Wizards’ Chess Log

5th June 2017, Monday

* Searching for APIs for speech recognition.
* Found IBM Watson Speech to Text, CMUSphinx, and other APIs.
* Decided to create a 2-D GUI for chess on Python.

6th June 2017, Tuesday

* No API was found to be easily installable or open source.
* Decided to use MATLAB for speech recognition.
* Installed MATLAB 2013a.
* Learning commands related to audio files on MATLAB.
* Based on above, wrote v0.1 code for speech recognition.
* Code takes in audio from microphone and stores it in a WAVE file.
* Finding methods to filter audio signals in MATLAB.

7th June 2017, Wednesday

* Wrote v0.2 of speech recognition code.
* Tried using LMS Adaptive Filter to remove noise from audio input.
* But, found that noise was not significantly removed by the said filter.
* Removing the filter in next version and try to compare 2 audio files using Cosine Similarity principle.
* Wrote v0.3 of speech recognition code.
* Using Cosine Similarity principle to compare the 2 audios.
* But, difference in start and end of the audios means that output comes not same even if the audios are same.
* Wrote v0.4 of speech recognition code.
* Eliminated problem of different start and end points using cross-correlation concept.
* But, speech recognition is not very accurate.

8th June 2017, Thursday

* Found a research paper explaining speech recognition using Mel Frequency Cepstrum Computation. Link can be found in resources.
* Will try to implement the method mentioned in the said paper.
* Found code which returns Mel Cepstral coefficients(2-D array or matrix) of input speech signal.
* First, cross correlation will be used to eliminate different start points.
* Then, will find difference of 2 matrices containing coefficients and calculate average difference.
* If average difference is less than certain value, 2 audios should be same.
* Wrote v0.5 of speech recognition code using above ideas and concepts.
* Speech recognition is somewhat accurate.
* Algorithm needs more fine-tuning.
* Changed decision to build GUI in QT Creator using C++ instead of Python.

9th June 2017, Friday

* Installed QT Creator on Ubuntu.
* Started learning C++ and QT Creator.
* Designed a chess-board for testing.

10th June 2017, Saturday

* Prepared for presentation.
* Wrote v0.6 of speech recognition code which takes 8 primary recordings and 8 recordings for comparison.
* Recognition is not yet accurate.
* Algorithm needs changes.

11th June 2017, Sunday

* Wrote v0.6.1 of speech recognition code which takes 8 primary recordings and infinitely takes single recordings for recognition (also corrected indentation and reduced memory usage).
* Written a part in a separate function to enhance readability, and propose to do similarly in future.
* Recognition algorithm and results are same as in v0.6
* Found a method called Dynamic Time Warping to compare 2 signals.
* Inbuilt MATLAB function is available for this, plan to implement this method in next version.
* Cannot use said function as it is available from MATLAB R2016a onwards, while we are using MATLAB R2013a.

12th June 2017, Monday

* Rechecking CMU Sphinx for speech recognition as it is getting difficult to implement in MATLAB.
* Successfully installed PocketSphinx and tested for recognition, found accuracy to be good.
* But, the application is based on the Linux command-line, so need to find a method to implement it according to our needs.
* Designed a chess board in QT Creator in QT Canvas 3D application project.
* Modified the base GUI created, the project files have the Images folder inside, so application can load the images without changes to source URL for all images.
* Also, intended base file is not compiling yet, needs to be checked.
* Now compiling, after installing OpenGL.
* Images were not loading, because they were not added to the resources file. Done that and images load correctly.

13th June 2017, Tuesday

* Prepared project report with expected timeline.
* Prepared a block diagram for the project.
* Learnt some animation techniques for QT GUI.
* Applied successfully on chess board we designed earlier.

14th June 2017, Wednesday

* Decided to shift to Sphinx4 instead of PocketSphinx due to good compatibility and ease of use with Java.
* Started learning basics of Java.
* Started learning to write functions in QT GUI for animation.
* Wrote small function but wasn’t able to pass parameter to it.

15th June 2017, Thursday

* Trying to implement speech recognition using Sphinx4 in Java.
* Code compiles and runs, but does not go beyond a certain point.
* Learning to write functions in QT GUI.
* Learnt parallel and sequential animation in QT.
* Finding methods to pass parameters in function for animation to move pieces.

16th June 2017, Friday

* Found that Linux has problems with microphones, so switched to Eclipse(Java IDE) on Windows from Eclipse on Ubuntu 14.04.
* Same code as yesterday now works correctly. Now learning file handling in Java to store recognized text in a file.
* Started preparing for probable presentation(with demonstrations for both GUI and speech recognition) on Saturday.
* Added entire source code for Java program using Sphinx4 which implements voice recognition of digits to Drive.
* Code recognizes with over 80% accuracy, but also gives random outputs on no input.
* This is due to it finding the closest match to the noise. Need to correct this in future versions.
* Found some animation techniques to move pieces at specified place of the player’s choice.

17th June 2017, Saturday

* Prepared for presentation.
* Wrote code for animation of some pieces.
* Due to wrong arrangement of boxes and images in code, images were going behind the boxes in animation.
* Solved that problem by rearranging the position of boxes and images in code.
* Prepared demonstration code of animation for presentation.

18th June 2017, Sunday

* Learning file handling in Java to store speech recognition output to a text file.
* Wrote v0.1 of Java code for speech recognition.
* Now stores the last recognized text in a text file.
* Rest of the code and functionality is same as in the base version.
* Plan to eliminate random outputs on no input in the next version, and to take only a single input on execution of program.
* Decided to write code for taking input from API in C++ file in QT Creator.
* The C++ file will take text input and give orders to qml for animation.
* For writing C++ code started learning C++.

19th June 2017, Monday

* Wrote v0.2 of Java code for speech recognition
* Now added a method(functions are called methods in Java) to count and return number of words recognized in the file class.
* Now storing recognized text in file only if it has 4 words, because only four words will be needed to specify the initial and final position of any piece.
* This eliminates the problem of random outputs to a good extent as random outputs tend to be 1 or 2 words at the most.
* Wrote function (takeInput.cpp) in C++ to take input from text file and convert it into integers.
* Started writing function for validation of input i.e. checking whether the input move is valid or not.

20th June 2017, Tuesday

* Wrote function for validating move of any Rook. Works correctly.
* Wrote the code in a new file(rook\_validation.cpp) rather than in the same primary function(validation\_v0.1.cpp) due to the length of the code.
* Wrote code for 2D array of size 8 x 8 in C++ file to keep track of the game.
* Searched for methods to connect qml and C++ files.
* Wrote program to handle qml file through C++ but program did not work.
* Searching for other methods to connect qml and C++.
* Wrote function for validating move of any Bishop. Works correctly.
* Made v0.2 of primary validation function as a consequence.
* Wrote function for validating move of any Queen. Works correctly.
* Made v0.3 of primary validation function as a consequence.
* Reused functions of rook and bishop for this function.

21st June 2017, Wednesday

* Decided to shift from QT Creator to Java for creating chess GUI.
* Wrote code to design a basic chess board .
* Started learning Java.
* Started learning animation in Java.

22nd June 2017, Thursday

* Due to shift of GUI to Java and implementation of Speech Recognition also in Java, decided to shift entire project to Java.
* Wrote v0.4 of validation code, but in Java. Converted the function into a class and added all other functions(methods) like rook, bishop, queen to the class. Returns true or false for input move’s validity.
* Wrote v0.5 of validation class.
* Added king method. Checks validity of move, but does not check whether king goes into check or checkmate. Not accounting for castling move yet.
* Plan to make methods for checking check condition for both kings and castling validity separately.
* Added pawn method. Accounts for straight basic movement, diagonal killing movement, and initial two step movement. Does not yet account for en passant special move, will be added in future versions.
* Wrote v0.6 of validation class. Added knight method to check validity of move.
* Validation class is basically complete with the exception of special moves like castling and en passant which cannot be added at this stage. This is due to the complexity involved in the rules for mentioned special moves. They can be added at a later stage when game tracking is clearer.
* Learnt how to write methods in Java.

23rd June 2017, Friday

* Learnt how to import images.
* Wrote code for animation in vertical, horizontal and diagonal direction.
* The code for animation of knight is not yet successfully written.
* Wrote code to design chess board.
* Added labels successfully to the main frame.

24th June 2017, Saturday

* Prepared for presentation.
* Made GUI of chess board. Imported images on chessboard successfully.
* Decided to write single method for animation of all pieces.
* Wrote method for animation of pieces.
* But not working properly yet due to repaint method. The repaint method cannot get called if there is no paint method.
* Searching for solution to call repaint method in animation method without calling paint method.
* Wrote v0.3 of speech recognition code in Java. Added a method to count number of recognized words to get exactly 4 words. Added a method to convert words into integers and process them into array coordinates from chess board coordinates.
* Main program class written alongside now takes input through microphone and displays validity of input move.
* Used new language model and dictionary files which are based on a corpus containing 2 words per line of digit words instead of just one. This helps improve accuracy and reduce time for recognition of 4 words to some extent.

25th June 2017, Sunday

* Wrote code for animation of pieces successfully.
* One condition is not working properly. Finding solution for that.
* For making animation successful, created new class and included main frame into animation method.
* The four parameters i.e. final and initial position coordinates will be passed through this method by user to perform the animation on the piece.

26th June 2017, Monday

* Added animation class i.e. ChessBoard class to chess project. Added convert class containing methods to convert between array, board and GUI coordinates.
* Wrote v0.2 of main program to now take microphone input, validate it, and perform move in GUI. Works correctly.
* Added buttons, profiles, player’s name label.

27th June 2017, Tuesday

* Added tracking class with methods to update the 2D array and to store its contents in a text file. It can store 2 instances of the 2D array to help implement the Undo feature later.
* Cleared the main program because it was getting lengthy, will be using it to test new methods for now. Will update it to run the entire program after implementing tracking completely.
* After calling animation method, a new window is opening every time.
* The last move performed is stored but for next move, a new window is opening. Finding solution for this.
* Implemented primary check method. It returns true if king is in check. Sub-methods are present which check this for a specific piece.
* Sub-method for Rook and Bishop added.
* Added colour change feature to player’s label for indicating which player is currently making his move. The active player will be green and other one will be red.
* Sub-method for Queen and Pawn added. Not yet accounting for pieces with same number due to pawn promotion.

28th June 2017, Wednesday

* Added sub-methods for Knight and King. Completed primary check methods for both kings.
* Added labels to main frame to show check, checkmate and winner on screen.

29th June 2017, Thursday

* Added storeBoard and loadBoard methods for tracking, but not functional yet.
* Added checkmate methods for both sides, but not working correctly yet.
* The accuracy of Sphinx4 with the current language model and dictionary model is lower than expected. This may be due to the fact that it has to recognize 4 words. Finding methods to improve accuracy.

30th June 2017, Friday

* Corrected method for checkmate. Manually resetting the board i.e. assigning values to each element of board array, rather than equating names of arrays.
* This is due to pass by reference concept i.e. equating names of arrays gives address of one array to another rather than copying all values.
* Checkmate condition for both kings working correctly now. But, only when no other piece can make a move. So, if a piece can remove check from king, it is not yet accounted for. Will be done in the next version.

1st July 2017, Saturday

* Completed checkmate method for both kings. If a piece can remove check from king, that is also now accounted.
* Added variables for storing no. of pieces giving check simultaneously, which can be maximum 2. In case it is 1, another variable stores the piece which is giving check to the king. Using these variables for detection of checkmate.
* Corrected implementation of tracking methods used for storing, loading, and updating the board. Loading method will be used to implement undo.

2nd July 2017, Sunday

* Added method to check for draw. Draw is declared if insufficient material is available for either side to win. Draw is also declared according to 50 moves rule.
* Added variables and code to update those variables in the tracking class. Variables are used to facilitate checking of the 50 moves rule.
* Added stalemate methods for both sides. Modified draw method and stored part of it in a separate method which tells which pieces are present on the board for a particular side.
* Added code to main method to control the game, animation and speech recognition. Getting an exception after the first move that line not available(related to speech recognition).
* Also, after each move, animation takes place in a new window. So, after 10 moves, we will have 10 windows open.
* Need to solve these issues in the next version.

3rd July 2017, Monday

* Finding a solution to the exception. Found this comment by a Sphinx4 developer on StackOverflow regarding this exception, “This is a bug in sphinx4 that recognizer doesn't release the resource properly and Windows java doesn't allow to open the microphone the second time”. But, no definite or simple solution yet.

4th July 2017, Tuesday

* No progress in solving this problem.
* Tried editing the Sphinx library by decompiling the class files, and compiling them and replacing them. Still does not work correctly.

5th July 2017, Wednesday

6th July 2017, Thursday

* Could not solve the problem, so now will try to use mouse click control instead.

27th July 2017, Thursday

* Managed to fix the exception by restructuring the main method to keep speech recognition on at all times.
* Added code to main program to call the update method to update the board, which was not being done earlier.
* Modified the speech class, it now only contains methods to convert text to integers and the word counting method. Rest of the code was moved to the main method to bypass the exception.
* Changed OR condition between main checking condition in main method to AND.
* Added code to check whether output of find function is -1 and -1, and skip those cases.

28th July 2017, Friday

* Added an array to indicate whether each piece still exists on board or was captured. Added code to toggle visibility of pieces based on this array.
* However, the piece which gets captured disappears instantly, after the validity of the move which gets it captured is proved true.
* This can be changed in future versions by adding some sort of delay mechanism.

29th July 2017, Saturday

* Added code to implement labels showing when check, checkmate, draw, etc. occurs.
* Not working correctly yet.
* Started video editing to make video for YouTube.

30th July 2017, Sunday

* Added flow charts in Project Reports folder which explain the code briefly
* Note that auxiliary functions like undo, restart, or labels stating check, checkmate, etc. are not yet implemented. However, methods required for their execution are already implemented in the source code.

9th August 2017, Wednesday

* Project video has been uploaded to YouTube. [This is the link.](https://www.youtube.com/watch?v=-olxLe8mwtw)