Freehand acquisition of unstructured scenes

Presented by Mihai Mudure
July 2006
Goals

- Acquire interactively approximate models of unstructured scenes

- Inside-looking-out modeling case
  - Currently working on outside looking in case, to be extended to inside-looking-out

- Freehand
Unstructured scenes

- Scenes that contain many small surfaces
  - Leafy plants, messy desks, coats on a rack
Unstructured scenes

- Detailed modeling requires
  - Huge time investment
  - Expensive acquisition hardware
Challenges

- **Data acquisition**
  - Acquire depth information from many viewpoints

- **Interactivity**
  - The operator must be able to get feedback during data acquisition and guide the scanning
Challenges

- Tracking the acquisition device
- Modeling
Our solution

- Use the ModelCamera for acquisition
  - Acquires color frames enhanced with 45 depth samples
- Evolving model is a colored point cloud
- Point cloud displayed as we scan
**Our solution**

- **Tracking**
  - Previous approach: we used calibrated features (checkers)
    - Not very robust for long sequences
    - Operator had to concentrate on maintaining registration
  - ModelCamera mounted on a mechanical tracking arm
Our Solution

- Modeling
  - Disconnected representation
    - Splatting
  - Connected representation (triangle mesh)
    - Create an approximate mesh for each desired view
    - Color the mesh by projective texture mapping
Our solution

- Mesh generation
  - Project points onto the desired view
  - Splat
  - Triangulate in 2D
  - Unproject each pixel covered by a splat into 3D, each such point will be vertex of the 3D mesh

- Advantages
  - Minimizes the size of the skins in the desired view
Mesh
Mesh example
Mesh
Coloring

- Which reference images to use?
  - Project reference COPs onto a sphere centered around the object
  - Triangulate projections
  - When rendering, project the desired view COP onto the sphere, find the triangle and color using the corresponding reference cameras

- Assumption: the entire object is visible in the reference images
Coloring
Order reference cameras by the distance between the desired view COP projection and the reference camera COP projections onto the sphere.

For each desired view pixel find the pixel in the reference image where the corresponding 3D point projects.

Compare the depth of the point with the depth in the reference image (zbuffers for reference images are pre-computed).

If the point is visible in the reference image, assign color.
Coloring skins

- No good solution
- Skins are approximations of the surface, thus will get incorrect color from the reference cameras
- Currently, skin pixels which remain without color after the previously described step will get color from the closest reference camera, regardless of visibility
Results
Future work

- Extending the method to inside-looking-out case
  - Mesh generation method would work as is for a room model!
  - Adapting the coloring for the case when only part of the scene is visible in the reference views
- Speeding up rendering
Thank you