

9. Creación de sistemas L con LS-SketchBook

9.1 La definición del sistema L

La definición de un sistema L especifica aspectos que tienen que ver con la propia definición formal del sistema L, reglas de producción, axioma, etc., y su interpretación geométrica. La definición de un sistema L viene a ser el equivalente al código fuente de un programa escrito en un lenguaje de programación. En realidad, puede considerarse como un lenguaje simplificado y especializado en la definición de sistemas L.

Por ejemplo, la siguiente definición corresponde a una de las estructuras arbóreas de Honda:

```
SistemaL MonopodicoHondaA;

// estructura monopódica arbórea según Honda

define r1=0.9; // razón de contracción en el tronco
define r2=0.6; // razón de contracción en las ramas
define a0=45; // ángulo de ramificación respecto al tronco
define a2=45; // ángulo de ramificación en las ramas laterales
define wr=1/sqrt(2); // razón de estrechamiento
define d=FibAngle; // ángulo de divergencia

indicecolor = 1;
elementobase = {
  forma = cilindro;
  nocaras = 8;
};

noterminales = {A,B,C};
axioma = A(50,10);
derivaciones = 10;

producciones = {
  p1: A(l,w) : true -> "(w)F(l)[&(a0)B(l*r2,w*wr)]/(d)A(l*r1,w*wr);
  p2: B(l,w) : true -> "(w)F(l)[- (a2)$C(l*r2,w*wr)]C(l*r1,w*wr);
  p3: C(l,w) : true -> "(w)F(l)[+ (a2)$B(l*r2,w*wr)]B(l*r1,w*wr);
};
```

La definición formal del sistema L es obligatoria pero los aspectos relacionados con la interpretación geométrica son opcionales. Al omitir alguno de ellos se asume su valor por defecto.

Cada sentencia o definición de un aspecto del sistema L se termina con un punto y coma que actúa como separador de sentencia. Los conjuntos se delimitan con llaves. Aunque no hay límite en la longitud de cada línea de texto, una misma definición puede ocupar varias líneas, por ejemplo en el caso de reglas de producción con condiciones o sucesores demasiado grandes, para hacer la definición más legible.

Los identificadores de los parámetros y los símbolos de los módulos son sensibles a las mayúsculas. No así las palabras reservadas para la definición del sistema L o las funciones. Pueden contener números pero deben comenzar por una letra.

Para una descripción detallada de la sintaxis de las definiciones de los sistemas L se puede consultar el anexo que contiene la gramática completa de las definiciones.

Nombre del sistema L

Cada sistema L posee un nombre o identificador a modo de título descriptivo. Generalmente se coloca al principio de la definición. Por ejemplo

```
SistemaL PodaBasitona;
```

El nombre del sistema L no tiene por qué coincidir con el nombre del archivo DSL en el que se guarda su definición.

Comentarios

Se pueden escribir notas o comentarios en la definición. El analizador sintáctico ignora todos los caracteres que van después del símbolo // hasta el final de línea. Por ejemplo:

```
define w0 = 1/sqrt(2);           // grosor inicial del tronco
```

Definición de parámetros constantes

Cada sistema L puede definir un conjunto de parámetros constantes con un significado particular. Su cometido es muy similar al de las constantes de los lenguajes de programación. Facilitan la lectura e interpretación de las expresiones matemáticas y estructuran la información relevante del sistema L. Su definición se hace con la palabra reservada *define*.

La siguiente definición incluye varios parámetros significativos en la construcción del sistema L:

```
SistemaL MonopodicoHondaC;
```

```
define r1=0.9;           // razón de contracción en el tronco
define r2=0.8;           // razón de contracción en las ramas
define a0=45;            // ángulo de ramificación respecto al tronco
define a2=45;            // ángulo de ramificación en las ramas laterales
define wr=1/sqrt(2);     // razón de estrechamiento
define d=FibAngle;      // ángulo de divergencia
```

```
noterminales = {A,B,C};
axioma = A(50,10);
derivaciones = 7;
```

```

producciones = {
  p1: A(1,w) : true -> "(w)F(1)[&(a0)B(1*r2,w*wr)]/(d)A(1*r1,w*wr);
  p2: B(1,w) : true -> "(w)F(1)[- (a2)$C(1*r2,w*wr)]C(1*r1,w*wr);
  p3: C(1,w) : true -> "(w)F(1)[+(a2)$B(1*r2,w*wr)]B(1*r1,w*wr);
};

```

La definición de un parámetro constante puede estar en función de otro parámetro constante siempre y cuando haya sido definido previamente.

Alfabeto de símbolos

Según la definición formal de los sistemas L, las letras de todos los módulos pertenecen a un alfabeto de símbolos. En ese alfabeto se encuentran de manera preestablecida todos aquellos símbolos estándar reconocidos por el intérprete de cadenas de módulos. Su significado es invariable para todos los sistemas L, por ejemplo, los símbolos de giro, de ramificación, etc. Pero, generalmente, también será necesario añadir otros símbolos a los que se otorga otro sentido particular del sistema L. Estos símbolos definidos por el usuario no son interpretables y cuando el intérprete de cadenas de módulos los encuentra simplemente los ignora.

La forma de agregar estos últimos símbolos al alfabeto del sistema L es

```
noterminales = {A,B,C,D};
```

En este caso se han añadido los símbolos *A*, *B* y *C*. Son referidos como no terminales porque no tienen valor en la última cadena generada en el proceso de derivación. El símbolo de todos los módulos que aparecen en la definición del axioma y las reglas de sustitución deben pertenecer al alfabeto del sistema L.

Los símbolos que pueden utilizarse son todos los caracteres ASCII comprendidos entre el número 33 y el número 127, ambos incluidos, excluyendo todos aquellos que tienen un significado preestablecido, es decir,

```
F, f, [, ], %, -, +, |, &, ^, /, \, #, !, @, ", ' ,
```

y aquellos que crearían ambigüedades en el análisis sintáctico de las cadenas de módulos, o sea:

```
;, :, <, >, (, ), ,
```

Cadenas de módulos

Una cadena de módulos está formada por una secuencia lineal de módulos. Cada módulo consta de un símbolo y, posiblemente de un conjunto de parámetros agrupados entre paréntesis y separados por comas, como si se tratase de una función. Según la forma de los parámetros se hablará de módulos reales, formales o predecesores.

Módulos reales

Los parámetros de los módulos reales, si los hay, son números reales. Su significado depende de su posición y del símbolo del módulo. En el caso de los símbolos interpretables, los primeros parámetros, es decir, los situados más a la izquierda, son los que se utilizan para ajustar la interpretación del módulo. Los demás son usados libremente para incluir otra información, pero el intérprete no puede hacer uso de ellos. Por ejemplo, en la siguiente cadena de módulos reales

```
F(1.5,18.0)++/(30.0)A(3,0)
```

el símbolo de trazado tiene asociados dos parámetros. Pero sólo el primero de ellos, 1.5, es utilizado por el intérprete. Los módulos de giro a la izquierda no tienen ningún parámetro por lo que a la hora de realizar la interpretación el intérprete les asignará el valor del ángulo establecido por defecto. El módulo A , no es terminal por lo que el número de sus parámetros es indiferente.

Módulos predecesores

Los módulos formales son los que aparecen en el predecesor de las reglas de producción. Nunca forman cadenas de módulos. Sus parámetros son los identificadores de los parámetros formales de la regla de producción. En la regla de producción

$$p_1 : A(n) : n > 10 \rightarrow F[+FB][-FC]A(n-1);$$

el módulo predecesor es $A(n)$ y el parámetro formal de la regla de producción es n . En el caso de que el nombre de un parámetro formal coincida con el de un parámetro constante, el primero tiene mayor prioridad, y cualquier aparición en una expresión de la regla de producción correspondiente al parámetro formal hará referencia al parámetro formal. Es decir, el parámetro formal actúa como si se tratase de una variable local frente a una global, el parámetro constante.

Durante el proceso de producción de cadenas, el módulo predecesor determina parcialmente, en función de su símbolo y número de parámetros, si una regla es concordante o no con un módulo real.

Módulos formales

Los módulos formales forman los sucesores de las reglas de derivación. Sus parámetros son expresiones en las que pueden participar los parámetros formales de su respectiva regla de derivación, además de todas las funciones, constantes, etc. Su valor permanecerá indefinido si incluye algún parámetro formal y sólo podrá ser evaluado en el momento en que la regla se aplica a un módulo real concreto.

Axioma

El axioma del sistema L consiste en una cadena de módulos reales. Se establece con la palabra reservada *axioma*, como en

$$\text{axioma} = FA(1); \quad // \text{ crecimiento a partir de un entrenodo}$$

La definición del axioma debe terminar siempre con un punto y coma, al igual que el resto de definiciones del sistema L. Ello no significa que el punto y coma sea un módulo más del axioma; de hecho, no se admite el punto y coma como símbolo no terminal, ya que provocaría ambigüedades en el análisis sintáctico de las cadenas de módulos.

Reglas de producción

El conjunto de reglas de producción del sistema L se define mediante la palabra reservada *producciones*. Siempre debe existir al menos una regla de producción. Cada regla consta de un identificador, un módulo sucesor, una condición y el sucesor. Por ejemplo:

$$\begin{aligned} \text{producciones} = \{ \\ p_1 : A(v) : v < 16 \rightarrow [-FB(v)][+FB(v)]FA(v+1); \\ p_2 : B(v) : v > 0 \rightarrow FB(v-1); \\ \}; \end{aligned}$$

La definición de cada regla debe terminar siempre con un punto y coma, al igual que el resto de definiciones del sistema L. Ello no significa que el punto y coma sea un módulo más del sucesor; de hecho, no se admite el punto y coma como símbolo no terminal, ya que provocaría ambigüedades en el análisis sintáctico de las cadenas de módulos.

Para indicar que la condición de la regla se verifica siempre se puede escribir

$p1 : A : \text{true} \rightarrow [-\text{FB}][+\text{FB}]FA;$

El orden de definición de las reglas puede ser determinante. En el caso de que varias reglas concuerden con un mismo módulo, siempre prevalecerá la definida con anterioridad.

Sensibilidad al entorno

La sensibilidad al entorno ofrecida por LS-SketchBook es bastante reducida y se limita a la prevista por los sistemas L sensibles al contexto, bastante inferior a la ampliación dada por los sistemas L abiertos. Sin embargo, la forma de implantarla es mucho más simple, tanto desde el punto de vista de uso como del de ejecución.

La sensibilidad al entorno se introduce a través de un conjunto de variables de entorno a las cuales se puede tener acceso en cualquier momento de la producción como si se tratasen de funciones sin parámetros. El nombre de las variables y su significado es el siguiente:

Nombre de la variable de entorno	P_x, P_y, P_z	H_x, H_y, H_z	L_x, L_y, L_z	U_x, U_y, U_z	N_r
Significado de la variable	Posición	Vector Frente (H)	Vector Izquierda (L)	Vector Arriba (U)	Nivel de recursión

Tanto la posición como los vectores de orientación se han separado en tres variables, una por cada coordenada espacial. Su valor es siempre el actual en el momento de acceder a ellas al igual que el nivel de recursión que indica el orden de ramificación o, lo que es lo mismo, el número de estados que en ese momento dado están memorizados (ramificaciones iniciadas y no completadas).

Por ejemplo, la siguiente regla de producción tiene como condición para su aplicación el que la posición actual, en el plano XY, no se encuentre dentro del círculo centrado en el origen y de radio r . En caso de que se verifique, el primer módulo del sucesor establece como color actual aquél cuyo índice de color es igual al nivel de recursión para, por ejemplo, mostrar en distintos colores cada rama de la estructura según sea su orden.

$p1 : A : \text{sqrt}(\text{sq}(P_x) + \text{sq}(P_y)) > r \rightarrow \#(N_r) + (10)\text{FB};$

Un aspecto que hay que subrayar es el relativo a los identificadores de las variables de entorno. Es posible definir un parámetro formal o un parámetro de usuario con el mismo nombre de alguna de las variables de entorno, solamente que prevalecerá entonces el significado de estos últimos con lo que se perderá el acceso a esa variable de entorno.

Generador de números aleatorios

A menudo resulta necesario añadir factores aleatorios al sistema L. Bien sea para modificar el valor de determinados parámetros de manera que no se aprecie demasiada regularidad o para poder seleccionar al azar una entre varias producciones. Las funciones

```
randomu(a,b)
```

y

```
randomn(m,d)
```

calculan un número al azar siguiendo una distribución uniforme en el intervalo $[a,b]$, en el primer caso, o una normal de media m y desviación típica d , en el segundo.

La semilla del generador de números aleatorios se toma al azar a menos que se proporcione una determinada, por ejemplo

```
semillaaleatoria = 850;
```

De esta manera se tiene siempre la misma serie de números aleatorios consiguiendo que, aun introduciendo factores aleatorios, siempre se alcancen las mismas cadenas de módulos.

9.2 La producción de cadenas de módulos

Número de derivaciones

El número de etapas de derivación que se realizan en el proceso de producción de cadenas se fija con la palabra reservada *noderivaciones*. Por ejemplo

```
noderivaciones = randomu(4,9); // edad de la estructura aleatoria
```

El resultado de la expresión en la parte derecha siempre se redondea al entero más próximo.

9.3 La interpretación geométrica

Símbolos reconocidos por el intérprete de cadenas de módulos

El intérprete de cadenas de módulos es capaz de reconocer los siguientes símbolos y los interpreta de la forma descrita:

Cambio en la orientación

Símbolos que modifican la orientación de la tortuga al hacerla girar sobre si misma.

Símbolo/ Módulo	Nombre	Significado
+(a)	Giro a la izquierda	Realiza un giro a la izquierda de a grados alrededor del eje U. Si no se indica ningún

		parámetro se toma el ángulo por defecto.
-(a)	Giro a la derecha	Realiza un giro a la derecha de a grados alrededor del eje U. Si no se indica ningún parámetro se toma el ángulo por defecto.
	Media vuelta	Giro de 180 grados alrededor del eje U. Es equivalente a +(180) o -(180).
^(a)	Inclinación hacia arriba	Realiza un giro hacia arriba de a grados alrededor del eje L. Si no se indica ningún parámetro se toma el ángulo por defecto.
&(a)	Inclinación hacia abajo	Realiza un giro hacia abajo de a grados alrededor del eje L. Si no se indica ningún parámetro se toma el ángulo por defecto.
/(a)	Ladeo a la izquierda	Realiza un giro hacia la izquierda de a grados alrededor del eje H. Si no se indica ningún parámetro se toma el ángulo por defecto.
\(a)	Ladeo a la derecha	Realiza un giro hacia la derecha de a grados alrededor del eje H. Si no se indica ningún parámetro se toma el ángulo por defecto.

Cambio de posición y trazado de objetos

Símbolos que desplazan la tortuga a la vez que trazan, o no, un elemento base.

Símbolo/ Módulo	Nombre	Significado
f(d)	Salto adelante	Desplazamiento hacia adelante una distancia d sin trazar nada. Si no se indica ningún parámetro se toma la distancia por defecto.
F(d)	Trazado	Desplazamiento hacia delante trazando un elemento base de longitud d. Si no se indica ningún parámetro se toma la distancia por defecto.

Ramificaciones

Símbolos que aumentan o disminuyen el nivel de ramificación o que podan ramas.

Símbolo/ Módulo	Nombre	Significado
[Guarda estado	Memoriza el estado actual de la tortuga, incluyendo su posición, orientación, atributos de trazado, etc.
]	Recupera estado	Restablece el estado actual de la tortuga a partir del último estado previamente memorizado.
%	Corte	Corta el resto de la rama, es decir, todos los símbolos que siguen hasta alcanzar el próximo símbolo de recuperación de estado.

Cambio de atributos de trazado

Símbolos que modifican las propiedades del trazado de los elementos base.

Símbolo/ Módulo	Nombre	Significado
“(g)	Aumento del grosor	Fija el ancho de la línea al valor indicado por g. Si no se indica ningún parámetro se aumenta el ancho actual la cantidad determinada por el incremento de grosor por defecto.
`(g)	Disminución del grosor	Fija el ancho de la línea al valor indicado por g. Si no se indica ningún parámetro se aumenta el ancho actual la cantidad determinada por el incremento de grosor por defecto.
#(n)	Incremento del índice de color	Establece el índice de color al valor indicado por n. Si no se indica ningún parámetro se incrementa el índice del color la cantidad determinada por el incremento de índice de color por defecto.
!(n)	Decremento del índice de color	Establece el índice de color al valor indicado por n. Si no se indica ningún parámetro se incrementa el índice del color la cantidad determinada por el incremento de índice de color por defecto.
@(a,b,c)	Establece color	Establece como color actual el definido por las tres componentes correspondientes al espacio de color usado en la interpretación: r, g y b en el espacio RGB, o h, s y b en el espacio HSB. Sin ningún parámetro utiliza el color señalado por el índice actual de la tabla de colores.

Valores por defecto en la interpretación

Todos los módulos que efectúan operaciones de giros o desplazamientos asumen un ángulo o una distancia por defecto, a falta de que se indique otro. Es decir que si, por ejemplo, en una cadena se introduce un módulo de giro sin ningún parámetro, sólo su símbolo, al interpretarlo se hará un giro de tantos grados como valga el ángulo por defecto. La ventaja que tiene esto es que durante la producción de cadenas se ahorra el espacio y el tiempo necesario para manejar un parámetro que puede tener siempre el mismo valor y aparecer repetido en un gran número de módulos. Posteriormente, el intérprete de cadenas de módulos se encarga de completar toda la información ausente con los valores establecidos por defecto.

Parámetro	distancia	ángulo	índice de color	grosor	variación del ancho de línea
Valor por defecto	1	90°	1	1	0.1
Modificador	<i>paso</i>	<i>angulo</i>	<i>indicecolor</i>	<i>ancholinea</i>	<i>incdecancholinea</i>

Cada uno de estos valores se puede modificar en la definición del sistema L. Su valor no tiene por qué ser exactamente un valor constante, sino que puede venir determinado por otros parámetros. Por ejemplo:

```
paso = sqrt(2);  
ángulo = round(360/(N+1));  
grosor = 0.2;
```

Colores

Existen dos modos de establecer el color de los objetos: mediante tablas de colores o mediante espacios de color.

La primera opción consiste en reunir en una tabla las definiciones de todos los colores usados por el sistema L. Cada color tiene asignado un índice. Durante la interpretación geométrica el color que se utiliza para trazar los objetos gráficos es el correspondiente al índice de color actual. Ese índice se puede fijar, incrementar o decrementar con los símbolos respectivos. La tabla de colores que se debe aplicar se indica mediante el nombre del archivo en el que se guarda, por ejemplo:

```
tablacolores = "cobrizos.TCL";
```

En ocasiones el color de una parte determinada de la planta viene fijado por determinados parámetros sólo conocidos durante la interpretación. Por ejemplo, puede depender de la edad u orientación de esa parte. Esto se opone a la definición estática de las tablas de colores. Por ello se hace necesario introducir otro símbolo que defina el color directamente. La definición se hace mediante tres parámetros cuyo significado depende del modelo de color escogido.

```
modelocolor = RGB;
```

selecciona el modelo RGB. Los tres parámetros que determinan el color son las contribuciones de cada uno de los colores básicos rojo, verde y azul. Aunque este modo de indicar un color es el más habitual, no resulta sencillo pensar en un color en esos términos. Por el contrario,

```
modelocolor = HSB;
```

emplea tres componentes mucho más intuitivas: el matiz, la saturación o pureza del color y su luminosidad. El inconveniente de usar el modelo HSB es que siempre hay que convertir los colores a modo RGB, que es el que utilizan los ordenadores en sus elementos gráficos. Esa conversión es rápida y sólo se efectúa en el proceso de interpretación geométrica por lo que generalmente no se notará la diferencia.

En ambos casos, los valores de los parámetros deben estar comprendidos entre 0 y 1, salvo en el caso del matiz que se especifica mediante un ángulo entre 0° y 360°. En cualquier caso, todos los valores fuera del intervalo correcto son saturados, es decir, se mantienen en el valor máximo permitido si lo superan o el mínimo si lo rebajan.

Primitivas gráficas simples

La representación más elemental de un sistema L se hace con líneas. Las líneas pueden tener diferente grosor pero en ningún caso tienen volumen, tanto si se trata de figuras planas como de estructuras espaciales. Por ello las estructuras representadas mediante líneas tienen siempre un carácter muy esquemático.

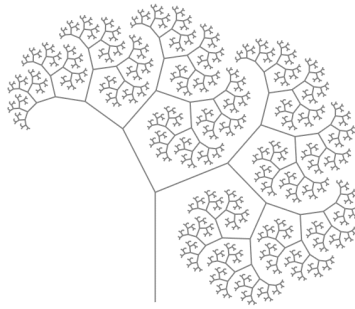


Figura 55: Representación mediante líneas de una ramificación.

Para determinados tipos de estructuras, como curvas fractales, la representación con líneas es suficiente. Pero, por lo general, la geometría de estructuras botánicas exige formas con volumen. Una primera solución consiste en emplear primitivas gráficas simples en lugar de líneas. Las formas más sencillas son rectángulos, bloques y cilindros.

La forma de indicar la primitiva de trazado es la palabra reservada *elementobase* para determinar distintos aspectos del elemento constructivo básico de la geometría. Por ejemplo:

```
elementobase = {
  forma = cilindro;
};
```

Si no se indica lo contrario, la forma empleada por defecto es la línea. Las formas simples disponibles son líneas, rectángulos, cruces, bloques y cilindros.

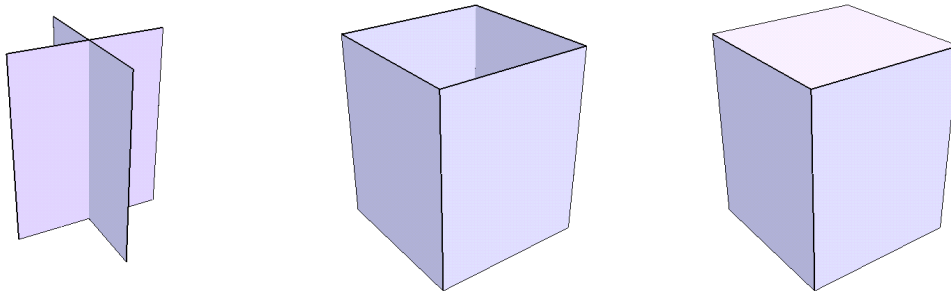


Figura 56: Primitivas gráficas simples (de izquierda a derecha): cruz, bloque abierto y bloque cerrado.

Según se trate de una u otra forma, se pueden modificar algunas de sus propiedades. En el caso de los bloques y cilindros se permite escoger entre cuerpos abiertos o cerrados.

```
elementobase = {
  forma = bloque;
  cubierta = abierto;
};
```

Y en el caso de los cilindros se puede seleccionar el número de caras laterales.

```
elementobase = {
  forma = cilindro;
  cubierta = cerrado;
```

```
nocaras = 16;  
};
```

El número mínimo de caras es 3 y el máximo 24.



Figura 57: Primitivas gráficas simples: cilindro de 16 caras abierto (izquierda) y cerrado (derecha).

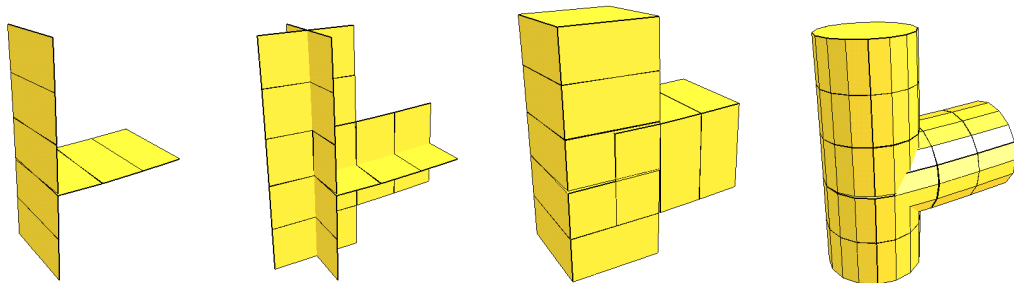


Figura 58: Comparación de la construcción de la base de una ramificación con las primitivas simples (de izquierda a derecha): rectángulos, cruces, bloques y cilindros de 16 caras.

La ventaja de las primitivas simples se deriva de que, precisamente al ser tan simples, el coste computacional de su representación es bastante reducido. Para realizar los primeros esbozos de una estructura pueden ser adecuadas, en especial si el número de elementos de la estructura es elevado. Pero, a cambio de esa ventaja, presentan dos problemas principales a la hora de construir estructuras a partir de su yuxtaposición: el cambio de grosor provoca escalonamientos y, a partir de cierto grado de curvatura más bien bajo, se nota la discontinuidad entre primitivas simple adyacentes.



Figura 59: Problemática de las primitivas simples: escalonamiento producido por la variación del grosor (izquierda) y discontinuidades en las curvaturas (derecha).

La solución de los inconvenientes de las primitivas simples viene de la mano de la generalización del concepto de cilindro: el cilindro generalizado.

Cilindros generalizados

Al modelar formas orgánicas, en particular plantas, es muy conveniente disponer de un mecanismo que permita crear elementos con curvaturas suaves. Un enfoque es utilizar objetos elementales, como cilindros o bloques, para aproximar con mayor o menor detalle la curvatura. Si bien ese método permite controlar el grado de curvatura mediante las mismas reglas de producción tiene el grave inconveniente de que aumenta sobremanera el tamaño de las cadenas de módulos.

El cilindro generalizado es una extensión del cilindro tradicional. Difiere en dos características fundamentales. El eje deja de ser un simple segmento rectilíneo y pasa a ser un trayecto curvo y, la sección de corte puede tener cualquier forma, no sólo un disco, y variar en tamaño o incluso forma a lo largo del trayecto que actúa como eje. Gracias a esta versatilidad el cilindro generalizado sirve adecuadamente para representar una gran variedad de formas, en especial muchas formas naturales.

Construcción del eje

LS-SketchBook construye los ejes de los cilindros utilizando curvas de Hermite de tres dimensiones. Una curva de Hermite es un tipo de curva paramétrica cúbica. Permite interpolar suavemente y sin singularidades los valores entre dos puntos conociendo las tangentes en dichos puntos. Otras curvas paramétricas cúbicas bastante conocidas son curvas de Bezier, los splines Catmull-Rom, los splines de Kochanek-Bartels, también llamados splines TCB o los B-splines. Todos ellos se utilizan frecuentemente en distintos ámbitos de la informática gráfica, no sólo para construir curvas o superficies sino también para parametrizar transformaciones de objetos.

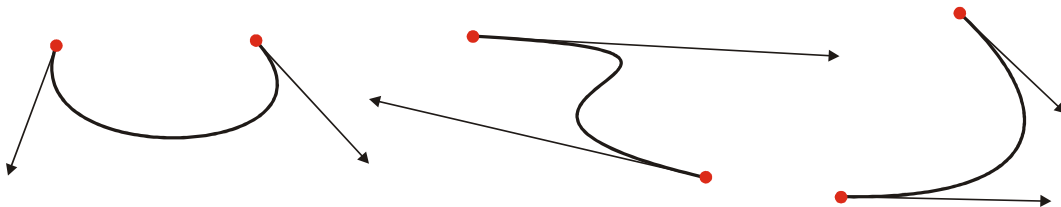


Figura 60: Algunas curvas de Hermite con los puntos de control y sus tangentes respectivas.

Las curvas paramétricas cúbicas se construyen empleando un conjunto de funciones combinadas entre sí y aplicadas a los elementos de control, puntos o tangentes. Las cuatro funciones que forma la base de Hermite son las siguientes, con t perteneciente al intervalo $[0,1]$.

$$h_1(t) = 2t^3 - 3t^2 + 1$$

$$h_2(t) = -2t^3 + 3t^2$$

$$h_3(t) = t^3 - 2t^2 + t$$

$$h_4(t) = t^3 - t^2$$

Los valores $t=0$ y $t=1$ corresponden a los extremos de la curva, es decir, los puntos de control. Al tratarse de polinomios sus derivadas siempre están definidas con lo que se evitan singularidades en el valor de la pendiente en cualquier punto.

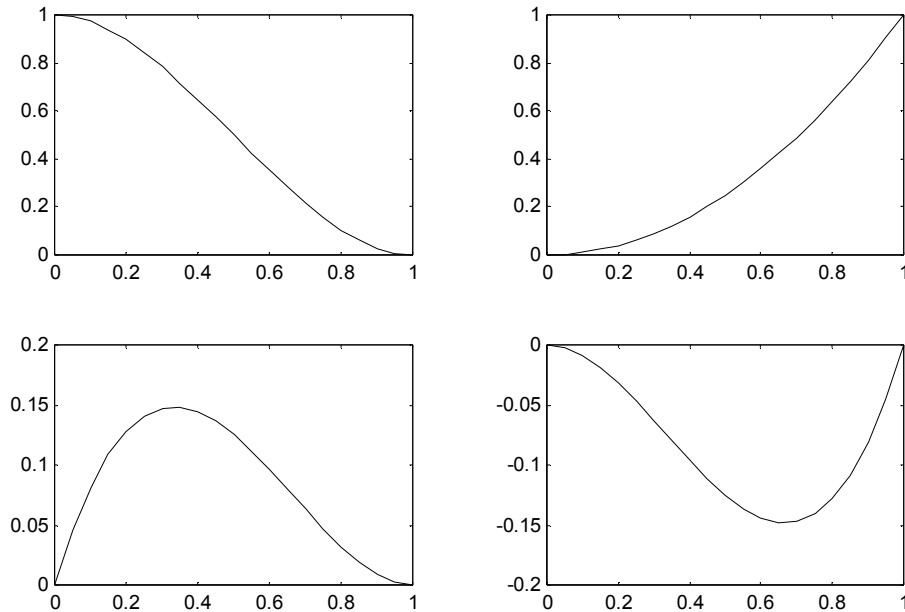


Figura 61: Los cuatro polinomios que forman la base de funciones de Hermite.

La ecuación paramétrica de las curvas de Hermite es de la forma

$$Q_H(t) = TM_H G_H$$

donde la matriz M_H contiene los coeficientes de los polinomios que forman la base de Hermite, la matriz G_H las posiciones y tangentes de los puntos de control y la matriz T es la matriz de tiempos. Expresado matricialmente, la ecuación es

$$Q_H(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} P_{1x} & P_{1y} & P_{1z} \\ P_{2x} & P_{2y} & P_{2z} \\ R_{1x} & R_{1y} & R_{1z} \\ R_{2x} & R_{2y} & R_{2z} \end{bmatrix}$$

siendo P la posición y R el vector tangente en el primer y segundo punto de control.

En el contexto de la geometría de la tortuga, cada punto de control queda marcado por las sucesivas posiciones de la tortuga y las tangentes en dichos puntos las fija el vector H que señala la dirección frontal de la tortuga. El módulo de las tangentes se calcula como la distancia euclídea entre los dos puntos de control multiplicado por el coeficiente de tensión que es el mismo para ambas. De esta manera, el peso de las tangentes varía junto con la longitud de la curva impidiendo que, por ejemplo, tangentes de módulo muy pequeño apenas tengan influencia en curvas muy largas y alteren su curvatura.

Una vez conocida la ecuación paramétrica de cada tramo del eje es fácil obtener la situación de todos los puntos intermedios necesarios situados entre los dos puntos de control. Cuantos más puntos se interpolen más suave aparecerá la curvatura del cilindro generalizado. Pero esa mayor precisión también conlleva mayores exigencias de cálculo,

especialmente en la fase de representación. Si la ecuación paramétrica de la curva que constituye el tramo del eje es $Q_H(t)$ la posición del i -ésimo de los n puntos intermedios se calcula evaluando $Q_H(i/(n+1))$. La tangente en ese punto se halla evaluando $Q'_H(i/(n+1))$.

La sección de corte

Las secciones de corte o contornos del cilindro generalizado usadas son poligonales definidas en los archivos CON de contorno. Están formados por una serie de vértices ordenados, no más de 100, y pueden ser abiertas o cerradas. Si son cerradas el último vértice se une con el primero. Las coordenadas de los vértices vienen expresadas respecto a un sistema de referencia local al contorno. Es importante que el centro de la poligonal se sitúe en el origen de dicho sistema de referencia para que al construir el cilindro generalizado quede correctamente alineado con el eje del cilindro generalizado.

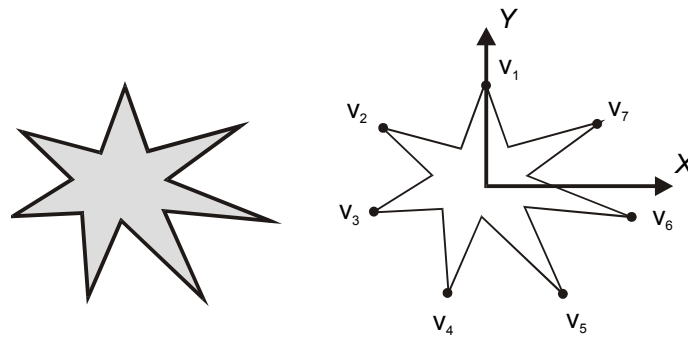


Figura 62: Un contorno poligonal cerrado con sus vértices definidos en un sistema de referencia local.

Al construir el cilindro generalizado el contorno debe ser trasladado desde su plano de referencia local hasta el punto del eje correspondiente del cilindro. Y también debe orientarse adecuadamente. La situación de los vértices C_i del contorno en su nueva situación se obtienen con

$$C_i = P + r(v_{ix}U + v_{iy}V)$$

De esa manera el punto $(0, 0, 0)$ en el sistema de referencia del contorno se hace coincidir exactamente con el punto del eje del cilindro generalizado. Por otro lado, la orientación del contorno se establece alineando la dirección de los ejes X e Y del sistema de referencia del contorno con la marcada por los vectores U y V del punto del eje respectivamente.

Los vectores U y V de cada punto de control equivalen a los opuestos de la dirección izquierda (L) y superior (U) de la tortuga en el punto. En los puntos intermedios a los puntos de control se interpolan utilizando las ecuaciones paramétricas de la curva de Hermite. Ambos vectores siempre forman un ángulo de 90° entre sí y determinan el plano cuya normal es el vector tangente en el punto del eje (la dirección frontal del vector H de la tortuga). Es decir

$$U \times V = T$$

La misma operación se repite sucesivamente sobre cada uno de los puntos del eje del cilindro generalizado, y, finalmente, se forma la superficie del cilindro generalizado triangulando los vértices de las secciones adyacentes.

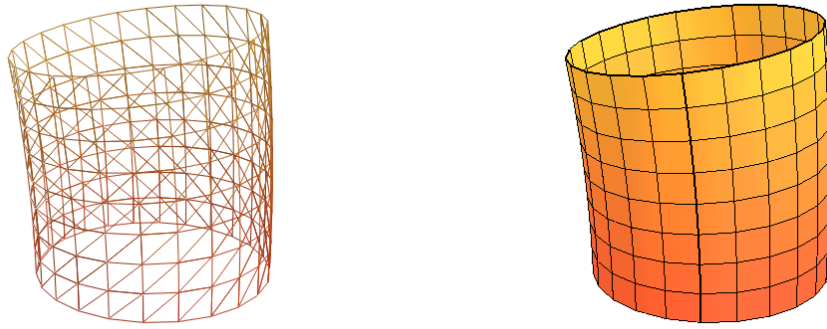


Figura 63: Construcción de la superficie de un cilindro generalizado.

Tanto el grosor como el color en cada sección intermedia también debe ser interpolado en base al definido en las secciones correspondientes a los puntos de control. En este caso, la interpolación es lineal. En el caso del color, se interpola independientemente cada una de sus componentes en el espacio RGB. El resultado es un cambio progresivo del color y del grosor de un punto de control a otro a lo largo del cilindro generalizado.

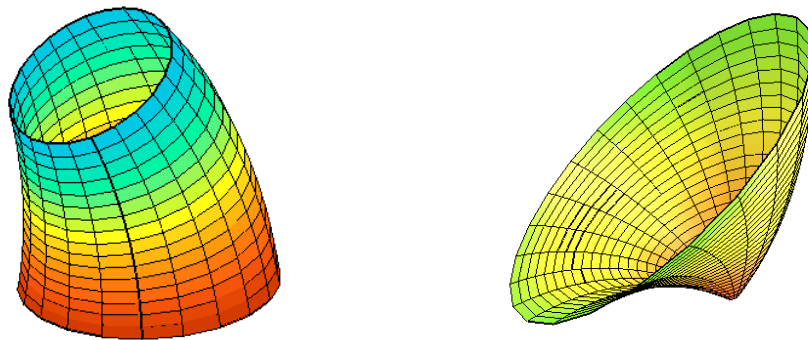


Figura 64: Interpolación del color y grosor en dos cilindros generalizado definidos por 3 y 4 puntos de control.

Ramificaciones de cilindros generalizados

Otro aspecto que resta por solucionar es el modo en que se tratan las ramificaciones. Por simplicidad, cada vez que se inicia una nueva rama, se crea un nuevo cilindro generalizado, cuyo primer punto de control pertenece al eje principal del cuál nace la rama. No obstante, la orientación, fundamentalmente la tangente o vector H , en ese punto se toma una vez producida la ramificación (después del símbolo \rfloor). En caso contrario, al mantener la orientación del punto correspondiente al eje principal se produciría una fuerte curvatura en la base de la rama.



Figura 65: Ramificación de un cilindro generalizado. La base de la rama parte desde uno de los puntos de control del eje principal, pero con la orientación establecida por la dirección de la ramificación.

El resultado es que ambos cilindros se intersecan, como se puede apreciar en la representación de líneas, pero, a menos que el grosor en la base de la rama sea mayor que el del eje principal, cosa poco habitual, la parte interior permanecerá oculta por la superficie.

Definición

La definición completa de un cilindro generalizado como elemento base incluye diferentes propiedades de distinto significado. Una definición típica podría ser la siguiente:

```
elementobase = {
  forma = tubo;
  nobandas = 8;
  tension = 1.2;
  controlextremos = central;
  contorno = "estrella 7p.con";
};
```

Las cuatro propiedades controlan aspectos fundamentales de la forma del cilindro generalizado: la precisión usada en su construcción, su grado de curvatura, la forma de construir los extremos y su sección.

Número de bandas

El número de bandas es el número de puntos intermedios interpolados entre cada par de puntos de control menos uno. Así, si se calcula un punto intermedio, el tramo del eje queda dividido en dos subtramos o bandas. Cada banda esta comprendida entre dos puntos adyacentes bien sean de control o intermedios.

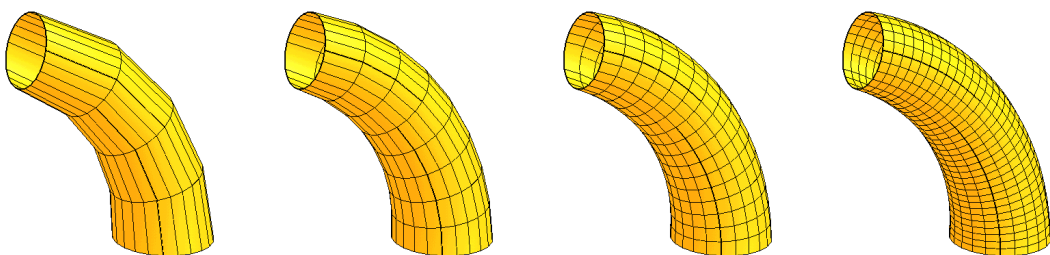


Figura 66: Cilindro generalizado construido con un número creciente de puntos intermedios (de izquierda a derecha): 1, 2, 4 y 8 bandas.

Tensión

Al hablar del escalado de las tangentes en función de la separación de los puntos de control que determinaban una curva de Hermite se mencionó un factor constante llamado coeficiente de tensión. Su valor controla la redondez de la curva.

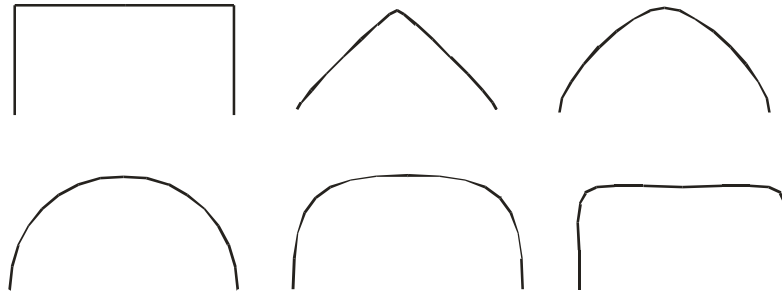


Figura 67: Efecto de la tensión en la curvatura del cilindro generalizado. El trazo original (esquina superior izquierda) ajustado con distintos valores de tensión (de izquierda a derecha y de arriba a abajo): 0.2, 0.7, 1.2, 1.7 y 2.5.

El valor de tensión que se toma por defecto y que parece producir formas más similares a las naturales es 1.2. A medida que su valor se aproxima a cero la curvatura va disminuyendo hasta que todos los puntos intermedios se encuentran alineados. Con valores superiores del coeficiente de tensión aumenta el poder de las tangentes en la ecuación paramétrica respecto al de las posiciones, haciendo que todos los puntos intermedios se sitúen a lo largo de las líneas marcadas por las tangentes en cada punto de control.

Selección de los puntos de control en los extremos

A la hora de construir el cilindro generalizado a partir de los segmentos rectos que se trazan durante la interpretación, con el símbolo de trazado, se debe decidir de qué manera se escogen los puntos de control. En principio, se marcarán en el punto central de cada segmento trazado. La excepción la constituyen el primer y último segmento del cilindro generalizado. Si también se marcaran en su punto central se descartaría medio segmento. En determinadas figuras esa supresión no se llegaría a apreciar, pero en las formadas por varios cilindros generalizados conectados entre sí se produciría una fragmentación. El caso más típico es el de las estructuras ramificadas en las que el punto de unión de las ramas con el eje principal es justamente el primer vértice del cilindro generalizado.

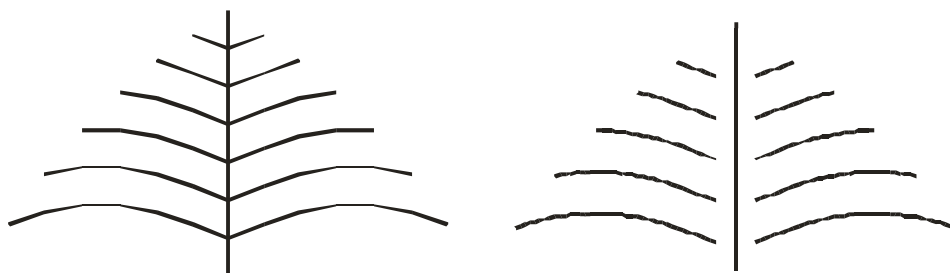


Figura 68: Fragmentación provocada al escoger como punto de control el punto medio de los segmentos inicial y final de las curvas de Hermite que representan las ramas.

La solución parece estar en seleccionar como puntos de control en esos segmentos no sus puntos centrales, sino sus vértices exteriores. Sin embargo, para otros tipos de figuras esa sería una mala solución. Así que dependiendo del tipo de figura se deberá escoger una u otra solución cambiando el valor de la propiedad *controlexremos* del cilindro generalizado. Las tres opciones son: *central*, *extremos* y *cerrado*.

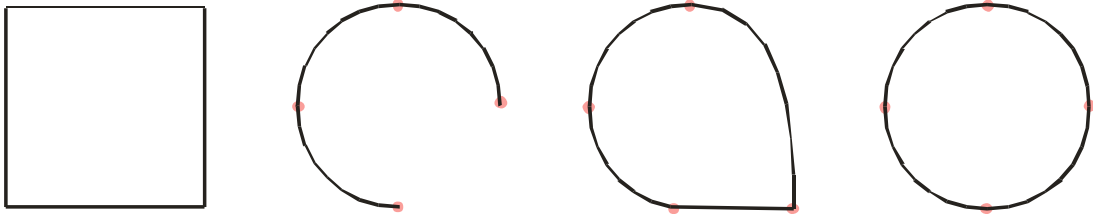


Figura 69: Situación de los puntos de control extremos. Las aristas originales (izquierda) se aproximan mediante una curva de Hermite. Los puntos de control en la primera arista y la última se marcan con (de izquierda a derecha) control central, extremos y cerrado.

La primera solución es la adecuada para trazar curvas abiertas o aisladas. La segunda sería la necesaria para estructuras ramificadas y, la tercera para curvas cerradas.

Contorno (sección de corte)

La definición de los contornos se halla en los archivos de contorno. Cada contorno puede tener hasta 100 vértices y ser abierto o cerrado.

LS-Sketchbook incluye un conjunto de contornos de uso bastante general: discos, estrellas, cruces, polígonos regulares y algunas poligonales abiertas. De todas maneras, debido a la sencillez de los archivos de contornos, resulta fácil añadir nuevos contornos adecuados a cada necesidad.

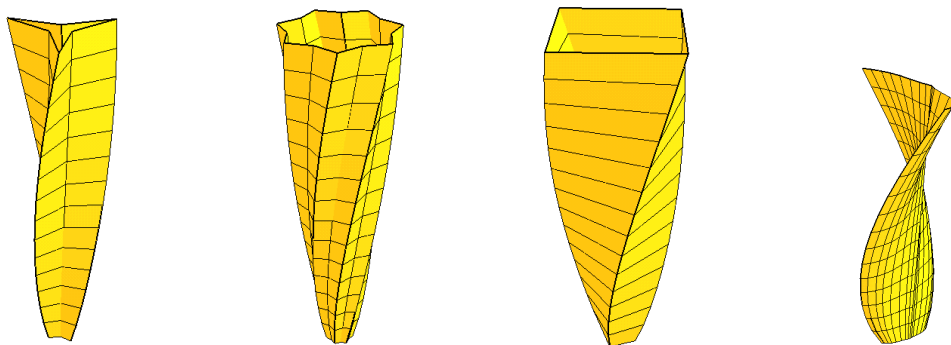


Figura 70: Diferentes contornos en un cilindro generalizado.

Si se omite el contorno, el cilindro generalizado estará formado únicamente por su eje.

9.4 Visualización

La última etapa en el ciclo de construcción del sistema L es su visualización. A través de su representación gráfica se puede verificar fácilmente si cumple los propósitos con los que fue diseñado o si, por el contrario, presenta algún tipo de falta.

Tipos de representación

LS-Sketchbook dispone de cinco estilos de representación: vértices, triángulos, líneas, sólido y combinado.

La representación más simple muestra tan sólo los vértices de los objetos gráficos que componen la escena. Su carencia de información se justifica en las ocasiones en que las escenas contienen un elevado número de objetos y su representación exige un cierto tiempo, suficiente para impedir mover la escena de manera interactiva. Se puede manipular en modo vértices y una vez situada pasar al estilo sólido para ver todos los detalles. También puede ser útil si lo que se busca en la representación es más bien obtener la forma global y aproximada de la escena, con densidades distintas según las regiones, a modo de nube de puntos, en vez de una descripción más detallada de cada uno de los objetos que la forman.

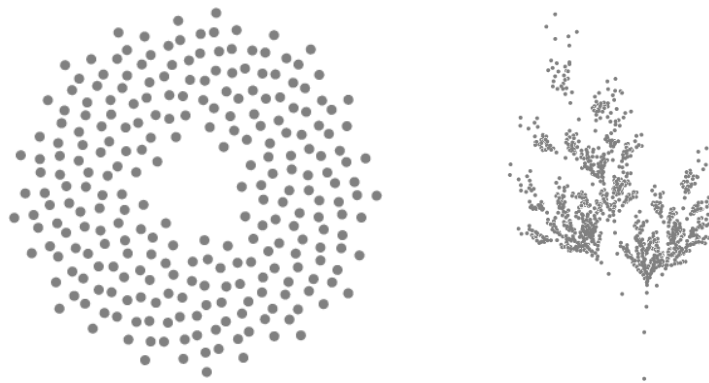


Figura 71: Representación con vértices de un patrón de filotaxis plano y de una ramificación.

Los estilos de líneas y triángulos corresponden al estilo tradicionalmente denominado *wireframe*; básicamente se trata del mismo estilo, salvo que uno de ellos omite las divisiones en triángulos de todas las caras de los objetos gráficos para evitar una sobrecarga en la visualización. En cualquier caso, el grosor de las líneas se puede modificar para adecuarlo a la complejidad de la escena, el tamaño de la ventana o la resolución de la pantalla.

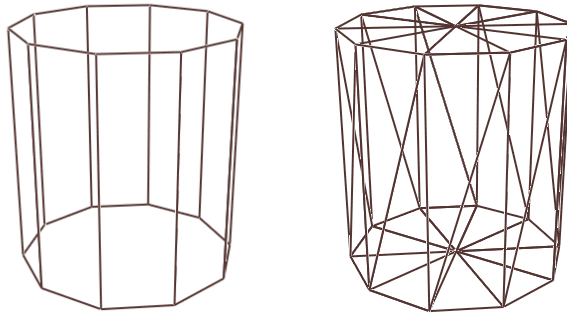


Figura 72: Representación mediante líneas y triángulos.

En el caso de figuras planas, la representación con líneas es la más adecuada, puesto que no es necesario dotar a los objetos de ningún volumen. En cambio, en escenas espaciales no es suficiente con representar las aristas de los cuerpos. También se representan sus caras que forman las superficies de los objetos.

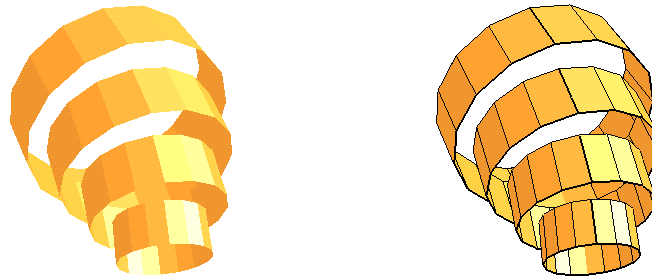


Figura 73: Representación en estilo sólido y combinado.

Finalmente, el estilo combinado consiste en una representación de superficies sólidas en las que se marcan con líneas negras las aristas de las caras.

Suavizado de líneas y puntos (antialiasing)

El antialiasing evita la aparición de líneas escalonadas en las representaciones de las aristas de los objetos suavizando sus bordes. Sólo actúa sobre las líneas y los puntos, no sobre los polígonos. En el caso de los vértices, su representación se hace mediante círculos, convenientemente suavizados, en vez de mediante cuadrados.

Un efecto secundario del suavizado proporcionado por OpenGL, es que, en las líneas o puntos que se superponen, se aprecia un borde ligeramente más claro alrededor de cada punto o línea que oculta parte de los objetos situados detrás. El modo de evitarlo es desactivar el buffer de profundidad. Sin embargo, esa medida sólo es apropiada para escenas planas, en las que ningún objeto se halla por delante de otro. La desactivación del buffer de profundidad en las escenas tridimensionales impide la determinación de las caras ocultas.

El uso del suavizado de líneas y puntos supone un esfuerzo computacional que amplía considerablemente el tiempo de representación, a menos de que se disponga de soporte por hardware a través de OpenGL.

Iluminación

Uno de los principales factores que determinan de manera decisiva la apariencia de un objeto es su iluminación. Hasta el punto de que un objeto no iluminado puede perder por completo su aspecto tridimensional. Los objetos representados en las ventanas de visualización están iluminados por una fuente de luz fija situada en la esquina superior derecha de la escena. Ese foco de luz ayuda en gran medida a dotar a las escenas de volumen. Aun así, la iluminación puede ser opcional por dos razones. Primero porque algunas de las escenas pueden ser planas, por lo que la iluminación no añade ninguna información útil, y segundo, porque puede complicar los cálculos realizados y disminuir con ello la velocidad de representación.

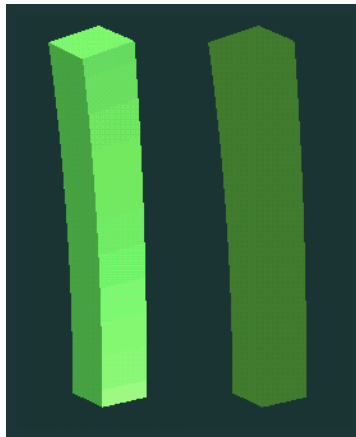


Figura 74: Visualización con la iluminación activada (izquierda) y desactivada (derecha).

A la iluminación de la escena también contribuye en alguna medida la luz ambiental, es decir aquella luz residual que proviene de multitud de fuentes pero sólo de forma dispersa. Esta luz no produce reflejos y al incidir en los objetos desde todas las direcciones no produce sensación de volumen. Así como la fuente de luz principal es de color blanco, la luz ambiental puede ser del mismo color que el fondo de la vista, para mantener cierta coherencia entre ambos colores. Pero esa similitud conlleva, generalmente, perder el verdadero color de los objetos, ya que se ven iluminados por luces coloreadas. Por tanto, para no falsear los colores originales de los objetos, lo idóneo es emplear una luz ambiental sin color, con un tono grisáceo.

Sombreado

Las superficies o líneas se pueden colorear de dos maneras distintas. Con el sombreado plano cada cara o segmento se representa con su color. Con el sombreado suavizado se interpolan los colores de los puntos interiores utilizando los colores de los vértices en los extremos de la cara o del segmento.

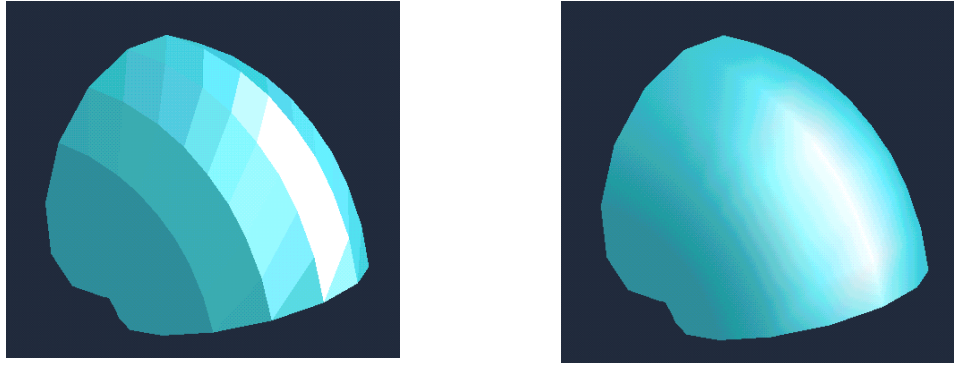


Figura 75: Superficie representada con sombreado plano (izquierda) y suavizado (derecha).

El método de sombreado resulta especialmente adecuado en la representación de superficies curvas e iluminadas.

Lo que se suaviza es sólo el color y no los vértices, por eso el borde de las superficies curvas representadas con sombreado suavizado sigue siendo igual de brusco, o suave, que al usar sombreado plano.

Referencias visuales

En ocasiones, en particular con objetos tridimensionales, es útil representar la escena junto con algunas referencias visuales. Para ver claramente la situación del suelo, se puede representar un plano con un área restringida a la región de la escena ocupada. El plano es parcialmente transparente, con el fin de que no oculte detalles cuando se gira la escena y queda en primer plano, por delante de algunos o todos los objetos de la escena.

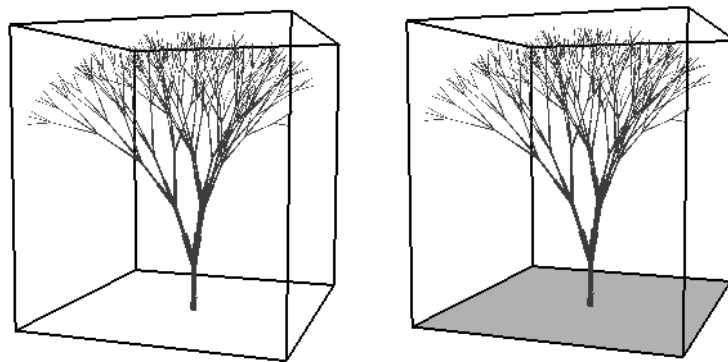


Figura 76: Representación de la escena con la opción 'Ver bounding box' activada. Se puede representar simultáneamente el suelo (derecha) para dar más referencias visuales.

Y para percibir claramente la región espacial que abarca la escena se pueden representar las aristas de la caja más pequeña que contiene a todos los objetos que forman parte de la escena. Con ambas opciones funcionando simultáneamente se eliminan ambigüedades que pueden aparecer.

Secuencias de crecimiento

Uno de los mecanismos clave y mejor aprovechados de los sistemas L es el mecanismo del crecimiento. Al fin y al cabo, las reglas de producción reflejan precisamente un proceso de transformación que se sucede a lo largo de sucesivos momentos en el tiempo. No sólo resulta interesante visualizar la etapa final del crecimiento sino que también tiene importancia la observación del proceso de desarrollo de la estructura en el tiempo. En ocasiones puede ser, de hecho, más importante abordar el carácter dinámico de la estructura que el estático.

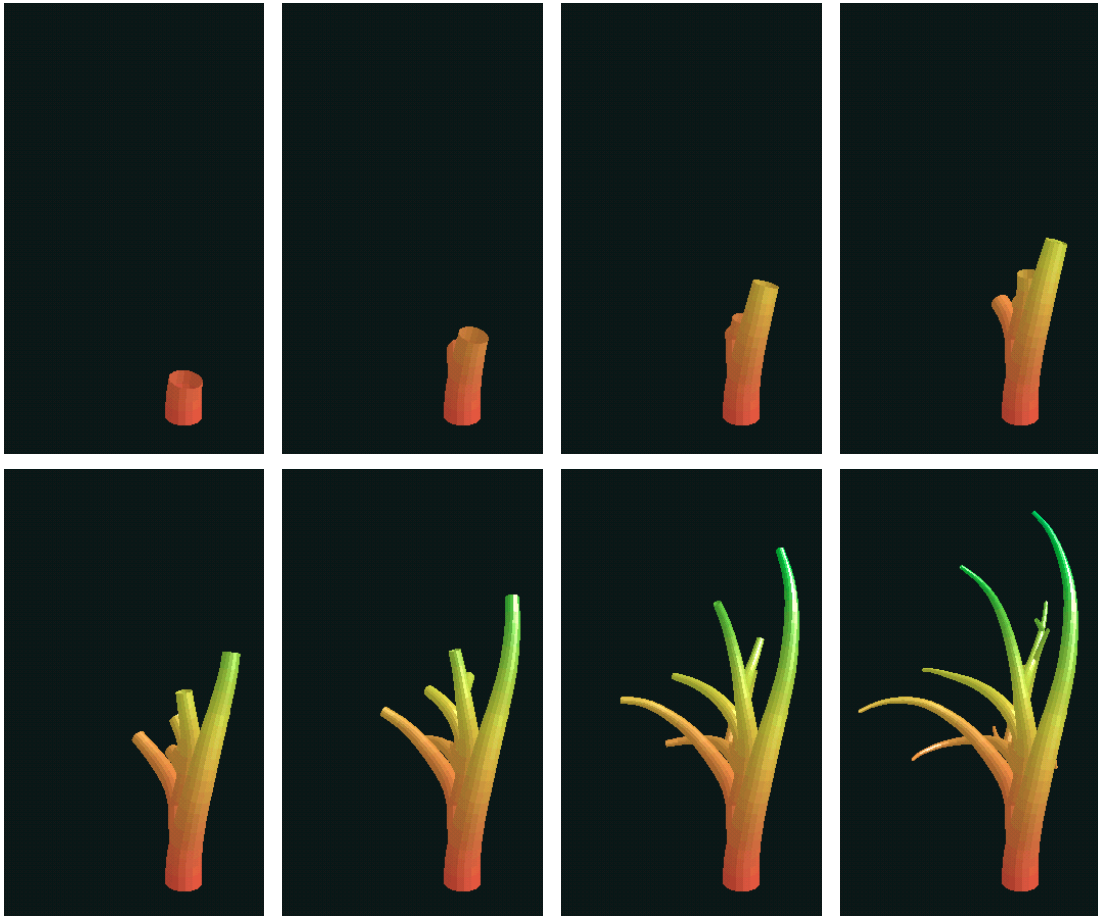


Figura 77: Secuencia de crecimiento de una estructura ramificada capturada en las etapas 5, 10, 15, 20, 25, 30, 35 y 40 de su desarrollo.

Con LS-SketchBook se puede observar la secuencia de crecimiento de una estructura. Al término de cada etapa de derivación la cadena de módulos se interpreta y se visualiza. El proceso prosigue con las siguientes etapas de derivación hasta alcanzar la última. El intervalo de tiempo transcurrido entre dos instantáneas del desarrollo de la estructura depende del tamaño de la cadena de módulos correspondiente. De manera que las últimas etapas aparecen de forma más distanciada.

9.5 Formatos de archivos

Dos de los principales objetivos en la elección de los formatos de los archivos necesarios para guardar la diversa información de los sistemas L es la facilidad de uso y

la compatibilidad con otras aplicaciones. Por una parte, se han usado archivos de texto en vez de binarios, siempre que el tamaño de estos no creciera desmesuradamente, ya que resulta fácil abrirlos con cualquier editor de texto. Tales el caso de la definición de los sistemas L, las cadenas de módulos y las tablas de colores. En el caso de las tablas de colores se ha añadido además otro formato binario. Su cometido es suplir de alguna manera la falta de algunas funciones que no tiene LS-SketchBook, como son las necesarias para crear de forma rápida y precisa tablas de colores.

Finalmente, exportar la geometría producida por los sistemas L es fundamental, con vistas a poder incorporarlas en otras aplicaciones diversas, generalmente aquellas que ofrecen un *render* más potente, y donde es posible integrarlas en otros modelos o escenas creadas con otros medios. De la misma manera, guardar imágenes de las ventanas de visualización también puede resultar útil si no se necesita más calidad en el *render* de los objetos producidos.

Sistema L (DSL)

Los archivos DSL contienen la definición del sistema L. Son archivos de texto sin formato editables con cualquier editor de texto de Windows. También se pueden guardar con extensión TXT en vez de DSL sin que cambie en nada su contenido, que es el mismo que se ve en el editor de sistemas L.

Por ejemplo:

```
SistemaL MonopodialPorHonda;

// estructuras arbóreas monopódicas de Honda

define r1 = 0.9; // tasa de contracción en el tronco
define r2= 0.6; // tasa de contracción en las ramas
define a0= 45; // ángulo de las ramas con el tronco
define a2 = 45; // ángulo de ramificación en las ramas
define wr = 1/sqrt(2); // tasa de reducción del grosor según da vinci
define d = FibAngle; // ángulo de divergencia

noterminales = {A,B,C};
axioma = A(1,10);
derivaciones = 6;

producciones = {
  p1: A(1,w) : true -> !(w)F(1)[&(a0)B(1*r2,w*wr)]/(d)A(1*r1,w*wr);
  p2: B(1,w) : true -> !(w)F(1)[- (a2)$C(1*r2,w*wr)]C(1*r1,w*wr);
  p3: C(1,w) : true -> !(w)F(1)[+ (a2)$B(1*r2,w*wr)]B(1*r1,w*wr);
};
```

La estructura de la definición se ajusta a las reglas gramaticales descritas en el capítulo dedicado a la sintaxis de la definición de sistemas L.

Plantilla de sistema L (PSL)

Cada vez que se crea un nuevo sistema L el editor de sistemas L abre el archivo *plantilla.psl* y lo utiliza como base para escribir el nuevo sistema L a partir él en vez de crear un nuevo archivo completamente en blanco. Cualquier sistema L se puede guardar con extensión *psl* de forma que sirva posteriormente como esquema para crear otros sistemas L. Sin embargo, para que el editor de sistemas L lo utilice habrá de guardarse con el nombre *plantilla.DSL*. Si no se encuentra la plantilla se comenzará con una definición en blanco.

La plantilla que se utiliza por defecto es la que contiene todas las opciones de definición de un sistema L, convenientemente agrupadas por función:

```
SistemaL ;

define = ;
define = ;
define = ;
define = ;

paso = ;
angulo = ;
ancholinea = ;
incdecancholinea = ;
indicecolor = ;
tablacolors = "";
elementobase = {
    forma = ;
    cubierta = ;
    nocaras = ;
    nobandas = ;
    tension = ;
    contorno = "";
};

semillaaleatoria = ;
modelocolor = ;

noterminales = {};
axioma = ;
derivaciones = ;

producciones = {
    p1 :: -> ;
    p2 :: -> ;
};
```

Al igual que los archivos DSL las plantillas de sistema L también son simples archivos de textos.

Cadenas de módulos (CAM)

Las cadenas de módulos se guardan en archivos de texto con extensión CAM. De manera que es posible leerlos o modificarlos con cualquier editor de texto. Por eso mismo se puede escoger guardarlas en archivos con extensión TXT.

El formato usado en los valores numéricos de los parámetros así como los separadores utilizados quedan fijados por las opciones seleccionadas para la presentación en el editor de cadenas de módulos. Realmente, la cadena se guarda con la misma apariencia que toma en el editor, salvo por el ajuste de línea, que no se emplea en el archivo. Por ejemplo, una cadena guardada en un archivo CAM se vería en el bloc de notas, con la opción de ajuste de línea, de la siguiente manera:

```
!(1.732) F(221.800) /(45.000) !(1.732) F(50.000) [ &(18.950) F(50.000) A
] /(94.740) [ &(18.950) F(50.000) A ] /(132.630) [ &(18.950) F(50.000) A
```

Hay que tener en cuenta que, así como se usan las opciones de formato de presentación de la cadena relativas al espaciado y formato numérico, también se usa la opción que delimita el número de módulos de la cadena presentada, por lo que cadenas muy largas pueden guardarse de manera incompleta a menos que se cambie la opción correspondiente.

No es posible guardar cadenas si no se presentan en el editor.

Autocad DXF

DXF es probablemente el formato de intercambio de datos más extendido en aplicaciones CAD de ordenadores personales. Originalmente fue desarrollado por los creadores de AutoCAD y ha alcanzado una gran popularidad debido principalmente al gran número de estaciones AutoCAD existentes. La gran mayoría de sistemas de CAD, así como gran parte de aplicaciones de ilustración vectorial, pueden importar y exportar archivos DXF.

Un archivo DXF es un archivo de texto ASCII formado por cuatro secciones: cabecera, tablas, bloques y entidades. Cada sección agrupa un conjunto de valores, cada uno de los cuales ocupa dos líneas en el archivo: el código de grupo y el valor. El código de grupo es un entero entre 0 y 999 que indica el significado del valor que le sigue. Por ejemplo, un comentario, un índice de color, un vértice, etc. La presencia de todos los códigos de grupo no es obligada y son suficientes unos pocos para poder describir la geometría de una escena.

Los archivos DXF que produce LS-SketchBook solamente contienen una marca, como comentario, identificando la aplicación que ha producido el archivo, la sección de cabecera con los límites de la escena, la sección de bloques vacía y la sección de entidades donde se incluye la información geométrica de la escena exportada. Las líneas se exportan como primitivas de tipo LINE, los tubos que constan sólo de eje, sin contorno, como POLYLINE y todos los demás objetos se descomponen en triángulos y se exportan como 3DFACE. Por ejemplo:

```
999
DXF creado por LS-SketchBook 1.0b
0
SECTION
2
HEADER
9
$EXTMIN
10
-25.903774
20
-13.951898
30
0
9
$EXTMAX
10
2.8672676
20
14.951884
30
0
0
ENDSEC
0
SECTION
2
BLOCKS
0
ENDSEC
0
SECTION
2
ENTITIES
0
3DFACE
10
0.62602532
20
0.91569221
30
0
11
1.059038
```

```

21
1.1656922
31
0
12
1.1060255
22
0.08430779
32
0
13
1.1060255
23
0.08430779
33
0
...
0
ENDSEC
0
EOF

```

Como 3DFACE exige cuatro vértices y los objetos que se exportan son triángulos el cuarto vértice y el tercero son los mismos. Todos los objetos se exportan en la misma capa y no se incluye información sobre el color.

El tamaño resultante de los archivos DXF es muy grande, mayor incluso que sus equivalentes RAW. Como promedio, cada triángulo de la escena puede requerir unos 150 bytes. Lo cual lleva a un tamaño de 1.4 MB para una escena de 10000 triángulos y 28.6 MB para una de 200000 triángulos. Afortunadamente la información de los archivos DXF, al igual que los RAW, permiten factores de compresión muy elevados, del orden del 80%.

Líneas y triángulos (RAW)

La geometría de las escenas está completamente construida a partir de triángulos o líneas. Cuando se exporta la información de la escena en un archivo de formato RAW se escribe un triángulo, o línea, en cada línea del archivo. Es decir, 3 vértices, si se trata de un triángulo, o 2 si es una línea. Cada vértice tiene 3 coordenadas espaciales, escritas con una precisión de 8 dígitos. Por ejemplo, las siguientes cuatro líneas representan dos rectángulos, formado cada uno de ellos por dos triángulos adyacentes:

```

0.7532239 0.56802613 0 1.7493949 0.65545189 0 0.84064972 -0.4281449 0
1.7493949 0.65545189 0 0.84064972 -0.4281449 0 1.8368206 -0.34071913 0
-1.2020762 0.29156953 0 -1.995677 0.90000832 0 -0.59363735 1.0851703 0
-1.995677 0.90000832 0 -0.59363735 1.0851703 0 -1.387238 1.6936091 0

```

El archivo es de texto con lo que puede alcanzar tamaños realmente grandes con escenas muy complejas. Una escena formada por 10000 triángulos ocupa cerca de 800 KB y una de 200000 triángulos alrededor de 15.5 MB. Aproximadamente, 90 bytes por triángulo o 50 bytes por línea. Por el contrario, tiene la ventaja de que se puede leer fácilmente.

Imagen bitmap de Windows (BMP)

BMP es el formato estándar de Windows para conservar imágenes. A pesar de sus limitaciones, es uno de los formatos más ampliamente admitidos. Soporta imágenes desde 1 bit hasta 24 bits, con paletas de colores o no, y con posibilidad de usar compresión RLE si se emplean paletas de colores. No soporta transparencia.

Un archivo BMP se organiza en tres o cuatro secciones, según la cantidad de colores de la imagen: una cabecera con información del archivo, otra con información de la

imagen, una tabla de colores, sólo en caso de que la imagen sea indexada, y, finalmente, los datos de la imagen que serán, según se trate, índices de color a la tabla de colores o bien valores de los componentes RGB.

Los archivos BMP escritos por LS-SketchBook son siempre de 24 bits y sin compresión.

Definición de contornos (CON)

La descripción de los contornos usados en la construcción de tubos se almacena en archivos de texto estructurados con extensión CON. Su formato es muy simple y fácil de modificar o crear con otra aplicación. Como ejemplo, gran parte de los archivos de contornos proporcionados han sido creados con Matlab, donde es bastante cómodo programar funciones para construir determinado tipo de figuras, como estrellas o polígonos regulares y, a continuación, exportarlos en a un archivo de texto con el formato requerido. Así, la descripción de una estrella de cuatro puntas tendría la siguiente forma:

```
poligonal
cerrado
0.0000000e+000 1.0000000e+000
2.8284271e-001 2.8284271e-001
1.0000000e+000 6.1230318e-017
2.8284271e-001 -2.8284271e-001
1.2246064e-016 -1.0000000e+000
-2.8284271e-001 -2.8284271e-001
-1.0000000e+000 -1.8369095e-016
-2.8284271e-001 2.8284271e-001
```

La primera línea distingue la clase del contorno. Un contorno poligonal interpreta los puntos del contorno como los vértices de una línea poligonal. Un contorno de tipo *spline* los toma como puntos de control a partir de los cuales construye la curva del contorno.

La segunda línea indica si el contorno es cerrado o abierto. En el primer caso, al trazar el contorno, el primer punto se unirá con el último formando una curva cerrada.

Finalmente, aparecen las coordenadas de cada uno de los puntos del contorno, dispuestos en dos columnas. La primera columna corresponde a la coordenada x y la segunda a la coordenada y . Las coordenadas se refieren al sistema de referencia local del contorno. En la construcción de los tubos el contorno se sitúa de tal manera que se hace coincidir el origen de su sistema de referencia con un punto perteneciente al eje del tubo.

Tabla de colores (TCL)

Los archivos con extensión TCL son archivos de texto, muy similares a los archivos ini de Windows. Cada uno de los 256 colores de la tabla de colores tiene una sección propia numerada con su número de orden en la tabla. Cada una de las secciones contiene cuatro campos con los valores del nombre y los componentes rojo, verde, azul y *alpha* del color. Por ejemplo:

```
[1]
nombre=celeste 1
red=0
green=255
blue=255
alpha=255

[2]
nombre=celeste 2
```

```
red=0
green=254
blue=255
alpha=255
```

...

```
[256]
nombre=celeste 256
red=255
green=0
blue=255
alpha=255
```

Los valores de los componentes son enteros entre 0 y 255. La componente *alpha* se usa para dotar de diferentes grados de transparencia a los objetos y también varía en el mismo rango.

Tabla de colores RGB (RGB)

Al igual que los archivos TCL, los archivos RGB también son archivos de texto. Sin embargo, su estructuración es muy distinta. Cada línea del archivo contiene los valores de los componentes RGB del color separados por espacios. La primera línea corresponde al primer color de la tabla, la segunda al segundo, y así sucesivamente. Las componentes se escriben como números reales con cuatro decimales significativos. No se incluyen ni el nombre ni la componente *alpha*.

La tabla RGB equivalente al ejemplo del apartado anterior sería:

```
0.0000    1.0000    1.0000
0.0000    0.9961    1.0000
...
1.0000    0.0000    1.0000
```

El formato RGB puede ser útil en los casos en los que la tabla de colores se crea mediante otra aplicación desde la cuál los datos de los colores se pueden exportar fácilmente con este formato, por ejemplo programas hechos en C, Fortran, Matlab, etc.

Tabla de colores de Adobe (ACT)

El formato de las tablas de colores con extensión ACT es muy simple. Tan sólo contienen los 768 bytes necesarios para guardar los componentes RGB de los 256 colores de la tabla. El orden que siguen, desde el primer color de la tabla es RGBRGB ..., es decir, un orden entrelazado.

La utilidad de este formato radica en que es el mismo que emplean algunas aplicaciones de Adobe, en particular Photoshop, para sus tablas de colores en imágenes indexadas. De esta manera resulta más sencillo crear una tabla de color basándose en los colores de alguna imagen ya existente. Por ejemplo, se puede tomar la fotografía de una planta, abrirla con Photoshop y reducir su número de colores a 256, o menos, los más representativos de la planta. A continuación se exporta la tabla de colores desde Photoshop y se incorpora en el modelo de sistema L. Conviene realizar la reducción de colores sin usar *dither* puesto que de esa manera se introducen colores extra para simular otros mediante tramados.



Figura 78: Fotografía original de la planta a modelar. Por efectos de la luz aparecen más tonalidades de cada color. No obstante puede ser práctico disponer de varias de ellas.

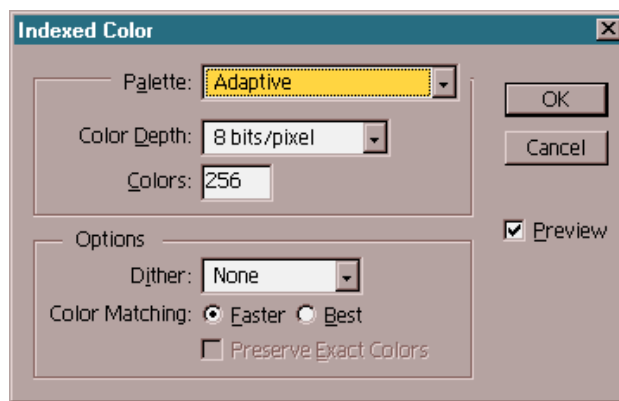


Figura 79: Cuadro de diálogo de Photoshop para convertir la imagen original de 24 bits a otra de 256 colores. En la conversión no se utiliza dither.

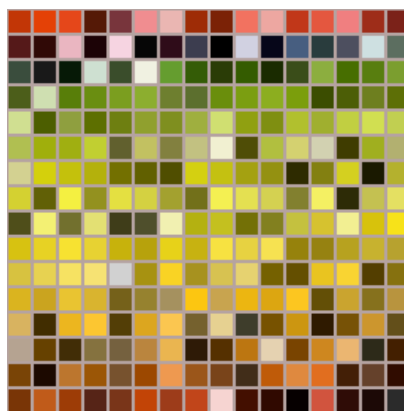


Figura 80: La tabla de colores obtenida que se puede guardar con formato ACT y utilizar en un sistema L.