Robotic 3D Vision
Prof. Dr. Jörg Stückler, Rui Wang
Winter Semester 2017/2018

Computer Vision Group
Department of Informatics
Technical University of Munich

Exercise Sheet 1

Topic: Image Formation, Extended Kalman Filter

Submission deadline: Monday, 15.11.2017, 23:59
Hand-in via email to rob3dvis-ws17@vision.in.tum.de

General Notice

All exercises can be done in teams of up to three students. Please hand-in your solution before the submission deadline, indicating names and matriculation numbers of your team members. Teams are encouraged to present their submitted solution during the exercise sessions.

Exercise 1.1: Image Formation

In this exercise, you will implement basic image processing, projection using the pin-hole camera model and image undistortion using Matlab.

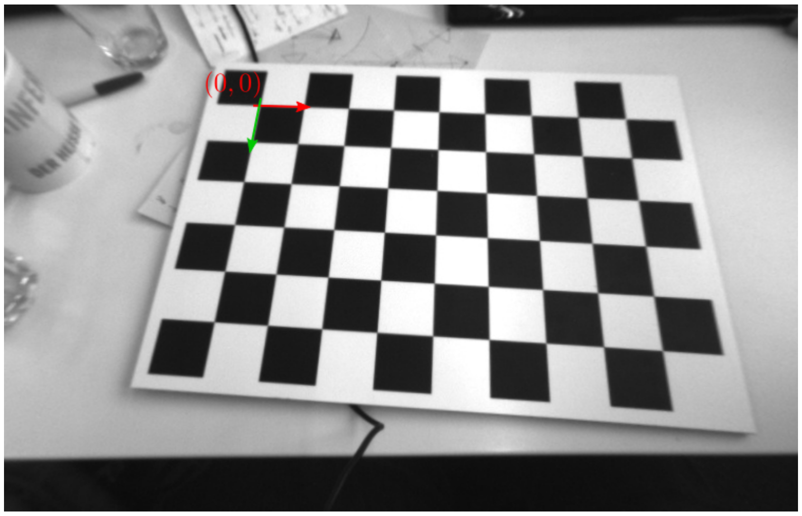

Figure 1: Checkerboard coordinate frame.

(a) Obtain the image data for this part of the exercise from the course webpage. The archive contains two images `sample_undistorted.png` and `sample_distorted.png`. Read the sample images (`imread`) and convert them to grayscale using `rgb2gray`.

(b) Start with the undistorted sample image and draw all the corners in the checkerboard pattern by projecting their 3D coordinates into the image. For this, we define a coordinate frame on the checkerboard whose origin is in the upper left inner corner of the pattern (see Fig. 1). The x-axis is parallel to the longer side of the pattern with increasing values towards the right in the sample image. The y-axis points downwards along the shorter side of the pattern in the image. The length between the corners on the checkerboard is 0.04 m.

The transformation from camera frame to checkerboard frame is given by $\omega_1 = -0.372483192214, \omega_2 = 0.0397022486165, \omega_3 = 0.0650393402332, t_1 = -0.107035863625, t_2 = -0.147065242923, t_3 = 0.398512498053$ in axis-angle representation $(\omega_1, \omega_2, \omega_3)$ for the rotational part and in meters (t_1, t_2, t_3) for the translational part. The camera intrinsics for the undistorted image are specified by focal lengths $f_x = 420.506712, f_y = 420.610940$ and camera center $c_x = 355.2082980, c_y = 250.3367870$.

For projecting the points on the checkerboard, first create the set of 3D points in the checkerboard frame. You can use the function `meshgrid`. Transform the points into the camera frame and project them into the image plane according to the pinhole camera model into pixel coordinates using the given camera intrinsics. Overlay the projected points with the image.

(c) Now repeat the projection and visualization of the checkerboard corners from the previous step in the distorted image. To this end, you need to apply the following distortion model to the normalized image coordinates $\overline{\mathbf{y}} = \pi(\mathbf{T}\overline{\mathbf{x}})$ before mapping them to pixel coordinates using the camera matrix $\overline{\mathbf{y}}_p = \mathbf{C}\overline{\mathbf{y}}_d$:

$$\mathbf{y}_d = (1 + k_1 r^2 + k_2 r^4)\mathbf{y}, r := \|\mathbf{y}\|_2 \,. \qquad (1)$$

The distortion parameters are $k_1 = -0.296609$ and $k_2 = 0.080818$.

(d) Determine the horizontal and vertical field of view of the camera, measured horizontally and vertically across the image center in the distorted and the undistorted case. Hint: Use the following iterative approach to determine undistorted from distorted image coordinates:

```
function UNDISTORT($\mathbf{y}_d$)
    $t \leftarrow 0$
    $\mathbf{y}_t \leftarrow \mathbf{y}_d$
    repeat
        $r = \|\mathbf{y}_t\|_2$
        $r_d = (1 + k_1 r^2 + k_2 r^4)$
        $\mathbf{y}_{t+1} = \frac{1}{r_d}\mathbf{y}_d$
        $t \leftarrow t + 1$
    until $\|\mathbf{y}_t - \mathbf{y}_{t-1}\|_2 <$ threshold
    return $\mathbf{y}_t$
end function
```

Exercise 1.2: Extended Kalman Filter

In this exercise, you will implement a robot localization algorithm based on the Extended Kalman Filter. We assume the robot moves in the 2D plane, for example, a wheeled robot with differential drive that moves on the floor inside a building. This means the robot state $\mathbf{x}_t = (x_t, y_t, \theta_t)^\top$ is 3-dimensional and composed of the 2-dimensional position x_t, y_t in the plane and the robot heading θ_t. We model the robot motion with an odometry-based motion model in this exercise, i.e. the state-transition model is

$$\mathbf{x}_t = g(\mathbf{x}_{t-1}, \mathbf{u}_t) + \boldsymbol{\epsilon}_t := \mathbf{x}_{t-1} + \begin{pmatrix} u_{tr}\cos(\theta_{t-1} + u_{r_1}) \\ u_{tr}\sin(\theta_{t-1} + u_{r_1}) \\ u_{r_1} + u_{r_2} \end{pmatrix} + \boldsymbol{\epsilon}_t, \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{d_t}). \quad (2)$$

The action $\mathbf{u}_t = (u_{tr}, u_{r_1}, u_{r_2})^\top$ is given by translational (u_{tr}) and rotational (u_{r_1}, u_{r_2}) motion measurements obtained from wheel odometry. For the noise covariance of the state-transitions, we assume

$$\boldsymbol{\Sigma}_{d_t} = \begin{pmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.01 \end{pmatrix} \quad (3)$$

The robot measures the range r and bearing ϕ to 2D landmark points $l_j = (l_{j,x}, l_{j,y})$ in the environment in the horizontal plane. It measures multiple landmarks in a time step for which we assume the association $c_{t,i} = j$ of measurements $\mathbf{z}_{t,i} = (r_{t,i}, \phi_{t,i})^\top$ to landmarks j known. The observation model is

$$\mathbf{z}_{t,i} = h(\mathbf{x}_t, c_{t,i}) + \boldsymbol{\delta}_{t,i} := \begin{pmatrix} \left\|(x_t, y_t)^\top - (l_{j,x}, l_{j,y})^\top\right\|_2 \\ \operatorname{atan2}(l_{j,y} - y_t, l_{j,x} - x_t) - \theta_t \end{pmatrix} + \boldsymbol{\delta}_{t,i}, \boldsymbol{\delta}_{t,i} = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{m_{t,i}}). \quad (4)$$

For the observation noise of an individual landmark measurement, we assume

$$\boldsymbol{\Sigma}_{m_{t,i}} = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix}. \quad (5)$$

The complete observation model in each time step, $\mathbf{z}_t = h(\mathbf{x_t}) + \boldsymbol{\delta}_t$ with $\boldsymbol{\delta}_t = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{m_t})$ stacks the M measurements in a single vector $z_t = (z_{t,0}^\top, \ldots, z_{t,M-1}^\top)^\top \in \mathbb{R}^M$. Analogously, we write $h(\mathbf{x}_t) := (h(\mathbf{x}_t, c_{t,0})^\top, \ldots, h(\mathbf{x}_t, c_{t,M-1})^\top)^\top$. The covariance $\boldsymbol{\Sigma}_{m_t}$ is formed from the individual measurement covariances,

$$\boldsymbol{\Sigma}_{m_t} = \begin{pmatrix} \boldsymbol{\Sigma}_{m_t,0} & 0 & \cdots & 0 \\ 0 & \boldsymbol{\Sigma}_{m_t,1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \boldsymbol{\Sigma}_{m_t,M-1} \end{pmatrix}. \quad (6)$$

(a) Determine the analytic Jacobians of the state-transition function $g(\mathbf{x}, \mathbf{u}_t)$ and the observation function $h(\mathbf{x})$ for the robot pose $\mathbf{x}$.

(b) Obtain the code sample and data for this part of the exercise from the course webpage. The archive contains three folders: data, matlab, plots. Implement EKF prediction and correction to localize the robot by finalizing the code in files `prediction_step.m` and `correction_step.m`.

(c) Run your code and report the final robot pose estimate after processing the whole dataset. Visualize your final result using the provided plotting functions. Plot the current state estimate in each time step into a png image and generate a video from the result sequence using avconv.

Submission instructions

A complete submission consists both of a PDF file with the solutions/answers to the questions on the exercise sheet and a ZIP file containing the source code that you used to solve the given problems. Note all names and matriculation numbers of your team members in the PDF file. Make sure that your ZIP file contains all files necessary to compile and run your code, but it should not contain any build files or binaries. Please submit your solution via email to `rob3dvis-ws17@vision.in.tum.de`.