N89-24016

VARIABLE PLANT SPACING

Prepared by

Jim Bledsoe
Lee Weiss

Fall 1987

SUMMARY

The goal of this project was to develop a system for varying the spacings between soybean plants as they grow to maximize the number of plants grown in a given volume. The project was studied to aid in the development of NASA's Controlled Ecological Life Support System (CELSS). The resulting design consists of plant trays which are three dimensional trapezoids arranged into circles in a compact geometrical configuration. These circles are stacked together in back to back pairs to form a long cylinder. In each growth tray, plants will be housed in individual containers containing a nutrient delivery system and a plant support mechanism. Between the containers, a "half" trellis has been designed to space the plants for maximum space efficiency. The design allows for localized seeding and harvesting mechanisms due to the chambers' geometrical configuration. In addition, the components have been designed for ease of cleaning and minimal maintenance. Next semester, the individual components will be constructed and tested to determine the success of the design.

TABLE OF CONTENTS

INTRODUCTION

Problem Definition

Conservation of vehicle space is crucial since there is a limited volume available. Focusing on the space allotted for growing plants, research must be undertaken to develop a system to maximize the use of this space. Plants such as soybeans require less space to grow as a seedling than they do as a mature plant ready for harvesting. There is potential for utilizing this size difference as a foundation on which to conduct pertinent research in the area of optimizing plant spacing.

Project Description

The purpose of this project is to design and build a system for growing plants in space. This design will conserve space and maximize the number of plants per volume by spacing plants for maximum space efficiency. Preliminary investigations have been directed toward soybean plants since soybean plants require horizontal and vertical spacing unlike other crops like wheat which only require vertical spacing [7]. Results of a three dimensional space saving design are potentially more valuable than that of one or two dimensional design.

Design Criteria

Based on NASA's guidelines for research on CELSS [1], the following general design criteria have been established for this project:

1. There should be three dimensional plant spacing to minimize volume required for growing plants.
2. Production of soybeans should be continuous.
3. System weight should be minimized.

4. Nutrients must be contained and shielded from light.
5. The plants must be rigidly supported.
6. Air circulation must be provided from root to canopy.
7. Each plant must be provided adequate lighting.
8. Maximum utilization of automation should be used to minimize the duties of the crew.
9. Access to seeding and harvesting mechanisms should be incorporated into the design.
10. An automated system should be able to clean and reprocess the growth medium.
11. There should be a minimum of maintenance on components and any required maintenance should be capable of being performed by an automated system.

Some of the above criteria were considered more carefully than others. The primary criteria are:

1. There should be three dimensional plant spacing to minimize volume required for growing plants.
2. Nutrients must be contained and shielded from light.
3. The plants must be rigidly supported.
4. Maximum utilization of automation should be used to minimize the duties of the crew.
5. Access to seeding and harvesting mechanisms should be incorporated into the design.
6. An automated system should be able to clean and reprocess the growth medium.

Background Information

The design of an independent, self-sustaining plant growth chamber will become increasingly important to the future of food production. Its importance lies in the ability of a plant growth chamber to support human life in a wide variety of environments.

Plant growth chambers will realize their potential in manned, deep space exploration, where storage and weight restrictions necessitate the efficient production of food crops.

Plant growth systems have been previously created which incorporate hydroponics into the design to maximize production in a minimum of space [1,2,3,4,5]. Most of these designs that alter the spacing between plants during growth do not provide for expansion in more than one or two dimensions.

CONCEPTS AND DESIGNS

Geometrical Configurations

The initial stages of investigation involved examining existing systems for growing plants in space [1,2,3,4,5]. These were evaluated and used as references for preliminary designs.

Tray shape. In designing a tray to hold the plants, the approach taken was to custom fit a tray to the plants. This approach facilitates a better solution to the problem of tray shape than to fit plants to a tray. Due to the sigmoidal growth curve for soybean plants (Figure 1), the natural spacing of the rows would be in the shape of a sigmoid curve. As a result, the trays should be constructed into trapezoids along the horizontal and vertical planes approximating the sigmoid lines (Figure 2).

Integration of Growth Trays into Chambers. Simplified paper models of the trapezoidal shape were constructed and arranged like building blocks into many different configurations. These models helped visualize the three dimensional shape of the growth chamber.

Integration with Planting and Harvesting Systems. An automated process would be responsible for planting. Seeding would entail placing a germinated seed onto or into the support device where it would grow. The growth trays will be shaped and configured to allow the seeding mechanism to perform its task easily and as quickly as possible.

Likewise, the configuration of the plants in the tray must allow for the mature plants to be harvested without disturbing the other plants. Cutting the stem above the supporting mechanism would easily separate the canopy area from the root area. The edible portion would be transported to a processor to make food, and the cleaning system would remove the root portion.

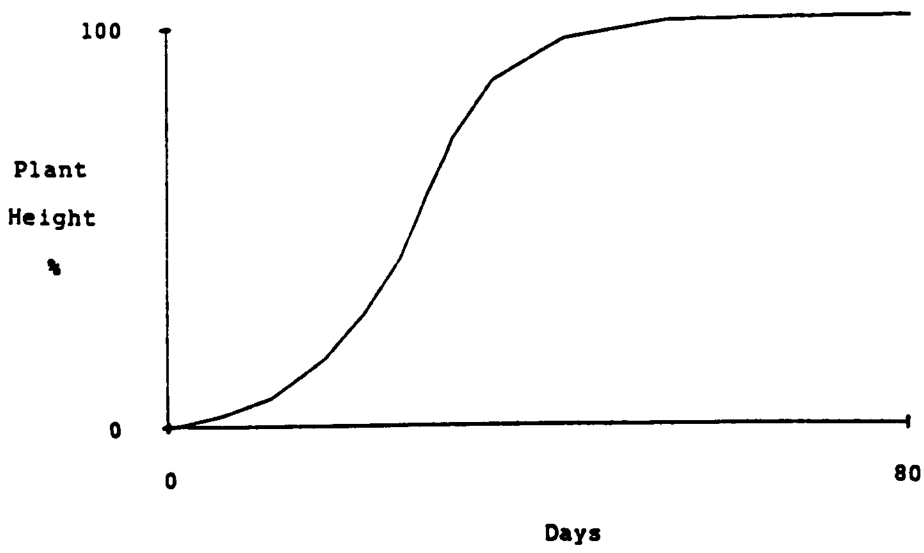


Figure 1. Soybean Growth Curve

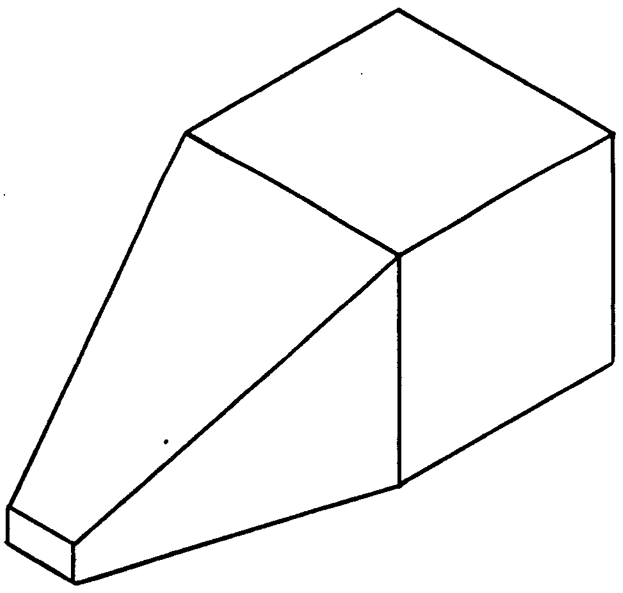


Figure 2. Trapezoidal Growth Tray

Plant Movement

Varying Plant Spacing During Growth. Varying the spacing of the plants is essential for conservation of space. One method developed was a mechanical "half" trellis (Figures 3a,3b). This trellis would support each individual plant in a separate container and would facilitate the independent movement of the plants. The trellises would provide stable, predictable expansion since they are constructed of rigid materials. Ends of the trellises will travel in tracks placed at the sides of the growth chamber. Nonlinear spacing of the plants is enhanced because as the trellis is expanded as it proceeds through the chamber, it also becomes thinner, providing movement perpendicular to the expansion of the trellis.

A computer program was written to animate the trellis design to help visualize how the individual plants would move as the trellises were expanded (Appendix A). The program enables the user to adjust the number of plants on a trellis, the number of trellises on a tray, and the maximum to minimum width ratio of the supporting track. The current version (V 1.0) of the program only allows for single step trapezoids, but later versions will allow two step trapezoids. Other future enhancements include having the program pack the rows together as closely as geometrically possible while evenly scaling the ages of the rows of plants. This will help determine the function required to move the endpoints of the trellises along the track.

An alternate system for plant spacing is an accordion tube. It is similar to a design referred to as an accordion tray [1]. The accordion tube system is simply a single row of plants on an expandible tube (Figure 4). By having single rows of plants, the tubes may be spaced separately and add another dimension of expansion over the existing method. The tube is a small hose which resembles a dryer hose. Holes for plant stems are placed at regular intervals along the tube.

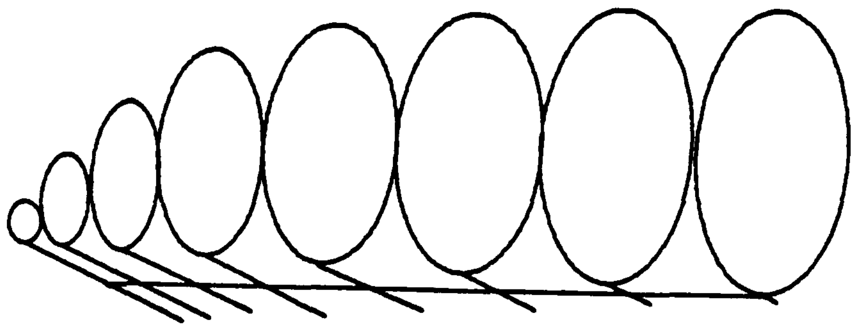


Figure 3a. Trellis Design - Side View

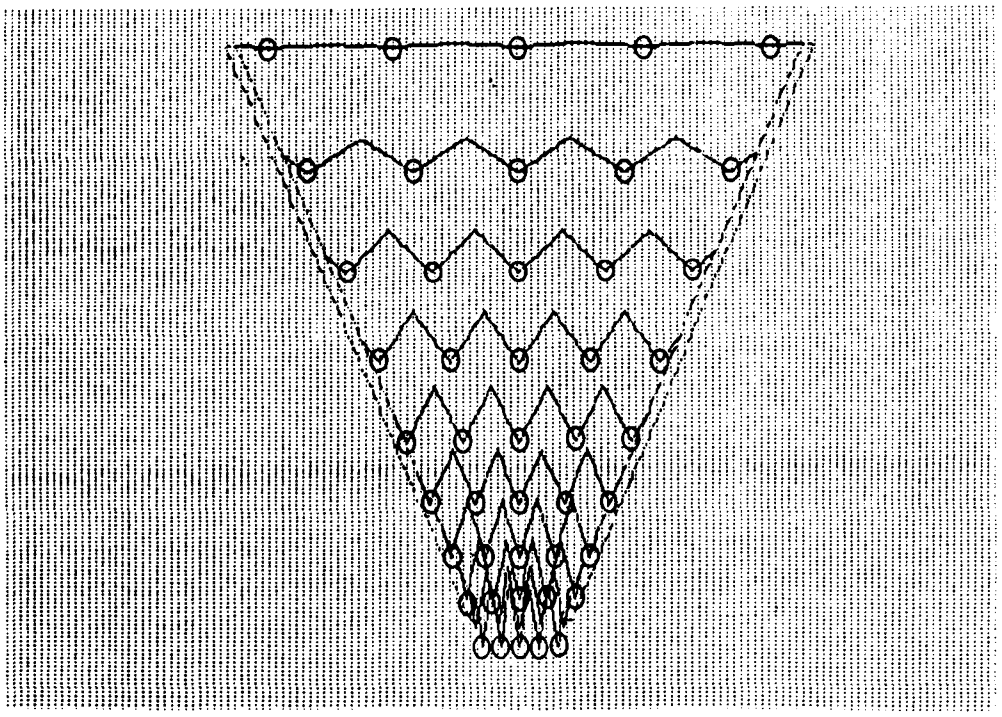


Figure 3. Computer Simulation of Trellis Support

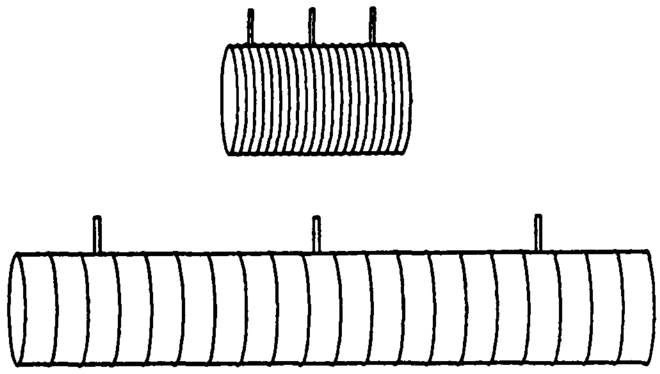


Figure 4. Accordian Tube System

Varying Row Spacing. There are several different methods for moving the rows of plants apart. The first is a simple motorized robot. Two linear stepper motors with simple grasping or pushing mechanisms would travel along the outside of the growth trays. The motors would be controlled by a computer. This system would be reprogrammable in mid flight.

A similar motorized system that is spaced by mechanical stops along a track instead of a computer was also studied. This system would not be adjustable once the track was constructed.

Another system involves having mechanical levers or linkages alter the space between rows in proportion to the amount that the rows are stretched [5]. This design is also non adjustable once it is constructed.

Plant Growth

Each plant could grow in containers that support individual plants or in large containers that support many plants. Regardless of the number of plants per container, the containers must provide nutrients, stem support, and a mechanism for plant spacing.

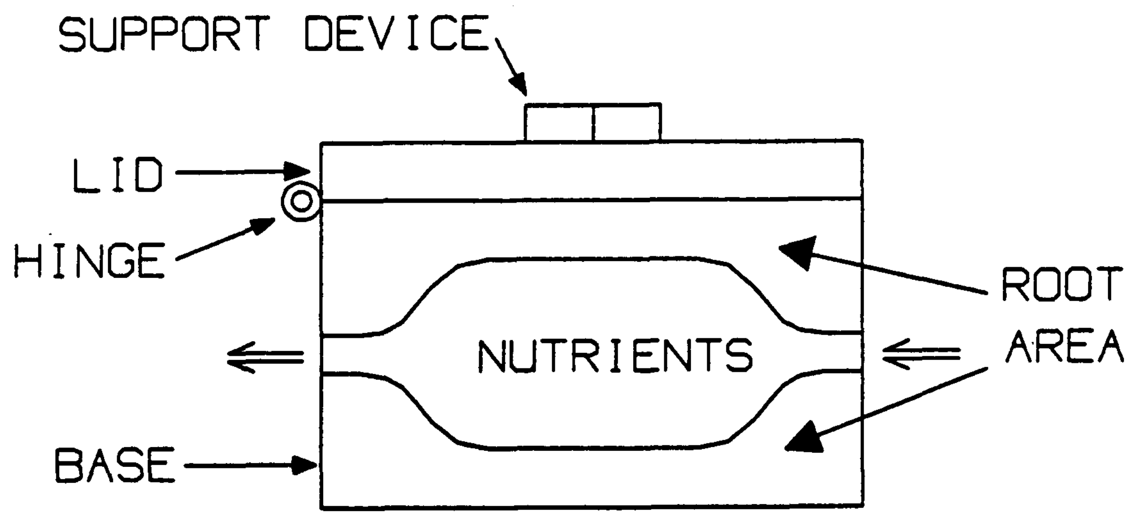


Figure 5a. Membrane Sac Delivery System

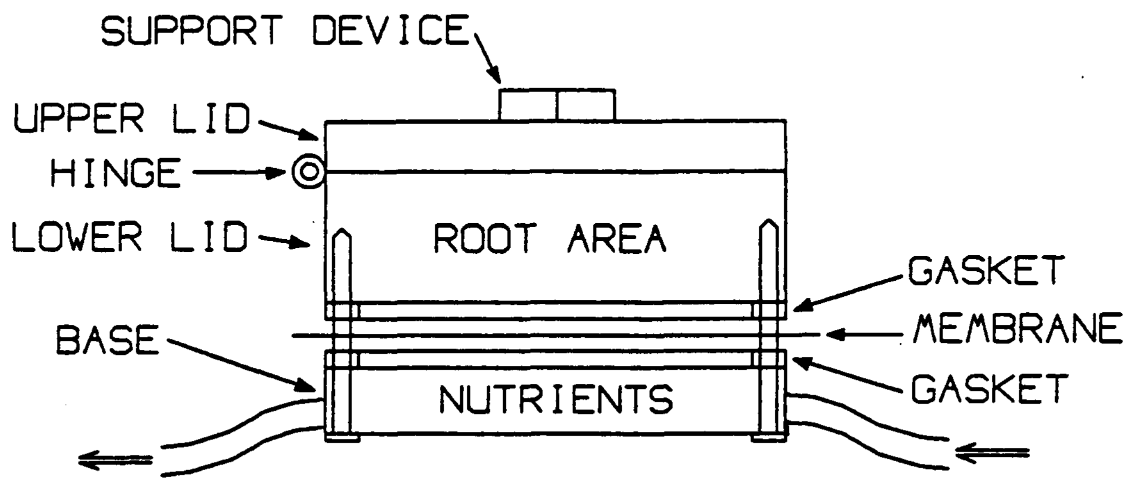


Figure 5b. Membrane Barrier Delivery System

Nutrient delivery. Two possible designs were developed for the nutrient delivery system: A membrane "sac" and a membrane barrier. The membrane "sac" is a membrane shaped into a tube; the ends of the tube serve as input and output locations for the nutrients (Figure 5a). The roots would grow around the membrane "sac" and would obtain the nutrient solution due to surface tension of the roots on the membrane and pressure differences between the atmosphere and the nutrient inside the membrane.

The membrane barrier design has a membrane stretched through the center of the plant container (Figure 5b). It separates the nutrient solution from the roots and provides a planar barrier as compared to the membrane "sac" which is cylindrical.

One drawback to both of these designs is that the container will be the same size for a seedling and a mature plant. There is a minimum root area that will support a mature plant. This area will limit the minimum size of the container and as a result, limit the ability to conserve space.

Plant support. Several different methods were developed for supporting the plants. They are: pneumatic donut, rubber diaphragm, and foam rubber. The pneumatic donut consists of an inflated circular tube shaped like a donut with the plant stem supported in the middle (Figure 6a). As the stem increases in thickness, the donut would slowly release air and decrease the pressure on the stem.

Each row of plants in this design must have a feedback mechanism to sense as well as vary the pressure in the donuts. Since the donut would be very difficult to manufacture with the resources available, this design was not tested.

The rubber diaphragm consists of four separate pieces of rubber sheeting with four equally spaced diametrical cuts in each piece. The four sheets are stacked on top of each other offset at 11.25 degrees (Figure 6b). This forms a very tight closure around the opening. A model of the rubber diaphragm design was built from drafting film instead of rubber and was tested for its ability to support a wooden shaft.

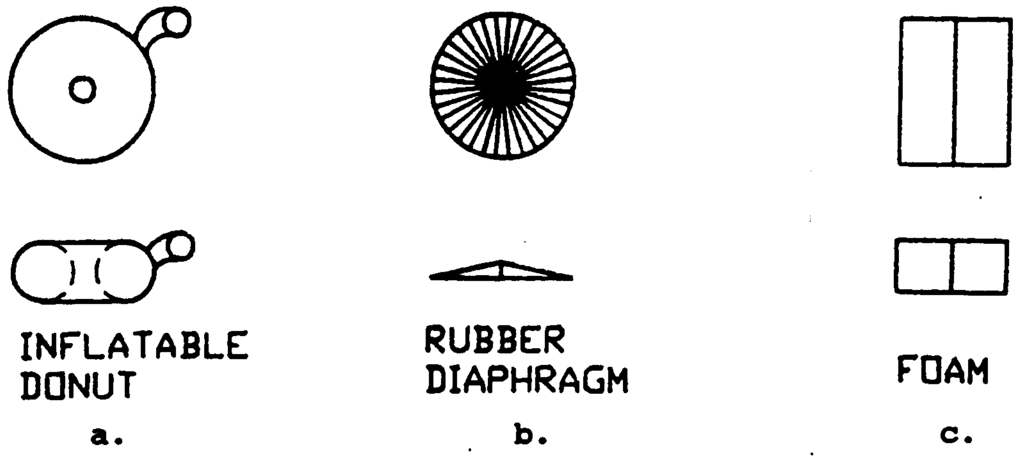


Figure 6. Plant Support Methods

The foam rubber design is very simple (Figure 6c). It consists of two rectangular pieces of foam placed over the opening of the container. The seed or stem would easily be held in place and would not constrict the stem. This design is very inexpensive, lightweight, simple to operate, has low maintenance, and simple to manufacture.

Cleaning and Maintenance

Cleaning. The independent plant containers would need to be refurbished before replanting soybeans in them. The optimal design would not interrupt the movement of the containers through the chamber but should allow the containers to be properly cleaned without removing them from the tray. The biomass in the container must be removed before the containers could be sterilized. This could be accomplished through an automated process where the containers could be opened and washed.

Maintenance. Since the duration of the proposed mission would be approximately three years [9], maintenance requirements are an essential aspect in the design of all components. All parts should be able to endure the entire mission or be serviceable in flight with a minimum of spare parts. Materials should be carefully selected because parts would be turning or sliding across each other or flexing repeatedly.

Any maintenance duties required would be performed by an automated system, or a crew member if only occasional light maintenance was required. The equipment needed for maintenance should be as small and light as possible.

RESULTS TO DATE

Geometrical Configurations

Tray Shape. In order to maximize space utilization, plants will be arranged in rows with equally aged plants in the same row and spacing will expand to follow the natural growth shape for the plants. The simplified shape is a trapezoidal shape approximating the sigmoid growth curve. When the plants are growing in the trays along each row, both the width and height will vary sigmoidally.

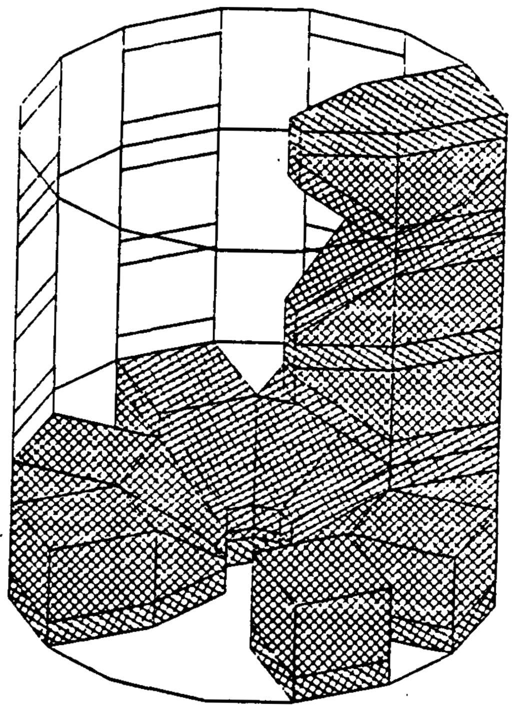


Figure 7. Cylindrical Growth Tray Arrangement

Integration of Growth Trays into Chamber. Through the design of paper models of the growth trays, the most efficient design consisted of trays arranged in a circles, two circles placed back to back against each other, and circle pairs stacked on top of each other (Figure 7). This configuration forms a near perfect cylinder, which would be compatible with the space shuttle cargo bay [1].

This arrangement spaces the growth trays efficiently, but the overall cylinder has several empty spaces. One empty space occurs in the center of each circular pair of trays. In addition, triangular shaped empty spaces near the perimeter of the chamber were created that run uninterrupted for the entire length of the cylinder. There is also a cylindrical empty space that runs lengthwise in the chamber.

The efficiency of this configuration was calculated for a set of estimated values (Appendix B). The estimations and assumptions for the calculations were:

1. A mature soybean plant is 3 feet high and 1 foot wide.
2. A soybean plant reaches maturity height in 40 days and is harvested after 80 days.
3. A soybean plant will grow with roots contained to a volume of 3.5 x 3.5 x 3 to 4 inches deep.
4. Cylinder size inside diameter is 13 feet 8 inches [1].

Calculated volume of plant growth and support area took up 70.3% of the volume and the remaining 29.6% was empty space for equipment and storage. The total volume of one circle of trays and empty space was 1,054,000 in^3 and contained 168 plants. A control volume consisting of concentric circles of plants at maturity including lighting and support totalled 1,014,000 in^3 and contained only 142 plants. The volume per plant ratio for the plant area only decreased by 46% under the control configuration. Volume per plant ratio including support and lighting but without empty space was 38% less and 12% less when including empty space. This shows that space is conserved even under the worst case situation. A realistic value for space reduction is 38% when empty spaces are utilized for other purposes.

The empty spaces could be used in a number of ways. For example, the space in the center could be utilized for seeding equipment, storage, or any other equipment that would require a large unpartitioned space. Also, different crops could utilize some or all of these empty spaces. The triangular and cylindrical spaces running the length of the cylinder would be well suited for air ducts, nutrient tubing, electrical cables, or pathways for moving materials or allowing crew members to move between levels of circular trays.

Integration with Planting and Harvesting Systems. The support containers will be seeded in a row at one end of the tray. The seedling side of each tray forms a circle in the center of the chamber. The centralization of the containers to be seeded facilitates uncomplicated incorporation of the seeding mechanism into the chamber design.

A similar situation exists with the mature plants which are located at the perimeter of the growth chamber. This allows the harvester to move on a track along the outside of the trays to gather all the mature plants.

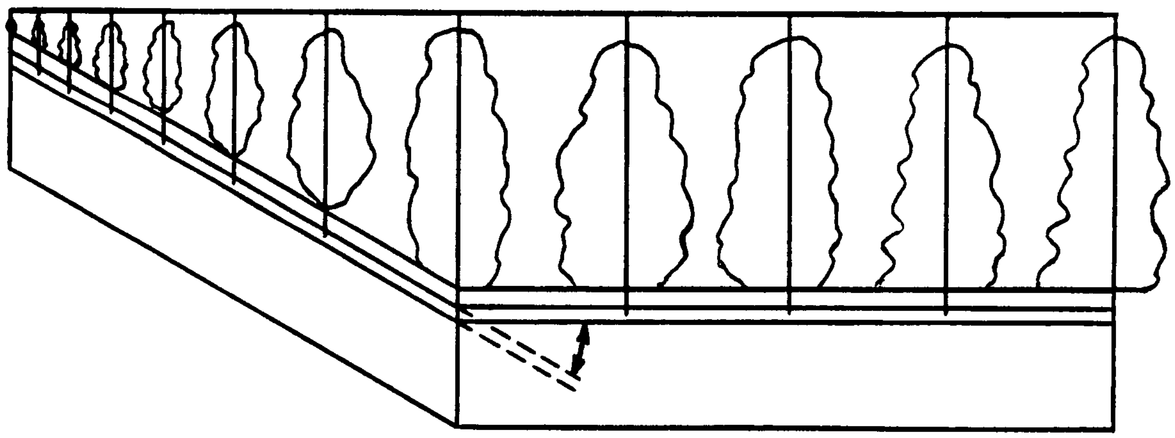


Figure 8. Plant Spacing During Growth

Plant Movement

Varying plant spacing during growth. Initial experiments showed that the trellis design will be able to perform the task of spacing the plants in rows quite efficiently. Problems arose when integrating the trellis design into the current tray and chamber configuration. The trellises must travel in a plane during the rapid growth stages and then shift to an intersecting plane (Figure 8). The trellises must travel across this discontinuity and still support the plants vertically.

The accordion tube has many flaws. One major drawback with the accordion tube design is the large expansion required. The hose must be able to expand at least four times its relaxed length to allow a plant 3 square inches to grow to 1 square foot [6]. This design would also be difficult to clean and reprocess because there would be no way to open the tube up for cleaning and successfully reseal it. For these reasons, the trellis design is preferred over the accordion tube.

Varying Row Spacing. The preferred method for varying the spacings between rows is using a limited robot system controlled by a computer. This is merely a conceptual design thus no further research will be performed in this area.

Plant Growth

Nutrient delivery. The membrane barrier was chosen as the preferred design for nutrient delivery. It is simpler to build and maintain than the membrane sac. The membrane barrier design facilitates removal of the plant roots from the container for cleaning without disturbing the membrane. Also, the membrane barrier lends itself to other potential designs. For example, the use of a more durable, porous metal such as stainless steel could be developed as a barrier [5]. If the membrane should fail on one of the containers, provisions in the design have been made to allow replacement of the membrane.

Plant support. The rubber diaphragm model constructed proved to be successful in its primary goal of rigidly supporting the wooden shaft at all angles of rotation, but had undesirable side effects. One problem with this system is that the diaphragm cannot accept a seed very easily. The diaphragm provides support at a very thin ring around the stem or seed, and the seed would easily slip out if not placed exactly in the center. A seedling would adapt more easily since it would be less prone to slip out.

Another problem discovered while testing the model was that considerable force was exerted on the shaft. If this were a plant stem, the excessive force might injure the plant [8]. The force was not measured, nor is a value of maximum force allowable on a stem known, but because of this force, this design may not be investigated next semester.

The design offering the most promising results is the foam rubber technique. It is simple, easy to construct, and performs the task of supporting the plant early in the growth stage. As the plant becomes larger, the roots will slowly enlarge and completely fill the root area in the container. The roots will then support the plant as a normal plant would in soil. Because of the simplicity of the foam design, this will be the design that will be constructed and tested next semester.

Cleaning and Maintenance

Cleaning. The plant containers were designed with a hinged lid on the container for access to the roots left behind by the harvester (Figure 9). An automated process will open the containers while they are on the underside of each tray and remove the root masses. The containers will then be cleaned by chemical and physical means. A cleaning solution may be pumped through the nutrient delivery system to cleanse the membranes. The cleaning portion of the design is conceptual and will not be constructed or tested.

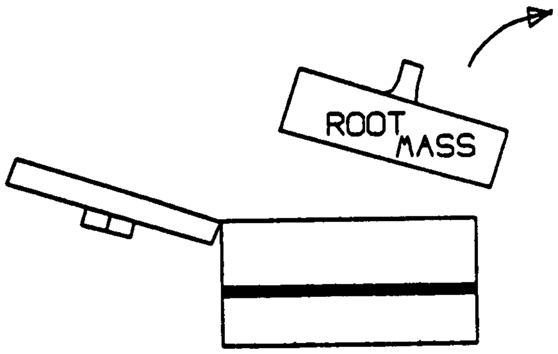


Figure 9. Cleaning of Plant Growth Containers

<u>Maintenance</u>. The plant containers were also designed to allow replacement of the membrane (Figure 5b). The membrane to be used is a plastic PVC based membrane that looks like paper [5]. The membranes should be able to perform for the entire length of the mission. If one should fail, the membranes are replaceable by crew or automation. Screws are removed from the bottom of the container, the base separated from the lower lid, and the membrane removed. A new membrane is slipped in place and the screws re-installed. A watertight seal is provided by rubber gaskets attached to the base and lower lid.

CONCLUSION

The designs presented constitute the components of a system for varying the spacings between soybean plants. The overall shape of the growth trays, as determined by the growth curve for soybeans, is a three dimensional trapezoid. These trays are arranged into pairs of circles back to back and stacked for the entire length of a large cylinder. Within the growth trays, plants will be housed in individual containers incorporating a nutrient delivery system and a plant support mechanism. Nutrients will be provided by a sheet of PVC membrane. The plants will be supported by being placed in between two pieces of foam. Individual containers will be spaced by a half trellis designed to vary the distance between plants for optimum volume usage. The design allows for localized seeding and harvesting mechanisms. In addition, the components have been designed for minimal maintenance and ease of cleaning.

Initial calculations from estimates on plant growth indicate that 38% less volume is required to grow a soybean plant using the tray and chamber configurations developed. This value is comparing the area required per plant in the trapezoidal configuration versus a cylinder housing concentric circles of plants.

PLANS FOR SECOND SEMESTER EFFORT

In the second semester, more data will be gathered in the area of soybean growth. More detailed data on the rate of growth at each stage of plant growth is required to quantitatively design a system for maximum efficiency to conserve space. Once detailed data is gathered and incorporated into the design, then a definite quantitative value of the efficiency may be calculated to determine the overall success of the project.

Also in the second semester, the trellis and membrane barrier designs will be constructed and tested. A test bed to hold the trellises has already been constructed. Modifications to the trellis design and other spacing methods such as the accordion tube will also be tested as well. A plant container will be constructed using the membrane barrier design. It will be tested with live plants to determine whether the roots will have enough area to grow in. If both of these systems succeed, then a final model will be built and retested.

REFERENCES

1. M. Oleson, R.L. Olson, Controlled Ecological Life Support System (CELSS) Conceptual Design Option Study, NASA Contract Report 177421, June 1986

2. R.P. Prince, J.W. Bartok Jr., Plant Spacing for Controlled Environment Plant Growth. Transactions of the ASAE, vol 21 [2]. p.332-336, 1978.

3. R.P. Prince, H.V. Koonz, Lettuce Production from a Systems Approach. ASAE.

4. R. Hinkle, T. Dreschel, The Tubular Membrane Growth System, Bionetics Corporation, August 1985.

5. Jeff Bohren, 1987, Personal Communication, University of Florida, Gainesville, Fl.

6. Ralph P. Prince, 1987, Personal Communication, Biomedical Operations and Research Office, J. F. Kennedy Space Center, Titusville, Fl.

7. Steph Syslo, 1987, Personal Communication, University of Florida, Gainesville, Fl.

8. Kent Tambling, 1987, Personal Communication, University of Florida, Gainesville, Fl.

9. 86-87 EGM 4000/1 Design Class, Final Report for the Advanced Space Design Program, p.3, 1987

APPENDIX A

Trellis Animation Program

The following is the listing for the trellis animation program. The program was written in C on an Amiga 1000 computer. The purpose of the program was to help visualize the effects of plant spacing for the half trellis design.

```
rm=75
include <intuition/intuition.h>
#include <graphics/display.h>
#include <graphics/gfxbase.h>
#include <graphics/gfxmacros.h>
#include <exec/memory.h>
#include <exec/execbase.h>
#include <stdio.h>
#include <hardware/custom.h>
#include <hardware/dmabits.h>
#include <libraries/dos.h>
#include <math.h>

/*
*******************************************************
**************
*
*  Simulations of support and expansion mechanisms.
*  Jim Bledsoe   Nov 1987   University of Florida  EGM
4000/1
*  For the NASA CELSS project and the USRA.
*
*******************************************************
*************** */

extern void *OpenLibrary();
extern struct Screen *OpenScreen();
extern struct Window *OpenWindow();
extern struct IntuiMessage *GetMsg();
extern struct MsgPort *CreatePort();

#define IBPtr struct IntuitionBase *
#define GBPtr struct GfxBase *
#define OneHundred 100.0

#define ScreenX 640L
#define ScreenY 200L
#define ScreenMode HIRES   /*HIRES, INTERLACE, and
NULL*/

struct IntuitionBase *IntuitionBase;
struct GfxBase *GfxBase;
struct Window *w = NULL;
struct Screen *s = NULL;
struct ViewPort *vp = NULL;
struct RastPort *rp = NULL;
struct BitMap bmbuff;
struct RastPort rpbuff;
struct TmpRas tmpras;
UBYTE dobuffer[20] = "0";
UBYTE undobuffer[20] = "0";
struct IntuiText
t0,t1,t2,t3,t10,t11,t12,t20,t21,t22,t30,t31;
struct MenuItem
```

```
m0,m1,m2,m3,m10,m11,m12,m20,m21,m22,m30,m31;
struct Menu menu1,menu2,menu3,menu4,menustop;
PLANEPTR area_raster = NULL;
PLANEPTR area_raster2 = NULL;
LONG trayulx,trayuly,trayurx,trayury;
LONG trayllx,traylly,traylrx,traylry;
LONG xdiff,ydiff;
FLOAT mx,my;
USHORT animate_flag = FALSE;
USHORT done = FALSE;
USHORT gad_added = FALSE;
LONG num_segs,num_trels,max_stretch_angle,timeconst;
LONG num_divs;
FLOAT bar_length;
struct trellis {
  FLOAT Y;
  USHORT Visible;
  LONG PlantSize;
};
struct trellis trellarray[51];

struct StringInfo info = {
  dobuffer, undobuffer, 0,20,0, 0,0,0,0,0, 0,0, NULL
};

USHORT BorderVectors[] = {0,0,206,0,206,13,0,13,0,0};
struct Border gborder = {
  -2,-3, 3,0,JAM1, 5, BorderVectors, NULL
};

struct IntuiText gtext = {
  3,0,JAM2, 79,12, NULL,"Input window!!?!",NULL
};

struct Gadget gadget = {
  NULL, ScreenX/2-102,80, 203,10, GADGHCOMP,
  LONGINT | RELVERIFY | STRINGCENTER,
  STRGADGET, (APTR)&gborder, NULL, &gtext, 0,
(APTR)&info,1,NULL
};

struct NewScreen ns = {
  0, 0, ScreenX, ScreenY, 3, 5, 1, ScreenMode,
CUSTOMSCREEN, NULL,
  "Graphical simulation of a Trellis support.  V1.01",
NULL, NULL
};

struct NewWindow nw = {
  0, 2, ScreenX, ScreenY-2, 5, 1,
  MENUPICK | CLOSEWINDOW | GADGETUP,
  WINDOWCLOSE | WINDOWDEPTH | ACTIVATE | SMART_REFRESH,
  NULL, NULL, "Graphical simulation of Trellis support.
V1.01",
  NULL, NULL, 0, 0, ScreenX, ScreenY, CUSTOMSCREEN
};

/*************************/

  animate() {
    register USHORT i,j;
    static LONG xl,xr,dy,y;
    static LONG sx,sy;
    static FLOAT yprop,x,ddx;
    SetRast(&rpbuff,0L);
    draw_tray();
    SetAPen(&rpbuff, 1L);
    for (i=0; i<num_trels; i++) {
      x = sqrt(trellarray[i].Y*OneHundred) +
timeconst/OneHundred;
      if (x > OneHundred) x -= OneHundred;
      trellarray[i].Y = x*x/OneHundred;  /*function of
movement*/
      yprop = trellarray[i].Y/OneHundred;
      xl = trayllx - (LONG)(xdiff*yprop);
      xr = traylrx + (LONG)(xdiff*yprop);
      y =  traylly - (LONG)(ydiff*yprop);
      ddx = (FLOAT)(xr - xl)/num_divs;
      dy = sqrt(bar_length*bar_length - ddx*ddx);
      Move(&rpbuff, (LONG)(xl*mx),(LONG)(y*my));
      for (j=1; j<num_segs; j++) {
        sx = xl + (1 + 2*(j-1))*ddx;
        sy = (j%2) ? y + dy : y - dy;
        Draw(&rpbuff, (LONG)(sx*mx),(LONG)(sy*my));
      }
      Draw(&rpbuff, (LONG)(xr*mx),(LONG)(y*my));
      Draw(&rpbuff, (LONG)(xl*mx),(LONG)(y*my));
    }
    ClipBlit(&rpbuff, 4,12, rp, 4,12,
ScreenX-8,ScreenY-14, 0xC0);
  }

  init_trellises() {
    register USHORT i;
    for (i=0; i<50; i++) {
      trellarray[i].Y = 0.0;
      trellarray[i].Visible = FALSE;
      trellarray[i].PlantSize = 10; /*not implemented
yet*/
    }
    for (i=0; i<num_trels; i++) {
      trellarray[i].Y =
OneHundred*i*i/num_trels/num_trels; /*function of
motion*/
      trellarray[i].Visible = TRUE;
      trellarray[i].PlantSize = 10; /*not implemented
yet*/
    }
```

```
   }

   get_input() {
     ULONG class,code,value;
     ULONG menu_num, item_num;
     struct Gadget *gadgptr;
     struct IntuiMessage *message;
     while((message=(struct IntuiMessage
*)GetMsg(w->UserPort))!=NULL) {
       class = message->Class;
       code = message->Code;
       if ((class == GADGETUP) || (class == GADGETDOWN))
{
         gadgptr = (struct Gadget *)message->IAddress;
         printf("Illegal!  there aren't supposed to be
any gadgets!!\n");
       }
       ReplyMsg(message);
       switch (class) {
         case CLOSEWINDOW:
           done = TRUE;
           break;
         case MENUPICK:
           menu_num = MENUNUM(code);
           item_num = ITEMNUM(code);
           if (menu_num==0) {
               switch (item_num) {
                 case 0:
newmax:
                   value = prompt_mes("Enter new top
width (10-960)",
                       10,960,trayurx-trayulx);
                   if (value<traylrx-trayllx) goto
newmax;
                   trayulx = 500 - value/2;
                   trayurx = 500 + value/2;
                   calc_consts();
                   break;
                 case 1:
newmin:
                   value = prompt_mes("Enter new bottom
width (10-950)",
                       10,950,traylrx-trayllx);
                   if (value>trayurx-trayulx) goto
newmin;
                   trayllx = 500 - value/2;
                   traylrx = 500 + value/2;
                   calc_consts();
                   break;
                 case 2:
                   value = prompt_mes("Enter new track
height(10-600)",
                       10,600,traylly-trayuly);
                   trayuly = 350 - value/2;
```

```
                    traylly = 350 + value/2;
                    trayury = trayuly;  traylry = traylly;
                    calc_consts();
                    break;
                  case 3:
                    value = prompt_mes("Enter adjustment
(+=up -=down  max 200)",
                        -200,200,0);
                    trayuly -= value;
                    traylly -= value;
                    trayury = trayuly;  traylry = traylly;
                    calc_consts();
                    break;
                  default:
                    printf("Illegal menu item number!\n");
                    break;
                } /*switch item_num*/
            }
            else if (menu_num==1) {
                switch (item_num) {
                  case 0:
                    num_segs = prompt_mes("Enter number of
trellis segments (1-100)",
                        1,100,num_segs);
                    calc_consts();
                    break;
                  case 1:
                    num_trels = prompt_mes("Enter number
of trellises (1-50)",
                        1,50,num_trels);
                    calc_consts();
                    break;
                  case 2:
                    max_stretch_angle = prompt_mes("Enter
max stretch angle (1-45)",
                        1,45,max_stretch_angle);
                    calc_consts();
                    break;
                  default:
                    printf("Illegal menu item number!\n");
                    break;
                } /*switch item_num*/
            }
            else if (menu_num==2) {
                switch (item_num) {
                  case 0:
                    timeconst = prompt_mes("Enter timing
constant (1-1000)",
                        1,1000,timeconst);
                    break;
                  case 1:
                    animate_flag = FALSE;
                    break;
                  case 2:
                    animate_flag = TRUE;
                    ClearMenuStrip(w);
                    SetMenuStrip(w,&menustop);
                    break;
                  default:
                    printf("Illegal menu item number!\n");
                    break;
                } /*switch item_num*/
            }
            else if (menu_num==3) {
                switch (item_num) {
                  case 0:
                    mx = (FLOAT)ScreenX/1000;  my =
(FLOAT)ScreenY/700;
                    m30.Flags = ITEMTEXT | HIGHCOMP |
ITEMENABLED | CHECKIT | CHECKED;
                    m31.Flags = ITEMTEXT | HIGHCOMP |
ITEMENABLED | CHECKIT;
                    SetRGB4(vp,  0L, 0L, 0L, 0L);
/*white*/
                    SetRGB4(vp,  1L, 15L, 15L, 15L);
/*white*/
                    SetRGB4(vp,  2L,  9L,  9L,  9L);
/*grey*/
                    SetRGB4(vp,  5L, 12L, 5L, 15L);
/*purple*/
                    break;
                  case 1:
                    mx = (FLOAT)ScreenX/955.55;  my =
(FLOAT)ScreenY/700;
                    m30.Flags = ITEMTEXT | HIGHCOMP |
ITEMENABLED | CHECKIT;
                    m31.Flags = ITEMTEXT | HIGHCOMP |
ITEMENABLED | CHECKIT | CHECKED;
                    SetRGB4(vp,  0L, 15L, 15L, 15L);
/*white*/
                    SetRGB4(vp,  2L,  0L,  0L,  0L);
/*grey*/
                    SetRGB4(vp,  5L, 15L, 15L, 15L);
/*purple*/
                    break;
                  default:
                    printf("Illegal menu item number!\n");
                    break;
                } /*switch item_num*/
            }
            else if (menu_num == 31) {
              animate_flag = FALSE;
              ClearMenuStrip(w);
              SetMenuStrip(w,&menu1);
            }
            else {
                printf("Illegal menu number!\n");
                break;
```

```
          } /*if menu_num*/
        default:
          break;
      } /*switch class*/
    } /*while message*/
  }

  LONG prompt_mes(string,min,max,current)
  LONG min,max,current;
  char string[]; {
    LONG value;
    ULONG class;
    struct IntuiMessage *message;
    ClearMenuStrip(w);
    ClipBlit(rp, 90,50, &rpbuff, 90,50, ScreenX-160,80,
0xC0);
    SetAPen(rp, 0);
    SetOPen(rp, 1);
    RectFill(rp, 100,60, ScreenX-100,120);
    SetAPen(rp, 1);
    gtext.IText = string;
    gtext.LeftEdge = 102 - 4*strlen(string);
    itoa(current,dobuffer);
    itoa(current,undobuffer);
    AddGadget(w, &gadget, 0);
    RefreshGadgets(&gadget,w,NULL);
    value = min-1;
    while (value<min || value>max) {
      class = NULL;
      while(class != GADGETUP) {
        message = (struct IntuiMessage
*)GetMsg(w->UserPort);
        class = message->Class;
      }
      value = atoi(dobuffer);
    }
    RemoveGadget(w, &gadget);
    ClipBlit(&rpbuff, 80,50, rp, 80,50, ScreenX-170,80,
0xC0);
    SetMenuStrip(w,&menu1);
    return(value);
  }

  itoa(n,s)
  char s[];
  LONG n; {
    SHORT i,sign;
    if ((sign=n)<0) n = -n;
    i = 0;
    do {
      s[i++] = n%10 + '0';
    } while ((n /= 10) > 0);
    if (sign<0) s[i++] = '-';
    s[i] = '\0';
    reverse(s);
  }

  reverse(s)
  char s[]; {
    SHORT c,i,j;
    for (i=0, j=strlen(s)-1; i<j; i++, j--) {
      c = s[i];
      s[i] = s[j];
      s[j] = c;
    }
  }


main () {
  USHORT i,j;
  IntuitionBase = (IBPtr)
OpenLibrary("intuition.library", 0L);
  GfxBase = (GBPtr) OpenLibrary("graphics.library", 0L);
  if (IntuitionBase && GfxBase) {
    if (nw.Screen = s = OpenScreen(&ns)) {
      if (!(w = OpenWindow(&nw))) {
        CloseScreen(s);
        goto out;
      }
    } else {   /*screen not opened*/
      goto out;
    }
    /*ShowTitle(s,1L);*/
    vp = &w->WScreen->ViewPort;  rp = &s->RastPort;
    InitBitMap(&bmbuff,3,ScreenX,ScreenY);
    for (i=0; i<3; i++) {
      if ((bmbuff.Planes[i] =
(PLANEPTR)AllocRaster(ScreenX,ScreenY))==NULL) {
        printf("Could not get video memory!!!
Grrrr!!!\n");
        if (i) {for (j=0; j<i; j++)
FreeRaster(bmbuff.Planes[j], ScreenX,ScreenY);}
        goto big_out;
      }
    }
    InitRastPort(&rpbuff);  rpbuff.BitMap = &bmbuff;
    set_color_registers();  set_menu();
    ScreenToFront(s);  SetMenuStrip(w,&menu1);
    init_vars();

    main_program();

    ClearMenuStrip(w);
    for (i=0; i<3; i++) FreeRaster(bmbuff.Planes[i],
ScreenX,ScreenY);
big_out:
    ScreenToBack(s);
    CloseWindow(w);  CloseScreen(s);
    }
  out:
    if (GfxBase) CloseLibrary(GfxBase);
    if (IntuitionBase) CloseLibrary(IntuitionBase);
  }

    set_color_registers() {
      SetRGB4(vp,  0L,  0L,  0L,  0L); /*black*/
      SetRGB4(vp,  1L, 15L, 15L, 15L); /*white*/
      SetRGB4(vp,  2L,  9L,  9L,  9L); /*grey*/
      SetRGB4(vp,  3L,  3L,  3L, 13L); /*blue*/
      SetRGB4(vp,  4L,  1L,  9L,  1L); /*green*/
      SetRGB4(vp,  5L, 12L,  5L, 15L); /*purple*/
      SetRGB4(vp,  6L, 15L, 14L,  0L); /*yellow*/
      SetRGB4(vp,  7L, 10L, 10L, 12L); /*silver*/
      SetDrMd(rp, JAM1);  SetAPen(rp, 1L);
      SetDrMd(&rpbuff, JAM1);  SetAPen(&rpbuff, 1L);
    }

    main_program() {
      while (!done) {
        if (animate_flag) animate();
        get_input();
      }
    }

    draw_tray() {
      SetAPen(&rpbuff, 2L);
      Move(&rpbuff,
  (LONG)(trayulx*mx),(LONG)(trayuly*my));
      Draw(&rpbuff,
  (LONG)((trayulx-10)*mx),(LONG)(trayuly*my));
      Draw(&rpbuff,
  (LONG)((trayllx-10)*mx),(LONG)(traylly*my));
      Draw(&rpbuff,
  (LONG)(trayllx*mx),(LONG)(traylly*my));
      Draw(&rpbuff,
  (LONG)(trayulx*mx),(LONG)(trayuly*my));
      Move(&rpbuff,
  (LONG)(trayurx*mx),(LONG)(trayury*my));
      Draw(&rpbuff,
  (LONG)((trayurx+10)*mx),(LONG)(trayury*my));
      Draw(&rpbuff,
  (LONG)((traylrx+10)*mx),(LONG)(traylry*my));
      Draw(&rpbuff,
  (LONG)(traylrx*mx),(LONG)(traylry*my));
      Draw(&rpbuff,
  (LONG)(trayurx*mx),(LONG)(trayury*my));
    }

    init_vars() {
      trayulx = 232;
      trayllx = 462;
      traylrx = 538;
    trayurx = 768;
    trayuly = 60;
    trayury = 60;
    traylly = 614;
    traylry = 614;
    num_segs = 6;
    num_trels = 7;
    max_stretch_angle = 5;
    timeconst = 50;
    mx = (FLOAT)ScreenX/1000;  my = (FLOAT)ScreenY/700;
    calc_consts();
  }

  calc_consts() {
    xdiff = trayllx - trayulx;
    ydiff = traylry - trayury;
    num_divs = 2 + 2*(num_segs - 2);
    bar_length =
(FLOAT)(trayulx-trayurx)/num_divs/cos(max_stretch_angle/
57.29578);
    init_trellises();
  }

  CreateMes(x,left,top,mesg)
  struct IntuiText *x;
  short left,top;
  UBYTE *mesg; {
    x->FrontPen = 0;  x->BackPen = 1;
    x->DrawMode = JAM1;
    x->LeftEdge = left;  x->TopEdge = top;
    x->ITextFont = NULL;
    x->IText = mesg;
    x->NextText = NULL;
  }

  CreateItem(name,item,next,left,top,flags)
  UBYTE *name;
  USHORT left,top;
  ULONG flags;
  struct MenuItem *item;
  struct MenuItem *next; {
    item->NextItem = next;
    item->LeftEdge = left;
    item->TopEdge = top;
    item->Width = 250;
    item->Height = 10;
    item->Flags = ITEMTEXT | HIGHCOMP | ITEMENABLED |
flags;
    item->MutualExclude = NULL;
    item->ItemFill = (APTR)name;
    item->SelectFill = NULL;
    item->Command = NULL;
    item->SubItem = NULL;
  }
  set_menu() {
    CreateMes(&t0, CHECKWIDTH+2, 1, "Alter max width");
    CreateItem(&t0, &m0, &m1, 5, 0, 0);
    CreateMes(&t1, CHECKWIDTH+2, 1, "Alter min width");
    CreateItem(&t1, &m1, &m2, 5, 10, 0);
    CreateMes(&t2, CHECKWIDTH+2, 1, "Alter height");
    CreateItem(&t2, &m2, &m3, 5, 20, 0);
    CreateMes(&t3, CHECKWIDTH+2, 1, "Shift vertically");
    CreateItem(&t3, &m3, NULL, 5, 30, 0);
    CreateMes(&t10, CHECKWIDTH+2, 1, "Alter num of
segments");
    CreateItem(&t10, &m10, &m11, 5, 0, 0);
    CreateMes(&t11, CHECKWIDTH+2, 1, "Alter num of
trellises");
    CreateItem(&t11, &m11, &m12, 5, 10, 0);
    CreateMes(&t12, CHECKWIDTH+2, 1, "Alter max stretch
angle");
    CreateItem(&t12, &m12, NULL, 5, 20, 0);
    CreateMes(&t20, CHECKWIDTH+2, 1, "Change speed");
    CreateItem(&t20, &m20, &m21, 5, 0, 0);
    CreateMes(&t21, CHECKWIDTH+2, 1, "Stop");
    CreateItem(&t21, &m21, &m22, 5, 10, 0);
    m21.Flags = ITEMTEXT | HIGHCOMP ;
    CreateMes(&t22, CHECKWIDTH+2, 1, "Start");
    CreateItem(&t22, &m22, NULL, 5, 20, 0);
    m22.Flags = ITEMTEXT | HIGHCOMP | ITEMENABLED |
CHECKIT | CHECKED;
    CreateMes(&t30, CHECKWIDTH+2, 1, "Set for screen");
    CreateItem(&t30, &m30, &m31, 5, 0, 0);
    m30.Flags = ITEMTEXT | HIGHCOMP | ITEMENABLED |
CHECKIT | CHECKED;
    CreateMes(&t31, CHECKWIDTH+2, 1, "set for printer");
    CreateItem(&t31, &m31, NULL, 5, 10, 0);
    m31.Flags = ITEMTEXT | HIGHCOMP | ITEMENABLED |
CHECKIT;
    menu1.NextMenu = &menu2;
    menu1.LeftEdge = 0;  menu1.TopEdge = 0;
    menu1.Width = 100;  menu1.Height = 100;
    menu1.Flags = MENUENABLED;
    menu1.MenuName = "Track";
    menu1.FirstItem = &m0;
    menu2.NextMenu = &menu3;
    menu2.LeftEdge = 100;  menu1.TopEdge = 0;
    menu2.Width = 100;  menu1.Height = 100;
    menu2.Flags = MENUENABLED;
    menu2.MenuName = "Trellis";
    menu2.FirstItem = &m10;
    menu3.NextMenu = &menu4;
    menu3.LeftEdge = 200;  menu1.TopEdge = 0;
    menu3.Width = 100;  menu1.Height = 100;
    menu3.Flags = MENUENABLED;
    menu3.MenuName = "Animation";
    menu3.FirstItem = &m20;
```

```
        menu4.NextMenu = NULL;
        menu4.LeftEdge = 300;  menu1.TopEdge = 0;
        menu4.Width = 100;  menu1.Height = 100;
        menu4.Flags = MENUENABLED;
        menu4.MenuName = "Aspect ratio";
        menu4.FirstItem = &m30;
        menustop.NextMenu = NULL;
        menustop.LeftEdge = 0;  menu1.TopEdge = 0;
        menustop.Width = ScreenX-2;  menu1.Height = 10;
        menustop.Flags = MENUENABLED;
        menustop.MenuName = "
    Stop Animation!";
        menustop.FirstItem = NULL;
      }
```

APPENDIX B

<u>Calculations of Volume and Efficiency of Trapezoidal Arrangement</u>

The following is the derivation and results of the calculations for efficiency of the trapezoidal tray arrangement measured against mature plant spacing in concentric circles.

Results of Volume Calculations:

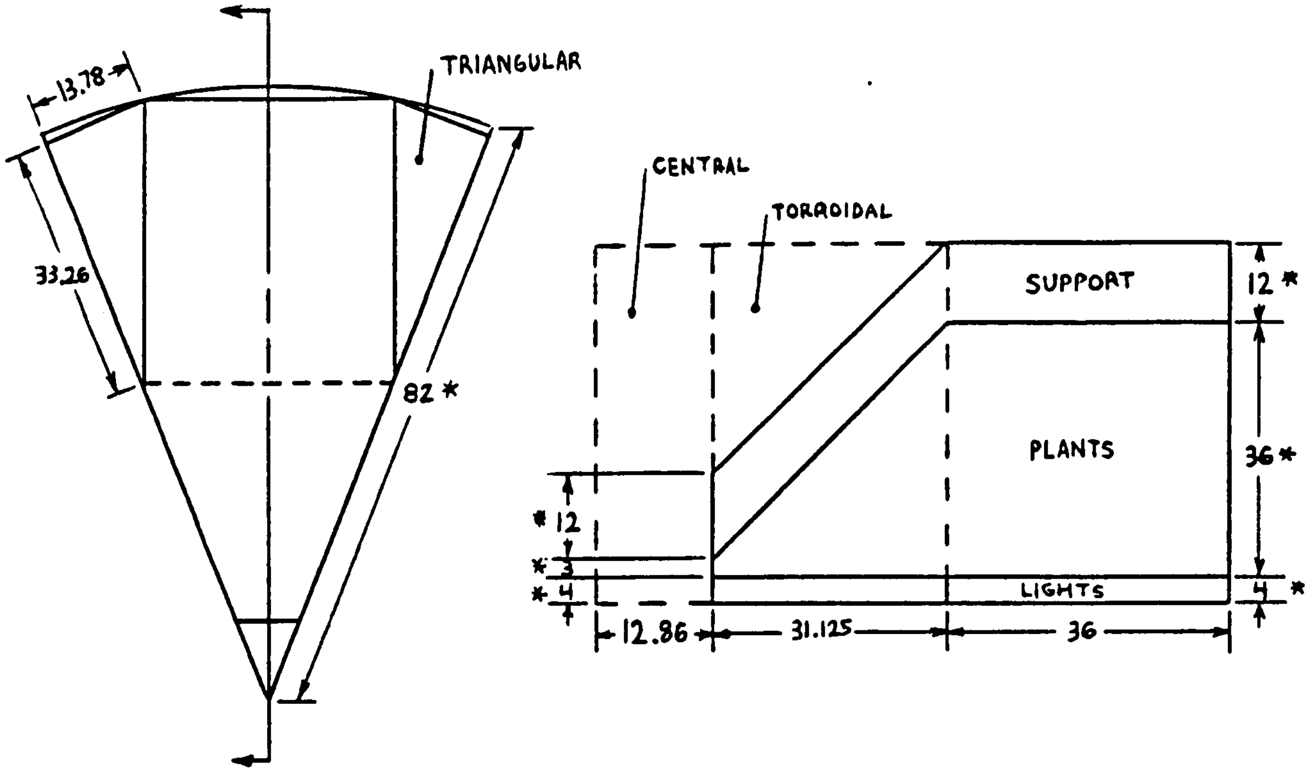
Unlabeled values are in cubic inches.

Mature height: 36 inches	Mature width: 12 inches
Cylinder diameter: 82 inches	Minimum width: 3.5125 inches

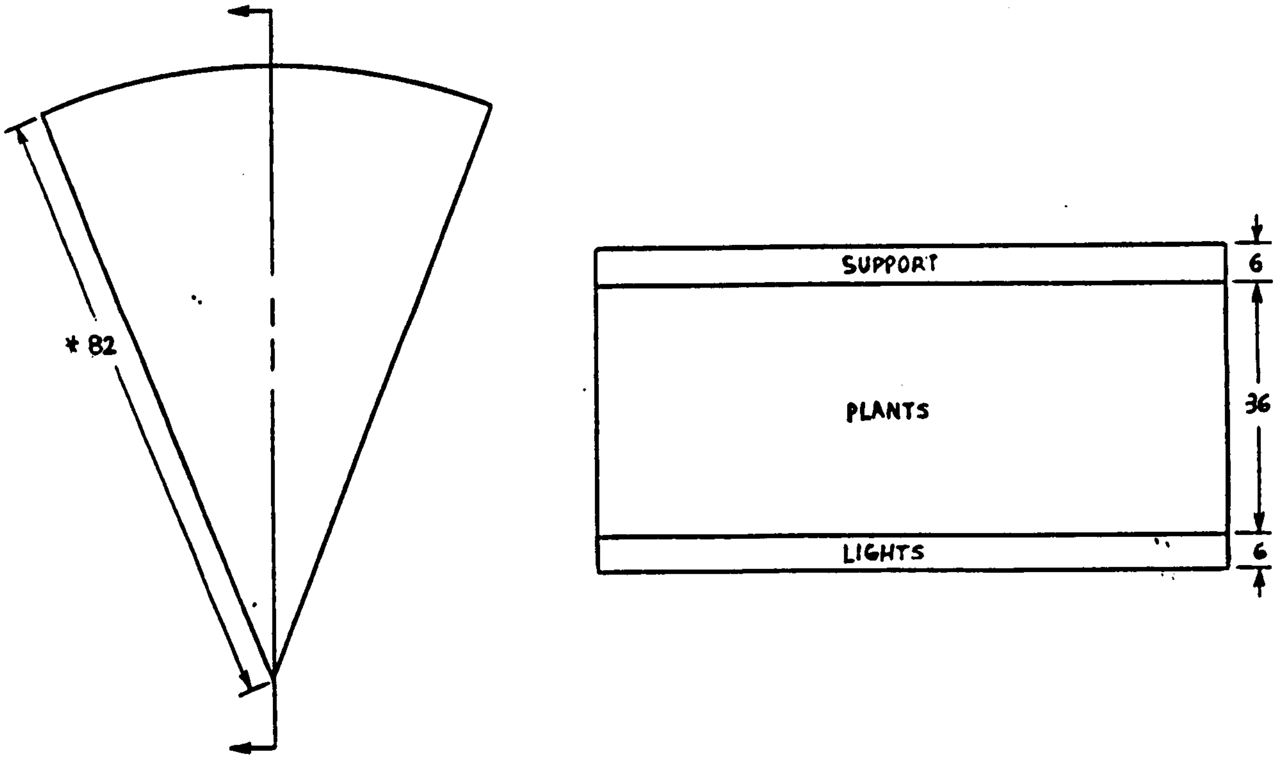
Area Description	Single Tray Volume	Percent of Total Volume	Eight Tray Volume
Plants	60,363	45.8	482,901
Lights	8,089	6.1	64,712
Support	24,267	18.4	194,135
Plants, lights and support	92,719	70.3	741,748
Triangular empty space	23,827	18.1	190,613
Central empty space	3,570	2.7	28,564
Torroidal empty space	11,598	8.8	92,784
Total empty space	38,995	29.6	311,961
Summed Total volume	131,714	99.9	1,053,709
Calculated Total volume	132,403	100.0	1,059,222
Cylinder volume	137,306	100.0	1,098,452
Control volume	137,306	100.0	1,098,452
Number of plants in trapezoid design	21		168
Number of plants in cylinder	17.75		142
Percent increase in number of plants			+18 %

Results of Efficiency Calculations (Figure 10):

Volume Description	Volume/Plant	% Difference
Plant volume for trapezoid	2,874	
Plant volume for control	5,355	
		-46 %
Plant, light, and tray volume for trapezoid	4,415	
Plant, light, and tray volume for control	7,141	
		-38 %
Total volume for trapezoid	6,272	
Total volume for control	7,141	
		-12 %



Trapezoidal



Cylindrical

Figure 10. Efficiency Calculations Diagrams