Simulate RoadRunner Scenarios with Actors Modeled in MATLAB

You can use a MATLAB ®System object™ to model the behavior of an actor in RoadRunner Scenario. A System object is a specialized MATLAB object designed specifically for implementing and simulating dynamic systems with inputs that change over time. For more information, see What Are System Objects?

In an actor behavior modeled using a MATLAB System object, you can retrieve run-time actor attributes from a scenario and update their values to move an actor around. For example, you can get the velocity of a vehicle by using the getAttribute function, and then increase the velocity until the vehicle catches up to another moving vehicle.

This page contains examples that highlight different actor behaviors modeled in MATLAB. To simulate the actor behavior created in MATLAB, you must connect RoadRunnerand MATLAB. For more information, see Connect MATLAB and RoadRunner to Control and Analyze Simulations.

Model Actor Behavior with Accelerated Motion

Model Actor Behavior with Accelerated Motion

This example shows how to create a MATLAB System object that models a vehicle with accelerated motion.

This example assumes that:

- You have a RoadRunner license and the product is installed. For more information, see Install and Activate RoadRunner(RoadRunner).
- You have a RoadRunner Scenario license and the product is installed.

Set Up Cosimulation Environment

Start RoadRunner application interactively by using the roadrunnerSetup function. When the function opens a dialog box, specify the location of the RoadRunner installation folder and project folder.

```
rrApp = roadrunnerSetup;
```

Add these files to the appropriate folders within your RoadRunner project:

- `Accelerated_Motion.rrscene` — Scene file for the example.
- `AccMotion.rrscenario` — Scenario file built on the scene, Accelerated_Motion.rrscene.
- `Acc.rrbehavior.rrmeta` — Behavior asset file that links behavior in the AcceleratedMotion.m MATLAB System object to a vehicle in the scenario.

```
copyfile("Accelerated_Motion.rrscene",fullfile(rrApp.status.Project.Filename,"Scenes"));
copyfile("AccMotion.rrscenario",fullfile(rrApp.status.Project.Filename,"Scenarios"));
copyfile("Acc.rrbehavior.rrmeta",fullfile(rrApp.status.Project.Filename,"Assets","Behaviors"));
```

Open the scene `Accelerated_Motion.rrscene`.

```
openScene(rrApp,"Accelerated_Motion");
```

Open the scenario `AccMotion.rrscenario`.

```
openScenario(rrApp,"AccMotion");
```

Connect to the RoadRunner Scenario server to enable cosimulation by using the createSimulation function.

```
ss = createSimulation(rrApp);
```

Start the simulation.

```
set(ss,"SimulationCommand","Start");
```

MATLAB System object for Actor with Accelerated Motion

The programming logic that enables a vehicle actor to display accelerated motion is implemented by MATLAB System object AcceleratedMotion. Hence, this MATLABSystem object represents the behavior of the actor in a RoadRunner Scenario. For more information about the working of a MATLAB System object, see Create SystemObjects.

MATLAB System object Code for Actor with Accelerated Motion

The main code snippets in `AcceleratedMotion` are explained below:

Ensure that the MATLAB System object executes in sync with the scenario by setting its sample time to the step size of the scenario.
Find the `ScenarioSimulation` object that corresponds to the scenario opened in RoadRunner by using the `Simulink.ScenarioSimulation.find` function. Get the stepsize of the scenario and assign it to the sample time of the object `obj`.

```
function st = getSampleTimeImpl(obj) obj.mScenarioSimulationHdl = ...
Simulink.ScenarioSimulation.find("ScenarioSimulation"); obj.mStepSize =
obj.mScenarioSimulationHdl.get("StepSize"); st = createSampleTime(obj, ...
"Type","Discrete","SampleTime",obj.mStepSize);
end
```

Retrieve the `ActorSimulation` object to which the MATLAB System object is attached.
In `setupImpl`, call the `Simulink.ScenarioSimulation.find` function to find the ActorSimulation object to which MATLAB System object `obj` is attached. During scenario simulation, MATLAB calls the function `setupImpl` only once, at the start. For more information about the working of a MATLAB System object, see Create SystemObjects.

```
obj.mActorSimulationHdl = Simulink.ScenarioSimulation.find( ... "ActorSimulation","SystemObject",obj);
```

Retrieve and store the current values of pose and velocity of the actor in the scenario by using the `getAttribute` function.

```
obj.mActor.pose = ... obj.mActorSimulationHdl.getAttribute("Pose");
obj.mActor.velocity = ... obj.mActorSimulationHdl.getAttribute("Velocity");
```

Calculate the simulation time elapsed since the last step to update the distance moved by the vehicle in this time.
During scenario simulation, MATLAB calls the `stepImpl` function at each time step. The `stepImpl` function contains the programming logic that enables the behavior of the vehicle actor. In the `stepImpl` function, calculate the simulation time elapsed since the last step. Then, assign the current time to the last simulation time so that the elapsed time can be calculated in the next iteration of the `stepImpl` function.

```
currentTime = obj.getCurrentTime;
elapsedTime = currentTime - obj.mLastTime;
obj.mLastTime = currentTime;
```

Get the pose and velocity of the actor in the previous time step.

```
velocity = obj.mActor.velocity; pose = obj.mActor.pose;
```

Calculate the current pose to move the vehicle forward.
If the simulation time is less than 3 seconds, update the translation component of the actor pose matrix to accommodate the distance that the vehicle traveled in the elapsed time. The distance traveled in the *x*- direction, `pose(1,4)`, is the *x*- component of the current velocity multiplied by the elapsed time. The distance travelled in the *y*-direction, `pose(2,4)`, is the *y*- component of the current velocity multiplied by the elapsed time, and so on.
If the simulation time is more than 3 seconds, then also multiply the translation component of the actor pose matrix by a factor of 3 to accelerate the car.

For more information about the actor pose matrix, see What Is a RoadRunner Pose Matrix?

```
if(currentTime < 3.0) pose(1,4) = pose(1,4) + velocity(1) * elapsedTime;
```