Message specification

This document describes the behavior, type and parameters of each message.

General idea: We use an own container for each message to avoid unspecific names of container items like "Parameter1". Each message is then packed in an abstract container of type *MsgObject*. The underlying library for sending/receiving is protobuf from Google.

Contents

1 Agent → VR

1.1 Agent movement

Execute a movement of the agent.
Message name: MsgAgentMovement

actionID Int32, a randomizied value to identify the action.

degree Float32, the direction to walk in degree(0 to 360°). This direction is relative to the world- or global coordinate system.

distance Float, the distance to walk.

The execution will be only started by this message, hence it is executed asynchronously. During the execution, the VR sends messages of type *MsgActionExecutationStatus* with *status=0*. If the execution is finished, the agent will receive a message of type *MsgActionExecutationStatus* with *status=1*.

1.2 Agent movement to a certain position

Execute a movement of the agent to a certain position along a path. Currently only usable in the SpartialCognition scene which has space grid and A* search.
Message name: MsgAgentMoveTo

actionID Int32, a randomizied value to identify the action.

x Float value, the X-coordinate of the target point.

y Float value, the Y-coordinate of the target point.

z Float value, the Z-coordinate of the target point.

1.3 Cancel movement to a certain position

Cancels a movement to a certain point (MsgAgentMoveTo).
Message name: MsgAgentCancelMoveTo

actionID Int32, a randomizied value to identify the action.

1.4 Eye movement

This command rotates the eyes of the agent.
Message name: MsgAgentEyemovement

actionID Int32, a randomized value to identify the action.

panLeft Float32, the rotation angle of the left eye in horizontal direction (positive values rotate it leftwards). The view angle range is -30 to +30°.

panRight Float32, the rotation angle of the right eye in horizontal direction (positive values rotate it leftwards). The view angle range is -30 to +30°.

tilt Float32, the rotation angle of the left and right eye in vertical direction. The view angle range is -30 to +30°.

The eye movement is executed immediately, hence the VR sends no messages of *MsgActionExecutationStatus* because nobody would see the animation of the eyes. This command should also execute an animation to move/rotate the head in the desired direction.

1.5 Eye fixation

This command fixates the eyes at a certain point in the world coordinate system.
Message name: MsgAgentEyefixation

actionID Int32, a randomized value to identify the action.

targetX Float32, the X-coordinate of the target point.

targetY Float32, the Y-coordinate of the target point.

targetZ Float32, the Z-coordinate of the target point.

The VR executes the fixation immediately and hold it until a message of type *MsgAgentEyemovement* occurs.

Because nobody would see the animation of the eyes, this command should also execute an animation to move/rotate the head in the desired direction. At the moment, there is no check if the eyes can really be rotated in the specific direction, hence the agent could also see things at its back.

1.6 Environment Reset

Send a command that sets the VR back to a chosen state. This message should be used for restarting the whole experiment.
Message name: MsgEnvironmentReset

Type Optional Int32, a parameter that selects one configuration if there are more than one.

1.7 Trial Reset

Send a command that sets partly the VR back to an chosen state. This message should be used for starting a new trial in an experiment.
Message name: MsgTrialReset

Type Optional Int32, a parameter that selects one configuration if there are more than one.

1.8 Grasp an object at a certain position

Try to grasp an object at the given position, specified in the viewfield coordinate system of the left eye.
Message name: MsgAgentGraspPos

actionID Int32, the value to identify the current action. Coordinate 0/0 is in the upper left corner.

targetX Float value, the X-coordinate of the position in the coordinate system of the left eye.

targetY Float value, the Y-coordinate of the position in the coordinate system of the left eye.

1.9 Grasp a certain object

Try to grasp an object with the given ID.
Message name: MsgAgentGraspID

actionID Int32, the value to identify the current action.

objectID Int32, the value to identify the target object. The mapping value to unity object depends on the current scenario.

1.10 Release a grasped object

Releases an object if grasped and held.
Message name: MsgAgentGraspRelease

actionID Int32, the value to identify the current action.

1.11 Point at a certain position

Point at a position, specified in the viewfield coordinate system of the left eye. Coordinate 0/0 is in the upper left corner.
Message name: MsgAgentPointPos

actionID Int32, the value to identify the current action.

targetX Float value, the X-coordinate of the position in the coordinate system of the left eye.

targetY Float value, the Y-coordinate of the position in the coordinate system of the left eye.

1.12 Point at a certain object

Point at an object with the given ID.
Message name: MsgAgentPointID

actionID Int32, the value to identify the current action.

objectID Int32, the value to identify the target object. The mapping value to unity object depends on the current scenario.

1.13 Interact with an object at a certain position

Interact with an object at a given position, specified in the viewfield coordinate system of the left eye. Coordinate 0/0 is in the upper left corner. The type of interaction should be spezified and implemented in the VR itself.
Message name: MsgAgentInteractionPos

actionID Int32, the value to identify the current action.

targetX Float value, the x coordinate of the position in the coordinate system of the left eye.

targetY Float value, the y coordinate of the position in the coordinate system of the left eye.

1.14 Interact with a certain object

Interact with an object with the given ID. The type of interaction should be spezified and implemented in the VR itself.
Message name: MsgAgentInteractionID

actionID Int32, the value to identify the current action.

objectID Int32, the value to identify the target object. The mapping value to unity object depends on the current scenario.

1.15 Turn

Turns the Agent around the vertical axis.
Message name: MsgAgentTurn

actionID Int32, the value to identify the current action.

degree Float value, the angle for the clockwise turn relative to the current direction of the agent.

2 VR → Agent

2.1 External reward

Delivers reward to the agent.
Message name: MsgReward

reward float, the user-specified external reward for the agent.

2.2 Grid sensor

This delivers the coordinates of the agent, like a GPS device. It is standardly disabled and could be enabled by *AgentScript::SendGridPosition=true*.
Message name: MsgGridPosition

targetX Float32, the X-coordinate of the agent.

targetY Float32, the Y-coordinate of the agent.

targetZ Float32, the Z-coordinate of the agent.

targetRotationX Float32, the X-coordinate of the agent's rotation in the world.

targetRotationY Float32, the Y-coordinate of the agent's rotation in the world.

targetRotationZ Float32, the Z-coordinate of the agent's rotation in the world.

2.3 Action execution status

Delivers the execution status of the last action.
Message name: MsgActionExecutationStatus

actionID Int32, the value to identify the action.

status Int32, An enum describing the execution status of the action: 0= in execution; 1 = finished; 2 = aborted because of a new action of the same type.

2.4 Collision

Detect an collision.
Message name: MsgCollision

actionID Int32, the value to identify the current action.

colliderID Int32, the ID of the collided item.

2.5 Eye position

Delivers status information of the eyes of the agent.
Message name: MsgEyePosition

rotationPositionX Float value, X-coordinate of current rotation positon.

rotationPositionY Float value, Y-coordinate of current rotation positon.

rotationPositionZ Float value, Z-coordinate of current rotation positon.

rotationVelocityX Float value, X-coordinate of current rotation alteration (per frame).

rotationVelocityY Float value, Y-coordinate of current rotation alteration (per frame).

rotationVelocityZ Float value, Z-coordinate of current rotation alteration (per frame).

2.6 Head Motion

Delivers status information of the agent.
Message name: MsgHeadMotion

accelerationX Float value, speedup X-coordinate of current movement (per frame).

accelerationY Float value, speedup Y-coordinate of current movement (per frame).

accelerationZ Float value, speedup Z-coordinate of current movement (per frame).

rotationAccelerationX Float value, speedup X-coordinate of current rotation (per frame).

rotationAccelerationY Float value, speedup Y-coordinate of current rotation (per frame).

rotationAccelerationZ Float value, speedup Z-coordinate of current rotation (per frame).

rotationVelocityX Float value, X-coordinate of current rotation alteration (per frame).

rotationVelocityY Float value, X-coordinate of current rotation alteration (per frame).

rotationVelocityZ Float value, X-coordinate of current rotation alteration (per frame).

velocityX Float value, X-coordinate of current movement alteration (per frame).

velocityY Float value, Y-coordinate of current movement alteration (per frame).

velocityZ Float value, Z-coordinate of current movement alteration (per frame).

2.7 Image

Send stereo image data. Make sure that the bool member *SendImages* in the function *void Update ()* of *AgentScript* is true.

Message name: MsgImages

leftImage Byte [], the byte data of the left image in png-format.

rightImage Byte [], the byte data of the right image in png-format.

2.8 Menu item

Send a command created by the user controlling the VR.
Message name: MsgMenu

eventID Int32, an enum to identify the event: 0 = start simulation, 1= stop simulation.

parameter Optional string, an string to send additional parameters.

3 Data container

A container class which can contain all types of messages. It can also contain a debug message. VR → Agent. Agent → VR.
Message name: MsgObject

msgAgentMovement Optional MsgAgentMovement

msgAgentMoveTo Optional MsgAgentMoveTo

msgAgentCancelMoveTo Optional MsgAgentCancelMoveTo

msgAgentEyemovement Optional MsgAgentEyemovement

msgAgentEyefixation Optional MsgAgentEyefixation

msgEnvironmentReset Optional MsgEnvironmentReset

msgTrialReset Optional MsgTrialReset

msgAgentGraspPos Optional MsgAgentGraspPos

msgAgentGraspID Optional MsgAgentGraspID

msgAgentGraspRelease Optional MsgAgentGraspRelease

msgAgentPointPos Optional MsgAgentPointPos

msgAgentPointID Optional MsgAgentPointID

msgAgentInteractionPos Optional MsgAgentInteractionPos

msgAgentInteractionID Optional MsgAgentInteractionID

msgAgentTurn Optional MsgAgentTurn

msgReward Optional MsgReward

msgGridPosition Optional MsgGridPosition

msgActionExecutationStatus Optional MsgActionExecutationStatus

msgCollision Optional MsgCollision

msgEyePosition Optional MsgEyePosition

msgHeadMotion Optional MsgHeadMotion

msgImages Optional MsgImages

msgMenu Optional MsgMenu

msgAnnarNetwork Optional MsgAnnarNetwork

msgStartSync Optional MsgStartSync

msgStopSync Optional MsgStopSync