Sean Yang

Pd 4

10/24/18

Journal 7

This week I continued to do more research on knn. Initially, I planned to write my own knn code because I wanted to get the distance among feature vectors. I assumed scikit-learn doesn’t give the closest neighbour’s distance from reading knn tutorials online. Writing my own code turns out to be unnecessary and in theory much slower than the version implemented in scikit-learn. I was going to implement a brute force version, but after reading document from scikit-learn, I learned that K-D tree in low dimension and ball-tree in high dimension (D>=20) are both faster than brute force. With D as number of dimensions, brute force has query efficiency of O(DN), while other two method have efficiency of O(Dlog(N)). I also found the method to query closest neighbour: query(*X*, *k=1*, *return\_distance=True*, *dualtree=False*, *breadth\_first=False*). There is a return\_distance parameter which returns the distance of closest neighbour. This further proves that going in depth to actually reading document from library teaches me more and can often save lot of work.

Although I found that by looking at feature vector from pca there is not a big difference among feature vectors, since I need to implement a knn for pixel based classification anyways, I will test out knn on feature vector generated using 68 key point. I also tried to use a new type of cascade called lbp cascade. This cascade is three times faster than haar cascade according to online sources. It seems to be fairly accurate so far and it gives a smaller bounding box compared to haar cascade. This is bad for keypoint detection but good for gesture capture.