Multiple View Geometry: Exercise Sheet 5

Prof. Dr. Daniel Cremers, Robert Maier, Rui Wang, TU Munich
http://vision.in.tum.de/teaching/ss2016/mvg2016

Exercise: June 9th, 2016

Part II: Practical Exercises

This exercise is to be solved **during the tutorial**.

Harris Corner Detector

In this exercise you will implement the Harris corner detector. *Hint: Again, it is possible to implement this exercise without loops. However, we recommend to start with a loop-based version, and only afterwards replace the loops by single function calls. For quick testing, we provide a small image `small.png`*

1. Download `ex5.zip` from the website.
2. Compute the structure tensor as introduced in the lecture for every pixel of `img1`. As weighting function use a Gaussian Kernel with a standard deviation of $\sigma_w = 3$ pixel, and an integration window size of $2\sigma_w + 1$ pixel. *Hint: use `normpdf` and `conv2`.*

   The structure tensor M for pixel x, y is a 2×2 matrix, which summarizes the structure of the image in the vicinity (determined by σ_w) of the pixel x, y. It can e.g. be interpreted as (weighted) covariance matrix of the image gradient around the pixel, and is computed as

   $$M(x,y) := \sum_{(x',y')} w(x'-x, y'-y) \begin{pmatrix} I_x^2(x',y') & I_x(x',y')I_y(x',y') \\ I_x(x',y')I_y(x',y') & I_y^2(x',y') \end{pmatrix} \quad (1)$$

   where I_x and I_y are the x and y image gradient computed using central differences, and w is the weighting function, here a zero-mean two-dimensional Gaussian with covariance $\sigma_w^2 I_{2\times 2}$.
3. Compute the scoring function $C(x,y) := \det(M(x,y)) + \kappa\, \text{trace}^2(M(x,y))$ using $\kappa = 0.05$. Visualize the scoring function using `imagesc`. *Hint: if you cannot see much, try to display a non-linearly transformed scoring function, e.g. sign(C)* $\cdot |C|^{\frac{1}{4}}$.
4. Find all pixels x for which $C(x) > \theta$, and which are a local maximum of the scoring function, i.e., all four adjacent pixel have a lower score (non-maximum suppression). Use $\theta = 10^{-7}$. Display the found Harris Corners using the provided function `drawPts`.
5. Try different values for σ_w - what do you observe?

Patch Tracking

1. Compute the local velocity of each pixel from `img` to `img2`, using the formula from the slides. You can re-use the computed structure tensor from the first exercise. Visualize your result using `imagesc` *Hint: Implementing this exercise without loop is somewhat tricky.*
2. *Optional: Implement Lucas-Kanade Patch-Tracking, by iteratively re-computing the estimated velocity at the new position. Use it to track the Harris Corners found in exercise 1.*