3D Scene Reconstruction : camera location

By Allyshia Sewdat
allyshias@gmail.com
School of Information Technology and Engineering, University of Ottawa

The Idea

Structure from Motion

Is it possible to recreate a 3D scene using 2D images?

Structure from motion (a process):

- recreates 3D structure from motion signals over time
- uses camera parameters and correspondences between 2D images of an object to recreate its 3D structure.

Several software bundles have been released that compute 3D geometric information using 2D images as input.

One of them is Bundler.

The Bundler

Bundler :

- is a software package created by Noah Snavely of the University of Washington
- is written in C++
- recreates scene geometry and camera locations using a given collection of images.

A significant application of Bundler : the Photo Tourism project, a collaboration between the University of Washington and Microsoft Research, yielding an interactive system for browsing large collections of photographs in 3D.

Photo Tourism

Noah Snavely, Steve Seitz, Kevin Chiu, Andy Hou - University of Washington
Richard Szeliski - Microsoft Reasearch
Photo : http://phototour.cs.washington.edu/bundler

- University of Washington's new way to browse photos.
- Input: large collections of photos of touristic attractions, taken from photo sharing web sites, such as *Flickr*.
- Aim: to compute photograph **location** and **orientation** to "reconstruct" a 3D model of the *likeness* of the original scene.
- User Interface allows user to navigate the scene by viewing it from different camera view points.
- Navigation involves selecting areas to view from a point cloud, and "stepping back" to see a larger picture from a given angle.
- The user views the scene entirely by looking at the pictures, carefully placed in the 3D coordinate system.
- The collections include pictures from varying cameras, of varying time of day, light, resolution and zoom levels.

The Question

We want to create an application that **navigates** around an object or a 3D location.

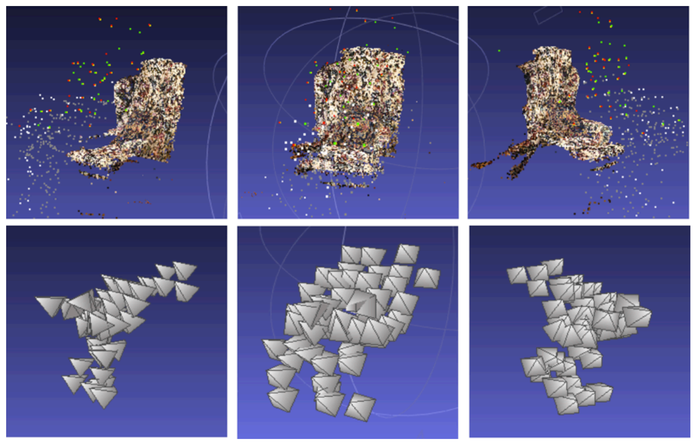
Can we do this using the Bundler Structure from Motion software package?

The Method

- **Download** and **configuration** of Bundler binaries and Cygwin environment (emulation of Unix for Windows), as well as other necessary binaries (SIFT keypoint matcher, Approximate Nearest Neighbour package) in preparation for running Bundler executables.
- Attempt to **run Bundler** with a series of test samples.
- **Troubleshooting** with Bundler's included Perl and Bash scripts when it did not yield expected results. According to the Bundler User Manual, a specific output text file containing rotation matrices, translation vectors and other keypoint and camera information should result from running the bundler. Modification to the Perl scripts provided with the Bundler was necessary to achieve this with our samples.
- **Success** with the output, which included .ply (polygon format) files, containing 3D coordinates and RGB colors of the keypoints found by Bundler. These points were viewed in Meshlab, an application for viewing and manipulating polygons and polyhedrons.
- Writing of Perl scripts to compute **camera positions** from output matrices and translation vectors, and to draw camera pyramids in order to visually verify the accuracy of the Bundler's output.
- Programming of a **Java application** allowing user to navigate around the input images based on the position of each corresponding image: navigation is performed by moving left, right, up and down around the image using keyboard arrows.

From image collection to 3D points

A collection of pictures taken from an ordinary digital camera from different angles around an object are used as input for the Bundler's structure from motion process.



Point cloud and camera pyramids retrieved from Bundler

Top row : matched points between the 54 images taken for this sample model, viewed in Meshlab
Bottom row: pyramids representing some of the cameras located by the Bundler

How Bundler Works

1- Detect **keypoints** points in each image
2- **Match** keypoints between pairs of images
3- Recover **camera parameters**
4- **Map camera locations in 3D coordinate system**

Keypoint Matching

- **Keypoints** are distincitve points found in a 2D image: often corners of the objects in the picture.
- A keypoint has a set of descriptors
- Points with equivalent descriptors are found between images.
- Matches are organized into ***tracks***, connected sets of matching keypoints accross images.
- Tracks which contain more than 1 keypoint in a single image are considered inconsistent.
- Consistent tracks containing at least 2 keypoints are kept for next phase.
- Bundler uses SIFT (Scale Invariant Feature Transform) Keypoint Detector (Lowe 2004), a software tool for detecting keypoints in each image.
  Approach : key points of an image (identified by difference-of-Gaussian function)
  → feature vectors (SIFT keys)
  → estimation of model parameters
  → object model
- Approximate Nearest Neighbour package (Arya et al 1998) is used for matching keypoint descriptors between pairs of images.

Camera locations

- Camera focal length is used to determine relative location of camera based on 3D location of each track.
- Parameters are estimated for a single pair of cameras, and one camera is added at a time to the system.

Bundler output

- A 3x3 rotation matrix (R) is given for each camera: determines the rotation around each of the x, y, and z axes required to find the angle of the camera.
- A translation vector (t) is given to determine the distance of a given camera from the origin of the coordinate system.

Viewing direction of the camera = R' + [0 0 -1]
3D position of a camera = -R' * t

What are the camera pyramids?

The **viewing frustum** of a camera is the area of the visible world that the camera "sees" at a given moment.

The angular extent of this observable space is called the **field of view.**

The frustum is in the shape of a truncated pyramid, and consists of 6 planes: near, far, top, left, right and bottom.

In our representations, the complete pyramid shape allows for the summit to represent the camera itself in a given pose - illustrating the pose for one particular image. The pyramid base represents the far plane (i.e. The furthest visible plane from the camera).

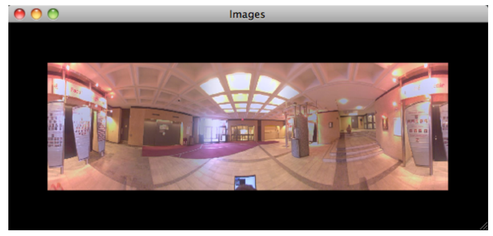
The pyramids are oriented in the same direction as the camera **viewing direction**, or the direction toward which the camera was pointed when it took the corresponding picture.

Applications

Object exploring software

Collection of photographs of one object
→ camera locations from Bundler
→ Java application allowing navigation around object

"Adjacent" images associated using camera coordinates



Screenshot of Java navigation application displaying photo of a Colonel By lobby (University of Ottawa) taken with panoramic camera

NAVIRE : Virtual Navigation in Image-Based Representations of Real World Environments

360 panoramic images
→ camera pose estimation
→ image-based representation of real environment

Currently underway at the VIVA lab, University of Ottawa

Though Bundler computes **relative** camera positions, it is possible to superimpose the resulting model on a map, satellite, image, floor plan, or other *geo-referenced* image to determine absolute geocentric coordinates for each camera.

Conclusion

The Bundler is an effective tool which can be applied, in lieu of Global Positioning Systems, in the navigation of real environments based on image viewing.

While this project was centered on camera position and basic navigation functionalities derivable from this, further work can be done by manipulating camera location data in order to provide more accurate and fluid navigation.

Also, the Bundler offers the reconstruction of a **sparse 3D model**, which can be used for photo browsing, or image-based navigation, and does not intend to recreate a *realistic* 3D model of an object. Other technologies exist, however, which do reconstruct dense 3D models. These open the field of study toward entirely new concepts, which could be explored as a follow-up to the object navigation application.

Acknowledgements

Research was conducted under the supervision and with the guidance of Professor Robert Laganière, VIVA Lab, University of Ottawa, and with the assistance of the members of the VIVA Lab.

Other acknowldegments:

David Lowe, University of British Columbia: SIFT Keypoint Detector, 2004.
http://www.cs.ubc.ca/~lowe/keypoints

Noah Snavely, Steven M. Seitz, Richard Szeliski, "Photo Tourism: Exploring Photo Colletions in 3D". SIGGRAPH Conf. Proc, 2006.

Noah Snavely, University of Washington : Bundler, and Bundler v0.4 User's Manual. 2009.
http://phototour.cs.washington.edu/bundler

Noah Snavely, Steve Seitz, Kevin Chiu, Andy Hou - University of Washington, Richard Szeliski - Microsoft Reasearch : Phototourism
http://phototour.cs.washington.edu

Sunil Arya and David M. Mount, University of Maryland, 2010 : ANN: A Library for Approximate Nearest Neighbor Searching
http://www.cs.umd.edu/~mount/ANN/

Undergraduate Research Opportunity Program, University of Ottawa