3D Avatar for Automatic Synthesis of Signs for The Sign Languages

Diego Addan Gonçalves
Universidade Federal do Paraná - Brazil
diegoaddan@gmail.com

Eduardo Todt
Universidade Federal do Paraná - Brazil
todt@inf.ufpr.br

Laura Sanchez Garcia
Universidade Federal do Paraná - Brazil
laura@inf.ufpr.br

ABSTRACT

This paper discusses a synthesis system that generates, from a XML input representing gesture descriptors, a vector of configuration parameters that are executed by a 3D Avatar for use in the animation of Sign Languages. The development of virtual agents able to reproduce gestures of sign languages is very important to the deaf community, since in general they also have difficulties to read conventional texts. In this research project, a consistent combination of 3D editor Blender, CMarkup parser and graphics engine Irrlicht was used to develop a novel approach to sign synthesis, based on a recent XML model that describes hand gestures using shape, location, movement and orientation descriptors. The described experiments validate the proposed implementation model, which constitutes a promising alternative in the area of synthesis of signals for computational applications of Sign Languages.

Keywords

Gesture synthesis, HCI, Accessibility and usability, deaf community, graphics engine.

1 INTRODUCTION

Research and development of systems which use virtual agents for Sign Language interpretation and generation is still scarce, lacking investment and motivation to boost development in this direction. It is not broadly known that, in general, deaf subjects also have difficulties in the process of conventional text reading, due to the lack of meaning to the phonetic representation of syllables. An example of this is the social context where resources that facilitate the use of Sign Languages are fundamental but not currently available. For instance, environments such as hospitals, could have means to provide important instructions, such as emergency warnings, through a virtual agent, offering to deaf people the same speed in communicating information as provided by conventional text and voice alternatives.

The construction of a synthesis system between traditional written languages and Sign Languages depends on the definition of the inputs, such as video recognition of gestures or written symbols, and outputs, like a representation through a 3D avatar. In order to produce outputs in a generic and parametric way, it is necessary to have a base, or descriptive model, of the phonological aspects of Sign Language and also a system capable of interpreting these data.

If the data input is a video of captured gestures, the system can recognize the signals and then supply the data to the model that generates descriptive interpretation in the form of written characters representation or another descriptive representation. If the input data is in written character, or given by descriptive representation, the system can make the synthesis to the interpretation through a 3D avatar.

The existing tools to the visual representation of Sign Languages (Section II) have limited resources and do not generate movements automatically. In general the user has to drag a list of movements to the virtual agent to generate the desired animations, in a tediously and time consuming process.

These tools are based on complete and indivisible signs, being restricted to a very limited knowledge base.

Bearing in account the formal models presented in the literature, and the limitations of existing systems, this work aimed to contribute to the academic scenario in order to develop a service for automatic synthesis of signals of a Sign Language through a 3D avatar.

 ISBN 978-80-86943-66-4

2 DESCRIPTION MODEL OF THE SIGN LANGUAGE LIBRAS AND EXISTING SYSTEMS

In this section, the necessary concepts related to computational models appropriate to describe the signs of Sign Languages and tools of textual interpretation are discussed, as well as other works related to computational systems for representation of Sign Languages using 3D Avatars.

2.1 Formal Representation of Sign Languages

The first visual architecture of schematic representation of signals, based on ASL, American Sign Language, was introduced in 1983, and it presented concepts of animation from a skeleton with movements based on formal modeling [1].

Since then, other studies have been developed adding new parameters, such as the type of motion, speed, repetition and symmetry of the arms [2] [3], and also using standard XML representation [4].

Recently a formal model that incorporates the parameters used in signals, structured through hierarchically organized classes, formally representing the Libras in an appropriated form oriented to computational use was developed [5].

This last model was chosen as the reference in the current work, although it was necessary an adaptation, described in the following sections, for use in 3D avatar. Figure 1 shows an example of some parameters and values of the formal model and its corresponding representation in an avatar.

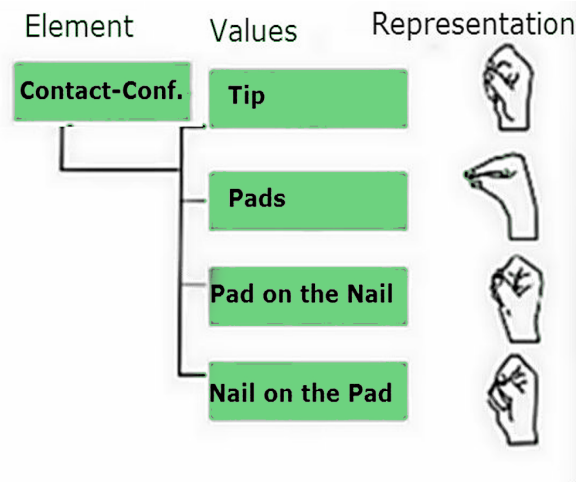

Figure 1: Here some contact configurations are show, where different finger positions indicate different types of contact of fingers.

2.2 Existing Tools for Representing Sign Language through 3D Avatars

From these formal models, and other forms of representation of Sign Languages, some software simulation of Sign Languages have been developed for commercial or academic purposes.

Gibet [6] was one of the first researchers to integrate the use of 3D for animation to a Sign Language. His work was based on a 3D arm with two joints. Two years later a early fingers configuration model was proposed, outlining some words, letter by letter, conceptualizing the idea of an animated hand that could be accessed over the internet using VRML [7].

Chadwick, Haumann and Parent [8] presented a model for creating an animatable body, human or not, based on three layers: skeleton, muscles and skin. The main commercial computer graphics tools work with a system like this, since only the layer of muscle is not indispensable to obtain a full animatable model based on deformation of polygons.

Research for an outline application for the synthesis of a Sign Language is a constant concern, expanding its outputs to the web. In [9] a model is presented based on two main classes, the client and server. The first consists of a common web browser, with support for technologies such as WebGL (HTML5 API that allows for the rendering of 3D graphics in web browsers) or O3D (open source API for creating 3D applications in web browsers) and Java Script (scripting language for web browsers). The server class receives requests from the client side and does the communication and translation animation, followed by conversion needed to pass the results to the browser on the client side.

Among these systems, it is relevant to quote those from VCom Gesture Builder [10], Max's Einfach Teilhaben [11] and Sign 4 Me [12], which propose the representation of Sign Languages through a 3D Avatar.

Yi, Harris and Descalu [13] also suggest that a major problem in these systems is the fact that users must know the native language of the software. The authors also reinforce the idea that the evaluated systems don't have a simple and intuitive interface, hindering the interaction of beginners. An important point in which the authors focus is that these tools may not have extended their content being limited to a set of pre-defined phrases or words. In other words, these systems rely on user controlled content, not being automatically animation software.

We conclude that the main limitations of these existing systems arise from the characteristic that they didn't use parameters based on a formal model, and so the animations are indivisible full signs, not generated automatically. Also they spell letter by letter a syllabic representation of the signal meaning, assuming that the deaf

user has the necessary knowledge of a traditional written language in order to understand the gesture output. Moreover, most of these software are commercial, being neither freely distributed nor open source.

Thus, one attractive alternative to overcome these limitations is to develop a system that makes the synthesis of the input parameters based on a formal model of computing Sign Language, producing an output through a 3D avatar, as shown in Fig.2, where the synthesis process is the most important stage.

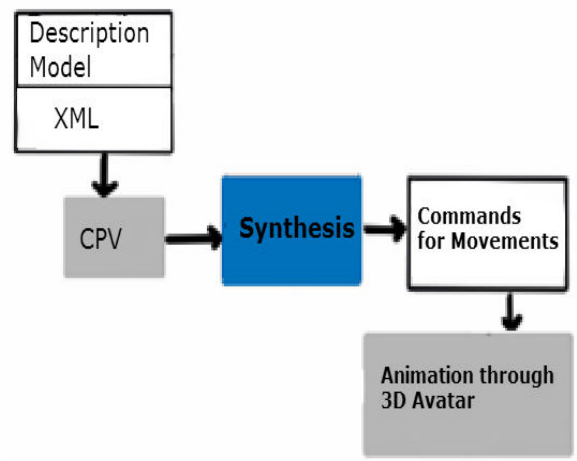

Figure 2: Block diagram of the proposed system, where CPV means Configuration Parameters Vector (Described in Section IV), that represents sequences of the required poses for each joint of the avatar's skeleton.

3 DEVELOPMENT OF 3D AVATAR AND ANIMATION SYSTEM

To implement the synthesis process it was first necessary to create a 3D humanoid object concerning the Avatar 3D. Its construction involved the following steps: design of the avatar mesh, research of avatar modeling techniques and choice of an appropriate option, definition of the necessary articulation points, and rigging construction. The last one is the structure used in the animation process and its connection to the mesh defines the reactions in the mesh to movements of the controller structure connected to it.

The modeling technique used was the poly-by-poly, following the concept art that the mesh was constructed from polygon extrudes [14]. At the end of this process an object was obtained with 2,276 polygons, which may be considered a low-count poly object. Fig.3 shows the concept art and the corresponding 3D Avatar mesh that was built using this technique.

In the following, the animation structure and rigging process for the 3D avatar was built. This structure includes controllers for the hands, arms, shoulders, torso

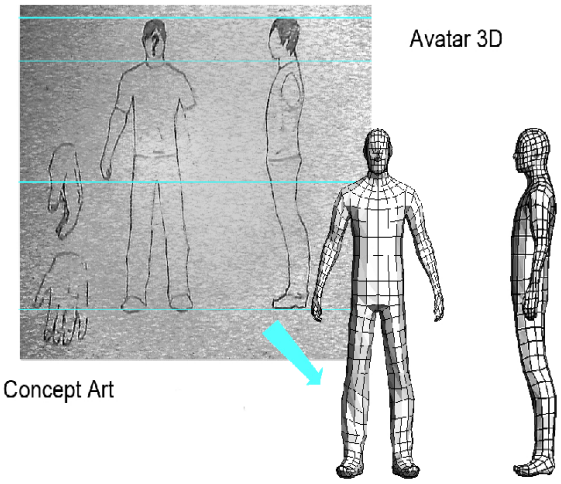

Figure 3: Concept Art and 3D Avatar.

and head. Facial expressions were not yet integrated at this point, but will be explored in future work.

The structure of animation controllers was integrated with the mesh through a technique where a weight is applied at each vertex of the structure relative to a particular controller.

Then UVW (map coordinate technique) mapping of the mesh was performed, where 3D structure is unwrapped into a 2D plane, enabling the application of the texture material needed to give to the avatar is superficial finishing [14].

With that, a texture was applied in the surface of the object for visual effects, finalizing the process of creating the 3D avatar. The most important point in the 3D avatar construction, beyond the concern to keep the mesh with few polygons, was to adjust both the physical model (mesh), and the animation framework, with the parametric model applied to the system.

The structure has a particular emphasis on hands, which were the most used in the tests and are the attribute of greatest importance in the formal model. In the future it is important to consider the Facial Expressions, which were not detailed in the formal model [5]. For the time being the implemented parameters were the hands and arms. Thus, at the rigging these points were considered to beof greater importance in terms of realism in the distribution of the weights of the vertexes and the spatial changes (collision with other parts).

4 AUTOMATIC SYNTHESIS OF SIGNS

This section describes the behaviour of the proposed system for the automatic synthesis system, generating the transcription to an output of hand and arms elementary movements through a virtual agent.

4.1 CPV - Configuration Parameters Vector

The descriptive model of Libras for computational use is divided into elements and sub-elements. The hand, for example, is considered as an element in formal model and the configuration of the hand is a sub-element with their respective values, like spatial coordinates or hand configuration. These parameters are represented by a vector of configuration parameters, called Configuration Parameters Vector (CPV).

This CPV is based on the formal model elements and are used XML external files to organize and register them, descriptively, separating the parameters by hierarchical packages that have elements and sub-elements and their values.

Set the schema formal model [5], a data structure based on tree was built to organize the elements used in the following test. This tree retains the hierarchical condition of the elements, and its construction prioritized elements concerning the arms and hand expressions.

The hand settings values, considering the Brazilian Sign Language Libras, have 61 possible values, and therefore it was decided that the rotation control and direction of the palm are the variable values of the application.

This representation differs from the formal model that suggests to control each finger for each motion value. Including the fingers would bring an unnecessary computational cost, so was defined use a "hand configuration" as elements. Was used for the tests the Libras hand settings and values for all other parameters defined in the model, such as motion, contact, direction of rotation, arm position and palm position.

Fig.4 shows an example of the created tree, on an AVl tree model. This tree graphic refers to the nodes of the sub-tree of non-manual expressions, where each node represents a value in the struct, hierarchy-dependent, based on formal parametric model. These sub-trees was expanded adding their specific parameters of articulations, movements and directions. Was built one sub-tree for each parameter and their elements.

This data structure was used for the construction of the CPV inputs. The interesting thing in CPV input format used in this work is that, since the entries are organized hierarchically like a tree, it is possible in the future use these algorithms for path decision making and optimization by the nodes.

Since an input is obtained, the system read the input file and recognize its structure, elements and values. The elements are the nodes that make up the chain and their values are the positions that each element of the CPV should take.

Fig.5 illustrates how this initial process works.

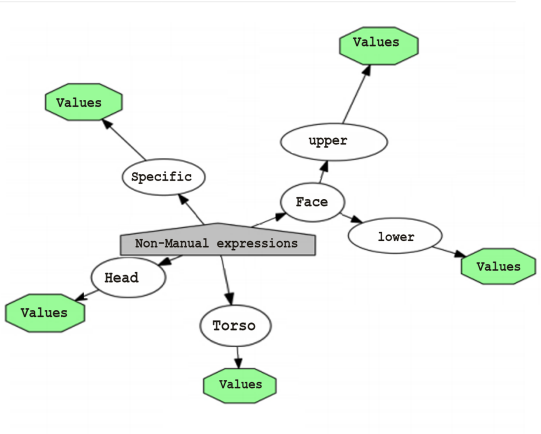

Figure 4: Sub-tree of CPV, hierarchically organized.

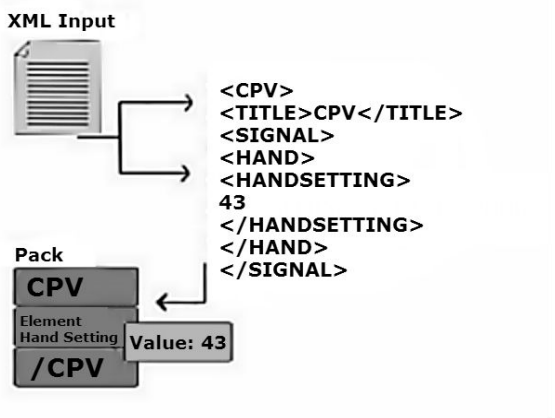

Figure 5: CPV interpretation.

We then conducted integration tests, that in principle were independent of the use of graphics engine (tool required for the next stage of the work) between the main application in C++ and the input file for the CPV. The test procedure was to create a function in the main application that can read the XML input file, extract and recognize their data, and pass the information to the application variables that can be used in the process or run time.

The data in test file were divided into two packs separated by tags, the first was called "element1" with the value "1", and a second pack called "element2" with the value "2". Basically the system should read this file by the application, separate the packs properly in hierarchical order and identify their values.

These tests were conducted using the parser library CMarkup that allows navigating XML files. With the system being able to read and identify elements of the input, the next procedure was the implementation in the graphics engine, where synthesis of the obtained data and generation of the output through the 3D avatar happens.

It is important to emphasize that the application of the formal model parameters used in virtual reality brings a new and promising scenario for the sign representation of Sign Language systems, correcting the main conceptual problem of this system is that the construction of phrases from a principle without unity, dealing with sentences like elements and not independent letters.

4.1.1 Automatic synthesis through the 3D avatar

The process of synthesis from data found in XML input to an output to Avatar 3D, is based on the 3D editor Blender, the parser CMarkup and graphics engine Irrlicht. The system software was implemented in C ++.

In order to build the animation and the corresponding call parameters, two paths can be taken. The first method is to make the code structure generate the animation movements of the graphics engine making each call associated to a pivot point bones. In this sense, each bone must be moved following the hierarchical structure of formal model, descending from the main element to the last sub-element chain. The second method are moving the mesh of 3D Avatar using the weight of each vertex, through an skeleton structure based in curves.

The system code indicates which bone will be moved and its coordinate position (in X, Y, Z environment position). After moving the first element of the chain of CPV it performs the same procedure on the following chain element (child node), and repeats the procedure until the last element in the chain is achieved.

In this process, the system only needs the textured Avatar 3D with the animation structure rigged, all calculations and procedures are executed from the main system through the graphic engine, which control animation structures.

Fig.6 shows the operation of this method.

A full test of the system was performed, integrating the parser to the environment with the graphics engine and using the methods to control the animation structure, in order to identify the input file and the information contained therein.

For the experiments, an entry with two elements was used, one representing an arm movement and the other specifying a hand configuration, one of the 61 standard possibilities from hand in Libras [15].

The mapping of these two configuration was done in the 3D editor, defining chains of sequential frames, using a morph modifier, for each block, by calculating the spatial alteration of each point of controller structure.

After recognizing the elements blocks for the CPV, each value was passed to a variable in the main application, so it could be used on the application process. Then a comparison with the values extracted was performed and, then, the application recognized the input block,

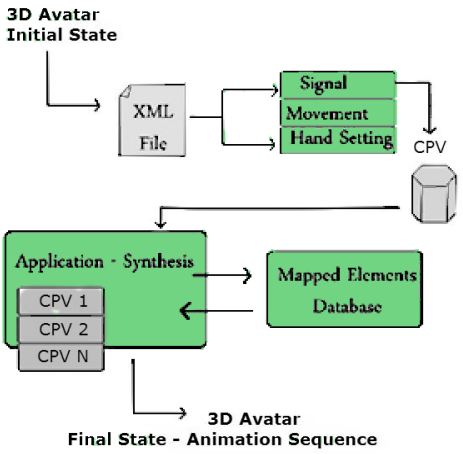

Figure 6: Synthesis Process. Begins by parse, extract the data configuration parameters, and applies for coordinates in the virtual environment rendering through 3D Avatar with the mapped movements.

searched the database and executed the mapped elements, exported to the mesh. These comparisons follows the hierarchical sequence of formal model.

In the following there is an slice of pseudo-code, concerning the recognition process of the input elements and passing parameters to the output generation process in the graphics engine Algorithm 1.

Data: XML Input
Result: Output in Graphic Engine - synthesis through 3D avatar.
while *(XML.Element = nextParameter AND VariableRoot.element != Null)* **do**
    read current element;
    if *GetChildData == avatarParameter* **then**
        gets coordinates;
        avatar = nodeSetFrameLoop(Value);
        counter = nextPointer;
        readNextParameter(counter)
    else
        counter = nextPointer;
        readNextParameter(counter);
    end
end

Algorithm 1: Read XML Parameters

Therefore, it could be verified that the output generated by 3D Avatar followed exactly the order described in the XML input. Fig.7 shows the output of 3D Avatar animation using the method described.

5 VALIDATION

For validation purposes a new test was conducted, running the entire process. The sign for the word "motorcycle" representation in Libras was chosen to be applied

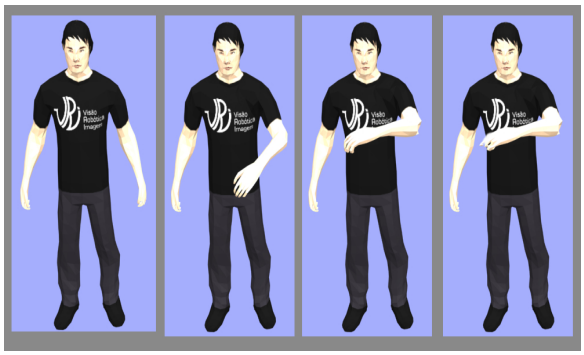

Figure 7: Animation Output.

to the complete synthesis process. As the test conducted in the previous section, the initial step was to build the XML input regarding the formal model blocks.

The validation protocol aimed testing the extraction of inputs through external XML file, evaluating the mapping of individual elements as well as the deformation in the 3D mesh.

Then we carried out the process of mapping the elements defining what will be the representation of a particular bone, or bones, related to a specific input. The bones constitute the articulated skeleton structure of the avatar.

For the "motorcycle" signal, the following elements were defined for the CPV: symmetry between the bones of the forearm and hand, bone turnover regarding the forearm at 90 degrees, hand setting, rotating the wrist of the right hand 90 degrees, repeating 3 times the last element.

Then we performed the synthesis process, where the description has been read from the XML file in order to find the CPV elements described earlier, and the system could find all blocks of configuration elements in hierarchical sequence. Since the call order and the elements were mapped, the system generated the corresponding output through 3D avatar as shown in Fig.8.

A test was built with two additional configuration parameters related to hand position without symmetry, besides rotation and movement of the arm. The tests followed, in general, to analyze the process unit from reading the CPV to exit through the graphic engine. In all cases the two methods used were shown in previous section, in order to assess behavioral differences between the different forms of the mesh control.

All tests presented in this paper were performed on a machine with the following settings: AMD Turion II X2, video card ATI Radeon HD 5470, 4 GB DDR3 memory.

As for performance and run time, all tests were processed in less than 1 second, since that technique, although involving 3D graphics, that usually constitute a heavy process, uses no complex lighting system, nor physical effects as secondary objects, scenarios or meshes which are not part of the 3D Avatar, which could make the rendering process heavy.

Concerning the results, the second method of synthesis process presented in the previous section, which uses a database of mapped elements in Blender itself, proved to be more advantageous than the first one, as discussed bellow.

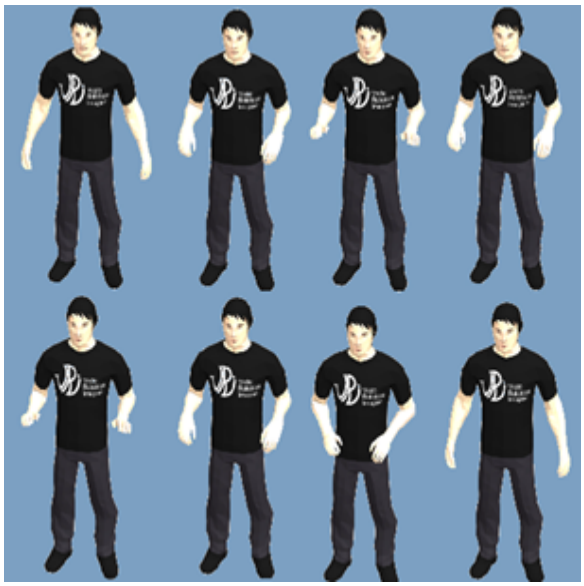

Figure 8: Motorcycle Animation Output.

The first method of testing the avatar 3D, in some cases initiated in a false position, which reflected across the output animation. Furthermore, since in this case the mapping process and calculations of positions happens at run time, the rate of frames per second in some tests fell by cutting approximately one frame to a sequence of forty.

In the second case, most of the processing is done during the mapping of the elements in 3D editor, and the processing of each block in the synthesis process, occur by calling blocks of pre-calculated position frames, which secure a less processing load.

This difference in the two methods proves valuable in the future, where the system has to calculate not only a sign, or part of it, but a entire conversation.

Still, the use of graphics engine lets you view, in the near future, the integration of these methods to the development of concepts for mobile devices or web browsers. Thus the spread and use of this kind of system achieves a significant portion of users, deaf or not, making this type of application, natural part of current systems.

6 CONCLUSIONS

The main contribution of this work is the development of a synthesis system through a 3D avatar for use in automatic Sign Languages animation, based on a formal

computational parametric model. With further work, this system can be used by the deaf community in fact, already working with elements of ACP, and not ready with signs or spelling, solving the problems of existing systems.

This formal model provides that the terms and words are represented within a real context and use of the deaf community, unlike the known systems. The traditional way, by spelling, generates an animation without unity, and difficult to understand. This step is important for the development of computational systems of representation in virtual reality facing the deaf community start of a right principle, using a formally input accepts.

The general contributions of this work are the application of conceptual formal model [5] in a real virtual reality environment, and the developed algorithm to extract the CPV information and translate into coordinate information for the animation of the 3D avatar. As a restriction, in this moment, can be cited especially the process of mapping entries to frames of position of animation, as well as inclusion in the model of facial expressions and improvements in rigging animation.

Facial animations and non-manual movements are not yet implemented in the formal model, however with the results of this work can be defined as building these new parameters following the same principles developed for the construction of the system to the arm and hands.

7 REFERENCES

[1] J. Loomis, H. Poizner, U. Bellugi, A. Blakemore, and J. Hollerbach, "Computer graphic modeling of american sign language," *SIGGRAPH Comput. Graph.*, vol. 17, no. 3, pp. 105–114, Jul. 1983. [Online]. Available: http://doi.acm.org/10.1145/964967.801139

[2] R. Quadros and L. Karnopp, "Língua de sinais brasileira: Estudos linguísticos," *Artmed*, vol. 1, 2007.

[3] W. Stokoe, "Sign language structure: An outline of the visual communication systems of the American deaf," *JOURNAL OF DEAF STUDIES AND DEAF EDUCATION*, vol. 10, no. 1, pp. 3–37, WIN 2005.

[4] H. Sagawa and M. Takeuchi, "A teaching system of japanese sign language using sign language recognition and generation," in *Proceedings of the Tenth ACM International Conference on Multimedia*, ser. MULTIMEDIA '02. New York, NY, USA: ACM, 2002, pp. 137–145. [Online]. Available: http://doi.acm.org/10.1145/641007.641035

[5] D. Antunes, C. Guimaraes, L. Garcia, L. Oliveira, and S. Fernandes, "A framework to support development of sign language human-computer interaction: Building tools for effective information access and inclusion of the deaf," in *Research Challenges in Information Science (RCIS), 2011 Fifth International Conference on*, May 2011, pp. 1–12.

[6] S. Gibet, "Synthesis of sign language gestures," in *Conference Companion on Human Factors in Computing Systems*, ser. CHI '94. New York, NY, USA: ACM, 1994, pp. 311–312. [Online]. Available: http://doi.acm.org/10.1145/259963.260372

[7] S. Geitz, T. Hanson, and S. Maher, "Computer generated 3-dimensional models of manual alphabet handshapes for the world wide web," in *Proceedings of the Second Annual ACM Conference on Assistive Technologies*, ser. Assets '96. New York, NY, USA: ACM, 1996, pp. 27–31. [Online]. Available: http://doi.acm.org/10.1145/228347.228353

[8] J. E. Chadwick, D. R. Haumann, and R. E. Parent, "Layered construction for deformable animated characters," *SIGGRAPH Comput. Graph.*, vol. 23, no. 3, pp. 243–252, Jul. 1989. [Online]. Available: http://doi.acm.org/10.1145/74334.74358

[9] M. Hrúz, P. Campr, Z. Krňoul, M. Železný, O. Aran, and P. Santemiz, "Multi-modal dialogue system with sign language capabilities," in *The Proceedings of the 13th International ACM SIGACCESS Conference on Computers and Accessibility*, ser. ASSETS '11. New York, NY, USA: ACM, 2011, pp. 265–266. [Online]. Available: http://doi.acm.org/10.1145/2049536.2049599

[10] "Gesture Builder software," http://www.vcom3d.com/?id=gesturebuilder, accessed: 2015-05-05.

[11] "Max E.T. software," http://www.einfach-teilhaben.de/DE/StdS/Home/stds_node.html, accessed: 2015-05-05.

[12] " Sign 4 Me software," http://www.vcom3d.com/, accessed: 2015-05-05.

[13] B. Yi, F. C. Harris, Jr., and S. M. Dascalu, "From creating virtual gestures to "writing" in sign languages," in *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '05. New York, NY, USA: ACM, 2005, pp. 1885–1888. [Online]. Available: http://doi.acm.org/10.1145/1056808.1057047

[14] K. L. Murdock, *3ds Max 2012 Bible*. Wiley Publishing, 2011.

[15] A. Porfirio, K. Lais Wiggers, L. Oliveira, and D. Weingaertner, "Libras sign language hand configuration recognition based on 3d meshes," in *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*, Oct 2013, pp. 1588–1593.