Mode Switching Control Using Lane Keeping Assist and Waypoints Tracking for Autonomous Driving in a City Environment

Nadjim Horri
School of Mechanical, Aerospace and Automotive Engineering
Coventry University, Coventry, CV15FB
Email: nadjim.horri@coventry.ac.uk

Olivier Haas
Institute of Future Transport and Cities
Coventry University, Coventry, CV15FB
Email: olivier.haas@coventry.ac.uk

Sheng Wang
Institute of Future Transport and Cities
Coventry University, Coventry, CV15FB
Email: sheng.wang@coventry.ac.uk

Mathias Foo
School of Engineering,
University of Warwick, Coventry, CV4 7AL, UK.
Email: M.Foo@warwick.ac.uk

Manuel Silverio Fernandez
Institute of Future Transport and Cities
Coventry University, Coventry, CV15FB
Email: Manuel.Silverio.Fernandez@coventry.ac.uk

Submitted [15/09/2021]

Abstract
This paper proposes a mode switching supervisory controller for autonomous vehicles. The supervisory controller selects the most appropriate controller based on safety constraints and on the vehicle location with respect to junctions. Autonomous steering, throttle and deceleration control inputs are used to perform variable speed lane keeping assist, standard or emergency braking and to manage junctions, including roundabouts. Adaptive model predictive control with lane keeping assist is performed on the main roads and a linear pure pursuit inspired controller is applied using waypoints at road junction where lane keeping assist sensors present a safety risk. A multi-stage rule based autonomous braking algorithm performs stop, restart and emergency braking maneuvers. The controllers are implemented in MATLAB® and Simulink™ and are demonstrated using the Automatic Driving Toolbox™ environment. Numerical simulations of autonomous driving scenarios demonstrate the efficiency of the lane keeping assist mode on roads with curvature and the ability to accurately track waypoints at cross intersections and roundabouts using a simpler pure pursuit inspired mode. The ego vehicle also autonomously stops in time at signalled intersections or to avoid collision with other road users.

Keywords: autonomous vehicle, lane keeping assist, switched, adaptive model predictive control, braking, pure pursuit.

There is an increasing demand for methods that will enable cars to drive with high levels of autonomy, while dealing with dynamical, control and safety constraints. This paper considers the mode switching control of autonomous vehicles with an emphasis on path following, under constraints on the trajectory including traffic light signals, road users and the need to maintain a safe distance from preceding vehicles. The application under consideration is the simulation of a vehicle navigating a distance of 500m in a city environment, including cross intersections, roundabouts and traffic lights before reaching a car park. Real life driving scenarios often require hybrid systems with the ability to switch between controllers when the control objectives change. Mode switching is often implemented between adaptive cruise control (ACC) laws, as in (1) where cruise, following or approach modes were successfully selected with their distance, speed and acceleration references under real traffic conditions. In (2), Ioannou and Chien developed ACC for platooning applications by implementing a switching logic in a look ahead model. Faster traffic flows were obtained compared to driver models. A similar headway switching logic was extended in (3) to vehicle following and platooning in automated highway systems, with emergency situations handling. In (4), autonomous cruise control was further developed with the ability to change lanes and vary speed limits depending on the traffic conditions. In (5), ACC was used together with a traffic dependent switching logic. In (6), model predictive control (MPC) was used for the optimal combination of flow metering and speed limits to reduce the overall congestion time. Platooning is however beyond the scope of this paper, where a time headway is only considered with respect to a maximum of one lead vehicle at any time. Other aspects of autonomy may involve the ability to avoid obstacles, either using MPC or path planning methods, such as jump point search (JPS) (7, 8). Autonomous braking is increasingly considered because of its proven impact on driving safety through well defined testing procedures (9). Autonomous braking algorithms often employ partial braking to decelerate, followed by full braking to avoid collision, as in (10) where a tradeoff was achieved between ride comfort and safety. In (11), a 39% reduction in the number of fatal injuries was obtained by combining forward collision warning with autonomous braking. This capability was extended in (12) to the ability to autonomously brake on roads with curvature. Autonomous driving strategies were also developed to account for traffic light signals. Predictive cruise control was used in (13) where upcoming traffic signal predictions were exploited to improve fuel economy and reduce trip time by adjusting speeds to reach intersections when traffic lights turn green. In (14), MPC was used to optimize fuel economy on roads with signalled intersections.

This paper focuses on path following on roads with curvature and signalled intersections with the ability to stop but also restart autonomously. Lane keeping and manoeuvring using MPC methods has been the subject of increased research interest in recent years. Predictive control is also increasingly being considered using active set or interior point methods, with hybrid affine models where a mode is typically selected based on the discrete states of a state machine and on discrete events being triggered when those states exceed thresholds (15). In (16), a Stanley lateral MPC controller was used to optimize lateral and yaw errors, while avoiding obstacles during double lane changes. In (17), MPC was performed with autonomous lane keeping using lane keeping assist (LKA) sensor points. Other approaches have focused on reducing computational demand using an explicit MPC approach where a multiparametric quadratic program is solved offline. This allows for pre-computed gains to be used in different state space regions for real time implementation as shown in (18,19). Nonlinear and time varying MPC were also used in (20) to maintain stability during active steering control on slippery roads up to relatively high speeds.

The ability to adapt MPC to unexpected emergencies or to changing paths constraints is important to many driving scenarios in city environments. The latter was shown in (21), where adaptive MPC was used to control the steering at constant speeds using a set membership-based switching strategy to deal with curvature changes during autonomous lane keeping. In (22), MPC was used with five driving modes depending on a congestion dependent emergency coefficient and on the estimated acceleration of a lead vehicle, with improvements compared to single mode MPC in terms of driving comfort and safety. In (23), adaptive MPC is used for automated fallback maneuvers in presence of sensor malfunctions. Adaptive MPC formulations with responsibility-sensitive safety distance constraints were introduced in (24). The robustness of adaptive MPC to scenario linked uncertainty, including driving at a junction, is investigated

in (25). A chance constrained scenario MPC approach was used to safely change lanes in (26). MPC was used in (27) for path planning by enforcing collision avoidance constraints while maintaining driving comfort by incorporating a lane dependent potential field within the cost function. A hybrid MPC approach was used as an adaptive cruise control method in (28) where the objective was to follow a lead vehicle. In (29), switched MPC controller was developed to switch between local linear MPC controllers depending on tyre conditions. A switched MPC controller was also proposed in (30) with mode switching between tracking error models, respectively based on heading error, heading error rate and sideslip. Mode switching with simpler controllers is less common, with few exceptions as in (31), where ACC uses nonlinear MPC for throttle control and more reactive PI control for the more urgent autonomous braking and an additional switching logic is used to automatically transition from driver commands to ACC. The switching between MPC controllers is therefore increasingly considered, but mode switching between MPC and simpler more reactive controllers has not received sufficient attention. Another reason to switch to more reactive controllers is the known possible adverse effect of certain sensors such as LKA systems as reported by Thatcham Research in (32) for scenarios requiring a lane change or to stop following lanes. In this paper, mode switching is applied between adaptive MPC and a simpler pure pursuit controller, which is known to be efficient at low speeds. In pure pursuit, a virtual target is assumed. The steering command is then either calculated from a required curvature and look ahead distance from the vehicle's rear axle to the virtual target position as in (33) or from a line of sight (LOS) requirement from the centre of gravity (CoG) of the vehicle to the virtual target as in (34). In this case, the car model still accounts for the fact that steering commands are applied with the front wheels, while heading is defined from the CoG.

The main contributions of the paper are as follows:

- a mode switching cruise and steering controller is applied with an adaptive MPC LKA mode on the main road to optimize path following and a safer and more reactive waypoints-based linear pure-pursuit type controller at road intersections, where this simpler approach is known to be efficient because of the shorter distances and lower speeds at those locations.
- the mode switching controller only uses LKA sensors on the main road where they are safe. It uses waypoints instead of LKA sensing at road junctions including cross intersections and roundabouts, where the discontinuities in lane patterns otherwise present the safety risk of an interruption in LKA sensor data, potentially leading to a wrong lane, even assuming a large LKA sensor field of view (FoV). Driving at junctions using LKA indeed presents lane keeping safety risks similar to the ones reported in (32).
- an autonomous braking logic is proposed, which differentiates between stopping accurately for traffic lights and emergency braking for other road users (ORUs) with additional safety margins, followed by a safe restart mode in both cases.
- The pure pursuit law projects the heading towards a virtual target between the current and next waypoints, which differs from conventional LOS pure pursuit where the next waypoint is tracked.

The solutions are developed using features of the Automatic Driving Toolbox in MATLAB®/Simulink 2020a, including LKA, optical and radar sensors and MPC toolbox libraries, as well as the Driving Scenario Designer. It is noteworthy that the LKA block in Simulink systematically fails at junctions and that the proposed approach circumvents this issue, which opens new prospects for users of Matlab's Automatic Driving Toolbox, particularly for path following simulation and implementation. The supervisory mode switching control, pure pursuit inspired mode and autonomous braking algorithms were all developed as part of the InnovateUK Assured Parking project (reference: 105095). One of the objectives of the project is to evaluate and develop city driving and parking capabilities for autonomous vehicles. The road model has similar features to the ASSURED CAV City at the HORIBA MIRA campus, Nuneaton, UK. A desired itinerary is realized using waypoints centered on the desired lanes with desired speeds, based on speed limits and arcs connecting waypoints on the roads that intersect.

The remainder of this paper is organized as follows: The equations of vehicle dynamics are described in the next section. The supervisory switching logic is introduced in the following section for the autonomous

driving problem under consideration. A mode switching steering controller is then presented before being extended to steering and speed control. An autonomous braking controller is introduced as an additional control mode to stop for traffic lights or ORUs before restarting. The numerical simulation analyzis is then presented, which demonstrates the effectiveness of the proposed mode switching controller with adaptive LKA MPC and pure pursuit waypoint tracking modes for path following on roads with curvature, cross intersections, roundabouts and of the autonomous braking mode using onboard obstacle sensing and traffic light information. This is followed by a discussion on the generality of the approach before concluding the paper.

Equations of Vehicle Dynamics

The lateral and longitudinal vehicle dynamics are described by a 'bicycle' type model similar to the one in (20) and given by:

$$\begin{aligned} \ddot{y} &= -\dot{x}r + \frac{F_y}{m} \\ \dot{r} &= \frac{M_z}{I_{zz}} \\ r &= \dot{\psi} \end{aligned} \qquad (1)$$

$$\ddot{x} = \dot{y}r + \frac{F_x}{m} \qquad (2)$$

where x, y are the Cartesian coordinates of the position vector in two dimensions, r is the yaw rate, I_{zz} is the yaw inertia, F_x, F_y are the forces acting along the longitudinal and lateral body axes, M_z is the yawing moment and ψ is the yaw angle. The yawing moment is a function of tyre dependent forces and arm lengths and is given by $M_z = F_{yr}\, l_r + F_{yf}\, l_f$, where l_r and l_f are respectively the rear and front tyre arm lengths with respect to the center of gravity of the car. F_{yr} and F_{yf} are the forces on the rear and front tyres, respectively and are given in (20). They depend on tyre stiffness, steering angle δ , velocity components $\dot{x}, \dot{y}$ and yaw rate r. The forces F_x and F_y are linked to the steering, acceleration and deceleration commands as follows:

$$\begin{aligned} \frac{F_x}{m} &= \frac{\delta_\tau - F_{Brake}}{m} \\ F_y &= F_{yr} + F_{yf} \end{aligned} \qquad (3)$$

where δ_τ is the throttle force acting along the longitudinal axis and F_{Brake} is the deceleration force applied by the brakes.

The model can be compactly written in vector form as:

$$\dot{\mathbf{x}} = \boldsymbol{f}(\mathbf{x}, \mathbf{u}) \qquad (4)$$

with:

$$\mathbf{x} = [x, \dot{x}, y, \dot{y}, \psi, r]^T$$

$$\mathbf{u} = \left[\frac{F_x}{m}, \delta\right]^T$$

For convenience, we define $\mathbf{p} = [x, y]^T$ as the position vector, $D = \frac{F_{brake}}{m}$ as the deceleration obtained by autonomous braking and V as the longitudinal velocity. In practice, $\frac{F_x}{m} = \frac{\delta_\tau}{m}$ during path following when the throttle is used. When autonomous braking is used, the throttle is automatically disabled and $\frac{F_x}{m} = -D$.

Supervisory Mode Switching Decision Layer for Autonomous Vehicle Control

The autonomous car in our study is required to perform the following autonomously:

(i) switch between adaptive MPC and linear pure pursuit inspired waypoint tracking using Algorithm 2 to drive on roads with junctions and only use LKA when it is safe. Note that Algorithm 1 assumes steering at a constant speed and was developed as a control design stage before being extended to Algorithm 2,

(ii) brake using Algorithm 3 – 'Autonomous braking controller' to adhere to road traffic regulations and stop in the case of emergencies due to the presence of ORUs in the path of the vehicle. The obstacle detection and pose estimation were obtained using sensor fusion from optical and radar sensors at the front of the ego vehicle.

(iii) restart when the traffic light is green or when the way is clear again, also using Algorithm 3.

The supervisory decision layer determines which of the acceleration, steering and braking controllers are activated based on three sets of activation thresholds:

(i) thresholds specifying forward crossing/collision warning time t_c before traffic lights or obstructions created by ORUs,

(ii) thresholds specifying the time gap t_h that will determine the safety distance s with respect to preceding vehicles,

(iii) lane keeping thresholds l and l_c to respectively impose vertical deviation constraints and to indicate if LKA sensing is safe to use.

A forward collision warning (FCW) status variable determines if without deceleration, collision with another road user will occur. Likewise, a forward stop line crossing warning (FSLCW) status variable determines if the traffic line will be crossed and the light is amber. The calculation of FCW time is described in the multi-stage braking section. The positions of the vertices of quadrilateral road intersection areas at the junctions are assumed to be obtained using the onboard map. Those regions will be taken to be rectangular in the numerical simulation section. The supervisory algorithm can be summarized as follows for scenarios requiring steering, acceleration and braking commands:

if (Green light) and (no road users detected within the safety distance)
    FCW status = 0 (No FCW)
    FSLCW=0 (No FSLCW)
    braking status = 0 (No braking)
    call the mode switching controller for steering/acceleration (*Algorithm 2*)
else if (amber or red traffic light) or (road user detected)
    call the autonomous braking controller (*Algorithm 3*)
end if

Note that within Algorithm 3, FCW status will be set to 1, before braking status is set to 1 (autonomous braking). The throttle of Algorithm 2 is then set to 0 during multistage braking. When the braking status is reset to 0 (green light), the brakes are released, and the throttle commands of Algorithm 2 are reactivated. A finite state machine enables the car to start again when the light turns back to green and if there is no obstacle in front of the vehicle.

Mode Switching Control for Path Following with Steering Inputs

The control design for the autonomous vehicle control is done in stages, where the lane following is first formulated for steering at constant speeds before being extended to acceleration and steering commands. The mode switching steering controller, which is presented in this section, has two modes: a LKA MPC controller on the main road and a waypoints tracking pure pursuit inspired steering controller at road intersections. Mathematically proving the conditions on mode switching stability is beyond the scope of this application driven paper. The stability of the mode switching controller was verified for a wide range of switching conditions and it is conjectured that stabilizing gains exist when continuity is ensured for the

desired speeds and positions at the intersection between the junction or roundabout area and the main road area, which are defined in algorithms 1 and 2. The next two subsections respectively describe the adaptive MPC mode and the pure pursuit inspired waypoints tracking mode in the constant speed case.

Adaptive MPC Steering Controller Mode

This subsection is a precusor to the steering and acceleration controller. It focuses on steering and assumes that the speed is constant. It allows to independently evaluate the steering performance. This preliminary simplifying assumption will however be relaxed in the proposed 3DoF controller.

A reduced order control system is therefore considered first, with $\mathbf{u} = \delta, \mathbf{x} = [x, y, \psi, r]$. The optimization problem under consideration is the minimization of a quadratic cost function under control effort and lateral error limitations:

$$\begin{gathered} \min_{\delta} J = \int_0^{\infty} \left(\mathbf{e^T Q e} + \mathbf{u^T R u}\right) dt \\ st. |\delta| < \delta_{Max} \\ \left|e_y\right| < l \end{gathered} \tag{5}$$

where $\mathbf{e} = \mathbf{x}_d - \mathbf{x}$ and $\mathbf{x}_d$ is a desired state vector trajectory, e_y is the lateral error with respect to the lane center, δ_{Max} is the maximum steering angle and l is a constraint on maximum lateral deviation from the path. The coordinates x_d, y_d are obtained from the LKA sensor when LKA is sufficiently accurate.

Using adaptive MPC, it is possible to incorporate controller and state constraints. The prediction model is adapted to changing operating conditions. The plant model is linearized by a first order Taylor expansion around operating points $(\bar{\mathbf{x}}, \bar{\mathbf{u}})$ along the road curvature. In adaptive MPC, the nominal operating point $(\bar{\mathbf{x}}, \bar{\mathbf{u}})$ evolves with time and the linearization matrices are updated:

$$\begin{gathered} \dot{\mathbf{x}} = \mathbf{A_c}(\bar{\mathbf{x}})\mathbf{x} + \mathbf{B_c}\mathbf{u} + \mathbf{N}(\bar{\mathbf{x}}) \\ \mathbf{y} = \mathbf{C_c}\mathbf{x} \end{gathered} \tag{6}$$

$$\mathbf{A_c}(\bar{\mathbf{x}}) = \left[\frac{\partial \boldsymbol{f}}{\partial \mathbf{x}}\right]_{\bar{\mathbf{x}}}, \mathbf{B_c}(\bar{\mathbf{x}}) = \left[\frac{\partial \boldsymbol{f}}{\partial \mathbf{u}}\right]_{\bar{\mathbf{x}}} = \begin{bmatrix} 0 & \frac{1}{m} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{K_f}{m} & 0 & \frac{l_f K_f}{I_{zz}} \end{bmatrix}$$

The matrix C_c is $I_{6\times 6}$ in the full 3DoF model and a concatenation of $I_{2\times 2}$ and $0_{2\times 2}$ matrix in the special case of a steering only model. After first order discretization, the discrete state and control matrices $\mathbf{A}_k$, $\mathbf{B}_k$ remain constant around the operating point and do not necessarily change at each time k. Matrices $\mathbf{A}_k$ and $\mathbf{B}_k$ are hereinafter referred to as $\mathbf{A}$ and $\mathbf{B}$.

The predicted states for i=1, N where N is the prediction horizon, can then be written as a function of past states over a prediction horizon and past control inputs over a control horizon N_c:

$$\begin{gathered} \mathbf{x}(k+i|k) = \mathbf{A}^i \mathbf{x}(k) + [\mathbf{A}^{i-1} \quad \mathbf{A}^{i-2} \quad \dots \quad \mathbf{I}]\, \mathbf{B}\, \boldsymbol{U}_c(k), i = 1, N \\ \boldsymbol{U}_c(k) = \boldsymbol{U}_c(k-1) + \Delta \boldsymbol{U}_c(k) \end{gathered} \tag{7}$$

where $\boldsymbol{U}_c(k) = \begin{bmatrix} \mathbf{u}(k\,|k) \\ \vdots \\ \Delta\mathbf{u}(k+N_c-1|k) \end{bmatrix}, \Delta \boldsymbol{U}_c(k) = \begin{bmatrix} \Delta\mathbf{u}(k\,|k) \\ \vdots \\ \Delta\mathbf{u}(k+N_c-1|k) \end{bmatrix}$

For n_u inputs, assuming no lead vehicle, the optimal vehicle path following control problem can be written as a quadratic programming problem minimizing:

$$\begin{gathered} J = \sum_{i=1}^{N} \|\mathbf{x}(k+i|k) - \mathbf{x}_{\mathrm{ref}}(k)\|_Q^2 + \sum_{j=1}^{N} \|\boldsymbol{\Delta U}(k+i|k)\|_R^2 \\ st. \left|\mathrm{u}_j\right| < u_{j_{Max}}, j = 1, n_u \\ \left|e_y\right| < l \end{gathered} \tag{8}$$

Pure Pursuit Steering Controller Mode

The conventional stabilizing steering controller at cross intersections or when entering and leaving roundabouts in the constant speed case will simply consist of a Proportional plus Integral (PI) controller:

$$\delta_{PI}(t) = kp\left(\psi_c(t) - \psi(t)\right) + k_i \int_0^t \left(\psi_c(\tau) - \psi(\tau)\right) d\tau \qquad (9)$$

where k_p and k_i are the proportional and integral controller gains respectively, ψ and ψ_c are the actual and desired yaw angles respectively. The desired yaw ψ_c is derived using a line of sight (LOS) pure pursuit inspired approach, which is recursively applied at the current position between the current waypoint *i* and the next waypoint *i*+1, for *i*=1, n_p, where n_p is the number of waypoints. Pure pursuit is also naturally complementary to MPC as it presents an analogy with it, given the tendency to look ahead with a LOS. A similar approach was proposed in (34) but the virtual target in that paper was simply the next waypoint. In this paper, the target can be adjusted depending on the relative positions with respect to the current and next waypoints. A weighting parameter λ is defined here as:

$$\lambda = \frac{\left\langle \Delta\mathbf{p}, \Delta\mathbf{p}_{\boldsymbol{waypoints(i,i+1)}} \right\rangle}{\left\| \Delta\mathbf{p}_{\boldsymbol{waypoints(i,i+1)}} \right\|^2} \qquad (10)$$

where $\langle ., . \rangle$ represents the scalar product, $\boldsymbol{\Delta p}$ is the relative position with respect to current waypoint *i* and $\boldsymbol{\Delta p_{waypoints(i,i+1)}}$ is the position vector from waypoint *i* to waypoint $i+1$. The projection parameter λ is designed to be closer to 1 when the vehicle is closer to $\mathbf{p}_{i+1}$ and closer to zero when the vehicle is closer to $\mathbf{p}_i$. This safety improvement makes the controller more reactive between waypoints. The target position can then be obtained as:

$$\mathbf{p}_T = (1-\lambda)\mathbf{p}_i + \lambda\mathbf{p}_{i+1} \qquad (11)$$

where $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$ respectively represent the current and next waypoint positions. A similar approach is used to develop realistic drivers' models by Mathworks. The desired heading is then finally given by:

$$\psi_c = atan\left(\frac{\Delta y}{\Delta x}\right) \qquad (12)$$

where Δx *and* Δy are respectively the longitudinal and lateral components of the vector $\mathbf{p}_T - \mathbf{p}$. Note that the desired heading could also have been written as a function of look ahead distance and road curvature.

The waypoints are taken to be at the center of the desired lane on the road map and on the arcs joining roads at road intersections. The desired speed is constant. This condition will later be relaxed with the variable speed controller and the pure pursuit law can be used to track variable or piecewise constant desired speeds. In the next subsection, the constant speed mode switching steering controller will combine the two controllers from the last two subsections.

Steering Controller arbitrator

The steering control objective is to follow a desired path including junctions and only change steering with a constant speed. The mode selection switches to the adaptive MPC steering controller mode on the main road before reaching road intersections, whenever the lane keeping MPC and waypoint tracking are sufficiently different (above safety threshold ε) which is typically the case with road curvature and the LKA sensing indicates a correct lane, which is characterized by a cross track error $|e_y|$ smaller than a threshold l_c. LKA is used by default whilst operating in the adaptive MPC mode because of its accuracy and the threshold ε is kept small to avoid unnecessary switching to waypoints on curved main roads. The threshold ε is only used to activate simpler PI control if the road was straight. To avoid accidental switching on the main road, it is possible to require differences smaller than ε for the last few time steps in memory, but this was found not to be necessary for our scenarios with road curvature. At intersections and when entering or leaving roundabouts, the pure pursuit inspired waypoints tracking controller mode is used for path following.

A set-based approach is used to define road intersection areas where pure pursuit PI control is used. Let F denote the full road map, in other words $F = \left\{(x, y) \in R^2, \underline{x} \leq x \leq \overline{x} \text{ and } \underline{y} \leq y \leq \overline{y}\right\}$, where $\underline{x}, \overline{x}, \underline{y}, \overline{y}$ are respectively the minima and maxima of of x and y in the road map area. Let J denote the union of all intersection areas Γ_j in the map, either between two roads or between a road and a roundabout:

$$\Gamma = \bigcup_{j=1}^{NJ} \Gamma_j \qquad (13)$$

where N_J is the number of road intersections on the desired path, $\Gamma_j = \left\{(x, y) \in R^2, \underline{x}_j \leq x \leq \overline{x}_j \text{ and} \underline{y}_j \leq y \leq \overline{y}_j\right\}$ represents a road intersection box (rectangular area in simulation examples) and $\underline{x}_j, \underline{y}_j, \overline{x}_j, \overline{y}_j$ are elements of F, respectively representing the minima and maxima of x and y in that box area.

The mode switching steering algorithm is summarized below:

Algorithm 1 - Steering Controller Arbitrator Algorithm

In Algorithm 1, steering control is applied at a constant speed to evaluate the steering performance independently from speed control and uses adaptive LKA MPC mode in $F \backslash \Gamma$ (main roads) provided that LKA is enabled by other safety and accuracy improvement thresholds. It uses the PI control mode in Γ (road intersection areas) or if LKA is disabled due to other safety and accuracy thresholds. Algorithm 3 (autonomous braking and restart) cannot be activated from Algorithm 1, as that would imply the use of variable speeds. This mode switching approach will be generalized to variable speeds in Algorithm 2, from which Algorithm 3 can then be activated.

if ($\mathbf{p} \in F \backslash \Gamma$ and $\left|e_y\right| \leq l_c$ and $|\delta_{\text{mpc}} - \delta_{\text{PI}}| > \varepsilon$)
    Apply adaptive MPC, δ_{mpc}, to solve Equation 5 (with LKA by default)
else if ($\mathbf{p} \in \Gamma$ or $\left|e_y\right| > l_c$ or $|\delta_{mpc} - \delta_{\text{PI}}| \leq \varepsilon$)
    Apply the waypoint tracking PI pure pursuit controller of Equation 9
end if

Mode Switching Controller with Steering and Acceleration Commands

The mode switching controller of this section also has an adaptive MPC LKA mode and a linear waypoint tracking pure pursuit Mode, but both control modes have two inputs, steering and acceleration. The state vector $\mathbf{x} = [x, \dot{x}, y, \dot{y}, \psi, r]^T$ and the control input vector $\mathbf{u} = [F_x, \delta]^T$ will therefore have full dimension.
The control objective of Algorithm 2 is to follow a desired path with variable desired speeds when necessary (ie. linearly decreasing desired speeds at the junctions when leaving the previous road or roundabout and linearly increasing desired speeds when entering the next road or roundabout). The ego vehicle will also maintain a safe minimum distance headway s from the lead vehicle, specified as a constant time headway t_h. We define $\Delta\delta$ and $\Delta\tau$ respectively as the differences between MPC and pure pursuit inspired PI waypoint tracking for the steering and throttle commands. The mode selector will switch to the adaptive MPC steering and acceleration controller whenever the differences in absolute value of $\Delta\delta$ and $\Delta\tau$ between the lane keepg MPC and the pure pursuit waypoints tracking control inputs exceed thresholds ε_1 on $\Delta\delta$ ε_2 on $\Delta\tau$, and that the vehicle is not at a road intersection area, such that LKA sensing is reliable. Otherwise, waypoints-based pure pursuit inspired PI control with acceleration and steering commands is used to control speed and yaw to the desired virtual target position. The fact that the pure pursuit controller is reactive is another safety advantage at junctions compared to MPC.

Adaptive MPC Steering and Acceleration Controller Mode

The adaptive MPC algorithm will be identical to the one of Equations 6-15, but with additional constraints.

$$\min_{\mathbf{u}} J = \int_0^{\infty} \left(\mathbf{e^T Q e} + \mathbf{u^T R u}\right) dt \quad st \ |\delta| < \delta_{Max} \quad \tau < \tau_{max} \quad |e_y| < l \quad |d_{EL}| > s(t_h) \tag{14}$$

where l is an upper bound constraint on the cross track error $|e_y|$, d_{EL} is the additional constraint on the distance headway from the ego to the lead vehicle and as previously defined, $\mathbf{u} = [\delta, \tau]^{\boldsymbol{T}}$ as it can be assumed that braking is not active and $F_x = \frac{\boldsymbol{\tau}}{\boldsymbol{m}}$ when this controller is applied. Autonomous braking is only used in the autonomous braking mode.

Linear Steering and Speed Controllers in the Waypoints Tracking Mode

The conventional steering control law for the 3DoF controller is still given by Equation 9 and the controller for longitudinal speed V is likewise given by:

$$\delta_{\tau\, PI} = S_{brake}\left(k_p\left(V_c(t) - V(t)\right) + k_i \int_0^t \left(V_c(\tau) - V(\tau)\right) d\tau\right) \tag{15}$$

where V_c is the desired longitudinal speed.

Note that the throttle will be multiplied by a status variable S_{brake} that will be equal to 1 nominally but will be switched to zero when a braking maneuver is activated ($F_{brake} \neq 0$) due to traffic lights turning amber or red or to prevent collision with other road users impeding the vehicle path.

The 3DoF mode switching algorithm, summarized in Algorithm 2, uses steering and acceleration commands to control speed, position and heading.

Algorithm 2 – Autonomous Steering and Acceleration Supervisory Controller

This algorithm only runs when Algorithm 3 is not running (FCW status =0, Braking status =0). For compactness, δ and τ are both denoted u_i in this algorithm, with i =1 and i =2 for the steering and throttle components of **u**, respectively. The sets Γ, F are defined as in the steering case.

if (**p** ∈ $F\backslash\Gamma$ and $|e_y| \leq l_c$ and $|u_{i\,mpc} - u_{i\,PI}| > \varepsilon_i, i = \{1,2\}$)
    Apply adaptive MPC $\mathbf{u}_{\text{mpc}}$ steering and acceleration that solves Equation 14
else if (**p** ∈ Γ or $|e_y| > l_c$ or $|u_{i\,mpc} - u_{i\,PI}| \leq \varepsilon_i, i = \{1,2\}$)
    Apply waypoint tracking PI controllers from Equations 9, 15
end

Note that in the adaptive MPC mode, the control input vector **u**(k) is applied at the current time step *k*. A loop predicts and stores inputs and outputs over the prediction horizon N and the control horizon N_c for possible future use (if MPC mode remains active) by solving Equation 14.

Autonomous Braking with Start-Stop Sequencing

The traffic light signals are programmed as a periodic finite state machine sequence. A multi-stage braking algorithm mode can be activated at any time to allow the car to stop for road users in the path of the ego vehicle, detected using optical and radar sensors, and to stop for for traffic lights when necessary. The vehicle is also programmed to restart when the traffic lights turn back to green or when there are no more

road users in the path of the ego vehicle. The methodology and algorithms for autonomous braking and start-stop sequencing are detailed hereinafter. Without loss of generality, autonomous braking is assumed to take place on straight parts of the road circuit, which is a realistic assumption near traffic lights. Autonomous braking is implemented using a finite state machine to select realistic progressive deceleration stages in the range from 0 to 1g.

Multi-Stage Braking Algorithm

The algorithm is activated when a time to collision (TTC) or time to traffic light (TTTL) becomes lower than the forward collision/intersection warning time. The FCW and a forward stop line crossing warning (FSLCW) times are given by:

$$FCW\ time = \frac{V}{D_{FCW}} \qquad (16a)$$

$$FSLCW\ time = \frac{V}{D_{FSLCW}} \qquad (16b)$$

where V is the norm of vector $\boldsymbol{V} = \dot{x}\vec{\imath} + \dot{y}\vec{\jmath}$ and $\vec{\imath}, \vec{\jmath}$ are respectively unit vectors along the x and y axes of the road map and D_{FCW} is the FCW deceleration, representing the nominal deceleration that an average driver would apply. Similarly, the FSLCW time will be needed to calculate the TTTL and this deceleration will be chosen such that $D_{FSLCW} \leq D_{FCW}$. For the simulated scenarios, traffic lights were placed at locations where the road is straight and along the x or y axis and nominal decelerations were also assumed for simplicity. Multi-stage braking uses predefined deceleration commands D_i, i=1, n, such that $D_{i+1} < D_i$ and D_n is the maximum deceleration at the last stage n. In the numerical simulation analyzis, we take n=3. Each deceleration stage has an associated braking time t_{bi}:

$$t_{bi} = \frac{V}{2\,D_i}\ , i = 1, n \qquad (17)$$

The division by 2 in Equation 17 is sometimes overlooked in commercial software but is necessary to brake at the correct position because for a constant deceleration stage to reduce the initial speed V to 0, the average of the decreasing speed over the deceleration time is $\frac{V}{2}$, assuming that the vehicle was not initially accelerating. The TTTL is given by:

$$\text{TTTL} = \frac{\|\mathbf{p} - \mathbf{p}_{TL}\|}{V} \qquad (18)$$

where $\mathbf{p}_{TL}$ is the desired CoG position of the car to stop at the traffic light stop line. Note that the TTTL simply reduces to $\frac{x - x_{TL}}{\dot{x}}$ when the traffic light is along the x axis for example. The automamous braking logic is similar for pedestrians and ORUs but the difference is that exact stoppage position is not enforced. Instead, the breaking time for ORUs is given by:

$$t_{bi\,ORU} = k\frac{V}{2\,D_i} \qquad (19)$$

where k is a gain that can be tuned empirically and depends on the speed limit and the detection sensors specifications. Geometrically, detection is possible if the ORU is within the sensor FoV and range $\frac{l_w}{\tan(FoV)} \leq d_{V-O} \leq r_{max}$, where r_{max} is the maximum sensor range, d_{V-O} is the distance to the ORU and l_w is lane width. The additional time needed to account for the fact that the vehicle is not yet aligned with a crossing pedestrian and that the braking distance is smaller than the detection range can be calculated as $\frac{r_{max} - \frac{l_w}{\tan(FoV)}}{\frac{V}{2}}$. In practice, an analytical formula helps to inform a first guess, but the value of k will be tuned manually because of the uncertainty on obstacle detection, which is characterized by the sensor detection probability and the scenario. Good repeatability was observed by applying the same k value to different roads and scenarios with the same speed limit and using the same obstacle sensors specifications.

The autonomous braking algorithm for other road users in Algorithm 3 is otherwise very similar to stopping for traffic lights, with the differences being that TTTL is replaced by the TTC and that the restart condition (light turning back to green) is replaced by a condition to check that the road user has been outside detection range for more than a set time $t_{restart}$.

The multi-stage braking algorithm is summarized in Algorithm 3 as follows:

Algorithm 3 - Autonomous Braking Algorithm

This algorithm is run:

- if (traffic light is amber or red and $\varepsilon_{SL} < d_{V-SL} < L$), where ε_{SL} is a distance ahead of the stop line that is too short to enable the ego vehicle to stop if the light changes to amber. L is the maximum distance from the stop line to start monitoring the traffic lights if the required $D \geq D_{\min}$. L will be taken to be equal to the range of the optical sensor.
- or if another road user is suddenly detected and emergency braking is required.

There is an additional 0.1s detection time delay to detect traffic lights or ORUs. That delay is implemented by default within the Automatic Driving Toolbox radar and optical sensors blocks in the case of road users and was verified with Horiba Mira to be realistic. To account for this and stop before the traffic light stop line, the car stop line is taken to be - 0.5m before the traffic light stop line. Let n be the number of braking stages.

FCW status =1 or FSLCW=1 (Collision or stop line warning, autonomous braking algorithm activated)
while i$\leq n$ (n=3 in 3 stage braking)
if (TTTL<t_{bi}) and (TTC<tbi$_{road_user}$)
$D = D_i$, (Note that $D_{i+1} > D_i$)
(use Equation 17 to compute D_i if $TTTL<t_{bi}$ and $TTC \geq t_{bi\,ORU}$,
otherwise use Equation 19)
Braking status = 1 (Deactivate throttle controller of Algorithm 2: $\delta_T = 0$)
if (car not stopped)
i=i+1
else if car stopped
FCW status = 0
FSLCW =0
Apply D_0 to keep the car stationary (assuming zero or low road inclination)
if traffic light is green or ORU out of collision area for duration $t_{restart}$
D=0 (Disable autonomous braking)
Braking status = 0 (Reactivate throttle controller of Algorithm 2)
end if (green light road user out of risk)
end if (car stop condition)
end if (TTTL or TTC condition)
end while

Note that for single stage braking (n=1) in algorithm 3, the condition to compute TTTL (or TTC) is:

$$-\frac{D}{2}\,TTTL^2 + V\,TTTL + \|\boldsymbol{p} - \boldsymbol{p_{TL}}\| = 0 \qquad (20)$$

where D is the nominal deceleration value (typically 4 m.s^{-2}). In the simulation analyzis, only longitudinal axis components of vectors **p** and **V** (with norm V) are used with a traffic light along that axis.

The autonomous braking algorithm was implemented using Stateflow and dedicated Matlab functions within Matlab/Simulink to implement the state machine part of the logic that differentiates between the cases of traffic lights and ORUs.

Numerical Simulation Analyzis

The vehicle parameters used for the simulation analyzis are mass m=1575 kg, I_{zz}=2875 mNs^2, right and left arm lengths are l_r= 1.6m and l_f=1.2m respectively. The front and rear tyre stiffness are K_f=19.6 kN/rad and K_r=33 kN/rad, respectively. Lane width is 3.75m. Initial conditions and desired speeds are scenario dependent. All scenarios are designed using MATLAB's Driving Scenario Designer tool and are preloaded with waypoints centered within the desired lanes on the desired itinerary. Functionalities of the Automatic Driving Toolbox are used for LKA and obstacle sensing. All three scenarios presented in this section have an ego vehicle with feedback control and may include other actors, such as lead cars or ORUs (pedestrians), with predefined paths.

The path following objective is to track a desired trajectory with waypoints centered within the lane. This is done using adaptive MPC when LKA is enabled on the main road (F\Γ) and using waypoints tracking PI control for cruise and pure pursuit steering control at road intersections or when entering and leaving roundabouts, where the lane pattern is typically temporarily discontinued, leading to a failure in LKA sensing. An additional time constant of 0.5 s is added between the throttle input and the vehicle. In Algorithms 1 and 2, the lateral MPC constraints are a maximum lateral deviation in absolute value l= 0.5 m, a maximum steering angle in absolute value δ_{max} =0.6 rad. A threshold l_c =1.5 m is also used as a safety feature to indicate the unlikely but risky possibility of a wrong lane. The number of waypoints is scenario dependent, with more waypoints and finer spacing at the junction areas.

Obstacle detection sensors with moderate technical specifications are only used in scenarios 2 and 3 where speed is variable to react to the sensors. The radar sensor on the vehicle has a range of 100m and a FoV of 19.85 degrees to detect lead vehicles at a relatively close range. The optical sensor has a larger FoV of 43.6 degrees and a range of 150m. Those technical specifications are within the usual range for modern obstacle sensors in autonomous vehicles (see (35)) and were obtained from Horiba-Mira to meet the requirements of Innovate UK project 'Assured Parking'. The sensor detections were derived for realistic simulated pedestrian motion, which was defined using Matlab's Driving Scenario Designer.

Scenario 1: Switched Controller with Lane Keeping MPC Steering and Waypoint Steering Modes

For this scenario, Algorithm 1 is used and simulation time t_f =54 seconds. The total number of waypoints is 37, most of which are used as a backup on the main road in case LKA sensing fails or at the roundabout. In this scenario, the set Γ is the union of two intersection areas between the roundabout and the two roads in Figure 1 and N_J=2. The MPC prediction horizon is 4 s and the control horizon is 0.2 s. The weights of the MPC controller are equal to 1 for lateral tracking and 0.1 for the control input. The desired speed is assumed to be fixed at 17 m.s^{-1} (38 mph). The gains of the Proportional plus Integral (PI) steering mode are k_p =1, k_i=0.02 in the steering scenario. The MPC constraints are a lateral deviation of +/- 0.5 m, a steering command in the range [-0.6, 0.6] rad. The threshold $\varepsilon = 0.002$ of similarity between linear waypoint tracking and LKA MPC is low, such that LKA MPC is used on the main road (F\Γ), where there are no lane pattern issues and MPC is more efficient due to the road curvature. Figure 2 shows the times when the LKA MPC controller is used (index=1) and when the pure pursuit waypoint tracking mode is used (index =0). The latter correspond to the ego vehicle entering and leaving the roundabout, from 37 s to 39 s, then from 46 s to 51 s, respectively. It can be seen in Figure 3 that the steering command remains within the steering constraints and steering angle reaches a maximum of 0.6 rad at the activation time of the waypoint tracking mode. Figure 4 shows that the lateral deviation changes sign when the curvature changes direction and remains bounded within +/-0.5 m.

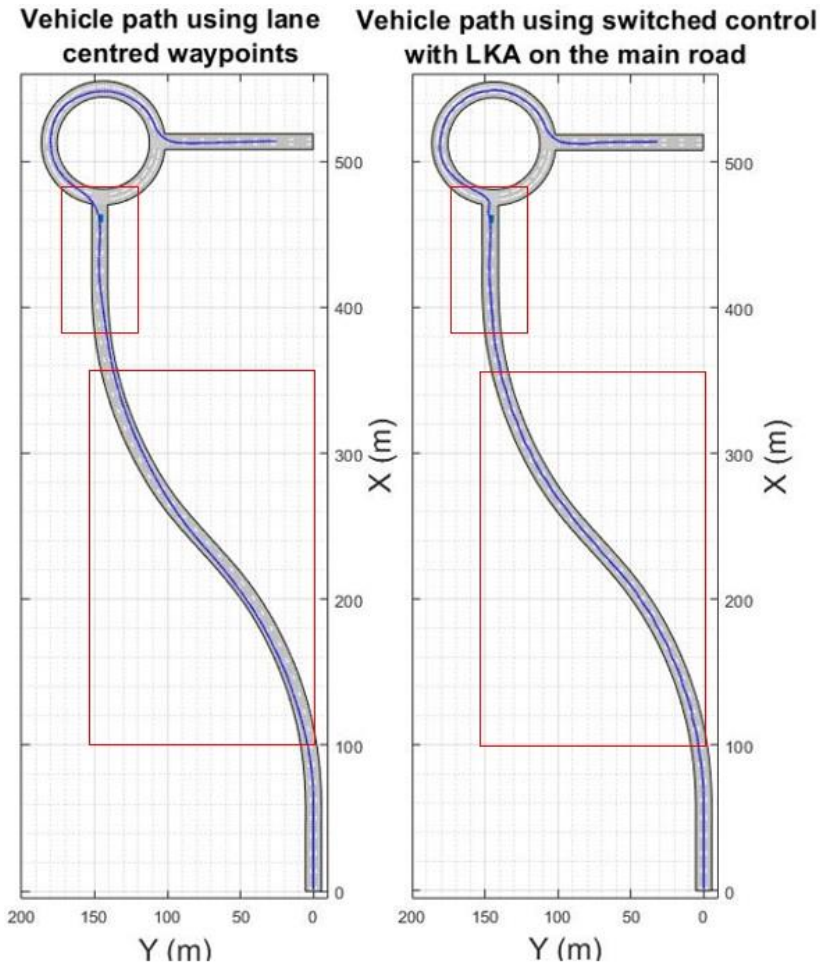

Figure 1. Comparison between Lane keeping assist (right) and waypoint tracking (left) as an adaptive MPC mode for Algorithm 1 on the main road

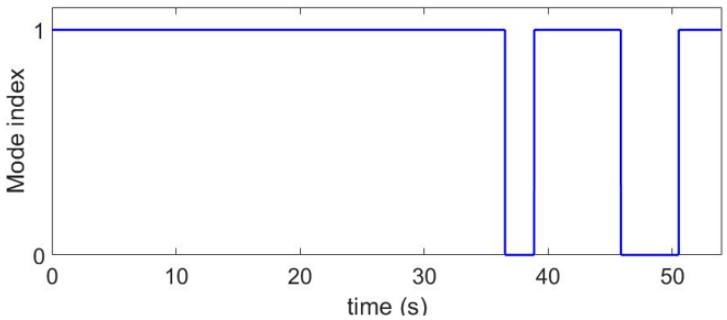

Figure 2. Mode index for steering only control, where 1 indicates that LKA MPC is activated and 0 indicates that waypoint tracking is activated.

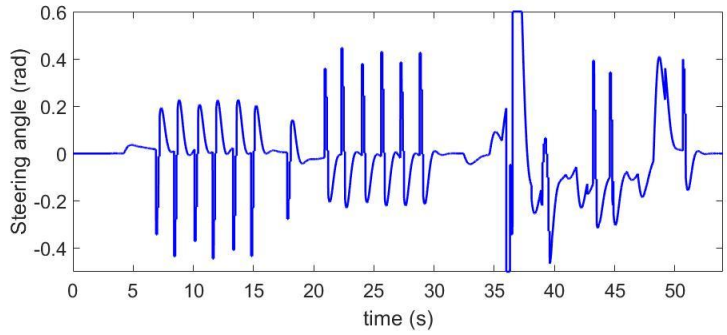

Figure 3. Steering angle input using the switched steering controller

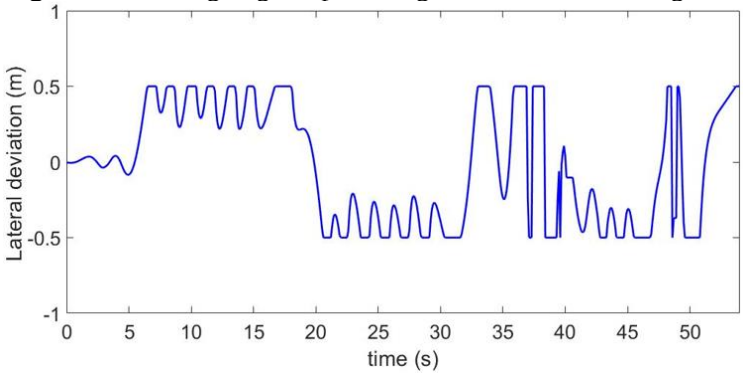

Figure 4. Lateral deviation using the switched steering controller

In Table 1, the performance metrics show that the use of the mode switching controller with adaptive MPC and linear stabilizing modes has enhanced efficiency in terms of lateral integrated absolute error (IAE) and in terms of control effort. Both controllers meet constraints but the path following performance is enhanced. The use of waypoints improves yaw error, which is of less practical significance than the lateral position error.

Performance metric	Switched Steering controller (Algorithm 1)	Switched steering controller with waypoint tracking (no LKA) in both MPC/PI modes
$\int_0^{tf} \lvert\delta(t)\rvert dt$	5.15	3.41
Yaw IAE= $\int_0^{tf} \lvert\psi_e\rvert dt$	3.2	2.95
Lateral IAE= $\int_0^{tf} \lvert y_e\rvert dt$	17.95	21.21
Min-Max lateral deviation	[-0.5, 0.6]	[-0.6, 0.6]
Min-Max yaw	[-0.5, 0.5]	[-0.5, 0.5]

Table 1. Performance metrics for the switched steering controller simulation with and without LKA on the main road

Scenario 2: Switched Controller with Steering and Acceleration Inputs in MPC and Waypoint Tracking Modes

In this subsection, Algorithm 2 is used and speed is controlled to a target speed history, which is constant at 21 m.s^{-1} on the main road (47 mph) or adjusted to keep a safe distance with respect to the lead car. Desired speeds at the cross intersection consist of a linear deceleration when approaching the junction followed by a linear acceleration out of the junction. The full simulation uses 29 waypoints in total, including waypoints on the main road in case LKA sensing failed and waypoints at the junction. The additional constraints

compared to to scenario 1 are a prescribed time headway $t_h = 1.5$ s with respect to a lead vehicle and an acceleration in the range [-3, 2] m.s^{-2}. The thresholds of similarity $\varepsilon_1, \varepsilon_2$ between steering and acceleration using waypoint tracking and LKA MPC are both equal to 0.002, such that LKA MPC is used on the main road (F\Γ), where there are no lane pattern issues and Γ is the junction area in this example where N_J =1.

The car is shown to follow the road curvature at the center of the lane and turns at the junction where the lanes are discontinued before joining the correct road and lane and reactivating lane keeping assist. The MPC weights on the longitudinal tracking, change in acceleration and steering commands are all equal to 0.1, whereas the weight on lateral error is equal to 1. The prediction horizon is 3 s, the control horizon is 0.2 s, the sampling time period is 0.1 s. The gains of the PI controllers are k_p=1, k_i=0.05 for steering and k_p= 1.1, k_i=0.1 for speed control. Figure 5 shows that the ego car follows the road curvature accurately and remains centered on the lane. In Figure 6, the mode switching controller is shown to perform better when the adaptive MPC mode uses LKA sensing compared to the use of waypoints during that mode on the main road. The same waypoint tracking pure pursuit controller is used at the junction in both cases. Note that LKA simply cannot be used for comparison at junctions using Matlab's Automatic Driving Toolbox. It was indeed found that LKA sensing systematically fails at junctions, which would also be the case in practice, because of lane discontinuities and lane recognition issues, even with a very wide LKA sensor FoV. Using waypoints at junctions is therefore a safety improvement.

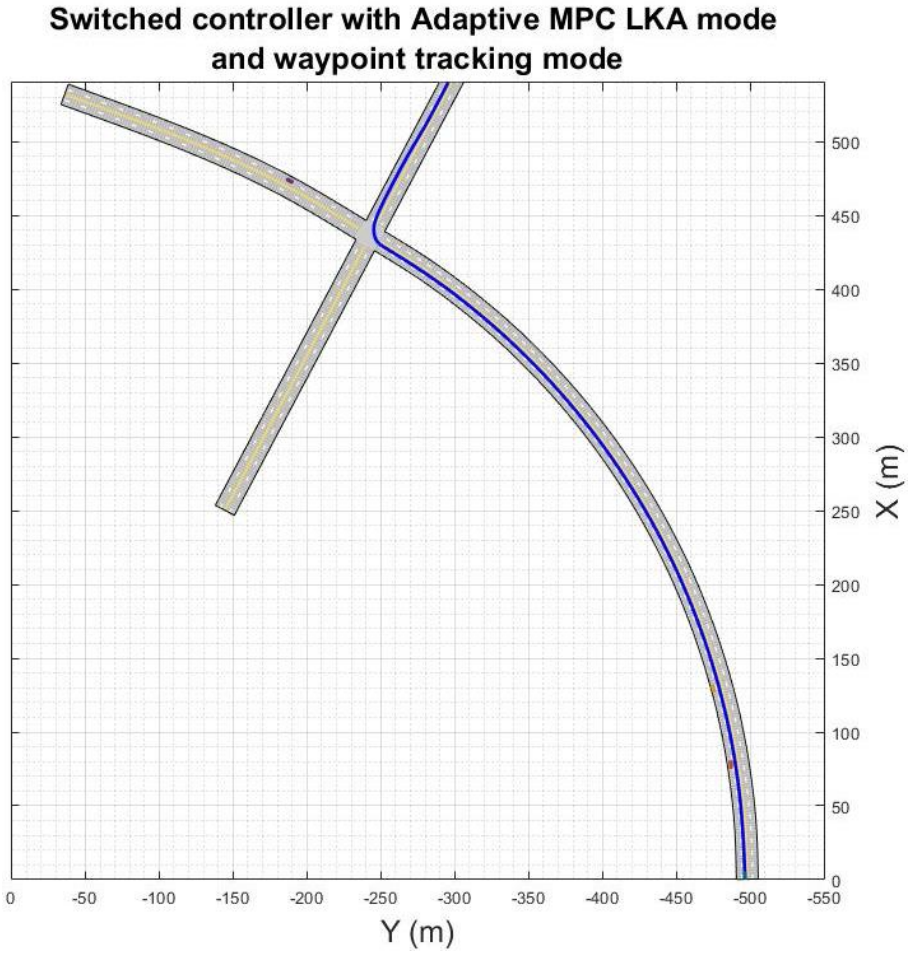

Figure 5. Path following on curved trajectory with a junction using the switched 3DoF controller

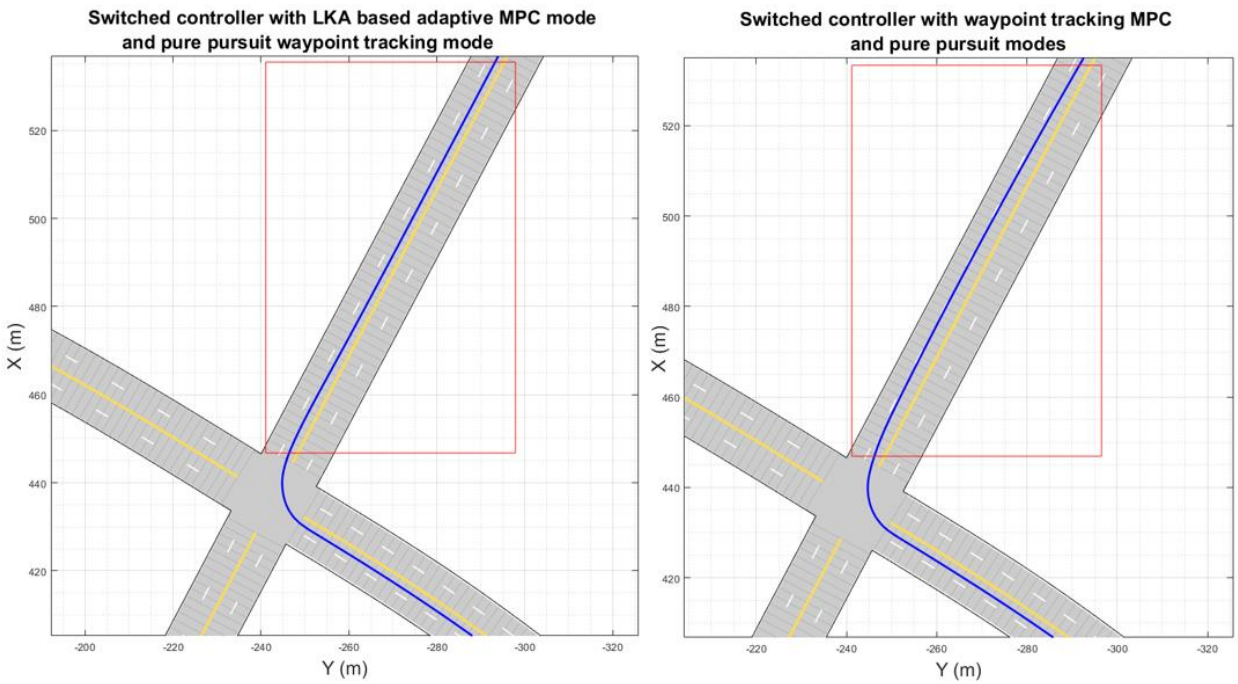

Figure 6. Comparison at the intersection between lane keeping assist (left) and waypoint tracking (right) as an adaptive MPC mode for Algorithm 2 on the main road

In Figure 7, the ego vehicle remains above the prescribed safety distance using MPC while the lead vehicle is detected. The control logic switches from mode index =1 to waypoint tracking (mode index =0) for 1 s at time t=31 s, when the ego vehicle reaches the junction, before lane keeping assist is enabled when the ego vehicle exits the junction. Figure 8 shows that the acceleration commands remain within the prescribed maximum acceleration limits of +/-2 $m.s^{-2}$. Figure 9 shows that gentle steering commands are used by the adaptive MPC controller mode of Algorithm 2 compared to the steering only controller of Algorithm 1, steering then reaches and recovers from a peak in steering but remains within prescribed steering constraints. In Figure 9, the lateral deviation remains within +/- 0.5 m.

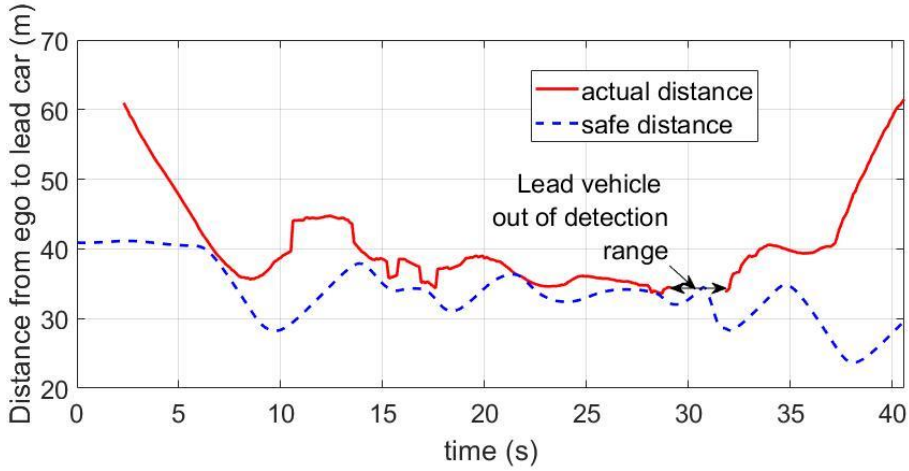

Figure 7. Distance to lead car and safety distance

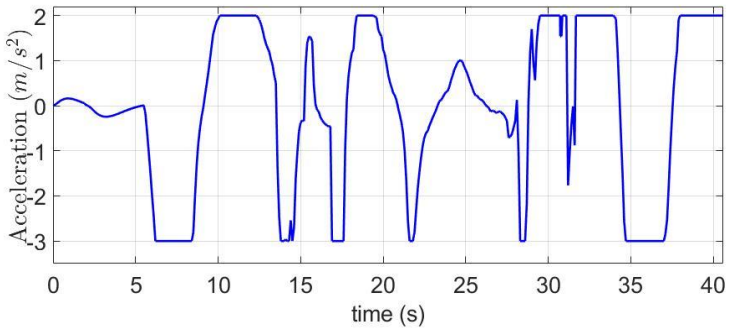

Figure 8. Acceleration commands using the 3DoF switched controller

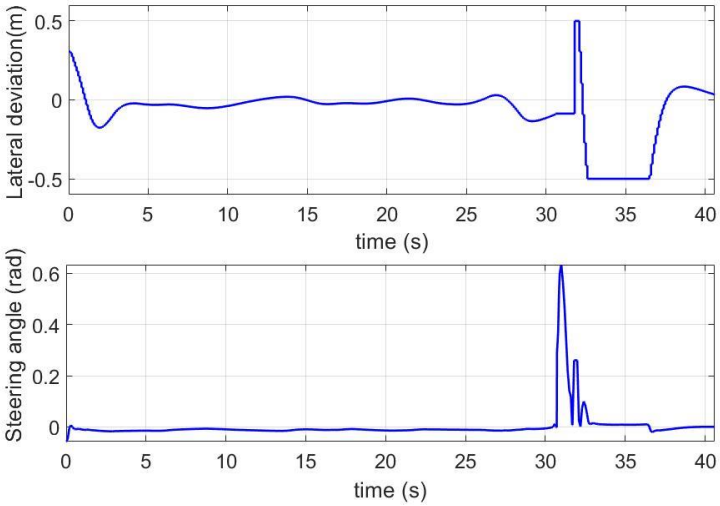

Figure 9. Lateral deviation and steering angle using the 3DoF switched controller (in radian)

All controller constraints were met during the MPC stage. A linear speed deceleration and acceleration profile was used when entering and exciting road junctions. In Table 2, good tracking performance is obtained by adding speed control to the mode switching controller. The MPC LKA stage is more accurate as predicted in terms of yaw and ateral control, but the performance of the controller remains admissible in terms of integrated control inputs and tracking error when the linear stabilizing control mode is incorporated in the full simulation.

Performance metric	Switched controller during MPC LKA stage only (t=0 to tf=27.5s)	Algorithm 2 controller during full simulation time (t=0 to tf=40s)	Switched controller with waypoints tracking in both MPC/PI modes (t=0 to t=40s)
$\int_0^{tf} \lvert\delta(t)\rvert dt$	0.37	0.78	0.92
$\int_0^{tf} \lvert\delta_\tau(t)\rvert dt$	15.9	26.25	34.55
Yaw IAE= $\int_0^{tf} \lvert\psi_e\rvert dt$	0.158	0.305	0.412
Lateral IAE= $\int_0^{tf} \lvert y\rvert dt$	1.3	2.9	3.41

Table 2. Performance metrics for the switched controllers with steering and acceleration commands

Scenario 3: Autonomous Braking for Traffic Tights and a Pedestrian

In this scenario, Algorithm 3 is used on a straight road along the Y-axis of the driving circuit. The traffic light stop line y-position coordinate is 8 m and the initial ego car y-position is at -30 m. The traffic light is located before a road intersection and a pedestrian y-position was set at a 50m in a collision course, after the road intersection. The pedestrian speed along the X-axis is 1.4 $\mathrm{m.s^{-1}}$. The number of braking stages was selected to be n=3 as it was found to be more accurate than single stage braking. The distance from the vehicle Centre of Gravity (CoG) to the front of the vehicle is 2.5m. A standard sensor detection probability of 0.9 was used for both optical and radar sensors in Matlab's driving scenario designer tool to account for the fact that pedestrian detection may not be systematic and immediate. Higher values of the detection probability were found to have a very limited effect. The multiplicative margin k in Equation 18 was taken to be equal to 1.4 to brake safely for the pedestrian in this typical scenario where the maximum speed is 12 m/s (assuming a 30 mph speed limit). This speed limit dependent gain was determined empirically and was found to work successfully on other scenarios with the same speed limit and sensor characteristics. It would be higher for a higher speed limit. The ability to restart when the traffic lights go green is also demonstrated. The maximum acceleration is conservatively set at 2m.s^{-2}. The effect of sensor update time period t_s is evaluated by numerical simulations, assuming that both the camera and radar have the same operating frequency $f_s = \frac{1}{t_s}$. Note that t_s= 0.1 s was the technical specification obtained from Horiba Mira to meet the requirements of Innovate UK project 'Assured Parking', which is within the typical 10Hz to 20Hz range of current obstacle sensing technology (see (35)) but the effect of this parameter is evaluated to assess the robustness to an unexpectedly low sensor frequency.

Initially, the traffic light is green (car running Algorithm 2 on a straight portion of the road map for simplicity). The traffic light then switches to amber, then red. The FCW time is calculated for a deceleration of $4ms^{-2}$. The traffic light is amber for 3 s before switching to red for 13 s, then amber and red for 2 s before turning back to green for 15 s, which is a typical sequence.

Figure 10 shows that the autonomous braking strategy uses multiple stages for the traffic light, but quickly selects either an intermediate deceleration of 6.8 m.s-^{2} if t_s= 0.1 s or t_s= 0.3 s or the highest level of deceleration available of $9.8ms^{-2}$ for a degraded sensor specification of t_s=0.5s. The maximum 1g deceleration level is realistic for modern vehicles equipped with anti-lock brake systems (36), but this deceletation level is only needed in emergencies such as this one.

Figure 11 shows that the vehicle stops at the correct (traffic light stop line) position when the traffic light turns red, then starts successfully again when the light turns back to green. After that, the vehicle successfully stops again safely at a CoG position 3.6 m before the crossing pedestrian and avoids a collision with the pedestrian for t_s= 0.1 s and t_s= 0.3 s. The distance from the front of the vehicle is 1.1m in both cases. The same figure however also shows that the collision risk is not avoided with t_s =0.5 s with both x and y positions differences below 0.1m . This is explained by the fact that the pedestrian is detected on time (around 23.5s) when t_s= 0.1 s or t_s= 0.3 s, but slightly too late (approximately t=23.8s) with t_s= 0.5 s. The vehicle automatically restarts when the pedestrian is no longer in a collision course.

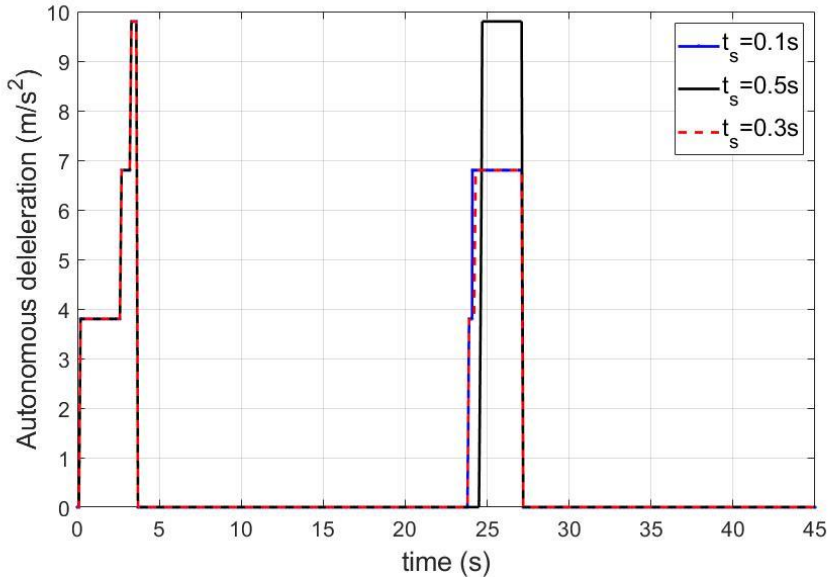

Figure 10. Deceleration during stop-start sequences for different sensor update time periods

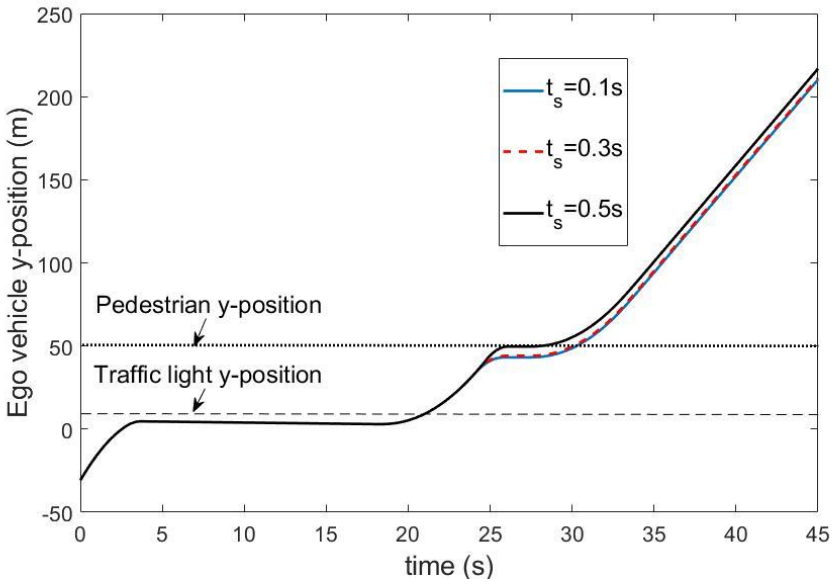

Figure 11. Sequence with autonomous braking for traffic light then road user for different sensor update time periods

Discussion on the Generality of the Proposed Approach

The proposed mode switching control approach was designed under commonly used assumptions such as zero inclination and straight roads near traffic lights, but the approach is systematic, provided that regions are defined at road junctions for the application of the pure pursuit controller. In simulation scenarios, the number of junctions N_J was either taken to be equal to 1 or 2, but the theory extends to larger road maps (larger N_J). All mode switching control algorithms were successfully tested on several road layouts with real time driving animations and scenarios involving different speed limits and junctions. To further automate the process, it is possible to store certain parameters (such as the gain k of Equation 19 in a lookup table to automatically change them when the speed limit changes. Several parameters are a function of the

vehicle and of the sensors and can be determined by numerical simulations of the proposed algorithms. Some algorithms including autonomous braking were successfully further verified using co-simulation by interfacing Matlab/Simulink with the high-fidelity simulation enviromnment of IPG Carmaker was successfully tested, but those features were not included due to paper length and to the mode switching control focus of the paper. As future work, alternatives to the pure pursuit algorithm can be considered for the waypoints tracking at junctions to further enhance performance at the expense of simplicity. The MPC mode can also be further enhanced for higher speed driving circuits or more complex junctions. Extending the switching logic to platooning is also an area for further development. The use of artificial intelligence was beyond the scope of the paper, but it could also be considered to further improve the mode switching.

Conclusions

This paper has demonstrated uninterrupted accurate autonomous path following in a city environment using threshold-based mode switching control with steering and acceleration commands. The first control mode is applied on the main roads and uses lane keeping assist based on adaptive MPC for accurate path following at desired cruise speeds. The second mode uses a safer and more reactive pure pursuit inspired controller with waypoints at junctions including cross intersections and roundabouts to maneuver at a reduced velocity at these locations where lane keeping assist sensing is less reliable. An autonomous braking mode with a stop-start logic overrides either mode to accurately stop and restart at traffic lights or to avoid collisions with other road users. Numerical simulations show that the switched controller meets the prescribed constraints on road curvature, acceleration, steering commands, time headway and lateral deviation. The proposed approach is systematic, provided that the speed limits are known and that road intersection areas are defined. This software development enhances the closed loop control demonstrations capabilities provided with the Automatic Driving Toolbox™ by including junctions and roundabouts and only employing lane keeping assist capabilities when it is safe.

Acknowledgment

The authors thank collaborators at Horiba-Mira and acknowledge that this work is developed with funding from Innovate-UK Project Park-IT (Reference: 10509).

Author Contributions

The authors confirm contribution to the paper as follows: System integration, controllers design and implementation: Nadjim Horri; Review and evaluation of the paper, control analyzis and operating conditions: Olivier Haas., Contribution to data collection and verification of simulation scenarios: Yang Sheng; Review and evaluation of the MPC approach and technical writing: Mathias Foo; Data collection from the Assured Parking project: Manuel Silverio Fernandez.

References

1. Zhenhaia, G., Juna , W., Hongyua, H. , Weia , Y., Dazhib , W. Multi-Argument Control Mode Switching Strategy for Adaptive Cruise Control System, *Procedia Engineering*, Vol. 137, 2016, pp. 581- 589.
2. Ioannou, P.A., Chien C.C. Autonomous Intelligent Cruise Control. *IEEE Transactions on Vehicle Technol*ogy, Vol. 42, No. 4, 1993, pp. 657–672.
3. Raza, H., Ioannou, P. Vehicle Following Control Design for Automated Highway Systems, *IEEE Control Systems Magazine*, Vol. 16, No. 6, 1996, pp. 43-60.
4. Zhang, Y., Ioannou, P.A. Combined Variable Speed Limit and Lane Change Control for Highway Traffic, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 18, No. 7, July 2017, pp. 1812-1823.
5. Kesting, A., Treiber, M., Schönhof, M., Helbing, D. Extending Adaptive Cruise Control to Adaptive Driving Strategies, *Transportation Research Record: Journal of the Transportation Research Board*, No. 2000, Transportation Research Board of the National Academies, Washington, D.C., 2007, pp. 16–24.

6. Hegyi, A., De Schutter, B., Hellendoorn, H. Model Predictive Control for Optimal Coordination of Ramp Metering and Variable Speed Limits, *Transportation Research Part C: Emerging Technologies*, Vol. 13, No. 3, 2005, pp. 185-209.
7. Ducho, F., Babinec, A., Kajan, M., Beno, P., Floreka, M., Tomas, F., Ladislav, J. Path Planning with Modified A Star Algorithm for a Mobile Robot, *Procedia Engineering*, Vol. 96, 2014, pp. 59 – 69.
8. Zhou, K., Yu, L, Long, Z., Mo, S. Local Path Planning of Driverless Car Navigation Based on Jump Point Search Method Under Urban Environment. *MDPI Future Internet*, Vol. 9, No. 51, 2017.
9. Hulshof, W., Knight, I.., Edwards, A., Avery, M., Grover, C. Autonomous Emergency Braking Test Results, *23rd International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, paper Number 13-0168, 2013.
10. Bae, JJ., Lee, MS., Kang, N. Partial and Full Braking Algorithm According to Time-to-Collision for Both Safety and Ride Comfort in an Autonomous Vehicle. *International Journal of Automotive Technology*, Vol. 21, 2020, pp. 351–360.
11. Kusano, K.D., Gabler, H.C. Safety Benefits of Forward Collision Warning, Brake Assist, And Autonomous Braking Systems In Rear-End Collisions, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 13, No.4, art. no. 6180219, 2012, pp. 1546-1555.
12. Lee, J., Kim, G., Kim, B. Study on the Improvement of a Collision Avoidance System for Curves. *Appl. Sciences*, Vol. *9*, No. 5380, 2019.
13. Asadi, B., Vahidi, A. Predictive Cruise Control: Utilizing Upcoming Traffic Signal Information for Improving Fuel Economy and Reducing Trip Time, *IEEE Transactions on Control Systems Technology*, Vol. 19, No. 3, art. no. 5454336, 2011, pp. 707-714.
14. Kamal, M.A.S., Mukai, M., Murata, J., Kawabe, T. Model Predictive Control of Vehicles on Urban Roads for Improved Fuel Economy, *IEEE Transactions on Control Systems Technology*, Vol. 21, No. 3, art. no. 6214590, 2013, pp. 831-841.
15. Borrelli, F., Bemporad, A., Morari, M. Predictive Control for Linear and Hybrid Systems. Cambridge University Press, 2017, pp. 16-24.
16. Abdelmoniem, A., Osama, A., Abdelaziz, M., Maged, SA. A Path-Tracking Algorithm Using Predictive Stanley Lateral Controller. *International Journal of Advanced Robotic Systems*, 2020, pp. 1-11.
17. Xu, Y., Chen, B.Y., Shan, X., Jia W.H., Lu, Z.F., Xu G. Model Predictive Control for Lane Keeping System in Autonomous Vehicle, *7th International Conference on Power Electronics Systems and Applications - Smart Mobility, Power Transfer & Security* (PESA), Hong Kong, December 2017.
18. Lee, J., Chang, H-J. Analyzis of Explicit Model Predictive Control for Path-Following Control, *PLoS One*, Vol. 13, No. 3: e0194110, 2018.
19. Kamat, S. Model Predictive Control Approaches for Lane Keeping of Vehicle, IFAC PapersOnline, Vol. 53, No. 1, 2020, pp. 176-182.
20. Falcone, P., Borrelli, F., Asgari, J., Tseng, H.E., Hrovat, D. Predictive Active Steering Control for Autonomous Vehicle Systems, *IEEE Transactions on Control Systems Technology*, Vol. 15., No. 3, 2007, pp. 566-580.
21. Bujarbaruah, M., Zhang, X., Tseng, H.E., Borrelli, F., Tseng, H.H. Adaptive MPC for Autonomous Lane Keeping, *14th International Symposium on Advanced Vehicle Control* (*AVEC*), Beijing, China, July 2018.
22. Qu, T., Zhao, J., Gao, H.; Cai, K., Chen, H., Xu, F., Multi-Mode Switching-Based Model Predictive Control Approach for Longitudinal Autonomous Driving with Acceleration Estimation, *IET Intelligent Transport Systems*, Vol. 14, No. 14, 2020, pp. 2102-2112.
23. Xue, W., Zheng, R., Yang, B., Wang, Z., Kaizuka, T. and Nakano, K. An Adaptive Model Predictive Approach for Automated Vehicle Control in Fallback Procedure Based on Virtual Vehicle Scheme. *Journal of Intelligent and Connected Vehicles*, Vol. 2, No. 2, 2019, pp. 67–77.
24. Chai, C., Zeng, X., Wu, X., Wang, X. Evaluation and Optimization of Responsibility-Sensitive Safety Models on Autonomous Car-Following Maneuvers, *Transportation Research Record: Journal of the Transportation Research Board*., Vol. 2674, No. 11, 2020, 662–673.

25. Leurent, E., Efimov, D., Maillard, O-A. Robust Estimation, Prediction and Control with Linear Dynamics and Generic Costs, *Proceedings of the 34th Conference on Neural Information Processing Systems* (NeurIPS 2020), Vancouver, Canada., 2020.
26. Cesari, G., Schildbach, G., Carvalho, A., Borrelli, F. Scenario Model Predictive Control for Lane Change Assistance and Autonomous Driving on Highways, *IEEE Intelligent Transportation Systems Magazine*, Vol. 9, No. 3, 2017, pp. 23-35.
27. Liu, C., Lee, L., Varnhagen, S., Tseng, E. Path Planning for Autonomous Vehicles using Model Predictive Control, *IEEE Intelligent Vehicles Symposium* (*IV*), Redondo Beach, CA, USA, June 2017.
28. Corona, D., Lazar, M., De Schutter, B., Heemels, M. A Hybrid MPC Approach to the Design of a Smart Adaptive Cruise Controller, *Proceedings of the 2006 IEEE International Conference on Control Applications*, Munich, Germany, October 4-6, 2006.
29. Di Cairano, S., Tseng, H.E., Bernardini, D., Bemporad, A. Steering Vehicle Control by Switched Model Predictive Control, *IFAC Proceedings*, Vol. 43, No. 7, 2010, pp. 1-6.
30. Sun, C., Zhang, X., Zhou, Q., Tian, Y. A Model Predictive Controller with Switched Tracking Error for Autonomous Vehicle Path Tracking, *IEEE Access*, Vol. 7, 2019, pp. 53103–53114.
31. Shakouri, P., Ordys, A. Nonlinear Model Predictive Control approach in design of Adaptive Cruise Control with Automated Switching to Cruise Control, *Control Engineering Practice*, Vol. 26, No. 1, 2014, pp. 160-177.
32. Grover, C., Avery, M., Knight, I. Technologies for the Prevention of Run Off Road and Low Overlap Head-On Collisions, *Proceedings of the 24th International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, Gothenburgh, Sweden, 2015.
33. Yang, X.; Xiong, L.; Leng, B.; Zeng, D.; Zhuo, G. Design, Validation and Comparison of Path Following Controllers for Autonomous Vehicles. *MDPI Sensors*, Vol. 20, No. 6052, 2020.
34. Chen, Y., Zhu, J., Pure Pursuit Guidance for Car-Like Ground Vehicle Trajectory Tracking, *Proceedings of 2017 the ASME Dynamic Systems and Control Conference*, Tyson Coner, Virginia, USA, October 11-13, 2017.
35. Müller, F. Survey on Ranging Sensors and Cooperative Techniques for Relative Positioning of Vehicles, *Sensors*, Vol. 17, No. 2: 271, 2017.
36. Kudarauskas, N. Analyzis of Emergency Braking of a Vehicle, Transport, Vol. 22, No. 3, 2007, pp. 154-159.