Translation of Sign Language Into Text Using Kinect for Windows v2

Preeti Amatya, Kateryna Sergieieva, Gerrit Meixner
UniTyLab
Heilbronn University
Heilbronn, Germany
emails: preetiamatya@gmail.com, kateryna.sergieieva@hs-heilbronn.de, gerrit.meixner@hs-heilbronn.de

***Abstract*—This paper proposes methods to recognize and translate dynamic gestures of the German Sign Language (Deutsche Gebärdensprache, DGS) into text using Microsoft Kinect for Windows v2. Two approaches were used for the gesture recognition process: sequence matching using Dynamic Time Warping algorithm and a combination of Visual Gesture Builder along with Dynamic Time Warping. For benchmarking purposes, eleven DGS gestures, which were provided by an expert user from Germany, were taken as a sample dataset. The proposed methods were compared on the basis of computation cost and accuracy of these gestures. The computation time for Dynamic Time Warping increased steadily with increasing number of gestures in the dataset whereas in case of Visual Gesture Builder with Dynamic Time Warping, the computation time remained almost constant. However, the accuracy of Visual Gesture Builder with Dynamic Time Warping was only 20.42% whereas the accuracy of Dynamic Time Warping was 65.45%. On the basis of the results, we recommend Dynamic Time Warping algorithm for small datasets and Visual Gesture Builder with Dynamic Time Warping for large datasets.**

Keywords-Sign Language; Deutsche Gebärdensprache; DGS; German Sign Language; Dynamic Time Warping; Visual Gesture Builder.

I. INTRODUCTION

Sign language is a visual language [27] used by the deaf people around the world. According to the World Federation of the Deaf, there are about 70 million deaf people in the world who use sign language as their mother tongue [1]. It is often compared to spoken languages in terms of "modality difference" [19]. A spoken language is perceived auditorily whereas a sign language is perceived visually [19]. However, sign languages are not based upon the spoken languages [8]. Instead, it has its own grammar, syntax, semantics, and morphology [14][20], which makes it a highly structured language [17]. It is not a universal language [8] and it differs according to the deaf communities across the world. But, most of the sign languages are named after a country, for instance: in the USA – American Sign Language (ASL), in Germany – German Sign Language or Deutsche Gebärdensprache (DGS), in the UK – BSL (British Sign Language), in Poland – Polish Sign Language (PSL), etc. It involves the process of a gestural interaction where gestures consist of signs, which differ from each other by minor changes [3]. These changes include a change in handshape, motion, location, non-manual cues like facial expressions [3], lip movements, and head movements, which complicates the recognition process.

According to the Stokoe's notation, a sign in a sign language comprises of three different features, which are: place where it is made, the distinctive configuration of the hand or hands making it, the action of the hand or hands [28]. In terms of gestures, motion and configuration can be classified according to their positions, which may be static or dynamic. Harling classified the hand gestures into four categories, as follows [11]:

Static Hand Posture, Static Hand Location – SPSL:

1) SPSL includes the most of fingerspelling and numbers of a sign language, which does not involve hand motions. For instance, spelling 'A' in ASL (see Figure 1).

Figure 1. "A" fingerspelling in ASL [29]

2) Dynamic Hand Posture, Static Hand Location – DPSL: DPSL includes sign gestures, which do not have real meaning of a word. It can also be an acronym. For instance, spelling "OK" in ASL. To spell "OK" first "O" is spelled and then "K" (see Figure 2).

Figure 2. "OK" fingerspelling in ASL [29]

3) Static Hand Posture, Dynamic Hand Location – SPDL: SPDL includes gestures like "Thank you" in ASL (see Figure 3) where a hand posture is almost flat and location changes from touching the chin to in-front of and slightly below the chin.

 ISBN: 978-1-61208-616-3

Figure 3. “Thank you” gesture in ASL [21]

4) Dynamic Hand Posture, Dynamic Hand Location – DPDL: it includes gestures like “Hello” in ASL where the hand posture (like the position of thumb in “Hello”) changes along with the location (See Figure 4).

Figure 4. “Hello” gesture in ASL [21]

Signing words of sign language using fingerspelling is a tedious task, and generally, deaf people do not prefer to use it as the main form of communication [24]. Also, most sign languages do not include fingerspelling but include gestures, which represent whole words [26].

This paper is based on the whole word representation of the German Sign Language (DGS) gestures, which are dynamic in nature, i.e., it involves some hand motions. Hence, SPDL and DPDL types of gestures were used for recognition using Microsoft Kinect for Windows v2 [6]. These gestures are used as input to the sign language translator application, which was built for this research. Gesture recognition is done on the basis of two approaches: using Dynamic Time Warping (DTW) algorithm (see Section 4.5) and Visual Gesture Builder (VGB) along with DTW (see Section 4.7). After a gesture is recognized, it is translated into text.

The rest of this paper is structured as follows: Section 2 describes related work done in the field of recognizing sign language. Section 3 describes the methodology used for the project. Section 4 explains the implementation of the approach in detail. Section 5 presents the performance benchmarks and evaluations of both variations of the application in terms of a central processing unit (CPU) cost and accuracy. The deductions made from the benchmarks and results, as well as further work, are discussed in Section 6.

II. Related Work

A lot of research has been done in the field of sign language recognition using various approaches. Oszust and Wysocki recognized isolated words of PSL on the basis of “features which include Kinect’s skeletal image” and “features describing hands as skin colored regions” [18]. Cooper recognized sign language on the basis of linguistic subunits using Markov Models and Sequential Pattern Boosting based on openNI frameworks [3]. For learning appearance based on subunits (location, motion, hand arrangements), hand segmentation and the position of the face required a user needs to wear data gloves [3].]

Starner recognized ASL sentences using single camera based on Hidden Markov Model (HMM) with word accuracy of 99.2% without modeling the fingers [26]. In his method, a user needs to wear distinct hand gloves [26]. Videos were captured at 5 frames per second with a 320x243 pixel resolution [26]. Fang and Gao proposed Transition Movement Model (TMM) for large vocabulary continuous sign language recognition [9]. The devices used for experimentation were Cybergloves and Pohelmus 3SPACE-position trackers [9]. Zhang, Zhou, and Li recognized sentences on the basis of Discrete HMM and DTW [30]. DTW was used for determining the end point of each sign [30].

Another widely used approach to gesture recognition is gesture classification with a help of machine learning techniques. Neural networks [23], support vector machine [25] or nearest neighbor [5] are often used for such purposes. Although, these methods show high accuracy level, they require not only determining the signing motion, but also design, fine-tuning and training of algorithmic model for classification.

In contrast to these approaches for sign language recognition, the methods used in this paper do not use hand segmentation approaches, and they do not use data gloves. Since our methods do not consider hand posture and focus on isolated words, only DTW algorithm was used for determining the signing motions performed by a user. The only constraint in DTW based approach is that a user has to stay in standstill positions to make a gesture, whereas VGB with DTW has no such constraints. The body joints tracking feature provided by Microsoft Kinect SDK 2.0 was used for sign language recognition, which does not require further color segmentation. Furthermore, we use VGB, which was released with Microsoft Kinect SDK 2.0 to recognize the start and end positions. Our training sets for start and end positions of a sign language is done by VGB application, which provides more custom features for our datasets. For instance: ignoring hands, ignoring lower body joints. Also, the VGB datasets can be experimented prior to recognition using VGB view to determine if the dataset is appropriate for sign language recognition. Finally, this paper also investigates the potential of DTW algorithm and VGB with DTW to recognize sign language gestures.

 ISBN: 978-1-61208-616-3

III. METHODOLOGY

To provide gesture recognition for DGS and its translation into the text, by means of DTW and VGB-with-DTW-based Translators, both learning machines have to be trained to perform this task. For these purposes, the system should be developed which cover data collection, which includes collecting data from the user, specifically filming videos with DGS gestures; data preprocessing, that has to transform video files into gesture dataset in a format acceptable for the translators; and translators training.

For the sample dataset, 11 DSG gestures were selected for recording. Video recording was decided to make with Microsoft Kinect for Windows v2 with SDK 2.0. It was chosen for this project because it can detect full body movements [6] using raw color frames and depth images. The sensors of Kinect include a color camera, a depth sensor, and IR (Infra-Red) emitter. The depth sensor and IR emitter of Kinect has a resolution of 512x424 pixels, and the camera has 1080p resolution [6]. It can capture frames at the rate of 30 FPS [6].

In this research Kinect ability for capturing body joints based on its depth images is used. The 2D body coordinates above the hip region are chosen because most sign language gestures include hand motions. For simplicity, hand postures and non–manual cues were not used for recognition.

Figure 5 shows the overall architecture of the sign language translator system.

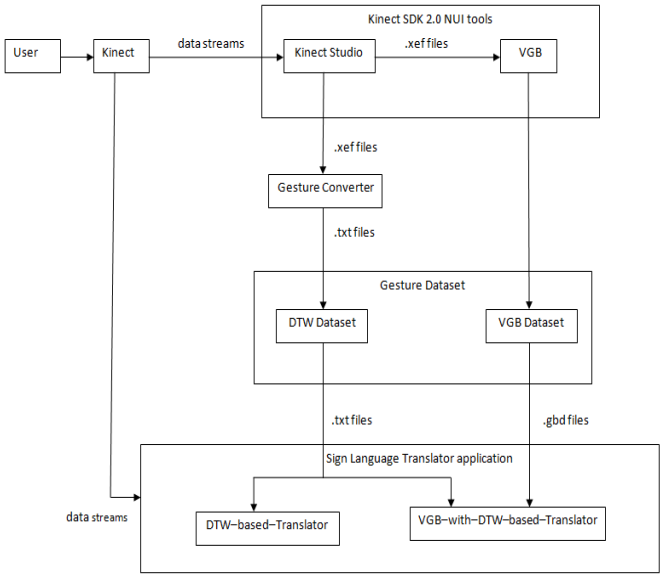

Figure 5. Overall System Architecture

An expert user is allowed to sign DGS gestures in front of the Microsoft Kinect for Windows v2. The Kinect captures the data streams from the user, and these data streams are recorded using Kinect Studio. The video files from the Kinect Studio (.xef files) are used for training datasets in VGB. These .xef files are processed using Gesture Converter, which returns DTW dataset (.txt files) after normalization (see Section 4.3). On the other hand, VGB datasets are created using the Visual Gesture Builder. After training VGB datasets, the generated .gbd files are used as a dataset in VGB-with-DTW-based Translator.

The .txt files from the DTW datasets are used in DTW-based Translator (see Section 4.5) as well as VGB-with-DTW-based Translator (see Section 4.7).

When a normal user signs a sign language gesture, the data streams are captured by sign language translator application (see Sections 4.5 and 4.7) and are translated into text (see Section 4.6).

IV. IMPLEMENTATION

4.1 Input Data

As mentioned before, input data for the system is data streams, which represents videos of gestures, filmed by Kinect Studio.

Kinect Studio is a recording tool, which can record 2D and 3D data captured by Kinect, along with the orientation of the body. Kinect studio records the data in a .xef file format. These files contain audio, video, depth information recorded by a user.

Recording clips with Kinect Studio was done by starting the Kinect service and user performing the gestures in front of Kinect. The Kinect records the videos in .xef format. These .xef files are used in VGB for training datasets.

4.2 Visual Gesture Builder

VGB is a tool, which was developed to recognize custom gestures using Kinect. VGB uses a data-driven solution for gesture recognition [8]. Gesture detection through VGB is done by training the gestures provided by users (content creation) rather than code writing [8]. The processes in VGB are described below in detail.

At first, VGB was used to tag the start and end positions of sign language gestures. A gesture in VGB is a posture in our context. A gesture in VGB was created with its custom features, which include “Body Side”, “Gesture Type” and “Training Settings”. A body side in VGB is differentiated into three categories, “Left”, “Right” and “Any”. “Any”-body side was chosen for training gestures. Discrete gesture type allows VGB to train gestures using AdaBoost, which is a machine learning meta-algorithm to improve performance [10]. “Ignore Lower Body” was chosen to train all the start and end positions of the gestures, and lower body was considered for training “Standstill” position.

After creating a gesture using VGB, start and end positions of sign language gestures were tagged separately (see Figure 6). The blue frames in Figure 6 represent a positive training set of data, whereas, all other frames, which are untagged, are referred to as negative training data. These frames are weak classifiers for AdaBoost. For tagging data, VGB provides custom input parameters. The custom input parameters chosen for the start and end positions of gestures are shown in Figure 6. Building gestures in VGB result in .gbd files. These .gbd files are used in application to detect gestures.

 ISBN: 978-1-61208-616-3

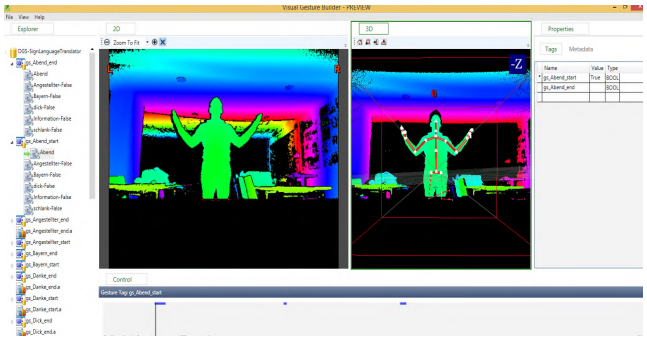

Figure 6. Clips tagged for "Abend_start" gesture

VGB view can be used to find out whether the start and end gestures of a sign language has been recognized correctly. It consists of all the gestures, which were tagged in VGB. When Microsoft Kinect SDK 2.0 captures the input, VGB view shows the confidence of gestures based on the input. A higher level of confidence shows more accuracy of the gestures performed by the user. Figure 7 is an example of a gesture performed using VGB view.

Project Settings

Name	Value	Type
Accuracy Level	0,95	FLOAT
Number Weak Classifiers at Runtime	1000	INT
Filter Results	True	BOOL
Auto Find Best Filtering Params	True	BOOL
Weight Of False Positives During Auto Find	0,5	FLOAT
Manual Filter Params: Num Frames To Filter	5	INT
Manual Filter Params: Threshold	0,001	FLOAT
Duplicate And Mirror Data During Training	False	BOOL
% CPU For Training	95	INT
Use Hands Data	False	BOOL
Ignore Left Arm	False	BOOL
Ignore Right Arm	False	BOOL
Ignore Lower Body	True	BOOL

Figure 7. Custom parameters for abend_start gesture

In Figure 8, the "dick" (in English: thick) gesture in DGS is performed. At first, a start position is detected followed by the end position. A spike shows the confidence of the start and end positions of the gesture.

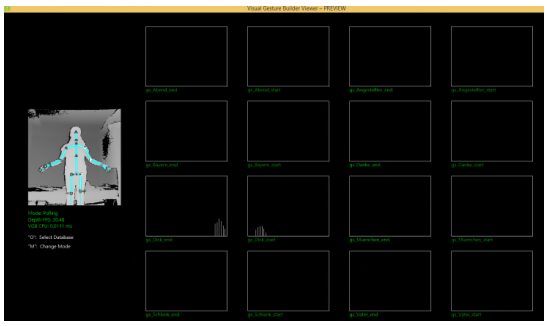

Figure 8. An example of viewing .gbd file using VGB view

The VGB view allows developers to verify if the training is good enough to be used in an application. After the .gbd file was viewed using VGB view, it was added in our application to detect the start and end positions of gestures.

4.3 Normalization of coordinates

Microsoft Kinect for Windows v2 SDK 2.0 can track 25 body joints using its depth images. To determine the hand gesture motion, the following body joints in 2D – HandRight, HandLeft, ElbowRight, ElbowLeft, WristRight, WristLeft, ShoulderRight and ShoulderLeft were chosen as the joints of interest because they contribute to identifying hand gestures [2] in a sign language.

In this case, normalization is carried out by shifting the origin from the Kinect to the user body position. It is done to eliminate the variations in the joined coordinates due to the height of a person or his position in the camera's field of view [2]. The distance between joints "ShoulderLeft" and "ShoulderRight" are taken for the variations due to a person's size [2]. Thus, the joint "SpineShoulder" is selected as the origin of the user body position. Normalization is done by subtracting the shoulder elements from joints coordinates when a user is not in the center of depth image [2]. While normalizing, every frame is separated by the delimiter "@" and consists of twelve X and Y coordinates of joints of interest except "ShoulderRight" and "ShoulderLeft". While performing hand gestures, "SpineShoulder" is considered as origin, which is approximately the midpoint of "ShoulderRight" and "ShoulderLeft". These normalized coordinates are used as sequence X in sequence matching using DTW (see Sections 4.5 and 4.7).

4.4 Gesture Dataset

4.4.1 DTW Dataset

DTW dataset consists of .txt files, which are resulted from the Gesture Converter after Normalization of coordinates (see Section 4.3). The .txt files consist of X and Y coordinates of body joints for a gesture.

4.4.2 VGB Dataset

The VGB dataset consists of clips from Kinect Studio recordings where start position and end position of each sign language gesture were tagged separately and considered as separate postures. While creating a gesture project, start or end keyword was added as a suffix of the gesture. For example: "gs_Abend_start". The "Gesture Type" was chosen as Discrete (AdaBoostTrigger). It allows VGB to train gestures using AdaBoost machine learning technique. In Figure 5, "Angestellter-false" in "Abend_start" gesture consists of "Angestellter" gesture, which was tagged as a negative dataset. Unlike tagging start and end positions, negative tagging includes tagging of motion of a gesture. For every gesture, negative training of the gestures should be done to reduce false positives. Figure 5 shows tagging of

 ISBN: 978-1-61208-616-3

"Abend_start" gesture for training. Multiple portions of the same gesture are tagged to improve accuracy.

4.5 DTW-based Translator

The body joints provided by Kinect can be used for sequence matching using the Dynamic Time Warping (DTW) algorithm.

DTW is a technique to find an optimal alignment between two given sequences under certain restrictions [16]. A distance measurement between time series can be used to find similarity between them [22]. It is used to cope up with time deformation and different speeds of time-dependent data [16]. The time and space complexity of DTW is O(nm) [13], which means every point in a series compares every other point in a time series.

For a given two sequences Q = (q_1, q_2, ..., q_i, ..., q_n) of length n and C = (c_1, c_2, ..., c_j, ..., c_m) of length m, a cost matrix is constructed where the (i^{th}, j^{th}) element of the matrix contains the distance $d(q_i, c_j)$ between two points q_i and c_j, i.e., $d(q_i, c_j) = (q_i - c_j)^2$ [13].

Each matrix element (i, j) corresponds to alignment between the points q_i and c_j [13]. A warping path W is a set of matrix elements, which defines the mapping between Q and C [13]. The k^{th} element of element W is defined as $w_k(i, j)$, where W = (w_1, w_2, ..., w_k, ..., w_K) and where $max(m, n) \leq K < m+n-1$ [13]. K is the length of warp path [13][22]. The minimum distance of a warp path is called an optimal warp path [22]. Its distance can be calculated using [22]: if $w_k=(i, j)$, w_k+1 is equal (i', j'), where $i \leq i' \leq i+1$, and $j \leq j' \leq j+1$, then

$$\sum_{K=1}^{K} Dist(W) = Dist(w_{ki}, w_{kj}). \quad (1)$$

In (1) Dist(W) is the Euclidean distance of the warp path W and Dist(w_{ki},w_{kj}) is the distance between two data points [22]. The value of a cell in a cost matrix is given by [22]:

$$D(i, j)=Dist(i, j)+min[D(i-1, j),D(i, j-1),D(i-1, j-1)] \quad (2)$$

The warp path to D(i, j) must pass through one of those three grid cells, and the least distance among the three neighboring cells are added to the Euclidean distance between the two points [22]. While filling the matrix, it is filled one column at a time from the bottom up, from left to right [22].

By using this approach, the cost matrix was plotted for two-time series. The normalized input from Kinect was captured in between standstill positions (sequence X in Figure 9) versus gesture from the DTW dataset (sequence Y) one at a time. The shortest distance (minimum cost) was calculated from the top row of the cost matrix. It was divided by the length of the gesture dataset sequence, with which it was compared against [4].

This resulted value was compared with a threshold value (t=2), which is defined manually. If the resulted value is less than or equal to t, the gesture is recognized.

4.6 Translation into Text

Since the gesture from Kinect (sequence X in Figure 9) is compared against multiple gestures in DTW dataset, only those gestures were chosen, which satisfies constraint (t less or equals to 2). DTW-based Translator returns a list of possible matches from the dataset for gesture input. Based upon the DTW principle, less cost results in more similarity in time series. Hence, the gesture, which has least cost was chosen to be a recognized gesture. After a gesture is recognized, the application gets the filename of the matched gesture, which is printed as the recognized gesture.

4.7 VGB with DTW-based Translator

DTW-based Translator utilizes high CPU and a significant amount of memory to compare the multiple gestures. Also, the application has to compute any gesture between standstill position even if they were not sign language gestures. The concept behind VGB with DTW approach is to avoid comparing every gesture in the DTW dataset for gesture recognition. It can be done by identifying the start position of the sign language gesture. If a start position is detected, it is highly probable that a user has signed a correct sign language gesture.

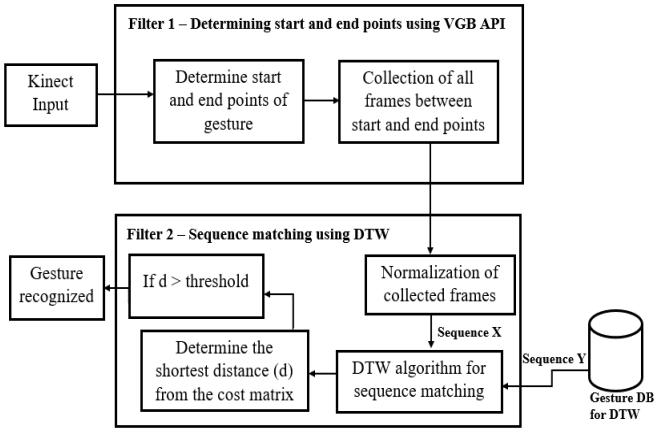

Figure 9. VGB-with-DTW-based Translator

In Figure 9, when the user body frame arrives as a Kinect input, each of these frames are processed by using the VGB API. It uses AdaBoost machine technique to recognize the start or end position of the sign language gestures. AdaBoost is a supervised machine learning algorithm [15], which generates a strong classifier out of the set of weak classifiers [10][12]. While training gesture datasets in VGB using AdaBoost, weak classifiers, i.e., angles using inferred joints, speed rejecting inferred joints, etc., are used to generate the strong classifier along with a confidence level. When a frame from Kinect is recognized as a gesture, VGB API returns the gesture name. When the start position of a gesture is detected, the application collects the body frames unless the end position of the same gesture is found. In the meantime, if it detects other gestures, the stored frames are cleared, and the collection of frames begins as soon as start

 ISBN: 978-1-61208-616-3

position is detected again. After the frames are collected between start and end positions of the same gesture, they are normalized and passed for sequence matching using DTW. In this case, the application knows the gesture name and, by using this gesture name as a reference, it compares with only one gesture from DTW dataset, which matches the gesture. It was done to find if the motion of the sign language gestures were performed correctly. Using this technique, false positives, which were observed using the DTW-based Translator, can be significantly reduced.

V. RESULTS

The eleven gestures of DGS used in Germany: Abend, Angestellter, Bayern, Danke, Dick, Gebärdensprache, Information, München, Schlank, Un- möglich and Vater were used to compute the cost of computation and accuracy. These gestures, which were used as datasets for DTW and VGB were provided by an expert user. To compute the cost of computation and accuracy, "Abend"-gesture was chosen against the varying number of gestures in the dataset. For DTW–based Translator DTW threshold was chosen as 2, whereas for VGB-with-DTW based translator DTW threshold was chosen as 2.5.

In VGB-with-DTW based translator time taken to compute a single gesture in dataset includes the sum of time taken to compute start and end position of gesture from VGB along with time taken to compute the gesture in DTW dataset. Figure 10 shows time taken to compute the "Abend"-gesture using both approaches.

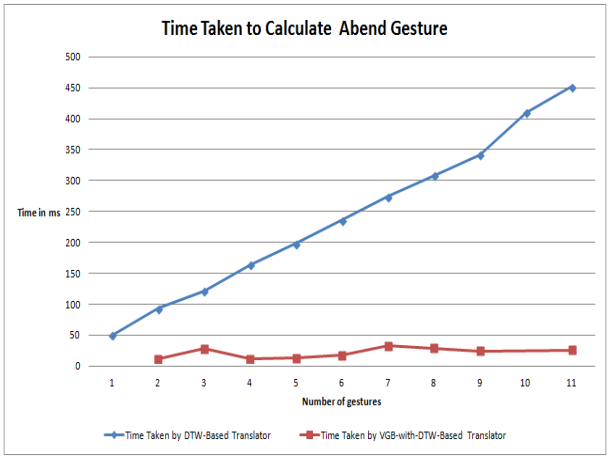

Figure 10. Time calculation for "Abend"-gesture using both approaches

5.1 Accuracy of DTW-based Translator

To compute the accuracy for DTW-based Translator, every gesture was performed 10 times by a novice user. Before performing the gestures, they were studied from the recordings of an expert user. Table 1 shows the number of detected, not detected, and false positives for the gestures using DTW-based Translator. The accuracy column shows accuracy in percentage for detected gestures. "Schlank" had 90% accuracy whereas "Unmöglich" had 40% accuracy.

TABLE I. ACCURACY OF DTW-BASED TRANSLATOR

Detected Gestures	No. of Obs.	Detected	Not Detected	False Positives	Accuracy(%)
Abend	10	6	3	1	60
Angesteller	10	7	3	0	70
Bayern	10	5	5	0	50
Danke	10	8	0	2	80
Dick	10	6	3	1	60
Gebärdensprache	10	5	5	0	50
Information	10	6	1	3	60
München	10	8	2	0	80
Schlank	10	9	1	0	90
Unmöglich	10	4	2	4	40
Vater	10	8	1	1	80
Total	110	72	26	12	
Overall Accuracy(%)		65.45	23.64	10.91	

Table 2 shows the gestures, which were recognized as false positives for DTW- based Translator.

TABLE II. GESTURES WITH FALSE-POSITIVE RECOGNITION

Detected Gesture	Recognized as
Abend	Gebärdensprache
Danke	Bayern
Dick	Unmöglich
Information	Dick
Unmöglich	Bayern, Danke
Vater	Bayern

Figure 11 is based on Table 1. For "Abend"-gesture, 6 gestures were detected, where one was false positive. The highest number of false positives were observed in "Unmöglich"-gesture, whereas the lowest number of false positives were observed in "Angesteller", "Bayern", "Gebärdensprache", "München" and "Schlank". The least detected gesture was "Unmöglich" whereas the most detected gesture was "Schlank".

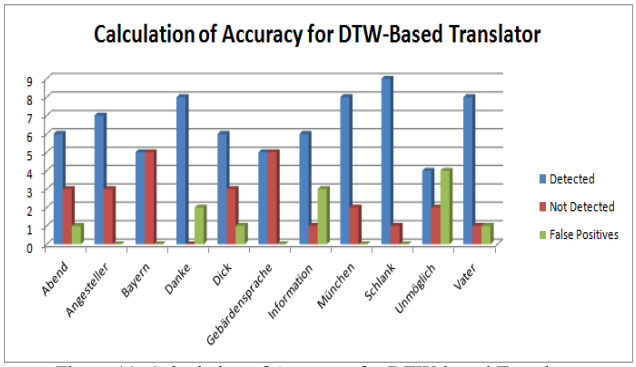

Figure 11. Calculation of Accuracy for DTW-based Translator

5.2 Accuracy of VGB-with-DTW-based Translator

Table 3 was calculated by performing a gesture several times by a novice user. "Abend" gesture was performed 15 times, but it was recognized only 4 times with no false positives. Some gestures like "Information" and "Vater" were not detected at all. The best-detected gesture was "Bayern", "Dick", and "München".

 ISBN: 978-1-61208-616-3

TABLE III. ACCURACY OF VGB-WITH-DTW-BASED TRANSLATOR

Detected gesture	No of obs.	Detected	Not detected	False Positives	Accuracy(%)
Abend	15	4	11	0	26.67
Angestellter	24	1	23	0	4.17
Bayern	15	7	8	0	46.67
Danke	14	4	10	0	28.57
Dick	14	6	8	0	42.86
Gebärdensprache	17	2	15	0	11.76
Information	23	0	20	3	0
München	12	5	7	0	41.67
Schlank	16	4	12	0	25
Unmöglich	23	6	17	0	26.09
Vater	18	0	18	0	0
Total	191	39	149	3	
Overall accuracy(%)		20.42	78.01	1.57	

Figure 12 shows the number of detected, not detected, and false positives for the gestures for VGB-with-DTW-based Translator recognized gesture.

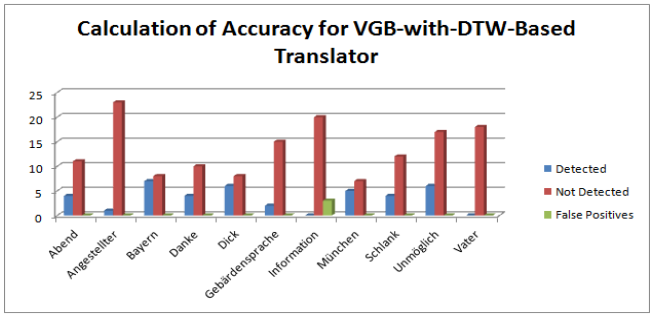

Figure 12. Calculation of accuracy for VGB-with-DTW-based Translator

5.3 Overall AccuraciesVGB with DTW-based Translator

DTW-based translator detected 65.45% of gestures with 10.91% false positives whereas VGB-with-DTW-based Translator detected 20.42% of gestures with 1.57% of false positives. More detections but also more false positives were observed in DTW-based Translator.

VI. DISCUSSION AND FUTURE WORK

In Figure 10, the time, which was taken to compute "Abend"-gesture using DTW–based translator, increased with increasing number of gestures in the dataset whereas VGB-with-DTW based translator almost remained same. This behavior was expected because DTW-based translator has to compare with more numbers of gestures on increasing the gestures in the dataset whereas VGB-with-DTW based translator has to compare with only one designated gesture. The constant outcome was expected for the VGB-with-DTW based translator, but some irregularities were observed in Figure 10. One of the possible reasons might be VGB not being able to detect the same frame as a start or end position of gestures in each observation. The accuracy of detected gestures of DTW–based Translator (see Table 1) was better than VGB-with-DTW based translator (see Table 3), although, some false positives were observed in DTW–based translator. From the false positive recognition (see Table 2) it can be inferred that:

- Usually, single-handed gestures have single-handed gestures as false positives. Similarly, two-handed gestures have two-handed gestures as false positives (Exception: "Dick").
- Most of the false positives were caused by "Bayern"-gesture. It was false positive for "Danke", "Unmöglich" and "Vater". If we observe the right-hand movement of "Bayern"-gesture, coordinates of right hand differ slightly because it is signed by forming small circles around the chest. The small variation in its coordinates might have resulted in false positive gesture for many cases.

From the accuracy of DTW-based Translator, it can be inferred that:

- Accuracy highly depended upon how user performed the gesture.
- Gestures involving long motions were difficult to recognize. For example, "Gebärdensprache", "Unmöglich".
- Short Gestures that included variation in coordinates were easier to recognize. For example: "Danke".
- However, lengthy gestures involving a greater variation of coordinates were difficult to recognize ("Abend", "Unmöglich").
- Long gestures affected Natural User Interaction of the user. It was observed that, while performing a long gesture like "Gebärdensprache", the gesture displayed in the application did not synchronize with the user body movement.
- When gestures were performed accurately, i.e., in accordance with the expert user, it was easier to recognize.
- The accuracy of VGB-with-DTW based translator was lower than DTW–based Translator. Some factors that may have affected its accuracy are:
- Detected gestures from VGB, i.e., start and end positions have very low confidence (<50%).
- The translator was not able to detect start and end points precisely. One reason might be due to a number of frames tagged in VGB. Several frames were considered for tagging standstill position, but only a small portion of frames was used for tagging start and end positions of gestures.
- "Angestellter_start" was not detected precisely, compared to "Angestellter_end". For example: the start position of "München", as well as the end position of "Unmöglich", were difficult to be detected. The reason might be that the start and end positions of a gesture closely resemble with other gestures.
- A single gesture "Bayern" was detected twice, because of multiple start and end points in the gesture. In such cases, the accuracy of this translator was even lower, because the partially completed gesture is not considered as a single gesture.

Nevertheless, this application is a proof of concept that VGB and DTW can be used for sign language recognition.

VII. CONCLUSION

In this paper, two different methods – DTW and VGB along with DTW were proposed to recognize dynamic sign language gestures.

To reduce complexity in application development, several constraints were taken into account, i.e., ignoring non-manual cues, the configuration of hands and considering

 ISBN: 978-1-61208-616-3

only the signs that involve some motion. Using DTW–based translator-approach the input gesture was compared against all available gestures in dataset whereas in VGB–with–DTW–based translator, the concept of determining the start position and end position of a sign language were used to guess the correct gesture, which was then verified by DTW computation. The DTW-based translator had some false positive detections and the time computation increased with the increasing number of the gestures in the dataset. Usually, similar gestures were the false positives. Some latency was observed using this approach, but the accuracy was comparatively better than translator based on VGB–with–DTW–based translator. The key factor for the increase in accuracy was a small dataset, and the boundary condition (standstill) for the gestures were easily recognized. However, DTW–based translator is unrealistic for large numbers of the dataset. In VGB-with-DTW-based translator, false positive detections were significantly lower compared to that of DTW-based translator. But, the accuracy decreased significantly. The lower accuracy was mostly because of not being able to precisely detect start and end points of similar sign language gestures. This approach had less computation time because of less noise in the gesture. However, the accuracy also reduced significantly because of the low detection rate of start and end gestures.

From benchmarks of the translator applications, it can be concluded that the DTW-based Translator has higher accuracy (65.45%) than that of VGB-with-DTW-based Translator (20.42%). But, the VGB-with-DTW-based Translator performed better regarding CPU consumption than the DTW-based Translator.

REFERENCES

[1] "Sign language". Available from http://wfdeaf.org/human-rights/crpd/sign-language; Last accessed 20.01.2018.

[2] S. Celebi, A. S. Aydin, T. T. Temiz, and T. Arici, "Gesture recognition using skeleton data with weighted dynamic time warping". In VISAPP (1), pp. 620-625, 2013.

[3] H. Cooper, E. J. Ong, N. Pugeault, and R. Bowden, "Sign language recognition using subunits". The Journal of Machine Learning Research 13, 1, pp. 2205-2231, 2012.

[4] "Dynamic time warping to recognize gestures". Available from https://social.msdn.microsoft.com/Forums/en-US/4a428391-82df-445a-a867-557f284bd4b1/dynamic-time-warping-to-recognize-gestures?forum=kinectsdk; Last accessed 20.01.2018.

[5] N. Gkigkelos and C. Goumopoulos, "Greek sign language vocabulary recognition using Kinect," presented at the Proceedings of the 21st Pan-Hellenic Conference on Informatics, Larissa, Greece, 2017.

[6] "Kinect tools and resources". Available from https://developer.microsoft.com/en-us/windows/kinect; Last accessed 20.01.2018.

[7] "Visual gesture builder: A data driven soultion to gesture detection". Available from https://onedrive.live.com/view.aspx?resid=1A0C78068E0550B5!77743&app=WordPdf; Last accessed 20.01.2018.

[8] K. Emmorey, "Language, cognition, and the brain: Insights from sign language research". Psychology Press, 2001.

[9] G. Fang, W. Gao, and D. Zhao, "Large-vocabulary continuous sign language recognition based on transition-movement models". In: IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, 37, 1, pp. 1-9, 2007.

[10] Y. Freund, and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting". In: Journal of computer and system sciences, 55, 1, pp. 119-139, 1997.

[11] P. A. Harling and A. D. N. Edwards, "Hand tension as a gesture segmentation cue". In: Proc. of Gesture Workshop on Progress in Gestural Interaction, pp. 75-88, 1996.

[12] T. Hastie, R. Tibshirani, and J. Friedman, "Boosting and Additive Trees". In: The Elements of Statistical Learning. Springer Series in Statistics. Springer, New York, NY, 2009.

[13] E. Keogh and C. A. Ratanamahatana, "Exact indexing of dynamic time warping". In: Journal of Knowledge and Information Systems, 7, 3, pp. 358-386, 2005.

[14] S. K . Liddell, "Grammar, gesture, and meaning in American Sign Language". Cambridge University Press, 2008.

[15] O. Mozos, C. Stachniss, and W. Burgard, "Supervised learning of places from range data using adaboost". In: Proc. of the IEEE International Conference on Robotics and Automation, pp. 1730-1735, 2005.

[16] M. Müller, "Dynamic time warping". Information retrieval for music and motion, Springer, 2007.

[17] S. Ong and S. Ranganath, "Automatic sign language analysis: a survey and the future beyond lexical meaning". In: IEEE Transactions on Pattern Analysis and Machine Intelligence, 27, 6, pp. 873-891, 2005.

[18] M. Oszust and M. Wysocki,. "Some approaches to recognition of sign language dynamic expressions with kinect". In Human-Computer Systems Interaction: Backgrounds and Applications 3. Springer, pp. 75-86, 2014.

[19] R. Pfau, M. Steinbach, and B. Woll, "Phonetics, phonology and prosody". In: Sign Language: An International Handbook, pp. 4-77, Walter de Gruyter, 2012.

[20] R. Pfau, M. Steinbach, and B. Woll, "Sign language: An international handbook". vol. 37, Walter de Gruyter, 2012.

[21] Quizlet LLC; Available from: http://quizlet.com/12296633/abc-1-asl-vocab-flash-cards; Last accessed 20.01.2018

[22] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space". In: Journal of Intelligent Data Analysis, 11, 5, pp. 561-580, 2007.

[23] M. S. Santos, E. B. Pizzolato, S. Feuerstack, "A Real-Time System to Recognize Static Gestures of Brazilian Sign Language (Libras) alphabet using Kinect". In: IHC 2012 Proceedings, 2012.

[24] A. Sarkadi, "Vocabulary learning in dyslexia: the case of a hungarian learner". In: Language learners with special needs: An international perspective. Clevedon: Multilingual Matters 194, 2008.

[25] C. Sun, T. Zhang, C. Xu, „Latent support vector machine modeling for sign language recognition with Kinect." In: ACM Trans. Intell. Syst. Technol. 6, 2, pp. 1-20, 2015

[26] T. Starner and A. Pentland "Real-time american sign language recognition from video using hidden markov models". In: Proc. of the International Symposium on Computer-Vision, pp. 265-270, 1995.

[27] W. C. Stokoe, "Sign language structure: An outline of the visual communication systems of the american deaf". In: Journal of deaf studies and deaf education, 10, 1, pp. 3-37, 2005.

[28] W. C. Stokoe, D. C. Casterline and C. G. Croneberg, "Introduction to a dictionary of american sign language". In: Linguistics of American Sign Language, pp. 243-258, 2000.

[29] W. Vicars, "ASL University". Available from http://www.lifeprint.com; Last accessed 20.01.2018.

[30] J. Zhang, W. Zhou, and H. Li, "A Threshold-based HMM-DTW Approach for Continuous Sign Language Recognition". In Proc. of the International Conference on Internet Multimedia Computing and Service, pp. 237-240, 2014.

 ISBN: 978-1-61208-616-3