Using Audio-Visual Speech Recognition Techniques in Augmented Reality Environment

Mohammad Reza Mirzaei, Seyed Ghorshi, and Mohammad Mortazavi
School of Science and Engineering
Sharif University of Technology
International Campus, Kish Island, IRAN
mrmirzaei@kish.sharif.edu, {ghorshi, mortazavi}@sharif.edu

Abstract—**Recently, many researches show that Augmented Reality (AR) and Automatic Speech Recognition (ASR) technologies can help people with disabilities. Today, we have seen great advances in AR and ASR technologies that were led to get much better results in each field. Audio-Visual Speech Recognition (AVSR) is one of the advances in ASR technology that combining audio, video and facial expressions to capture narrator's voice. This paper examines the ability of combining AR and AVSR technologies to implement a new system for helping deaf people. Our proposed system can take narrator's speech and converts it into readable text and show it on AR environment display, so deaf person can easily read the narrator's speech. The evaluation results show that this system has lower word error rate compared to ASR and VSR in different noisy condition. It also shows its reliability and usability to helping deaf people in different places.**

Keywords-Augmented Reality; Audio-Visual Speech Recognition; Augmented Reality Environment; Communication; Deaf People;

I. Introduction

Today, using new technologies for helping people with disabilities is highly regarded and much research in this area is underway. The important issue is how to combine and use these technologies together to make them more applied and get the best results from them. One technology that has a lot of interest recently and various researches have been carried out is Augmented Reality (AR). AR is a low-cost technology that has 3D computer graphics and one special feature known as marker [1]. A number of recent studies have shown that Virtual Reality (VR) and AR can be used to help people with disabilities [2], [3]. Another technology that is able to help disabled people is Automatic Speech Recognition (ASR). ASR is the ability of a computer to interpret human speech and translate it into text or commands [4]. Noise is the most important issues in speech recognition systems, especially when using them in outdoors [5]. Recent advances in ASR technology had led to combine audio, video and facial expressions to capture the voice of narrator [6], [7]. This technique is called Audio-Visual Speech Recognition (AVSR). The results of using AVSR in noisy places show that it can improve the recognition accuracy in ASR systems [8] and also have shown that making a real-time AVSR system is now possible [9].

Communication with people is a basic need for everyone but some people are not able to communicate well with other people due to some disabilities. One group of these disabled people is deaf people. Communication with normal people is the main problem among deaf people. Usually deaf people learn and use sign-language well to communicate with each other, but normal people have no desire to learn it. By this reason, many normal people feel awkward or become frustrated trying to communicate with deaf people, especially when no interpreter is available. As we know, deaf people cannot hear or recognize sounds very bad, but most of them are able to see very well. We take advantage from this ability of deaf people to make a new system for helping them. Our proposed system uses the audio, video and facial expressions of narrator to make the narrator's speech visible for them with the help of AR and AVSR technologies. With our proposed system, deaf people can see what narrator says and also normal people do not need to learn sign-language to communicate with deaf people. So, our proposed system solved the main problems of deaf people which is communication with normal people and vice versa. In this paper, we proposed a new system that combines AR and AVSR technologies. This system is a new system with multiple features and can be used as a novel system for helping deaf people, but one of the main goals of it, is helping deaf people to easily communicate with normal people.

The rest of this paper is organized as follows: In Section 2 we explain our proposed system and a brief discussion about it. In Section 3 the structure of our proposed system is presented. Section 4 describes our system design and its hardware, software and system requirements. In Section 5 the experimental results of our proposed system are presented, and finally in Section 6 we summarize and conclude the paper.

II. Proposed System

Our proposed System includes a variety of technologies. It consists of the AR engine, the AVSR engine, the Joiner Algorithm, Audio-Visual contents, Automated Process Scripts, and Face Detection techniques. This system uses the combination of these technologies to make speech visible for deaf people. Figure 1 shows the overall view of our proposed system with an Ultra Mobile PC (UMPC). Of course deaf people can use AR Head Mounted Display (HMD) or mobile phones to see the AR environment.

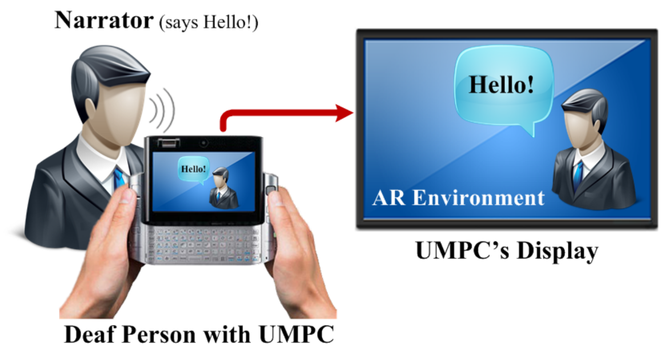


Figure 1. Overall view of our proposed system.

In our system's scenario, the AVSR engine collects the speech and facial expressions from the detected narrator and the AR engine realizes the scenario. The Joiner Algorithm has task of combining AR and AVSR engines to work together. For achieving some goals we need to use some automated process scripts that can be integrated to the system in the future. Our system uses the AVSR engine to recognize speech of narrator and convert it to text, and uses the AR engine to display this text as a dynamic object in AR environment. This system uses the built-in camera and microphone on UMPC, or AR-HMD or mobile phones for getting video and speech of narrator, and uses their displays for showing the objects in AR environment. The important features in our system are that for detecting narrator in the environment we proposed and used face detection techniques instead of marker detection, and also we used AVSR techniques instead of ASR techniques. With these methods, deaf people can use our proposed system in any noisy environment without need to carry marker.

III. System Structure

In this section, we briefly explain the structure of our proposed system and the components and technologies that we used in this system. Figure 2 shows the structure of our proposed system.

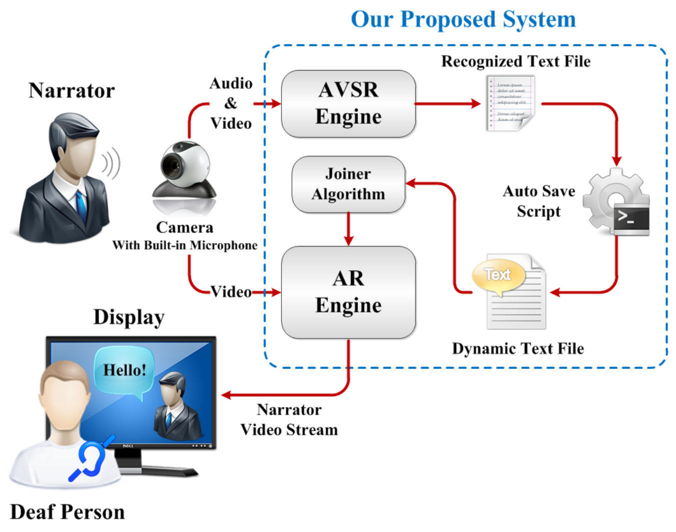


Figure 2. Our proposed system structure.

Our proposed system consists of two main parts: Hardware part and Software part. In hardware part some hardware requirements like camera, microphone and display are required and in software part we developed the core of our system that consists of the AR and AVSR engines, the Joiner Algorithm and the Auto Save Script. All these parts can be considered in an integrated system.

The deaf person focuses the UMPC's camera on the narrator. This camera can be a web camera that connected to or embedded in a computer or mobile phone, or mounted on AR-HMD. The camera captures the video and the built-in microphone, or external microphone, captures the voice (speech) of narrator. These two parameters are considered as key parameters in our system, because both of them needed to process separately.

When video frames are given from the camera, the AR engine identifies the face of narrator and uses it as a marker. Simultaneously, the AVSR engine captures audio and video of detected narrator and converts him/ her speech to a recognized text file. But this output text file is not completely dynamic and cannot be used in the Joiner Algorithm, because it changes frequently in time without being saved when narrator says some words. For making it completely dynamic and usable for the Joiner Algorithm, we proposed and wrote an Auto Save Script to save this recognized text file immediately and gives us the text file that we called it "Dynamic Text File". It uses as our system's text strings database for the Joiner Algorithm. Therefore, every time narrator says something, the words save immediately in recognized text file and our system always has the updated version of this recognized text file. To make the combination of AR and AVSR engines, we proposed and wrote the Joiner Algorithm. The Joiner Algorithm loaded dynamic text file strings on a speech bubble image for use it in the AR engine. The AR engine matches the Joiner Algorithm output to its face position information and augments it on narrator video frames in AR environment display. Finally, deaf person can easily see and read the narrator's speech in AR environment display near the face of narrator.

IV. System Design

Our proposed system designed to be a portable system to use easily by deaf people. We developed the AR engine with Adobe Flash Platform in Adobe Flex Integrated Development Environment (IDE) that is called "Adobe Flash Builder". Adobe Flex is an Open-Source Software Development Kit (SDK) released by Adobe Systems for development and deployment cross-platform applications based on Adobe Flash Platform [10]. So, our system is a cross-platform system and can be easily adapts and uses in many portable devices.

Our proposed system can be developed with an internal AVSR engine; also it can associate with an external AVSR engine. For initial testing and for showing the power of it, we used an Open-source AVSR engine called "Audio-Visual Continuous Speech Recognition (AVCSR)" by Intel research group [11]. This AVSR engine can get the narrator's video file as input and analyze it to show the results by using ASR, VSR and AVSR engines. So we can easily compare the results of our proposed system in different scenarios.

A. Hardware Requirements

For running our system in basic state, we do not need very powerful hardware. It can be running as a very lightweight application. Generally for running our system we need some specific hardware requirements that are shown in Table 1.

TABLE I. HARDWARE REQUIREMENTS

Our proposed system hardware requirements	
Processor	Intel Mobile or Core Due 1.0 GHz
Memory	1 GB
Camera	640*480 VGA sensor or more, compatible with computers that have either USB 2.0 or USB 1.1, or Built-in cameras.
Microphone	All types of external microphone, or Built-in microphones.

In our system, the AVSR engine can be more powerful and may need some specific hardware in the future. So, our system's hardware requirements depend on its AVSR engine. We can consider that all hardware requirements are in an integrated system.

For initial testing, using Intel AVCSR application in Microsoft Windows XP Operating System (OS), we used an UMPC with specified hardware like Intel Dual Core 1.8 GHz processor, 4 GB memory, and 5 mega pixels High-Definition (HD) sensor external web camera with built-in noise cancelling microphone. This system works fine with this particular hardware and gives very good results in Windows XP OS. Also, the external AVSR engine runs without any problems in our tested UMPC by specified hardware.

B. Software Design

Our system had been divided into three main software parts: The AR engine, the AVSR engine, and the Joiner Algorithm. For each of these parts, a group of libraries and toolkits for face detection, audio and video recognition, tracking, text object loading and rendering have been used.

1) The AVSR Engine

For getting different results and due to experimental work, we used a powerful external AVSR engine. The Intel AVCSR application package was found to be the best choice. It is a package that developed by Intel research group for testing and evaluation of audio and visual speech recognition and combination of these as an AVSR system [11]. It uses a set of audio and visual features to increase the accuracy of speech recognition in noisy environments. The Intel AVCSR application uses OpenCV (Open Source Computer Vision by Intel) and contains a set of Visual C++ functions for visual feature extraction as well as binary code for audio feature extraction and the fusion model [11]. Generally, the Intel AVSR engine first detects and tracks the face and mouth of the narrator in consecutive frames. Then, it extracts a set of visual features from the mouth region. Also, it obtains the acoustic features from the audio channel that consists of Mel Frequency Cepstral Coefficients (MFCC) [12]. Finally it models jointly the audio and visual observation sequences by using a Coupled Hidden Markov Model (CHMM) [12], Figure 3.

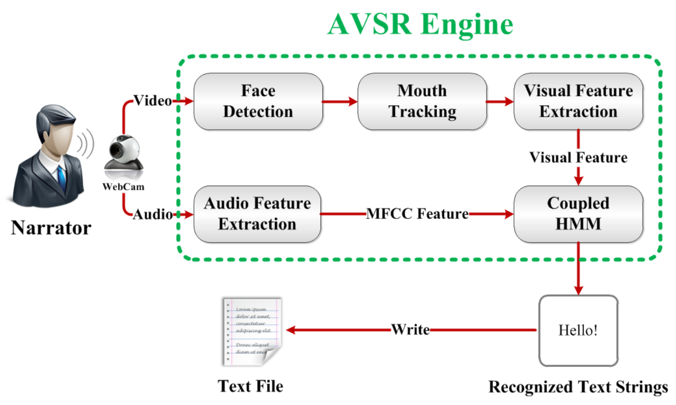


Figure 3. Block diagram of the AVSR engine.

The Intel AVCSR application is an offline system in which the recorded audio-visual data are loaded and processed from Audio Video Interleave (AVI) files. In Intel AVCSR application, besides AVCSR decoder, two more recognition engines, the audio-only decoder (the audio-only speech recognition engine, ASR) and the visual-only decoder (lip-reading engine, VSR) are integrated into the system [11]. We can use the test data in an appropriate format to compare the performance and the result of the different engines in our proposed system. The task of AVSR engine is processing the input audio and video, and also facial expressions of narrator to converts them into a recognized text strings. In general, most AVSR engines use the method for speech recognition that is shown in Figure 3 and the result is recognized as text strings that are written in a text file. Our system uses the dynamic version of this output recognized text file as an input to the Joiner Algorithm.

As it is said, the Intel AVCSR application is an offline system and cannot work in real-time, so we need to record and open the narrator's .AVI file to process by AVCSR; also we cannot normally use its results in our system. For making the results usable in our system, we need to do some changes in the Intel AVCSR application source codes. Since the Intel AVCSR is an Open-source package, we could easily change some codes to retrieve the recognized strings and write them respectively into a .TXT file as our system's text database. Of course by using a real-time AVSR engine in the future [9] we do not need to do any changes. For initial testing we used a .TXT file to save recognized text strings. Also, our system can be developed with internal temporary text array to save recognized text strings. Every time the .AVI file is opened, the Intel AVCSR processes it and analyses recognized words in its engines. Then wrote these strings from its engines on the text file separately, but it does not save in this file. The Joiner Algorithm always needs the updated version of this text file, which means every word that the Intel AVCSR recognizes as a text string must be saves immediately in the text file. For doing this, we proposed and wrote an Auto Save Script to save the text file immediately and give us the updated version of it. We called this "Dynamic Text File". This text file is used as the input to the Joiner Algorithm which is shown in Figure 4. To avoid increasing the size of the text file with useless text strings, we programmed our system to completely clean this text file every time it runs by user.

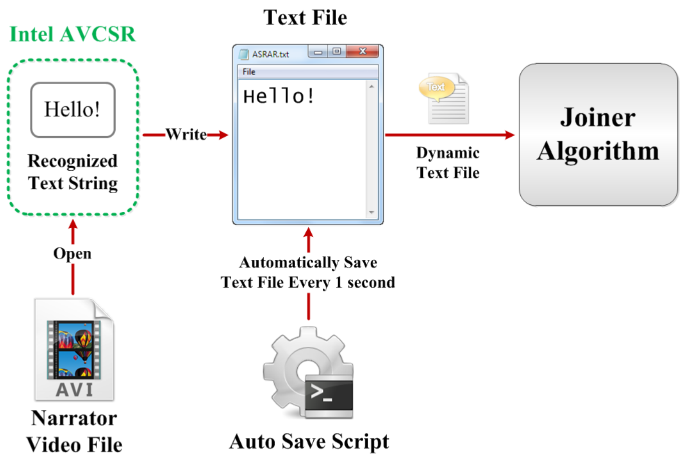


Figure 4. The Intel AVCSR and Auto Save Script working process.

The Auto Save Script is a small computer script or application to do the saving operation automatically in specified times. For initial testing in Microsoft Windows XP OS, this script is written by VBScript (Visual Basic Scripting Edition) programming language that can be compiled and used easily in Windows OS. It can also be developed by other programming languages or as internal coding in our system. This script does auto saving the text file every one second. Thus, the text file saves immediately when the AVSR engine recognize the word and wrote it on the text file. So, we always have the updated version of this text file, as dynamic text file, for using it in the Joiner Algorithm.

2) *The Joiner Algorithm*

For making relations between the AR and AVSR engines we proposed and wrote a module called "Joiner Algorithm". With this module the AR engine can get and use the output of the AVSR engine. So this module enables us to combine AR and AVSR engines to work together. The Joiner Algorithm is a Flash application with Action Script version 3.0 (AS3) coding [13] for loading dynamic text file strings on speech bubble image to use it on the AR engine, Figure 5.

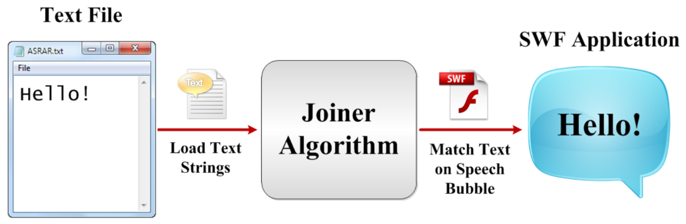


Figure 5. The Joiner Algorithm working process.

For the initial phase of testing our system, we've developed the Joiner Algorithm to load and show only the last word in the text file. This development is optional and developer can easily change it to show more words. With this development, the Joiner Algorithm is always looking up the last word in the text file. When it found a new word, loads and shows it on speech bubble image. The output of the Joiner Algorithm is a .SWF application that we can easily use it to show in AR environment by using AR toolkits.

3) *The AR Engine*

In the AR engine, FLARToolkit, FLARManager, OpenCV and PaperVision3D with AS3 are used. The purpose of using these toolkits is for their availability to export our system as .SWF, hence allowing the availability of making it as a cross-platform application. Also these libraries allow us to show the output of the Joiner Algorithm, that is a .SWF file, in AR environment. The FLARToolKit is AS3 ported version of the ARToolKit. The ARToolKit is a computer vision tracking library that allows for the creation of AR applications that overlay virtual imagery on the real world [14]. The ARToolKit was originally developed by Hirokazu Kato of Nara Institute of Science and Technology in 1999 and was released by the University of Washington HIT Lab [15], and the FLARToolKit was developed by Spark Project team (Libspark) [16]. Libspark had a huge impact on flash advancements. They are porting existing solid ARToolkit libraries from C/ C++ into flash and released the FLARToolkit [16]. FLARManager is a lightweight framework that makes it easier to build AR applications for Flash and compatible with a variety of tracking libraries and 3D frameworks [17].

As mentioned, our system uses face detection instead of marker detection. Most AR applications are used marker to detect and identify objects. Marker is a square arrays system that added to objects or environment and allowing a computer vision algorithm to calculate the camera pose in real time and shows the 3D objects on environment [18]. If marker is used, we must design and create specific marker in our system and the users must be always carrying the marker. So we decided to develop our system as a marker less AR application. In this case, we are preventing to do some marker detection challenges that make our system more complex. Face detection is the best choice for our system because the only thing that we need is to detect the face of narrator as a marker. We do not need to use face databases for portrait functions in face recognition technology. So, this makes our system very lightweight. The technology that we used for face detection is OpenCV. OpenCV is a library of programming functions for real-time computer vision and is a powerful platform [19]. OpenCV has many application areas that include facial recognition system too. OpenCV face detection uses a type of face detector called a Haar Cascade Classifier. Haar-like features are digital image features that are used in object recognition [19]. Our system uses AS3 version of OpenCV for facial detection, webcam object detection, head tracking, 3D library integration, etc. The face detection part of OpenCV ported to AS3 language by Ohtsuka Masakazu in Spark Project team (Libspark) [20]. We found a ported version of OpenCV for AS3 called "Marilena Object Detection". Marilena is AS3 library classes for object detection and decent facial recognition [20]. It has the best performance among other AS3 libraries for doing face detection. We are joining the Marilena library to FLARToolkit library with Adobe Flash Builder IDE to develop the AR engine. Using this, helps the AR engine uses face detection instead of marker detection in our system. We used the combination of these toolkits and libraries, near using face detection techniques, to make and develop the AR engine in Adobe Flash Builder IDE. The block diagram of the AR engine is shown in Figure 6.

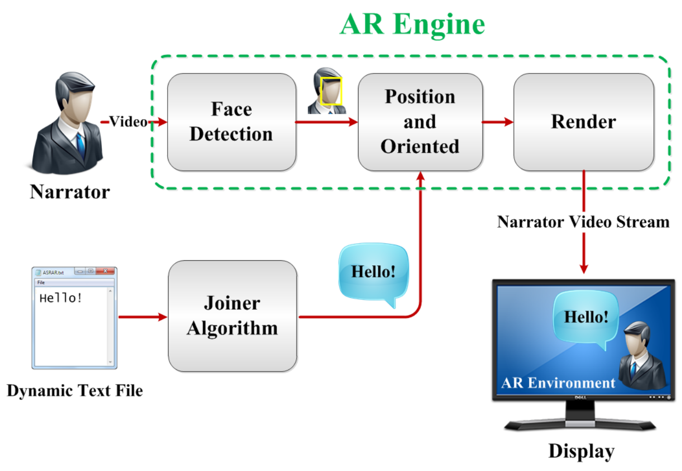


Figure 6. Block diagram of the AR engine.

In the AR engine, after capturing the video frames from the narrator by camera, the face detection module identifies the face of current narrator in the environment and sends this information to Position and Oriented module. For initial testing, we've developed the face detection module to detect only the face of one person in the environment and use it as the face of narrator. In this method, our system knows where the narrator's face is located in the video frames. Simultaneously, the Joiner Algorithm loads the last string in dynamic text file and shows it on speech bubble image and sends its output, which is a .SWF file, to Position and Oriented module. Then the Position and Oriented module matches the speech bubble .SWF file to information from the face detection module, which is the position of narrator's face, and sends them to Render module. Finally, Render module augments all of this information on narrator's video frames in AR environment display, near the face of narrator. With this process, narrator's speech becomes visible in display and deaf person can easily see what narrator says, Figure 7. This will be very exciting for deaf person and exactly is the goal that we were seeking to achieve it by our proposed system.

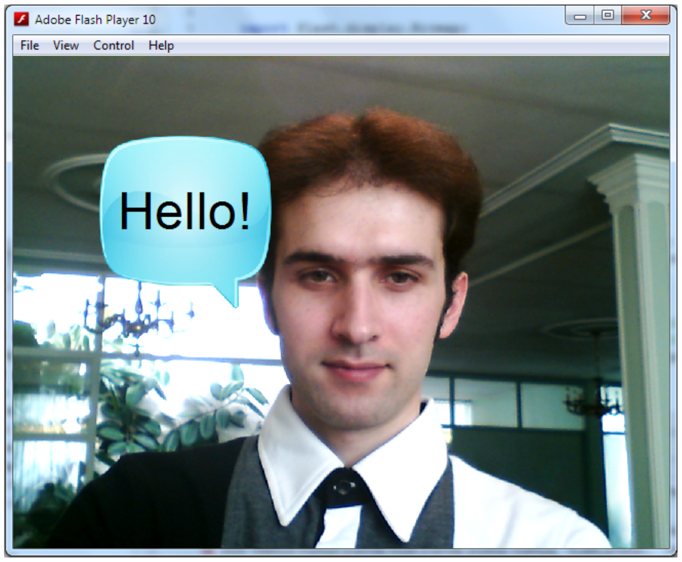


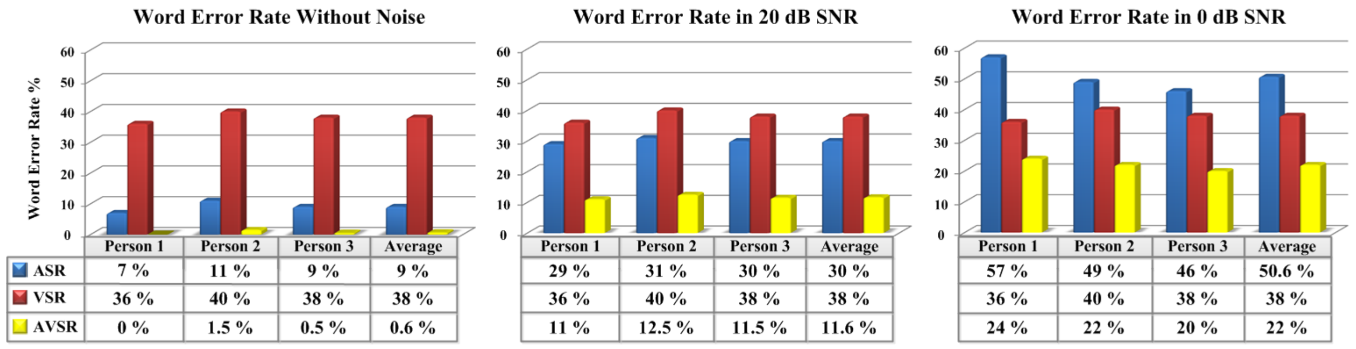
Figure 7. The AR environment in our proposed system.

V. Evaluation Results

To evaluate our proposed system, we tested it in three different noisy environments with three different persons, by the Intel AVCSR application. The Intel AVCSR allows us to add noise to the input video file concurrently in various acoustic SNR. Our video recording files are completely without noise, so we can review and analyze the various conditions imposed by external noise in our system. Also we recorded these video files from three different persons with different facial expressions that let us to test the AVSR engine in different facial conditions. The facial and environmental conditions are very important in these tests and will lead to changes in test results. The time between saying a word and showing it in AR environment directly depends on the power of AVSR engine. The Intel AVCSR engine needs approximately 10 seconds to process video file and get results, but if we use a real-time AVSR engine [9] in the future, this time decreases to less than 2 or 1 seconds. For getting the better results, we used an UMPC with powerful hardware specifications that described in hardware requirements section and for recording testing video files we used a powerful digital video recorder near using a powerful external microphone with noise cancelling feature. We also assumed that the distance between narrator and video recorder is only 1 meter and narrator speaks slowly with the best accent. It was very important for us to know whether or not our system can work in different environments in terms of its performance.

A. *Classification of Tests*

We classified our tests according to different conditions that a deaf person may be in. These tests are classified to: Without Noise, 20 dB SNR, and 0 dB SNR. Each of these conditions simulates a particular noisy environment for our system. We collect and review the results of Word Error Rate in each test and observed that our proposed system can work successfully and performs good results in all tests. The following will explain the test results and how our proposed system will work in different noisy condition. After opening and processing video file with the Intel AVCSR application, it wrote the output results of each engine separately in the text file that will be used for testing our proposed system. These tests have been done on all three persons and the results have been shown in Figure 8. It can be seen form Figure 8 that in different noisy condition (i.e. without noise, 20 dB SNR, and 0 dB SNR) the AVSR system has lower word error rate compared to ASR and VSR. Therefore, our system can work fine on average in different noisy environments by using Intel AVCSR application. All words that the Intel AVCSR application wrote in the text file, after processing video file that take about 10 seconds, were shown immediately and correctly in AR environment, near the face of narrator, without any delay in the system. Testing our system with this AVSR engine has some different results that depend on many parameters like hardware, noises, narrator accent, facial expressions, etc. We compared the average word error rate of each test in the Intel AVCSR application in equal conditions, and the results have shown in Figure 9. The results show that our system (AVSR System) and the Intel AVCSR application still has lower word error rate in 0dB, 10 dB, 20dB, 30dB SNR and without noise compared to ASR and VSR.



Word Error Rate Without Noise

	Person 1	Person 2	Person 3	Average
🟦 ASR	7 %	11 %	9 %	9 %
🟥 VSR	36 %	40 %	38 %	38 %
🟨 AVSR	0 %	1.5 %	0.5 %	0.6 %

Word Error Rate in 20 dB SNR

	Person 1	Person 2	Person 3	Average
🟦 ASR	29 %	31 %	30 %	30 %
🟥 VSR	36 %	40 %	38 %	38 %
🟨 AVSR	11 %	12.5 %	11.5 %	11.6 %

Word Error Rate in 0 dB SNR

	Person 1	Person 2	Person 3	Average
🟦 ASR	57 %	49 %	46 %	50.6 %
🟥 VSR	36 %	40 %	38 %	38 %
🟨 AVSR	24 %	22 %	20 %	22 %

Figure 8. Word error rate of three persons with average, under various acoustic SNR.

As it can be seen from Figure 9, with noise reduction feature the percentage of word error rate decreases. Therefore, our system only depends on how much the AVSR engine is powerful and advances in this area have a direct impact on the system and makes our system more powerful in the future. It also shows that our system can be very useful for helping deaf people in different noisy places.

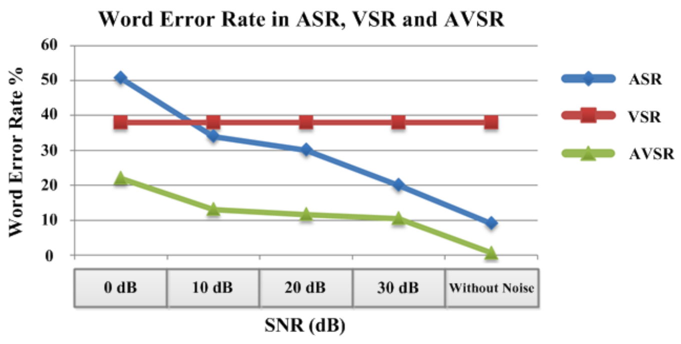


Figure 9. Word Error Rate in the Intel AVCSR engines.

VI. Conclusion

This paper is aimed to design and develop a new system to help deaf people by combining AR and AVSR technologies. We do this by finding a common factor between AR and AVSR technologies that is the text string. Our proposed system can makes speech visible for deaf people in AR environment display. We have shown that we can reach new approach by combination of existing technologies. Also we show that our proposed system solved the main problems of deaf people which is communication with normal people and vice versa. The results of testing our system showed that this system can act very well in many situations that the deaf person might be. These results also showed that our system is consistent and will provide acceptable results with new real-time AVSR engines in the future. The improvements and advancements in AR and AVSR technologies bring bright future for our proposed system. The only limitation of our system is that it depends on the power of AVSR engines that could easily solve with AVSR engines improvement that has been seen in recent years. Hopefully, our system can be an alternative tool for deaf people in acquiring visual literacy and appropriate communication process skills and cause to normal people do not feel awkward or become frustrated to communicate with deaf people.

References

[1] W.R. Sherman and A.B. Craig, *Understanding Virtual Reality*, Morgan Kaufmann Publisher, 2003.

[2] N.M. Zainuddin, H.B. Zaman, and A. Ahmad, "Developing Augmented Reality Book for Deaf in Science: The Determining Factor," *Int. Symposium in Information Technology*, IEEE, 2010, pp. 285-288.

[3] N.M. Zainuddin and H.B. Zaman, "Augmented Reality in Science Education for Deaf Students: Preliminary Analysis," presented at Regional Conf. on Special Needs Education, Faculty of Education, Malaya Univ., 2009.

[4] A. Kalra, S. Singh, and S. Singh, "Speech Recognition," *The International Journal of Computer Science and Network Security*, vol. 10, no. 6, pp. 216-221, Jun. 2010.

[5] G. Bailly, E. Vatikiotis, and P. Perrier, *Issues in Visual and Audio-Visual Speech Processing*, MIT Press, 2004.

[6] I. Lipovic, *Speech and Language Technologies*, InTech, 2011.

[7] S.W. Chin, L.M. Ang, and K.P. Seng, "Lips Detection for Audio-Visual Speech Recognition System," presented at Int. Symposium on Intelligent Signal Processing and Communication Systems, Thailand, 2008.

[8] R. Navarathna, P. Lucey, D. Dean, C. Fookes, and S. Sridharan, "Lip Detection for Audio-Visual Speech Recognition In-Car Environment," in *Proc. 10th Int. Conf. on Information Science, Signal Processing and their Applications*, 2010, pp. 598-601.

[9] P. Shen, S. Tamura, and S. Hayamizu, "Evaluation of Real-time Audio-Visual Speech Recognition," presented at Int. Conf. on Audio-Visual Speech Processing, Japan, 2010.

[10] Adobe Systems, Inc., "Adobe Flash Builder," Jun. 2011; http://www.adobe.com/products/flash-builder.html.

[11] Open Computer Vision Library, "Open AVSR Alpha 1," May. 2011; http://sourceforge.net/projects/opencvlibrary/files/obsolete/.

[12] X.X. Liu, Y. Zhao, X. Pi, L. Liang, and A.V. Nefian, "Audio-Visual Continuous Speech Recognition Using A Coupled Hidden Markov Model," in *Proc. 7th Int. Conf. on Spoken Language Processing*, Denver, CO, Sept. 2002, pp. 213-126.

[13] R. Braunstein, M.H. Wright, and J.J. Noble, *ActionScript 3.0 Bible*, Wiley,2007.

[14] W. Hohl, *Interactive Environment with Open-Source Software*, Springer Vienna Architecture, 2008.

[15] ARToolworks, Inc., "ARToolKit," May. 2011; http://www.hitl.washington.edu/artoolkit/.

[16] Spark Project Team, "FLARToolKit," May. 2011; http://www.libspark.org/wiki/saqoosha/FLARToolKit/en.

[17] Transmote, "FLARManager: Augmented Reality in Flash," May. 2011; http://words.transmote.com/wp/flarmanager/.

[18] S. Cawood and M. Falia, *Augmented Reality: A Practical Guide*, Pragmatic Bookshelf, 2008.

[19] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*, O'Reilly Media, 2008.

[20] Spark Project Team, "Marilena Face Detection," Jun. 2011; http://www.libspark.org/wiki/mash/Marilena.