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

channel $getRobotPosition : \mathbb{R} \times \mathbb{R}$
channel $getRobotVelocity : \mathbb{R} \times \mathbb{R}$
channel $getRobotAcceleration : \mathbb{R} \times \mathbb{R}$
channel $getRobotAngles : \mathbb{R}$
channel $getRobotAngularVelocity : \mathbb{R}$
channel $getRobotAngularAcceleration : \mathbb{R}$

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

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

2.2.3 Update Actions that interrupt Robot Movement

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

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

2.2.4 Communication Actions that occur on the time step

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

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

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

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

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

2.2.5 Input Event Buffers

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

2.2.6 Environment Main Action

$$
\begin{array}{l}
EnvironmentLoop \mathrel{\widehat=} time := 0 \,;\; \mu X \bullet \\
\quad RobotMovement \,;\; \left(\begin{array}{l} \big(time \le step\big)\ \&\ EnvironmentUpdate \\ \Box \\ \big(time > step\big)\ \&\ Communication \triangle_0\ time := 0 \end{array} \right) ;\; X
\end{array}
$$

$$
\textbf{channelset}\ triggerChannels == \{\!|\ obstacleTriggered\ |\!\}
$$

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

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

end

2.3 Mapping

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

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

$\bullet\ MoveCall$

end

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

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

$\bullet\ StopEvent$

end

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

2.4 Composition

channelset $getSetChannels == \{|$
$\quad getRobotPosition, getRobotVelocity, getRobotAcceleration,$
$\quad getRobotAngles, getRobotAngularVelocity, getRobotAngularAcceleration,$
$\quad setRobotVelocity, setRobotAcceleration,$
$\quad setRobotAngularVelocity, setRobotAngularAcceleration$
$|\}$

process $RoboWorld \mathrel{\widehat=}$
$\quad (Environment \ [\![\ getSetChannels\ ]\!]\ Mapping) \setminus getSetChannels$