Prof. Dr.-Ing. A. Bruhn
Computer Vision and Intelligent Systems Group
Institute for Visualization and Interactive Systems
University of Stuttgart

Assignment 5

Programming Exercise 5.1 (Gradient Constancy)

You can download the file `copcv19_ex05.tgz` from ILIAS. To unpack the archive, use

```
tar xzvf copcv19_ex05.tgz
```

1. Supplement the routine `compute_motion_tensor()` in the C programme `horn_schunck.c` with missing code so that it computes the motion tensor of the *gradient constancy assumption*. You can approximate the required second order derivatives f_{xx}, f_{xy}, f_{yy}, f_{xt}, and f_{yt} by first computing f_x, f_y, and f_t and then applying a simple central difference scheme for additional $x-$ and $y-$ derivatives (without averaging). In order to compile your programme please use the contained makefile. The compiled programme is then executed by

   ```
   ./frontend <input_image1.pgm> <input_image2.pgm> <zoom_ratio> [ground_truth.F]
   ```

   where the integer parameter `zoom_ratio` is in general set to 1. The use of a ground truth file `ground_truth.F` is optional and triggers the computation of the average angular error (AAE).

2. Use the provided image pair `yos1.pgm` and `yos2.pgm` to optimise your results with respect to the average angular error (AAE).

Programming Exercise 5.2 (Backward Registration)

Use the same code for the second task.

3. Supplement the routine `backward_registration()` in the same C programme with missing code so that it compensates the second image by a given flow field.

4. You can use this routine by pressing $F8$ after computing a displacement field. The motion compensated second frame ist then written out as file `frame2_bw.pgm`. Use the Linux command

   ```
   animate <frame1.pgm> <frame2_bw.pgm>
   ```

   to visually compare the first and the motion compensated second visually for your best results for the sequences `yos1.pgm` and `yos2.pgm` as well as `rhein1.pgm` and `rhein2.pgm`. Do the results make sense in both cases?