Technische Universität München
Fakultät für Informatik
Prof. Gudrun Klinker, Ph.D.
Andreas Langbein, Adnane Jadid, David Plecher

Summer Term 2018
Exercise Sheet 1
April 18, 2018

Introduction to Augmented Reality

Some general hints about the homework:
Try to accomplish the the exercises at home. Every subsequent week, we will discuss your problems and hopefully also your results together. We will use C++ for the exercises!

Exercise 1 (H) Warmup

Download the marker file (Marker-Cube.pdf) from the lecture website and print it. Also download the video file (MarkerMovie.mp4) containing a scene with several markers.

Exercise 2 (P,H) Getting to know OpenCV

In the first part of the exercises, we will create a simple marker-based optical tracker. For image processing, we will use the library *OpenCV*.

(a) Download *OpenCV 3.2.0* from `http://opencv.org/` and install it on your computer.

(b) Read the documentation and get an overview over the different functions.

(c) If you use VisualStudio, create a new Win32 console project with an empty .cpp file. Add the OpenCV folders to the include/library paths and the OpenCV library names to the linker option. You might want to add OpenCV dll files to your project folder. If you use CMake, an exemplary CMakeLists.txt can be found on the lecture website.

(d) Create a program which displays live webcam stream or the video stream. Use the HighGUI functions `cv::namedWindow, cv::VideoCapture` and `cv::imshow`. The program should quit when ESC (keycode 27) is pressed. If you use a webcam, print an error message if no camera was found. Then, instead, show the video from the file.

Exercise 3 (P,H) Thresholding

In order to detect monochrome markers in the camera image, we first convert the color image into a white-and-black image.

(a) First, convert the camera image to a grey image by `cv::cvtColor`, as the thresholding algorithms work only with grey images. `cv::cvtColor` requires a conversion type–`CV_BGR2GRAY`, indicating from color to gray scale.

(b) Two thresholding functions exist: `cv::threshold` and `cv::AdaptiveThreshold`. Try both methods and show the result in a new window. For `cv::threshold`, create a slider using `cv::createTrackbar` to easily change the thresholding parameter.

(c) Experiment with the parameters. Choose parameters such that the markers are clearly visible under all lighting conditions and viewing angles. They should have a continuous frame which does not merge with the environment.

(d) Briefly describe the advantages and drawbacks of both tresholding algorithms.