High-level Computer Vision
Exercise 1

05.05.2016

Thomas Pohl (2537675)
Guillermo Reyes (2556018)
Daniel Schaefer (2549458)
Dominik Weber (2548553)

Code Annotations

- Throughout our distance functions we allowed 1D, 2D **and 3D** histograms. We also implemented support for 3D histograms because we expect this to be a likely case although it didn't happen in our exercises.

Question 1: Image Filtering

c) **What happens when you apply the following filter combinations?**

1. Gaussian Blur
2. filter for the partial y-derivative
3. filter for the partial x-derivative
4. same result as in 3 (commutativity)
5. same result as in 2 (commutativtiy)

d) We can see two effects:
First of all there is a nice directional behaviour of the partial derivatives which isn't really surprising but can be observed in the fine structures of the face or the wooden mushroom. dy captures here vertical information while dx captures horizontal information.
The second effect are the different values for sigma which determine our Gaussian presmoothing: Lower values preserve almost every structure but also consist of noise while higher values tend to lose fine information but can eliminate noise.

Question 3: Object Identification

c) Our testsresults can be found in the '/Tests3c.txt' file. It basically explains all our testing in addition to what you can read in 4b below.
The best combinations have been intersect distance with num_bins=30 on the rg-histogram with 81 hits. The same amount of hits also occured for chi2 on the rg-histogram using num_bins=30.

Question 4: Performance Evaluation

b) We plotted the behaviour of precision and recall for many 'num_bin' values with all combinations of histogram and distance function. You can find these results in the folder '/RPC-Curves'. The name describes the type of histogram and the number is the value which was used for 'num_bins'. Honestly the best way to compare these histograms and distance functions is to look at the pictues. Regardles a short description of what is happening:

- **dxdy-histogram:**
  The dxdy-histogram is absolutely useless for 'num_bins' values like 5. Afterwards it increases in precision and recall until around 'num_bins' 20 with chi2 being the best distance function followed by intersect and l2. Compared to the other RPC-curves all distance functions stay fairly close in precision and recall. From 'num_bins' 20 to 100 the l2 distance function decreases a little bit regarding both precision and recall but the results of chi2 and intersect stay really close.
- **rg-histogram:**
  The rg-histogram already has really good results for 'num_bins'=5 and has its peak in precision and recall at around 10-20. Its consistently chi2, intersect l2 from best to worst. The difference between chi2 and intersect is fairly significant this time and l2 is decreasing for growing 'num_bins' values just like in the dxdy histogram. The results of chi2 and intersect also decrease with growing 'num_bins' but very very slowly compared to l2.
- **rgb-histogram:**
  the behaviour is really similar to the rg-histogram. For values like 'num_bins'=5 the rg-histogram is significantly superior to the rgb histogram but at around 10 rgb and rg are fairly even in all distance functions. Just like in the other histograms chi2 is consitently the best, closely followed by intersect and then quite a big gap to l2. The behaviour stays basically exactly the same as in the rg-histogram.
- **Summary:**
  Overall we think it's pretty safe to say that chi2 is the distance algorithm of choice here regarding precision and recall for these instances. The l2 distance algorithm is basically always the worst by a large margin. Histogram choice is really close between rg and rgb but especially with higher 'num_bins' rg is superior to rgb histogram. Dxdy is worse in basically every case.
  Regarding 'num_bins' there is apparently a sweet spot between 15-60 for rgb (assuming you use chi2). For rg its 10-100 and for dxdy its 20-100 (it's still really really bad compared to rg or rgb).