⌂ / Simulation / ParkingArea

ParkingArea

Parking Areas

Areas for parking outside the road network (either road-side parking or car parks) can be defined using the `<parkingArea>` element. This accomplishes the following purposes:

- arbitrary parking positions and angles can be defined to visualize fishbone or parallel parking
- parking space outside the road network can be limited to a set capacity
- automatic rerouting to an alternative parking area can be triggered whenever a parking area becomes full

Definition

A road-side parkingArea is defined as in the following:

```
<parkingArea id="ParkAreaA" lane="a_0" startPos="200" endPos="250" roadsideCapacity="5" angle="45" length="30"/>
```

Additionally, individual parking spaces can be defined:

```
<parkingArea id="ParkAreaB" lane="b_0" startPos="240" endPos="260" roadsideCapacity="0" width="5" length="10" angle="30">
    <space x="853" y="623"/>
    <space x="863" y="618"/>
    <space x="873" y="613"/>
    <space x="883" y="608"/>
    <space x="893" y="603"/>
    <space x="848" y="611" width="4" length="8" angle="120"/>
    <space x="858" y="606" width="4" length="8" angle="120"/>
    <space x="868" y="601" width="4" length="8" angle="120"/>
    <space x="878" y="596" width="4" length="8" angle="120"/>
    <space x="888" y="591" width="4" length="8" angle="120"/>
</parkingArea>
```

The total capacity of a parking area is given by the sum of its *roadsideCapacity* and the number of its `<space>` elements.

The parkingArea supports the following attributes:

Attribute Name	Value Type	Value Range	Default	Description
id	string	id		The name of the parking area; must be unique
lane	string	valid lane id		The name of the lane the parking area shall be located at
startPos	float	-lane.length < x < lane.length (negative values count backwards from the end of the lane)	0	The begin position on the lane (the lower position on the lane) in meters
endPos	float	-lane.length < x < lane.length (negative values count backwards from the end of the lane)	lane.length	The end position on the lane (the higher position on the lane) in meters, must be larger than *startPos* by more than 0.1m
friendlyPos	bool	*true,false*	*false*	whether invalid stop positions should be corrected automatically (default *false*)
name	string	simple String		Arbitrary text to describe the parking area. This is only used for visualization purposes.

Attribute Name	Value Type	Value Range	Default	Description
roadsideCapacity	int	non-negative	1 (if no parking space is defined, else 0)	The number of parking spaces for road-side parking
onRoad	bool		false	Whether vehicles remain on the road while parking. **Note:** If set to *true*, only roadsideCapacity is used and no `<space>`-definitions are permitted.
width	float	positive	3.2	The width of the road-side parking spaces
length	float	positive	(endPos - startPos) / roadsideCapacity	The length of the road-side parking spaces
angle	float (degree)		0	The angle of the road-side parking spaces relative to the lane angle, positive means clockwise
lefthand	bool		*false*	Whether road-side parking spaces should be drawn on the left side of the lane
acceptedBadges	string list			The set of keywords which give access to the parking area (an empty value means free access to all)

Custom parking spaces

The space element supports the following attributes:

Attribute Name	Value Type	Value Range	Default	Description
x	float			The x-position in meters of the parking vehicle
y	float			The y-position in meters of the parking vehicle
z	float		0	The z-position in meters of the parking vehicle
width	float		width value of the parent parking area	The width of the parking space
length	float		length value of the parent parking area	The length of the parking space
angle	float (degree)		absolute angle of the parent parking area (lane angle + angle attribute)	Absolute angle of the parking space
slope	float (degree)		0	Slope angle of the parking space

Caution
Please note that parking areas must be added to a config via the *--additional-files* parameter (see *additional-file*).

Letting Vehicles stop at a parking area

To declare a vehicle that stops at a parkingPlace, a `<stop>`-definition must be part of the vehicle or its route:

```
<vehicle id="0" depart="0">
    <route edges="e1 e2 e3"/>
    <stop parkingArea="pa0" duration="20"/>
</vehicle>
```

What is defined here is a vehicle named "0" that stops as at parkingArea "pa0". Note, that the lane of that parking area must belong to one of the edges "e1, e2, e3" of the vehicles route.

For a complete list of attributes for the "stop"-element of a vehicle see Definition_of_Vehicles,_Vehicle_Types,_and_Routes#Stops.

Modelling Maneuvering Times when Entering and Leaving the Parking Space

When setting the (boolean) option **--parking.maneuver**, vehicles will spend extra time on the road when leaving and entering a parkingArea. This time depends on the angle of the parking lot relative to the road lane and can be configured with the vType attribute *maneuverAngleTimes*. This is a comma-separated list of number-triplets of the form *ANGLE ENTERINGTIME LEAVINGTIME*:

```
<vType id="example" maneuverAngleTimes="10 3.0 4.0,80 1.6 11.0,110 11.0 2.0,170 8.1 3.0,181 3.0 4.0"/>
```

The value corresponding to the closest angle is used. The value of maneuverAngleTimes is initialized to a vClass-specific value:

- default: `maneuverAngleTimes="10 3 4,80 1 11,110 11 2,170 8 3,181 3 4"`
- truck, trailer, coach, delivery: all times doubled compared to default
- bicycle, moped: `maneuverAngleTimes="181 1 1"`

Rerouting when the current parkingArea is full

If a vehicle reaches a parkingArea that is filled to capacity it must wait on the road until a space becomes available or reroute to a new parking area.

Restricting the access to the parking area

In the real world, the access to parkings may be restricted to different kinds of users, such as residents, customers or employees. As these access rights may vary a lot depending on the location, they are modeled using a separate permission system from the vehicle classes. The access to the parking area can be restricted in the following way:

- The parking area defines a set of accepted badges (keywords) which model single access rights in the attribute **acceptedBadges**, e.g. `acceptedBadges="clients_supermarket employees_supermarket"`
- The vehicles or vehicle types define the set of owned badges which give them access, e.g. `parkingBadges="employees_supermarket home_johndoe"`
- Any parking area which is not restricted (empty **acceptedBadges** attribute) or where at least one of the accepted badges is present in the vehicle or vehicle type definition can be accessed

Caution
Parking badges defined in the vehicle definition override vehicle type settings.

Importing / Generating Parking Areas

The following tools exist to obtain parking area definitions

- generateParkingAreas.py: probabilistic generation for a whole network
- generateParkingLots.py: generate space definitions to fill up a given shape
- netedit: define parking areas visually
- netconvert --parking-output: imports roadside parking areas from OSM
- SAGA generateParkingAreasFromOSM.py: imports all kinds of parking areas from OSM

TraCI

Some information regarding parking areas can be accessed directly using traci.simulation.getParameter() calls.

- **parkingArea.capacity**: total number of parking spaces (roadsideCapacity + number of `<space>` elements)
- **parkingArea.occupancy**: number of vehicles parking at this parking area

This page was last edited on: 12 December 2024

Follow us on | Documentation built with MkDocs