Introduction to Computer Vision
TTIC 31040
Winter 2022
Image gradients
Keypoint descriptor

Lecture 4: Cameras and calibration

TTIC 31040: Introduction to Computer Vision

Instructor: Greg Shakhnarovich

TTI-Chicago

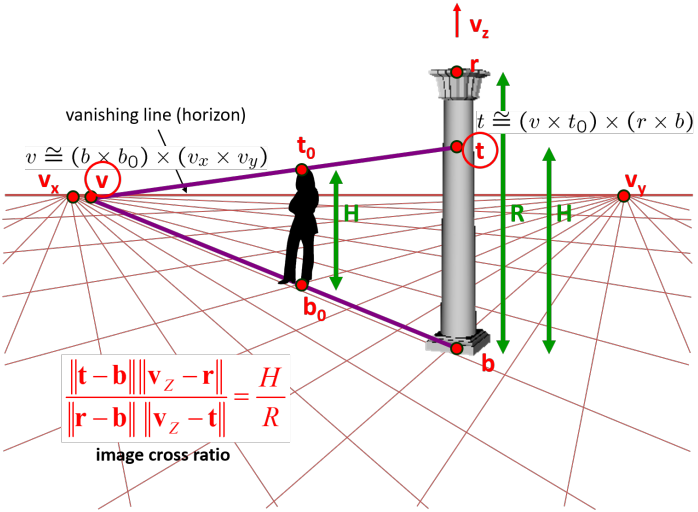
January 24, 2022

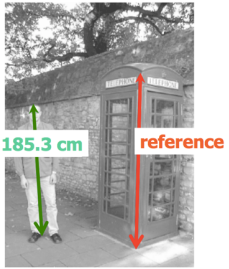
Administrivia

- Plan to have a quiz in week 4 (online, format TBD)
  Expect 30-40 min effort
- Reminder: office hours
  Mon 12:30: in person+Zoom (Greg)
  Wed 12:30: Zoom for now (Haochen)
  Thu 1pm: Zoom (Greg and/or Haochen)
- Start working on PS2 ASAP!
- Psets will sometimes foreshadow material from upcoming lectures (casualty of super short quarters...) If the pset mentions a concept not covered yet and does not explain it, just wait!

Review: single view metrology

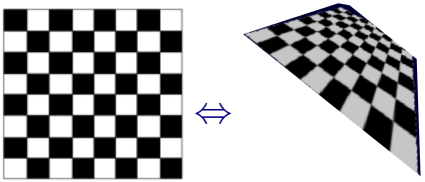
- Can get partial information about the scene from vanishing points/lines:

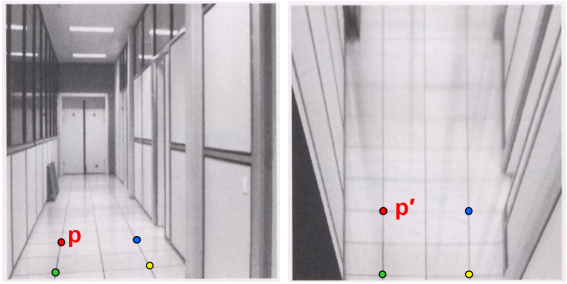




Review: estimating homographies

- To rectify an image w.r.t. a given plane: find at least four correspondences, then solve the direct linear transform

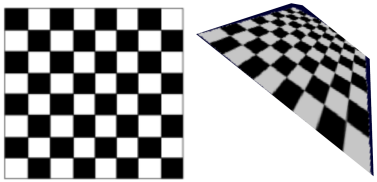




Homographies

- Recall: projective transformation 2D to 2D, preserving linearity: parameterized by $\mathbf{H}_{3\times 3}$

$$\lambda \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$



- Direct Linear Transform: solving for hs
- Each correspondence (pair of matched point coordinates in the two images) yields

$$\begin{aligned} x'(h_7 x + h_8 y + 1) &= h_1 x + h_2 y + h_3, \\ y'(h_7 x + h_8 y + 1) &= h_4 x + h_5 y + h_6 \end{aligned}$$

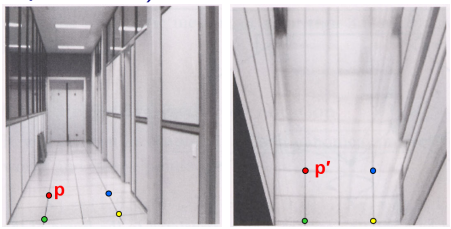
- Equivalent to solving a constrained linear system

$$\mathbf{h}^* = \underset{\|\mathbf{h}\|=1}{\operatorname{argmin}} \|\mathbf{A}\mathbf{h}\|^2$$

where $\mathbf{h} \in \mathbb{R}^9$ is the flattened $\mathbf{H}$

Homography estimation

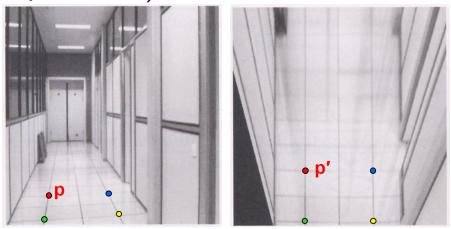
- To rectify an image: find points in known configuration (usually a rectangle with known aspect ratio)



$\mathbf{p}$: points in the origina image;
$\mathbf{p}'$: coordinates of where you want these points to go in the rectified image

Homography estimation

- To rectify an image: find points in known configuration (usually a rectangle with known aspect ratio)

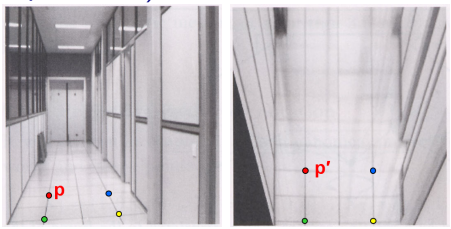


$\mathbf{p}$: points in the origina image;
$\mathbf{p}'$: coordinates of where you want these points to go in the rectified image

- Estimate $\mathbf{H}$; then use $\mathbf{H}$ to calculate for each pixel in the target the "source" pixels in the original, and interpolate to create a new image

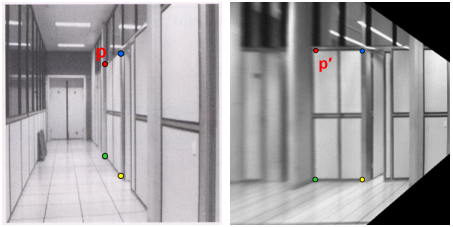
Homography estimation

- To rectify an image: find points in known configuration (usually a rectangle with known aspect ratio)



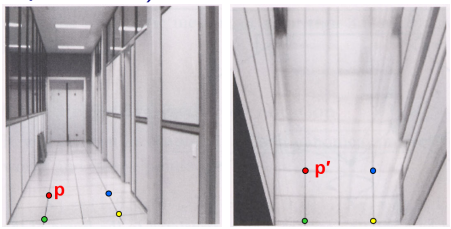
$\mathbf{p}$: points in the origina image;
$\mathbf{p}'$: coordinates of where you want these points to go in the rectified image

- Estimate $\mathbf{H}$; then use $\mathbf{H}$ to calculate for each pixel in the target the "source" pixels in the original, and interpolate to create a new image



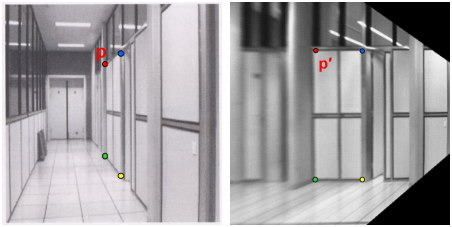
Homography estimation

- To rectify an image: find points in known configuration (usually a rectangle with known aspect ratio)



$\mathbf{p}$: points in the origina image;
$\mathbf{p}'$: coordinates of where you want these points to go in the rectified image

- Estimate $\mathbf{H}$; then use $\mathbf{H}$ to calculate for each pixel in the target the "source" pixels in the original, and interpolate to create a new image

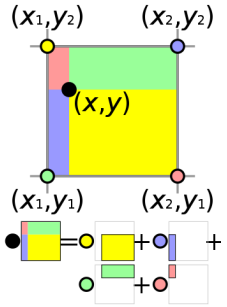


Interlude: interpolation

- Generally, geometric transformation maps pixel (i, j) to fractional coordinates (x', y')
- Nearest neighbor "interpolation": take value from closest pixel

$$(x, y) = \operatorname{argmin} \left\{ (x - x')^2 + (y - y')^2 \right\}$$

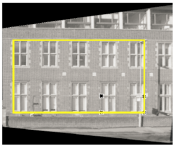
- Bilinear interpolation: find four pixels closest to (x', y') (four corners of a unit pixel "cell")
- Estimate value as the weighted average of the values in the corners, weighted by how far they are from (x', y')



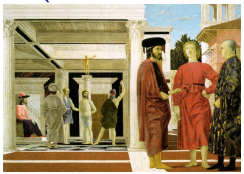
src: Wikipedia : Cmglee

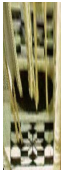
Homography for image rectification

- Caution: makes sense for planar scenes (or scene elements) only!



from Hartley & Zisserman

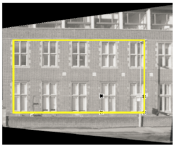




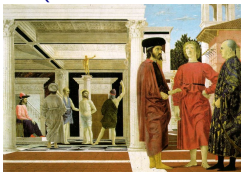
Flagellation, Piero della Francesca

Homography for image rectification

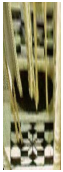
- Caution: makes sense for planar scenes (or scene elements) only!



from Hartley & Zisserman



Flagellation, Piero della Francesca

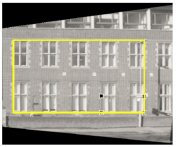


- Can play with homography to simulate new viewpoint

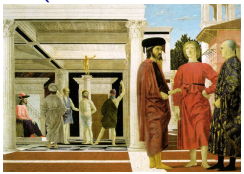
Vermeer, The Music Lesson

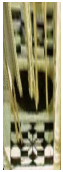
Homography for image rectification

- Caution: makes sense for planar scenes (or scene elements) only!



from Hartley & Zisserman





Flagellation, Piero della Francesca

- Can play with homography to simulate new viewpoint

Vermeer, The Music Lesson

- Need a bit more than just a single homography – e.g., model a scene as a *collection* of planar elements

3D reconstruction from single view

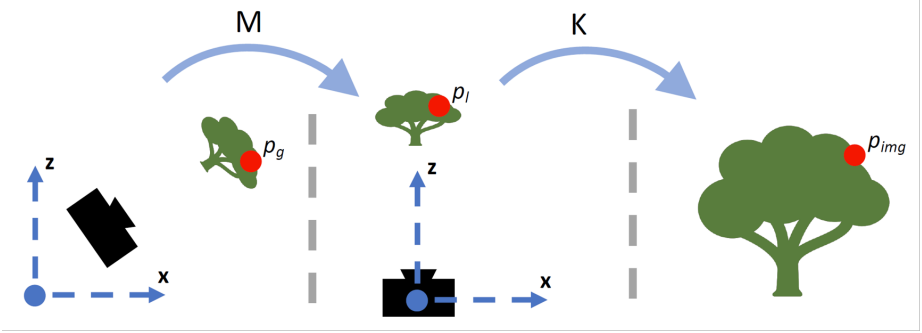
src: Criminisi et al., 2000

Roadmap

- Today: discuss real cameras and calibration
- Next up: multiple cameras!
- After that: our first taste of 3D reconstruction

Mapping from scene to image

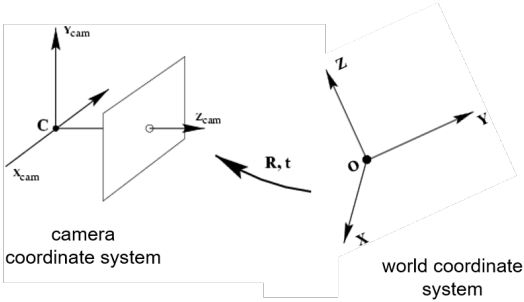
$$\begin{bmatrix} \text{2D point} \\ \text{(3x1)} \end{bmatrix} = \begin{bmatrix} \text{Camera to pixel coord. trans. matrix} \\ \text{(3x3)} \end{bmatrix} \begin{bmatrix} \text{Perspective projection matrix} \\ \text{(3x4)} \end{bmatrix} \begin{bmatrix} \text{World to camera coord. trans. matrix} \\ \text{(4x4)} \end{bmatrix} \begin{bmatrix} \text{3D point} \\ \text{(4x1)} \end{bmatrix}$$



src: P. Hedman

Camera coordinates vs. world coordinates

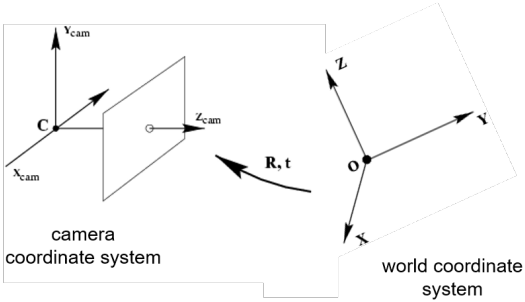
- The camera coordinate system (frame) is related to the world coordinate frame by a rotation and translation
- 6 DoF to recover



$$\tilde{\mathbf{X}}_c = \mathbf{R}\left(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}\right)$$

Camera coordinates vs. world coordinates

- The camera coordinate system (frame) is related to the world coordinate frame by a rotation and translation
- 6 DoF to recover

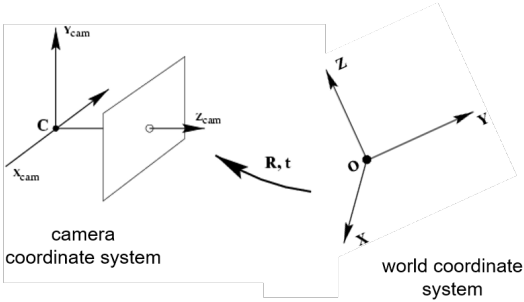


$$\tilde{\mathbf{X}}_c = \mathbf{R}\left(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}\right)$$

- $\tilde{\mathbf{X}}_c$: point in camera frame (inhomogeneous)

Camera coordinates vs. world coordinates

- The camera coordinate system (frame) is related to the world coordinate frame by a rotation and translation
- 6 DoF to recover

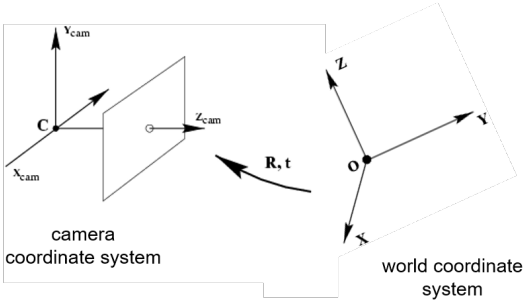


$$\tilde{\mathbf{X}}_c = \mathbf{R}\left(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}\right)$$

- $\tilde{\mathbf{X}}_c$: point in camera frame (inhomogeneous)
- $\tilde{\mathbf{X}}$: point in world frame (inhomeogeneous)

Camera coordinates vs. world coordinates

- The camera coordinate system (frame) is related to the world coordinate frame by a rotation and translation
- 6 DoF to recover

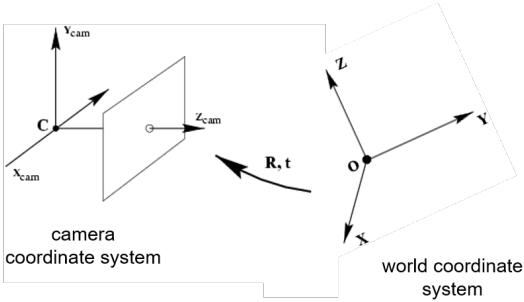


$$\tilde{\mathbf{X}}_c = \mathbf{R}\left(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}\right)$$

- $\tilde{\mathbf{X}}_c$: point in camera frame (inhomogeneous)
- $\tilde{\mathbf{X}}$: point in world frame (inhomeogeneous)
- $\tilde{\mathbf{C}}$: camera center in world frame (inhomogeneous)

Camera coordinates vs. world coordinates

- The camera coordinate system (frame) is related to the world coordinate frame by a rotation and translation
- 6 DoF to recover



$$\tilde{\mathbf{X}}_c = \mathbf{R}\left(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}\right)$$

- $\tilde{\mathbf{X}}_c$: point in camera frame (inhomogeneous)
- $\tilde{\mathbf{X}}$: point in world frame (inhomeogeneous)
- $\tilde{\mathbf{C}}$: camera center in world frame (inhomogeneous)
- $\mathbf{R}$: rotation matrix

Camera rotation and translation

$$\tilde{\mathbf{X}}_c = \mathbf{R}\left(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}\right)$$

Camera rotation and translation

$$\tilde{\mathbf{X}}_c = \mathbf{R}\left(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}\right) \qquad \begin{pmatrix} \tilde{\mathbf{X}}_c \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix} \begin{pmatrix} \tilde{\mathbf{X}} \\ 1 \end{pmatrix}$$

Camera rotation and translation

$$\tilde{\mathbf{X}}_c = \mathbf{R}\left(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}\right) \qquad \begin{pmatrix} \tilde{\mathbf{X}}_c \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix} \begin{pmatrix} \tilde{\mathbf{X}} \\ 1 \end{pmatrix}$$

$$\mathbf{X}_c = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{X}$$

Camera rotation and translation

$$\tilde{\mathbf{X}}_c = \mathbf{R}\left(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}\right) \qquad \begin{pmatrix} \tilde{\mathbf{X}}_c \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix} \begin{pmatrix} \tilde{\mathbf{X}} \\ 1 \end{pmatrix}$$

$$\mathbf{X}_c = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{X}$$

$$\mathbf{x} = \mathbf{K}\,[\mathbf{I}\,|\,\mathbf{0}] \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{X}$$

- $\mathbf{x}$: point in image coordinates (homogeneous)
- $[\mathbf{I}|\mathbf{0}]$: 3×4 projection matrix
- $\mathbf{K}$: 3×3 2D transformation (calibration matrix)

Camera rotation and translation

$$\tilde{\mathbf{X}}_c = \mathbf{R}\left(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}\right) \qquad \begin{pmatrix}\tilde{\mathbf{X}}_c \\ 1\end{pmatrix} = \begin{bmatrix}\mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1\end{bmatrix}\begin{pmatrix}\tilde{\mathbf{X}} \\ 1\end{pmatrix}$$

$$\mathbf{X}_c = \begin{bmatrix}\mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1\end{bmatrix}\mathbf{X}$$

$$\mathbf{x} = \mathbf{K}\,[\mathbf{I}\,|\,\mathbf{0}]\begin{bmatrix}\mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1\end{bmatrix}\mathbf{X}$$

$$\mathbf{x} = \mathbf{K}\left[\mathbf{R}\,|\,-\mathbf{R}\tilde{\mathbf{C}}\right]\mathbf{X},$$

- $\mathbf{x}$: point in image coordinates (homogeneous)
- $[\mathbf{I}|\mathbf{0}]$: 3×4 projection matrix
- $\mathbf{K}$: 3×3 2D transformation (calibration matrix)

Camera rotation and translation

$$\tilde{\mathbf{X}}_c = \mathbf{R}\left(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}\right) \qquad \begin{pmatrix}\tilde{\mathbf{X}}_c \\ 1\end{pmatrix} = \begin{bmatrix}\mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1\end{bmatrix}\begin{pmatrix}\tilde{\mathbf{X}} \\ 1\end{pmatrix}$$

$$\mathbf{X}_c = \begin{bmatrix}\mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1\end{bmatrix}\mathbf{X}$$

$$\mathbf{x} = \mathbf{K}\,[\mathbf{I}\,|\,\mathbf{0}]\begin{bmatrix}\mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1\end{bmatrix}\mathbf{X}$$

$$\mathbf{x} = \mathbf{K}\left[\mathbf{R}\,|\,-\mathbf{R}\tilde{\mathbf{C}}\right]\mathbf{X}, \qquad \mathbf{t} = -\mathbf{R}\tilde{\mathbf{C}}$$

- $\mathbf{x}$: point in image coordinates (homogeneous)
- $[\mathbf{I}|\mathbf{0}]$: 3×4 projection matrix
- $\mathbf{K}$: 3×3 2D transformation (calibration matrix)

Camera rotation and translation

$$\tilde{\mathbf{X}}_c = \mathbf{R}\left(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}\right)$$

$$\begin{pmatrix} \tilde{\mathbf{X}}_c \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix} \begin{pmatrix} \tilde{\mathbf{X}} \\ 1 \end{pmatrix}$$

$$\mathbf{X}_c = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{X}$$

$$\mathbf{x} = \mathbf{K}\,[\mathbf{I}\,|\,\mathbf{0}] \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{X}$$

$$\mathbf{x} = \mathbf{K}\left[\mathbf{R}\,|\,-\mathbf{R}\tilde{\mathbf{C}}\right]\mathbf{X}, \qquad \mathbf{t} = -\mathbf{R}\tilde{\mathbf{C}}$$

$$\mathbf{x} = \mathbf{K}\,[\mathbf{R}\,|\,\mathbf{t}]\,\mathbf{X}$$

- $\mathbf{x}$: point in image coordinates (homogeneous)
- $[\mathbf{I}|\mathbf{0}]$: 3×4 projection matrix
- $\mathbf{K}$: 3×3 2D transformation (calibration matrix)
- $[\mathbf{R}\,|\,\mathbf{t}]$: 3×4 camera *pose* (extrinsics) matrix

Interlude: rotation matrices

- A rotation in 3D is defined by an axis $\mathbf{v}$, $\|\mathbf{v}\| = 1$, and an angle θ (3 DoF)
- Can be computed as $\mathbf{x}' = \mathbf{R}\mathbf{x}$ with an orthonormal matrix $\mathbf{R}$
- Rodrigues' formula:

$$\mathbf{R} = \mathbf{I} + \sin\theta\mathbf{V} + (1 - \cos\theta)\mathbf{V}^2$$

where

$$\mathbf{V} = \begin{bmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{bmatrix}$$

Interlude: rotation matrices

- A rotation in 3D is defined by an axis $\mathbf{v}$, $\|\mathbf{v}\| = 1$, and an angle θ (3 DoF)
- Can be computed as $\mathbf{x}' = \mathbf{R}\mathbf{x}$ with an orthonormal matrix $\mathbf{R}$
- Rodrigues' formula:

$$\mathbf{R} = \mathbf{I} + \sin\theta\mathbf{V} + (1 - \cos\theta)\mathbf{V}^2$$

where

$$\mathbf{V} = \begin{bmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{bmatrix}$$

note: $\mathbf{V} \cdot \mathbf{v} = \mathbf{v} \times \mathbf{v}$

Interlude: rotation matrices

- A rotation in 3D is defined by an axis $\mathbf{v}$, $\|\mathbf{v}\| = 1$, and an angle θ (3 DoF)
- Can be computed as $\mathbf{x}' = \mathbf{R}\mathbf{x}$ with an orthonormal matrix $\mathbf{R}$
- Rodrigues' formula:

$$\mathbf{R} = \mathbf{I} + \sin\theta\mathbf{V} + (1 - \cos\theta)\mathbf{V}^2$$

where

$$\mathbf{V} = \begin{bmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{bmatrix}$$

note: $\mathbf{V} \cdot \mathbf{v} = \mathbf{v} \times \mathbf{v}$

- Another interpretation of $\mathbf{R}$: its rows give the rotated basis, replacing the original $(1, 0, 0)$, $(0, 1, 0)$, $(0, 0, 1)$
- Many other representations for rotations!

Camera calibration

- To define the mapping between the world and the image, we need to know

$$\mathbf{P} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]$$

Camera calibration

- To define the mapping between the world and the image, we need to know

$$\mathbf{P} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]$$

- **Intrinsic parameters** in $\mathbf{K}$:
  principal point;
  distance to camera;
  pixel sizes

Camera calibration

- To define the mapping between the world and the image, we need to know

$$\mathbf{P} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]$$

- **Intrinsic parameters** in $\mathbf{K}$:
  principal point;
  distance to camera;
  pixel sizes
- **Extrinsic parameters** in $\mathbf{R}, \mathbf{t}$: location and orientation of the camera ("six DoF")
- If you move your camera: change $\mathbf{R}, \mathbf{t}$, keep $\mathbf{K}$
- If you change the camera but keep the location and orientation: change $\mathbf{K}$, keep $\mathbf{R}, \mathbf{t}$

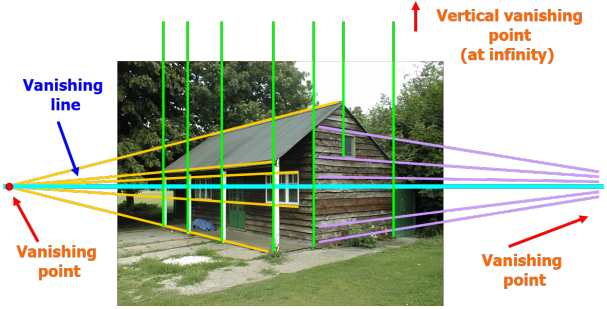
Camera calibration (extrinsics)

- If we have some known reference coordinates in the scene, we can *calibrate* the camera (estimate $\mathbf{P}$) – later
- What if we do not have known points?

Camera calibration (extrinsics)

- If we have some known reference coordinates in the scene, we can *calibrate* the camera (estimate $\mathbf{P}$) – later
- What if we do not have known points?

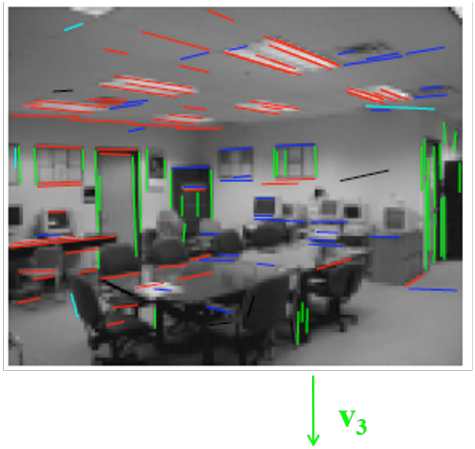
- If we have vanishing points for 3 orthogonal directions, we can use that!



src: slide:Efros, photo:Criminisi

Calibration from vanishing points

- Intuition: align the world coordinate system with the observed coordinate system of the three orthogonal directions in the image



Calibration from vanishing points

- Intuition: align the world coordinate system with the observed coordinate system of the three orthogonal directions in the image

- Here $\mathbf{v}_1$, $\mathbf{v}_2$ are finite (albeit outside of image), $\mathbf{v}_3$ is at infinity

Calibration from vanishing points

$$\mathbf{P} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{bmatrix}$$

- Decide (w.l.o.g.) that the world coordinates are axis-aligned with the three directions you have
- $\mathbf{p}_1 = \mathbf{P} \cdot (1, 0, 0, 0)$: the vanishing point in the x direction (in the world)

Calibration from vanishing points

$$\mathbf{P} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{bmatrix}$$

- Decide (w.l.o.g.) that the world coordinates are axis-aligned with the three directions you have
- $\mathbf{p}_1 = \mathbf{P} \cdot (1, 0, 0, 0)$: the vanishing point in the x direction (in the world)
- Similarly for $\mathbf{p}_2$, $\mathbf{p}_3$ for y, z

Calibration from vanishing points

$$\mathbf{P} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{bmatrix}$$

- Decide (w.l.o.g.) that the world coordinates are axis-aligned with the three directions you have
- $\mathbf{p}_1 = \mathbf{P} \cdot (1, 0, 0, 0)$: the vanishing point in the x direction (in the world)
- Similarly for $\mathbf{p}_2$, $\mathbf{p}_3$ for y, z
- $\mathbf{p}_4 = \mathbf{P} \cdot (0, 0, 0, 1)$: projection of the origin (of the world system)
- But: only known up to scale (independent per column)...

$$\mathbf{P} = [a\mathbf{v}_x \quad b\mathbf{v}_y \quad c\mathbf{v}_z \quad \mathbf{o}]$$

Calibration from vanishing points

- Recall (assuming square pixels, and f in pixel units):

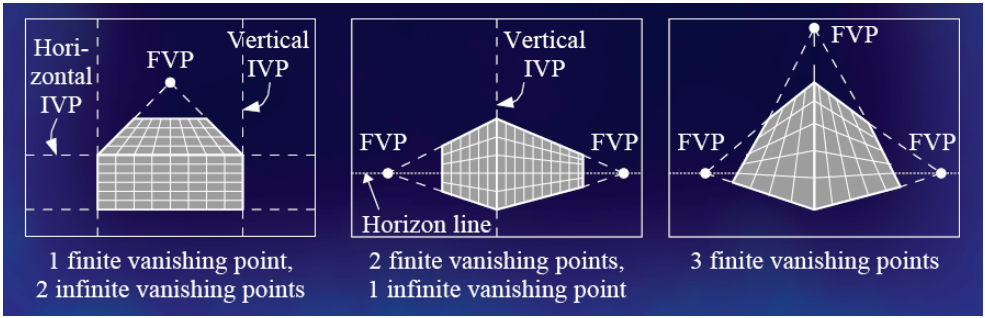
$$\mathbf{K} = \begin{bmatrix} f & 0 & \beta_x \\ 0 & f & \beta_y \\ 0 & 0 & 1 \end{bmatrix}$$

- Can show:

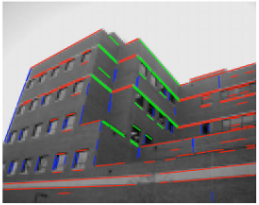
$$\mathbf{v}_i^T \mathbf{K}^{-T} \mathbf{K}^{-1} \mathbf{v}_j = 0$$

  i.e., each pair of vanishing points gives a constraint on $\mathbf{K}$
- Three constraints, three unknowns

Calibration from vanishing points



Cannot recover f; principal point is the 3rd vanishing point



Can solve for f, principal point

Calibration from vanishing points

- Let $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ be the three orthogonal (vanishing) directions in the scene
- Constraints: $\lambda_i \mathbf{v}_i = \mathbf{K}\mathbf{R}\mathbf{e}_i$
- Once we have $\mathbf{K}$:

$$\lambda_i \mathbf{K}^{-1}\mathbf{v}_i = \mathbf{R}\mathbf{e}_i$$

- Let

$$\mathbf{R} = [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3],$$

then $\mathbf{R}\mathbf{e}_i = \mathbf{r}_i$

Calibration from vanishing points

- Let $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ be the three orthogonal (vanishing) directions in the scene
- Constraints: $\lambda_i \mathbf{v}_i = \mathbf{K}\mathbf{R}\mathbf{e}_i$
- Once we have $\mathbf{K}$:

$$\lambda_i \mathbf{K}^{-1}\mathbf{v}_i = \mathbf{R}\mathbf{e}_i$$

- Let

$$\mathbf{R} = [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3],$$

then $\mathbf{R}\mathbf{e}_i = \mathbf{r}_i \quad \Rightarrow \mathbf{r}_i = \lambda_i \mathbf{K}^{-1}\mathbf{v}_i$

Calibration from vanishing points

- Let $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ be the three orthogonal (vanishing) directions in the scene
- Constraints: $\lambda_i \mathbf{v}_i = \mathbf{K}\mathbf{R}\mathbf{e}_i$
- Once we have $\mathbf{K}$:

$$\lambda_i \mathbf{K}^{-1}\mathbf{v}_i = \mathbf{R}\mathbf{e}_i$$

- Let

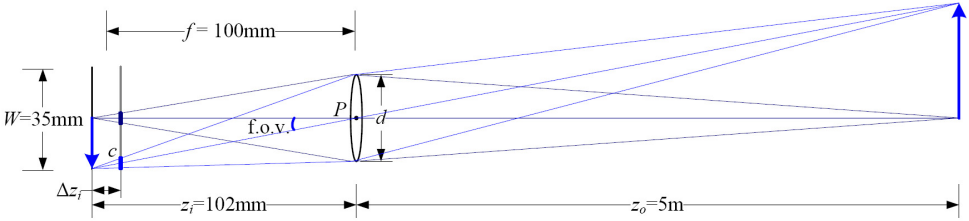
$$\mathbf{R} = [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3],$$

  then $\mathbf{R}\mathbf{e}_i = \mathbf{r}_i \quad \Rightarrow \mathbf{r}_i = \lambda_i \mathbf{K}^{-1}\mathbf{v}_i$
- Finally, get λ_i from $\|\mathbf{r}_i\|^2 = 1$

Calibration from vanishing points: summary

- Step 1: solve for $\mathbf{K}$ from vanishing points
- Step 2: get $\mathbf{R}$ from vanishing points using known $\mathbf{K}$
- Properties:
  - Could be automated (will talk later about line detection)
  - Extrinsics w.r.t. arbitrary coordinate frame
  - Only works when observe 3 orthogonal directions in the scene
  - Noisy (depends on calculating vanishing points)
  - Problems when vanishing points are at infinity
  - Later: calibration with a known set of points ("calibration object")

The thin lens model



- f focal length, z_o distance to object plane, z_i distance to focal plane

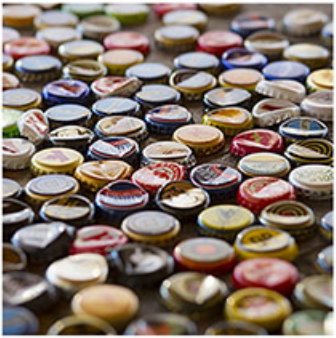
$$\frac{1}{z_o} + \frac{1}{z_i} = \frac{1}{f}$$

  when $z_o \to \infty$, $f \approx z_i$ (like in pinhole model)
- Moving focal plane by Δz_i moves image out of focus
  creates *circle of confusion* c
- Field of view determined by sensor size W vs. aperture size d
- F-number $f/\# = \frac{f}{d}$

Depth of field

- Depth of field is the range of distances to the camera (depths) in which objects are (approximately) in focus

f/1.8



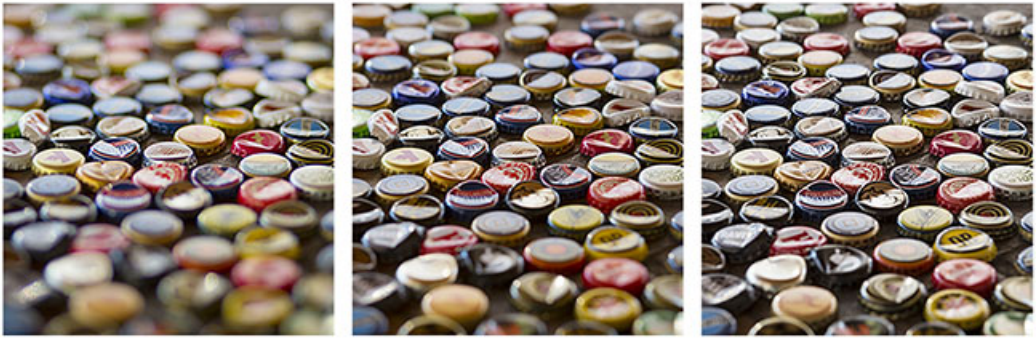
f/4.5

f/22

src: www.better-digital-photo-tips.com

Depth of field

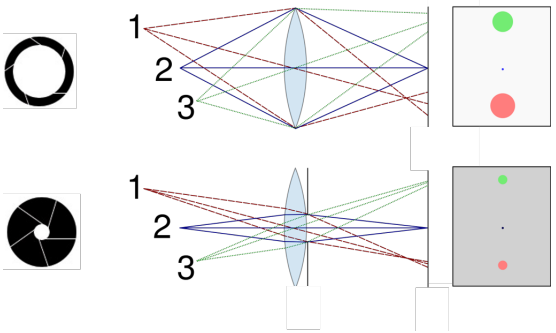
- Depth of field is the range of distances to the camera (depths) in which objects are (approximately) in focus



src: www.better-digital-photo-tips.com

- Why would it depend on the f/#?

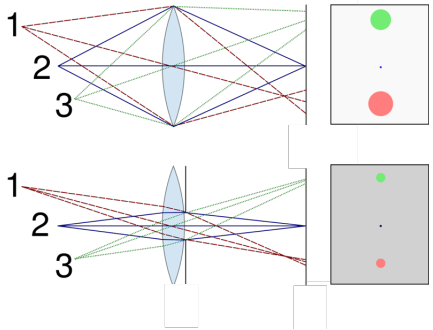
Depth of field



Depth of field

- Smaller aperture = larger depth of field
- Smaller aperture = less light; need to increase exposure!

src: Wikipedia



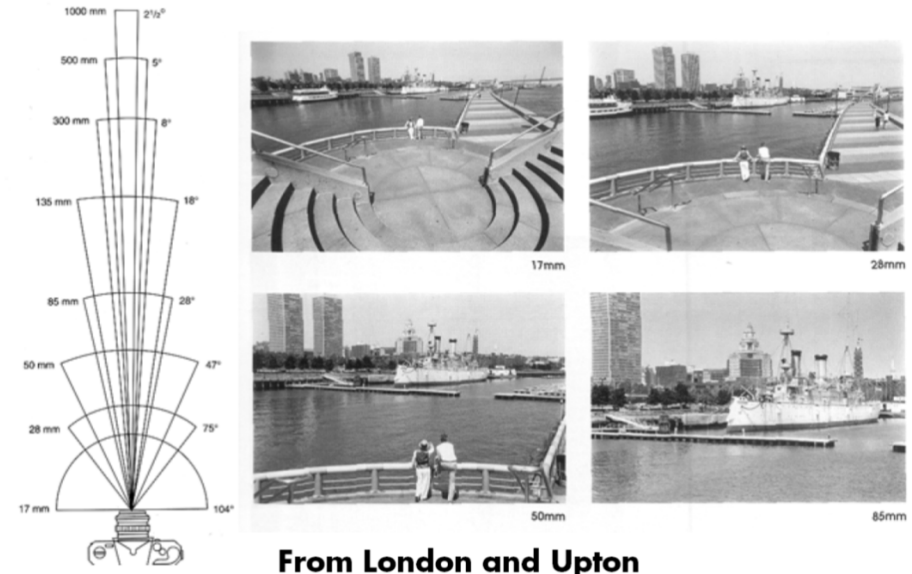
small aperture

large aperture

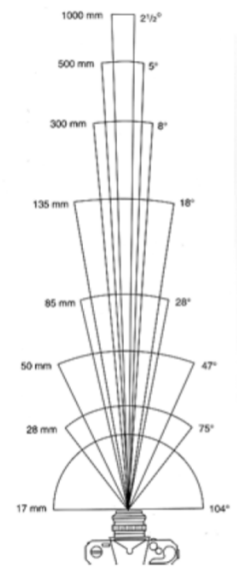
src: A. Efros

Field of view

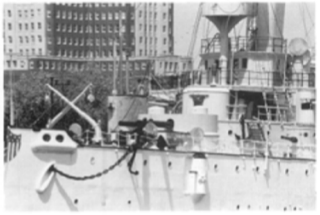
- Field of view: the angular extent of the world observed by the camera src: A. Efros



Field of view



135mm



300mm

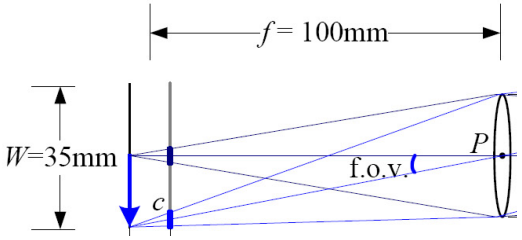
From London and Upton

Field of view

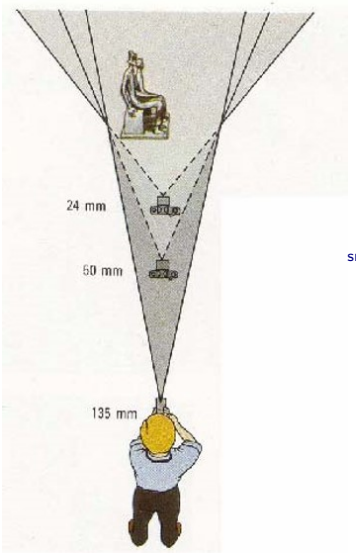
- FoV depends on f and W

$$\theta = \tan^{-1}\left(\frac{W}{2f}\right)$$

- Increase f = decrease FoV



Field of view and perspective



src: F. Durand

Large FoV/small f
Close to camera

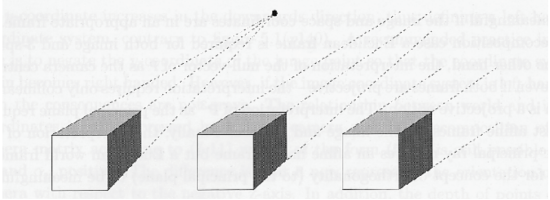
Small FoV/large f
Far from camera

Field of view and perspective

- Recall (Lecture 2): telephoto lenses (large f, small FOV) approximate weak perspective (scaled orthographic) projection

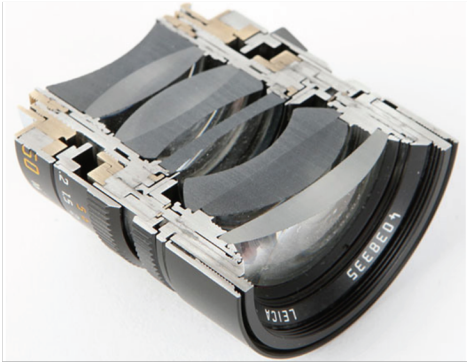
Wide angle Standard Telephoto

src: F. Durand



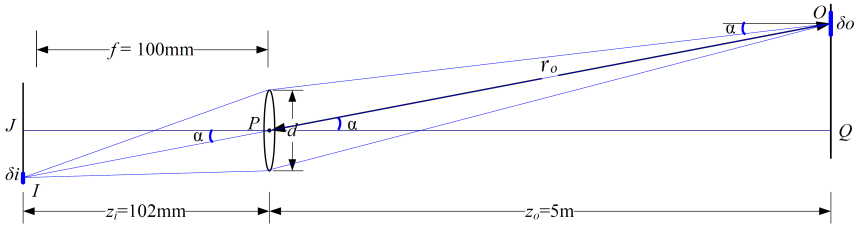
Real lenses in cameras

- The thin lens model is a primitive approximation of the real (compound) lenses in modern cameras



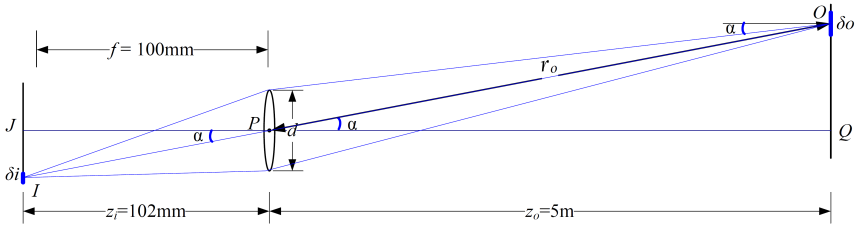
Real lenses: vignetting

- Vignetting: tendency of the brightness to fall off away from image center



Real lenses: vignetting

- Vignetting: tendency of the brightness to fall off away from image center



- Amount of light depends (among other factors) on angle α

src: F. Durand

Vignetting correction

- Correction: lighten corners (need to specify parameters)

src: juzaphoto.com

Vignetting correction

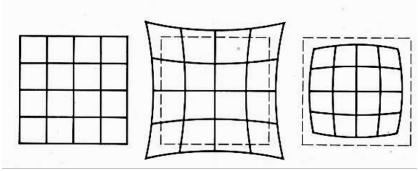
- Correction: lighten corners (need to specify parameters)

src: juzaphoto.com

- Besides *natural* vignetting, there is also *mechanical* vignetting (obstruction by internal hardware in the lens) and *optical* vignetting in multiple lens assemblies (loss of light intensity towards the periphery due to shading of rear lenses by forward ones)

Real lenses: radial distortion

- Real (wide-angle) lenses distort relative to pinhole model: lines no longer preserved



barrel

pin cushion

fisheye

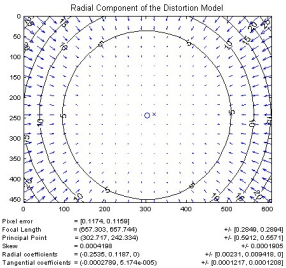
src: R. Szeliski

Radial distortion: modeling and correction

- Pinhole model: $x = f\frac{X}{Z}$
- Radial distortion: $x' = \lambda x$,

$$\lambda = 1 + k_1 r^2 + k_2 r^4 + \dots$$

where $r^2 = x^2 + y^2$

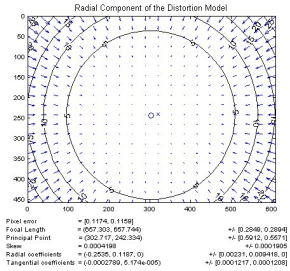


Radial distortion: modeling and correction

- Pinhole model: $x = f\frac{X}{Z}$
- Radial distortion: $x' = \lambda x$,

$$\lambda = 1 + k_1 r^2 + k_2 r^4 + \dots$$

where $r^2 = x^2 + y^2$



- Many techniques for estimating parameters and undistorting (rectifying) images

src: D. Ramanan

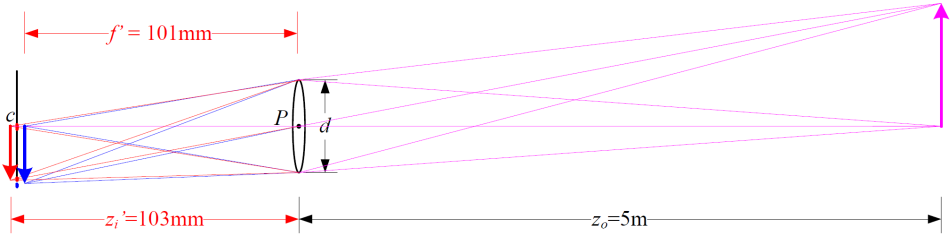
Radial distortion correction

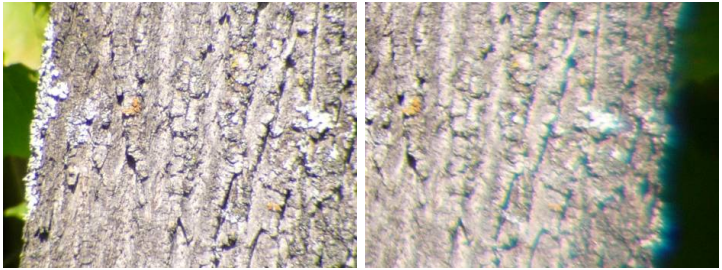
- Note: usually requires sacrificing pixels near the border

src: juzaphoto.com

Real lenses: chromatic aberration

- Actual focusing distance varies with light wavelength





src: Lana Lazebnik

Digital camera artifacts

Noise

- low light is where you most notice noise
- light sensitivity (ISO) / noise tradeoff
- stuck pixels

In-camera processing

- oversharpening can produce halos

Compression

- JPEG artifacts, blocking

Blooming

- CCD charge overflowing into neighboring pixels

Color artifacts

- Color moire
- Purple fringing from microlenses

src: Lana Lazebnik