Imprecision of calibrated Cameras

Paul G .Plöger

SEE course

Vers 1.1 , 26.4.2015

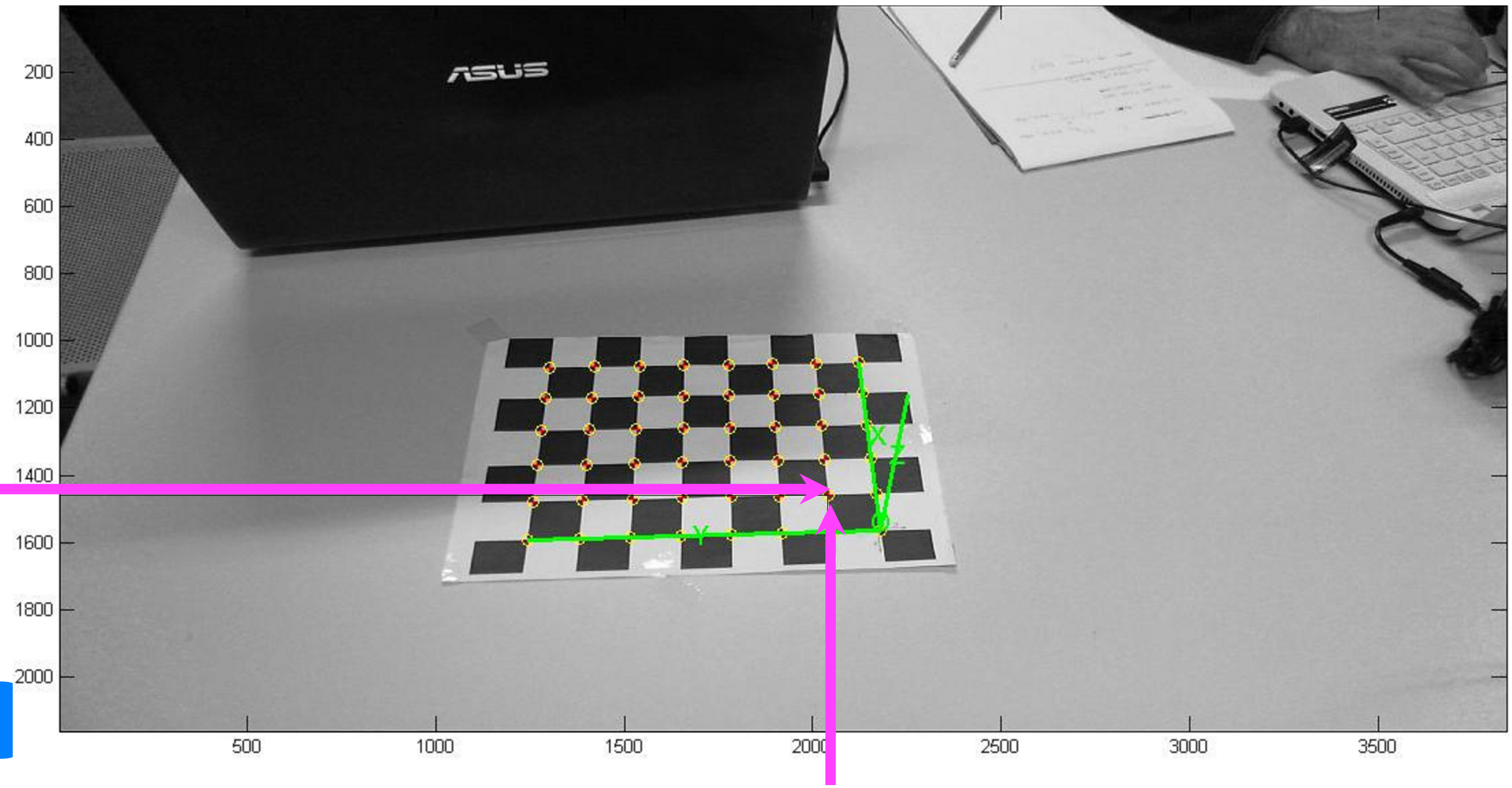
Problem description

Given:

- a 2d image with:
  - a coordinate system
  - an **observed point** in the image
- a camera (modeled by a matrix ***P***)

Tasks:

- A: Determine the observed point in 3d
- B: And: how precise is this ???



Formalization

Given:

image ***I***, point ***x*** ε ***I***, homogeneous coordinates

x = [row;col;1], e.g [2039;1459;1] (see right =>)

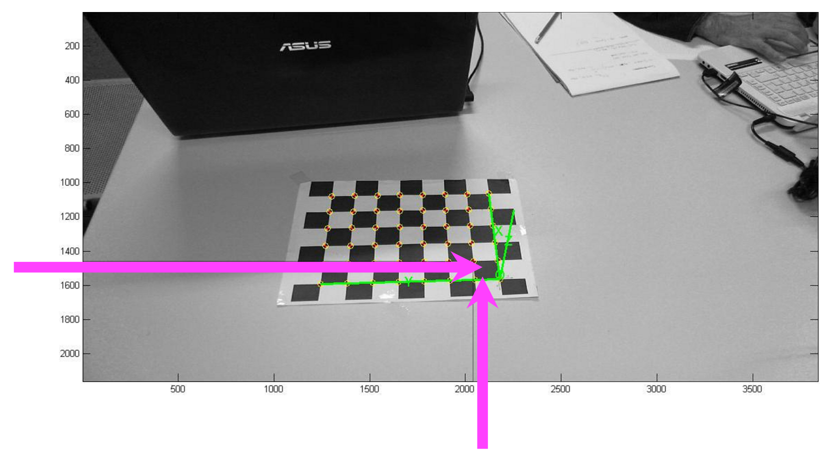
a camera model:

a matrix ***P*** (4x3) with parts:

- ***K* intrinsic parameters** (matrix (3x3), homogeneous)
- ***R*, *C* extrinsic parameters** (R = rotation matrix C = camera center)

Task:

Determine a *unique* point ***X*** (=[***a;b;c***;1]) s.t. ***PX*** = ***x***



Observe:
This task is **UNDERDETERMINED**!
Why?

Example for *P,K,R,C* from Hartley / Zissermann Chap. 6, p163

```
% K :== intrinsic camera params
% R :== rotation: world -> cam
% C :== camera center in world
% P :== camera model
% M = K*R
% P = [M –M*C], is a 3x4 matrix
format longG; P = [
 353.553 339.645 277.744 -1449460
 -103.528 23.3212 459.607 -632525
 0.707107 -0.353553 0.612372 -918.559
];
C = [1000;2000;1500]
M = [
 353.581904 339.673062 277.72616
 -103.525936 23.320992 459.607424
 0.70711 -0.35355 0.61237
]
```

```
K = [
 468.2 91.2 300
 0 427.2 200
 0 0 1
];
R = [
 0.4138 0.90915 0.04708
 -0.57338 0.22011 0.78917
 0.70711 -0.35355 0.61237
];
% Sanity check:get back C from P
X = det(P(:,[2,3,4]));
Y = -det(P(:,[1,3,4]));
Z = det(P(:,[1,2,4]));
T = -det(P(:,[1,2,3]));
X/T
         -1000.00073078916
Y/T
         -2000.00195199755
Z/T
         -1500.00028314237
```

Question after precision

We will get ***P*** from ***K*, *R*, *C***

all via a process called

CAMERA CALIBRATION

How does the imprecision of the measured camera parameters influence the pixel coordinates on the image plane {green}?

A: Strategy

1. Start in world frame {W} and map to the camera frame {C}

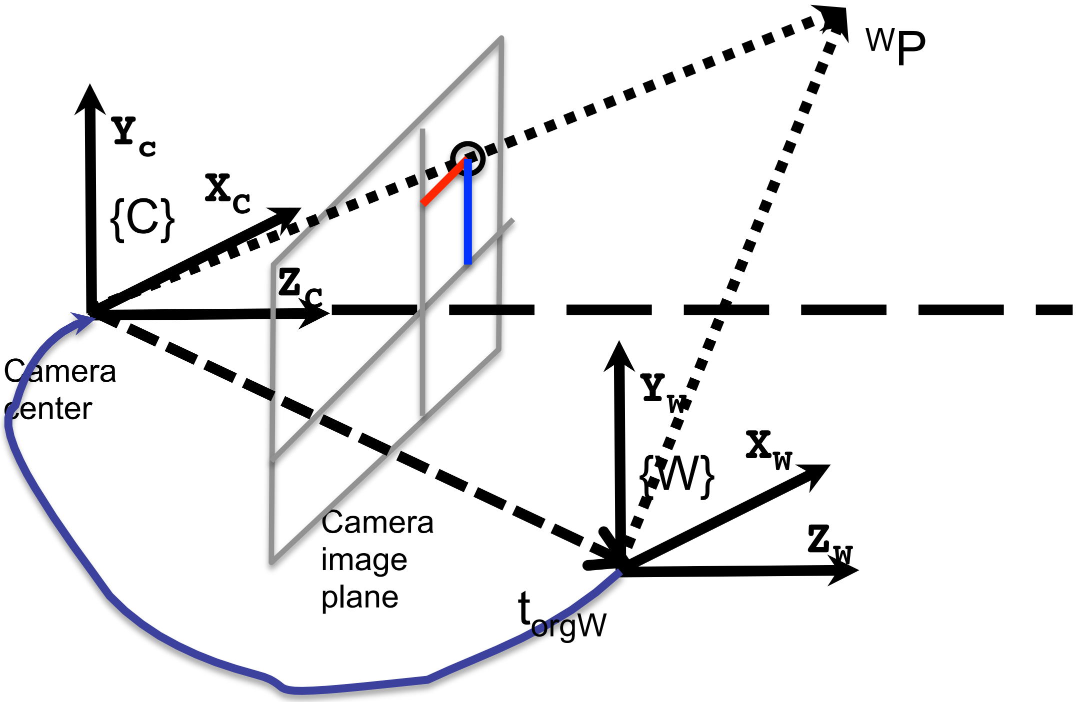
   {W} -> {C}

2. Project to image plane (still using camera frame coordinates)

   {C} -> {grey}

3. map to image (pixel) coordinates

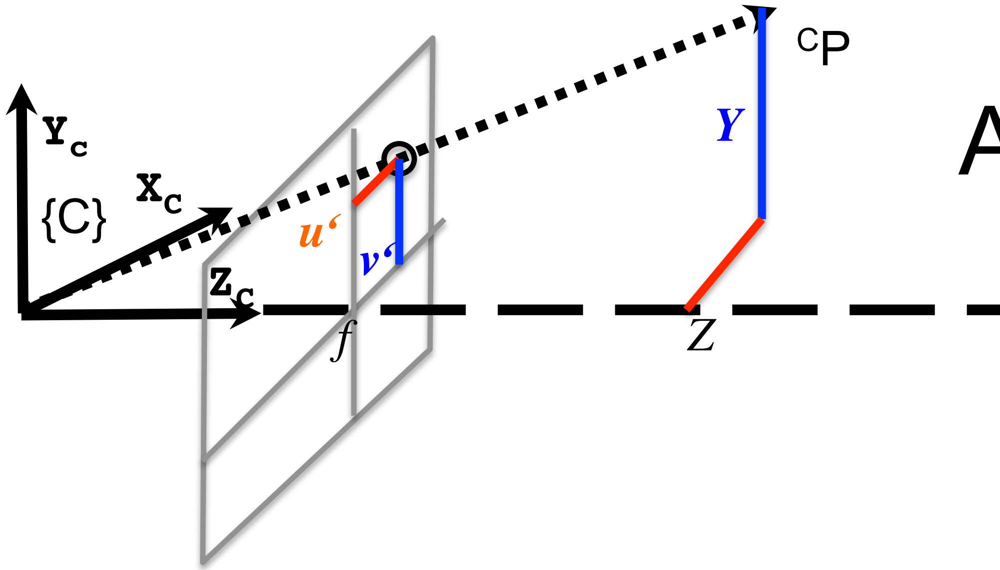
   {grey} -> {green}



A1: {W} -> {C}

$$r11 = c\beta\, c\gamma$$
$$r12 = s\alpha\, s\beta\, c\gamma - c\alpha\, s\gamma$$
$$r13 = c\alpha\, s\beta\, c\gamma + s\alpha\, s\gamma$$
$$r21 = c\beta\, s\gamma$$
$$r22 = s\alpha\, s\beta\, s\gamma + c\alpha\, s\gamma$$
$$r23 = c\alpha\, s\beta\, s\gamma - s\alpha\, c\gamma$$
$$r31 = -s\beta$$
$$r32 = s\alpha\, c\beta$$
$$r33 = c\alpha\, c\beta$$
$$\alpha, \beta, \gamma \quad Euler\ angles$$

$$^{C}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = {}^{C}_{W}T_{\text{hom}} \; ^{W}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{C}_{W}\begin{bmatrix} r11 & r12 & r13 \\ r21 & r22 & r23 \\ r31 & r32 & r33 \end{bmatrix} & \vec{t}_{orgw} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix} \; ^{W}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$



A2: {C} -> {gray}

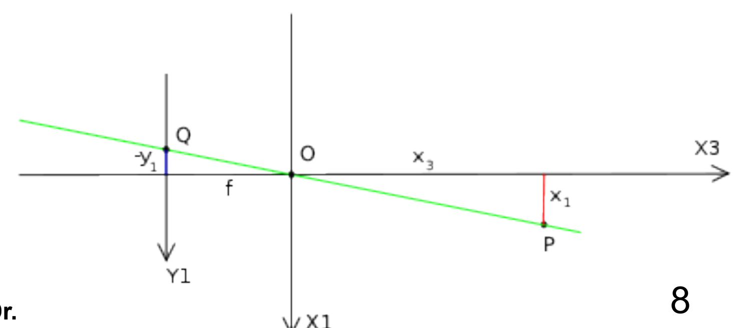
f *focal length*

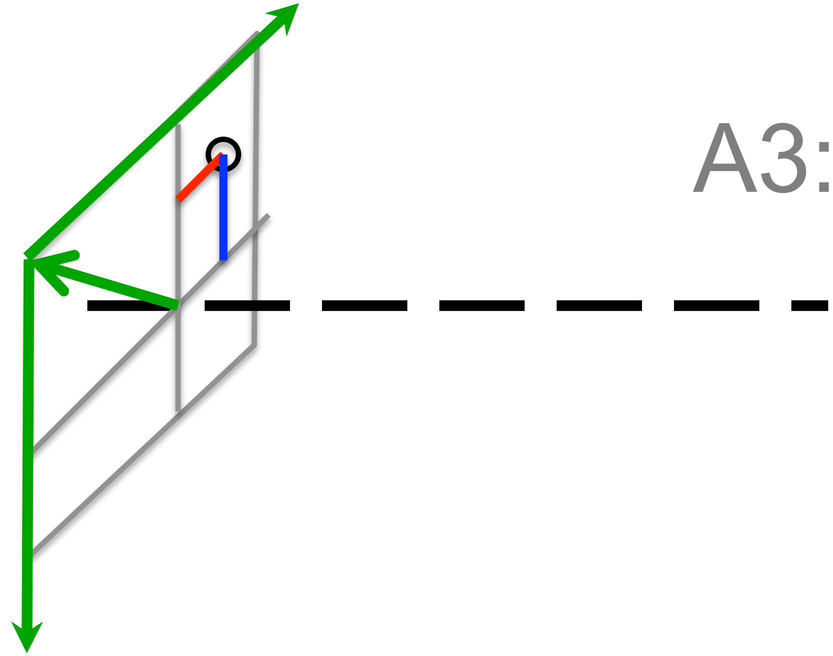
$$\frac{u'}{f} = \frac{X}{Z} \Leftrightarrow u' = \frac{f}{Z} X$$

$$\frac{v'}{f} = \frac{Y}{Z} \Leftrightarrow v' = \frac{f}{Z} Y$$

$$^{grey}\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \frac{f}{Z}\begin{bmatrix} X \\ Y \\ \frac{Z}{f} \end{bmatrix} \sim {}^{C}\begin{bmatrix} X \\ Y \\ \frac{Z}{f} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix} {}^{C}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} := P$$





A3:{grey} -> {green}

$$
{}^{Green}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} := \overset{K}{\begin{bmatrix} fc(1) & \alpha_c * fc(1) & cc(1) \\ 0 & fc(2) & cc(2) \\ 0 & 0 & 1 \end{bmatrix}} \; {}^{gray}\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix}
$$

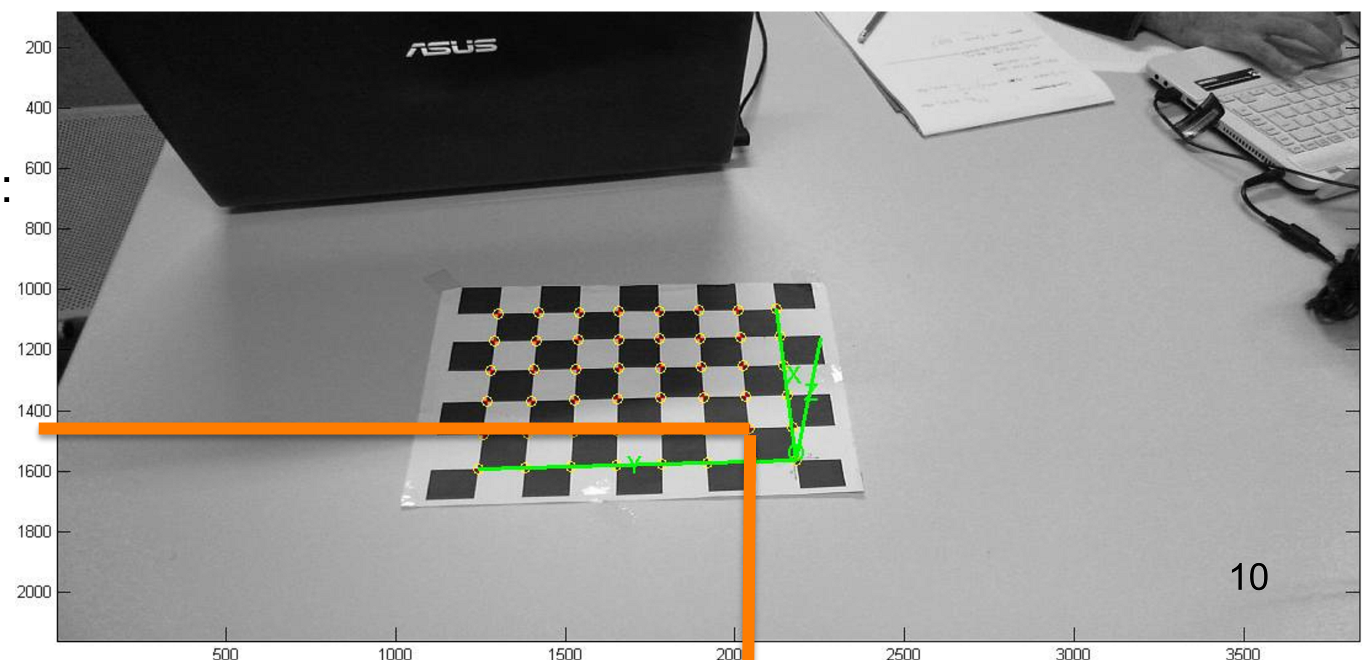
Now Data from Kilian + David, from LEA, mat file

```
C =
 -355.4000
   92.1800
  605.5000

K =
   1.0e+03 *
   3.050393312453318                   0   1.986372170712007
                   0   3.024051420072965   0.99500971740[illegible]72
                   0                   0                   0

R =
  -0.059269656395942  -0.997265685484274   0.044139102690120
  -0.751404608620741   0.015460942212210  -0.659660574393697
   0.657174422793689  -0.072264180764005  -0.750266396824681
```

Must be

1 !!!!!

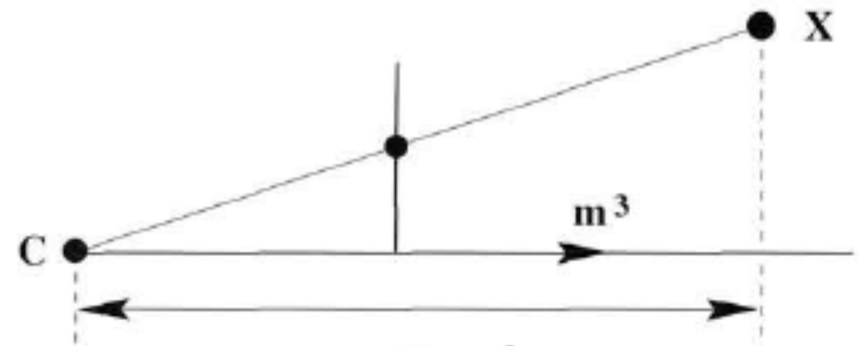
I added a new point:
in WORLD
(30,30,0,1) **[mm]**
in image
(2039,1459,1)



Hochschule
Bonn-Rhein-Sieg

Solution idea

Use:



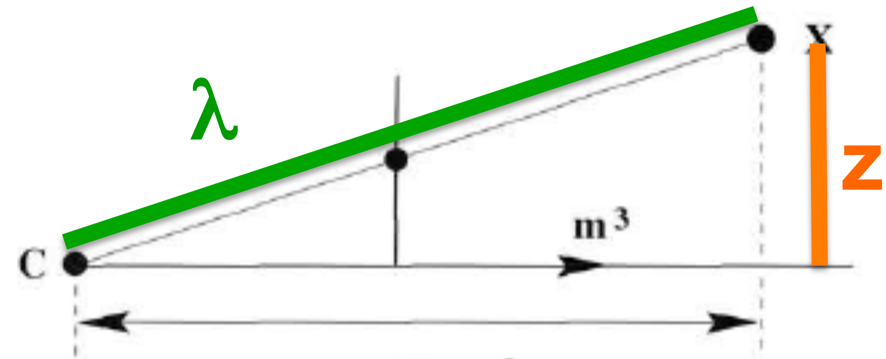
$$X(\lambda) = P^{+}x + \lambda C$$

$i.e.\ X(\lambda)$ *is any point on the ray which connects the image point* x *and the camera center* C.

Problem : what is λ?

Solution idea

Use:



$$(1) \quad X(\lambda) = P^{+}x + \lambda C$$

i.e. $X(\lambda)$ *is any point on the ray which connects the image point* x *and the camera center* C.

Problem: what is λ*?*

Answer:
If we would know **ONE compoment of X (in real world)** then eq. (1) can be used to determine **one unique lambda** from this one known point!
Having the λ determine x and y for X vector!

...cont...

```
M = K*R;
P=[M -M*C];
X=det(P(:,2:4))
Y=-det(P(:,[1,3,4]))
Z=det(P(:,[1,2,4]))
T=-det(P(:,[1,2,3]))
X/T
 -355.4
Y/T
 92.18
Z/T
 605.5
C(4)=1;
```

```
Pplus = pinv(P);
Ximage = [2037;1459;1];
myZ = 0.0;%KNOWN! My point is on table
syms lamb;
Xlamb = Pplus*Ximage+lamb*C;
r = solve(Xlamb(3)-myZ*Xlamb(4),lamb);
r = double(r);
Xq = subs(Xlamb(1),lamb,r);
Yq = subs(Xlamb(2),lamb,r);
Zq = subs(Xlamb(3),lamb,r);
Wq = subs(Xlamb(4),lamb,r);
Xq = eval(Xq/Wq)
Yq = eval(Yq/Wq)
Zq = eval(Zq/Wq)
```

Xq =
29.573195399613
Yq =
30.6556816805163
Zq =
6.06473223866988e-16

This is correct up to 0,7 mm !!!

Partial result of output of Caltech calibration toolbox

```
Calibration results after optimization (with uncertainties):

Focal Length:          fc = [ 657.30254   657.74391 ] ± [ 0.28487   0.28937 ]
Principal point:       cc = [ 302.71656   242.33386 ] ± [ 0.59115   0.55710 ]
Skew:             alpha_c = [ 0.00042 ] ± [ 0.00019  ]   => angle of pixel axes
Distortion:            kc = [ -0.25349   0.11868   -0.00028   0.00005   0.00000
Pixel error:          err = [ 0.11743   0.11585 ]

Note: The numerical errors are approximately three times the standard deviation
```

$$T_{\text{hom}} \Rightarrow undistorted$$

$$P \Rightarrow distorted\ only\ by\ variance\ in\ F\ (fc)$$

$$K \Rightarrow distorted\ by\ known\ \sigma_{\alpha}, \sigma_{cc}, \sigma_{fc(1)}, \sigma_{fc(2)}$$

$$G(u,v)\ is\ distorted\ by\ known\ \sigma_{kc}$$

Non linear distortion G(u,v)

Unfortunately u‘,v‘ dont land where expected!!

Lens distortion [u‘‘,v‘‘]= G(u‘,v‘):

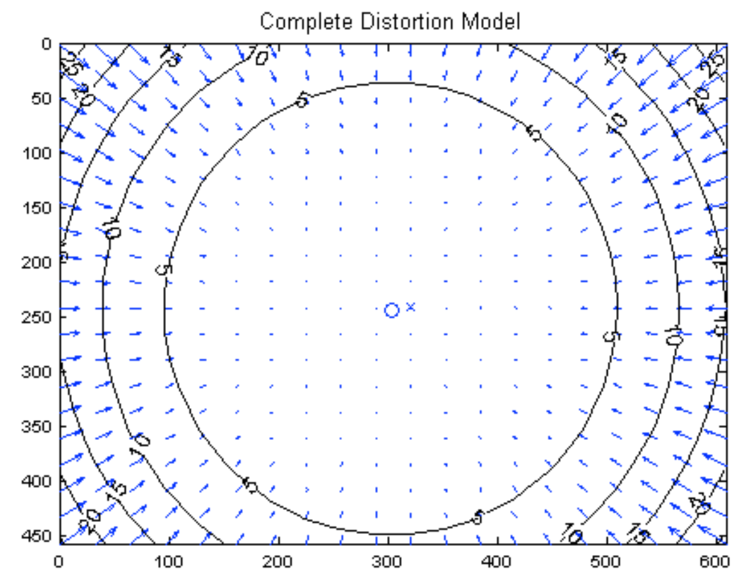
$$
{}^{green}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K * {}^{gray}\begin{bmatrix} u'' \\ v'' \\ 1 \end{bmatrix} = K * G(P * T_{\text{hom}} \, {}^{W}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix})
$$

$$
\begin{bmatrix} u'' \\ v'' \end{bmatrix} = G(\begin{bmatrix} u' \\ v' \end{bmatrix}) = \begin{bmatrix} G_1(u',v') \\ G_2(u',v') \end{bmatrix} =
$$

$$
\begin{bmatrix} u'(1 + kc(1)r^2 + kc(2)r^4 + kc(5)r^6) \\ v'(1 + kc(1)r^2 + kc(2)r^4 + kc(5)r^6) \end{bmatrix} + \begin{bmatrix} du \\ dv \end{bmatrix}
$$

$$
\begin{bmatrix} du \\ dv \end{bmatrix} = \begin{bmatrix} 2kc(3)u'v' + kc(4)(r^2 + 2u'^2) \\ kc(3)(r^2 + 2v'^2) + 2kc(4)u'v' \end{bmatrix}
$$

$$
r = \sqrt{u'^2 + v'^2}
$$



Pixel error	= [0.1174, 0.1159]	
Focal Length	= (657.303, 657.744)	+/- [0.2849, 0.2894]
Principal Point	= (302.717, 242.334)	+/- [0.5912, 0.5571]
Skew	= 0.0004198	+/- 0.0001905
Radial coefficients	= (-0.2535, 0.1187, 0)	+/- [0.00231, 0.009418, 0]
Tangential coefficients	= (-0.0002789, 5.174e-005)	+/- [0.0001217, 0.0001208]

Total

$$T_{\text{hom}}\ \textit{undistorted}$$

$$P\ \textit{distorted only by variance in}\ F$$

$$G(u,v)\ \textit{is distorted by known}\ \sigma_{kc}$$

$$K\ \textit{is distorted by known}\ \sigma_{\alpha}, \sigma_{cc}, \sigma_{fc(1)}, \sigma_{fc(2)}$$

```
Calibration results after optimization (with uncertainties):

Focal Length:          fc = [ 657.30254   657.74391 ] ± [ 0.28487   0.28937 ]
Principal point:       cc = [ 302.71656   242.33386 ] ± [ 0.59115   0.55710 ]
Skew:             alpha_c = [ 0.00042 ] ± [ 0.00019  ]   => angle of pixel axes = 89.97595 ± 0.01092 degrees
Distortion:            kc = [ -0.25349   0.11868   -0.00028   0.00005   0.00000 ] ± [ 0.00231   0.00942   0.00
Pixel error:          err = [ 0.11743   0.11585 ]

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

B: FINALLY: $S_{G(kc(1...6))}$

If S_{zi} are empirical variances and if z_i's are independent (i.e. $S_{zi\,zj} \approx 0$ for $i \neq j$) then it holds:

$$S_G^2 = \sum_i S_{z_i}^2 \left(\frac{\partial G}{\partial z_i} \right)^2$$

$$^{green}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = KG(PT_{\text{hom}}\;{}^{W}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix})$$

In case that G has imprecise coefficients and imprecise input values, take partial derivative w.r.t. all varying inputs and neglect second order terms.

$$S^2_{u''} = \sum_{i=1}^{6} S^2_{kc(i)} \left(\tfrac{\partial G_1}{\partial kc(i)}\right)^2 =$$

$$S^2_{kc(1)}(r^4u^2) + S^2_{kc(2)}(r^8u^2) + S^2_{kc(3)}(4u^2v^2) +$$

$$S^2_{kc(4)}(r^2 + 2u^2)^2 + S^2_{kc(5)}(r^{12}u^2)$$

$$S^2_{v''} = \sum_{i=1}^{6} S^2_{kc(i)} \left(\tfrac{\partial G_2}{\partial kc(i)}\right)^2 =$$

$$S^2_{kc(1)}(r^4v^2) + S^2_{kc(2)}(r^8v^2) + S^2_{kc(4)}(4u^2v^2) +$$

$$S^2_{kc(3)}(r^2 + 2v^2)^2 + S^2_{kc(5)}(r^{12}v^2)$$

$$|r| \leq \sqrt{\max v^2 + \max u^2}$$

$$|v| \leq \max v$$

$$|u| \leq \max u$$

$S^2_{u''} + S^2_{v''}$

Imprecision in K

$$S_u^2 = S_{fc1}^2 \frac{\partial K_1}{\partial fc1} + S_\alpha^2 \frac{\partial K_1}{\partial \alpha} + S_{cc1}^2 \frac{\partial K_1}{\partial cc1}$$

$$= S_{fc1}^2 (u + \alpha v)^2 + S_\alpha^2 fc1^2 v^2 + S_{cc1}^2$$

$$S_v^2 = S_{fc2}^2 \frac{\partial K_2}{\partial fc2} + S_{cc2}^2 \frac{\partial K_2}{\partial cc2}$$

$$= S_{fc2}^2 v^2 + S_{cc2}^2$$

Total

Combine both

Propagate all effects though

Task : rehease this now for *YOUR* data !!!

Hint: use now the height of the robot‘s LEDs over ground!