Министерство образования республики Беларусь

Белорусский национальный технический университет

Факультет информационных технологий и робототехники

Кафедра «Системы автоматизированного проектирования»

КУРСОВОЙ ПРОЕКТ ПО ДИСЦИПЛИНЕ:

«3D моделирование»

на тему: *“ Получение реалистичного изображения 3-х мерного объекта, представляющего собой полигональную сетку, его преобразование и проецирование”*

Исполнитель: студент гр.10702314
Сухоцкий М.А.

Руководитель: ст. преподаватель
Сиденко Л. А.

Минск 2016

Оглавление

Введение

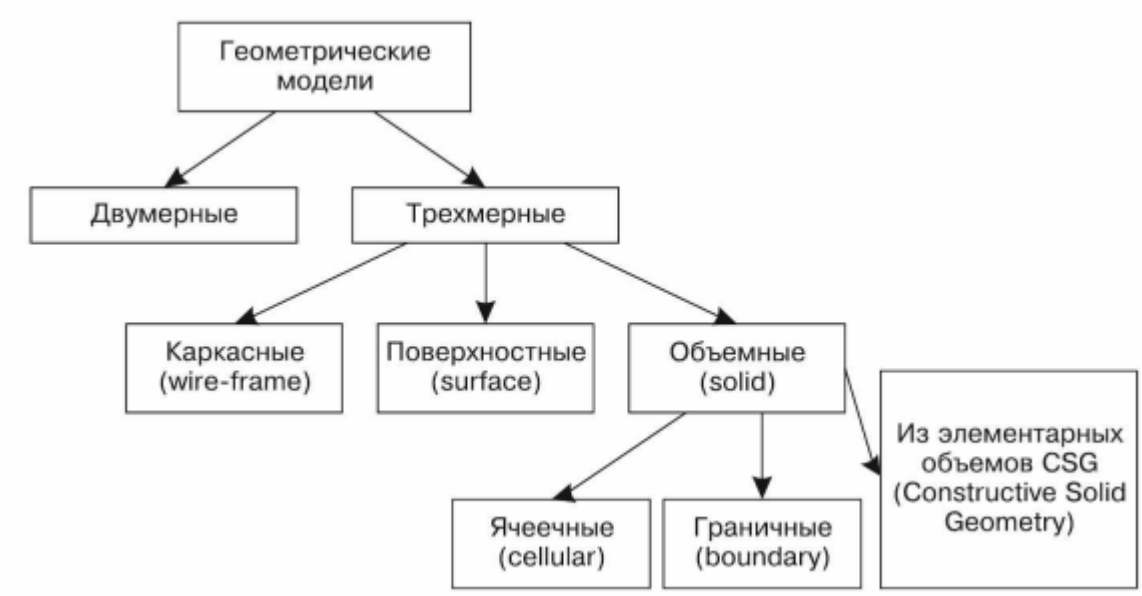
В последнее время, когда развитие компьютерных технологий достигло значительных результатов, все чаще обращаются к задачам моделирования трехмерных объектов. Почти все программное обеспечение, применяемое в машиностроении, архитектуре и других отраслях производства, содержит в себе приемы трехмерного моделирования.

Задача трехмерного моделирования состоит в том, что нужно определить структуру данных, описывающую трехмерное тело, на основе полученной структуры получить возможность изменять объект, изменять его положение в пространстве. Также к задачам трехмерного моделирования относятся задачи проекционного черчения, реализации реалистического изображения объекта с учетом таких факторов как свет, материал, шероховатость поверхностей и т.д.

Решение этой задачи облегчает работу пользователя (конструктора, инженера), связанную с проектированием каких-либо объектов, т.к. традиционное проекционное черчение не дает реального изображения предмета. Реальное изображение объекта легче воспринимается человеком, что делает работу более комфортной, а, следовательно, и более продуктивной.

Все сказанное доказывает то, что трехмерное моделирование это одна из самых важных задач, решаемых в САПР сегодня.

В 3D моделировании различают следующие модели:



- каркасная – представляет собой форму детали в виде конечного множества линий, лежащих на поверхности детали. Для каждой линии

известны координаты концевых точек и указана их инцидентность ребрам или поверхностям;

- поверхностная – отображает форму детали с помощью задания ограничивающих ее поверхностей, например, в виде совокупности данных о гранях, ребрах и вершинах;
- объемные – отличаются тем, что в них в явной форме содержатся сведения о принадлежности элементов внутреннему или внешнему по отношению к детали пространству.

Применяют следующие подходы к построению геометрических моделей:

- задание граничных элементов – граней, ребер, вершин;
- кинематический метод, согласно которому задают двумерный контур и траекторию его перемещения; след от перемещения контура принимают в качестве поверхности детали;
- позиционный подход, в соответствии с которым рассматриваемое пространство разбивают на ячейки (позиции) и деталь задают указанием ячеек, принадлежащих детали; данный метод является очень громоздким и используется редко;
- метод конструктивной геометрии – представление сложной детали в виде совокупностей базовых элементов формы и выполняемых над ними теоретико-множественных операций.

1.Описание объекта и получение ГМ

Целью курсового проекта является разработка приложения, позволяющего получать реалистичное изображения 3-х мерного объекта, представляющего собой полигональную сетку, его преобразование и его проецирование. Объект представляет собой трехмерную модель сферы. Язык исполняемого приложения – C#.
Исходные данные к проекту: сфера радиуса R

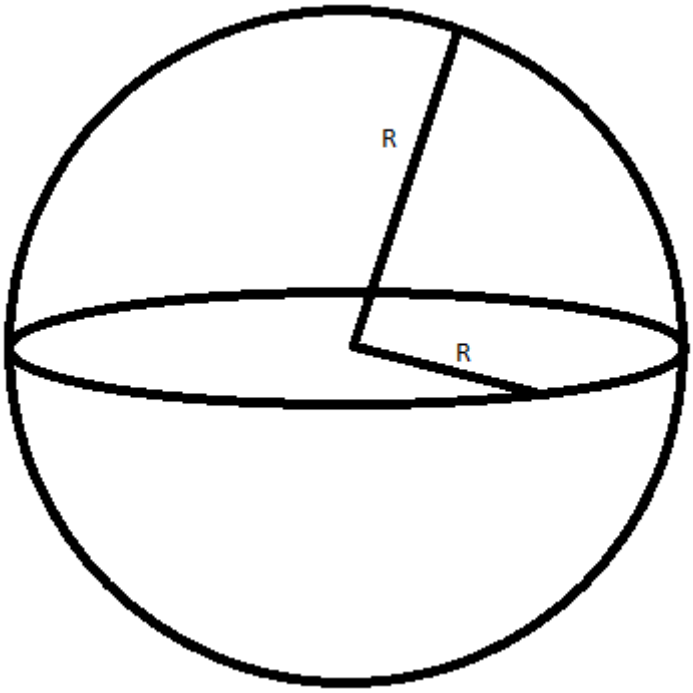


Рис. 1.1 – сфера

R– радиус сферы;

Объект представляет собой набор граней. Грань в свою очередь это набор рёбер, ребро - состоит их двух точек (начальная и конечная), точка описывается тремя координатами (x, y, z). Указав необходимые параметры (размеры) фигур, происходит расчёт координат вершин.

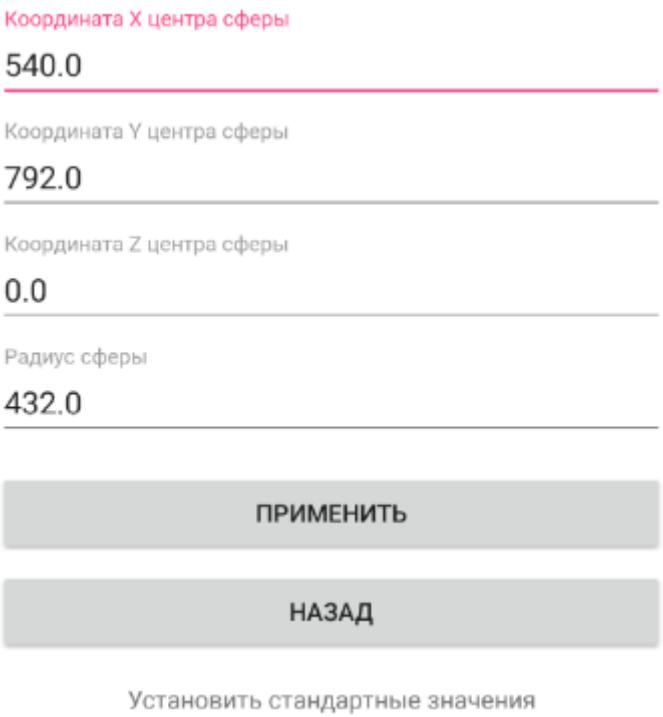


Рис.1.2 Значение параметров модели

В зависимости от значения аппроксимации, количество точек будет изменяться. С помощью геометрических формул получим X, Y и Z - координаты узлов для исходного положения объектов. За исходное положение примем нулевые смещения по всем осям и нулевые значения углов поворота объекта вокруг каждой из координатных осей.

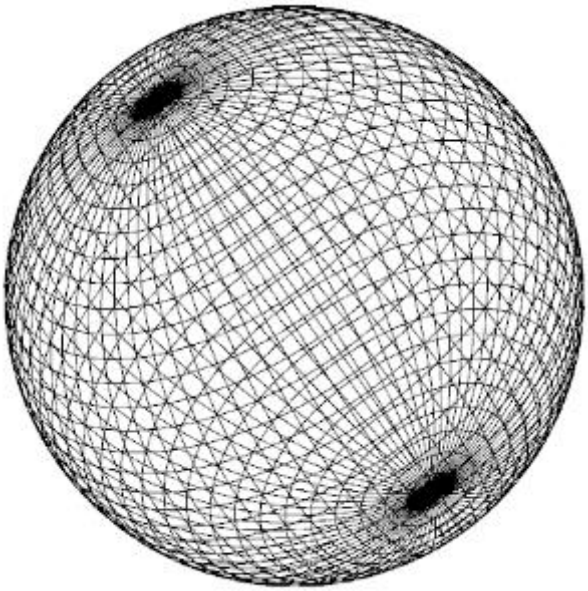


Рис. 1.3. Вид 3-х мерного объекта

2. Геометрические преобразования модели

К геометрическим преобразованиям относятся перемещение, поворот и масштабирование.

Геометрические преобразования ведутся в однородных координатах, где каждая точка имеет вид:

$$P = (W \cdot x, W \cdot y, W \cdot z, W) = (x, y, z, 1)$$

При таком представлении удобно оперировать большим числом вершин. Преобразование объекта в однородных системах координат ведется путем умножения каждой точки на матрицу преобразования. Все геометрические преобразования в трехмерном пространстве выполняются относительно центра системы координат.

2.1. Перенос

Точки на плоскости можно перенести в новые позиции путем добавления к координатам этих точек констант переноса. Матрица переноса имеет следующий вид.

$$P' = P \cdot T(Dx, Dy, Dz)$$

$$T(Dx, Dy, Dz) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ Dx & Dy & Dz & 1 \end{bmatrix}$$

P – начальная точка,

P’ – точка после выполнения переноса,

T – матрица переноса,

Dx – значение переноса относительно оси OX.

Dy – значение переноса относительно оси OY.

Dz – значение переноса относительно оси OZ.

2.2. Поворот

Точки могут быть повернуты на определенный угол относительно начала координат.

$$P' = P \cdot R(\varphi)$$

$$Rz(\varphi) = \begin{bmatrix} \cos\varphi & \sin\varphi & 0 & 0 \\ -\sin\varphi & \cos\varphi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad Rx(\varphi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi & 0 \\ 0 & -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$Ry(\varphi) = \begin{bmatrix} \cos\varphi & 0 & -\sin\varphi & 0 \\ 0 & 1 & 0 & 0 \\ \sin\varphi & 0 & \cos\varphi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

P – начальная точка,

P’ – точка после выполнения поворота,

Rz – матрица поворота вокруг оси OZ,

Rx – матрица поворота вокруг оси OX,

Ry – матрица поворота вокруг оси OY,

φ – угол поворота.

2.3. Масштабирование

Масштабирование производится относительно начала координат. Матрица масштабирования выглядит следующим образом

$$P' = P \cdot S(Sx, Sy, Sz)$$

$$S(Sx, Sy, Sz) = \begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

P – начальная точка,

P’ – точка после выполнения масштабирования,

Sx - коэффициент масштабирования относительно оси OX.

Sy - коэффициент масштабирования относительно оси OY.

Sz - коэффициент масштабирования относительно оси OZ.

3. Проецирование

В общем случае проецирование – это преобразование n-мерного пространства в m-мерное, где n > m. Мы же под проецированием будем понимать преобразование 3-хмерного пространства в 2-мерное.

В зависимости от способа проецирования различают *центральные* и *параллельные* проекции. Параллельные в зависимости от соотношения между направлением проецирования и нормалью к проецируемой плоскости делятся на *прямоугольные* и *косоугольные*. К прямоугольным проекциям относятся *ортографические*, *аксонометрические* проекции.

Параллельные проекции названы так потому, что центр проекции бесконечно удален и все проецирующие лучи параллельны. При описании центральной проекции мы явно задаем ее центр проекции, в то время как, определяя параллельную проекцию, мы указываем направление проецирования. Поскольку проекция отрезка сама является отрезком, то достаточно спроецировать одни лишь конечные точки и соединить их.

Центральная проекция порождает визуальный эффект, аналогичный тому, к которому приводят фотографические системы или зрительная система человека, и поэтому используется в случаях, когда желательно достичь определенной степени реалистичности. Этот эффект называется перспективным укорачиванием: по мере увеличения расстояния от центра до объекта размер получаемой проекции уменьшается. Это, с другой стороны, означает, что хотя центральная проекция объектов является реалистичной, она оказывается непригодной для представления точной формы и размеров объектов: из проекции нельзя получить информацию об относительных расстояниях; углы сохраняются только на тех гранях объекта, которые параллельны проекционной плоскости; проекции параллельных линий в общем случае не параллельны.

Для построения ортографических проекций выбираются различные двухмерные системы координат: для фронтальной – Y(X), для горизонтальной – Z(X), для профильной – Y(Z).

Все проекции получаются перемножением векторов точек объекта в однородных координатах на матрицы соответствующих преобразований. Двумерный вектор (x, y)в однородных координатах записывается в виде (wx, wy, w), где $w \neq 0$. Число w называется масштабным множителем. Для того, чтобы из вектора, записанного в однородных координатах получить вектор в обычных координатах необходимо разделить первые две координаты на третью: $(wx / w, wy / w, w / w) \rightarrow (x, y, 1)$. Трехмерный вектор записывается аналогично, но после x,y добавляется координата z.

Параллельная проекция порождает менее реалистичное изображение, поскольку отсутствует перспективное укорачивание, хотя при этом могут иметь место различные постоянные укорачивания вдоль каждой из осей.

Проекция фиксирует истинные размеры (с точностью до скалярного множителя), и параллельные прямые остаются параллельными. Как и в случае центральной проекции, углы сохраняются только на тех гранях объекта, которые параллельны проекционной плоскости.

Параллельные проекции разделяются на два типа в зависимости от соотношения между направлением проецирования и нормалью к проекционной плоскости. Если эти направления совпадают, т.е. направление проецирования является нормалью к проекционной плоскости, то проекция называется ортографической. Если же проекторы не ортогональны к проекционной плоскости, то проекция называется косоугольной.

В инженерной графике наиболее широко используемыми видами ортографических проекций являются вид спереди, вид сверху и вид сбоку, в которых проекционная плоскость перпендикулярна главным координатным осям, совпадающим вследствие этого с направлением проецирования. Поскольку каждая проекция отображает лишь одну сторону объекта, часто совсем непросто представить себе пространственную структуру проецируемого объекта, даже если рассматривать сразу несколько проекций одного и того же объекта. Но тем не менее такие чертежи позволяют определять реальные размеры объекта.

3.1. Фронтальная проекция

Для получения фронтальной проекции все точки модели объекта необходимо умножить на матрицу фронтальной проекции:

$$M_{фр} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.2. Горизонтальная проекция

Для получения горизонтальной проекции все точки модели объекта необходимо умножить на матрицу горизонтальной проекции:

$$M_{гор} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.3. Профильная проекция

Для получения профильной проекции все точки модели объекта необходимо умножить на матрицу профильной проекции:

$$M_{проф} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Примеры ортографических проекций:

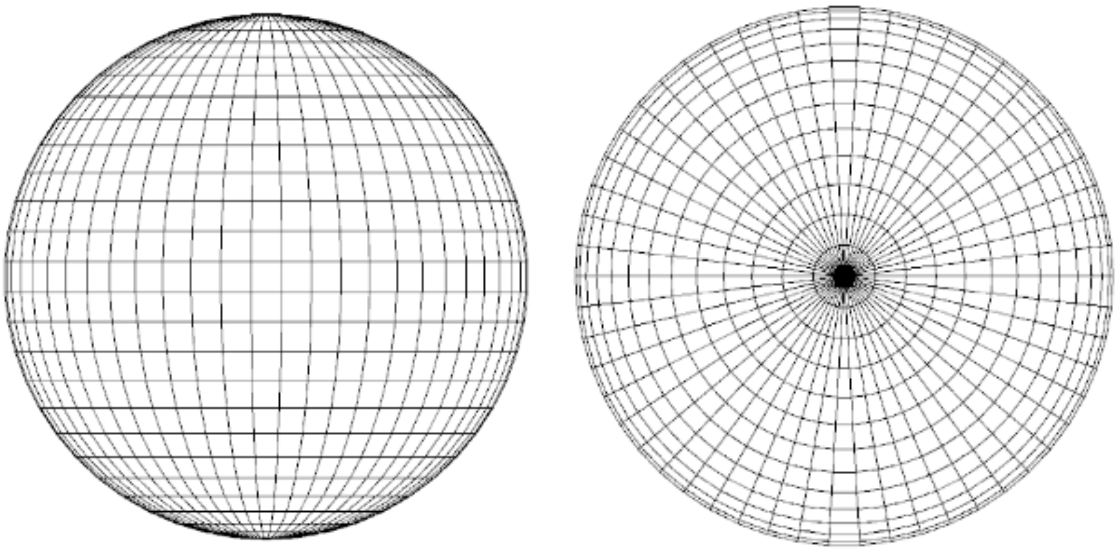


Рис.3.1 – Ортографические проекции

3.4. Аксонометрическая проекция

В случае аксонометрических проекций используются проекционные плоскости, не перпендикулярные главным координатным осям, поэтому на них изображается сразу несколько сторон объекта, так же как и при центральном проецировании, однако в аксонометрии укорачивание постоянно, тогда как в случае центральной проекции оно связано с расстоянием от центра проекции. При аксонометрическом проецировании сохраняется параллельность прямых, а углы изменяются; расстояния же можно измерить вдоль каждой из главных координатных осей (в общем случае с различными масштабными коэффициентами).

Для построения аксонометрической проекции необходимо задать два угла: φ и ψ, которые отвечают за повороты относительно осей X и У. Далее

все точки модели объекта необходимо умножить на матрицу аксонометрической проекции:

$$M_{акс} = \begin{bmatrix} \cos\psi & \sin\varphi\sin\psi & 0 & 0 \\ 0 & \cos\varphi & 0 & 0 \\ \sin\psi & -\sin\varphi\cos\psi & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Пример аксонометрической проекции для углов $\varphi = 45$ и $\psi = 35$ на рис. 3.2.

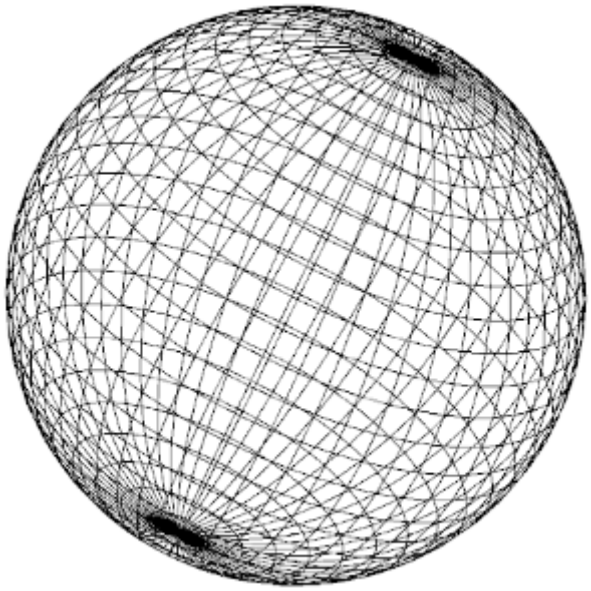


Рис.3.2 – Аксонометрическая проекция

3.5. Косоугольная проекция

Косоугольные проекции также являются параллельными, причем проекционная плоскость перпендикулярна главной координатной оси. Сторона объекта, параллельная этой плоскости, проецируется так, что можно измерять углы и расстояния. Проецирование других сторон объекта также допускает проведение линейных измерений (но не угловых) вдоль главных осей.

Для построения косоугольной проекции необходимо задать L и α. Далее все точки модели объекта необходимо умножить на матрицу косоугольной проекции:

$$M_{кос} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ L\cos\alpha & L\sin\alpha & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

где α – угол между осью OX и проекцией оси OZ на плоскость XOY, а l – масштабный коэффициент.

Пример косоугольной проекции для L= 1 и α =30:

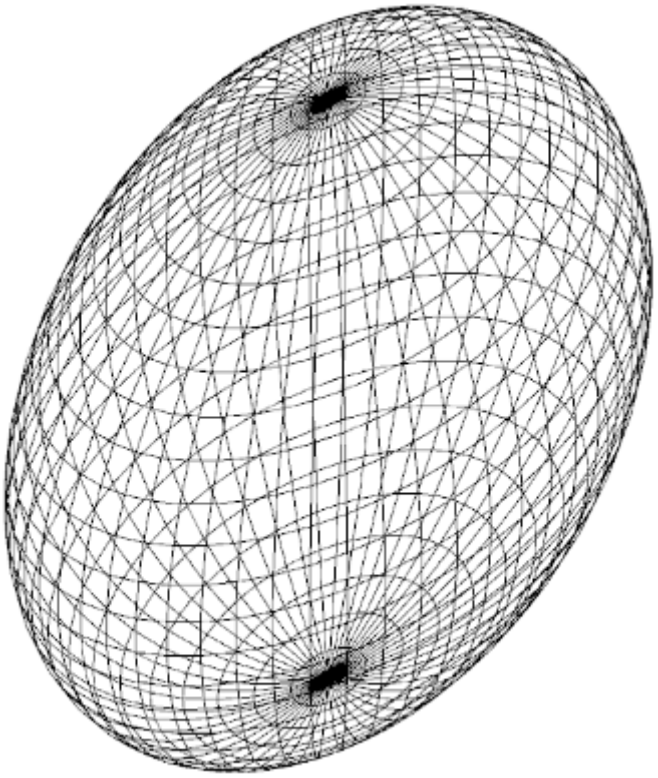


Рис.3.3 – Косоугольная проекция

3.6. Перспективная проекция

3.6.1. Без видового преобразования

Когда пучок проецирующих лучей исходит из заданного центра проекции, то параллельные отрезки на плоскости проекции уже не будут параллельными, за исключением случая, когда они лежат в плоскости, параллельной проекционной. При проецировании нескольких параллельных прямых их проекции пересекаются в так называемой точке схода. Если совокупность прямых параллельна одной из координатных осей, то их точка схода называется главной. Таких точек может быть не

больше трех. Основное свойство центральных проекций заключается в том, что более удаленные предметы отображаются в меньшем масштабе.

Расстояние от точки зрения до картинной плоскости называется фокусным (d).

$$M_{пер} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1/d \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$P' = P \cdot M_{пер} = [x \ y \ z \ 1] \cdot M_{пер} = [x \ y \ z \ z/d] = [\frac{x}{z/d} \ \ \frac{y}{z/d} \ \ d \ 1]$$

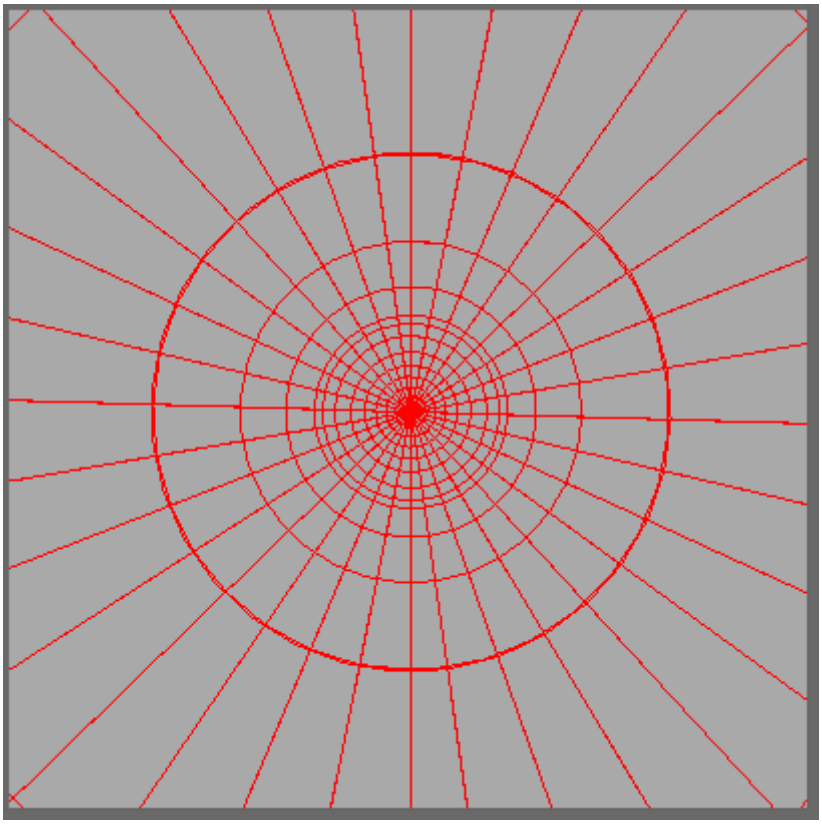


Рис.3.4 – Пример перспективной проекции (точка зрения внутри объекта).

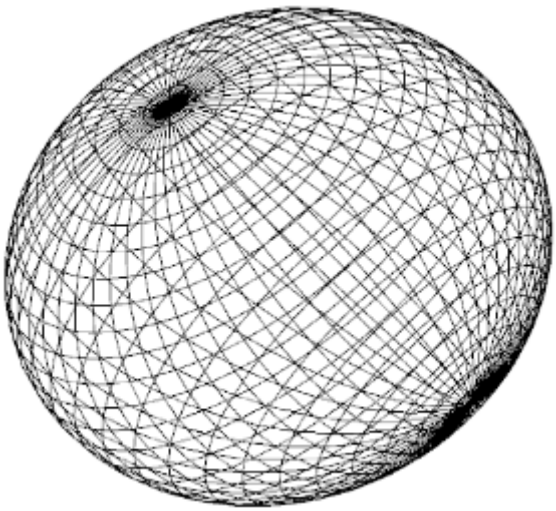


Рис.3.5 – Пример перспективной проекции (точка зрения вне объекта).

3.6.2. С видовым преобразованием

Все предыдущие проекции были ограничены строгим размещением картинной плоскости и центром проекции. Видовые преобразования позволяют размещать картинную плоскость относительно объекта произвольно. По сути, это преобразование системы координат.

Координаты объекта в видовой системе координат:

$$[x_E\, y_E\, z_E\, 1] = [x\ \ y\ \ z\ \ 1] \cdot V$$

, где

V – матрица видового преобразования,

x_E, y_E, z_E – *координаты объекта в видовой системе координат*

Для видового преобразования необходимо выполнить следующие действия:

1. Перенос:

$$T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -X_E & -Y_E & -Z_E & 1 \end{pmatrix}$$

, где

Xe, Ye, Ze – координаты точки зрения.

2. Поворот системы координат вокруг оси я на угол ($\frac{\pi}{2}-\theta$) по часовой стрелке:

$$Rz=\begin{bmatrix} \cos(\frac{\pi}{2}-\theta) & \sin(\frac{\pi}{2}-\theta) & 0 & 0 \\ -\sin(\frac{\pi}{2}-\theta) & \cos(\frac{\pi}{2}-\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3. Поворот вокруг оси X на угол ($\pi-\varphi$) против часовой стрелки:

$$Rx=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\varphi-\pi) & \sin(\varphi-\pi) & 0 \\ 0 & -\sin(\varphi-\pi) & \cos(\varphi-\pi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4. Изменить направление оси OX:

$$S=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

В результате результирующая матрица видового преобразования примет вид:

$$V=T\cdot R_z\cdot R_x\cdot S$$

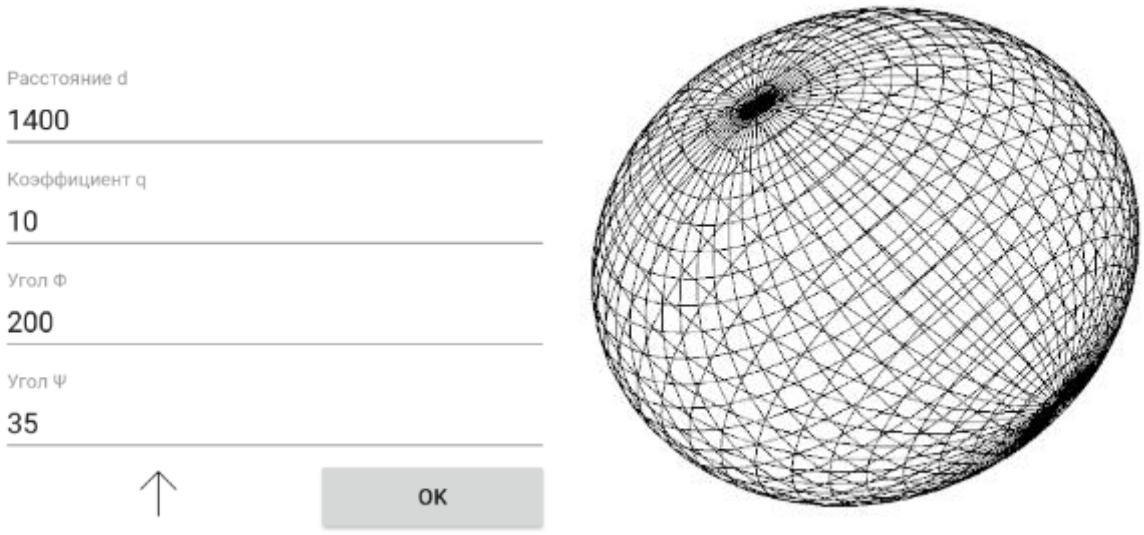


Рис.3.5 – Пример перспективной проекции с видовым преобразование.

4. Удаление невидимых линий

Сложность задачи удаления невидимых линий и поверхностей привела к появлению большого числа различных способов ее решения. Многие из них ориентированы на специализированные приложения. Единого решения этой задачи, годного для различных случаев, естественно, не существует: для каждого случая выбирается наиболее подходящий метод. Например, для моделирования процессов в реальном времени требуются быстрые алгоритмы, в то время как для формирования сложного реалистического изображения, в котором представлены тени, прозрачность и фактура, учитывающие эффекты отражения и преломления цвета в мельчайших оттенках, фактор времени выполнения уже не так существенен. Подобные алгоритмы работают медленно, и зачастую на вычисления требуется несколько минут или даже часов. Существует тесная взаимосвязь между скоростью работы алгоритма и детальностью его результата. Ни один из алгоритмов не может достигнуть хороших оценок для этих двух показателей одновременно.

Для удаления невидимых линий используется алгоритм Робертса. Он представляет собой первое известное решение задачи об удалении невидимых линий. Это математически элегантный метод, работающий в объектном пространстве. Алгоритм прежде всего удаляет из каждого тела те ребра или грани, которые экранируются самим телом. Затем каждое из видимых ребер каждого тела сравнивается с каждым из оставшихся тел для определения того, какая его часть или части, если таковые есть, экранируются этими телами. Поэтому вычислительная трудоемкость алгоритма Робертса растет теоретически, как квадрат числа объектов. Это в сочетании с ростом интереса к растровым дисплеям, работающим в пространстве изображения, привело к снижению интереса к алгоритму Робертса. Однако математические методы, используемые в этом алгоритме, просты, мощны и точны. Кроме того, этот алгоритм можно использовать для иллюстрации некоторых важных концепций. Наконец, более поздние реализации алгоритма, использующие предварительную приоритетную сортировку вдоль оси z и простые габаритные или минимаксные тесты, демонстрируют почти линейную зависимость от числа объектов.

Определение нелицевых граней

Пусть F — некоторая грань многогранника. Плоскость, несущая эту грань, разделяет пространство на два подпространства. Назовем положительным то из них, в которое смотрит внешняя нормаль к

грани. Если точка наблюдения – в положительном подпространстве, то грань – **лицевая**, в противном случае – **нелицевая**. Если многогранник выпуклый, то удаление всех нелицевых граней полностью решает задачу визуализации с удалением невидимых граней.

Для определения, лежит ли точка в положительном подпространстве, используют проверку знака скалярного произведения (l, n), где l – вектор, направленный к наблюдателю, фактически определяет точку наблюдения; n – вектор внешней нормали грани. Если $(l, n) > 0$, т. е. угол между векторами острый, то грань является лицевой. Если $(l, n) < 0$, т. е. угол между векторами тупой, то грань является нелицевой.

В алгоритме Робертса требуется, чтобы все изображаемые тела или объекты были выпуклыми. А значит идеально подходит для сферы.

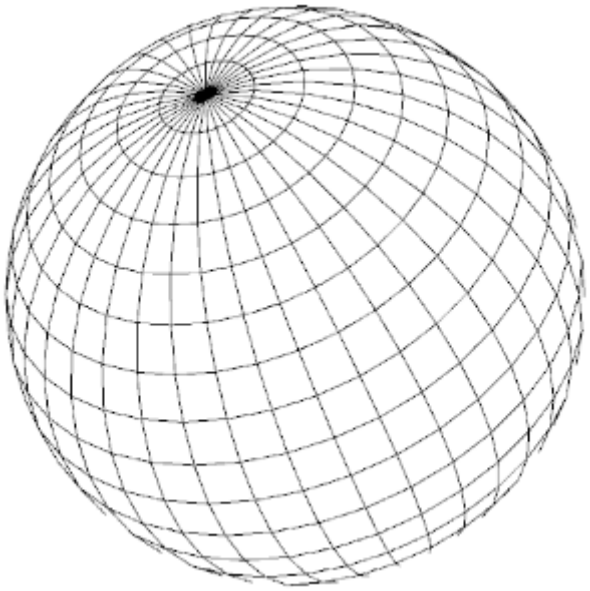


Рис.4.1 – Удаление невидимых линий

5. Закраска изображения с учётом освещения

Рассмотрим, как можно определить цвет пикселей изображения поверхности согласно взаимному расположению поверхности и источника света и наблюдателя.

Диффузное отражение**.** Этот вид отражения присущ *матовым* поверхностям. Матовой можно считать такую поверхность, размер шероховатостей которой уже настолько велик, что падающий луч рассеивается равномерно во все стороны. Такой тип отражения характерен, например, для гипса, песка бумаги. Диффузное отражение описывается законом Ламберта, согласно которому интенсивность отраженного света пропорциональна косинусу угла между направлением на точечный источник света и нормалью к поверхности.

Рис. 5.1.

Рассеянный свет. Ia*Ka. В итоге мы получаем значение интенсивности света:

$$I = I_a k_a + I_L k_d \cos\theta$$

k_d – коэффициент диффузного отражения (k_d=1);

I_L – интенсивность источника света;

I_a – интенсивность отраженного света;

k_a – коэффициент диффузного отражения рассеянного света;

θ – угол между направлением на источник света и нормалью к поверхности.

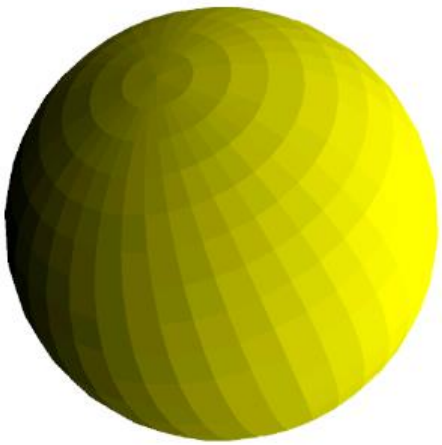


Рис. 5.2. Освещение (источник света справа от объекта)

Расчет интенсивностей выполняется при отрисовке полигонов.
Нормаль рассчитывается как вектор:

$$\vec{a} = (x, y, z), \text{ где}$$

$$x = y_1z_2 + y_2z_3 + y_3z_1 - y_2z_1 - y_3z_2 - y_1z_3$$

$$y = z_1x_2 + z_2x_3 + z_3x_1 - z_2x_1 - z_3x_2 - z_1x_3$$

$$z = x_1y_2 + x_2y_3 + x_3y_1 - x_2y_1 - x_3y_2 - x_1y_3, \text{ где}$$

$(x_1y_1z_1), (x_2y_2z_2), (x_3y_3z_3)$ – координаты трех точек грани.

Координаты центральной точки грани рассчитывается, как среднее арифметическое соответствующих координат всех точек грани.
Последним шагом является вычисление:

$$\cos(\vec{a}\vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| \cdot |\vec{b}|}, \text{ где}$$

$\vec{b}$ - вектор от источника света; $\vec{a} \cdot \vec{b} = x \cdot x_L + y \cdot y_L + z \cdot z_L$;

$|\vec{a}|$ и $|\vec{b}|$ – длины векторов $\vec{a}$ и $\vec{b}$.

Программная реализация: координаты нормали находятся следующим образом:

```
public Vector
getVectorNormal()
{
                        Vector prevVector = getCrossProduct(new
Vector(points.get(0), points.get(1)), new Vector(points.get(0),
points.get(2)));
                        Point3D C = new Point3D(points.get(0).x + prevVector.x,
points.get(0).y + prevVector.y, points.get(0).z + prevVector.z);
                        Point3D H = points.get(0);
                        Point3D C2 = new Point3D(2 * H.x - C.x, 2 * H.y - C.y, 2 *
H.z - C.z);
                        double first = getDistBetwTwoPoints3D(center, C);
                        double second = getDistBetwTwoPoints3D(center, C2);
                        if (first > second) {
                            return new Vector(points.get(0), C);
                        } else {
                            return new Vector(points.get(0), C2);
                        }
                    }
```

Параллельно вычисляются координаты средней точки грани:

```
public static Point3D
getMassCenter(List<Point3D>
points) {
                                    double xSum = 0, ySum = 0, zSum = 0;
                                    for (Point3D point : points) {
                                        xSum += point.x;
                                        ySum += point.y;
                                        zSum += point.z;
                                    }
                                    return new Point3D(xSum / points.size(), ySum /
                                points.size(), zSum / points.size());
                                }
```

Вычисляются значение косинуса θ:

```
public static Vector
getCrossProduct(Vector
v1, Vector v2) {
                                return new Vector(
                                        v1.y * v2.z - v1.z * v2.y,
                                        v1.z * v2.x - v1.x * v2.z,
                                        v1.x * v2.y - v1.y * v2.x
                                );
                            }
```

Теперь когда все данные есть, вычисляется интенсивность цвета грани:

```
public double
getLightCoefficient()
{
                                    lightCoefficient = 0.5 + 0.9 * 0.6 *
                            getDotProduct(vectorLight, vectorNormal) / (vectorLight.length *
                            vectorNormal.length);
                                    lightCoefficient = lightCoefficient > 1 ? 1 :
                            lightCoefficient;
                                    lightCoefficient = lightCoefficient < 0 ? 0 :
                            lightCoefficient;
                                    return lightCoefficient;
                                }
```

Заключение

В результате курсового проекта поставленная задача была выполнена успешно.

Были решены следующие задачи:

- получена 3-х мерная геометрическая модель объекта, ее отображение и проекции;
- создано программное приложение, которое обеспечивает задание параметров геометрического объекта пользователем;
- реализован алгоритм удаления невидимых линий;
- реализован алгоритм создания реалистичного освещения объекта;
- создан простой и интуитивно понятный пользовательский интерфейс.

Литература

1. Сиденко Л.А. Компьютерная графика и геометрическое моделирование.— СПб.: Питер, 2008.
2. Н.Н.Голованов Геометрическое моделирование. Москва Издательство «Физматлит» 2002
3. Конспект лекций.