SPEA-K:

Speech Experiments with Audapter - Kids

SPEA-K design by Frances Ingram-Bate and Ludo Max

SPEA-K programming and documentation by Frances Ingram-Bate, Kwang S. Kim, Hantao Wang, and Elise LeBovidge

SPEA-K requires an install of the Audapter software by Shanqing Cai et al.

User Manual

Version 1.0.0

September 2021

Contact: MaxLab@uw.edu

Manual Contents

0. Using SPEA-K and Audapter in published work

If you publish research that used SPEA-K, please cite the software as follows:

Ingram-Bate, F., Kim, K. S., Wang, H., Max, L. (2021). SPEA-K: Speech Experiments with Audapter – Kids [software]. GitHub repository, https://github.com/MaxLabSoftware/SPEA-K

Whenever you cite SPEA-K, you should also cite Audapter. The Audapter manual requests that you cite:

Cai, S., Boucek, M., Ghosh, S. S., Guenther, F. H., Perkell, J. S. (2008). A system for online dynamic perturbation of formant frequencies and results from perturbation of the Mandarin triphthong /iau/. In *Proceedings of the 8th International Seminar on Speech Production*, Strasbourg, France, pp. 65-68.

Tourville, J. A., Cai, S., Guenther, F.H. (2013). Exploring auditory-motor interactions in normal and disordered speech. *Proceedings of Meeting on Acoustics*, *19*(1), 060180. https://doi.org/10.1121/1.4800684

1. Overview of SPEA-K and Audapter

SPEA-K is a freely available MATLAB program that allows users with or without programming experience to test children in speech experiments with auditory feedback perturbations. This is achieved by (a) letting the user configure all details of the experimental manipulations and elicited speech utterances in a graphical user interface (GUI) and storing the selected settings in experiment configuration files, (b) eliciting speech from the child during an interactive "board game" designed to run on a touchscreen monitor, and (c) implementing auditory feedback manipulations of the child's speech with the freely available Audapter MATLAB program previously developed by Shanqing Cai and others.

SPEA-K is used to *design* and *administer* the experiment, but is currently not optimized to *analyze* the obtained data. Nevertheless, the results of Audapter's online formant and fundamental frequency tracking (i.e., Audapter's own real-time acoustic measures of the participant's speech) are available to the experimenter after the session. For final data preparation, we recommend additional, rigorous acoustic analyses and visual inspection of all individual tokens.

Although the user interacts only with SPEA-K – while Audapter runs all data acquisition and processing "under the hood" – we strongly recommend familiarizing yourself with the technical details of Audapter itself if possible. A manual for Audapter is available at http://sites.bu.edu/guentherlab/files/2016/11/AudapterManual.pdf.

**WARNING*: It is important to note that SPEA-K was created to make it feasible for a greater number of speech researchers and clinicians to conduct auditory-motor learning studies. For this reason, the program allows users to configure a wide variety of parameters through its simple GUI format. However, this convenience comes at the cost of some loss in optimization. That is, although all general aspects of the experiment are configurable in this manner (e.g., experimental phases, number of trials, type of perturbation, etc.) not all input parameters to the Audapter signal processing software are configurable in the current release version of SPEA-K. We consider this to be a reasonable trade-off as even some advanced users of Audapter often leave at least some of those input parameters set to their default values. One example is the

nDelay parameter; but see Kim, Wang, & Max (2020) for how fine-tuning of this parameter can reduce the feedback delay caused by signal processing in Audapter:

Kim, K. S., Wang, H., Max, L. (2020). It's about time: minimizing hardware and software latencies in speech research with real-time auditory feedback. *Journal of Speech, Language, and Hearing Research, 63*(8), 2522–2534. https://doi.org/10.1044/2020_JSLHR-19-00419

2. Downloading and installing SPEA-K and Audapter

2.1. Hardware and software requirements

Basic hardware requirements include a microphone connected to a *microphone in* channel of a computer audio interface, a *line out* channel of the audio interface connected to a headphones amplifier or mixer, and a pair of headphones or insert earphones. The audio interface typically uses USB to connect with a recording computer (notebook or desktop). High-quality audio equipment is of course strongly recommended. It should be noted also that the choice of audio interface can greatly affect the total latency of the auditory feedback loop from microphone pickup to headphones/earphones playback. For “real-time” auditory feedback, we currently recommend the RME BabyFace Pro audio interface or the newer BabyFace Pro FS. Latency testing results for a few common audio interfaces are available in Kim et al. (2020; see citation above).

Basic software requirements for SPEA-K include an installation of both MATLAB and Audapter in the Windows operating system. At this time (September 2021), we have used and tested SPEA-K only on computers with the Windows 10 operating system and MATLAB R2020a, R2020b, or R2021a. Audapter should be installed following the instructions included in the Audapter manual. Audapter will then be in the local directory C:\speechres. SPEA-K expects to find Audapter in this particular directory.

Regarding audio drivers, it is technically possible to use the free, hardware independent ASIO4ALL driver. However, in order to minimize the hardware latency (and thus also the total feedback loop latency which includes both hardware latency and software processing latency), it is very strongly recommended to operate the available audio interface with its own manufacturer's default ASIO driver (see the above Kim et al., 2020 citation for more details regarding driver effects on feedback delay). If the interface’s specific driver is installed, SPEA-K provides the option to select it as part of creating an experiment configuration file.

2.2. Obtaining and installing the software

MATLAB can be purchased from The Mathworks at https://www.mathworks.com. If you are affiliated with an academic institution, MATLAB licenses may be available through your IT support. Install MATLAB following the instructions provided by The Mathworks or your institution’s IT support.

Audapter is available at no cost from GitHub: https://github.com/shanqing-cai/audapter_MATLAB. An Audapter manual is available at no cost from http://sites.bu.edu/guentherlab/files/2016/11/AudapterManual.pdf. Install Audapter following the instructions provided in the manual.

SPEA-K is available at no cost from GitHub: https://github.com/MaxLabSoftware/SPEA-K. There is no installation required, but the cloned (or downloaded) SPEA-K directory as well as its

subdirectories must be added to MATLAB's path. For those new to MATLAB, this can be accomplished by navigating to the Home tab. Under the Environment header, you can open the Set Path window and click on "Add with Subfolders…" (see Figure 1). Navigate to your SPEA-K directory, select the entire folder, and then save and close this Set Path window. Now SPEA-K commands can be used in the MATLAB Command Window.

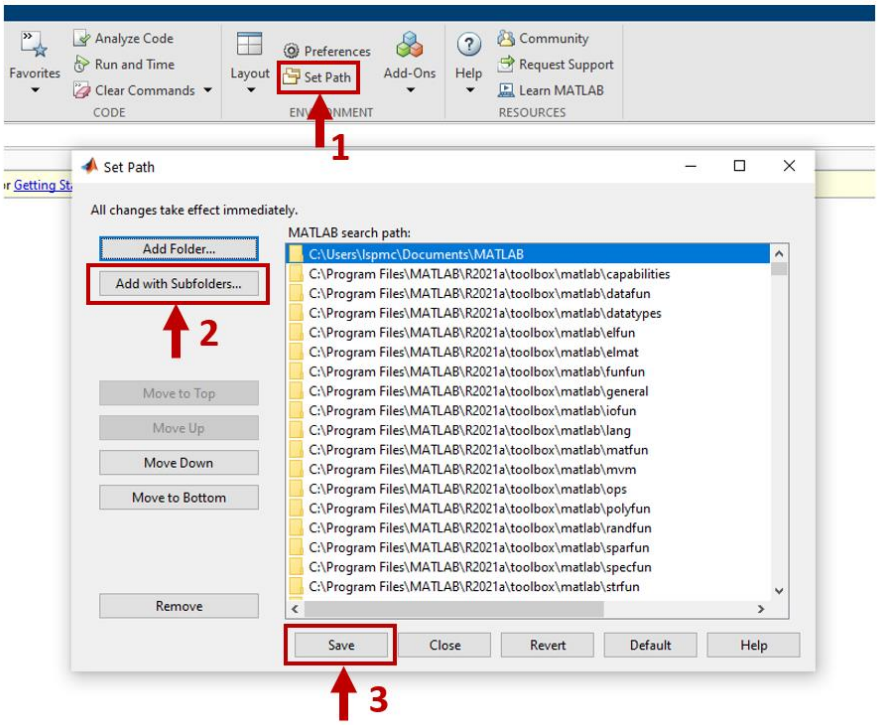

Figure 1. Adding SPEA-K to the MATLAB path. First, the Set Path window needs to be opened using the shortcut under MATLAB's Home tab (1). Then the SPEA-K directory can be added (with subfolders!) (2) before these settings are saved (3) and this window is closed.

2.3. Starting the software

SPEA-K can be started by typing the command `SPEAK` into the MATLAB workspace and pressing the Enter key on the keyboard. This assumes that all software is correctly installed, and the SPEA-K directory and subdirectories are already added to MATLAB's path.

Doing so will bring up SPEA-K's start window which provides options to either start designing/configuring a new experiment or to run an experiment with a previously saved configuration file (Figure 2).

When choosing the first option ("Create a New Experiment Design"), the user is presented with a new graphical user interface (GUI) where various aspects of the experiment can be easily configured (see *Section 3: Configuring an experiment in SPEA-K* below). When the user is finished making selections for each group of configuration parameters, clicking *Save and Exit* will save all settings to a .mat file and return the user back to the start window.

The second option in this start window ("Load a Previously Created Design") is to run an experiment with parameters that had been previously configured. Just load a previously created

.mat file that contains configuration settings, and SPEA-K's board game window will appear to initiate the speaking game.

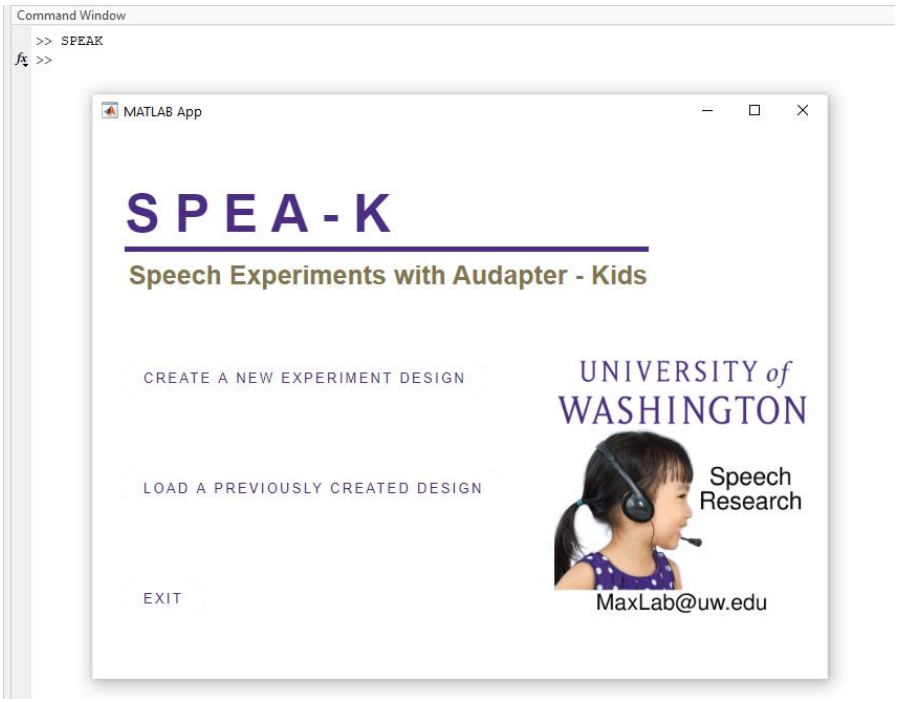

Figure 2. Running SPEA-K by typing the command `SPEAK` into the MATLAB workspace and pressing the Enter key on the keyboard.

3. Configuring an experiment in SPEA-K

Prior to starting the design of a new experiment in SPEA-K, the user should decide on, among other things, which auditory perturbation to implement (e.g., formant shift up or down, gradual or sudden introduction of the perturbation, perturbation size), how many trials should be included in each phase of the experiment (e.g., baseline, ramp, hold, wash-out), which test words to use, etc.

The user should also save to their local hard drive a set of pictures (e.g., downloaded from the internet): one picture per target word. For example, if the child should be prompted to produce 30 trials of “bus”, 30 trials of “bug”, and 30 trials of “duck”, then the user will need to supply one .jpg image of a bus, one of a bug, and one of duck, with the filenames *bus.jpg*, *bug.jpg*, and *duck.jpg*. SPEA-K will use the image filenames to label each image, so this naming convention is necessary.

Within SPEA-K, all experimental parameters are then entered in a user-friendly GUI, without the need for any MATLAB programming. The settings are saved in a configuration file (.mat) that can be loaded for each participant in the study. SPEA-K was designed to run multiple participants using the same configuration file, and will automatically create subject folders and organize the experimental data based on the subject ID that is provided at the beginning of each launch of the game. If the same subject ID has previously been entered for the same experiment, a new folder will still be created (i.e., no data will ever be overwritten). Note that

different configuration files can be created to collect data for multiple conditions. In addition, a shorter configuration file can be created to be run as a “practice block” to explain the instructions to the participant and allow them to familiarize themselves with the interface.

The configuration GUI contains various sections (organized as separate window tabs) for which information is entered in text boxes or selections are made from drop-down menus. The following sections are included.

3.1. Selecting an audio driver (“Audapter” tab)

After running SPEA-K and selecting “Create a New Experiment Design”, the user is first prompted to select the audio driver that they want to use (Figure 3). The choice of audio driver affects the behavior of Audapter, the perturbation software that runs behind-the-scenes. Of particular relevance for speech auditory-motor learning experiments is the feedback delay inherent in the experimental apparatus. Using the recommended audio driver for your audio interface is therefore critical for optimum performance (see the above Kim et al., 2020 citation for more details regarding driver effects on feedback delay).

Pressing the “Find audio driver(s)” button will populate a list of available drivers to select. Once the correct driver is selected, press “SAVE and NEXT” to move to the next step of experimental configuration.

**NOTE*: We want to reiterate that it is very strongly recommended to operate the available audio interface with its own manufacturer's default ASIO driver. If the interface’s specific driver is installed, SPEA-K will find the driver during this step of experimental configuration. However, if the interface’s specific driver is not installed, it will NOT appear in this list.

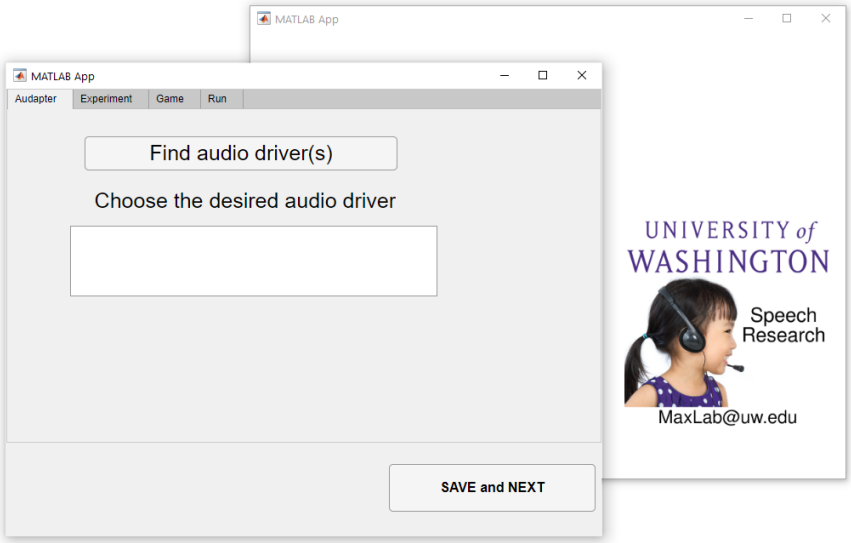

Figure 3. SPEA-K’s audio driver selection window.

3.2. Perturbation configuration ("Experiment" tab)

After selecting the audio driver, the user is prompted to supply the perturbation details (Figure 4, top). You may select one of three types of auditory feedback perturbation:

- Formant shift = applies a frequency shift to formants 1 and 2 (F1 and F2) of the participant's auditory feedback
- Pitch shift = applies a frequency shift to the fundamental frequency (F0) of the participant's auditory feedback
- Delay = applies a temporal manipulation by delaying the participant's auditory feedback

For formant shifts, the user may then further specify the units, direction, and magnitude of the desired perturbation. The available perturbation units are Hertz (Hz) and mels in absolute terms, or cents and percentage in relative terms (i.e., as a percentage). The magnitude and direction of the F1 and F2 perturbations can be separately adjusted. To apply a downward shift, simply place a negative sign (-) in front of the value.

For pitch shifts, the perturbation can only be applied in relative terms (cents or percentage). To apply a downward shift, simply place a negative sign (-) in front of the value.

For an applied delay, the perturbation magnitude will be in milliseconds (ms).

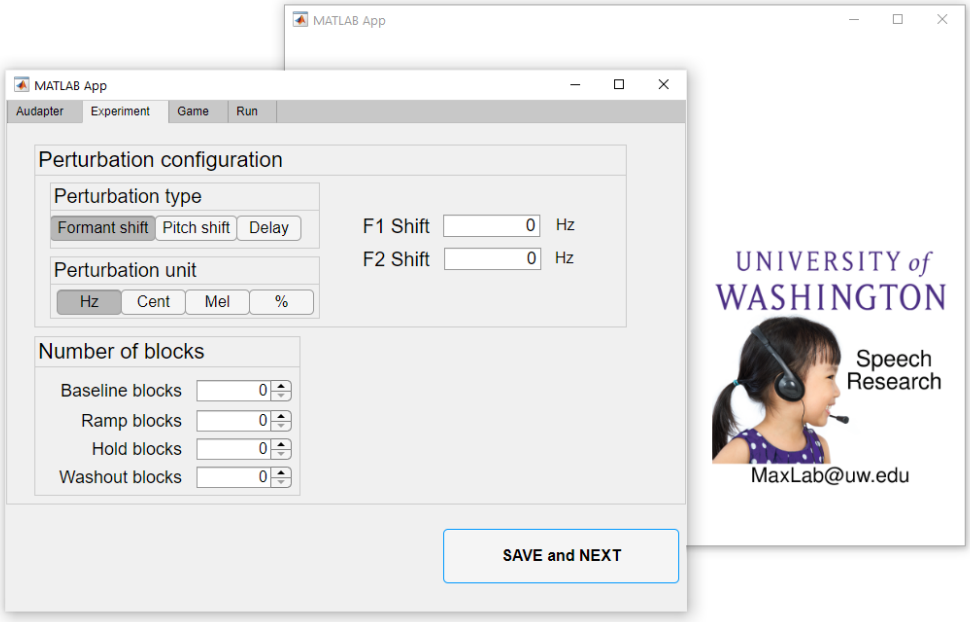

Figure 4. SPEA-K's "Experiment" tab with the perturbation and experimental phase settings.

3.3. Number of blocks ("Experiment" tab)

Here the user can specify the desired perturbation phases of the experiment, as well as the desired number of blocks for each (Figure 4, bottom).

One block includes one repetition of each of the target words. In other words, if you have 5 target words (i.e., 5 different images for the participant to name), one block would equal 5 trials – a complete set of one repetition of each image.

The four possible experimental phases are:

- Baseline = a pre-perturbation phase under typical auditory feedback
- Ramp = a perturbation phase that evenly ramps up the perturbation magnitude over a set number of blocks
- Hold = a perturbation phase that applies the full perturbation magnitude
- Washout = a post-perturbation phase under typical auditory feedback, typically used to monitor the participant's return to baseline (or lack thereof) after auditory-motor learning

The following two examples are included for clarity.

Example 1: The user would like the participant to produce 30 repetitions of each word, all under a delay of 50 ms.

- After setting the delay to 50 ms under *Perturbation type*, the user would input 30 into the *Hold blocks* cell. All other cells should remain at 0 since the user does not want any trials to be produced under typical auditory feedback (e.g., baseline or washout blocks), nor do they want the delay to be slowly increased over time (e.g., ramp blocks).

Example 2: The user would like the participant to produce 15 repetitions of each word under typical feedback before increasing the amount of F1 shift over the next 15 repetitions until reaching a maximum of +50 Hz that is maintained for another 30 repetitions. They would also like a washout phase of 15 repetitions to mirror the baseline.

- After setting the first formant shift to +50 Hz under *Perturbation type*, the user would input 15 into the *Baseline blocks* cell, 15 into the *Ramp blocks* cell, 30 into the *Hold blocks* cell, and 15 into the *Washout blocks* cell.

Once the perturbation settings are finalized, click on "SAVE and NEXT" to move to the next step of experimental configuration.

3.4. Choosing the stimuli: uploading game pictures ("Game" tab)

In this tab, the user will select the directory where all of the image files are saved (Figure 5). The files should be in a standard format for images (e.g., JPEG or PNG).

As previously mentioned, SPEA-K is programmed such that the filenames of the images are used to display the target word to be produced above each image. Therefore, if the experimenter would like the participant to produce the words "bus", "bug" and "duck", the three images should have the corresponding filenames of *bus.jpg*, *bug.jpg*, and *duck.jpg*.

After the directory is selected, a sample picture will appear in the GUI. If this image looks correct, the user can click on “SAVE and NEXT” to export the .mat configuration file.

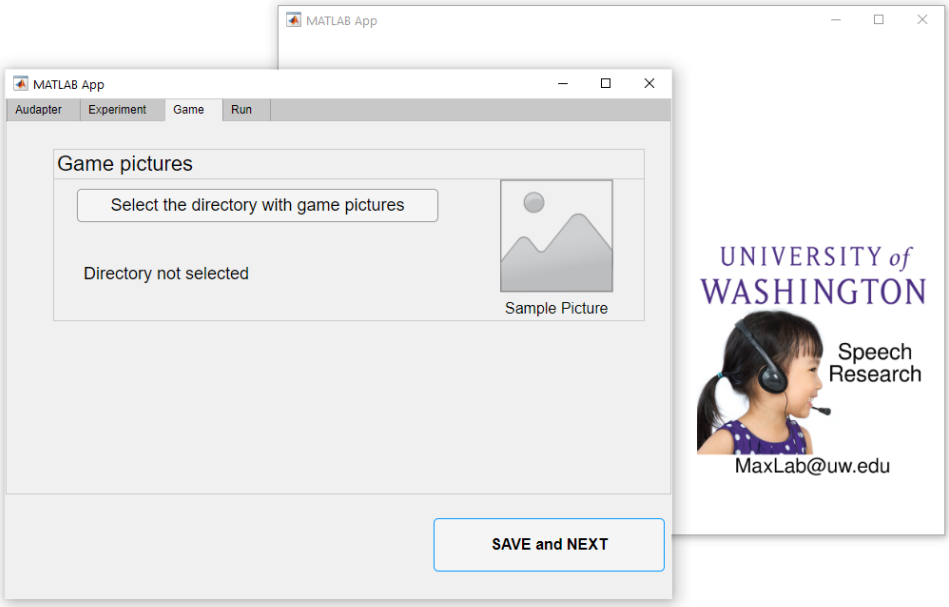

Figure 5. Selecting the directory where all target word image files are saved.

3.5. Saving the configuration file (“Run” tab)

At this point, the configuration file is complete! The user has two options moving forward: simply saving the configuration file or saving the file and running the game.

If you select “SAVE the config file”, you will be prompted to name your file and select where you’d like the file saved.

If you select “SAVE and RUN SPEAK”, you will be prompted to name your file and select where you’d like the file saved. In addition, the game will automatically open with your newly created configuration file!

4. Playing the board game

4.1. Opening the game

Once the user has a configuration file created and saved, they can load the file to initiate the board game (see *Section 2.3: Starting the software*). Alternatively, the game can automatically be started at the end of the configuration step (see *Section 3.5: Saving the configuration file (“Run” tab)*).

Before the game is started, the user will see a snapshot of the configuration file settings for verification before proceeding (Figure 6). Follow the GUI instructions (“*Do you want to run the experiment with this configuration file?*”) to launch SPEA-K.

You will first be prompted to enter a subject ID before the game is launched. SPEA-K was designed to run multiple participants using the same configuration file, and will automatically create subject folders and organize the experimental data based on the unique subject ID. If the same subject ID has previously been entered for the same experiment, a new folder will still be created (i.e., no data will ever be overwritten). Note that different configuration files can be created to collect data for multiple conditions. In addition, a shorter configuration file can be created to be run as a "practice block" to explain the instructions to the participant and allow them to familiarize themselves with the interface.

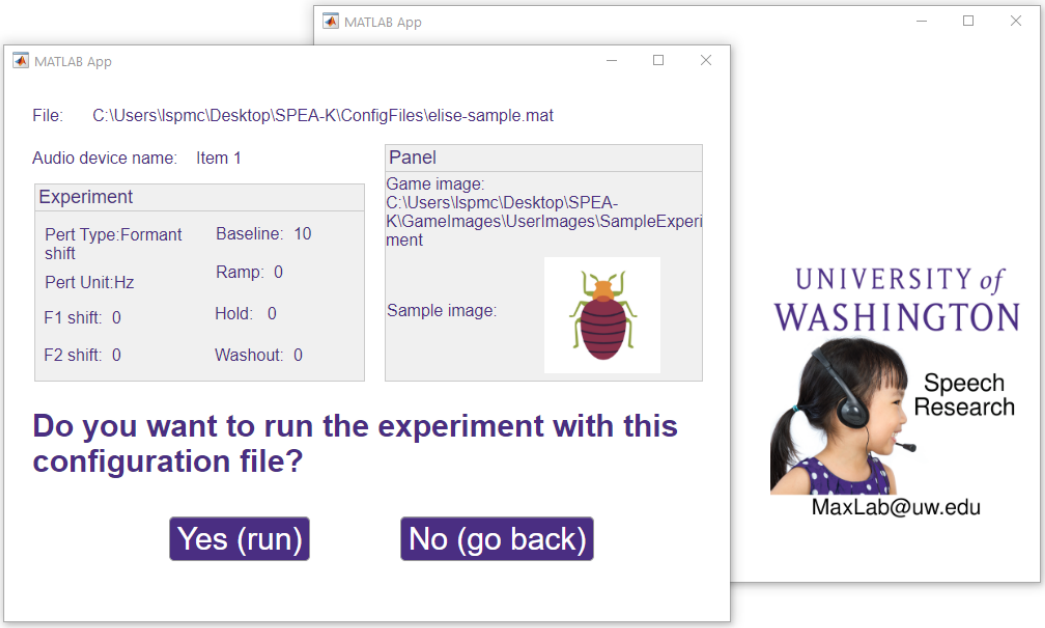

Figure 6. Confirmation window to verify the configuration file prior to launching the game.

4.2. Game overview

The objective of the game is to move along the track and reach the end. As of now (September 2021), the only game layout is a dinosaur who moves along the track, collecting its eggs on the way back to its nest. Future releases of SPEA-K can offer other game layouts. The current game design is based on the actual cardboard game used in Kim et al. (2020):

> Kim, K. S., Daliri, A., Flanagan, R., Max, L. (2020). Dissocaited development of speech and limb sensorimotor learning in stuttering: Speech auditory-motor learning is impaired in both children and adults who stutter. *Neuroscience, 451*(15), 1–21. https://doi.org/10.1016/j.neuroscience.2020.10.014

To play the game, the participant (usually a young child sitting beside the experimenter) will sift through the "cards" in the middle of the board to find the image that matches the tile directly in front of their character piece. As mentioned above, the default piece is currently a dinosaur. After a match is made, the dinosaur will advance one space forward on the track. As more matches are made, the dinosaur will continue moving along the track, collecting its eggs, to eventually reach its nest and end the game.

Given that the software was created for speech auditory-motor adaptation experiments, the game is designed such that each image/card is named aloud as the participant "flips" through them. Based on the settings that the experimenter selected in their configuration file, the number of images in the stack (as well as the length of the game board's track) will vary. The software is designed to always yield the correct number of trials as specified in the configuration file. In addition, the participant will always reach the end of the track. In other words, the game is rigged such that the participant will always win.

4.2. Game controls

To interact with the game, the user (or participant, depending on their age, ability, and the experimenter's preference) will use a touchscreen monitor to make selections. Given that selections are made in a MATLAB window, SPEA-K can also be played using a mouse or trackpad.

To begin the experiment, "START GAME" must be pressed (Figure 7). Pressing this button brings up the first trial and begins data acquisition. The experimenter may leave the window on this opening screen prior to the participant's arrival to expedite data collection. Once everyone is ready to begin and the instructions are clear, "START GAME" can be pressed. We suggest creating a shorter configuration file with different target words to run first to allow the participant to familiarize themselves with the interface and the task instructions before diving directly into the experimental session.

Figure 7. The board game window after launch, using an example configuration file. The length of the track will vary depending on the number of blocks and the number of images selected during the configuration process.

The experiment will begin by presenting the first image card in the middle of the board. The participant should produce each word (i.e., name each image) as they appear in the middle of the board. After each word is produced, there are two options to continue to the next trial:

- If the image matches the tile directly in front of the dinosaur, the participant may click on the icon on the track to identify the match (Figure 8). Doing so will advance the dinosaur one tile forward on the board.

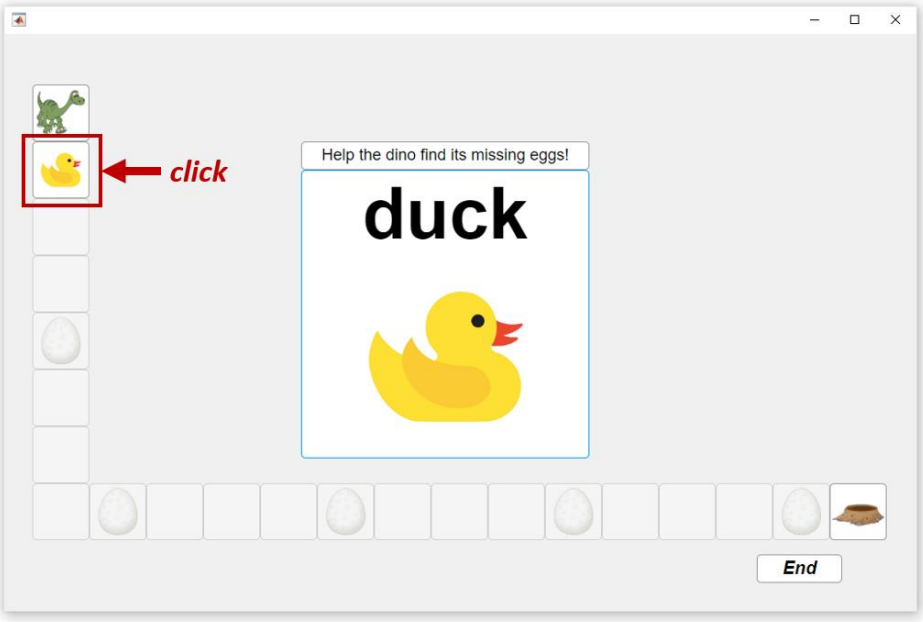

Figure 8. An example trial where the image matches the tile directly in front of the dinosaur. The participant would need to click on the icon on the track to identify the match and advance the game piece forward.

- If the image does NOT match the tile directly in front of the dinosaur, the participant can continue "flipping through" the stack of images by clicking on the card in the middle of the board (Figure 9). The participant must continue producing each word and clicking through the images until a match is made.

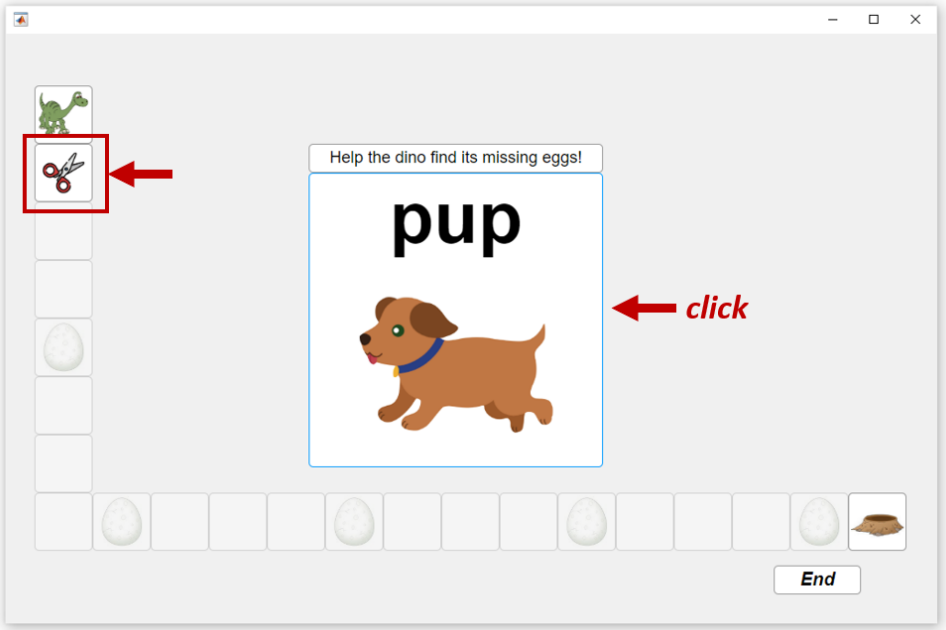

Figure 9. An example trial where the image does NOT match the tile directly in front of the dinosaur. The participant would need to click on card in the middle of the board to continue searching for matches.

Gameplay will continue in this manner until the end of the track is reached. Once the character piece has reached the end of the board, a trophy will appear to congratulate the player.

*NOTE: Upon completion, the “END” button in the bottom right of the screen should be pressed to exit the game in order to properly exit Audapter.

5. Miscellaneous notes

5.1. Data output

SPEA-K will run Audapter behind-the-scenes and save the acoustic data from each trial. This data will be saved as a .mat file under the SPEA-K folder:

[SPEA-K folder]\Experiment\[name of the configuration file]\Data\[subject code]

Data saved includes the raw acoustic data in addition to Audapter’s online formant and fundamental frequency tracking (i.e., Audapter’s own real-time acoustic measures of the participant’s speech). However, for final data preparation, we *strongly* recommend additional, rigorous acoustic analyses and visual inspection of all individual tokens.

5.2. Formant tracking optimization

Future releases of SPEA-K may include information on how to optimize the behind-the-scenes formant tracking by Audapter.