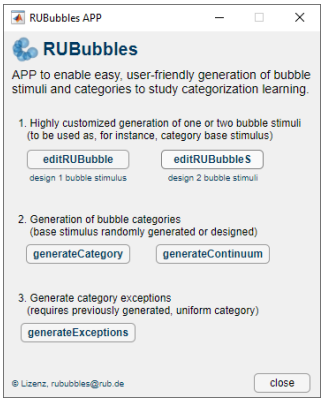
RUBubblesAPP 1.1

This app can be used to precisely customize individual base stimuli (editRUBubble, editRUBubbleS), to create full bubble categories (generateCategory and generateContinuum) and to design category exceptions (generateExceptions). It allows an easy-to-understand manipulation of stimulus parameter and contains direct visualizations of parameter changes. This ReadMe is intended to give some more detailed user instructions and explain specific app components.



Version and revision history

Version 1.1 (last update: 19.03.2021)

Code/App modifications:

- Correction of 'generateExceptions' and 'editRUBubble_EXC' (closing of window, density and size slider)

MATLAB App

To use the RUBubblesAPP as an app within your MATLAB environment download the file 'RUBubblesAPP.mlappinstall' and install it. You will then find the app in 'APPS', 'MY APPS'. It runs on MATLAB2020b. Earlier versions might lack necessary functions or features. The 'Image Processing Toolbox' is necessary to draw custom sphere positions. It is recommended to create one folder (e.g. 'bubbleAPP') and use subfolders within to save distinct bubble categories.

Standalone Desktop App

To use the RUBubblesAPP independently of MATLAB download the file 'RUBubblesAPP_Installer.exe' and follow the installation instructions. You might need to install MATLAB Runtime (Release 2020b, https://de.mathworks.com/products/compiler/matlab-runtime.html). The app can then be used irrespective of the MATLAB version or even without MATLAB. It is recommended but not obligatory to create one folder (e.g. 'bubbleAPP') and use subfolders within to save distinct bubble categories.

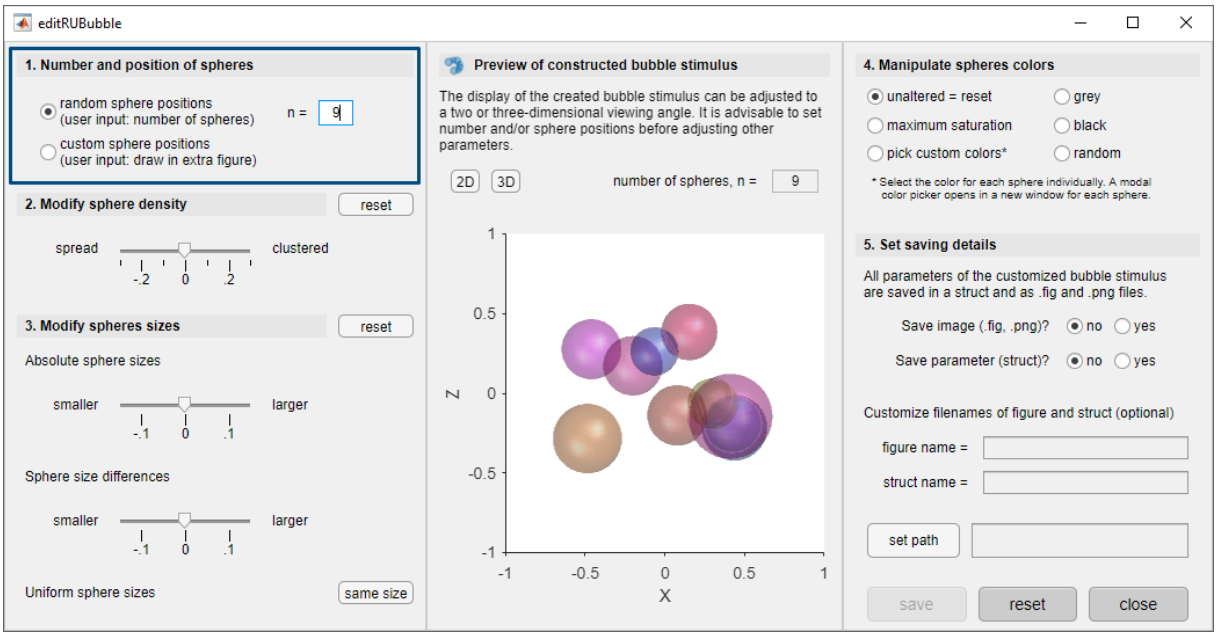
Licence and contact

Contact:
Aylin Apostel, M.Sc. Biology, Neural Basis of Learning, Department of Psychology, Institut of Cognitive Neuroscience, Ruhr-University Bochum, Universitätsstr. 150, GA 04/146, 44801 Bochum, Germany

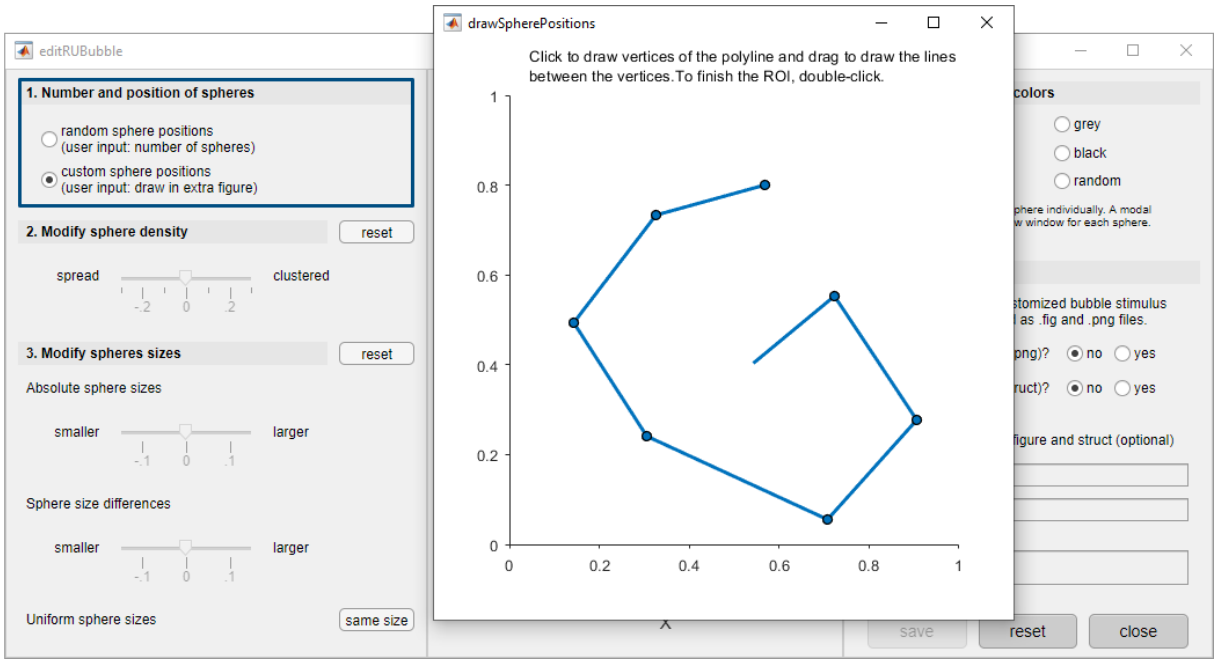
mail: rububbles-app@ruhr-uni-bochum.de

editRUBubble

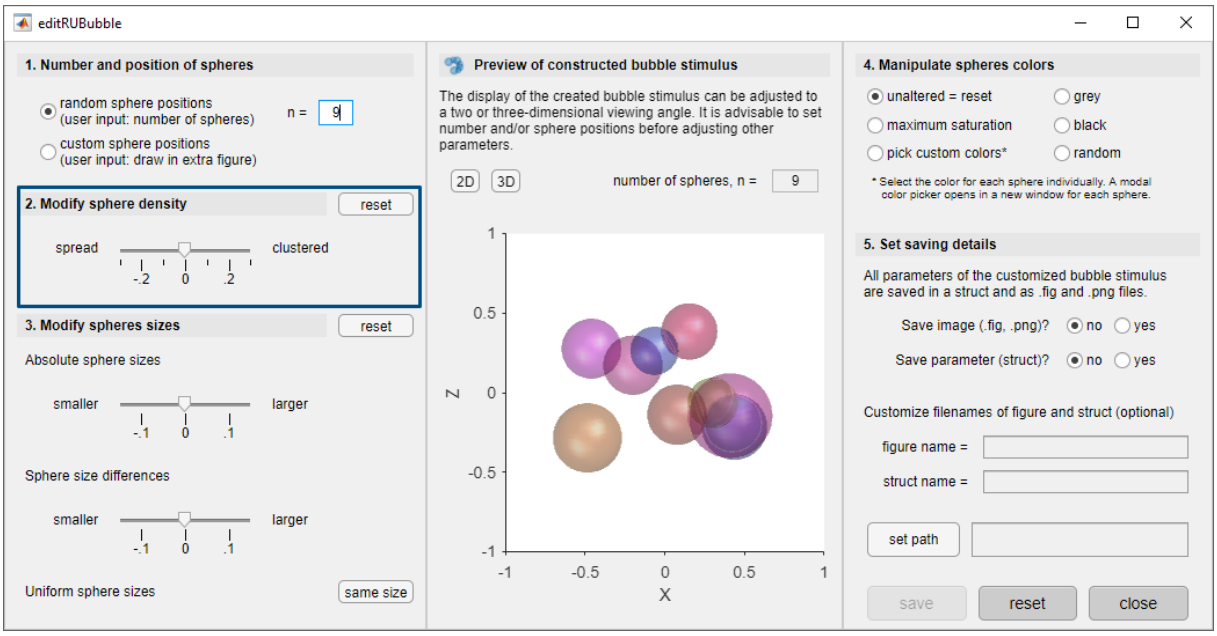
The button ‘editRUBubble’ opens a window to design one bubble stimulus. By default, the created stimulus is saved as ‘bubl’ (figure, formats: ‘.fig’, ‘.png’) and ‘par’ (struct containing all stimulus parameter). Both filenames can be modified. Before stimulus features can be customized (such as density, size, color), the number (and position) of spheres must be set. When both, absolute sphere size and sphere size differences, should be modified it is important to first change the absolute size and then size differences (order of size sliders important for proper functioning).



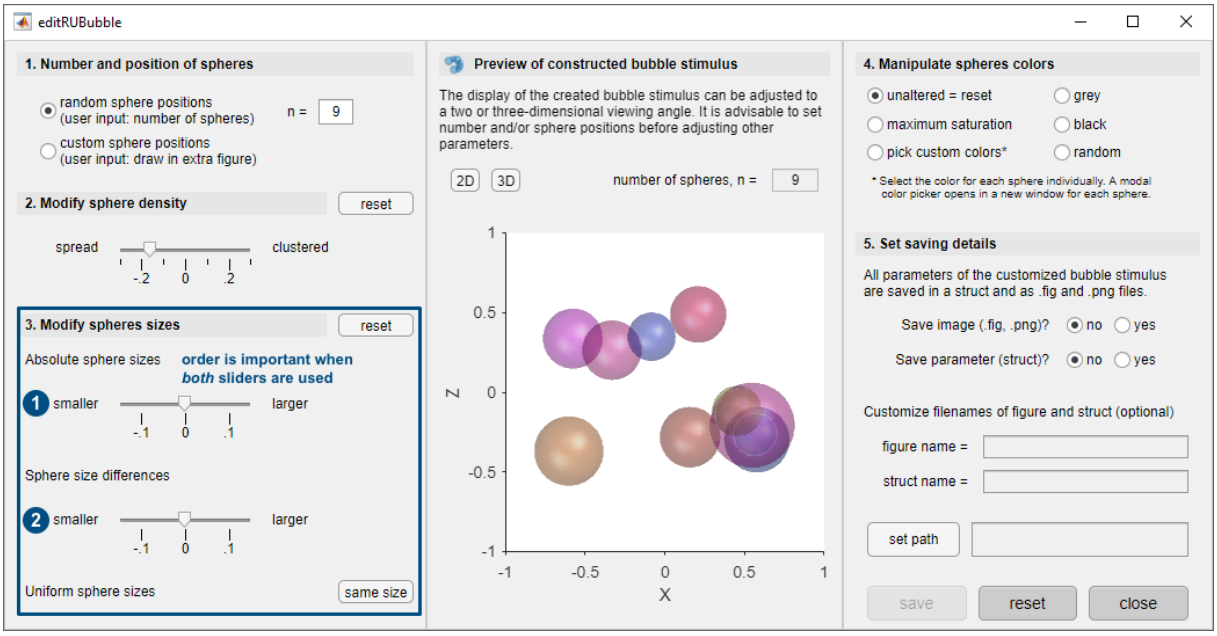
1: Random sphere positions. Only the number of spheres needs to be set. Initial values of sphere size and color are chosen randomly and can be modified subsequently.



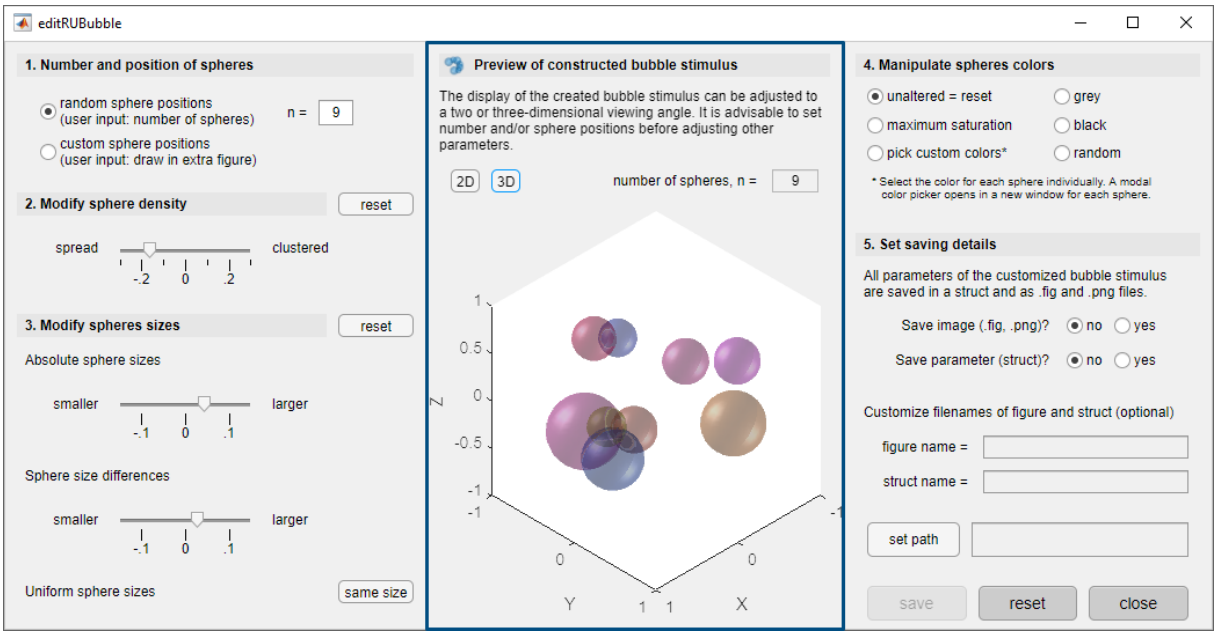
2: Custom sphere positions. A new window opens in which the user can mark sphere positions. Double-click to finish and return to the main window. The number of vertices defines the number of spheres. Sphere position is set as drawn by the user, sphere size and colors are chosen randomly and can be adjusted later. The resulting bubble stimulus is visualized in the middle panel. The ‘Image Processing Toolbox’ is necessary to use this feature within the MATLAB app.



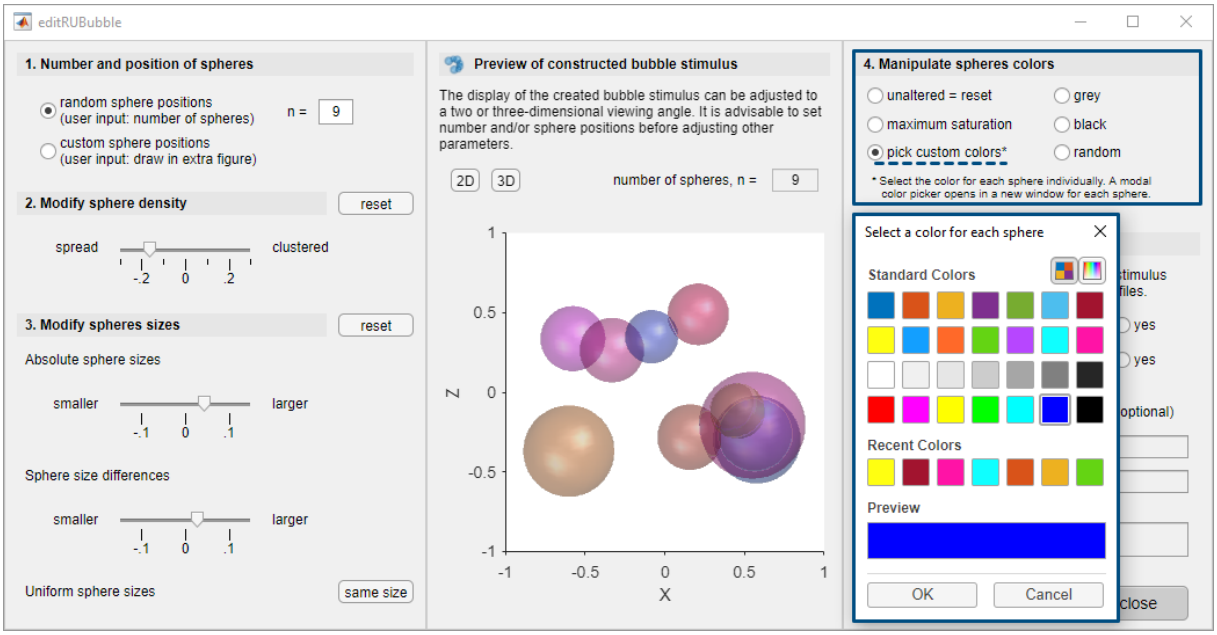
3: Post hoc customization of sphere density. Spheres can be shifted away from or towards the common center. Use the reset button next to the related heading to restore the original values.



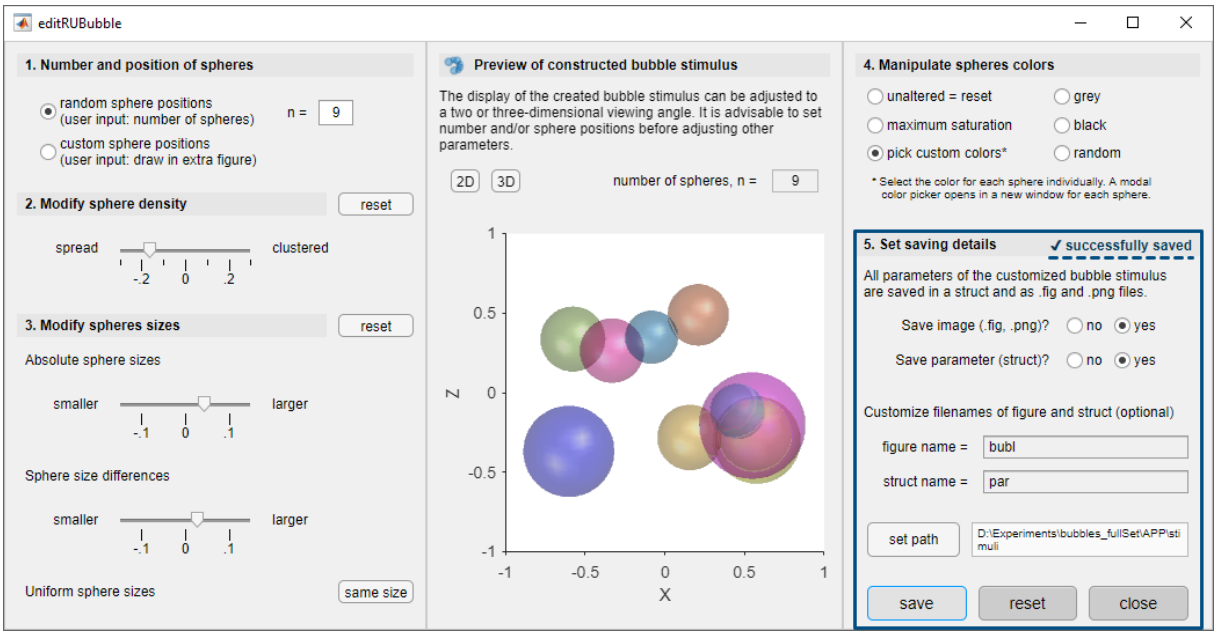
4: Two slider can be used to customize sphere sizes. For correct functioning, absolute sphere size has to be manipulated before sphere size differences (only if both size slider are used). It is further possible to set all spheres to the same size (same size button). Use the reset button next to the related heading to restore the original values.



5: The middle panel contains a preview of the constructed bubble stimulus. It can be viewed and saved as 2D or 3D image (2D image per default).



6: Sphere colors can be adjusted by setting all spheres to black or grey, to new random colors, or maximum saturation. The user can further pick the specific colors using a modal color picker (opens new selection window for each sphere). The new colors will be assigned randomly and the preview of the bubble stimulus will be updated after all novel colors were selected. Set the color selection to 'unaltered = reset' to restore the original sphere colors.

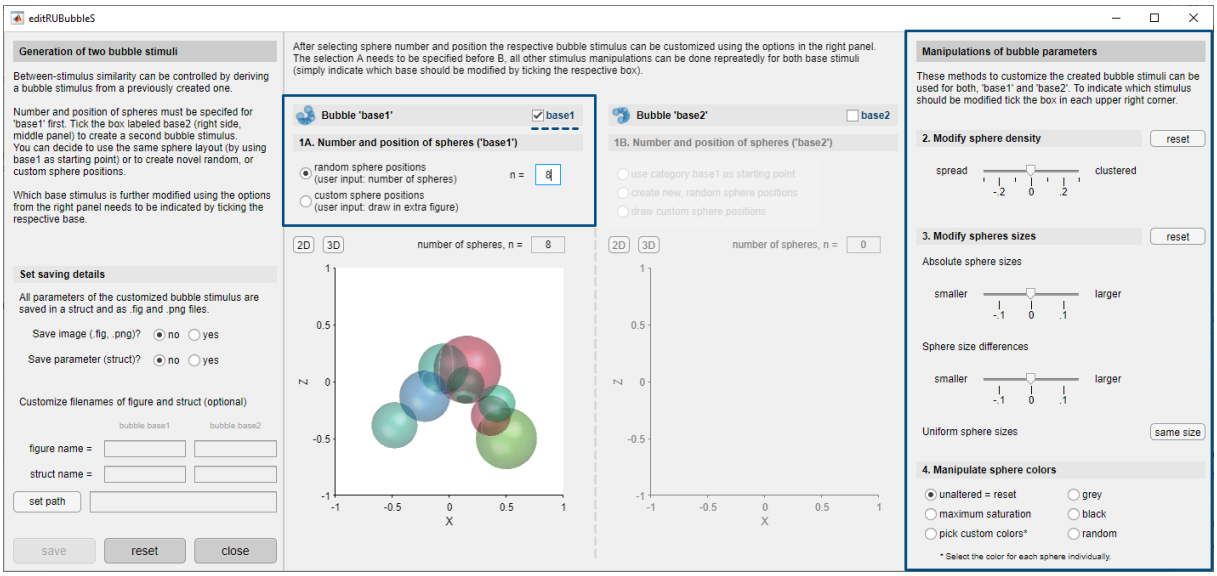


7: The designed bubble stimulus can be saved as figure ('bubl', file formats: '.fig' and '.png') and struct containing all stimulus parameter ('par'). Both filenames can be changed. If either filename already exists in the selected folder the user is warned in an extra window and asked whether the existing file/files should be overwritten. A notice next to the related heading informs the user about successful saving.

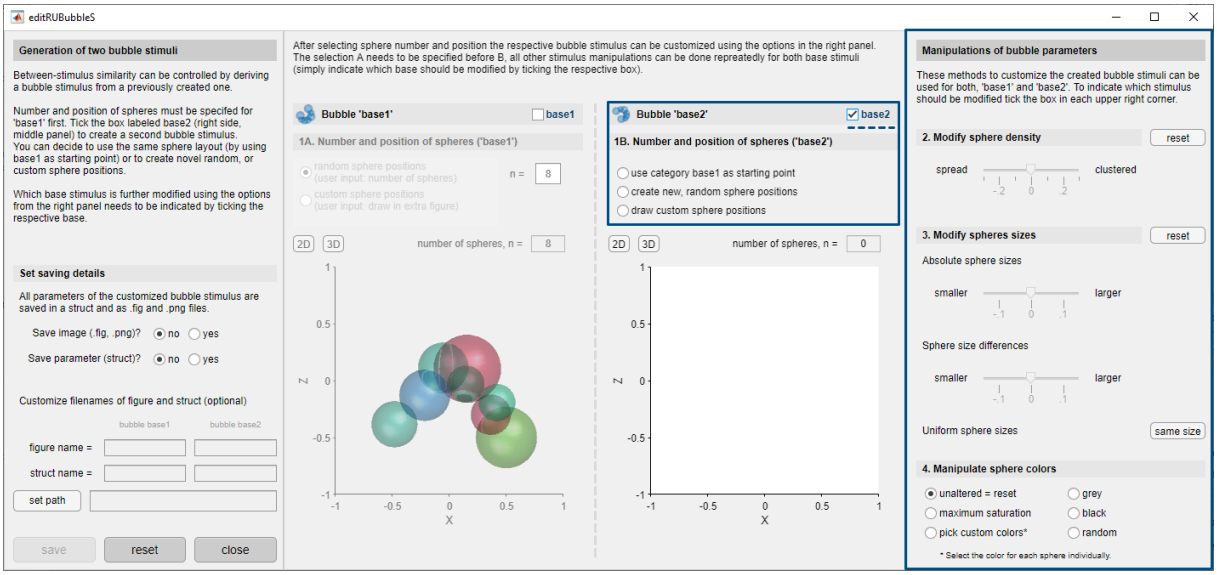
editRUBubbleS

The button 'editRUBubbleS' opens a window to design two bubble stimuli simultaneously. This allows it to adjust between-stimulus similarity (or between category similarity if both stimuli are later used as base stimuli for the generation of bubble categories). By default, the created stimuli are saved as 'bublA' and 'bublB' (figures, formats: '.fig', '.png') and 'parA' and 'parB' (structs containing all stimulus parameter). All filenames can be modified.

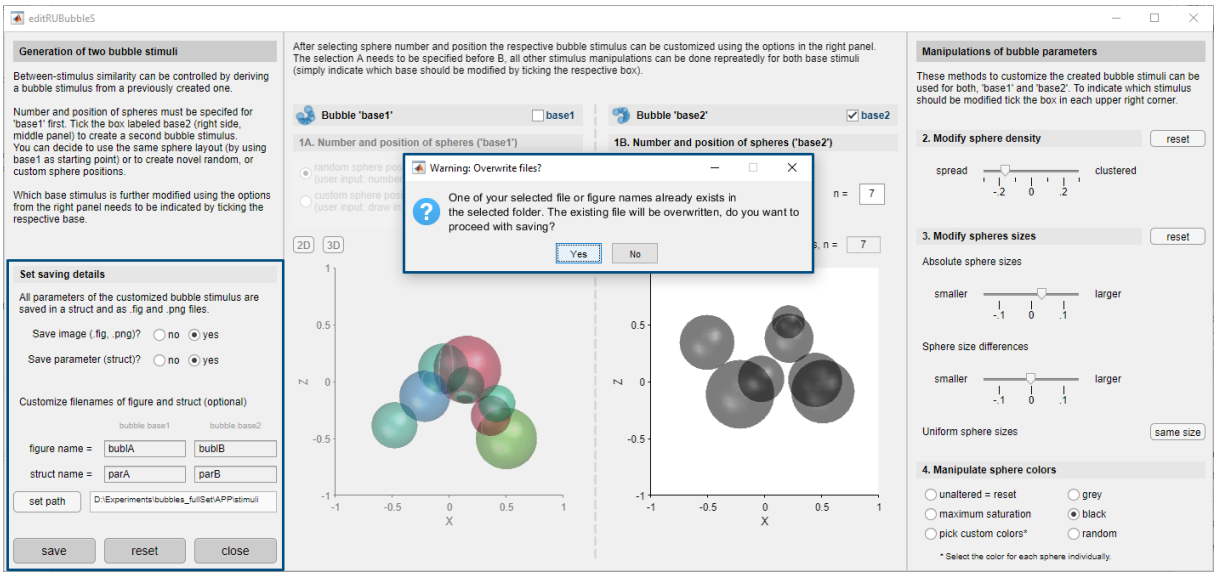
Bubble base1 has to be created first (options for customization as in editRUBubble). To start the creation of base2 tick the respective box (right side, middle panel). Base2 can now either be based on base1 or on novel (random or custom) sphere positions and then adapted.



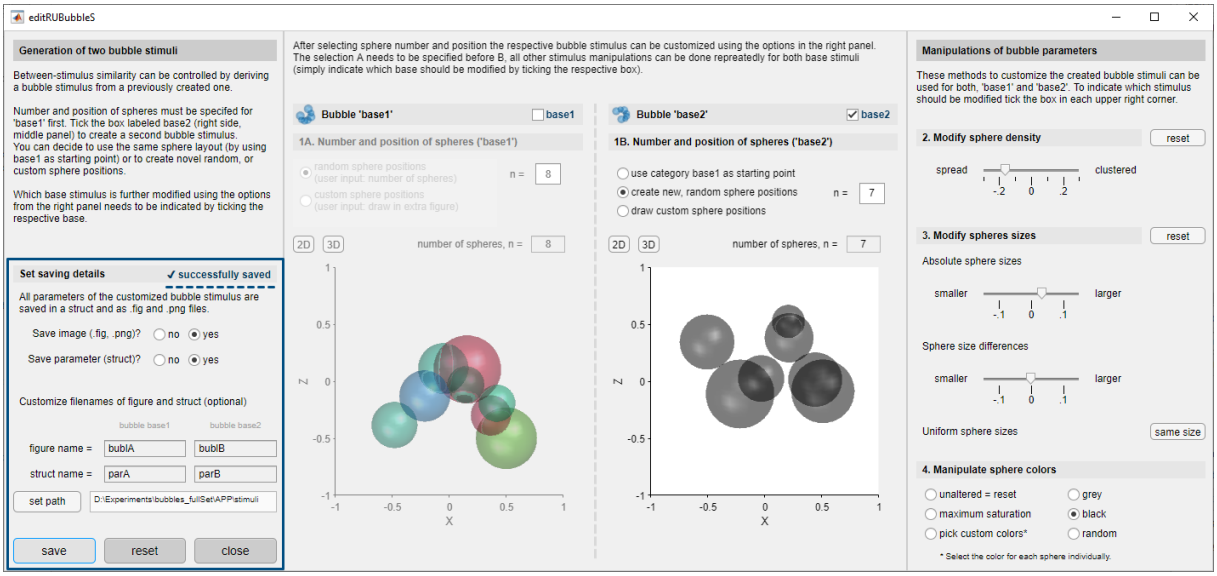
8: Creation of base1. Start by setting number and position of spheres for base1 (the user can choose between random or custom sphere positions). Thereafter, the created bubble stimulus can be customized using slider and color specifications in the right panel (functionality as in editRUBubble).



9: Creation of base2. Tick the box of base2 in order to enable the specification of sphere number and position for base2 (the user can choose between base1, random or custom sphere positions). Thereafter, base2 can be adapted as base1 using the options in the right panel.



10: The app produces a warning if the selected folder already contains a figure or struct with the same filename. The user can choose to overwrite the existing files or cancel and either modify the filenames or change the saving path.



11: A notice next to the heading 'Set saving details' informs the user about successful saving.

generateCategory

The button ‘generateCategory’ opens a window to create a complete bubble category. A bubble category is generated by either defining the minimum and maximum deviation from the category base stimulus (option A, uniform distribution) or the standard deviation, which describes the width of the parameter distribution (option B, Gaussian distribution).

The user can design a specific base stimulus (opens another window – analogous to editRUBubble), use a randomly generated base stimulus with a selected number of spheres or load a pre-designed bubble stimulus. After specifying the requested number of category members, the user has to select the parameter distribution: uniform or Gaussian. Figures of all category members are numbered consecutively and saved as ‘stim#’. Parameters of all bubble stimuli are saved in the struct ‘stim’ and another struct ‘catParameter’ contains all parameters of the base stimulus and details of the category generation (selected distribution, delta/sigma values per parameter, number of similarity level).

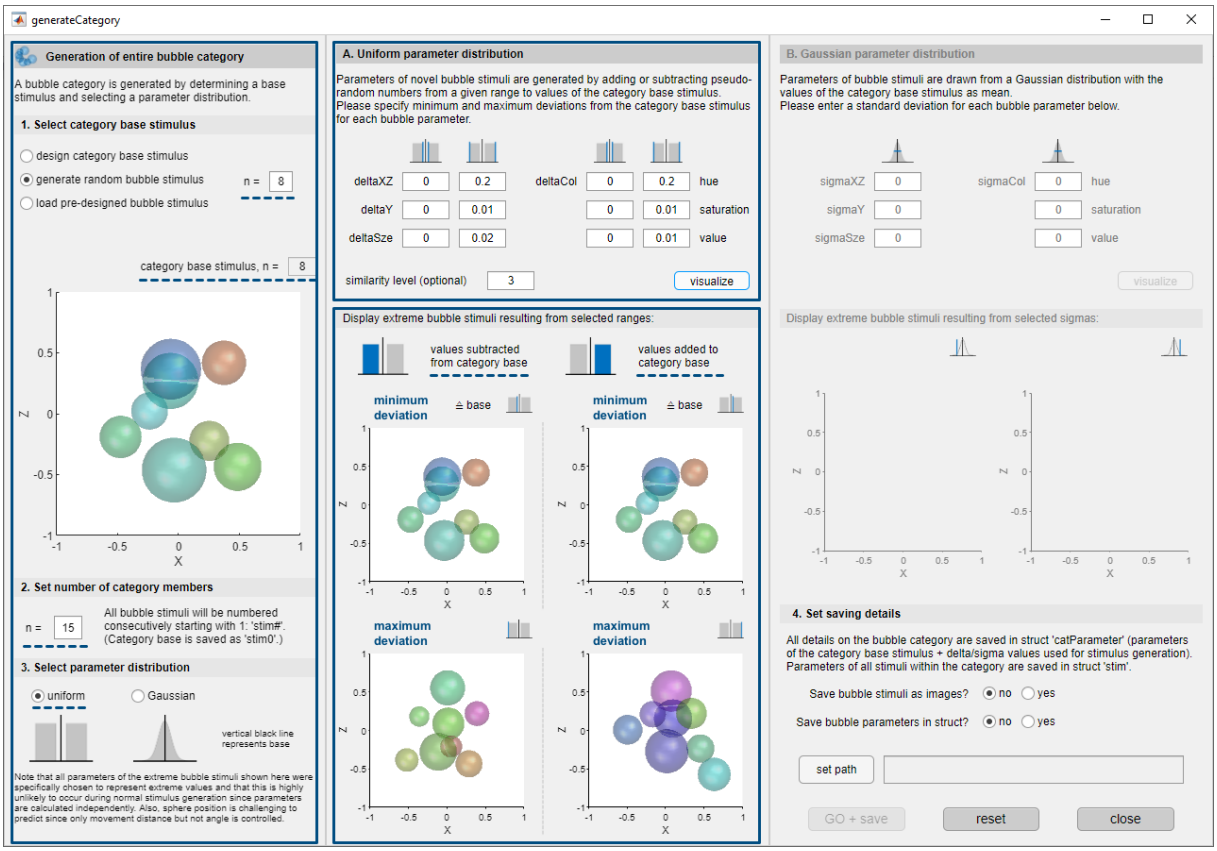
A. Uniform parameter distribution

The user has to specify the minimum and maximum deviation from the base stimulus for each parameter. Visualized are bubble stimuli constructed using these extreme values. Both upper figures show the minimum deviations (left: subtracted from base stimulus, right: added to base stimulus). If all minimum values are set to 0, the depicted bubble stimuli are identical to the base stimulus (also indicated above the figure ‘≙ base’). The lower figures visualize the maximum deviations (left: subtracted from base stimulus, right: added to base stimulus). Input values can be between 0 and 0.5 with the first number being the minimum value (minimum ≤ maximum). It is advisable to use small input values for size, saturation and value: Minor numeric changes lead to big changes in sphere size and sphere color should mostly be manipulated using hue.

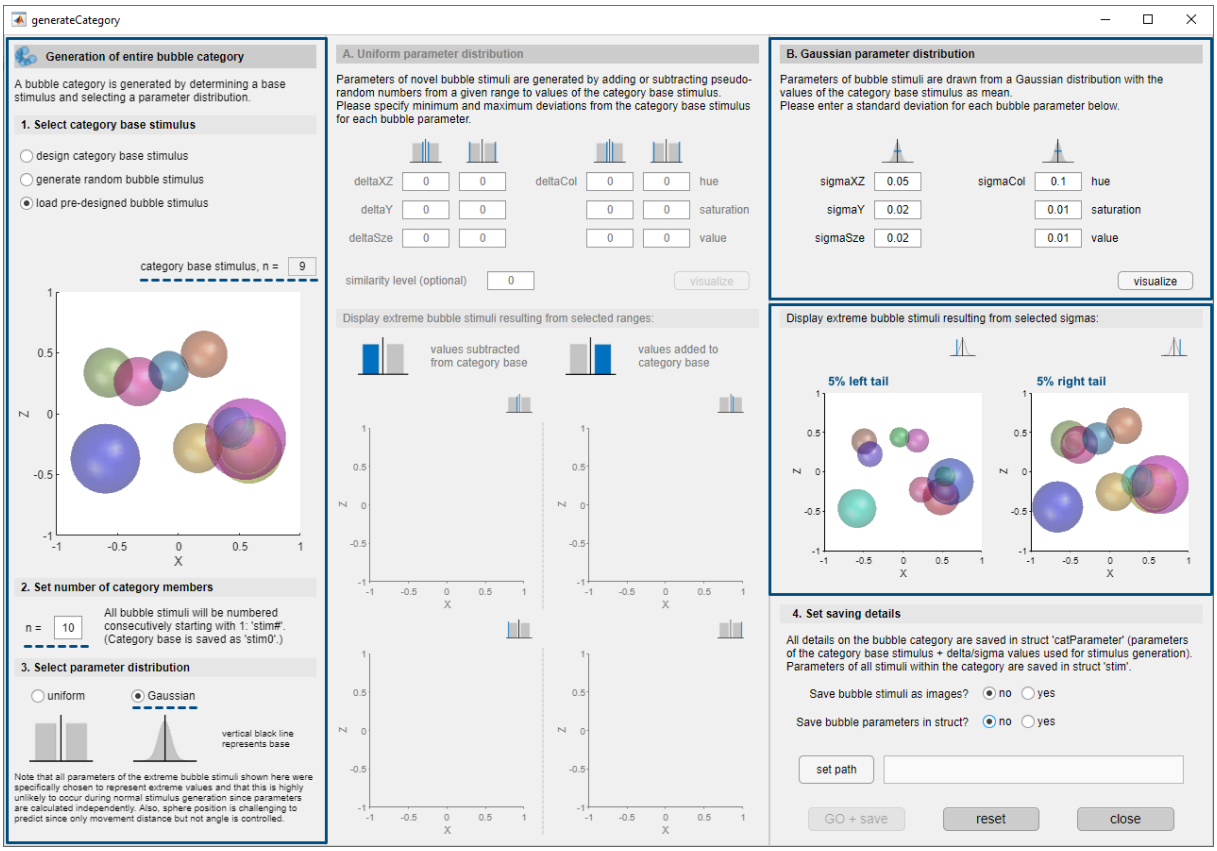
When generating a category based on uniform parameter distributions the user can further specify the number of similarity levels. Defined parameter ranges are then subdivided according to the number of similarity levels and used to create bubble stimuli that show small (lower level) and high (upper level) deviations from the base stimulus. If the requested stimulus number cannot be divided by the number of similarity levels the app produces more stimuli than indicated, for example the values numStim = 15, simLevel = 3 result in the generation of 16 stimuli instead of 15.

B. Gaussian parameter distribution

The user has to specify the standard deviation, which describes the width of the parameter distribution with the value of the base stimulus as mean. Visualized are bubble stimuli constructed using parameter values at 5% from the tails of the Gaussian distribution. Input values can be chosen between 0 and 0.5. However, as explained for option A it is advisable to use small values for sphere size, saturation and value.



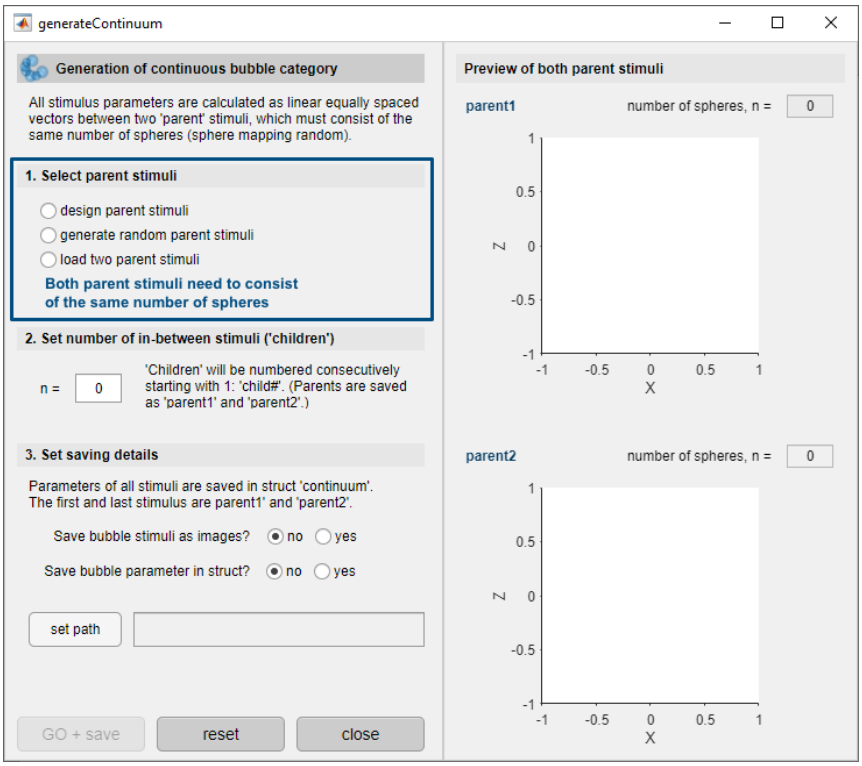
12: Bubble category created using a uniform parameter distribution. The category is based on a randomly generated base stimulus consisting of 8 spheres and delta ranges specified for each parameter. Exemplary stimuli visualized in the center panel outline minimum and maximum deviations from the base stimulus (both for a subtraction on the left and addition on the right). Since all minimum values were set to 0 in this example both upper stimuli are identical to the base stimulus (also indicated above the figure '≙ base'). As in this example, the maximum deviations for size, saturation, value (and Y coordinate) should be chosen from lower values. A notice next to the heading 'Set saving details' informs the user about successful saving.



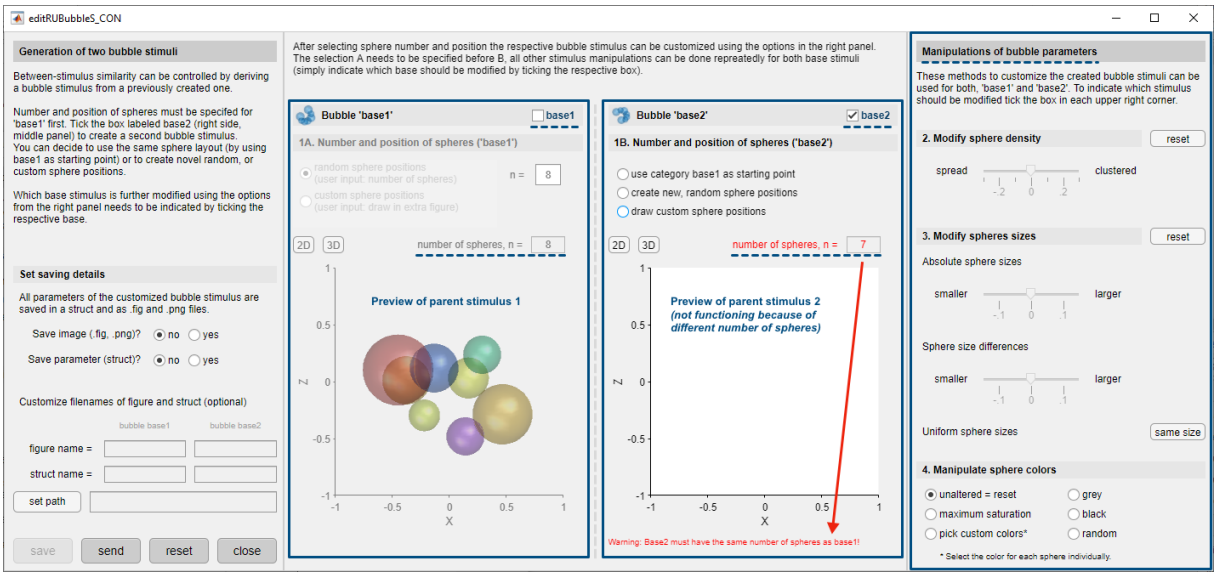
13: Bubble category created using a Gaussian parameter distribution. A pre-designed bubble stimulus was used as category base stimulus. Standard deviation values were specified for each bubble parameter. Exemplary stimuli visualized in the right panel outline resulting bubble stimuli using values at 5% from the tails of the Gaussian distribution. As in this example, it is advisable to use small standard deviations for the parameters size, saturation and value. A notice next to the heading 'Set saving details' informs the user about successful saving.

generateContinuum

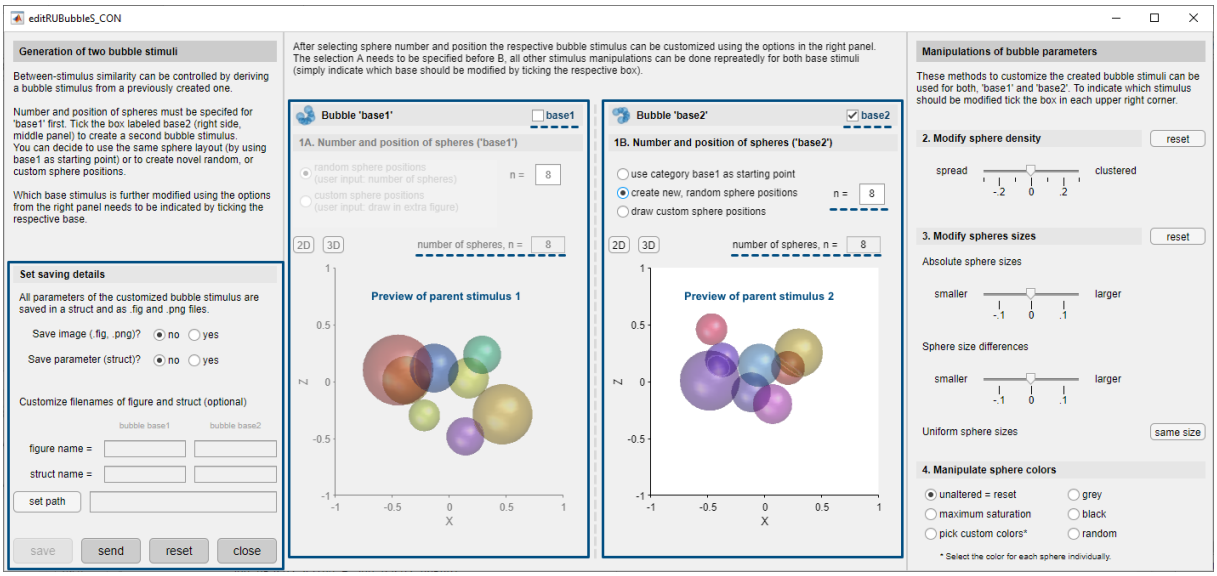
The button 'generateContinuum' opens a window to create a continuous bubble category. All category members are calculated as in-between stimuli based on two parent stimuli. Parent stimuli must consist of the same number of spheres and can either be designed (opens another window – analogous to editRUBubbleS), randomly generated (sphere number set by user) or uploaded. Figures of all in-between stimuli are numbered consecutively and saved as 'child#'. Stimuli of both parent stimuli are saved as 'parent1' and 'parent2'. Parameters of all bubble stimuli are saved in the struct 'continuum' (parents as first and last stimulus).



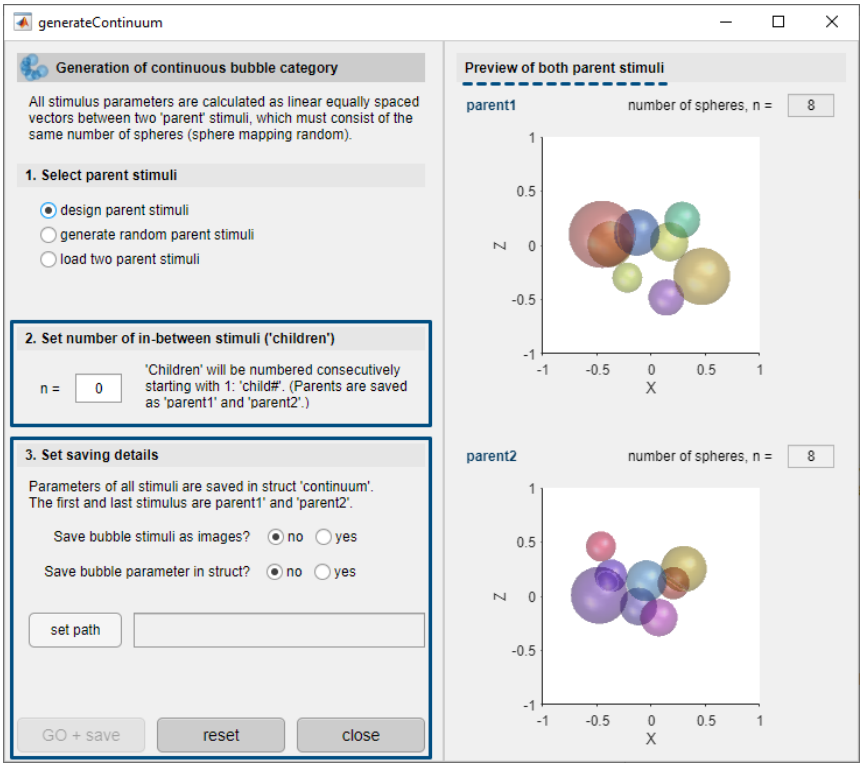
14: A continuous category is generated based on two parent stimuli, which can be specifically designed, randomly generated or uploaded. To enable the calculation of linear equally spaced vectors for each parameter, both parent stimuli need to consist of the same number of spheres.



15: Parent stimuli can be specifically designed in another window similar to 'editRUBubbleS'. The number of spheres must be the same between both base stimuli and the user is warned if, for instance, drawn sphere positions differ in sphere number from the previously created base1.



16: When sphere positions of base2 are selected randomly, the number of spheres is set automatically based on the sphere number of base1. The created parent stimuli can be sent back to create a continuous category and/or additionally saved within this window. A notice next to the heading 'Set saving details' informs the user about successful saving and sending.



17: The right panel contains a preview of both parent stimuli (in this example, specifically designed), which are used to create a continuous category. Before setting the saving details, the user has to specify the desired number of in-between stimuli. A notice next to the heading 'Set saving details' informs the user about successful saving.

generateExceptions

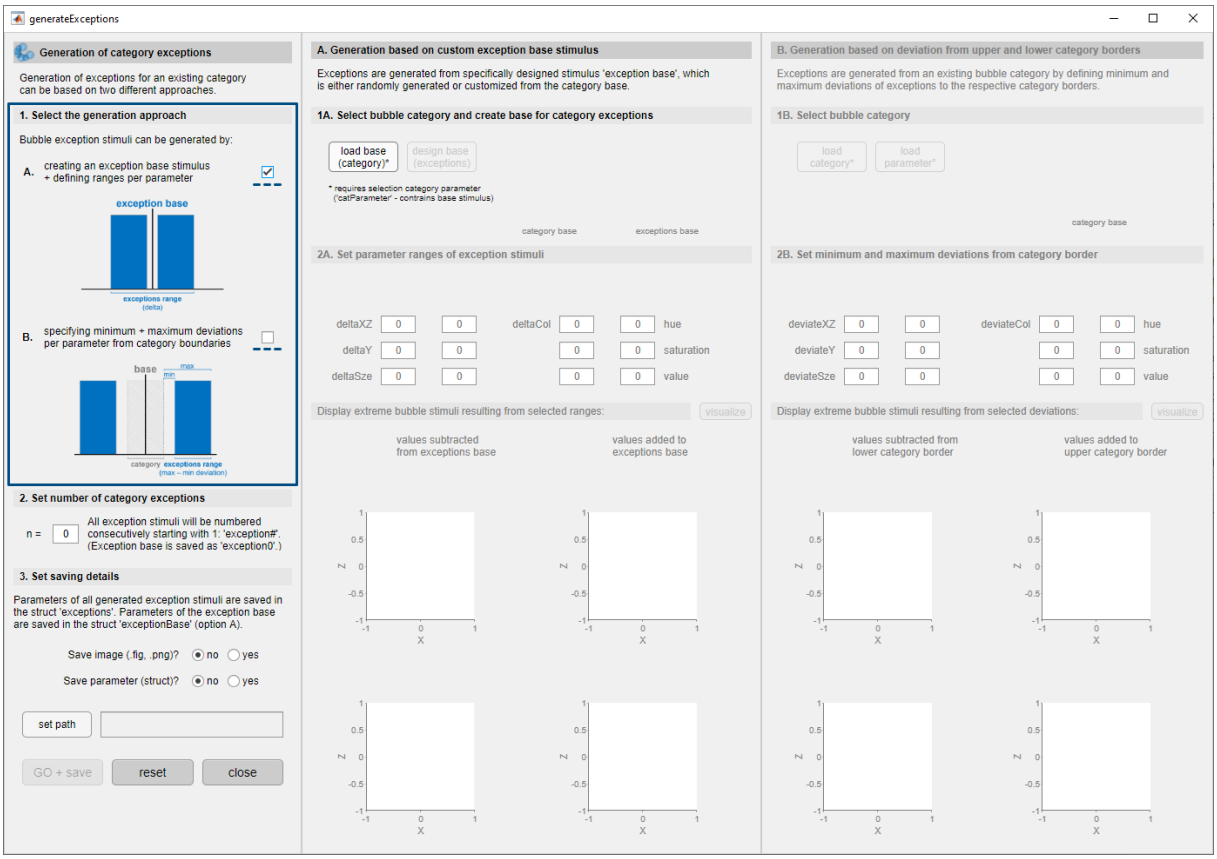
The button ‘generateExceptions’ opens a window to create category exceptions for an already generated bubble category, which has been created based on a uniform parameter distribution. The user can choose between two generation approaches: exceptions can either be created based on a custom exception base stimulus (option A) or by defining the minimum and maximum deviations from the category boundaries (option B). Figures of all exception stimuli are numbered consecutively and saved as ‘exception#’. Parameters of all bubble stimuli are saved in the struct ‘exceptions’ and all details of the exception stimulus generation are saved in struct ‘exParameter’ (exception base stimulus/category borders, delta/deviation values per parameter). In option A, the created exception base is saved as ‘exception0’ (figure) and ‘exceptionBase’ (struct).

A. Exception base stimulus

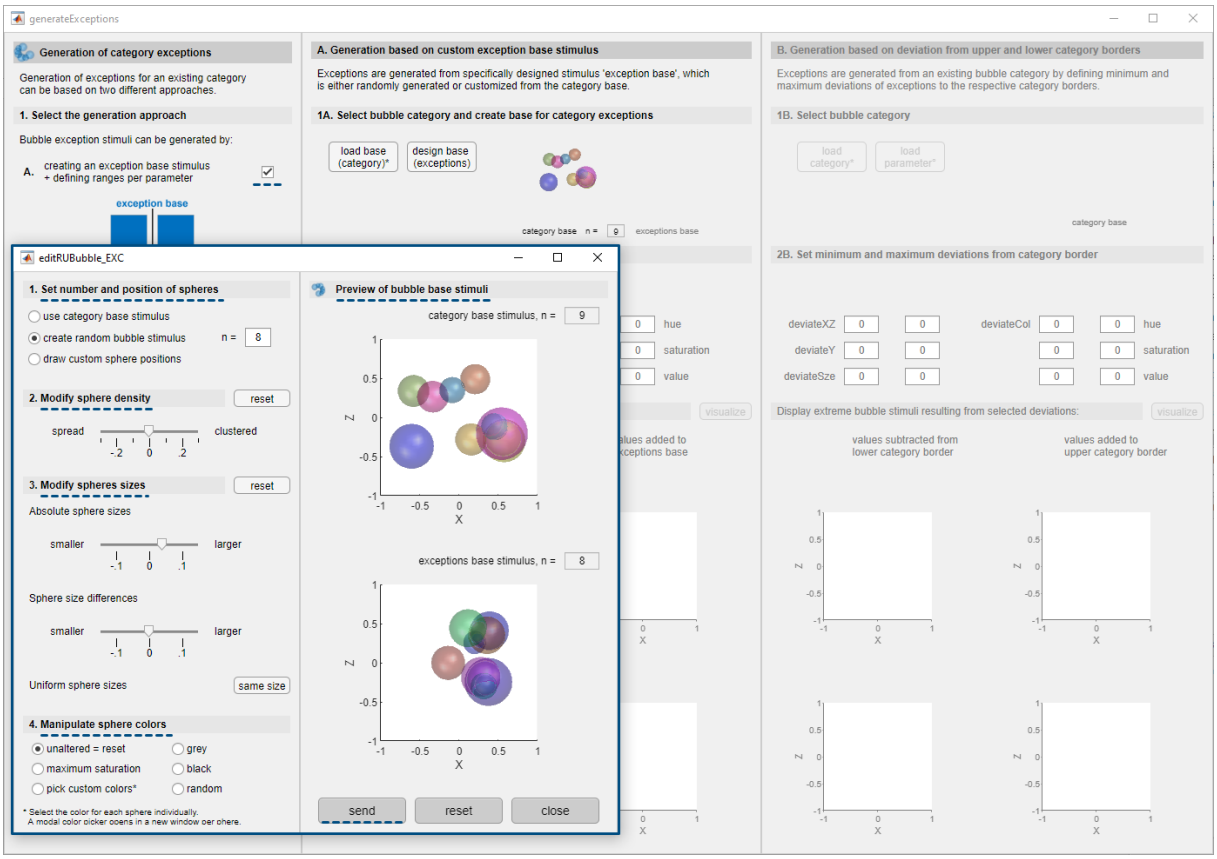
The user first has to upload the struct ‘catParameter’ of a bubble category. Thereafter, the exception base stimulus has to be created (opens another window: ‘editExceptionBase’, slightly different design but with same operating principle as ‘editRUBubble’). Both base stimuli are then visualized in the upper section of the middle panel. As for the generation of a uniformly distributed category, the user then has to specify the minimum and maximum deviations from the exception base stimulus in order to define parameter ranges for all exception stimuli (ranges, minimum ≤ maximum, values between 0 and 0.5, smaller numbers for size, saturation, and value advisable). Both upper figures show the minimum deviations (left: subtracted from exception base stimulus, right: added to exception base stimulus). If all minimum values are set to 0, the depicted bubble stimuli are identical to the exception base stimulus (also indicated above the figure ‘≙ exceptions base’). The lower figures visualize the maximum deviations (left: subtracted from exception base stimulus, right: added to exception base stimulus).

B. Deviation from category boundaries

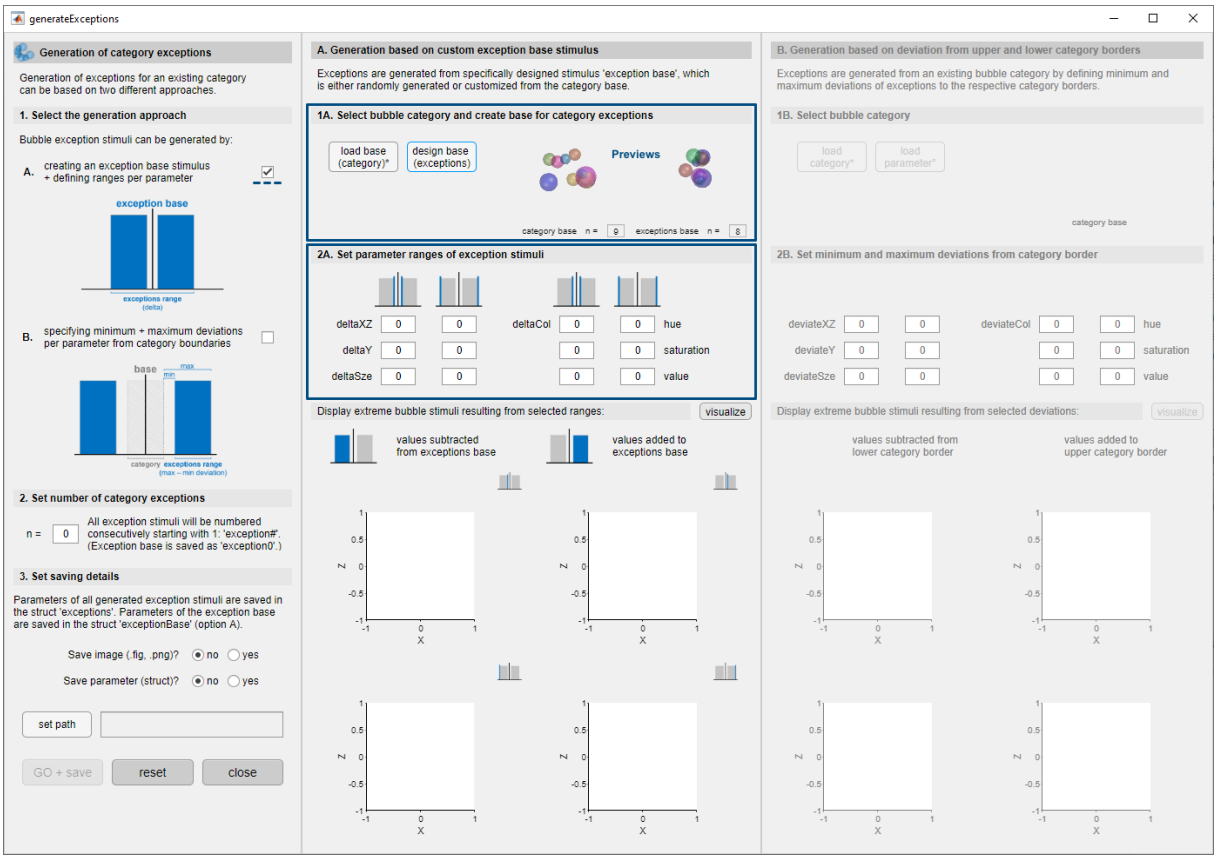
The user first has to upload the category and all generation details: structs ‘stim’ and ‘catParameter’. Checks within the heading indicate which struct has already been uploaded. Subsequently, the category base stimulus is visualized in the upper section of the right panel. To create the parameters of exception stimuli, the user has to specify the minimum and maximum deviations from the category borders. Values within these ranges will then be added or subtracted from the upper and lower category limits, respectively. Input values can be chosen between 0 and 0.5. However, as explained before, it is advisable to use small values for sphere size, saturation and value.



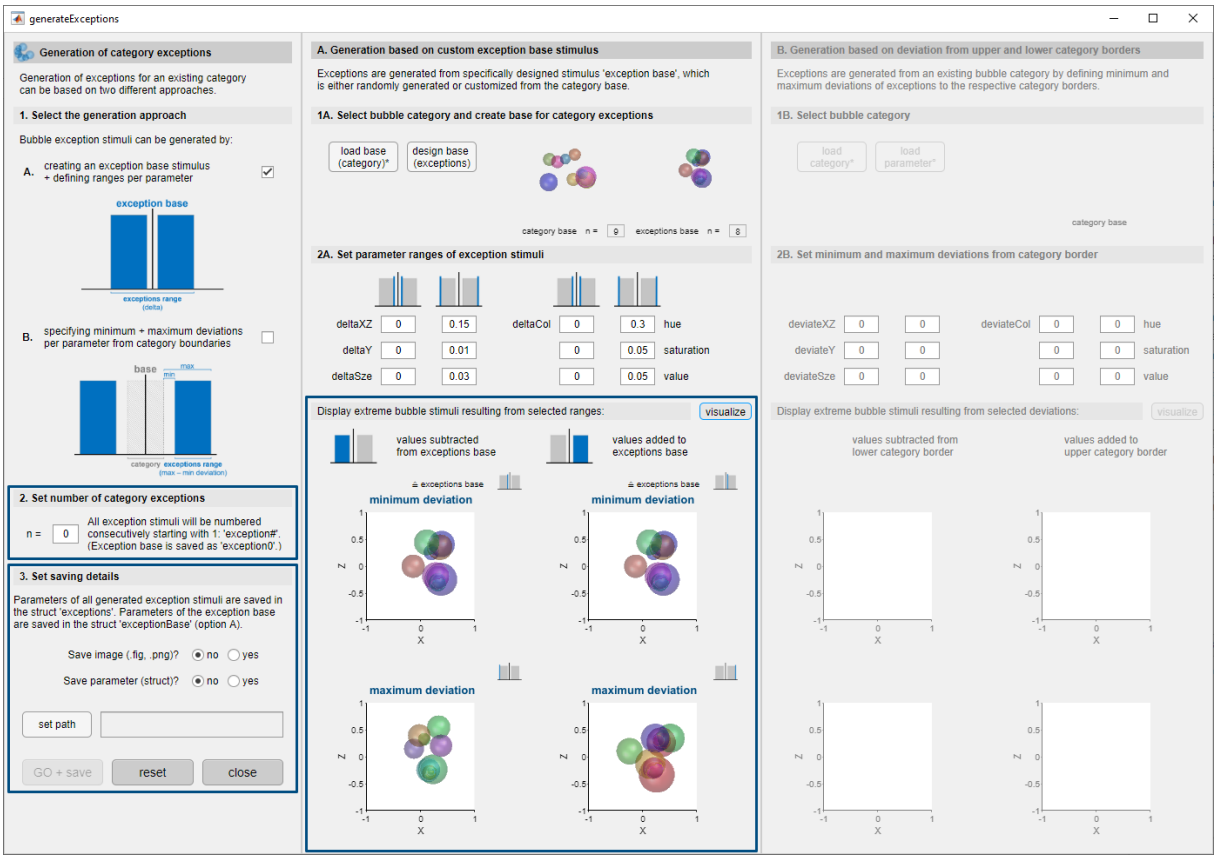
18: To generate category exceptions the user can either design an exception base stimulus and then define minimum and maximum deviations (option A) or define minimum and maximum deviations from the category borders (option B). The generation approach is selected by ticking either box in the left panel next to the respective description. In this example, option A is chosen.



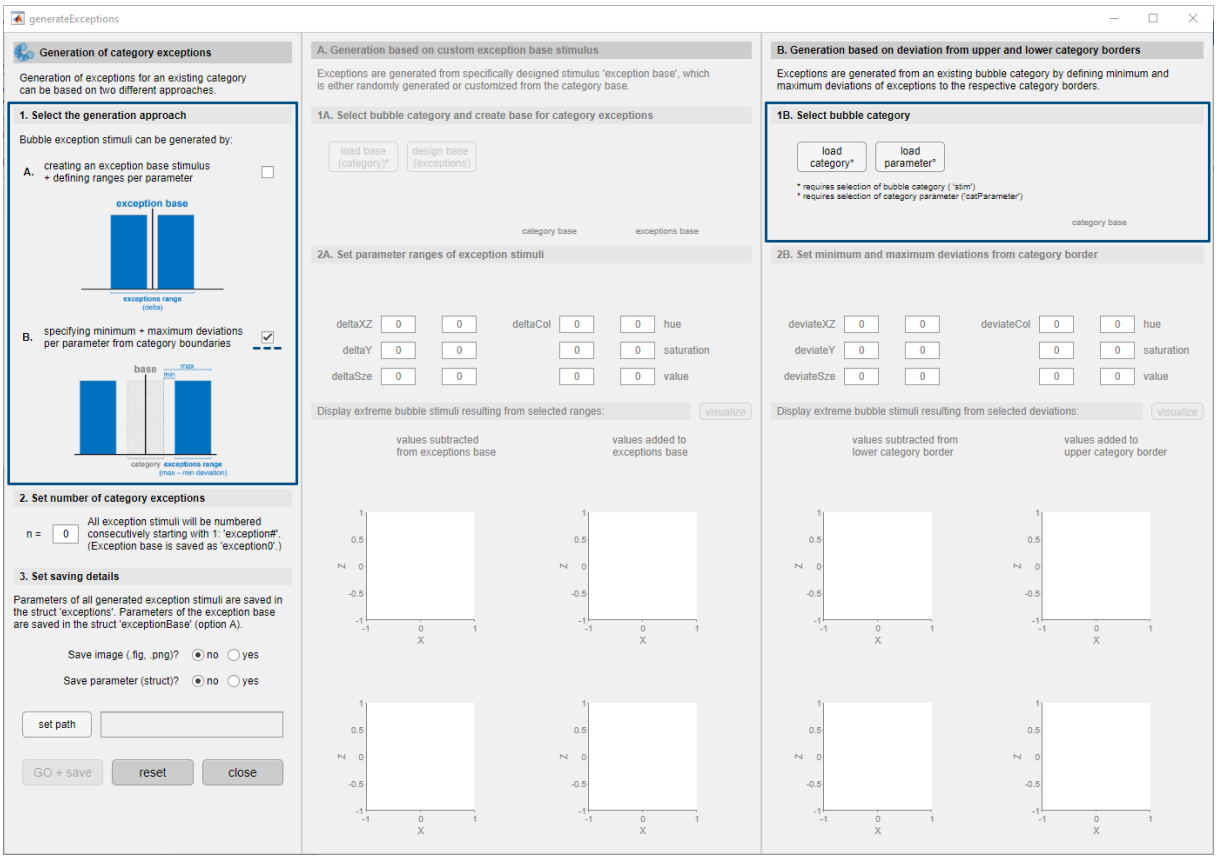
19: Creation of an exception base stimulus (option A). After uploading parameters of a bubble category, the user has to design an exception base stimulus. For this, another window opens which visualizes the category base stimulus and enables the user to specifically design an exception base stimulus, which is then sent back to the 'generateExceptions' window.



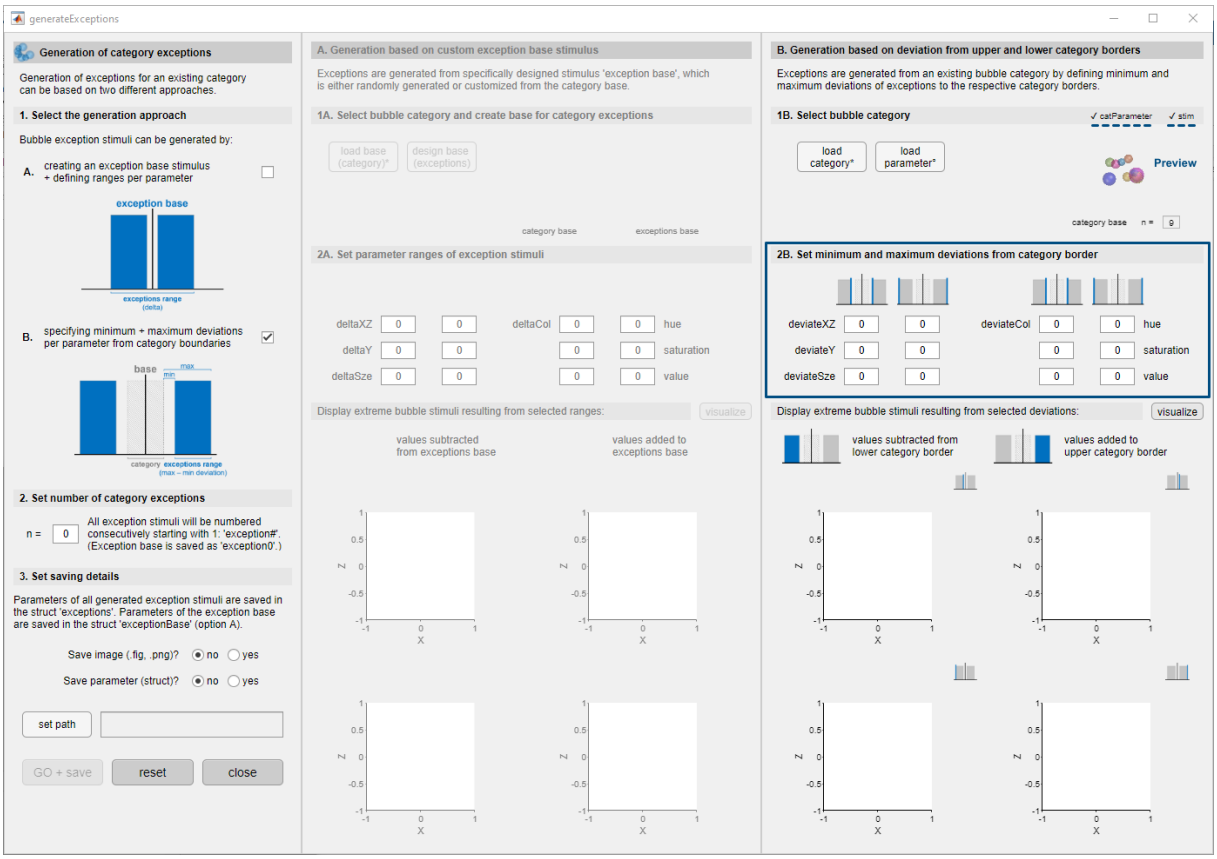
20: Both base stimuli (category and exception base) are visualized in the upper part of the middle panel. Now, parameter ranges can be specified for each bubble parameter (minimum ≤ maximum, values between 0 and 0.5, smaller numbers for size, saturation, and value advisable).



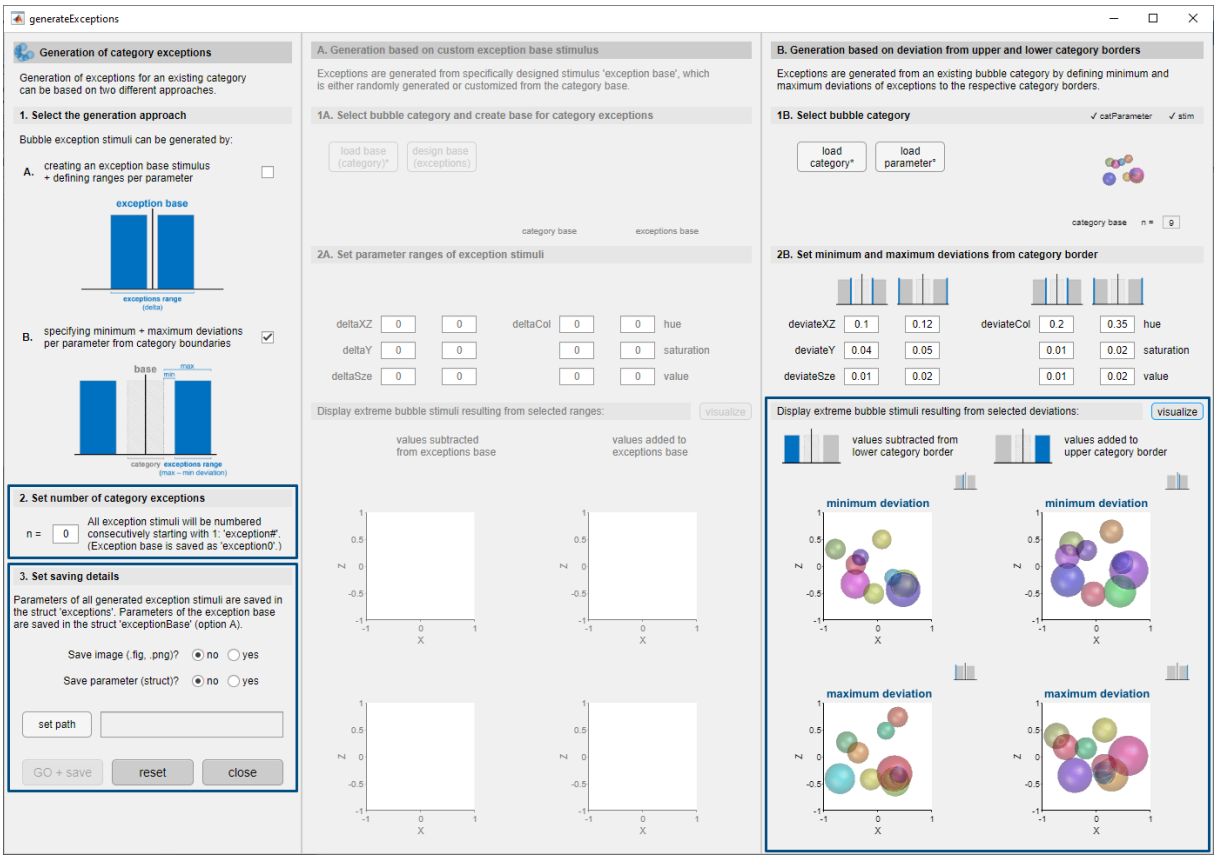
21: Visualization of exemplary bubble stimuli using the specified minimum and maximum deviation values per parameter. Since all minimum values were set to 0 in this example both upper stimuli are identical to the exception base stimulus (additionally indicated above each figure: '≙ exceptions base'). Now, the number of exception stimuli and saving details should be specified. A notice next to the heading 'Set saving details' informs the user about successful saving.



22: To use parameter deviations from upper and lower category borders for the creation of exception stimuli, the box of option B needs to be selected (left panel). In the right panel, the bubble category and generation details have to be uploaded (button 'load category' → input: struct 'stim' and button 'load parameter' → input: struct 'catParameter'). Checks within the heading '1B. Select bubble category' indicate which struct has already been uploaded (see next image).



23: After all category details have been uploaded, the category base stimulus is visualized in the upper part of the right panel and minimum and maximum deviations from the category borders can be specified (minimum ≤ maximum, values between 0 and 0.5, smaller numbers for size, saturation, and value advisable).



24: Visualization of exemplary bubble stimuli created based on the specified minimum and maximum values per parameter. Now, the number of exception stimuli and saving details should be specified. A notice next to the heading 'Set saving details' informs the user about successful saving.