МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
Кафедра «Комп'ютерної інженерії»

МЕТОДИЧНІ ВКАЗІВКИ

щодо організації самостійної роботи студентів при виконанні індивідуальних завдань з дисципліни «Апаратно-програмні засоби систем комп'ютерної графіки»

(для студентів заочної, заочної прискореної з наданням денних освітніх послуг і денної форм навчання напряму підготовки 6.050102 «Комп'ютерна інженерія»)

Розглянуто на засіданні кафедри комп'ютерна інженерія
__31_ _*серпня*_2010 р., протокол №_1__
Завідувач кафедри, д.т.н., проф.
_____ *В.А. Святний*

Затверджено на засіданні навчально-методичної комісії напряму підготовки 6.050102 "Комп'ютерна інженерія"
(протокол №_1_від _31_ _*серпня*_2012 р.)

Голова комісії, д.т.н., проф..
_____ *В.А. Святний*

Донецьк – 2012

УДК 681.3

Методичні вказівки щодо організації самостійної роботи студентів при виконанні індивідуальних завдань з дисципліни «Апаратно-програмні засоби систем комп'ютерної графіки» (для студентів заочної, заочної прискореної з наданням денних освітніх послуг і денної форм навчання напряму підготовки 6.050102 «Комп'ютерна інженерія») / Уклад.: Р.В. Мальчева. - Донецьк, ДонНТУ, 2013. - 24 с.

Наведені методичні вказівки щодо виконання розрахунково-графічних та контрольних робіт, структура, варіанти завдань та необхідний довідковий матеріал.

Надані приклади та опис апаратурної реалізації етапів базового алгоритму синтезу зображень. Наведений приклад використання системи моделювання Active HDL для перевірки роботи функціональної схеми для заданого етапу синтезу.

Призначений для студентів заочної, заочної прискореної з наданням денних освітніх послуг і денної форм навчання напряму підготовки 6.050102 «Комп'ютерна інженерія».

Укладач: Р.В. Мальчева, доцент

Рецензент: Н.С. Костюкова, доцент

Відпов. за випуск: В.А. Святний, зав. каф., проф.

ВСТУП

Основною задачею курсу є систематизація та придбання знань і навичок студентами по засвоєнню алгоритмів створення та відображення графічної інформації, основ проектування графічних систем, особливо для роботи у режимі реального часу, засвоєння типових алгоритмів, а також знайомство з останніми досягненнями світових технології в області архітектурних і алгоритмічних рішень при проектуванні графічних систем.

Вміння і навички закріплюються у студентів на базі розробки програм створення 2D та 3D – графічних об’єктів, а також розробки функціональної схеми спеціалізованого пристрою для реалізації заданого алгоритму обробки, а також програми його моделювання.

РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА №1
«Синтез двомірного зображення трьомірного об’єкту»

Мета: У результаті виконання роботи студенти повинні знати послідовність та основні математичні вирази для реалізації алгоритму синтезу 3d примитіву.

Етап 1: Підготувати опис об'єкта у власній (локальній) системі координат.

В якості 3d примитіву розглядається випуклий багатогранник, тобто будь-який випуклий об’єкт, апроксимований полігональною моделлю. Особливостю такої фігури є те, що усі грані одноприорітетні, тобто не захиляють одна одну.

Таблиця 1.1 - Тип об'єкта (по останній цифрі залікової книжки)

0	пряма піраміда з основою з 3 вершин
1	пряма піраміда з основою з 4 вершин
2	пряма піраміда з основою з 5 вершин
3	усічена пряма піраміда з основою з 3 вершин
4	усічена пряма піраміда з основою з 4 вершин
5	усічена пряма піраміда з основою з 5 вершин
6	похила піраміда з основою з 4 вершин
7	похила піраміда з основою з 5 вершин
8	усічена похила піраміда з основою з 4 вершин
9	усічена похила піраміда з основою з 5 вершин

Опис повинний включати:

а) габаритний радіус об'єкта,

б) кількість граней об'єкта,

в) опис граней:

- кількість вершин грані,
- координати вершин,
- параметри нормального вектора.

Можливий окремний опис координат вершин. У цьому випадку при опису грані задається список вершин по годинній стрільці з видимої сторони.

Етап 2: Виконати розрахунок двомерного зображення об'єкта.

а) для заданих векторів положення систем координат об’єкту та спостерігача відносно глобальній системи координат виконани розрахунок математичного апарату перетворення геометричних параметрів з однієї системи координат до іншої;

б) проаналізовати відімость об'єкту вцілому та його окремих граней;

в) виконати необхідні геометричні перетворення, масштабування, відсікання, проецірування, відсікання по площині екрана, видові перетворення.

Спостерігач знаходиться в центрі системи координат, розташованої у світовій системі в точці (-300, 0, 0), кути Эйлера рівні (0,0,0).

Положення системи координат об'єкта задане в табл.2.

Вікно спостереження розміром (1.5 X 1.0) метра розташовано на відстані 0.6 метра перпендикулярно осі Ox системи спостерігача.

Таблиця 1.2 - Положення системи координат об'єкта
(по передостанній цифрі)

0	20, 0, 0, 45, 60, 30
1	-50, 0, 0, 30, 45, 60
2	50, 0, 0, 60, 30, 45
3	100, 0, 0, 30, 60, 90
4	-100, 0, 0, 45, 60, 90
5	150, 0, 0, 30, 50, 70
6	-150, 0, 0, 50, 80, 40
7	200, 0, 0, 60, 80, 40
8	-200, 0, 0, 80, 60, 30
9	-20, 0, 0, 90, 60, 45

У якості моделі вікна використовується екран дисплея розміром (270 X 210) мм і растром 780 X 640 пикселів.

Етап 3: Оформлення звіту з розрахунково-графічної роботи.

Зміст звіту:
1. Назва курсу, № роботи, ПІБ студента, № залікової книжці. Завдання.
2. Рисунок постановки задачі синтезу з указівкою положень систем координат об'єкта і спостерігача у світовій системі координат для одного кадру, що відповідає завданню. Попередню оцінку видимих граней для цього ж кадру.
3. Рисунок об'єкта синтезу в локальній системі координат із указівкою нумерації вершин, граней, нормалей.
4. Вміст бази даних опису об'єкта візуалізації з поясненням або розрахунком нормалей кожної грані.
5. Опис етапів синтезу зображення з приведенням основних математичних виражень.
6. Результати розрахунків по кожному етапу для кожної грані.
7. Двомірна картинка зображення об'єкта на екрані (за результатами розрахунків).
8. Перелік посилань.

РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА №2
«Розробка та дослідження заданого етапу алгоритм синтезу зображення трьомірного об'єкту»

Мета: У результаті виконання роботи студенти повинні вміти розробляти спеціалізовані пристрої щодо реалізації етапів алгоритму синтезу зображень.

Етап 1: Вивчення заданого етапу алгоритму синтезу зображення (табл.2.1).

На цьому етапі доцільно виконати підготовку декілька наборів тестових даних для перевірки працездатності вибраної реалізації алгоритму, а також розрахунок очікуваних результатів роботи алгоритму. При цьому можливе використання програмної моделі, розробленої у будь-якому середовищі. Для проведення розрахунків можливе використання математичних пакетів (Matlab, Excel).

Таблиця 2.1 – Етап алгоритму (по номеру в журналі групи)

1	Розрахунок прямої матриці (типу A)
2	Розрахунок зворотній матриці (типу B)
3	Розрахунок сумісній матриці (типу C)
4	Визначення видімості об'єкту (з розрахунком піраміди)
5	Формування списку об'єктів (використання симетричного дерева)
6	Формування списку об'єктів (використання асиметричного дерева)
7	Визначення видімості грані (скалярний додаток)
8	Перетворення координат вершин з системи координат об'єкту в систему координат постеригача
9	Перетворення параметрів векторів з системи координат об'єкту в систему координат постеригача
10	3d-2d перетворення (проецирування)
11	3d відсікання по площані вікна
12	Зміна мірила (перетворення з міліметрів екрану у пікселі)
13	Растризація сегменту прямої лінії (метод ЦДА)
14	Растризація сегменту прямої лінії (метод Брезенхема)
15	Растризація сегменту прямої лінії (метод Люка)
16	Растризація дугі окружності (метод Брезенхема)
17	Растризація дугі еліпсу (метод Брезенхема)
18	Растризація дугі параболи (метод Руа)
19	Растризація дугі гіперболи (метод Брезенхема)
20	Зміна мірила (перетворення з метрів окна у міліметри екрану)
21	3d відсікання по площані вікна
22	2d відсікання по межі «+Xe»
23	2d відсікання по межі «-Xe»
24	2d відсікання по межі «+Ye»
25	2d відсікання по межі «-Ye»

Етап 2: Розробка функціональної схеми для 2-3 способів використання паралельної обробки даних.

На цьому етапі виконується розробка варіантів функціональної організації спеціалізованого пристрою. Доцільно пропонувати послідовне використання паралельної обробки даних. Наприклад, рішення 1 використовує паралельне обчислювання для координат однієї вершіни, рішення 2 додатково до рішення 1 пропонує параллельну обробку вершін.

Етап 3: Розробка моделі пристрою для реалізації етапу алгоритму. Дослідження ефективності способів використання паралельної обробки даних.

Для моделювання доцільно використовувати систему HDL, застосування інших систем також можливе.

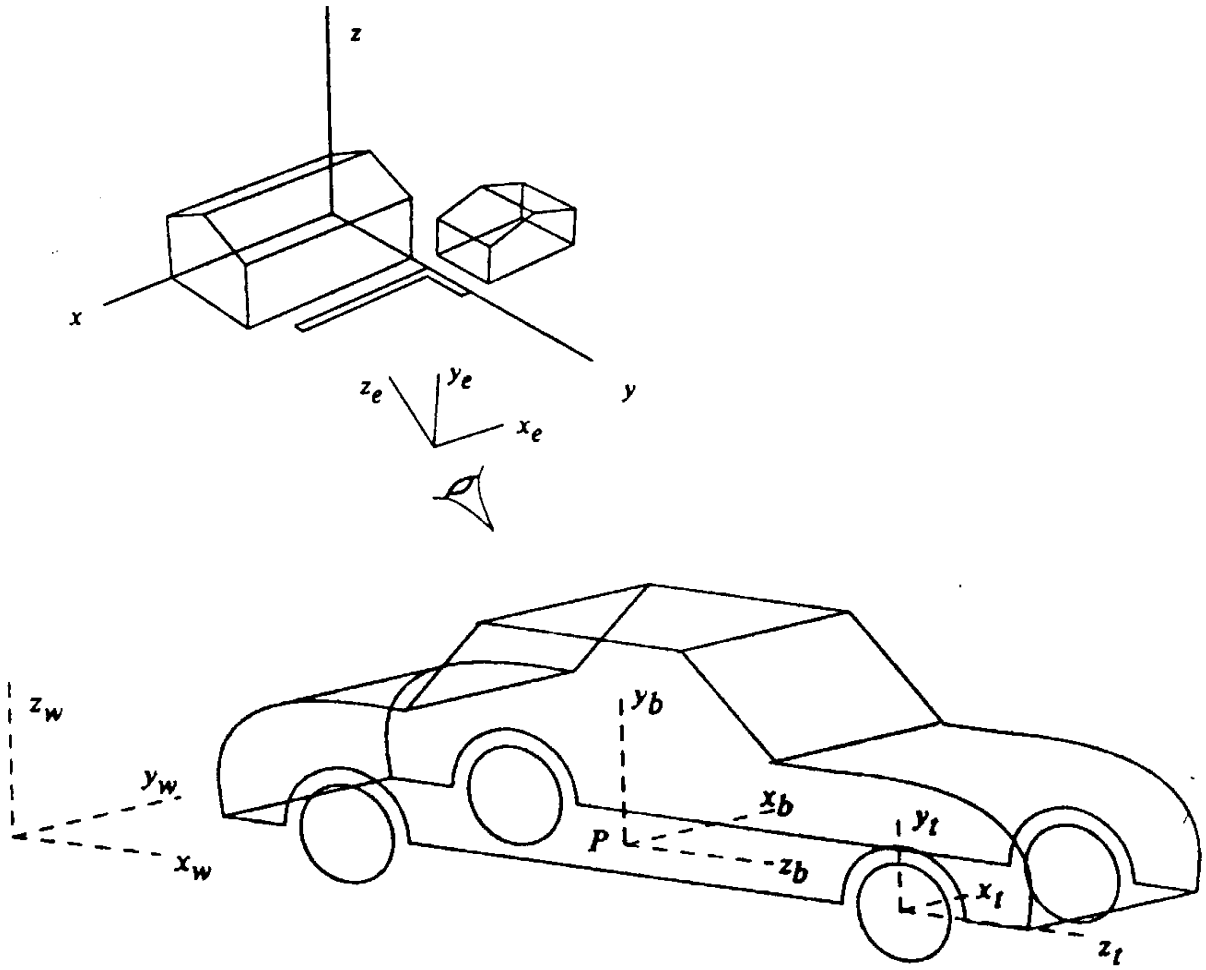
Етап 4: Оформлення звіту з розрахунково-графічної роботи.

Зміст звіту:

1. Назва курсу, № роботи, ПІБ студента, № по журналу групи. Завдання.
2. Опис етапу алгоритму (математичні вирази, схема алгоритму).
3. Тестови набори даних (вхідні – вихідні) з наведенням розрахунків.
4. Варіанти функціональної організації спеціалізованого пристрою.
5. Модель пристрою.
6. Результати моделювання.
7. Висновки та рекомендації з метою вдосконалення.
8. Перелік посилань.

Додаток А
Матеріали для виконання розрахунково-графічної роботи №1

А1. Використання різних систем координат



Основні операції.

Перенос

$$[x'\ y'\ z'\ 1] = [x\ y\ z\ 1]\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ Tx & Ty & Tz & 1 \end{bmatrix},$$

де Tx, Ty, та Tz - компоненти переноса центру системи координат відносно попереднього.

Зміна мірила

$$[x'\ y'\ z'\ 1] = [x\ y\ z\ 1]\begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Повернення відносно однієї вісі

$$[x'\ y'\ z'\ 1] = [x\ y\ z\ 1]\begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Розрахунок матриць перетворення з однієї системи координат до іншої.

$$MB = \begin{bmatrix} \cos\psi\cdot\cos\theta & \sin\theta & -\sin\psi\cdot\cos\theta \\ \sin\psi\cdot\sin\gamma - \cos\psi\cdot\cos\gamma\cdot\sin\theta & \cos\theta\cdot\cos\gamma & \sin\theta\cdot\sin\psi\cdot\cos\gamma + \cos\theta\cdot\sin\gamma \\ \cos\psi\cdot\sin\gamma\cdot\sin\theta + \sin\psi\cdot\cos\gamma & -\sin\gamma\cdot\cos\theta & \cos\psi\cdot\cos\gamma - \sin\theta\cdot\sin\psi\cdot\sin\gamma \end{bmatrix}$$

```
class Matr
{
      double m[3][3];                    //опис
public:
      Matr (double,double,double,int);  //int - тип матриці
      Matr ();
      // Функции
      void print_matr (Matr);
      //operations with Matr
      friend Matr operator * (Matr&,Matr&); //one_Matr*second_Matr=>Matr
      friend Point operator ^ (Point&,Matr&); //Point*Matr=>Point
};
/********************** КОНСТРУКТОРИ **********************/
Matr::Matr()
      { m[0][0] = m[1][1] = m[2][2] = 1;
        m[1][0] = m[0][1] = m[1][2] = 0;
        m[2][0] = m[2][1] = m[0][2] = 0;
      }
Matr::Matr(double ps,double te, double ga, int pr)
{
//    ps,ga,te - кути системи координат, град;
//    pr     - тип матриці
//    pr=1 з локальної до глобальної (типу A)
//    pr=2 з глобальної до локальної (типу B)
      double psr,ter,gar;
      double ps_s,ps_c;
      double te_s,te_c;
      double ga_s,ga_c;
      double psga_ss,psga_sc,psga_cs,psga_cc;
      //градуси до радіан 0,017454=3.14/180
      psr = ps*0.017454;
      ter = te*0.017454;
      gar = ga*0.017454;
      //sin and cos
      ps_c = cos(psr); ps_s = sin(psr);
      te_c = cos(ter); te_s = sin(ter);
      ga_c = cos(gar); ga_s = sin(gar);
      //часткові додатки, які використовуються двічі
      psga_ss=ps_s*ga_s;
      psga_cc=ps_c*ga_c;
      psga_sc=ps_s*ga_c;
      psga_cs=ps_c*ga_s;
```

```
      // пряма матриця
      if (pr==1)
            {
            m[0][0] =  ps_c*te_c;
            m[0][1] =  psga_ss-psga_cc*te_s;
            m[0][2] =  psga_cs*te_s+psga_sc;
            m[1][0] =  te_s;
            m[1][1] =  te_c*ga_c;
            m[1][2] = -te_c*ga_s;
            m[2][0] = -ps_s*te_c;
            m[2][1] =  te_s*psga_sc+psga_cs;
            m[2][2] =  psga_cc-te_s*psga_ss;
            }
      // трансформована матриця
      if (pr==2)
            {
            m[0][0]  =  te_c*ps_c;
            m[0][1]  =  te_s;
            m[0][2]  = -te_c*ps_s;
            m[1][0]  =  psga_ss - psga_cc*te_s;
            m[1][1]  =  ga_c*te_c;
            m[1][2]  =  psga_sc*te_s+psga_cs;
            m[2][0]  =  psga_cs*te_s+psga_sc;
            m[2][1]  = -ga_s*te_c;
            m[2][2]  =  psga_cc - psga_ss*te_s;
            }
}
```

A2 Розрахунок параметрів нормального вектору.

```
typedef struct
{
double x,y,z;
} v;
void main()
{
 FILE *f; f=fopen("res.dat","w+");
 clrscr();
 int d;
 v p[3];
 for(int i=0; i<3;i++)
  {
   fprintf(f,"Input point %d:",i+1);
   scanf("%d",&d); p[i].x=(double)d;
   scanf("%d",&d); p[i].y=(double)d;
   scanf("%d",&d); p[i].z=(double)d;
   fprintf(f,"p[%d]:%10.2lf,%10.2lf,%10.2lf\n",
   i+1,p[i].x,p[i].y,p[i].z);
  }
 v a,b,n;
 a.x=p[1].x-p[0].x;
 a.y=p[1].y-p[0].y;
 a.z=p[1].z-p[0].z;
 b.x=p[2].x-p[0].x;
 b.y=p[2].y-p[0].y;
 b.z=p[2].z-p[0].z;
 fprintf(f,"\n a:%10.2lf,%10.2lf,%10.2lf",a.x,a.y,a.z);
 fprintf(f,"\n b:%10.2lf,%10.2lf,%10.2lf",b.x,b.y,b.z);
 // Векторний додаток
 n.x=a.y*b.z-b.y*a.z;
 n.y=a.z*b.x-b.z*a.x;
 n.z=a.x*b.y-b.x*a.y;
 fprintf(f,"\n n:%10.2lf,%10.2lf,%10.2lf",n.x,n.y,n.z);
```

```
// Довжина вектору
double l=sqrt(n.x*n.x+n.y*n.y+n.z*n.z);
// Нормалізація вектору
 n.x=n.x/l;
 n.y=n.y/l;
 n.z=n.z/l;
 fprintf(f,"\n After normalization n:%8.6lf,%8.6lf,%8.6lf",n.x,n.y,n.z);
 }
```

Зміст файлу з результатами:

Input point 1:p[1]: -100.00, -100.00, 100.00

Input point 2:p[2]: -50.00, 100.00, -25.00

Input point 3:p[3]: 0.00, -100.00, 150.00

a: 50.00, 200.00, -125.00

b: 100.00, 0.00, 50.00

n: 10000.00, -15000.00, -20000.00

After normalization n:0.371391,-0.557086,-0.742781

А3. Синтез многогранника

А3.1 Общий алгоритм синтеза

Для синтеза изображения трехмерного объекта необходимо подготовить базу данных описания объекта (в соответствии с принятым способом аппроксимации и требованиями алгоритма синтеза) и загрузить ее, при возможности, в оперативную память.

В общем случае, процесс синтеза – это бесконечный цикл смены кадров изображения. Смена кадров должна выполняться с частотой не менее 20 кадров в секунду (требования непрерывности «картинки» на экране). Реально, исходными данными для процесса синтеза являются результаты динамики движения объекта и наблюдателя. Для простейших примеров для задания и изменения векторов положения можно использовать изменение параметров при помощи управляющих клавиш или в цикле по некоторому закону.

А3.2. Последовательность синтеза одного кадра

Ввод положений систем координат и расчет матриц преобразования.

В каждом кадре как наблюдатель, так и объекты сцены могут изменять свое расположение. Для отработки этих изменений в начале синтеза каждого кадра вводятся параметры (Xo,Yo,Zo,Po,Qo,Go,Xн,Yн,Zн,Pн,Qн,Gн), определяющих взаимное расположение центра отображаемой сцены (СКО) и наблюдателя (СКН) в мировой системе координат. По введенным значениям рассчитываются матрицы преобразований из одной системы в другую:

- А - преобразование из системы координат отображаемой сцены в мировую систему координат (по углам СКО, признак =1);
- В - преобразование из мировой системы координат в систему координат наблюдателя (по углам СКН, признак =2);
- С - преобразование из системы координат отображаемой сцены в систему координат наблюдателя по формулам:

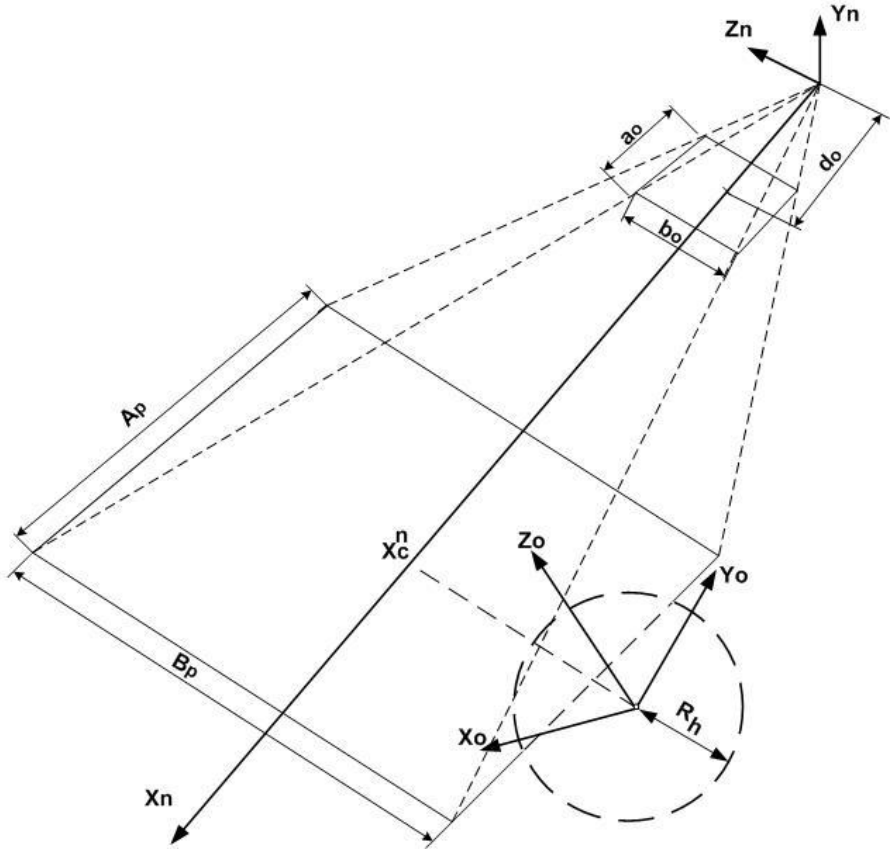
C11=B11*A11+B12*A21+B13*A31
C12=B11*A12+B12*A22+B13*A32
C13=B11*A13+B12*A23+B13*A33
C14=B11*(Xo-Xн)+B12*(Yo-Yн)+B13*(Zo-Zн)

C21=B21*A11+B22*A21+B23*A31
C22=B21*A12+B22*A22+B23*A32
C23=B21*A13+B22*A23+B23*A33
C24=B21*(Xo-Xн)+B22*(Yo-Yн)+B23*(Zo-Zн)
C31=B31*A11+B32*A21+B33*A31
C32=B31*A12+B32*A22+B33*A32
C33=B31*A13+B32*A23+B33*A33
C34=B31*(Xo-Xн)+B32*(Yo-Yн)+B33*(Zo-Zн)

Аналитические выражения для вычисления значений коэффициентов приведенных матриц получены путем последовательного применения операций преобразований 3-D координат.

Задача отсечения относительно видимого объема

Необходимо отметить, что решение задачи удаления невидимых поверхностей обычно начинается на этапе формирования дисплейного списка кадра, когда из базы выбираются объекты и/или их элементы, потенциально видимые через экран (пирамида видимости показана на рисунке ниже) в текущем положении наблюдателя.



Прежде чем выполнять преобразование координат объекта в систему координат наблюдателя (процесс, требующий больших затрат времени), целесообразно выполнить проверку: виден ли вообще (хотя бы частично) данный объект при текущем взаимном расположении объекта и наблюдателя. Объект может находиться за наблюдателем (направление взгляда противоположно расположению объекта) или находиться перед наблюдателем, но настолько высоко (низко, слева или справа), что не будет виден через окно наблюдения, т.е. его изображение не попадет на плоскость окна (экрана). Наличие такой информации может существенно сократить временные затраты при синтезе текущего кадра изображения.

При определении потенциальной видимости объекта задача решается в системе

координат наблюдателя, поэтому в качестве первого шага алгоритма на данном этапе необходимо выполнить перевод координат центров объектов из мировой системы координат в систему координат наблюдателя, т.е. умножить их на матрицу В. Затем последовательно произвести анализ видимости каждого объекта. При определении видимости достаточно представлять объекты их габаритными шарами, что значительно упрощает анализ и в то же время позволяет отбрасывать заведомо невидимые в данном кадре объекты.

Для каждого объекта входными данными на этом шаге являются:

- размеры $a_o \times b_o$ экрана, расположенного перпендикулярно оси наблюдения (совпадает с осью Ox системы координат наблюдателя) ;

- расстояние от экрана до наблюдателя, d_o;

- координаты центра объекта, $(x_c^n; y_c^n; z_c^n)$ в СКН;

- радиус габаритного шара объекта, R_h .

Исходя из размеров экрана, видимости на расстоянии d равна:

$$\begin{aligned} Vis = &(x_c^n > d_o - R_h) \,\&\, (y_c^n < \frac{B_p}{2} + R_h) \\ &\&\, (y_c^n > -\frac{B_p}{2} - R_h) \,\&\, (z_c^n < \frac{A_p}{2} + R_h) \\ &\&\, (z_c^n > -\frac{B_p}{2} - R_h) \end{aligned} \qquad (3.1)$$

где A_p, B_p – основание пирамиді, определяемое из віражения:

$$\frac{x_c^n}{d_o} = \frac{A_p}{a_o} = \frac{B_p}{b_o}, \;\; A_p = a_o \cdot \frac{x_c^n}{d_o}; \;\; B_p = b_o \cdot \frac{x_c^n}{d_o}, \qquad (3.2)$$

Анализ видимости грани

На этом этапе в начале выполняется анализ направляющего вектора каждой грани объекта, при этом нормаль грани преобразуется в систему координат наблюдателя

NXi'=NXi*C11+NYi*C12+NZi*C13
NYi'=NXi*C21+NXi*C22+NZi*C23
NZi'=NXi*C31+NYi*C32+NZi*C33

Затем координаты одной вершины грани объекта преобразуются в систему координат наблюдателя в соответствии со следующими соотношениями:

X1'=X1*C11+Y1*C12+Z1*C13+C14
Y1'=X1*C21+X1*C22+Z1*C23+C24
Z1'=X1*C31+Y1*C32+Z1*C33+C34

и рассчитывается скалярное произведение вектора наблюдения на нормаль грани

S=X1'*(NXi')+(Y1')*(NYi')+ (Z1')*(NZi')

Для видимой грани S<0.

3.2.4. Задача преобразования координат объекта в систему координат наблюдателя

На следующем этапе координаты каждой лицевой грани объекта преобразуются в систему координат наблюдателя в соответствии со следующими соотношениями:

Xi'=Xi*C11+Yi*C12+Zi*C13+C14
Yi'=Xi*C21+Xi*C22+Zi*C23+C24
Zi'=Xi*C31+Yi*C32+Zi*C33+C34

Задача отсечения по плоскости окна

Затем проверяется положение каждой грани относительно оконной плоскости. Если грань вся лежит перед окном, то она отбрасывается, если вся за окном, то передается для дальнейшей обработки. В случае, если грань пересекает плоскость, которой принадлежит окно, рассчитываются значения координат точек пересечения грани с плоскостью окна в соответствии со следующими выражениями:

Хпп=Dэ

Yпп=(Dэ-Хпэ)*(Yзэ-Yпэ)/(Хзэ-Хпэ)+Yпэ

Zпп=(Dэ-Хпэ)*(Zзэ-Zпэ)/(Хзэ-Хпэ)+Zпэ,

где индексы переменных имеют следующий смысл:

пп -точка пересечения,

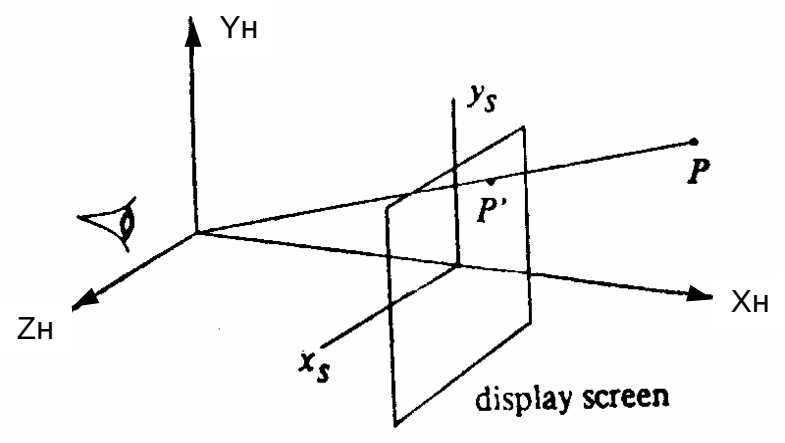
пэ -точка перед экраном,

зэ- точка за экраном.

Dэ- расстояние от точки наблюдения до экранной плоскости.

3.2.6. Проецирование на плоскость окна и перевод в экранные координаты

Перспективна проекція точки Р на площину вікна або дисплея.



Проецирование трехмерных координат на оконную плоскость производится по формулам:

Xiэ=Kx*Zi/Xi

Yiэ=Ky*Yi/Xi

Этап масштабирования координат – перевод в экранные координаты (пикселы) с учетом линейных размеров экрана, которые в общем случае могут не совпадать с размерами окна, обычно, совмещается с этапом проецирования, при этом масштабные коэффициенты учитываются при расчете коэффициентов проецирования Кх и Ку.

3.2.7. Отсечение по границам экрана

Затем следует этап отсечения граней по границам экрана. Для получения координат точек пересечения грани с границами экрана выполняются проверки по всем четырем границам. При использовании формата с плавающей точкой для отсечения по границе "+Хэ" применяются следующие соотношения:

Xiэ=Хэ

Yiэ=(Хэ-Хipр)*(Yipс-Yipр)/(Хipс-Хipр)+Yipр

Для остальных границ соотношения аналогичны.

При использовании целочисленного значения целесообразнее использовать метод половинного деления.

Додаток Б

Приклади виконання етапів розрахунково-графічної роботи №2

Б1. Відсікання одного ребра грані відносно правої межі екрану

Розглянемо загальний алгоритм (рис.Б.1).

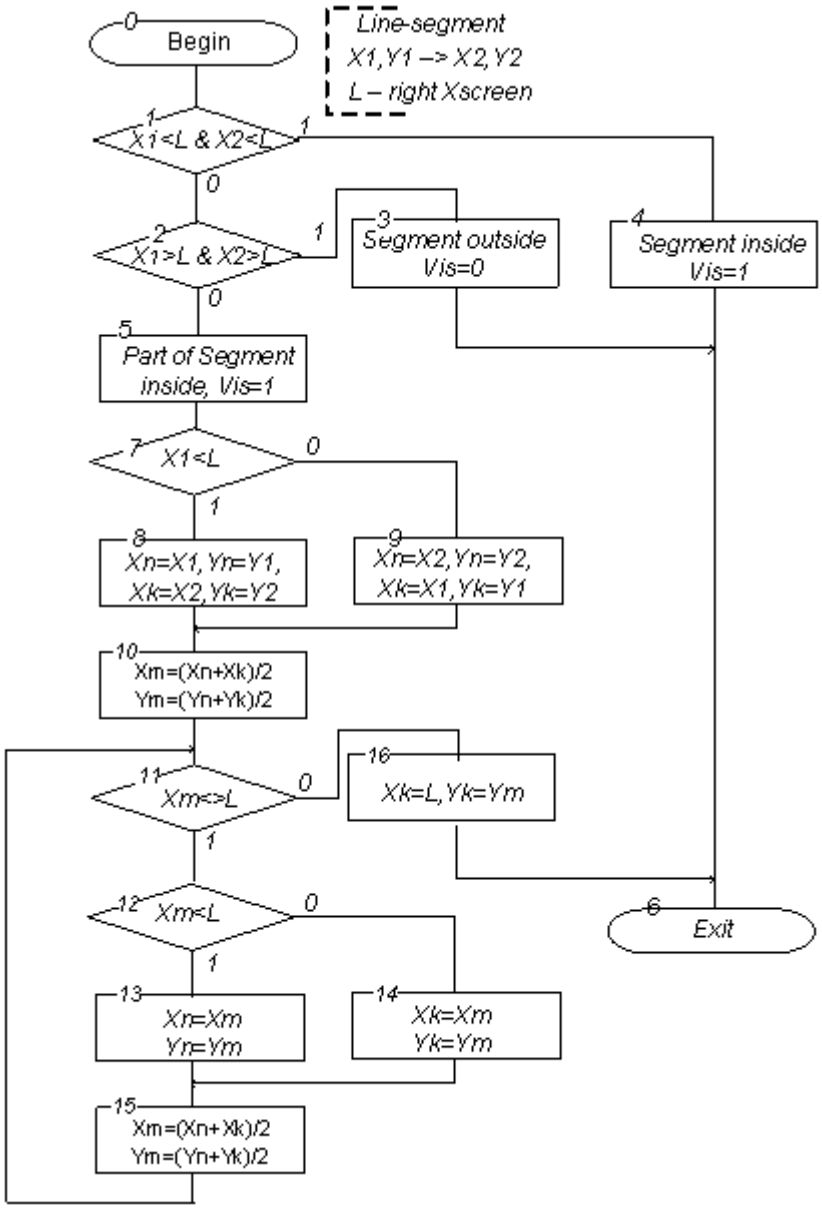


Рис.Б.1. Алгоритм відсікання відносно правої межі екрану

Розглянемо варіанти використання паралельних обчислювань сумісно з використанням UML [6] для оформлення алгоритмів. Алгоритм пошуку крапки перетину з межею екрану наведений на рис.Б.2.

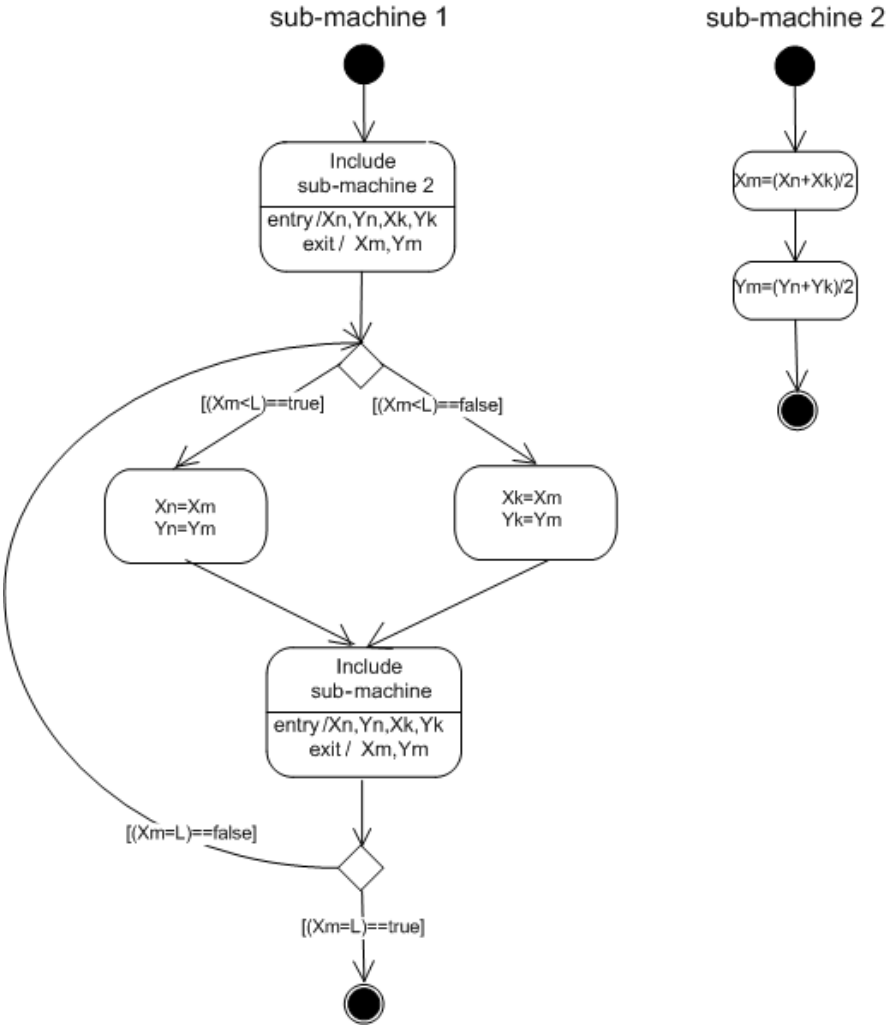


Рис.Б.2. Алгоритм пошуку крапки перетину

```
  -- Architecture of middle-point-subdivision block
  entity sum_div is
     port(Xn: in integer;
            Xk: in integer;
          Yn: in integer;
            Yk: in integer;
          Xm: out integer:=0;
          Ym: out integer:=0
            );
  end sum_div;
  ------------------------------------------------------
  architecture sum_div1 of sum_div is
  begin
     process (Xn,Xk,Yn,Yk)
     begin
           Xm<=(Xn+Xk)/2;
           Ym<=(Yn+Yk)/2;
     end process;
end sum_div1;
  ------------------------------------------------------------
  -- Architecture of processor for cross-point calculation
  entity cpu is
     port(Xn: in integer;
          Xk: in integer;
          Yn: in integer;
          Yk: in integer;
```

```
         L:  in integer;
         Xc: out integer:=0;
         Yc: out integer:=0;
           Res: out integer:=0
           );
end cpu;

architecture cpu1 of cpu is
component sum_div is
    port(Xn,Xk,Yn,Yk: in integer:=0;
         Xm,Ym: out integer:=0);
end component sum_div;
signal Xna,Xka,Yna,Yka,Xca,Yca: integer:=0;
begin
   U1 : sum_div port map(Xna,Xka,Yna,Yka,Xca,Yca);
process
variable Xn1,Xk1,Yn1,Yk1,Xc1,Yc1: integer;
variable Res1: integer:=0;
 begin
 Xn1:=Xn;
 Xk1:=Xk;
 Yn1:=Yn;
 Yk1:=Yk;

 if Xk /= L and Xn /= L then
 m1: loop
--   Xc1:=(Xn1+Xk1)/2;
--   Yc1:=(Yn1+Yk1)/2;
   Xna<=Xn1;
   Xka<=Xk1;
   Yna<=Yn1;
   Yka<=Yk1;
   Xc1:=Xca;
   Yc1:=Yca;
   if Xc1>L then
      Xk1:=Xc1;
      Yk1:=Yc1;
   else
      Xn1:=Xc1;
      Yn1:=Yc1;
   end if;
  wait for 100 ns;
  exit m1 when Xc1 = L;
  end loop m1;
 else
  wait for 10 ns;
  if Xk = L     then
     Xc1:=Xk;
     Yc1:=Yk;
  else
    Xc1:=Xn;
     Yc1:=Yn;
  end if;
 end if;
 Xc<=Xc1;
 if Res1=0 then
 Yc<=Yc1;
 end if;
 Res<=1;
 Res1:=1;
 end process;
end cpu1;
```

Варіант 1 – паралельний розрахунок середньої крапки для X та Y (Sub-machine 2 розподілена на 2 частки).

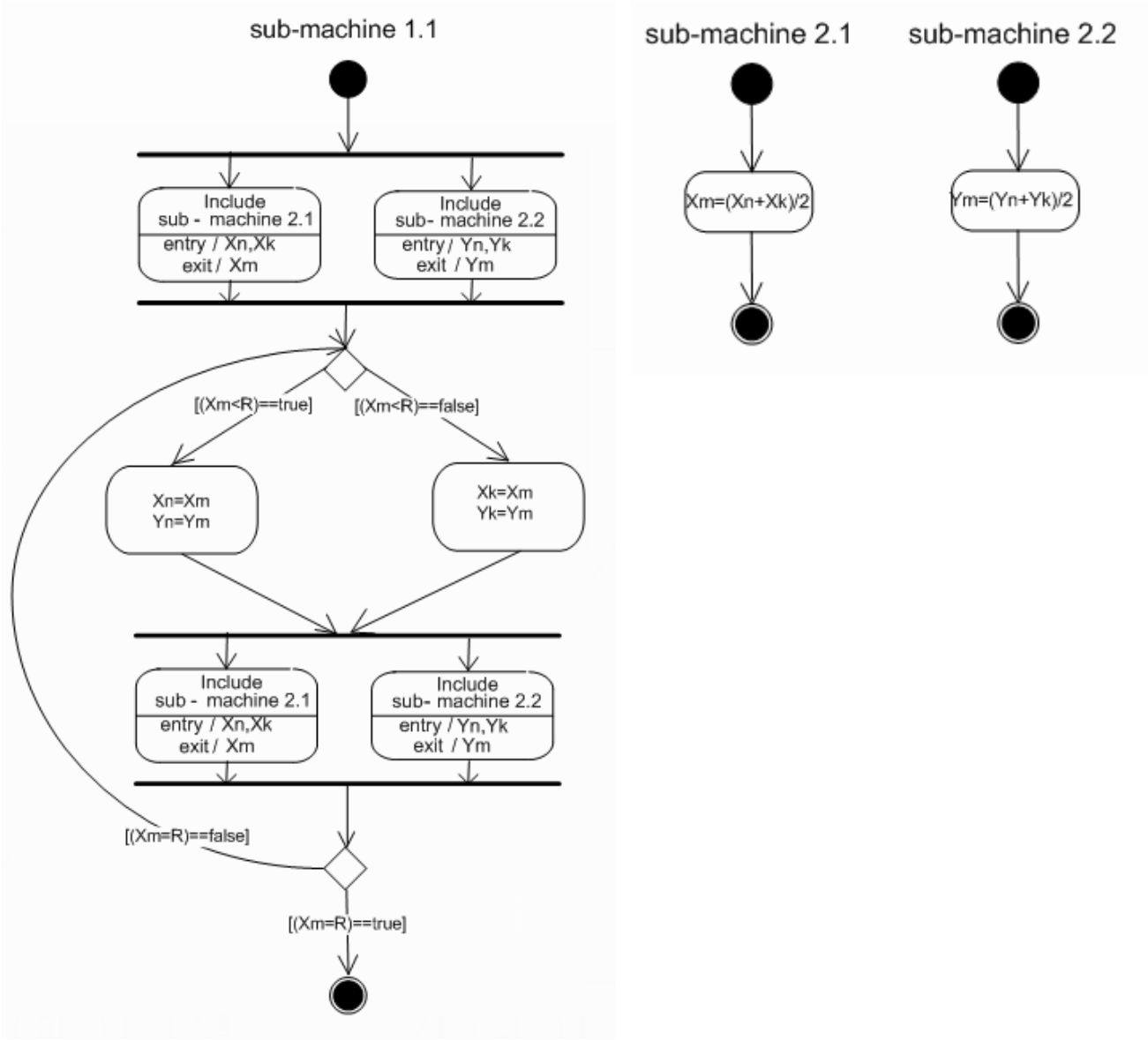


Рис.Б.3. Алгоритм пошуку крапки перетину: варіант 1

З використанням Sub-machine 1.1 отримаємо Sub-machine для відсікання відносно усіх 4-х меж екрану (Clip L, Clip R, Clip T, Clip B), як наведено на рис.Б.4 - ліва частина.

Варіант 2 – паралельний пошук крапок перетину для правої та лівої X- межі (оскільки вони незалежні) , а потім паралельний пошук крапок перетину для верхньої та ніжній Y - межі (рис.Б.4 – права частина).

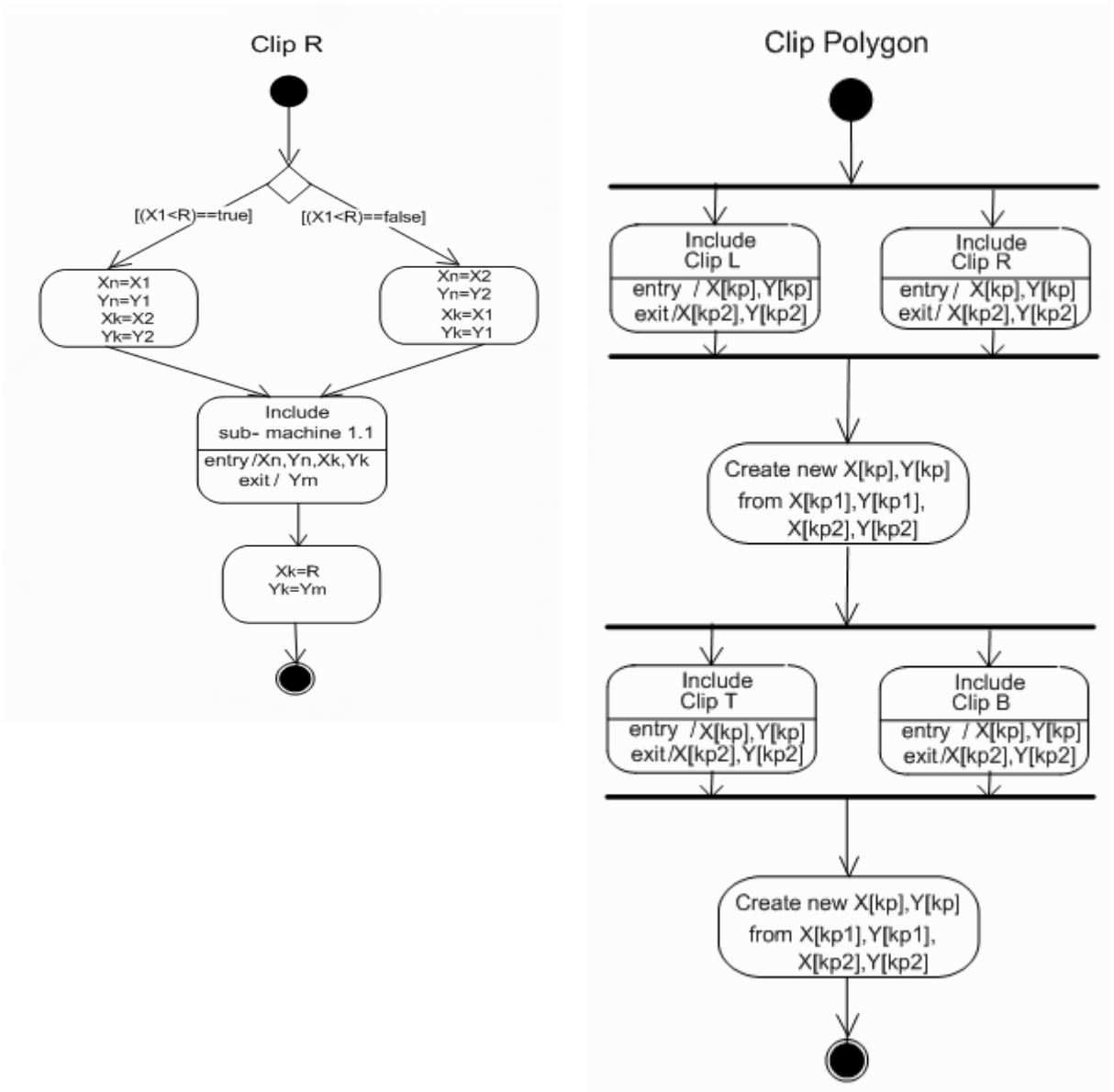
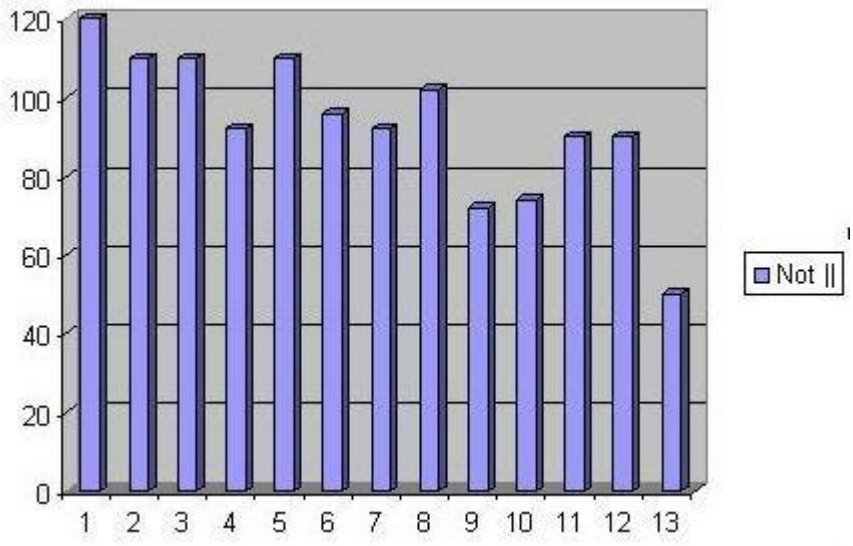


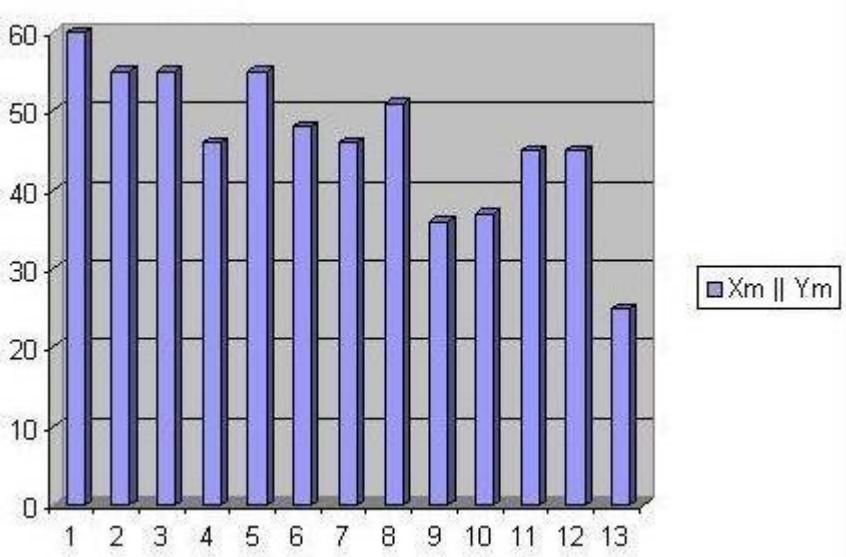
Рис.Б.4. Алгоритм відсікання полігону

Дослідження ефективності проведені для 13 полігонів. Отримані наступні діаграми кількості використаних операцій «+>>» (поділ на 2 реалізований зсувом). На рис.Б.5. наведена діаграма кількості використаних операцій «+>>» для послідовного алгоритму та варіанту 1 (розрахунок середньої крапки паралельно для X та Y).

На рис.Б.6. наведено порівняння діаграм кількості використаних операцій «+>>» для всіх варіантів реалізації алгоритму.



а)



б)

Рис.Б.5. Кількість використаних операцій «+>>»
а) послідовний алгоритм
б) розрахунок середньої крапки паралельно для X та Y (варіант 1)

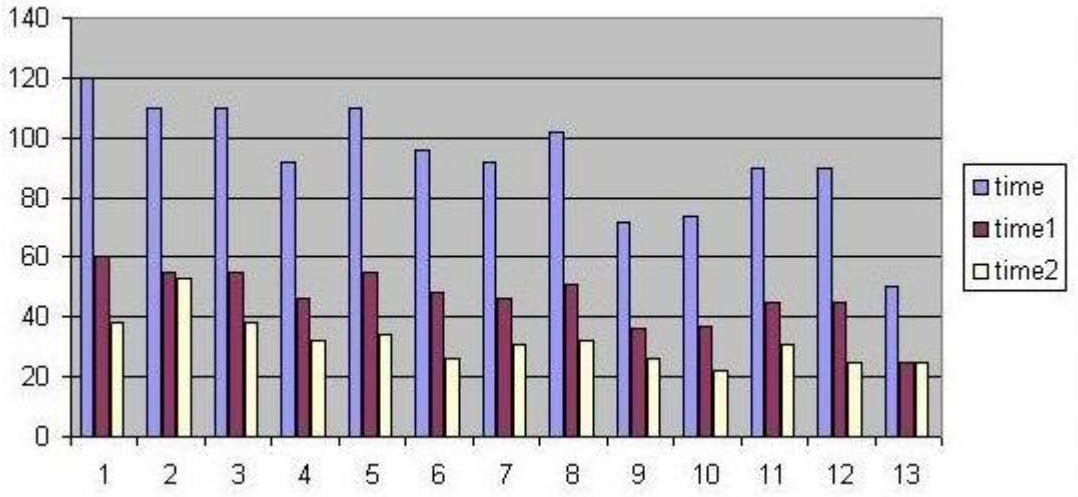
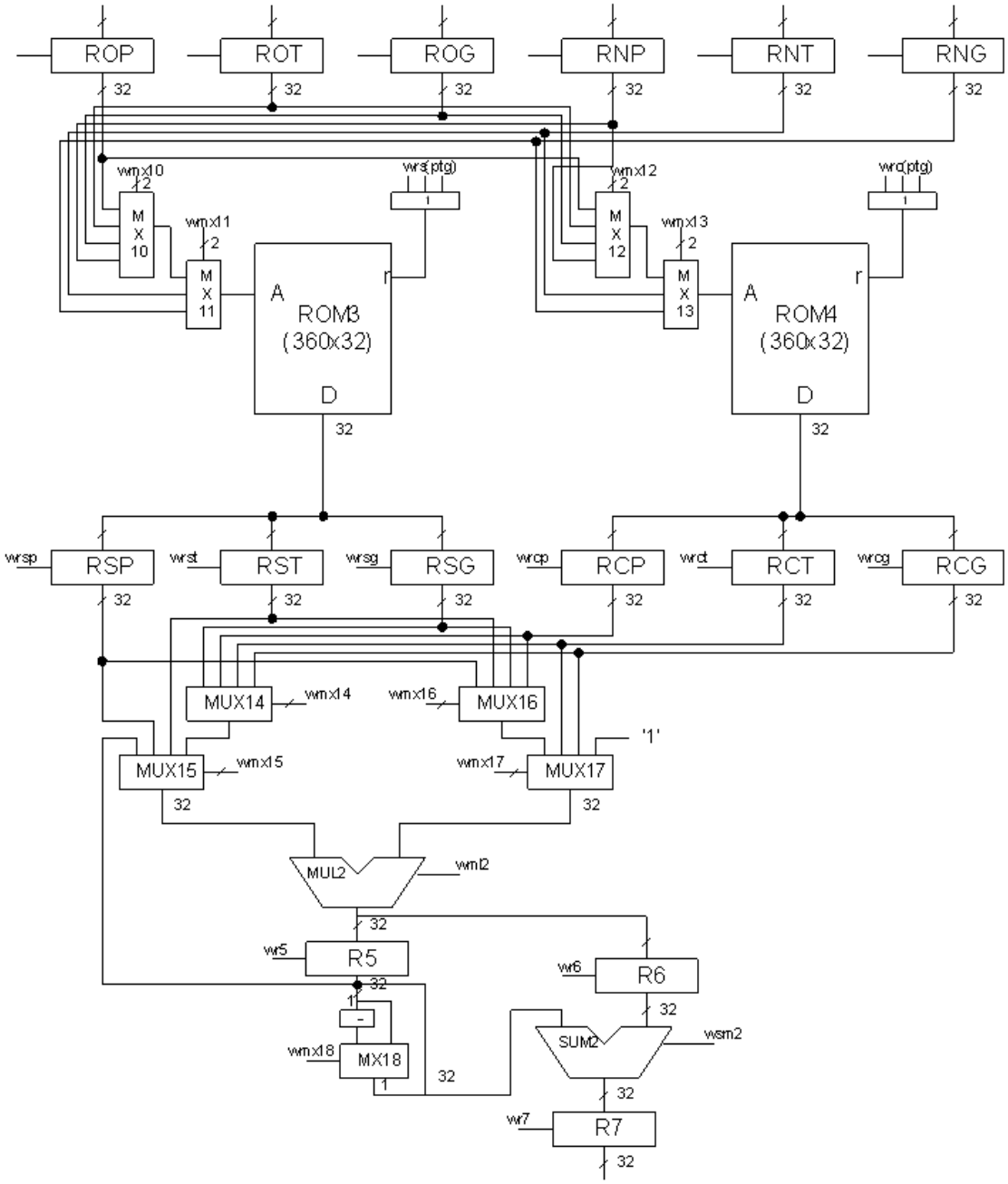


Рис.Б.6. Діаграма кількості операції “+>>”:
time – послідовна схема; time1 – “Xm || Ym”; time2 - “R || L + T || B”

Б2. Приклад функціональної організації процесору для розрахунку матриці.



ПЕРЕЛІК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Лекції з курсу «Апаратно-програмні засоби систем комп'ютерної графіки» (для студентів напрямку підготовки «Комп'ютерна інженерія») / Уклад.: Р.В. Мальчева. - Донецьк, ДонНТУ, 2013. - 132 с.

2. Веселовська Г.В. Основи комп'ютерної графіки. Навчальний посібник (с грифом МОН України). – К.: Центр учбової літератури, 2004. – 392 с.

3. Вельтмандер П.В. Машинная графика. Учебное пособие в 3-х книгах. - Новосибирск: НГУ, 1997.
Кн.1. Вводный курс. - 123 с., ил.
Кн.2. Основные алгоритмы. - 193 с., ил.
Кн.3. Архитектуры графических систем. – 90 с., ил.

4. Foley J., Dam A. Computer Graphics. Principles and practice. - 2nd ed. In C. - AWPC, 1997. – 1175 p.

5. Головашин, В.Л. Основы компьютерной графики : учебное пособие / В.Л. Головашин, С.А. Вязовов, С.И. Лазарев. – Тамбов: Изд-во Тамб. гос. техн. ун-та, 2008. – 80 с. [pdf - версия]

6. S.J. Mellor and M.J. Balcer. Executable UML. A Foundation for Model - Driven Architecture. Indianapolis: Addison-Wesley, 2002.

Навчальне видання

Методичні вказівки щодо організації самостійної роботи студентів при виконанні індивідуальних завдань з дисципліни «Апаратно-програмні засоби систем комп’ютерної графіки»

(для студентів заочної, заочної прискореної з наданням денних освітніх послуг і денної форм навчання напряму підготовки 6.050102 «Комп’ютерна інженерія»)

Укладач: Мальчева Раїса Вікторівна