Dynamic Obstacle Avoidance - Test Specification Report

Gianvincenzo Daddabbo, Gaetano Gallo, Alberto Ruggeri, Martina Tedesco, Alessandro Toschi

26-May-2021 10:06:07

Table of Contents

<u>1. Dynamic_obstacle_avoidance</u>	3
1.1. Dynamic_obstacle 100km/h	3
<u>1.1.1. 0° 100km/h</u>	3
<u>1.1.2. 20° 100km/h</u>	5
<u>1.1.3. 45° 100km/h</u>	6
<u>1.1.4. 70° 100km/h</u>	8
<u>1.1.5. 90° 100km/h</u>	9
<u>1.1.6. 110° 100km/h</u>	11
<u>1.1.7. 135° 100km/h</u>	12
<u>1.1.8. 160° 100km/h</u>	14
<u>1.1.9. 180° 100km/h</u>	15
<u>1.1.1020° 100km/h</u>	17
<u>1.1.1145° 100km/h</u>	19
<u>1.1.1270° 100km/h</u>	20
<u>1.1.1390° 100km/h</u>	22
<u>1.1.14110° 100km/h</u>	23
<u>1.1.15135° 100km/h</u>	25
<u>1.1.16160° 100km/h</u>	26
1.1.17. 1000m curvature clockwise 100km/h	28
1.1.18. 500m curvature clockwise 100km/h	30
1.1.19. 300m curvature clockwise 100km/h	31
1.1.20. 300m curvature counterclockwise 100km/h	33
1.1.21. 500m curvature counterclockwise 100km/h	34
1.1.22. 1000m curvature counterclockwise 100km/h	36
1.2. Dynamic obstacle 40km/h	37
<u>1.2.1. 0° 40km/h</u>	37
<u>1.2.2. 20° 40km/h</u>	39
<u>1.2.3. 45° 40km/h</u>	40
<u>1.2.4. 70° 40km/h</u>	42
<u>1.2.5. 90° 40km/h</u>	43
<u>1.2.6. 110° 40km/h</u>	45
<u>1.2.7. 135° 40km/h</u>	47
<u>1.2.8. 160° 40km/h</u>	48
<u>1.2.9. 180° 40km/h</u>	50
<u>1.2.1020° 40km/h</u>	51
<u>1.2.1145° 40km/h</u>	53
<u>1.2.1270° 40km/h</u>	54
1.2.1390° 40km/h	56
1.2.14110° 40km/h	58
1.2.15135° 40km/h	59
1.2.16160° 40km/h	61
1.2.17. 1000m curvature clockwise 40km/h	62
1.2.18. 500m curvature clockwise 40km/h	

1.2.19. 300m curvature clockwise 40km/h	65
1.2.20. 300m curvature counterclockwise 40km/h	
1.2.21. 500m curvature counterclockwise 40km/h	
1.2.22. 1000m curvature counterclockwise 40km/h	

1. Dynamic_obstacle_avoidance

Test Details

_	•	. •	
Desc	rın	tı∩	n
レしろし	ין ו	CIO	

This report describes the tests performed for the Obstacle Avoidance regarding a dynamic obstacle moving at a lower speed respect to the ego-vehicle considering different simple scenarios at 40km/h and 100km/h.

1.1. Dynamic_obstacle 100km/h

1.1.1.0° 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
map = [0 0; 1000 0];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y_{rec}(end+1:end+p+20) = Y_{rec}(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6:
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.2. 20° 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
map = [0 \ 0; 1000 \ 364];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+p+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0_{kin} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map, V obst, Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start_dyn_obst
       start_dyn_obst1;
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.3.45° 100km/h

PostLoad Callback

%% Set Speed V = 100/3.6; %% Scenario Loading map = [0 0; 1000 1000];

% Evaluate total distance covered by the route on the map distance = odometer(map);

%% Reference signal

% Upsample map based on speed and timestep

[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);

% Extend the reference signal to avoid index over limits

 $X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});$

Y rec(end+1:end+p+20) = Y rec(end);

Theta_rec(end+1:end+p+20) = Theta_rec(end);

% Define initial condition based on map

```
x0_{kin} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
           point B1;
dyn_obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point_B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start_dyn_obst
       start dyn obst];
X_obstacle_dyn = dyn_obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.4. 70° 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
map = [0 0; 1000 2747];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eve(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map,V obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point B:end)...
          repmat(V_obst,length(X_ostacolo(point_B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start dyn obst
```

```
start_dyn_obst];
X_obstacle_dyn = dyn_obstacle1(:,1);
Y_obstacle_dyn = dyn_obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.5. 90° 100km/h

PostLoad Callback

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
map = [0 0; 0 1000];
```

% Evaluate total distance covered by the route on the map distance = odometer(map); %% Reference signal % Upsample map based on speed and timestep [X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);

```
% Extend the reference signal to avoid index over limits
X rec(end+1:end+p+20) = X_rec(end);
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6:
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point_B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start_dyn_obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.6. 110° 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
map = [0\ 0; -364\ 1000];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map,V obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B];
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.7. 135° 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
map = [0 0; -1000 1000];
```

```
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y_{rec}(end+1:end+p+20) = Y_{rec}(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0_{dyn} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V 0 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start_dyn_obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.8. 160° 100km/h

```
%% Set Speed
V = 100/3.6:
%% Scenario Loading
map = [0 0; -2747 1000];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0_{dyn} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V 0 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.9. 180° 100km/h

PostLoad Callback

%% Set Speed V = 100/3.6;

```
%% Scenario Loading
map = [0 0; -1000 0];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point B:end)...
          repmat(V_obst,length(X_ostacolo(point_B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start dyn obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.10. -20° 100km/h

PostLoad Callback

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
map = [0 0; 1000 -364];
```

% Evaluate total distance covered by the route on the map distance = odometer(map);

%% Reference signal

% Upsample map based on speed and timestep

[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);

% Extend the reference signal to avoid index over limits

 $X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});$

Y rec(end+1:end+p+20) = Y rec(end);

Theta_rec(end+1:end+p+20) = Theta_rec(end);

% Define initial condition based on map

```
x0_{kin} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map,V obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
           point B1;
dyn_obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start_dyn_obst = [X_ostacolo(point_B) Y_ostacolo(point_B) V_obst];
obstacle = [start dyn obst
       start_dyn_obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta_obstacle_dyn = dyn_obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.11. -45° 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
map = [0 \ 0; 1000 - 1000];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eve(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map,V obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point B:end)...
          repmat(V_obst,length(X_ostacolo(point_B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start dyn obst
```

```
start_dyn_obst];
X_obstacle_dyn = dyn_obstacle1(:,1);
Y_obstacle_dyn = dyn_obstacle1(:,2);
theta_obstacle_dyn = dyn_obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.12. -70° 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
map = [0 0; 1000 -2747];

% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
```

```
% Extend the reference signal to avoid index over limits
X rec(end+1:end+p+20) = X_rec(end);
Y_rec(end+1:end+p+20) = Y_rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eve(6)*1000;
% Obstacle definition
V obst = 50/3.6:
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point_B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start_dyn_obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.13. -90° 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
map = [0 \ 0; 0 \ -1000];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

$1.1.14.-110^{\circ}\ 100 km/h$

PostLoad Callback

%% Set Speed V = 100/3.6; %% Scenario Loading map = [0 0; -364 -1000];

```
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X_{rec}(end+1:end+p+20) = X_{rec}(end);
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0_{dyn} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V 0 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map, V obst, Ts);
point B = round(length(X ostacolo)*0.4);
number_obstacles = [point_B
           point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point B:end)...
         repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start dyn obst
      start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.15. -135° 100km/h

```
%% Set Speed
V = 100/3.6:
%% Scenario Loading
map = [0 0; -1000 -1000];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0_{dyn} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V 0 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.16. **-160° 100km/h**

PostLoad Callback

%% Set Speed V = 100/3.6;

```
%% Scenario Loading
map = [0 0; -2747 -1000];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point_B:end)...
          repmat(V_obst,length(X_ostacolo(point_B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start dyn obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.17. 1000m curvature clockwise 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
[X_rec, Y_rec, Theta_rec] = curve_generator(-1000,V,Ts);
map = [X_rec Y_rec];
distance = odometer(map);
% Extend the reference signal to avoid index over limits
X_rec(end+1:end+p+20) = X_rec(end);
Y_rec(end+1:end+p+20) = Y_rec(end);
Theta_rec(end+1:end+p+20) = Theta_rec(end);
% Define initial condition based on map
x0_kin = [X_rec(1) Y_rec(1) Theta_rec(1) V]';
x0_dyn = [X_rec(1) Y_rec(1) Theta_rec(1) V 0 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
eqoStates.Plant = x0 kin';
```

```
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V_{obst} = 50/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map,V obst,Ts);
point B = round(length(X ostacolo)*0.4);
number_obstacles = [point_B
           point B1;
dyn_obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point B:end)...
         repmat(V obst,length(X ostacolo(point B:end)),1)];
start_dyn_obst = [X_ostacolo(point_B) Y_ostacolo(point_B) V_obst];
obstacle = [start_dyn_obst
      start_dyn_obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta_obstacle_dyn = dyn_obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.18. 500m curvature clockwise 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
[X rec, Y rec, Theta rec] = curve generator(-500,V,Ts);
map = [X rec Y rec];
distance = odometer(map);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map, V obst, Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn_obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point B:end)...
          repmat(V_obst,length(X_ostacolo(point_B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start dyn obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y_obstacle_dyn = dyn_obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.19. 300m curvature clockwise 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
[X_rec, Y_rec, Theta_rec] = curve_generator(-300,V,Ts);
map = [X_rec Y_rec];
distance = odometer(map);
% Extend the reference signal to avoid index over limits
X_rec(end+1:end+p+20) = X_rec(end);
Y_rec(end+1:end+p+20) = Y_rec(end);
Theta_rec(end+1:end+p+20) = Theta_rec(end);
% Define initial condition based on map
x0_kin = [X_rec(1) Y_rec(1) Theta_rec(1) V]';
x0_dyn = [X_rec(1) Y_rec(1) Theta_rec(1) V 0 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
```

```
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V_{obst} = 50/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map,V obst,Ts);
point B = round(length(X ostacolo)*0.4);
number_obstacles = [point_B
           point B1;
dyn_obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point B:end)...
         repmat(V obst,length(X ostacolo(point B:end)),1)];
start_dyn_obst = [X_ostacolo(point_B) Y_ostacolo(point_B) V_obst];
obstacle = [start_dyn_obst
      start_dyn_obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta_obstacle_dyn = dyn_obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.20. 300m curvature counterclockwise 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
[X rec, Y rec, Theta rec] = curve generator(300,V,Ts);
map = [X rec Y rec];
distance = odometer(map);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map, V obst, Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start_dyn_obst = [X_ostacolo(point_B) Y_ostacolo(point_B) V_obst];
obstacle = [start dyn obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta_obstacle_dyn = dyn_obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.21. 500m curvature counterclockwise 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
[X_rec, Y_rec, Theta_rec] = curve_generator(500,V,Ts);
map = [X_rec Y_rec];
distance = odometer(map);
% Extend the reference signal to avoid index over limits
X_rec(end+1:end+p+20) = X_rec(end);
Y_rec(end+1:end+p+20) = Y_rec(end);
Theta_rec(end+1:end+p+20) = Theta_rec(end);
% Define initial condition based on map
x0_kin = [X_rec(1) Y_rec(1) Theta_rec(1) V]';
x0_dyn = [X_rec(1) Y_rec(1) Theta_rec(1) V 0 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
eqoStates.Plant = x0 kin';
```

```
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V_{obst} = 50/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map,V obst,Ts);
point B = round(length(X ostacolo)*0.4);
number_obstacles = [point_B
           point B1;
dyn_obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point B:end)...
         repmat(V obst,length(X ostacolo(point B:end)),1)];
start_dyn_obst = [X_ostacolo(point_B) Y_ostacolo(point_B) V_obst];
obstacle = [start_dyn_obst
      start_dyn_obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta_obstacle_dyn = dyn_obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.1.22. 1000m curvature counterclockwise 100km/h

```
%% Set Speed
V = 100/3.6;
%% Scenario Loading
[X rec, Y rec, Theta rec] = curve generator(1000,V,Ts);
map = [X rec Y rec];
distance = odometer(map);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 50/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map, V obst, Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start_dyn_obst = [X_ostacolo(point_B) Y_ostacolo(point_B) V_obst];
obstacle = [start dyn obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta_obstacle_dyn = dyn_obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2. Dynamic_obstacle 40km/h

1.2.1. 0° 40km/h

PostLoad Callback

%% Set Speed V = 40/3.6; %% Scenario Loading map = [0 0; 1000 0];

% Evaluate total distance covered by the route on the map distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X rec(end+1:end+p+20) = X rec(end);

```
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec(1)} Y \text{ rec(1)} Theta \text{ rec(1)} V]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0 kin';
eqoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number_obstacles = [point_B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point_B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start_dyn_obst = [X_ostacolo(point_B) Y_ostacolo(point_B) V_obst];
obstacle = [start dyn obst
       start dyn obst];
X_obstacle_dyn = dyn_obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.2. 20° 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
map = [0.0; 1000 364];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map, V obst, Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B];
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.3.45° 40km/h

PostLoad Callback

%% Set Speed V = 40/3.6; %% Scenario Loading map = [0 0; 1000 1000];

```
% Evaluate total distance covered by the route on the map
distance = odometer(map):
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y_{rec}(end+1:end+p+20) = Y_{rec}(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0_{dyn} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V 0 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eve(6)*1000;
% Obstacle definition
V obst = 10/3.6:
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn_obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start_dyn_obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.4. 70° 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
map = [0 0; 1000 2747];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0_{dyn} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V 0 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.5. 90° 40km/h

PostLoad Callback

%% Set Speed V = 40/3.6;

```
%% Scenario Loading
map = [0 \ 0; 0 \ 1000];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point B:end)...
          repmat(V_obst,length(X_ostacolo(point_B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start dyn obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.6. 110° 40km/h

PostLoad Callback

%% Set Speed V = 40/3.6; %% Scenario Loading map = [0 0; -364 1000];

% Evaluate total distance covered by the route on the map distance = odometer(map);

%% Reference signal

% Upsample map based on speed and timestep

[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);

% Extend the reference signal to avoid index over limits

 $X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});$

Y rec(end+1:end+p+20) = Y rec(end);

Theta_rec(end+1:end+p+20) = Theta_rec(end);

% Define initial condition based on map

```
x0_{kin} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
           point B1;
dyn_obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point_B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start_dyn_obst
       start dyn obst];
X_obstacle_dyn = dyn_obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.7. 135° 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
map = [0 0; -1000 1000];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eve(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map,V obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point B:end)...
          repmat(V_obst,length(X_ostacolo(point_B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start dyn obst
```

```
start_dyn_obst];
X_obstacle_dyn = dyn_obstacle1(:,1);
Y_obstacle_dyn = dyn_obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.8. 160° 40km/h

PostLoad Callback

%% Set Speed

```
V = 40/3.6;
%% Scenario Loading
map = [0 0; -2747 1000];
% Evaluate total distance covered by the route on the map distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
```

```
% Extend the reference signal to avoid index over limits
X rec(end+1:end+p+20) = X_rec(end);
Y_rec(end+1:end+p+20) = Y_rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eve(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point_B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start_dyn_obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.9. 180° 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
map = [0 0; -1000 0];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map, V obst, Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B];
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.10. -20° 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
map = [0 0; 1000 -364];
```

```
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y_{rec}(end+1:end+p+20) = Y_{rec}(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eve(6)*1000;
% Obstacle definition
V_{obst} = 10/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map, V obst, Ts);
point B = round(length(X ostacolo)*0.4);
number_obstacles = [point_B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start_dyn_obst = [X_ostacolo(point_B) Y_ostacolo(point_B) V_obst];
obstacle = [start dyn obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.11. -45° 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
map = [0 \ 0; 1000 - 1000];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0_{dyn} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V 0 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.12. -70° 40km/h

PostLoad Callback

%% Set Speed V = 40/3.6;

```
%% Scenario Loading
map = [0\ 0;\ 1000\ -2747];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point_B:end)...
          repmat(V_obst,length(X_ostacolo(point_B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start dyn obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.13. -90° 40km/h

PostLoad Callback

%% Set Speed V = 40/3.6; %% Scenario Loading map = [0 0; 0 -1000];

% Evaluate total distance covered by the route on the map distance = odometer(map);

%% Reference signal

% Upsample map based on speed and timestep

[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);

% Extend the reference signal to avoid index over limits

 $X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});$

Y rec(end+1:end+p+20) = Y rec(end);

Theta_rec(end+1:end+p+20) = Theta_rec(end);

% Define initial condition based on map

```
x0_{kin} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map,V obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
           point B1;
dyn_obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start_dyn_obst = [X_ostacolo(point_B) Y_ostacolo(point_B) V_obst];
obstacle = [start dyn obst
       start_dyn_obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta_obstacle_dyn = dyn_obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.14. -110° 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
map = [0.0; -364 - 1000];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eve(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map,V obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point B:end)...
          repmat(V_obst,length(X_ostacolo(point_B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start dyn obst
```

```
start_dyn_obst];
X_obstacle_dyn = dyn_obstacle1(:,1);
Y_obstacle_dyn = dyn_obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.15. -135° 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
map = [0 0; -1000 -1000];

% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X rec, Y rec, Theta rec] = reference generator(map,V,Ts);
```

```
% Extend the reference signal to avoid index over limits
X rec(end+1:end+p+20) = X_rec(end);
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \text{ dyn} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V } 0 \text{ 0}]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eve(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point_B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start_dyn_obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.16. -160° 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
map = [0.0; -2747 -1000];
% Evaluate total distance covered by the route on the map
distance = odometer(map);
%% Reference signal
% Upsample map based on speed and timestep
[X_rec, Y_rec, Theta_rec] = reference_generator(map,V,Ts);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map, V obst, Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B];
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.17. 1000m curvature clockwise 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
[X_rec, Y_rec, Theta_rec] = curve_generator(-1000,V,Ts);
map = [X rec Y rec];
```

```
distance = odometer(map);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0_{kin} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V]';
x0_{dyn} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V 0 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V_{obst} = 10/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map, V obst, Ts);
point B = round(length(X ostacolo)*0.4);
number_obstacles = [point_B
            point_B];
dyn_obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start_dyn_obst = [X_ostacolo(point_B) Y_ostacolo(point_B) V_obst];
obstacle = [start dyn obst
       start dyn obst];
X obstacle dyn = dyn obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.18. 500m curvature clockwise 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
[X rec, Y rec, Theta rec] = curve generator(-500,V,Ts);
map = [X rec Y rec];
distance = odometer(map);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta_rec(end+1:end+p+20) = Theta_rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6:
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
             point B];
dyn obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point B:end)...
           repmat(V obst,length(X ostacolo(point B:end)),1)];
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.19. 300m curvature clockwise 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
[X_rec, Y_rec, Theta_rec] = curve_generator(-300,V,Ts);
map = [X_rec Y_rec];
distance = odometer(map);
% Extend the reference signal to avoid index over limits
X rec(end+1:end+p+20) = X rec(end);
```

```
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec(1)} Y \text{ rec(1)} Theta \text{ rec(1)} V]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0 kin';
eqoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number_obstacles = [point_B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point_B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start_dyn_obst = [X_ostacolo(point_B) Y_ostacolo(point_B) V_obst];
obstacle = [start dyn obst
       start dyn obst];
X_obstacle_dyn = dyn_obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(later-	

Enabled	Name	Definition	Requi reme nts
		al_dev > 5 && lateral_ dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.20. 300m curvature counterclockwise 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
[X rec, Y rec, Theta rec] = curve generator(300,V,Ts);
map = [X_rec Y_rec];
distance = odometer(map);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0_{dyn} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V 0 0]';
extended map = [X rec Y rec Theta rec repmat(V,length(X rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X ostacolo, Y ostacolo, theta ostacolo] = reference generator(map, V obst, Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start dyn obst
       start dyn obst];
```

```
X_obstacle_dyn = dyn_obstacle1(:,1);
Y_obstacle_dyn = dyn_obstacle1(:,2);
theta_obstacle_dyn = dyn_obstacle1(:,3);
```

Assessments

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, veri- fy(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.21. 500m curvature counterclockwise 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
[X_rec, Y_rec, Theta_rec] = curve_generator(500,V,Ts);
map = [X_rec Y_rec];
distance = odometer(map);
% Extend the reference signal to avoid index over limits
X_rec(end+1:end+p+20) = X_rec(end);
Y_rec(end+1:end+p+20) = Y_rec(end);
Theta_rec(end+1:end+p+20) = Theta_rec(end);
% Define initial condition based on map
```

```
x0_{kin} = [X_{rec}(1) Y_{rec}(1) Theta_{rec}(1) V]';
x0 \text{ dyn} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V } 0 \text{ 0}]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn_obstacle1 = [X_ostacolo(point_B:end) Y_ostacolo(point_B:end) theta_ostaco-
lo(point_B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start_dyn_obst
       start dyn obst];
X_obstacle_dyn = dyn_obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	

Enabled	Name	Definition	Requi reme nts
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	

1.2.22. 1000m curvature counterclockwise 40km/h

```
%% Set Speed
V = 40/3.6;
%% Scenario Loading
[X rec, Y rec, Theta rec] = curve generator(1000,V,Ts);
map = [X rec Y rec];
distance = odometer(map);
% Extend the reference signal to avoid index over limits
X \operatorname{rec}(\operatorname{end}+1:\operatorname{end}+\operatorname{p}+20) = X \operatorname{rec}(\operatorname{end});
Y rec(end+1:end+p+20) = Y rec(end);
Theta rec(end+1:end+p+20) = Theta rec(end);
% Define initial condition based on map
x0 \text{ kin} = [X \text{ rec}(1) \text{ Y rec}(1) \text{ Theta rec}(1) \text{ V}]';
x0 \, dyn = [X \, rec(1) \, Y \, rec(1) \, Theta \, rec(1) \, V \, 0 \, 0]';
extended_map = [X_rec Y_rec Theta_rec repmat(V,length(X_rec),1)];
egoStates.Plant = x0 kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
V obst = 10/3.6;
[X_ostacolo, Y_ostacolo, theta_ostacolo] = reference_generator(map,V_obst,Ts);
point B = round(length(X ostacolo)*0.4);
number obstacles = [point B
            point B1;
dyn obstacle1 = [X ostacolo(point B:end) Y ostacolo(point B:end) theta ostaco-
lo(point_B:end)...
          repmat(V obst,length(X ostacolo(point B:end)),1)];
start dyn obst = [X ostacolo(point B) Y ostacolo(point B) V obst];
obstacle = [start dyn obst
       start dyn obst];
X_obstacle_dyn = dyn_obstacle1(:,1);
Y obstacle dyn = dyn obstacle1(:,2);
theta obstacle dyn = dyn obstacle1(:,3);
```

Enabled	Name	Definition	Requi reme nts
True	Left lane as- sessment 1	At any point of time, At any point of time, if an obstacle is detected: verify(lateral_dev >= 2 && lateral_dev <= 6) must be true must be true	
True	Left lane as- sessment 2	At any point of time, At any point of time, verify(lateral_dev < 6) must be true must be true	
True	Safe over- take assess- ment	At any point of time, At any point of time, if an obstacle is detected: verify(duration(lateral_dev > 5 && lateral_dev < 3,sec) < 1) must be true must be true	
True	Lateral accel- eration as- sessment	At any point of time, At any point of time, verify(duration(Lateral_ acceleration >= 2,sec)<=0.5) must be true must be true	