MPC-L Static Obstacle Avoidance - Test Specification Report

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

22-Jun-2021 23:11:55

Table of Contents

1. Static_obstacle_avoidance 2

1.1. Multiple_static_obstacle_20km/h 2

1.1.1. 0° 20km/h 2

1.1.2. 20° 20km/h 3

1.1.3. 45° 20km/h 4

1.1.4. 70° 20km/h 4

1.1.5. 90° 20km/h 5

1.1.6. 110° 20km/h 6

1.1.7. 135° 20km/h 7

1.1.8. 160° 20km/h 8

1.1.9. 180° 20km/h 8

1.1.10. -20° 20km/h 9

1.1.11. -45° 20km/h 10

1.1.12. -70° 20km/h 11

1.1.13. -90° 20km/h 12

1.1.14. -110° 20km/h 12

1.1.15. -135° 20km/h 13

1.1.16. -160° 20km/h 14

1.1.17. 1000m curvature clockwise 20km/h 15

1.1.18. 500m curvature clockwise 20km/h 16

1.1.19. 300m curvature clockwise 20km/h 16

1.1.20. 300m curvature counterclockwise 20km/h 17

1.1.21. 500m curvature counterclockwise 20km/h 18

1.1.22. 1000m curvature counterclockwise 20km/h 18

1. Static_obstacle_avoidance

Test Details

Releases	Current (2019b)

1.1. Multiple_static_obstacle_20km/h

Test Details

Releases	Current (2019b)

1.1.1. 0° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
        10000 10000 0];
```

1.1.2. 20° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
        10000 10000 0];
```

1.1.3. 45° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
            10000 10000 0];
```

1.1.4. 70° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
            10000 10000 0];
```

1.1.5. 90° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
          0 10000 0];
```

1.1.6. 110° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
        -10000 10000 0];
```

1.1.7. 135° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
        -10000 10000 0];
```

1.1.8. 160° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
          -10000 10000 0];
```

1.1.9. 180° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
          -10000 0 0];
```

1.1.10. -20° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
            10000 -10000 0];
```

1.1.11. -45° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
         10000 -10000 0];
```

1.1.12. -70° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
         10000 -10000 0];
```

1.1.13. -90° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
            0 -10000 0];
```

1.1.14. -110° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
        -10000 -10000 0];
```

1.1.15. -135° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
            -10000 -10000 0];
```

1.1.16. -160° 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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_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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
          -10000 -10000 0];
```

1.1.17. 1000m curvature clockwise 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
          -1200 -800 0];
```

1.1.18. 500m curvature clockwise 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
        -1000 -400 0];
```

1.1.19. 300m curvature clockwise 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
          -600 -240 0];
```

1.1.20. 300m curvature counterclockwise 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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
idx = round(length(extended_map)*0.5);
```

```
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
          -600 240 0];
```

1.1.21. 500m curvature counterclockwise 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
%% 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)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0;
          -1000 400 0];
```

1.1.22. 1000m curvature counterclockwise 20km/h

Test Details

Releases	Current (2019b)

PreLoad Callback

```
%% Set Speed
V = 20/3.6;
```

```
%% 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)];
egoStates.Plant = x0_kin';
egoStates.Covariance = eye(6)*1000;
% Obstacle definition
idx = round(length(extended_map)*0.5);
obstacle = [extended_map(idx,1) extended_map(idx,2) 0
            -1200 800 0];
```