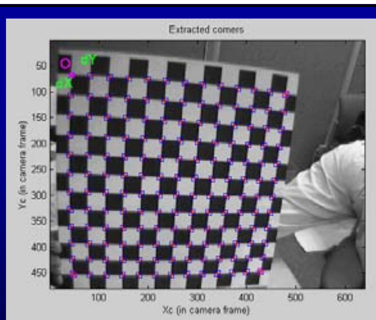


Camera calibration

Illustrations courtesy of Jean-Yves Bouguet

Motivation

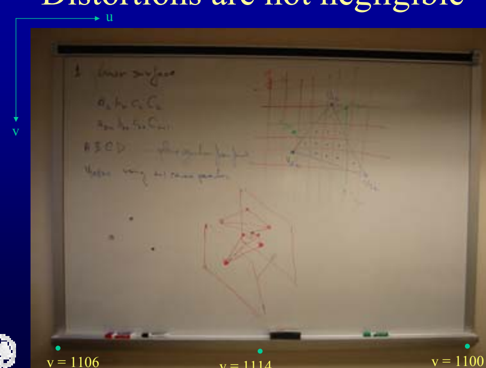
- Problem definition
  - for each pixel in an image, determine the corresponding 3D ray: find the camera model
- Applications
  - projective texture mapping
  - image stitching
  - depth from stereo
  - depth from structured light
  - color / depth registration
  - building ray databases
  - many more, whenever one needs to relate an image to another image or to the 3D scene that was captured

Planar pinhole camera model

- Ideal model
  - no distortions
  - infinitely small aperture ("everything is in focus")
- Pixel ray correspondence given by

$$ray(u,v) = (\dot{C} + \bar{c} + u\bar{a} + v\bar{b})$$

Distortions are not negligible



Camera parameters

- Intrinsic parameters
  - define the internal structure of the camera and are invariant to the location and orientation of the camera
- Extrinsic parameters
  - define the location and orientation of the camera

Camera parameters

- Intrinsics
  - Focal length: 2x1 vector (to allow for pixels with different width / height) f_1, f_2
  - Principal point: 2x1 vector (pixel coordinates of projection of COP onto image plane) u_0, v_0
  - Skew coefficient: scalar (angle between the x and y pixel axes) α
  - Distortion: 5x1 vector (radial and tangential distortion coefficients) $k_{1, 2, 3, 4, 5}$
- Extrinsics
  - three rotation angles and three translation scalars

"Plumb Bob" distortion model

$$\dot{P}(x_p, y_p, z_p)$$

$$p_n = \begin{bmatrix} \frac{x_p}{z_p} \\ \frac{y_p}{z_p} \end{bmatrix} = \begin{bmatrix} u_n \\ v_n \end{bmatrix}$$

$$r^2 = u_n^2 + v_n^2$$

$$p_{nd} = \begin{bmatrix} u_{nd} \\ v_{nd} \end{bmatrix} = (1 + k_1 r^2 + k_2 r^4 + k_5 r^6) p_n + \begin{bmatrix} 2k_3 u_n v_n + k_4 (r^2 + 2u_n^2) \\ k_3 (r^2 + 2v_n^2) + 2k_4 u_n v_n \end{bmatrix}$$

$$p_d = \begin{bmatrix} f_1 (u_{nd} + \alpha v_{nd}) + u_0 \\ f_2 v_{nd} + v_0 \end{bmatrix}$$

- 5 distortion coefficients
  - given point P (in camera frame)
  - p_n are normalized coordinates
  - p_{nd} are distorted n. c.
  - p_d are distorted coordinates

Calibration

- Estimate camera parameters using a list of known scene (3D) – image correspondences

$(x_0, y_0, z_0, u_0, v_0)$
$(x_1, y_1, z_1, u_1, v_1)$
…
$(x_n, y_n, z_n, u_n, v_n)$

Overview

- Introduction
- Camera calibration
  - establishing 3D – 2D correspondences
  - extracting camera parameters
  - error analysis
  - calibration with known intrinsics
- Approximate calibration of intrinsics

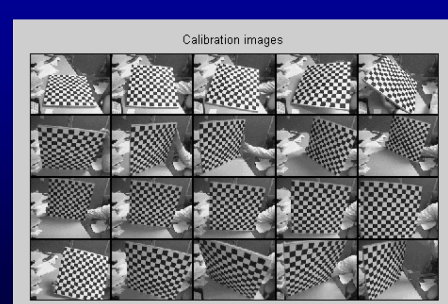
Overview

- Introduction
- Camera calibration
  - establishing 3D – 2D correspondences
  - extracting camera parameters
  - error analysis
  - calibration with known intrinsics
- Approximate calibration of intrinsics

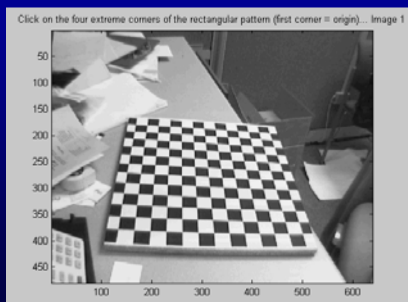
Scene – image correspondences

- Design scene with fiducials that
  - are easily detectable
  - are covering uniformly the fov of camera
  - have known geometry
- Typical calibration target is a bw checkerboard

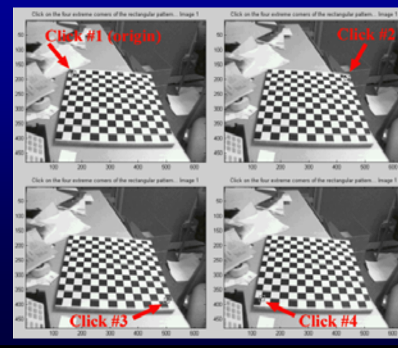
Take pictures of grid



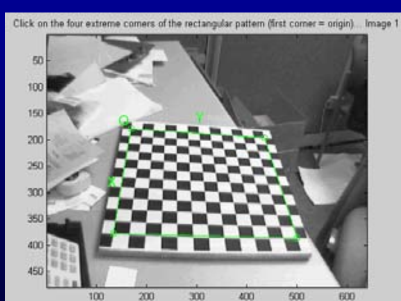
Corner extraction: finding grid



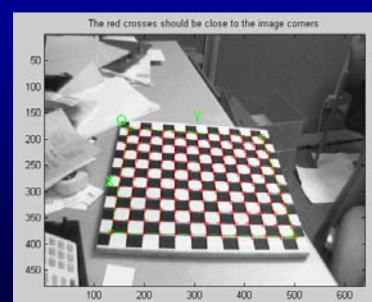
Corner extraction: finding grid



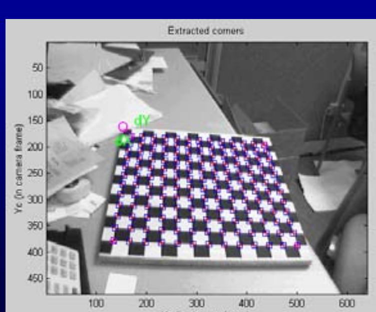
Corner extraction: finding grid

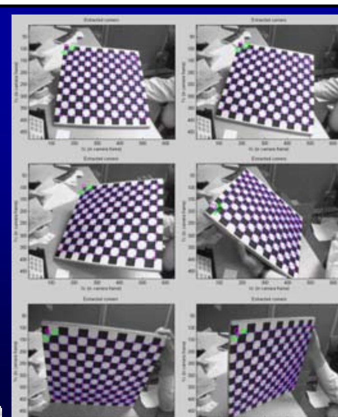


Corner extraction: prediction



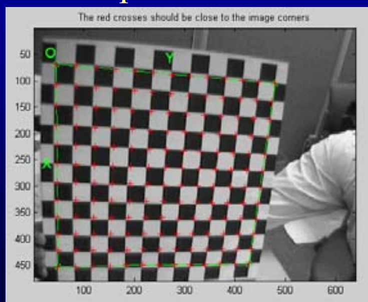
Corner extraction: search



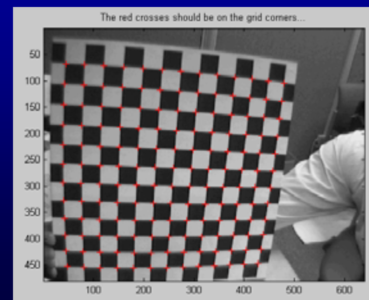


Corner extraction: search

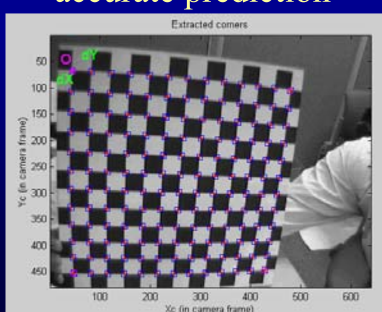
Corner extraction: inaccurate prediction



Corner extraction: estimate distortion for more accurate prediction



Corner extraction: OK with more accurate prediction



Overview

- Introduction
- Camera calibration
  - establishing 3D – 2D correspondences
  - extracting camera parameters
  - error analysis
  - calibration with known intrinsics
- Approximate calibration of intrinsics

Overview

- Introduction
- Camera calibration
  - establishing 3D – 2D correspondences
  - extracting camera parameters
  - error analysis
  - calibration with known intrinsics
- Approximate calibration of intrinsics

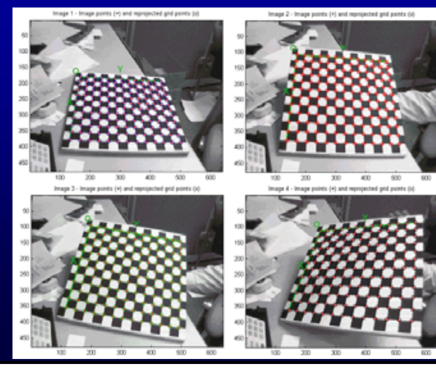
Calibration

- Search for camera parameters that best fit the correspondences given
- Over-constrained system, one wants to minimize the error

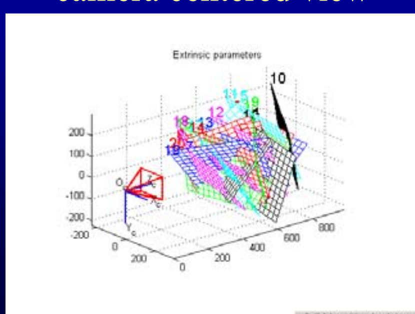
Calibration

```
Aspect ratio optimized (est_aspect_ratio = 1) -> both components of fc are estimated (DEFAULT).
Principal point optimized (center_optim=1) - (DEFAULT). To reject principal point, set center_optim=0
Skew not optimized (est_alpha=0) - (DEFAULT)
Distortion not fully estimated (defined by the variable est_dist):
     Sixth order distortion not estimated (est_dist(5)=0) - (DEFAULT) .

Main calibration optimization procedure - Number of images: 25
Gradient descent iterations: 1...2...3...4...done
Estimation of uncertainties...done


Calibration results after optimization (with uncertainties):

Focal Length:          fc = [ 657.29599   657.74627 ] ± [ 0.28562   0.29017 ]
Principal point:       cc = [ 302.98382   242.47752 ] ± [ 0.58016   0.55480 ]
Skew:             alpha_c = [ 0.00000 ] ± [ 0.00000  ]   => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:            kc = [ -0.25340   0.11884   -0.00027   0.00002  0.00000 ] ± [ 0.00232   0.00945   0.00012   0.00012  0.00000 ]
Pixel error:          err = [ 0.11801   0.11592 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).
```

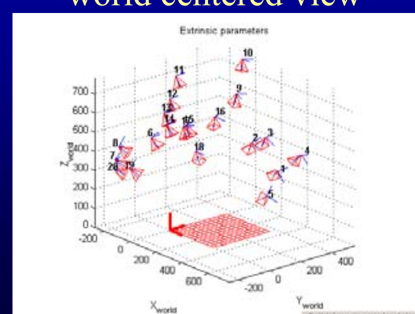

Reprojection of corners



Extrinsic parameters found – camera centered view



Extrinsic parameters found – world centered view



Overview

- Introduction
- Camera calibration
  - establishing 3D – 2D correspondences
  - extracting camera parameters
  - error analysis
  - calibration with known intrinsics
- Approximate calibration of intrinsics

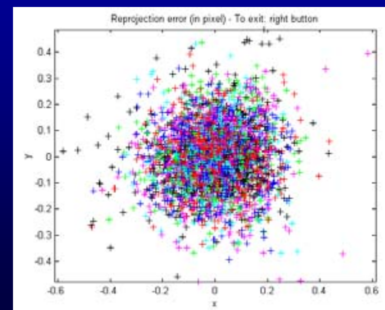
Overview

- Introduction
- Camera calibration
  - establishing 3D – 2D correspondences
  - extracting camera parameters
  - error analysis
  - calibration with known intrinsics
- Approximate calibration of intrinsics

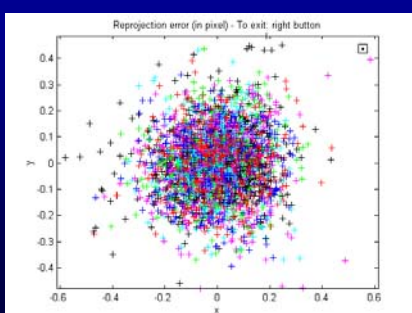
Error analysis

- Error
  - difference between corner location projected using calibrated camera parameters and actual corner location
- Acceptable errors
  - depends on application
  - “less than one pixel”

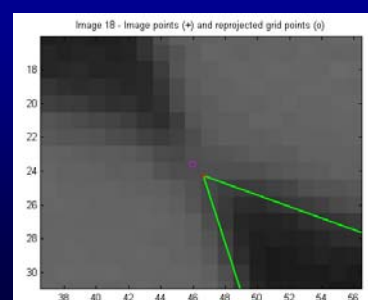
Error examples



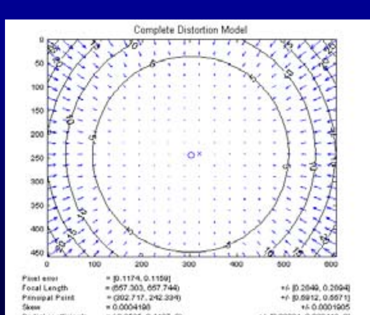
Inspecting individual error points



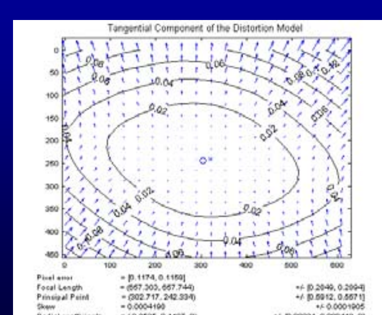
Inspecting individual error points



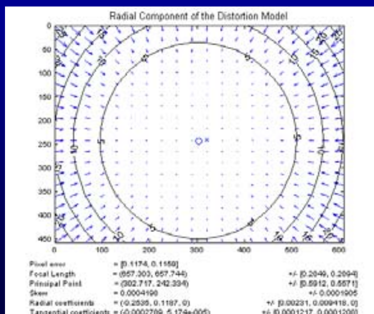
Distortion model visualization



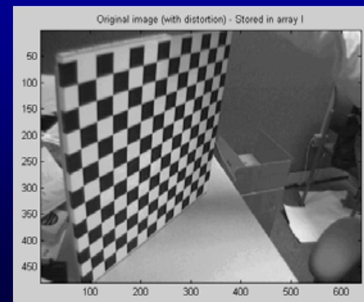
Distortion model visualization



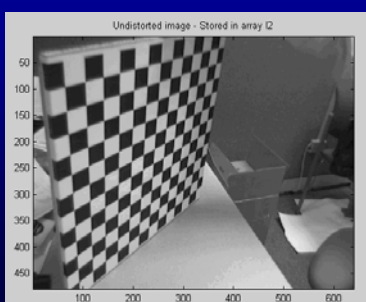
Distortion model visualization



Undistorting images: distorted



Undistorting images: undistorted



Overview

- Introduction
- Camera calibration
  - establishing 3D – 2D correspondences
  - extracting camera parameters
  - error analysis
  - calibration with known intrinsics
- Approximate calibration of intrinsics

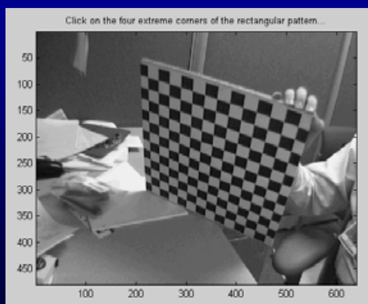
Overview

- Introduction
- Camera calibration
  - establishing 3D – 2D correspondences
  - extracting camera parameters
  - error analysis
  - calibration with known intrinsics
- Approximate calibration of intrinsics

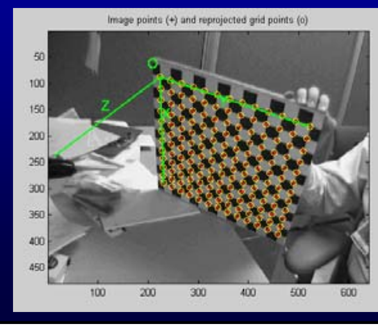
Pose estimation

- Determine position and orientation of known camera
- Use scene – image correspondences and intrinsic parameters
- Useful in many applications
  - projective texture mapping
  - image stitching
  - lightfield construction
  - video compositing, etc.

Pose estimation



Pose estimation



Overview

- Introduction
- Camera calibration
  - establishing 3D – 2D correspondences
  - extracting camera parameters
  - error analysis
  - calibration with known intrinsics
- Approximate calibration of intrinsics

Overview

- Introduction
- Camera calibration
  - establishing 3D – 2D correspondences
  - extracting camera parameters
  - error analysis
  - calibration with known intrinsics
- Approximate calibration of intrinsics

Approximate calibration of intrinsics

- Print out black and white grid
  - As large as possible, at least 40 cm x 40 cm
  - OK to use several letter-sized pages; measure the global consistency of the grid
- Aim camera perpendicularly to the grid
  - Entire field of view covered by grid
  - Frame margins as parallel as possible to the grid lines
  - Measure distance from grid to camera f_{cm} in cm
  - In image, measure width w_{cm} and height h_{cm} of patch of grid in cm (using known size of checkers)
- Approximate intrinsics
  - Set pixel width to 1; set pixel height to $(h_{cm}/h) / (w_{cm}/w)$, where w and h are the image dimensions in pixels
  - Assume square pixels, and that the COP projects in the center of the image
  - $a=(1, 0, 0)$, $b=(0, -pix_h, 0)$, $c=(-w/2, -h/2*pix_h, -f_{cm}/(w_{cm}/w))$, $C=(0, 0, 0)$