный стандарт обмена графическими данными.

MCAD, Mechanical CAD – автоматизация механического проектирования.

MDA, Mechanical Design Automation – автоматизация механического проектирования.

MES, Manufacturing Execution System – автоматизированная система управления производственными процессами.

MPM, Manufacturing Process Management – управление производственными процессами, цифровое производство.

MPS, Master Production Schedule – основной производственный план.

MRP, Material Requirement Planning – планирование потребности в материалах.

MRP II, Manufacturing Resource Planning – планирование производственных ресурсов.

NC, Numerical Control – числовое программное управление (ЧПУ).

NURBS, Non-Uniform Rational B-Spline – неоднородный рациональный B-сплайн.

PDM, Product Data Management – управление данными об изделии.

PLM, Product Lifecycle Management – управление жизненным циклом изделия.

SCM, Supply Chain Management – управление цепочками поставок.

SGC, Selective Geometric Complexes – селективные геометрические комплексы.

STEP, STandard for Exchange of Product model data – стандарт обмена модельными данными об изделии.

STL, STereoLithography – стереолитография.

Список литературы

1. Гриньон П., Ушаков Д. Учет инженерных ограничений на различных этапах PLM // Препринт 9; ЗАО ЛЕДАС. – Новосибирск, 2004. – 40 с.
2. Грувер М., Зиммерс Э. САПР и автоматизация производства / пер. с англ. – М.: Мир, 1987. – 528 с.
3. Еремченко А., Ершов А. Два новых метода декомпозиции задач с геометрическими ограничениями // Препринт 11; ЗАО ЛЕДАС. – Новосибирск, 2004. – 36 с.
4. Ермолин Е., Казаков А. Подход к моделированию механических систем твердых тел // Препринт 10; ЗАО ЛЕДАС. – Новосибирск, 2004. – 60 с.
5. Жермен-Лакур П., Жорж П. Л., Пистр Ф., Безье П. Математика и САПР: В 2-х кн. / пер. с франц. – М.: Мир, 1989. – 264 с.
6. Зенкевич О. Метод конечных элементов в технике / пер. с англ. – М.: Мир, 1975.
7. Ландау Л. Д., Лифшиц Е. М. Теоретическая физика: учеб. пособие. В 10 т. Т. I. Механика. – М.: Наука, 1988. – 216 с.
8. Ли К. Основы САПР (CAD/CAM/CAE). СПб.: Питер, 2004. – 560 с.
9. Маслов Л. Б. Численные методы механики. – Ивановский государственный энергетический университет. Публикация доступна в электронном виде по адресу <http://www.ispu.ru/library/lessons/Maslov/index.html>.
10. Митрофанов С. П. Групповая технология машиностроительного производства: В 2-х т. – Л.: Машиностроение, 1983. – 786 с.
11. Прейс С., Снытников Н., Ушаков Д. Collaborative Design-2 (LCS): Концепция, архитектура и вычислительные методы // Препринт 8; ЗАО ЛЕДАС. – Новосибирск, 2004. – 40 с.
12. Сапрыкин В. Н. Техническая механика: учеб. – М.: Экзамен, 2005. – 560 с.
13. Сидоров В.. Программирование в ограничениях с черными ящиками // Препринт 2; ЗАО ЛЕДАС. – Новосибирск, 2003. – 39 с.
14. Стародубов В. Роль и место PLM в линейке ERP, CRM и SCM // Журнал CIO № 7 (28) 2004. <http://www.cio-world.ru/analytics/review/35177>.
15. Ушаков Д., Телерман В. Системы программирования в ограничениях (обзор). Системная информатика. – Новосибирск: Наука; Сибирская издательская фирма РАН, 1999.

16. Хофманн К. М. Введение в двумерные задачи удовлетворения геометрических ограничений // Препринт 14, ЗАО ЛЕДАС. – Новосибирск, 2004. – 68 с.
17. Altafini C. and Frezza R. Motion on submanifolds of noninvariant holonomic constraints for a kinematic control system evolving on a matrix Lie group // Systems and Control Letters, 50(3): 241–250, 2003.
18. Catmull E. and Clark J. Recursively generated B-spline surfaces on arbitrary topological meshes // Computer-Aided Design 10 (Sept. 1978), 350–355.
19. Chaikin G. An algorithm for high speed curve generation // Computer Graphics and Image Processing 3 (1974), 346–349.
20. Doo D. and Sabin M. Behaviour of recursive division surfaces near extraordinary points // Computer-Aided Design 10 (Sept. 1978), 356–360.
21. Dulmage A. L. and Mendelsohn N. S. Coverings of Bipartite Graphs // Canad. J. Math. 10 (1958), 517–534.
22. Ershov A., Ivanov I., Preis S., Rukoleev E., Ushakov D. LGS: Geometric Constraint Solver // Preprint 4, LEDAS Ltd. – Novosibirsk, Russia (2003).
23. Evans M. Consultant's Viewpoint on Product Lifecycle Management // Financial Times Information Technology (2002). Публикация доступна в электронном виде по адресу <http://specials.ft.com/ftit/april2002/FT3BK49PJZC.html>.
24. Fares C. and Hamam Y. Collision Detection for Rigid Bodies: A State of the Art Review // Proceedings of 15th International Conference on Computer Graphics and Applications GraphiCon'2005. – Novosibirsk, Institute of Computational Mathematics and Mathematical Geophysics, Novosibirsk, 2005 (6–14).
25. Farin G., Hoschek J. and Kim (eds.) M.-S. Handbook of Computer Aided Geometric Design // Elsevier Science (North Holland), 2002. – 848 р.
26. Farinacci C. PLM and CRM: A Not-So-Odd Couple // CMC Report (2004). Публикация доступна в электронном виде по адресу <http://www.insightexec.com/cgi-bin/item.cgi?id=130900>.
27. Gillespie R. B. and Colgate J. E. A Survey of Multibody Dynamics for Virtual Environments // Proceedings of the ASME Dynamic Systems and Control Division. ASME, 1997, DSC-Vol. 61 (45–54).
28. Han J.-H., Pratt M. and Regli W. Manufacturing Feature Recognition from Solid Models: A Status Report // IEE Trans. on Robotics and Automation (2000).

29. Hoffmann C. M., Lomonosov A and Sitharam M. Finding solvable subsets of constraint graph // Principles and Practice of Constraint Programming CP'97 (1997), 463–477.
30. Jermann C., Mathis P., Neveu B., Schreck P., Trombettoni G. Decomposition of Geometric Constraint Systems: A Survey // Int. J. Comp. Geometry & Applications (2005), in print.
31. Laumond J.-P. Motion Planning for PLM: State of the Art and Perspectives // Int. J. Product Lifecycle Management. Vol. 1. No. 2. 2006.
32. Mortenson M. E. Mathematics for Computer Graphics Applications. – 2nd ed. – Industrial Press Inc., 1999. – 354 p.
33. On-Line Geometric Modeling Notes. Computer Science Department, University of California, Davis. Публикация доступна в электронном виде по адресу <http://graphics.idav.ucdavis.edu/education/CAGDNotes/homepage.html>.
34. Owen J. Algebraic solution for geometry from dimensional constraints // Proc. 1st ACM Symposium on Solid Modeling and CAD/CAM Applications, ACM Press (1991), 397–407.
35. PLM and ERP Integration: Business Efficiency and Value // CIMdata Report (2005). Публикация доступна в электронном виде по адресу http://www-03.ibm.com/solutions/plm/doc/content/bin/PLM_to_ERP_Integration_White_Paper_Final.pdf.
36. Product Lifecycle Management: «Empowering the Future of Business» // CIMdata Report (2002). Публикация доступна в электронном виде по адресу http://www.ariondata.com/Contenidos/Hojas_de_datos/CIMdata_PLM_Empowering_Future_of_Business.pdf.
37. Shah J. J. and Mäntylä M. Parametric and Feature Based CAD/CAM. – John Wiley & Sons, Inc., 1995. – 619 p.
38. Trainelli L. The Vectorial Parameterization of Rotation and Motion // Scientific report DIA-SR 02–18, Politecnico di Milano, 2002. Публикация доступна в электронном виде по адресу <http://www.aero.polimi.it/~trainell/publications/papers/vecpar.pdf>.

Книги издательства «ДМК Пресс» можно заказать в торгово-издательском холдинге «АЛЬЯНС-КНИГА» наложенным платежом, выслав открытку или письмо по почтовому адресу: **123242, Москва, а/я 20** или по электронному адресу: **orders@aliants-kniga.ru**.

При оформлении заказа следует указать адрес (полностью), по которому должны быть высланы книги; фамилию, имя и отчество получателя. Желательно также указать свой телефон и электронный адрес.

Эти книги вы можете заказать и в Интернет-магазине: www.aliants-kniga.ru.

Оптовые закупки: тел. **(495) 258-91-94, 258-91-95**; электронный адрес **books@aliants-kniga.ru**.

Ушаков Дмитрий Михайлович

Введение в математические основы САПР: курс лекций

Главный редактор *Мовчан Д. А.*
dm@dmk-press.ru

Корректор *Чанинова А. А.*

Верстка *Чанинова А. А.*

Дизайн обложки *Мовчан А. Г.*

Подписано в печать 18.08.2010. Формат 60×90 $1/16$.

Гарнитура «PetersburgС». Печать офсетная.

Усл. печ. л. 16. Тираж: 1000 экз.

Web-сайт издательства: www.dmk-press.ru