A.2 GUI: User Manual

This document is written as a guide for the Matlab GUI: Sensor Placement Optimization.

A.2.1 Intended Use

The UI is made to simplify the problem formulation of the Sensor Placement Problem in 3D for sensors defined by the field of view, range, and price. The UI supports **.wrl* (VRML) files with nodes defined as Indexed Face Set as its input. The process for defining the problem follows the procedure shown below, with further explanations throughout this document.

1. Import VRML model
2. Add floor
3. Add cameras
4. Define placement lines
5. Define regions of interest
6. Define optimization parameters
7. Generate optimization code
8. Visualize results

A.2.2 About

The GUI is made as part of a master thesis at the University of Agder 2018 by Vegard Tveit.

A.2.3 Requirements

The program is made using Matlab 2017a. It is assumed that the user has some programming experience with Matlab, but most of the code is properly commented for easier understanding and bug fixes. It is recommended to use separate software for interpreting, changing or converting the 3D VRML files. A proposed software is Meshlab.

In the VRML file, the geometry nodes must be defined as indexed face set. Also, the *transform* node must be defined as *layout*. In the VRML file, this should look as following: *DEF layout Transform {*.

The user should have some knowledge regarding computational geometry and optimization for best use of the software, but this is not required. The software is set up for an example program where the user is only required to follow the necessary steps without doing any modifications. By being able to see the intended options for a given problem, the user can get a better understanding of how to use the UI.

A.2.4 File

As a safety feature, in case the program crashes, or errors are made, the UI stores information for each significant step in **.mat* files. If the program shuts down and is started again, the **.mat* files can be

used as a method of restoring previous data. This method replaces the commonly used *save as* and *load* functions, but the result is the same.

A.2.5 Edit

The *Edit* menu contains the function for defining the problem.

Edit VRML

The *Edit VRML* menu opens Matlab's VRML editor. The full documentation for this program can be seen in `https://se.mathworks.com/help/sl3d/the-3d-world-editor.html` (retrieved 23.01.2018), and will not be described in this document.

Floor

To determine the shape of the polygon enclosed by the walls, a function for adding a floor to the problem space is included. The "Add Floor" menu opens a figure window and a dialog box where the corner coordinates are specified. There are no limits on the number of defined corners. The coordinates should be separated by spaces, and it should be checked that there are equally many y- and z-coordinates as x-coordinates. To get the corner coordinates, the data cursor in the figure toolbar can be used. Another tip is when in "Rotate 3D" mode, the view can be changed by right-clicking in the figure window. By choosing XZ view, it is easier to get the correct x- and z-coordinates. When the dialog box is closed, the floor is added to the figure for visualization and saved to the Matlab workspace as well as added to the Indexed Face Set in the VRML model.

Camera

The camera menu is the most extensive sub-menu, where the sensors are defined as well as the placement lines for the sensors. Firstly, the camera parameters are specified in the "Add Camera" menu. Additionally, the price can be specified for each sensor in any desired unit in the dialog box.

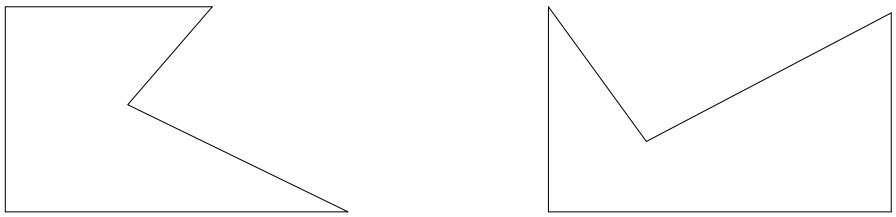
After the cameras are defined, the user needs to specify where the sensors can be placed. This is done in the "Add Lines" menu, which opens a dialog box and a figure. The lines should be straight and defined by the start and end point of the line. Similarly to the procedure of adding the floor, the coordinates can be found using the data cursor tool. When the close button is pushed, the dialog box closes, and the lines are saved to the Matlab workspace. The lines that should be used for sensor placement must have the *Accepted* choice checked. In this way, the program understands which lines to use, and which to ignore. When the lines are specified, *lines.mat* is saved with the information from the dialog box. This mat file must not be deleted before the JSON file is generated since it will be used later in the program.

The lines can be visualized using the "Visualize Lines" menu. A new dialog box is opened, where the accepted lines are displayed. By pressing the *Visualize* button, a figure window opens with the 3D model (shown in red) as well as the placement lines (shown in blue).

Region of Interest

A region of Interest enables the user to have k-coverage possibilities. First, the user should specify whether the grid generation should be along the vertical or horizontal axis. To get a better understanding of why this is relevant, two figures are shown in Fig. A.2. Fig. A.2b shows an example of a polygonal floor plan, where it would be easiest to generate the fishnet along the vertical axis. The reason for this is that for this polygon, every point along the vertical axis represents a unique point at the polygonal edge. Contrarily, in Fig. A.2a, every point along the horizontal axis represents a unique

point along the polygonal edge. It should be noted that the grid generation starts at the bottom left corner of the polygon.



(a) Figure for Horizontal Grid Generation (b) Figure for Vertical Grid Generation

Figure A.2: Directions of Grid Generation

The next step is to add the Region of Interest. This is done by clicking the "Add Region of Interest" option, which opens an input window and a figure of the generated grid of the polygon representing the floor. The user has to specify the coordinates for the Region of Interest according to Fig. A.3.

In Fig. A.3, a ROI volume is presented for visualization. The subscript of either 0 or 1 indicates the corner coordinates at either Y0 or Y1, respectively.

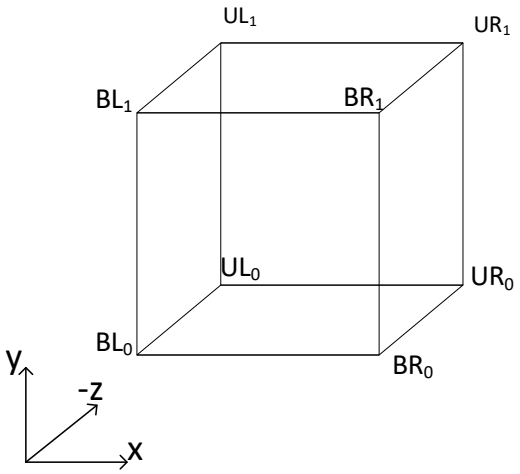


Figure A.3: Region of Interest Cube Coordinates and Indices

Additionally, the user has to specify the weight option, either 1, 2 or 3. A weight of 1 corresponds to a 2-covered region of interest, which is the standard option. 2 is a 3-covered region of interest, and 3 in a zero-covered region of interest, which means that the region is of no interest concerning coverage.

The user can add multiple ROIs, and display them in a list using the "Visualize Region of Interest" function, which also displays the region of interest in the 3D scene.

A.2.6 Optimization

Setup Parameters

Here, the main parameters for the optimization accuracy are defined. Firstly, the room height at floor and roof level must be established. Next, the height of the voxels is specified by determining the total number of voxels in the vertical direction. Finally, the placement lines accuracy is defined by specifying the number of placement points per length unit along the defined lines.

Generate JSON

When the parameters are defined, as well as all other necessary parameters are set up to describe the problem, the JSON file can be generated. Generating the JSON file may take some time. When the file is created, a figure is shown that graphically shows the output regarding obstacles, placement points, and regions of interest.

A.2.7 Visualization

In the *Visualization* menu, the result from the optimization algorithm can be seen.

Load Optimization Results

When *Load Optimization Results* is chosen, the user must specify the position *(x,y,z)* of all sensor to be placed along with the number of sensors and their respective pan angles. For now, the sensor parameters are considered to be fixed, but this can easily be changed in the callback functions. The user must also specify the filename of the **.mat* file where the *optim* data is stored from the UI output.