# Ranger RoboWorld

## 1 RoboWorld Document

```
## ARENA ASSUMPTIONS ##
The arena is two-dimensional.
The gradient of the ground is 0.0.
The arena has obstacles.
## ROBOT ASSUMPTIONS ##
The robot is a point mass.
## ELEMENT ASSUMPTIONS ##
One quarter of the arena contains obstacles.
The obstacles are point masses.
## MAPPING OF INPUT EVENTS ##
The event obstacle occurs when the distance from the robot to an obstacle is
   less than 0.5 m.
## MAPPING OF OUTPUT EVENTS ##
When the event stop occurs, the velocity of the robot is set to 0 m/s towards
   the orientation of the robot, and the angular velocity of the robot is set
   to 0 rad/s.
## MAPPING OF VARIABLES ##
## MAPPING OF OPERATIONS ##
When the operation move(lv,av) is called, the velocity of the robot is set to
   lv m/s towards the orientation of the robot, and the angular velocity of
   the robot is set to av rad/s.
## MAPPING OF VARIABLES ##
```

## 2 RoboWorld CyPhyCircus Semantics

### 2.1 Channels

$$
\begin{array}{l}
\mathbf{channel}\ getRobotPosition : \mathbb{R} \times \mathbb{R} \\
\mathbf{channel}\ getRobotVelocity : \mathbb{R} \times \mathbb{R} \\
\mathbf{channel}\ getRobotAcceleration : \mathbb{R} \times \mathbb{R} \\
\mathbf{channel}\ getRobotAngles : \mathbb{R} \\
\mathbf{channel}\ getRobotAngularVelocity : \mathbb{R} \\
\mathbf{channel}\ getRobotAngularAcceleration : \mathbb{R}
\end{array}
$$

**channel** $setRobotVelocity : \mathbb{R} \times \mathbb{R}$
**channel** $setRobotAcceleration : \mathbb{R} \times \mathbb{R}$
**channel** $setRobotAngularVelocity : \mathbb{R}$
**channel** $setRobotAngularAcceleration : \mathbb{R}$

$InOut ::= in \mid out$

**channel** $obstacle : InOut$
**channel** $stop : InOut$
**channel** $moveCall : \mathbb{R} \times \mathbb{R}$

**channel** $obstacleTriggered : \mathbb{B}$

## 2.2 Environment

$step : \mathbb{R}$

$obstaclePositions : \mathbb{P}(\mathbb{R} \times \mathbb{R})$
$collisionRadius : \mathbb{R}$

$detectionRadius : \mathbb{R}$

**process** $Environment \mathrel{\widehat=}$ **begin**

### 2.2.1 Environment State

*EnvironmentState*

$pos : \mathbb{R} \times \mathbb{R}$
$vel : \mathbb{R} \times \mathbb{R}$
$acc : \mathbb{R} \times \mathbb{R}$
$ang : \mathbb{R}$
$angVel : \mathbb{R}$
$angAcc : \mathbb{R}$
$time : \mathbb{R}$

**state** $EnvironmentState$

### 2.2.2 Robot Movement

$$
\begin{array}{l}
\mathit{RobotMovement} \mathrel{\widehat{=}} \\
\quad \left( \begin{array}{ll}
\frac{\mathrm{d}pos}{\mathrm{d}t} = vel & \frac{\mathrm{d}vel}{\mathrm{d}t} = acc \\
\frac{\mathrm{d}ang}{\mathrm{d}t} = angVel & \frac{\mathrm{d}angVel}{\mathrm{d}t} = angAcc \\
\frac{\mathrm{d}time}{\mathrm{d}t} = 1 &
\end{array} \right) \\
\qquad \triangle \left( \begin{array}{l}
(\exists\, obs : obstaclePositions \bullet norm\,(pos - obs) < collisionRadius \\
\qquad \land\ vel > 0) \\
\lor\ (time > step)
\end{array} \right)
\end{array}
$$

### 2.2.3 Update Actions that interrupt Robot Movement

$$
\begin{array}{l}
\mathit{CollisionDetection} \mathrel{\widehat{=}} \\
\quad \big(\exists\, obs : obstaclePositions \bullet norm\,(pos - obs) < collisionRadius\big) \mathrel{\&} \\
\qquad vel, acc := (0,0), (0,0) \\
\quad \Box \\
\quad \big(\neg\ \exists\, obs : obstaclePositions \bullet norm\,(pos - obs) \geq collisionRadius\big) \mathrel{\&} \\
\qquad \mathbf{Skip}
\end{array}
$$

$$\mathit{EnvironmentUpdate} \mathrel{\widehat{=}} \mathit{CollisionDetection}$$

### 2.2.4 Communication Actions that occur on the time step

$$
\begin{array}{l}
\mathit{InputTriggers} \mathrel{\widehat{=}} \mathit{Obstacle\_InEventMapping} \\
\mathit{Obstacle\_InEventMapping} \mathrel{\widehat{=}} \\
\quad \big(\exists\, obs : obstaclePositions \bullet norm\,(pos - obs) < detectionRadius\big) \mathrel{\&} \\
\qquad obstacleTriggered.\mathbf{True} \longrightarrow \mathbf{Skip} \\
\quad \Box \\
\quad \big(\neg\ \exists\, obs : obstaclePositions \bullet norm\,(pos - obs) \geq detectionRadius\big) \mathrel{\&} \\
\qquad obstacleTriggered.\mathbf{False} \longrightarrow \mathbf{Skip}
\end{array}
$$

$$
\begin{array}{l}
\mathit{GetPosition} \mathrel{\widehat{=}} getRobotPosition!pos \longrightarrow \mathbf{Skip} \\
\mathit{GetVelocity} \mathrel{\widehat{=}} getRobotVelocity!vel \longrightarrow \mathbf{Skip} \\
\mathit{GetAcceleration} \mathrel{\widehat{=}} getRobotAcceleration!acc \longrightarrow \mathbf{Skip} \\
\mathit{GetAngles} \mathrel{\widehat{=}} getRobotAngles!ang \longrightarrow \mathbf{Skip} \\
\mathit{GetAngularVel} \mathrel{\widehat{=}} getRobotAngularVelocity!angVel \longrightarrow \mathbf{Skip} \\
\mathit{GetAngularAcc} \mathrel{\widehat{=}} getRobotAngularAcceleration!angVel \longrightarrow \mathbf{Skip}
\end{array}
$$

$$
\begin{array}{l}
\mathit{SetVelocity} \mathrel{\widehat{=}} setRobotVelocity?newVel \longrightarrow vel := newVel \\
\mathit{SetAcceleration} \mathrel{\widehat{=}} setRobotAcceleration?newAcc \longrightarrow acc := newAcc \\
\mathit{SetAngularVel} \mathrel{\widehat{=}} setRobotAngularVelocity?newAngVel \longrightarrow angVel := newAngVel \\
\mathit{SetAngularAcc} \mathrel{\widehat{=}} setRobotAngularAcceleration?newAngAcc \longrightarrow angAcc := newAngAcc
\end{array}
$$

$$\begin{array}{l}
\mathit{GetSetVariables} \mathrel{\widehat{=}} \\
\qquad \mathit{GetPosition} \mathrel{\Box} \mathit{GetVelocity} \mathrel{\Box} \mathit{GetAcceleration} \\
\qquad \Box \\
\qquad \mathit{GetAngles} \mathrel{\Box} \mathit{GetAngularVel} \mathrel{\Box} \mathit{GetAngularAcc} \\
\qquad \Box \\
\qquad \mathit{SetVelocity} \mathrel{\Box} \mathit{SetAcceleration} \mathrel{\Box} \mathit{SetAngularVel} \mathrel{\Box} \mathit{SetAngularAcc}
\end{array}$$

$$\begin{array}{l}
\mathit{Communication} \mathrel{\widehat{=}} \\
\qquad \mathit{InputTriggers} \ ; \ \Big( \ \mathit{GetSetVariables} \ ; \ \mathit{Communication} \ \Big)
\end{array}$$

### 2.2.5 Input Event Buffers

$$\begin{array}{l}
\mathit{Obstacle\_Buffer} \mathrel{\widehat{=}} \mathbf{var}\ \mathit{obstacleTrig} : \mathbb{B} \bullet \mathit{obstacleTrig} := \mathbf{False}; \\
\qquad \left( \begin{array}{l}
\mathit{obstacleTriggered}?b \longrightarrow \mathit{obstacleTrig} := b \\
\Box \\
\big(\mathit{obstacleTrig} = \mathbf{True}\big) \mathbin{\&} \mathit{obstacle.in} \longrightarrow \mathbf{Skip}
\end{array} \right) ; \ \mathit{Obstacle\_Buffer} \\
\mathit{InputEventBuffers} \mathrel{\widehat{=}} \mathit{Obstacle\_Buffer}
\end{array}$$

### 2.2.6 Environment Main Action

$$\begin{array}{l}
\mathit{EnvironmentLoop} \mathrel{\widehat{=}} \mathit{time} := 0 \ ; \ \mu X \bullet \\
\qquad \mathit{RobotMovement} \ ; \ \left( \begin{array}{l}
\big(\mathit{time} \leq \mathit{step}\big) \mathbin{\&} \mathit{EnvironmentUpdate} \\
\Box \\
\big(\mathit{time} > \mathit{step}\big) \mathbin{\&} \mathit{Communication} \ ; \ \mathit{time} := 0
\end{array} \right) ; \ X
\end{array}$$

**channelset** $\mathit{triggerChannels} == \{\!| \ \mathit{obstacleTriggered} \ |\!\}$

**nameset** $\mathit{EnvVars} == \{\mathit{pos}, \mathit{vel}, \mathit{acc}, \mathit{ang}, \mathit{angVel}, \mathit{angAcc}, \mathit{time}\}$

$$\begin{array}{l}
\bullet\ \mathit{pos}, \mathit{vel}, \mathit{acc} := (0,0), (0,0), (0,0) \ ; \ \mathit{ang}, \mathit{angVel}, \mathit{angAcc} := 0, 0, 0; \\
\qquad (\mathit{EnvironmentLoop} \ [\![ \ \mathit{EnvVars} \mid \mathit{triggerChannels} \mid \varnothing \ ]\!] \ \mathit{InputEventBuffers}) \\
\qquad\qquad \setminus \mathit{triggerChannels}
\end{array}$$

**end**

## 2.3 Mapping

$\mathbf{process}\ Move\_OperationMapping \mathrel{\widehat=} \mathbf{begin}$

$$\begin{array}{l}
MoveCall \mathrel{\widehat=} \\
\quad moveCall?ls?as \longrightarrow getRobotAngles?yaw \\
\qquad \longrightarrow setRobotVelocity!(ls * (sin\ yaw), ls * (cos\ yaw)) \\
\qquad \longrightarrow setRobotAngularVelocity!as \longrightarrow MoveCall
\end{array}$$

$\bullet\ MoveCall$

$\mathbf{end}$

$\mathbf{process}\ Stop\_OutputEventMapping \mathrel{\widehat=} \mathbf{begin}$

$$\begin{array}{l}
StopEvent \mathrel{\widehat=} \\
\quad stop.out \longrightarrow setRobotVelocity!(0, 0) \\
\qquad \longrightarrow setRobotAngularVelocity!0 \longrightarrow StopEvent
\end{array}$$

$\bullet\ StopEvent$

$\mathbf{end}$

$\mathbf{process}\ Mapping \mathrel{\widehat=} Move\_OperationMapping \interleave Stop\_OutputEventMapping$

## 2.4 Composition

$$\begin{array}{l}
\mathbf{channelset}\ getSetChannels == \{\!| \\
\qquad getRobotPosition, getRobotVelocity, getRobotAcceleration, \\
\qquad getRobotAngles, getRobotAngularVelocity, getRobotAngularAcceleration, \\
\qquad setRobotVelocity, setRobotAcceleration, \\
\qquad setRobotAngularVelocity, setRobotAngularAcceleration \\
\quad |\!\}
\end{array}$$

$$\begin{array}{l}
\mathbf{process}\ RoboWorld \mathrel{\widehat=} \\
\quad (Environment \llbracket\ getSetChannels\ \rrbracket\ Mapping) \setminus getSetChannels
\end{array}$$