Image-based Tree Modeling

Ping Tan, Gang Zeng, Jingdong Wang, Sing Bing Kang & Long Quan
Introduction

• Approach for generating 3D models of natural-looking trees

• Trees are hard to model because of their inherent geometric complexity

• Image based approach
Bushes vs. Trees

- **Bushes** – Terrestrial flora with large leaves (relative to bush size)
- **Trees** – Large terrestrial flora with small leaves (relative to tree size)
Features of a modeling tool

- Ease of model generation
- Model editability
- Realism
- All three are difficult to achieve simultaneously
Techniques for modeling of Trees

- Techniques can be classified into two
  - Rule based: Rules or grammars to create branches or leaves
  - Image based: Use images of tree for modeling
Rule based modeling

• Use generative L-system or geometric rules to produce realistic looking trees
• Some techniques even use rules of plant growth
• Disadvantages
  – Requires expertise for effective use
  – Based on the assumption that branch shape and leaf follow a particular pattern
  – Considerable effort is spent in modeling unexpected local modifications like stunted growth
Image based modeling

• Previous Works
  – Single Image, Multiple Images
  – Simple L-system fit for branch generation
  – Branching rules in voxel space

• Disadvantages
  – Cannot be edited or animated easily
Image based modeling

• Paper’s approach
  – Designed to reduce the user interaction by using image data as much as possible
  – Branch recovery is mostly automatic with user given some editability options
  – Imaged based does not mean that we model each and every leaf from the source images
Overview

- Three steps in modeling
  - Image capture and 3D point recovery
  - Branch recovery
    - Reconstruction of visible branches
    - Reconstruction of occluded branches
  - Leaf population
Overview

Source Images → Structure from motion → Reconstruction of visible branches → Reconstruction of occluded branches → Image Segmentation → Textured 3D model
Image Capture

• Use a hand-held camera

• Capture images of a tree from a number of different overlapping views

• 10 to 20 images with a coverage of about 120° to 200° around the tree is enough for almost every tree
3D point recovery

• Run structure of motion on the images to recover the camera parameters and the 3D point cloud
• We assume that the surface of the tree has been extracted in each image so we know that 3D point cloud is of the tree
• Depending on the spatial distribution of the camera and geometric complexity of the tree, significant areas of the tree may be missing
Branch recovery

- Two step process
  - Reconstruction of visible branches
  - Reconstruction of occluded branches
- Three assumptions in branch recovery
  - Cloud of 3D points has been partitioned into points belonging to the branches and leaves
  - The tree trunk and its branches are assumed to be unoccluded
  - Structures of visible branches are highly representative of those that are occluded
Reconstruction of visible branches

- Construct visible branches using the 3D point cloud
- 3D points would be in the form of multiple point clusters due to the occlusion of branches
- Each cluster is called a branch cluster
- Three steps
  - Graph construction
  - Extraction branch clusters
  - Determining branch geometry
Graph Construction

- Each 3D point represents vertex of a graph
- Compute distance between vertices
  \[ d(p, q) = (1 - \alpha)d_{3D} + \alpha d_{2D} \text{ with } \alpha = 0.5 \]
  \[ d_{2D} = \sum_i \frac{1}{n_i} \sum_j |\nabla I_i(x_{ij})|, \]
- 2D distance is set to infinity if line is projected outside branch area
Forming branch clusters

- Connect neighboring pixels if their distance is smaller than threshold value.
- Each connected component is considered as branch cluster.
- Process each branch cluster to compute branch geometry which includes computing skeleton node and branch radius.
Branch geometry

- Bottom-Up approach
- Start with cluster that contains lowest 3D point (root).
- Compute shortest path between root and all other points.
- Edges are kept only if they are part of shortest paths.
Branch geometry

- Divide edges into segments of predefined length.
- Compute centroid of points in each segment as skeleton node.
- Compute standard deviation of points in each segment as radius of skeleton node.
User Intervention

- Branch refinements
  - Adding or removing skeleton nodes
  - Adjusting radius of branch
  - Add or discard identified branches
- Connecting two branch clusters
Reconstruction of occluded branches

- Based on assumption that the branch structure is locally self-similar.
- Reconstructed branches are used to replicate occluded branches.
- Sub trees of visible branches are candidate replication blocks.
- If replication blocks cannot be constructed from visible branches then they are chosen from another similar tree.
Replication Blocks
Branch reconstruction

- Replication blocks provide structure samples and tree silhouette specify boundary conditions.
- Reconstruction of branches is also governed by reconstructed 3D points.
- Depending on their availability growth can be:
  - Unconstrained – 3D points are not available
  - Constrained – 3D points are available
Unconstrained Growth

- Randomly select end point of branch
- Priority is given to thicker branches or to those which are closer to tree trunk
- Randomly select replication block and attach it to branch end point after applying appropriate scaling
- Cap reconstructed branch as per extracted silhouette
Constrained Growth

- Growth is governed by reconstructed 3D points.
- Goal is to Minimize $\sum_i D(p_i, Tree)$ over
  $\{p_i | i = 1, ..., n_{3D}\}$
- Where $D(p_i, tree)$ is the smallest distance between a given point $p_i$ and the branch endpoints of Tree.
- Calculate $\sum_i D(p_i, Tree)$ for each sub tree and select one with minimum $\sum_i D(p_i, Tree)$
Optimization

- Determine influence cone at end point of each branch.
- Consider only those points which fall within this influence cone.
- Problem reduces to:
- Minimize \( \sum_{p_i \in \text{Cone}} D(p_i, \text{Subtree}) \)
- Where Cone is the set of points within the influence cone associated with tree node.
Populating tree with leaves

- Add leaves directly to branches using simple guidelines
  - Simple and does not require analysis of source images
  - Results are not realistic
- Add leaves by analyzing source images
  - Segmenting and clustering images
  - Produces realistic results
Process

- Apply mean shift color to produce homogeneous regions
- Apply split or merge to fit regions within prescribed size
- Cluster regions based on mean shift colors
- Fit leaf model to selected regions
- Determine position and orientation of each leaf
Mean shift filtering

- Data clustering algorithm used to construct homogeneous regions
- Takes into consideration both spatial location and color value

\[
K_{h_s,h_r}(x) = \frac{C}{h_s^2 h_r^2} k_E \left( \left| \frac{x_s}{h_s} \right|^2 \right) k_E \left( \left| \frac{x_r}{h_r} \right|^2 \right)
\]

\[
k_E(x) = 1 - x \text{ if } 0 \leq x \leq 1
\]

\(h_s \) ranged from 6 to 8 and \(h_r \) from 3 to 7.
Mean shift filtering

- Average out discontinuities due to bad lighting and occlusions
Region split and merge

- Build graph with each pixel as node.
- Edge between two neighbors if their mean-shift color difference is below a threshold.
- Connected components are considered as regions.
- Regions are split or merged if their size do not fall under valid range.
Image Segmentation
Form clusters

- Each region is candidate leaf.
- Merge regions to form clusters.
- Keep clusters associated with brightest colors.
- Fit each region with leaf model.
- User Interaction
  - Split or merge regions
  - Accept or reject particular cluster
Position and orientation of leaf

- Determine Position
  - Nearest 3D point
  - Nearest branch point
- Determine Orientation
  - Initially set orientation to be parallel to image plane
  - If more than 3 points project onto region use SVD to determine orientation.
  - Or choose orientation whose projection is closest to given region.
  - Discard redundant leaves.
Synthesize leaves

- Leaf density is ratio of number of leaves to length of branch.
- Leaf density should be evenly distributed.
- Synthesize leaves on branches with lowest leaf density.
Conclusion

- Described a system for constructing realistic-looking tree models from images
- Automatic techniques for recovering branches with user intervention
- The system currently requires that images be pre-segmented into tree branches, tree leaves and background.
References

- Image-based tree modeling – Ping Tan, Gang Zeng, Jingdong Wang, Sing Bing Kang, Long Quan
  - http://www.youtube.com/watch?v=uR7tUe5hnU0
So we can model
Thank You