# How to use the Roculus visualization for ROSIE.

Daniel Bug

2014-08-21

## 1 Setting up ROSIE

Here are the different steps to be done on the robot before the application start:

- Turn on the key and start-up robot base and computer.
- When the login screen shows up, go into the EBC switch menu on the robot base and turn the last two 24.0V outputs ON. (7-0 and 7-1, 24V).
- Switch on the sidekick computer by pressing the small red button at the side of the terminal.
- Open a console and begin to start up the robot operating system. There is a startup script for a *tmux* session that will have a set of all interesting terminals for you: `Ctrl+R 'start.sh' ->` . It takes a while to start teh first time. The first terminal is a display of the running processes (htop) and you can browse the terminals by using `Ctrl+b, n` or `Ctrl+b, p` to get to the next/previous terminal.
  The following things need to run:
  - `rosie_core`
  - `rosie_robot`
  - `rosie_navigation` (You will probably have to init the navigation in rviz manually to have to robot localized correctly.)
  - `rosie_head_camera`, which needs to be run on the sidekick. (Check that the terminal is an ssh session on the strands-sidekick, before you start the camera launch file).
- The application does not listen to the pure camera topics, but - in order to save bandwidth - relies on topics that are republished from a *drop* node (`Ctrl+R 'drop'`):
  - Open a new (tmux) terminal by typing `Ctrl+b, c`
  - drop-node for the rgb images: `rosrun topic_tools drop /head_xtion/rgb/image_color/compressed 9 10 /head_xtion/rgb/image_color/reducedBW/compressed &`
  - drop-node for the depth images: `rosrun topic_tools drop /head_xtion/depth/hw_registered/image_rect/compressedDepth 9 10 /head_xtion/depth_registered/image_rect/reducedBW/compressedDepth &`

The `9 10` parameters indicate that we will drop 9 out of 10 images, which will give around 3 fps for the video stream. You can work with higer values, but there will be a limit at which you start trading off a better frame rate against the position and orientation updates. For dropping 2/3 e.g. the images might take to much bandwidth for the transforms to be updated regularly and the stream will look nice, but appear at the wrong place.
The topics should print a notification on the terminal saying that they are advertising the new topic. If you don't see the message the head camera was probably not started correctly. There might be a short-cut, but I used to kill all terminals and launch everything from scratch again.

# 2 Starting the Application

Roculus relies on the folder structure. During start-up the roculus directory will be searched for multiple config files and map-components. They are explained in the next section.

- To start the application:
  - On the visualization PC, make sure you are connected to rosienet and that your `/etc/hosts` file lists the correct IP adresses.
  - Check with `echo $ROS_MASTER_URI` that this environment variable is pointing to the robot, i.e. `http://scitosstrands:11311`
  - If you want to use the gamepad launch the teleoperation node in a separate terminal:
    `roslaunch scitos_teleop teleop_joystick.launch`
  - Go to the roculus folder: `roscd roculus`
  - Start roculus: `rosrun roculus roculus_node`, the start-up time can be in minutes, if the program is loading multiple room scans into the environment. (You can check the terminal output. To get there use `Alt+Tab`).

# 3 Configs, Resources and Map Data

As mentioned there are several configs and resources used by the application:

- ogre.cfg: contains the screen and render settings for ogre, if this file can not be read, ogre will show a config dialog to recreate such a file.
- plugins.cfg: specifies the plugins that will be loaded on start-up. Probably just the cg and particle library is needed, the rest will be commented out.
- resources.cfg: points to the materials/textures/shaders for the game:
  - media/sibenik.zip: one image from this archive is used as default initialization for textures
  - media/rosie.zip: everything for the robot avatar
  - media/game.zip: everything for the game... meshes for the keys/treasure/locks, their materials, etc.

  - media/vertexColor.material: contains the materials for the thesis application, (blank material, video stream texture, snapshot texture)
  - media/projection3D.cg: the cg shader programs that actually perform the reprojection from camera geometry to 3D snapshot. The color transformation for the sepia look if defined here as well.

- map directory: using the simpleXMLparser from Rares and the multiroom parser, all patrol-recordings in this folder are loaded into the environment during start-up

# 4 Functions on the Keyboard

The keyboard will be used as a supervisor input. It can:

- ESC (3 times): End the application
- i: reinitialize the game
- p: toggle first person mode
- a,d,s,w,(arrows), PgUp, PgDn: Move around in the world in free view
- SPC: select a navigation target
- m: toggle visibility of the map
- v: toggle the visibility of the preloaded environment
- F3: toggle visibility of the frame rate display
- F4: toggle visibility of the application info display

# 5 Functions on the Gamepad and Mouse (Player Input)

The mouse can only select navigation targets by clicking (the selection still involves looking at the waypoint).
The gamepad has 4 possible functions:

- buttons 1-4: select a navigation target
- button 8: reset the oculus orientation to look in the direction of the robot
- cross-joystick (x-direction): look 120deg behind you (to the left/right)
- only if the supervisor unmouted the player from first-person: you can fly around with the joysticks (left: forward+backward/turn, right: up+down and step left/right)