A real-time applicable dynamic hand gesture recognition framework

Thomas Kopinski
University Ruhr West
Computer Science Institute
46236 Bottrop, Germany
Email: thomas.kopinski@hs-rw.de

Alexander Gepperth
ENSTA ParisTech
828 Blvd des Maréchaux
91762 Palaiseau, France
Email: alexander.gepperth@ensta-paristech.fr

Uwe Handmann
University Ruhr West
Computer Science Institute
46236 Bottrop, Germany
Email: uwe.handmann@hs-rw.de

***Abstract*—We present a system for efficient dynamic hand gesture recognition based on a single time-of-flight sensor. As opposed to other approaches, we simply rely on depth data to interpret user movement with the hand in mid-air. We set up a large database to train multilayer perceptrons (MLPs) which are subsequently used for classification of static hand poses that define the targeted dynamic gestures. In order to remain robust against noise and to balance the low sensor resolution, PCA is used for data cropping and highly descriptive features, obtainable in real-time, are presented. Our simple yet efficient definition of a dynamic hand gesture shows how strong results are achievable in an automotive environment allowing for interesting and sophisticated applications to be realized.**

I. Introduction

Hand gesture recognition, as an intuitive supplementary means, finds its way into many fields of human-machine interaction (HMI) and our every day life. The challenges are manifold and strongly depend on the given task as e.g. fingers may occlude themselves, the user has little time to react to system feedback or changing environmental conditions can present additional difficult hurdles which need to be reckoned with. Within the automotive environment, hand gestures performed mid-air can have different application scenarios such as controlling infotainment systems, HUDs or displaying vehicle information. The driver should, in a typical driving scene, be able to focus on the environment and has a limited interaction space due to different objects in the vehicle interior.
We present a hand gesture recognition system with a single depth sensor allowing us to embed the system into any kind of environment regardless its lighting conditions, as we make use of a single time-of-flight sensor (ToF-sensor). We demonstrate how our setup is easily embedded into a car interior and provide interesting interaction scenarios (cf. Fig.1). Our approach is almost purely data-driven as we set up a large database to train multilayer-perceptrons (MLPs) for classifying static poses that define each of the targeted dynamic gestures. We present a simple yet robust method of defining dynamic hand gestures in such a way that there is no need to define complex models or implement sophisticated tracking techniques. This paper is set up as follows: After we present an extensive overview over state of the art methods in Sec.II we give a definition of our approach and outline the main differences. We go on to describe our system setup in Sec.III and describe the advantages of doing so. Furthermore in Sec.IV we present our database comprising a large number of data samples which builds the foundation for our system. We then describe the employed holistic point cloud descriptors in Sec.V as well as the parametrization of the MLPs and the underlying fusion technique in Sec.VI. The main idea of defining dynamic hand gestures is outlined in Sec.VII. We prove that our system is able to perform well in real-time by extensive test runs on unseen participants followed by a statistical analysis of our experiments in Sec.VIII. We conclude with a discussion of this approach and by giving an outlook on future work in Sec.IX.

II. Related Work

Dynamic Gesture recognition poses two main problems, referred to as the spatial segmentation (where does the gesture start and end) and the temporal segmentation (when does it start and end), together denoted as spatiotemporal segmentation in [1]. In this article, the authors propose a complex framework consisting of multiple lower- and higher-level modules processing information between each other. Our work differs, in that we have a purely depth-based and data-driven approach. Our system exploits the already established static hand gesture recognition framework and opposed to [1] we prove that dealing with the subgesture problem is a viable approach realisable efficiently in real-time. Moreover our suggestion does not have to deal with detecting the optimal sequence within a timeframe and classification is done reliably by an MLP.
Kurakin et al. [2] make use of a Kinect sensor to solve the classification task of disambiguating between 12 dynamic American Sign Language (ASL) gestures. However the Kinect is not really applicable under daylight conditions, their feature selection is more complex and the test set is very small which makes it hard to compare the efficiency of the system. Moreover our approach does neither rely on tracking nor on normalization of the hand cluster.
Vision based approaches typically rely on the detection of hand pixels [3], employ tracking algorithms and HMMs to detect dynamic gestures [4] or finite-state machines (FSMs) to define a dynamic movement via a sequence of static states. The visual approach in [5] disambiguates between six ASL signs with an FSM, however it forms complex feature vectors to encode finger movements coming from a glove. FSMs are also used in the approach from [6] where the hand and the head of the user are tracked. The process of feature generation is comparatively complicated and the gesture test set is rather simple while the FSM model itself can become very complex. Our approach avoids these cumbersome steps, allows for uncertainties to occur while remaining easy to define overall.

The problem of detecting hand gestures is tackled in [7] via statistical modeling which is something we want to avoid as it adds more complexity to the task. Malima et al. [8] show a fast approach relying heavily on skin color for detection and segmentation and on calculating the center of gravity to define the hand region itself. However it remains unclear how expressive this method is as the tests conducted do not seem to be extremely representative.
While the approaches for hand gesture recognition are covered in a more general way in [9], Suarez et al. [10] give a more extensive overview of hand gesture recognition with depth information.
The main differences to our approach are that we rely solely on depth data, so calibration is not necessary, and neither is the definition of a complex hand model. Aiming at interesting application scenarios, we define complex dynamic hand gestures easily by our approach arguing that dealing with a subgesture problem can be dealt with efficiently in real-time, as we show in this contribution. Moreover we show that by our definition we retain the possibility to easily extend our system to contain more dynamic gestures and even more complex ones.

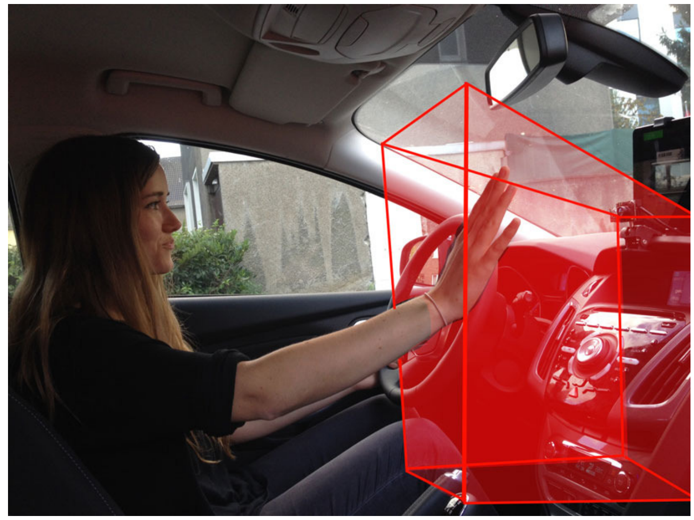

Fig. 1. A typical application scenario: The driver interacts with the infotainment system. The ToF-sensor captures the sensitive VOI marked in red.

III. System Setup

Our hand pose recognition pipeline consists of a ToF-sensor, a point-cloud cropping module, a feature transformation module, a neural network architecture and a graphical user interface displaying the user feedback for the detected static hand poses as well as the dynamic hand gestures. The ToF-sensor is recording the nearby user environment (see Fig.1) and is connected to a standard laptop running the Ubuntu OS. In an initial step all surrounding data points except for a designated volume of interest (VOI) are cropped. The resulting point cloud is again reduced to the minimum data needed via the PCA algorithm (principal component analysis) resulting in fingers, palm and wrist only. The remaining point cloud data is transformed via the descriptor described in Sec.V into a histogram forming a so-called feature vector characterizing the shape of the cloud. The recognition task is done by MLPs trained on a large database yielding a score for the each class for a designated point cloud at any point in time t. All detections of static and dynamic hand poses are visualized in a GUI (cf. Fig.4) displaying the inactive state, the current static hand pose and the currently (if any) performed dynamic hand gesture. The system is able to work at a framerate of up to 20Hz which is more than sufficient as we will demonstrate later on.

IV. Static hand pose database

Our hand pose database builds the foundation of our hand gesture recognition system. We recorded 600000 samples coming from 20 different persons with the Camboard Nano sensor (see [11] for details on this low-cost ToF sensor). Our intention was to define a set of hand poses which is difficult to disambiguate while being also meaningful in such a way as that each hand pose can have a clearly recognizable application scenario in the field of HMI. Additionally we wanted the poses to provide the basis for the definition of dynamic hand gestures as described later on. Since the orientation and position of the hand can vary significantly depending on the user and the application scenario the participants were asked to translate and rotate their hand during recording in order to capture this variance. Moreover, in order to deal with the scaling problem, for each of the poses three ranges were defined - near, intermediate and far. During the recording we made sure that for the 3000 samples recorded from each person per hand pose equally many, i.e. 1000, were recorded in each range. The data was captured by a single ToF-sensor which was set to a adequate illumination time for near-range interaction and cropped the scene appropriately in order to get rid of irrelevant background data. The database comprises ten different hand poses denoted a-j (cf.Fig. 2).

V. PCA and Point Cloud Descriptors

A. Principal Component Analysis for Point Cloud Cropping

The main directions of the cloud are found using Principal Component Analysis (PCA) [12]. PCA aims to find uncorrelated basis vectors for an arbitrary set of data vectors. Eigenvectors (also termed "principal components") are ordered by the variance of data points projected onto them, allowing efficient data compression by omitting principal components of low variance. This algorithm is applied as shown below, using as input the set of n 3D coordinates of points in a point cloud denoted x_j, $j \in [0, n]$).

- The mean value $\bar{x} = \frac{1}{n} \cdot \sum_{j=1}^{n}(x_j)$ is computed.
- The scatter matrix is calculated :

  $$S = \sum_{j=1}^{n}(x_j - \bar{x})(x_j - \bar{x})^\top$$

  This matrix can be used as maximum-likelihood estimate of the covariance matrix.
- The Eigenvectors of this matrix yield the principal components.

We intend to cut off 'unnecessary' parts of the cloud, i.e. outliers and elongated parts of the forearm. In this case, the principal components correspond to orthogonal vectors that

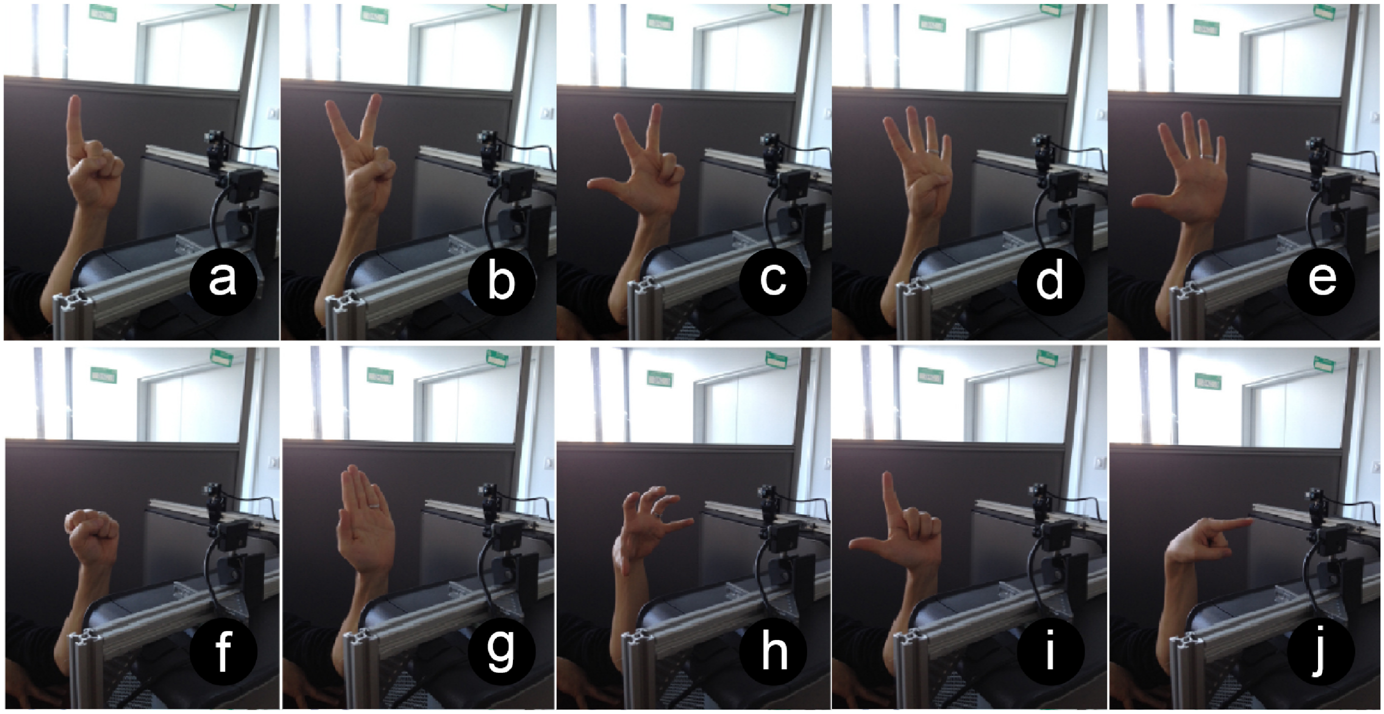

Fig. 2. The static hand pose database consisting of ten different hand poses denoted a-j.

represent the most important directions in the point cloud. The vector with the most important y-component allows to recognize the axis hand-forearm.

The wrist, as the link between the hand and the forearm, is detected in order to determine a limit for the cropping. The employed method assumes that the distance between the endpoint of the fingers and the centroid is an upper bound of the distance between the centroid and the wrist.

To find the endpoint of the hand towards the direction of the fingers, tests are made along the axis, starting at the centroid and moving progressively upward. At each step, we determine whether there are points within a designated small neighborhood around the axis. The upper end of the hand is marked if this number of neighboring points equals 0. Then the bottom limit for the wrist is fixed at the same distance from the centroid, but in the inversed direction along the y-axis. All points below this wrist limit are cut out which is exemplarily shown in Fig.3.

B. Forming descriptive feature vectors from Point Clouds

The PFH-Descriptor (PFH-Histogram) [13] is a local descriptor which relies on the calculation of normals. It is able to capture the geometry of a requested point for a defined k-neighbourhood. Thus, for a query point and another point within its neighbourhood, four values (the point features or PFs) are being calculated, three of which are angle values and the fourth being the euclidean distance between these two points. The angle components are influenced by each point's normal, so in order to be able to calculate them, all the normals have to be calculated for all points in the cloud. Therefore we are able to capture geometric properties of a point cloud in a sufficient manner, depending on the chosen parameters. These parameters have been thoroughly examined in our previous work which led for example to an optimal choice for the parameter n, the radius for calculation of the sphere which encloses all points used to calculate the normal of a query point. One major drawback is the fact that the PFH-descriptor cannot be easily embedded into a real-time applicable system as the computation cost becomes too high, when extended to a global descriptor. To overcome this issue, we present a modification of the PFH-Descriptor.

Our version of the PFH-Descriptor makes use of its descriptive power while maintaining the real-time applicability. Using the PFH in a global sense would mean having to enlarge the radius so that every two point pairs in the cloud are used to create the descriptor. This quickly results in a quadratically scaling computation problem as a single PFH-calculus would have to be performed 10000 times for a point cloud of 100 points. Given the fact that our point clouds have a minimum size of 200 points up to 2000 points and more, this is not feasible for our purposes. Therefore we randomly choose 10000 point pairs and use the quantized PFs to build a global 625-dimensional histogram. We calculate one descriptor per point cloud which forms the input for the neural network. We have conducted numerous experiments with this descriptor in various application scenarios and found it to be well balanced in terms of descriptiveness and computation cost.

VI. Neural Network architecture and Fusion technique

We trained two MLPs and divided the database accordingly to allow for the implementation of a sophisticated fusion technique. Both MLPs have three layers - input, hidden and output layer. Extensive parameter search in work conducted so far yielded this network structure with 50 hidden neurons in

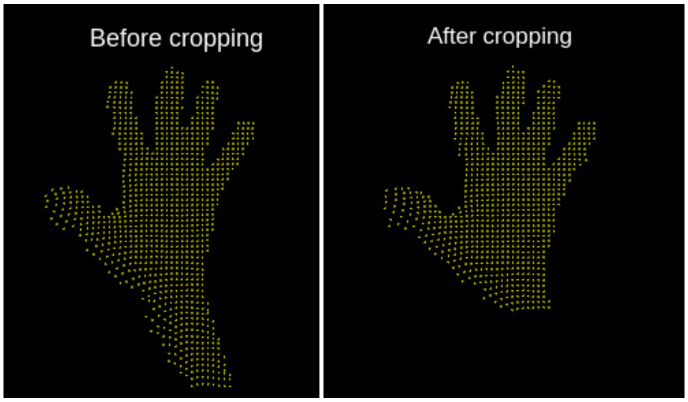

Fig. 3. Point cloud before PCA-cropping (left) and after (right).

each MLP and standard parameters for training. Each output layer comprises 10 neurons corresponding to the 10 hand pose classes. The first MLP has an input layer of size 625, corresponding to the size of the feature vector while the second MLP has an input layer of size 635 - the size of the feature vector added to the number of output neurons of the first MLP. Each Point Cloud is transformed into a histogram of length 625 - as described in Sec.V and fed into the first MLP. The MLP processes the feature vector, determines the neuron values in the output layer and concatenates these values again with the feature vector which is then presented as input into the second MLP. The neuron with the highest activation in the second MLP corresponds to the designated class. For more information please refer to our preceding work in [11],[14] as the theory is beyond the scope of this paper. We have tested various techniques for this problem and this fusion approach resulted in the best generelization performance. The neural network architecture was implemented using the FANN library [15].

VII. Dynamic Hand Gestures

We define hand gestures as being dynamic, i.e. changing in state over time, and they can be contrasted against static hand poses which in turn do not change in state. Therefore, in an in-car infotainment system, a static hand pose as the *one* pose (cf.Fig. 2, hand pose 'a') could be connected to selecting the first audio channel while a dynamic zooming in/out gesture could be applied in a typical maps application. Our approach makes use of the simple fact that a dynamic hand gesture must have a clearly distinguishable starting pose and a clearly distinguishable ending pose. Consequently a 'grabbing' movement can be defined by starting as hand pose 'h' and ending as hand pose 'f' in our hand pose database. This is a clearly defined feature and serves as a universal definition in that any kind of dynamic gesture can be captured in this sense. The number of theoretically definable gestures therefore sums up to $n(n-1) = 90$ gestures definable from our static database, as any case is bidirectional i.e. a gesture from 'a' to 'b' can be performed vice versa.

We denote a static hand pose as a state s at any given point in time t: s_t. A sequence of n occurrences of a certain hand pose is defined as $< s_{t=0}, ..., s_{t=n} >$. During the interaction phase our gesture recognition module takes consecutive snapshots which are interpreted by the system via a voting scheme. For a series of 10 consecutive snapshots, a static hand pose is recognized by the most frequent occurrence within this series if the occurrence is above a certain threshold. In order to take into account that our framerate can vary between 5-20Hz, a threshold of 7 yields satisfactory performance in terms of recognition rate and user acceptance as the feedback has to be provided to the user and in order to suppress too frequent changes.

We use this as a basis of defining dynamic hand postures within this time series as follows. For a dynamic gesture any occurrence of the starting state at any given point in time s_t^{st} followed by any occurrence of the ending state s_{t+m}^{en} with $m \geq 1$ within the observed time series corresponds to the classification of the sequence as containing the dynamic gesture: $< s_{t=0}, ..., s_t^{st}, ..., .s_{t+m}^{en}, ..., s_{t=n} >$. This the most simplified notation which allows for the fact that misclassifications may occur in between the detection of the starting state and the detection of the ending state. The only condition being made here is that both classifications must occur within a certain timeframe and that the starting state must be detected before the ending state.

In order to stabilize recognition results, a simple extension of the definition above can be made. As soon as one occurrence of a starting state is made this starting point of a gesture is only taken as valid if it is immediately followed by one or multiple occurrences of the same state, i.e. $s_t^{st} = s_{t+1}^{st}$. The same rule can be applied for the the ending state of a dynamic gesture. The restriction that these consecutive occurrences of states must form uninterrupted subchains within the observed timeframe suffices to define a robust dynamic gesture recognition pipeline, recognizing dynamic hand gestures well in real-time as we will see in Sec.VIII. Of course a proper choice of parameters is immanent and strongly depends on factors such as framerate, classification rate and user feedback. The choice of the length of the observed timeframe restricts other parameters as the length of the subchains for starting and ending sequences. The benefit of this simple definition is the fact that we allow for uncertain states or even misclassification to occur in between starting and ending sequences of a dynamic gesture as well as within the timeframe as a whole. Additionally, as we found out during the testing phase, this approach provides extra flexibility as every user has a different way of performing a gesture and thus an otherwise more restrictive definition of a dynamic gesture can be too obstructive.

VIII. Experiments and results

The first problem to define is the sensible area or volume of interest (VOI). As opposed to systems working with 2D gestures the user has no way of knowing whether she/he has entered the sensitive area or not. To this end we decided to describe the approximate VOI to each user and let them interact freely. For each recording we asked the user to enter the designated VOI and perform the corresponding gesture.

We have conducted a series of tests with 10 different persons whose data is not contained in the database, i.e., the results given in Table I show the generalisation performance of our system to previously unseen persons. We explained to each person how our system works and realised a GUI containing the system's response whether a gesture was recognized or not. For the experiments in this contribution, we defined a

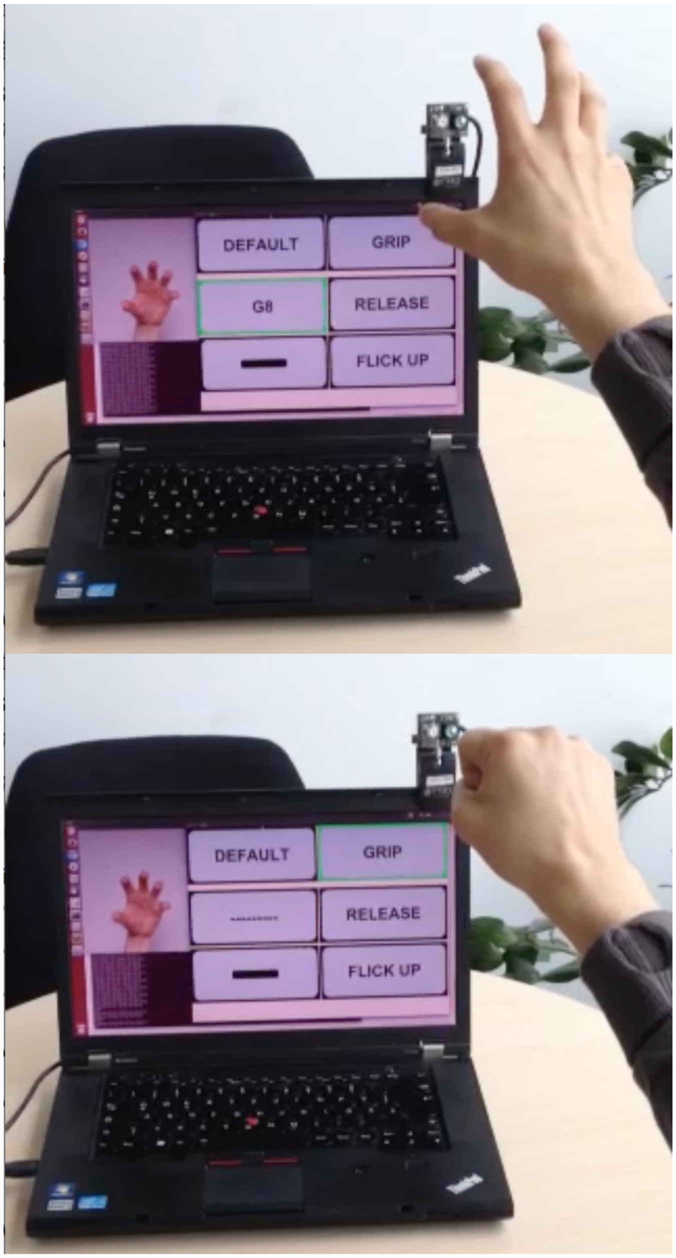

Fig. 4. Demo setup: The user performs the grabbing gesture defined by the starting pose 'h' (top) and the ending pose 'f' (bottom) (cf. Fig.2 for notations)

timeframe of length $n = 10$, meaning that from the moment user input is generated we observe the last 10 consecutive snapshots and the corresponding classifications in order to determine whether a dynamic gesture is contained or not. Moreover, we found that for a timeframe of this size, two identical consecutive starting poses and two identical consecutive ending poses suffice to efficiently detect a dynamic gesture.

Four gestures were defined to this end: Grab, release, zoom in and zoom out. Each gesture can be defined via an unambiguous static state from our database. The grabbing motion (shown in Fig. 4) is defined as starting with hand pose 'h' and ending in hand pose 'f'. The corresponding release gesture is the exact inverse starting with hand pose 'f' and ending with hand pose 'h' (cf. Fig.2). The pinching/zooming gesture is defined analogously and can be seen as the same gesture known from pinching/zooming in 2D in e.g. a typical maps application. To this end, pinching is defined as starting with hand pose 'i' and ending with hand pose 'f'. Consequently the inverse movement from 'f' to 'i' defines the zooming gesture cf. Fig.2. All users found the concept easy to grasp and interacted with our system by this means naturally.

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
grab	10	6	3	5	5	8	6	7	5	4
release	7	6	9	8	9	8	8	9	9	7
zoom in	10	10	10	10	10	10	10	10	10	10
zoom out	9	10	7	10	9	9	10	8	9	9

TABLE I. EACH GESTURE WAS PERFORMED 10 TIMES BY EVERY PERSON. A COLUMN ENTRY REPRESENTS THE NUMBER OF CORRECTLY RECOGNIZED SAMPLES PER PERSON AND GESTURE.

The overall classification rate is 82.25% averaged over all persons and gestures. There is a 100% recognition rate for zooming in, followed by 90% for zooming out, 80% for release and 59% for grabbing. This shows that with a robust detection mechanism for static hand poses our approach resembles a viable solution. However misclassifications still occur as our data recordings show in Tab. II for all hand gestures, although this amounts to only a few cases as the statistics show. In the case of zooming in/out, the misclassifications sum up to 16 and 3 cases respectively, which in turn makes up for 1% or 0.1% of all the cases. For grab/release the numbers are higher, namely 151 and 78 misclassifications respectively which in turn makes up for 10% and less than 5% of all classifications. Comparing these number to the figures in Tab. I helps explaining why the individual gestures perform more poorly as it seems evident that more misclassifications of static hand poses impair the performance of the system. However it also shows that misclassifications are allowed to happen while a gesture is still recognized correctly, which shows the flexibility of our approach. A more in-depth analysis of our recordings reveals that misclassifications occur in 153 cases of all the correctly recognized gestures performed by the participants within the timeframe and between starting and ending sequence. Nevertheless our approach helps to remain robust by dismissing these samples. This shows that such a simple definition of a dynamic gestures is able to provide a satisfactory and stable performance under challenging conditions in real-time. As these statistics also indicate, users tend to remain longer in the final positions of a gesture, nearly 3-4 times longer in some cases (cf. Tab. II). Hence e.g. in the case of 'grabbing' the number of detected ending states (state 'f' - 1160 samples) is more than 3 times higher than the number of detected starting states (state 'h' - 338 samples). Why that is the case is subject to further analysis but it helps to provide further stability mechanisms for the problem at hand.

	a	b	c	d	e	f	g	h	i	j	sum
grab	24	0	3	4	0	1160	13	338	107	0	1649
release	0	0	4	11	0	489	4	1126	59		1693
zoom in	5	0	0	0	0	875	0	11	669	0	1560
zoom out	1	1	0	0	0	308	1	0	1361	0	1361

TABLE II. ALL HAND POSE CLASSIFICATIONS SUMMED UP OVER ALL PERSONS AND SEQUENCES.

The individual results per person differ strongly. Person 1 seemed to be at ease with the system as only 4/40 gestures were not recognized while the result for person 3 are the weakest with 11 misclassifications in total. What seems interesting is the fact that, although the gestures are defined inversely, the results differ significantly. In the case of zooming in/out there is a 10% drop in classification performance and for the release/grab gesture this sums up to about 20% in misclassification rate. The former result can be interpreted in such a way as that misclassifications influence the overall results and may interrupt a sequence leading to mostly no classification at all by our system. The latter is more difficult to explain, but the data recorded suggests that for most persons it is easier to grasp the notion of a releasing gesture than that of a grabbing gesture as evidently most users did not finish the movement to the final state and thus our system was unable to determine whether the motion was finished or not. Most users left their hand half open and in their mind had already finished the movement. This problem can be easily fixed by a relaxed restriction of our definition or by simple user guidance or training.

The average sequence length for performing a gesture is 16.34 which shows that defining the timewindow of length n=10 absolutely suffices for our purposes. One has to take into account that each input cloud above a certain threshold (w.r.t. the point cloud size) is taken as input. This on the one hand suppresses noise and unwanted behaviour and on the other hand stabilizes overall results as 'meaningful' input is favoured. The average time the users needed to perform a gesture was about 1.5-1.7 seconds averaged over all gestures and persons.

However, there still needs to be conducted more research on why misclassifications occur and the figures differ for individual cases although the gestures defined are closely related.

IX. Discussion and future work

We have presented an in-car dynamic hand gesture recognition system based on ToF-sensor data and MLP classification. The sensor is mounted to the front console and captures the nearby driver environment, making our system sensible to user input. Depth data is cropped and transformed into a feature vector capturing the shape of the hand and classified by a sophisticated fusion technique optimized for this problem. The recognition of dynamic gestures from static poses happens via a simple start-end definition of the gesture and experiments on ten unknown persons show that for a real-time application this approach leads to very satisfying results. Our system setup requires only little calibration as the sensor needs to be directed into the car interior while the descriptors and the MLPs are robust enough to make up for invariances in hand orientations, calibration errors and noise resulting from sensor measurement errors or lighting influences.

Experiments show that results may vary for different persons as gestures are not easily definable in a general way but the parametrisation presented in this work builds a good starting point for further research. An optimal configuration with respect to e.g. the duration of a gesture very much depends on the given task, the setting and not least on the personal preferences of the user which may explain the partly large differences in classification rate for similarly defined gestures. Further work will address the most common errors and difficulties. We will improve the system by further refining the dynamic hand gesture definition and provide a statistical analysis of the task at hand. The possibilities provided by our gesture set are promising thus we will seek to define a larger dynamic gesture set driven by possible in-car applications. Our approach is moreover easily extendable to recurring gestures and even more complex cases as first research has already provided promising results in this direction. All research will be paired with a more complex testing environment by realising an infotainment system controllable by dynamic gestures. All data comprising point clouds, feature vectors, whole time sequences as well as timeframes and classification results will be made publicly available.

References

[1] J. Alon, V. Athitsos, Q. Yuan, and S. Sclaroff, "A unified framework for gesture recognition and spatiotemporal gesture segmentation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 9, pp. 1685–1699, 2009.

[2] A. Kurakin, Z. Zhang, and Z. Liu, "A real time system for dynamic hand gesture recognition with a depth sensor," in *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*. IEEE, 2012, pp. 1975–1979.

[3] M.-H. Yang, N. Ahuja, and M. Tabb, "Extraction of 2d motion trajectories and its application to hand gesture recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 8, pp. 1061–1074, 2002.

[4] A. Ramamoorthy, N. Vaswani, S. Chaudhury, and S. Banerjee, "Recognition of dynamic hand gestures," *Pattern Recognition*, vol. 36, no. 9, pp. 2069–2081, 2003.

[5] J. Davis and M. Shah, "Visual gesture recognition," in *Vision, Image and Signal Processing, IEE Proceedings-*, vol. 141, no. 2. IET, 1994, pp. 101–106.

[6] P. Hong, M. Turk, and T. S. Huang, "Gesture modeling and recognition using finite state machines," in *Automatic face and gesture recognition, 2000. proceedings. fourth ieee international conference on*. IEEE, 2000, pp. 410–415.

[7] S. Reifinger, F. Wallhoff, M. Ablassmeier, T. Poitschke, and G. Rigoll, "Static and dynamic hand-gesture recognition for augmented reality applications," in *Human-Computer Interaction. HCI Intelligent Multimodal Interaction Environments*. Springer, 2007, pp. 728–737.

[8] A. Malima, E. Ozgur, and M. Çetin, "A fast algorithm for vision-based hand gesture recognition for robot control," in *Signal Processing and Communications Applications, 2006 IEEE 14th*. IEEE, 2006, pp. 1–4.

[9] S. Mitra and T. Acharya, "Gesture recognition: A survey," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 37, no. 3, pp. 311–324, 2007.

[10] J. Suarez and R. R. Murphy, "Hand gesture recognition with depth images: A review," in *RO-MAN, 2012 IEEE*. IEEE, 2012, pp. 411–417.

[11] T. Kopinski, S. Magand, A. Gepperth, and U. Handmann, "A pragmatic approach to multi-class classification," in *Neural Networks (IJCNN), 2015 International Joint Conference on*. IEEE, 2015, p. to appear.

[12] I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2005.

[13] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz, "Aligning point cloud views using persistent feature histograms," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, 2008, pp. 3384–3391.

[14] T. Kopinski, A. Gepperth, and U. Handmann, "A simple technique for improving multi-class classification with neural networks," *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2015.

[15] S. Nissen, "Implementation of a fast artificial neural network library (fann)," *Report, Department of Computer Science University of Copenhagen (DIKU)*, vol. 31, 2003.