Introducción a la Fotografía 3D
UBA/FCEN Marzo 27 – Abril 12 2013
Clase 4 : Viernes Abril 5

Gabriel Taubin
Brown University
Course Schedule

- Introduction
- The Mathematics of 3D Triangulation
- 3D Scanning with Swept-Planes
- **Camera and Swept-Plane Light Source Calibration**
- Reconstruction and Visualization using Point Clouds
Modeling Lens Distortion

Without lens distortion distortion

\[
\begin{bmatrix}
  x_p \\
  y_p \\
\end{bmatrix} = \begin{bmatrix}
  f_x X^W / Z^W + c_x \\
  f_y X^W / Z^W + c_y \\
\end{bmatrix}
\]

Modeling Lens Distortion

Radial distortion

\[ x_{\text{corrected}} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \]
\[ y_{\text{corrected}} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \]

Modeling Lens Distortion

Tangential distortion

\[ x_{\text{corrected}} = x + [2p_1 y + p_2 (r^2 + 2x^2)] \]

\[ y_{\text{corrected}} = y + [p_1 (r^2 + 2y^2) + 2p_2 x] \]

Modeling Lens Distortion

\[
\begin{bmatrix}
  x_p \\
  y_p
\end{bmatrix} = (1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \begin{bmatrix}
  x_d \\
  y_d
\end{bmatrix} + \begin{bmatrix}
  2 p_1 x_d y_d + p_2 (r^2 + 2x_d^2) \\
  p_1 (r^2 + 2y_d^2) + 2p_2 x_d y_d
\end{bmatrix}
\]

Intrinsic Camera Calibration

- How to estimate intrinsic parameters and distortion model? (unknowns: focal length, skew, scale, principal point, and distortion coeffs.)
Intrinsic Camera Calibration

- How to estimate intrinsic parameters and distortion model? (unknowns: focal length, skew, scale, principal point, and distortion coeffs.)
- Popular solution: Observe a known calibration object (Zhang [2000])
Intrinsic Camera Calibration

- How to estimate intrinsic parameters and distortion model? (unknowns: focal length, skew, scale, principal point, and distortion coeffs.)
- Popular solution: Observe a known calibration object (Zhang [2000])
- Each 2D chessboard corner yields two constraints on the 6-11 unknowns
Intrinsic Camera Calibration

- How to estimate intrinsic parameters and distortion model? (unknowns: focal length, skew, scale, principal point, and distortion coeffs.)
- Popular solution: Observe a known calibration object (Zhang [2000])
- Each 2D chessboard corner yields two constraints on the 6-11 unknowns
- But, must also find 6 extrinsic parameters per image (rotation/translation)

\[ \Rightarrow \text{Result: Two or more images of a chessboard are sufficient} \]
CHAPTER 11

Camera Models and Calibration

Learning OpenCV

Vision begins with the detection of light from the world. That light begins as rays emanating from some source (e.g., a light bulb or the sun), which then travels through space until striking some object. When that light strikes the object, much of the light is absorbed, and what is not absorbed we perceive as the color of the light. Reflected light that makes its way to our eye (or our camera) is collected on our retina (or our image).

The geometry of this arrangement—particularly of the ray’s travel from the object, through the lens in our eye or camera, and to the retina or image—is of particular importance to practical computer vision.

A simple but useful model of how this happens is the pinhole camera model. A pinhole is an imaginary wall with a tiny hole in the center that blocks all rays except those passing through the tiny aperture in the center. In this chapter, we will start with a pinhole camera model to get a handle on the basic geometry of projecting rays. Unfortunately, a real pinhole is not a very good way to make images because it does not gather enough light for rapid exposure. This is why our eyes and cameras use lenses to gather more light than what would be available at a single point. The downside, however, is that gathering more light with a lens not only forces us to move beyond the simple geometry of the pinhole model but also introduces distortions from the lens itself.

In this chapter we will learn how, using camera calibration, to correct (mathematically) for the main deviations from the simple pinhole model that the use of lenses imposes on us. Camera calibration is important also for relating camera measurements with measurements in the real, three-dimensional world. This is important because scenes are not only three dimensional; they are also physical spaces with physical units. Hence, the relation between the camera’s natural units (pixels) and the units of the

Demo: Camera Calibration in Matlab

Extrinsic Camera Calibration

\[ X_C = RX_W + T \]
\{R,T\}?
Extrinsic Camera Calibration

\[ X_C = RX_W + T \]

{\( R, T \)?
Extrinsic Camera Calibration

\[ X_C = R_v X_v + T_v \]
How to map an image pixel to an optical ray?

Solution: Invert the *calibrated* camera projection model

But, also requires inversion of distortion model (which is non-linear)

Mapping implemented in Camera Calibration Toolbox with `normalize.m`

Result: After calibration, pixels can be converted to optical rays
Shadow Plane Calibration
Shadow Plane Calibration
Shadow Plane Calibration
Shadow Plane Calibration

\[ P(t) = \Lambda_h(t) \cap \Lambda_v(t) \]
\[ n(t) = \Lambda_h(t) \otimes \Lambda_v(t) \]
Alternatives for Shadow Plane Calibration

Alternatives for Shadow Plane Calibration

Alternatives for Shadow Plane Calibration

Point Cloud Reconstruction

\[ P = \Lambda_C(x, y) \cap \Pi_l(t) \]
Point Cloud Reconstruction

\[ P = \Lambda_C(x, y) \cap \Pi_l(t) \]
Demo: Putting it All Together

Loading object and reconstruction parameters...
Performing video processing...
+ finding the dividing line between reference planes...
  (click on two points on the dividing line)
+ define the reference area for the "vertical" plane...
  (click on top-left and bottom-right corners of reference area)
+ define the reference area for the "horizontal" plane...
  (click on top-left and bottom-right corners of reference area)
+ estimating per-pixel dynamic range and shadow thresholds...
+ estimating shadow boundaries...
+ estimating shadow crossing time(s)...
+ displaying video processing results...
Determined extrinsic calibration of reference plane(s)...
+ finding the extrinsic parameters of the "horizontal" plane....
  (click on the four extreme corners of the rectangular pattern,
  starting on the bottom-left and proceed counter-clockwise.)
+ finding the extrinsic parameters of the "vertical" plane....
  (click on the four extreme corners of the rectangular pattern,
  starting on the bottom-left and proceed counter-clockwise.)
Reconstructing 3D points using intersection with shadow plane(s)...
+ recovering implicit representation of shadow planes...
+ reconstructing 3D points...
Display reconstruction results and exporting VRML file...
+ displaying reconstruction results...
+ exporting VRML file...
#VRML V2.0 utf8
Shape {
  geometry IndexedFaceSet {
    coord Coordinate {
      point [
        1.633 -0.943 -0.667
        0.000  0.000  2.000
        -1.633 -0.943 -0.667
        0.000  1.886 -0.667
      ]
    }
    coordIndex [ 
      0 1 2  -1 3  1  0  -1 2 1 3  -1 2 3  0  -1
    ]
  }
}
Additional Reconstruction Examples


Course Schedule

- Introduction
- The Mathematics of 3D Triangulation
- 3D Scanning with Swept-Planes
- Camera and Swept-Plane Light Source Calibration

» Reconstruction and Visualization using Point Clouds