A REPLICA MODEL ON ON-LINE KBC GAME FOR MULTI PLAYERS USING VOICE BASED CONTROLLED MECHANISM

Abstract

One of the most popular game shows on Indian television is Kaun Banega Crorepati (KBC), also known as KBC. The player must correctly respond to each question in order to move on to the next one in this simple quiz game. The player receives prize money if they successfully respond to all the questions. While this game is intended to be enjoyable and entertaining, it also improves the viewers' general knowledge. The objective of paper is to develop a simple quiz game that blends elements of the KBC show into a game that a group of players can enjoy hands-free and with a little setup while playing on a big or small screen. The use of voice recognition makes this possible that includes elements taken directly from the game show and is fun, informative, easy to play, and maintains the spirit of the original show. The game should be able to recognize the player's voice and perform the right action.

Keywords: Crorepati, Spotting, Detection

Authors

Naga Jyothi P
Department of Computer Science and Engineering
GITAM School of Technology
GITAM (Deemed to Be University)
Andhra Pradesh, India
npothaba@gitam.edu1

Venkata Ramana Mancha
Department of Computer Science and Engineering
GITAM School of Technology
GITAM (Deemed to Be University)
Andhra Pradesh, India
vmancha@gitam.edu

Sahrudh D. V
Student
GITAM School of Technology
GITAM (Deemed to Be University)
Andhra Pradesh, India

Suresh. S
Chief Librarian
ANITS
Visakhapatnam, Andhra Pradesh, India
sureshs.nice@gmail.com4

I. INTRODUCTION

Imagine playing a game with all of your friends at a house party. People typically play games with several joysticks and controllers on TV. Now, not all of us can buy an Xbox or a PlayStation 5, and even if our model can, the only games that can be played in a group are some extremely violent titles, which won't be fun for everyone at the party. Higher education's face-to-face instruction needs to change in order to better serve the current generation of students [1]. Concerningly high levels of learner and teacher demotivation are a result of the outdated master class method of teaching. By allowing students to participate in their active thinking activity, active learning strategies increase student engagement and, as a result, motivation. One active learning method that can help in this area is the usage of gamification in the classroom[2]. As an illustration to both of these situations, our goal was to create a straightforward game that anyone could play and enjoy. "Kaun Banega Crorepati" is a TV show competition that is one of those games that anyone, regardless of age or gender, can play. So that model soughts to replicate the show into a playable computer game with minimal setup[3].

Since creating games is nothing new, we want to go one step further and make use of speech recognition. These days, the number of innovative voice recognition programmes is expanding really quickly. Web searches, dictation, phone calls, notes and reminders, and YouTube automatic subtitling are all common uses for automatic speech recognition. Although they do not allow full use of its potential, the widely used Mac OS and Windows operating systems readily contain integrated speech recognition tools. One of the most recent applications for voice recognition is the gaming environment[4][5]. Although it is a very promising field, not many computer games employ it. With the creation of a computer game based on "Kaun Banega Crorepati," our study aims to investigate the field of integrated speech recognition and text-to-speech synthesis. The purpose of the model was to design and develop a game that is capable of identifying the player's voice and doing the appropriate action.

II. LITERATURE REVIEW AND LIMITATIONS

To start looking into the various gesture technologies believed it would be best to design an application that displays the technology rather than an overly complicated game that only leverages the technology for its own purposes. To arrive at the conclusion that a quiz-style game has all the traits necessary to examine these technologies and their applications after looking at the capabilities and features of the already accessible technologies. The benefits and drawbacks of a number of gesture-based gadgets, such as the Myo Armband, Xbox Kinect, and Leap Motion. and found that the rate of error in the gesture readings was too high to produce a trustworthy working solution after evaluating a number of these devices[6]. Another finding is that those with mobility issues would not be able to play a game created using these technologies, which would shrink the market for the game [7]. Hence, the proposed model need concentrate on voice technology in our research. The problem with the present apps is that despite being a great tool, speech recognition hasn't been successfully incorporated into games. To investigate each voice-activated game available right now. The games that are currently accessible include ScreamTrain, Resonance, and OneHand Clapping[8][9]. That examined in these games, so our model found that none of them truly use a proper speech or word recognition sound; instead, they all rely

on arbitrary patterns or cliched sounds like shouting, humming, or clapping. These games are not voice-controlled and simply rely on audio.

Our research revealed a sizable market need for a challenging yet fun and entertaining voice-controlled game.

III. PROPOSED MODEL

The proposed model for game presents one question and four options—the letters A, B, C, and D—to the player. The player has to try to answer the question truthfully. Once the user chooses an answer, a final answer box will appear asking them to confirm their choice. If the user selects correctly, they will go on to the subsequent question and up the scoreboard. There are a total of 15 questions, and they increasingly get harder as the game progresses.

The user is aware of that the selection of voice-gestures is limited, but rather than simply adding more gestures for the sake of it, model wants to concentrate on how these gestures will be used in gameplay. By implementing this game in the classroom, the teacher becomes "the presenter" whereas one of the several students transforms into "the contestant". The rest of the classroom fulfills the role of "public". As soon as a contestant is placed in the "hot seat," the emcee begins to quiz them on recent class lecture subjects.

There are a few lifelines available to the participant that let audience members participate actively in the discussion. A range of incentives and awards are presented to the kids as contestants progress in the challenge. This game offers the students a fun, new method of stimulating their learning. People are more likely to attend class because of the game's built-in rivalry among the players, which motivates them to study at home before "the contest" and encourages them to do so. Teachers also welcome the students' increased participation and interest. Consequently, it can be said that the use of this strategy has a beneficial impact on the motivation and involvement of both students and teachers.

The proposed model pictured a group of gamers playing together on a big or small screen, enjoying our hands-free, straightforward setup. The goal of model was to incorporate components from the KBC show into a simple game that anyone could play using a hands-free setup on either a large or small screen as shown in figure 1.

1. Architectural View for Proposed Game Engine Model

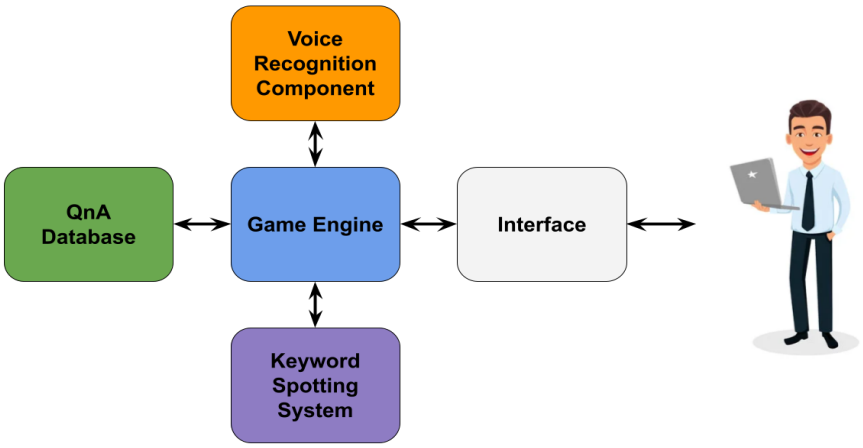

Figure 1: The Proposed Game Engine Model to play the game

2. Methodology: The keyword spotting (KWS) component is the most important part of this application. It needs to be made known right away that the software currently lacks a fully functional speech recognition functionality. Instead, a KWS block is used, and it works well for the situation at hand. Due to the need for an open vocabulary and dynamic terms selected especially for each enquiry, a KWS was adopted in place of an ASR. Word confidence scores may also be provided by KWS as a detection strategy [10][11]. As a result, the detected keywords may be approved or rejected depending on the score obtained. Traditional machine learning techniques are the foundation of the system for finding keywords. Hidden Markov Models and neural networks are employed. Three phases, or parts, make up the KWS module: extracting features, estimating with posteriors, and decryption as described in figure 2.

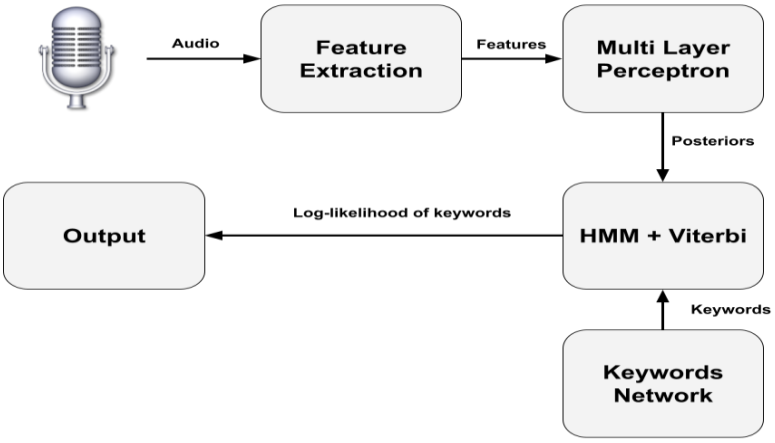

Figure 2: Keyword spotting process

Our keyword detection method starts with feature extraction. At this point, the input wav file is processed using the HCopy tool, and the related feature vectors are then acquired. Among other specifications, the input wav file must be 16 kHz sampled and 16 bit PCM coded for speech.

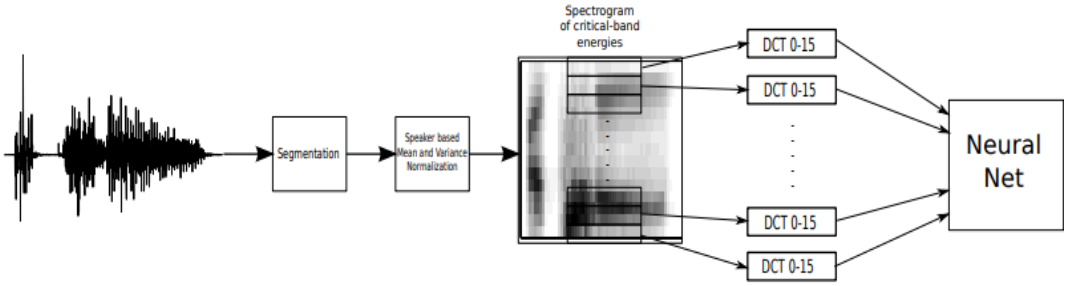

Figure 3: Features Extraction

The features extraction algorithm goes through the following steps shown in figure 3: 1. Create voice segments using a 25 ms Hamming window, then translate them into frequency domain using the Fourier transform (frame rate 10ms). 2. Map the linear frequency bins into the 23 critical bands in order to obtain 23 critical-band-warped energy (using the mel scale). 3. Use the DCT (discrete cosine transform) for each significant band across a 300 ms temporal span. 4. Just maintain the first 16 DCT coefficients (capturing the most of data variability). 5. Concatenate the DCT coefficients obtained for each critical band to get a vector with 368 components.

The Viterbi algorithm is employed to complete the decoding. The approach provides the log likelihood for the sequence of HMM states that is most likely given the observations for each keyword model. In order to estimate the log-likelihood ratios for each keyword, which will be utilized as scores, it also computes the log-likelihood for the background model. In this instance, the "null model" is the current keyword model, and the "alternative model" is a background model[12]. The decoding algorithm's steps are typically establishing a keyword network, building HMM sequences for each term, and concatenating HMMs in accordance with the keyword network, determining the log-likelihoods for each keyword and backdrop model using the Viterbi technique and establishing the positions for each keyword (log-likelihood ratio)[13][14]

IV. STAGES FOR PROPOSED GAME ENGINE MODEL

1. **File Reading Questions:** The game relies on players answering questions, therefore one criterion was that there be a wide variety of questions to keep players interested. The questions are loaded from a questions.json file to accomplish this. The ability to add more questions in the future without having to change of code is another advantage of this. Below is an example of four of the many questions from that file. The questions are shown above in JSON format. Therefore, in our model developed a Question object as shown below so that model could use these questions in a game.1500 distinct options are now available in the file. 15 questions are then randomly selected and stored in an array for every new game.

2. **Sound System:** Any game's sound effects are an essential element. In KBC, the game's use of sound effects can increase drama and tension. In order to improve the gameplay and the drama the game provides to the user, model enables to decide to replicate these sounds. To do this, a Sound Controller was included in the game. The initial setup a Sound Management singleton class to contain a collection of sound clips that would be loaded at various moments during the game. This originally worked, but as the game advanced in complexity, issues with voice recognition and synchronizing the delay of the sounds playing in coroutines began to arise.

   To solve this issue, proposed creates a static Sound Controller that worked differently. By loading all the sound files on awake, so that it is able to decrease memory usage and get rid of the delay that occurs when a sound clip is loaded before it can be played. The game incorporates the iconic KBC theme as well as all the serious background themes. The sounds are programmed with a rise in difficulty. 1-5 is easy, 6-10 is medium. From 10-16 the sound increases in terms of tension

3. **Scoreboard Management:** The KBC contestants advance according to the amount of money they receive for each question, starting at 100 rupees and rising all the way up to 1 crore rupees. The denomination scale in our game is set out similarly. The boards are disabled (i.e., set to false) before the beginning of the game and may only be enabled using the proper voice command. Model is able to reduce memory use and eliminate the lag that happens while the game is being played by loading all the image files on awake.

4. **Implementing Lifelines**: On the game show, lifelines assist the participant in getting out of a sticky circumstance or a challenging question. The show's lifelines typically consist of 50-50, audience polls, and calling a friend. The decision to not use all of the lifelines because our game is a computer game with no actual stakes. The interested part in the last lifeline, the Phone a Friend lifeline. Since this is our final semester and will leave on our separate routes, the bond among us developers wanted to build something that would be immortalized. Thus, inclusion a helps feature whereby speaking "help" would activate the lifeline, which would then select one of the four devs at random to respond to a query. The loaded the images into the environment but kept them disabled at the beginning of the game. They will be activated on calling help. A random number generator selects the dev to contact and finally the right option is chosen automatically.

5. **Answers Checking:** The questions are loaded from a questions.json file to accomplish this. Once the word is spoken, coroutines to select the answer are started. Each word has a switch case to lead to a coroutine as below.The selected answer is flashed and checked if correct or wrong, based on whether the game moves forward or exits to the main menu. If the question level is beyond 16, the game ends declaring the player the winner.

6. **Speech Recognition Engine:** A switch case statement is used here. The phrase or word recognized is converted into a string which is checked if it fits the correct case statement. The cases include the four alphabets—A, B, C, D—final answer, yes, no, play game, main menu, scoreboard, help, new game, and more.

7. **Results and Testing Discussions:** Although we are aware that the available voice-gestures are restricted, model had chosen to focus on how these gestures will be employed during gameplay rather than simply adding more gestures for the sake of it. To forward our ultimate goal of accessibility, to make sure the game consumed few resources and had an intuitive user interface. The model proposed configured the Voice Recognition in this way to avoid the user experiencing unexpected behavior while playing the game. The complete execution of this model is done under the python environment with JVM, AI enabled voice recognizer, and Unity 3D platforms and tools had been used.

Figure 4: Main Menu screen

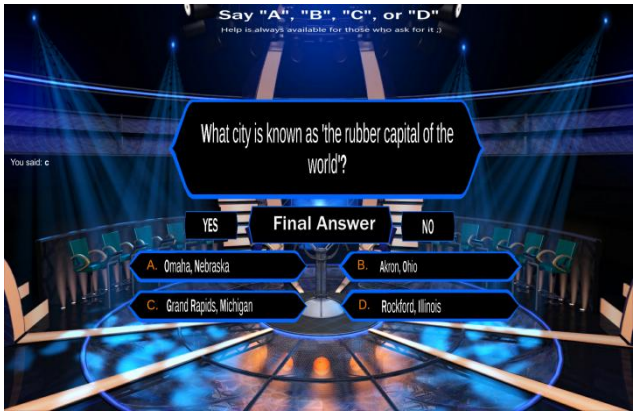

Figure 5: Question State

Figure 6: Getting a right answer lifeline help

Figure 7: Wrong answer

The images above depict *different* gaming screens. You have a choice between two ways to play or exit. Each time you move forward, the questions automatically load. Instructions are shown on the screen itself to avoid confusion. The user can freely change their mind before submitting an answer because after speaking, the prompt for the final response is also displayed. When a correct response is selected, the box illuminates green to indicate that it was selected. Red flashes to highlight both the incorrect and correct responses when the erroneous response is chosen. When help is required, one of the developers is contacted, and they immediately select the appropriate response. The help is only restricted twice every game, though. Once 15 questions are done, the game is complete and the player is declared the winner. The prototype of this model is illustrated in the figures 4, 5, 6 and 7

Table 1: Main Menu Contents

Test	Expected Result	Actual Result	PASS/FAIL
Saying "new game" should launch a new game	Game scene loads	Game scene loaded	PASS
Saying "play game" should launch a new game	Game scene loads	Game scene loaded	PASS
Saying "play" should launch a new game	Game scene loads	Game scene loaded	PASS
Saying "quit" should terminate the game	Game shuts down	Game shut down	PASS
Saying "exit" should terminate the game	Game shuts down	Game shut down	PASS
Clicking [new game] should launch a new game	Game scene loads	Game scene loaded	PASS

Table 2: In Game Scene Score details

Test	Expected Result	Actual Result	PASS/FAIL
Saying "new game" should launch a new game	Game scene loads	Game scene loaded	PASS
Clicking [new game] should launch a new game	Game scene loads	Game scene loaded	PASS
Saying "show scoreboard"	Scoreboard comes into view	scoreboard came into view	PASS
Saying "scoreboard"	Scoreboard comes into view	scoreboard came into view	PASS
Saying "score"	Scoreboard comes into view	scoreboard came into view	PASS
Saying "pause"	Scoreboard comes into view	scoreboard came into view	PASS
Saying ["a","b","c","d"]	Should select and highlight answer	answer was selected and highlighted	PASS
Saying ["a","b","c","d"]	Should select an answer and prompt for final answer	answer was selected and final answer prompted	PASS
Saying "help"	Should call a help and select the correct answer	a random dev was called and correct answer was chosen	PASS

Table 3: In Game Scene (Final Answer mode)

Test	Expected Result	Actual Result	PASS/FAIL
Saying "Yes"	Selects final answer	Final answer selected	PASS
Saying "Final Answer"	Selects final answer	Final answer selected	PASS
Saying "No"	Deselects final answer	Final answer deselected	PASS
Select correct answer	Correct answer highlighted and move to next question	Answer highlighted and new question loaded	PASS
Select correct answer to progress game	Position on scoreboard should increase	Scoreboard position increased by 1	PASS
Select wrong answer	Wrong answer highlighted and game ends	Wrong answer highlighted and game ends	PASS

To evaluate whether each voice command led to the right action, testing with each sound and identified whether it recognised. The model graded them on a scale of PASS or FAIL. The results of the speech recognised is as follows in table 1 and In game scene shows the current results of the contestants as shown in table 2 and table 3.

V. CONCLUSION AND FUTURE WORK

It was fascinating to learn about the vast diversity of technologies that are currently available to enable user input into a system without utilizing the traditional keyboard and mouse method from a research standpoint. Even though there are only a few games that make use of these technologies, it is abundantly evident that there are increasing numbers of implementations every day. There are surprisingly few voice-controlled games on the market, in contrast to our method, which recognised keywords from a user. It is a revelation throughout our investigation consider only to be interesting. Everything could find suggests that the lack of voice-controlled games is primarily due to issues with cross compatibility. The multiple libraries, packages, and software programmes used for voice input all appear to target different systems rather than providing a multi-platform solution. As a result, if a voice-controlled game were to be created for a number of platforms, each build would need a unique implementation. Utilizing Windows Speech Recognition for Windows and UWP platforms as well as Mobile Speech Recognition to target iOS and Android mobile devices.

All of our original objectives were met, and our plans were put into effect, in the final result. In light of this, to advise giving the model proposed another go because, much like every software problem, there are still a few things that could be done better. The idea may come across as dynamic and interactive both ways if an AI-generated voice is added. Cross-platform compatibility, in our opinion, would considerably improve the game by enabling its complete release to a large market. In order to accomplish this, the code may be changed to recognise the type of device being used and to customize the voice recognition library being used to the target device. A clock to limit time is also another extension possible. More in-game features could be added to improve the sense of authenticity. The model offers the three in-game options of Phone a Friend, 50-50, and Audience Poll to a player who is unsure about the answer.

REFERENCES

[1] Mateusz Felczak & Maria B. Garda (2023) Żulionerzy and the Polish Independent Video Games of the Early 2000s, Studies in Eastern European Cinema, 14:1, 25-38, DOI: 10.1080/2040350X.2022.2071519
[2] Söbke, H. Exploring (Collaborative) Generation and Exploitation of Multiple Choice Questions: Likes as Quality Proxy Metric. Educ. Sci. 2022, 12, 297. https://doi.org/10.3390/ educsci12050297
[3] Felczak, M., & Garda, M.B. (2022). Żulionerzy and the Polish Independent Video Games of the Early 2000s. Studies in Eastern European Cinema, 14, 25 - 38.
[4] Abigail Hotaling, James P. Bagrow, Efficient crowdsourcing of crowd-generated microtasks, PLOS ONE, 10.1371/journal.pone.0244245, 15, 12, (e0244245), (2020).
[5] Warshauer, M. (2013). On-line resources from the American Studies Resources Centre at LJMU Who Wants to Be a Millionaire: Changing Conceptions of the American Dream.
[6] Glickman, S., McKenzie, N., Seering, J., Moeller, R., & Hammer, J. (2018). Design Challenges for Livestreamed Audience Participation Games. Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play.
[7] Aydin, B.I., Yilmaz, Y.S., & Demirbas, M. (2017). A crowdsourced “Who wants to be a millionaire?” player. Concurrency and Computation: Practice and Experience, 33.

[8] Yuksel, B.F., Collisson, P., & Czerwinski, M. (2017). Brains or Beauty. ACM Transactions on Internet Technology (TOIT), 17, 1 - 20.
[9] César Morillas Barrio, Mario Muñoz Organero, Joaquin Sanchez-Soriano "Gamifying the Classroom: An example with the TV Game Who Wants to be a Millionaire?". 10th International Technology, Education and Development Conference, 7-9 March, 2016
[10] Ram, D., Motlícek, P., & Potard, B. (2016). Integration of Real-Time Speech Processing Technologies for Online Gaming.
[11] H. Gasimov, A. Triastcyn, P. Motlicek, H. Bourlard. "WHO WANTS TO BE A MILLIONAIRE?" IDIAP Research Institute. July 2012.
[12] Medcalf, R., & Griggs, G. (2015). "I Never Saw Him Play but We Were the Best of Friends for Years". Communication & Sport, 3, 334 - 347.
[13] M. Seeger. Pattern Classification and Machine Learning (Course notes). Probabilistic Machine Learning Laboratory, EPFL, 2012.
[14] P. Motlicek, F. Valente, I. Szoke. Improving acoustic based keyword spotting using LVCSR lattices. Proceedings of International Conference on Acoustic Speech and Signal Processing, Japan, 2012.

