# A Turn-Taking Based Number Guessing Game

Meral Kuyucu
*Dept. of Computer Engineering*
*Istanbul Technical University*
Istanbul, Turkey
korkmazmer@itu.edu.tr

İlknur Çelik
Dept. of Computer Engineering
*Istanbul Technical University*
İstanbul, Turkey
celikil17@itu.edu.tr

***Abstract*—This report summarizes a project completed within the scope of BLG 624E, Human Robot Interaction offered at ITU in the Spring of 2022. The proposed project is a turn-taking number guessing game in which the robot "thinks" of a number and a child guesses the number. With each turn of the child, the robot replies with either an up motion to indicate that the child should make a higher guess, a down motion to indicate the child should make a lower guess and a victory dance in honor of the child finding the correct answer. This project was implemented in Webots using Python. The related GitHub repository is public online [1].**

***Index Terms*—**

## I. Introduction

"Guess the Number" game is a human-robot interactive game designed to support the development of mathematical and social abilities of children in the developmental age. This turn-taking game is played with a humanoid robot, NAO. The game starts off by the robot "thinking" of a number, the rest of the game is conducted in turns. The child takes a guess at the number. The robot interacts with the child by providing a feedback by making physical and verbal communication according to the number he has in mind. The turn-taking procedure repeats and the child takes more turns until the number is guessed correctly. In the end, the robot performs a victory dance. A demo video is provided in the GitHub repository. Recently, studies have been carried out on the use of robots in classrooms and games to facilitate, develop and support children's learning process [1]. These studies lie at the heart of the motivation for this particular project.

Information about the technologies used is given in Part II, a detailed analysis of the game, applied algorithms and models, and implementation of the project are discussed thoroughly Part III. An evaluation of this project in the light of Human-Robot Interaction principles is established in Section IV. The difficulties encountered in the development process are listed in Section V. Finally, potential future work, such as features that can be incorporated to the project and concluding remarks can be found in Sections VI and VII respectively.

## II. Technologies Utilized

This section breaks down the technologies necessary to create and run this project.

[1]GitHub Repository Link

### A. Python Programming Language - For All Tasks

The Python programming language is the language in which the entire project is written. It is used in all three major components of the project:

- Speech Server Configuration
- Virtual Robot Simulation
- Extraction of Robot Movements (from external simulator)

### B. Webots Simulator - NAO Simulation

Webots is an open source robot simulation program developed by Cyberbotics. The physical simulation of NAO was done in Webots. This means, the motions that NAO carries out were interpreted and realized by Webots. Also connection and communication with the speech server was also done in the simulator.

### C. Google Speech API - Speech to Text & Text To Speech

Speech recognition is done using Google API. Google Speech API is responsible for converting the child's commands into digits (Speech to Text) and converting the phrases the robot should say given in text into speech (Text To Speech).

### D. Socket Programming - Communication Between Speech API & Robot

Webots does not enable auditory inputs or any form of speech recognition in the simulator, although we know NAO has this capability. To mimic NAO's speech, we had to find a solution. We used socket programming and created a speech server which enables communication between Google Speech API and the simulator. The server listens to requests from the robot. When a request arrives indicating that it is the child's turn, the server listens for the user's voice commands and sends them to the Google API for STT conversion. Once the text format arrives at the server, it evaluates and makes another request to Google Speech API based on the comparison of the child's number and the robot's number. A spech responce generated from the Google Speech API is sent back to the server and played aloud for the child to hear.

### E. Choregraphe - Motion Generation

Webots does not provide a facilitating interface for the generation of motion files. The only way to conduct a motion is to provide a comma separated value file containing the joint angles of the robot in the timestamps at which they should

be conducted. This file is extremely difficult to manipulate. Thus, motion files were obtained using Choregraphe. In the application, the desired movements were made with the NAO robot via the GUI interface, and the values of the joints were saved as a Python file. This file is different from the Webots motion file, however, we were able to extract the data from the Choregraphe file and convert it to the Webots format motion file using basic file processing techniques.

## III. Implementation

### A. Game Loop

This turn-based game is started with the robot presenting the basic game rules to the child. The robot starts off the game by generating a random number as the goal number and says to the child: "Let's play a game. I am going to think of a number. You try to guess it, and I will tell you if you are correct. Make a guess!" At this point, the speech server begins to listen for the child's input. The child is expected to say a number out loud. After the child completes his/her phrase, the speech server makes an API request and obtains the text representation of this speech input. The server sends this information back to the simulator and the comparison mechanism in the simulator compares the guess with the goal number. If the guess is too low, the robot is programmed to lower its arm, at which point the server plays the audio "That was a good guess, try something lower. Make a guess." In the other case, the robot raises its arm and the sound server plays the sound "That was a good guess, try something higher. Make a guess. " If the child guesses the correct number, the robot says "That's correct, way to go!". Next, the Macarena is played from the sound server and the robot dances to the music. The architecture of the system is given in Figure 1.

### B. Incorporation of Speech

Because the Webots simulator does not support audio input, we had to program this part of the project independently and run it simultaneously. The robot has speech capabilities outside of the simulator, so our aim here was to imitate that capability. We programmed a simple server in python. When the simulator client first connects to this host, the initial game message is given. In this message, the rules of the game are explained as presented in the Game Loop section. Afterward, when the participant presses a button, another message is sent to the server indicating that the participant is ready. After adjusting to the background noise, the server listens for speech. The server continues listening until the amplitude of the sound reaches the environment conditions before the child started speaking. This recording is then used to make the API request. When the request yields a response, this response is sent to the simulator for the robot to perform the appropriate up, down or win motion. The message is also evaluated in the server to make the robot speak either an encouraging phrase for the child to try again, or inform the child that the guess was correct. In the latter case, the music audio file is played. In the simulator side, the robot does the corresponding action.

### C. Preparing and Interfacing the Motions

Choregraphe was used for preparing and interfacing the motions. Values of all joint angles at timestamps were recorded. To do this, separate up and down movements of the robot arm were modeled manually in the Choregraphe GUI interface. The values of the joint angles were obtained through the node available in the Choregraphe for the dance motion. The obtained data were extracted as a Python file. This file as it is does not work with the Webots simulator. However, it contains three lists:

- Name of Each Joint
- Time Stamps for Each Recording Taken
- Recording of the Angle Values of Each Joint

These lists are extracted from the python file by means of file processing and a CSV file compatible with webots is created. In summary, the values of the joint angles obtained by both manual manipulation and pre-existing motion nodes in Choregraphe, we created three motion files compatible with Webots. Using these motion files, the NAO robot performs the desired movements in the Webots environment.

## IV. Evaluation in Terms of HRI

In this section, we focus on how this project relates to the principles we discussed in class.

### A. Proximity

Children are known to learn better from 3D interaction, as opposed to a purely virtual interaction. This is why we chose an physical robot. The form of interaction as we will get to it, is speech. To be able to interact with the robot by means of speech, the child must be close to the robot. Although the robot is in simulation for development purposes, if this project were to be realized, we would expect the robot and the child to be proximately co-located for the best experience.

### B. Type of Interaction

The addressees of the communication are a robot and a child. Therefore, the structure of the group is human-robot individual interaction.

### C. Nature of Task

The nature of the task is both social oriented and edutainment oriented. The robot we designed is a social robot. A social robot can be defined as an autonomous robot that communicates with other autonomous entities (humans) by adhering to social rules associated with its role. The child has the opportunity to interact with an autonomous agent and take turns with it, this can potentially teach the child turn-taking and patience skills. At the same time, we expect that this game contributes to the mathematical development of children. With the robot's inputs, the child is expected to learn relationships between numbers such as greater than, less than, and equal to.

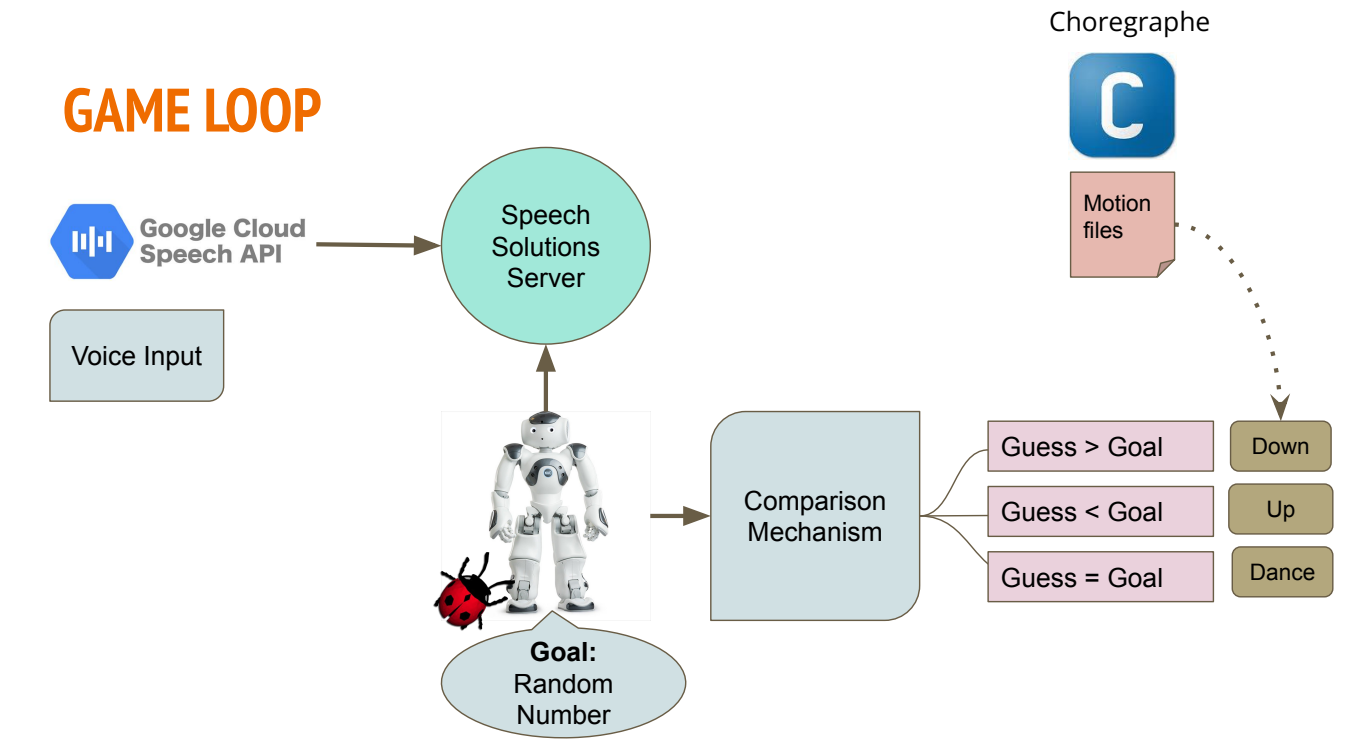


Fig. 1. Architecture of the system. Components include: Robot Simulator, Choregraphe Motion Translation, STT & TTS Support.

### D. Duration of Interaction

The child is playing a turn taking game. This game can finish very quickly if the child is very lucky, but on average, the game should take over a few minutes as the child thinks for an answer and watches the robot's feedback for each turn. Because this is a game, the children playing can keep coming back for more runs without getting bored. Perhaps they'll join the robot in the Macarena dance. This characteristic of the developed project can be summarized as repeated relatively long interactions.

### E. Information Exchange

Another important component between human and robot interaction is information exchange. Efficient interactions are characterized by productive exchanges between human and robot. In our robot, the voice communication media is used in giving commands to the robot as well as receiving feedback from the robot. At the same time, the robot physically communicates by means of arm movements. The dance can also be counted as a communication of victory, or joy.

### F. Motivation of Communication

The motivation for communication is spontaneous. The child can play the game any time he/she wants. Once the game starts, the robot listens for a while, so the the child can take time to think before answering.

### G. Level of Autonomy

Traditional human-robot interaction is exclusively in the form of "master-slave" communication. That is, it pertains to commanding and monitoring. To be more thorough, the interaction model is unidirectional. The human "speaks" and the robot "listens". The robot may ask for clarification, or take the command and perform the necessary actions to realize the command. As a result, the performance of the system relies on the operator's competence with the interface of the robot. To increase system capacity, flexibility, and create collaboration, human-robot communication must be enriched and bidirectional. Our robot was developed for the purpose of an interaction model in which humans and robots communicate as peers.

### H. Embodiment

The physical embodiment of artificial agents is very important and contributes greatly to the quality of communication. The robot in our project has an anthropomorphic structure. A Anthropomorphic robot is a robot that resembles a human body. Anthropomorphic robots are especially useful in scenarios where the human body is necessary. For the purposes of this study, it was important that the child could bond with the robot and feel comfortable with it so that interaction was established. We most often speak to and hear spoken speech from humans, which is why we believed an anthropomorphic shape was essential for the purposes of our design.

### I. Affordance

Affordance is essentially what the robot is capable of doing. The affordances of the robot preferred for this study are extensive, however, we only use a narrow scope of the robot's affordances. The robot is able to move as it completes arm motions, and dance movements. Additionally, the robot can speak with the child. This can be refferred to as socialization, however, the scope of speech is not necessarily very wide. Thus, we will refer to the second importance affordance as speech.

## V. Challenges

There are some challenges that we've experienced in this study. The first major issue was the selection of the simulator. We were both working on M1 machines. Many proprietary or open source software systems do not yet support Apple silicon. Therefore, we tried but were unable to use ROS2, CoppeliaSIM, and Chorographe on our personal machines. Webots was the only only simulator that we were able to find that was both capable of carrying out (at least most of) our tasks and was functional with our computers.

The second issue was that of speech recognition. Although NAO can speak and recognize speech, the simulators we saw do not provide support for audio communication with the virtual twin of the robot. We decided to solve this issue by carrying the speech recognition to the cloud. However, we were faced with another issue. The simulator does not allow for busy waiting, and cloud API calls tend to have a significant amount of latency. It takes a while to connect to the Google API to define speech, convert from voice to text, and also text to voice and send the result to the server. The Webots simulator threw an error when we tried threading and the sleep method of the time module in Python. This is why we had to implement busy waiting loops after making a request to the server via socket programming to solve this issue.

Finally, the motion files were not available as ready files. Webots also did not offer a facility to create a motion file. There was no GUI, only joint names and numbers were very difficult to interpret. We have solved this by using Cherographe on another device and exporting the necessary files. Then we manipulated those files to match the Webots motion files.

## VI. Future Work

In the future, multiple capabilities could be added to this project:

- A more complex game loop which allows children to stop playing when they are bored and initiate more games back to back. (i.e. The robot asks the child to play again. )
- More forms of communication. The child should be able to communicate the numerical inputs to the robot by means of showing fingers, or writing the number down on a card, or maybe choosing a card with numbers pre-written. This way, the child can be more immersed in terms of sensory stimulants and have a better learning experience.
- The roles of the child can be reversed in the game so as to provide a form of reinforcement.
- The speech recognition is not fool-proof currently. So the robot does not yet know how to respond to non-numerical inputs. This can be established by making the robot ask the child to make a numerical guess when the input is not in the necessary format.

## VII. Conclusions

To sum up, a peer-to-peer Guess the Number Game was simulated using the NAO robot as a social agent. This game is designed to support both the mathematical and social abilities and developments of children. The project was implemented using Python, Webots, and Choregraphe.

## References

[1] C. Lytridis, C. Bazinas, G. A. Papakostas, and V. Kaburlasos, "On measuring engagement level during child-robot interaction in education," in *Robotics in Education*, M. Merdan, W. Lepuschitz, G. Koppensteiner, R. Balogh, and D. Obdržálek, Eds. Cham: Springer International Publishing, 2020, pp. 3–13.