This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement n: 761999

# EasyTV: Easing the access of Europeans with disabilities to converging media and content.

## D2.5 Sign language animation final implementation

**EasyTV Project**

*H2020. ICT-19-2017 Media and content convergence. – IA Innovation action.*

**Grant Agreement n°: 761999**

Start date of project: 1 Oct. 2017

Duration: 30 months

Document. ref.:

## Disclaimer



The EasyTV Consortium consists of the following partners:

| | Partner Name | Short name | Country |
|---|---|---|---|
| 1 | Universidad Politécnica de Madrid | UPM | ES |
| 2 | Engineering Ingegneria Informatica S.P.A. | ENG | IT |
| 3 | Centre for Research and Technology Hellas/Information Technologies Institute | CERTH | GR |
| 4 | Mediavoice SRL | MV | IT |
| 5 | Universitat Autònoma Barcelona | UAB | ES |
| 6 | Corporació Catalana de Mitjans Audiovisuals SA | CCMA | ES |
| 7 | ARX.NET SA | ARX | GR |
| 8 | Fundación Confederación Nacional Sordos España para la supresión de barreras de comunicación | FCNSE | ES |
| 9 | Unione Italiana dei ciechi e degli ipovedenti | UICI | IT |

| | |
|---|---|
| **PROGRAMME NAME:** | H2020. ICT-19-2017 Media and content convergence. – IA Innovation action |
| **PROJECT NUMBER:** | 761999 |
| **PROJECT TITLE:** | EASYTV |
| **RESPONSIBLE UNIT:** | CERTH |
| **INVOLVED UNITS:** | CERTH |
| **DOCUMENT NUMBER:** | D2.5 |
| **DOCUMENT TITLE:** | Sign language animation final implementation |
| **WORK-PACKAGE:** | WP2 |
| **DELIVERABLE TYPE:** | Demonstrator |
| **CONTRACTUAL DATE OF DELIVERY:** | 31-01-2020 |
| **LAST UPDATE:** | 31-01-2020 |
| **DISTRIBUTION LEVEL:** | PU |

**Distribution level:**

**PU** = *Public,*

**RE** = *Restricted to a group of the specified Consortium,*

**PP** = *Restricted to other program participants (including Commission Services),*

**CO** = *Confidential, only for members of the LASIE Consortium (including the Commission Services)*

# Document History

| VERSION | DATE | STATUS | AUTHORS, REVIEWER | DESCRIPTION |
|---|---|---|---|---|
| v.0.1 | 02/12/2019 | Draft | Georgios Gerovasilis (CERTH/ITI)<br>Nikolaos Kaklanis (CERTH/ITI)<br>Konstantinos Votis (CERTH/ITI)<br>Dimitrios Tzovaras (CERTH/ITI) | Table of contents definition and document structure |
| v.0.2 | 20/12/2019 | Draft | Georgios Gerovasilis (CERTH/ITI)<br>Nikolaos Kaklanis (CERTH/ITI)<br>Konstantinos Votis (CERTH/ITI) | First draft with all sections partially completed |
| v.0.3 | 08/01/2020 | Draft | Dimitris Konstantinidis (CERTH/ITI)<br>Kosmas Dimitropoulos (CERTH/ITI) | Added Chapter 4.5 |
| v.0.4 | 23/01/2020 | Complete version to be sent for peer-review | Georgios Gerovasilis (CERTH/ITI)<br>Nikolaos Kaklanis (CERTH/ITI)<br>Konstantinos Votis (CERTH/ITI) | All sections fully completed |
| v.1.0 | 31/01/2020 | Final version with peer-review comments addressed - ready to be submitted | Georgios Gerovasilis (CERTH/ITI)<br>Nikolaos Kaklanis (CERTH/ITI)<br>Konstantinos Votis (CERTH/ITI)<br>Dimitrios Tzovaras (CERTH/ITI) | Final version with peer-review comments addressed |

# Definitions, Acronyms and Abbreviations

| ACRONYMS / ABBREVIATIONS | DESCRIPTION |
|---|---|
| 3D | Three-Dimensional |
| C3D File | Coordinate 3D File |
| IK | Inverse Kinematics |
| LOD | Level Of Detail |
| Motion Builder (Autodesk) | MB |
| MoCap | Motion Capture |
| SSL | Spanish Sign Language |

# Table of Contents

# List of Figures

# Executive Summary

This document is the final version of the deliverable D2.5 concerning the development and production of the EasyTV sign language animation method. This method will animate signer avatar focusing on realism. A signer avatar will receive the recorded sign language both gestures and facial expressions and simulated in realistic way through the joint animation of both face and hand motions. In this version, the requirements and the architecture of the module are outlined. Also, an analysis of the phases that compose the module is given, regarding the development and improvement of avatar and the connection with crowdsourcing platform and other services.

# 1. INTRODUCTION

The implementation of sign language service on the EasyTV project aims to visualize the data which have been obtained from the crowdsourcing platform, using an animated avatar. The service animates a signer avatar with high accuracy and realism in order to increase sign comprehesion. The process of the reproduction of the recorded signed language focuses on synchronization of the joint animation of both face and hand motions and their embeddings into graph topologies. The product of this service is ready to be used on different devices such as Desktop PCs (equipped with a web-browser) or an Android TV..

The present document is organized as follows:

- **Chapter 2** presents the updated and final architecture of sign language 3D production system.
- **Chapter 3** analyses the intermediate tests feedback and the actions taken.
- **Chapter 4** describes the final development of sign language avatar.
- **Chapter 5** describes the connection with Crowdsourcing platform and the implementation with other services of EasyTV.
- **Chapter 6** presents some indicative use cases.
- **Chapter 7** outlines the conclusions of this work.

# 2. SIGN LANGUAGE 3D REPRODUCTION SYSTEM UPDATED ARCHITECTURE

This chapter presents the aspects concerning the architecture of EasyTV signer avatar service. These include the interconnection with other EasyTV services as well as, the internal components that compose the avatar development.

## 2.1. Input & output

In the deliverables D1.3 *"First release of the EasyTV system architecture"* and D2.1 *"Sign Language animation preliminary development and production"* the architecture of the Sign Language Production Module has been defined. The module consists of three submodules, a) sign language crowdsourcing platform, b) capturing module and c) the humanoid avatar. The complete architecture including interactions with other components is shown in Figure 2-1.

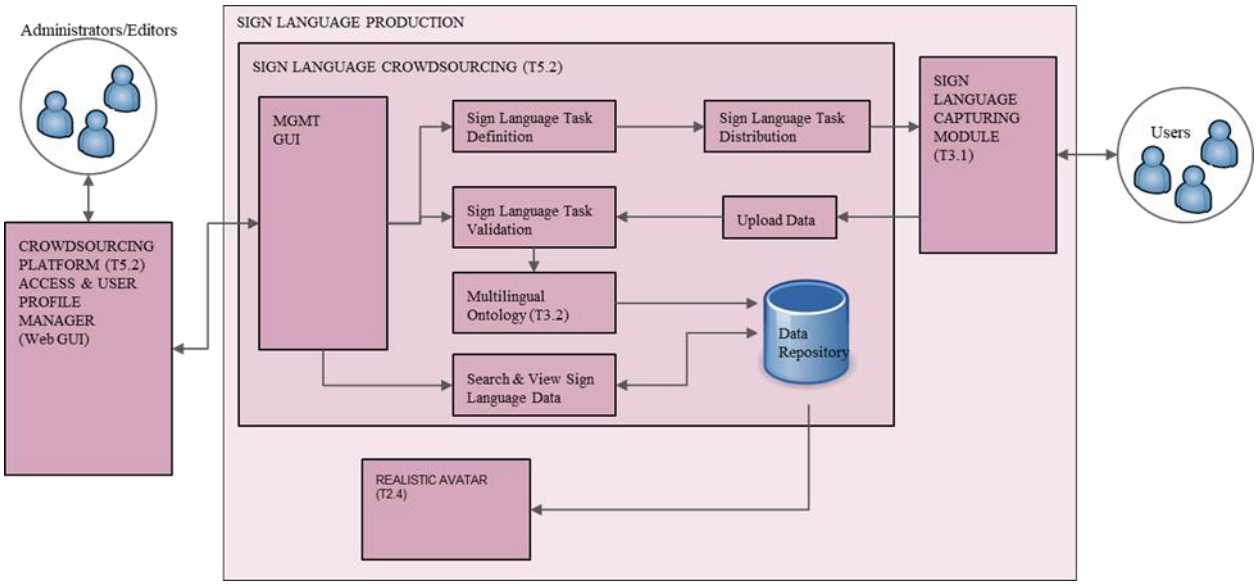


**Figure 2-1 Complete Architecture**

As mentioned in D2.1, the output of the Sign Language Capturing Module is a collection of motion data, captured from human signers. EasyTV adopts a novel architecture for providing signing avatar animation through a series of processing steps applied on motion capture data, eventually leading to crowdsourcing ready-to-play motion files as shown in Figure 2-2.

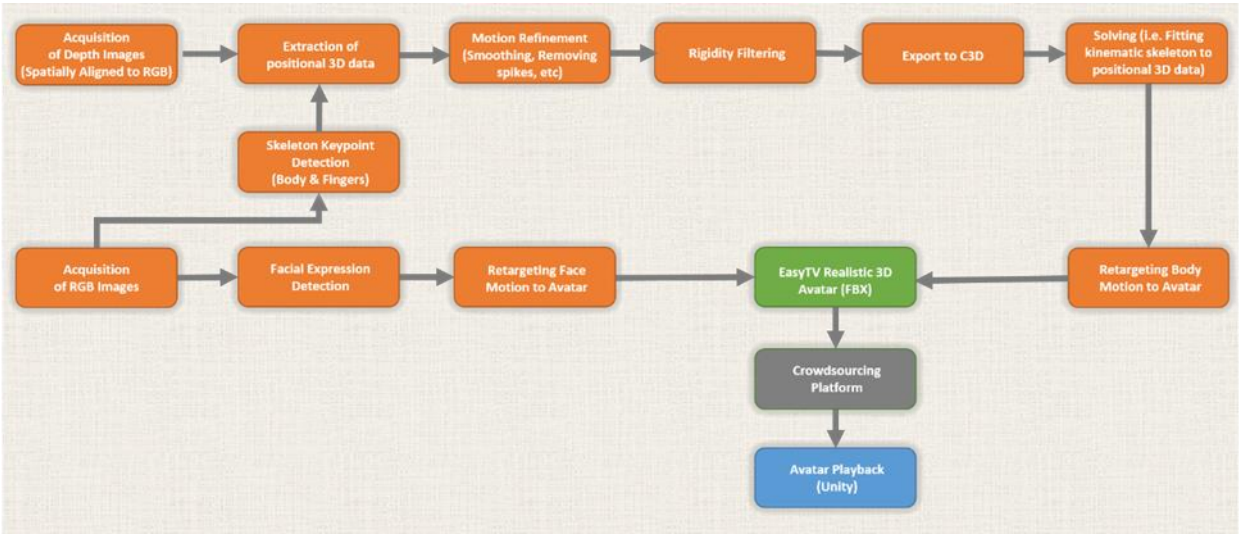


**Figure 2-2** Architectural scheme for generating crowdsourced 3D animation for Sign Language in EasyTV.

The first steps include the acquisition of RGB and depth images, the construction of 3D data from 2D keypoint detection algorithms, as well as, data filtering for motion refinement and exporting to an industry standard motion file format. Then, motion is retargeted to the avatar to produce 3D animation. Although animation for the body and face is generated on the same 3D avatar, the two retargeting procedures are different. At the final step, animation data is exported into another industry standard file format and uploaded, after a bit of optimization and categorization, on a data repository inside the Crowdsourcing Platform for later use by the EasyTV services and modules. The purpose is to convert the recorded signs to human moves from avatar representing the signer. Therefore, the recorded data will be the main input for the 3D reproduction system. A subtitle file from subtitles service can be provided through CS App and HbbTV to the avatar service in order to define which one from the recording signs should be reproduced. The output of the 3D reproduction system will be an avatar signer visualizing the recorder sign, stored in crowdsourcing repository.

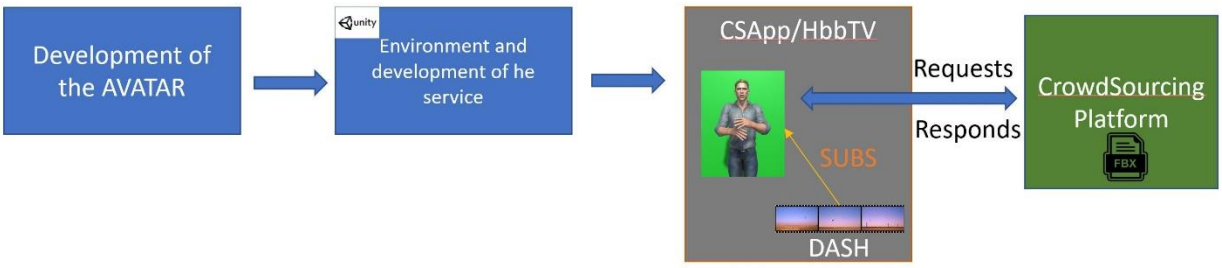


**Figure 2-3 AVATAR Service pipeline**

As shown in Figure 2-3 the Avatar Service is implemented in CSApp. The user who navigates inside CSApp can enable Avatar Service. When the Service as activated a window with an avatar appears in the right down of the screen. All DASH streams, CSApp includes, come along with metadata files. One category of these files is “Subtitles”. A subtitle file contains text and timestamps. Each DASH Stream can contain one or more subtitle file, in different languages. When a DASH file is chosen and the Avatar Service is activated, CSApp can provide a link which contains the subtitle file of the specific stream. After a deserialization of the file, avatar creates a table with most of the words of the subtitle in queue and its timestamps. When the timestamp indicates, avatar makes a request to the crowdsourcing platform in order to receive the motion file of the specific word and reproduce it.

# 3. FEEDBACK FROM INTERMEDIATE TESTS AND ACTIONS TAKEN

This chapter describes the content which used for the intermediate tests and a short resume regarding the development. The results are presented in table and then the following actions are also described in short.

## 3.1. Content of Intermediate tests

For the intermediate tests a sequence of three phrases was decided to be shown to the final users. Specifically three short videos in Spanish Sign Language (SSL) was provided by FCNSE in order to be reproduced from Avatar Service. The users had to view a video with the avatar to sign these three phrases. Thus, they could view the video more than once in order to catch up with some details maybe missed. In the end of view of each phrase, a side-by-side video showing the human signer in original video and the avatar is presented in order to have a better judgement for the differences and to understand all the words of the phrases. Figure 3-1 shows a frame of this video.

**Figure 3-1 Intermediate test video**

In D2.1 the fully development procedure of creating an avatar was described in detail. Considering the final result of the D2.1 the avatar that was used in intermediate tests had some changes and upgrades. The lip sync method was replaced by a manually added blendshape system for the face. Specifically, some blendshapes were added and a combination of them could form almost every facial expression of the signer. At this stage of development there were not any recording data regarding the face and therefore all the expressions were added manually.

Additionally, avatar's appearance in general has also some updates. The most obvious was the quality and the color of the clothes. It was preferred a strong color with an updated texture in order to have a more clearly view for the hands. Also, the general texturing over the skin had an improvement without any significant size cost to the final file. A major issue for the tests was to examine the level of comprehensiveness a virtual avatar can reach. Therefore, there were many refinements regarding the movement, especially the abnormal poses. Working almost in every frame the corrections was basically aimed to have an identical video with original regarding time and positions, especially body-arms-fingers movement. Lastly, the visual environment was improved. A wall and a floor were added in order to create a more natural feeling for the viewer and a completely new lighting was developed, using as a base the 3-point light set up which is commonly used in television and cinematography.

## 3.2. Results

After the viewing, each user had to fill out a questionnaire. The main purpose of the questionnaire was to define the level of comprehensiveness regarding the signs and the acceptance the viewer had of a non-human signer. Also, through targeted comments from the user the major issues could be defined. Figure 3-2 shows these results

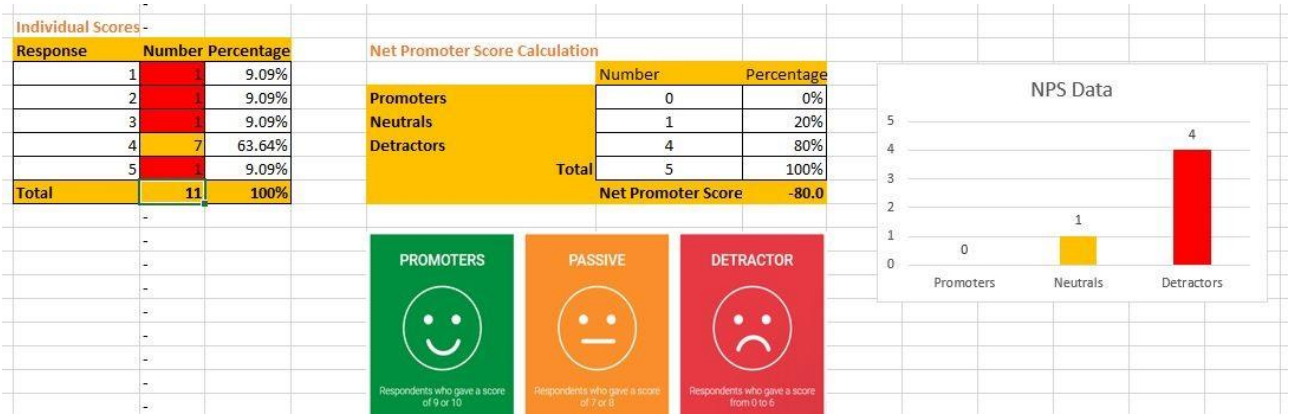


**Figure 3-2 Intermediate test results**

As it can be seen on the figure there was in general terms a negative opinion about the avatar. Considering the comments, it seems that the major issue was the lack of actual face recording data. Even with the additional blendshapes a manual procedure could not perform as a whole the exact facial expressions. Also, the manual refinements on the rest of the body seemed clumsy and mechanical. Even though an identical movement seems in the video, the comments were not positive enough. The reason might be that the avatar as technology and appearance did not convinced the community that it is able to replace a human signer.

Considering the results of the tests and the comments the next actions were clear enough. The most important was to focus in facial data and give as close to reality identical expressions as possible. The main aim was to improve and maybe replace the blendshape system that it was used till now. But that was depend from the capturing module system and the data could store. Additionally, a more realistic avatar had to be redesigned in order to reach to more human alike standards the community could accept. Though, any possible upgrade regarding texture and appearance must be occurred without any significant size cost. The smooth movement of arms could be ensured replacing manual refinements with limitations over specific parts and joints. All the actions taken are described in detail in the following chapters.

# 4. SIGN LANGUAGE AVATAR

This chapter discusses the methodology used to rig and texturing a 3D model. Limitations were added in order to avoid possible abnormal positions. Face development is also described in detail due to the importance in comprehensive of sign language.

## 4.1. Avatar Rig and texture

The preliminary development of a humanoid avatar with Sign Language capabilities was described in document D2.1 *"Sign language animation preliminary development and production"*. Specifically, a high texture-quality avatar was developed with a rig, matching the one was used on T3.1 Recording Module. Therefore, all skeleton joints were corresponding to the specific points the recording module could capture. A (local) JSON File provided the positions of each joint in specific timestamps, which were stored during capturing. Figure 4-1 shows the keypoints of the skeleton.

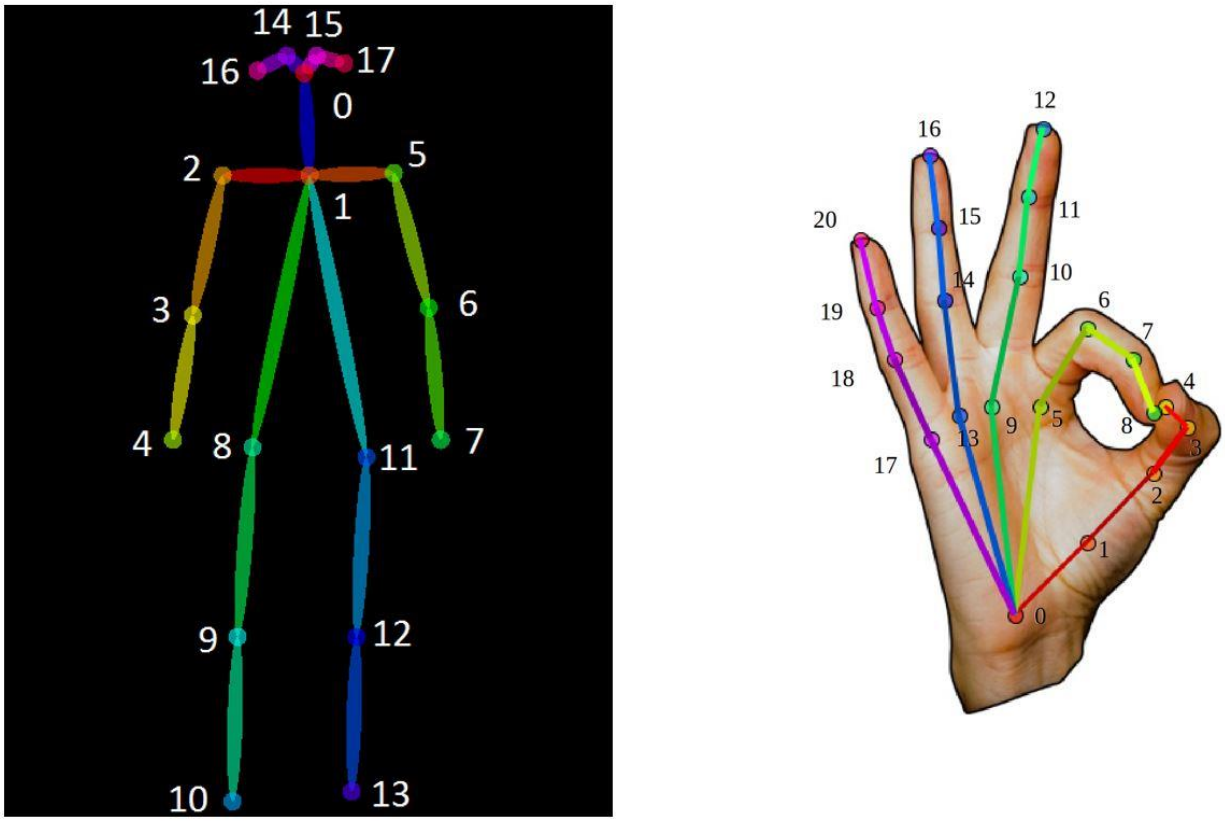


**Figure 4-1 Body and finger keypoints**

The main issue was that using only the positions of a joint on local coordinates could not deliver the expected results on 3D World Space. The lack of rotations caused some random results and the usage of Inverse Kinematics (IK) could not solve the issue. Therefore, the avatar had a lot of abnormal movement, especially with arms penetrating the body, even after adding limitations to joints and making some basic corrections.

At this early stage of development there was no face data to implement. It was already known that the importance of the facial movement is very critical in all sign languages. Considering the lack of facial data a lip synchronization approach was followed in order the avatar to pronounce each word. The main aim was firstly to help possible “lip readers” to have a better comprehensive experience and secondly to avoid have a static face, which will cause confusion and possible negative opinions regarding the realism.

Furthermore, another issue was the proper order of timestamps which define the whole movement and the order of the words too. In some cases, the avatar was moving to slow, especially in complicated movement where most of the joints were changing, causing desynchronization with the rest of the words and its speech. Figure 4-2 shows the avatar in its final form as described in D2.1.

**Figure 4-2 AVATAR**

In general, the avatar could sign the recordings from a local file but with a lot of manual refinements and the lack of facial expressions proved to be overwhelming for realism.

The first changes of the avatar were about its general appearance. The most important change was the skin texture and color. The main objections about not to be realistic were the comic-alike color and tone the skin emitted. Additionally, the eyes was almost look like as a part of the skin reducing the level of realism in even lower levels. Figure 4-3 shows all the textures for the avatar. The general procedure was described and analyzed in D2.1.

**Figure 4-3 Avatar 's texture**

For this occasion, red tone was added when needed in order to give a more human-alike look. Also, improvements were made regarding some parts of the face. All these corrections occurred after testing the avatar inside the scene with the specific light was intended to use, which will be described in the next chapter.

Another major change was about clothing. First, an upgrading in resolution was chosen without affecting the final size of the avatar. As mentioned before, the size of the final avatar is really important because it will be implemented in a web based application (web GL or Android TV) and it would be preferred to have the smaller possible size without have any significant reduction of quality for the user. Also, a one-color shirt was chosen without further details to avoid any distractions and to create a monochromic contrast with hands during signing. After all these changes, the avatar was re-tested independently for occlusions and weight errors during the movement.

The improved avatar was used for the Intermediate tests. Considering the results and the comments a slightly different approach was further applied. As mentioned before the final service must be implemented with web features (WebGL / Android). Therefore, all the design and development choices were aiming at high standards for a mobile version. Regarding the updated approach an avatar with 4K textures was designed and then downgraded for a mobile version. The reduction concerns the number of polygons a texture consists of. The visual result for a viewer, especially considering that the avatar will be appear in a part of a TV or in a tablet, would not have any significant difference. Figure 4-4 gives an example of the difference in clothes for the updated avatar. Furthermore, the shirt designed with the sleeves up in order to emphasize and distinguish the movement of the arms, as suggested.

**Figure 4-4 High Quality clothes**

The need for a more human alike avatar, as the test showed, indicates that the skin had to be improve even more. Especially in the face all the movement had to be clearer and more distinct for the viewer. Figure 4-5 shows the updated skin and Figure 4-6 shows the picture of the final avatar that was used, indicates more clearly the large number of polygons that form the skin. More polygons means higher quality texture. The most important parts of the face (mouth, eyes and nose) was also enhanced with even more polygons in order to show the lightest detail.

**Figure 4-5 Updated Skin**

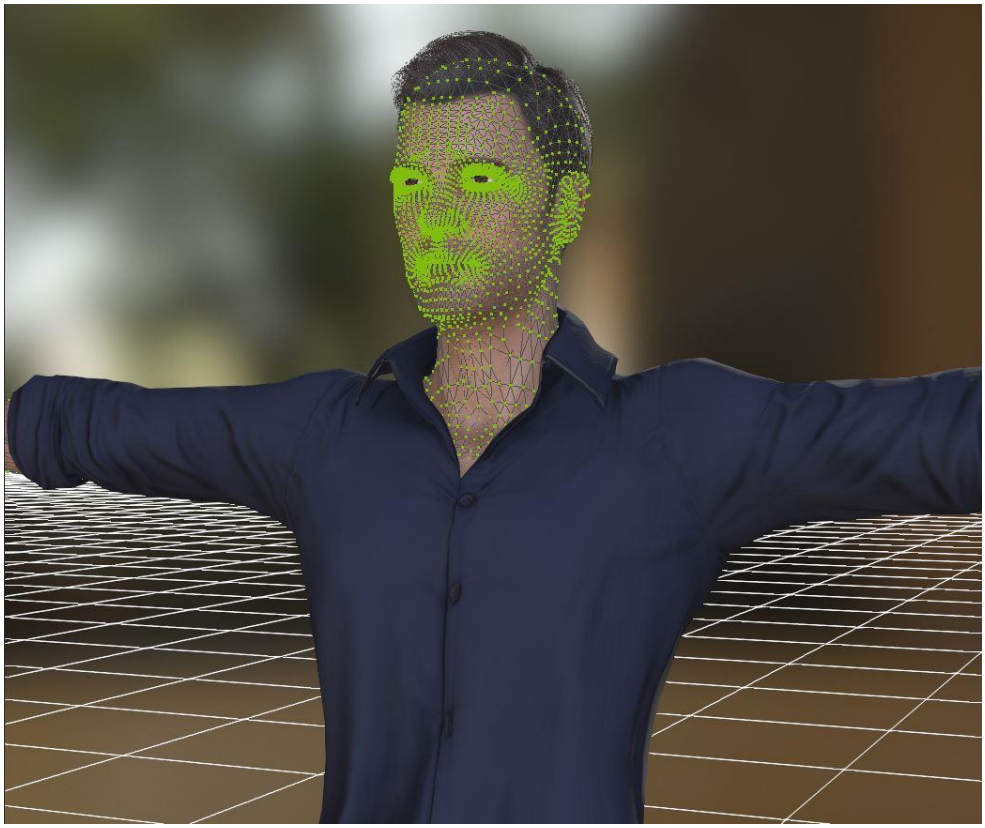

**Figure 4-6 High poly texture**

## 4.2. Updated Rig

The capturing module have updated the rig regarding the body. The main difference was that the most important bone moved from the neck to the hips. It has to be mentioned that this is the most commonly used composition of a skeleton concerning avatar development. Figure 4-7 shows the new rig. Keypoints 1 and 8 consists as the most important root joints.

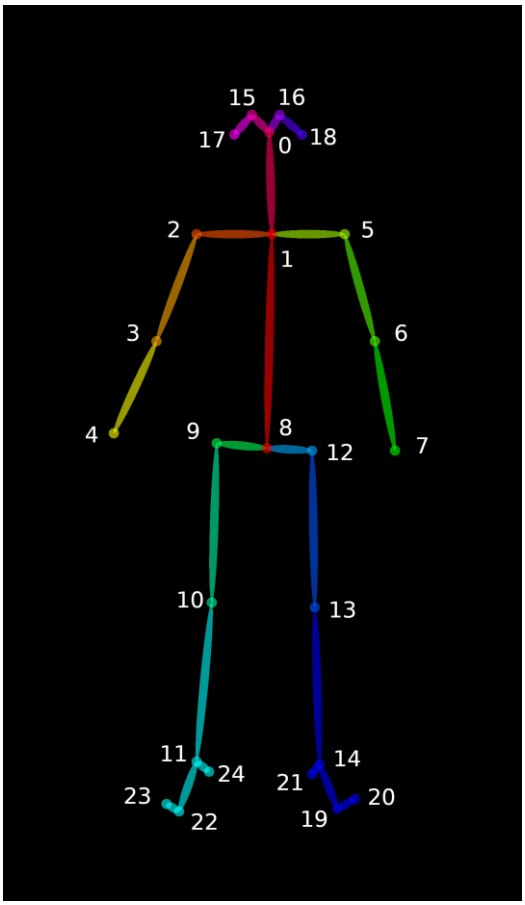


**Figure 4-7 Capturing module updated skeleton**

The new keypoints had to be corresponded to the avatar joints. Therefore, rig was updated as it can be seen in Figure 4-8 . Also, some bones added that do not participate to the reproductions of the data. The reason was to help fixing possible visual defects due to mesh errors and skin weights. Their role was currently supportive. Before final export, the avatar was tested again in extreme poses and all the visual errors were fixed.

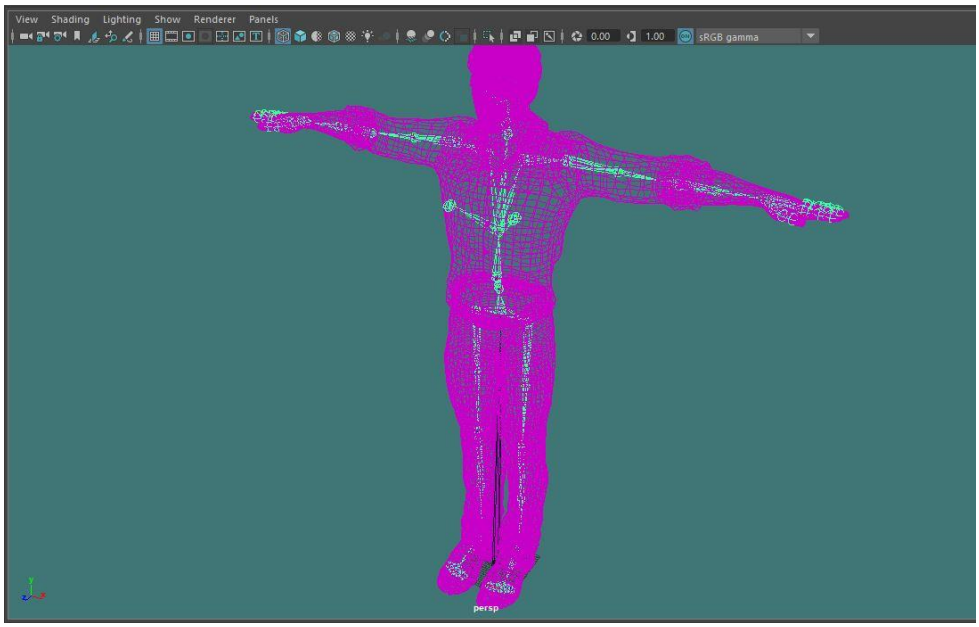


**Figure 4-8 Avatar Rig**

## 4.3. Face

As mentioned before, face expressions have the most important role in Sign Language. Words who have same sign but different expressions have different meaning. Therefore, the challenge is to create an avatar who can reproduce all the expressions of the signer with the slightest detail.

For the intermediate tests, there was not recorded data for the face. As mentioned before a Lip synchronization method was tested, where the avatar could pronounce each phoneme using a combination of blendshapes. As appeared to be, a lip sync technique was very difficult to implement due to low level of comprehensiveness. Lip reading is a method that is used by people who want to communicate directly and fast but is not so commonly used in community. So, it proved that it would have an uncertain result and acceptance. Therefore, it was decided to add by hand all the face expressions as shown in the video.

In previous lip sync method, a set of blendshapes was created just for form the phonemes for the letters. In order to accurate reproduce all the face expressions additional blendshapes were added and the older ones were tested to the updated face texture. Figure 4-9 shows some of these blendshapes.

**Figure 4-9 A set of blendshapes**

A combination of blendshapes consists of a facial expression as was made from signer. Figure 4-10 shows a frame from the video was used for the tests. The signer has a specific facial expression and avatar uses some of the blendshapes like “right eye blink” and “open mouth” in order to make identical expression.

**Figure 4-10 Avatar using blendshapes**

With the use of face recording data, a different approach was applied. In the same JSON file as before face keypoints were added, with its positions. Therefore, the same keypoints had to be matched with spots over the face of the avatar. Figure 4-11 shows the facemap that was used from capturing module.

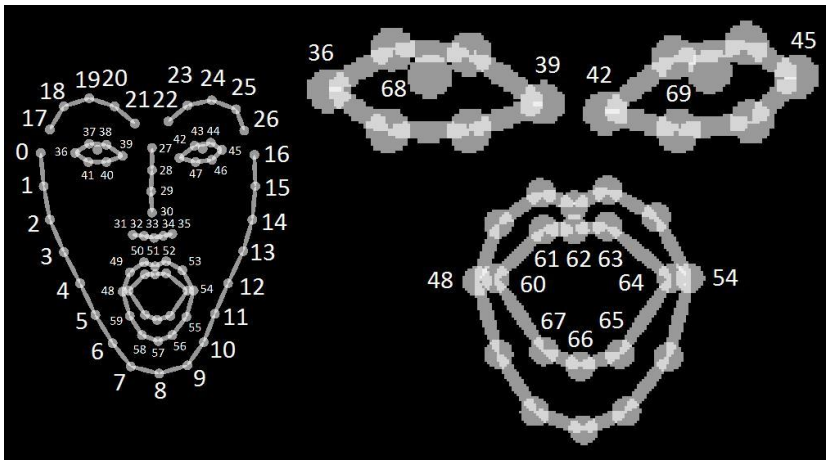


**Figure 4-11 Face Keypoints**

Regarding the avatar these key points must be represented either from blendshapes either with bones (of a rig) which will “drag” the texture in the specific position points. Blendshapes method is commonly used by giving some weights (i.e. 0-100) and have the corresponding result. For example, if the blendshape is “open mouth” 0 stands for a mouth which is closed and 100 stands for a mouth wide open. JSON files consists of position data and therefore a face rig approach was applied. Figure 4-12 shows some of the keypoints from external point of view.

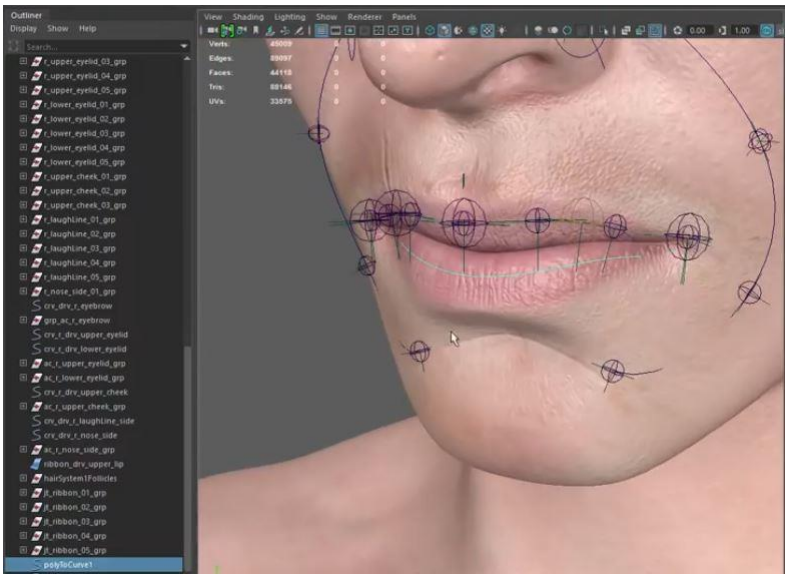

**Figure 4-12 Avatar's face keypoints**

In order to create new bones a connection with pre-existent rig is better to be established. For face rig the most appropriate connection is over the neck joint. Figure 4-13 shows the added face rig.

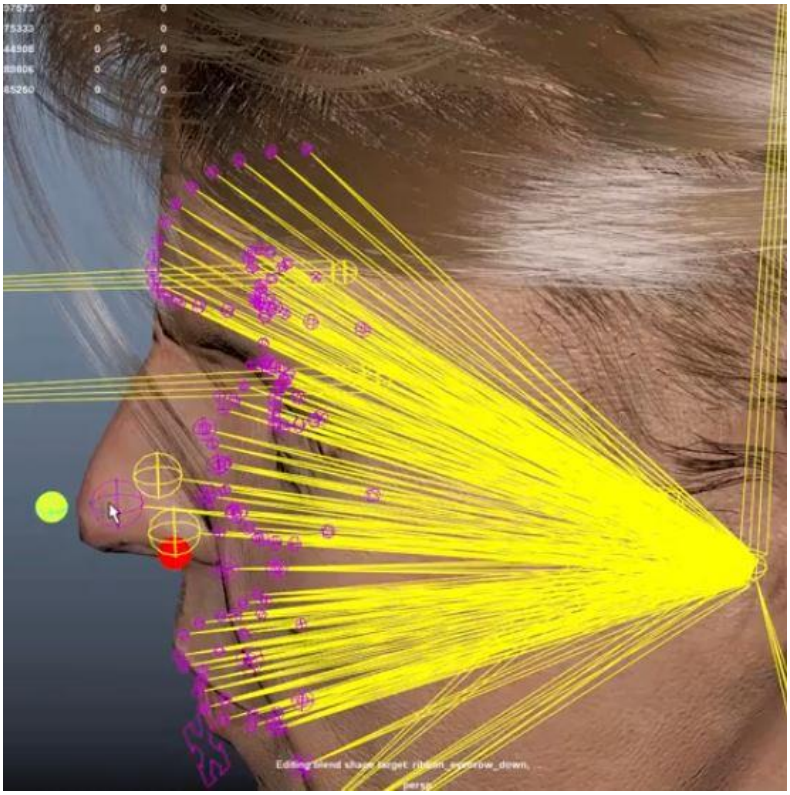

**Figure 4-13 Face rig**

When a new rig is added the most important aim is to customize how each joint affects the skin using weights. More weight means more transformation for the skin. In order to avoid abnormal poses each joint over the face must be checked, especially in extreme poses. Additionally, the area of the nearest joints can be affected with extra weight. Customizing the poses in extreme positions can prevent and restrict data spikes from recorded files. Figure 4-14 shows a frame from the procedure

was followed.

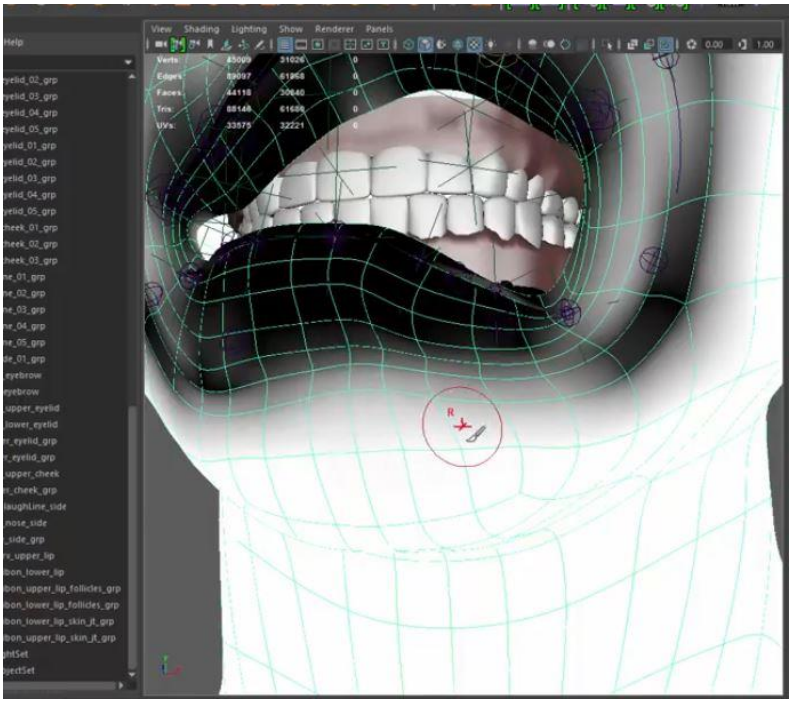

**Figure 4-14 Skin weights**

The biggest issue was that all the keypoints' positions of the same timestamp was not appropriate synched and the face has many distinct spikes. Furthermore, different signer face shapes could provide with no appropriate data. For the final form a new approach using an updated blendshape method was applied. Recording module could record in their data specific expressions. Each of these expressions is a combination of blendshapes which are attached in avatar's face. For example the expression "smile" consists of blendshapes like "open mouth", "blink right eye", "blink left eye", eyebrows up" etc. and in different weight sizes. Figure 4-15 shows an example of a blinking eye and its weights as described before.

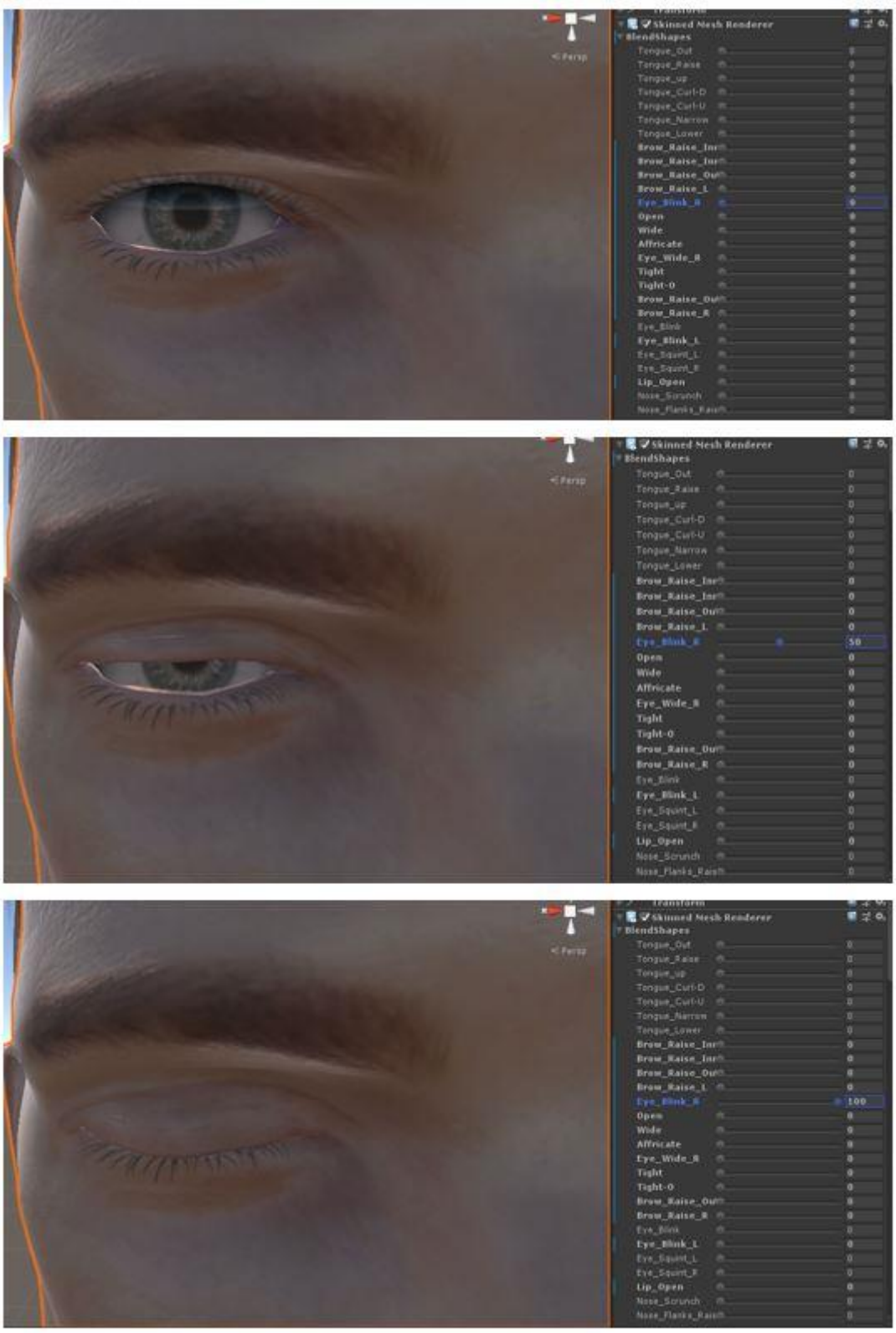


**Figure 4-15 Blendshape Eye Blink**

Using the avatar's blendshapes, the capturing module has created a set of face expression which consists of a set of these blendshapes. Each signer expression on its turn consists of the facial expressions above. The biggest advantage of this method is that any face could be used without any difference to the avatar's blendshapes and positions. Furthermore, a new combination of even more

blendshapes could cover all the expressions that the capturing module can record.

## 4.4. World Environment

One of the most critical aspects of comprehensiveness is the environment around avatar. At first, it is very useful for the user to watch the avatar inside a common and daily environment for him. A wall or a place around in a city can give the illusion that avatar is acting like a real human. One other aspect, and probably the most important, is lighting. As it happens in every video file, a bad lighting can change the aspect of what really happening or even totally destroy the quality. Therefore, after D2.1 a whole update occurred regarding lights. Unity[1] can place different kind of lights, i.e. spotlight, direct light. A combination of them can have the same visual result as a studio can have. Using three-point lighting set up, which is probably the most common used for one human cases, the shadows can be controlled and the contrast to be balanced. The key light is the main light source. It shines directly to the avatar and establishes the overall look. The fill light provides balance to the key light by filling the rest of the avatar's face with a softer light. The back light creates a flattering rim of light around the subject, separating him or her from the background. In avatar's use case one more light had to be added in front of the face as a fill light due to the animated texture. Also, one more light was added in front of the chest in order to have some additional contrast between hands and shirt and the signs to be clearer to the viewer. Figure 4-16 and Figure 4-17 shows an example of this development.


**Figure 4-16 Wall and lighting**

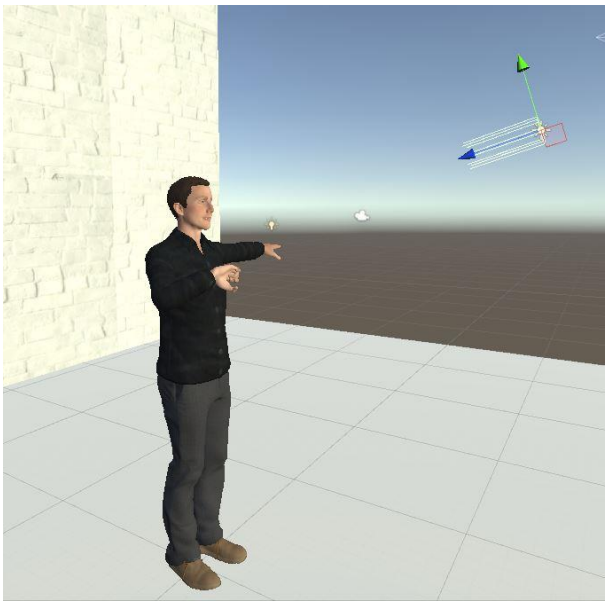

**Figure 4-17 Three point lighting**

Considering the feedback and comments from the intermediate tests, a kind different approach was followed. At first, the wall was replaced from a green solid background which can very easily be replaced with a blue one. A green or a blue color can be used from television producers in order to be replaced with a scene and give each time a different perspective to the viewer. Also, it is very commonly seen these colors to television and cinematography, thus the viewer audience is very familiar to.

Using a lot of same color in the scene can change the lighting appearance, if it is used like the way it was described before. In the contrary with lightmaps, light probes[2] provide a way to capture and use information about light that is passing through the empty space. The difference is that while lightmaps store lighting information about light hitting the surfaces in the scene, light probes store information about light passing through empty space. The primary use of light probes is to provide high quality lighting (including indirect bounced light) on moving objects, like hands, in the scene. The secondary use of light probes is to provide the lighting information for static scenery when that scenery is using Unity's LOD[3] system. As a result, the lighting is aiming mostly the avatar, creating a group of light around it. As a result, there is a clear view of the avatar ignoring thein a high level possible other reflections. Figure 4-18 shows an example of the new method.

**Figure 4-18 Green background and light probe**

Figure 4-19 shows the avatar in its final form. The high quality texture is obvious a main difference with previous model and the details in face are even more human alike than before.

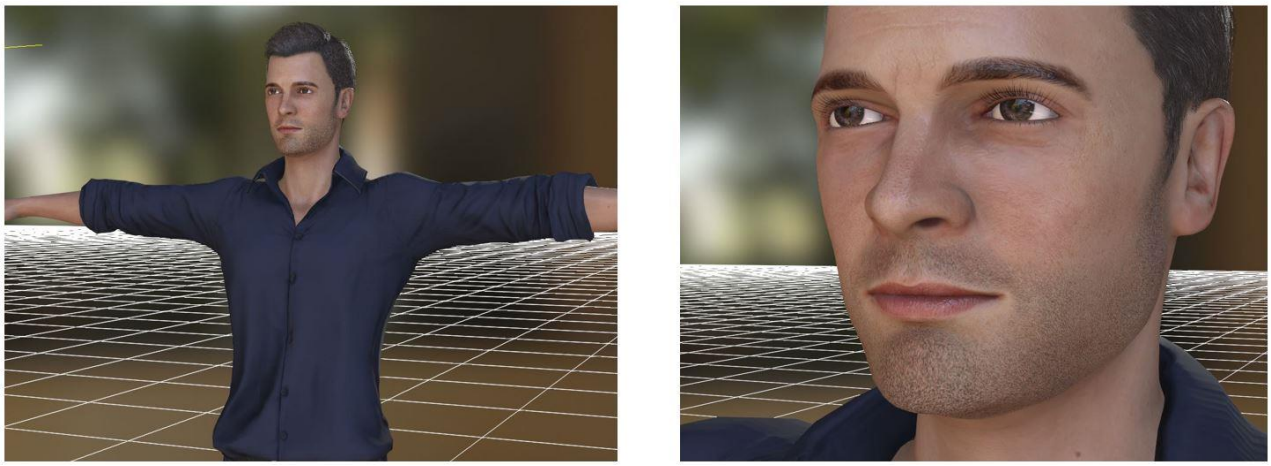

**Figure 4-19 Final form of the Avatar**

## 4.5. Recording Data

Animation requires motion to be encoded as translation, rotation, and scale values of a given object. As mentioned earlier in D3.7 *"Sign language capturing technology final version"*, a C3D file contains only positional data, i.e. translation values of markers (in our case, skeletal joints). Autodesk's MotionBuilder (MB) [4] software offers the 'Flexible MoCap' pipeline with which a kinematic skeleton

can be fitted to positional data within a C3D file. The first step is importing the C3D data to the current scene. Then, the animator has to place a generic 3D human model (MB's Actor) properly close to the corresponding keypoints in the scene as seen in Figure 4-20. A 'markerset' is then created to map the Actor's body parts and fingers to the corresponding 3D keypoints. Finally, the solving process is executed to fit the constrained skeleton model to the 3D point cloud. The above process is executed just once. Solving to any other motion data of the same structure and scale is executed automatically (i.e., without the animator's intervention), using the same Actor model.

**Figure 4-20 Solving using the Actor asset in Autodesk's MotionBuilder.**

An extra step included in the EasyTV pipeline and necessary for achieving finer results in the solving process is rigidity filtering. In this step, a skeleton of fixed bone lengths is defined and serves as a reference for scaling the bone lengths captured for each signer. This is done by calculating the unit vectors for each bone detected for a signer, at each frame, and then use the reference bone lengths to hierarchically scale all bones. Except for providing finer solving results, this process also brings invariance to different body sizes of signers, as well as, camera distances.

When the solving process completes, the Actor's motion can drive any rigged 3D model using motion retargeting. The only requirement is that the 3D model has to be 'characterized'. Characterization refers to the process of mapping a 3D model's rigged parts to a standard body model recognized by the retargeter. Since a 3D model can be rigged in any way, retargeting the same motion to different avatars might produce different results. After motion retargeting, animation is baked onto the character's skeleton so that the Actor is no longer needed for animation playback. Finally, a FBX motion file is exported. This file can then be imported to animate any 3D model having the same characterization.

In order to retarget facial motion, a setup is created for the 3D avatar in a way similar to the 'characterization' process required in the body retargeting process. The setup for the face requires a certain number of facial expressions to be stored for a given 3D avatar. It is preferred that the facial rig of the avatar includes a set blendshapes that can generate all these expressions rather than offering just low level facial controllers. Any number of expressions can be omitted in this setup, but the avatar will not be able to reproduce them in the retargeting phase. For every signing session, facial tracking is applied on the RGB images and the motion of the eyes, brows, nose and mouth is extracted. It is important to enhance the face tracker accuracy by providing a neutral frame with the keypoints properly placed on the signer's face. Finally, the extracted motion can be retargeted to any avatar having the aforementioned setup.

Although the retargeted body motion can be exported in a FBX motion file without including the avatar, the same does not hold for face motion. This is due to an inability of MotionBuilder's FBX exporter to separately store blendshape/morph information without including the corresponding 3D model within the file. For that reason, the final output of the EasyTV animation pipeline are FBX files that also include the animated avatar (i.e., animation files) instead of just motion information (i.e.,

motion files). This raises a problem on space requirements for storing animation files, since each file containing the avatar model reaches approximately 80MB in size and is impractical for uploading and storing to a remote crowdsourcing repository. In order to resolve the issue, a low-spec version of the avatar (i.e., without any assets) has been created especially for storing body and face animation information. This way, space requirements are reduced to ~10MB per file.

## 4.6. Abnormal movement and Corrections

Considering all the methods above, a motion file is received from the crowdsourcing platform and the avatar reproduces the movement. Even though, there are still very common issues with abnormal poses. Figure 4-21 shows an example of these poses.

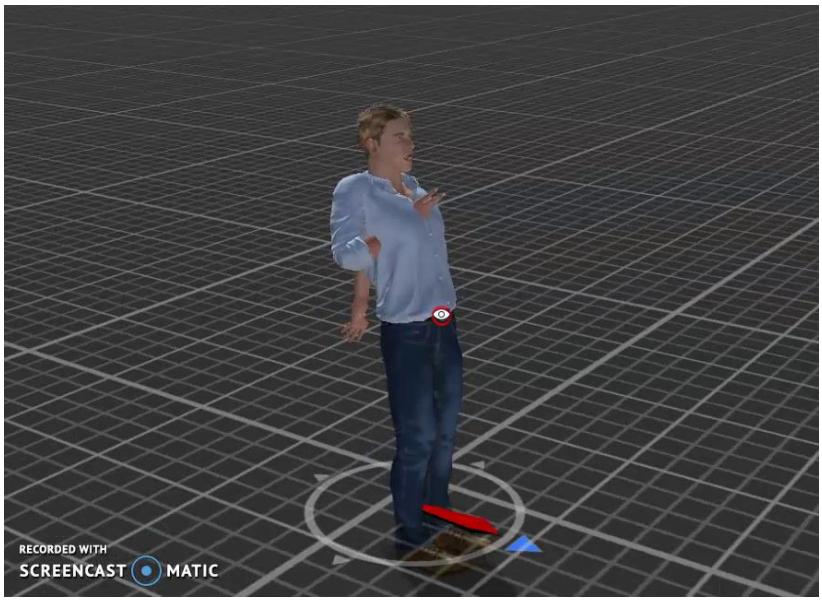


**Figure 4-21 Arm penetrating body**

In order to avoid as much as possible these poses further features were added to the avatar. At first there were added limitations to the most important joints as shoulders and wrists. Figure 4-22 shows how the limitations are applied. Specific angles were appointed, blocking any movement beyond them.

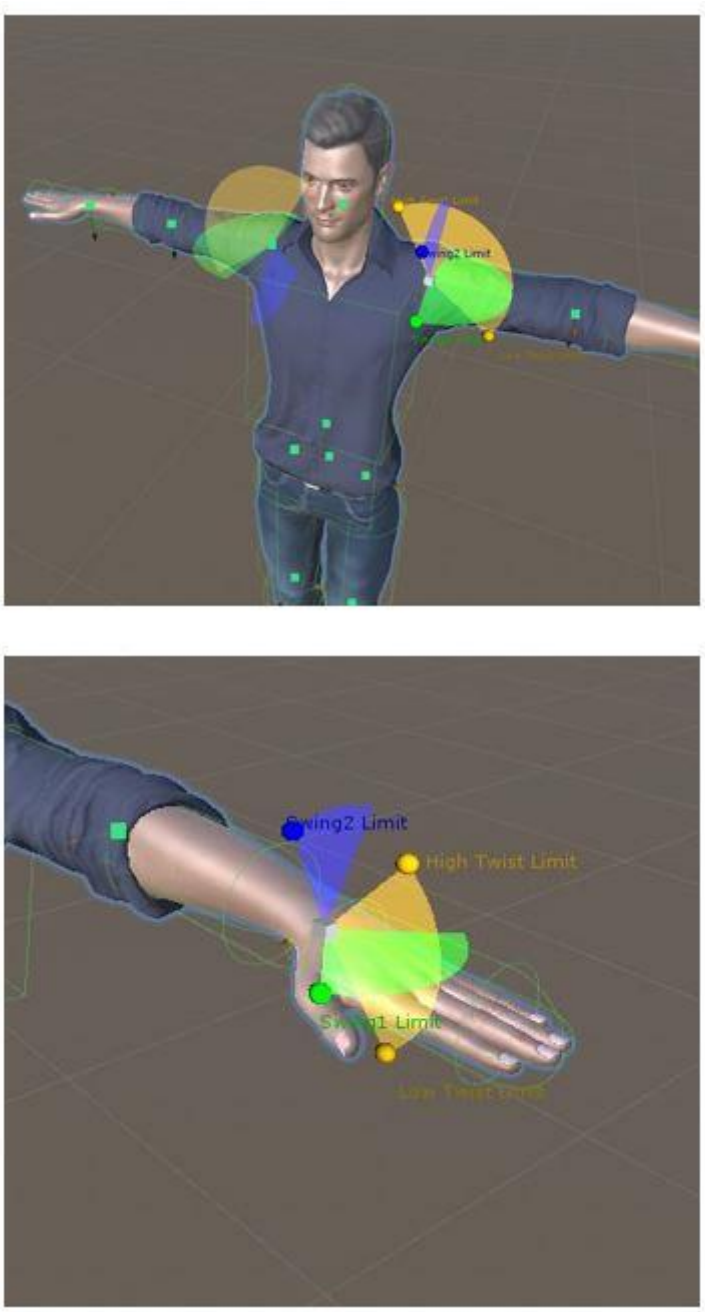


**Figure 4-22 Joint limitations**

Furthermore, in order to avoid collisions between body parts colliders were added over the main bones. Figure 4-23 shows the set of colliders were added over the body.

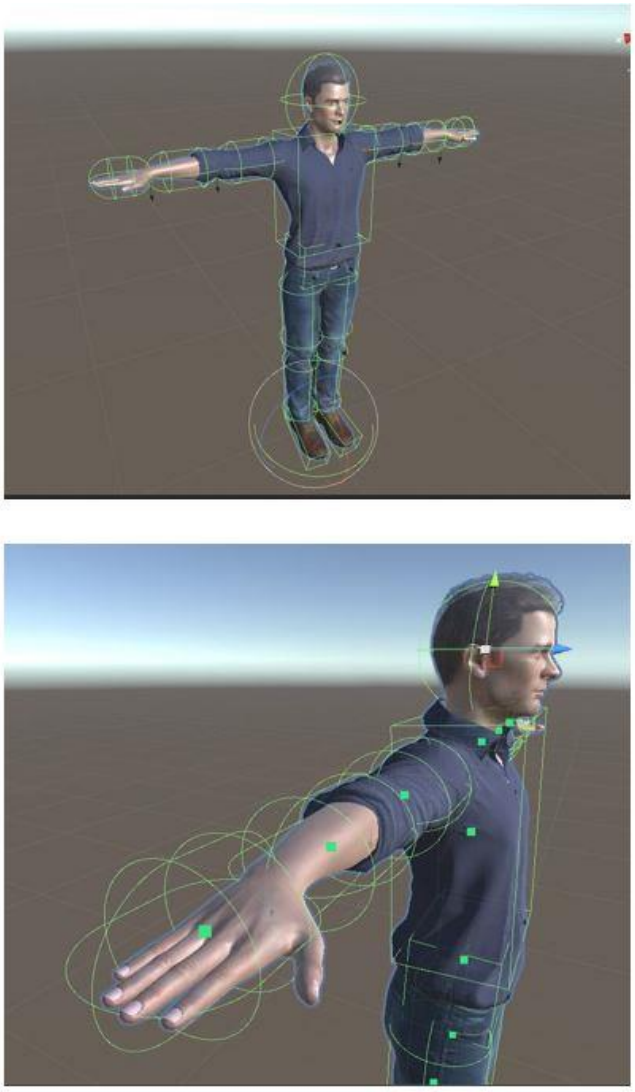

**Figure 4-23 Colliders**

The size of the colliders was even reduced and adopted in the size of avatar. Though, a jittering issue is created when preventing specific movement. When the motion file indicates that a joint has to move in a specific point, a repetitive movement trying to reach this point is created in a small area. The reason is that the limits prevent the joint to reach the point and during updates the result is a continuous back/forward movement. Figure 4-24 shows a part of this issue with blue skeleton to indicate the preventing position. Regarding fingers only the angle limitation was added. Due to very short distances between finger joints the collision part was chosen only for the hand as a whole. Otherwise, a continuously jittering was consisted with the slightest move.

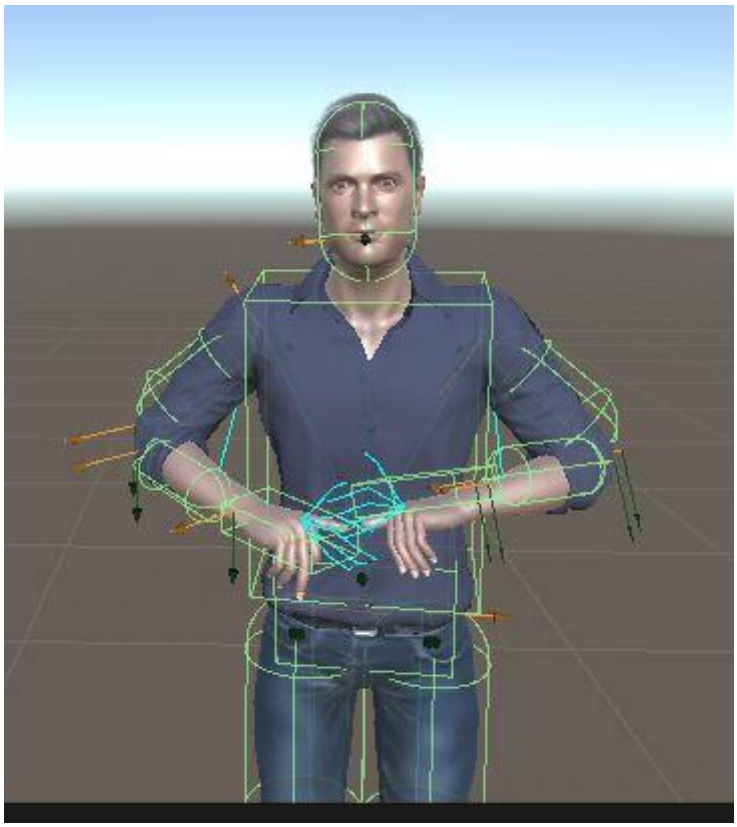

**Figure 4-24 Jittering issue**

As a solution a balance between weight and collisions needs to be found. Whenever the abnormal movement is more extreme the jittering is becoming clearer to the viewer. Adjusting weight to the colliders can reduce the power of collisions but with a possible loss to realism.

# 5. INTEGRATION WITH THE EASYTV PLATFORM

This chapter examines the methods were used for connecting with crowdsourcing platform and make requests. It also describes the communication with other EasyTV Services.

## 5.1. Integration with the Crowdsourcing Platform

As described in the architecture, the crowdsourcing platform actually works as a data manager. All the recordings from capture module are stored inside platform and waiting for requests from avatar service. Figure 5-1 shows the basic architecture regarding Ontology. The signs are stored as concepts and they have a JSON metadata file besides the basic MoCap file.

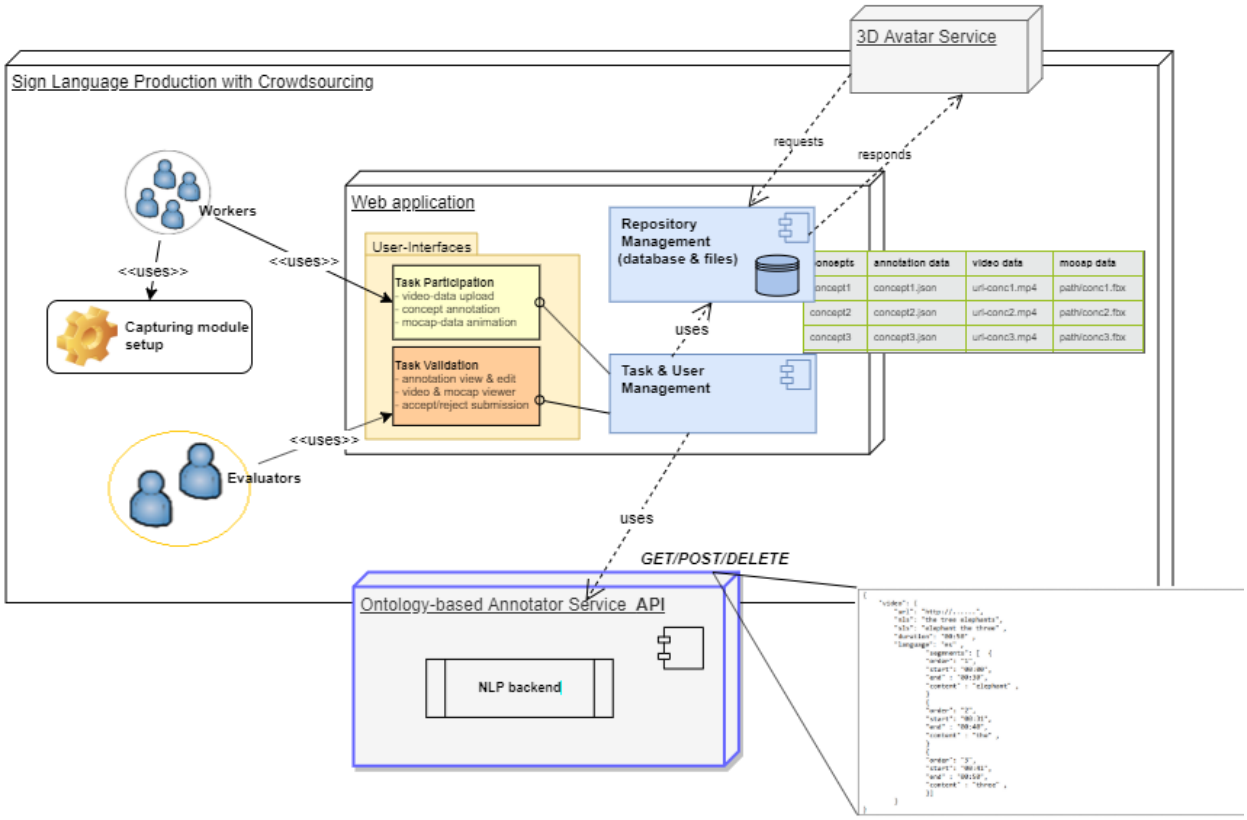


**Figure 5-1 Architecture for Ontology with Avatar Service**

The API Request method provided by Unity can handle communication the same way any other method is used in Web. Using POSTMAN in the Crowdsourcing platform can have the two figures below. Figure 5-2shows the param-requests and Figure 5-3 shows the header with the authentication hash-key which is used between partners. So, when the avatar revice makes a GET request can receive the fbx motion file, along with other metadata information. If there is not the specific word the API returns a 404-Not found response.

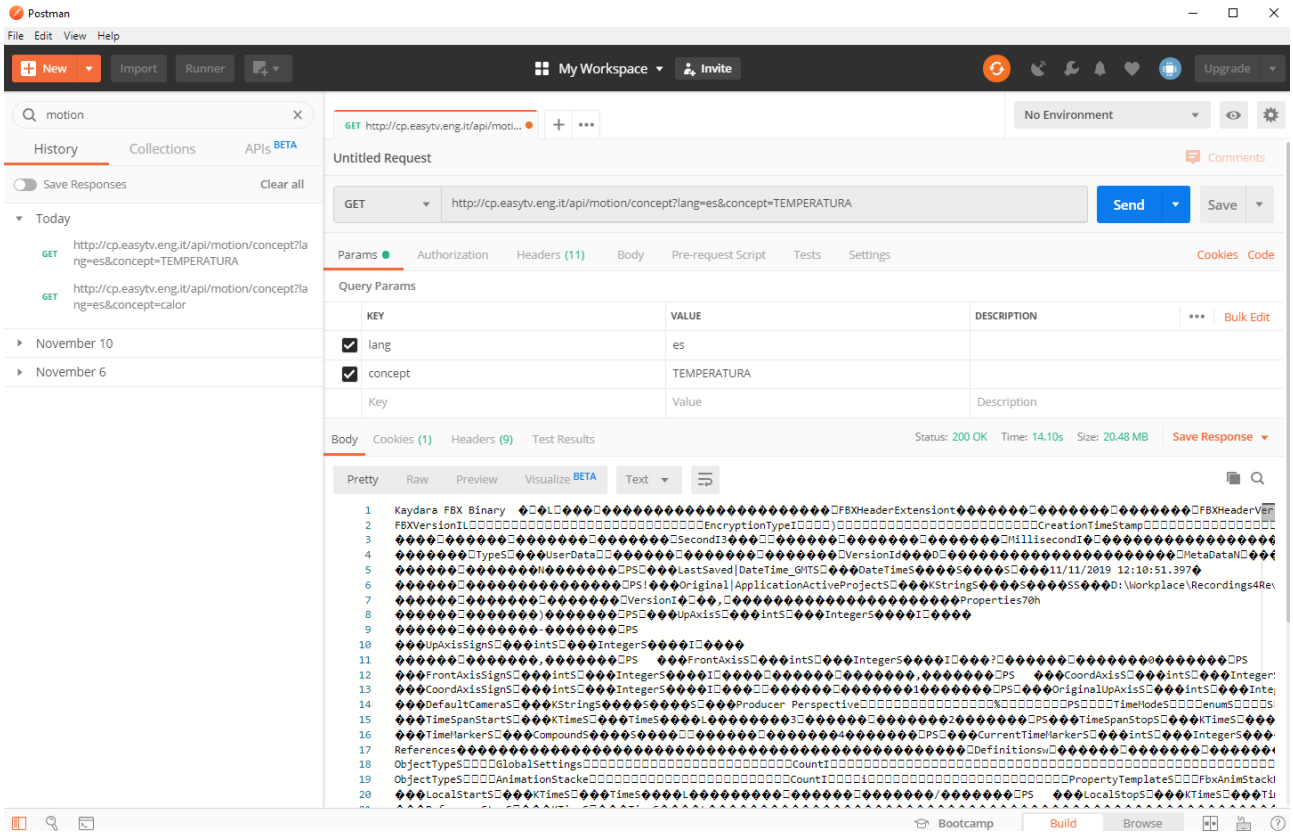


**Figure 5-2 Request Parametrs**

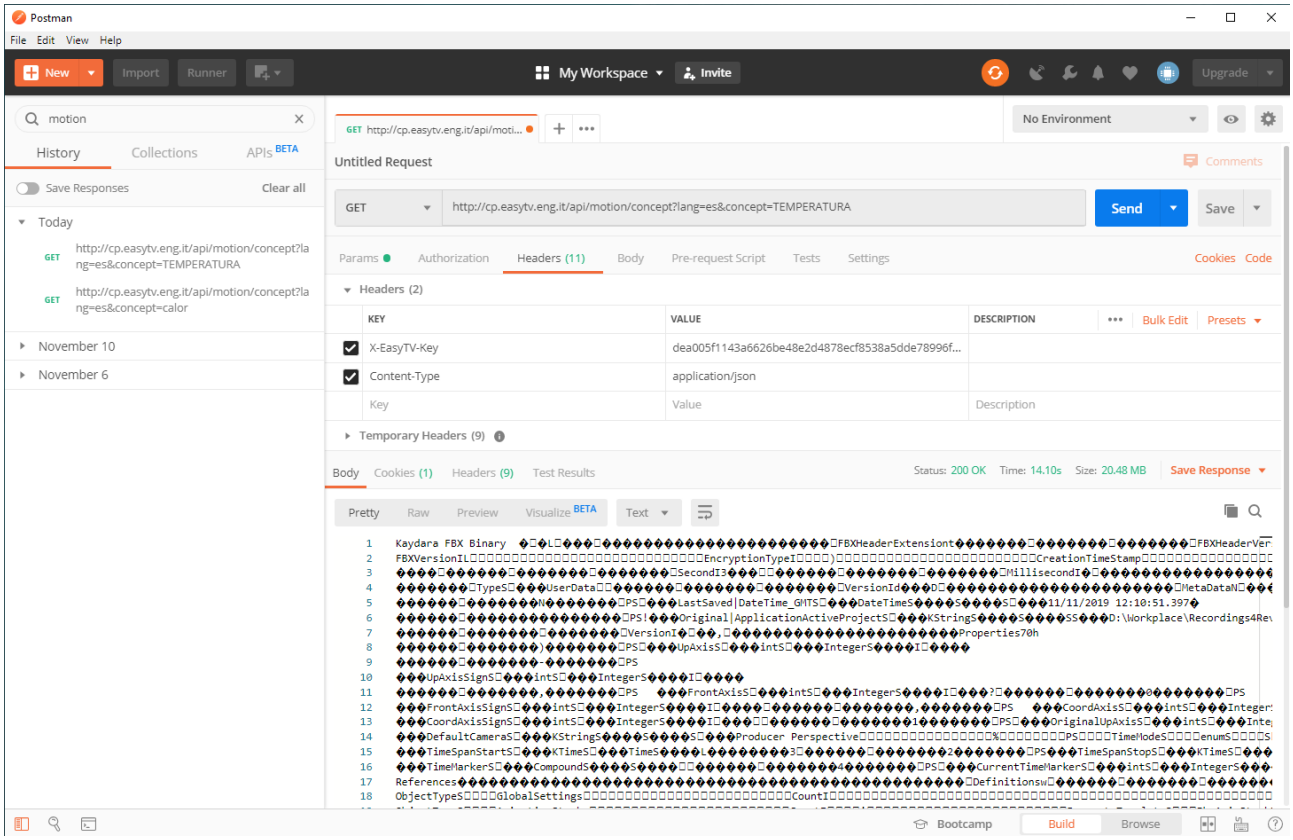


**Figure 5-3 Header Request**

## 5.2. Deserialization of subtitles file

During the intermediate tests each recording motion data file included a different phrase. These phrases were used as an example at first during the integration between avatar service and the crowdsourcing platform. But as described above the motion files are stored in words, which are called concepts. Therefore, an approach where only words are requested was followed. First of all, it should be mentioned that is currently impossible from this service to adopt all syntax and grammar rules of ever sign language the crowdsourcing platform has. On the contrary, a general rule was used for all sign languages aiming in a typical comprehensiveness for the meaning of the phrase. The subtitle file is written on a specific form with timestamps. For example, a sentence appears on screen for some seconds before being replaced with the next sentence. The specific sentence has the same timestamp for all words. These words are becoming part of a table with the same order as they appear. But before that the articles and a set of words which does not commonly signed are being removed. The final result is a GET request with a queue of basic words, like verbs and nouns. Of course, the order is the same as appears in the subtitle file.

## 5.3. Annotations - time stamps

All the recordings are starting and finishing with the signer in start position, which in case of Kinect is usually an A-position. This means that in every transition between words the avatar has to move its arms up and down which of course is not acceptable for the sign language. In order to avoid such abnormal behavior an additional information was added in metadata file. Specifically in recording module the user can define where the actual sign starts. This information stands on header of each request. So, starting from second word of the queue that is found in repository on GET request, the actual movement starts from the second that indicates that this is the beginning of the sign. The same approach is been followed for the end of each word. As a final result the avatar makes a continuous movement signing a sequence of words approaching the normal sign language and avoids to lower hands in each word.

## 5.4. Integration with the CSApp / HBBTV

CS App is an Android application. The most appropriate method to export the sing language service is as a Gradle Android project. During exporting some rules must be added. For example the “quit” method must be defined in order to avoid having some open issues when it is called. In any case all the methods and functions the service use are exposed in an approach where the CSApp can control. The final result is a small window where the avatar appears when the users call it. The same window shows on HbbTV but this time the service is exported as a webGL file. In that case service is called in a frame with specific dimensions defined from HbbTV. Figure 5-4 shows the menu for activation of the Service and Figure shows an example of the service enabled in CSApp.

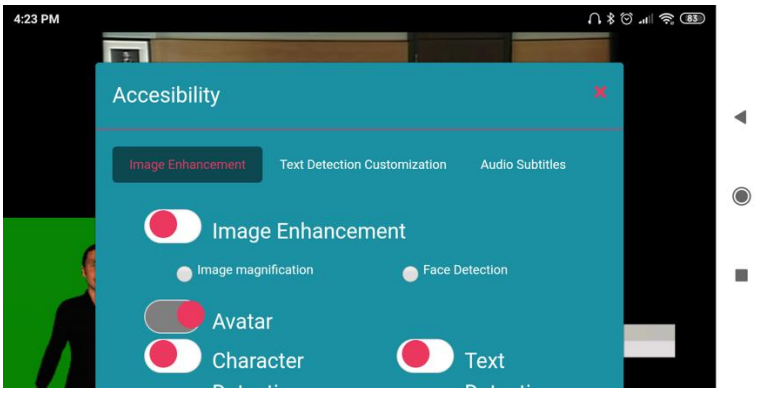


**Figure 5-4 CSApp menu**


**Figure 5-5 Avatar Service in CSAPP**

As described above all the integration with the crowdsourcing platform occurs directly and with simple http requests. Therefore, there is no need to further integration with CSApp, despite rules that allows the communication. Though, CSApp is responsible for extracting the accurate subtitle file from DASH stream and provide it to the avatar service. Each DASH stream has metadata information and subtitles in different languages is one of them. If the avatar service is enabled CSApp can find the specific link in the requested language and provide it to the avatar service. The link obtains the subtitles in specific form and always timestamped. The procedure after receiving the subtitle was described above.

# 6. INDICATIVE USE CASE: WEATHER FORECAST SCENARIO

This chapter describes the use case that was used in the final development of the service. In order to try out the system in real terms a thematology had to be decided regarding the videos that will be used. It would seem unlikely to capture and store such an amount of signs that they could respond in any video. Therefore, it was decided to create a set of words and phrases relative to weather. Temperature, clouds, rain are some of the words that were captured as seen in Figure 6-1

**Figure 6-1 Signer in final recordings**

Figure 6-2 shows the signer, the motion file that recorded through the Capturing module and the final result on the avatar.

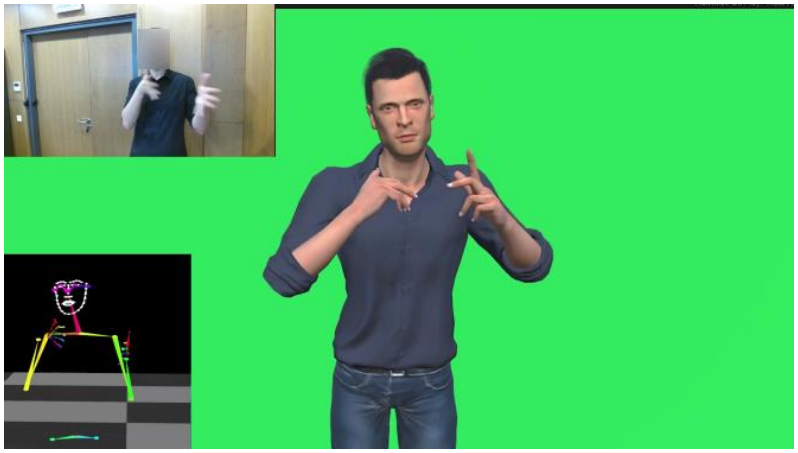

**Figure 6-2 Recordings and Avatar**

Weather forecasts were chosen, and subtitles were added with the form that has been used during development. Figure 6-3 shows a frame from one of the weather forecasts[5] was chosen with subtitles enabled.


**Figure 6-3 Weather forecast with subtitles**

When the user calls a specific weather forecast and enables the Avatar Service, CSApp provides the subtitle file to the Service. The Avatar Service is requesting for the words in specific timestamp. If the words are available avatar reproduces the sign. Otherwise, receiving a 404 respond and remains in idle state waiting for the next available Word to appear. Using the last figure as example, the words temperature and high are stored in the platform. So, when these words appear on subtitles, the avatar, which already requested the words of the subtitle file, reproduces the signs from the stored motion file. Figure 6-4 shows how the Avatar Service appears on the CS App. HbbTV has the same workflow. For the purpose of the video Avatar window was positioned on the up-left side of the screen.


**Figure 6-4 CSApp with Avatar Service**

Time handle is also an issue for sign language. Weather forecast videos are usually short and with fast speech from the presenter. Therefore, Avatar Service in the request loads a set of the upcoming words ready to be reproduced. Furthermore, the method that was used during the deserialization is close enough to a TV sign language forecast. Due to the restricted time, signer has to cut off a lot of words but the viewer with hearing disabilities can understand the weather forecast.

# 7. CONCLUSIONS

This deliverable presented the final version of the EasyTV signer avatar, a humanoid avatar that reproduces signs which have been recorded from a human signer. The need of a more realistic humanoid avatar as possible pushed the development to AAA graphics along with a balanced size of the file. Face development has also extra care cause to the significance to the comprehensiveness of sign language. The visual details were emphasized for a clearer view to the user. The environment and especially the lighting performance were tested accordingly with avatar's form and texture for better results. The final avatar has been integrated into the CSApp and HbbTV in Android Gradle and WebGL format respectively. Regarding the workflow, all the signs are captured and stored in the Crowdsourcing platform. The CSApp provides the subtitle link of the corresponding video to the Avatar Service and in its turn requests the words that need to be reproduced from the Crowdsourcing platform.

For future connections with other services inside the EasyTV platform, the usage of a similar to the subtitle file is necessary for the communication with the data repository. Therefore, the speech-to-text Service can be a part of the workflow. The text will have the role of the subtitles file in the Avatar Service. Using the same method a further interaction with the user can be established as the avatar can perform a basic Q&A use case. The main issue to the above use cases is the correct use of time and the synchronization between words and movement.

[1] “Unity 3D.” [Online]. Available: https://unity.com/.

[2] “Light Probes.” [Online]. Available: https://docs.unity3d.com/Manual/LightProbes.html.

[3] “LOD.” [Online]. Available: https://docs.unity3d.com/Manual/LevelOfDetail.html.

[4] “Autodesk MotionBuilder.” .

[5] World Meteorological Organization, “Weather Forecast,” *CC-BY 4.0*, 2014. [Online]. Available: https://www.youtube.com/watch?v=G_JmdMINI_Q. [Accessed: 31-Jan-2020].