Gesture Recognition

Simon Weickert
University of Applied Sciences Würzburg-Schweinfurt
Leipzig, Germany
simon.weickert@student.fhws.de

***Abstract*—In this report the basic steps for recognizing gestures are explained and visualized. From filtering and quantizing the captured data to training the model. Followed by the analysis of how well new date gets recognized by the trained system.**

***Index Terms*—Gesture Recognition**

I. Introduction

In the very short span of five days we learned the basics of gesture recognition. Starting from generating the data on the phone, proceeding with the steps of a basic recognition system. We learned how to filter the data, quantize it with k-Means Algorithm, train a Hidden Markov Model and how to use Naive Bayes to classify the data to the gestures. Additionally I got quickly introduced to python and now also learned to use LaTeX. I would say that I have been party successfull as I got to understand the basics of gesture recognition and a new programming language and can continue to do my own research in this field. My training results are limited though as the time didn't allow for more than I would have wished for.

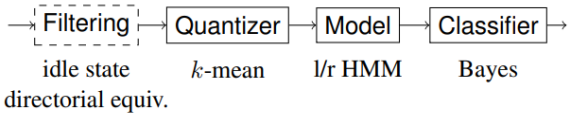

Fig. 1: gesture recognition pipeline [1]

This figure shows an overview of the pipeline that we got introduced to and learned to use for our own datasets.

II. Data Capturing

Multiple gestures have been captured in different ways on my Huawei P30 with the SensorRecorder-app by Toni Fetzer. First of all I started with the classic hand waving, which was done thrice. The movement consisted of the same ingredients:

Hand up $\rightarrow$ *Hand right* $\rightarrow$ *Hand Left* $\rightarrow$ *Hand down*

The difference was lying in the time duration and pauses in between the different states.

1) One hand waving and spending three seconds for each transition (12sec in total)
2) Same as first but with 3 sec stops in between the transitions (24sec in total)
3) Multiple short duration hand wavings (30sec in total)

A buffer time of 3 seconds has been added to the start and end of each run. These buffer times were then cut out afterwards to eliminate random high frequency data from touching the phone (this noise can be seen on the second diagram of Figure 2.

Other actions have been captured as well, like doing a polishing movement and swinging the phone fast in a plastic bag.

III. Filtering

Filtering is important to simplify and reduce the data size. An idle filter has been used to eliminate the noise when no action is being made. There are two special cases which were tested to see how much noise the phone sensors produce.

1) Phone laying on the table
2) Phone being held in the hand

In both cases the intention was to hold still and effectively don't produce any data. But due to the lack of accuracy of the hardware there is still a lot of noise produced. The filter reduces all the vector data to zero if the average of them is higher than delta. This delta was originally set to 0.5 which worked alright for some data and bad for other ones depending on the speed of the gestures. After

examining the idle states it got clear that a delta of 0.2 was sufficient enough to easily ignore lying idle states and also ignore the resting in hand idle states. For very slow movements like in the slow hand waving files where each transition took 3 seconds a delta of 0.15 worked best.

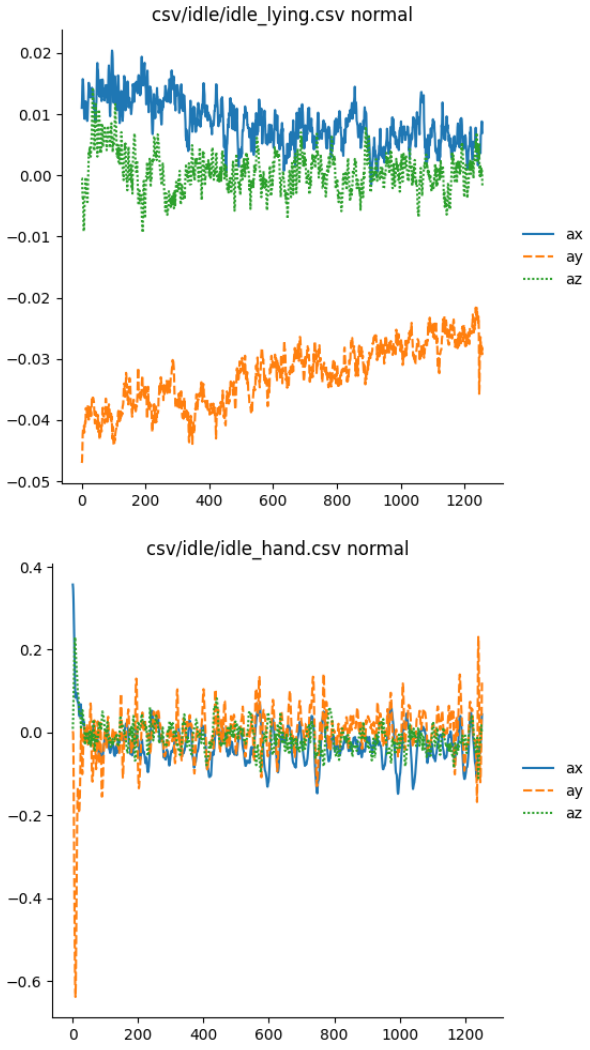

Fig. 2: top: phone lying on desk
bottom: phone hold in hand

Another way of drastically reducing the data size can be achieved with directorial equivalence filtering. The idea is that if the vector v_t didn't change by a significant amount (e.g. it doesn't exceed epsillon which is set to 0.2 m/s²) compared to each of its four predecessors v_{t-1}, v_{t-2},v_{t-3},v_{t-4} then the change basically gets nullified.

IV. Quantization

Using k-Means to get quantized data is a very efficient and easy approach to get cluster centers.

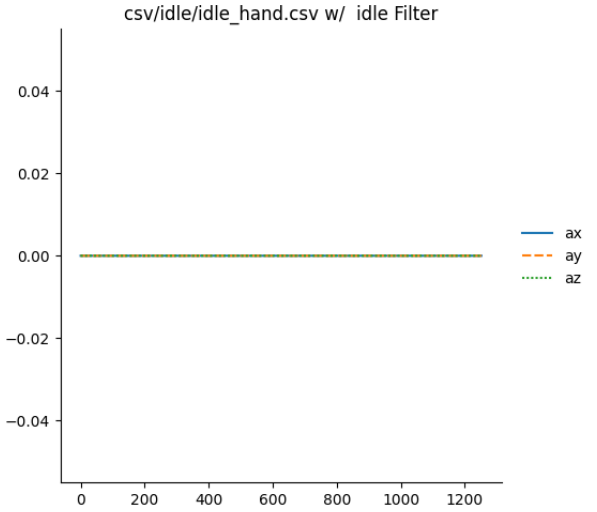

Fig. 3: idle cases filtered with delta = 0.5

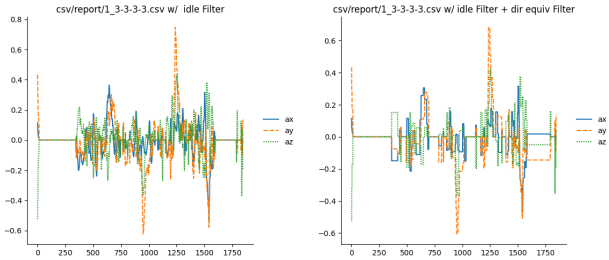

Fig. 4: slow hand waving with idle filter and on the additionally with the directorial equivalent filtering

Which were used to quantize the data.

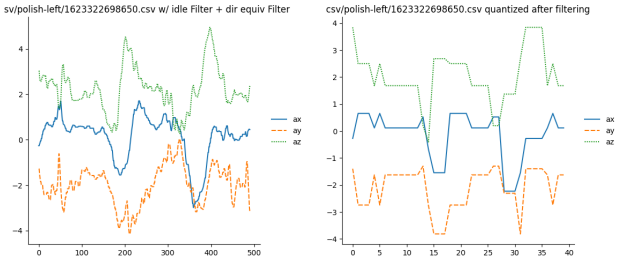

Fig. 5: example of polishing left gesture: only filtered (left) vs quantized (right)

In the paper [1] there has been made investigations on which size of clusters work best. They concluded that 14 clusters for a 3-D coverage is sufficient. I Thought that eight cluster centers would be enough as the movements (waving, polishing, swinging) are all happening on a 2-D plane. But to have nicely aligned clusters like seen in [1] I need to define the centers manually. Then the problem

emerges that it's hard to define how exactly the plane is aligned and how far off the origin the vectors should be placed. As this was out of scope I just tried different cluster numbers $k = 4$, $k = 8$, $k = 14$, $k = 18$ and indeed Quantization using 14 clusters performed best for my data set.

V. Dimensionality Reduction

For the Reduction of the sample size I tested two different methods. One approach was resampling the data using decimation. Which means that every n^{th} data point of the sample gets eliminated where n is depending on the target length. The other approach was to remove duplicates as only the transitions hold the important information. These kinds of reduction seemed to be sufficient enough, so no further experiments were done with spectral methods.

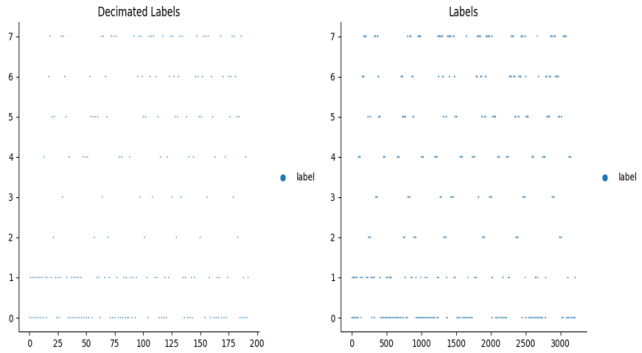

Fig. 6: removed duplications

VI. Hidden Markov Model

The Hidden Markov Model (HMM) receives a number of hidden states and is trained with a sequence of labels, where each label stands for a group of closely related acceleration vectors. The HMM returns a most likely state sequence corresponding to the input data. I trained multiple models with the different gestures.The Model makes an estimation based on the observations. The sequence of data was afterwards forwarded to the multinomial Naive Bayes classifiers which outputs the propabilities of fitting to the gestures.

VII. Evaluation

The data on which the model was trained got easily detected by the system as expected. When trying to put fresh test data into the system, it sometimes struggled to detect it, going under 30% in some cases. I noticed that my Naive Bayes Classifiers recognized a data set where I did some fast hand wavings as polishing gesture. I also used some random datesets to test if true negatives correctly happen e.g. a new gesture like holding the phone in the hand and then doing burpees (sport exercise where you go from lying to jumping and repeat). Indeed this gesture was correctly identified as new and got propabilities of 0-4% which was to be expected as it is completely different to all the training gestures.

Acknowledgment

Thank you Valentin Zitzmann for joining me in the programming sessions.
Learning python, debugging and testing out data with you made it more interesting Thank you Marcel Kyas for holding the lectures and introducing us to the world of Motion Tracking / Gesture Recognition. Also thanks for sharing interesting facts about Iceland.

References

[1] T. Schlömer, B. Poppinga, N. Henze and S. Boll, *Gesture recognition with a wii controller*, Proc. 2nd Int. Conf. Tangible Embedded Interact. (TEI), pp. 11-14, 2008.