ISSN 2278-3091
Volume 9, No.1.1, 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse6491.12020.pdf
https://doi.org/10.30534/ijatcse/2020/6491.12020

JARI – Malaysian Sign Language Translator

Sofiah Sajap[1], Norhanifah Murli[2], Nurul Amirah Paham[3]
[1] Universiti Tun Hussein Onn Malaysia, sofiahsajap@gmail.com
[2]Universiti Tun Hussein Onn Malaysia, hanifah@uthm.edu.my
[3]Universiti Tun Hussein Onn Malaysia, amerpaham@gmail.com

ABSTRACT

This paper is a documentation of the development of an Android mobile application 'JARI' that is specifically designed and developed as Malaysian Sign Language (MSL) learning application. The application name, 'JARI' is an acronym that stands for 'belaJAR Isyarat' (Learn Sign). The development of this project is triggered by the lack of available mobile MSL learning application in the market today. The main objective of this development is to design an MSL learning application on Android platform that demonstrate the MSL gesture by using 3-Dimensional (3D) animation. The application allows user to enter their own input in the form of word or simple phrases by providing speech input or by just typing in normally. The input is then translated to an animated 3D MSL gesture. 'JARI' allows user to translate a sentence to MSL or spell a word by using fingerspelling method. 'JARI' is developed by adapting the Multimedia Mobile Content Devel-opment (MMCD) methodology, the animated MSL was modelled and animated by using MakeHuman and Blender, while the interface of the application is fully developed by using Android Studio. Android Speech Application Programming Interface (API) is used to enable the speech recognition function in the application. This application is foreseen to be beneficial among deaf communities in Malaysia whom intended to learn the basic of MSL conveniently and effectively.

Key words : Android Speech API; animation, JARI, Malaysian Sign Language.

1. INTRODUCTION

Malaysian Sign Language (MSL or Bahasa Isyarat Malaysia) is the principal language of the deaf community in Malaysia. It was born when the Malaysian Federation of the Deaf was established in 1998 and the users has expanded among deaf leaders and participants. It was based on American Sign Language (ASL), but the two are considered different languages. According to Ethnologue [1], the population of the deaf community in Malaysia is approximately 29,500 but the population of the signers of Malaysian Sign Language is 58,700 as of 2013 and the number is probably growing.
Speech Recognition is the inter-disciplinary sub-field of computational linguistics that develops methodologies and technologies that enables the recognition and translation of spoken language into text by computers. In this mobile application, the speech recognition technology is used with the help of Android Speech API to detect speech and will later translate it to a sign language gesture.

This mobile application is based on Android platform. It will enable user to translate simple Bahasa Malaysia words or phrases to MSL. It will also help beginners and non-signers who have interest in learning MSL to learn it in easy, interactive and interesting way. The main reference for the development of this project is 'Bahasa Isyarat Malaysia' which was published by the Malaysian Federation of the Deaf. The book is the first and official compilations of all the signs used by the deaf community in Malaysia.

The rest of the paper is organized as follows: Section II describes the related works that helps in the development of this application. Section III will discuss in detail about MMCD, which is the methodology used for this application development. Section IV presents the system analysis and design and Section V discusses on the implementation and testing of the application. Finally, section VI discusses and concludes the development advantages, limitation and future works for this project.

2. LITERATURE REVIEW

This section will discuss related concepts and technique related to this project. First, an insight about the history of MSL will be discussed. After that, the elements of MSL which consists of Sign Language and Finger Spelling will be elaborated. The existing MSL learning resources will also be reviewed alongside with the technology that can be used to assist in learning MSL. The last component of this this chapter is the review of existing sign language learning application.

2.1. MSL in Brief and Its Elements

Long before MSL was first introduced in Malaysia, deaf community in Malaysia already use Penang Sign Language (PSL) to communicate with each other. PSL began surfacing as an informal method for students in Primary School for the Deaf in Penang to communicate with each other [2]. When the school was first established in 1982, its main focus was on oralism (i.e.: lip-reading) and thus signing was prohibited from being used in their formal education. But the student still

communicates with each other discreetly by making small signs near down their hips. As a result, PSL was born. PSL was popular among deaf student in Penang and soon began spreading among deaf community in Malaysia. It was used as a primary language among deaf community in Malaysia before American Sign Language (ASL) was introduced.

According to [2], ASL was introduced in Malaysia by a former pharmacist whose name is Mr. Tan Yap that has dedicated his life to improve the deaf community living condition in Malaysia. MSL was developed with a great influence from ASL [2]. It was based on Bahasa Malaysia as this is the language that is widely spoken among Malaysian. MSL however does not follow any structured grammar system existed and it has its own set of grammar system that was derived from daily observation of the deaf community. Nowadays, MSL is still the most widely used sign language in Malaysia and it is very important as a communication and interpretation tool among the deaf community in Malaysia.

MSL technique consist of sign language and finger spelling. Although technically, finger spelling is capable of representing every word and is also easier and faster to learn, but to use finger spelling alone in conversing in sign language is rather impractical and tiring in daily life. It is almost like spelling every word in a sentence instead of just speaking normally. The definition of sign language and finger spelling is further elaborated below.

2.1.1. Sign Language

Sign Language according to Merriam-Webster [3] are a formal language employing a system of hand gestures for communication (as by the deaf) and an unsystematic method of communicating chiefly by manual gestures used by people speaking different language.

In short, sign language can be defined as a system that involves the movement of hand and can be used to transmit information from one person to another. In MSL, words are formed by using a hand gesture or a combination of it. The hand gesture sometimes is influenced by the observation of deaf people to their surroundings. The term 'signers' is used for people who are able to communicate using sign language.

2.1.2. Finger Spelling

Finger spelling is a method used in MSL that is used to spells words [4]. It is often used to spell out name and places, or words that cannot be represented by hand gesture. It also can be used to spell out words which signers are unsure of the sign. Basically, each letter is represented by a 'hand pose' and the combination of this 'hand pose' is used to spell a word. In MSL, there are 26 'hand pose' that represent each alphabet and 19 'hand pose' that represent numbers.

2.2. MSL Learning Resources

As an initiative to promote MSL to Malaysian, many non-government organization (NGO) such as Malaysia Federation of the Deaf (MFD), the Lions Club and Young Men's Christian Association Kuala Lumpur (YMCA) occasionally conducted class to non-signers interested in learning MSL. The class was instructed by professional translator known as '*Jurubahasa Isyarat'* (JBI) at specific location, usually around Lembah Klang, Kuala Lumpur. Every interested participant usually need to pay fees to the organizer to join the class.

The NGOs also have produced and printed several books over the years as a learning reference of MSL. The books are usually rich with illustrations to depict the sign language. These books can be bought through the association itself and it usually cost around RM5.00 to RM50.00 each depending on the learning level and scope. By using these books, non-signers can learn MSL flexibly with low cost. But learning MSL by books are usually difficult as MSL involves hand motions and signs illustrated in the books cannot truly depicts the sequence of the hand motions [5].

2.3. Technology to Improve MSL Learning Method

As mentioned in previous section, to learn MSL in mobile way by using books is actually difficult. But as technology revolves, there are various method and improvements available that can be implemented in MSL learning method. For example, before smartphone exist, there are already several projects that have been initiated by using the electronic media such as through video compact disc to assist in the teaching learning of MSL [5].

But when Apple iPhone first introduced with its own operation system (OS), iOS in 2007, it has dramatically revolutionized the way electronic media works [6]. Smartphone is considered as a fantastic device as it is able to handle most task from making calls and keeping notes to gaming and learning. The debut of Android OS by Google in 2007 have also open the door to technology companies to produce more innovative and affordable smartphone device [6]. Today, almost every person in the world owns at least one smartphone that is heavily involved in their daily routine.

2.2.1. Android OS

Android is a widely adopted open-source project and Linux-based mobile phone operating system developed by Google that powers phones, watches and even car stereos [7]. The first version of Android which was Android 1.0 was introduced on September 2008 and it only includes very little but useful application such as Gmail and YouTube [8]. The development of Android is very rapid and vicious and over the last 10 years since it was first developed, Google has produced a total of 16 versions of Android OS. The latest Android version is Android 8.0 Oreo which was publicly released on 21st August 2017.

Android 2.3 Gingerbread which was released on December 6, 2010 was the first Android version to have received a better audio, graphical and input enhancements for game developer [9]. This upgradation allows Android devices to support application with 3D animation and graphics. Google later include OpenGL ES 3.0 support in Android 4.3 Jelly Bean

which was released on July 24, 2013 which supports a smoother and faster 3D graphic and animation [10].

Since Android 1.6 Donut, Google had included a built-in text-to-speech engine that able to receive speech input from user and convert it into text [11]. This engine is later upgraded to have a better dictation and voice integration on Android 4.0 Ice Cream Sandwich [12].

2.2.2. Android Speech API

Speech recognition is a technology that allow speech or spoken input into systems, which is somehow similar to other methods of input like typing, clicking or selecting. [13]. Speech recognition able to extract and identify words and phrases from spoken language and convert them to a machine-readable format. The Android Speech API provides recognition control, background services, intents, and support multiple languages used around the globe [14]. This API was first introduced in Android API Level 3. The Android Speech API can be integrated as a plus feature in Android applications to increase the applications' productivity and improve their work flow.

The classes that comes with the API are *SpeechRecognizer* and *RecognizerIntent*. In Android programming, *Intent* is used to pass messages between components, such as Activities, Services, Broadcast Receivers and Content Providers. In Speech Recognition API, the most important intent is *RecognizerIntent.ACTION_RECOGNIZE_SPEECH* which is used to define the request. To start the recognition process, the intent requires only one extra data source which is either *RecognizerIntent.EXTRA_LANGUAGE_MODEL*, which specify the recognize language to default locale (English) or *RecognizerIntent.EXTRA_LANGUAGE* which is used to set the recognize language other than default.

As for now, Android Speech API supports more than 100 languages including Bahasa Malaysia. Once the recognition results are returned, they are saved in the data bundle associated with *RecognizerInten.EXTRA_RESULTS* intent.

3. METHODOLOGY

JARI is an Android application that able to translate a text or phrases provided by user to a 3D animation of MSL. The input is acquired from user by three ways, which are from speech input, text input and word directory. The application run on devices that operates on Android 4.4 (Android KitKat/API Level 19) or later. The target users for this mobile application are teenagers and adults who is a non-signer or a beginner in sign language and interested in learning basic knowledge about MSL. The language used in this application is Bahasa Malaysia since the MSL is based on this language.

The application covers fingerspelling and words from 8 first chapters from the main reference book, 'Bahasa Isyarat Malaysia' [15], which was published by the Malaysian Federation of the Deaf. The 8 chapters are Greetings (Ucapan), Pronouns (Kata Ganti Nama), Family (Keluarga), Questions (Kata Tanya), Conjunctions (Kata Penghubung), Auxiliary Verb (Kata Bantu), covered in these chapters or to spell the word that are not covered by using fingerspelling method. The application will also provide a brief information about the Deaf community in Malaysia and about MSL as an additional knowledge.

In developing the application, MMCD [16] methodology is adapted. It is formulated to suit the needs of developing a mobile learning application. MMCD is based on the characteristics of an agile development model. This methodology is also designed to help the developer to speed up the development activities and to ensure that mobile learning application development is performed as planned. The methodology encompasses of five main components or stages which are the application idea creation stage, structure analysis stage, process design stage, main function development stage, and testing stage. The project workflow for this project in each stage in MMCD is summarised in Table 1. In each stage, all activities, deliverables and tools are identified and listed.

Table 1: Project Workflow

Stage	Activities	Deliverable	Tool
Application Idea Creation	Identify: • Problem statement • Objective • Scope • Software & Hardware requirement Application content	• Proposal Application idea creation checklist	Microsoft Word
Structure Analysis	Analyze: • Navigation • Object	• Application structure checklist	• Microsoft Word
Process design	Create: • Navigation structure • Flowchart • Storyboard Design: • Objects • 3D model • Interface	• Navigation structure • Flowchart • Storyboard • 3D Model • Application Interface	• Microsoft Word • Make Human • Blender • Android Studio
Main function development	• Coding and programming of application.	• Prototype	• Android Studio
Testing	Perform: • Alpha Testing • Beta Testing	• Output • Survey form	• Android Studio • HockeyApp • Google Forms • Microsoft Excel

Table 1 shows the project workflow of this project which was fractioned according to the stages involved in MMCD methodology. Project workflow is done to help in tracking the processes and tasks involved in a project to ensure that there are no repeatable process occurs during project execution.

MMCD methodology is summarized as in Figure 1. It starts with the process of generating the application idea and then move on to analyzing the structure. Structure analysis includes navigation and object analysis. The next stage is process design which developer will start designing the object in the application and write single function prototype before moving on to the stage of developing the main function which will produce an application prototype. The last stage of MMCD is the application prototype testing process.

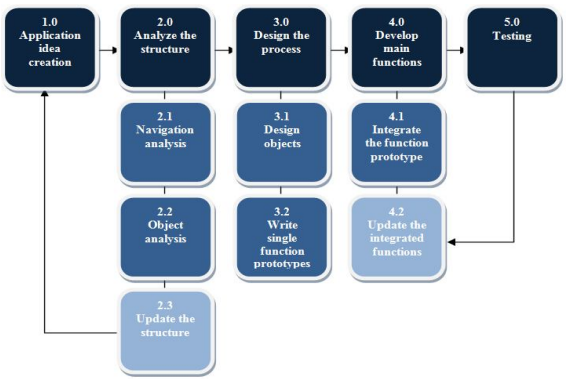

Figure 1: MMCD Methodology [16]

3.1 Application Idea Creation

This application is developed to be a mobile learning application that can helps in learning MSL. Although, there were many MSL learning application that exist in the market, but most of them are not interactive, does not able to effectively demonstrate the sign language using the right medium and have a poor user interface design. This application will allow user to provide their own input and demonstrate the sign language in 3D animation. User requirement as well as hardware and software requirement are recognized.

3.2User Requirement Analysis

User requirement analysis has been conducted for this project using interview method. The interview was done to acquire precise information on what modules should be included in the applications. The interviewee involved are a representative member from MFD who is also a professional MSL translator, Mrs. Hafidah Awaludin and deaf personnel who is also an illustrator and graphic artist who works for MFD, Mr. Zaine Bujal.

From the interview session, Mrs. Hafidah stated that there are two ways of communicating in MSL which are fingerspelling and hand gesture. When people came to learn MSL, they were usually taught MSL word by word and to communicate, they will later form a sentence by combining the MSL words. Mrs. Hafidah also stressed that the grammar structure for MSL is actually very different than those for Bahasa Malaysia or English. The main focus of MSL in creating a complete sentence is to deliver an information in the most straightforward and brief as possible but still able to convey the real message.

In the term of designing and modelling a 3D model that is appropriate for learning MSL, the input from Mr. Zaine was acquired. Mr. Zaine pointed out that the outfit for the 3D model should be designed to be short-sleeve so that the hand gesture can be clearly seen and observed by user. He also shared that the blue color is one of the most appropriate color theme to be used for application since the color could represents communication, calm, and logic – which is the principal component when creating a learning environment for MSL. Mr. Zaine also suggested to include the information about The Deaf Culture (Budaya Pekak) in the application as a way to attract and educate user on the deaf community in Malaysia.

3.3 Hardware and Software Requirement

Hardware and software requirement analysis in the early stage is crucial as it will reduce the potential of having to add additional hardware mid-development phase and thus will delay the development period and will probably increase the development cost. The basic hardware needs for this project are computer, a mouse and a mobile device for testing and debugging purpose. The software required to model and rig the 3D model that will be used in the application are Blender and MakeHuman. Meanwhile, to build and develop the application, Android Studio. The designing of model and object used in this application is done by using Adobe Photoshop CS6 and Adobe Illustrator CS6. HockeyApp is used to distribute the application for Beta Testing process.

3.4 Analyse the Structure

Structure analysis involves navigation analysis and object analysis. The purpose of conducting navigation analysis is to measure the usability of the application while object analysis is carried out to analyze the characteristic and traits of objects that are contained in the application

3.4.1 Navigation Analysis

Navigation analysis is conducted to inspect the complete navigation path in the application. Graphical Hierarchical Task Analysis (HTA) is used as a tool to complete the navigation analysis. The HTA analyse the application's task to translate a word or phrase to the 3D animation of MSL. There are 3 sub-tasks under the main task, which are accept input from user, choose category and list all words. These sub-tasks assist the application to accomplish its main task which is to translate a word or phrase 3D animated MSL.

3.5 Object Analysis

Object Analysis is done to identify the list of objects that is contained inside the application. The objects involved include buttons, image, icons, logo, and animation. The object analysis for this project is documented in Table 2.

Table 2: Object Analysis for Each Settings and Fragments

Settings/Fragment	Object/s
3D Animations	Consist of animation: • Idle • A – Z • 17 words in *Ucapan* • 10 words in *Kata Nama* • 10 words in *Keluarga* • 10 words in *Kata Tanya* • 7 words in *Kata Hubung* • 8 words in *Kata Bantu* • 13 words in *Perasaan* • 15 words in *Kata Kerja*
Splash Screen	• A 16:9 screen wallpaper that contains the application logo and name, JARI.
Introduction/App Tutorial Screen	• A short animation on how to use the app
Fragment *Laman Utama*	• 3D animation • Voice input button (Microphone button) • Text input area • Set animation speed button • Replay animation button
Fragment *Kategori*	• List of words in *Ucapan, Kata Nama, Keluarga, Kata Tanya, Kata Hubung, Kata Bantu, Perasaan* and *Kata Kerja* category
Fragment *Direktori*	• List of all words that contained in animation database, arranged alphabetically
Fragment *Budaya Pekak*	• Infographics about The Deaf Culture

3.6 Design the Process

Process design helps in identifying and designing the process of the application. To assist to the means, flowchart and storyboarding are used to give a visual depiction of how the application may looks like and works in real life. Figure 2(a)-(d) show the main storyboards of JARI in which Figure 2(a) shows the splash screen which appears for 5 seconds every time the application is launched; Figure 2(b) shows the navigation drawer or the main navigation menu in JARI. To open the drawer, user simply needs to swipe the screen from left, or to click the top left button; Figure 2(c) shows the JARI main screen. User can provide their input by typing in the keyword or by providing speech. User can also switch between 'Bina Ayat' (Build sentences) and 'Mengeja' (Fingerspelling) by swiping the screen to left and right; and finally Figure 2(d) shows the category screen that listed all the words covered in JARI according to the respective category. To switch between category, user needs to slide the screen to left or right. When user clicks on a word, they will be redirected to the Main Screen and the 3D gesture will be played immediately.

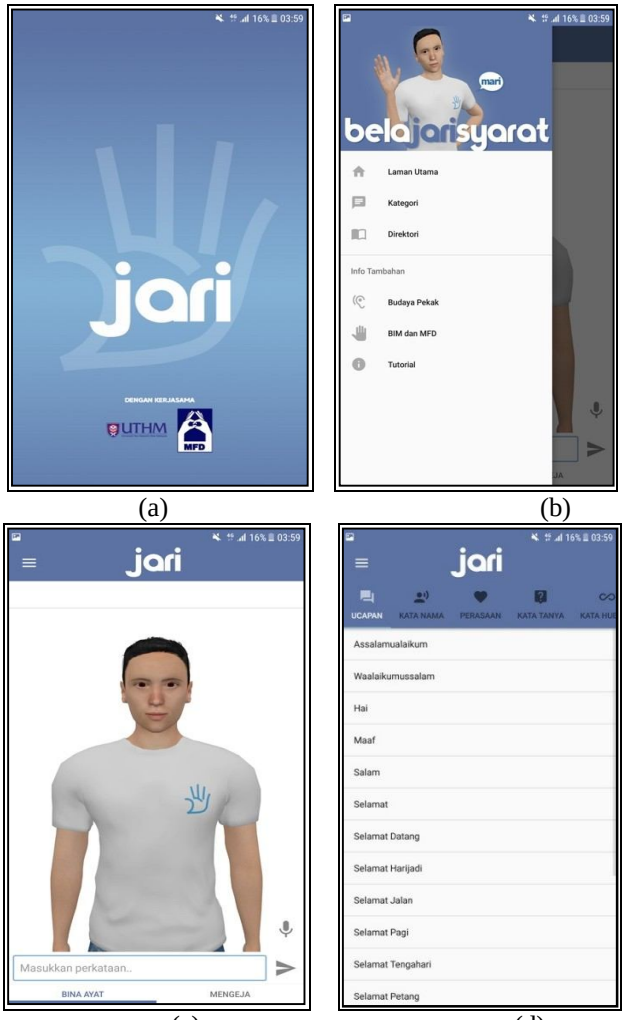

Figure 2: JARI main storyboards (a) Splash Screen (b) Navigation Drawer
(c) Main Screen (d) Category Screen

3.7 Main Function Development

In this section, the implementation and testing procedure for MSL Translator application, JARI are discussed in detail. First, the process of modelling and animating the 3D Model that was used in the application is elaborated. The development of the application interface is then detailed focusing to the implementation of the application's navigation method. Next, the implementation of Android Speech API and the integration of 3D animation in the application is explained. Finally, the testing procedure used for this which consist of Alpha and Beta testing project is detailed.

3.8 Modelling and Animating the 3D Model

The human model used in this application is modelled by using MakeHuman. MakeHuman is an open source and powerful software that was developed for modelling and prototyping a realistic human model. In MakeHuman, the body parts, facial features and the skin of the 3D model are modelled and exported to MHX2 (MakeHuman eXchange) format so that it could be imported in Blender software to define its clothing material and rigging process. Figure 3 shows the model is modelled in MakeHuman.

Figure 3: The 3D Human Model Created in MakeHuman

After the 3D model is exported in Blender, the rigging of the model is defined. The model, which is also known as mesh in Blender, need to be rigged in order to animate it. Since MSL generally involved many hand, finger and facial movements, most part of the body is rigged with digital skeleton, which can act as a handle to move and animate the mesh into desired pose. A total of 53 bones including facial and torso bones are used for this mesh. After rigging is done, the animation for each MSL movements is composed. The animation is done in Blender by implementing the key frame method, in which only the important pose involved in the animation are animated in specific time sequence marked as the key frame.

Figure 4: Four Key Frame Poses Created in Blender for MSL Animation of the Word '*Selamat*'

Figure 4 shows the example of using key framing in animating the MSL for the word '*Selamat*' (greetings). Four key frames are created in the timeline and the important poses is animated only in those key frames. Blender will then calculate the movements path in between the key frame to create a smooth animation.

3.8.1 Development of the Application's Interface

In Android Studio, the application interface layout is defined in XML while the behaviour of the interface is programmed by using Java language. The overall interface design of JARI is heavily influenced by Material Design, which is a design language that was developed by Google as a guideline to ensure the design consistency across Android applications.

In JARI, the Navigation Drawer is used as the main menu and navigation tools for user to navigate across the applications. Navigation Drawer is a UI panel that is hidden on the left side of the application when it is not in use but appears when user swipes a finger from the left edge of the screen or when user touches the drawer icon placed in the app bar. While Navigation Drawer is used as the main navigation tools in the application, there were also another navigation tool that is used in JARI which is known as Tabs. Tabs provide a lateral navigation between screens by using a horizontal finger gesture (swiping to left or right).

3.8.2 Implementing Android Speech API

Android Speech API is used in JARI to accept input from user in the form of speech and translate it to a 3D animated MSL. This allows user to click on the recording button and speak, and the application will process the speech input into text and use the text to search for the right animation.

As mentioned in Section II, Android Speech API can be initialized by creating a new Intent and including the RecognizerIntent class to start a new Speech Recognition activity. The code segment shown in Figure 5 shows how this API is implemented in this application.

```
Intent intent = new

Intent(RecognizerIntent.ACTION_RECOGNIZE_SPE
ECH);
intent.putExtra(RecognizerIntent.EXTRA_LANGU
AGE_MODEL,

RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
intent.putExtra(RecognizerIntent.EXTRA_LANGU
AGE, "ms-MY");
intent.putExtra(RecognizerIntent.EXTRA_PROMP
T,
        getString(R.string.speech_prompt));

try {
    startActivityForResult(intent,
REQ_CODE_SPEECH_INPUT);
}
    catch (ActivityNotFoundException a){

}
```

Figure 5: Code Segment for the Implementation of Android Speech API

The `RecognizerIntent` class, as shown in the above figure, requires a few necessary flags or data source in order to set its behaviour. The description of the flags used in the above code segment is summarised in Table 3.

Table 3: Recognizer Intent's Flags with its Description

RecognizerIntent Flags	Description
`RecognizerIntent.ACTION_RECOGNIZE_SPEECH`	Start the activity that will prompt user for speech and sent it through a speech recognizer.
`RecognizerIntent.EXTRA_LANGUAGE_MODEL`	Informs the recognizer which speech model to prefer when performing ACTION_RECOGNIZE_SPEECH
`RecognizerIntent.LANGUAGE_MODEL_FREE_FORM`	Use a language model based on a free-form speech recognition
`RecognizerIntent.EXTRA_LANGUAGE`	Set the language model which is Bahasa Malaysia by tagging "ms-MY"
`RecognizerIntent.EXTRA_PROMPT`	Display text "Cakap sesuatu" to prompt user to speak.
`RecognizerIntent.EXTRA_RESULTS`	An ArrayList<String> of the recognition results after performing the speech recognition.

After the recognition is done, `startActivityForResult()` is called to get the result from the `RecognizerIntent`. The result is acquired by using `getStringArrayListExtra()` with `RecognizerIntent.EXTRA_RESULTS` as the flags for the function. Figure 6 shows that the recognition result is then stored as an array of strings and is passed to a class to display the appropriate 3D animation.

```
@Override
public void onActivityResult (int requestCode,
int resultCode, Intent data){
    super.onActivityResult(requestCode,
resultCode, data);
 switch (requestCode){
  case REQ_CODE_SPEECH_INPUT:
   if (resultCode == RESULT_OK && null != data){
                ArrayList<String> result =
data.getStringArrayListExtra(RecognizerInten
t.EXTRA_RESULTS);
       String text = result.get(0);
       translateTextToAnim(text);
       }
   break;
    }
}
```

Figure 6: Code Segment for `onActivityResult()`

3.8.3 Displaying and Playing 3D Animation

The 3D animation is displayed in the application by using `VideoView` with `MediaPlayer`. The `VideoView` class can load media from various sources, takes care of computing its measurement from the media so that it can be used in any layout manager and provides various display option. `MediaPlayer` class is used to control the animation such as to play, pause and replay the animation.

To attach the animation to `VideoView`, URI class is used. URI stands for Uniform Resource Identifier and is usually used to tell a content provider, which in this case refers to `VideoView`, of what should be accessed by reference. It acts as an immutable one-to-one mapping to a resource or data. In Figure 7, `Uri.parse` method is used to parse the idle animation that is stored in raw resource, as an URI object so that the media resource can be set as an immutable reference to be played inside `VideoView`. The idle animation is displayed when the application is first launched or when the application is not in used.

```
public Uri idleAnimation (){
    String uriPath2 =
 "android.resource://com.example.sofiahsajap
.ijbi/" + R.raw.idle;
    Uri uri2 = Uri.parse(uriPath2);
    loopAnimation();
    return uri2;
}
```

Figure 7: Code Segment to Setup an Animation by Using URI

Figure 8 shows the code to play a parsed URI object which refers to the animation, inside `VideoView`. `setVideoURI()` class is used to set the video path by using URI reference while `start()` class is used to start the animation immediately.

```
animplay = (VideoView)


getView().findViewById(R.id.anim_play);
final Uri uri = idleAnimation();
animplay.setVideoURI(uri);
animplay.requestFocus();
animplay.start();
```

Figure 8: Code Segment for `on Activity Result()`

One of the key features in JARI is it allows user to choose whether to build a sentence or spell a word from the list of word and alphabet in JARI and translate them to the 3D animation of MSL. Hence every words and alphabets are stored as a single entity instead of storing it as a complete sentence or word. This method is used as it will help reducing the application size and will give users the flexibility of creating their own sentence or word. To enable this, the following technique is implemented:

Step 1: If user choose to build sentence, the input received from user is stored as an array of strings. But if user choose for fingerspelling method, the input received is stored as an Array of characters.

Step 2: Start a loop through the array objects.

Step 3: The objects for each of the array is passed through a Java class named checkAnimation that was created to check whether the animation exist in the application.

Step 4: If the animation does not exist, the class will return Boolean value false, and the application will display blank animation with Toast message to indicates user to opt out for fingerspelling method instead (if user choose to build MSL sentence).

Step 5: To play the animation sequentially, the method setOnCompletionListener is used to make sure that each animation is played right after each other only after the previous animation is complete.

Step 6: Stop the loop when it reaches the end of the array.

4. RESULTS AND DISCUSSION

The Beta release of JARI is uploaded on HockeyApp and is distributed and tested among Beta users. The testing process is performed by adapting the Technology Acceptance Model (TAM) to investigate user's perception and intention of using JARI application to learn MSL. TAM consists of four constructs, including perceived ease of use, perceived usefulness, attitude toward using and behavioral intention [17]. Based on these four constructs of TAM, a questionnaire is created by including questions that relate to the constructs. The relationship between the questions and the item in the questionnaire is as summarized in Table 4.

The questionnaires are then distributed among 20 selected Beta users that consists of young adults and adult that intended on learning MSL. The questionnaire proposed several statements based on TAM constructs and users were asked to give their level of agreement or of disagreement with the items using five-point Likert scale. The data and feedback collected from the survey were than statistically analyzed and the result from the survey were tabulated into bar charts as shown in Figure 9 – Figure 12.

Perceived ease of use refers to a level of easiness that user feels when using JARI to learn MSL. Based on Figure 9, the user's level of ease of use falls at a satisfying level. Majority of users strongly agreed that learning to use JARI is easy (10 users), JARI provides a clear instruction on how to use the application (11 users) and the operation of JARI does not took too much time (10 users). A majority number of users agreed that JARI is easy to use (11 users). However, one of the user disagreed on the operation of JARI does not require too much time.

Perceived of usefulness refers to user's perspective on whether the application is able to help them learn about MSL and the deaf communities. Referring to Figure 10, many users agreed that JARI is useful in helping them to learn on how to build a sentence in MSL (12 users) while half of them strongly agreed that the application able to help them to spell in MSL (10 users). Most users also strongly agreed that JARI can increase their proficiency in MSL (11 users), help them learn about the deaf communities (14 users) and help them learn MSL better (14 users).

Attitude toward use refers to user's satisfaction level when they're using the application to learn MSL. The feedback shows that users have a positive attitude towards the application. As shown in Figure 11, 18 users strongly agreed that JARI is a good idea and 14 users felt that using the application is a fun experience. 13 users strongly agreed that JARI makes the experience of learning MSL more interesting. There are equal numbers of users that strongly agreed and agreed that they like using JARI to learn MSL.

Figure 12 shows the behavioral intention of users on using JARI on a long-term scale. The result shows that 10 users agreed that they intended; 10 users predicted; and 8 users planned; on using the application again in the next 6 months.

Table 4: Relationship Between TAM Constructs and the Measured Items

Constructs	Measured Items
Perceived ease of use	[PEU1] I think JARI is easy to use [PEU2] I think that learning to use JARI is easy [PEU3] I think that JARI provide a clear guidance on how to use the application [PEU4] I think that the operation of JARI does not require too much time
Perceived usefulness	[PU1] I think JARI is useful in helping me to learn on how to build a sentence in MSL [PU2] I think JARI is useful in helping me to learn on how to spell in MSL [PU3] I think JARI can increase my proficiency in MSL [PU4] I am able to learn about deaf communities in Malaysia by using JARI [PU5] If I use JARI, I can learn MSL better
Attitude toward use	[AT1] I think JARI is a good idea [AT2] I think JARI is fun [AT3] I think JARI makes the experience of learning MSL more interesting

	[AT4] I like using JARI to learn MSL
Behavioural intention	[BI1] I intend to use JARI again in the next 6 months [BI2] I predict I would use. JARI again in next 6 months [BI3] I plan to use JARI again in the next 6 months

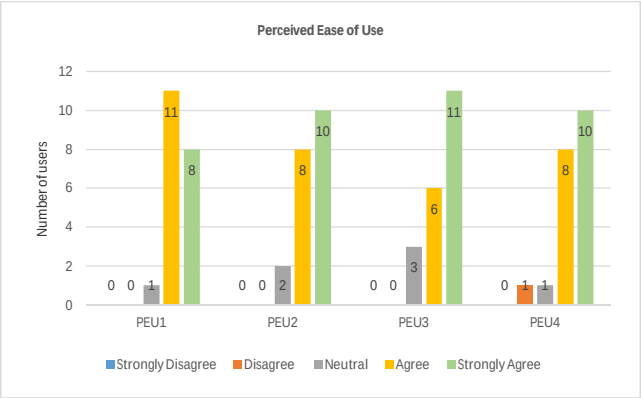

Figure 9: Users' perception on ease of use

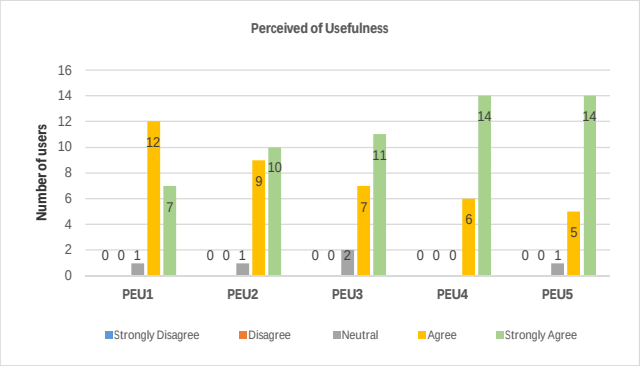

Figure 10: Users' perception on MSL usefulness

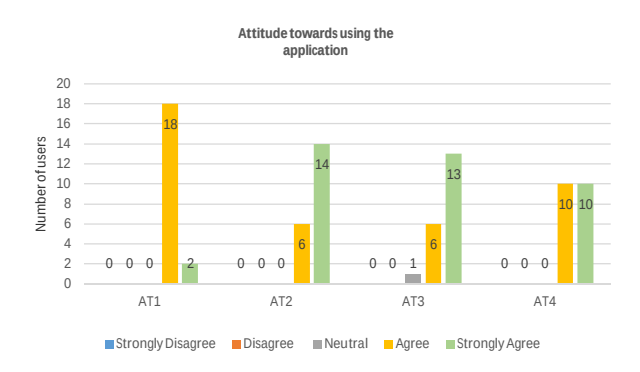

Figure 11: User's attitude towards MSL

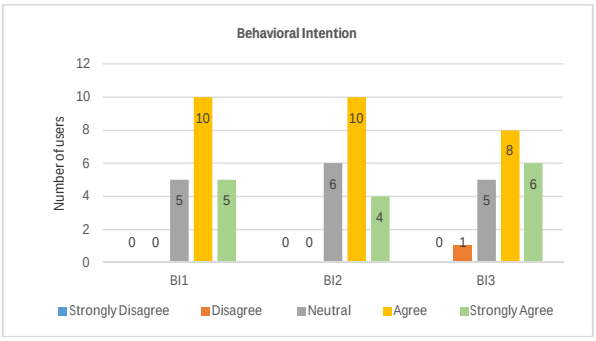

Figure 12: User's behavioural intention

5. CONCLUSION

Overall, all of the objectives of this project, which are to design a MSL learning application on Android platform that use Speech Recognition technology, to develop an interactive learning mobile application that translate words or phrases to 3-Dimensional animated sign gesture and to perform testing for the application, have been successfully achieved. JARI application does have its own advantages. However, with limited time and given resources, there were also limitations to the application. Hence, further works and improvements need to be made on the application to ensure its usability in long-term context. This section discusses the advantages and limitations identified in JARI application in term of its ease of use, usability and functionality. Besides, the future works for the application are further elaborated based on its potential as an on-going project**.**

5.1 Advantages of JARI

The MSL Translator mobile application or JARI is among the few Android application that exists in the market today that use 3D animation to demonstrate the MSL hand gesture. Compared to pictures and textual description, 3D animation can be more helpful in demonstrating the sign language since some sign language involve a complex hand movement, and thus it needs a detailed animation to precisely describe the movements.

Besides, JARI also able to accept input directly from user in the form of text and also speech. Hence, instead of scrolling and browsing through list of categories and words, user can directly type the word or 'speak' to the application, and the application will display the corresponds animation. User can also use this application for fingerspelling and build their own sentences in MSL by simply providing their input. There are hundreds of probabilities of words and sentences that can be made by user to be translated in MSL by using this application.

Aside from using this application as MSL learning tool, user can also explore and learn more about deaf communities in *Budaya Pekak*. *Budaya Pekak* provides interesting and riveting facts that revolves around the deaf cultures among the deaf communities in Malaysia.

5.2 Limitations of JARI

The development of JARI was executed in a very short time period and with limited resources, therefore it has some limitations that might need time to be resolved. For example, there are limited word choices that can be demonstrated in MSL, since the application only covers the first eight chapters from the MSL main reference book. This affect user's experience in using the application since user is unable to view the MSL gesture for a complete sentence if one of the word in the sentence is not present in the application database and needs to opt out to fingerspelling method instead. Even though fingerspelling is generally accepted as a form of sign language, it requires more time and more hand movements to form a word.

JARI also use a different user interface approach compared to existing MSL learning application in which user needs to browse through the list of categories and words to view its corresponds MSL gesture. Therefore, it might be difficult and confusing for first-time users to use this application.

5.3 Future Works

The future works on JARI should focus on breaking through its current limitations. Improvements need to be made to ensure that it can provide a pleasant user experience in future. More 3D animations of MSL and if possible, all the MSL gesture contained in the MSL main reference book need to be included in the application so that user can have a wide possibility and much more abilities to translate any words to 3D animated sign language. With the wide range of words covered in the application, there is a great possibility that JARI can also serves as a universal MSL translator in future.

The applications also were currently developed only for Android platform. The plan of developing the applications on other mobile operating system platform such as iOS and Windows Mobile OS should be considered so that JARI can be made available to more users and cater to larger market.
at the bottom of the column in which it is cited; do not put footnotes in the reference list (endnotes). Use letters for table footnotes (see Table I).

ACKNOWLEDGEMENT

We would like to thank Universiti Tun Hussein Onn Malaysia for supporting this project under the TIER 1 grant.

REFERENCES

1. Gordon, R. & Grimes, B. (2005) **Ethnologue.** Dallas, Texas, SIL International
2. Hurlbut, H. M. (2000). **A preliminary survey of the signed languages of Malaysia**.
3. Sign language. 2018. In *Merriam-Webster.com*. Retrieved at May 15, 2018, from https://www.merriam-webster.com/dictionary/sign-language
4. Muharam, M. (2007). ***Translator of Malay*** text to Kod Tangan Bahasa Melayu.
5. Ow, S. H., Mokhtar, S. & Zainuddin, R. (2007) ***A Review on the Teaching and Learning Resources for the Deaf Community in Malaysia.*** Chiang Mai University Journal of Social Sciences and Humanities.
6. Reed, B. (2010). A brief history of smartphones. PC World, 18.
7. Karch, M. (2017, October 02) What is Google Android, lifewire. Retrieved from https://www.lifewire.com/what-is-google-android-1616887
8. Riley, S. (2015, July 2). Android Version History: ***A timeline of New Feature, tom's guide***, Retrieved from https://www.tomsguide.com/us/android-version-history, news-21211.html
9. Android 2.3 ***Platform and Updated SDK Tools*** (2010, December 06), Developers Blog, Retrieved from https://android-developers.googleblog.com/2010/12/android-23-platform-and-updated-sdk.html
10. Whitman, R. (2013) ***Demystifying Android 4.3 Jelly Bean, Extreme Tech,*** Retrieved from https://www.extremetech.com/mobile/162275-demystifying-android-4-3-jelly-bean
11. Ryan, P. (2009) Google releases Android 1.6; ***Palm unleashes*** WebOS 1.2, Ars Technica.
12. Velazco, C. (2011) A Quick Android 4.0 ***Ice Cream Sandwich Feature Rundown***, techcrunch, Retrieved from https://techcrunch.com/2011/10/18/a-quick-ice-cream-sandwich-feature-rundown/
13. Unuth, N. (2016***) What is Speech Recognition?*** lifewire, Retrieved from: https://www.lifewire.com/what-is-speech-recognition-3426721
14. Liu, C. (2018) ***Exploring the Android Speech API for Voice Recognition***, developer.com. Retrieved from https://www.developer.com/
15. **Bahasa Isyarat Malaysia** (3rd ed.). (2014). Puchong, Selangor: Persekutuan Orang Pekak Malaysia
16. Saifudin, W. S. N., Salam, S., & Abdullah, M. H. L. (2012) Multimedia Mobile Content Development Framework and Methodology for Developing M-Learning Applications, *Journal of Technical Education and Training (JTET)*, Vol 14(1).
17. Davis, F. D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance. MIS Quarterly, Vol. 13, No. 3, 319-340. https://doi.org/10.2307/249008