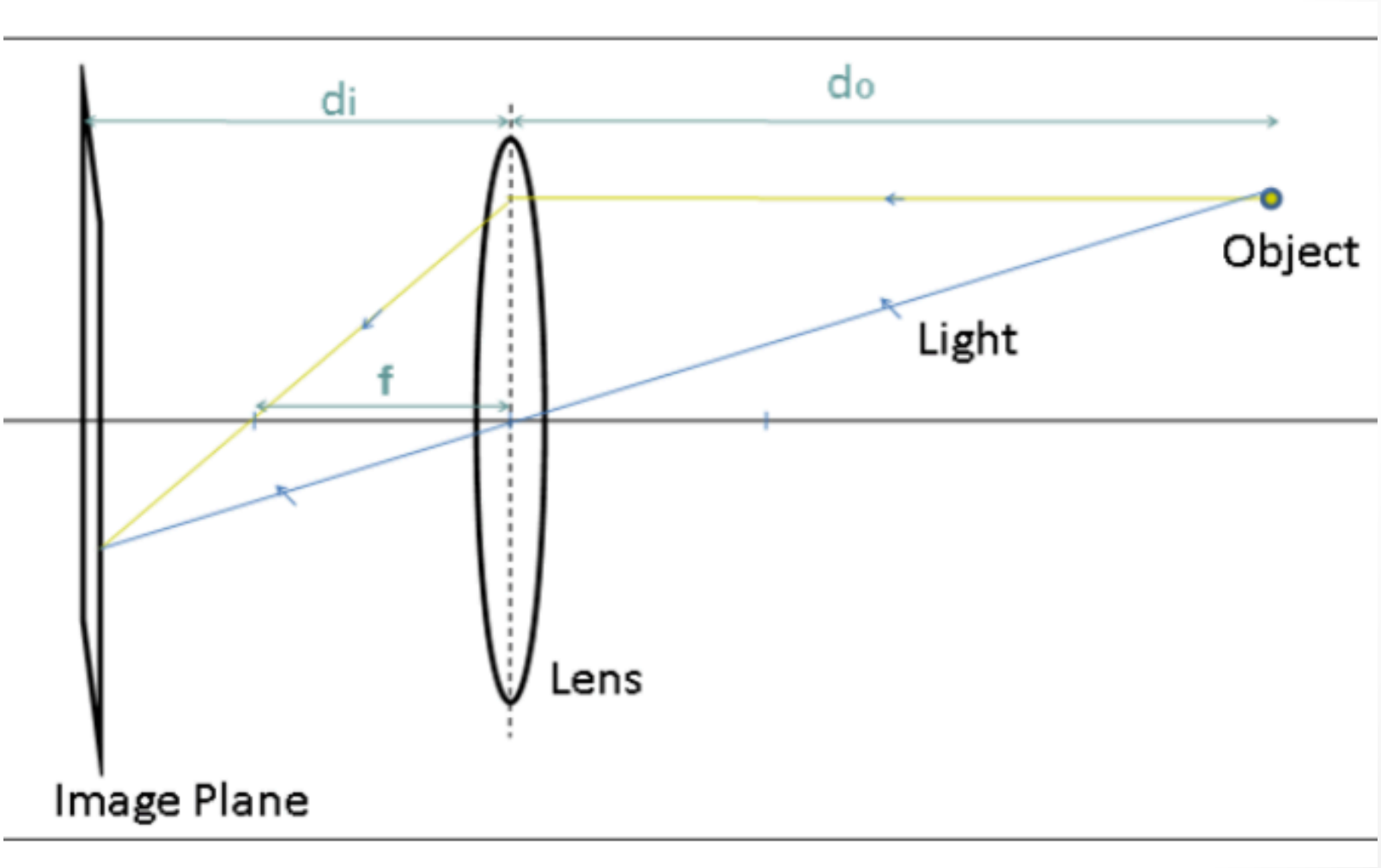
Modelo de Cameras e Calibração

Prof. Dr. Geraldo Braz Junior

Slides baseados em nas notas de aula de Fei-Fei li

Cameras reais



Distância Focal

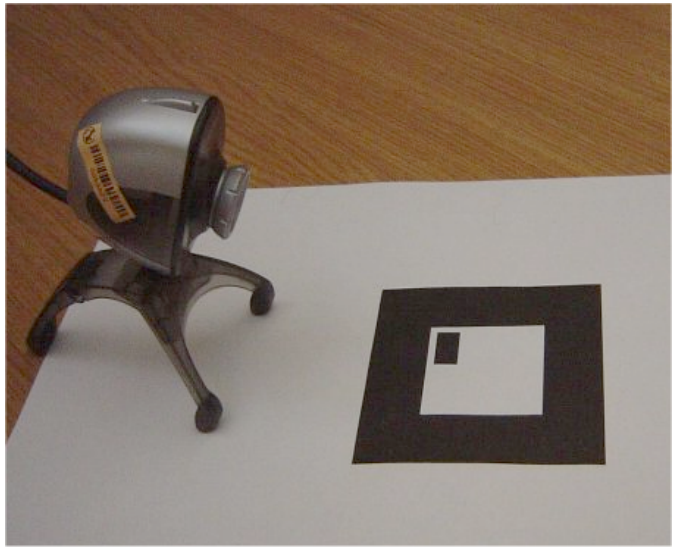
- **do** representa a distância da lente para o objeto
- **di** representa a distância da lente para o plano da imagem (distancia focal f)

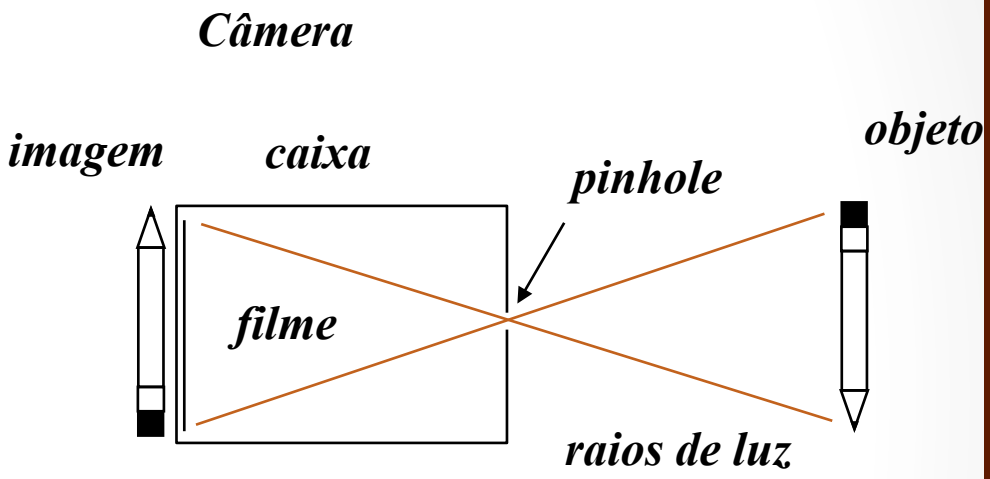
$$\frac{1}{f} = \frac{1}{do} + \frac{1}{di}$$

Simplificações

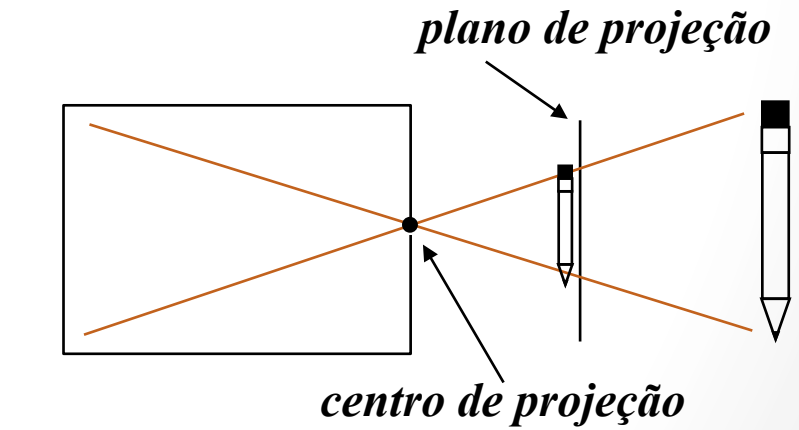
- do >> di
  - plano da imagem pode ficar localizado sobre a distância focal
- Simplificação: posiciona o plano da imagem sobre o plano da lente para não obter imagens invertidas
  - inverte o sinal de f

Pinhole



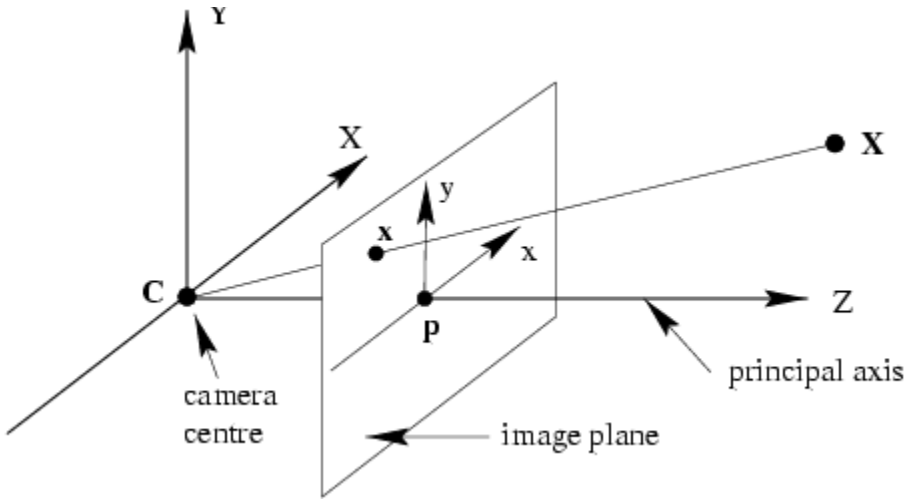


"pinhole"



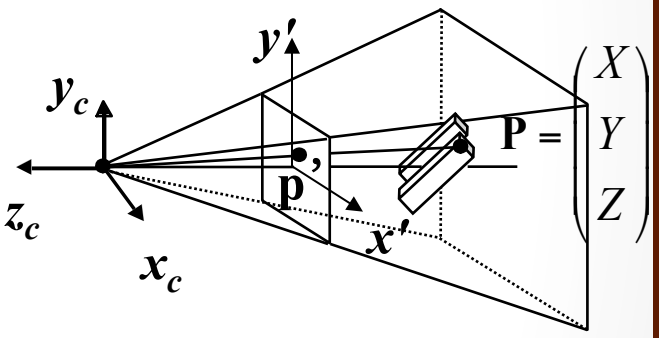
Notação

Visão



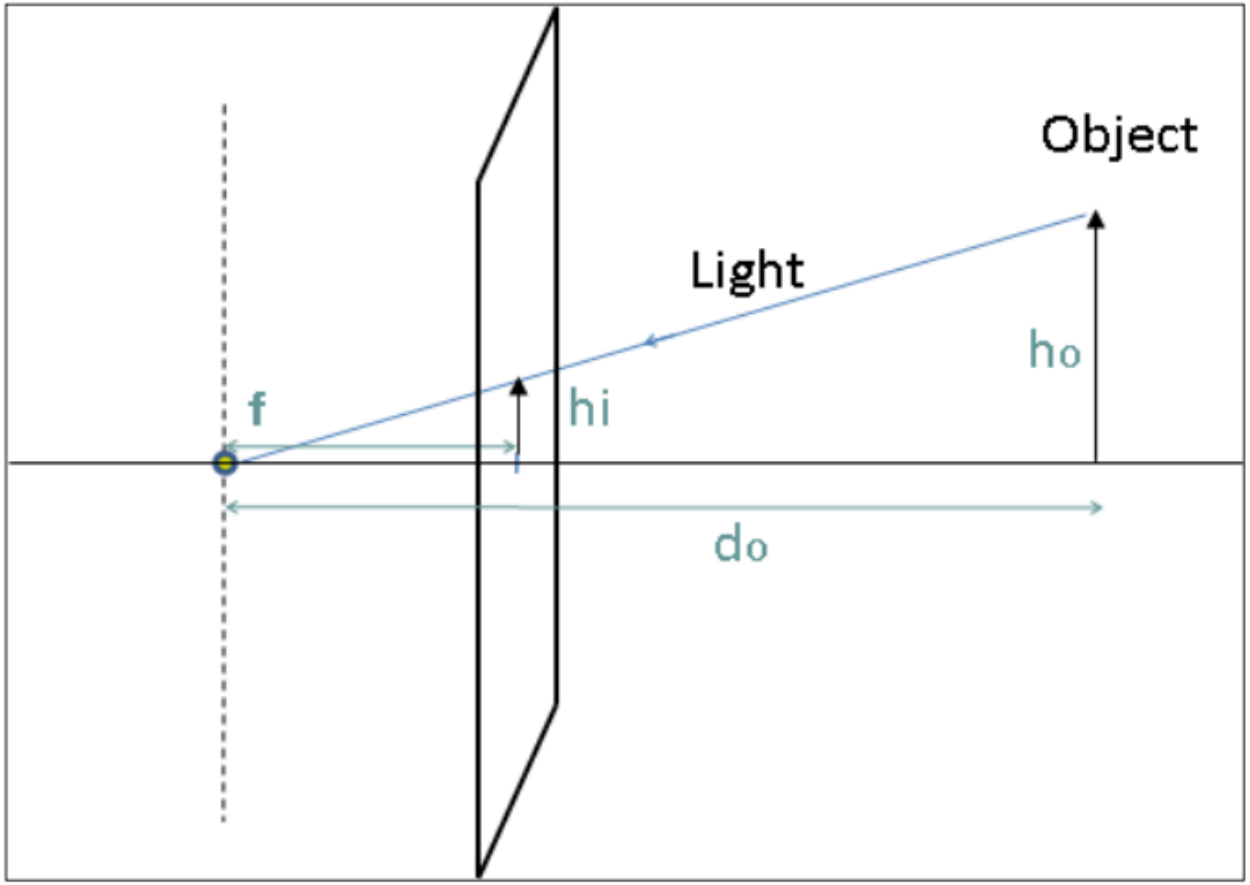
$$x = f\frac{X}{Z}, \quad y = f\frac{Y}{Z}$$

Modelo



$$x = -f\frac{X}{Z}, \quad y = -f\frac{Y}{Z}$$

Notação



$$hi = f\,\frac{ho}{do}$$

Como relacionar 3D -> 2D?

- Coordenadas homogêneas

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

homogeneous image coordinates

$$(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

homogeneous scene coordinates

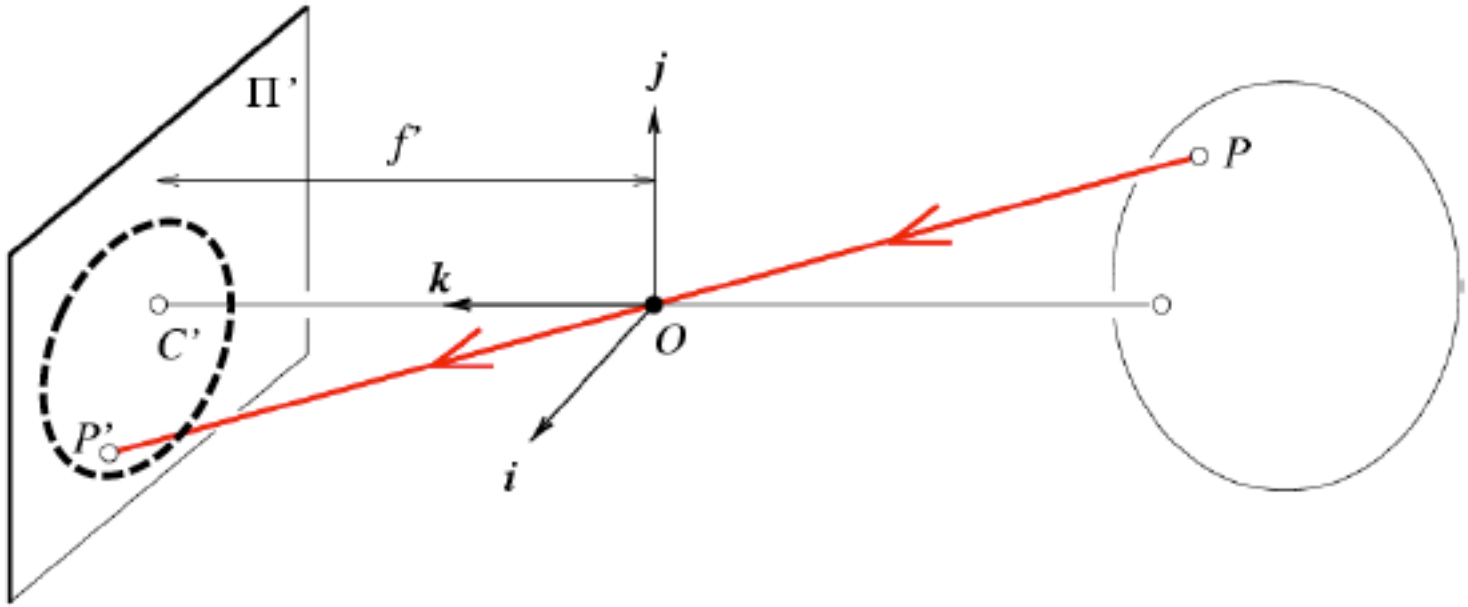
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

$$\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

Então precisa aplicar projeção!

In homogeneous coordinates:

$$P' = \begin{bmatrix} f\ x \\ f\ y \\ z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



M é a matriz ideal

- Assume que:
  - tamanho de pixel uniforme
  - centro ótico em (0,0)
  - sem distorções
  - sem rotações
  - camera em (0,0,0)

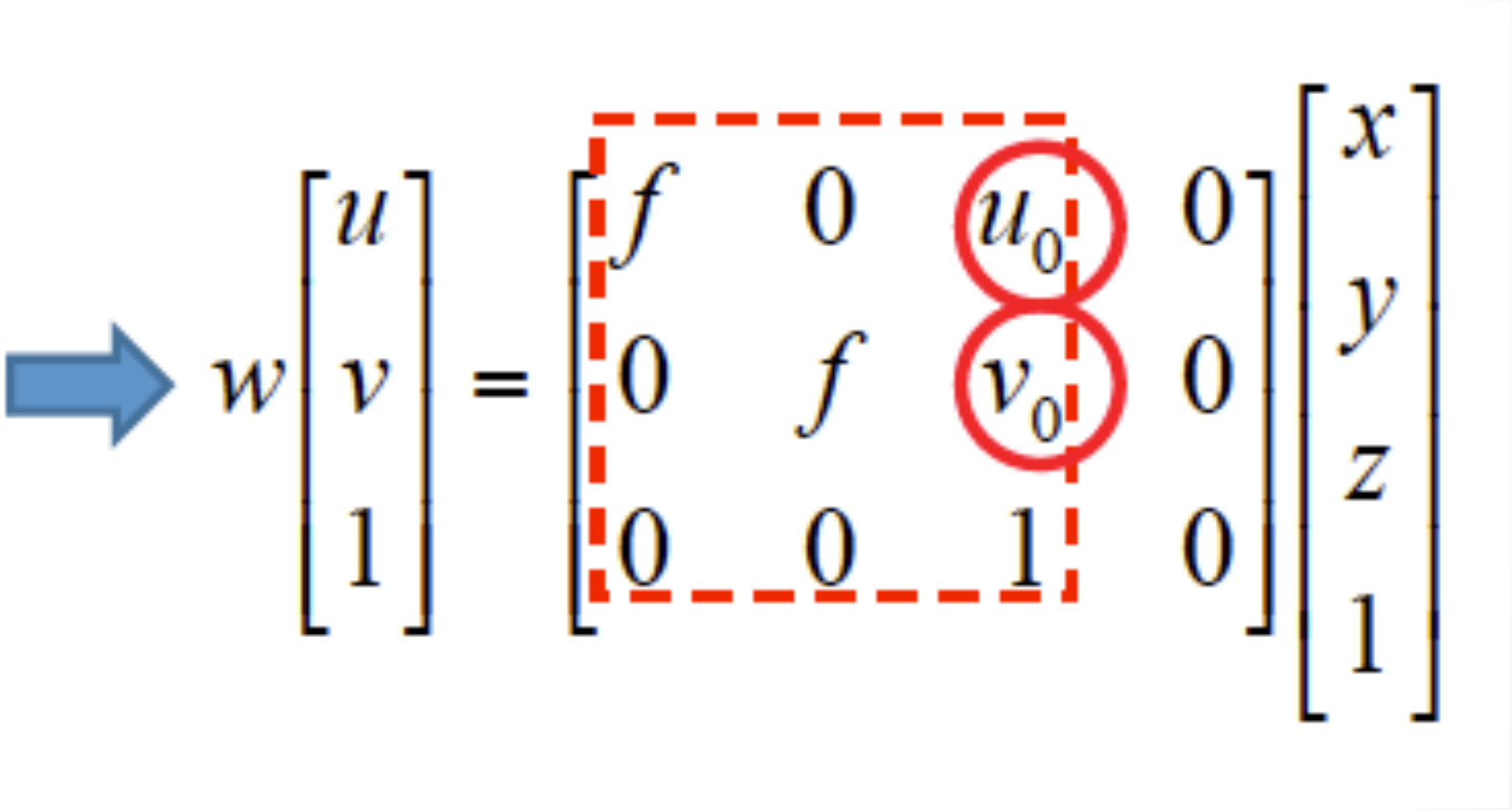
Analisando: decompondo

$$P' = \begin{bmatrix} fx \\ fy \\ z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = K \begin{bmatrix} I & 0 \end{bmatrix} P$$

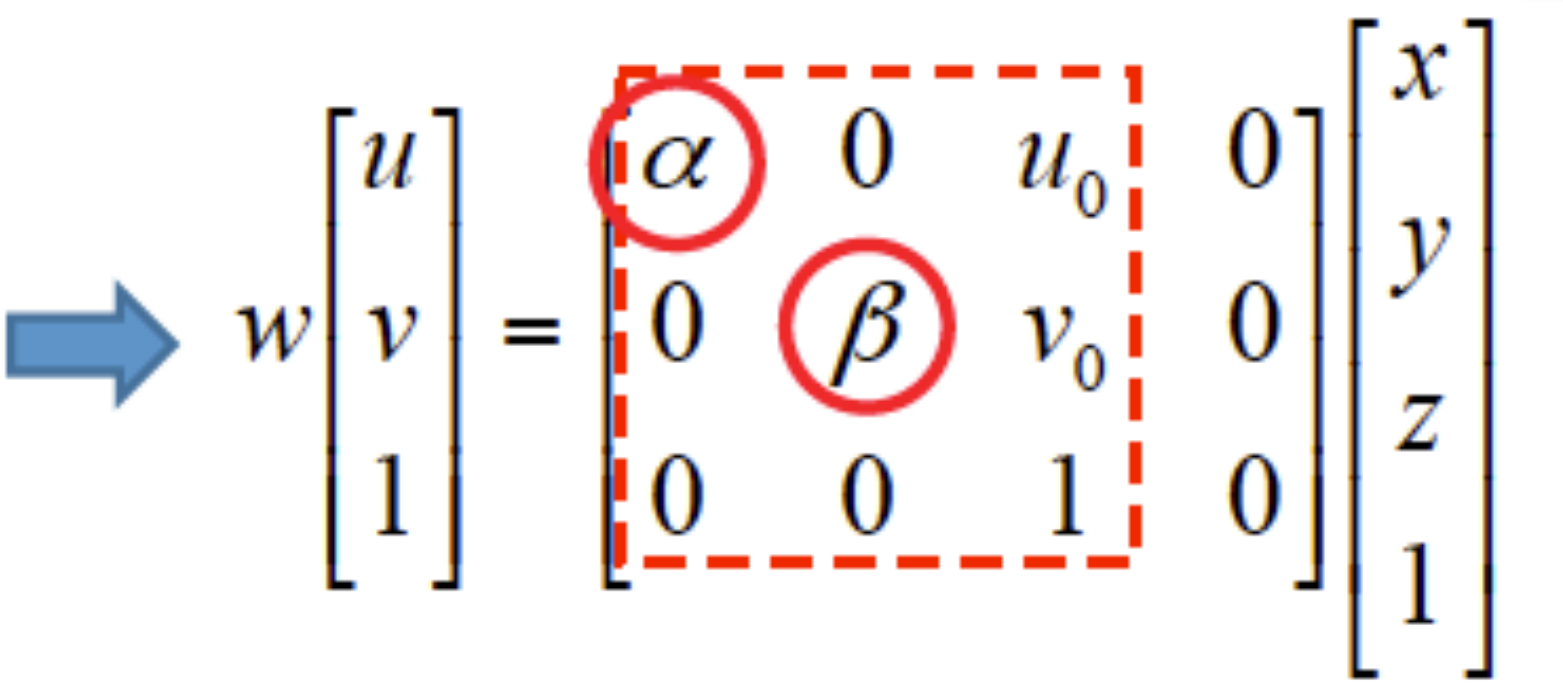
K

K é conhecido como parâmetros intrinsecos da câmera

Incluindo centro variável



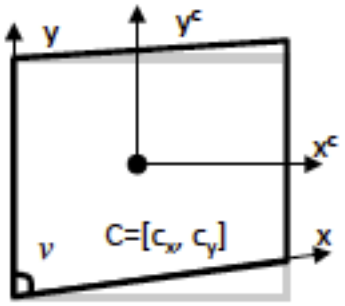
Pixels retangulares



$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & u_0 & 0 \\ 0 & \beta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Minimizando Distorção

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & s & u_0 & 0 \\ 0 & \beta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

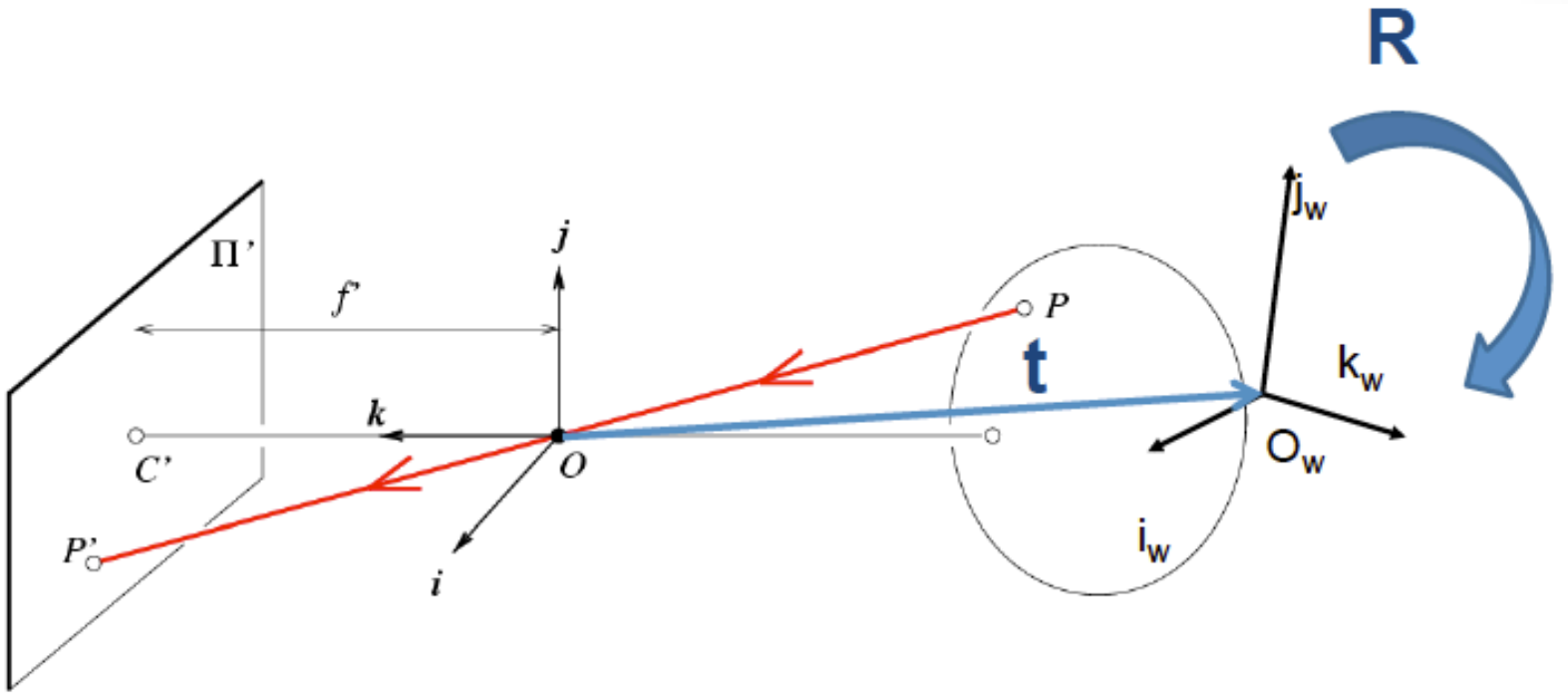


Parâmetros Intrinsecos

$$P' = K\begin{bmatrix} I & 0 \end{bmatrix} P \Rightarrow w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & s & u_0 & 0 \\ 0 & \beta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Intrinsic parameters

Parâmetros Extrínsecos: Rotação e Translação



Parâmetros Extrínsecos

$$P' = K\begin{bmatrix} R & \bar{t} \end{bmatrix} P \Rightarrow w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & s & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Como estimar?

- São ao todo 5+9+3 variáveis
- Simplificações aplicáveis?

Projeção ortográfica

- Aplicáveis de acordo com a aplicação
- distância infinita, projeção paralela

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

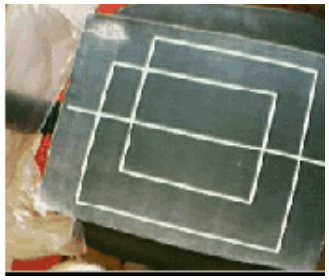
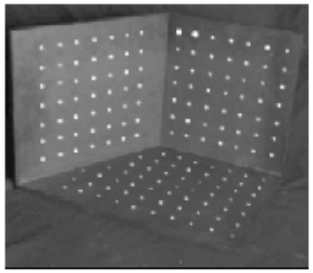
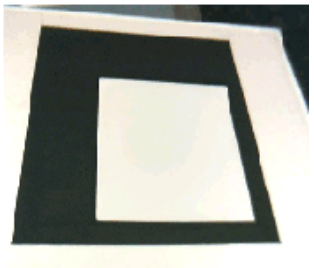
Projeção Ortográfica Escalada

- Dimensões dos objetos são pequenas se comparadas com a distância
- paralela

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 0 & s \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

E se não for ortogonal?

- Calibração de câmera
  - **Problema**: obter os parâmetros extrínsecos (**R**, **T**) e intrínsecos (**K**) da transformação projetiva de câmera.
  - **Dados**: *n* pares de pontos correspondentes (P_i, p_i) na cena e na imagem.


Calibração de câmeras

- Calibração ↔ estimação de parâmetros → otimização
- Alguns métodos: Tsai, Zhang

$$\min_{K,R,T} \sum | p_i - f_{K,R,T}(P_i) |^2$$

p_i: pontos da imagem

$f_{K,R,T}$: projeção (função não linear)

P_i: pontos da cena

Calibração com Opecv

- Usa uma função básica com o padrão de um tabuleiro de xadrez (marcador)
- Necessita de várias projeções (Zhang)

Detectando o tabuleiro

```
vector<cv::Point2f> imageCorners;
Size boardSize(6,4);
```

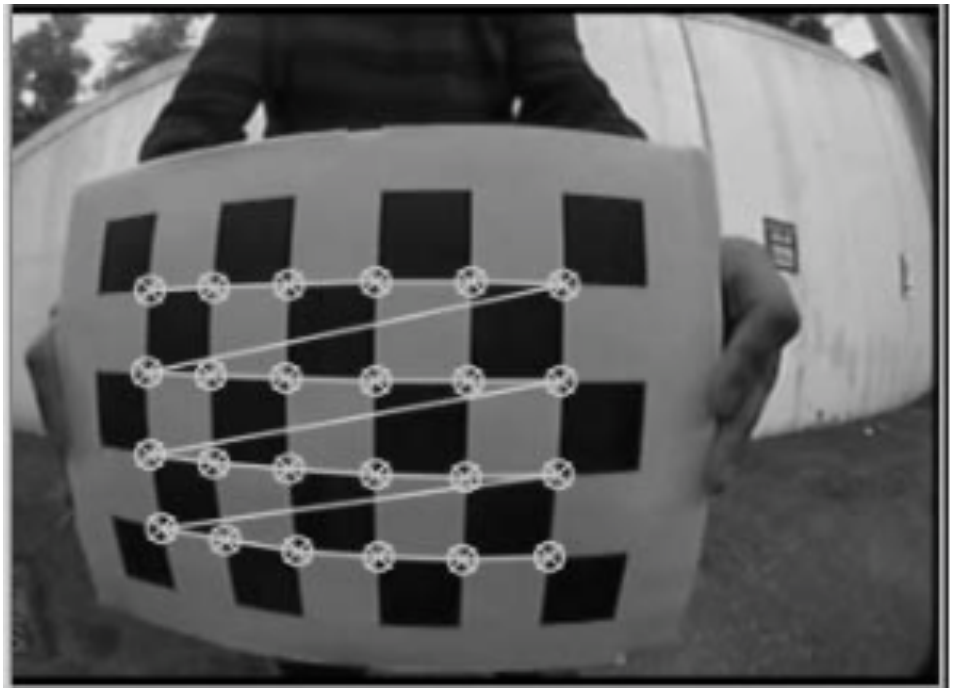
bool found = **findChessboardCorners**(image, boardSize, imageCorners);

-> imagem com o tabuleiro num ponto de vista

-> boardSize: quantos cantos deseja detectar

Detectando o tabuleiro

- **drawChessboardCorners** (image, boardSize, imageCorners, found)



Para calibrar

- Deve inserir um conjunto de imagens como esta

- Duas listas
  1. pontos num modelo ideal em 3D
  2. pontos localizados na imagem

- As duas configuram a correspondência do modelo para a imagem

Calibrando

vector<Mat> rvecs, tvecs;

calibrateCamera(

objectPoints, //3D

imagePoints, // pontos na imagem

imageSize, // tamanho da imagem

cameraMatrix, // output parâmetros intrinsecos

distCoeffs, // output matriz de distorção

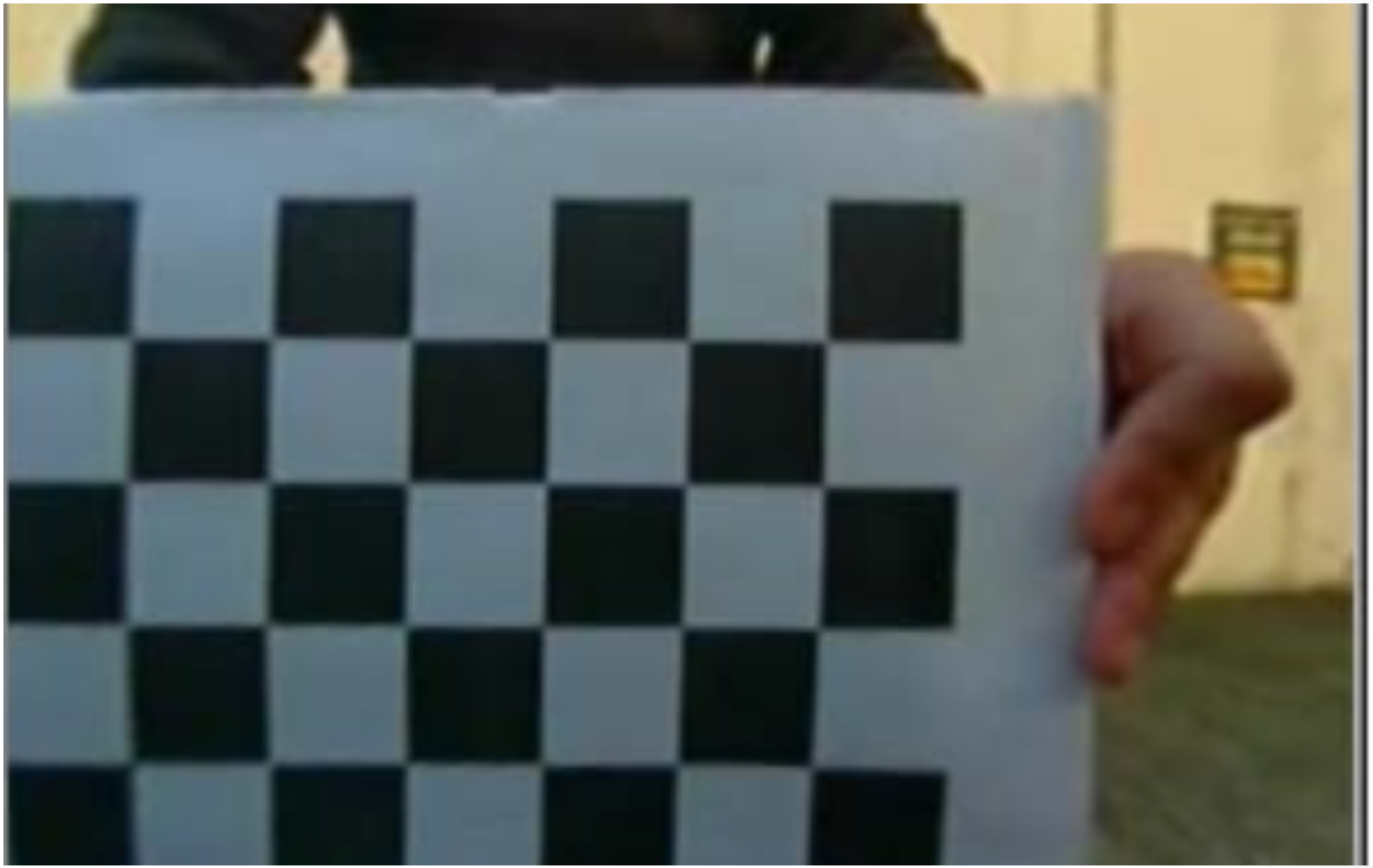
rvecs, tvecs, // parâmetros extrínsecos

flag); //

Reduzindo distorção

initUndistortRectifyMap(
cameraMatrix, // parâmetros intrinsecos
distCoeffs, // parâmetros de distorção
cv::Mat(), // opcional
cv::Mat(), // opcional
image.size(), // tamanho da imagem distorcida
CV_32FC1, // tipo de saida
map1, map2); // funções de mapeamento para x e y

remap(image, undistorted, map1, map2,
cv::INTER_LINEAR); // tipo de interpolação



Outras abordagens – Use keydetector

Some o Opengl