A real-time applicable 3D gesture recognition system for Automobile HMI

Thomas Kopinski[1], Stefan Geisler[1], Louis-Charles Caron[2], Alexander Gepperth[2] and Uwe Handmann[1]

Abstract— **We present a system for 3D hand gesture recognition based on low-cost time-of-flight(ToF) sensors intended for outdoor use in automotive human-machine interaction. As signal quality is impaired compared to Kinect-type sensors, we study several ways to improve performance when a large number of gesture classes is involved. Our system fuses data coming from two ToF sensors which is used to build up a large database and subsequently train a multilayer perceptron (MLP). We demonstrate that we are able to reliably classify a set of ten hand gestures in real-time and describe the setup of the system, the utilised methods as well as possible application scenarios.**

I. Introduction

As "intelligent" devices enter more and more areas of everyday life, the issue of man-machine interaction becomes ever more important. As interaction should be easy and natural for the user and also not require a high cognitive load, non-verbal means of interaction such as hand gestures will play a decisive role in this field of research. With the advent of low-cost Kinect-type 3D sensors, and more recently of low-cost ToF sensors (400-500€) that can be applied in outdoor scenarios, the use of point clouds seems a very logical choice. This presents challenges to machine learning approaches as the data dimensionality and sensor noise are high, as well as the number of interesting gesture categories. In this article we build upon earlier results [1] and demonstrate how a system can be developed and integrated into a car in order to be able to classify a gesture alphabet of ten hand poses. Our approach is purely data-driven, i.e. by extending and applying a Pointcloud descriptor to our needs we are able to set up a real-time applicable system which is robust versus daylight interferences, invariant to rotation and translation problems and moreover works without the need to formalise a possibly complicated hand model. We will first discuss the related work relevant for our research (Sec. II) and then go on to describe the setup of our system within an automobile environment. Subsequently we describe the sensors and the used database in Sec. IV. In Sec. V we go on to give an account of the used different holistic point cloud descriptors and explain the meaning of the parameter variations we will test. Sec. VI summarises the implemented NN classes and the choice of parameters. The key questions we will investigate in Sec. VII concern the generalisation error of the NN and the performance of our system in a live demonstration as well as offline and then we outline the procedure of the Live Demonstration in VIII. Lastly in IX we give an outlook of potential improvements of our system as well as our next steps.

II. Related work

Depth sensors allow for an easy and robust solution for recognising hand poses as they can easily deal with tasks as segmentation of the hand/arm from the body by simple thresholding as described in [2]. Several surveys have made use of this feature with various approaches to segmentation. Moreover it is possible to make use of the depth information to distinguish between ambiguous hand postures [3]. Nevertheless, it has not been possible to achieve satisfactory results utilising only a single depth sensor. Either the range of application was limited or the performance results were dissatisfying. Usually a good performance result was achieved with a very limited pose set or if designed for a specific application [4]. However for our purposes we need to employ several hand poses which are partly very difficult to disambiguate. ToF-Sensors - although working at stereo-frame rate - generally suffer from a low resolution which of course makes it difficult to extract proper features. Improved results can be achieved when fusing Stereo Cameras with Depth Sensors, e.g. in [5]. In [6] a single ToF-Sensor is used to detect hand postures with the Viewpoint Feature Histogram.

Various approaches make use of the Kinect's ability to extract depth data and RGB data simultaneously [7]. However this approach relies heavily on finding hand pixels in order to be able to segment the hand correctly. Moreover, approaches utilising the Kinect sensor will always suffer from changing lighting conditions which in our case is no drawback as ToF-sensors show robust results in such situations. [8] also make use of the Kinect sensor's ability to acquire RGB and depth data simultaneously albeit using a hand model as a basis for hand pose detection. Nevertheless this algorithm also relies on finding skin-coloured pixels to allow for segmentation in 2D and 3D as well as tracking the hand. To our knowledge there is no comparable work which is placed in the automotive environment. [9] give a extensive overview of the methods and applications used for hand gesture recognition. One of their insights is that most applications are in the field of robot control, interactive displays/tabletops/whiteboards or sign language recognition. In [10] a case study is made of how the Kinect can be utilised to control E-Mail functions in a car through set of

[1] Thomas Kopinski, Stefan Geisler and Uwe Handmann are with the Hochschule Ruhr West - University of Applied Sciences, Lützowstraße 5, 46236 Bottrop, Germany thomas.kopinski, stefan.geisler, uwe.handmann@hs-ruhrwest.de

[2] Alexander Gepperth and Louis-Charles Caron are with ENSTA ParisTech, 828 Blvd des Maréchaux, 91762 Palaiseau, France alexander.gepperth, lcaron@ensta-paristech.fr

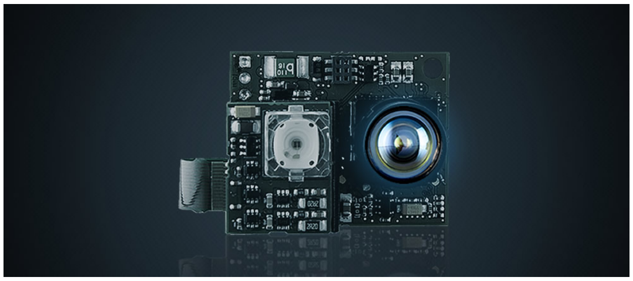

Fig. 1: The Camboard nano

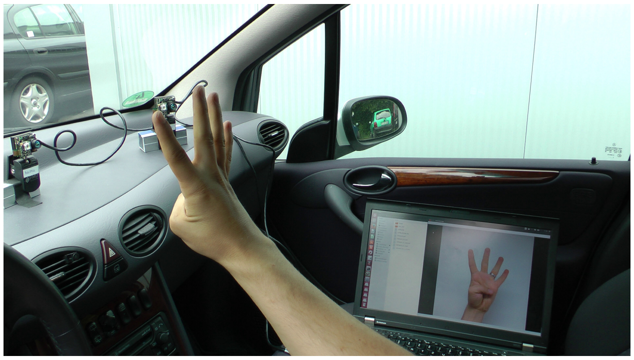

Fig. 2: A sample recording from the Live Demonstration

six hand gestures. While the results remain unclear, except for the fact that gestures could be well accepted as a means of control in a car, the gesture set remains small and the effect of different lighting conditions on the results is not discussed. More comprehensive overviews are given in [11] and [12].

Beneath the technology development research is conducted on how to design intuitive user interfaces. Bailly et al. investigate and compare different menu techniques in [13]. Wilson and Benko developed a system with several projectors and depth cameras named LightSpace [14].

In-car scenarios have been developed for several years as the the driver can keep his hands close to the steering wheel while being able to focus on the surrounding environment. Pointing capabilities could be interesting to control content in the head-up displays. A good overview is given in [15].

Such scenarios demand robust data extraction techniques which is provided by the aforementioned ToF-sensor. Our approach shows that it is possible to achieve satisfactory results relying solely on depth data when detecting various hand poses. In merging information from a second depth sensor we are able to boost our results significantly while always retaining the applicability under various lighting conditions - one of the greatest advantages of ToF-sensors compared to e.g. the frequently used Kinect sensor.

III. System Setup

We integrated two ToF Sensors into a car, both fixed to the centre console and connected to a Laptop with a Linux system installed, handling the computation task. Taking our previous results into consideration, we found a 30 degree angle for the setup of the cameras to be sufficiently suitable in order to be able to disambiguate even the more difficult hand poses. To this end, we placed one camera in the centre of the console and the other one slightly shifted and rotated to the right from the driver's perspective due to obvious space-requirements (cf. Figure 2). As opposed to our previous research, we focus on recognising the subject's (i.e. driver) right hand for the desired hand poses. Therefore we defined a desired Volume of Interest (VOI) within which we want to identify hand poses. Due to driver behaviour, possible range and convenience as well as obstacle occlusion (e.g. steering wheel) our VOI is of trapezoidal shape with a depth of 27-35cm, a maximum width of 60cm and a minimum width of about 45cm enclosed by the FoV (Field of View) of the camera frustum. The total height covered by our cameras ranges from 30-35cm. It is important to note that both cameras have been set up with the same parameters in order to roughly have the same distance to the recorded object as well as the same VOI at any given point t in time. Furthermore we cover a space big enough to recognise the most important movements in the car. Usually the driver has his hand on the steering wheel or close to it or, in other situations leans onto the armrest, which differs significantly in position and allows for longer interaction with our system. By defining our VOI as described, we are able to cover these possibilities.

IV. The Database

The database was recorded using two ToF-Sensors (cf. Figures 1 and 2) of type Camboard nano which provide depth images of resolution 165x120px with a frame rate of 90fps.

The illumination wavelength is 850nm which makes the cameras applicable in various light conditions whilst maintaining robustness versus daylight interferences. Since the ToF-principle works by measuring the time the emitted light needs to travel from the sensor to an object and back pixelwise, the light is modulated by a frequency of 30MHz in order to be able to distinguish it from interferences. In a multi-sensor setup however this may lead to a distortion of measurements since both sensors have the same modulation frequency. To avoid such measurement errors, the data was recorded by taking alternating snapshots from each sensor. As can be seen in Figure 2 the cameras are mounted in a fixed position at a distance of approx. 25cm and a 30 degree angle from the recorded object. This allows for a recording of the database such that the hand can be placed in an equal distance of about 20cm - 45cm from each camera to the centroid of the resulting point cloud data set and therefore each camera can also be calibrated to its needs. For the current experiments, focus has been put on the recognition of static *hand poses* which are contrasted to dynamic *hand gestures*. Each set of poses was recorded with a variation of the hand posture in terms of translation and rotation of the hand and fingers. Moreover the driver was able to place her/his arm on the armrest which is reasonable in order to allow for creating a reproducible scenario which is also realistic in terms of applicability as possible (longer) user interaction might well occur in this manner. This results in an alphabet of ten hand poses: Counting from 1-5 and *fist*, *stop*, *grip*, *L*, *point* denoted by *a-j* (cf. Figure 3). For each pose, a set of 2000 point clouds was recorded for each camera. Since we recorded hand poses from ten different persons independently, this yields a data set of 400.000 samples. Each person was asked to move and rotate the hand to a sufficient degree as to ensure enough variance in the pointclouds. This is needed, since our approach is meant to be as invariant to translation and rotation as possible, also considering the fact that hand size varies significantly between the probands.

The main advantage of using a ToF camera is that it allows for outdoor use as e.g. contrasted to a Kinect. This is well demonstrated in our database as well as in the live demonstration as we recorded many samples over varying lighting conditions during bright or dim daylight without impeding the performance of the system. In the chosen probands, both male and female, the size of the hand ranged from 8,5cm - 9,5cm in width and from 17,0cm - 19,5cm in length.

V. Point cloud descriptors

All used global descriptors were calculated using methods of the publicly available Point Cloud Library (PCL) which were adapted to our needs.

A. The PFH-Descriptor

The PFH-Descriptor (PFH-Histogram) [16] is a local descriptor which relies on the calculation of the normals. It is able to capture the geometry of a requested point for a defined k-neighbourhood. So for a query point and another point within this 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 we extend it to be a global descriptor. To overcome this issue, we present a modification of the PFH-Descriptor in the following section.

B. 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 camera and concatenate all obtained descriptors in order to serve as inputs to the neural network.

VI. Neural network classification and fusion

With M cameras, our system produces M descriptors per frame according to the methods described above. We use a multilayer perceptron (MLP) network[17] to model the multi-class classification function, either implicitly by concatenating all N descriptors followed by NN classification (”early fusion”), or explicitly, by performing classification individually on each of the N descriptors and then combining results (”late fusion”). Neural networks contain a bias unit at each layer, the training algorithm is ”RProp”[17] with hyperparameters $\eta^+ = 1.2$, $\eta^- = 0.6$, $\Delta_0 = 0.1$, $\Delta_{\text{min}} = 10^{-10}$ and $\Delta\text{max} = 5$. Network topology is NK-16-10 (hidden layer sizes from 10-500 were tested without finding significant performance improvements), K indicating the size of the used descriptors and N the number of cameras, here $N = 2$. Activation functions are sigmoid throughout the network. As the MLP classifiers have 10 output neurons, corresponding to the 10 gesture classes, with activities o_i, the final classification decision is obtained by taking the class of the neuron with the highest output. However, we do not necessarily wish for every classification to be taken seriously, and thus implement a simple extension based on a *confidence measure* $\text{conf}(\{o_i\})$. Final decisions are thus either taken by determining the neuron with maximal output, or by applying the confidence measure in the following way:

$$\text{class} = \begin{cases} \text{argmax}_i o_i & \text{if conf}(\{o_i\}) > \theta_{\text{conf}} \\ \text{no decision} & \text{else} \end{cases}$$

The confidence measure performs a mapping from $\mathbb{R}^{10} \rightarrow \mathbb{R}$ and shall be denoted ”confOfMax”:

$$\text{confOfMax}(\{o_i\}) = \max\ o_i \qquad (1)$$

For performing late fusion, that is, obtaining two independent classifications o_i^1, o_i^2 based on each camera's features, we simply calculate the arithmetic mean of both output vectors: $o_i^F = 0.5(o_i^1 + o_i^2)$. This intrinsically takes into account the variance in each response, as an output distribution strongly peaked on one class will dominate a flat (or less peaked) distribution. The resulting output distribution o_i^F can then be subjected to the decision rule of Eqn. (1).

VII. The Experiments

The experiments were conducted in such manner as to test the influence of the various parameters on the classification results. We tested various settings for the NN parameters and ended up having three layers of 1250, 16 and 10 neurons -

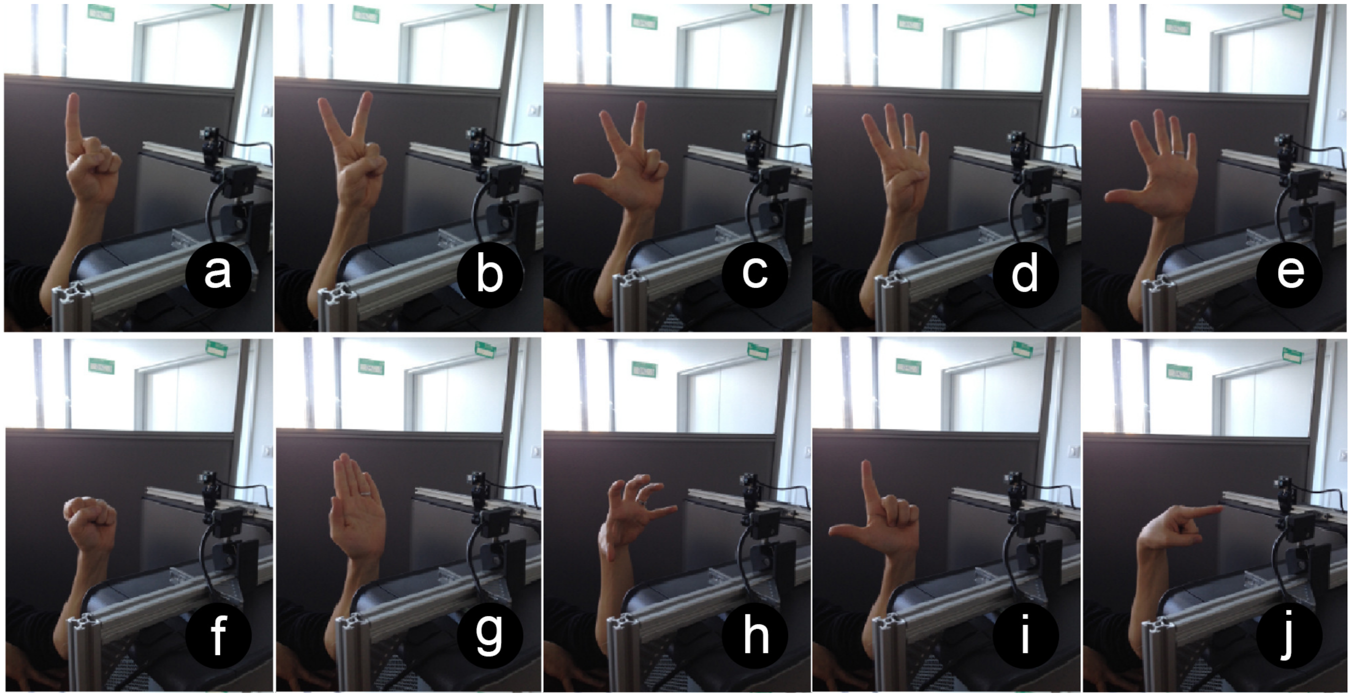

Fig. 3: The hand gesture database: A set of ten hand gestures, counting from 1-5 and *fist*, *flat*, *grip*, *L*, *point*

i.e. one hidden layer - for the input, hidden and output layer respectively. The number of neurons for the hidden layer was chosen as such because the overall performance of the NN peaked at this value during the course of the experiments.

The used NN code was taken from the OpenCV library. We used Rprop as the training algorithm and a sigmoid activation function. The input for the network is formed by taking the input cloud and create the customised descriptor for it. Doing this for each camera and concatenating them together forms the input which is fed into the NN. This procedure is done for training, testing as well as for the live demonstration described later on.

A. *Baseline performance*

The first experiment splits all gesture and person samples 50/50 for training and classification, which leads to an overall recognition rate of 95 percent without even applying the confidence measure as indicated in eqn.(1). Table I displays the number of correctly classified hand gestures enumerated a-f (cf. Fig. 3). One row adds up to ∼10000 samples, because this is how many were included in the set to test the classification error of the NN. From Tab. I we obtain an overall classification error of 6.3%. A few things can be observed:

	a	b	c	d	e	f	g	h	i	j
a	9503	87	17	14	11	48	28	4	129	104
b	96	9470	183	109	1	36	16	10	29	30
c	145	406	8913	181	27	21	45	46	97	20
d	19	64	107	9191	241	22	138	56	8	18
e	21	15	15	191	8986	8	679	41	11	8
f	410	29	26	66	49	8971	153	34	72	85
g	13	8	8	75	30	17	9758	26	18	8
h	7	7	50	94	180	10	20	9454	54	31
i	238	30	45	17	8	25	10	12	9444	74
j	347	33	15	29	11	66	25	15	72	9193

TABLE I: The overall classification results on the complete data set. The overall classification error is 6.3% while this varies from 2.5% to 10.8% for individual gesture classes.

- overall the performance of the net seems to vary between gestures: most notably c, e and f are the ones on which the NN performs worst, as opposed to gestures a and g, which are recognised best - 9503, 9758 respectively out of nearly 10000 samples recognised correctly.
- for every gesture - but above all, for the ones recognised worst - we can identify the one, for which it was mistaken the most as for instance gesture b is likely to be held for gesture c and vice versa; in other words, ’two’ is likely to be held for ’three’ and ’fist’ is likely to be held for ’one’. This makes sense as these poses differ only slightly in terms of pointcloud size, shape or appearance in general. Similar observations can be made for gesture pairs (c ↔ e), (e ↔ g) or (f ↔ a).
- it is important to note, that while one gesture may be mistaken for another, the reverse is not always the case, at least not in the frequency (compare f → a and a → f)

B. *Generalization to unknown persons*

Table II represents the performance of the NN, trained using all persons except one, on data from the person not contained in the training set. Hence each row now adds up to ∼2000 samples. The overall classification error for this experiment is about 14% - nearly twice as large if compared to the table before. We can also derive a set of interesting observations here:

- the network performs best on poses f, h, i while the worst results are on a and e
- similar to the experiment described above, there are poses likely to be mistaken for others - here most notably: a ↔ i and e ↔ d. While this makes sense overall (compare the hand gestures visualised in the database in Fig. 3 for similarities), the described cases differ from the ones above and it is left to interpretation where this difference comes from. If a certain hand pose from a person differs significantly from the equivalents included in the training set, this case is prone to error.
- we can derive, that it is obviously possible generalise well by training a NN on our database for our given task while the more interesting fact is that this works better for some cases that others. It is e.g. clearly observable that hand pose *a* is very likely to be interpreted as hand pose *i* which can occur due to the fact that this certain proband poses in a similar way for this specific task.

	a	b	c	d	e	f	g	h	i	j
a	1329	3	2	0	0	4	0	0	626	36
b	20	1710	237	11	0	3	0	4	9	6
c	7	116	1799	24	2	2	3	22	22	3
d	0	1	11	1812	145	0	1	29	0	1
e	2	0	3	349	1391	0	68	185	0	2
f	43	0	1	0	0	1900	3	0	40	13
g	1	0	2	10	251	14	1719	3	0	0
h	0	2	17	0	3	0	2	1942	27	7
i	35	6	20	2	0	1	3	22	1907	2
j	176	51	3	1	2	32	1	4	36	1674

TABLE II: The classification results on a data set with one person excluded from the training set and entirely used for testing. The overall classification error is about 14% while this varies from 3% to 35% for individual gesture classes.

C. Improvement of baseline performance by thresholding confidence

Tables III and IV refer to a set of experiments conducted on the whole data set split into equal parts (∼100000 samples for training and testing each) but this time comparing the confidences to a threshold Θ, in order to classify only those samples we deem to be 'confident enough'. As before, several experiments have been run and we shall outline the observations:

- we have tested various confidence parameters Θ for the output neurons ranging between values [0.5,0.95] with 0.05 values for the steps taken; we chose to exemplary show the effect by taking 2 sample results for the values 0.65 and 0.95
- we allow only for accepting the classified category by the NN if the value of the highest neuron is above the chosen confidence value, else we reject the sample. This leads to the fact that with a higher chosen confidence value more samples are rejected (6776 rejected samples for Θ = 0.65 - Tab. III and 34005 rejected samples for Θ = 0.95 - Tab. IV).
- The desired effect of sorting out cases in which we are unsure is achieved by this parameter if we compare these results to the previous ones (cf Tab.I) as the number of false positives drops in all cases.
- We are able to retain most of the true positives and improve the overall classification error which drops to 3.63%
- The disambiguation problem remains for the most difficult cases - as an example gesture *f* is likely to be held for gesture *a* (cf. Fig. 3)

	a	b	c	d	e	f	g	h	i	j
a	9343	49	10	3	4	21	8	0	86	77
b	58	9286	90	57	0	15	5	6	14	12
c	117	269	8470	95	13	9	23	21	62	7
d	6	43	27	8818	104	8	63	22	2	6
e	13	8	6	112	8584	5	573	14	5	2
f	359	15	3	29	8	8607	57	14	42	34
g	9	4	4	35	20	11	9683	15	6	5
h	2	5	27	32	88	6	6	9066	39	11
i	198	11	23	10	3	10	3	4	9278	49
j	281	14	7	9	6	29	10	8	45	8880

TABLE III: The classification results with confidence threshold $\Theta = 0.65$, the overall Classification error is 3.63% - rejected 6776 samples

When applying a very high threshold of Θ = 0.95, the following observations can be made:

- raising Θ close to 1 results in many samples being excluded from accepting as confident (cf. Tab. IV) as more than a third (34005) of all samples were rejected
- in most cases the recognition of false positives drops to (or close to) 0; those cases which are difficult to disambiguate are reflected clearly in the results
- some classes 'suffer' more than others in terms of recognition or rejection rate, e.g. hand poses *c* and *e* are subject to many exclusions which can be interpreted in such a way as that these gestures are probably more ambiguous than others
- we can also state the fact that as (fairly) many samples remain as false positives in some cases (300 in the case of gesture f → a). Confidence must be high for several neurons (i.e. hand poses) in cases like this.
- we are able to lower the classification error to 1.3% on average at the cost of the fact that some gestures are better recognised than others - this can be seen in the video of our live demonstration: we stabilise the overall performance of the system as a whole while recognition results fluctuate for a few examples

	a	b	c	d	e	f	g	h	i	j
	7104	3	0	0	1	4	1	0	8	14
b	13	7655	4	1	0	2	1	0	0	1
c	112	50	4242	9	0	0	2	0	9	1
d	3	4	0	5957	5	0	8	1	0	0
e	9	0	0	12	4120	1	114	2	1	0
f	300	2	0	1	0	5843	4	0	2	7
g	8	1	0	3	0	3	8766	3	1	0
h	0	0	1	3	3	0	1	6336	7	0
i	38	0	0	1	0	0	1	0	7623	9
j	51	0	0	1	1	0	0	0	5	6633

TABLE IV: The classification results with confidence threshold $\Theta = 0.95$, the overall Classification error is 1.31% - 34005 rejected samples

D. Improvement of generalization to unknown persons

Tab. V and Tab. VI show the results of our system when evaluating it on data of a person excluded from the training set while employing the confidence measure. We show exemplary results for the confidence parameters Θ = 0.75 and Θ = 0.85 (cf. Tab. V and Tab. VI respectively). Analogous to the experiments before we derive the following observations:

- The effect of introducing Θ when generalising on unknown data can be described as similar albeit there exist slight differences, namely the fact that we are testing on a smaller number of samples which quickly leads to a significant drop in the recognition of true positives. Overall the system performs well offline - though slightly worse compared to when employing the whole database.
- we achieve error rates close to 0 (or 0) in most of the cases with a (comparatively) small choice of Θ (cf. Tab. V)
- a good choice of Θ is crucial for our classification task as too many samples may be ruled out (cf. gesture *g* in Tab. VI) too early which leads to poor performance results in some cases.
- we are unable to achieve an equally good classification error as before due to the fact that too many samples are rejected (more than 65% in the case of Θ = 0.85) which makes our system too unstable for some cases. As we describe later on, we are averaging results over time when utilising our system. Hence it remains rather acceptable to expect some false positives - which are then ruled out by a simple maximising function - than to increase the confidence in such manner as that too few samples are included. The latter case results in some hand gestures nearly not being recognised at all.
- averaging results over time does not help if Θ is chosen too high. We achieve best results for our Live Demonstration with Θ ranging in [0.6,0.8]
- an overall error measurement has to be taken cautiously, as the number of true positives varies significantly in between the classes *a-j*

	a	b	c	d	e	f	g	h	i	j
a	429	2	0	0	0	0	0	0	333	0
b	1	1119	17	0	0	0	0	0	3	2
c	1	53	919	2	0	0	0	6	4	0
d	0	0	0	1402	8	0	0	0	0	0
e	0	0	0	261	403	0	3	33	0	0
f	1	0	0	0	0	1084	0	0	0	0
g	0	0	0	1	3	1	470	0	0	0
h	0	0	17	0	2	0	0	1727	12	2
i	0	0	0	0	0	0	0	0	1569	1
9	47	0	0	0	0	1	1	0	4	474

TABLE V: The classification results with confidence threshold $\Theta = 0.75$, the classification error is 7.89% with 9560 rejected samples

VIII. The real-time system

The real-time implementation is set up and described as in section III. The proband is seated and positioned in the same manner as the other probands during the recordings. The right arm is placed onto the armrest in order to be able to interact with the system properly. The lighting conditions for this experiment were the same (bright sunlight) as those for the other probands before. The proband tests all ten hand poses in an arbitrary manner, the cameras record the environment as described in Sec. III + Sec. IV and feed the transformed data into the NN which classifies the result. Every recognized hand pose is displayed, with the system running at a frequency of 5-6 Hz.

	a	b	c	d	e	f	g	h	i	j
a	115	0	0	0	0	0	0	0	195	0
b	0	759	4	0	0	0	0	0	0	1
c	1	23	143	1	0	0	0	1	1	0
d	0	0	0	951	2	0	0	0	0	0
e	0	0	0	96	216	0	0	10	0	0
f	1	0	0	0	0	940	0	0	0	0
g	0	0	0	1	0	1	12	0	0	0
h	0	0	2	0	1	0	0	1507	4	1
i	0	0	0	0	0	0	0	0	1154	1
j	16	0	0	0	0	0	1	0	4	203

TABLE VI: The classification results with confidence threshold $\Theta = 0.75$, the classification error is 5.78% with 13610 rejected samples

Test results are in general comparable to the offline system although an exact measurement is not possible as annotations were not yet created for the real-time in-car scenario. Based on a visual inspection of results, we observe very few errors if probands' hand gestures are a part of the training set, otherwise performance suffers but not beyond a certain point. The recognition rate in the latter case is impeded by the fact that many of the chosen poses are difficult to disambiguate (see the used gesture alphabet in Fig. 3), e.g., hand pose 'two' vs. 'three' or 'L' vs. 'one'.

We were able to boost the performance of the real-time system by using the "confOfMax" confidence measure as in the offline experiments. Confidence thresholds in the range of $[0.6, 0.9]$ yield satisfactory results, by determining whether the system is 'sure' enough to recognise and classify a sample. Of course a balance has to be found between the need to have very accurate results, and the need to maintain a sufficiently high frequency of results. As we achieve already a satisfactory frame rate which can conceivably be boosted by optimizing the code, we believe that it is acceptable to reject half of the incoming samples in exchange for a high recognition accuracy.

IX. Conclusion and outlook

In this article, we present a real-time hand gesture recognition system based on inexpensive and robust time-of-flight cameras, intended for human-machine interaction in an automotive environment. Tests on an offline database show excellent generalisation performance for a set of 10 static gestures. We furthermore present an in-car, real-time implementation of this system which is tested, in a non-rigorous way for the time being, on persons whose hands are not at all present in the training data, while still achieving very good performance given that a classification problem with 10 classes is studied. In particular, what we demonstrate is that already a computationally simple confidence measure boosts performance considerably at the cost of ignoring uncertain samples, which to our mind is strongly preferable to proposing an incorrect classification.

In future work, we will build upon these foundations and try to use more powerful heuristics to bring the recognition rate close to 100 percent. Particularly, we intend to investigate the use of other, more advanced confidence measures, as well as the principle of temporal continuity (i.e., classifications do not usually fluctuate very rapidly). Beyond that, it is conceivable to devise schemes that compare output neurons not only based on their activation but also on statistical co-occurrence properties established during training, which might improve the disambiguation of problematic classes.

REFERENCES

[1] Thomas Kopinski, Alexander Gepperth, Stefan Geisler, and Uwe Handmann. Neural network based data fusion for hand pose recognition with multiple tof sensors. *ICANN*, 2014.

[2] S. Oprisescu, C. Rasche, and B. Su. Automatic static hand gesture recognition using tof cameras. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pages 2748–2751. IEEE, 2012.

[3] E. Kollorz, J. Penne, J. Hornegger, and A. Barke. Gesture recognition with a time-of-flight camera. *International Journal of Intelligent Systems Technologies and Applications*, 5(3):334–343, 2008.

[4] S. Soutschek, J. Penne, Jo. Hornegger, and J. Kornhuber. 3-d gesture-based scene navigation in medical imaging applications using time-of-flight cameras. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*, pages 1–6. IEEE, 2008.

[5] Y. Wen, C. Hu, G. Yu, and C. Wang. A robust method of detecting hand gestures using depth sensors. In *Haptic Audio Visual Environments and Games (HAVE), 2012 IEEE International Workshop on*, pages 72–77. IEEE, 2012.

[6] T. Kapuściński, M. Oszust, and M. Wysocki. Hand gesture recognition using time-of-flight camera and viewpoint feature histogram. In *Intelligent Systems in Technical and Medical Diagnostics*, pages 403–414. Springer, 2014.

[7] Matthew Tang. Recognizing hand gestures with Microsoft's kinect. *Web Site: http://www.stanford.edu/ class/ee368/Project_11/ Reports/Tang_Hand_Gesture_Recognition. pdf*, 2011.

[8] Iason Oikonomidis, Nikolaos Kyriazis, and Antonis A Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *BMVC*, pages 1–11, 2011.

[9] Jesus Suarez and Robin R Murphy. Hand gesture recognition with depth images: A review. In *RO-MAN, 2012 IEEE*, pages 411–417. IEEE, 2012.

[10] Andreas Riener, Michael Rossbory, and Alois Ferscha. Natural dvi based on intuitive hand gestures. In *Workshop UX in Cars, Interact*, page 5, 2011.

[11] Zhou Ren, Jingjing Meng, and Junsong Yuan. Depth camera based hand gesture recognition and its applications in human-computer-interaction. In *Information, Communications and Signal Processing (ICICS) 2011 8th International Conference on*, pages 1–5. IEEE, 2011.

[12] Juan Pablo Wachs, Mathias Kölsch, Helman Stern, and Yael Edan. Vision-based hand-gesture applications. *Communications of the ACM*, 54(2):60–71, 2011.

[13] Gilles Bailly, Robert Walter, Jörg Müller, Tongyan Ning, and Eric Lecolinet. Comparing free hand menu techniques for distant displays using linear, marking and finger-count menus. In *Human-Computer Interaction–INTERACT 2011*, pages 248–262. Springer, 2011.

[14] Andrew D Wilson and Hrvoje Benko. Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 273–282. ACM, 2010.

[15] Carl A Pickering, Keith J Burnham, and Michael J Richardson. A research study of hand gesture recognition technologies and applications for human vehicle interaction. In *3rd Conf. on Automotive Electronics*. Citeseer, 2007.

[16] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Aligning point cloud views using persistent feature histograms. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 3384–3391. IEEE, 2008.

[17] S. Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall, 1999.