Chapter 4

Plant Module and Experimental Design

This chapter describes the structure of prototype, the L-system prototype, the qualitative model, the L-system string and plant interpretation, the data collection, the growth function, the quantitative model, the visualization, and the model evaluation. The plant module is based on the bracketed L-systems, which represent the structure of plant.

4.1 Structure of Prototype

The biological data from the actual plant observation are very important for simulating the plant development. A flow diagram of a prototype of simulation and visualization of the plant growing is shown in Figure 4.1.

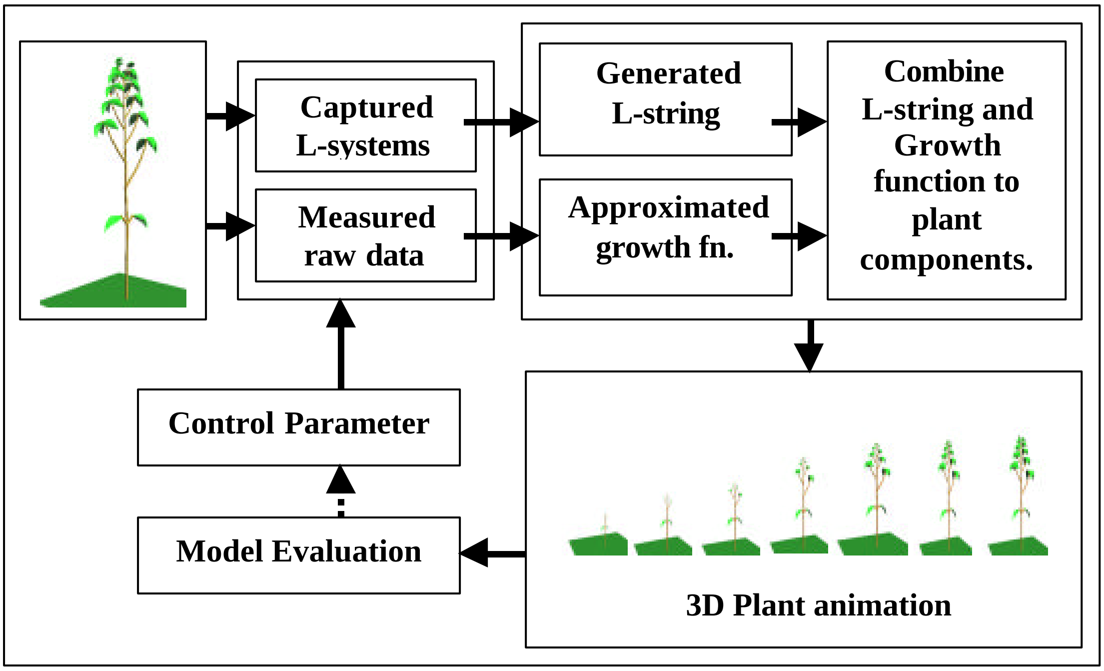


Figure 4.1: Diagram of plant simulation and visualization.

This section presents a prototype for creating computer models that capture the development of plants using L-systems and mathematical model incorporating biological data. The L-system is used for qualitative model in order to represent the plant topology and development. This method has six steps, (1) defining a qualitative model constructed from observations of plant growth in their life cycle, (2) measuring key characteristics collected from actual plants, (3) converting raw data to growth functions based on sigmoidal function approximations, (4) defining a quantitative model composed from the qualitative model and growth function, (5) visualizing the quantitative model, and (6) evaluating the models. The design of L-system in this thesis is based on bracketed L-systems.

4.2 L-system Prototype

The design of L-system in this thesis is based on bracketed L-systems. The symbols are used to represent the plant components. All of these symbols are described in Table 4.1.

Table 4.1: Symbols used in plant growth L-system.

Symbols	Meaning
I	To generate the plant internodes
i	To generate the plant short internodes
P	To generate the plant petioles
p	To generate the plant short petioles
A	To generate the plant apices
L	To generate the plant leaves
F	To generate the plant flowers
+	Roll counter-clockwise to positive Z-axis by angle δ_z, using rotation matrix $R_z(\delta_z)$
-	Roll clockwise to positive Z-axis by angle δ_z, using rotation matrix $R_z(-\delta_z)$
&	Roll counter-clockwise to positive Y-axis by angle δ_y, using rotation matrix $R_y(\delta_y)$
^	Roll clockwise to positive Y-axis by angle δ_y, using rotation matrix $R_y(-\delta_y)$
\	Roll counter-clockwise to positive X-axis by angle δ_x, using rotation matrix $R_x(\delta_x)$
/	Roll clockwise to positive X-axis by angle δ_x, using rotation matrix $R_x(-\delta_x)$
|	Roll back, using rotation matrix $R_y(180)$

Symbols	Meaning
[	Push the current state of the turtle onto a pushdown stack to create a new branch
]	Pop a state from the stack and make it the current state of the turtle to close the branch

The simple L-system string is

I [-I] I [+I] I [/I] I [\I] I I where $\delta_x = \delta_y = \delta_z = 70$, and $\alpha_x = \alpha_y = \alpha_z = 0$.

This string has ten internodes, six for main stem, and four for petiole in four directions. The visualized image is shown in Figure 4.2. The matrices $R_z(-\delta_z)$, $R_z(\delta_z)$, $R_x(-\delta_x)$, and $R_x(\delta_x)$ are used for calculating the symbol string [-I], [+I], [/I], and [\I], respectively.

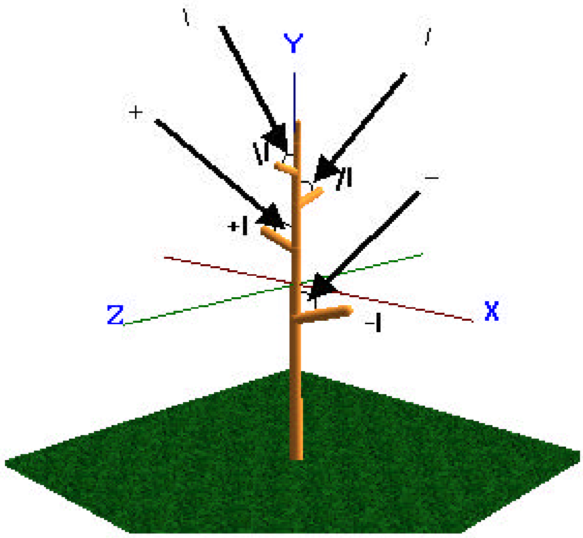


Figure 4.2: A simple L-system Interpretation.
I [-I] I [+I] I [/I] I [\I] I I.

The L-system description of a plant consists of a set of iterations, a set of directional and sizing parameters, initial string, a set of production rules, and a set of terminating productions. The statement

Plant_Name {
    Iterations=*N*
    Angle=***d***
    Diameter=*D*
    Axiom=***w***
    Production 1

Production 2
…
Production *n*
Endrule
Endproduction 1
Endproduction 2
…
Endproduction *m*
}

is the format of L-system.

The meaning of each keyword is given as follows:

Plant_Name

Plant_Name is a name of plant module.

Iterations=N

This input sets the number of iterations for selecting and rewriting the production rules. Each production rule is selected according to the appearance of the symbols in the current string. N is an integer greater than –1.

*Angle=**d***

This angle(***d***) is used to set the angle of a the branch. For example, '-' is to turn right by an angle ***d***, '^' is to pitch up by an angle ***d***, and '/' is to roll right by an angle ***d*** degree.

Diameter=D

This diameter is used to set the diameter of the first internode. The other diameter of internodes and petioles are set follow by the equation of first internode. The unit of *D* is centimeters.

*Axiom=**w***

This string is used to set the start status of the plant. Every start stem is located at the origin (0,0,0), and pointed towards the positive Y axis. The three angles for a three-dimensional space ($\alpha_x,\alpha_y,\alpha_z$) are set to zero for the first internode.

Production 1… Production n

Each production consists of a predecessor and a successor. The format of production is given below.

Predecessor=Successor

The *predecessor* is a symbol and the *successor* is a symbol string. The symbol is a set of character, and the symbol string is a set of character string.

Endrule

To terminate the rewriting of a production rule, a terminating symbol must be substituted to the corresponding symbol used by the previous production rule. The substitution rules are defined in the *endproduction* 1 to the *endproduction* m. The *endproduction* rules are called at the N^{th} iteration.

Endproduction 1 ... Endproduction m

The format of *endproduction* is the same as the production. The *endproduction* is a symbol that used to specify special plant symbols. These symbols are not defined in Table 4.1.

Character "{" and "}"

The character *"{"* and *"}"* are the beginning and the end of L-systems structure, respectively.

4.3 Qualitative Model

The modeling process begins with the specification of the qualitative model. It captures the aspects of a plant which can be obtained through the observations and are deemed essential to its form and development. These include the topology and the sequence of activities of various plant modules. The main components of the plant are distinguished and their developmental stages are identified. The connections between these components are also defined. In this thesis, the qualitative parameters are obtained from a soybean.

The qualitative parameters of the soybean model consists of three main parts: internodes, petioles, and leaves. The simulation begins with first pair of leaves. This

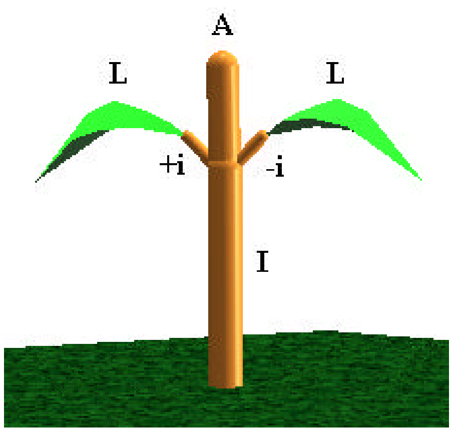


Figure 4.3: Axiom = I[+iL][-iL]A.

is captured by the L-system axiom, or the initial string of modules. The axiom in Figure 4.3 represents an internode *I*, a pair of leave *L* with their short internode *i*, and an apex *A*. The apex *A* is initially contained within the petioles. After some iterations, the apex *A* is substituted by an internode *I*, a right petiole [-*P*], an internode *I*, a left petiole [+*B*] and an apex *A* shown by the following production rule.

A = I[-P]I[+B]A

Each petiole consists of some internodes, some short petioles, a left leaf, a right leaf, and a middle leaf. The production rules of the left and right petioles are defined as follows:

P = IIII[\pL][/pL][-pL]
B = IIII[\pL][/pL][+pL]

The "Endrule" is defined as follows:

B = IL
P = IL
A = IL

The above *Endrules* are called at the last iteration. The left petiole (B), the right petiole (P) as well as the apex (A) are substituted by an internode and a leaf. By using the previous defined production rules with some specific parameters, the L-system description of a soybean is given in the following format.

```
Soybean {
        Iterations=6
        Angle=45
        Diameter=1.5
        Axiom=I[+iL][-iL]A
        A = I[-P]I[+B]A
        P = IIII[\pL][/pL][-pL]
        B = IIII[\pL][/pL][+pL]
        Endrule
        B = IL
        P = IL
        A = IL
}
```

In the above L-systems code, the number of iteration is six; the initial branch angle is 45 degrees, and the diameter of first internode is 1.5 centimeters. After the sixth iteration, the *Endrule* productions are called to terminate the substitution process.

The last symbol string at the sixth iteration is

I[+iL][-iL]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL][-pL]] I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL] [-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL] [/pL][-pL]]I[+IIII[\pL][/pL][+pL]]**A**

The string *P* and *B* are not appear in the last string of the above symbol string, but *A* remains in the last letter. *A* is replaced with *endproduction A=IL*. Observe that the *endproduction B=IL* and *P=IL* is not necessary to define in the soybean module. After the above string is replaced with the *endproduction,* the final string is

I[+iL][-iL]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL][-pL]] I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL] [-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL] [/pL][-pL]]I[+IIII[\pL][/pL][+pL]]**IL**

After the sixth iteration and the *Endrule* is substituted to the last symbol string, the symbol string is interpreted by the turtle's interpretations of the L-systems definition. This string is transformed to the associated plant structure with the growth function in each time step.

4.4 L-system String and Plant Interpretation

The last string of generated L-system is interpreted as the plant structure which consists of internodes, petioles, leaves, apices, and flowers that compose to the main stem. The branches is defined from the bracket symbols “[” and “]”. The example of the L-system prototype is given in an example 2.

The simplest L-system is given for understanding the L-system code and L-system interpretation in *Plant*2 prototype.

```
Plant2{
    Iterations=1
    Angle=45
    Diameter=2
    Axiom=IA
    A=[+pL][-pL]I[-I1][+I1]I[/I2][\I2]IF
    1=[/iL][\iL]P
    2=[-iL][+iL]P
    Endrule
    P=iiF
}
```

The *Plant2* prototype has one iteration, the angle of 45 degrees, and two units of diameter. An axiom consists of the internode *I* and the apex *A*. At the first iteration, the apex *A* is substituted by left leaf short petiole and leaf [+pL], right short petiole and leaf [-pL], internode *I*, left petiole and special symbol [-I1], right petiole and special symbol [+I1], internode *I*, back petiole and special symbol [/I2], front petiole and special symbol [\I2], internode *I*, and flower *F*. The second production, a symbol 1 is replaced by back short internode and its leaf [/iL], front short internode and its leaf [\iL], and the petiole *P*. The third production, a symbol 2 is replaced by short right petiole and its leaf [-iL], short left petiole and its leaf [+iL], and the petiole *P*. The *endproduction P* is substituted by two short internode and flower for every petiole *P* in the last L-system string. After the first iteration, the last string is given below.

I[+pL][-pL]I[-I[/iL][\iL]iiF][+I[/iL][\iL]iiF]I[/I[-iL][+iL]iiF][\I[-iL][+iL]iiF]IF

The internode order, leaf order, and flower order of the L-system string are described in Figure 4.4. The internode order is counted from the first to the last L-system symbol string for symbol *I*, *i*, *P*, and *p*. The leaf (L) and flower (F) order are attached to the previous internode or petiole. Every internode and petiole has an attribute for leaf and flower. The attribute of internode is described in Section 4.8.

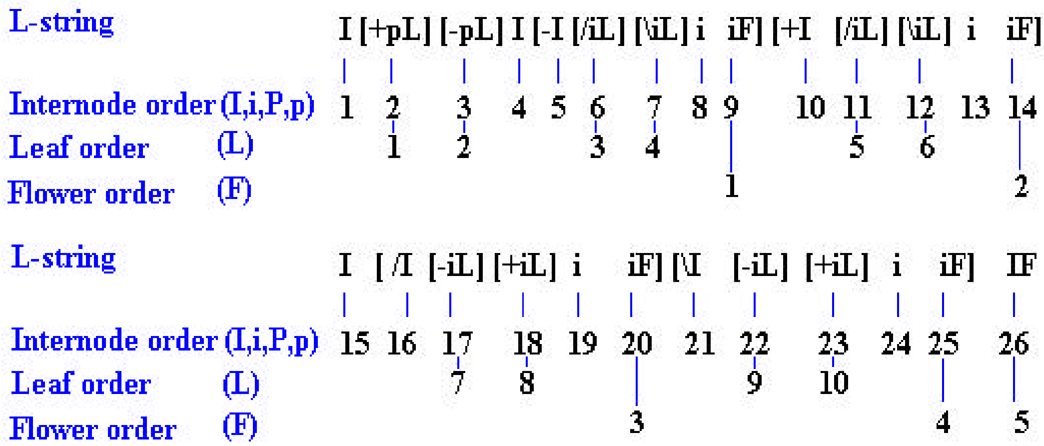


Figure 4.4: The component order of L-system string.

The visualization of the L-system string is presented in Figure 4.5. It shows only the internodes and petioles. The previous nearest component of the internode or the petiole is the parent of component *i*. Figure 4.6 shows the number of internode order.

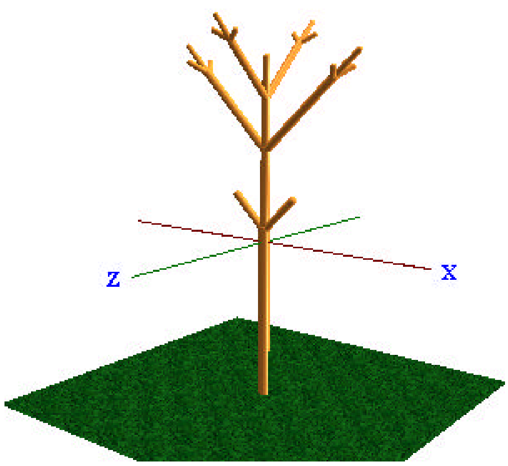


Figure 4.5: All internodes and petioles.

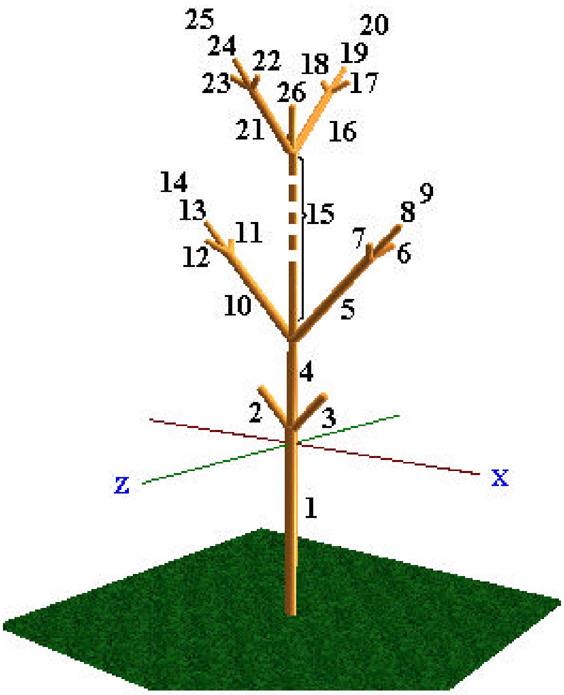
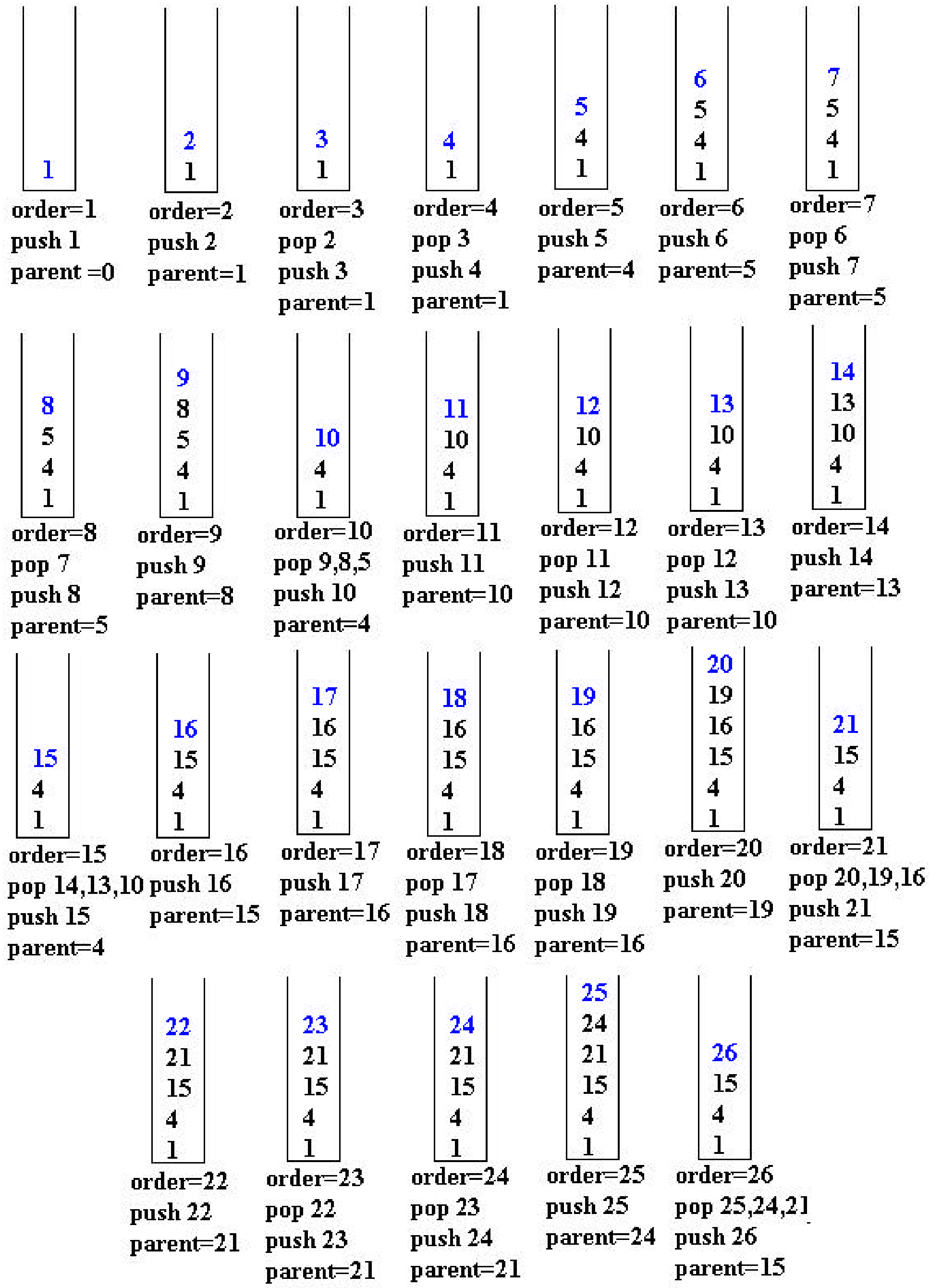


Figure 4.6: All internodes number.

In Figure 4.6, the number of each internode is counted following the L-system string as in Figure 4.4. It is used to determine the parent of each component. The number of symbol [in the last L-system string is used to generate the main stem of plant. The level of each component is used to calculate the initial time of each component. The number of bracket [is calculated from the following equation.

Number of Bracket [= (Number of symbol [ ) – (Number of symbol])

The parent is used to set the attribute of each component in the visualization process, for the example in Figure 4.6, the internode 1 is the parent of the internode 2, 3 and 4. The parent value of component can be calculated from the number of bracket [and the number of main internode in the current state. Each stack of every state with the order number of internode or petiole of the internode *I*, short internode *i*, petiole *P*, or short petiole *p* are shown in Figure 4.7.



I[+pL][-pL]I[-I[/iL][\iL]iiF][+I[/iL][\iL]iiF]I[/I[-iL][+iL]iiF][\I[-iL][+iL]iiF]IF

Figure 4.7: The parent of each internode and petiole.

At the last order of Figure 4.7, the main stem of the plant will be arranged in the stack. For example, the main stem of this example is the first, the fourth, the 15th, and the 20th internode, respectively. The level of each component is considered from the order of the parent calculated from the following equation:

Level of component i (L_i) = (Level of parent component i) + 1

where the level of the parent component 0 is zero. Figure 4.8 shows the level of the plant. It is used to set the initial time of its component. It will be shown in Section 4.5. The level of component, the main stem order, the parent of component, and the number of bracket [are shown in Table 4.2.

where
L_i : Level of component
M_i : Main stem order
P_i : Parent of component
$N([)$: Number of bracket [

Table 4.2: The value of leaf number, flower number, number of [, parent of each component, main stem order, and level of each component.

Component number *i*	Symbol	Leaf number	Flower Number	$N([)$	P_i	M_i	L_i
1	I	-	-	0	0	1	1
2	p	1	-	1	1	-	2
3	p	2	-	1	1	-	2
4	I	-	-	0	1	2	2
5	I	-	-	1	4	-	3
6	i	3	-	2	5	-	4
7	i	4	-	2	5	-	4
8	i	-	-	1	5	-	4
9	i	-	1	1	8	-	5
10	I	-	-	1	4	-	3
11	i	5	-	2	10	-	4
12	i	6	-	2	10	-	4
13	i	-	-	1	10	-	4
14	i	-	2	1	13	-	5
15	I	-	-	0	4	3	3
16	I	-	-	1	15	-	4
17	i	7	-	2	16	-	5
18	i	8	-	2	16	-	5
19	i	-	-	1	16	-	5
20	i	-	3	1	19	-	6
21	I	-	-	1	15	-	4
22	i	9	-	2	21	-	5
23	i	10	-	2	21	-	5
24	i	-	-	1	21	-	5
25	i	-	4	1	24	-	6
26	I	-	5	0	15	4	4

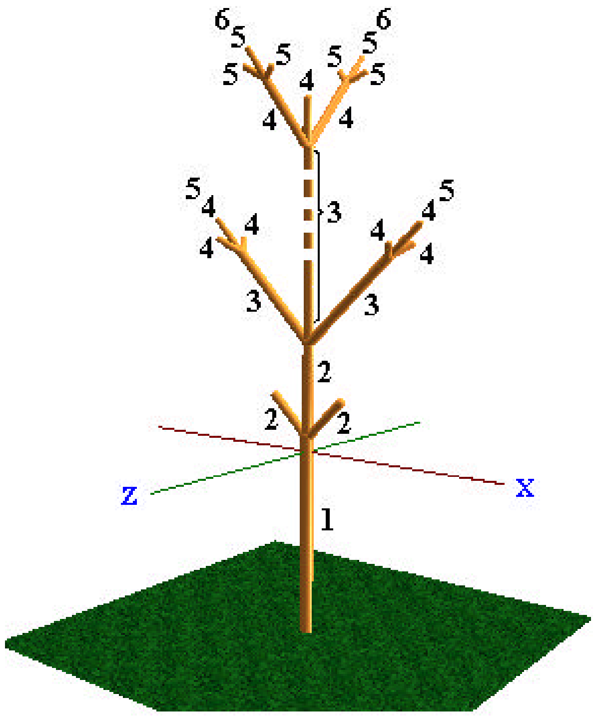


Figure 4.8: The level of each component.

From this prototype, the leaf *L* and the flower *F* are added to the system and the visualization is shown in Figure 4.9.

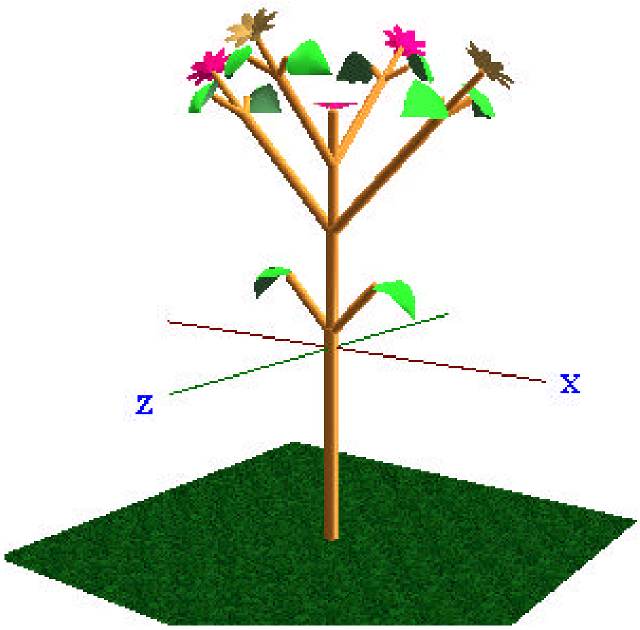


Figure 4.9: The visualized image after adding leaf and flower to the system.

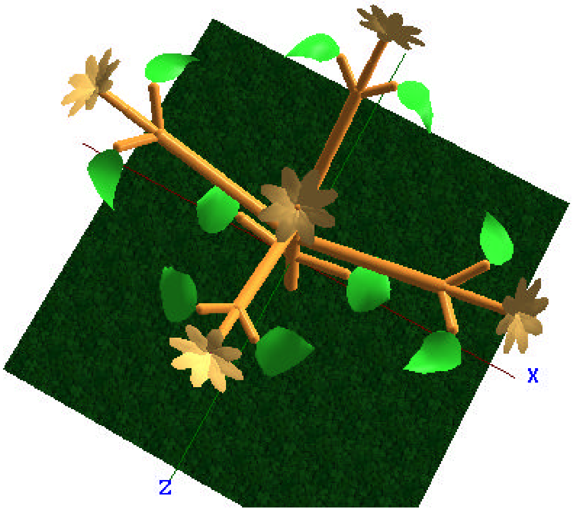


Figure 4.10: The top view of *Plant2* prototype.

In the case of soybean, the L-system string is more complicated, so we reduced the process to two iterations. The last L-system string is given below.

I[-iL][+iL]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL]
[/pL][-pL]]I[+IIII[\pL][/pL][+pL]]IL

The internodes and petioles of the soybean model are shown in Figure 4.11, and the number of internode and petiole are shown in Figure 4.11. The level of each component and its parent is shown in Figure 4.13 by following the concept of the *Plant2* prototype. After adding the leaf *L* to the plant, the visualization is shown in Figure 4.14, and the top view of its topology is expressed in Figure 4.15.

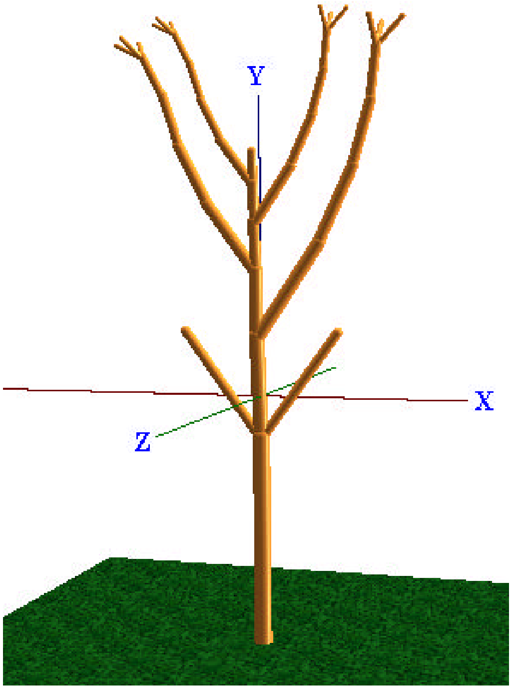


Figure 4.11: The internode and petiole of soybean at second iteration.

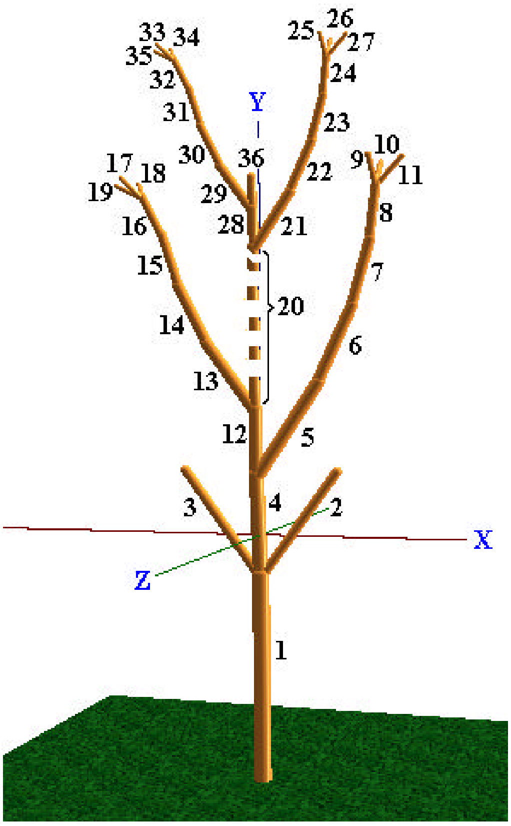


Figure 4.12: The internode and petiole order number.

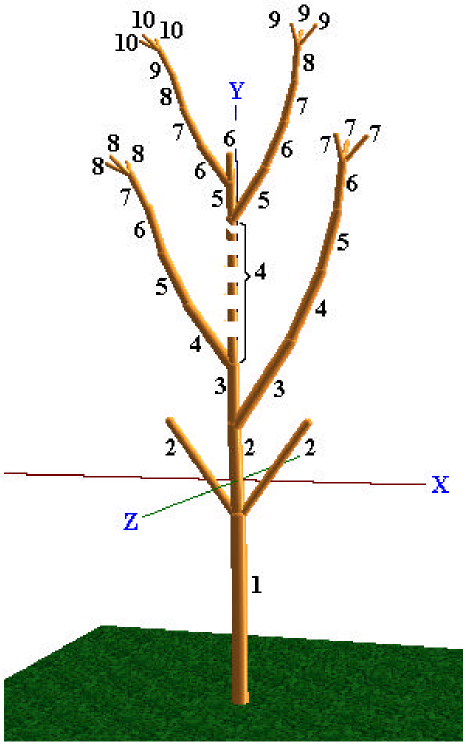


Figure 4.13: The level of all internode and petiole.

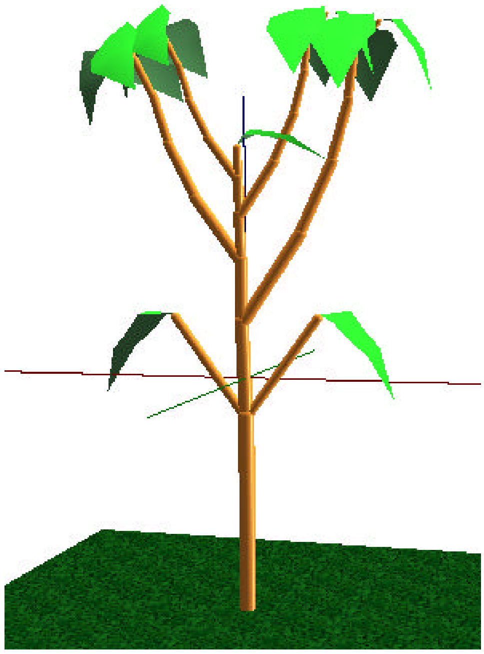


Figure 4.14: The soybean and its leaves.

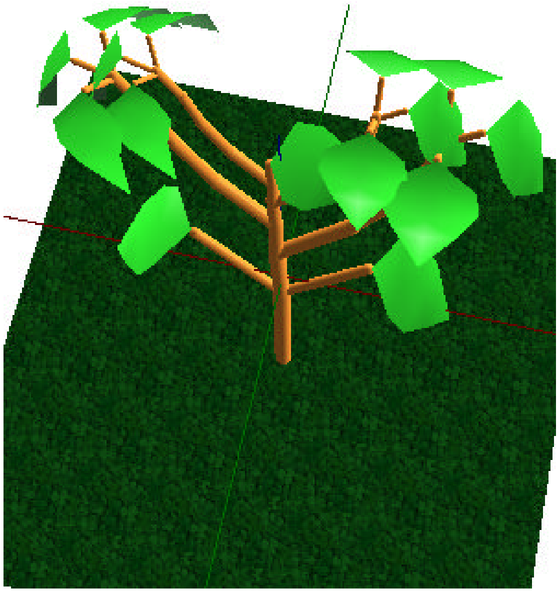


Figure 4.15: The top view of the soybean.

The structure of rewriting algorithm is similar to Pascal. The structure of the production rule consists of the predecessor and successor which is given below.

```
Type production = record
begin
        Pred    : Character
        Succ    : String
end
```

The variables are defined as follows:

i, j, and k	: a positive integer,
iter	: a number of the L-system iterations,
rulenum	: a number of production rules,
allstr	: a last L-system string,
rule	: an array of the production rule,
prev	: a previous string of current character,
last	: a next string of current character,
Endrule_Check	: a flag of the endrule keyword, it is true if there are endrule keyword,
Length()	: a length function return the number of argument string,
EndRuleNum	: a number of endproduction rules,
Endrule	: an array of the production rule,

The rewriting algorithm of the L-system is given below.

Rewriting Algorithm

```
BEGIN
  FOR i := 1 TO iter DO
  BEGIN
        FOR j := 1 TO rulenum DO
        BEGIN
              k := 1
              WHILE (k <= length(allstr)) DO
              BEGIN
                    IF  allstr[k]=rule[j].pred THEN
                    BEGIN
                          prev := Copy(allstr,1,k-1)
                          last := Copy(allstr,k+1,length(allstr)-k)
                          allstr := prev + rule[j].succ + last
                          k := k+length(rule[j].succ)
                    END
                    ELSE  k:=k+1
              END
        END
  END

  IF EndRule_Check THEN
  BEGIN
        FOR j := 1 TO EndRuleNum DO
        BEGIN
              k := 1
              WHILE (k <= length(allstr)) DO
              BEGIN
                    IF  allstr[k]=EndRule[j].pred THEN
                    BEGIN
                          prev := Copy(allstr,1,k-1)
                          last := Copy(allstr,k+1,length(allstr)-k)
                          allstr := prev + EndRule[j].succ + last
                          k := k+length(EndRule[j].succ)
                    END
                    ELSE  k:=k+1
              END
        END
  END
END.
```

After the the rewriting process is finished, the L-system string is interpreted by the L-system interpretation algorithm, the structure and variables are defined as follows:

The structure of plant is given the following type:

```
Type TTree=record
        StringType     : Char;
        Length         : Real;
        BigT           : Real;
        T              : Real;
        t_start,t_stop : Real;
        Angle_H, Angle_U, Angle_L : Glfloat;
        Angle          : GLfloat;
        angle_azimut   : GLfloat;
        allchild       : Integer;
        child          : array[1..maxchild] of integer;
        myparent       : Integer;
        mylevel        : Integer;
        Leaf           : Boolean;
        Flower         : Boolean
End.
```

The description of TTree attribution is given as follows:

StringType : a symbol of the L-system string,
Length : a length of internode, or petiole,
BigT : a maximum life time of internode or petiole,
t : a time variable between the internode or petiole life cycle,
t_start,t_stop : a start time and stop time of the internode or petiole,
Angle_H, Angle_L, Angle_U : a angle respect to X, Y, Z axis, respectively,
allchild : a number of all child of internode or petiole,
child : an array of child internode or petiole,
myparent : a parent of the internode or the petiole,
mylevel : a level of the internode or the petiole,
Leaf : a flag for internode leaf, it is TRUE if there is a leaf,
Flower : a flag for internode flower, it is TRUE if there is a flower,

The variables are defined as follows:

CountNode : a number of symbols *I, A, P, i,* and *p,*
Showmessage() : a procedure that display the argument message,
Exit : an exiting procedure,
NOT : a boolean operator,
Maxatree : a maximum length of the L-system string,
Atree : an array of plant TTree type,
Maxatree : a maximum number of symbols *I, A, P, i,* and *p,*
Current_Str[k] : a k^{th} symbol of the L-system string,
Main_Stem[k] : an array of the main stem order at k^{th} level,
MainStemOfBracket[k] : a number of bracket [at the k^{th} level,
Bracket : a number of Bracket [,
Ang_U : an angle respect to Z axis,
Ang_L : an angle respect to Y axis,

Ang_H : an angle respect to X axis
TopStack : a length of stack,
Stack_Angle_U[TopStack] : a Topstack angle respect to Z axis,
Stack_Angle_L[TopStack] : a Topstack angle respect to Y axis,
Stack_Angle_H[TopStack] : a Topstack angle respect to X axis,
RU_Change : a flag of turtle angle on Z axis,
RL_Change : a flag of turtle angle on Y axis,
RH_Change : a flag of turtle angle on X axis,
AllNode : a number of all component *I, i, A, P, p,*
Current_Node : a current symbol of the L-system string,
TopMainStem : a current length of the main stem, and
MainStem[TopMainStem] : a main stem order at TopMainStem level.

The L-system interpretation algorithm is given below.

L-system interpretation algorithm

```
BEGIN
        CountNode := 0;
        FOR k := 1 TO length(allstr) DO
        BEGIN
           IF allstr[k] IN ['I','A','P','i','p'] THEN
                 CountNode := CountNode + 1
           IF NOT (allstr[k] in ['I','i','A','P','p','L','F','[',']','-','+','/','\','^','&','|']) THEN
           BEGIN
                 Showmessage(allstr[k]+'is not in {I,i,A,P,p,L,F,[,],/,\,+,-,^,&,|}')
                 EXIT
           END
        END

        IF CountNode > Maxatree THEN
        BEGIN
           Showmessage('The String is too long.')
           EXIT
        END

        FOR k:= 1 TO CountNode DO
        BEGIN
                 atree[k].Angle_U := 0
                 atree[k].Angle_L := 0
                 atree[k].Angle_H := 0
                 atree[k].Leaf := False
                 atree[k].Flower := False
        END

        IF CountNode > MaxaTree THEN
        BEGIN
                 Showmessage('The number of node is too much.')
                 EXIT
        END
```

```
        ELSE
                FOR k:= 1 TO length(allstr) DO
                BEGIN
                        Current_Str[k] := 0
                        Main_Stem[k] := 0
                        MainStemOfBracket[k] := 0
                END

//------------ Plant generator start ------------
        k := 1
        WHILE k<=length(allstr) DO
        BEGIN
                CASE allstr[k] OF
                '[' :   BEGIN
                                Bracket := Bracket + 1
                                MainStemOfBracket[Bracket] := 0

                                Ang_U := 0
                                Ang_L := 0
                                Ang_H := 0

                                Stack_Angle_U[TopStack] := Ang_U
                                Stack_Angle_L[TopStack] := Ang_L
                                Stack_Angle_H[TopStack] := Ang_H

                        END

                ']' :   BEGIN
                            Bracket := Bracket - 1
                            Current_Str[TopStack] := 0  // Clear the top of Stack

                            Ang_U := Stack_Angle_U[TopStack]
                            Ang_L := Stack_Angle_L[TopStack]
                            Ang_H := Stack_Angle_H[TopStack]

                            TopStack := TopStack - MainStemOfBracket[Bracket+1]
                            MainStemOfBracket[Bracket+1] := 0

                            RU_Change := True
                            RL_Change := True
                            RH_Change := True
                        END

        'A','P','I','i','p' :
                        BEGIN
                           AllNode := AllNode + 1
                           TopStack := TopStack + 1
                           Current_Node := CountNode + 1
                           Current_Str[TopStack] := Current_Node
```

```
   MainStemOfBracket[Bracket] := MainStemOfBracket
           [Bracket] + 1

IF (RU_Change) THEN
BEGIN
        IF (allstr[k+1] IN ['I','A','P','i','p']) THEN
        BEGIN
                RU_Change :=True
                atree[Current_Node].Angle_U := Ang_U
        END
        ELSE    RU_Change :=False
END

IF (RL_Change) THEN
BEGIN
        IF (allstr[k+1] IN ['I','A','P','i','p']) THEN
        BEGIN
                RL_Change :=True
                atree[Current_Node].Angle_L  := Ang_L
        END
        ELSE    RL_Change :=False
END

IF (RH_Change) THEN
BEGIN
        IF (allstr[k+1] IN ['I','A','P','i','p']) THEN
        BEGIN
                RH_Change :=True
                atree[Current_Node].Angle_H  := Ang_H
        END
        ELSE    RH_Change :=False
END

IF Bracket = 0 THEN   // Check for MainStem
BEGIN
        inc(TopMainStem)  // = 0 is empty node.
        Main_Stem[TopMainStem] := Current_Node
        IF (allstr[k-1] IN ['[','I','A','P',']','i','p']) THEN
                atree[Current_Node].angle  := 0

        Ang_U  := 0.01*Random(5)
        Ang_L  := 0.01*Random(5)
        Ang_H  := 0.01*Random(5)

        IF allStr[k] = 'I' THEN
                atree[Current_Node].StringType  := 'I'
        ELSE IF allStr[k] = 'i' THEN
                atree[Current_Node].StringType  := 'i'
        ELSE    atree[Current_Node].StringType  := allstr[k]
END
```

```
            ELSE  IF allStr[k] IN ['I','i'] THEN
            BEGIN
            IF allStr[k] = 'I' THEN
                  atree[Current_Node].StringType  := 'P'
            ELSE    atree[Current_Node].StringType  := 'p';

            // Reset the angle of Petiole after first Node

      IF allStr[k-1] IN ['I','i'] THEN
      BEGIN
            IF RU_Change AND (atree[Current_Node1].Angle_U>0)
            THEN
                  Ang_U  := 0.05
            ELSE  Ang_U  := -0.05

            IF RL_Change AND (atree[Current_Node1].Angle_L>0)
            THEN
                  Ang_L  := 0.05
            ELSE  Ang_L  := -0.05

            IF RH_Change AND (atree[Current_Node1].Angle_H>0)
            THEN
                  Ang_H  := 0.05
            ELSE  Ang_H  := -0.05
            END
      END
      ELSE
            atree[Current_Node].StringType  := allstr[k]

            // Set Leaf attach to this node
            IF allstr[k+1]='L' THEN
            BEGIN
                  atree[Current_Node].Leaf  := True
                  k :=k+1
            END
            // Set Flower attach to this node
            ELSE IF allstr[k+1]='F' THEN
            BEGIN
                  atree[Current_Node].Flower := True
                  k:=k+1
            END

            atree[Current_Node].myparent := Current_Str[TopStack-1]

            atree[Current_Node].Angle_U := Ang_U
            atree[Current_Node].Angle_L := Ang_L
            atree[Current_Node].Angle_H := Ang_H

            atree[Current_Node].mylevel := Bracket
      END
```

```
        '-' :   BEGIN
                    Ang_U := Ang_U - sigma
                    RU_Change := True
                END

        '+' :   BEGIN
                    Ang_U := Ang_U + sigma
                    RU_Change := True
                END

        '|' :   EGIN
                    Ang_U := Ang_U + 180.0
                    RU_Change := True
                END

        '&' :   BEGIN
                    Ang_L := Ang_L + sigma
                    RL_Change := True
                END

        '^' :   BEGIN
                    Ang_L := Ang_L - sigma
                    RL_Change := True
                END

        '\' :   BEGIN
                    Ang_H := Ang_H + sigma
                    RH_Change := True
                END

        '/' :   BEGIN
                    Ang_H := Ang_H - sigma
                    RH_Change := True
                END
        END
        k := k + 1
  END
END.
```

4.5 Data Collection

The data of each component are collected from an actual soybean. They are the internodes length, diameter, leaves length and width, petioles length corresponding to the time of its life cycle. The actual data are obtained daily from three soybeans for 61 days. The raw data will be used for approximating to the sigmoidal growth function. The data of soybean were collected manually using rulers and a protractor. The collected data is shown in Appendix B. We ignore all data that are not vegetative state such as underground part, and the reproductive states. The physiology of soybean is shown in Figure 4.16. The internode length, petiole length, leaf length and width are designed as in Figure 4.17, Figure 4.18, and Figure 4.19, respectively.

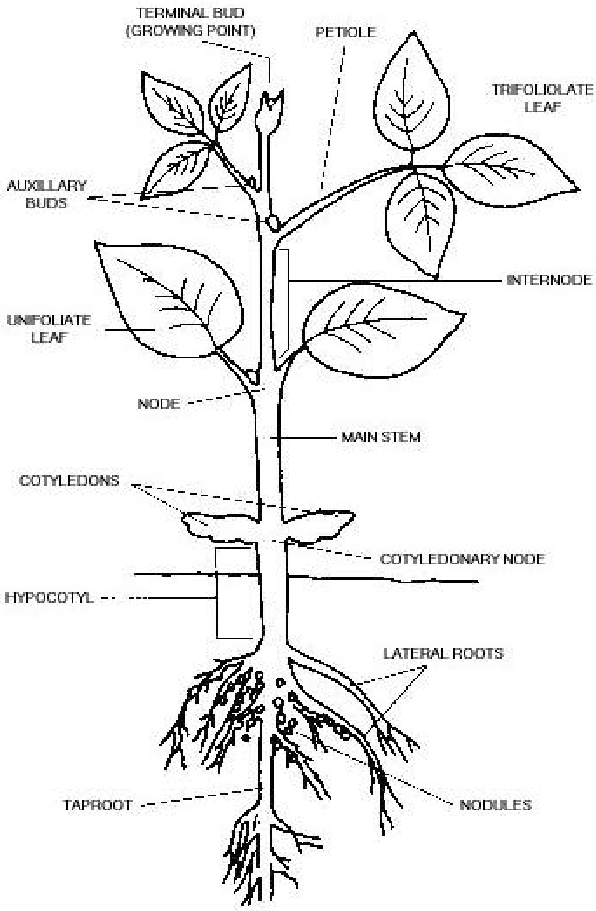


Figure 4.16: The soybean physiology.

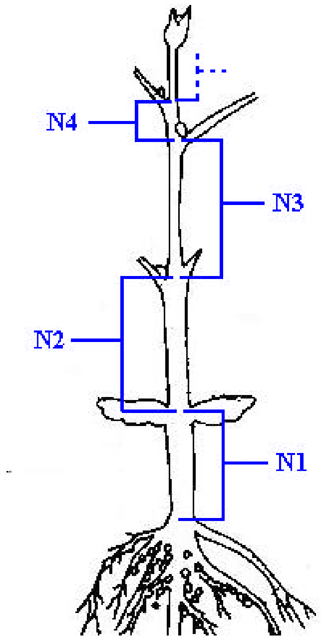


Figure 4.17: Internode data, Ni is the order of internode.

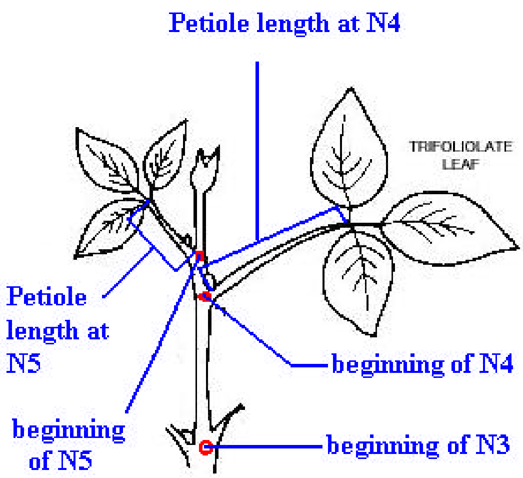


Figure 4.18: Petiole length data.

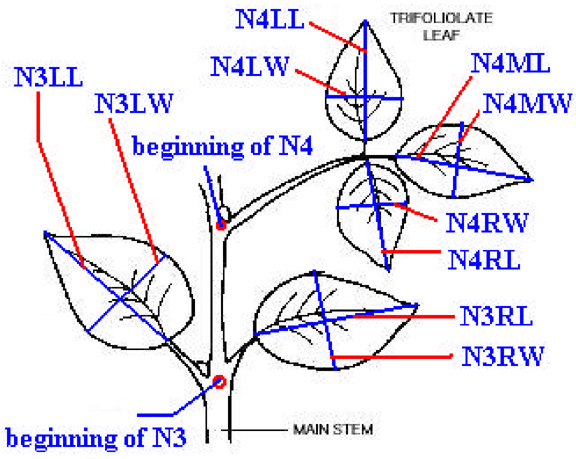


Figure 4.19: Leaf length and width data.

The description of leaf measurement is shown in Figure 4.19. At the beginning of third internode, there are two leaves. The fourth internode and upper internode, there are trifoliolate – three leaves. The symbol description of each leaf is given below.

NiLL : the left leaf length of i^{th} intenode,
NiLW : the left leaf width of i^{th} intenode,
NiRL : the right leaf lenght of i^{th} intenode,
NiRW : the right leaf width of i^{th} intenode,
NiML : the middle leaf length of i^{th} intenode,
NiMW : the middle leaf width of i^{th} intenode,

4.6 Growth Function

The raw data in Section 4.5 is approximated as a sigmoidal growth function in shown Figure 4.20.

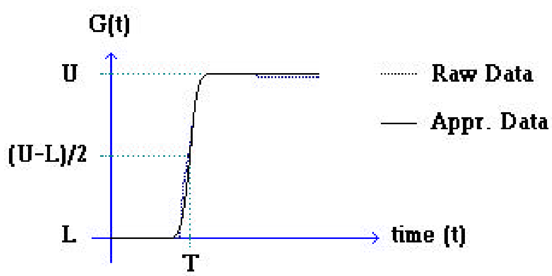


Figure 4.20: Sigmoidal curve approximation.

The raw data is converted to the growth function *G(t)* of length or width at time *t* and is given below.

$$G(t) = L + \frac{U - L}{1 + e^{m(T-t)}}$$

where	L	: the minimum value of length or width,
	U	: the maximum value of length or width,
	m	: the approximated slope of raw data,
	T	: the time at (U – L)/2
	t	: the independent time variable

In the prototype, the system read data from the user interface and compute the approximation function as a sigmoidal curve the algorithm is as the following.

The Sigmoidal curve approximation Algorithm

1. Read the raw data of length, width, diameter of the internode, the petiole, the leaf, the apex, and the flower corresponding to a time *t*.
2. Compute the minimum value *L* and the maximum *U* from step 1.
3. Compute the time *T* the size (U-L)/2.
4. Compute the appropriate slope of the sigmoidal curve.
   4.1 Compute all error of each slope. The slope *m* will be increased from zero to one with step size 0.01.

```
for i := 0 to 100 do
  begin
     e_i := 0.00;
     m := 0.01*i;
     for j := 1 to n do
     begin
        G(t_j) := L + (U-L)/(1+exp(m*(T-t_j)));
        e_i := e_i + | G(t_j) - y_j |;
     end;
end;
```

where n is a number of day, e_i is an error at value i, $G(t_j)$ is and approximated of growth value at time j, y_j is a raw data at time j, and L,U,m are as the previous

4.2 Compute the appropriated slope at the minimum error.

```
slope_tmp := e_0;
slope_ok := 0;
for i:= 1 to 100 do
begin
   if slope_tmp > e_i then
   begin
      slope_tmp := e_i;
      slope_ok := i;
   end;
end;
```

Figure 4.21: An algorithm for calculating the simoidal curve.

Besides the growth function, there are other functions, which are used to control all the components of the plant topology, such as the length of each internode from the first internode to the last internode. The function is

$$Y_i = c(a)^{ni} \qquad \ldots(3)$$

where Y_i is the length of internode i, c is a constant, a is a real value greater than zero, and n_i is the level of internode i.

The initial time of each component is specified by the following linear equation

$$B_i = \mathbf{b} n_i + b \qquad \ldots(4)$$

where B_i is the initial time of component i, $\mathbf{b}$ is the acceleration rate of B_i, n_i is the level of component i, and b is a constant.

Every component of plant is controlled by a self-growth function likes the one shown in Figure 4.16 with either same or different slope m, time T, the maximum U, and the minimum L.

4.7 Quantitative Model

The quantitative model combines the L-systems string with the approximated growth functions of each component. The plant model can simulate its growth with continuous development in the virtual reality form. In Section 4.3, the qualitative model or the L-system string is defined and generated to the plant structure. In Section 4.6, the approximated growth function is constructed from the raw data corresponding to the time in their life cycle. The slope m, the minimum value L, the maximum value U, and the time T are set to its component.

Every component of the plant such as the internode I, the petiole P, the leaf L, the flower F, and the apex A are controlled by a self-growth function likes the one shown in Figure 4.16 with either same or different slope m, the time T, the maximum U, and the minimum L.

For example, the L-system string of *Soybean* prototype in Section 4.3 is shown below:

I[-iL][+iL]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]I[-IIII[\pL][/pL][-pL]]I[+IIII[\pL][/pL][+pL]]IL

The value of m, L, U, T of every component is shown in the Table 4.3

Table 4.3: The value of *L, U, m, T*..

Symbols	*L* value	*U* value	*m* value	*T* value
I and i	0.1275	4.039	0.65	7
P and p	0	6.8737	0.38	8
L width	0	2.7909	0.49	11
L length	0	5.2613	0.54	11
A	0.1275	4.039	0.65	7

The approximated growth function of each component is shown in Figure 4.22. The internode *I* and the short internode *i* are shown in Figure 4.22(a), the petiole *P* and the short petiole *p* are shown in Figure 4.22(b), Figure 4.22(c) shows the leaf length *L* , the leaf width *L* is shown in Figure 4.22(d), and Figure 4.22(e) shows the apex *A*.

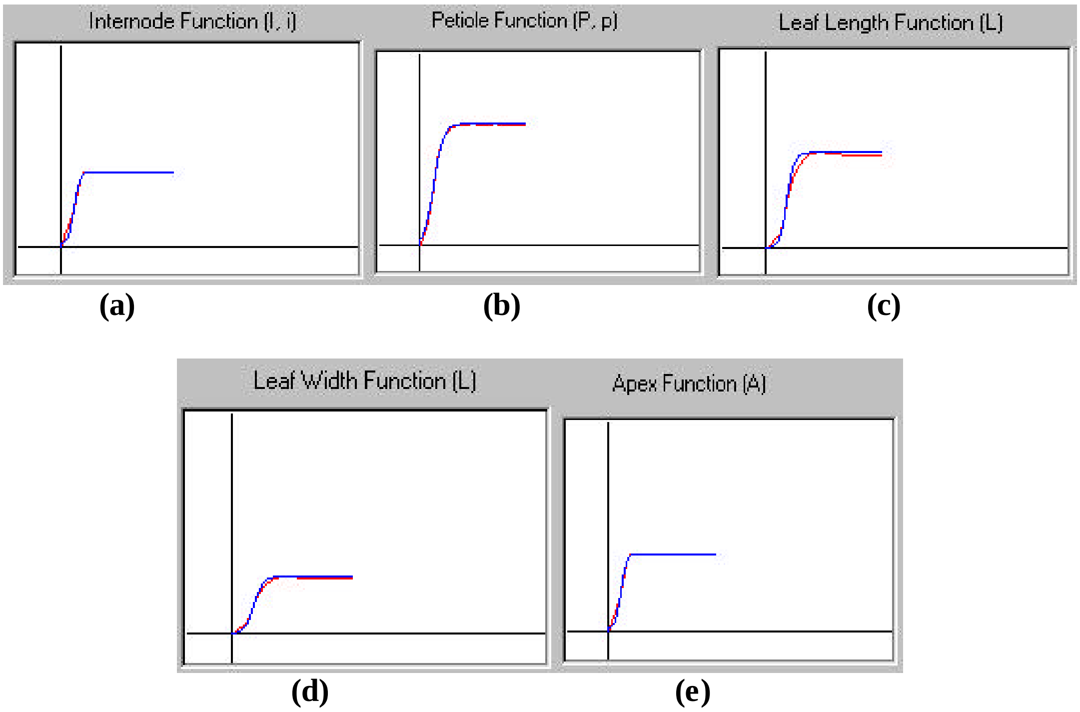


Figure 4.22: The approximated growth function of *I, i, P, p, L,* and *A*.

Figure 4.23 illustrates a graphical image of a soybean drawn from the L-system string defined in the production rules in Section 4.3. Figure 4.23(a) shows the L-system string obtained after the rewriting process. The plant structure in Figure 4.23(b) is constructed from the L-system string shown in Figure 4.23(a) and its graphical image of the axiom of the plant is illustrated in Figure 4.23(g).

Figure 4.23(d) gives the details of a part of soybean. Figure 4.23(c) is the left petiole component. The right petiole is shown in Figure 4.23(e). In Figure 4.23(f), the raw data is converted to the growth function corresponding to each symbol.

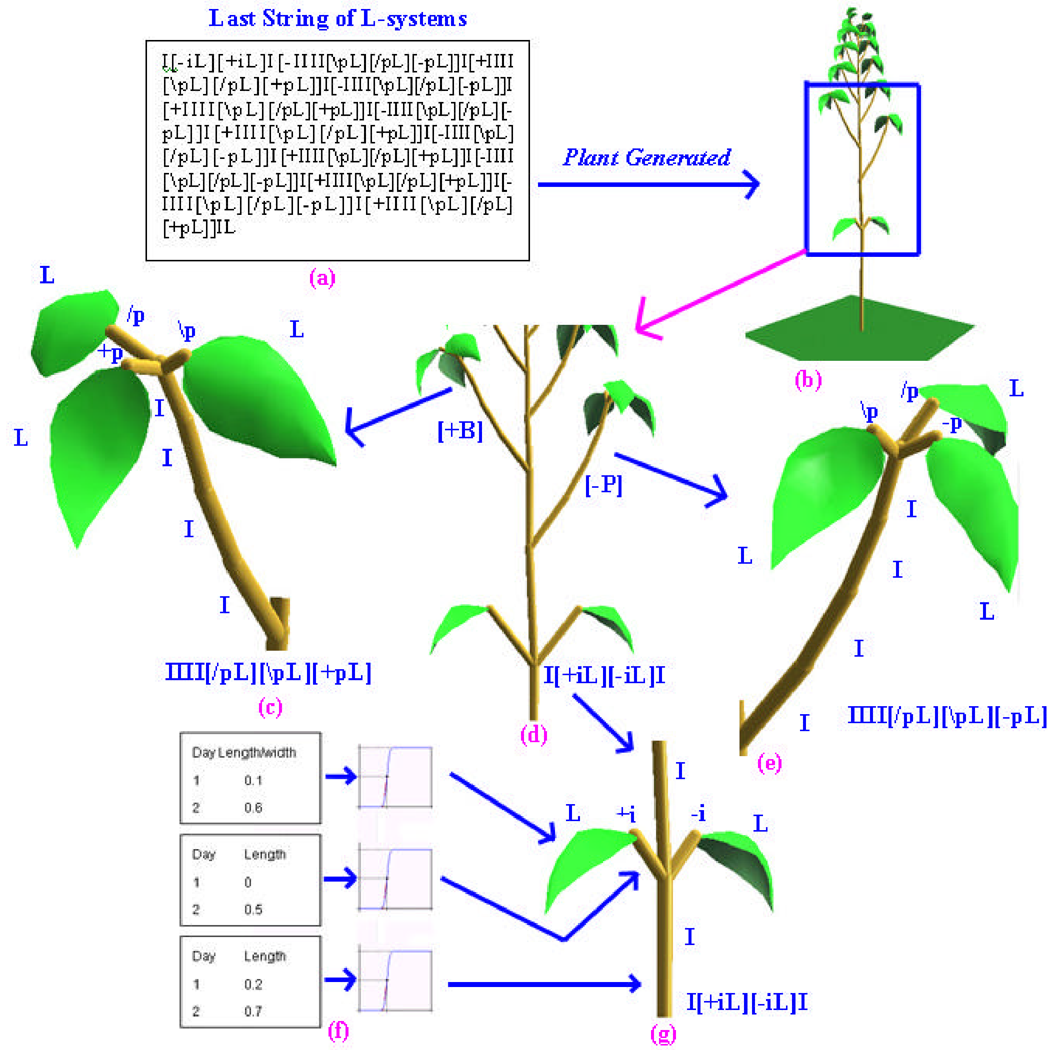


Figure 4.23: Structure of simulation generated from the production rules of soybean in Section 4.3.

4.8 Visualization

This section describes the visualization method of each component such as the internode *I*, the short internode *i*, the petiole *P*, the short petiole *p*, the leaf *L*, the apex *A*, and the flower *F*. Each component uses the primitive geometric shape; for example, cylinder, line, triangular polygon, sphere, rectangular polygon, and texture bitmap.

4.8.1 Internode

A cylinder represents the plant internode, while a sphere represents the internode joint. Figure 4.24 shows the internode of plant. The short internode is similar to internode but it is not the same length. The internode and their ground is shown in Figure 4.25.

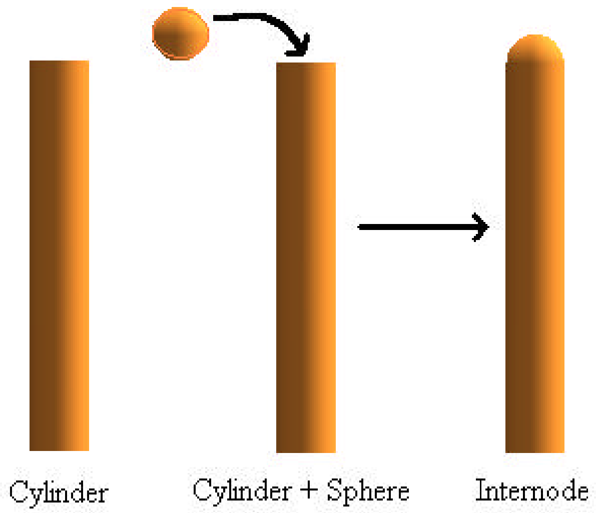


Figure 4.24: Internode of plant topology.

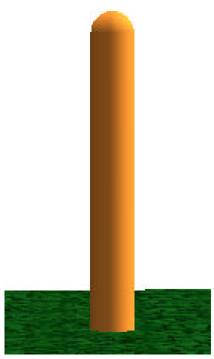


Figure 4.25: Internode plant.

The structure of internode in the prototype is defined by the following type:

```
Procedure InterNode(wx,wy:real)
begin
        Cylinder(wx,wx,wy,slices,stacks)
        Sphere(wx,longitudes,latitudes)
end
```

From above procedure, internode receive 2 arguments, *wx* and *wy* where *wx* represents the base radius of the internode, and *wy* represents the height of the internode. A cylinder has five arguments, namely, base radius, top radius, height, edges, and slices. Spheres have three arguments, namely, radius, latitudes, and longitudes. The cylinder and the sphere are given in Figure 4.26.

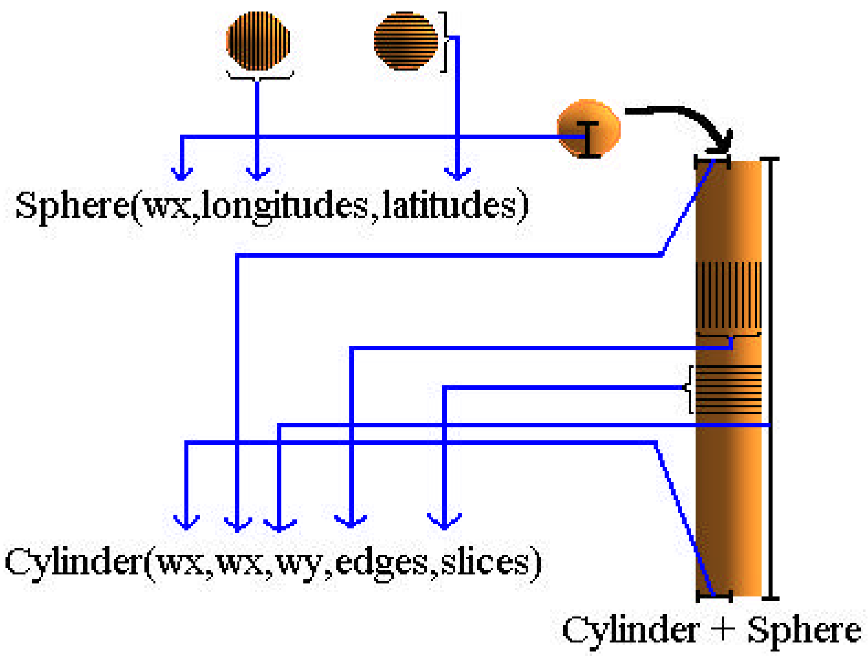


Figure 4.26: The cylinder and sphere argument.

4.8.2 Petiole

The petiole and the main stem have the same topology as internode with either same or different size. The petiole component is represented by the symbol *I*, *i*, *P*, or *p*. Figure 4.27 shows the petiole and the main stem of the plant topology. The short petiole structure is similar to the petiole with either same or different size. The main stem was given in Section 4.4. Its direction is upward to Y-axis.

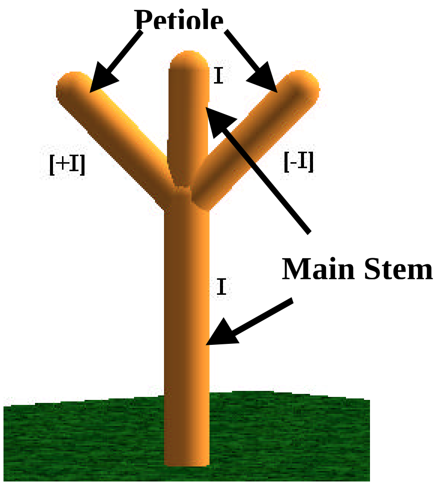


Figure 4.27: The petiole and the main stem of plant topology I[-I][+I]I.

4.8.3 Leaf

A plant leaf is very important for plant visualization to look more realistic. The plant is defined by more than three points in the three-dimensional space. It consists of the *source point* which is attached to the internode or the petiole tip. In this thesis, the leaf library is designed using the following format.

Name=Leaf_Name
Source=S
Point=P_{1x} P_{1y} P_{1z}
Point=P_{2x} P_{2y} P_{2z}
Point=P_{3x} P_{3y} P_{3z}
...
Point=P_{nx} P_{ny} P_{nz}
Triangle=T_{11} T_{12} T_{13}
Triangle=T_{21} T_{22} T_{23}
Triangle=T_{31} T_{32} T_{33}
...
Triangle=T_{m1} T_{m2} T_{m3}

The meaning of each keyword is given as follows:

Name=Leaf_Name

Leaf_Name is the name of leaf.

Source=S

The *source point* is the point in set of the leaf point. It is attached to the tip petiole or the internode that has leaf.

Point=P_{nx} P_{ny} P_{nz}

The keyword is a coordinate (x,y,z) of leaf point. The value P_{nx} , P_{ny} , and P_{nz} are the x, y, z at the point n^{th}. They are a real value.

Triangle=T_{m1} T_{m2} T_{m3}

The triangle consists of three points. The value T_{m1} ,T_{m2} ,T_{m3} are the first point, the second point, and the third point of the m^{th} triangle. They are member in the set of the leaf point.

For example, the leaf prototype is given below.

```
Name=Leaf,Soybean
Source=2
Point=0 0 0
Point=0 0 0
Point=0 0 0
Point=-33 -50 8
Point=0 -50 11
Point=33 -50 8
Point=-44 -100 11
Point=0 -100 17
Point=47 -100 11
Point=-33 -150 5
Point=0 -150 12
Point=33 -150 8
Point=0 -200 0
Point=0 -200 0
Point=0 -200 0
Triangle=1 4 2
Triangle=4 5 2
Triangle=2 5 6
Triangle=2 6 3
Triangle=4 7 5
Triangle=7 8 5
Triangle=5 8 9
Triangle=5 9 6
Triangle=7 10 8
Triangle=10 11 8
Triangle=8 11 12
Triangle=8 12 9
```

Triangle=10 13 11
Triangle=13 14 11
Triangle=11 14 15
Triangle=11 15 12

From the above illustrates a soybean leaf prototype. The second point is the *source point*. There are 15 points (size 5x3), and 16 triangular polygons. Figure 4.28 shows the soybean leaf topology. At the first, there are 15 points likes Figure 4.28(a) in plane XY coordinate. The first point and the third point are set similar to the second point as same as the 13^{th} point and the 15^{th} point are set to the 14^{th} point. The seventh point is moved leftward while the ninth is moved rightward. The fifth point, the eighth point, and the 11^{th} point are moved upward. The result is shown in Figure 4.28(b). All triangular polygon is set to every triangle in Figure 4.28(c). In Figure 4.28(d), the 13^{th} point, 14^{th} point, and 15^{th} point are moved to downward. It makes the leaf look like longer than the old leaf.

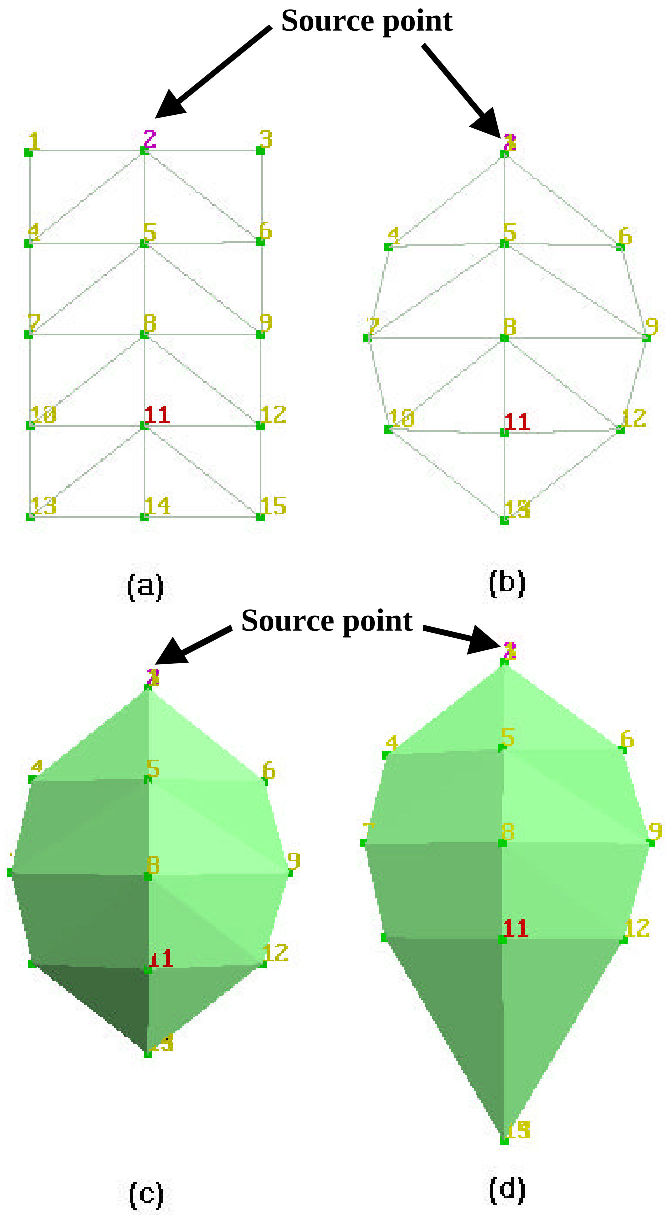


Figure 4.28: The leaf of plant topology.

The leaf grid in this thesis is designed for *J*x*K* points, where *J*, *K* is 3, 5, 7, 9, or 11. The leaf size supports odd number, because this thesis assumes that the leaf is symmetric with respect to the column. A size of leaf depends on its complicated structure.

In Figure 4.29 shows the simple plant with its leaves. The source point of leaf is attached to the internode tip. Therefore the second point is attached to the internode. The leaves in this thesis can be resized and rotated following the user adjustment.

The simple L-system of Figure 4.29 is I[-iL][+iL]A. The previous symbol of the leaf symbol *L* must be only the internode *I*, the short internode *I*, the petiole *P*, or the short petiole *p*. Therefore, the leaf cannot separate from the internode *I*, the short internode *i*, the petiole *P*, or the short petiole *p*.

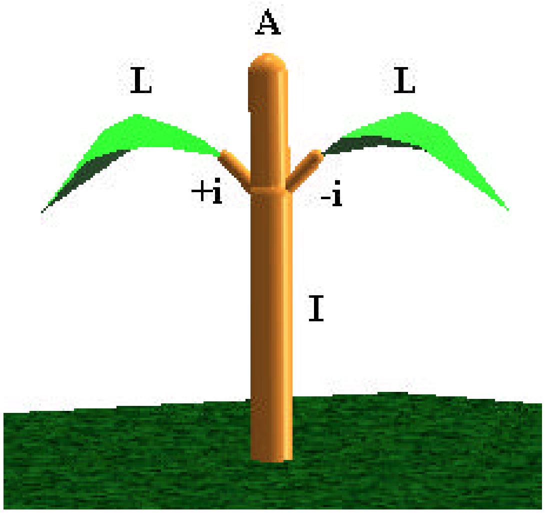


Figure 4.29: A simple plant with its leaves I[-iL][+iL]A.

4.8.4 Flower

A flower of plant in this thesis is defined only rounded flower. The number of petals will be changed by user interface. The petal structure is similar to the leaf structure. It has a source point, set of point, set of triangular polygon. For example, Figure 4.30 shows a simple plant with its leaves and flower with the L-system as follows:

I [-i L] [+i L] I i F

The plant consists of eight components, there are two internodes and one short internode for main stem. There are two petioles, left petiole, and right petiole with their leaf. The flower *F* must follow the internode *I*, the short internode *i*, the petiole *P*, or the short petiole *p*. The symbol *F* is imported from the flower library as well as the leaf library.

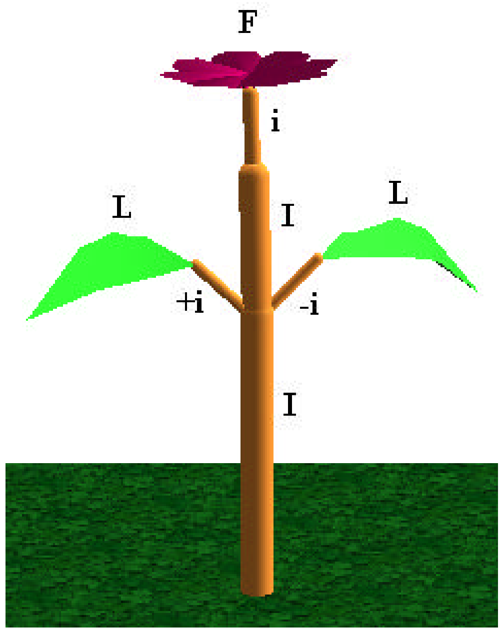


Figure 4.30: A simple plant with its leaves and flower : I[-iL][+iL]IiF.

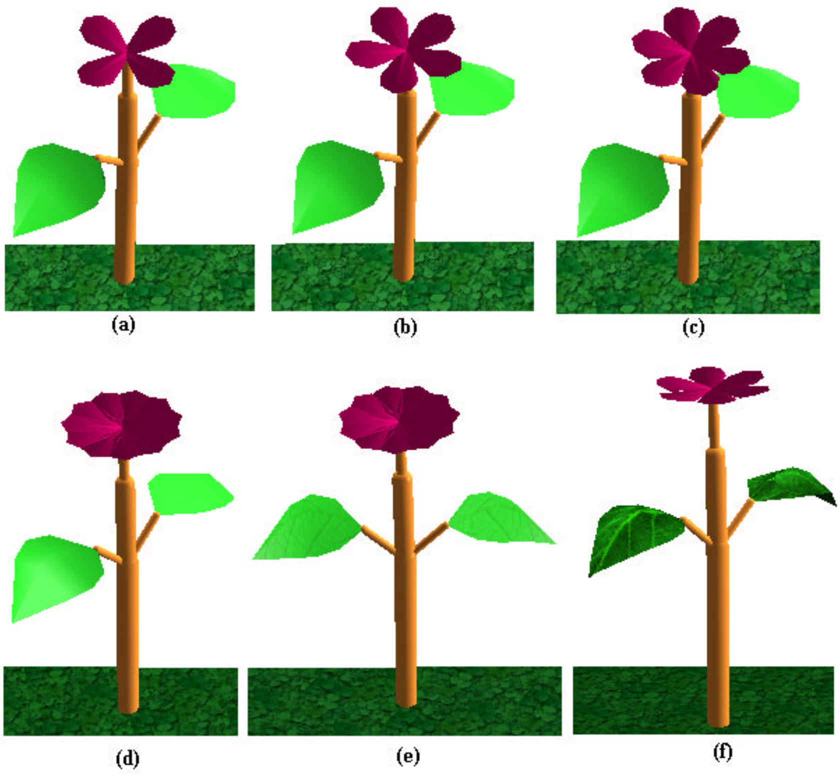


Figure 4.31: Some flower plant topology.

The flower is composed by some petals which user can set the number of petals. It is defaulted at eight petals. In Figure 4.31 shows some flower plant with their leaf. Figure 4.31(a), there are four petals, five petals in Figure 4.31(b), six petals in Figure 4.31(c), 11 petals in Figure 4.31(d), 11 petals in Figure 4.31(e) with leaf texture, and six petals with leaf texture mapping in Figure 4.31(e).

4.8.5 Texture mapping

In this thesis, a texture is only set to leaf component. The texture is designed for 2-dimensional texture mapping to 3-dimensional space of leaf polygon. The texture size must be $2^n x 2^m$ pixels, where *n*, *m* is 3, 4, 5, 6, 7, or 8. It is cropped from the actual leaf and set to the appropriate size. The texture will be changed the color from user adjustment. The user can create the new texture or select from the given texture library.

The texture coordinate is mapped to range of [0,1]. For example, if the texture size is 64x64 pixels, the texture coordinate will map to 1x1. Figure 4.32 shows the mapping coordinate. The texture coordinate (0,0) is mapped to p0 (0,0), (63,0) to p1 (1,0), (0,63) to p2(0,1), (63,63) to p3(1,1), (31,31) to p4(0.5,0.5) as well as other points.

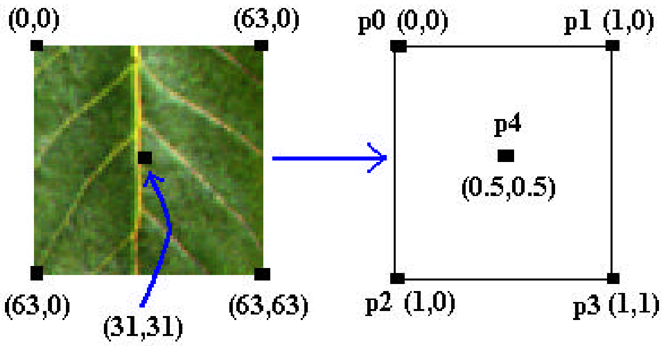


Figure 4.32: The texture coordinate.

Figure 4.33 shows a texture mapping to the leaf polygon. A leaf coordinate in three-dimensional space (Figure 4.33(a)) is mapped to range of [0,1] coordinate in XY plane (Figure 4.33(b)) corresponding to texture coordinate in Figure 4.33(d), and the texture coordinate is mapped to the actual leaf coordinate as Figure 4.33(c).

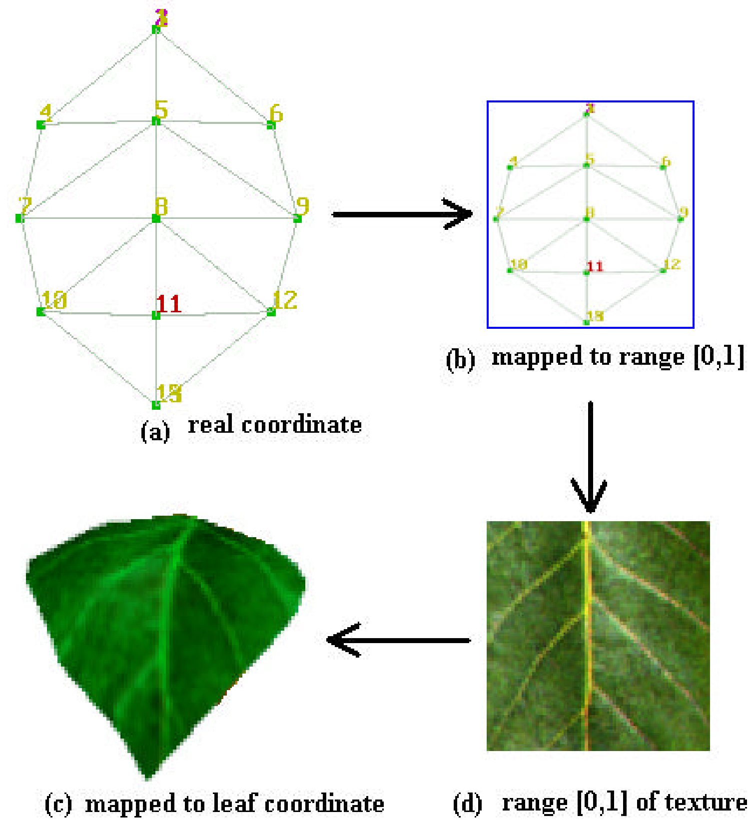


Figure 4.33: A texture mapping with leaf polygon.

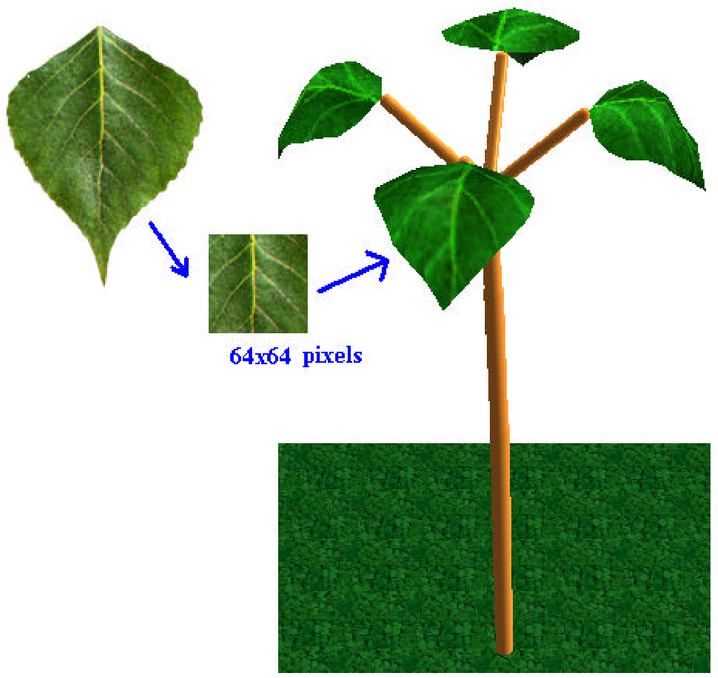


Figure 4.34: The leaf texture mapping.

The flow of leaf texture mapping is shown in Figure 4.34. The texture is scanned by scanner from the actual leaf, and cropped to the appropriate size, then map to the three-dimensional space of leaf point. The three-dimensional leaf can be changed the material color. Figure 4.35 shows a simple plant with its textured leaves and flowers. The L-system string is

I[+iL][-iL]I[-IF][+IF][/IF][\IF]iF

There are two internodes *I*, and a short internode *i* for main stem, a left short internode and its leaf [+iL], a right short internode and its leaf [-iL]. There are four internodes and their flowers in four directions, and a center short internode with its flower. The top view of the simple plant is shown in Figure 4.36.

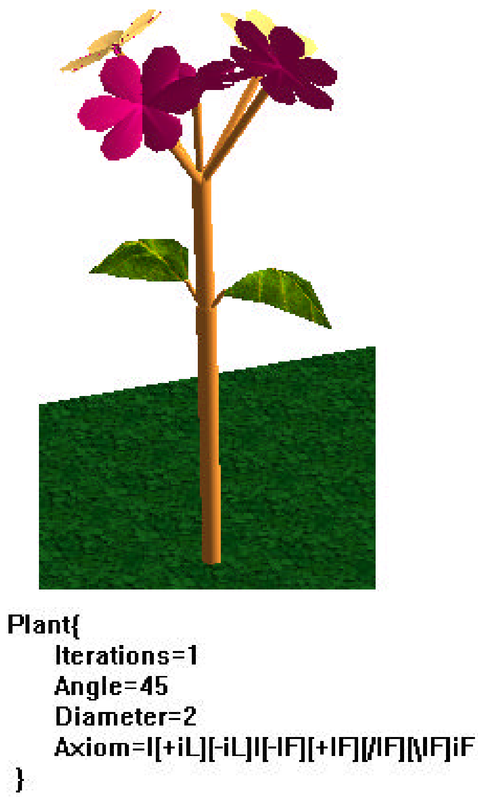


Figure 4.35: A simple plant with its leaves and flowers.

Figure 4.36: Plant and their component with textured leaves.

In the case of soybean, the texture of the soybean leaf is cropped from the actual soybean in this thesis experiment. It is mapped to the leaf polygon of soybean plant in Figure 4.37.

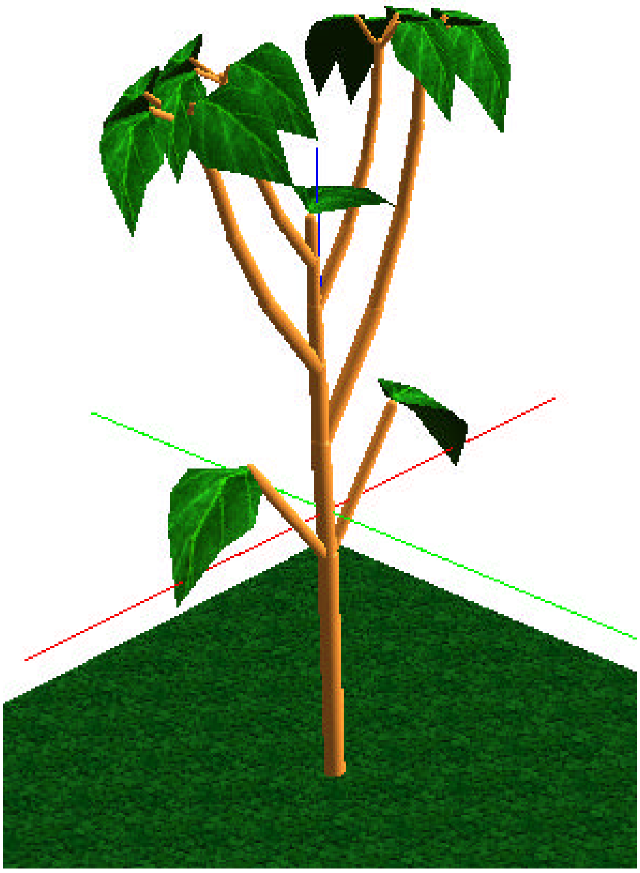


Figure 4.37: A soybean and textured leaves with *Soybean* prototype using two iterations.

Figure 4.38 visualize the *Soybean* prototype in Section 4.3 using six iterations as well as in Figure 4.39 is added the soybean texture to its leaves.

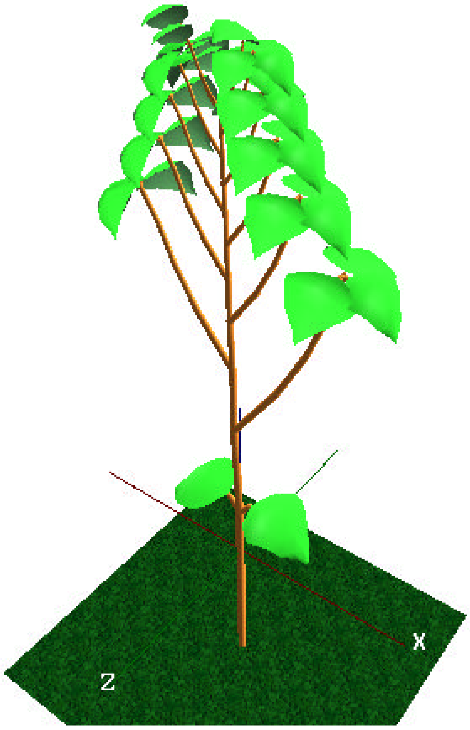


Figure 4.38: The soybean as follows Section 4.3 using 6 iterations.

Figure 4.39: The soybean as follows Section 4.3 using 6 iterations with textured leaves.

4.9 Model Evaluation

This thesis has the capability to adjust the parameters of the plant model interactively. This allows the designer to verify the production rules and to modify the appearance of the graphical image of the generated plant in real time mode. In addition, if there are any flaws presented in the plant model due to the production rules, the designer can edit the rules and recompile the L-system description.