A Calculation Method of Plant Similarity Giving Consideration to Different Plant Features

Wei-long Ding [1,2,*], Shui-sheng Wu[1], Nelson Max[2], Fu-li Wu[1,2], Li-feng Xu[1]

[1] (College of Computer Science & Technology, Zhejiang University of Technology, Hangzhou 310023, China)

[2] (Department of Computer Science, University of California, Davis, 95616, USA)

Abstract A method to compute the similarity between different plants is proposed, using features of a plant's topological structure and peripheral contour, as well as its geometry. The topological structures are described using tree graphs, and their similarity can be calculated based on the edit distance of these graphs. The peripheral contour of a plant is abstracted by its three-dimensional convex hull, which is projected in several directions. The similarity of the different projections is calculated by an algorithm to compute the similarity of two-dimensional shapes. The similarity of the geometrical detail is computed by considering the geometrical properties of different level branches. Finally the overall similarity between different plants is calculated by combining these different similarity measures. The validity of proposed method is evaluated by detailed experiments.

Keywords plant shape, similarity calculation, topological structure, contour feature, geometrical details

0 Introduction

Morphology is a basis for classifying plants into different types [1,2]. The similarities of the morphologies, structures, and habits between different plant species is related to their closeness or distance in phenotype relationships. Thus the discrimination between different plants is a critical step in the classification and retrieval of vegetation. Traditional discrimination methods rely mainly on manual operation, and thus may be subjective, labor intensive, and unsuited for rapid classification and retrieval of vegetation.

A plant-structure simulation model which can accurately describe the geometry and topology of a plant has significance in scientific research on evapotranspiration, the ideal plant type design of crops, and the optimization of cultivation measures [3]. However, in judging whether a model is precise or not, one must calculate the degree of similarity between the reconstructed 3D model and the real plant. Therefore, the definition and calculation of the similarity between different plants has important theoretical significance and practical value.

Compared to research on the similarity of three-dimensional models [4, 5], the comparison of DNA and protein sequences [6], and the similarity of malicious code [7], research on plant structure similarity is considerably weaker. Related research on plant structure similarity includes three aspects: 1) similarity of plant architectures [8-10]; 2) plant species identification based on blade similarity [11-14]; and 3) similarity of tree-structured data [15-17]. The methods to calculate the similarity between two plant architectures include global comparison methods, analytical comparison methods, and tree-graph-based comparison methods [8]. In global comparison methods, the similarity of two plant architectures is calculated by using parameters such as fruit production, stem diameter, and crown size. This method can roughly compare the similarity of global structure, but cannot compare in detail the plant topology and organ arrangement geometry. The analytical comparison methods first statistically analyze the topology and the spatial distributions of organs of a plant, and then use these features to compare the similarity between two plants. The tree-graph based comparison method employs edit

distance [15] to describe the similarity between two plants. As it requires complex mathematical calculation and frequent operations, this method is somewhat complicated. Some proposed methods to identify the type of species of plants are based on the similarity of the blade [11-14]. However, those studies only focus on the leaves, but neglect the similarity of plant topology. The study of similarity of tree-structured data [15-17] also pays attention only to the abstract tree graph, but there are significant differences with the real tree structures. In summary, plant structure similarity needs further study.

The objective of this paper is to present a method to compute the similarity between different plants which comprehensively considers plant topology, 2D projections of peripheral contour features, and inner details. The experimental results show that the proposed method can effectively calculate the similarity between different real plant architectures, and can distinguish different plant species, families, or genera based on their similarity.

1 Definition of the similarity of plant morphology

We consider the similarity of three aspects of plant morphology: 1) topology, which describe the structural relationship between various organs, 2) the peripheral outlines of a plant and the contour of each branch, and 3) the inner features, which describe the geometric characteristics, such as branching angles and diameters of the different organs. Assume that the similarities of plants in n different aspects have been calculated as the feature vector $s=(s_0, s_1, \ldots, s_{n-1})^T$ with each $s_i \in [0,1]$. Also assume some empirical weighting factors (in the range (0,1]) are assigned to each feature respectively, in the row vector $w=(w_0,w_1,\ldots,w_{n-1})$. Then the weighted average similarity is calculated as:

$$\overline{S} = \frac{\sum_{i=0}^{n-1} w_i s_i}{\sum_{i=0}^{n-1} w_i} \quad (1$$

Let $S_m = \max\{s_0, s_1, \ldots, s_{n-1}\}$. Then our formula for calculating the similarity of plants is:

$$S = b\overline{S} + (1-b)S_m \quad (2$$

where b is an empirical constant. Note that S and $\overline{S}$ are both in [0,1].

To calculate the similarity of plants, we consider seven features. The first two are the similarity S_t of topological structure and the similarity S_{3g} of peripheral outline. The other five are similarities of different inner features, which include: the average value of the angle between the branches and the stem (S_a), the diameter ratio of lateral branches to the stem (S_d), the width-to-height ratio of the peripheral outline (S_{wh}), the average value of the angle between second-order branches and first-order branches (S_{a1}), and the cross-sectional area ratio of lateral branches to the stem (S_{s1}).

2 Similarity of two plant topologies

2.1 The description of plant topology

The topological structure of a plant is determined by the relationships and distribution of internodes and nodes, usually represented by a tree graph [8,15]. A tree graph G is defined as a collection of vertices V and directed edges E, denoted as $G = \{V, E\}$. A vertex corresponds to a node. An edge corresponds to an internode connecting two nodes, and is represented by an ordered pair (v_i, v_j) (where v_i and v_j respectively represents the vertices). Edges are separated into two classes according to the geometry of the plant. An axial edge from v_i to v_j is one that continues in the direction from v_i's parent to v_i. Other edges are non-axial. We consider only trees with a single edge starting at the root, and

this edge is also axial. In contrast to the articles [8,15], in this article the vertices only represent the nodes, not including leaves, flowers, fruits, and other organs. Unless otherwise specified, a node in the following sections also indicates the vertex of a tree graph. Two types of edges between nodes are used to identify the different axes on a given plant: a precedent relationship (denoted by '<') and a branching relationship (denoted by '+') [8]. For example, in Figure 1, v_6 is a child of node v_1 and the edge (v_1, v_6) is axial, so their relationship is precedent, denoted as $v_1<v_6$; v_6 is a node of the main stem, and v_5 is a node of a branch, so the edge (v_5, v_6) is non-axial and their relationship is branching, denoted as v_6+v_5. Similarly, we have $v_6<v_7$, $v_7<v_2$, v_7+v_8. Extending the axial relationship across multiple internodes, we have $v_1<v_2$, $v_7<v_4$, and v_1+v_4.

$T[v]$ indicates the full sub-tree whose root is node v (the collection of nodes that includes node v and all its descendant nodes), and $|T|$ indicates the number of nodes of the tree graph T. If v is not the root and the edge to v from its parent is axial, then the sub-tree $T(v)$ is called an axial sub-tree, and if this edge is non-axial, then $T(v)$ is called a branch sub-tree. The number of internodes along the growth direction of the terminal buds of a branch sub-tree is called the depth of a branch sub-tree. For example, in Figure 2, the depths of branch sub-trees B, C , B', and C ' are 4, 3, 3, and 2 respectively.

2.2 Branch Degradation of Tree Graphs

In a simplified tree graph, a plant's topology is defined as the relation and distribution of the connections between the nodes. Therefore, we only consider the branching difference when comparing the similarity of plant topologies. To give a larger weight to the branching, we do branch degradation on the tree graphs. Branch degradation is performed by repeatedly removing any nodes v which have a parent node, and exactly one child node which is connected to v by an axial edge and connecting the parent of v to the child of v by a single edge. We call the resulting graph after branch degradation a branch degradation tree. Branch degradation can remove redundant nodes on sub-trees of a tree graph. For example, the nodes v_1, v_2, v_3, v_4, v_5 on the left of Figure 1 all satisfy the above conditions, so we remove all of them to get the branch degradation tree shown on the right of this figure.

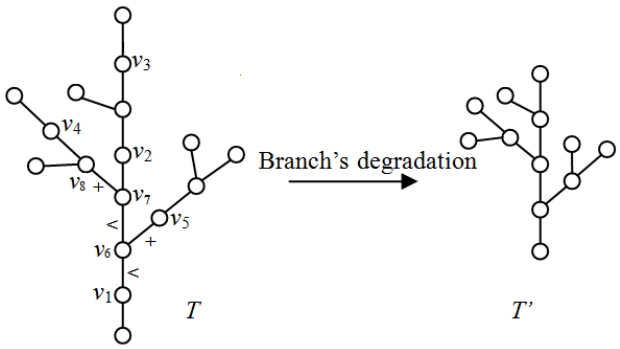


Figure 1 Branch's degradation of tree graphs

2.3 Edit operations between tree graphs, and mapping constraints

The measure of the topological similarity between two plants is determined by calculating the edit distance between the two corresponding tree graphs. The articles [8,15] define the distance between two tree graphs as the minimum effort needed to transform a source graph T_s into a target graph T_d using edit order S, which is a sequence of n insertion, deletion or substitution operations. In the process of transforming T_s into T_d, a technique named "mapping" is widely employed to characterize the effect of a sequence of edit operations on a tree graph. A mapping is intuitively a description of how the sequence transforms T_s into T_d, ignoring the order in which the edit

operations are applied [8, 22]. By definition, an edit mapping M is a collection of ordered pairs (v_i, v_j) where v_i is a node of T_s and v_j is a node of T_d.

There may be many mappings from T_s into T_d, with different costs. To get the minimum cost and preserve certain structural properties during the graph transformation, the article [8] came up with three constraints to restrict the operations for the edit mapping. For any node pairs (v_1,v_2), (w_1,w_2), and (u_1,u_2) in M, the three constraints are: 1) $v_1=w_1$ if and only if $v_2=w_2$; 2) $v_1<w_1$ if and only if $v_2<w_2$; 3) the least common ancestor of v_1 and w_1, which is denoted as lca(v_1,w_1), satisfies lca(v_1,w_1)<u_1 or lca(v_1,w_1)+u_1 if and only if lca(v_2,w_2)<u_2 or lca(v_2,w_2)+u_2. The first "equality" constraint establishes a one-to-one correspondence between a subset S_s of T_s and a subset S_d of T_d. Vertices of T_s not in S_s are the one that are deleted, and vertices of T_d not in S_d are the one that are inserted. The second constraint maintains the ancestor relationship when mapping. The third constraint maintains the correspondence between the branches of T_s and T_d, that is, any vertex in a branch sub-tree S_{sb} of T_s can only be mapped onto a branch sub-tree S_{db} of T_d.

In order to apply the graph edit distance to the comparison of real plant topologies, to consider all branching information contained in the tree description, and to limit and thus speed up the search for the best mapping, we propose three new additional constraints: the mapping constraint of branches, the mapping constraint for the depth of sub-tree internodes, and the mapping constraint of the number of sub-tree nodes. They can be described as follows:

A) **Mapping constraint of main trunk**

This constraint is to ensure that the main stems of the source tree and the target tree are aligned with each other. Assuming that $v \in T_d$, $w \in T_s$, $(v, w) \in M$, v is the root node of T_d, v_0 is a child node of v, the edge from v to v_0 is an axial edge; w is the root node of T_s, w_0 is a child node of w, the edge from w to w_0 is an axial edge; v and w have child nodes $\{v_0, v_1, v_2, \ldots, v_n\}$ and $\{w_0, w_1, w_2, \ldots, w_m\}$ respectively, and those nodes satisfy $\{v<v_0, v+v_1, v+v_2, \ldots, v+v_n, w<w_0, w+w_1, w+w_2, \ldots, w+w_m\}$. Let $T[v_0]$ be the axial sub-tree of node v and $T[w_0]$ be the axial sub-tree of node w, A mapping constraint of main truck is that any vertex in $T[v_0]$ of T_d can only be mapped onto $T[w_0]$ of T_s. For example, in Figure 2, A and A' represent the axial sub-trees of node v and node w respectively, B, C and B', C' represent the branch sub-trees of node v and node w respectively. According to this constraint, A can only be mapped onto A'. If A is mapped onto B' or C', this mapping will be an invalid mapping.

B) **Mapping constraint for the depth of branch sub-tree**

The second constraint ensures a sub-tree is mapped according to the depth of the sub-tree. Let $T[v_1],T[v_2],\ldots,T[v_n]$ be the branch sub-trees of T_s and $T[w_1]$, $T[w_2],\ldots,T[w_m]$ be the branch sub-trees of T_d. First we sort the lists of $T[v_i]$ and $T[w_j]$ respectively in descending order of their depth. For sub-trees whose depth are equal, we select the order arbitrarily. (This can cause the cost computed by formula (3) below to be greater than the true minimum transformation cost.) Given this ordering of sub-trees, the constraint requires the first sub-tree in the branch sub-trees ordering of T_s is mapped onto the first sub-tree in the branch sub-trees ordering of T_d, the second one in the branch sub-trees ordering of T_s is mapped onto the second one in the branch sub-trees ordering of T_d, and so on. For instance, in Figure 2, if the mapping between B and B' and mapping between C and C' are established, then this constraint is satisfied.

C) Mapping constraint of the number of sub-tree nodes

This constraint ensures the sub-tree is mapped according to the number of the sub-tree nodes. It is easy to understand the number of nodes in a sub-tree $T[v_i]$, which means how many nodes a sub-tree has. For example, in Figure 2 the number of nodes of B, C, B', C' are 7, 8, 11, 3 respectively. During a

sub-tree mapping, firstly we sort the sub-trees $T[v_1],T[v_2],\ldots,T[v_n]$ and $T[w_1]$, $T[w_2],\ldots,T[w_m]$ respectively in descending order of node number. Taking B, C, B' and C' as an instance, after sorting the new order of them is B', C, B and C'. According to this constraint, there should establish mappings between B' and C, and between B and C'. If two sub-trees are with the same depth, the order will be determined by this constraint.

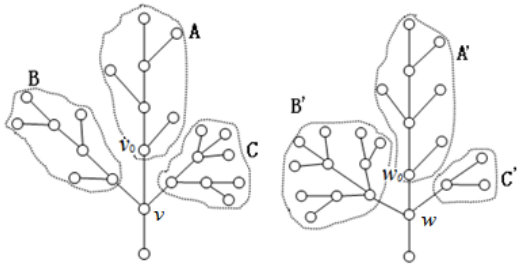


Figure 2 Three mapping constraints

2.4 Calculation of topological similarities

The edit distance between tree graphs is used to calculate the similarity of topologies in this article. For the two tree graphs, let one tree be the source tree T_s, and the other be the target tree T_d. After the branch degradation on both T_d and T_s, we perform certain insertions and deletions to transform T_s to T_d, according to a mapping M.

The total cost $D_t(T_s, T_d)$ of the transformation is defined recursively as follows:

$$D_t(T_s,T_d) = D_t(T[p(T_s)],T[p(T_d)]) + \sum_{i=1}^{\max\{|b(T_s)|,|b(T_d)|\}} D_t(T[b(T_s,i)],T[b(T_d,i)]) \quad (3$$

In this equation, $p(T)$ represents all the axial successor nodes of the root node (excluding branching nodes). $b(T)$ represents all the branching nodes of the root node. $b(T,i)$ represents the i^{th} node in this set using the ordering specified in the previous section. Assume $b(T,i)=\emptyset$ when $i>|b(T)|$, and $D_t(\emptyset,T)=D_t(T,\emptyset)=|T|$. $|T|$ is the number of nodes in tree T. $D_t(\emptyset,T)$ represents the cost of transforming an empty tree to T (insertions), while $D_t(T,\emptyset)$ represents the cost of transforming the tree T to an empty tree (deletions). Note that only insertions and deletions add to the cost when they occur in the base cases of this recursion, when one of the two trees is empty. There is no cost for the substitution considered in [8] and [15], because we do not label the nodes.

After computing the cost recursively using equation (3), we can clamp the result into the region between 0 and 1 using linear transformations and thus get the similarity value of topological structure (ranging from 0 to 1 with 1 representing the maximum similarity). The transformation equation is defined as follows:

$$S_t(T_s,T_d) = 1 - \frac{D(T_s,T_d) + ||T_s| - |T_d||}{2\max(|T_s|,|T_d|)} \quad (4$$

The edit distance calculated using our extra mapping conditions A), B) and C) may not be the minimum edit distance, but close to the minimum. The first reason is we focus more on conserving the natural characters of a plant rather than transferring it into a pure mathematical graph problem. And the second reason is that we use a greedy algorithm to calculate the edit distance in our program. The greedy algorithm always gives the best choice in each step when solving a problem. The greedy algorithm may not be able to get global optimal solution for all problems, but for many questions on a wide range, it can produce an optimal solution or approximate the optimal solution. And our recursive method of computing the edit distance using equation (3) executes much more quickly than the algorithms in [8] and [15].

3 Calculation of similarities between outlines and between inner details

Since the 3D structure of a real plant is very complicated, we used the 3D convex hull to represent the plant outline. To calculate the similarity value between plant outlines, we compute the 3D convex hulls, get the 2D projections in multiple directions, calculate the similarities between each pair of 2D projections, and finally calculate the 3D similarity from the similarities of the 2D projections.

In addition to topological structures and 3D outlines, different kinds of plants also differ a lot in inner details, such as the angles of branching, the proportion between the diameters of branches and those of trunks, and so on. In order to improve the algorithm's accuracy, this article will also take these detailed characteristics into account when measuring the similarities.

3.1 Calculation of similarities between 2D shapes

A 2D shape is a connected area on a plane, represented by a polygonal outline. To calculate the similarity between two 2D shapes, we represent the vertices of the polygons as characteristic vectors $V=((x_0,y_0),(x_1,y_1),\ldots,(x_n,y_n))$. Then the source shape and target shape need to be standardized. In the following section, when we use 2D shape similarity for estimating 3D similarity of plants, we standardize a vertical orientation of the main trunk or stem. But we still need to account for possible differences from translation and scaling. Thus we do the following:

Step 1: calculate the minimum coverage circle [18] of the shape;

Step 2: do translation so that the center of this circle is located at the origin;

Step 3: do scaling so that the radius of this circle is 1 (unit circle).

Since the distance between the two vectors can be calculated easily when the number of vertices is the same, we resample each polygon with K new vertices, as follows.

Divide the unit minimum coverage circle into K equal parts, by taking a ray from the center at every $2\pi/K$ radians (see Figure 3). There can be no intersections, one intersection, or multiple intersections between the ray and the polygon. If there are no intersections, mark the intersection between the ray in the reverse direction and the polygon. If there is one intersection, mark it. If there are multiple intersections, mark the farthest one from the center. Then we have the coordinates of K marked points for both the source shape (x_{si}, y_{si}) and the target shape (x_{di}, y_{di}). For both shapes, we can get approximate polygon outlines, denoted by G_s and G_d respectively, by connecting the K points. Figure 3 depicts how we resample the polygons by equally spaced rays in 2D. To generalize this method to 3D, we can project the source 3D shape G_{3s} and target 3D shape G_{3d} into different directions to get theirs 2D projections at first, and then resample the polygons according to the method given above.

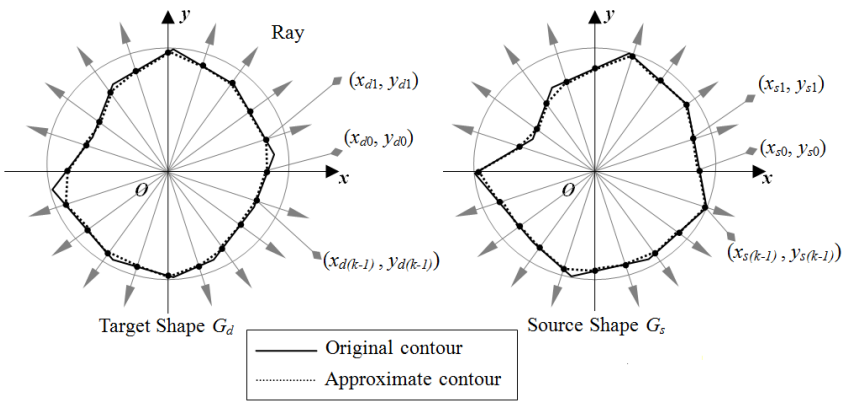


Figure 3 Approximation of 2D Shapes (K=20)

We can calculate the similarity of the resampled polygons by Euclidean Distance, using the

following equation:

$$S_g(G_s,G_d)=1-\frac{\min\left\{\sqrt{\sum_{i=0}^{K-1}[(x_{di}-x_{si})^2+(y_{di}-y_{si})^2]},\sqrt{\sum_{i=0}^{K-1}[(x'_{di}-x_{si})^2+(y'_{di}-y_{si})^2]}\right\}}{\sqrt{K}} \quad (5$$

where (x_{si}, y_{si}) is the coordinate of the i^{th} vertex in G_s, (x_{di},y_{di}) is the coordinate of the i^{th} vertex in G_d, $G_d^{'}$ denotes the shape we get by rotating 180 degree around the y-axis from G_d, and (x'_{di}, y'_{di}) is the coordinate of the i^{th} vertex in $G_d^{'}$.

3.2 Calculation of the similarities between the plant outlines

To describe the outline of a plant, we use a 3D convex hull that includes all the nodes in the tree graph [19]. Let H_d and H_s respectively represent the 3D shapes of the convex hulls of the target tree graph and the source tree graph. For the target H_d and the source H_s, we first project them into different directions and get the 2D projections. For example, in Figure 4, the dashed-line arrows mark the projection directions, and then we can get eight 2D projections for H_d and eight 2D projections for H_s. The projection planes of P_{s05}, P_{s06}, P_{s07}, and P_{d05}, P_{d06}, P_{d07} in Figure 4 are each represented by only a line segment in order not to hide the information of other projections. Then we calculate the 3D similarity based on the similarities of those 2D projections. The steps are:

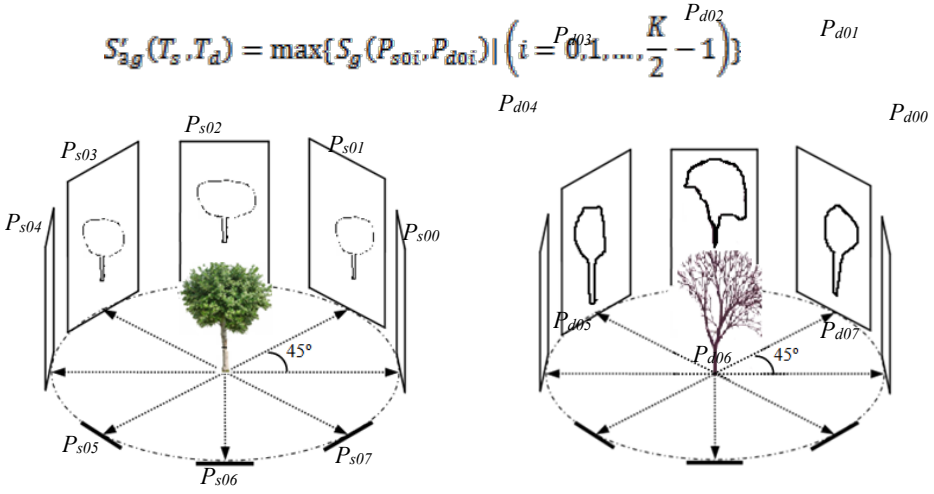
Step 1: Extract the vertices of the convex hull of shape H_d and H_s respectively.

Step 2: Let O_d, O_s be the center of the minimum coverage sphere [20] of H_d and H_s respectively, and make them coincide with the origin of the coordinate axes. Then scale the 3D shapes so that their minimum coverage spheres turn into unit spheres with radius equal to 1.

Step 3: Suppose P is the root point of a tree graph and O is the center of the minimum coverage sphere, determine whether the line segment PO is on the y-axis or not. If not, make them coincide with each other by a rotation about P.

Step 4: Let P_{d0i} be the projection of the target 3D convex hull onto the negative z-direction after rotating $2\pi i/K$ radians around the y-axis, and P_{s0i} be the projection of the source convex hull with the same process. Then the equation to calculate the similarity of contours $S'_{3g}(T_s,T_d)$ between T_s and T_d is:

$$S'_{3g}(T_s,T_d)=\max\{S_g(P_{s0i},P_{d0i})\mid\left(i=0,1,\ldots,\frac{K}{2}-1\right)\} \quad (6$$



1) The source H_s 2) The target H_d

Figure 4 The projections from 3D convex hull to many 2D shapes (K=8)

However, such similarity between 3D convex hulls is not enough to describe the similarity of

branches between T_s and T_d. Therefore, we also need to consider the similarities between non-completely degraded sub-trees at all levels. For a certain node v of a tree T, $T[v]$ denotes a complete sub-tree of T with v as the root. $T[v]$ is a non-completely degraded sub-tree if: (1) $T[v]$ does not degrade to a linked list; (2) $T[v]$ has more than two internodes; and (3) v is the root or has at least one sibling node. For example in Figure 5, the non-completely degraded sub-tree of level-0 is the whole tree $T[v_1]$; the non-completely degraded sub-trees of level-1 are $T[v_2]$, $T[v_3]$ and $T[v_4]$; and the non-completely degraded sub-trees of level-2 are $T[v_5]$ and $T[v_7]$.

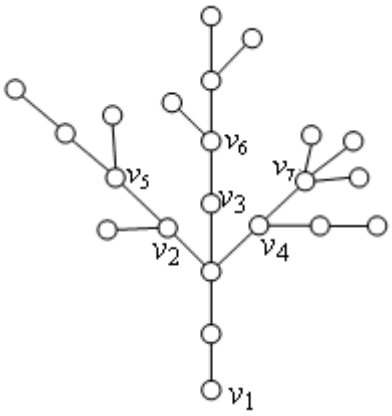


Figure 5 Non-degraded completely sub-tree

Assume T_d has non-completely degraded sub-trees of levels 0, 1, …, m, and T_s has non-completely degraded sub-trees of levels 0, 1, …, n. Let $MS(T,i)$ denote the non-completely degraded sub-tree that is the most similar with the tree T at i^{th} level. Then the final similarity between T_s and T_d can be calculated as:

$$S_{3g}(T_s,T_d)=\begin{cases}\dfrac{\sum_{i=0}^{\min(m,n)} S'_{3g}(MS(T_s,i),MS(T_d,i))}{\min(m,n)}, & \min(m,n)\neq 0\\ 0, & \min(m,n)=0\end{cases} \quad (7$$

3.3 Calculations of similarities between plant inner details

The inner detail of a plant can be described by geometric attributes of plant organs and topological connections between different organs. In this article, we consider the average branching angles on the main trunk, the ratios of diameters and cross-sectional area of main trunk to 1st level branches, the ratio of overall width to height, and the average branching angles of 2nd level branches on 1st level branches. Obviously, there must be some similarities between these parameters and proportions if two plants are similar. We use the following formulas to calculate the values of S_a, S_d, S_{wh}, S_{a1}, and S_{s1}.

For any positive numbers x, y, with $x \le y$, the similarity of these two parameters can be calculated as:

$$s(x,y)=\begin{cases}1-\dfrac{1}{2}\left(\dfrac{y}{x}-\dfrac{x}{y}\right), & 1\le\dfrac{y}{x}<\sqrt{2}+1\\ 0, & \dfrac{y}{x}\ge\sqrt{2}+1\end{cases} \quad (8)$$

For example, assume θ_s and θ_d are the average branching angle in radians of T_s and T_d, respectively, and then by using the equation (9) we have the similarity of average branching angle S_a:

$$S_a(T_s,T_d)=s(\theta_s,\theta_d) \quad (9)$$

The range of the result is also from 0 to 1 with 1 indicating the maximum similarity, i.e. exactly the same.

Similarly, we can calculate other similarities of plant details, including S_d, S_{wh}, S_{a1}, and S_{s1}. Then, considering also equations (4) and (7), we can use equations (1) and (2) to get the final weighted combination of these similarities between two plants.

4 Experiments

Based on the methods above, we implemented an experimental program on Windows XP with QT as GUI, and Visual C++ 2005 as IDE. The program is implemented in C++ with the OpenGL library. The experimental machine has 2GB RAM, an Intel Pentium D 2.80GHz CPU and a Nvidia GeForce 7300GT graphic card.

This section briefly illustrates the use of the proposed method in different application contexts. To show the effect of our algorithm as a means to calculate the similarity between different plant shapes, two examples have been selected: the first one illustrates that the proposed method can be used for theoretical plants; the second one illustrates that our method can be used for different species of trees in nature.

4.1 Similarities of simulated topological structures

We simulated 10 theoretical plants representing 10 different models which can be easily modeled as tree graphs, which are shown Figure 6. Each theoretical plant made up of ten elementary entities. The organization of the connections between their entities are different. We calculated the similarities between each pair using equation (4), and the result is shown in Table 1. The values range from 0 to 1 with 1 indicating exactly the same between two tree structures. From Table 1 we can see the similarities between T_0 and T_7, T_0 and T_9 are 0.18 which show that these two tree structures have maximal topological distance. The calculated results are reasonable because the similarity between unbranched plants and plants with many branches is very low. T_2 and T_5 have a high similarity because the difference between their structures is small.

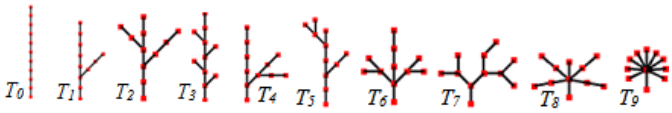


Figure 6 Simulated tree structure

Table 1 Simulated tree structure pairwise similarity

	T_0	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
T_0	1.00	0.5	0.33	0.20	0.33	0.25	0.22	0.18	0.28	0.18
T_1	0.5	1.0	0.75	0.50	0.75	0.60	0.54	0.30	0.66	0.46
T_2	0.33	0.75	1.00	0.71	0.60	0.83	0.46	0.27	0.54	0.40
T_3	0.20	0.50	0.71	1.00	0.42	0.62	0.35	0.21	0.40	0.31
T_4	0.33	0.75	0.60	0.42	1.00	0.50	0.76	0.40	0.54	0.40
T_5	0.25	0.60	0.83	0.62	0.50	1.00	0.40	0.23	0.46	0.35
T_6	0.22	0.54	0.46	0.35	0.76	0.40	1.00	0.55	0.57	0.44
T_7	0.18	0.30	0.27	0.21	0.40	0.23	0.55	1.00	0.37	0.30
T_8	0.28	0.66	0.54	0.40	0.54	0.46	0.57	0.37	1.00	0.75
T_9	0.18	0.46	0.40	0.31	0.40	0.35	0.44	0.30	0.75	1.00

4.2 Similarities of real plant structures

In this section, we apply our method to assess similarities in a sample of 5 species of trees: maple, camphor, osmanthus, gingko, and cherry, with 5 trees for each species. The length and diameter of the main trunk and the 1st and 2nd level branches, the spatial orientation of each branch, and the width and height of each tree have been measured using a tape measure, a digital caliper, and a digital protractor. The data were analyzed manually and the topologies of the trees in Figure 7 were obtained based on this data. We calculated the similarities between each pair of trees based on our method. Then we took the average of the results as the similarities between different kinds of plants, as shown in Table 2.

We can observe the following from the Table 2: 1) The similarities between trees of the same species are all above 0.81; 2) Similarities between plants with different branching methods are relatively low. The similarities between the monopodial branching plant Ginkgo and other sympodial branching plants are mostly lower than 0.5; 3) Similarities between plants with the same branching method are all above 0.72. Hence, it is reasonable to say that the method proposed in this article can differentiate different plants.

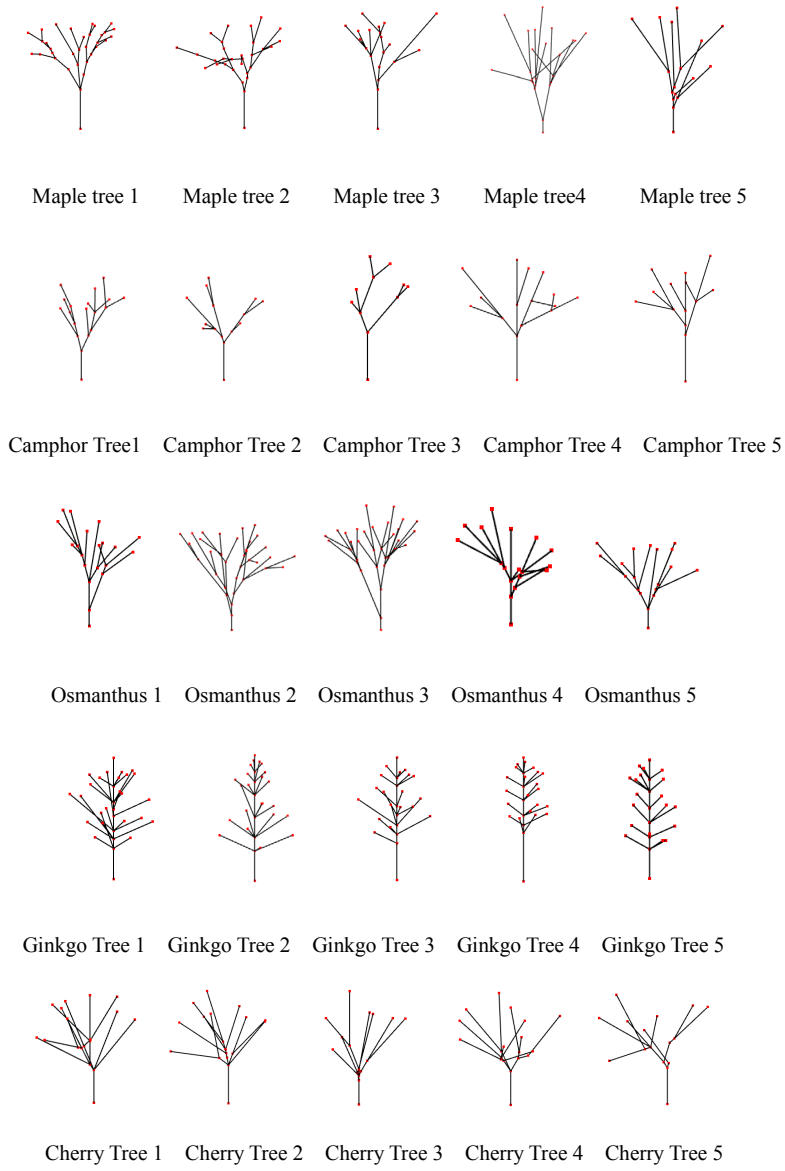


Figure 7 Five kinds of trees' topology

Table 2 Similarity between two different species

	Maple tree	Camphor tree	osmanthus trees	Ginkgo Tree	cherry tree
Maple tree	0.82	0.72	0.75	0.51	0.72
Camphor tree	0.72	0.82	0.74	0.59	0.75
osmanthus trees	0.75	0.74	0.81	0.53	0.75
Ginkgo Tree	0.51	0.59	0.53	0.91	0.55
cherry tree	0.72	0.75	0.75	0.55	0.83

4.3 Values of parameters

The constant K in equation (5) will affect the accuracy of describing a 2D contour. In order to determine the appropriate value of K, we designed an experiment on irregular tree contours (see Figure 8). We calculated the similarities S using the method proposed for the value of K from 0 to 120. From Figure 9 we can see that as K increases, S approaches a stable value and S barely changes after K = 30. Therefore we took K = 30 in this article. However, value of K may vary when comparing other kinds of shapes, such as cups, cars or jigsaws.

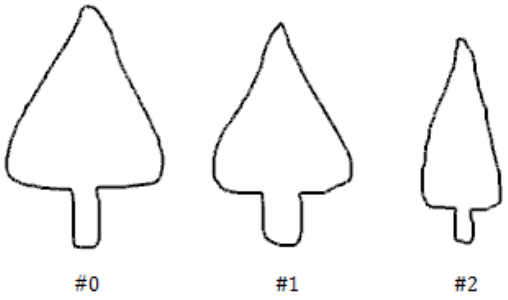


Figure 8 Plants peripheral irregular contours

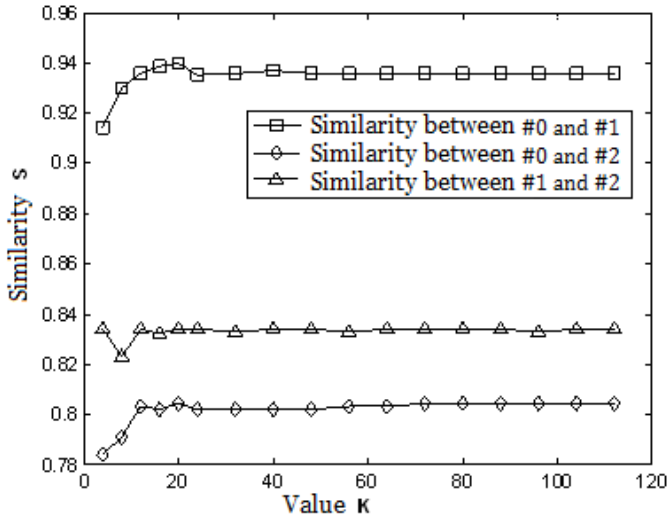


Figure 9 Relation between similarity S and constant K

As for the experience constants b, w_0, w_1, w_2, w_3, w_4, w_5, and w_6, we did experiments to determine their values. We choose the empirical constant b=0.698 after many experiments. Then we changed the other weights from 0.1 to 0.9 with interval 0.1, and tried all the possible combinations with equation (3). When choosing the weights, we wanted to maximize the similarities between the same kind of the

plants and minimize the similarities between different kinds of plants. We wrote a program to remove all the unlikely combinations and then chose manually from the 300 combinations left. The final chosen weights are: w_0=0.7, w_1=0.2, w_2=0.9, w_3=0.2, w_4=0.3, w_5=0.1, and w_6=0.2, for the seven similarities S_t, S_{3g}, S_a, S_d, S_{wh}, S_{a1}, and S_{s1} respectively .

4.4 Performance comparison with the algorithms of [8] and [15]

When comparing the similarity based on the edit distance of two tree graphs, articles [8, 15, 21] use mapping of nodes to describe a series of edit operations. In order to find the minimum cost mapping, which is an *NP*-complete problem, in polynomial time, they add three constraints, for example, the ancestor constraint and the independent sub tree constraint, to conserve the ancestor relation and sub-tree structure during the mapping. This article comes up with three new constraints: a mapping constraint of branches, a mapping constraint for the depth of sub-tree internodes and a mapping constraint of the number of sub-tree nodes. The first constraint concerns mapping of the main stems, while the second and the last constraint are used for mappings of sub-trees instead. With these six constraints, the method will perform more closely to the way that human beings commonly differentiate plants and also perform more quickly, although it may not get the minimum transformation cost.

Article [8] used a multi-level tree graph to represent the structure of a plant. The nodes included not only internodes, but also leaves, flowers, fruits and other organs. However, as a matter of fact, the structure is mainly determined by the connection properties of the internodes. So the method used in this article avoids the unnecessary cost for substituting different kinds of nodes. Article [8] didn't consider the characteristics of the peripheral contour and the inner detail of a plant. In contrast to our research, the study of article [15] focused on the problem of computing an edit based distance between abstract unordered labeled trees, and did not consider the geometry of real tree structures. Moreover, both [8] and [15] consider labeled graphs, so there is a cost associated to a label change from a "substitution". We do not use labels, so there is no need for substitution in this article.

5 Conclusions

This article proposes a method to compute the similarity between different plants. This method considers the similarities between plant topological structure, geometric shapes, and inner details, and gives a combined similarity value as a weighted average of these similarities. When calculating similarities between topological structures, we use a simplified tree graph to represent the topological structure of a plant and calculate the cost of transformation between two graphs using the edit distance method and branch degradation. When calculating similarities between geometric shapes, we propose an algorithm to calculate the similarities between two 2D polygons and extend it into 3D so that it can be used to measure the similarities between two plant outline shapes. When calculating the similarities between plants inner details, we consider geometric attributes of plant organs and the connection angles between different organs. The experimental results have proved that the method we proposed can calculate the similarities between different kinds of plants effectively.

For future work, we plan to apply this algorithm to a structure-based plant recognition software system. Moreover, we want to come up with a more effective computation method based on hardware acceleration. We are using the 3D convex hull when calculating the similarities. However, the convex hulls are just rough approximations of the plant outlines. We can use irregular 3D shapes to improve the accuracy of the approximation. Our 3D outline similarity method in section 3.2 assumes that the orientations of the two trees are similar after rotation to make the main trunk lie along the vertical y axis. However, a further rotation of one of the trees about y axis might make the 2D projections more similar,

and we can consider this in future. Last but not least, more details, such as leaf outlines, leaf veins, and bark texture, can also be taken into consideration.

Acknowledgements

This work was supported by the National Natural Science Foundations of China (31471416, 31301230) and Natural Science Foundations of Zhejiang Province (LY14C130015). The authors are grateful to the anonymous reviewers whose comments helped to improve this paper.

References

[1] Ma Yinxiao, Yao Min. Application of SVM in Plant Classification. Chinese Science Bulletin, 2007, 23(3): 404-407.

[2] Zheng Xiaodong, Wang Xiaojie, Gao Jie. Automatic Feature Extraction of Leaf Shape Designed for Angiosperm Taxonomy. Chinese Agricultural Science Bulletin, 2011, 27(15): 149-153.

[3] Guo Yan, Li, Bao Guo. New advances in virtual plant research. Chinese Science Bulletin, 2001, 46(1): 273－280.

[4] Schneider M, Behr T. Topological Relationships Between Complex Spatial Objects. ACM Trans. on Database Systems, 2006, 31(1): 39-81.

[5] Pan Xiang, Zhang Sanyuan, Ye Xiuzi. A Survey of Content-Based 3D Model Retrieval with Semantic Features. Chinese Journal of Computers, 2009, 32(6): 1069-1079.

[6] Peng Qunsheng, Hu Min. Approaches for 3D Protein Structure Similarity Comparison-a Survey. Journal of Computer-aided Design & Computer Graphics, 2006, 18 (10):1465-1471.

[7] Yang Yi, Su Purui, Ying Lingyun, et al. Dependency-based malware similarity comparison method. Journal of Software, 2011, 22(10): 2438-2453.

[8] Ferraro P, Godin C. A distance measure between plant architectures. Annals of Forest Science, 2000, 57(5-6):445-461.

[9] Godin C, Caraglio Y. A multiscale model of plant topological structures. Journal of Theoretical Biology, 1998, 191:1-46.

[10] Ferraro P, Godin C, Prusinkiewicz P. A structural method for assessing self-similarity in plants. In: Proc 4th International Workshop on Functional-Structural Plant Models, Montpellier, France, 2004:56–60.

[11] Nam Y, Hwang E, Kim D. A similarity-based leaf image retrieval scheme: Joining shape and venation features. Computer Vision and Image Understanding, 2008, 110(2):245-259.

[12] Caballero C, Aranda M C. Plant species identification using leaf image retrieval. In: Proc of ACM International Conference on Image and Video Retrieval (CIVR), 2010: 327-334.

[13] Du J X, Wang X F, Zhang G J. Leaf shape based plant species recognition. Applied Mathematics and Computation, 2007, 185(2): 883-893.

[14] Goeau H, Joly A, Selmi S, et al. Visual-based plant species identification from crowdsourced data. In: Proc of 19th ACM international conference on Multimedia, 2011: 813-814.

[15] Zhang K. A new editing based distance between unordered labeled trees. Combinatorial Pattern Matching, 1993,684:254-265.

[16] Kaizhong Zhang. Algorithms for the constrained editing distance between ordered labelled trees and related problems. Pattern Regonition, 1995, 28, 463-474.

[17] Rui Yang, Panos Kalnis, Anthony K. H. Tung. Similarity evaluation on tree-structured data. In: Proceedings of the 2005 ACM SIGMOD international conference on Management of data, 2005:754-765.

[18] Skyum S. A simple algorithm for computing the smallest enclosing circle. Information Processing Letters, 1991, 37(3):121-125.

[19] Thomas H.Cormen, Charles E.Leiserson, Ronal L.Rivest, et al. Introduction to Algorithm(Second Edition). Beijing: Higher Education Press, 2002.

[20] Emo Welzl. Smallest enclosing disks(balls and ellipsoids). Lecture Notes in Computer Science, 1991,555:359–370.

[21] Zhang K, Jiang T. Some MAX SNP-hard results concerning unordered labeled trees. Information Procecssing Letters, 1994, 49(5):249-254.