Panorama Stitching and Augmented Reality
Local feature matching with large datasets

- **Examples:**
  - Identify all panoramas and objects in an image set
  - Identify all products in a supermarket
  - Identify any location for robot localization or augmented reality
Matching in large unordered datasets
Matching in large unordered datasets
Nearest-neighbor matching

- Solve following problem for all feature vectors, $x$:
  \[
  \forall j \quad NN(j) = \arg \min_i \|x_i - x_j\|, \quad i \neq j
  \]

- Nearest-neighbour matching is the major computational bottleneck
  - Linear search performs $dn^2$ operations for $n$ features and $d$ dimensions
  - No exact methods are faster than linear search for $d>10$
  - Approximate methods can be much faster, but at the cost of missing some correct matches. Failure rate gets worse for large datasets.
K-d tree construction

Simple 2D example

Slide credit: Anna Atramentov
K-d tree query
Approximate k-d tree matching

Key idea:
- Search k-d tree bins in order of distance from query
- Requires use of a priority queue
Fraction of nearest neighbors found

Results:
- 100,000 uniform points in 12 dimensions.
- Speedup by several orders of magnitude over linear search.
Panorama stitching (with Matthew Brown)
Panorama stitching (with Matthew Brown)
Bundle Adjustment

- New images initialised with rotation, focal length of best matching image
Bundle Adjustment

- New images initialised with rotation, focal length of best matching image
Multi-band Blending

- Burt & Adelson 1983
  - Blend frequency bands over range $\propto \lambda$
2-band Blending

Low frequency ($\lambda > 2$ pixels)

High frequency ($\lambda < 2$ pixels)
Multi-band Blending

- Linear blending
- Multi-band blending
Automatic Straightening
Automatic Straightening

• Heuristic: user does not *twist* camera relative to horizon

• Up-vector perpendicular to plane of camera x vectors

\[
\left( \sum_i X_iX_i^T \right) u = 0
\]
Automatic Straightening
Gain Compensation

• No gain compensation
Gain Compensation

- Gain compensation

- Single gain parameter $g_i$ for each image

\[ e = \sum_{i} \sum_{j} \sum_{u_i \in R(i,j)} (g_i I_i(u_i) - g_j I_j(u_j))^2 \]
Panoramas from handheld consumer cameras

- Free working demo available: *Autostitch*
- Commercial products: Serif, Kolor, others coming

Show in Java applet: [Browser demo](#)
Autostitch usage in www.flickr.com

Over 20,000 panoramas posted by users of free Autostitch demo
Public images from Flickr

Surprise: Many users want borders to be visible
Applications:
- Film production (already in use)
- Heads-up display for cars
- Tourism
- Medicine, architecture, training

What is needed:
- Recognition of scene
- Accurate sub-pixel 3-D pose
- Real-time, low latency
Augmented Reality
(David Lowe & Iryna Gordon)

- Solve for 3D structure from multiple images
- Recognize scenes and insert 3D objects

Shows one of 20 images taken with handheld camera
System overview

input images → feature extraction → SIFT features → feature matching → 2-view matches

object-to-scene transformation

virtual object insertion → scene model

bundle adjustment

match validation → multi-view matches

virtual object rendering

camera pose estimation

model recognition

augmented video frame → current video frame

offline online
Bundle adjustment: an example

20 input images

50 iterations: error = 0.2 pixels
Incremental model construction

- **Problems:**
  - computation time increases with the number of unknown parameters
  - trouble converging if the cameras are too far apart (> 90 degrees)

- **Solutions:**
  - select a subset of about 4 images to construct an initial model
  - incrementally update the model by resectioning and triangulation
  - images processed in order determined by the spanning tree
3D Structure and Virtual Object Placement

- Solve for cameras and 3D points:
  - Uses bundle adjustment (solution for camera parameters and 3D point locations)
  - Initialize all cameras at the same location and points at the same depths
  - Solve depth-reversal ambiguity by trying both options

- Insert object into scene:

Set location in one image, move along epipolar in other, adjust orientation
Augmentation Example