Cheng ZL, Zhang XP, Chen BQ. Simple reconstruction of tree branches from a single range image. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 22(6): 846~858 Nov. 2007

Simple Reconstruction of Tree Branches from a Single Range Image

Zhang-Lin Cheng[1,2] (程章林), Xiao-Peng Zhang[1,2] (张晓鹏), and Bao-Quan Chen[3] (陈宝权)

[1] *Sino-French Laboratory LIAMA, Institute of Automation, Chinese Academy of Sciences, Beijing 100080, China*

[2] *National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences Beijing 100080, China*

[3] *Digital Technology Center, University of Minnesota, Minneapolis, MN 55455, U.S.A.*

E-mail: {zlcheng, xpzhang}@nlpr.ia.ac.cn; baoquan@cs.umn.edu

Revised September 12, 2007.

Abstract 3D modeling of trees in real environments is a challenge in computer graphics and computer vision, since the geometric shape and topological structure of trees are more complex than conventional artificial objects. In this paper, we present a multi-process approach that is mainly performed in 2D space to faithfully construct a 3D model of the trunk and main branches of a real tree from a single range image. The range image is first segmented into patches by jump edge detection based on depth discontinuity. Coarse skeleton points and initial radii are then computed from the contour of each patch. Axis directions are estimated using cylinder fitting in the neighborhood of each coarse skeleton point. With the help of axis directions, skeleton nodes and corresponding radii are computed. Finally, these skeleton nodes are hierarchically connected, and improper radii are modified based on plant knowledge. 3D models generated from single range images of real trees demonstrate the effectiveness of our method. The main contributions of this paper are simple reconstruction by virtue of image storage order of single scan and skeleton computation based on axis directions.

Keywords cylinder fitting, generalized circular cylinder, skeleton, tree branch modeling

1 Introduction

Trees are indispensable objects in natural environment. 3D reconstruction and digitalization of a tree is very important in virtual environments. Botanically faithful tree models are important elements in landscape simulation, cityscape design, virtual reality, computer games, and other fields of 3D computer graphics. However, trees can be difficult to model due to their geometric complexity.

The research of tree modeling has a long history, and there are many approaches to construct natural-looking tree models. In the early days, 3D tree models are either constructed from some botanical and geometric rules[1,2], or designed with the help of an interactive editing interface[3]. These methods can produce a quite realistic impression of plants, but it is difficult to construct a 3D model of a given real tree. Recently some modeling techniques based on the digitization of a real tree have been developed[4,5], but they deal with trees with leaves, and put special emphasis on the overall impression, and produce a plausible tree skeleton even with the widespread occlusions introduced by leaves. In this paper we focus on reconstructing branching structures of a bare tree and generating a faithful branch model.

In fact, constructing a 3D model for trunk and branches is a key step of tree modeling, since branches determine the architecture of a tree and the shape of tree crown, which is important for visual realism. In botanic-faithful virtual tree modeling, branches and their topological structure are generated at first, and then other organs (e.g., leaves, flowers or fruits) are subsequently added.

Besides its importance in computer graphics, geometric modeling of tree branches is also significant in the application of forestry and ecosystems. The model of branches contains important morphological parameters, such as location and direction of a branch, tree height, diameter at breast height (DBH) etc. These parameters are important for estimating wood volume and shape of the crown for forestry management. Therefore, branch modeling is of ecological and economical interest for forest management[6,7] and timber industry[8,9].

To create a 3D branch model, the surface information of branches should be obtained at first. Apart from manual measurements with very simple tools,

Regular Paper

This work is supported by the National High Technology Development 863 Program of China under Grant Nos. 2006AA01Z301 and 2006AA10Z229, the National Natural Science Foundation of China under Grant Nos. 60674128, 60073007, and 60473110, and Beijing Municipal Natural Science Foundation under Grant No. 4062033.

currently two kinds of electrical instruments, digital cameras and range scanners, are used to acquire both the measurement and appearance information of a tree.

A number of photographs of a given tree can be captured from different overlapping views with a digital camera. Standard computer vision techniques[10−12] are applied to recover 3D points on the surface of the tree from these photographs. It is simple to acquire photographs using a camera, but the accuracy of recovered points is limited due to the insufficiency of camera resolution and errors generated in camera calibration for 3D reconstruction.

Range scanners are used to directly obtain surface points of a tree, and the accuracy of these points can be a few millimeters or even less. A single scan acquires one-side information of the tree, and the sample points are located regularly on the projection plane in the form of a range image. Multiple scans from different views are acquired if possible to obtain more surface information. After registration of multiple scans, a point cloud representation is obtained with points distributed on the tree branch surfaces.

In this paper we proposed a simple and efficient approach to constructing a 3D model of the trunk and branches of a real tree from a single range image. The source data for modeling is captured by a single scan of a bare tree using a laser scanner. Currently we assume a single scan as the basic input to our algorithm and this decision is based on the following reasons. First, a single scan maintains a regular distribution of points in the form of a range image, so conventional image processing techniques in 2D space can be applied, thus simplifying the modeling process. Multiple scans, once they are registered, become unorganized point clouds, therefore, no image structure can be leveraged. We think it is more worthwhile to perform tree modeling from individual scans and then combine all these individual models together, as indicated in our future work (Section 8). This way, we leverage both the image storage order of individual scans and the availability of multiple scans. On the other hand, multiple scans are not always obtainable. For example, a tree can be surrounded by other trees or buildings, and it is visible from only a peculiar viewpoint. Also in reality, a single scan can already capture most of the tree structure.

Trees have irregular branch shape and cross section area; like in other tree reconstruction work, we do not aim to recover the detail shape of a tree, especially from just a single range image. The incompleteness of the input data and the limited data accuracy and resolution, as well as abundant occlusions between branches, all prevent us from performing an exact reconstruction. In this work we generalize the representation of trunk and branches as circular cylinders. A generalized circular cylinder is defined as the volume of a circle orthogonal to and sweeping along a given 3D axis. It can be constructed by a series of skeleton nodes and their corresponding radii[11,13].

The remainder of the paper is organized as follows. After discussing previous work on tree modeling in Section 2, we provide an overview of our branch modeling pipeline in Section 3. Following the description of the efficient circular cylinder fitting method in Section 4, we present the theoretical base for the estimation of axis direction on each point using cylinder fitting in Section 5. Skeleton nodes and corresponding radii are computed at each cross-section, and then the cylinder model is constructed for the trunk and branches in Section 6. Experimental results are explained in Section 7. Conclusion and future work are finally presented in Section 8.

2 Related Work

Many approaches for creating a 3D model of a tree have been proposed in the past decades, and they can be roughly classified as three categories: botanical modeling, geometric modeling, and digitization of real trees.

Botanical Modeling. In the early days, people tried to simulate the growth process of natural plants. Trees were synthesized by some botanical rules or grammars. Among these rule-based methods, L-systems presented by A. Lindenmayer and P. Prusinkiewicz were widely used to describe the growth process of living organisms based on the guidance of fractal patterns[2,14].

AMAP model[1] was constructed based on bud life circles with botanical knowledge and some typical data measurement. This technique reflects the growing mechanism of plants, including space occupation and the location of leaves, flowers or fruits.

GreenLab models are presented[15] as a mathematical model that simulates interactions between plant structure and function. The model accurately reproduced the dynamics of plant growth, architecture and geometry of various annual and woody plants. As a result of internal competition for resources, the variability of leaf size and compensatory growth following pruning can be simulated also.

These approaches are not easy to use for a non-botanist because they have many parameters to control the architecture, and the input data (rules and parameters) are normally rather indirectly related to the output (3D geometry). The final shape and appearance of output model are difficult to control. So it is not appropriate to construct 3D models of real

trees using botanical modeling methods[5].

Geometric Modeling. Geometric modeling of trees is not strictly confined by botanical laws, but seeks to generate visually realistic shapes[16]. A 3D model of each branch is constructed with generalized circular cylinders from given 3D skeleton points in Bloomenthal's approach[13]. The trunk and side branches are connected through surface intersection then.

Generalized circular cylinder is a simplistic and popular way of modeling plant branch shapes through Frenet frames. Specifically, prism model is a simplified application of generalized cylinder for branches[13], and it is widely used in popular tree generators. The model of a maple tree in Bloomenthal's work has a deep impression of reality.

The combination of a rule-based approach with traditional geometric modeling techniques enables the efficient and effective generation of plants in Xfrog[3,16]. It allows easy generation of many branching objects including flowers, bushes, trees, and even non-botanical things. It is also a convenient tool for interactive plant modeling.

The approaches described in [17, 18] construct 3D models interactively using freehand sketches. All these approaches do not aim at 3D model reconstruction of real trees either.

Digitization of Real Trees. With recent advances in digital camera and laser scanning techniques, image-based and laser-scanning based methods have been developed to create 3D models of real trees in nature.

Shlyakhter *et al.*[19] constructed a 3D tree model from a set of photographs. In this approach the visual hull of a tree is produced from the segmented silhouettes at first, and then a plausible skeleton is constructed with the medial axis of the visual hull. To compute the branch thickness and to synthesize leaves, the L-system fit is applied with the skeleton as an axiom. More recently, Teng *et al.*[11] recovered 3D trunk model from 2D skeletons in two calibrated photographs. These methods generate only approximate skeletons and radii with limited accuracy.

Quan *et al.*[5] also used photographs for modeling, and they focused on the modeling of leaves, with branches interactively designed by the user. In these image-based methods, 3D information is computed from corresponding points in different images. The accuracy of reconstructed 3D points from 2D images is limited due to inevitable noise in input images and errors in estimated camera parameters. Therefore, image-based methods pay more attention to the visual impression than the faithfulness of the reconstructed model. With this thought, Neubert *et al.*[20] proposed a method to produce approximative models from input photographs using particle flows, which avoids the recovery of 3D point cloud.

With the help of range scanners, 3D shape information of trees can be acquired easily and accurately. Some methods have been proposed to modeling trees or extracting tree skeletons from the range scanner data. In the approach of Xu *et al.*[4], a skeleton is produced by connecting the centroid of points that have a similar length of the shortest path to a root point. The skeletons obtained in this way are dependent on the position of root points in each disconnected branches, and the automatic selection of root points is not easy, especially when there are too much occlusions. Additionally, as the skeleton nodes are sparse in this approach, the smoothness and fidelity of final skeletons should be improved. The corresponding radius of each skeleton node is estimated with the allometric theory, and thus is not directly derived from the scanned data. The very recent approach of Tan *et al.*[21] uses the image based methods to get a point cloud of trunk and branches, and applied the similar procedure as [4] to compute visible branches.

In the work of Gorte *et al.*[22] and Pfeifer *et al.*[9], tree skeletons are constructed with 3D mathematical morphology in 3D voxel space, and stem thickness in each skeleton point is computed with cylinder fitting. Their method requires multiple-scan to obtain dense point clouds. Many branches, especially those in the upper crown, are not reconstructed. Also the reconstructed branches are disconnected. The drawback of this approach is that the skeletonization is noise sensitive.

Previous work on modeling real trees all dealt with trees with foliage which disturb the actual reconstruction of branches. In contrast, we focus on branching structures and use a bare tree as input, which can be done by scanning in winter or pre-distinguishing branches and leaves based on color and position.

3 Overview

With a range image of a bare tree as input, the branch modeling method consists of two main parts: estimation of axis directions (Section 5) and construction of hierarchical skeletons (Section 6) as shown in Fig.1.

The estimation of axis directions can be divided into five steps. First, the range image of branches is segmented into several patches based on depth discontinuity (jump edges), so that each patch contains only structurally connected branches, which can be seen as a rough decomposition of different branches (Subsection 5.1). In each patch, mid-points between two boundary points in horizontal and vertical directions are calculated as two sets of coarse skeleton points.

These coarse skeletons are good approximation of actual skeletons and provide local shape information for further cylinder fitting (Subsection 5.2). Based on these coarse skeletons, branch patches are further decomposed into sub-patches at ramifications, so that each sub-patch only contains points from the same branch (Subsection 5.3). In the neighborhood of each coarse skeleton point, circular cylinder fitting is performed to compute primary axis directions (Subsection 5.4). After getting these primary axis directions, a robust axis voting technique is applied to obtain axis direction for each point (Subsection 5.5).

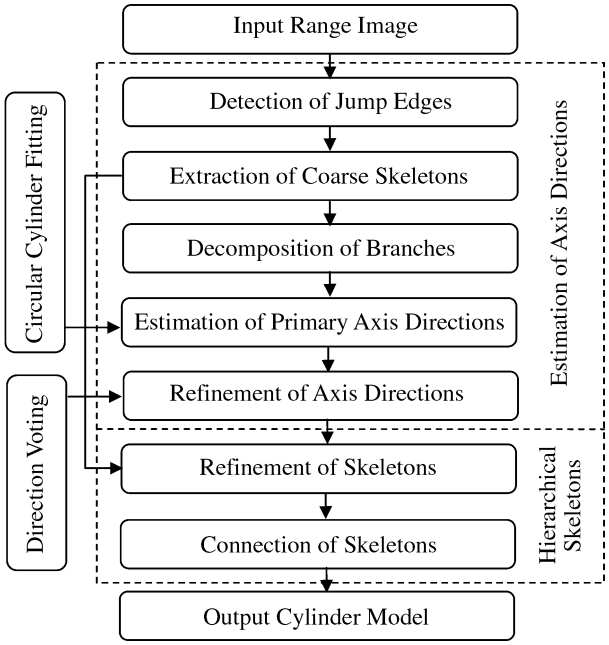

Fig.1. Pipeline of branch modeling.

The construction of hierarchical skeletons consists of two steps. Based on axis directions, invalid coarse skeleton points are rejected, which result in a coarse skeleton on the surface. Refined skeleton nodes that lie in the center of branches and corresponding radii are computed in the neighborhood of remaining coarse skeleton points (Subsection 6.1). Then all skeleton nodes are connected to form a hierarchical structure and improper radii are modified based on plant knowledge (Subsection 6.2). Finally, a cylinder model of the trunk and main branches are constructed using the hierarchical skeleton and corresponding radius at each skeleton node (Subsection 6.3).

Circular cylinder fitting is a theoretical base for the computation of axis directions. An effective cylinder fitting method is presented and explained in the next section.

4 Circular Cylinder Fitting

Circular cylinder fitting deals with minimizing the mean square distance from a set of points to a circular cylinder surface. There are mainly two types of methods for the least-square fitting of circular cylinder: algebraic method and geometric method, which minimize the algebraic and geometric distances respectively[23].

The algebraic method can be solved with linear techniques casted as an eigenvalue problem. However the solution may be very different from the optimum cylinder, especially in the case that fitted points cover only a small angle around the whole circumference. Geometric method is usually formulated as a five-parameter nonlinear optimization problem. It is solved with iterative techniques, where a proper estimation of the initial solution is very important to avoid local minimization.

Given n 3D points $\boldsymbol{p}_i$ $(1 \leqslant i \leqslant n)$, the objective function to be minimized is defined as

$$\mathcal{F} = \sum_{k=1}^{n} d_i^2, \tag{1}$$

where d_i is the Euclidean (geometric) distance of point $\boldsymbol{p}_i$ to the fitted cylinder surface. Circular cylinders have five degree of freedom, two for the direction of axis $\boldsymbol{a}$, two for the point $\boldsymbol{q}$ that the axis passes through, and one for the radius r. They can be parameterized without constraints by $\boldsymbol{s} = (\phi, \theta, \alpha, \rho, r)$ as follows[24]:

$$\boldsymbol{a} = (\cos\phi \sin\theta,\ \ \sin\phi \sin\theta,\ \ \cos\theta)^{\mathrm{T}},$$
$$\boldsymbol{q} = \rho\ (\boldsymbol{a}_x \cos\alpha + \boldsymbol{a}_y \sin\alpha),$$

where

$$\boldsymbol{a}_x = (-\sin\phi,\ \ \cos\phi,\ \ 0)^{\mathrm{T}},$$
$$\boldsymbol{a}_y = (\cos\phi\cos\theta,\ \ \sin\phi\cos\theta,\ \ -\sin\theta)^{\mathrm{T}}.$$

Then the geometric distance is computed with the following formula:

$$d_i = \|(\boldsymbol{p}_i - \boldsymbol{q}) \times \boldsymbol{a}\| - r. \tag{2}$$

Lukács *et al.*[24] found an approximate distance function $\tilde{d}_i$ that is faithful to the true distance function d_i:

$$\tilde{d}_i = d_i + \frac{d_i^2}{2r} = \frac{\|(\boldsymbol{p}_i - \boldsymbol{q}) \times \boldsymbol{a}\|^2 - r^2}{2r}. \tag{3}$$

Then the objective function (1) can be transformed to

$$\min \sum_{k=1}^{n} \tilde{d}_i^{\,2}. \tag{4}$$

To find the solution of (4), we use the Levenberg-Marquardt method[25]. Any such algorithm requires some good initial estimation of the solution. We give an effective method for estimating the initial values of $\boldsymbol{a}$, $\boldsymbol{q}$, and r in Subsection 5.4, which takes full advantage of the shape property of branches.

5 Estimation of Axis Directions

Since the branch is regarded as a generalized circular cylinder, every point on the branch surface can be assigned an axis direction, which is perpendicular to the corresponding cross-section. The first stage of our branch modeling system is computing the axis direction of each point. To do that, jump edges are detected at first to partition disconnected branches in the range image, and then coarse skeleton points and initial radii are computed from the contour of each branch. Finally, a small segment of the branch centered in each skeleton point is fitted with a circular cylinder to estimate the axis direction.

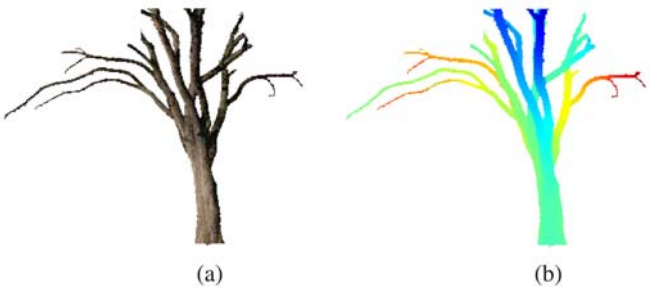

Fig.2. Branch images extracted from a single scan of a maple tree. (a) Texture image. (b) Range image.

Each pixel in the range image except background corresponds to a point in 3D space by the 3D-2D projection, and we regard them as the same throughout the paper for simplicity. Fig.2 shows a range image and the corresponding texture image extracted from a single scan of a real tree by discarding leaves and small branches. The point size (diameter) is 5 centimeters.

5.1 Jump Edge Detection

Depth discontinuity in a range image often means boundaries of different surfaces which are called jump edges, such as boundaries between the tree and background, boundaries between different branches induced by occlusion. Jump edges can be identified as points that have depth difference greater than a threshold, where depth difference of a point is defined by the maximum difference relative to its neighbors. However, this hard thresholding can cause undesirable artifacts, such as interrupted or dangling lines, because the range image may contain heavy noise, and adjacent points on two near branches may have small depth difference while points on a highly sloped surface have the big one. Hysteresis thresholding can alleviate these effects by using two thresholds $t_{\min}$ and $t_{\max}$ ($t_{\min} < t_{\max}$)[26]. Points with depth difference greater than $t_{\max}$ are regarded as *secure boundary points*. Conversely, points with difference lower than $t_{\min}$ are not boundary points. Points with difference between $t_{\min}$ and $t_{\max}$ are regarded as *potential boundary points* which are used to bridge the gaps between boundaries. These potential points are rejected if they are not connected to any secure boundary points by a path of potential boundary points. Fig.3(a) shows secure boundary points in black color and potential boundary points in gray color.

Point size is an effective index for the selection of jump thresholds. $t_{\min}$ and $t_{\max}$ are set as $C_{\min}$ and $C_{\max}$ times the point size respectively. For the tree in Fig.2, $C_{\min}$ and $C_{\max}$ are set as 2 and 4 respectively. The purpose of detecting jump edges is to separate geometrically disconnected branches, so jump edges are not necessary to be one-pixel wide.

After discarding the points on jump edges, branches are segmented into several patches, noted as $\{\boldsymbol{R}_k | k = 1, 2, \ldots, n\}$. Each patch $\boldsymbol{R}_k$ corresponds to a cluster of structurally connected branches. Fig.3(b) shows jump edges (in black) and segmented patches in different colors.

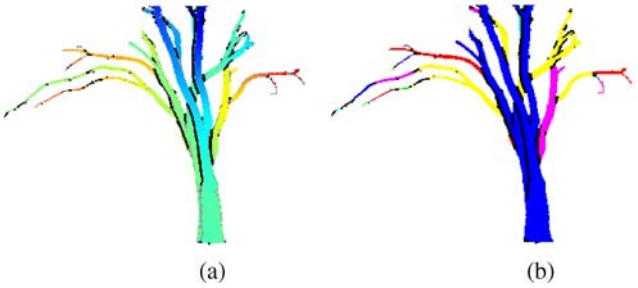

Fig.3. Detection of jump edges. (a) Boundary pixels. (b) Segmented patches.

5.2 Coarse Skeleton Extraction

For a tree-shaped object, the midpoint between two boundary points on a scanning line is a good approximation of skeleton point when the scanning line is approximately perpendicular with the axis direction of the object[27,28]. In contrast, midpoints with scanning line parallel to the axis direction are not skeleton points. A midpoint with the angle between scanning line and axis direction bigger than 45° is referred to as *valid skeleton point*, otherwise *invalid skeleton point*.

As shown in Fig.3(b), the trunk and branches are

partitioned into several patches after jump edge detection. We compute midpoints from the contour of each patch in horizontal and vertical scanning directions respectively, and at the same time obtain the distance between two corresponding boundary points.

Let $\{\boldsymbol{m}_i^{(k)} | i = 1, 2, \ldots, n^{(k)}\}$ represent the horizontal (vertical) midpoints in patch $\boldsymbol{R}_k$, where $n^{(k)}$ is the number of horizontal (vertical) scanning lines in patch $\boldsymbol{R}_k$, $d_i^{(k)}$ denotes the distance (number of pixels) between two boundary pixels in patch $\boldsymbol{R}_k$ that lie on the same horizontal (vertical) scanning line with $\boldsymbol{m}_i^{(k)}$. $d_i^{(k)}$ indicates the initial diameter of the cross-section that passes through midpoint $\boldsymbol{m}_i^{(k)}$.

If the horizontal (vertical) distance between midpoint $\boldsymbol{m}_i^{(k)}$ and its neighboring midpoint $\boldsymbol{m}_{i+1}^{(k)}$ (or $\boldsymbol{m}_{i-1}^{(k)}$) is less than a small integer Γ (Γ is set to 3 pixels by default), we say these two midpoints are *neighborhood connected*.

It can be observed that valid skeleton points are neighborhood connected, while invalid skeleton points are not or the number of neighborhood connected points is small. Additionally, midpoints $\boldsymbol{m}_i^{(k)}$ with very large diameter $d_i^{(k)}$ are invalid skeleton points. Based on these facts, we can discard some invalid skeleton points by connecting horizontal and vertical midpoints respectively. The remaining midpoints are served as coarse skeleton points. They may still contain some invalid skeleton points, but it does not matter because these coarse skeleton points are used as initial centers for cylinder fitting and invalid skeleton points are further rejected after obtaining axis directions.

Fig.4 shows the horizontal and vertical midpoints and extracted coarse skeleton points, where gray pixels show the segmented patches in Fig.3(b), blue pixels are extracted coarse skeleton points and red pixels are detected invalid skeleton points. It can be seen that horizontal (vertical) midpoints are good approximation of skeleton points in vertical (horizontal) parts of a branch.

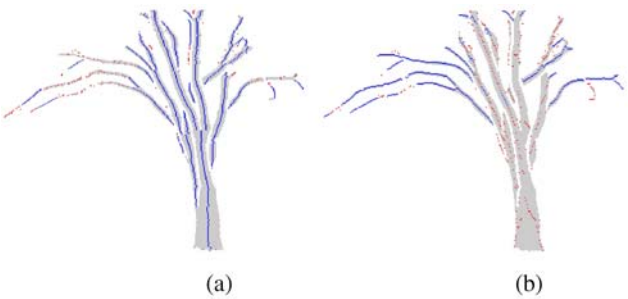

Fig.4. Extraction of coarse skeletons in horizontal and vertical directions. (a) Horizontal. (b) Vertical.

5.3 Branch Shape Decomposition

Nevertheless, some patches may contain ramifications, i.e., some neighboring points in the same patch lie in different branches, which can destroy the following cylinder fitting. It is better to segment the patches into sub-patches, so that every sub-patch contains points only from the same branch. Note that accurate partition of different branches from a range image is very difficult because there is often no clear boundaries between different branches. Here we do not aim at accurate partition because of the irregular shapes of a tree at ramifications. We found that the partition of branches just in horizontal or vertical direction at ramification points is a simple and effective solution.

Neighborhood connected midpoints belong to the same branch, while midpoints in different branches are not neighborhood connected. Based on this fact, a series of sub-patches can be obtained after extracting neighborhood connected horizontal and vertical skeleton points respectively in each patch $\boldsymbol{R}_k$. The sub-patch is a set of neighborhood connected midpoints and all other points lie on the same scanning line with any one of these midpoints.

Fig.5(a) shows the sub-patches of vertical branches and Fig.5(b) vertical partition of horizontal branches. Note that the decomposition of branches is performed using horizontal and vertical skeleton points respectively, and some branches that contain both horizontal and vertical skeleton points are partitioned into different horizontal and vertical sub-patches.

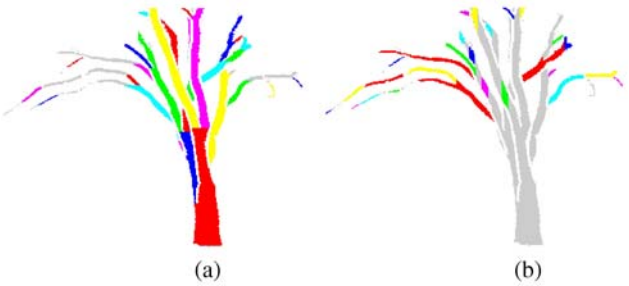

Fig.5. Segmentation of branch patches in horizontal direction and vertical directions based on coarse skeleton points. (a) Horizontal sub-patch. (b) Vertical sub-patch.

5.4 Primary Axis Direction Computation

After obtaining coarse skeleton points ($\boldsymbol{m}_i^{(k)}$) and initial diameters ($d_i^{(k)}$), cylinder fitting is performed in the neighborhood of each coarse skeleton point to compute the primary axis direction. The neighborhood $\mathbb{N}(\boldsymbol{m}_i^{(k)})$ is set as the intersection of a window mask ($N_i^{(k)} \times N_i^{(k)}$) centered in $\boldsymbol{m}_i^{(k)}$ and the sub-patch that

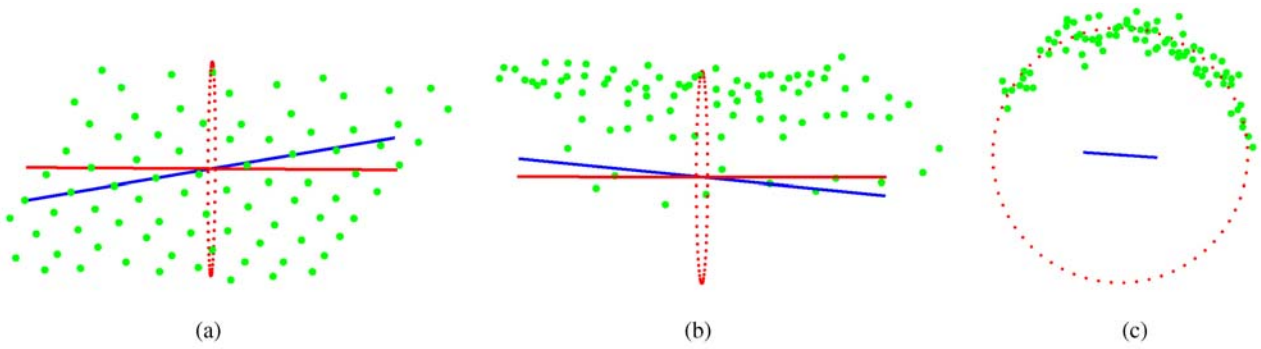

Fig.6. Cylinder Fitting to a point set (green points) from three views. (a) View from front. (b) View from top. (c) View from left.

$\boldsymbol{m}_i^{(k)}$ lies in. The sub-patch ensures that all or most fitted points belong to the same branch. If $m_i^{(k)}$ is a horizontal (vertical) skeleton point, the corresponding horizontal (vertical) sub-patch which it lies in is used. Note that if $m_i^{(k)}$ is both a horizontal and vertical skeleton point, two cylinders are fitted in the horizontal and vertical neighborhoods. The size of window mask $N_i^{(k)}$ is changed with the variety of diameter $d_i^{(k)}$ at each skeleton point. This adaptive neighborhood avoid cylinder fitting in a too small or too big section of a branch. If the scanning data contain heavy noise or the cross-sections of branches deviate highly from circles, the size of neighborhood is set to be big, relative to the diameter $d_i^{(k)}$, otherwise set to be small. In our experiments, $N_i^{(k)} = 2d_i^{(k)} + 1$, the length of neighborhood in the axis direction is larger than that in the direction of cross-section.

The method presented in Section 4 is applied to perform cylinder fitting. The initial estimations of solutions $\boldsymbol{a}, \boldsymbol{q}$, and r are computed as follows.

1) Since the size of neighborhood is larger than the diameter of cross-section, the direction of the first principal component of fitted points often represents the axis direction. The initial axis direction $\boldsymbol{a}$ is estimated by computing the eigenvector corresponding to the biggest eigenvalue.

2) The point $\boldsymbol{q}$ that the axis passes through is estimated by the centroid of fitted points.

3) The initial radius r for this fitting is set to $(d_i^{(k)} \times s)/2$, where s is the point size.

Fig.6 shows an example of axis directions computed using the cylinder fitting method. The blue line represents the initial axis direction estimated by the direction of the first principal component, and the red line is the resulting axis of cylinder fitting and red points cross-section. The sub-figures show three orthogonal views.

Fig.7(a) shows the primary axis directions of coarse skeleton points. These axis directions are projected to the range image plane for better visualization. In most cases, the angle between the initial estimated axis (blue arrowhead) and the resulting axis (red arrowhead) is small, which indicates that the direction of the first principal component is a good approximation of the axis direction and local optimization is avoided to some extent due to the fine initial estimation.

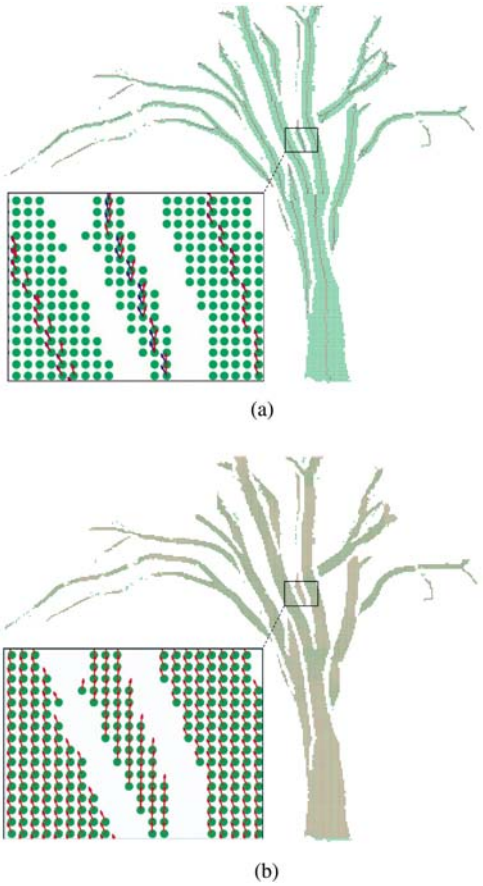

Fig.7. Axis direction refinement from the initial estimation to the refined axis direction. (a) Initial estimation of axis directions. (b) Refined axis direction.

5.5 Axis Direction Refinement

As mentioned earlier some points are both horizontal and vertical skeleton points, two circular cylinders are fitted in the horizontal and vertical neighborhood of each point, which generate two corresponding axis directions for the point. Additionally, each point $\boldsymbol{p}$ on the branch lies in the neighborhood $\mathbb{N}(\boldsymbol{m}_j^{(k)})$ of several coarse skeleton points $\boldsymbol{m}_j^{(k)}$, $(j = 1, 2, \ldots, n(\boldsymbol{p}))$, so several axis directions $\boldsymbol{a}_j^{(k)}$ are generated for point $\boldsymbol{p}$ after cylinder fitting.

To collect these axis directions, we refine them by smoothing, which is similar to image filtering in principle. This is performed by finding a consensus (voting) direction, i.e., finding a unit vector $\tilde{\boldsymbol{x}}$ that maximize the square sum of inner products between $\boldsymbol{x}$ and these axis directions $\boldsymbol{a}_j^{(k)}$:

$$\max_{\|\boldsymbol{x}\|=1} \sum \langle \boldsymbol{x}, \boldsymbol{a}_j^{(k)} \rangle^2. \tag{5}$$

This is a typical constrained optimization problem. It is characterized by the well-known necessary and sufficient optimality condition

$$\nabla S(\boldsymbol{x}) + \lambda \nabla C(\boldsymbol{x}) = 0,$$

where $S(\boldsymbol{x})$ is the objective function to be maximized, $C(\boldsymbol{x})$ the constrained function, and λ the *Lagrange multiplier*,

$$\begin{aligned} S(\boldsymbol{x}) &= \sum \langle \boldsymbol{x}, \boldsymbol{a}_j^{(k)} \rangle^2 = \boldsymbol{x}^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{x}, \\ C(\boldsymbol{x}) &= \boldsymbol{x}^{\mathrm{T}} \boldsymbol{x} - 1, \\ \boldsymbol{A} &= [\boldsymbol{a}_1^{(k)}, \boldsymbol{a}_2^{(k)}, \ldots, \boldsymbol{a}_n^{(k)}]^{\mathrm{T}}. \end{aligned} \tag{6}$$

The solution is solved by eigen-analysis of the matrix built from co-variances, i.e., $\tilde{\boldsymbol{x}}$ is the eigenvector corresponding to the biggest eigenvalue of the matrix $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$. This solution is used as a refined axis direction of point $\boldsymbol{p}$.

Fig.7(b) shows all refined axis directions in the range image plane. Fig.8 shows a local view of the refined axis directions in 3D around a ramification on the branch surface.

6 Construction of Cylinder Model

After axis directions have been estimated, skeleton node and corresponding radius of each cross-section can be computed. Ideally, a skeleton node is the center of a circular cross-section, and it is better to compute the skeleton node by cylinder fitting. However, several factors make the radius of fitted cylinder deviate from true dimension, and skeleton node deviate from actual center. First, the scanning data may cover only a small portion of the whole circumference due to occlusion or limited scanner resolution. Second, the cross-section of branches are not circular in most cases, especially the cross-section around a ramification. Third, the scanning data inevitably contain noise. As a solution, we use the centroid of each cross-section as a skeleton node, and compute the corresponding radius from the branch thickness $d_i^{(k)}$ in range image.

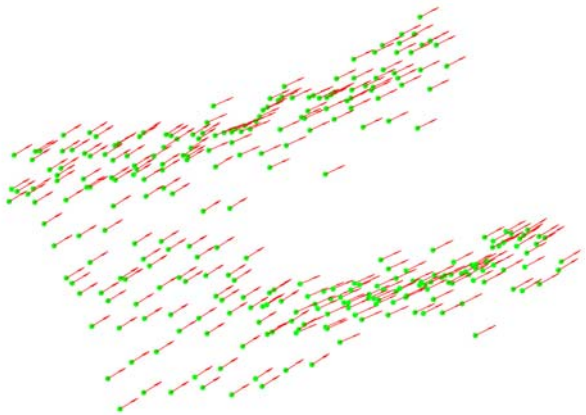

Fig.8. Refined axis directions on the branch surfaces around a ramification.

6.1 Computation of Skeleton Nodes and Radii

In order to construct a proper 3D skeleton, we take advantage of the horizontal and vertical coarse skeleton points. Some of these coarse skeleton points so far are invalid skeleton points. These invalid skeleton points, according to their definition, can be distinguished by the angle between scanning line and axis direction. If the angle is smaller than 45°, the skeleton point is rejected as invalid skeleton point. Fig.9 shows the remaining coarse skeleton points by rejecting invalid skeleton points in Figs.4(a) and 4(b), where blue points come from horizontal skeleton, and red points come from vertical skeleton.

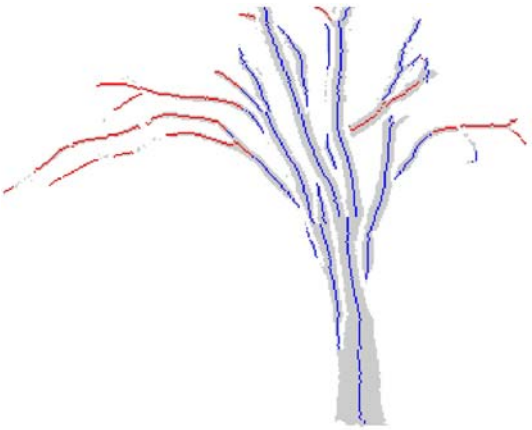

Fig.9. Coarse skeleton points.

Computation of Skeleton Nodes. The coarse skeleton points lie on the surface of branches, and they provide initial position for the centering skeleton nodes. Each coarse skeleton point determines a cross-section of a branch. The center of this cross-section can be computed by the weighted centroid of neighboring points, where the neighborhood is the same as that in cylinder fitting. The computed centroid may not be on the same cross-section with the coarse skeleton point because of the asymmetry neighborhood induced by irregular branch surface or occlusion, and it should be translated in the axis direction to the cross-section.

Let $\boldsymbol{m}^{(2)}$ be a coarse skeleton point, $\boldsymbol{a}$ be the axis direction of $\boldsymbol{m}^{(2)}$, $\boldsymbol{a}_i^{(k)}$ be axis directions of neighboring points $\boldsymbol{p}_i^{(k)}$, $\boldsymbol{c}$ be the weighted centroid of all points in $\mathbb{N}(\boldsymbol{m}^{(2)})$, then skeleton node $\boldsymbol{m}^{(3)}$ can be computed by the following equation

$$\boldsymbol{m}^{(3)} = \boldsymbol{c} - \langle \boldsymbol{c} - \boldsymbol{m}^{(2)}, \boldsymbol{a} \rangle \cdot \boldsymbol{a}, \tag{7}$$

where

$$\boldsymbol{c} = \frac{\sum \boldsymbol{p}_i^{(k)} w_i}{\sum w_i},$$

$$w_i = w\Big(\frac{|\langle \boldsymbol{p}_i^{(k)} - \boldsymbol{m}^{(2)}, \boldsymbol{a} \rangle|}{\delta_1}\Big) w\Big(\frac{|\langle \boldsymbol{a}_i^{(k)}, \boldsymbol{a} \rangle|}{\delta_2}\Big),$$

δ_1 is adapted as point size s times a constant (e.g., 5), and δ_2 is computed by the variance of $\langle \boldsymbol{a}_i^{(k)}, \boldsymbol{a} \rangle$.

In the computation of skeleton node, weights w_i are used to weaken the influence of points $\boldsymbol{p}_i$ that are far away from $\boldsymbol{m}^{(2)}$ in the axis direction $\boldsymbol{a}$. Points with axis direction largely different from $\boldsymbol{a}$ are also assigned with small weights. The weight function w is a smoothly vanishing, locally supported function. In particular we adopt Wendland's radial function (8) as in [29].

$$w(x) = (1-x)^4(1+4x). \tag{8}$$

Fig.10(a) shows the principle to calculate skeleton nodes in (7). Since the coarse skeleton points are acquired on every scanning line in the range image, which are dense with pixel accuracy, the computed skeleton nodes are also dense with high accuracy and can finely describe the shape of branches as shown in Fig.11.

Computation of Radii. As mentioned earlier, cylinder fitting provide radius information for every skeleton node but with some radii deviating from true dimension, and it is difficult to distinguish these improper radii, so here we compute the radius at each skeleton node with the branch thickness. Let d be the average thickness of branch at point $\boldsymbol{m}^{(2)}$ and its neighboring points, α the angle between the scanning line and the 2D axis direction of $\boldsymbol{a}$ in 2D image space. The branch radius r at point $\boldsymbol{m}^{(3)}$ is computed by the following equation:

$$r = \frac{d \times \sin\alpha}{2}. \tag{9}$$

Fig.10(b) shows the principle to calculate the radius of a skeleton node.

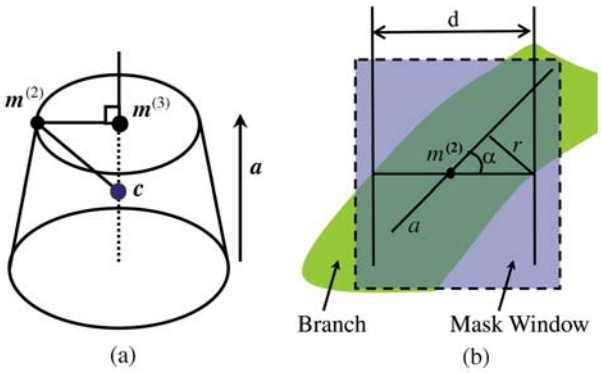

Fig.10. Computation of a skeleton point and its radius. (a) Skeleton points. (b) Radii.

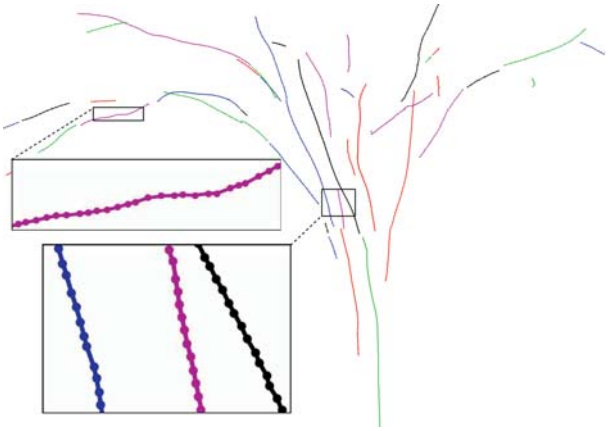

Fig.11. Skeleton segments.

6.2 Connection of Skeleton Nodes

In this step 3D skeleton nodes are connected with straight lines to produce a hierarchy of skeletons representing the shape of tree branches. This connection is performed in two steps: skeleton nodes corresponding to coarse skeleton points in the same sub-patch are connected first, which form a series of skeleton segments, and then these skeleton segments are connected to form a hierarchy of skeletons.

In the first step, since skeleton nodes are computed in the neighborhood of coarse skeleton points one by one and they lie in the same cross-section of a branch. These skeleton nodes can be connected in the same way as the connection of their corresponding coarse skeleton points in the range image.

After connecting skeleton nodes in each sub-patch, we get a series of disconnected skeleton segments as shown with different colors in Fig.11.

In the second step, each skeleton segment is connected to another segment. Let $\boldsymbol{p}$ be one end point of a skeleton segment, $\boldsymbol{a}$ be the axis direction at point $\boldsymbol{p}$, $\mathbb{M} = \{\boldsymbol{m}_i\}$ be a point set of neighbor skeleton nodes on other skeleton segments. A skeleton point $\boldsymbol{m}_j \in \mathbb{M}$ is connected to point $\boldsymbol{p}$ when the following conditions are satisfied:

1. the distance between point $\boldsymbol{m}_j$ and $\boldsymbol{p}$ is smaller than a distance threshold Δ;
2. the angle θ_i is smaller than a threshold Θ;
3. $\boldsymbol{m}_j$ has the smallest projection distance d_j to point $\boldsymbol{p}$ among all points in $\mathbb{M}$,

where

$$d_i = \langle \boldsymbol{m}_i - \boldsymbol{p}, \boldsymbol{a} \rangle,$$
$$\theta_i = \cos^{-1}\Big(\frac{d_i}{\|\boldsymbol{m}_i - \boldsymbol{p}\|}\Big).$$

The threshold Θ is set as 30° in our experiments. The distance threshold Δ depends on the occlusion between branches and is set by user. This way of connecting skeleton segments is similar to that of [4].

This connection is performed on every end point of each skeleton segment. If there exist many branches and large occlusion, this automatic connection may produce incorrect connection. So we designed an user interface based on Fig.9 to connect different skeleton segments, which is very convenient to perform in 2D space with the range image providing implicit 3D information. The connection result is a fully connected skeleton with the hierarchical structure of a tree as shown in Fig.12, where different colors represent different hierarchical levels.

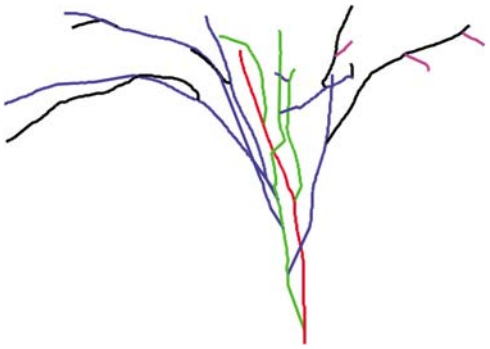

Fig.12. Skeleton hierarchy.

Since the radius at each skeleton node is computed by the branch thickness, the computed radii of partly hidden branches may be smaller than true dimensions. To alleviate these errors, the radii are refined from tip to root of the skeleton so that the radii of child skeleton nodes are not bigger than that of parent nodes.

6.3 Generation of Cylinder Model

Once the connected skeleton and corresponding radii are obtained, a cylinder model is constructed by generating a set of generalized cylinders with these skeleton and radii[11,13].

Given a list of skeleton points ($\boldsymbol{m}_i$) and radii (r_i), the procedure to construct the cylinder branch model is to find the discrete Frenet frames ($\boldsymbol{T}_i, \boldsymbol{N}_i, \boldsymbol{B}_i$) recursively from one end of the branch to the other end with as less torsion as possible at first. Then the generalized circular cylinder equation is piecewise computed as a spanned surface between two end circles.

The texture coordinates for each point on generalized circular cylinder are defined as the coordinates when it is isomorphically mapped to a planar square area.

7 Experiments and Discussions

In this section, we show reconstruction results from two range images. The first one is a range image (Fig.2) extracted from a single scan of a 20-meter tall maple tree using Riegl Inc' LMS-Z360i 3D image sensor. The leaves of this tree are discarded, and small branches are also rejected due to the difficulty in distinguishing them from leaves. Fig.12 shows the generated hierarchical skeleton with 19 branch segments by connecting 1399 skeleton nodes. The number of hierarchical levels of this example is 5.

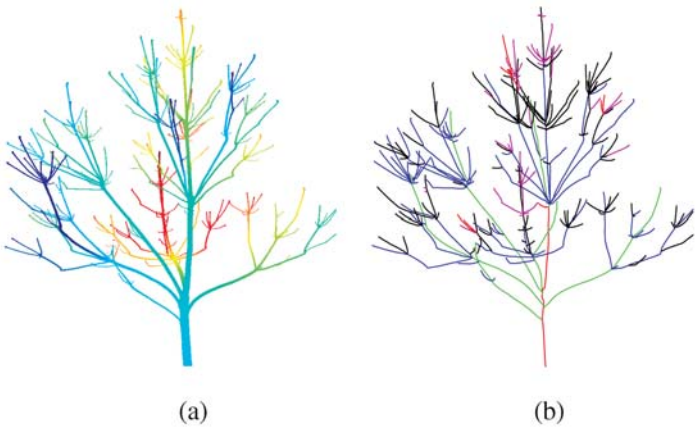

Fig.13. Sterculia tree model. (a) Range image. (b) Hierarchical skeletons.

Fig.13(a) shows the second range image generated by a single scan of a 6-meter tall sterculia tree without leaves in the early spring. Fig.13(b) shows the generated skeleton with 198 branch segments by connecting 11 550 skeleton nodes. The number of hierar-

chical levels of this example is 6. Note that some small branches in the middle of range image (Fig.13(a)) are discarded in skeleton connection due to heavy occlusions. Fig.13 illustrates that the generated skeleton is faithful to original tree due to the accurate and dense skeleton nodes.

The hierarchical connection for skeleton segments of the maple tree in Fig.12 is fully automatic. But for some skeleton segments of the sterculia tree, the automatic connection is not correct. We utilized the user interface to interactively modify the wrong connections in 2D image space with the range image for reference.

Fig.14 shows the constructed trunk and branches of the two trees, where Fig.14(a) and Fig.14(d) are shaded cylinder models. Bark textures are mapped on the cylinder models as shown in Fig.14(b) and Fig.14(e), where the self shadows are rendered in post-processing. Fig.14(c) and Fig.14(f) show the texture-mapped models from novel views.

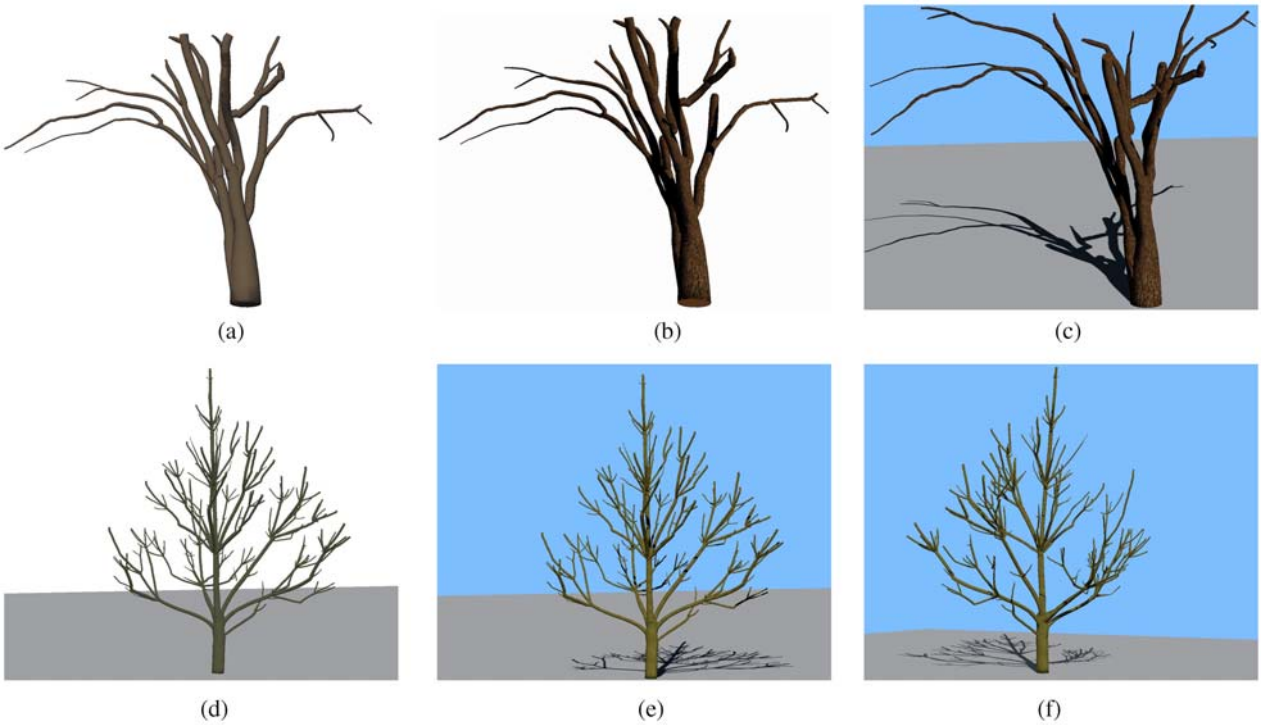

Fig.14. Rendering of branch models. (a) 3D maple model. (b) Textured maple model. (c) Another view of maple. (d) 3D sterculia model. (e) Textured sterculia model. (f) Another view of sterculia.

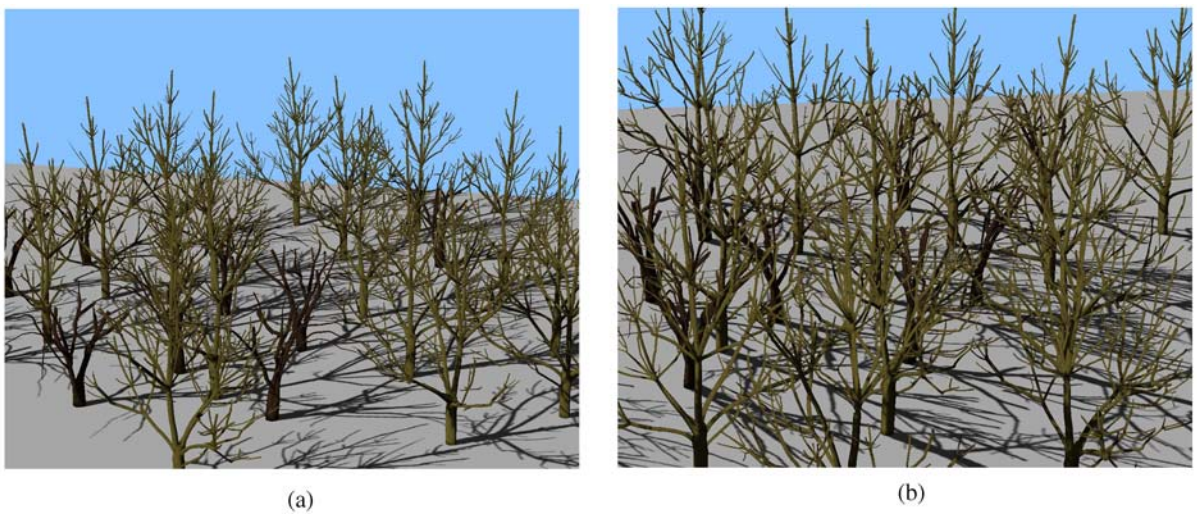

Fig.15. Virtual forest of defoliated maple and sterculia trees in winter. (a) Global view. (b) Another view.

To show the realism of the models acquired with our new technique, a virtual forest of defoliated trees is constructed in Fig.15 using these two cylinder models with random distribution and rotation on the ground. There are 26 samples in the forest with 9 maple trees and 17 sterculia trees. Fig.15(a) and Fig.15(b) show rendering results of this forest in two different views.

The rendering results in Figs.14 and 15 are synthesized using Phong local lighting model. Shadow effects are generated with z-buffer in post-processing using Reeves algorithm[30].

There are two limitations in this approach. First the skeleton segments are connected with straight lines, i.e., skeleton nodes for occluded branches are predicted in a very simple way. This may reduce the fidelity of generated model. Another restriction lies in the estimation of radius. Due to the incompleteness of input data and irregular cross-section area of branches, some radii computed by cylinder fitting highly deviate from true dimensions, and the thickness of branches in range images only provides rough information about the radius.

8 Conclusion and Future Work

We proposed a method to construct a cylinder model of the trunk and branches of a real tree from a single range image. The key idea is to estimate the axis direction at each point by cylinder fitting and to compute skeleton nodes based on these axis directions. Since generalized cylinder is a general primitive for plant modeling in computer graphics[11], our cylinder model of the trunk and branches can be used as input or measurement of a real tree for plant growth simulation[1].

The main advantage of this approach is the utilization of image storage order of single scan, which results in a simple reconstruction of branches mainly performed in 2D image space. Additionally, skeleton nodes are computed at each cross-section after getting axis directions, which results in dense skeleton nodes with high accuracy.

Due to the heavy occlusions among branches in a single scan, the prediction of hidden part is inherently difficult. In the future, we plan to construct occluded branches from a second or third auxiliary scan data, i.e., skeletons are reconstructed from individual scans and then they are registered and combined together. The mechanism of GreenLab plant growth model can be used to predict hidden parts of a branch in the crown. The result of this approach can be used as input of some plant growth modeling methods, e.g., GreenLab and L-system. The parameter estimation or parameter fit in GreenLab and L-systems can use the constructed cylinder model as an axiom to generate twigs and leaves as done by Shlyakhter[19].

Finally, this approach should be improved to cope with other trees with more complex crown, especially for the trees with more branches partially hidden by leaves or by other branches. Branch reconstruction should integrate with foliage construction, so that a full model of foliage tree can be reconstructed.

Acknowledgements We would like to thank all reviewers for their comments and instructive suggestions. We also thank Ms. Wei Ma for providing the scanning data of a sterculia tree, Dr. Marc Jaeger for his help in post-processing of the images in Figs.14 and 15, and Ms. Jia Liu for correcting some language mistakes in the manuscript.

References

[1] Phillippe de Reffye, Claude Edelin, Jean Françon *et al.* Plant models faithful to botanical structure and development. In *Proc. 15th Annual Conf. Computer Graphics and Interactive Techniques, SIGGRAPH '88*, New York, USA, 1988, ACM Press, pp.151~158.

[2] Prusinkiewicz P, Aristid Lindenmayer. The Algorithmic Beauty of Plants. New York: Springer-Verlag New York, Inc., USA, 1990.

[3] Oliver Deussen, Bernd Lintermann. A modelling method and user interface for creating plants. In *Proc. Conference on Graphics Interface'97*, Toronto, Ont., Canada, 1997, Canadian Information Processing Society, pp.189~197.

[4] Hui Xu, Nathan Gossett, Baoquan Chen. Knowledge-based modeling of laser-scanned trees. In *Proc. SIGGRAPH'05: ACM SIGGRAPH 2005 Sketches*, Los Angeles, California, USA, 2005, ACM Press, p.124.

[5] Long Quan, Ping Tan, Gang Zeng, Lu Yuan, Jingdong Wang, Sing Bing Kang. Image-based plant modeling. *ACM Trans. Graph.*, 2006, 25(3): 599~604.

[6] Aschoff T, Thies M, Spiecker H. Describing forest stands using terrestrial laser-scanning. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, XXXV(Part B5), ISPRS, 2004, pp.237~241.

[7] Bienert A, Maas H G, Scheller S. Analysis of the information content of terrestrial laserscanner point clouds for the automatic determination of forest inventory parameters. In *Proc. Workshop on 3D Remote Sensing in Forestry*, 2006.

[8] Thies M, Pfeifer N, Winterhalder D *et al.* Three-dimensional reconstruction of stems for assessment of taper, sweep and lean based on laser scanning of standing trees. *Scandinavian J. Forest Research,* 2004, 19(6): 571~581.

[9] Norbert Pfeifer, Ben Gorte, Daniel Winterhalder. Automatic reconstruction of single trees from terrestrial laser scanner data. In *Proc. ISPRS Conf.*, Instanbul, Turkey, Int. Archives of Photogrammetry and Remote Sensing, Vol. XXXV, B5, 2004, pp.114~119.

[10] David A Forsyth, Jean Ponce. Computer Vision: A Modern Approach. Prentice Hall, 2002.

[11] Chin-Hung Teng, Yung-Sheng Chen, Wen-Hsing Hsu. Constructing a 3D trunk model from two images. *Graph. Models*, 2007, 69(1): 33~56,

[12] Maxime Lhuillier, Long Quan. A quasi-dense approach to surface reconstruction from uncalibrated images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2005, 27(3): 418-~433.

[13] Jules Bloomenthal. Modeling the mighty maple. In *Proc. 12th Annual Conf. Computer Graphics and Interactive Techniques, SIGGRAPH'85*, 1985, ACM Press, pp.305∼311.

[14] Przemyslaw Prusinkiewicz, Mark James, Radomír Měch. Synthetic topiary. In *Proc. 21st Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH'94*, 1994, ACM Press, pp.351∼358.

[15] F Blaise, J F Barczi, M Jaeger, P Dinouard, P de Reffye. Simulation of the Growth of Plants — Modeling of Metamorphosis and Spatial Interactions in the Architecture and Development of Plants. Cyberworlds, Springer-Verlag, 2004, 1998, pp.81∼109.

[16] Bernd Lintermann, Oliver Deussen. Interactive modeling of plants. *IEEE Comput. Graph. Appl.*, 1999, 19(1): pp.56∼65.

[17] Makoto Okabe, Takeo Igarashi. 3D modeling of trees from freehand sketches. In *Proc. ACM SIGGRAPH 2003 Sketches & Applications, SIGGRAPH'03*, San Diego, California, USA, 2003, ACM Press, DVD-ROM.

[18] Makoto Okabe, Shigeru Owada, Takeo Igarashi. Interactive design of botanical trees using freehand sketches and example-based editing. *Computer Graphics Forum* (*Proc. of Eurographics'05*), 2005, 24(3): 487∼496.

[19] Ilya Shlyakhter, Max Rozenoer, Julie Dorsey *et al.* Reconstructing 3D tree models from instrumented photographs. *IEEE Comput. Graph. Appl.*, 2001, 21(3): pp.53–61.

[20] Boris Neubert, Thomas Franken, Oliver Deussen. Approximate image-based tree-modeling using particle flows. In *Proc. SIGGRAPH'07*, San Diego, CA, USA, 2007.

[21] Ping Tan, Gang Zeng, Jingdong Wang, Sing Bing Kang, Long Quan. Image-based tree modeling. In *Proc. SIGGRAPH'07*, San Diego, CA, USA, 2007.

[22] Ben Gorte, Norbert Pfeifer. Structuring laser-scanned trees using 3D mathematical morphology. In *Proc. ISPRS Conf.*, Instanbul, Turkey, Int. Archives of Photogrammetry and Remote Sensing, Vol. XXXV, B5, 2004, pp.929∼933.

[23] Vaughan Pratt. Direct least-squares fitting of algebraic surfaces. In *Proc. the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH'87*, 1987, ACM Press, pp.145∼152.

[24] Gabor Lukács, Ralph Martin, Dave Marshall. Faithful least-squares fitting of spheres, cylinders, cones and tori for reliable segmentation. In *Proc. the 5th European Conference on Computer Vision, ECCV'98*, Vol. I, London, UK, 1998, Springer-Verlag, pp.671∼686.

[25] William H Press, Saul A Teukolsky, William T Vetterling, Brian P Flannery. Numerical Recipes in C: The Art of Scientific Computing, New York: Cambridge University Press, NY, USA, 1992.

[26] Canny J. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 1986, 8(6): 679∼698.

[27] Jr D Dion, D Laurendeau, R Bergevin. Generalized cylinders extraction in a range image. In *Proc. the International Conference on Recent Advances in 3-D Digital Imaging and Modeling, NRC'97*, Washington DC, USA, 1997, IEEE Computer Society, pp.141∼147.

[28] Nevatia R, Binford T O. Description and recognition of curved objects. *Artificial Intelligence*, 1977, 8(1): 77∼98.

[29] Anders Adamson, Marc Alexa. Point-sampled cell complexes. *ACM Trans. Graph.*, 2006, 25(3): 671∼680.

[30] William T Reeves, David H Salesin, Robert L Cook. Rendering antialiased shadows with depth maps. In *Proc. SIGGRAPH'87*, 1987, ACM Press, pp.283∼291.

Zhang-Lin Cheng received the B.S. degree in automation from Changsha University of Electric Power in 2002. He is currently a Ph.D. candidate in Institute of Automation, Chinese Academy of Sciences. His research interests include point-based graphics and geometric modeling.

Xiao-Peng Zhang received his M.S. degree in mathematics from Northwest University in 1987, and the Ph.D. degree in computer science from Institute of Software, Chinese Academy of Sciences (CAS), in 1999. He is an associate professor in the Sino-French Laboratory LIAMA and National Laboratory of Pattern Recognition at Institute of Automation, CAS. His main research interest is computer graphics and pattern recognition. Dr. Zhang was invited as a foreign specialist for forest visualization in INRIA, the French National Institute for Research in Computer Science and Control from September 2002 to August 2004. He is also an associate professor of Graduate University of CAS, and the course he is teaching there is *computer graphics and application*. He received the National Scientific and Technological Progress Prize (Second Class) in 2004. His research work is supported by projects from the National Natural Science Foundation of China, National High-Tech Research and Development 863 Program of China, and the French National Research Agency.

Bao-Quan Chen received the M.S. degree in electronic engineering from Tsinghua University, in 1994, and a second M.S. degree (1997) and a Ph.D. degree (1999) both in computer science from the State University of New York at Stony Brook. He is an assistant professor of computer science and engineering at the University of Minnesota at Twin Cities, where he is also a member of the Digital Technology Center and Digital Design Consortium. His research interests generally lie in computer graphics and visualization, focusing specially on 3D data acquisition, illustrative rendering, visualization, and interactive techniques. His research is supported by the US National Science Foundation (NSF), Army Research, Microsoft Research, and a private donation. He is the recipient of the Microsoft Innovation Excellence Program 2002, the NSF CAREER award 2003, and McKnight Land-Grant Professorship of the University of Minnesota 2004∼2006. Dr. Chen has served, or is serving, on program and paper committees of several conferences in the field, most notably, IEEE Visualization (program co-chair 2004, general co-chair 2005/2006, 2007), and the Symposium on Point Based Graphics (2004/2005, papers co-chair 2006, program co-chair 2007). For more information, see http://www.cs.umn.edu/∼baoquan. He is a senior member of the IEEE.