Computer Vision II

Heiko Neumann, Christian Jarvers
Institute of Neural Information Processing
Ulm University

Assignment 5: Submission date (Moodle): February 2, 2023, 12:00.

This homework comprises two tasks. You only need to solve one of the two to pass the homework. Choose whichever one you prefer.

1 Tracking with Lucas-Kanade (10 points)

A key step in appearance-based tracking is to predict how the object or image patch we are tracking will look after a specific motion. This can be achieved by **warping** the image patch, i.e., projecting pixel values to new locations using a set of parameters $\mathbf{p}$ to define the transformation.

The Lucas-Kanade tracking algorithm introduced in the lecture (*chapter VI slide 15*) uses iterative gradient descent to find a parameter vector $\mathbf{p}$ that warps one image into another (i.e., that minimizes the error between the warped and target image). These parameters describe the motion that a tracked object or image patch made between two images. Implement this algorithm and use it to warp the image `box-00.png` into the image `box-01.png`.

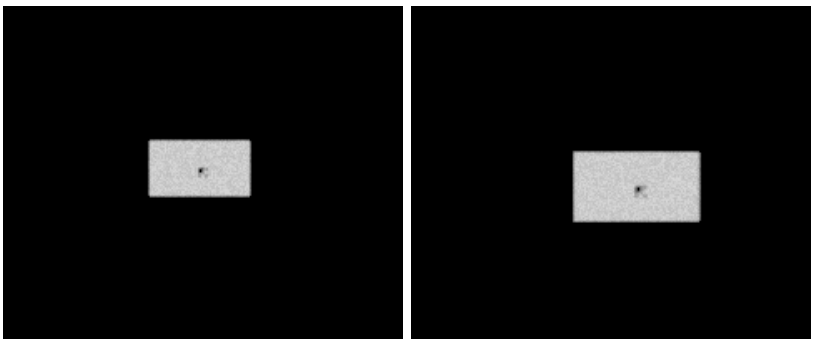

Figure 1: Example images `box-00.png` and `box-01.png`.

1. **Warping**: Implement the function `Iw = imwarp(img, p)`, which takes an image `img` and a parameter vector $\mathbf{p} = [p_1, p_2, p_3, p_4]$ as inputs and transforms the image content according to the warping function w, resulting in $I_w(x) = I(w(\mathbf{x}; \mathbf{p}))$:

$$w(\mathbf{x};\mathbf{p}) = \begin{pmatrix} p_1 & 0 & p_3 \\ 0 & p_2 & p_4 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{1.1}$$

2. **Interpolation**: Since the projected new locations may not match the pixel grid exactly, you need to interpolate the new pixel values. You can use the functions `interpolate` and `extrapolate` from the package `Interpolations.jl` for this purpose.
3. **Error minimization**: The Lucas-Kanade algorithm works by updating the parameter vector $\mathbf{p}$ iteratively to minimize the squared error $E = \sum_{x,y}(I_2 - I_w)^2$. For each step of the optimization procedure, record the parameter vector and error value. The algorithm should terminate if $\|\mathbf{p}_{k+1} - \mathbf{p}_k\| < 10^{-5}$ or after a maximum of 40 iterations.

   Begin by calculating the Jacobian matrix J for the given transformation $\mathbf{w}$. It does not change with iterations of the algorithm and therefore has to be calculated only once:

$$J = \frac{\partial \mathbf{w}(\mathbf{x};\mathbf{p})}{\partial \mathbf{p}} \tag{1.2}$$

   In contrast, the spatial derivatives in the warped image have to be recalculated after every update of $\mathbf{p}$. Using the column vector $\mathbf{n} = J^T \cdot \nabla I_w$, the warped image I_w, and the second image I_2, the iterative update can be written as:

$$\mathbf{p}_{k+1} = \mathbf{p}_k + \sum_{x,y} \mathbf{n} \cdot (I_2 - I_w) \cdot \left(\sum_{x,y} \mathbf{n} \cdot \mathbf{n}^T \right)^{-1} \tag{1.3}$$

4. **Results**: Show the difference between the warped original and the target image for every timestep. In addition, display the values for $\|\mathbf{p}\|$ and E across iterations, each in a separate graph. Start the iterative procedure with a parameter vector $\mathbf{p}$ which uses the identity as the warping function.

2 Condensation-Algorithm for Tracking (10 points)

The goal of this task is to track a rectangle over time using the Condensation-Algorithm (also known as particle filter) and SIR (sequential importance resampling). This algorithm tracks the rectangle in the face of several problems that arise from the image data:

- the motion direction changes
- the rectangle accelerates or slows down
- the rectangle is not always completely visible
- the rectangle moves out of the image and re-enters it at a different location
- the signal is corrupted by strong noise

The algorithm tracks the object with the help of a defined number of particles, which are initially distributed evenly over the search area and later accumulate around the rectangle. Each particle comprises 6 parameters. Four of these serve to estimate the properties of the object: position (x, y) and velocity (v_x, v_y). The other two parameters are particle weight w and cumulative weight c

Implement the full algorithm by executing the following steps for `t=1:100`:

- **Initialize** (only once): uniform distribution of all $N = 1000$ particles over the image; normally distributed initial velocities with $\sigma_{vx,vy} = 1.0$; particle weight $w_n = \frac{1}{N}$; cumulative weight $c_n = n \cdot \frac{1}{N}$
- Load the current image and convert it to a grayscale, floating point matrix.
- **Resampling step**: generate a new set of N particles. Each particle should be initialized by drawing a random number $r \in [0, 1]$ uniformly and identifying the particle i with the smallest cumulative weight $c_i \geq r$ (from the set of old particles). The new particle has the same parameters as i.
- **Prediction step** (drift + diffusion): the new particle positions are calculated according to their velocity (drift), followed by diffusion of the position with $\sigma_{x,y} = 0.5$ and diffusion of velocity with $\sigma_{vx,vy} = 1.0$.
- **Measurement**: use the function `measure_probability`, which calculates the new weights based on the distance to points in the image.
- Use the arithmetic mean of the 100 best particles to estimate the current position and velocity. For visualization, color these 100 best particles red and the remaining samples in blue. Estimate the tracked velocity via the difference of the current position from the previous one.

Compare your results to the ground-truth data in the file `position.mat`, i.e. generate a 2D-Plot of the real trajectory and superimpose your estimates on it. Visualize the velocities as two 1D-plots, one for v_x and one for v_y.

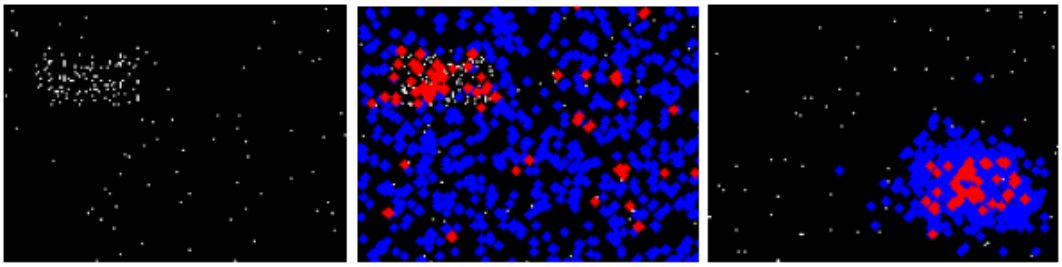

Figure 2: *Left*: rectangle to be tracked. *Middle*: initial distribution of particles with 100 best particles marked in red. *Right*: distribution of particles during tracking.

Submission procedure

- Please work in groups of 2 students. Submission exclusively in Moodle (it is sufficient if one group member submits the solution in Moodle).
- Please name all Julia files with the current assignment and exercise number, e.g. `sh01ex02.jl` for the second assignment of the first assignment sheet.
- Please present your results in a pdf-document with
  - the names of all group members
  - and a brief description of your results and images.
- Please submit all files as a zip-document.

Have fun!