RPC API for RoboSkate

TUM
Michael Zechmair

Contents

1 RoboSkate setup

RoboSkate should be able to start as a server. This means that, after startup, the game should be in a paused state, waiting for inputs from a client. The client will send a command to load a level, and then send commands to control the robot.

2 Communication Protocol

Communication between the NRP and RoboSkate should be performed via the RPC protocol. For this an RPC server has to be implemented within RoboSkate which can be accessed from a separate machine. This server should facilitate a number of RPC endpoints, which will be described in detail in the following section.

3 RPC Endpoints

3.0.1 /initialize

Description Initialization call. Used to perform initial setup operations. In this case, it should:

- Load a level
- Set the board to the starting position
- Fix camera behind board
- Return camera resolution

URL Structure `<roboskate_host_ip>\initialize`

Parameters JSON String:

level Level Number

Returns Returns an object containing:

Byte Success value. 0 For success, any other value indicates error

Integer Camera image height

Integer Camera image width

Example Input:

```
{
    "level": 1
}
```

Output:

```
0
1920
1080
```

3.0.2 /shutdown

Description Shutdown call. This should close the level and let RoboSkate wait for a new **initialize** call.

URL Structure `<roboskate_host_ip>\shutdown`

Parameters No parameters are passed

Returns Returns an object containing:

Byte Success value. 0 For success, any other value indicates error

Example Input: No input
Output:

```
0
```

3.0.3 /get_info

Description Get state call. Once a level is loaded, this call should be able to get board state information.

URL Structure `<roboskate_host_ip>\get_info`

Parameters No parameters are passed

Returns Returns an object containing:

Byte Success value. 0 For success, any other value indicates error

3 Float Board position

3 Float Board linear velocity

3 Float Board angular velocity

9 Float Board rotation matrix

3 Float Board crane joint angles

3 Float Board crane joint velocities

Example Input: No input
Output:

```
0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0
0.0 0.0 0.0
0.0 0.0 0.0
```

3.0.4 /set_info

Description Set state call. Once a level is loaded, this call should be able to set the board crane position

URL Structure `<roboskate_host_ip>\set_info`

Parameters An object containing:

1 Float Percentage time of "Left"/"Right"-key depresses (from -1.0 to 1.0, -1.0 means "Left"-key is depressed all the time, 1.0 means "Right"-key is depressed all the time)

1 Float Percentage time of ”D”/”A”-key depresses (from -1.0 to 1.0, -1.0 means ”D”-key is depressed all the time, 1.0 means ”A”-key is depressed all the time)

1 Float Percentage time of ”E”/”W”-key depresses (from -1.0 to 1.0, -1.0 means ”E”-key is depressed all the time, 1.0 means ”W”-key is depressed all the time)

Returns Returns an object containing:

Byte Success value. 0 For success, any other value indicates error

Example Input:

```
−1.0  0.9  0.0
```

Output:

```
0
```

3.0.5 /get_camera

Description Get camera data call. Once a level is loaded, this call should be able to get the camera output.

URL Structure `<roboskate_host_ip>\get_camera`

Parameters No parameters are passed

Returns Returns an object containing:

Byte Success value. 0 For success, any other value indicates error

Image Height*Width * 3 bytes Image RBG values

Example Input: No input
Output:

```
0
255  225  155  0  1  23  ...  234  245  128
```

3.0.6 /run_game

Description Run game call. Should run game for a specified amount of time

URL Structure `<roboskate_host_ip>\run_game`

Parameters Object containing:

Float Time (in seconds) to run simulation

Returns Returns an object containing:

Byte Success value. 0 For success, any other value indicates error

Float Total simulated time

Example Input:

```
0.01
```

Output:

```
0
31.28
```