

ISO Flock OSC Control

Set Simulation Rate

Set simulation update rate.

Command: /SetSimulationRate

Parameters v1: float (rate in miliseconds)

Add Space

Add new space for neighborhood calculations. Types of neighborhood calculations are: NTree or KDTree or ANN for neighborhoods between pointlike objects (such as agent parameters), RTree for neighborhoods between shapes, Grid for grid space such as force fields, or finally PermanentNeighbors for neighbors, that are manually managed. In case of grid spaces, a neighbor calculation mode needs to be supplied that can be either CellLocation (the grid value in the cell the neighbor is in) or GridLocation (list of grid values of all the cells that are next to the neighbor) or AvgLocationMode (value at position of neighbor that is linearly interpolated from the cells next to the neighbor) or AvgRegionMode (spatial position derived from the positions of the cells next to the neighbor weighted by the value of these cells). Also, for grid space, it needs to be specified whether the neighbors in the grid can modify the values of the grid, this so called grid update mode can assume one of the following values: NoUpdateMode (the grid doesn't change) or NearestReplaceMode (the values in the nearest grid cells or overwritten) or NearestAddMode (values are added to the nearest grid cells) or AvgReplaceMode (the values in the nearest grid cells are overwritten with linearly interpolated values) or AvgAddMode (linearly interpolated values are added to the nearest grid cells)

Command: /AddSpace

Parameters v1: string (unique but arbitrary name of space)
 string (type of neighborhood calculation: NTree or KDTree or ANN or RTree or Grid or PermanentNeighbors)
 int (dimension of space)

Parameters v2: string (unique but arbitrary name of space)
 string (type of neighborhood calculation: NTree or KDTree or ANN or RTree or Grid or PermanentNeighbors)
 float array (minimum position in space)
 float array (maximum position in space)

Parameters v3: string (unique but arbitrary name of space)
 string (type of neighborhood calculation: NTree or KDTree or ANN or RTree or Grid or PermanentNeighbors)
 string (name of grid: should be empty for the moment)
 string (grid neighbor calculation mode: CellLocation or GridLocation or AvgLocation or AvgRegion)
 string (grid update mode: NoUpdate or NearestReplace or NearestAdd or AvgReplace or AvgAdd)

Remove Space

Remove existing space for neighborhood calculations

Command: /RemoveSpace

Parameters v1: string (unique name of space)

Add Sender

Add new sender for network based communication. Communication supports both TCP and UDP based internet protocols. The format of the communication messages can either be OSC or the ISO Default format.

Command: /AddSender

Parameters v1: string (unique but arbitrary name of sender)
 string (ip address of communication target, can be empty for
 UDP mode -> broadcast)
 int (port number of communication target)
 string (communication protocol: UDP or TCP)
 string (communication message format: OSC or Default)

Remove Sender

Remove existing sender for network based communication.

Command: /RemoveSender

Parameters v1: string (unique name of sender)

Add Receiver

Add new receiver for network based communication. Communication supports both TCP and UDP based internet protocols. The format of the communication messages can either be OSC or the ISO Default format.

Command: /AddReceiver

Parameters v1: string (unique but arbitrary name of receiver)
 int (port number of communication source)
 string (communication protocol: UDP or TCP)
 string (communication message format: OSC or Default)

Remove Receiver

Remove existing receiver for network based communication.

Command: /RemoveReceiver

Parameters v1: string (unique name of receiver)

Add Swarm

Add new swarm.

Command: /AddSwarm

Parameters v1: string (unique but arbitrary name of swarm)

Remove Swarm

Remove existing swarm.

Command: /AddSwarm

Parameters v1: string (unique name of swarm)

Add Agents

Add agents to existing swarm.

Command: /AddAgents

Parameters v1: string (unique name of swarm)
 int (number of agents to add)

Remove Agents

Remove agents from existing swarm.

Command: /RemoveAgents

Parameters v1: string (unique name of swarm)
 int (number of agents to remove)

Add Parameter

Add new agent parameter to swarm.

Command: /AddParameter

Parameters v1: string (unique name of swarm)
 string (unique but arbitrary name of parameter)
 float array (parameter values)

Remove Parameter

Remove existing agent parameter from swarm.

Command: /RemoveParameter

Parameters v1: string (unique name of swarm)
 string (unique name of parameter)

Assign Neighbors

Activate neighborhood calculations for agent or swarm parameter. The neighborhood calculations are performed by a previously created space. The parameter can be either visible or invisible for neighborhood calculations. If its invisible, the parameter won't show up as neighbor. Furthermore, the radius within which neighbors are detected as well as the number of neighbors that a parameter may possess can also be defined)

Command: /AssignNeighbors

Parameters v1: string (unique name of swarm)
 string (unique name of parameter)
 string (unique name of space)
 int (parameter visible: 0 or 1)
 float (neighborhood search radius)
 int (maximum number of neighbors)

Remove Neighbors

Deactivate neighborhood calculations for agent parameter.

Command: /RemoveNeighbors

Parameters v1: string (unique name of swarm)
 string (unique name of parameter)
 string (unique name of space)

Set Parameter

Set parameter values. There is an ambiguity concerning the distinction between a SetParameter command that interpolates parameter values over a duration or that randomizes parameter values. This ambiguity occurs when both the second last and last float array contain only one float value. In this case, the last float value is interpreted as duration. In all other situations, when the last float array contains more than one float value, it is interpreted as upper randomization bound.

Command: /SetParameter

Parameters v1: string (unique name of swarm)
 string (unique name of parameter)
 float array (parameter values)

Parameters v2: string (unique name of swarm)

	string	(unique name of parameter)
	float array	(parameter values)
	float	(parameter values interpolation duration)
Parameters v3:	string	(unique name of swarm)
	string	(unique name of parameter)
	float array	(random parameter values lower boundary)
	float array	(random parameter values upper boundary)
Parameters v4:	string	(unique name of swarm)
	int	(agent index)
	string	(unique name of parameter)
	float array	(parameter values)
Parameters v5:	string	(unique name of swarm)
	int	(agent index)
	string	(unique name of parameter)
	float array	(parameter values)
	float	(parameter values interpolation duration)
Parameters v6:	string	(unique name of swarm)
	int	(agent index)
	string	(unique name of parameter)
	float array	(random parameter values lower boundary)
	float array	(random parameter values upper boundary)
Parameters v7:	string	(unique name of swarm)
	int	(agent index range lower boundary)
	int	(agent index range upper boundary)
	string	(unique name of parameter)
	float array	(parameter values)
Parameters v8:	string	(unique name of swarm)
	int	(agent index range lower boundary)
	int	(agent index range upper boundary)
	string	(unique name of parameter)
	float array	(parameter values)
	float	(parameter values interpolation duration)
Parameters v9:	string	(unique name of swarm)
	int	(agent index range lower boundary)
	int	(agent index range upper boundary)
	string	(unique name of parameter)
	float array	(random parameter values lower boundary)
	float array	(random parameter values upper boundary)

Randomize Parameter

Randomize parameter values. This command is useful to avoid an ambiguity in the SetParameter command (v2, v5, v8), which can't distinguish whether the last float parameter value indicates a duration or a randomization upper bound.

Command: /RandomizeParameter

Parameters v1:	string	(unique name of swarm)
	string	(unique name of parameter)
	float array	(random parameter values lower boundary)
	float array	(random parameter values upper boundary)

Parameters v2:	string	(unique name of swarm)
	int	(agent index)
	string	(unique name of parameter)
	float array	(random parameter values lower boundary)
	float array	(random parameter values upper boundary)

Parameters v3:	string	(unique name of swarm)
	int	(agent index range lower boundary)
	int	(agent index range upper boundary)
	string	(unique name of parameter)
	float array	(random parameter values lower boundary)
	float array	(random parameter values upper boundary)

Register Parameter

Register agent or swarm parameter for network communication. Parameters that have been registered send their values via a sender. These values can be normalized if requested.

Command: /RegisterParameter

Parameters v1:	string	(unique name of sender)
	string	(unique name of swarm)
	string	(unique name of parameter)

Parameters v2:	string	(unique name of sender)
	string	(unique name of swarm)
	string	(unique name of parameter)
	float array	(minimum parameter values for normalization)
	float array	(maximum parameter values for normalization)

Parameters v3:	string	(unique name of sender)
	string	(unique name of swarm)
	string	(unique name of parameter)
	int array	(lower and upper range of agents)

Parameters v4:	string	(unique name of sender)
	string	(unique name of swarm)
	string	(unique name of parameter)
	int array	(lower and upper range of agents)
	float array	(minimum parameter values for normalization)
	float array	(maximum parameter values for normalization)

Deregister Parameter

Reregister agent or swarm parameter from network communication.

Command: /DeregisterParameter

Parameters v1: string (unique name of sender)
 string (unique name of swarm)
 string (unique name of parameter)

Add Behavior

Add new agent behavior to swarm. The behavior must be one of the predefined behavior classes. The parameter values that the behavior operates on are specified by two string sequences. The first sequence lists the parameter names from the behavior reads the values. The second sequence lists the parameter names whose values the behavior modifies. If the behavior takes into account the neighbors of one or several parameters, these neighbors are specified by listing the name of the space that calculates the neighbors right after the name of the parameter, separated by a colon (no spaces)

Command: /AddBehavior

Parameters v1: string (unique name of swarm)
 string (unique but arbitrary name of behavior)
 string (class of behavior)
 string (names of input parameter (and neighbor spaces))
 string (names of output parameter (and neighbor spaces))

Remove Behavior

Remove existing agent behavior from swarm.

Command: /RemoveBehavior

Parameters v1: string (unique name of swarm)
 string (unique name of behavior)

Show Swarm

Activate the visualization of an existing swarm. To display the swarm agents, the parameter defining the agent's position needs to be specified. With these settings alone, the agents will be shown as spheres. If you specify the parameter that defines the agent's velocity and thus its orientation, the agents will be shown as oriented pyramids. Apart from the agents themselves, their trails can also be visualized. When show swarm is applied for the first time, the number of trail segments that are specified represents the maximum number of trail segments displayed. For subsequent applications of show swarm the number of trail segments cannot exceed this maximum value. By specifying a trail alpha attenuation factor larger than zero, the trail will gradually become transparent towards the end that is opposite of the agent.

Command: /ShowSwarm

Parameters v1:	string	(unique name of swarm)
	string	(unique name of agent position parameter)
	float array	(color (rgba) of agent with values ranging from 0 to 1)
	float	(size of agents)
Parameters v2:	string	(unique name of swarm)
	string	(unique name of agent position parameter)
	string	(unique name of agent velocity parameter)
	float array	(color (rgba) of agent with values ranging from 0 to 1)
	float	(size of agents)
Parameters v3:	string	(unique name of swarm)
	string	(unique name of agent position parameter)
	float array	(color (rgba) of agent with values ranging from 0 to 1)
	float	(size of agents)
	float array	(color (rgba) of agent trail with values ranging from 0 to 1)
	int	(number of trail segments)
	float	(attenuation factor for the trail alpha value)
Parameters v4:	string	(unique name of swarm)
	string	(unique name of agent position parameter)
	string	(unique name of agent velocity parameter)
	float array	(color (rgba) of agent with values ranging from 0 to 1)
	float	(size of agents)
	float array	(color (rgba) of agent trail with values ranging from 0 to 1)
	int	(number of trail segments)
	float	(attenuation factor for the trail alpha value)

Hide Swarm

Deactivate the visualization of an existing swarm. For a new show swarm command, the trail length value specifies the new maximum number of trail segments.

Command: /HideSwarm

Parameters v1: string (unique name of swarm)

Show Space

Activate the visualization of an existing neighborhood space. If the space calculates distances between points or shapes, the space will be visualized as lines connecting the neighboring agent parameters. If the space is a grid space, the grid values will be visualized as vectors positioned at regular intervals corresponding to the subdivision of the grid.

Command: /ShowSpace

Parameters v1: string (unique name of space)
float array (color (rgba) of space with values ranging from 0 to 1)

Parameters v2:	string	(unique name of space)
	float array	(color (rgba) of space with values ranging from 0 to 1)
	float	(scale factor for grid values to change visible vector length)

Set Display Color

Set background color of the swarm visualization window. The alpha value of this color affects the opacity of the entire window (on OSX)

Command: /SetDisplayColor

Parameters v1:	float array	(background color (rgba) of display window with values ranging from 0 to 1)
----------------	-------------	---

Set Display Position

Set position of virtual camera.

Command: /SetDisplayPosition

Parameters v1:	float array	(three dimensional coordinates of camera)
----------------	-------------	---

Set Display Orientation

Set orientation of virtual camera in Eulerian angles.

Command: /SetDisplayOrientation

Parameters v1:	float array	(three Eulerian angles specifying camera orientation)
----------------	-------------	---

Set Display Zoom

Set zoom of virtual camera.

Command: /SetDisplayZoom

Parameters v1:	float	(zoom factor for camera)
----------------	-------	--------------------------