On the Normality of the Projection Parameters

A. Furlan, D. Marzorati and D. Sorrenti

Outline

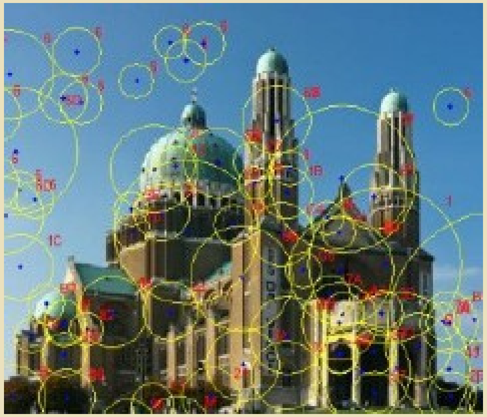
- Introduction
- Filtering in Computer Vision
- Error sources and common assumptions
- The camera model
- Camera calibration and errors: Zhang and Bouguet
- Proving the normality of the calibration parameters
- Using the Unscented Transform for a correct error propagation
- Results & Conclusions

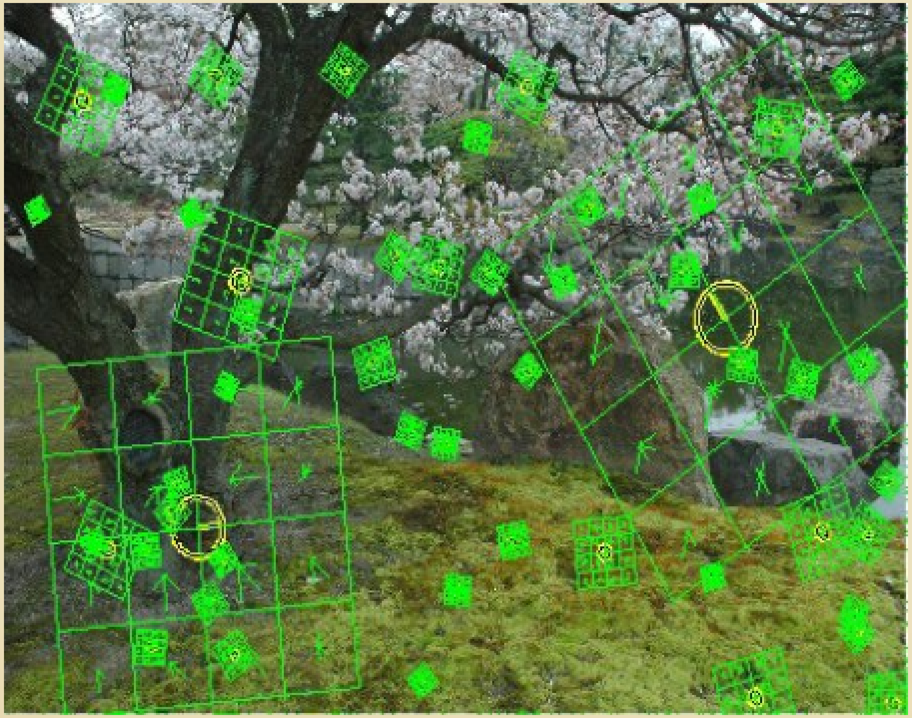
Introduction

- Robotic perception requires computer vision as well as other sensing means.
- The robotic agent can build its own representation of the world.
- Robot vision includes:
  - Object Tracking
  - Visual Simultaneous Localization and Mapping (SLAM)
- Both OT and SLAM use filtering

Image analysis – 1

- Feature extraction





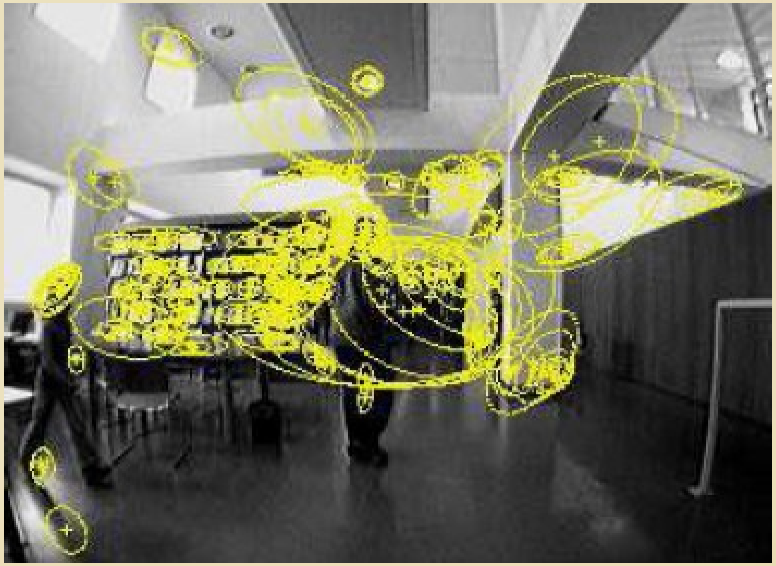
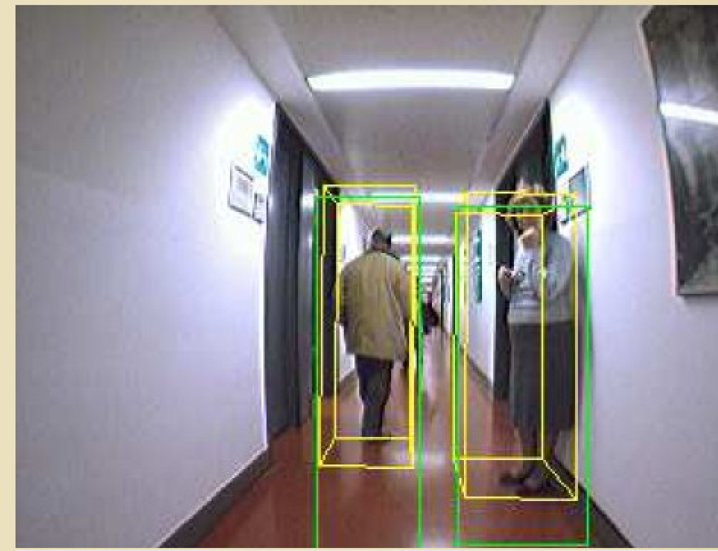
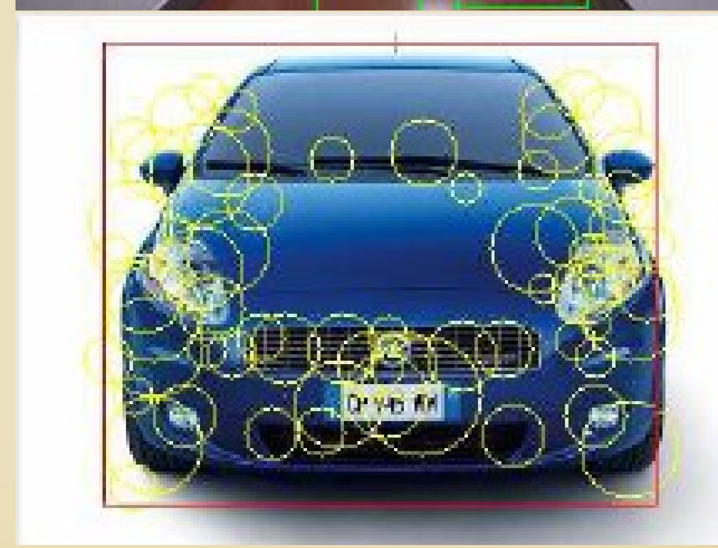


Image analysis – 2

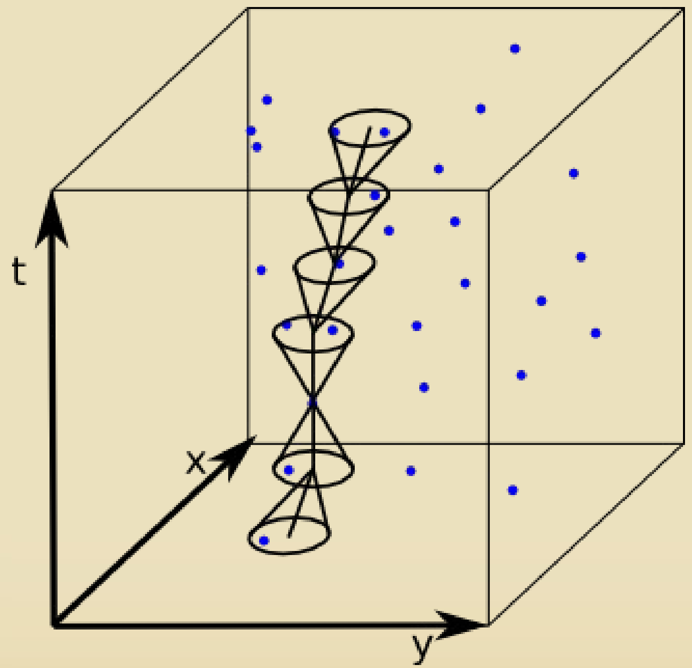
- Object detection

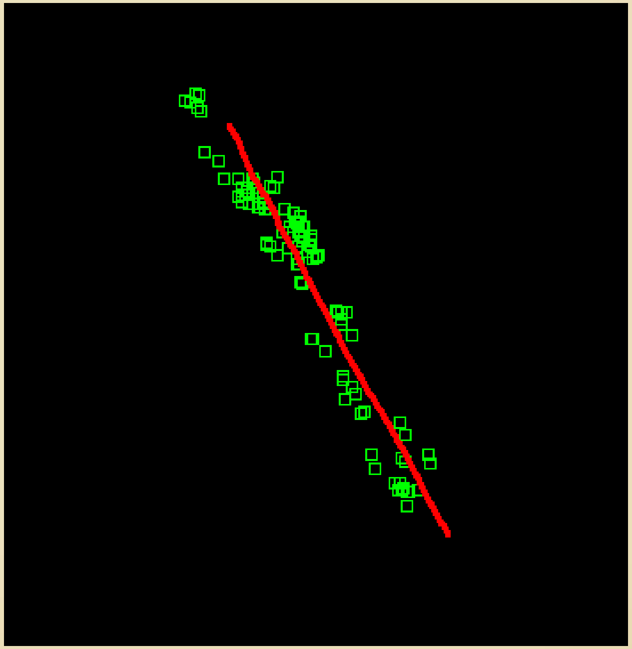




Object Tracking

- Perform space-time filtering
- Associate instances of objects over time





The data association problem

- Problem which arises when tracking multiple objects/features
- Good data association needs **good uncertainty estimation**
- Under-estimation may lead to complete divergence of the filter
- Over-estimation may impose the use of expensive techniques for discriminate against different objects

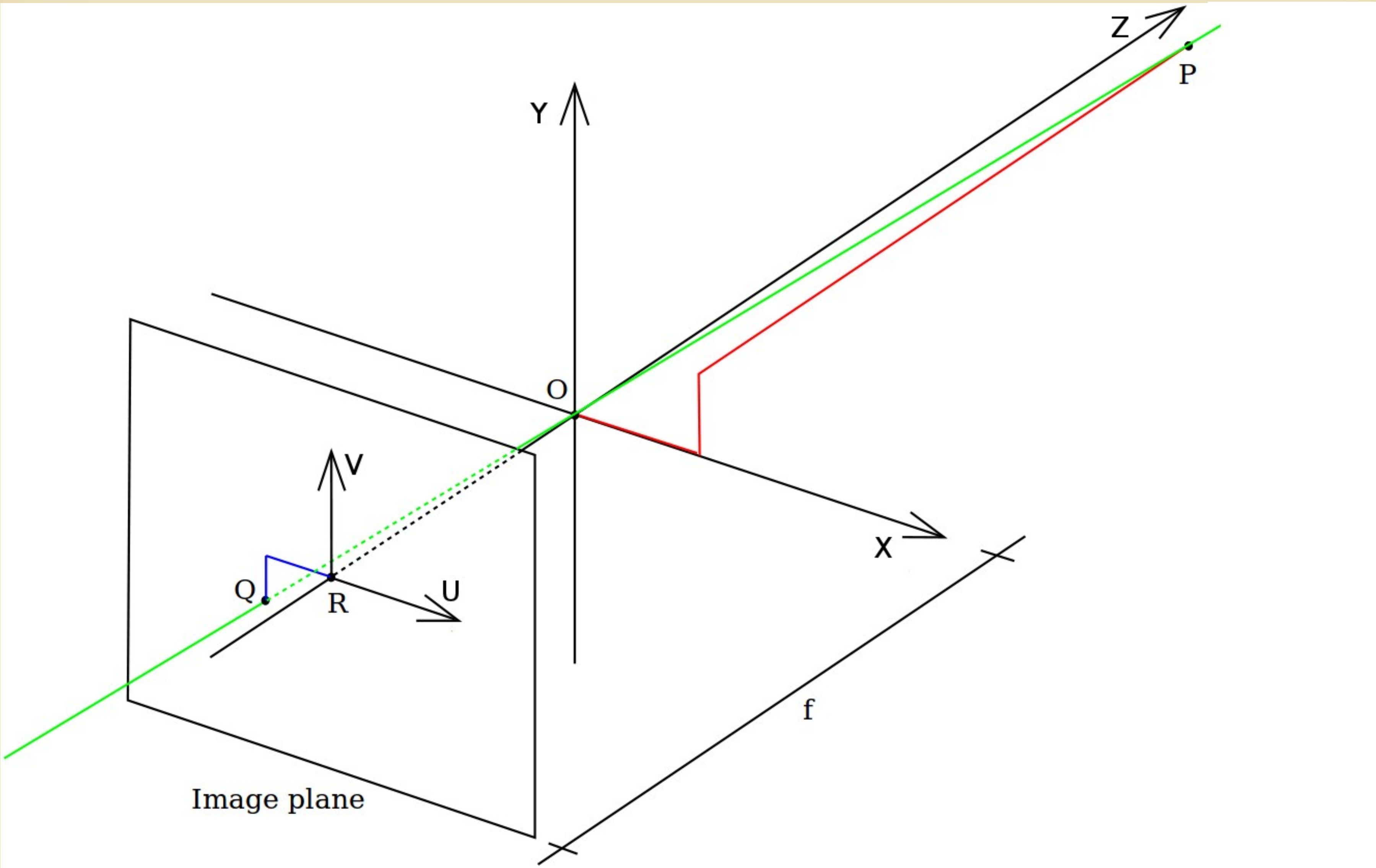
Error sources

- Errors in detection can be principally due to:
  - Detector precision
  - Inaccurate estimate of camera parameters
- At detection time, each detection is affected by an error with zero mean and std. deviation σ_{err}
- Camera calibration is performed only once
- Thus, calibration mistakes 'polarize' the error during detection

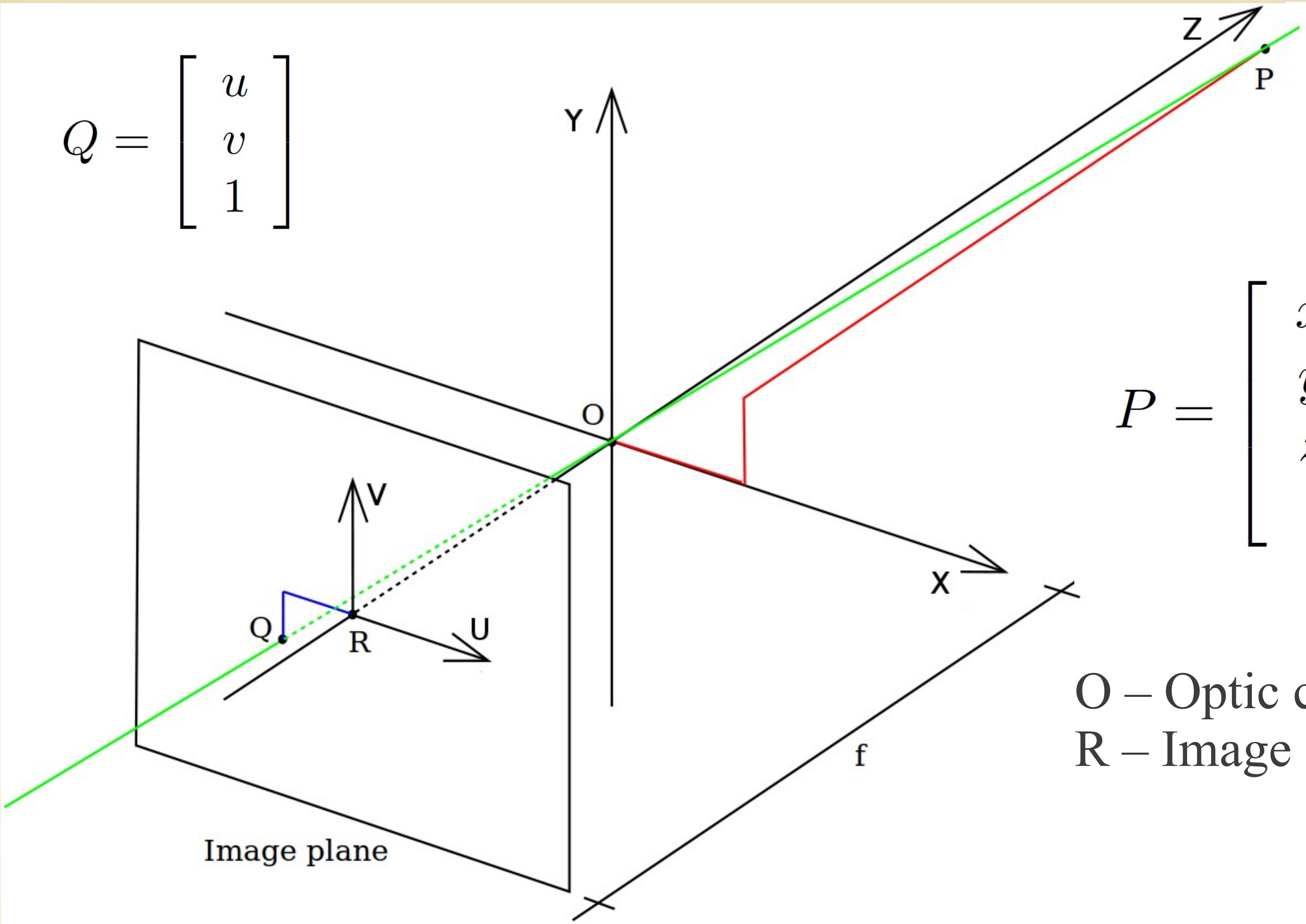
Common assumpions

- It is commonly assumed that:
  - Detector errors (i.e. 2D image points) are normally distributed
  - Calibration errors are normally distributed
- We focus on the demonstration of the second assumption
- It is not straightforward because of the non-linearity of the calibration process

The camera model



The camera model



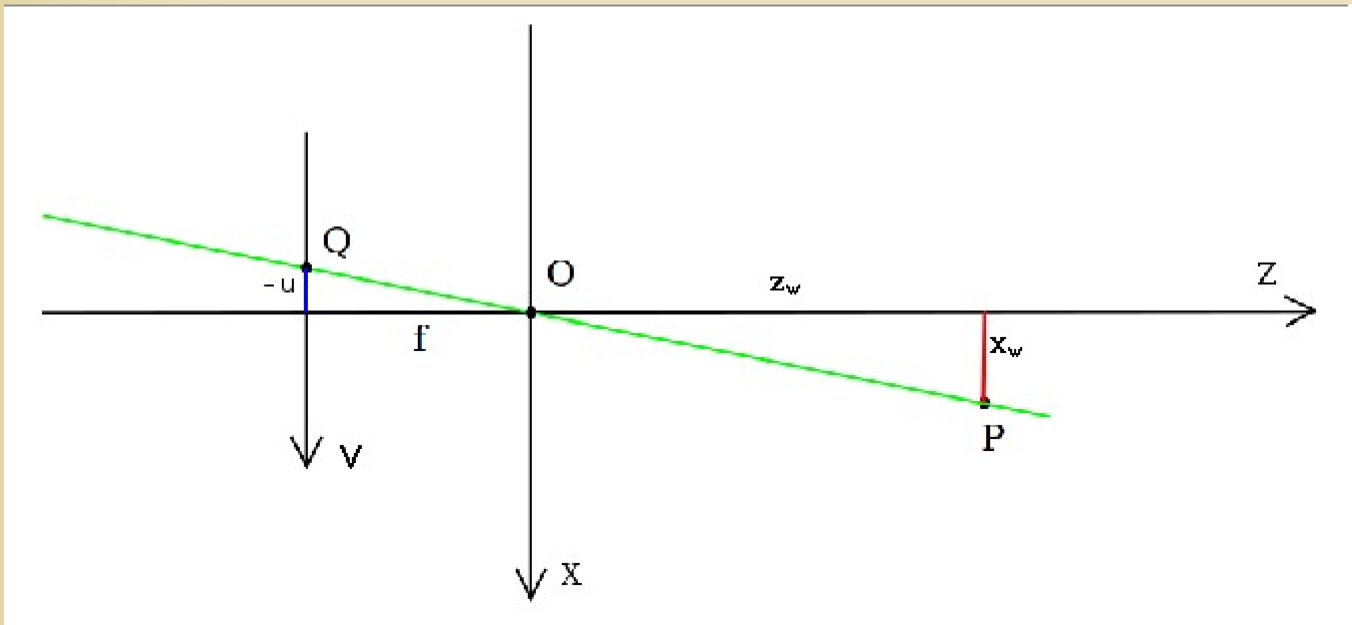
$$Q = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

$$P = \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

O – Optic center
R – Image center

The camera model

- Looking from the Y axis we can write:



$$\frac{-u}{f} = \frac{x_w}{z_w}$$

- Similarly we can write:

$$\frac{-v}{f} = \frac{y_w}{z_w}$$

Camera projection

- The projection is composed by:
  - a **roto-translation** to align the world frame reference with the camera
  - a **projection** performed by the camera matrix

$$z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A \begin{bmatrix} R & T \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

- The parameters of the roto-translation are called the extrinsic parameters
- The parameters of the projection are called intrinsic parameters

The camera matrix

$$A = \begin{bmatrix} \alpha & c & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

- (u_0, v_0) – coordinates of the principal point
- α and β – scale factors in image U and V axes
- c – skewness parameter

Camera calibration

- Procedure which estimates intrinsic and extrinsic camera parameters
- Different approaches in the 90's, e.g. DLT
- In 1999 Zhengyou Zhang proposed a new, very flexible method
- It became a milestone in camera calibration
- Jean-Yves Bouguet refined this method and developed libraries for OpenCV and Matlab

Calibration procedure – Zhang

- Very flexible technique
- Unlike former methods it does not require expensive equipment
- It only requires the camera to observe a planar pattern shown at a few differen orientations
- Two steps:
  - Closed form solution
  - Non-linear refinement (maximum likelihood)

Zhang – Closed-form solution

$$s\tilde{\mathbf{m}} = \mathbf{A}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\tilde{\mathtt{M}}$$

$$H = \lambda A[R\ T]$$

- Estimation of the **homography H** between the model plane and its image

Zhang – Constraints and definitions

$$\begin{bmatrix}\mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3\end{bmatrix} = \lambda \mathbf{A} \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix}$$

$$\mathbf{h}_1^T \mathbf{A}^{-T} \mathbf{A}^{-1} \mathbf{h}_2 = 0$$

$$\mathbf{h}_1^T \mathbf{A}^{-T} \mathbf{A}^{-1} \mathbf{h}_1 = \mathbf{h}_2^T \mathbf{A}^{-T} \mathbf{A}^{-1} \mathbf{h}_2$$

$$\mathbf{B} = \mathbf{A}^{-T} \mathbf{A}^{-1} \equiv \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{\alpha^2} & -\frac{c}{\alpha^2\beta} & \frac{cv_0 - u_0\beta}{\alpha^2\beta} \\ -\frac{c}{\alpha^2\beta} & \frac{c^2}{\alpha^2\beta^2} + \frac{1}{\beta^2} & -\frac{c(cv_0 - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} \\ \frac{cv_0 - u_0\beta}{\alpha^2\beta} & -\frac{c(cv_0 - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} & \frac{(cv_0 - u_0\beta)^2}{\alpha^2\beta^2} + \frac{v_0^2}{\beta^2} + 1 \end{bmatrix}$$

Zhang – Parameters extraction

$$\mathbf{h}_i^T \mathbf{B} \mathbf{h}_j = \mathbf{v}_{ij}^T \mathbf{b}$$

$$\mathbf{b} = [B_{11}, B_{12}, B_{22}, B_{13}, B_{23}, B_{33}]^T$$

$$\mathbf{v}_{ij} = [h_{i1}h_{j1}, h_{i1}h_{j2} + h_{i2}h_{j1}, h_{i2}h_{j2}, \\ h_{i3}h_{j1} + h_{i1}h_{j3}, h_{i3}h_{j2} + h_{i2}h_{j3}, h_{i3}h_{j3}]^T$$

$$\begin{bmatrix} \mathbf{v}_{12}^T \\ (\mathbf{v}_{11} - \mathbf{v}_{22})^T \end{bmatrix} \mathbf{b} = \mathbf{0} \, .$$

Zhang – Solving the system

$$\mathbf{V}\mathbf{b} = \mathbf{0}$$

- This system can be solved with SVD
- The solution is the right singular vector of V associated with the smallest singular value
- Once **b** is estimated, intrinsic and extrinsic parameters can be readily extracted

Zhang – Refinement

- Due to noise and imperfect modelization, the closed form solution is very rough
- It can be refined through maximum likelihood inference

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \|\mathbf{m}_{ij} - \hat{\mathbf{m}}(\mathbf{A}, \mathbf{R}_i, \mathbf{t}_i, \mathtt{M}_j)\|^2$$

- Non-linear minimization problem → Levenberg-Marquardt algorithm
- Use of the closed-form solution as an initial guess

Calibration procedure – Bouguet

- In Zhang's method there is no information about the error on the estimated parameters
- In the OpenCV library there is no information about the error on the estimated parameters
- In Matlab, Bouguet added to the Toolbox Calib an analysis of the 'residual' error
- It is calculated as the error between the measured image points and the re-projected ones

Calibration procedure – Bouguet

- Assuming that this error is normally distributed, it is described with $N(0,\sigma_{err})$
- The errors on the calibration parameters are also assumed to be normally distributed
- Thus they are computed by propagating σ_{err}
- Propagation is done by linearizing the projection function with a first order expansion
- This choice may generate:
    - Under-estimation due to linearization
    - Under-estimation due to 'unfortunate' calibration set

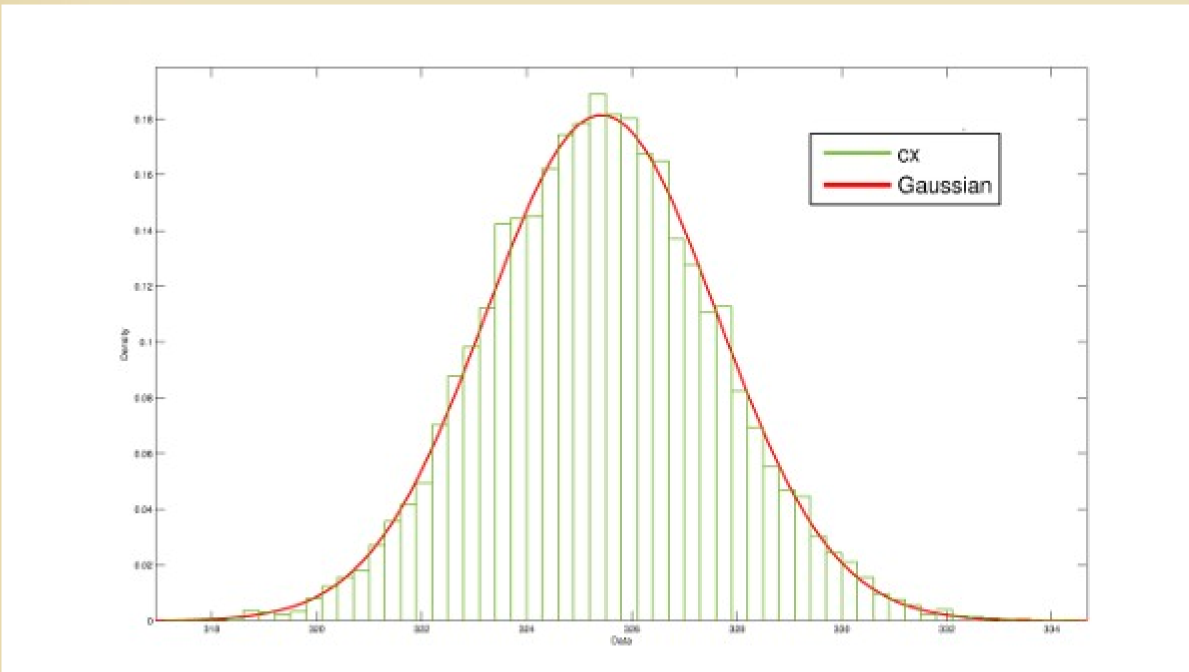
Our work

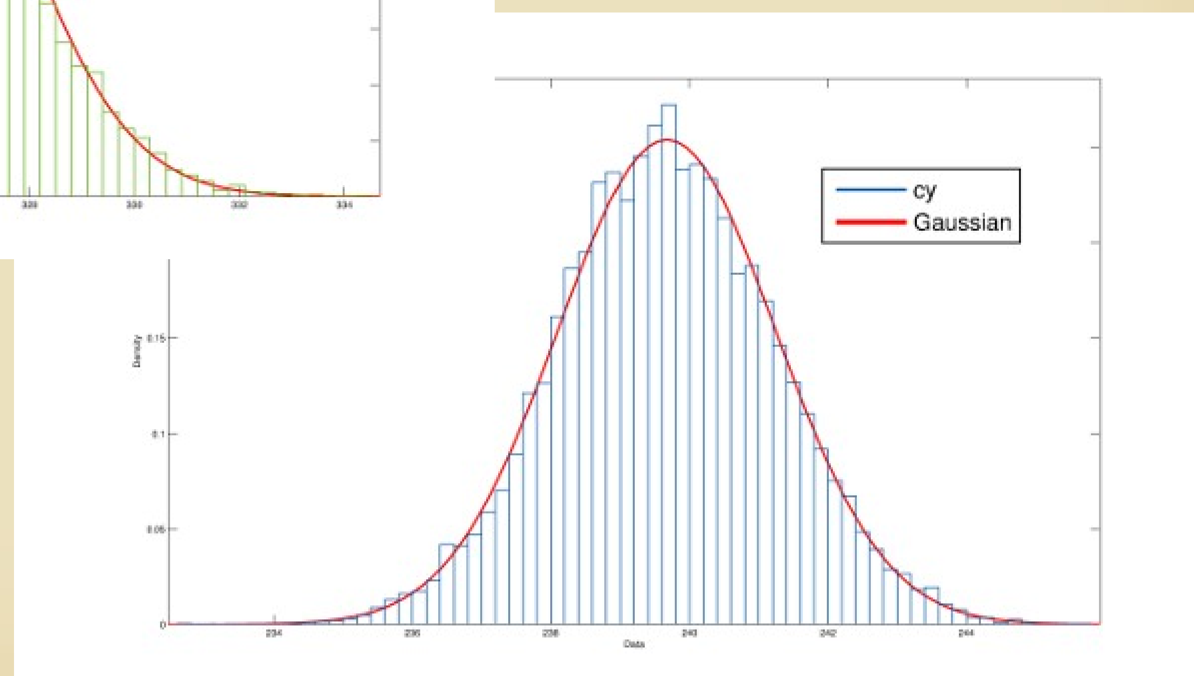
- We first prove that the error on the projection parameters is actually normally distributed
- Secondly we propose a method for calculating the true error distribution
- Our method enables the calibration procedure to take into account an a-priori information about the measure error

Proving the normality

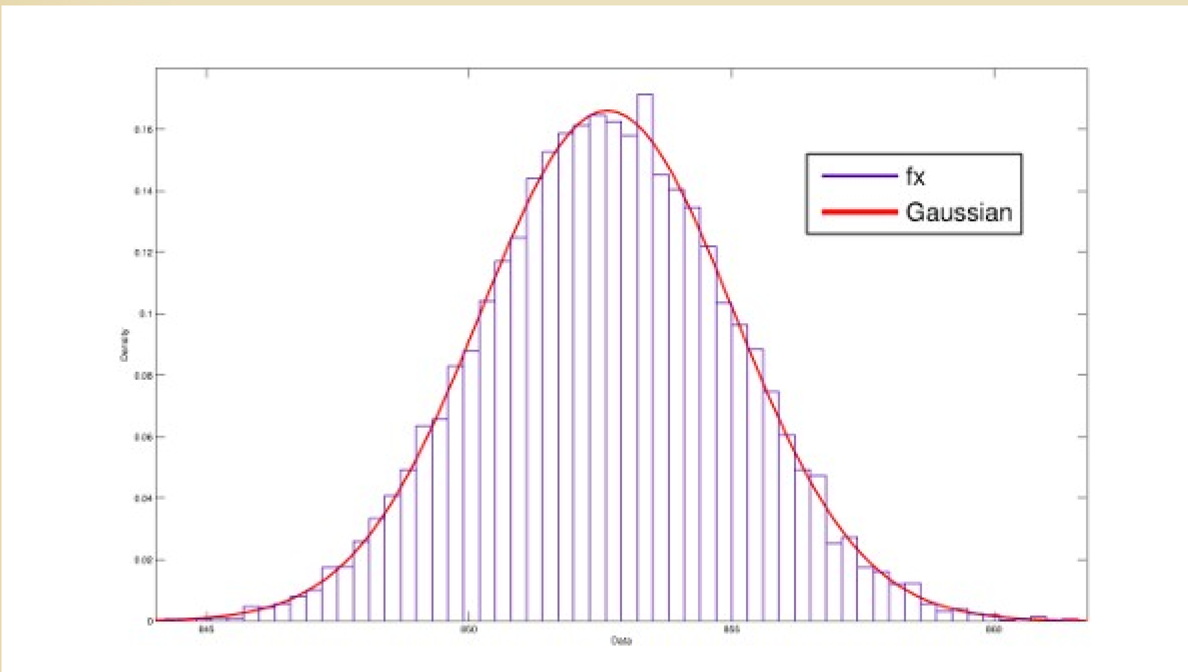
- We use a Particle Transform to prove that the error is normally distributed
- In particular, given an ideal calibration set:
  - We generate 2,000 particles (noisy calibration sets) by adding gaussian noise to the 2D points
  - Each particle is then used to perform a calibration with the OpenCV library
  - The resulting 2,000 calibration results show that the parameters are affected by an error which is normally distributed

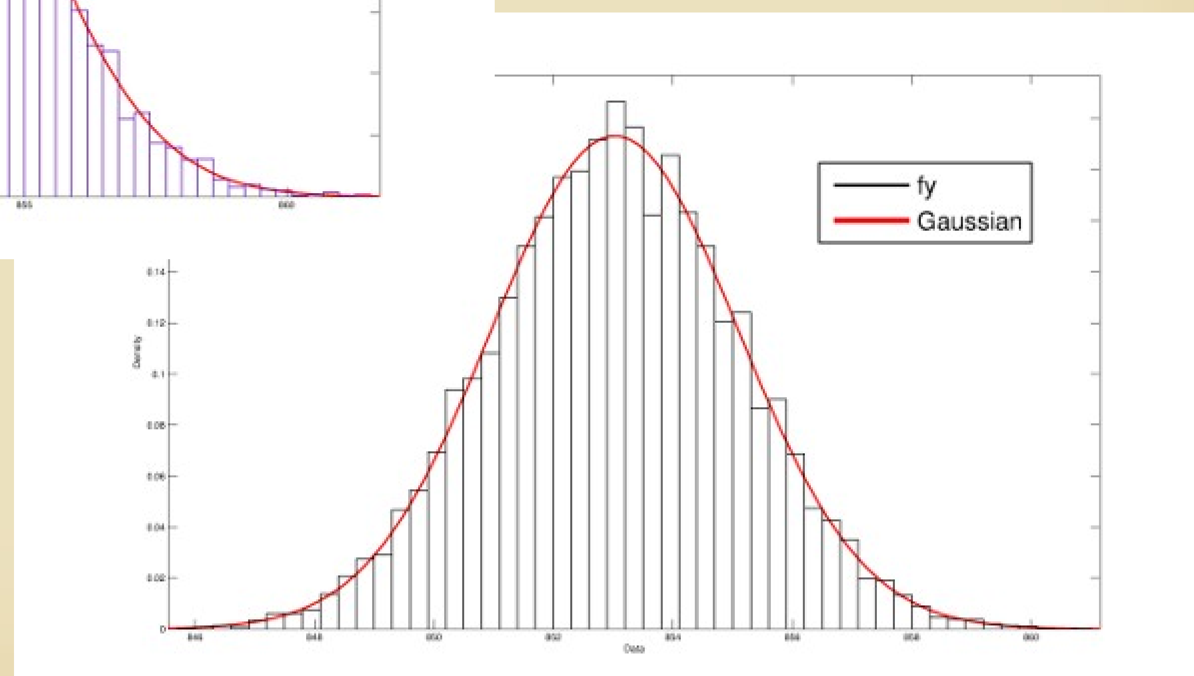
Proving the normality





Proving the normality





True error distribution

- Once we proved the normality of the projection parameters, we do not need the (computationally expensive) Particle Transform any more
- We can now use a transform which works well with normal distributions
- We propose the use of the Unscented Transform
- It can be used to propagate the a-priori knowledge through the calibration procedure

Using the Unscented Transform

- Advantages:
  - We do not need to generate tousands of particles and to make tousands of calibrations
  - We apply the calibration procedure only 2N times on a lightly modified calibration set
  - Instead of using the 2D points observed by the detector, we use them $\pm \sigma_{detector}$
  - The 2N calibration results give us the correct description of the error on the projection parameters

Pros and Cons

- Pros:
  - We can take into account the uncertainty of the detector
  - Particular cases (e.g. high lens distortion) will be well handled (radial distortion model uses 6th order polinomials → bad linearization)
  - Little computational overhead
- Cons:
  - For small $\sigma_{detector}$ the results may not be significantly different from the original method

Results

- Matlab Toolbox Calib procedure

```
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 ]
```

```
Pixel error:          err = [ 0.11743   0.11585 ]
```

Results

- Our calibration procedure ($\sigma_{detector} = 0.25$)

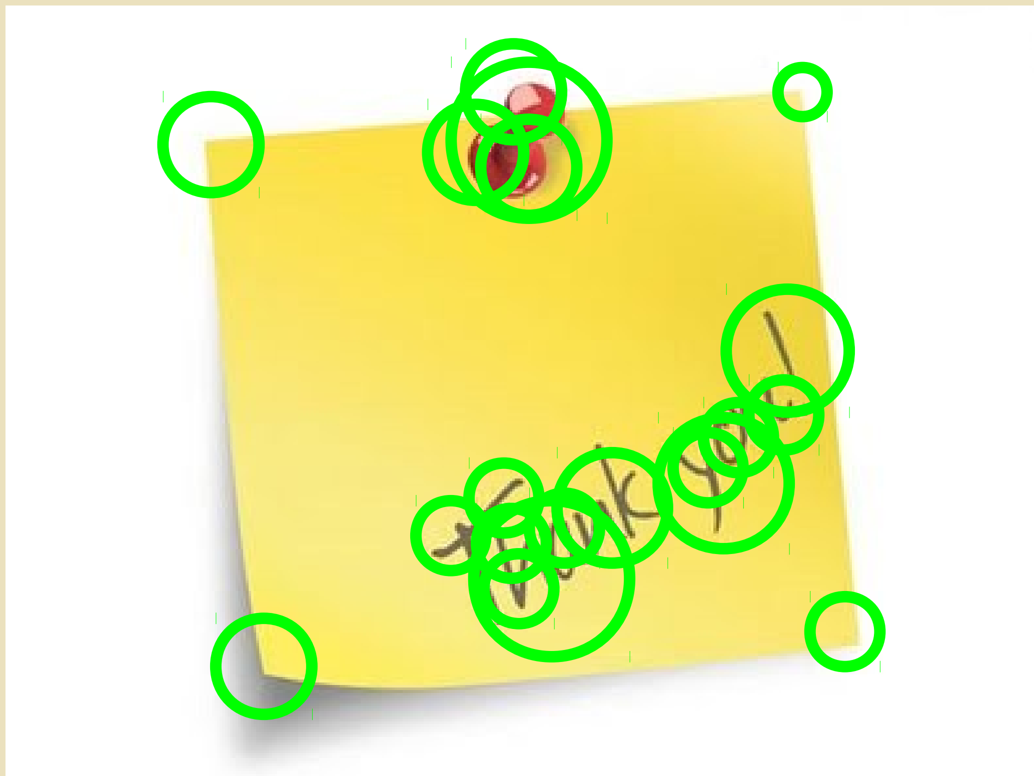
$$A = \begin{pmatrix} 849.884 & 0. & 320.083 \\ 0. & 849.856 & 240.046 \\ 0. & 0. & 1. \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} 2.62697 & -0.134707 & 2.616 & 0.315995 \\ -0.134707 & 3.73523 & -0.129208 & -0.111284 \\ 2.616 & -0.129208 & 2.65658 & 0.26562 \\ 0.315995 & -0.111284 & 0.26562 & 1.79191 \end{pmatrix}$$

Conclusions

- We numerically proved that the error on the projection parameters is normally distributed
- We proposed the use of the Unscented Transform instead of the Particle Transform for error propagation
- The proposed method is only a little slower than the original (highly parallelizable)
- The proposed method can take into account the a-priori uncertainty of the detector
- The proposed method gives the true error distribution, avoiding the linearization of the projection function


Thank you!


Thank you!