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

Assignment 1

Programming Exercise (Block Matching)

You can download the file `copcv19_ex01.tgz` from ILIAS. To unpack the archive, use

```
tar xzvf copcv19_ex01.tgz
```

1. Supplement the routine `matching_cost()` in the C programme `blockmatching.c` with the missing code such that it implements the presented SSD block matching algorithm.
   All other functions are already provided in the code. In order to compile your programme please use the contained makefile:

   Type `make` to compile the source code and create the executable `frontend`.
   Type `make clean` to remove previously build files (useful if something "fishy" is happening)

   The compiled programme is then executed by

   ```
   ./frontend <input_image1.pgm> <input_image2.pgm> <zoom_ratio>
   ```

   where the integer parameter `zoom_ratio` is in general set to 1. This opens an OpenGL frontend realised with GLUT where you can set the parameters and compute the result. Press `F1` for help!

2. Use the provided image pairs Tsukuba (`tsu1.pgm`, `tsu2.pgm`) and Yosemite (`yos1.pgm`, `yos2.pgm`) to evaluate the performance of the SSD block matching algorithm for different window sizes m.
   The maximum displacement magnitude d for limiting the search space should be chosen $d = 7$ for Yosemite (optic flow) and $d = 13$ for Tsukuba (stereo).

3. Supplement the routine `subpixel_refinement()` in the same C programme `blockmatching.c` with the presented parabola fitting strategy.
   Compare your results visually with and without sub-pixel refinement.

4. Supplement the main routine `BLOCK_MATCHING()` in the same C programme `blockmatching.c` with the presented occlusion detection approach using the forward/backward check. Use the forward flow if the estimation is reliable. If not, i.e. if you have identified an occluded pixel, set the corresponding displacement to zero.
   Vary the occlusion threshold and take a look at the resulting displacement field. Note that the programme either allows you to use subpixel refinement or occlusion detection!