

5. РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ РАСЧЕТНО-ГРАФИЧЕСКОГО МОДЕЛИРОВАНИЯ В САПР

5.1. Структура программного комплекса и данных

Комплекс интерактивного расчетно-графического программирования включает системы (подсистемы): векторной машинной графики "Вектор" и растровой машинной графики CG (в транспьютерном варианте реализации - TCG), а также проблемно-ориентированные модули (подсистема "Аппарат конструктора" для проектирования судовых линий, подсистема "Теоретический чертеж" (ТЧ) для проектирования теоретического чертежа корпуса судна и т.д.). Комплекс работает под управлением программы "Монитор графического диалога", обеспечивающей ведение графического диалога, создание сценария диалога, обработку ошибок, выполнение простейших расчетов и программирование на уровне пользователя, не являющегося квалифицированным программистом.

Организация комплекса по модульному принципу (рис. 5.1) обусловлена:

- необходимостью оптимального использования ограниченного объема оперативной памяти РС IBM,
- выполнением условия открытости системы.

Система "Вектор" состоит из следующих модулей.

Ves_Аппарат конструктора (Ves_АК) - модуль генерирования плоских и пространственных линий с поддержкой их в БД "Аппарата конструктора" (БДАК).

Ves.1PS - модуль задания простейших аналитических поверхностей (сферы, эллипсоида, конуса, цилиндров и их отсеков), а также поверхностей, моделируемых через точечный каркас по линейному, квадратичному, кубическому и сплайновым законам.

Ves_Line - модуль автономного задания и исследования аналитических, лекальных и составных линий как в самом модуле, так и в задаваемых пользователем отдельных файлах. В модуле реализована возможность распознавания пространственных линий по их плоским проекциям, представленным в формате .rsx. Передача линий, сгенерированных описываемым модулем, в другие модули осуществляется либо их включени-

ем в качестве базовых для этих модулей, либо через специальным образом организованные файлы обмена данными.

VEC_R - модуль формирования поверхностей вращения и винтовых. В качестве образующей линии используются линии из набора самого модуля и линии, генерируемые модулями Vec_Line и Vec_AK системы "Вектор" или системой AutoCAD.

VEC_CK - модуль генерирования кинематических и специального контура поверхностей по направляющей линии, задаваемой отрезком прямой или сплайном. В модуле реализованы известные методы формирования поверхностей по методу "Специального контура" (СК).

Vec_Gran - модуль рациональной линейной, квадратичной и кубической параметризации поверхностей по линиям контура, генерируемым модулем, или по линиям, генерируемым в БДАК или системой AutoCAD. Перечисленные методы моделирования традиционны и применяются во многих геометрических системах. Использование же гиперключевого метода - явного, полупараметрического и параметрического способов заданий поверхностей, а также аппарата неявных управляющих функций в сочетании с этими методами повышает гибкость модуля при его использовании в проектировании судовых форм.

VEC_Kf - модуль построения двумерных и трехмерных сеток, а также различных конформных рельефов и топологических преобразований над ними. Имеется возможность "чтения" сеток (массива точек в формате rlf), описанных пользователем на любом языке программирования, или "вызова" сеток из библиотеки программ системы "Вектор".

Vec.Solid - модуль задания простейших фигур - тел (параллелепипеда, различных многогранников, конуса, шара, цилиндра и т.п.) и решения с этими телами задач позиционного и метрического характера, а также задания трехмерных фигур через точечный каркас по линейному, квадратичному, кубическому и сплайновым законам.

Vec_S.R - модуль генерирования трехмерных фигур вращением или винтовым перемещением граней.

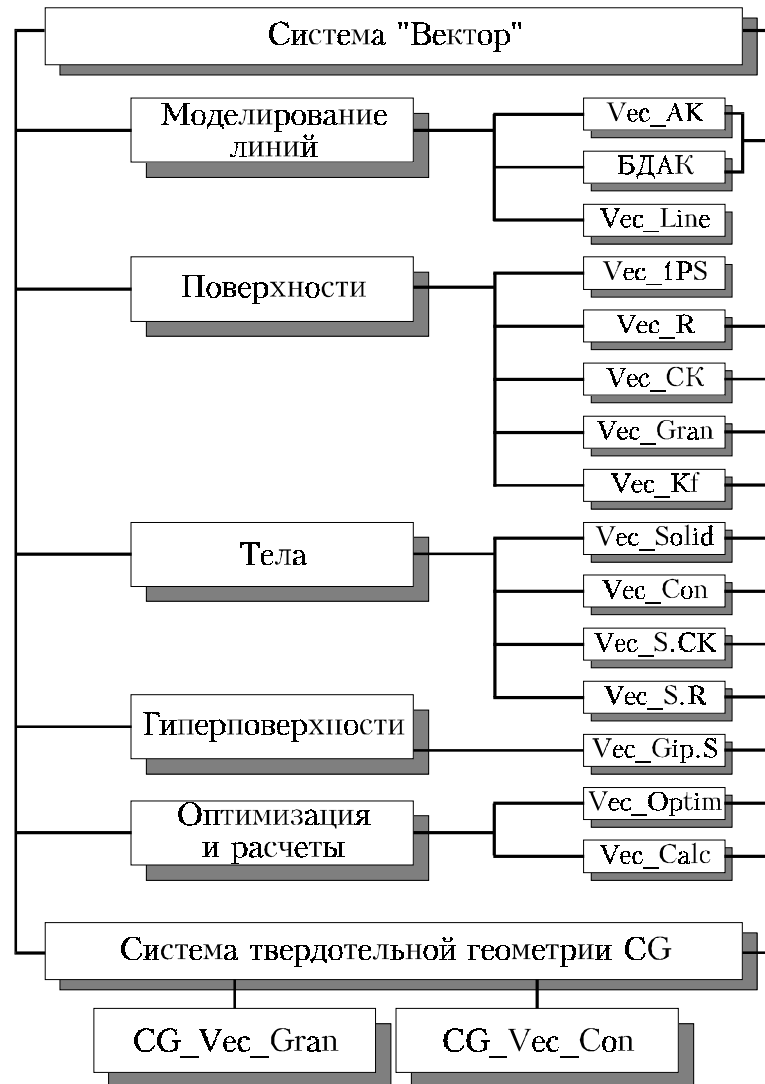


Рис. 5.1. Модульная структура системы "Вектор"

Vec_S.CK - модуль генерирования тел по методу специального контура (например, мгновенного преобразования от куба к гладким формам и далее к шару) движения граней по направляющим линиям (отрезку прямой или сплайновой кривой). В модуле возможности методов "СК", применяемых, в основном, для моделирования двумерных поверхностей, расширены в плане генерирования трехмерных фигур как тел. Данный подход наиболее удобен для проектирования и расчетов гребных винтов и турбинных лопаток.

Ves_Cop - модуль трансверсального формирования трехмерных фигур по их граням. Модуль построен по идеологии модуля Ves_Gran, но для трехмерного случая.

Ves_Gip.S - модуль формирования четырехмерных объектов в трехмерном пространстве (как преобразования во времени одного объекта в другой по тому или иному закону) и трехмерных тел как гиперсечений этих четырехмерных фигур.

Ves_Optim - модуль визуализации и расчета целевых функций (в стандартном варианте - размерность до пяти) в самых различных рассеянных ортогональных и аксонометрических проекциях комплексного чертежа. Автоматическое построение изолиний ЦФ обеспечивает определение зоны предпочтительного решения и зоны Парето в многокритериальных задачах. Из данного модуля можно непосредственно обращаться к программам генерирования ЦФ, что позволяет, последовательно "сужая" области ограничений, "подойти" к точке минимума или максимума ЦФ. Задание ЦФ описывается на языке СИ по специальной программе-графарету (двух - трех вариантов) и для пользователя не представляет затруднений. Модуль также обеспечивает возможность проверки различных топологических преобразований четырехмерных форм. Гиперсечения таких форм записываются в специальный файл и могут быть использованы в твердотельной системе CG при формировании сцен динамического преобразования объектов.

Организирующим элементом всех модулей является монитор графического диалога (МГД) со встроенным в него языком "Калькулятор". Монитор обеспечивает диалог пользователя с пакетами прикладных программ, прием и расшифровку директив, контроль правильности задания данных, запуск процедур и переход по диаграмме состояний. Задание директив осуществляется с помощью функциональной клавиатуры, джойстика или светового пера. Более подробное описание о МГД дано в прил. 3.

Язык систем, позволяющий составлять комбинации из математических предложений, директив графической системы и обращений к программам ПНП, определяет современный пользовательский инструмент интерактивного расчетно-графического программирования в прикладных задачах.

Во всех модулях программного комплекса на основе транслятора УАСС реализован язык пользователя "Калькулятор", который представляет собой расширение языка программирования СИ. Язык "Калькулятор" ориентирован на эксплуатацию системы пользователем, не обладающим квалификацией программиста.

Язык и графические возможности системы "Вектор" предоставляют пользователю удобный инструмент овладения методами геометрического программирования: элементами векторной и линейной алгебры, аналитической геометрии, комплексных чисел, дифференциального исчисления, оптимизации и т.д. В системе реализован минимум возможностей языка СИ и максимум графических возможностей, достаточные не только для решения инженерных задач, но и для проработки научных идей. При этом весь ход интерактивного процесса решения задач оформляется в библиотеку макрокоманд, которые в дальнейшем могут быть использованы в других задачах.

Язык " Калькулятор " включает следующие компоненты:

- ПЕРЕМЕННЫЕ :

n-n199 - целые: s-s199, x-x199,y-y199,z-z199 - вещественные: P(x,y,z)-p199(x199,y199,z199) - вещественные типа "точка". Переменные могут быть локальными (например;loc s1-s10) и глобальными /по умолчанию/. Нумерация регистров - для локальных переменных не ограничена.

- ОПЕРАЦИИ:

+ - сложение, - - вычитание, * - умножение, / - деление
= - присвоение.

- ОПЕРАТОРЫ:

? - условный оператор " если " /ставится после условия/
goto - переход /вперед/ по метке, состоящей до 8 символов /включая цифры/ английского алфавита. Перед меткой в первой позиции /перед пробелом/ должен стоять символ \$
error - обработка ошибочных условий
exit - выход из МК
free - освобождение объявленных ранее локальных переменных.

- ЛОГИЧЕСКИЕ ОТНОШЕНИЯ И ОПЕРАЦИИ:

<> - равенство > - больше < - меньше
& - и ! - или ^ - не

- ВСТРОЕННЫЕ ФУНКЦИИ:

sin - синус; cos - косинус; tan - тангенс;
 cotan - котангенс; log - натуральный логарифм;
 exp - показательная функция; sqrt - квадратный корень и т.д.

- ДИРЕКТИВЫ:

print\$on/off/ - включить /выключить/ печать текста МК
 msg\$off/on/ - выключить /включить/ сообщения расчетов

- ВЫЗОВ МК:

< имя МК > : < список присвоений >

Если МК начинается с n, s, p, x, y, z, то при ее вызове, перед именем МК ставится /без пробела/ символ \$. Возможен рекурсивный вызов МК, что позволяет организовывать циклы.

В среде системы геометрического кормплекса, поддерживаемого монитором МГД, можно выполнят ь также следующие действия:

- обращение к программам, написанным на языках СИ, ФОРТРАН, Бейсик;
- использование массивов $[x, y, z]$, полученных средствами данной системы или других систем, например ФОРАН;
- выполнение расчетов по программам пакета научных исследований SYS.SPPLIB для ЕС ЭВМ, адаптированного для ПЭВМ;
- редактирование макрокоманд без выхода из системы;
- запись рассчитываемых параметров в специальный файл с целью их сохранения для использования в следующем сеансе работы;
- визуализацию рисунков, получаемых в других модулях комплекса или других системах, в частности, системах AutoCad и Форан;
- чтение рисунков, получаемых в других системах и распознавание линий , снимаемых с помощью сканера.

В настоящее время с помощью системы "Вектор" создана библиотека макрокоманд по методам аналитической геометрии и векторной алгебры, используемая как дополнительный инструмент различных реалистических сцен и анимаций в системе "CG". Например, для создания эффекта движения объекта в лабиринте архитектурных форм необходимо не только задать его движение от точки А к точке В, но и выполнить преобразования типа совмещения центра сцены с направлением движения объекта и т.п.

Другим компонентом комплекса расчетно-графического программирования является система твердотельного моделирования CG - совме-

стная разработка с сотрудниками ИАПУ ДВО РАН. Эта система обеспечивает моделирование реалистических сцен и объектов по геометрическим примитивам самой системы, а также поверхностям, генерируемым модулями системы "Вектор". Связь между системами CG и модулями многомерного моделирования системы "Вектор" осуществляется на уровне обмена файлами. Модули *Ves_gran* и *Ves_con* системы "Вектор" подключены непосредственно к системе CG, образуя новые модули *CG_Ves_Gran* и *CG_Ves_Con*. Такая структура ПО позволяет наиболее эффективно использовать возможности векторного и растрового моделирования.

Связь комплекса с системой AutoCAD обеспечивает возможность использования сервиса этой системы для оформления чертежей моделирования объектов и вывода их на плоттер.

Предусмотрена возможность передачи в модуль *Ves_Calc* информации о геометрических образах из любого модуля системы "Вектор" (или автономной программы) в виде массива точек для расчета различных дифференциально-геометрических характеристик моделируемых форм (например, расчета площади, объема, построения эквидистанты кривой или поверхности, построения геодезических линий и т.п.), а также для получения всевозможных преобразований и изображений с выводом их на плоттер.

Все инвариантные и проблемно-ориентированные модули систем "Вектор" и "CG_Вектор" имеют собственный язык пользователя, могут использовать линии из базы данных модуля "Аппарат конструктора" в качестве формообразующих линий, а также обладают некоторыми возможностями полуавтоматического распознавания линий по их двум проекциям для формообразующих линий при генерировании геометрических форм.

Системы Вектор, CG, АК, "Теоретический чертеж" имеют организацию данных как на уровне СУБД, так и файловую.

Управление базой данных систем "Вектор" и "СГ"

Формирование структурированного представления сложного технического объекта осуществляется с помощью той или иной совокупности модулей системы "Вектор", которые при работе используют единую базу данных для формирования цифровой модели объекта. База данных состоит из графических данных, описывающих объект (например, значения переменных, описывающих кривые и поверхности, координаты точек граней объекта, его объема и т.п.), и вспомогательных данных, относящихся, например, к характеристикам положения в пространстве, типов используемых кривых и пр.

При разработке графической системы необходимо обеспечивать создание и корректировку базы данных как в диалоговом, так и пакетном режимах. Системные программы этого назначения объединены в подсистему "СУБД". На рис.5.2 приведена структура данных, описывающих один структурированный типоучасток.

"Головной узловой элемент" является основой структуры банка данных системы "Вектор". Он содержит кодовое название и пароль разрабатываемого проекта. Пароль необходим для санкционирования доступа пользователя к базе данных системы, который осуществляется следующим образом:

- при создании нового банка в головной элемент базы данных записывается пароль пользователя, состоящий из восьми буквенно-цифровых знаков;

- при инициализации системы "Вектор" производится запрос пароля у пользователя, который сравнивается с паролем, записанным в головной элемент. При несовпадении этих паролей доступ к базе данных системы "Вектор" невозможен.

Номер типоучастка содержится в ассоциаторе (AS), который является связующим звеном между головным узловым элементом и данными, описывающими типоучасток. С использованием "номера типоучастка" происходит последовательный просмотр содержимого ассоциаторов. При совпадении содержимого какого-либо ассоциатора с заданным номером типоучастка открывается доступ к данным выбранного типоучастка, а при несовпадении выводится сообщение, что такого типоучастка в базе данных нет.

Данные, необходимые для описания типоучастка, распределены по трем уровням.

Первый уровень составляет узловой элемент, связанный с соответствующим ассоциатором. Узловой элемент содержит координаты точек (xk, xc, xv) по оси x для определения плоских сечений $\psi_1(y)$, $\psi_0(y)$, $\psi_u(y)$, а также экстремальные значения точек линий $f_2(x)$, $\varphi_2(x)$, необходимые для построения сечения на генерируемой поверхности типоучастка, параллельного плоскости xz .

К узловому элементу первого уровня присоединены данные двух видов, которые идентифицируются двумя типами рингстартов (RS): 0 и I, соответственно.

Ко второму уровню (рингстарт I) присоединены данные, содержащие общую информацию о типоучастке и разнесенные по трем атрибутивным элементам.

"Атрибутный элемент I" содержит информацию о виде кривых, представляющих контур типоучастка в пространстве.

"Атрибутный элемент 2" используется при табличном задании кривых контура и содержит информацию о количестве опорных точек для кривых (значения переменных NF , NFI , $NPSIK$, $NPSIC$, NUA).

В "атрибутном элементе 3" хранится информация о видах задаваемого контура типоучастка (значение переменной $NUPR$) и сшивающей функции (значение переменной $NCHIV$), о степени сглаживания (значение переменной RO) для сплайновой аппроксимации со сглаживанием.

В третий уровень (рингстарт 0) собраны все необходимые данные для графического построения линий контура типоучастка. Этот уровень состоит из пяти подобных атрибутивных элементов, содержащих различные данные, в зависимости от вида используемых кривых, для каждой линии контура типоучастка.

Реализация описанной структуры базы данных системы "Декарт" осуществлена в среде систем управления базами графических данных "Уссур" [15] для ЕС ЭВМ и VISTA для ПЭВМ. Эта структура отвечает требованиям, налагаемым на диалоговую систему: обеспечивает легкость поиска и необходимую коррекцию данных проектируемого типоучастка.

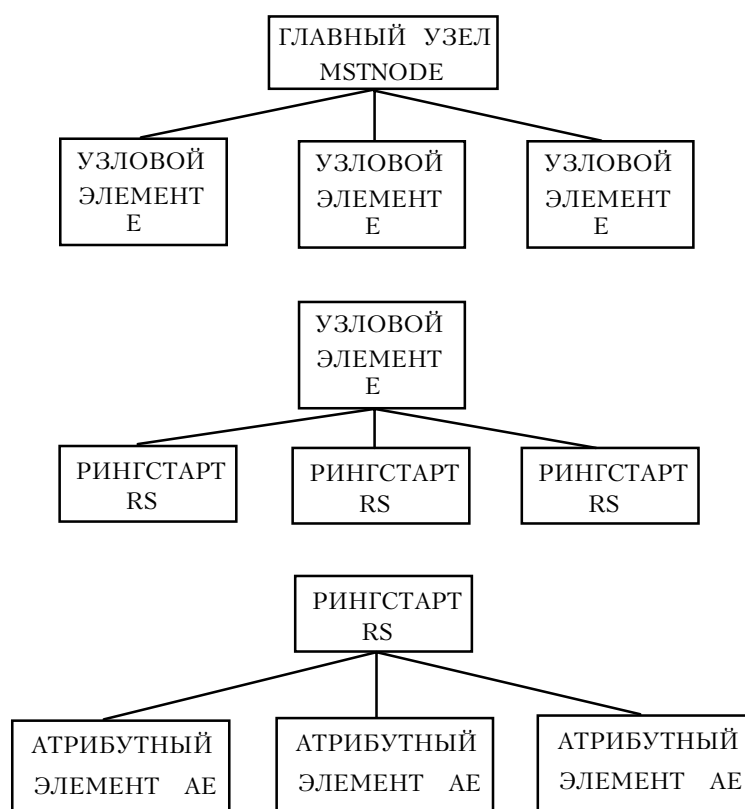


Рис.5.2. Общий вид структуры данных системы "Вектор"

Управление базой данных системы "CG" поддерживает два инструмента для работы с описанием геометрической сцены, ориентированные на разные режимы, но решающие одну задачу - задачу сохранения описания сцены во внешней памяти и восстановления его для последующей работы со сценой.

Первый инструмент - это макрокомандные средства, предназначенные для работы в интерактивном режиме. Они позволяют явно формировать или автоматически генерировать параметрическое описание сцены, хранить его в файле внешней памяти и читать описание в оперативную память для последующего редактирования и визуализации сцены.

Альтернативой инструменту макрокоманд являются средства организации оперативной базы данных для хранения описания сцены. Они позволяют пользователю программировать требуемую схему БД и ориентированы на пакетный режим работы. Преимущество этого способа в сравнении с макрокомандным состоит в более высокой скорости обработки, что проявляется заметнее для насыщенных сцен. Другое его пре-

имущество - возможность хранения в БД проблемно-ориентированных атрибутов, т.е. атрибутов, связанных с семантическими объектами.

Разработанные и описываемые ниже средства реализованы в среде СУБД VISTA.

Логическая организация БД представляет собой двухуровневое дерево, на первом уровне которого находятся конъюнкции, а на втором - элементы. Пользователь может записывать конъюнкции определенной на данной момент сцены в БД, либо считывать (добавлять) в сцену хранящиеся в БД конъюнкции. Доступ к конъюнкциям производится по ключу - имени конъюнкции.

По структуре диалога программы CG переход в состояние "Работа с БД" производится из состояния "Обработка сцены". В состоянии "Работа с БД" определены следующие операции:

- "открыть БД";
- "закрыть БД";
- "записать конъюнкцию";
- "прочитать конъюнкцию";
- "удалить конъюнкцию".

Операции "Открыть/Закрыть БД" введены для более эффективного использования оперативной памяти: открытая БД требует дополнительной памяти. Описание схемы БД в терминах языка описания данных СУБД VISTA даны в прил. 3.

5.2. "АППАРАТ КОНСТРУКТОРА" - подсистема формирования плоских и пространственных линий при генерации судовых обводов

Процесс формирования поверхности корпуса судна является итерационным и состоит из двух основных этапов - сначала строится некоторое приближение этой поверхности, затем путем численных расчетов уточняются ее детали, обеспечивающие достижение заданных параметров; в случае неудовлетворительных результатов проектирования осуществляется возврат в исходное состояние, строится новое приближение поверхности и т.д. [1]. На всех этапах проектирования поверхности корпуса конструктору приходится работать с кривыми сложной формы - формообразующими линиями и различными сечениями корпуса. Для облегчения этой работы в качестве удобного и мощного инструмента манипулирования плоскими геометрическими объектами предлагается подси-

стема построения формообразующих линий, названная "Аппаратом конструктора" ("АК").

5.2.1. Модель формообразующей линии

Геометрическая сложность поверхности корпуса судна отражается на сложности ее формообразующих линий. Математическое описание таких линий как цельных объектов в общем случае вызывает затруднения, а зачастую и просто невозможно. Поэтому на практике используется прием математического описания формообразующих линий по участкам. Исходя из этого, формообразующие линии определяются как составные объекты, образованные из последовательности независимых элементов - сегментов линии, каждый из которых может быть описан отдельно. Разбиение линии на сегменты и выбор способа описания каждого сегмента выполняется конструктором. Подсистема "АК" предоставляет пользователю возможность применять предварительно определенные типы сегментов (прямая, дуга окружности, параметрический сплайн) или вводить собственные описания для дополнительных типов сегментов (пользовательских).

Ввод в подсистему "АК" нового типа сегмента сводится к описанию параметров представления этого сегмента на специальном языке и созданию процедуры-функции, вычисляющей координатные значения точек сегмента. После выполнения процедуры генерации описания новый тип сегмента может использоваться точно так же, как и предопределенный тип.

Сегмент линии представляется упорядоченной парой точек, задающих начало и конец сегмента, и функцией, определяющей геометрию сегмента. Граничные точки сегментов задают разбиение формообразующей линии на смежные участки и называются узлами линии.

В процессе конструирования можно модифицировать узлы, тип и параметры функций для каждого сегмента. Соответствие задания сегмента требуемым критериям определяется пользователем на основании визуальной оценки форм сегмента и линии в целом, а также оценки их численных характеристик. Численные характеристики - производная в заданной точке, длина линии на заданном участке, площадь подграфика участка линии и др. - могут быть получены в режиме диалога с "АК".

Помимо задания непосредственных границ и параметров сегмента имеется возможность "сшивания" соседних сегментов, т.е. выбора таких границ и параметров, чтобы удовлетворялись условия пересечения или касания данного сегмента, других сегментов.

Совокупность формообразующих линий во внутреннем цифровом представлении составляет базу данных подсистемы "АК" (БДАК). База данных имеет двухуровневую иерархическую структуру. Первый уровень составляют линии как объекты. Формообразующие линии задаются в собственных локальных системах координат, называемых системами координат линий. Для каждой линии в БДАК хранится информация о расположении ее плоскости в пространстве и об области определения линии (минимальные и максимальные координаты ее точек).

Второй уровень иерархической структуры - сегменты, из которых состоят линии. Сегменты идентифицируются номером или координатами точки в области определения сегмента (прямоугольное окно, границы которого задаются координатами сегмента). Для каждого сегмента в БДАК хранятся его границы, тип и параметры функции, определяющей его геометрию.

5.2.2. Структура подсистемы построения формообразующих линий

Центральным элементом структуры подсистемы является база данных (рис.5.3), содержащая внутренние представления формообразующих линий. В базе данных хранятся как конечные результаты конструирования - завершенные формообразующие линии, так и результаты его промежуточных этапов, а также заготовки - фрагменты линий и отдельные их сегменты. База данных БДАК объединяет компоненты подсистемы в целое и связывает последовательные сеансы работы конструктора в непрерывный процесс. Работа с базой данных поддерживается системой управления структурированными данными СУБД. Язык манипулирования данными преобразуется модулем прикладного интерфейса в специальный проблемно-ориентированный язык, который используется в других программах подсистемы. Такое решение облегчает разработку геометрических модулей подсистемы и, при необходимости, предоставляет возможность заменить используемую СУБД.

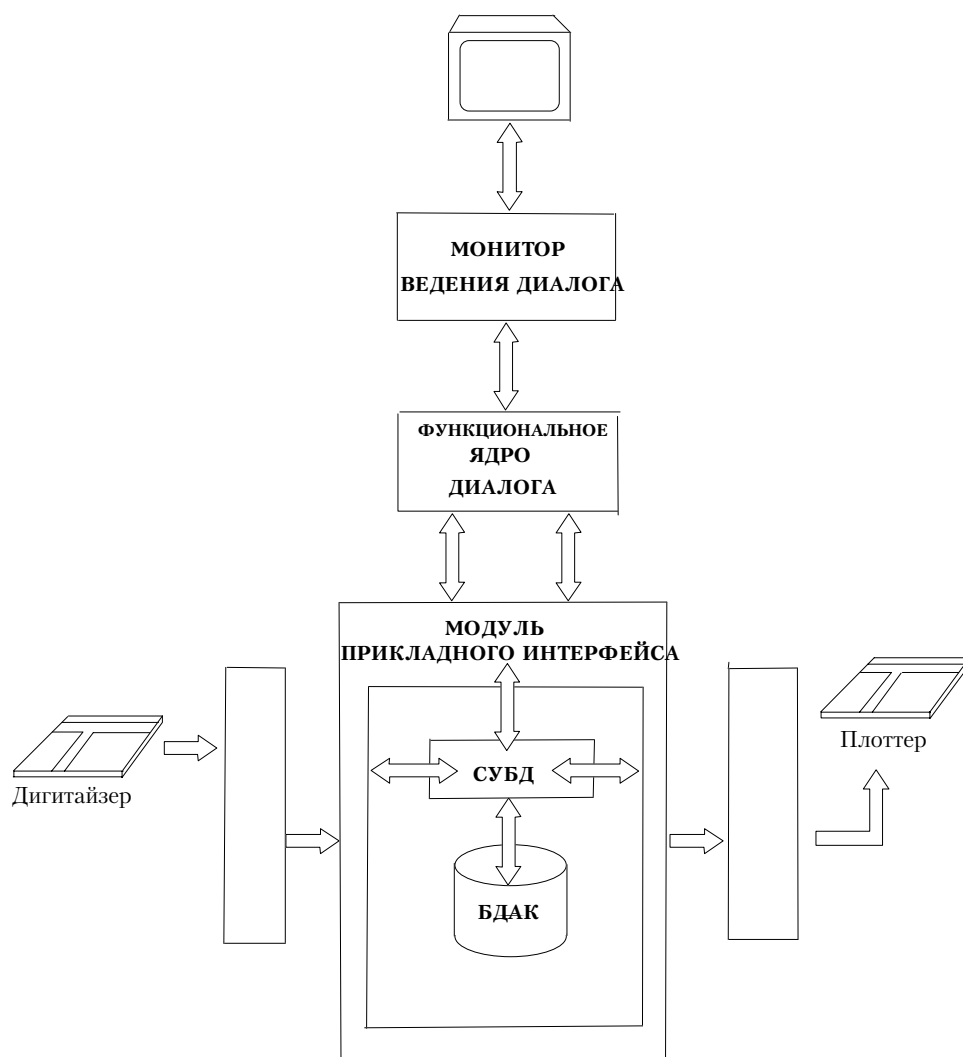


Рис.5.3. Структура построения формообразующих линий подсистемы "АППАРАТ КОНСТРУКТОРА" системы "Вектор"

Загрузка в БДАК исходных геометрических данных большого объема выполняется с помощью компонента "Загрузка БДАК". Исходные геометрические данные формируются на основе чертежно-графической документации путем непосредственной их оцифровки с использованием устройства ввода графической информации или подготавливаются в виде символьных файлов на основе заданных числовых таблиц.

Непосредственное конструирование формообразующих линий выполняется в режиме диалога конструктора-пользователя с подсистемой "АК". Эта часть подсистемы включает модуль "Функциональное ядро диалога", два монитора ведения диалога и пакетный монитор. Интерактивная программа образуется парой - модулем "Функциональное ядро

диалога" и одним из диалоговых мониторов. Модуль "Функциональное ядро" содержит ряд таблиц, описывающих схему диалога: диалоговые состояния, используемые меню и их взаимосвязи, типы вводимых данных, а также программы, реализующие функции, указанные в меню диалоговых состояний. Мониторы ведения диалога представляют собой таблично-управляемые мониторы, реализующие диалог на том или ином типе терминала - графическом дисплее .

Получение конечной документации по готовым формообразующим линиям в графической и числовой форме выполняется компонентом "Вывод чертежей и таблиц". Этот компонент обеспечивает оформление результатов конструирования в соответствии с определенными требованиями.

Рассмотренные компоненты подсистемы конструирования составляют ее базовый вариант, поддерживающий решение задачи конструирования формообразующих линий в исходной постановке. Модульная структура подсистемы позволяет расширить множество модулей при изменении этой постановки.

5.2.3. Интерактивное конструирование формообразующих линий

Основные операции по конструированию формообразующих линий поддерживаются следующими компонентами подсистемы "АК":

- графической интерактивной программой,
- интерактивной программой для алфавитно-цифрового дисплея,
- пакетной (неинтерактивной) программой.

Наиболее развитым набором операций обладает графическая интерактивная программа (ГИП), рассматриваемая в этом разделе. Две другие программы отличаются от нее тем, что они не поддерживают визуализации линий и графического ввода данных. Кроме того, в пакетной программе ввод данных производится из заранее подготовленного файла.

Диалог с оператором дисплея в ГИП осуществляется с использованием системы меню, с помощью которых конструктор выбирает требуемые ему операции. При выполнении геометрических функций используются координатные устройства дисплея (джойстик или кодирующий планшет), алфавитно-цифровая и функциональная клавиатура. Работа ГИП представляется диаграммой состояний, приведенной на рис.2. Каж-

дое из таких состояний характеризуется множеством команд, отражаемых в меню (возможные переходы между состояниями отмечены на рисунке стрелками). Оператору дисплея предоставлена возможность отмены действия последней введенной команды. Для этого используется специальная директива "отмена", которая восстанавливает состояние процесса конструирования, имевшееся перед вводом отменяемой команды.

Основным состоянием ГИП является состояние "Работа с линиями". В это состояние ГИП переходит при ее запуске, им же завершается сеанс работы ГИП. В состоянии "Работа с линиями" выполняются операции по регистрации в БДАК новых линий, удалению старых, созданию копий линий и объединению линий (теоретико-множественное объединение пары линий). В состоянии "Работа с линиями" также выполняются операции выдачи справок, например, списка всех линий, имеющих в БДАК.

Состояние "Геометрия линии" позволяет работать с линией в целом - имеется возможность модифицирования множества узлов линии и формирования отдельных сегментов. Последняя функция выполняется путем перехода в другое состояние - "Геометрия сегмента". При добавлении и удалении узлов геометрия одного или двух смежных сегментов перестает удовлетворять новому положению узлов, поэтому такие сегменты заменяются на прямолинейные. При переходе в состояние "Геометрия линии" необходимо установить текущую линию, с которой будет производиться работа.

Геометрия отдельного сегмента линии задается в состоянии "Геометрия сегмента". Работа на этом этапе ведется над сегментом, выбранным в состоянии "Геометрия линии". Предусматривается ряд альтернативных способов задания геометрии сегмента. Имеется две категории способов задания - независимое от других сегментов задание текущего сегмента путем указания условий на связь с другим сегментом (или таковой). Независимое задание геометрии сегмента производится в состоянии "Параметры сегмента данного типа" (состояния 4,5,6,7 на рис.5.4) и может выполняться несколькими способами. Например, дугу окружности можно задать одной средней точкой, величиной радиуса, направлением касательной на левом или правом конце (концевые точки дуги - границы сегмента считаются заданными). Задание геометрии сегмента путем указания условий его связей с другими сегментами позволяет выполнить

"сшивание" сегментов по тем или иным правилам. В результате выполнения операций этой группы модифицируется исходное описание сегмента, например, его граничные точки. Обеспечиваются три типа связи сегментов: пересечение, касание и вставка.

Операция "пересечение" позволяет продолжить сегмент до пересечения с заданным сегментом. В результате выполнения операции граничные точки текущего сегмента устанавливаются в точку пересечения.

Операция "касание" изменяет геометрию сегмента таким образом, чтобы он гладко "сшился" с заданным сегментом. Геометрия текущего сегмента должна иметь одну степень свободы. В этом случае также изменяются границы текущего сегмента.

Операция "вставка" изменяет геометрию сегмента таким образом, чтобы он гладко "вшился" между двумя заданными сегментами. Геометрия текущего сегмента должна иметь две степени свободы. Границы текущего сегмента устанавливаются в точки "сшивания".

Различные численные расчеты по хранимым в БДАК линиям объединены в состоянии "Расчеты на линии". Могут быть вычислены и выведены на дисплей, например, векторы касательных в заданных точках, радиусы кривизны, площадь под линией и т.п.

Состояние "Определение системы" координат ввода/вывода позволяет устанавливать различные системы координат для ввода и вывода графической информации.

Переходы между состояниями ГИП выполняются по зарезервированным для этой цели функциональным кнопкам. При завершении работы ГИП все введенные в процессе сеанса изменения в описаниях линий сохраняются в БДАК и могут использоваться в последующих сеансах работы.

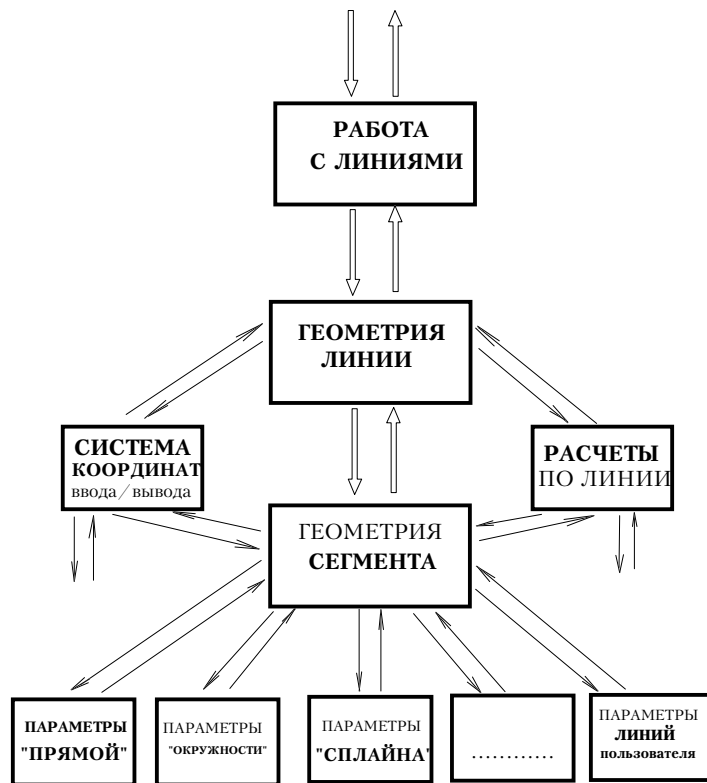


Рис. 5.4. Диаграмма состояний интерактивного конструирования формообразующих линий в АК

5.2.4. Алгоритмы -машинно - ориентированный подход к реализации элементарной геометрии в АК

Наибольшее распространение в САПР получили системы моделирования, имитирующие в части ввода информации действия чертежника за кульманом. Входной язык таких систем базируется на понятиях "сопряженность", "параллельность", "перпендикулярность". Удобным аппаратом математического описания этих понятий является теория функций комплексной переменной (ТФКП).

В рамках ТФКП прямые линии рассматриваются как окружности бесконечно большого радиуса, вследствие чего прямые и окружности не различаются. Поскольку обычно задачи на пересечение разделяются на подзадачи (пересечение отрезка с

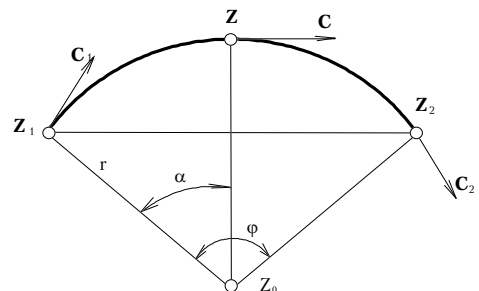


Рис. 5.5. Дуга на комплексной плоскости

отрезком, отрезка с дугой, дуги с дугой), представление дуг и отрезков функциями комплексной переменной упрощает их решение.

Зададим на комплексной плоскости (рис.5.5) направляющую дугу: $v = e^{i\varphi/2}$, где z_1 -начальная точка дуги, z_2 -конечная точка дуги, φ - центральный угол дуги ($\varphi > 0$ при вращении против ч.с., $\varphi < 0$ при вращении по ч.с.).т z_0 - центральный параметр дуги. Тройка (z_1, z_2, v) однозначно определяет дугу. Для отрезка центральный угол φ равен нулю и $v=1$, поэтому направленный отрезок на комплексной плоскости задается тройкой $(z_1, z_2, 1)$.

Параметрическое представление дуги и отрезка на комплексной плоскости имеет вид:

$$z = \frac{z_2 - hvz_1}{1 + hv}, \quad (5.1)$$

где параметром служит действительное число h : значению $h=0$ соответствует точка $z=z_1$, при $h \rightarrow \infty$ $z=z_2$.

Отображение h в точки дуги z непрерывно и взаимно однозначно. При $v=1$ формула (5.1) определяет отрезок с начальной точкой z_1 и конечной точкой z_2 .

В практических приложениях встречаются следующие задачи:

1) найти точки пересечения двух дуг (двух отрезков, отрезка и дуги) (рис.5.7);

2) построить отрезок с заданной начальной точкой, в своей конечной точке касающийся заданной окружности (рис.5.6,а);

3) построить касательный отрезок к двум окружностям (рис.5.6,б);

4) построить дугу с заданной начальной точкой, в своей конечной точке касающуюся заданной окружности или прямой;

5) построить касательную дугу к двум заданным объектам (рис.5.6,в);

6) построить дугу, касающуюся трех направленных окружностей в заданном порядке (задача Аполлония) (рис.5.6,г).

а)

б)

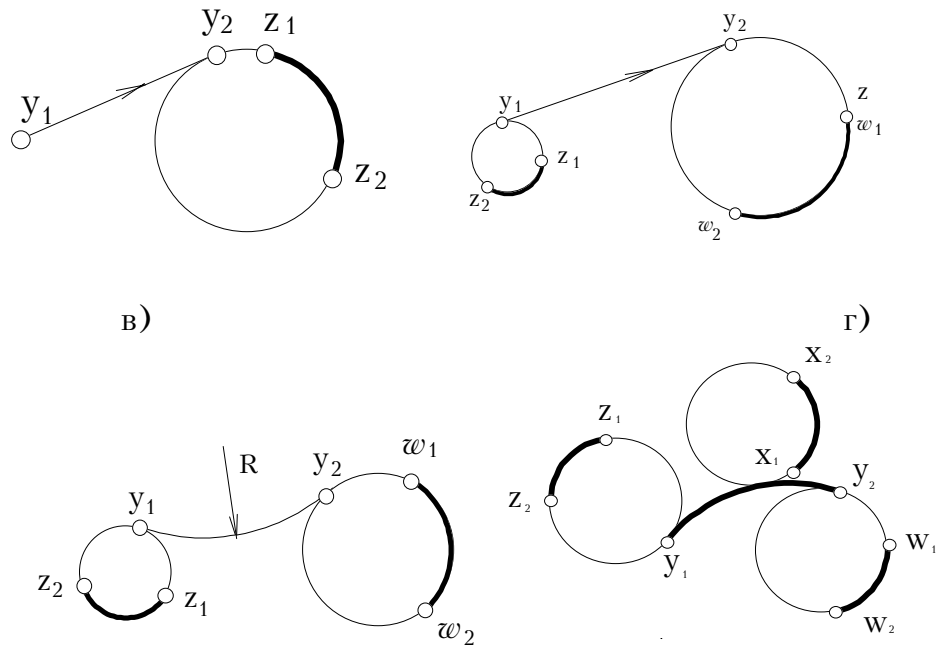


Рис.5.6

В решении этих задач можно выделить следующие шаги:

- составление билинейных уравнений, соответствующих поставленным задачам (шаг 1);
- вычисление коэффициентов билинейных уравнений (шаг 2);
- решение уравнений (шаг 3);
- вычисление параметров искомым уравнений (шаг 4).

Общей частью всех алгоритмов является шаг 3 этой схемы. Индивидуальные шаги 2 и 4 просты: коэффициенты уравнений вычисляются по исходным данным без предварительного анализа положения и выделения особых случаев.

Задача Аполлония (рис.5.6,г) решается по известной схеме геометрических преобразований:

- расширение третьей дуги на величину $-R$. Эта дуга сводится к точке;
- перенос на величину $-x_0$. Точка x_0 переходит в начало координат, точки начала/конца двух других дуг преобразуются по формуле: $z' = z - x_0$;
- инверсия.

Точка начала координат при инверсии перемещается в бесконечность, и задача Аполлония сводится к построению отрезка, касательного к двум окружностям

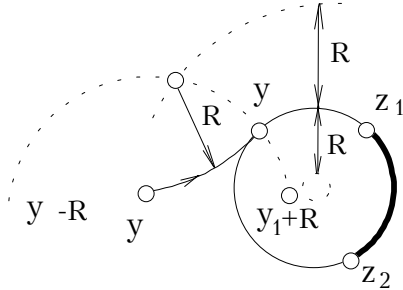


Рис.5.7

В качестве примера рассмотрим решение задачи 1: найти точку пересечения двух объектов (двух дуг, двух отрезков, отрезка и дуги).

Параметрическое представление первого объекта

$$z = \frac{z_2 + hvz_1}{1 + yv}. \quad (5.1)$$

Параметрическое представление второго объекта

$$z = \frac{\omega_2 + gv\omega_1}{1 + g\eta}, \quad (5.2)$$

где параметром служит неотрицательное вещественное число g . В точке пересечения должно выполняться равенство функций (5.1) и (5.2). Это равенство можно трактовать как уравнение с неизвестными h и g :

$$ahg + bh + cg + d = 0, \quad (5.3)$$

где

$$a = v\eta(z_1 - \omega_1); \quad b = v(z_1 - \omega_2); \quad c = \eta(z_2 - \omega_1); \quad d = z_2 - \omega_2.$$

Уравнение (5.3) решается аналитически как система двух билинейных уравнений с действительными коэффициентами. Возможны три случая:

- отсутствие решения (объекты не пересекаются);
- решение единственное (соответствует касанию объектов);
- уравнение имеет два решения (две точки пересечения объектов).

Проверка неотрицательности h, g определяет пересечение с основной частью дуги или дополнительной к окружности.

В системе "ВекторР" изложенная методика реализована в модулях Vec_Line (при файловой поддержке данных) и "Аппарат конструктора" (при работе с БДАК). С этой целью в базовом языке СИ для написания программ, реализующих выполнения алгебраических действий с комплексными числами, создан блок программ. В языке пользователя "Калькулятор" возможность работы с комплексными числами реализована через макрокоманды.

Более общие задачи элементарной геометрии, решаемые в системе "Вектор", приведены в талицах 5.1-5.6.

Инструмент комплексных чисел, (а именно - рельефы комплексных чисел) реализован в модуле системы "Вектор" и может использоваться для проектирования новых геометрических трехмерных форм (см. прил.2).

Задание прямой линии, окружности, таблично заданной кривой /ТЗК/, определение точек и переустановка начала/конца геометрического элемента линии /ГЭЛ/ по одной из заданных координат.

Таблица 5.1

N Задачи	Содержание типа задания
A1	Прямая b проходит через две заданные точки p1 и p2 .
A2	Прямая b проходит через точку p под заданным углом к оси x.
A3	Окружность c задается координатами центра и радиусом.
A4	Окружность c задается координатами центра и точкой, лежащей на окружности.
A5	Окружность c проходит через три заданные точки p1 , p2 , p3 .
A6	Окружность заданного радиуса проходит через точки p1 и p2 .
A7	Задание лекальной таблично заданной кривой (ТЗК).
A8	Определение точки на прямой по одной из ее координат.
A9	Определение точек на окружности по одной из их координат.
A10	Определение точек на ТЗК по одной из их координат.
A11	Переопределение начала и конца отрезка прямой.
A12	Переопределение начала и конца дуги окружности.
A13	Переопределение начала и конца визуализации ТЗК.
A14	Задание прямой линии, состоящей из нескольких участков.
A15	Задание окружности, состоящей из нескольких участков ее визуализации.
A16	Перезадание сегмента "прямая" дугой окружности заданного радиуса.
A17	Перезадание сегмента типа "сплайн" дугой окружности заданного радиуса.
A18	Перезадание сегмента типа "круг" и типа "сплайн" на сегмент типа "прям".

Определение точек пересечения ГЭЛ (прямая, окружность, ТЗК) и переустановка их начала/конца

Таблица 5.2

№ задачи	Содержание типа задания
Б1	Определение точки пересечения двух заданных прямых
Б2	Определение точки пересечения прямой b с окружностью a и окружности c с прямой b
Б3	Определение точек пересечения двух заданных окружностей.
Б4	Определение точек пересечения прямой b с ТЗК s .
Б5	Определение точек пересечения окружности a с ТЗК s и ТЗК c окружностью a .
Б6	Определение точек пересечения ТЗК s с ТЗК v
Б7	Пересечение двух прямых b_1 и b_2 .
Б8	Пересечение прямой b с окружностью a .
Б9	Пересечение окружности a с прямой b с переустановкой их концов.
Б10	Пересечение 2-х окружностей a_1 и a_2 с переустановкой их начала и конца.
Б11	Пересечение 2-х окружностей a_1 и a_2 с переустановкой их концов
Б12	Пересечение прямой b с ТЗК s с переустановкой конца прямой.
Б13	Вырез "дуга окружности" в прямой линии b .
Б14	Вырез "дуга окружности" в окружности a .
Б15	Вырез "дуга окружности" в ТЗК s .

Определение ГЭЛ через операции "Касание" и установка начала/конца касаемого сегмента

Таблица 5.3

№ задачи	Содержание типа задания
В1	Прямая b касается окружности a и проходит через точку p .
В2	Прямая b касается окружности a и проходит через точку p ($p \in a$)
В3	Прямая b_1 касается окружности a и параллельна прямой b
В4	Прямая b касается ТЗК s и проходит через точку p
В5	Прямая b_1 касается ТЗК s и параллельна прямой b
В6	Окружность c заданного центра o касается прямой b .
В7	Окружность c заданного центра o касается окружности a
В8	Окружность c заданного центра o касается ТЗК.
В9	Окружность c заданного центра o касается окружности a
В10	Окружность c касается окружности a и проходит через точки p_1 и p_2

V11	Окружность c касается прямой b и проходит через точки $p1$ и $p2$ ($p1 \in b$).
V12	Окружность c касается окружности a и проходит через точки $p1$ и $p2$ ($p1 \in a$)
V13	Окружность c касается ТЗК s и проходит через точки $p1$ и $p2$
V14	Окружность c касается ТЗК s и проходит через точки $p1$ и $p2$ ($p1 \in s$)
V15	Окружность c радиуса R касается прямой b и проходит через точку p
V16	Окружность c радиуса R касается прямой b и проходит через точку p ($p \in b$)
V17	Окружность c радиуса R касается окружности a и проходит через точку p
V18	Окружность c радиуса R касается окружности a и проходит через точку p ($p \in a$)
V19	Окружность c радиуса R касается ТЗК s и проходит через точку p
V20	Окружность c радиуса R касается ТЗК s и проходит через точку p ($p \in s$)
V21	
V22	

Задание прямой, дуги окружности, ТЗК в случае их сопряжения с ГЭЛ операцией "Вставка"

Таблица 5.4

N задачи	Содержание типа задания
Г1	Прямая b касается окружностей $a1$ и $a2$.
Г2	Прямая b касается окружности a и ТЗК s .
Г3	Прямая b касается ТЗК s и ТЗК v .
Г4	Окружность c радиуса R касается прямой b и окружности a .
Г5	Окружность c радиуса R касается окружностей $a1$ и $a2$.
Г6	Окружность c радиуса R касается прямой b и ТЗК s .
Г7	Окружность c радиуса R касается окружности a и ТЗК s .
Г8	Окружность c радиуса R касается ТЗК s и ТЗК v .
Г9	Окружность c касается прямых $b1$ и $b2$
Г10	Окружность c касается прямых $b1$ и $b2$ и проходит через точки p .
Г11	Окружность c касается прямых $b1$ и $b2$ и проходит через точки p ($p \in b1$ или $p \in b2$).
Г12	Окружность c касается окружности a , прямой b и проходит через точки p .

Г13	Окружность c касается окружности a , прямой b и проходит через точки p ($p \in b$).
Г14	Окружность c касается окружности a , прямой b и проходит через точки p ($p \in a$).
Г15	Окружность c касается окружности a_1, a_2 и проходит через точки p .
Г16	Окружность c касается окружности a_1, a_2 и проходит через точки p ($p \in a_1$ или $p \in a_2$).
Г17	Окружность c касается прямой b , ТЗК s и проходит через точки p .
Г18	Окружность c касается прямой b , ТЗК s и проходит через точки p ($p \in b$).
Г19	Окружность c касается прямой b , ТЗК s и проходит через точки p ($p \in s$).
Г20	Окружность c касается окружности a , ТЗК s и проходит через точки p .
Г21	Окружность c касается окружности a , ТЗК s и проходит через точки p ($p \in a$).
Г22	Окружность c касается окружности a , ТЗК s и проходит через точки p ($p \in s$).
Г23	Окружность c касается ТЗК s и ТЗК v и проходит через точки p .
Г24	Окружность c касается ТЗК s и ТЗК v и проходит через точки p . ($p \in s_1$ или $p \in s_2$).
Г25	ТЗК s касается прямых b_1 и b_2 .
Г26	ТЗК s касается прямой b и окружности a .
Г27	ТЗК s касается окружностей a_1 и a_2 .
Г28	ТЗК w касается 2-х ТЗК s и w .

Определение ГЭЛ через операции АК или дополнительные построения

Таблица 5.5

N Задачи	Содержание типа задания
Д1	Задание отрезка (длины s) прямой от заданной точки A на заданной прямой.
Д2	Задание дуги (длины s) от заданной точки A на заданной окружности.
Д3	Определение точки на окружности по длине хорды.
Д4	Определение точки на ТЗК по длине хорды.
Д5	Определение точки на ТЗК от заданной по длине кривой.
Д6	Задание прямой, проходящей через точку параллельно заданной прямой.

Д7	Задание прямой, проходящей через точку перпендикулярно заданной прямой.
Д8	Задание прямой, проходящей через точку под углом к заданной прямой.
Д9	Задание прямой как биссектриссы заданного угла.
Определение ГЭЛ через дополнительные построения	
Д10	Задание прямой, проходящей параллельно заданной прямой на заданном расстоянии.
Д11	Задание нормали к окружности.
Д12	Задание нормали к ТЗК.
Д13	Определение точки p на заданном расстоянии от прямых b_1 и b_2 .
Д14	Определение точки на ТЗК, наиболее удаленной от оси x .
Д15	Определение точки на ТЗК, наиболее удаленной от оси y .
Д16	"Притупить кромку" (срезать угол) отрезком заданной длины.
Д17	Задание отрезка l (длины s) между двумя пересекающимися прямыми ($l \perp b_1$ или $l \perp b_2$)
Д18	Задание отрезка (заданной длины и направления) между двумя пересекающимися прямыми
Д19	Задание отрезка l (длины s) между прямой b и окружностью a ($l \perp b$).
Д20	Задание отрезка (заданной длины и направления) между прямой b и окружностью a .
Д21	Задание дуги (заданной длины) между двумя пересекающимися прямыми.
Д22	Построение сопряжения "Прямая-окружность" при заданных длине отрезка прямой b , радиусе дуги и начал: отрезка - p_1 и дуги окружности - p_2 .
Д23	Определение точки p на окружности по заданному углу от заданной точки p_1 .
Д24	Определение точки p , симметричной заданной точке относительно прямой b .
Д25	Определение прямой b_2 , симметричной заданной прямой b_2 относительно прямой b_3 .
Д26	Определение ТЗК v , симметричной заданной ТЗК s относительно прямой b .
Задачи Аполлония	
Д27	Построение окружности c , касающейся прямых b_1, b_2, b_3 .
Д28	Построение окружности c , касающейся окружностей a_1, a_2, a_3 .
Д29	Построение окружности c , касающейся окружностей a_1, a_2 , и прямой b .
Д30	Построение окружности c , касающейся окружности a , и прямых b_1, b_2 .

Определение (задание, касание, сопряжение) окружности, центр которой лежит на носителе (прямой, окружности)

Таблица 5.6

N задачи	Содержание типа задания
E1	Окружность c радиуса R проходит через точку p . Центр c лежит на прямой b .
E2	Окружность c радиуса R проходит через точку p . Центр c лежит на окружности a .
E3	Окружность c проходит через точки p_1 и p_2 . Центр c лежит на прямой b .
E4	Окружность c проходит через точки p_1 и p_2 . Центр c лежит на окружности a .
E5	Окружность c радиуса R касается прямой b . Центр c лежит на прямой b_1 .
E6	Окружность c радиуса R касается прямой b . Центр c лежит на окружности a .
E7	Окружность c радиуса R касается окружности a . Центр c лежит на прямой b .
E8	Окружность c радиуса R касается окружности a_1 . Центр c лежит на окружности a_2 .
E9	Окружность c проходит через точку p и касается окружности a_1 . Центр c лежит на b .
E10	Окружность c проходит через точку p и касается прямой b_1 . Центр c лежит на b_2 .
E11	Окружность c проходит через точку p и касается прямой b . Центр c лежит на a .
E12	Окружность c проходит через точку p и касается окружности a_1 . Центр c лежит на a_2 .
E13	Окружность c касается прямых b_1 и b_2 . Центр c лежит на прямой b_3 .
E14	Окружность c касается прямых b_1 и b_2 . Центр c лежит на окружности a .
E15	Окружность c касается окружностей a_1, a_2 . Центр c лежит на прямой b .
E16	Окружность c касается прямой b_1 и окружности a . Центр c лежит на прямой b_2 .
E17	Окружность c касается прямой b и окружности a_1 . Центр c лежит на окружности a_2 .
E18	Окружность c касается окружностей a_1, a_2 . Центр c лежит на окружности a_3 .

5.3. Проблемно-ориентированная подсистема "ТЕОРЕТИЧЕСКИЙ ЧЕРТЕЖ" (ТЧ)

5.3.1. Постановка задачи

Задача построения судовой поверхности заключается в том, чтобы для заданной совокупности пространственных линий определить аналитически поверхность, содержащую эти линии.

Введем ряд терминов и обозначений, используемых далее в тексте.

При задании контура используется до трех линий каждого семейства. Образующие линии обозначаются символами u_0, u_c, u_1 , а направляющие - символами v_0, v_c, v_1 . Точки на линиях контура обозначаются соответственно символами $P_{u0}, P_{uc}, P_{u1}, P_{v0}, P_{vc}, P_{v1}$, а координаты этих точек - x_{u0}, y_{u0}, z_{u0} и т.п. Каждая пространственная линия контура задается двумя проекциями на координатные плоскости, причем образующие обычно задаются проекциями на плоскости XU и YZ , а направляющие - на плоскости XU и XZ . Две линии контура считаются согласованными, если они пересекаются. Точки пересечения образующих и направляющих называются узлами контура и обозначаются P_{ij} , где i, j принимают значения $0, c, 1$. Областью определения (поверхности) называется проекция поверхности на плоскость XU , ограниченная проекциями линий контура.

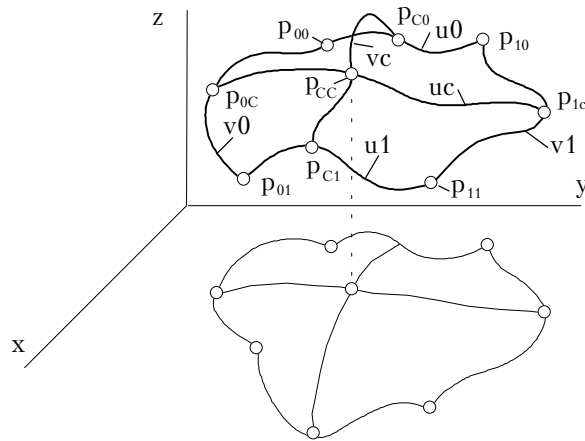


Рис.5.8. Контур параметрической поверхности

В приводимых ниже формулах обозначения типа $z_{u0}(y_{u0})$ указывают на то, что координата z точки линии u_0 задана явной функцией от координаты y , а обозначения типа $z_{u0}(u)$ или просто z_{u0} - на то, что линия u_0 задана параметрически.

Рассматриваются два общих метода построения поверхности:

- параметрическая модель, в которой поверхность задается формулами:

$$\begin{aligned}x &= x(u, v), \\ y &= y(u, v), \\ z &= z(u, v); \end{aligned}$$

- полупараметрическая модель, в которой поверхность задается формулами :

$$\begin{aligned}x &= x(y, v), \\ z &= z(y, v); \end{aligned}$$

и два частных случая полупараметрической модели, позволяющих задать плазовые поверхности (рис.5.9) в явном виде :

$$z = z(x, y),$$

где x, y, z - координаты точки поверхности.

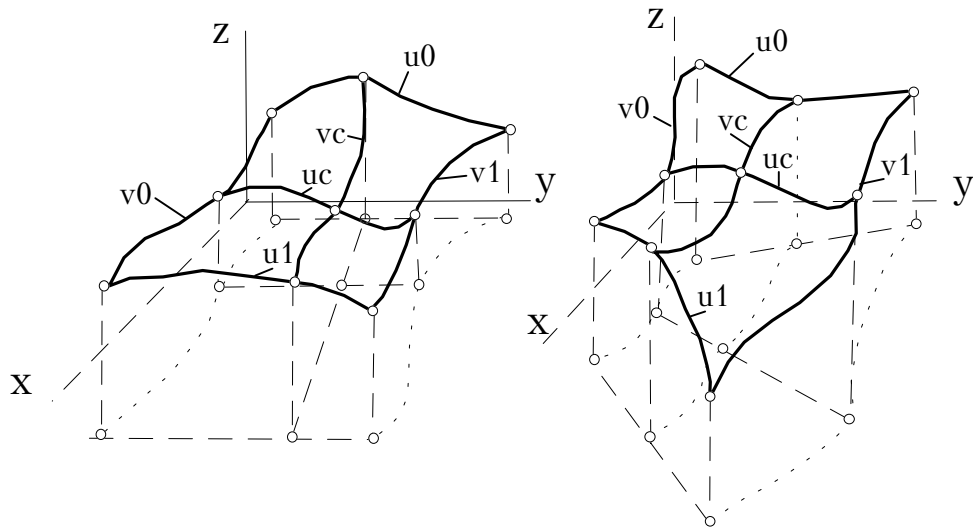


Рис. 5.9. Плазовые поверхности

- а) образующие лежат в плоскостях, параллельных yz ;
- б) образующие лежат в плоскостях, перпендикулярных xy .

Рассматриваемые математические модели различаются способами параметризации области определения. Функциональная зависимость z от входных переменных (u, v ; y, v и x, y) определяется по следующей одинаковой для всех моделей схеме. По входным переменным ищутся

вспомогательные точки на границах области определения; вычисляются значения z -координат линий контура в найденных точках и полученные значения "смешиваются" так, чтобы удовлетворить условию прохождения поверхности через линии контура.

Для определения точек на границах и "смешивания" z -координат используются весовые функции w_0, w_c, w_1 , позволяющие управлять формой поверхности.

Границы области определения необходимо задавать для всех моделей. В то же время Z проекции (XZ или YZ) некоторых линий контура могут отсутствовать. В таких случаях соответствующая линия контура считается не заданной. Разные варианты задания контура будут рассматриваться отдельно.

Для контура, состоящего из четырех линий (u_0, u_1, v_0, v_1) , можно дополнительно задать до четырех неявных управляющих линий (u_0, u_1, v_0, v_1) , XY проекции которых совпадают с XY проекциями соответствующих линий контура, а Z проекции задаются явными функциями от x или y . Эти линии также позволяют управлять формой поверхности.

5.3.2. Математическое представление судовой поверхности

5.3.2.1. Параметрическая модель поверхности

Для данной модели контур может состоять только из линий u_0, u_1, v_0, v_1 либо некоторого их подмножества. Преимущество этой модели по сравнению с полупараметрической моделью состоит в том, что границы области определения (проекции образующих и направляющих на плоскость XY) задаются произвольными параметрическими линиями. Z проекции образующих задаются явными функциями $z_{iu}(y_{ui})$, а Z проекции направляющих - явными функциями $z_{vi}(x_{vi})$, где i принимает значения 0,1. Ниже приводятся формулы параметризации области определения:

$$x(u, v) = x_{u_0} + x_{v_0} - x_{00} + \frac{(x_{v_1} - x_{v_0} - x_{10} + x_{00})(x_{u_1} - x_{u_0} - x_{01} + x_{00})}{x_{11} - x_{01} - x_{10} + x_{00}},$$

$$y(u, v) = y_{u_0} + y_{v_0} - y_{00} + \frac{(y_{v_1} - y_{v_0} - y_{10} + y_{00})(y_{u_1} - y_{u_0} - y_{01} + y_{00})}{y_{11} - y_{01} - y_{10} + y_{00}}.$$

Вспомогательные точки на границах области определения задаются формулами :

$$P_{ui}=P_{u0}(u), \quad P_{ui}=P_{u1}(u), \quad P_{v0}=P_{v0}(v), \quad P_{v1}=P_{v1}(v).$$

Вычисление z координаты точки поверхности производится по формулам п. 5.3.2.4.

5.3.2.2. Полупараметрическая модель поверхности

Для данной модели контур может состоять из линий $u_0, u_c, u_1, v_0, v_c, v_1$ либо некоторого их подмножества. Границы области определения для образующих задаются функциями $x_{ui}(y_{ui})$, для направляющих - функциями $y_{vi}(x_{vi})$. Z проекции образующих задаются функциями $z_{ui}(y_{ui})$, а Z проекции направляющих - функциями $z_{vi}(x_{vi})$, где i принимает значения 0,с,1. Ниже приводятся формулы параметризации области определения в зависимости от варианта задания контура:

1) при отсутствии линии u_c :

$$x(v, y) = w_0 \cdot x_{u0}(y_{u0}) + w_1 \cdot x_{u1}(y_{u1}),$$

где $w_0=1-v$, $w_1=v$;

2) при наличии линии u_c :

$$x(v, y) = w_0 \cdot x_{u0}(y_{u0}) + w_c \cdot x_{uc}(y_{uc}) + w_1 \cdot x_{u1}(y_{u1}),$$

где $w_0=1-3v+2v^2$, $w_c=4v(1-v)$, $w_1=v(2v-1)$.

Вспомогательные точки на границах области определения и на XY проекциях линий u_c и v_c задаются по координатно переменными x_{uj} , y_{uj} , x_{vj} , y_{vj} :

1а) при отсутствии линий u_c и v_c :

$$x_{vi} = w_0 \cdot x_{i0} + w_1 \cdot x_{i1},$$

$$y_{uj} = \frac{(y - y_{v1}(x_{v1})) \cdot (y_{1j} - y_{0i})}{y_{1v}(x_{1v}) - y_{0v}(x_{0v})} + y_{1j}$$

для $i, j=0,1$;

1б) при отсутствии линии v_c :

$$x_{vi} = \omega_0 \cdot x_{i0} + \omega_c \cdot x_{ic} + \omega_1 \cdot x_{i1},$$

$$y_{uj} = \frac{(y - y_{v1}(x_{v1})) \cdot (y_{1j} - y_{0i})}{y_{1v}(x_{1v}) - y_{0v}(x_{0v})} + y_{1j}$$

для $i=0, c, 1$; $j=0, 1$;

2а) при отсутствии линии u_c :

$$x_{vi} = \omega_0 \cdot x_{i0} + \omega_1 \cdot x_{i1},$$

$$y_{uj} = (1 - \omega_y) \cdot A_{0j} + \omega_y \cdot A_{cj},$$

где

$$\omega_y = \frac{y - y_{v0}(x_{v0})}{y_{v1}(x_{v1}) - y_{v0}(x_{v0})},$$

$$A_{0j} = \frac{(y - y_{vc}(x_{vc})) \cdot (y_{ci} - y_{0j})}{y_{vc}(x_{vc}) - y_{v0}(x_{v0})} + y_{cj},$$

$$A_{cj} = \frac{(y - y_{v1}(x_{v1})) \cdot (y_{1i} - y_{cj})}{y_{v1}(x_{v1}) - y_{vc}(x_{vc})} + y_{1j}$$

для $i=0, 1$; $j=0, c, 1$;

2б) при наличии линий u_c и v_c :

$$x_{vi} = \omega_0 \cdot x_{i0} + \omega_c \cdot x_{ic} + \omega_1 \cdot x_{i1},$$

$$y_{uj} = (1 - \omega_y) \cdot A_{0j} + \omega_y \cdot A_{cj},$$

где

$$\omega_y = \frac{y - y_{v0}(x_{v0})}{y_{v1}(x_{v1}) - y_{v0}(x_{v0})},$$

$$A_{0j} = \frac{(y - y_{vc}(x_{vc})) \cdot (y_{ci} - y_{0j})}{y_{vc}(x_{vc}) - y_{v0}(x_{v0})} + y_{cj},$$

$$A_{cj} = \frac{(y - y_{v1}(x_{v1})) \cdot (y_{1i} - y_{cj})}{y_{v1}(x_{v1}) - y_{vc}(x_{vc})} + y_{1j}$$

для $i=0, c, 1$; $j=0, c, 1$.

Вычисление z координаты точки поверхности производится по формулам п. 5.3.2.4.

5.3.2.3. Частные случаи

В рассмотренных ниже частных случаях задания контура поверхность задается как явная функция $z=z(x,y)$, что позволяет существенно ускорить процесс вычисления.

Для первого частного случая необходимо, чтобы образующие были параллельны плоскости YZ . Контур может состоять из линий $u_0, u_c, u_1, v_0, v_c, v_1$ либо некоторого их подмножества. При этом значение x координат вспомогательных точек на направляющих полагается равным $x(x_{v_i}=x)$, а весовые функции вычисляются по формулам:

- при отсутствии линии u_c -

$$w_0 = 1 - \frac{x - x_{00}}{x_{01} - x_{00}}, \quad w_1 = \frac{x - x_{00}}{x_{01} - x_{00}},$$

- при наличии линии u_c -

$$w_0 = \frac{(x - x_{c0})(x - x_{01})}{(x_{00} - x_{c0})(x_{00} - x_{01})}, \quad w_c = \frac{(x - x_{00})(x - x_{01})}{(x_{0c} - x_{00})(x_{0c} - x_{01})},$$

$$w_1 = \frac{(x - x_{00})(x - x_{01})}{(x_{01} - x_{c0})(x_{01} - x_{c0})}.$$

Остальные переменные вычисляются так же, как и в п. 5.3.2.1.

Для второго частного случая необходимо, чтобы образующие находились в плоскости ортогональной плоскости XU , причем контур может состоять из линий u_0, u_1, v_0, v_1 либо некоторого их подмножества. Вспомогательные точки на направляющих определяются как точки пересечения соответствующих направляющих с прямой, проходящей через точку пересечения проекций линий u_0 и u_1 на плоскость XU и точку с координатами x, y . Параметр v является линейной функцией угла между осью Y и определенной выше прямой. Остальные переменные вычисляются так же, как и в п. 5.3.2.1.

5.3.2.4. Определение z координаты точки поверхности

Для различных вариантов задания контура z координата точки поверхности определяется разными способами. Ниже представлены формулы для каждого допустимого варианта. Предполагается, что все вспомогательные переменные определены. Используется дополнительная вспомогательная переменная : $D_y = y_{v1}(x_{v1}) - y_{v0}(x_{v0})$

Заданы линии u_0, u_1, v_0, v_1

$$z = \omega_0 \cdot (T_0 + z_{u0}(y_{u0})) + \omega_1 \cdot (T_1 + z_{u1}(y_{u1})),$$

где

$$T_0 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{00})}{D_y}),$$

$$T_1 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_y}).$$

Заданы линии u_0, v_0, v_1

$$z = T_0 + z_{u0}(y_{u0}),$$

где

$$T_0 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{00})}{D_y}).$$

Заданы линии u_1, v_0, v_1

$$z = (T_1 + z_{u1}(y_{u1})),$$

где

$$T_1 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_y}).$$

Заданы линии u_0, u_1, v_0

$$z = \omega_0 \cdot (z_{v0}(x_{v0}) - z_{00} + z_{u0}(y_{u0})) + \omega_1 \cdot (z_{v0}(x_{v0}) - z_{01} + z_{u1}(y_{u1})).$$

Заданы линии u_0, u_1, v_1

$$z = \omega_0 \cdot (z_{v1}(x_{v1}) - z_{10} + z_{u0}(y_{u0})) + \omega_1 \cdot (z_{v1}(x_{v1}) - z_{11} + z_{u1}(y_{u1})).$$

Заданы линии v_0, v_1

$$z = (1 - \omega_y) \cdot z_{v_0}(x_{v_0}) + \omega_y \cdot z_{v_1}(x_{v_1}).$$

где

$$\omega_y = (y - y_{v_0}(x_{v_0})) / D_y.$$

Заданы линии u_0, u_1

$$z = \omega_0 \cdot z_{u_0}(y_{u_0}) + \omega_1 \cdot z_{u_1}(y_{u_1}).$$

Заданы линии u_0, v_0

$$z = z_{v_0}(x_{v_0}) - z_{u_0} + z_{u_0}(y_{u_0}).$$

Заданы линии u_1, v_0

$$z = z_{v_0}(x_{v_0}) - z_{u_1} + z_{u_1}(y_{u_1}).$$

Заданы линии u_0, v_1

$$z = z_{v_1}(x_{v_1}) - z_{u_0} + z_{u_0}(y_{u_0}).$$

Заданы линии u_1, v_1

$$z = z_{v_1}(x_{v_1}) - z_{u_1} + z_{u_1}(y_{u_1}).$$

Задана линия v_0

$$z = z_{v_0}(x_{v_0}).$$

Задана линия v_1

$$z = z_{v_1}(x_{v_1}).$$

Задана линия u_0

$$z = z_{u0}(y_{u0}).$$

Задана линия u_1

$$z = z_{u1}(y_{u1}).$$

Заданы линии u_0, u_c, u_1, v_0, v_1

$$z = \omega_0 \cdot (T_0 + z_{u0}(y_{u0})) + \omega_c \cdot (T_c + z_{uc}(y_{uc})) + \omega_1 \cdot (T_1 + z_{u1}(y_{u1})),$$

где

$$T_0 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{00})}{D_y}),$$

$$T_1 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_y}),$$

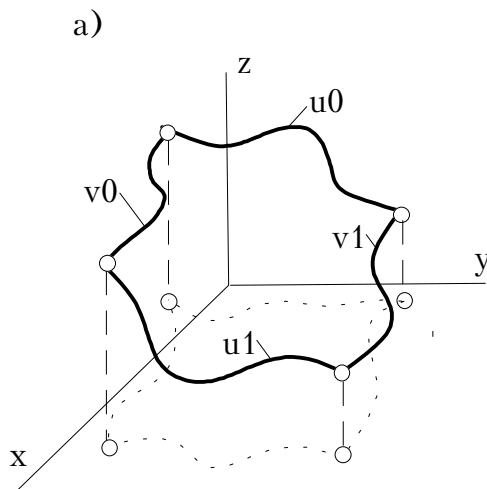
$$T_c = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{c1}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{c0})}{D_y}).$$

Заданы линии u_c, v_0, v_1

$$z = (T_c + z_{uc}(y_{uc})),$$

где

$$T_c = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{c1}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{c0})}{D_y}).$$



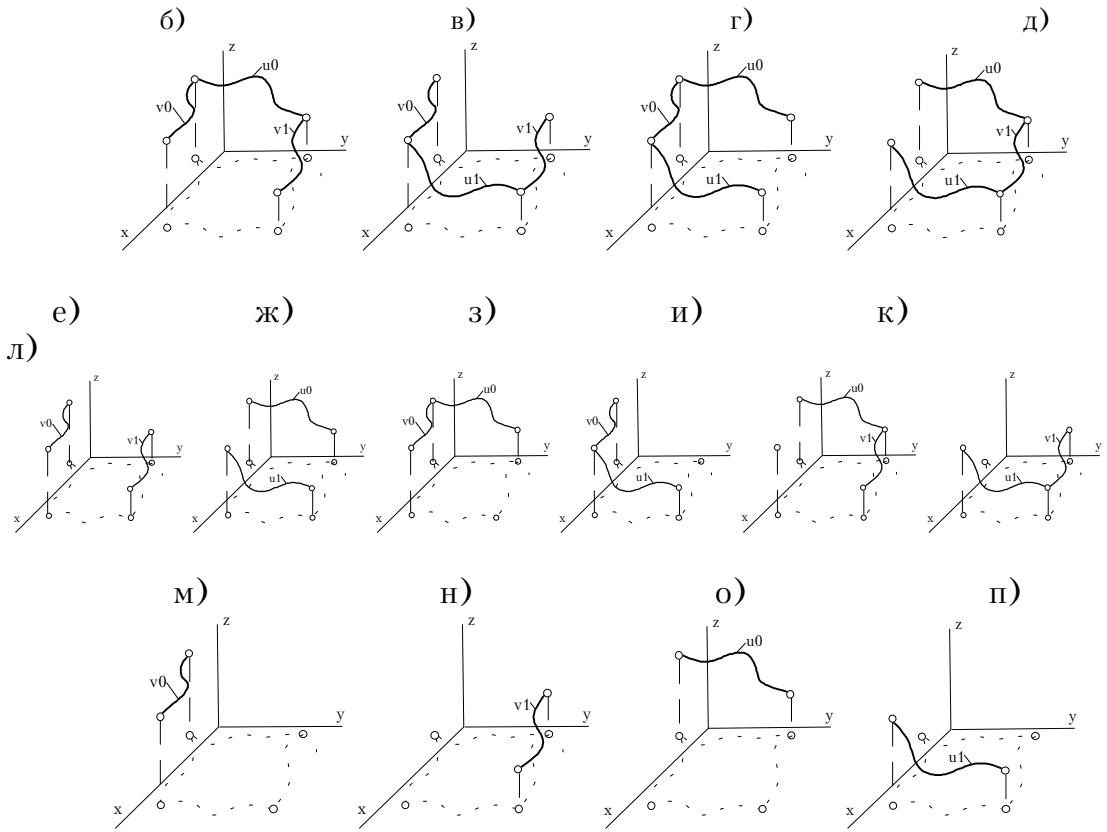


Рис.5.10. Задание поверхностей при различных сочетаниях линий на контуре из 4-х кривых

Заданы линии u_0, u_c, u_1

$$z = \omega_0 \cdot z_{u_0}(y_{u_0}) + \omega_c \cdot z_{u_c}(y_{u_c}) + \omega_1 \cdot z_{u_1}(y_{u_1}).$$

Заданы линия u_c

$$z = z_{u_c}(y_{u_c}).$$

Заданы линии u_0, u_1, v_0, v_c, v_1

$$z = \omega_0 \cdot (T_0 + z_{u_0}(y_{u_0})) + \omega_1 \cdot (T_1 + z_{u_1}(y_{u_1})),$$

где

$$T_0 = (1 - \omega_y) \cdot T_{00} + \omega_y \cdot T_{01},$$

$$T_1 = (1 - \omega_y) \cdot T_{10} + \omega_y \cdot T_{11},$$

$$\omega_y = (y - y_{v_0}(x_{v_0})) / D_y.$$

$$\begin{aligned}
T_{00} &= ((y - y_{v_0}(x_{v_0})) \cdot (z_{v_c}(x_{v_c}) - z_{c_0}) - \frac{(y - y_{v_c}(x_{v_c})) \cdot (z_{v_0}(x_{v_0}) - z_{00})}{D_{y_0}}), \\
T_{01} &= ((y - y_{v_0}(x_{v_0})) \cdot (z_{v_1}(x_{v_1}) - z_{11}) - \frac{(y - y_{v_1}(x_{v_1})) \cdot (z_{v_0}(x_{v_0}) - z_{01})}{D_{y_c}}), \\
T_{10} &= ((y - y_{v_0}(x_{v_0})) \cdot (z_{v_1}(x_{v_1}) - z_{c_1}) - \frac{(y - y_{v_1}(x_{v_1})) \cdot (z_{v_0}(x_{v_0}) - z_{0c})}{D_{y_0}}), \\
T_{11} &= ((y - y_{v_0}(x_{v_0})) \cdot (z_{v_1}(x_{v_1}) - z_{10}) - \frac{(y - y_{v_1}(x_{v_1})) \cdot (z_{v_0}(x_{v_0}) - z_{00})}{D_{y_c}}), \\
D_{y_0} &= y_{v_c}(x_{v_c}) - y_{v_0}(x_{v_0}), \\
D_{y_c} &= y_{v_1}(x_{v_1}) - y_{v_c}(x_{v_c}).
\end{aligned}$$

Заданы линии $u_0, u_c, u_1, v_0, v_c, v_1$

$$z = \omega_0 \cdot (T_0 + z_{u_0}(y_{u_0})) + \omega_c \cdot (T_c + z_{u_c}(y_{u_c})) + \omega_1 \cdot (T_1 + z_{u_1}(y_{u_1})),$$

где

$$\begin{aligned}
T_0 &= (1 - \omega_y) \cdot T_{00} + \omega_y \cdot T_{01}, \\
T_1 &= (1 - \omega_y) \cdot T_{10} + \omega_y \cdot T_{11}, \\
T_c &= (1 - \omega_y) \cdot T_{c0} + \omega_y \cdot T_{c1}, \\
T_{00} &= ((y - y_{v_0}(x_{v_0})) \cdot (z_{v_c}(x_{v_c}) - z_{c_0}) - \frac{(y - y_{v_c}(x_{v_c})) \cdot (z_{v_0}(x_{v_0}) - z_{00})}{D_{y_0}}), \\
T_{01} &= ((y - y_{v_0}(x_{v_0})) \cdot (z_{v_1}(x_{v_1}) - z_{11}) - \frac{(y - y_{v_1}(x_{v_1})) \cdot (z_{v_0}(x_{v_0}) - z_{01})}{D_{y_c}}), \\
T_{10} &= ((y - y_{v_0}(x_{v_0})) \cdot (z_{v_1}(x_{v_1}) - z_{c_1}) - \frac{(y - y_{v_1}(x_{v_1})) \cdot (z_{v_0}(x_{v_0}) - z_{0c})}{D_{y_0}}), \\
T_{11} &= ((y - y_{v_0}(x_{v_0})) \cdot (z_{v_1}(x_{v_1}) - z_{10}) - \frac{(y - y_{v_1}(x_{v_1})) \cdot (z_{v_0}(x_{v_0}) - z_{00})}{D_{y_c}}), \\
T_{c0} &= ((y - y_{v_0}(x_{v_0})) \cdot (z_{v_1}(x_{v_1}) - z_{11}) - \frac{(y - y_{v_1}(x_{v_1})) \cdot (z_{v_0}(x_{v_0}) - z_{01})}{D_{y_0}}), \\
T_{c1} &= ((y - y_{v_0}(x_{v_0})) \cdot (z_{v_1}(x_{v_1}) - z_{1c}) - \frac{(y - y_{v_1}(x_{v_1})) \cdot (z_{v_0}(x_{v_0}) - z_{0c})}{D_{y_c}}), \\
D_{y_0} &= y_{v_c}(x_{v_c}) - y_{v_0}(x_{v_0}), \\
D_{y_c} &= y_{v_1}(x_{v_1}) - y_{v_c}(x_{v_c}).
\end{aligned}$$

a)

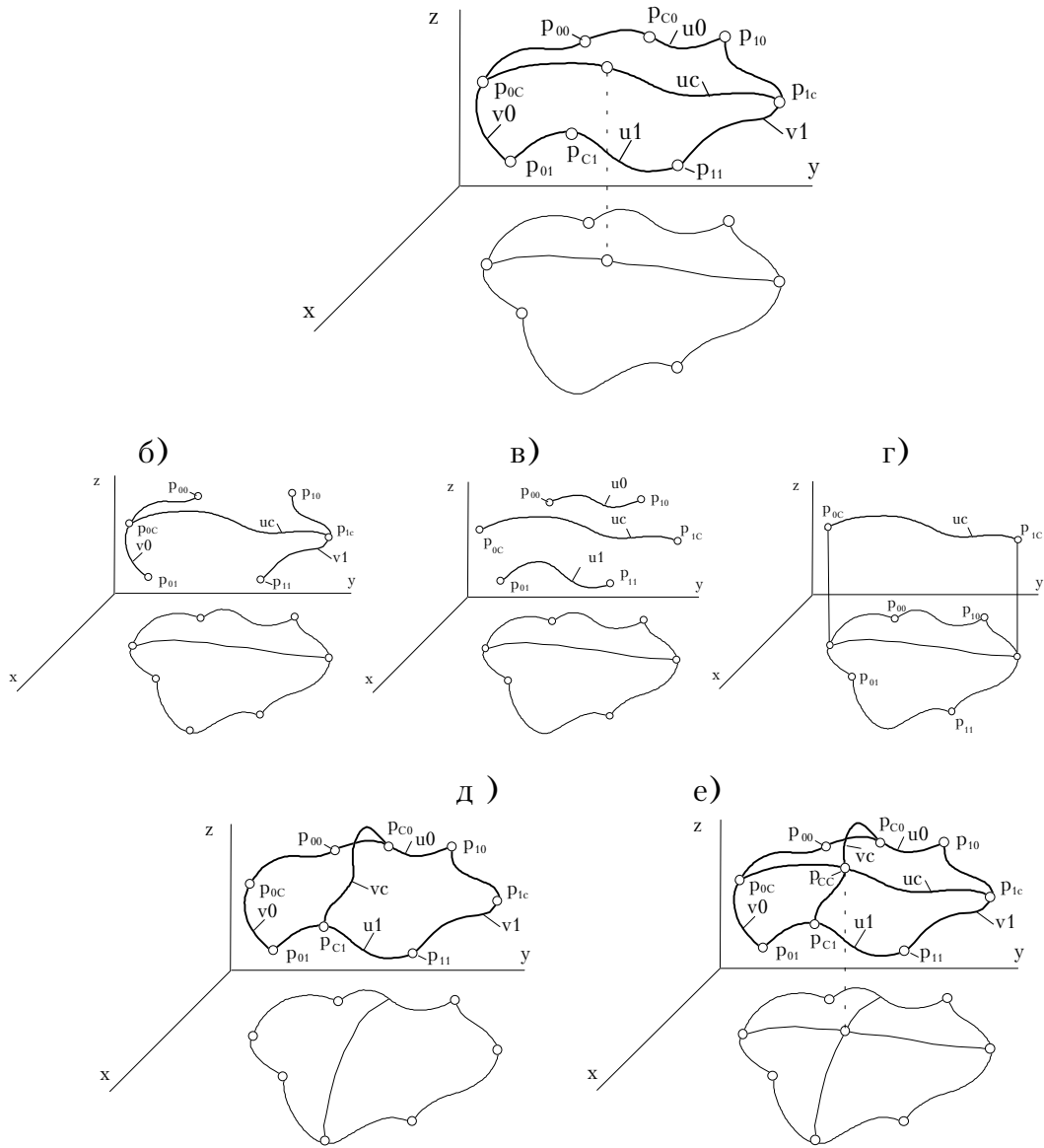


Рис.5.11. Явные управляющие функции (линии) поверхности

Заданы линии $u_0, u_1, v_0, v_1, r_{u0}$

$$z = \omega_0 \cdot [\omega_0 \cdot (T_0 + z_{u0}(y_{u0})) + \omega_1 \cdot (T_0 + z_{ru0}(y_{u0}))] + \omega_1 \cdot z_{u1}(y_{u1}),$$

где

$$T_0 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{00})}{D_y}),$$

$$T_1 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_y}).$$

Заданы линии $u_0, u_1, v_0, v_1, r_{u1}$

$$z = \omega_1 \cdot [\omega_1 \cdot (T_0 + z_{u1}(y_{u1})) + \omega_0 \cdot (T_1 + z_{ru1}(y_{u1}))] + \omega_0 \cdot z_{u0}(y_{u0}),$$

где

$$T_0 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{00})}{D_y}),$$

$$T_1 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_y}).$$

Заданы линии $u_0, u_1, v_0, v_1, r_{u0}, r_{u1}$

$$z = \omega_0 \cdot [\omega_0 \cdot (T_0 + z_{u0}(y_{u0})) + \omega_1 \cdot (T_0 + z_{ru0}(y_{u0}))] +$$

$$\omega_1 \cdot [\omega_1 \cdot (T_1 + z_{u1}(y_{u1})) + \omega_0 \cdot (T_1 + z_{ru1}(y_{u1}))],$$

где

$$T_0 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{00})}{D_y}),$$

$$T_1 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_y}).$$

Заданы линии $u_0, u_1, v_0, v_1, r_{v0}$

$$z = (1 - \omega_y) \cdot [\omega_0 \cdot z_{00} + \omega_1 \cdot z_{10}] + \omega_y \cdot [\omega_0 \cdot z_0 + \omega_1 \cdot z_1],$$

где

$$\omega_y = (y - y_{v0}(x_{v0})) / D_y,$$

$$z_{00} = T_{00} + z_{u0}(y_{u0}),$$

$$z_{10} = T_{10} + z_{u1}(y_{u1}),$$

$$z_0 = T_0 + z_{u0}(y_{u0}),$$

$$z_1 = T_1 + z_{u1}(y_{u1}),$$

$$T_0 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{00})}{D_y}),$$

$$T_1 = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_y}),$$

$$T_{00} = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{rv0}(x_{v0}) - z_{00})}{D_{y0}}),$$

$$T_{10} = ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{rv0}(x_{v0}) - z_{00})}{D_{y0}}).$$

Заданы линии $u_0, u_1, v_0, v_1, r_{v1}$

$$z = (1 - \omega_y) \cdot [\omega_0 \cdot z_0 + \omega_1 \cdot z_1] + \omega_y \cdot [\omega_0 \cdot z_{01} + \omega_1 \cdot z_{11}],$$

где

$$\begin{aligned}
 w_y &= (y - y_{v0}(x_{v0})) / D_y, \\
 z_{01} &= T_{01} + z_{u0}(y_{u0}), \\
 z_{11} &= T_{11} + z_{u1}(y_{u1}), \\
 z_0 &= T_0 + z_{u0}(y_{u0}), \\
 z_1 &= T_1 + z_{u1}(y_{u1}), \\
 T_0 &= ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{00})}{D_y}), \\
 T_1 &= ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_y}), \\
 T_{01} &= ((y - y_{v0}(x_{v0})) \cdot (z_{rv1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_y}), \\
 T_{11} &= ((y - y_{v0}(x_{v0})) \cdot (z_{rv1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_{yc}}).
 \end{aligned}$$

Заданы линии $u_0, u_1, v_0, v_1, r_{v0}, r_{v1}$

$$z = (1 - w_y) \cdot [w_0 \cdot z_{00} + w_1 \cdot z_{10}] + w_y \cdot [w_0 \cdot z_{01} + w_1 \cdot z_{11}],$$

где

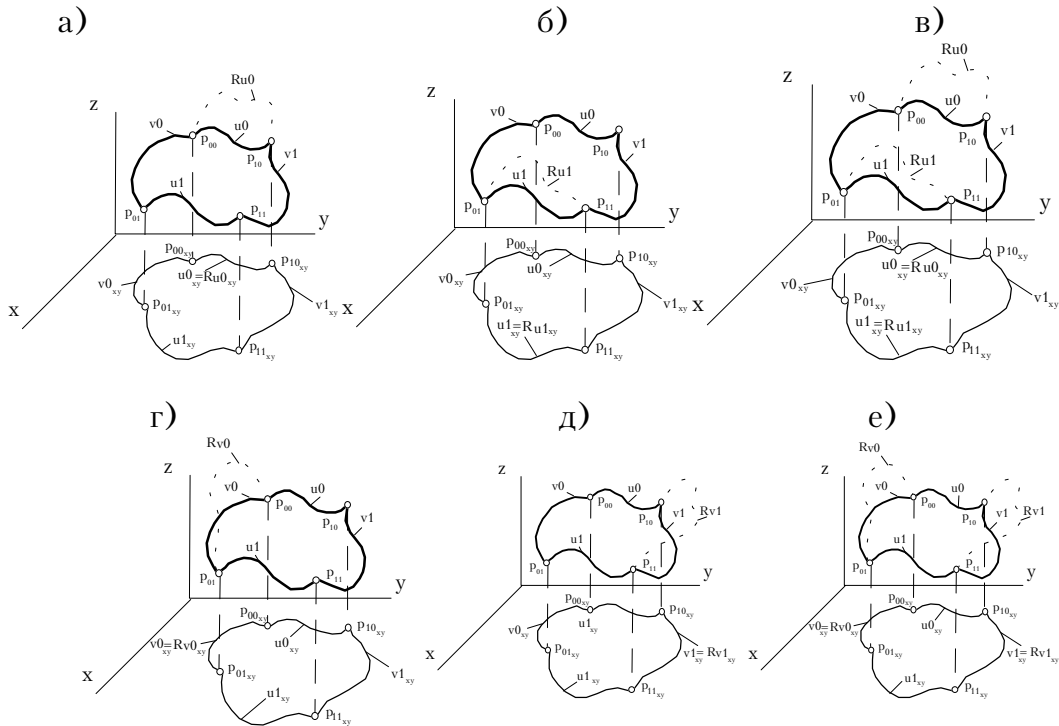
$$\begin{aligned}
 w_y &= (y - y_{v0}(x_{v0})) / D_y, \\
 z_{00} &= T_{00} + z_{u0}(y_{u0}), \\
 z_{10} &= T_{10} + z_{u1}(y_{u1}), \\
 z_{01} &= T_{01} + z_{u0}(y_{u0}), \\
 z_{11} &= T_{11} + z_{u1}(y_{u1}), \\
 T_{00} &= ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{rv0}(x_{v0}) - z_{00})}{D_y}), \\
 T_{01} &= ((y - y_{v0}(x_{v0})) \cdot (z_{rv1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{00})}{D_y}), \\
 T_{10} &= ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{rv0}(x_{v0}) - z_{00})}{D_y}), \\
 T_{11} &= ((y - y_{v0}(x_{v0})) \cdot (z_{rv1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_y}).
 \end{aligned}$$

Заданы линии $u_0, u_1, v_0, v_1, r_{u0}, r_{u1}, r_{v0}, r_{v1}$

$$\begin{aligned}
 z &= w_y \cdot \{(1 - w_y) \cdot [w_0 \cdot (w_0 \cdot z_0 + w_1 \cdot zr_0) + w_1 \cdot (w_1 \cdot z_1 + w_0 \cdot zr_1)] + \\
 &+ w_y \cdot [w_0 \cdot z_{00} + w_1 \cdot z_{11}]\} + (1 - w_y) \cdot (w_0 \cdot z_{01} + w_1 \cdot z_{11}),
 \end{aligned}$$

где

$$\begin{aligned}
 w_y &= (y - y_{v0}(x_{v0})) / D_y, \\
 z00 &= T_{00} + z_{u0}(y_{u0}), \\
 z10 &= T_{10} + z_{u1}(y_{u1}), \\
 z01 &= T_{01} + z_{u0}(y_{u0}), \\
 z11 &= T_{11} + z_{u1}(y_{u1}), \\
 z0 &= T_0 + z_{u0}(y_{u0}), \\
 z1 &= T_1 + z_{u1}(y_{u1}), \\
 zr0 &= T_0 + z_{ru0}(y_{u0}), \\
 zr1 &= T_1 + z_{ru1}(y_{u1}), \\
 T_0 &= ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{00})}{D_y}), \\
 T_1 &= ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_y}), \\
 T_{00} &= ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{rv0}(x_{v0}) - z_{00})}{D_y}), \\
 T_{01} &= ((y - y_{v0}(x_{v0})) \cdot (z_{rv1}(x_{v1}) - z_{10}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{00})}{D_y}), \\
 T_{10} &= ((y - y_{v0}(x_{v0})) \cdot (z_{v1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{rv0}(x_{v0}) - z_{00})}{D_y}), \\
 T_{11} &= ((y - y_{v0}(x_{v0})) \cdot (z_{rv1}(x_{v1}) - z_{11}) - \frac{(y - y_{v1}(x_{v1})) \cdot (z_{v0}(x_{v0}) - z_{01})}{D_y}).
 \end{aligned}$$



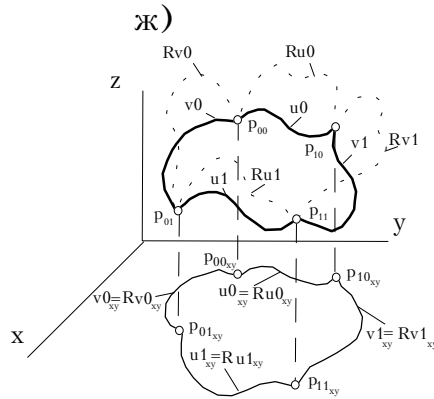


Рис.5.12.

5.3.3. Алгоритмы решения задачи.

Ниже излагаются алгоритмы построения сечений поверхности, полученной по математическим моделям.

Входными данными алгоритмов являются координаты x и y точки поверхности. Необходимо по этим координатам определить координату z поверхности. В частных случаях z вычисляется по формулам п.5.3.2.3 непосредственно. В общем случае для определения координаты z необходимо предварительно решить обратную задачу определения параметра v для полупараметрической модели или параметров u, v для параметрической модели по заданным x и y . Для решения этой задачи используется метод Ньютона решения систем нелинейных уравнений, реализованный программами из "Пакета научных программ на Фортране". Ниже представлены уравнения (системы уравнений) и формулы для вычисления частной производной (матрицы частных производных), необходимые для использования метода Ньютона в частных случаях задания контура в полупараметрической и параметрической моделях.

5.3.3.1. Полупараметрическая модель: контур задан линиями u_0, u_1, v_0, v_1 либо их подмножеством

Уравнение, решаемое относительно v :

$$F(v) = X(v, y) - x = 0,$$

где $X(v, y)$ - определяется по формулам п. 5.3.2.2.

Производная от F по v :

$$\frac{dF(v)}{dv} = \frac{dX(v, y)}{dv} = w_0 \frac{dx_{u0}}{dv} + w_1 \frac{dx_{u1}}{dv} + x_{u1} - x_{u0} ,$$

где

$$\frac{dx_{ui}}{dv} = \frac{Di \frac{dx_{ui}}{dy} \left[(y - y_{v1}) \frac{dy_{v0}}{dv} - (y - y_{v0}) \frac{dy_{v1}}{dv} \right]}{(y_{v1} - y_{v0})(y_{v1} - y_{v0})},$$

$$\frac{dy_{vj}}{dv} = \frac{dy_{vj}}{dx} (x_{j1} - x_{j0}),$$

$$D_i = y_{i1} - y_{0i}$$

для $i=0,1; j=0,1$.

5.3.3.2. Полупараметрическая модель: контур задан линиями u_0, u_1, u_c, v_0, v_1 либо их подмножеством

Уравнение, решаемое относительно v :

$$F(v) = X(v, y) - x = 0 ,$$

где $X(v, y)$ - определяется по формулам п. 5.3.2.2.

Производная от F по v :

$$\frac{dF(v)}{dv} = \frac{dX(v, y)}{dv} = w_0 \frac{dx_{u0}}{dv} + w_c \frac{dx_{uc}}{dv} + w_1 \frac{dx_{u1}}{dv} + x_{uc} \frac{dw_c}{dv} + x_{u1} \frac{dw_1}{dv} ,$$

где

$$\frac{dx_{ui}}{dv} = \frac{Di \frac{dx_{ui}}{dy} \left[(y - y_{v1}) \frac{dy_{v0}}{dv} - (y - y_{v0}) \frac{dy_{v1}}{dv} \right]}{(y_{v1} - y_{v0})(y_{v1} - y_{v0})},$$

$$\frac{dy_{vj}}{dv} = \frac{dy_{vj}}{dx} \left(x_{j0} \frac{dw_0}{dv} + x_{jc} \frac{dw_c}{dv} + x_{j1} \frac{dw_1}{dv} \right),$$

$$\frac{dw_0}{dv} = 4v - 3, \quad \frac{dw_c}{dv} = 4 - 8v, \quad \frac{dw_1}{dv} = 4v - 1,$$

$$D_i = Y_{1i} - Y_{0i}$$

для $i=0,c,1; j=0,1$.

5.3.3.3. Полупараметрическая модель: контур задан линиями u_0, u_1, v_0, v_1, v_c либо их подмножеством

Уравнение, решаемое относительно v :

$$F(v) = X(v, y) - x = 0,$$

где $X(v, y)$ - определяется по формулам п. 5.3.2.2.

Производная от F по v :

$$\frac{dF(v)}{dv} = \frac{dX(v, y)}{dv} = \omega_0 \frac{dx_{u0}}{dv} + \omega_1 \frac{dx_{u1}}{dv} + x_{u1} - x_{u0},$$

где

$$\frac{dx_{ui}}{dv} = \frac{dx_{ui}}{dy} \cdot \left[\frac{d\omega_y}{dv} (A_{ci} - A_{0i}) + (1 - \omega_y) \frac{aA_{0i}}{dv} + \omega_y \frac{dA_{ci}}{dv} \right],$$

$$\frac{dA_{0i}}{dv} = \frac{D_{0i} \left[(y - y_{vc}) \frac{dy_{v0}}{dv} - (y - y_{v0}) \frac{dy_{vc}}{dv} \right]}{(y_{vc} - y_{v0})(y_{vc} - y_{v0})},$$

$$\frac{dA_{ci}}{dv} = \frac{D_{ci} \left[(y - y_{v1}) \frac{dy_{vc}}{dv} - (y - y_{vc}) \frac{dy_{v1}}{dv} \right]}{(y_{v1} - y_{vc})(y_{v1} - y_{vc})},$$

$$\frac{d\omega_y}{dv} = \frac{(y - y_{v1}) \frac{dy_{v0}}{dv} - (y - y_{v0}) \frac{dy_{v1}}{dv}}{(y_{vc} - y_{v0})(y_{vc} - y_{v0})},$$

$$\frac{dy_{vj}}{dv} = \frac{dy_{vj}}{dx} (x_{j1} - x_{j0}),$$

$$D_i = Y_{1i} - Y_{0i}$$

для $i=0,1$; $j=0,c,1$.

5.3.3.4. Полупараметрическая модель: контур задан линиями $u_0, u_1, u_c, v_0, v_1, v_c$ либо их подмножеством

Уравнение, решаемое относительно v :

$$F(v) = X(v, y) - x = 0,$$

где $X(v, y)$ - определяется по формулам п. 5.3.2.2.

Производная от F по v :

$$\frac{dF(v)}{dv} = \frac{dX(v, y)}{dv} = \omega_0 \frac{dx_{u0}}{dv} + \omega_c \frac{dx_{uc}}{dv} + \omega_1 \frac{dx_{u1}}{dv} + x_{u0} \frac{d\omega_0}{dv} + x_{uc} \frac{d\omega_c}{dv} + x_{u1} \frac{d\omega_1}{dv},$$

где

$$\frac{dx_{ui}}{dv} = \frac{dx_{ui}}{dy} \cdot \left[\frac{d\omega_y}{dv} (A_{ci} - A_{0i}) + (1 - \omega_y) \frac{dA_{0i}}{dv} + \omega_y \frac{dA_{ci}}{dv} \right],$$

$$\frac{dA_{0i}}{dv} = \frac{D_{0i} \left[(y - y_{vc}) \frac{dy_{v0}}{dv} - (y - y_{v0}) \frac{dy_{vc}}{dv} \right]}{(y_{vc} - y_{v0})(y_{vc} - y_{v0})},$$

$$\frac{dA_{ci}}{dv} = \frac{D_{ci} \left[(y - y_{v1}) \frac{dy_{vc}}{dv} - (y - y_{vc}) \frac{dy_{v1}}{dv} \right]}{(y_{v1} - y_{vc})(y_{v1} - y_{vc})},$$

$$\frac{d\omega_y}{dv} = \frac{(y - y_{v1}) \frac{dy_{v0}}{dv} - (y - y_{v0}) \frac{dy_{v1}}{dv}}{(y_{vc} - y_{v0})(y_{vc} - y_{v0})},$$

$$\frac{dy_{vj}}{dv} = \frac{dy_{vj}}{dx} \left(x_{j0} \frac{d\omega_0}{dv} + x_{jc} \frac{d\omega_c}{dv} + x_{j1} \frac{d\omega_1}{dv} \right),$$

$$\frac{d\omega_0}{dv} = 4v - 3, \quad \frac{d\omega_c}{dv} = 4 - 8v, \quad \frac{d\omega_1}{dv} = 4v - 1,$$

$$D_{0i} = Y_{ci} - Y_{0i}$$

$$D_{ci} = Y_{1i} - Y_{ci}$$

для $i=0, c, 1; j=0, c, 1$.

5.3.3.5. Параметрическая модель: контур задан линиями u_0, u_1, v_0, v_1 либо их подмножеством

Система уравнений, решаемых относительно u и v :

$$F_1 = X(u, v) - x = 0,$$

$$F_2 = Y(u, v) - y = 0,$$

где $X(v, y)$ - определяется по формулам п. 5.3.2.1.

Матрица частных производных от функций F_1 и F_2 по u и v :

$$\frac{dF_1(u, v)}{du} = \frac{dx(u, v)}{du} = \frac{dx_{u0}}{du} + \frac{(x_{v1} - x_{v0} - x_{10} + x_{00}) \left[\frac{dx_{u1}}{du} - \frac{dx_{u0}}{du} \right]}{x_{11} - x_{01} - x_{10} + x_{00}},$$

$$\frac{dF_1(u,v)}{dv} = \frac{dx(u,v)}{dv} = \frac{dx_{v0}}{dv} + \frac{(x_{u1} - x_{u0} - x_{10} + x_{00}) \left[\frac{dx_{v1}}{dv} - \frac{dx_{v0}}{dv} \right]}{x_{11} - x_{01} - x_{10} + x_{00}},$$

$$\frac{dF_2(u,v)}{du} = \frac{dy(u,v)}{du} = \frac{dy_{u0}}{du} + \frac{(y_{v1} - y_{v0} - y_{10} + y_{00}) \left[\frac{dy_{u1}}{du} - \frac{dy_{u0}}{du} \right]}{y_{11} - y_{01} - y_{10} + y_{00}},$$

$$\frac{dF_2(u,v)}{dv} = \frac{dy(u,v)}{dv} = \frac{dy_{v0}}{dv} + \frac{(y_{u1} - y_{u0} - y_{10} + y_{00}) \left[\frac{dy_{v1}}{dv} - \frac{dy_{v0}}{dv} \right]}{y_{11} - y_{01} - y_{10} + y_{00}},$$

Для вычисления координат точек на линиях контура, заданных параметрически, использовались программы комплекса АК. Для явных функций линий контура также использовались программы "Пакета научных программ на Фортране".

5.3.4. Входные данные

5.3.4.1. Общее описание

Входные данные: задание на отрисовку подготавливаются пользователем на языке описания чертежей во входном текстовом файле STDIN. Разделителем в описании отдельных данных и ключевых слов является пробел. В некоторых случаях в качестве разделителя можно использовать запятую. Допускаются строки - комментарии, которые должны содержать в первой позиции символ \$.

5.3.4.2. Грамматика языка описания "ТН" в форме Бекуса - Наура

<чертежи> ::= ЧЕРТЕЖ <чертеж> | <чертежи> ЧЕРТЕЖ <чертеж>

<чертеж> ::= <рисунок> | <чертеж> <рисунок>

<рисунок> ::= ERROR

| <действие> КОПУС <фрагмент>

| <действие> БОК <фрагмент>

| <действие> ПАРАМ_СЕТКА <фрагмент>

| <аксонометрия> <фрагмент>

| <действие> ЛИНИИ <линии бд>
 <действие> ::= РИС <реж. подп.> <с.к.> | ПЕЧ | СОХР <имя лин.>
 | <аксонометрия>
 <реж. подп.> ::= | ПОДП
 аксонометрия ::= АКС <число> <с.к.>
 <с.к.> ::= | <параметр с.к.>
 | <с.к.> <необяз. запят.> <параметр с.к.>
 <параметр с.к.> ::= <ключ с.к.> = <число> | -X | -Y | XYUX
 <ключ с.к.> ::= ОТ_ФХ | ОТ_ФУ | ШИР | ВЫС | ШАГ | ОТ_X | ОТ_Y
 | ДО_X | ДО_Y | МАСШ
 <фрагмент> ::= <параметр фраг.> | <фрагмент> <параметр фраг.>
 <параметр фраг.> ::= <сетка> | <диап. сеч.>
 | <гран. 0> | <гран. 1>
 | <тип параметризации> | <контур>
 <сетка> ::= НА_СЕТКЕ <число> * <число>
 <диап. сеч.> ::= ПЕР <число> ПОС <число>
 <гран. 0> ::= ГРАН_0 <чис. или имя>
 <гран. 1> ::= ГРАН_1 <чис. или имя>
 <тип параметризации> ::= ПОЛУПАРАМ_МОДЕЛЬ ПАРАМ_МОДЕЛЬ
 <контур> ::= ДЛЯ_КОНТУРА <узлы> <линии>
 <узлы> ::= | <узел> | <узлы> <узел>
 <узел> ::= P00 = <число> <необяз. запят.> <число>
 | P10 = <число> <необяз. запят.> <число>
 | P01 = <число> <необяз. запят.> <число>
 | P11 = <число> <необяз. запят.> <число>
 | PC0 = <число> <необяз. запят.> <число>
 | PC1 = <число> <необяз. запят.> <число>
 | P0C = <число> <необяз. запят.> <число>
 | P1C = <число> <необяз. запят.> <число>
 | PCC = <число> <необяз. запят.> <число>
 <линии> ::= <линия> | <линии> <линия>
 <линия> ::= ОБР <индекс> <проекции обр.>
 | НАП <индекс> <проекции нап.>
 | УПР ОБР <индекс2> Z(Y) = <чис. или имя>
 | УПР НАП <индекс2> Z(X) = <чис. или имя>
 <проекции обр.> ::= <проекция обр.>

```

| <проекция обр.> <проекция обр.>
| ПЛОСКАЯ <проекция обр. лин.>
<проекция обр.> ::= X(Y) = <чис. или имя>
| Z(Y) = <чис. или имя>
| <имя лин.>
<проекция обр. лин.> ::= X(Y) = <имя лин.> | Z(Y) = <имя лин.>
<проекции нап.> ::= <проекция нап.>
| <проекция нап.> <проекция нап.>
| ПЛОСКАЯ <проекция нап. лин.>
<проекция нап.> ::= Y(X) = <чис. или имя>
| Z(X) = <чис. или имя>
| <имя лин.>
<проекция нап. лин.> ::= Y(X) = <имя лин.> | Z(X) = <имя лин.>
<чис. или имя> ::= <число> | <имя лин.>
<индекс> ::= <индекс2> | C:
<индекс2> ::= 0: | 1:
<имя лин.> ::= LXLNTYPE LXINT
<число> ::= LXFLOATP | LXINT
<линии бд> ::= <список линий> | <линии бд> <список линий>
<список линий> ::= LXLNTYPE <номера линий>
<номера линий> ::= LXINT
| <номера линий> <необяз. запят.> LXINT
<необяз. запят.> ::= | ,

```

В представленной выше форме ключевые слова и обобщенные лексемы напечатаны большими буквами. Используются следующие обобщенные лексемы:

ERROR - ошибочная лексема;

LXLNTYPE - тип линии БДАК - два произвольных символа;

LXINT - целое число;

LXFLOATP - вещественное число (с точкой).

Входной файл состоит из описания одного или нескольких чертежей теоретического чертежа судна. В начале описания каждого чертежа указывается ключевое слово ЧЕРТЕЖ. Каждый чертеж состоит из одного или нескольких описаний рисунков.

В описании рисунка указываются: действие, которое необходимо выполнить для фрагмента поверхности либо для линий БДАК; тип се-

чений поверхности (ключевые слова: КОРПУС, БОК), либо задание на отрисовку линий БДАК (ключевое слово ЛИНИИ); описание фрагмента либо линий.

Ключевое слово ПАРАМ_СЕТКА задает отрисовку проекций параметрической поверхности в плоскости XY, что используется при отладке модели поверхности.

Возможны следующие действия:

- рисовать ортогональные проекции сечений поверхности либо линии БДАК (ключевое слово РИС);
- рисовать аксонометрические проекции сечений поверхности либо линий БДАК (ключевое слово АКС);
- печатать значения координат сечений поверхности в текстовый файл в формате файлов типа CARD системы FORAN (ключевое слово ПЕЧ);
- сохранять сечения поверхности в БДАК (ключевое слово СОХР).

Для отрисовки ортогональных проекций указывается (опционно) режим подписи сечений ключевым словом ПОДП. В режиме подписи на рисунок выводятся значения каждого сечения поверхности.

Для любой отрисовки в описании рисунка задается система координат рисунка по отношению к системе координат устройства отображения.

Тип аксонометрической проекции задается углом между осью X на листе и осью X в пространстве (<число> после ключевого слова АКС).

Сохранение сечений поверхности производится в виде линий БДАК. Первое сечение сохраняется в линии с именем <имя лин.> (после ключевого слова СОХР); для следующих сечений имена линий имеют тот же тип, а номера их последовательно увеличиваются на 1.

Заданные значения большинства параметров сохраняются для последующих рисунков либо чертежей. Исключения из этого правила будут отмечены особо.

5.3.4.3. Параметры системы координат

Эти параметры позволяют "привязать" рисунок к полю отображения (листу бумаги либо экрану дисплея) и задать масштаб рисунка .

Вводится понятие физического окна рисунка (прямоугольника в плоскости "поле изображения"), задаваемое параметрами: $OT_ФХ$, $OT_ФУ$ - левый нижний угол и $ШИР$ и $ВЫС$ - соответственно, ширина и высота окна. Это окно связывается с логическим окном рисунка - прямоугольником в плоскости сечений поверхности либо в плоскости линий БДАК. Логическое окно задается параметрами: OT_X , OT_Y - левый нижний угол и $ДО_X$, $ДО_Y$ - правый верхний угол. При этом точка $OT_ФХ$, $OT_ФУ$ соответствует точке $OT_ФХ$, $OT_ФУ$.

Масштаб рисунка можно задавать одним из следующих способов:

1) непосредственное задание масштаба - необходимо задать параметр $МАСШ$;

2) задание масштаба по x - диапазону - необходимо задать параметр $ДО_X$ - в этом случае правая граница физического окна будет соответствовать правой границе логического окна;

3) задание масштаба по Y - диапазону - необходимо задать параметр $ДО_Y$ - в этом случае верхняя граница физического окна будет соответствовать верхней границе логического окна;

4) для отрисовки линий БДАК автоматическое задание системы координат при отсутствии параметров OT_X , OT_Y - в этом случае границы логического окна устанавливаются из габаритов первой из линий БДАК, входящих в рисунок, так, чтобы линия полностью вписывалась в физическое окно либо по X , либо по Y - диапазону;

5) сохранение значения масштаба, заданного в предыдущем рисунке при отсутствии параметров $МАСШ$, $ДО_X$, $ДО_Y$.

Параметры системы координат, задающиеся ключевыми словами - X , $-Y$, $XYYX$, применяются соответственно для обращения направления X - оси координат логического окна, Y - оси координат и для перемены X и Y осей.

Параметр $ШАГ$ позволяет задать плотность отрисовки отличных от прямых сегментов линий БДАК. Значение шага в мм определяет расстояние между соседними точками при отрисовке. При отрисовке на графопостроителе ЕС 7907 сегменты типа дуг окружности отрисовываются аппаратным способом.

Значения параметров OT_X , OT_Y , $ДО_X$, $ДО_Y$, $МАСШ$ не сохраняются при переходе к следующим рисункам либо чертежам.

Следует отметить, что изображение, выходящее за пределы физического окна, также отрисовывается (отсечение не производится).

5.3.4.4. Параметры фрагмента поверхности

В эту группу входят параметры, определяющие:

- сетку, на которой отрисовывается либо распечатывается сечение поверхности;
- диапазон сечений поверхности;
- границы сечений поверхности;
- тип параметризации;
- контур из формообразующих линий поверхности.

Сечение поверхности отрисовывается либо распечатывается на прямоугольной сетке, жестко связанной с системой координат поверхности, так, что существуют линии сетки, обязательно проходящие через оси координат. Шаги сетки по оси X и по оси Y задаются параметрами: НА_СЕТКЕ < шаг по X > * < шаг по Y >. По умолчанию < шаг по X > = < шаг по Y > = 10.

Отрисовываются либо распечатываются все сечения поверхности на линиях сетки, входящих в диапазон сечений, который задается параметрами ПЕР < число > и ПОС < число >, а также первое и последнее сечения, если они не совпадают с линиями сетки. По умолчанию первое сечение = последнему сечению = 0.

Для каждого сечения определяются значения координаты z поверхности на узлах сетки в диапазоне от границы 0 до границы 1. Эти значения распечатываются либо линейно интерполируются и отрисовываются.

Границы сечений задаются параметрами ГРАН_0 < число или имя линии > и ГРАН_1 < число или имя линии >. В случае задания границы линией БДАК границы вычисляются из параметров линии. Если границы не заданы, то они определяются из линий контура. Значения параметров границ не сохраняются при переходе к следующему рисунку либо чертежу.

Для контура, состоящего из четырех либо меньшего числа линий (U0, U1, 0V, 1V), параметр <тип параметризации> определяет математическую модель поверхности - параметрическую или полупараметри-

ческую. По умолчанию строится полупараметрическая модель поверхности.

Для задания параметров контура необходимо указать ключевое слово ДЛЯ_КОНТУРА. Формообразующие линии, составляющие контур, являются в общем случае пространственными линиями и задаются двумя проекциями на координатных плоскости. Вводится понятие области определения контура - это проекция контура на плоскость ХУ.

Узлы - это точки пересечения линий области определения контура. Узлы задаются в случае, когда невозможно определить их автоматически.

Форма задания узлов:

<обозначение узла>=<x - координата> <необяз. запятая> <y- координата>.

Контур может содержать произвольное число (до трех) образующих линий и произвольное число (до трех) направляющих линий, а также до двух неявных управляющих образующих линий и до двух неявных направляющих линий. Для образующих используется ключевое слово ОБР, для направляющих - НАП, для неявных образующих - УПР ОБР, для неявных направляющих - УПР НАП.

Индекс линии определяет местоположение линии в контуре и принимает значения: 0:, С:, 1:, где 0: - означает нулевая линия, С: - средняя и 1: - первая.

Проекции линии определяются ключевыми словами, задающими плоскость проекции, например, запись" ОБР 1: Z(Y)=" означает, что определяется проекция на плоскость ZY первой образующей. В качестве проекций можно указывать число (константу) либо имя линии БДАК. Можно также задавать формообразующие линии, указав ключевое слово ПЛОСКАЯ и имя линии в БДАК для одной из проекций. В этом случае вторая проекция считывается константой, а значение ее выбирается из параметров плоскости линии БДАК.

Необходимо отметить, что область определения контура должна задаваться, как минимум, четырьмя ХУ проекциями, т. е. должны быть заданы проекции :

ОБР 0: X(Y) = ...

ОБР 1: X(Y) = ...

НАП 0: Y(X) = ...

НАП 1: Y(X) =

5.3.4.5. Параметры линий БДАК

Для отрисовки линий БДАК указываются их имена. Если линии имеют одинаковый тип, их имена можно задавать в списочной форме, например : БТ 1,2,3,108,9,4.

5.3.5. Выходные данные

Выходными данными программы являются изображения, полученные на экране дисплея, либо временные файлы изображений , предназначенные для автономной отрисовки на графопостроителе, сечения поверхности, сохраненные в виде линий БДАК, и выходной текстовый файл, содержащий координаты сечений поверхности в формате файла типа CARD системы FORAN.

Сообщения содержат:

- строки, дублирующие строки текста входного файла,
- информационные сообщения,
- предупредительные сообщения,
- сообщения об ошибках.

5.3.5.1. Информационные сообщения

К информационным сообщениям относятся сообщения о способах задания масштаба:

- "непосредственное задание масштаба",
- "задание масштаба по x-диапазону",
- "задание масштаба по y-диапазону",
- "масштаб не изменяется ",
- "автоматическое задание с. к."

5.3.5.2. Предупредительные сообщения

К таким сообщениям относятся :

- сообщения о несовпадении Z-проекции образующих и направляющих в узлах (точках) пересечения соответствующих линий области определения контура, например, "рассогласование XXNNN(V) - YYMMM(V)=D", где XXNNN, YYMMM - имена линий БДАК для проекций образующей и направляющей, V - значение проекции в случае задания ее константой, D - разность между значениями проекций в узле;

- сообщения о невозможности вычисления v или u, v параметров области определения по заданным x, y, например, "***** ошибка 2 при вычислении v".

5.3.5.3. Сообщения об ошибках

Нормальное завершение программы имеет код 0. В противном случае выдается сообщение об ошибке, и программа завершает работу с кодом, отличным от 0.

В табл.5.7 представлены сообщения об ошибках, коды возврата программы и действия по устранению ошибок.

Ошибки программы

Таблица 5.7

Код возврата	Сообщения об ошибках	Действия по устранению
	SNTAX ERROR	Исправить ошибку в соответствии с грамматикой языка описания чертежей.
	Линия < имя линии > отсутствует в БДАК	Задать правильное имя линии БДАК
	Пустая линия	Задать правильное имя линии БДАК
	Недопустимый контур	Задать, как минимум, четыре линии области определения контура.
	Линии: < имя линии >, < имя линии >	Для указанных линий задать узел (точку пересечения) в явном виде.
	Пересечение не найдено	
	Границы не найдены	Задать допустимую область определения контура, либо задать границы в явном виде

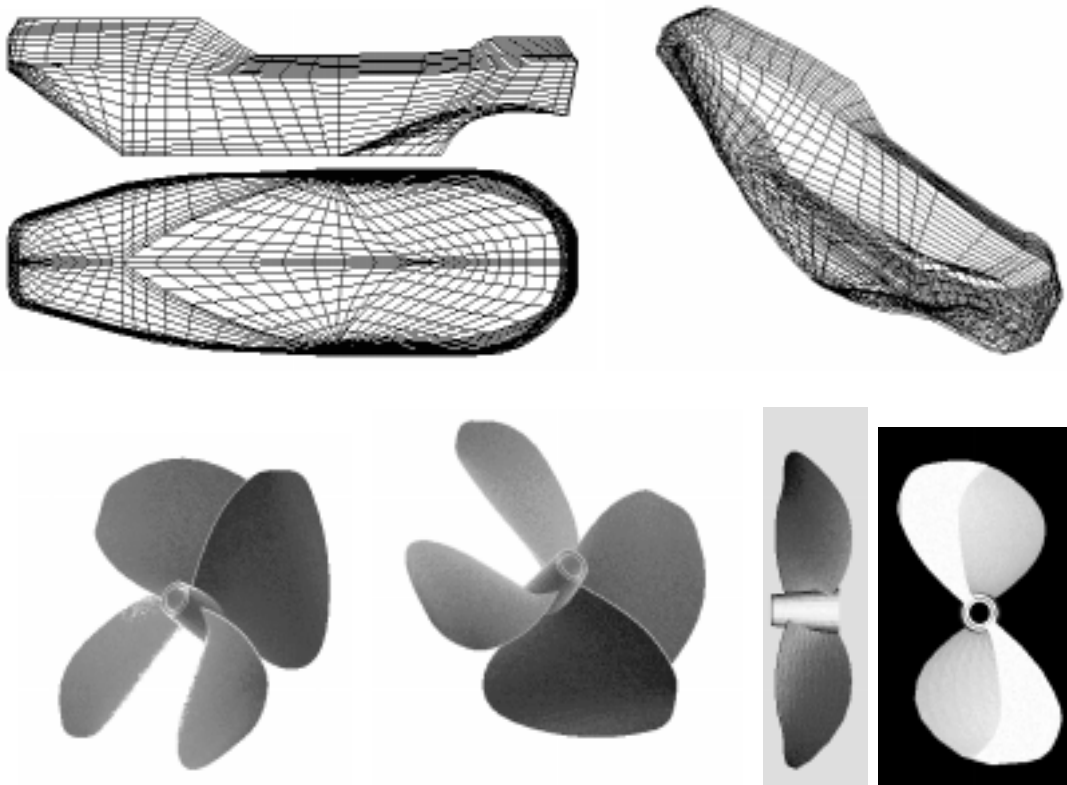


Рис.5.13. Примеры генерирования судовых форм в "ТЧ" и "СГ"

5.4. СГ - интерактивная графическая система создания и визуализации сложных геометрических объектов

Одной из важнейших проблем разработки САПР судов является описание и реалистическая визуализация в интерактивном режиме пространственных геометрических объектов. В конкретных прикладных задачах такими объектами могут быть как реальные объекты проектирования (деталь конструкции, корпус судна и т.п.), так и модельные объекты - модели реальных объектов и процессов. В первом случае наряду с визуализацией необходимо также построение точных чертежей проектируемых объектов.

Для модельных объектов достаточно иметь средства их описания и визуализации, а также обеспечить возможность вычисления различных геометрических характеристик в целях управления процессом конструирования. Примером таких задач является актуальная в машиностроении задача компоновки - оптимального размещения узлов и механизмов в замкнутом пространстве, решение которой состоит в замене реальных объектов упрощенными геометрическими, создании и визуализации на экране дисплея сцены модельных объектов, последующей оценке этой сцены оператором в интерактивном режиме. В результате выполнения последовательности таких итераций может быть получено оптимальное решение.

Из описанного алгоритма очевидным образом следует целесообразность разработки универсального математического и программного инструмента для конструирования и визуализации сложных пространственных объектов. Реализация такого инструмента в виде автономной программной системы, настраиваемой на различные входные структуры данных, позволит использовать эту систему как самостоятельно, так и в составе других прикладных систем типа САПР и АСНИ для визуализации сложных пространственных сцен.

Следует отметить, что безотносительно к выбранному математическому методу описания геометрических объектов и алгоритмам растровой визуализации особенностью любой реализованной системы такого назначения будет большое потребление вычислительных ресурсов. Например, визуализация практически важных сцен на ЭВМ типа ЕС-1066 или ПВМ с микропроцессором 80386/387 даже наиболее распространенным z-буферным алгоритмом требует от нескольких минут до десятков минут. Такая длительность, очевидно, неприемлема для интерактивного режима работы. Реальным выходом из этой ситуации может быть разработка параллельных алгоритмов и реализация их на многопроцессорных вычислительных системах.

Обсуждаемая задача может быть решена в нижеследующей постановке: необходимо разработать математические и программные средства интерактивного конструирования и растровой визуализации сложных пространственных сцен, отвечающие следующим требованиям:

- описание сложных геометрических объектов и сцен, составленных из объектов, осуществляется методом конструктивной геометрии;

- набор поддерживаемых геометрических примитивов, из которых конструируются объекты, должен быть достаточно широким (для целей приложения) и легко расширяемым;
- должна быть обеспечена возможность геометрических преобразований объектов сцены;
- вышеперечисленные возможности конструирования объектов и их преобразования должны поддерживаться средствами языка;
- алгоритмы растровой визуализации основываются на использовании **z**-буферов; при разработке и реализации алгоритмов преследуется цель минимизации вычислительных ресурсов;
- для повышения реалистичности изображений реализуются соответствующие механизмы (освещенность сцены источником света, зависимость освещенности от глубины сцены и т.д.);
- программная реализация на ПЭВМ выполняется на языке "СИ" в виде автономной интерактивной системы с настраиваемым входным интерфейсом;
- для повышения быстродействия системы распараллеливаются алгоритмы и программное обеспечение адаптируется к многопроцессорной вычислительной системе.

5.4.1. Метод и алгоритмы конструирования и визуализации геометрических объектов

В основу алгоритмического обеспечения разработанной системы положен метод конструктивной геометрии [221,223,224], сущность которого состоит в том, что описание геометрического объекта формируется с помощью теоретико-множественных операций и операций геометрического преобразования на ограниченном наборе типов примитивов. Внутренним представлением объекта при описании метода конструктивной геометрии является дерево (рис.5.14) с терминальными вершинами - примитивами (шары, цилиндры и т.д.) и нетерминальными вершинами - операциями объединения, пересечения, вычитания и геометрических преобразований.

При приведении этого дерева к дизъюнктивной нормальной форме (ДНФ) указанным операциям ставятся в соответствие логические операции: дизъюнкция и конъюнкция. Достоинствами такого подхода к описанию объектов являются простота определения семантики описаний (иначе говоря, просто построить предикат, который определяет, принадлежит точка объекту или нет) и простота алгоритмов теоретико-множественных операций. Однако у этого подхода есть и существенные недостатки, одним из которых является сложность получения изображений пространственных объектов, построенных методом конструктивной геометрии.

В диссертационной работе предлагается алгоритм конструктивной геометрии и реализующая его программная система построения методом растровых цветных изображений пространственных объектов.

Предлагаемый алгоритм имеет следующие достоинства:

- построение изображения не требует аппроксимации построенного объекта участками поверхности (например, многоугольниками [223]);
- изображение строится непосредственно на основе представления в виде дерева, терминальными вершинами которого являются примитивы, а нетерминальными - операции.

Класс примитивов заранее не ограничен. Примитивы определяются как абстрактные типы данных, над которыми разрешены следующие операции:

- 1) ортогональные геометрические преобразования,
- 2) нахождение точек пересечения с заданным лучом,
- 3) нахождение минимальных и максимальных значений (необязательно точных) координат примитива по осям трехмерного пространства.

Примитив может быть и составным объектом, например, многогранником, описанным через границу.

Рассмотрим более подробно организацию класса объектов, с которыми работает предлагаемый алгоритм. Объект будет описываться в виде дерева специального вида, представленного в ДНФ.

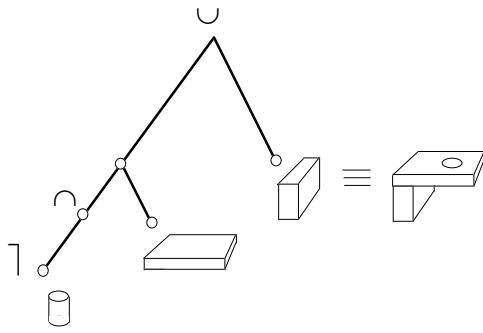


Рис. 5.14

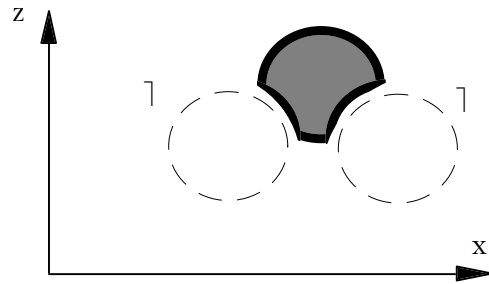


Рис. 5.15

Пусть P - любой примитив, над которым определены операции 1 - 3, либо его отрицание. Тогда описание пространственного объекта будет иметь вид

$$\bigcup_{i=1..n} \bigcap_{j=1..m} P_{ij}$$

Теоретико-множественные операции над объектами, представленными в такой форме, задаются достаточно просто, хотя и несколько сложнее, чем в случаях, когда объект представляется в виде произвольного булевого выражения. Операция объединения задается тривиально. Операции пересечения и вычитания задаются по известным формулам булевой алгебры:

$$\begin{aligned} & \left(\bigcup_{i=1..n} \bigcap_{j=1..m_i} P_{ij} \right) \cap \left(\bigcup_{i'=1..n'} \bigcup_{j'=1..m'_{i'}} Q_{i'j'} \right) = \\ & \bigcup_{i=1..4} \bigcup_{i'=1..n'} \left(\bigcap_{j=1..m_i} P_{ij} \cap Q_{i'j'} \right) - \left(\bigcup_{i=1..n} \bigcap_{j=1..m_i} P_{ij} \right) = \\ & \bigcap_{i=1..v} \bigcup_{j=1..m_i} IP_{ij} = \bigcup_{j_2=1..m} \bigcup_{j_1=1..m} \dots \bigcup_{j_n=1..m_n} \left(IP_{1j_1} \cap IP_{2j_2} \cap \dots \cap IP_{nj_n} \right). \end{aligned}$$

При построении результирующих дизъюнкций осуществляется грубая проверка: содержит ли данная конъюнкция хотя бы одну точку. Для этого строится пересечение параллелепипедов, аппроксимирующих примитивы (см. операцию 3 над примитивами), не имеющие знака "-". Если пересечение пусто, конъюнкция исключается, если пересечение не пусто, проверяется наличие в конъюнкции примитивов со знаком "-" таких, что параллелепипед, аппроксимирующий примитив, не пересекается с по-

строенным пересечением. Если такие примитивы есть, они исключаются из конъюнкции.

Операции геометрического преобразования объекта сводятся к преобразованию примитивов с использованием операции 1. Рассмотрим работу алгоритма построения растровых изображений для пространственных объектов, представленных в виде ДНФ. На первом этапе проводится геометрическое преобразование объекта, переводящее его в экранную систему координат (ЭСК). В этой системе координат плоскость $Z=0$ есть плоскость экрана, проецирование осуществляется вдоль оси z . Ориентация строки пикселей экрана совпадает с ориентацией оси x . Затем осуществляется сортировка примитивов, из которых состоит пространственный объект: каждой строке k пикселей экрана ставится в соответствие список S -примитивов таких, что минимальное значение примитива по оси y , найденное с помощью процедуры 3, попадает в данную строку.

На следующем этапе работы алгоритма происходит последовательный анализ строк с экрана. Для каждой строки k строится сечение z -буфера $Z=Z(x,y)$, где Z - расстояние до ближайшего участка поверхности выводимого объекта вдоль оси наблюдения, y - координата k -й строки пикселей, x - координата пикселей в строке. По двум соседним сечениям z -буфера (для строк k и $k-1$) строится строка изображения, которая выводится на растровое графическое устройство. Соседнее сечение z -буфера используется при этом для вычисления нормали к поверхности.

Освещенность в данной точке вычисляется по формуле

$$I = I_0 \cos(\theta) f(r),$$

где I - интенсивность источника света; θ - угол между нормалью к поверхности и направлением на источник света; r - расстояние от точки поверхности до источника света; f - функция убывания освещенности с глубиной (обычно $a/a + r$).

При последовательном анализе строк алгоритм использует текущий список S -примитивов, проекции которых пересекаются с данной строкой. Первоначально этот список пуст. При переходе к анализу новой строки в него добавляется список примитивов S (см. выше), проекции которых пересекаются с данной строкой и не пересекаются с предыдущей, и из него исключаются примитивы, для которых максимальное значение по оси y меньше значения y для данной строки k .

Таким образом, резко сокращается количество примитивов, которое надо рассмотреть при построении сечения \mathbf{z} -буфера, а если процедура Z дает точные значения минимального и максимального значения по оси y , то при построении сечения \mathbf{z} -буфера достаточно рассматривать только те примитивы, проекции которых заведомо пересекаются с данной строкой пикселей.

Рассмотрим теперь построение сечения \mathbf{z} -буфера для строки пикселей k на основе текущего списка примитивов. Первоначально $z = +$ для любого x . Затем для конъюнкции j исходного пространственного объекта, все примитивы (без знака) которого находятся в текущем списке примитивов S , строятся сечения \mathbf{z} -буфера $Z(x, y)$. Сечение \mathbf{z} -буфера для всего объекта последовательно модифицируется этими \mathbf{z} -буферами по формуле

$$Z(x, y) = \min(Z(x, y_n), z_j(x, y_n))$$

Сечение \mathbf{z} -буфера $Z(x, y)$ для конъюнкции j строится следующим образом (см. рис.5.15). Из текущего списка примитивов S выбирается очередной примитив P . Если все примитивы, входящие в конъюнкцию j и содержащие P без знака, входят одновременно и в текущий список примитивов S , и если сечение $Z(x, y)$ для конъюнкции j еще не строилось, то строится отрезок $[XMIN, XMAX]$, являющийся пересечением всех отрезков $[XMIN, XMAX]$, где $XMIN, XMAX$ - минимальное и максимальное значения x для примитивов без знака, входящих в конъюнкцию j . Если отрезок $[XMIN, XMAX]$ не пуст, строится таблица, в которой каждому пикселю x, y \mathbf{z} -буфера $Z(k, y)$ (где x лежит в интервале от $XMIN$ до $XMAX$) ставится в соответствие список S . В список S входят координаты точек пересечения луча, проведенного через точку x, y параллельно оси наблюдения z с поверхности всех примитивов, входящих в конъюнкцию j . Точки пересечения луча с примитивом можно найти в силу существования процедуры 2 (см. выше). Точки пересечения, входящие в список S , упорядочиваются по возрастанию координаты z . Затем для каждого пикселя x, y ищется расстояние $Z(x, y)$ вдоль оси наблюдения до ближайшей точки поверхности конъюнкции j . Для этого используется список S точек пересечения луча наблюдения, проходящего через пиксель x, y , с примитивами из конъюнкции j .

Рассмотрим процесс нахождения $Z(x, y)$. Будем двигаться по лучу из точки $Z=0, x=x, y=y$ вдоль оси z . Обозначим количество примитивов без знака из конъюнкции j , внутри которых будем находиться, через N , а

количество примитивов со знаком из конъюнкции j , внутри которых будем находиться, как N . Тогда, если при движении вдоль оси z пересекаем: первую граничную точку примитива со знаком из конъюнкции j , то $N:=N-1$; первую граничную точку примитива без знака из конъюнкции j , то $N:=N+1$; вторую граничную точку примитива без знака из конъюнкции j , то $N:=N-1$.

Если после пересечения очередной граничной точки $N=0$ и $N=N$ (общему количеству примитивов без знака из конъюнкции j), то z координата данной граничной точки дает искомое значение $Z(x,y)$. Действительно, $N=0$ и $N=N$ означает, что мы находимся внутри всех примитивов конъюнкции j без знака и вне всех примитивов конъюнкции j со знаком "+".

В качестве геометрических примитивов на текущий момент в системе выбраны и программно поддерживаются следующие примитивы: полупространство, шар, параллелепипед, цилиндр, конус, обобщенный цилиндр (образующая - произвольный контур), обобщенный тор (осевая - произвольная кривая в пространстве, образующая - окружность переменного радиуса $r=f(t)$, где t - параметр, зависящий от длины осевой).

Организация диалога базируется на использовании специально разработанного графического монитора, описание работы с которым дано в прил. 3.

5.4.2. Распараллеливание алгоритмов

Для анализа возможных методов распараллеливания общего алгоритма работы системы, описанного в предыдущей главе, представим его в виде укрупненных блок-схем (рис.5.16 и рис.5.17).

На рис.5.16 выделены три этапа преобразований графических данных:

- задание в диалоговом режиме (возможен и программный способ) описания сцены в мировой системе координат,
- преобразование данных в экранную систему координат с помощью графического монитора (при участии оператора-пользователя),
- построчное формирование растрового изображения, обеспечиваемое работой всех алгоритмов визуализации.

Для упрощения работы по адаптации ПО на многопроцессорной системе вначале была реализована простая схема распределения ПО систе-

мы между ПЭВМ и транспьютерным набором. По этой схеме первые два этапа преобразований (блок-схема рис.5.16) и визуализация выполнялись на ПЭВМ. Транспьютерный набор использовался для формирования изображения, организованного в виде одной задачи. Все алгоритмы этого этапа включают операции над числами с плавающей запятой. В результате тестирования было выявлено, что при такой схеме распределения ПО на сравнительно сложных геометрических сценах достигается ускорение в сравнении с функционированием на ПЭВМ в 10 - 12 раз.

Дальнейшие действия были направлены на реализацию распараллеливания применительно к этапу формирования изображения, для которого характерна трудоемкость алгоритмов. Такое распараллеливание работы системы возможно на двух уровнях. Первый уровень соответствует различным способам деления экранной матрицы изображения на составляющие области, для каждой из которых организуется отдельный процесс полного цикла обработки данных до формирования изображения. В этом случае каждый процесс заполняет свою область в результирующей матрице изображения, которая по завершению всех процессов передается на визуализацию.

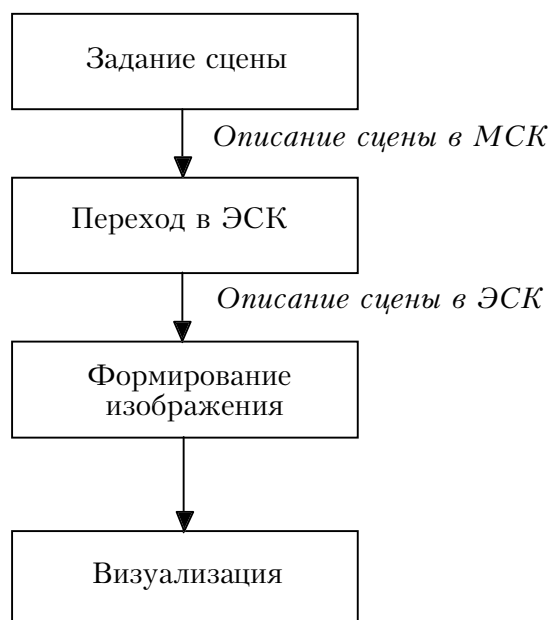


Рис.5.16. Алгоритм обработки сцены

Достоинством такого способа распараллеливания является то, что при этом не требуется модификации последовательно работающего алгоритма системы. Издержки здесь возникают за счет того, что при раздельной обработке отдельных частей изображения-экрана появляются граничные эффекты, для устранения которых необходимо предусмотреть грамотную "сшивку" смежных частей. Более существенный недостаток заключается в том, что при таком способе априори нельзя получить оптимального использования имеющихся процессов без дополнительного механизма адаптивного разделения. Причина состоит в том, что одинаковым по площади частям экрана могут соответствовать различные по насыщенности изображения и, в итоге, разные объемы вычислений.

Второй уровень распараллеливания связан с реализацией параллельных алгоритмов, т.е. алгоритмов с параллельной обработкой промежуточных данных. В этом случае глубина параллелизма определяется свойствами самого алгоритма. По-видимому, оптимальное распараллеливание предполагает реализацию на обоих выделенных уровнях распараллеливания.

Первоначально был реализован способ распараллеливания, соответствующий первому уровню. Из всех возможных разбиений - на квадраты (размером в пиксель и больше), горизонтальные полосы (от одной строки до нескольких), вертикальные полосы и возможные их комбинации - выбран способ разбиения матрицы-экрана на вертикальные полосы.

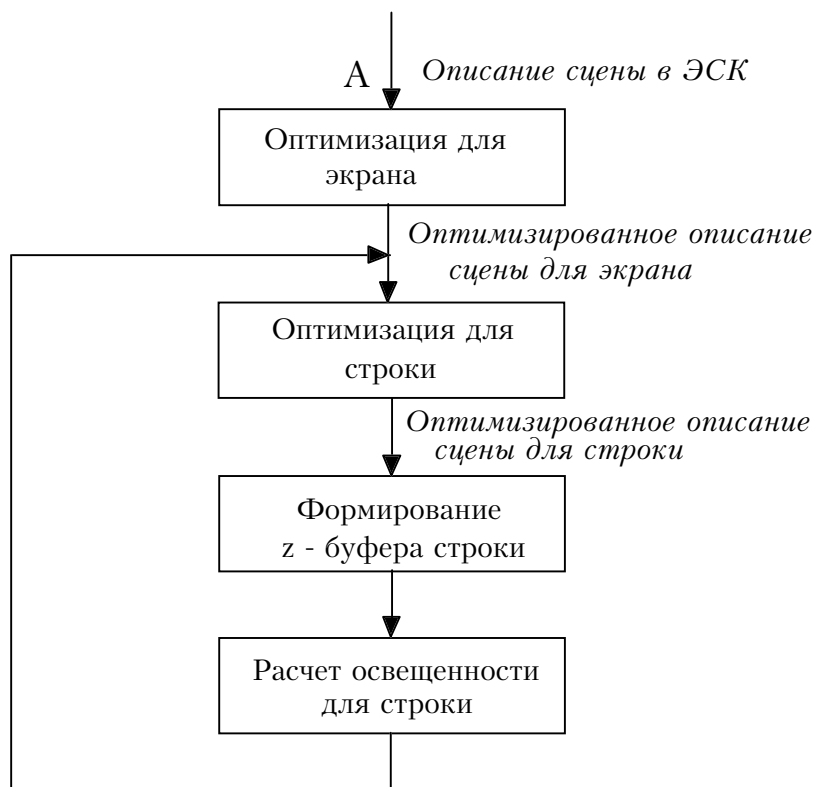


Рис. 5.17. Формирование растрового изображения

Применительно к блок-схеме на рис.5.17 распараллеливание в этом случае осуществляется в точке А. Этот способ в сравнении с другими перечисленными имеет следующие достоинства:

- каждая вертикальная полоса рассматривается как самостоятельный экран, применительно к которому работает весь последовательный алгоритм (от точки А), а модификация ПО в этом случае минимальна;
- трудоемкий этап "оптимизация для экрана" выполняется меньшее число раз в сравнении с другими способами разбиения;
- издержки на устранение граничных эффектов в сравнении с другими способами минимальны.

Реализация предложенного способа распараллеливания работы системы на сложных геометрических сценах обеспечила ускорение в 8-9 раз. В результате тестирования метода параллельной обработки генерирования сцен на компьютерном наборе в системе CG (рис.5.1) была создана отдельная система твердотельного моделирования на компьютерной технологии. Ниже приводятся ее описание и характеристики проведенных испытаний.

5.5. TCG - система проектирования трехмерных объектов в реальном времени на транспьютерной технологии

5.5.1. Назначение системы

Программа TCG предназначена для описания сложных трехмерных сцен и их реалистичной визуализации.

Сцены, как и в системе CG, описываются при помощи (конструируются из) трехмерных геометрических примитивов. Примитивы - это параметрически заданные тела в пространстве, например: шар, конус, параллелепипед. Параметры примитивов характеризуют их расположение, размеры и, в какой-то степени, форму. При конструировании экземпляры примитивов задаются указанием их типа и значений параметров. Экземпляры примитивов называются элементами. Реальные пространственные тела описываются как результаты операций пересечения, объединения и теоретико-множественного вычитания элементов, представляющих собой множества точек трехмерного пространства. Пересечение (вычитание) набора элементов называется конъюнкцией. Объединение набора конъюнкций называется объектом. При конструировании сцены определяется множество конъюнкций посредством задания их элементов и множество объектов. Объекты можно геометрически преобразовывать, т.е. перемещать, поворачивать и однородно растягивать (сжимать).

Визуализация производится для произвольных объектов. При визуализации предполагается, что источник освещения находится строго сзади на бесконечности. Точку зрения на сцену можно изменять.

5.5.2. Условия применения

Интерактивная часть программы TCG работает на ЭВМ типа IBM PC под управлением MS DOS. Расчет данных для визуализации производится на транспьютерной сети (см. п. "Алгоритмы визуализации").

Возможны три графических режима работы программы:

- 1) для адаптера VGA - режим 320 x 200 пикселей, 256 цветов;
- 2) для адаптера VGA - режим 640 x 480 пикселей, 16 цветов;
- 3) для адаптера EGA - режим 640 x 350 пикселей, 16 цветов.

Дополнительно возможна визуализация на транспьютерном графическом контроллере G170 (512 x 512 пикселей, 256 цветов).

Язык программирования программ для IBM PC - Си (транслятор MSC/QC). Язык программирования программ для транспьютерной сети - ZLC.

5.5.2. Логическая организация данных

Логическая организация данных представляет собой квазидерево, отличающееся от классического дерева тем, что некоторые узлы могут иметь общие подчиненные узлы. Корнем такого квазидерева является сцена, терминалами - элементы, нетерминальными узлами - объекты. Узлы, имеющие в подчинении элементы и только их - это конъюнкции. Объекты, не являющиеся конъюнкциями, не могут иметь в подчинении элементы. Элементы интерпретируются как экземпляры примитивов, а конъюнкции - как пересечения множества подчиненных им элементов, причем элемент входит в конъюнкцию со знаком минус, если производится пересечение с его дополнением (вычитание). Объекты, не являющиеся конъюнкциями, интерпретируются как произвольные подмножества конъюнкций. В подчинении объектов могут быть другие объекты (подобъекты), но реально они представляют все множество конъюнкций, подчиненных всем своим подобъектам. Таким образом, любой объект (в том числе и сцена) логически описывается как объединение пересечений элементов. Подобъекты введены для удобства задания объектов и операций с ними.

Имеются следующие типы элементов (примитивы): полупространство, шар, параллелепипед, шестигранник, цилиндр, конус, эллиптический цилиндр, выпуклое тело вращения, тор, конический тор, выпуклый тор, спираль, коническая спираль, выпуклая спираль, участок поверхности.

Примитивы: полупространство, шар, параллелепипед, цилиндр, конус, тор - обычные, определяемые в пространственной геометрии тела. Эллиптический цилиндр - это цилиндр, имеющий в сечении, ортогональном его оси, не окружность, а произвольный эллипс. Участок поверхности - это массив двумерных точек, полученный ранее любым методом и передаваемый в специальном формате в систему TCG.

Некоторые примитивы входят в классы - обобщенные примитивы. В частности, эллиптический цилиндр является представителем класса

"обобщенный цилиндр", а примитивы: выпуклое тело вращения, тор, конический тор, выпуклый тор, спираль, коническая спираль, выпуклая спираль - являются представителями класса "обобщенный тор". С точки зрения логической организации все примитивы эквивалентны. Различие состоит в том, что классы "обобщенные примитивы" легко можно расширить. Для этого необходимо запрограммировать новые функции (пользователя), описывающие формообразующие линии данного класса, описать новые примитивы в таблице и перекомпоновать программу. Для обобщенного цилиндра необходимо добавить функцию, вычисляющую координаты произвольной плоской параметрической линии - образующей цилиндра. Для обобщенного тора необходимо добавить функцию, вычисляющую координаты произвольной пространственной параметрической линии - оси обобщенного тора и функцию зависимости радиуса обобщенного тора от t -параметра.

Для выпуклого тела вращения осью является прямая, а функцией зависимости радиуса - синусоида, определенная на периоде. Для тора ось - это окружность; функция зависимости радиуса - константа. Для конического тора ось - окружность; функция зависимости радиуса - линейная. Для выпуклого тора ось - окружность; функция зависимости радиуса - синусоида на периоде. Для спирали ось определяется функцией типа:

$$\begin{aligned}x &= \sin(wt), \\ y &= \cos(wt), \\ z &= at,\end{aligned}\tag{5.4}$$

функция зависимости радиуса - константа. Для конической спирали ось - функция типа (5.4); функция зависимости радиуса - линейная. Для выпуклой спирали ось - функция типа (5.4); функция зависимости радиуса - синусоида на периоде.

Следует отметить, что примитивы: цилиндр, конус, эллиптический цилиндр - не ограничены. Для работы с ограниченными телами необходимо задать их пересечение с полупространствами или с какими-либо ограниченными телами.

Параметры примитивов приводятся в описании операций задания элементов (см. прил.3).

5.5.3. Алгоритмы визуализации

Ниже рассматривается алгоритм визуализации, реализованный в последовательном прототипе программы TCG. Алгоритм основан на методе вычисления z-буфера для каждой строки экрана.

1. Производится копирование описания всей сцены или той ее части, которую требуется визуализировать. Дальнейшая работа производится с полученной копией. Этот этап необходим в силу того, что геометрические характеристики сцены в процессе работы будут изменены, в то же время необходимо хранить оригинал сцены в мировой системе координат (МСК) для последующего редактирования сцены.

2. Описание сцены преобразуется из МСК в ЭСК в соответствии с ранее установленными параметрами ЭСК. Преобразование заключается в изменении геометрических характеристик (например, центра или точки привязки какого-либо элемента) посредством применения к ним матрицы геометрического преобразования. Объем работы на этом этапе не зависит от области отображаемой сцены (области зрения).

3. Выполняется глобальная оптимизация сцены. Этот этап заключается в: а) вычислении глобальных геометрических характеристик сцены, включающих габариты конъюнкций (минимальные и максимальные значения x , y - координат области конъюнкции); б) корректировке сцены - если конъюнкции не пересекаются с областью зрения, они удаляются из описания сцены.

4. В цикле по строкам экрана выполняются:

4.1) локальная оптимизация сцены; вычисляются сечения элементов с плоскостью строки и плоские (x_{\min} , x_{\max}) габариты конъюнкций;

4.2) вычисление z-буфера строки;

4.3) вычисление яркостей пикселей строки;

4.4) формирование образа видеобуфера строки и вывод его на экран.

Вычислительную схему используемого в системе алгоритма визуализации можно рассматривать как имеющую несколько измерений, в каждом из которых возможно распараллеливание вычислений.

Одним измерением является растровая матрица экрана, которую требуется сформировать в итоге обработки всей сцены. Здесь возможно фиксированное разделение экрана на отдельные области, обработку

(формирование) каждой из которых можно организовывать отдельно (одним транспьютерным модулем). В частности, в качестве таких областей удобно рассматривать горизонтальные или вертикальные полосы экрана. Основной недостаток такого подхода - возможное различие в трудоемкости параллельных процессов и, как следствие этого, неэффективное использование процессоров. Различие между областями возникает из-за того, что сцена располагается на экране неравномерно.

Этот недостаток устраняется, если области выбрать маленькими, а процессоры использовать как разделяемый ресурс, однако в этом случае возрастают затраты на передачу данных процессорам. Оптимизировать этот подход может специальная система динамического разбиения на области, которая обеспечивает равномерную загрузку процессоров.

Другим измерением в указанном выше смысле является последовательность локальных вычислительных процедур алгоритма, работающих на множестве исходных геометрических примитивов обрабатываемой сцены. Такими процедурами являются: вычисление минимаксов для строки, экрана, области; вычисление плоского сечения для строки экрана; вычисление z -буферной строки; вычисление яркости элементов строки для установленной проекции (параллельной/ перспективной). Следует при этом отметить, что отдельные локальные процедуры также могут распараллеливаться, например, на множестве растр-элементов.

Третьим измерением распараллеливания может быть множество геометрических примитивов исходной сцены. В этом случае процедурой обработки, рассматриваемой в качестве отдельного процесса, является цепочка вычислений до получения z -буферной строки примитива.

Помимо изложенных соображений в результирующей схеме распараллеливания необходимо учитывать следующий эмпирический факт: больший эффект достигается на больших объемах арифметических вычислений.

В результате учета всех рассмотренных возможностей распараллеливания разработана многоуровневая схема распараллеливания алгоритма визуализации: первый уровень - фиксированное или адаптивное разбиение экрана на области; второй уровень - вычисление плоских сечений и z -буферных строк на множестве примитивов; третий уровень - разбиение яркостной матрицы данной области на подобласти.

В данной версии TCG реализован метод фиксированного разбиения экрана на области - горизонтальные полосы. Разбиение на вертикальные полосы менее эффективно для вышеописанного алгоритма визуализации, т. к. в этом случае локальная оптимизация сцены дублируется на каждом процессоре. Каждая полоса обсчитывается отдельным процессом. Число процессоров-транспьютеров N и число процессов M задаются параметрами при запуске программы TCG. Если число процессов больше числа процессоров, процессоры используются как разделяемые ресурсы. Данный подход налагает ограничение на максимальное число используемых процессов: оно не может превышать число обрабатываемых строк (S).

Рассмотрим алгоритм работы программы TCG на транспьютерной сети. Для обеспечения задачи маршрутизации сообщений по транспьютерной сети разработаны пакет функций (TCGROUT), обеспечивающих посылку пакетов данных с произвольного транспьютера на любой другой, к которому существует физический путь доступа, и копирование пакетов данных на все рабочие транспьютеры сети. Для связи с HOST (IBM PC) машиной использовался пакет функций PTC, разработанный в ВНИИПВТИ.

Предполагается, что транспьютерная сеть содержит два выделенных узла (корневой транспьютер, выполняющий функции связи с IBM PC и функции планирования работ на сети, и транспьютер, связанный с графическим контроллером G170), а также ряд одинаковых рабочих узлов, на которых и выполняется собственно работа. Средствами пакета TCGROUTR можно настроить (конфигурировать) программу TCG на сеть произвольной конфигурации, удовлетворяющей вышеприведенным требованиям. Конфигурации сети, на которых была реализована данная версия программы TCG, приведены на рис.5.18 и рис.5.19.

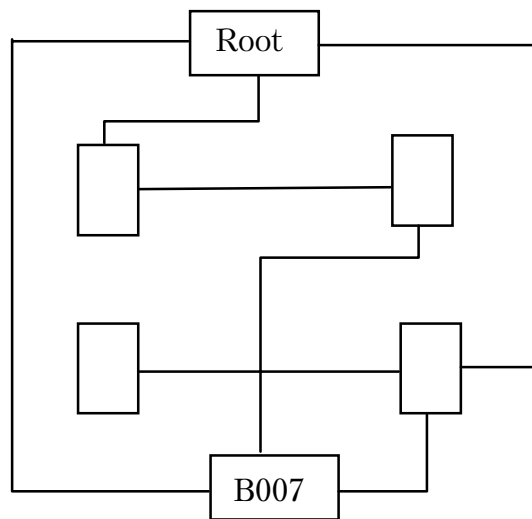


Рис 5.18. Конфигурация сети из 6 транспьютеров.

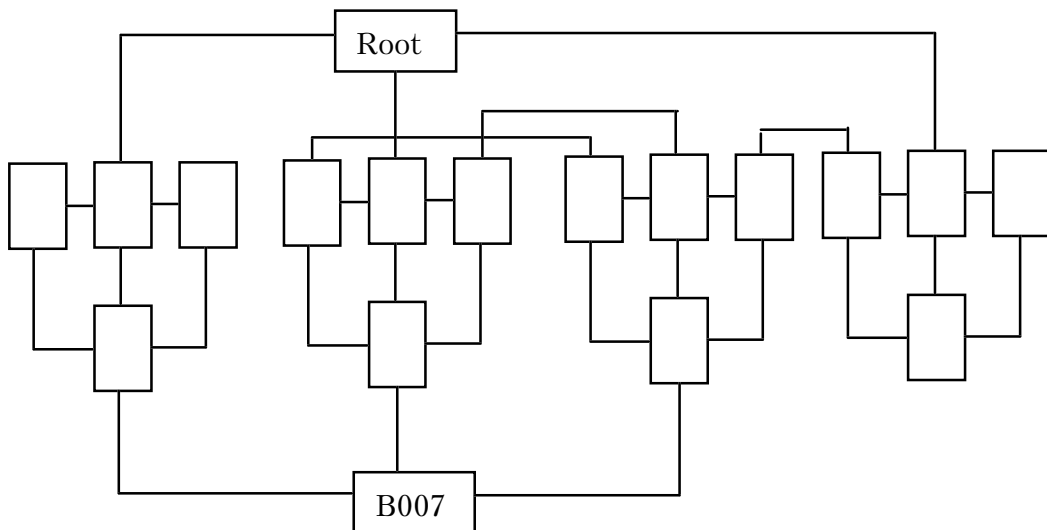


Рис 5.19. Конфигурация сети из 18 транспьютеров.

Транспьютерная сеть загружается в начале работы программы TCG. При загрузке сети на корневом и на всех рабочих транспьютерах запускаются задачи ROUTER, ответственные за пересылку пакетов. На корневом транспьютере также запускается задача-планировщик, распределяющая работы по рабочим транспьютерам, на транспьютере, связанном с G170 - задача, принимающая графические данные и выводящая их на экран дисплея G170, а на всех рабочих транспьютерах - задачи визуализации сцены. На все задачи ROUTER передается таблица маршрутизации, заданная в файле конфигурации (например, TCG.CFG).

В процессе диалога после задания сцены она копируется на рабочие транспьютеры. При выполнении операции "Визуализация" на рабочие

транспьютеры копируются дополнительные данные, и задаче-планировщику на корневом транспьютере передается команда "начать работу". Далее программа на IBM PC ожидает приема пакетов данных от транспьютерной сети. Возможны следующие пакеты:

- предупредительные текстовые сообщения;
- образы видеобуфера дисплея IBM PC (для режимов вывода на IBM PC);
- сообщение об окончании работы на транспьютерной сети.

Текстовые сообщения и образы видеобуфера выводятся на экран дисплея IBM PC. При получении сообщения об окончании работы программа на IBM PC продолжает работу в диалоге.

При получении команды "начать работу" эта программа запускает M процессов на N рабочих транспьютерах - передает задачам визуализации данные, описывающие полосу, и команду "начать работу". Если $M > N$, сначала запускаются N процессов. Задача-планировщик ожидает сообщений "конец работы" от задач визуализации и при получении такого сообщения запускает следующий процесс на освободившемся транспьютере. По завершении всех процессов задача-планировщик передает сообщение "конец работы" программе на IBM PC.

Задачи визуализации при получении команды "начать работу" выполняют пункты 2,3,4 вышепредставленного алгоритма визуализации. При получении образа видеобуфера строки он передается программе на IBM PC для режимов вывода на IBM PC либо задаче, обслуживающей G170. Когда все строки данной горизонтальной полосы визуализированы задача визуализации передает сообщение "конец работы" задаче-планировщику.

Определим ускорение k на N транспьютерах как отношение времени работы программы на одном транспьютере (t_1) ко времени работы программы на сети из N транспьютеров (t_N):

$$k(N) = t_1 / t_N \quad (5.5)$$

и эффективность распараллеливания как отношение этого ускорения к N (в процентах):

$$e = k(N) / N . \quad (5.6)$$

Эффективность распараллеливания показывает, какая часть потенциальной мощности транспьютерной сети задействована при визуализации данной сцены данным алгоритмом.

В идеальном случае $k(N)=N$ и $e=100\%$. Реально же $k(N)$ зависит от алгоритма распараллеливания. В рассматриваемом алгоритме в целях обеспечения непрерывности изображения между полосами необходимо для каждой полосы обсчитывать на одну строку больше высоты полосы, что приводит к дублированию вычислений для некоторых строк. В этом случае в предположении, что сцена однородна и $M=N$,

$$t_N = \frac{(S / N + 1) \cdot t_1}{S},$$

где S - число выводимых строк.

Зависимость $k(N)$ выражается следующим образом:

$$k(N) = N - \frac{1}{1 + N / S}. \quad (5.7)$$

В граничном случае, когда $N=S$, $k=N/2$, эффективность $e=50\%$.

Кроме того, в реальном случае время работы на одном транспьютере имеет некоторую постоянную составляющую (t_0), не зависящую от объема работ на этом транспьютере. Следовательно, и время работы на сети из N транспьютеров будет содержать эту составляющую. В эту составляющую входят накладные расходы на транспортировку сообщений, увеличивающиеся с ростом N , и время выполнения установочных операций, которые дублируются на каждом транспьютере. В представленном алгоритме распараллеливания пункт 2 и часть а) пункта 3 алгоритма визуализации (геометрическое преобразование сцены и вычисление глобальных геометрических характеристик) дублируются на каждом процессе и, таким образом, время выполнения этой части работы входит в постоянную составляющую. При учете постоянной составляющей время работы на сети из N транспьютеров выражается формулой:

$$t_N = t_0 + \frac{(S / N + 1)(t_1 - t_0)}{S}, \quad (5.8)$$

а зависимость $k(N)$:

$$k(N) = N - \frac{1}{1 + N / S + (t_0 / t_1)(N - 1 - N / S)}. \quad (5.9)$$

Во многих случаях членами N/S в (5.6) можно пренебречь и тогда формула (5.9) выражается более просто:

$$k(N) = N - \frac{1}{1 + (t_0 / t_1)(N - 1)} . \quad (5.10)$$

Из формулы (5.9) видно, что при t_0/t_1 , стремящемся к 1, $k(N)$ стремится к 1 ($e=1/N\%$), т. е. ускорения нет.

Из (5.10) следует, что при возрастании N (в области, где $t_0N/t_1 \gg 1$) $k(N)$ стремится к постоянной величине t_1/t_0 ($e=t_1/(t_0N)\%$), т. е. наступает насыщение (формула (5.9) дает в этом случае некоторое другое значение постоянной величины). Следовательно, для получения максимальной эффективности при увеличении числа транспьютеров необходимо уменьшать отношение t_1/t_0 . Величина t_0 в первую очередь зависит от сложности сцены (если не учитывать накладные расходы на транспортировку), которая примерно определяется числом конъюнкций и элементов в сцене, в то время как t_1-t_0 зависит от площади сцены (числа обрабатываемых пикселей). Таким образом, распараллеливание визуализации эффективно для сцен, требующих большого времени счета и менее эффективно для сцен с малым временем счета.

Повышение эффективности распараллеливания программы TSG возможно за счет уменьшения значения t_0 двумя способами.

Первый способ заключается в распараллеливании пункта 2 и части а) пункта 3 алгоритма визуализации. В этом случае можно сначала запускать ряд процессов, каждый из которых выполняет геометрические преобразования и вычисление глобальных характеристик одной конъюнкции или одного элемента, а затем уже запускать процессы, выполняющие обработку горизонтальных полос.

Второй способ заключается в выполнении пункта 2 и части а) пункта 3 алгоритма визуализации на высокоскоростном геометрическом процессоре, например, на i860.

Следует отметить, что неоднородность сцены уменьшает эффективность. Увеличение числа процессов по отношению к числу транспьютеров приводит к меньшей зависимости от неоднородности сцены и к увеличению эффективности распараллеливания. С другой стороны, в знаменателях формул (5.7) - (5.10) необходимо использовать значение M вместо N , что приводит к обратному эффекту. Для каждой сцены существует некоторое оптимальное значение $M \geq N$, для которого эффективность распараллеливания максимальна, но определить это оптимальное значение заранее, перед выводом сцены, довольно трудно. С другой

стороны адаптивное разбиение экрана на области может повлечь дополнительные накладные расходы.

Из формулы (5.9) можно получить (с учетом $M > N$) формулу, выражающую зависимость t_0/t_1 от заданных значений N , M , $k(N)$, S :

$$t_0 / t_1 = \frac{N / k(N) - 1 - V / S}{M - 1 - M / S} . \quad (5.11)$$

Данная версия программы TCG испытывалась на двух вариантах транспьютерной сети. Первый вариант использовался в основном для отладки и проверки общей работоспособности программы. Все анализируемые данные получены для второго варианта.

Второй вариант представляет собой сеть из 18 транспьютеров, 16 из которых используются в качестве рабочих. Для исследования влияния числа рабочих транспьютеров (N) программа TCG запускалась с числом процессов (M), меньшим 16. Этим имитировалась сеть с меньшим числом транспьютеров.

В табл.5.8 представлены исходные данные, полученные при визуализации сцены 1 - модельного города (рис.5.20) и данные, вычисленные из исходных по формулам (5.5), (5.6) и формуле (5.11) (см. п. "Алгоритмы визуализации"). Следует отметить, что оценка t_0/t_1 довольно приблизительно, т. к. формула (5.5) не учитывает неоднородности сцены.

В случаях $M \leq 16$ N принимается равным M . Для $M > 16$ N принимается равным 16.

Для сравнения приведены данные для режимов "GRAY" (вывод на IBM PC, разрешение 640 x 480) и "G170" (вывод на G170, разрешение 512 x 512). Расчет производился только для данных режима "G170".

Время визуализации той же сцены последовательным прототипом программы TCG (на IBM PC, в режиме "GRAY") равно 352.1 с.

Таблица 5.8

Число процессов M	Время виз.(с.) "GRAY"	Время виз.(с.) "G170"	$k(N)$	e	t_0/t_1
---------------------	-----------------------	-----------------------	--------	-----	-----------

1	119,3	95,0	-		
2	66,6	52,8	1,80	90%	0,108
4	41,0	33,7	2,82	71%	0,137
8	25,5	21,0	4,53	56%	0,108
16	14,1	11,6	8,19	51%	0,062
24	12,1	9,7	9,79	61%	0,026
26	11,2	9,5	10,00	63%	0,022
28	11,0	9,1	10,44	65%	0,018
30	12,2	10,5	9,05	57%	0,024
32	13,5	11,1	8,56	54%	0,026

Сцена 1 достаточно сложная: состоит из примерно 50 конъюнкций и занимает всю площадь вывода для экрана. Визуализация производилась в перспективной проекции.

На основании анализа данных табл.5.8 можно сделать следующие выводы.

1. Ускорение растет в зависимости от числа транспьютеров, но рост его замедляется.

2. Оптимальное число процессов для 16 транспьютеров и данной сцены равно 28.

3. Сильный разброс значений t_0/t_1 показывает, что данная сцена в большой степени неоднородна.

4. Теоретически значение t_0/t_1 для однородной сцены может только увеличиваться с ростом числа процессов. Большие значения t_0/t_1 в верхней части табл. 5.9 объясняются неточностью формулы (5.9) для неоднородной сцены. Поэтому в качестве оценки t_0/t_1 выбирается минимальное значение: $t_0/t_1=0.018$. Для такого значения t_0/t_1 насыщение ускорения наступит при $N \gg 76$. Постоянная составляющая в этом случае $t_0=1.7$ с., т.е. вне зависимости от числа транспьютеров и числа процессов время визуализации данной сцены данным алгоритмом, меньшее, чем 1.7 с., нельзя получить.

В табл.5.9 представлены исходные данные, полученные при визуализации сцены 2 - модельного города и данные, вычисленные из исходных по формулам (5.5), (5.6), (5.11). Сцена 2 (рис.5.20) отличается от сцены 1 (рис.5.20) только ракурсом зрения, поэтому значения t_0 для сцены 1 и для сцены 2 должны быть равны. Вывод изображения произво-

дился на дисплей G170 на весь экран (окно 512 x 512) и в окна размерами (256 x 256) и (128 x 128). Для сравнения в табл.5.9 представлены времена визуализации для метода распараллеливания, в котором экран разбивается на вертикальные полосы. Используется методика расчетов такая же, как и для табл.5.8. Расчет производился только для данных метода горизонтальных полос.

Время визуализации той же сцены последовательным прототипом программы TCG (на IBM PC, в режиме "GRAY") равно 298.5 с.

Таблица 5.9

Число процессов М	Время виз.(с.) "GRAY"	Время виз.(с.) X-полосы	k(N)	e	t0/t1
Окно 512 x 512					
1	79,5	79,5			
16	12,4	8,6	9,24	58%	0,047
32	18,7	9,3	8,55	53%	0,026
Окно 256 x 256					
1	22,5	22,5			
16	5,4	3,5	6,43	40%	0,095
32	9,4	4,6	4,89	31%	0,069
Окно 128 x 128					
1	7,4	7,4			
16	3,1	2,0	3,70	23%	0,215
32	5,7	3,2	2,31	14%	0,184

На основании анализа данных табл. 2 можно сделать следующие выводы.

1. Метод распараллеливания, основанный на разбиении экрана на вертикальные полосы, менее эффективен, чем метод, основанный на разбиении экрана на горизонтальные полосы. Это объясняется тем, что для первого метода в постоянную составляющую входит дополнительно время выполнения этапа 5.1 алгоритма визуализации.

2. Эффективность распараллеливания уменьшается с уменьшением размеров (площади) окна визуализации. Это объясняется тем, что составляющая времени визуализации t_1-t_0 уменьшается с уменьшением

площади окна, в то время как t_0 остается постоянной. Следовательно, t_0/t_1 увеличивается.

3. В качестве оценок для t_0/t_1 выбираются минимальные значения (см. 4 вывод для табл.5.9). Таким образом, для разных окон получаются следующие значения:

$$\begin{aligned} (512 \times 512) - t_0/t_1 &= 0.026; & t_0 &= 2.1 \text{ с.}; \\ (256 \times 256) - t_0/t_1 &= 0.069; & t_0 &= 1.6 \text{ с.}; \\ (128 \times 128) - t_0/t_1 &= 0.184; & t_0 &= 1.4 \text{ с.} \end{aligned}$$

Среднее значение постоянной составляющей (с учетом $t_0 = 1.7$ для сцены N 1) $t_0 = 1.7$ с. Насыщение ускорения для разных окон наступит при:

$$\begin{aligned} (512 \times 512) - N &\gg 38, \\ (256 \times 256) - N &\gg 14, \\ (128 \times 128) - N &\gg 5. \end{aligned}$$

В табл.5.10 представлены исходные данные, полученные при визуализации сцены 3 - куба с сферическими углублениями и данные, вычисленные из исходных по формулам (5.5), (5.6), (5.10). Эта сцена состоит из одной конъюнкции, содержащей четыре элемента. Визуализация производилась в параллельной проекции в окне (128 x 128).

Сцена визуализировалась в цикле из 100 шагов, причем на каждом шаге менялась точка зрения на сцену. В табл.5.10 приводится усредненное по 100 шагам время визуализации. Данные приводятся для одного процесса и для оптимального числа процессов.

Таблица 5.10

Число процессов М	Время виз.(с.)	k(N)	e	t0/t1
1	119,3			
40	0,09	8,55	53%	0,014

На основании анализа данных табл.5.10 можно сделать следующие выводы.

1. Постоянная составляющая $t_0 = 0.01$ с. Следует заметить, что в постоянную составляющую в данном случае входило время пересылки данных (матрицы преобразований) с IBM PC на все рабочие транспьютеры. Насыщение ускорения наступит при $N \gg 71$.

2. Из формулы (4) можно вычислить необходимое число транспьютеров для получения вывода изображений данной сцены в реальном времени: частота вывода 25 кадров/с.; время вывода 0.04 с. Расчетное значение - 32 транспьютера.

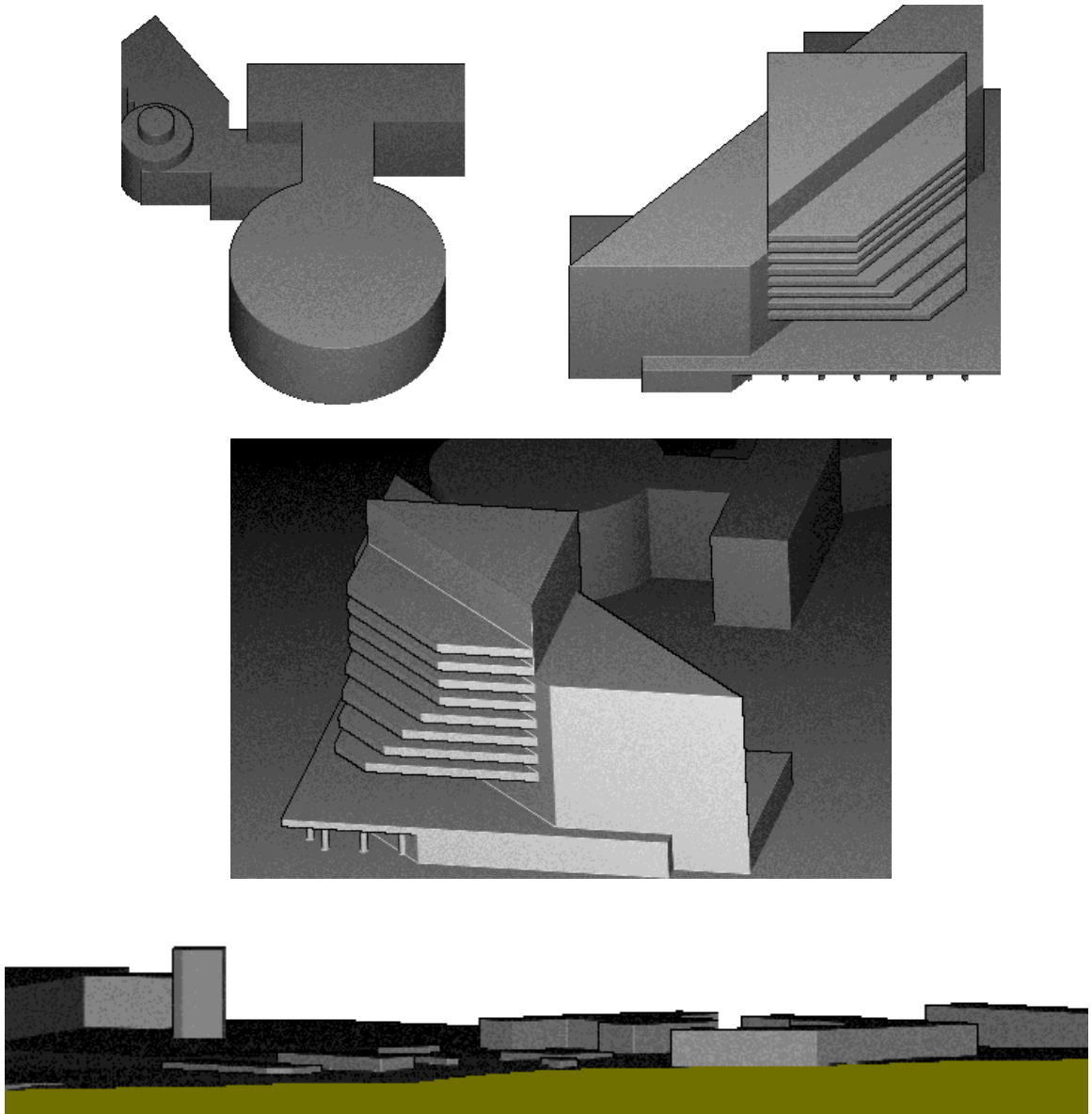


Рис.5.20

Выводы по пятой главе

В качестве практического воплощения концепции Описан программный комплекс расчетно-графического программирования в

САПР, при разработке и реализации которого использованы принципы и подходы графического моделирования, изложенные в главах 1-4.

1. Система "Вектор" является программным комплексом моделирования сложных геометрических форм различной размерности. Она работает под управлением системной программы "Монитор графического диалога", обеспечивающей ведение графического диалога, создание сценария диалога, обработку ошибок, выполнение простейших расчетов и программирование на уровне пользователя, не являющегося квалифицированным программистом.

2. Система "Вектор" организована по модульному принципу. Это обусловлено, во-первых, необходимостью оптимального использования ограниченного объема оперативной памяти РС IBM и, во-вторых, выполнением условия открытости системы.

3. Система "Вектор" состоит из модулей инвариантного и проблемно-ориентированного назначений, что обеспечивает, с одной стороны, дальнейшее развитие комплекса и, с другой стороны, выполнение производственных заказов по проектированию, в частности, корпусов судов и их конструкций.

4. На основе транслятора YACC реализован язык пользователя "Калькулятор", который представляет собой расширение языка программирования СИ. Язык "Калькулятор" ориентирован на эксплуатацию системы пользователем, не обладающим квалификацией программиста.

5. Другим компонентом комплекса расчетно-графического программирования является система твердотельного моделирования CG - совместная разработка с сотрудниками ИАПУ ДВО РАН. Эта система обеспечивает моделирование реалистических сцен и объектов по геометрическим примитивам самой системы, а также поверхностям, генерируемым модулями системы "Вектор". Связь между системами CG и модулями многомерного моделирования системы "Вектор" осуществляется на уровне обмена файлами. Модули Vec_gran и Vec_con системы "Вектор" подключены непосредственно к системе CG, образуя новые модули CG_Vec_Gran и CG_Vec_Con. Такая структура ПО позволяет наиболее эффективно использовать возможности векторного и растрового моделирования.

6. Связь комплекса с системой AutoCAD обеспечивает возможность использования сервиса этой системы для оформления чертежей моделирования объектов и вывода их на плоттер.

7. Предусмотрена возможность передачи информации в модуль Vec_Calc о геометрических образах из любого модуля системы "Вектор" (или автономной программы) в виде массива точек для расчета различных дифференциально-геометрических (например, расчет площади, объема, построение эквидистанты кривой или поверхности, построение геодезических линий и т.п.) характеристик моделируемых форм, а также для получения всевозможных преобразований и изображений с выводом их на плоттер.

8. Все инвариантные и проблемно-ориентированные модули систем "Вектор" и "CG_Вектор" имеют собственный язык пользователя, могут использовать линии из базы данных модуля "Аппарат конструктора" в качестве формообразующих линий, а также некоторые возможности полуавтоматического распознавания линий по их двум проекциям для формообразующих линий при генерировании геометрических форм.

9. Предложен перспективный шаг в развитии комплекса - методы распараллеливания алгоритмов на транспьютерной основе. Первый этап исследований и первые практические результаты в виде действующей программной системы твердотельного моделирования TCG показали, что реализация параллельной обработки графической информации на транспьютерной основе обеспечила высокую скорость визуализации - до 16 несложных сцен в сек.

10. Эти же исследования позволили выявить пути ускорения визуализации сцен:

- для достаточно простых сцен (типа сцены 3) режим реального времени достигается простым увеличением числа транспьютеров в сети;

- для повышения эффективности распараллеливания, особенно для сложных сцен, необходимо уменьшать постоянную составляющую t_0 за счет распараллеливания геометрического преобразования и вычисления глобальных характеристик сцены либо за счет выполнения их на высокоскоростном геометрическом процессоре;

- для уменьшения влияния неоднородности сцены, что, в свою очередь, приведет к повышению эффективности, необходимо

использовать метод адаптивного разбиения экрана на горизонтальные полосы.

ОСНОВНЫЕ ВЫВОДЫ И РЕЗУЛЬТАТЫ

Общепризнан факт, что осуществление процессов проектирования и технологической подготовки производства в графической среде способствует сокращению временных и материальных затрат, повышению качества этих процессов, поскольку в графической среде на компьютере можно быстро сгенерировать множество проектных решений и из них выбрать оптимальное, не прибегая к эмпирическому методу "проб и ошибок" в натуре.

В тоже время менее очевидна эффективность использования графических методов анализа и решения задач линейного и нелинейного программирования

В настоящей диссертационной работе представлен опыт совместного решения этих задач в плане создания математического и программного обеспечения, обладающего интегрированными возможностями расчетно-графического моделирования в инженерной деятельности.

Так, описанная в диссертационной работе система "Вектор": 1) обладает достаточно полным набором развитых графических возможностей для моделирования геометрических форм различной размерности; 2) имеет язык пользователя, обеспечивающий возможность осуществления алгебраических вычислений, манипулирования графическими операциями, обращения к программам, написанными на других языках; 3) имеет файловую и графические базы данных, средства диагностики работы программ, программы-сценарии ведения диалога; 4) является модульной, вследствие чего любой модуль в отдельности может быть поставлен на ПК с меньшими возможностями; 5) является открытой, т.е. имеется возможность дополнить ее новыми базовыми возможностями, вплоть до узкой специализации по тому или иному кругу задач.

В диссертации разработан и реализован в программном геометрическом комплексе алгоритм структурно-клеточного представления наиболее сложных геометрических форм двух, трех и более измерений. Разработана интерактивно-графическая система

моделирования многомерных объектов по их аксонометрическим и ортогональным проекциям. Сочетание в графической системе этого алгоритмического обеспечения с хорошим меню, языком элементарных арифметических, тригонометрических и логических операций, базой данных хранения графической информации, визуализацией моделируемых объектов в реалистических сценах дает возможность эффективно решать следующие задачи:

- генерировать внешние и внутренние формы корпуса судна;
- производить компоновку и размещение внутреннего его насыщения;
- моделировать многомерные процессы функциональных зависимостей с изображением их на экране дисплея;
- проводить различные топологические и конформные преобразования и, соответственно, исследовать модель в задачах полного их обтекания, распределения прочностных характеристик;
- изображать рельефы комплексных функций и решать с ними всевозможные прикладные задачи.

Система "Вектор" является также удобным инструментом для овладения методами геометрического программирования и алгоритмизации, элементами векторной и линейной алгебры, аналитической геометрии, комплексных чисел, дифференциального исчисления и оптимального управления.

В модулях системы "Вектор" скомпонованы программные возможности языка СИ и графические операции для решения инженерных задач и проработки научных идей. Программное обеспечение построено таким образом, что пользователю представляется возможность концентрировать свое внимание на постановке и анализе решаемой задачи и меньше отвлекаться на поиск прямых методов решения задач и технику программирования. Входной язык прост для понимания, весь ход интерактивного процесса решения задач оформляется в библиотеку макрокоманд, которые могут быть в дальнейшем использованы в других задачах.