Determining Geometry from Images

Richard Szeliski
Microsoft Research

SIGGRAPH Course on Image-Based Modeling, Rendering,
and Lighting
August 10, 1999

Computer Vision

Computer Vision is the inverse of Computer Graphics:
◆ computer graphics:
  – given a 3D model, render it
◆ computer vision
  – given some images, create a 3D model
This talk describes some techniques for recovering 3D geometry from images.
Motivation

◆ model building for virtual reality, animation, and CAD is slow and tedious
◆ animators and designers want photo-realistic (texture-mapped) models
◆ video input, display, and processing hardware becoming ubiquitous (multimedia)
◆ computer vision algorithms becoming more mature and reliable

Applications

◆ recover camera location to superimpose graphics on image [Gleicher 92]
◆ extract texture maps from real world [Beardsley96, Debevec96]
◆ create a 3-D model object or world model, without extensive interactive modeling
Applications (example)

- 3D model building example

3D model, octree, 3D curves, texture-mapped

Outline

- camera calibration
- pose estimation (view correlation)
- triangulation
- structure from motion
- feature matching (correlation)
- stereo matching (dense shape estimates)
- volumes (octrees) from silhouettes
- surface curves from profiles
- inverse texture mapping
- applications
Camera calibration

- determine camera *internal* (focal length) and *external* (pose) parameters from known 3D points
- forward projection equations

\[
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix}
= [R]_{c3}
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix}
+ t
\]

\[
\begin{bmatrix}
u \\
v \\
w
\end{bmatrix}
\sim
\begin{bmatrix}
U \\
V \\
W
\end{bmatrix}
= 
\begin{bmatrix}
f & 0 & u_c \\
0 & f & v_c \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix}
\]

Camera matrix calibration

- directly estimate 11 unknowns in 3×4 matrix projecting 3D ⇒ 2D

\[
\begin{bmatrix}
u \\
v \\
w
\end{bmatrix}
= 
\begin{bmatrix}
m_{00} & m_{01} & m_{02} & m_{03} \\
m_{10} & m_{11} & m_{12} & m_{13} \\
m_{20} & m_{21} & m_{22} & m_{23}
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z \\
1
\end{bmatrix}
\]

- bring denominator over, solve set of linear equations
Camera matrix calibration

◆ Advantages:
  – very simple to formulate and solve
◆ Disadvantages:
  – doesn't compute internal parameters
  – more unknowns than true degrees of freedom
  – need a separate camera matrix for each new view

Pose estimation

◆ once the internal camera parameters are known, can compute camera pose
◆ application: superimpose 3D graphics onto video
◆ possible solution techniques:
  – use standard calibration code [Tsai87]
  – use view correlation [Bogart91]
  – use through the lens camera control [Gleicher92]
  – other techniques from computer vision
Triangulation (Stereo)

◆ Problem: Given some points in correspondence across two or more images (taken from calibrated cameras), \((u_j, v_j)\), compute the 3D location \(X\).

Method I: intersect viewing rays in 3D, minimize:

\[
\arg \min_X \sum_j \min_{s_j} ||C_j + s_j V_j - X||^2
\]

- \(X\) is the unknown 3D point
- \(C_j\) is the optical center of camera \(j\)
- \(V_j\) is the viewing ray for pixel \((u_j, v_j)\)
- \(s_j\) is unknown distance along \(V_j\)

◆ advantage: geometrically intuitive
Triangulation (con't)

- **Method II**: solve linear equations in $X$
  - advantage: very simple

- Method III: non-linear minimization
  - advantage: most accurate (image plane error)

Structure from motion

- Given many points in *correspondence* across several images, $\{(u_{ij},v_{ij})\}$, simultaneously compute the 3D location $X_i$ and camera (or motion) parameters $M_j$

- two main variants: calibrated, and uncalibrated (sometimes associated with Euclidean and projective reconstructions)

- long history of research algorithms [Longuet81,Tomasi92,Weng93a,Szeliski94e,Beardsley96a]
Structure from motion (con't)

- Simple iterative algorithm used for face reconstruction [Pighin98] assuming roughly known geometry and pose
  - assume \((u_c, v_c) = (0, 0)\), but \(f\) is unknown
    
    \[
    u_j = s_j \frac{r'_j \cdot X_j + t'_j}{1 + \eta_j r'_j \cdot X_j}, \quad v_j = s_j \frac{r'_j \cdot Y_j + t'_j}{1 + \eta_j r'_j \cdot X_j}
    \]

    where \(\eta_j = 1/t^z_j\) is the inverse distance to object, and \(s_j = f_j/t^z_j\) is a world-pixel scale factor

- **advantage**: works well for narrow fields of view when \(f\) and \(t^z_j\) are hard to estimate

Structure from motion (con't)

- bring denominator over to l.h.s.
- iteratively solve for: \(s_j, X_i, R_j, t^x_j, t^y_j, \eta_j\)
- all equations are linear, except for \(R_j\), which is linearized by using a small angle (instantaneous velocity) approximation
Structure from motion  
(example)

◆ automatically track points in video sequence, validate consistent matches, and build 3D structure from point tracks [Beardsley96a]  
◆ uses both points and lines for reconstruction  
◆ final output is texture-mapped model

Structure from motion: limitations

◆ very difficult to reliably estimate structure and motion unless:  
  – large (x or y) rotation or  
  – large field of view and depth variation  
◆ camera calibration important for Euclidean reconstructions  
◆ need good feature trackers  
◆ postprocessing of the resulting 3-D points?
Feature matching (correlation)

- Find corresponding points in image video sequence
  - one simple technique: find two patches with minimal summed squared error [Anandan89]

\[
E_{xy}(u, v) = \sum_{k=x-w}^{x+w} \sum_{l=y-w}^{y+w} [I_1(k + u, l + v) - I_0(k, l)]^2
\]

Feature matching (optic flow)

- need sub-pixel precision to get best registration
- solution: Taylor series expansion of image function [Lucas81a]

\[
E(u + \delta u) = \sum_i (e_i + g_i \cdot \delta u)^2
\]

where \(x' = x + u\), \(e_i = I_1(x') - I_0(x)\), \(g_i = \nabla I_1(x')\)
Feature matching (optic flow)

- solve 2×2 system
  \[ \sum_i g_i g_i^T \delta u = - \sum_i e_i g_i \]
- use a coarse-to-fine pyramid to speed up search [Bergen92a]
- related to Brightness Constancy Equation [Horn81]
  \[ I_x u + I_y v - I_t = 0 \]

Stereo: epipolar geometry

- Match features along epipolar lines
Stereo: epipolar geometry

- For two images (or images with collinear camera centers), can find epipolar lines.
- Epipolar lines are the projection of the pencil of planes passing through the centers.
- Rectification: warping the input images (perspective transformation) so that epipolar lines are horizontal [Faugeras ‘93; Loop & Zhang ‘99].

Stereo: dense depth

- Apply feature matching criterion at all pixels simultaneously.
- Search only over epipolar lines (many fewer candidate positions).
- Can also match features such as lines.
Stereo: hierarchical matching

- Use coarse-to-fine search in an image pyramid to handle larger displacements
  [Bergen et al.'92]

Stereo: certainty modeling

- Compute certainty map from correlations

input  depth map  certainty map
Stereo: dense depth

- recovered depth map can be used for view interpolation [Chen93, Szeliski95, Seitz96]

[Dense Stereo Matching]

Dense Stereo Matching

- Advantages:
  - gives detailed surface estimates
  - multi-view aggregation improves accuracy

- Limitations:
  - narrow baseline ⇒ noisy estimates
  - fails in textureless areas
  - sparse, incomplete surface
  - sensitive to non-Lambertian effects
Stereo matching: limitations

◆ problems at and near occlusions
◆ incorrect color extraction, no partial occupancy in (mixed) border pixels

◆ solution: simultaneously recover disparities, colors, and opacities

Multi-Image Scene Recovery

◆ Goals of new stereo algorithm
  – simultaneously recover disparities, colors, and opacities (c.f. blue screen matting)
  – explicitly handle occlusions
  – true multi-frame setting [Collins, CVPR’96]
  – details in [Szeliski & Golland, ICCV’98]
Plane Sweep Stereo

- Sweep family of planes through volume
  - Each plane defines an image ⇒ composite homography

For each depth plane
- Compute composite (mosaic) image — *mean*
- Compute error image — *variance*
- Convert to confidence and aggregate spatially
- Select winning depth at each pixel
Plane Sweep Stereo

- "Stack of acetates" model (related to LDI)
  - warp and composite (over) back-to-front

Plane Sweep Stereo

- Compute visibility each input/layer pair
- Recompute means, confidences, and opacities
Voxel Coloring

- Generalizes plane sweep camera geometry
  - replace plane sweep with surface sweep
    [Seitz & Dyer, CVPR’97][Kutalakos & Seitz]

Results for dinosaur and rose
Stereo with Matting

◆ Estimate fractional opacities for pixels
  – adjust layer “sprites” (colors and opacities) to best match input images
  – optimization criteria:
    ✦ re-synthesis error
    ✦ color and opacity smoothness
    ✦ prior distribution on opacities
  – corresponds to MAP Bayesian estimator

Stereo with Matting

◆ SRI Trees sequence example

input images               stereo layers
Stereo with Matting

◆ Advantages:
  – true multi-image matching
  – deals with occlusions and mixed pixels

◆ Limitations:
  – too many degrees of freedom (volume)
  – breaks up surfaces into “voxels”
  – no “sub-pixel” depths

Layered Stereo

◆ Use arbitrarily oriented sprites
  [Baker,Szeliski,Anandan’98]

◆ Estimate 3D plane equation for each sprite
Layered Stereo

◆ Assign pixel to different “layers” (objects, sprites)

Layered Stereo

◆ Track each layer from frame to frame, compute plane eqn. and composite mosaic

◆ Re-compute pixel assignment by comparing original images to sprites
Layered Stereo

- Resulting sprite collection

Layered Stereo

- Re-synthesize original or novel images from collection of sprites
Layered Stereo Demo

- *SpriteViewer*: renders sprites with depth

![SpriteViewer](image)

Layered Stereo

- Per-pixel residual depth estimation
  - *plane plus parallax* [Anandan et al.]
  - *model-based stereo* [Debevec et al.]

  - better accuracy / fidelity
  - makes *forward warping* more difficult
Layered Stereo

◆ Advantages:
  – can represent occluded regions
  – can represent transparent and border (mixed) pixels (sprites have \textit{alpha} value per pixel)
  – works on texture-less interior regions

◆ Limitations:
  – fails for high depth-complexity scenes
  – may need manual initialization / control

Volumes from silhouettes

◆ extract binary \textit{silhouette} of object photographed against known background
◆ each silhouette + camera center defines enclosing conic region of space
◆ intersection of cones ⇒ bounding volume
◆ use octree representation of volume for efficiency [Szeliski93h]
Volumes from silhouettes

Cup on turntable example

Advantages:
- simple to implement, fairly robust
- fast execution
- complete (closed) surface

Disadvantages:
- only produces line hull
- limited resolution
- sensitive to classification (thresholding)
Surface curves from profiles

- extract and link edges in each image
- match edges across image sequence
- infer 3-D location from 2 or more matched edges:
  - for *stationary edge* (surface marking, sharp crease), use regular triangulation
  - for smooth self-occluding *profile* (limb), use 3 or more edges, fit circular arc [Szeliski94]

Coffee jar example
Surface curves from profiles

◆ Advantages:
- correct estimates at occluding contours
- good for smoothly curved objects
- provides intrinsic surface estimates, piecewise continuous surface mesh
- works on interior surface markings

◆ Disadvantages:
- fails in textureless interior areas
- incomplete surface (not closed)

Inverse texture mapping (photometry)

◆ recover color distribution over shape
◆ undo shading effects:
  - diffuse illumination
  - single source Lambertian
◆ weight contribution by surface normal
◆ smooth (and sharpen) results
  [Yu & Malik; Debevec]
Application: 3D face model building [Pighin98a]

- take several photos of a face from different views
- identify key points (eye and mouth corners, nose tip, ...) in each image
- recover camera position and coarse geometry using structure from motion
- add more correspondences, refine geometry, and interpolate to the rest of the mesh
Application: 3D face model building [Pighin98a]

- recover cylindrical texture map
- refine shape estimates using stereo
- animate by morphing between expressions

“neutral” → “joy”
3D face model-based tracking

- Use “analysis by synthesis” to match 3D face model parameters to input video

3D model-based effects

- Change viewpoint, identity, illumination, or add special effects (scars, tattoos, …)
Applications

◆ industrial applications
  – CAD/CAM
  – “3D Fax”: collaborative design
  – architecture
  – biomedical (surgery, prostheses)
  – special effects (FX), virtual studio
  – fashion & clothing

◆ consumer applications:
  – 3D world building (travel, home sales, home page, ...)
  – 3D model construction (art, hobby, ..)
  – 3D avatar construction (heads)
  – “3D videophone”
Applications: panoramas

To find out more

- general references on computer vision: [Ballard82,Horn86,Faugeras93,Nalwa93]
- recent survey of (some) 3D modeling techniques [Szeliski97]
- Workshop on Image-Based Modeling and Rendering: http://graphics.stanford.edu/workshops/ibr98/
Bibliography


---

Bibliography


