Computer Vision
Cameras

dsai.asia

Asia Data Science and Artificial Intelligence Master's Program

Co-funded by the Erasmus+ Programme of the European Union

Readings

Readings for these lecture notes:

- Hartley, R., and Zisserman, A. *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2004, Chapter 6-8.

Outline

A camera maps a 3D object space to a 2D image.

We focus on cameras that perform central projection, for which there are several camera models, each represented by a matrix and each a specialization of the general projective camera.

There are two main kinds of cameras — those with a finite center and those with a center at infinity. The main infinite camera is the affine camera.

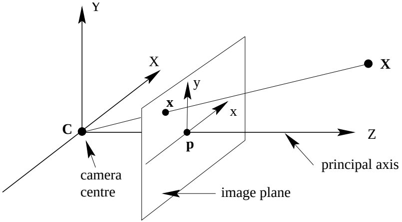
Outline

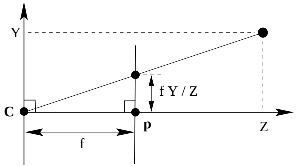
Finite cameras
The basic pinhole model

The pinhole camera uses central projection of points onto a plane.

The camera center or optical center, is the center of projection and the origin of a Euclidean coordinate system.

The image plane or focal plane is the plane $Z = f$.





Hartley and Zisserman (2004), Fig. 6.1

A few definitions:

- The principal axis is the axis orthogonal to the image plane intersecting the origin.
- The principal point is the intersection of the principal axis with the image plane.
- The principal plane is the plane through the camera center parallel to the image plane.

The transformation from a point in 3-space is just

$$(X, Y, Z)^T \mapsto (fX/Z, fY/Z)^T$$

In homogeneous coordinates, this is a linear transform:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}.$$

We write this compactly as

$$\mathtt{x} = \mathtt{P}\mathtt{X}$$

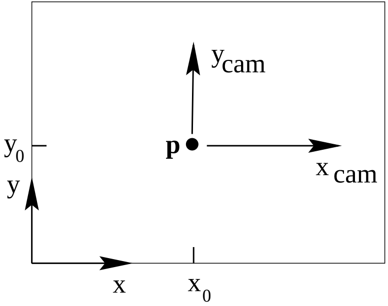
where

$$\mathtt{P} = \mathrm{diag}(f, f, 1) \left[\mathtt{I} \mid 0\right].$$

If the coordinate system in the image plane is not centered at the principal point, we write

$$(X, Y, Z)^T \mapsto (fX/Z + p_x, fY/Z + p_y)^T$$

where $(p_x, p_y)^T$ are the coordinates of the principal point.



Hartley and Zisserman (2004), Fig. 6.2

Now we write the transformation as

$$\mathtt{x} = \mathtt{K}\left[\mathtt{I} \mid 0\right] \mathrm{X}_{\mathrm{cam}}.$$

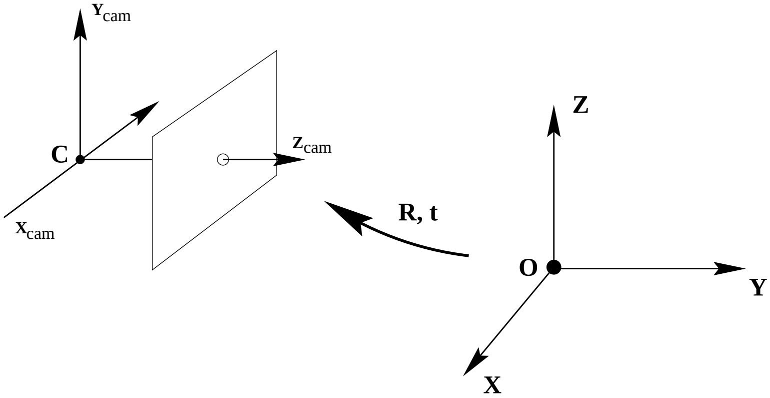
where $\mathtt{K}$, called the camera calibration matrix is

$$\mathtt{K} = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}$$

The notation $\mathrm{X}_{\mathrm{cam}}$ emphasizes that X is a 3D point in the camera coordinate frame.

The camera coordinate frame is a coordinate system whose origin is at the camera center and whose Z axis is the principal axis of the camera.

Now suppose our camera is rotated and translated with respect to a world coordinate frame:



Hartley and Zisserman (2004), Fig. 5.3

Suppose the 3D point $\tilde{\mathrm{C}}$ is the camera center in the world coordinate frame. Then we write

$$\mathrm{X}_{\text{cam}} = \begin{bmatrix} \mathtt{R} & -\mathtt{R}\tilde{\mathrm{C}} \\ 0^T & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} \mathtt{R} & -\mathtt{R}\tilde{\mathrm{C}} \\ 0^T & 1 \end{bmatrix} \mathrm{X}.$$

Putting the rigid transformation together with the camera projection gives us

$$\mathrm{x} = \mathtt{KR}\left[\mathtt{I} \mid -\tilde{\mathrm{C}}\right] \mathrm{X}$$

We write the general pinhole camera

$$P = KR[I \mid -\tilde{C}],$$

a matrix with 9 degrees of freedom (6 for the rigid transform, two for the principal point, and 1 for the focal length f).

The matrix K is said to contain the intrinsic parameters of the camera.

R and $\tilde{C}$ are the extrinsic parameters of the camera.

Usually we don't bother to make the camera center explicit and write

$$P = K[R \mid t]$$

where $t = -R\tilde{C}$.

On real-world cameras such as CCDs, the sensor cells may not be square, so we define a separate horizontal and vertical focal length α_x and α_y.

For added generality, we can consider camera where the horizontal and vertical axis are not orthogonal, introducing a skew parameter s.

With these modifications, we have the general finite projective camera

$$K = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}.$$

Finite cameras

General finite projective cameras

The general finite projective camera $\mathtt{P} = \mathtt{K}[\mathtt{R} \mid \mathtt{t}]$, then, has 11 degrees of freedom: 6 for the rigid transform, 2 for the principal point, 2 for the focal lengths, and 1 for the skew.

11 is also the number of DOF in a homogeneous rank 3 3×4 matrix.

Some properties of general finite cameras:

- Since $\mathtt{R}$ is orthogonal and $\mathtt{K}$ is necessarily invertible, the left 3×3 submatrix $\mathtt{M}$ of $\mathtt{P}$ must be non-singular.
- Any 3×4 matrix $\mathtt{P}$ with non-singular left-hand 3×3 submatrix $\mathtt{M}$ can be written in the form $\mathtt{K}[\mathtt{R} \mid \mathtt{t}]$ using the RQ factorization.
- The set of finite projective cameras is identical to the set of 3×4 matrices with non-singular left 3×3 submatrices.

If we remove the restriction that the left submatrix must be non-singular (but keep the restriction that $\mathtt{P}$ is rank 3), we obtain the general projective camera. We now consider its properties.

Outline

The general projective camera $\mathtt{P}$ maps $\mathrm{x} = \mathtt{P}\mathrm{X}$.

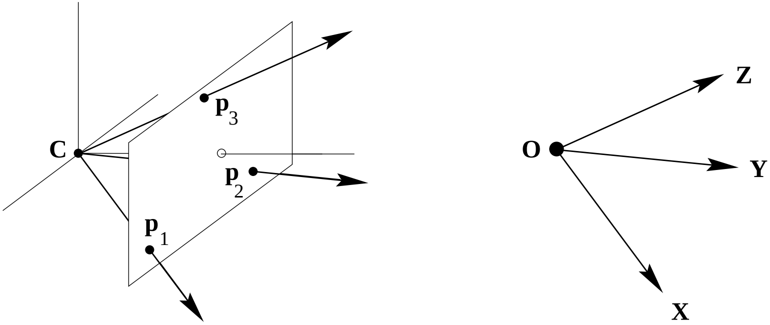
We divide $\mathtt{P}$ into blocks as $\mathtt{P} = [\mathtt{M} \mid \mathrm{p}_4]$.

If $\mathtt{P}$ is rank 3, it has a 1-D right null space generated by the 4-vector C, with $\mathtt{P}\mathrm{C} = 0$.

C represents the camera center in homogeneous coordinates, i.e., the point in $\mathbb{P}^3$ mapped to 0.

If C is a finite point in $\mathbb{P}^3$, i.e., $C_4 \neq 0$, the camera is finite; otherwise, the camera center is a point at infinity, and the camera is said to be infinite.

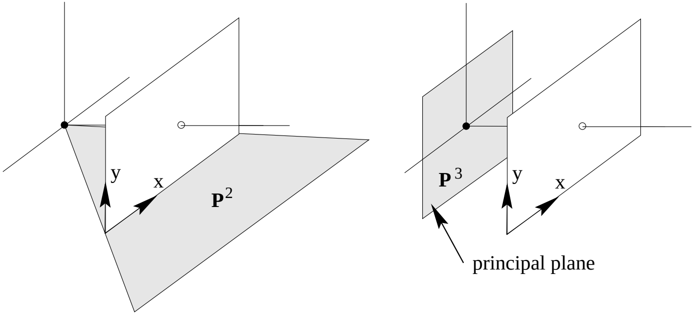
We obtain many other properties from the projective geometry of P. Its first three columns represent the vanishing points in the image of the world-coordinate X, Y, and Z axes.



Hartley and Zisserman, Fig. 6.4

The rows of P represent planes in $\mathbb{R}^3$:

- The third row is the principal plane (the plane through the camera center parallel to the image plane).
- The first and second rows represent the $\mathbb{R}^3$ plane corresponding to the lines $x = 0$ and $y = 0$ in the image, respectively.



Hartley and Zisserman, Fig. 6.5

The third row $\mathbf{m}^{3T}$ of $\mathtt{M}$ gives the direction of the principal axis.

The principal point is $\mathbf{x}_0 = \mathtt{M}\mathbf{m}^3$.

The backprojection of an image point x is the set of points in $\mathbb{R}^3$ mapping to that point.

To find the backprojection, we use the pseudoinverse:

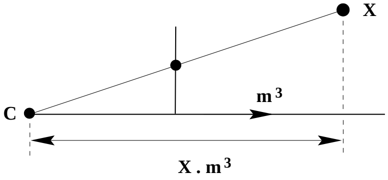
$$\mathrm{X}(\lambda) = \mathrm{P}^{+}\mathrm{x} + \lambda \mathrm{C}$$

(Recall that the line between two points in $\mathbb{R}^3$ is just the span of the two points in $\mathbb{P}^3$.)

If we have a camera matrix $[\mathtt{M} \mid \mathrm{p}_4]$ and project the point $\mathrm{X} = (X, Y, Z, 1)^T$ to $\mathrm{x} = \mathtt{PX} = w(x, y, 1)^T$, we obtain $w = \mathrm{m}^{3T}(\mathrm{X} - \tilde{\mathrm{C}})$,

w is the dot product of the ray $\mathrm{X} - \tilde{\mathrm{C}}$ with the principal ray direction.

If we normalize so that $\det \mathtt{M} > 0$ and $\|\mathrm{m}^3\| = 1$, then w is the depth of the point X from the camera center C in the direction of the principal ray.



Hartley and Zisserman, Fig. 6.6

In general, this means we can also write, given $\mathrm{X} = (X, Y, Z, T)^T$, $\mathtt{P} = [\mathtt{M} \mid \mathrm{p}_4]$, and $\mathtt{P}\mathrm{X} = w(x, y, 1)^T$, that

$$\mathrm{depth}(\mathrm{X}; \mathtt{P}) = \frac{\mathrm{sign}(\det \mathtt{M}) w}{T \|\mathrm{m}^3\|}$$

This can be a convenient way to test if an arbitrary point X is in front of an arbitrary camera $\mathtt{P}$ or not.

If $\mathtt{P}$ is a camera, we will often want to decompose it to obtain the intrinsic and extrinsic parameters explicitly.

To find the camera center C, we just obtain the right null vector of $\mathtt{P}$ as the last column of $\mathtt{V}$ in the SVD $\mathtt{UDV}^T = \mathtt{P}$.

If $\mathtt{P}$ is a finite camera, then $\mathtt{M}$ is non-singular, and we can find $\mathtt{KR} = \mathtt{M}$ using the RQ decomposition.

```
function [C,T,R,K] = decompose(P)
  % Extract camera geometry from finite camera matrix P
  [U,D,V]=svd(P);
  C = V(1:3,4)/V(4,4);
  [K,R] = rq(P(:,1:3));
  K = K/K(3,3);
  % If focal lengths come out negative, fix them
  fix_t = eye(3);
  if K(1,1) < 0, fix_t(1,1) = -1; end
  if K(2,2) < 0, fix_t(2,2) = -1; end
  K = K * fix_t;
  R = fix_t * R;
  % If R is oriented backwards, fix it
  if det(R) < 0
    R = -R;
  end;
  T = -R*C;
end
```

(Based on code by Rassarin Chinnachodteeranun, 2007)

The resulting calibration matrix $\mathtt{K}$ will have the form

$$\mathtt{K} = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$$

The skew s will normally be 0 for a real camera.

s can turn out to be non-zero under some transformations, e.g. when a rectifying homography $\mathtt{H}$ is applied to a real image and we re-decompose the effective camera matrix $\mathtt{HP} = \mathtt{K}[\mathtt{R} \mid \mathtt{t}]$.

The camera matrix $\mathtt{P}$ can be thought of as the composition of a 4×4 homography, a projection from $\mathbb{P}^3$ to $\mathbb{P}^2$, followed by a 3×4 homography:

$$\mathtt{P} = \mathtt{H}_{3\times 3} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathtt{H}_{4\times 4}.$$

For real-world cameras, the homographies will be Euclidean, but in practice we will allow general projectivities, and this will be useful at times.

Cameras at infinity are those for which $\mathtt{M}$ is singular. They may be divided into affine cameras and non-affine cameras.

There are four varieties of affine camera with different constraints on the form of $\mathtt{M}$:

- Orthographic projection
- Weak perspective
- General affine.

See text for details. We won't use them but they are useful in some 3D reconstruction methods.

There are other kinds of cameras not fitting our model, such as the line camera. See text for details.

Outline

Computing P

Why estimate P?

In many applications, we need to determine the camera matrix $\mathtt{P}$ that produced a given image.

- In camera calibration, we have a set of correspondences $\mathbf{x}_i \leftrightarrow \mathbf{X}_i$ and want to calculate $\mathtt{P}$ from the correspondences.
- In restricted camera estimation we have $\mathbf{x}_i \leftrightarrow \mathbf{X}_i$ and some prior knowledge about $\mathtt{P}$, for example that it is a pinhole camera or that the skew is 0, and want to find the best $\mathtt{P}$ meeting these constraints.

Normally in vision algorithms we assume a true projective camera (linear in homogeneous coordinates).

This is invalid when we have lens distortion, so we need techniques to deal with distortion also.

In any case, once we have $\mathtt{P}$ we can obtain $\mathtt{K}$, $\mathtt{R}$, and $\tilde{\mathbf{C}}$ using RQ decomposition and the SVD, as already discussed.

We first derive the linear solution (the DLT) for $\mathtt{P}$.

The problem, to find, given a set of correspondences $\mathrm{X}_i \leftrightarrow \mathrm{x}_i$, is very similar to the DLT for homography estimation.

From the relation $\mathrm{x}_i = \mathtt{P}\mathrm{X}_i$ we derive

$$\begin{bmatrix} 0^T & -w_i \mathrm{X}_i^T & y_i \mathrm{X}_i^T \\ w_i \mathrm{X}_i^T & 0^T & -x_i \mathrm{X}_i^T \\ -y_i \mathrm{X}_i^T & x_i \mathrm{X}_i^T & 0^T \end{bmatrix} \begin{pmatrix} \mathrm{P}^1 \\ \mathrm{P}^2 \\ \mathrm{P}^3 \end{pmatrix} = 0. \tag{1}$$

As before, the 3 equations are linearly dependent, so we only use the first two equations (as long as $w_i \neq 0$).

We stack our $2n$ equations to obtain the linear system $\mathtt{A}\mathrm{p} = 0$.

Since we have 11 DOF, we can obtain an exact solution from $5\frac{1}{2}$ correspondences (2 equations for each of 5 correspondences and 1 for a 6th correspondence) so long as the points are in general position.[1]

The solution, as it was with $\mathtt{H}$, is just the 1-D right null space of $\mathtt{A}$.

In the over-determined case we can minimize the algebraic error $\|\mathtt{A}\mathrm{p}\|$ subject to $\|\mathrm{p}\|$ by taking the last column of $\mathtt{V}$ in the SVD $\mathtt{UDV}^T = \mathtt{A}$.

[1] In the case of $\mathtt{P}$ estimation, degeneracy occurs if the camera and 3D points lie on a twisted cubic or the points lie on the union of a plane and a straight line through the camera center.

As in homography estimation, data normalization is crucial for linear minimization of algebraic error:

- As before, we perform isotropic scaling so that the image points have mean 0 and average distance $\sqrt{2}$ from the origin.
- For the 3D points, we do the same, so that the average distance from the origin is $\sqrt{3}$.

The DLT for $\mathtt{P}$ is thus identical to the DLT for $\mathtt{H}$ except for the slightly different construction of the matrix $\mathtt{A}$.

The DLT for $\mathtt{P}$ can easily be extended to work with line correspondences instead of point correspondences (see text).

As before, we like the normalized DLT due to its simplicity and stability but we prefer an optimal solution according to maximum likelihood estimation.

Under the assumption of perfect measurement of 3D points X_i and Gaussian errors in the image, we obtain the maximum likelihood estimate of $\mathtt{P}$:

$$\hat{\mathtt{P}} = \operatorname*{argmin}_{\mathtt{P}} \sum_i d(\mathrm{x}_i, \mathtt{P}\mathrm{X}_i)^2$$

Once again, the minimization is a nonlinear least squares problem which can be solved iteratively by Levenberg-Marquardt. This leads to the Gold Standard algorithm for estimation of $\mathtt{P}$.

Here is the full Gold Standard algorithm for computing P when world points X_i are accurately known (Hartley and Zisserman, 2004, Algorithm 7.1).

Gold Standard algorithm for P: Objective

Given $n \geq 6$ world to image point correspondences $\{\mathrm{X}_i \leftrightarrow \mathrm{x}_i\}$, determine the maximum likelihood estimate of P minimizing $\sum_i d(\mathrm{x}_i, \mathrm{P}\mathrm{X}_i)^2$.

Gold Standard algorithm for P: Algorithm

(i) **Normalization:** Compute similarity transforms $\mathtt{T}$ and $\mathtt{U}$ normalizing $\{\mathrm{x}_i\}$ and $\{\mathrm{X}_i\}$ then compute $\tilde{\mathrm{x}}_i = \mathtt{T}\mathrm{x}_i$ and $\tilde{\mathrm{X}}_i = \mathtt{U}\mathrm{X}_i$.

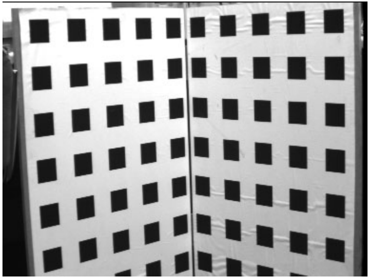
(ii) **Linear solution**: use the DLT approach to solve the linear system in Eq. 1 and obtain the $\tilde{\mathtt{P}}_0$ minimizing algebraic error.

(iii) **Geometric error minimization**: Beginning from $\tilde{\mathtt{P}}_0$, use Levenberg-Marquardt to find a new estimate $\tilde{\mathtt{P}}$ minimizing

$$\sum_i d(\tilde{\mathrm{x}}_i, \tilde{\mathtt{P}}\tilde{\mathrm{X}}_i)^2.$$

(iv) **Denormalization**: The camera matrix for the original (unnormalized) coordinates is obtained from $\tilde{\mathtt{P}}$ as

$$\mathtt{P} = \mathtt{T}^{-1}\tilde{\mathtt{P}}\mathtt{U}.$$



Hartley and Zisserman (2004), Fig. 7.1

We require a set of accurate 3D points not on the same plane.

Hartley and Zisserman got 197 corners with these steps:

(i) Canny edge detection
(ii) Straight line fitting
(iii) Intersecting the lines to obtain the imaged corners

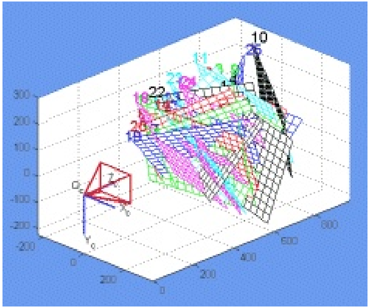
The pixel measurement accuracy was within 0.1 pixel.

The error $\sum_i d(\tilde{\mathrm{x}}_i, \tilde{\mathrm{P}}\tilde{\mathrm{X}}_i)^2$ was 0.365 for DLT only and 0.364 for the full Gold Standard algorithm.

If there is error in the 3D measurements, similar to computation of $\mathtt{H}$, we

- Simultaneously estimate $\hat{\mathrm{X}}_i$ and $\hat{\mathrm{x}}_i$ with $\hat{\mathrm{x}}_i = \mathtt{P}\hat{\mathrm{X}}_i$
- Use a Mahalanobis distance error measure reflecting our uncertainty in the measurements and the different units in image and 3D coordinates.

A planar calibration object can be used if multiple images are used.



(From the Caltech toolbox Web site)

The Caltech Camera Calibration Toolbox for Matlab (http://www.vision.caltech.edu/bouguetj/calib_doc/) is highly recommended. It extracts corners automatically and models radial distortion. It is part of OpenCV also.

If you do unconstrained camera calibration, you'll get non-zero skew estimates and principal point estimates not at the center of the image.

But sometimes we want to assume things like

- The skew $s = 0$
- The pixels are square $(\alpha_x = \alpha_y)$
- The principal point (x_0, y_0) is known
- $\mathtt{K}$ is completely known

These problems can be solved by obtaining the unrestricted camera with DLT then proceeding with a restricted parameterization using Levenberg-Marquardt.

See text for more efficient procedures for algebraic minimization with the common 9-parameter case of $s = 0, \alpha_x = \alpha_y$.

Hartley and Zisserman's experiments with 9-parameter restricted camera estimation obtained a residual of 0.601, compared to 0.364 for unrestricted estimation.

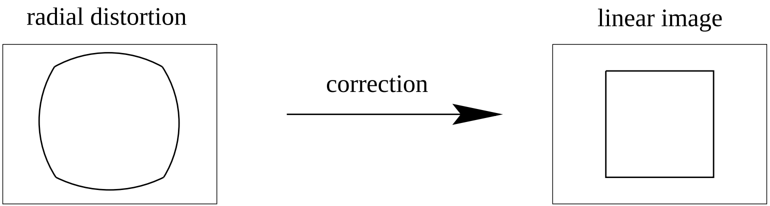
We see that less flexibility in the model means higher error, but probably values closer to the truth.

Outline

1. Introduction
2. Finite cameras
3. General cameras
4. Computing P
5. Radial distortion

Radial distortion

Cheap lenses (especially wide-angle lenses) introduce significant distortion into the image, so that the linear pinhole model no longer holds.



Hartley and Zisserman (2005), Fig. 6.5

To obtain accurate 3D information, we want to correct for distortion such as radial distortion.

Assume we have a point $\mathrm{X}_{\mathrm{cam}}$ in camera coordinates. Without distortion we have a corresponding ideal image point $(\tilde{x}, \tilde{y}, 1)^T = \mathtt{K}[\mathtt{I} \mid 0]\mathrm{X}_{\mathrm{cam}}$.

We model radial distortion by a radial displacement

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = \begin{pmatrix} x_c \\ y_c \end{pmatrix} + L_d(\tilde{r}) \begin{pmatrix} \tilde{x} - x_c \\ \tilde{y} - y_c \end{pmatrix}$$

where

- $(\tilde{x}, \tilde{y})$ is point's ideal image position
- (x_c, y_c) is the center of distortion
- (x_d, y_d) is the actual image position after distortion
- $\tilde{r}$ is the distance $\sqrt{\tilde{x}^2 + \tilde{y}^2}$ from the center of radial distortion
- $L_d(\tilde{r})$ is a distortion function, nonlinear in $\tilde{r}$

If we know the distortion parameters, we can correct the distortion by applying an undistortion function

$$\hat{x} = x_c + L_u(r_d)(x_d - x_c), \quad \hat{y} = y_c + L_u(r_d)(y_d - y_c)$$

where (x_d, y_d) are the measured coordinates, $(\hat{x}, \hat{y})$ are the corrected coordinates, (x_c, y_c) is the center of radial distortion, and $r_d^2 = (x_d - x_c)^2 + (y_d - y_c)^2$.

Note that the inverse distortion function $L_u(r) = 1/L_d(\tilde{r})$, but depending on the form of $L_d(\tilde{r})$, finding $\tilde{r}$ corresponding to r may not be easy.

The corrected coordinates $(\hat{x}, \hat{y})$ will obey a linear projective camera model.

Radial distortion

Even-order polynomail model

The most commonly used model is the even-order polynomial model

$$L_u(r_d) = 1 + \lambda_1 r_d^2 + \lambda_2 r_d^4 + \lambda_3 r_d^6 + \ldots \ .$$

$x_c, y_c, \lambda_1, \lambda_2, \lambda_3, \ldots$ are the distortion parameters, which must be estimated from image measurements.

Radial distortion

Division model

There have been objections to the even-order polynomial model.

According to some researchers, the model performs well for small distortion, but for severe distortion, a prohibitively large number of non-zero distortion parameters are required.

Fitzgibbon proposes an alternative model, the division model, as a more accurate approximation to the typical camera's true distortion function:

$$L_u(r_d) = \frac{1}{1 + \lambda_1 r_d^2 + \lambda_2 r_d^4 + \ldots}$$

An example of estimating the polynomial model distortion parameters using a cost function based on the straightness of imaged scene lines:

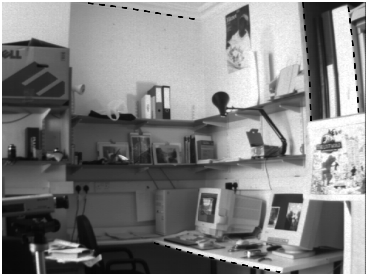


Image with radial distortion.

Undistorted version.

Hartley and Zisserman (2004), Fig. 6.6

Radial distortion

Example (division model)

An example of estimating the single-parameter division model based on automatic extraction and correction of circular arcs (distorted straight lines):

Image with radial distortion.

Undistorted version.

Bukhari, F. and Dailey, M.N, Automatic Radial Distortion from a Single Image, *Journal of Mathematical Imaging and Vision*, 45(1): 31-45, 2013. The distorted image is taken from `http://www.andromeda.com/people/ddyer/photo/wideangle.html`

Another example from Faisal's work:

Image with radial distortion.

Undistorted version.

The distorted image is taken from `http://www.flickr.com/photos/eirikso/with/3105820062/`

Radial distortion

Conclusion

Hartley and Zisserman found that the residuals in their camera calibration experiment dropped from 0.364 (unrestricted) and 0.601 (restricted) to 0.179 (unrestricted) and 0.380 (restricted) after radial distortion correction.

This improvement is for a camera with minimal distortion. On cheap cameras (e.g. Web cameras) correcting radial distortion is critical.

The Caltech toolbox automatically implements the polynomial model, automatically finding the distortion parameters along with estimation of $\mathtt{P}$.

The toolbox can undistort a given series of images or output an undistortion lookup table that you can plug into your own system.

The toolbox is not entirely plug-and-play! You must understand the image formation process and camera model, and you must ensure your measurements X_i and x_i are as accurate as possible.