Towards trajectory planning and collision avoidance for the voyager unmanned ground vehicle in dynamic environments

Dean Jeggels[1*], and *Jacobus A. A.* Engelbrecht[1]

[1]Department of Electrical and Electronic Engineering, Stellenbosch University, South Africa

Abstract. This paper presents a Model Predictive Contouring Control (MPCC) framework for trajectory planning on the CSIR Voyager differential-drive UGV, scoped to structured racetrack environments. The system unifies path following and short-horizon planning using a linear-spline track representation and curvilinear progress dynamics, targeting high progress with low contouring and lag error. Dynamic obstacles are assumed to have known, broadcast trajectories; collision avoidance is formulated either by penalizing vehicle-to-obstacle distance in the objective function or by enforcing minimum-distance inequality constraints over the prediction horizon. The framework is validated in MATLAB simulations. Real-time feasibility depends strongly on horizon selection: smaller prediction/control horizons achieve >10 Hz closed-loop rates, whereas longer horizons that improve lookahead can exceed the 10 Hz budget. The work establishes a baseline in simualtion for integrating dynamic-obstacle avoidance under known trajectories onto the Voyager platform; future efforts will progress toward onboard perception-driven obstacle prediction and constraints before deployment on the physical Voyager platform.

1. Introduction

Autonomous navigation in dynamic environments represents a fundamental challenge in robotics and artificial intelligence. As unmanned ground vehicles (UGVs) transition from controlled industrial settings to complex real-world applications like urban delivery and disaster response, their ability to balance high-speed operation with collision avoidance becomes critical to both operational success and public safety. An increase in demand for automated ground vehicles that can work safely alongside humans has led to the need for solving complex problems like real-time trajectory optimisation in uncertain dynamic environments. Prior work in this domain has looked at various strategies, from traditional model predictive control (MPC) frameworks [1] to learning-based approaches [2].

* Corresponding author: 23676787@sun.ac.za

While optimisation-based methods, as presented by Keanly et al. [3], showed effective static obstacle avoidance, they fail to account for dynamic obstacles. Recent advances in model predictive contouring control (MPCC) have shown promise for racing applications, but implementations typically assume static environments or rely on reactive rather than While optimisation-based methods, as presented by Keanly et al. [3], showed effective static obstacle avoidance, they fail to account for dynamic obstacles. Recent advances in model predictive contouring control (MPCC) have shown promise for racing applications, but implementations typically assume static environments or rely on reactive rather than predictive avoidance strategies [4]. Comparative studies between Pure Pursuit, MPC and MPCC by Kong et al. [5] reveal trade-offs between computational complexity and path optimality, highlighting the need for frameworks that integrate dynamic obstacle prediction directly into the optimisation process.

This research works towards extending their methodology to dynamic environments by incorporating predictions of moving obstacles into a real-time MPCC framework by Lam et al. [6] on the Voyager. The key contribution lies in formulating a joint optimisation problem that accounts for both the Voyager's dynamics and the anticipated trajectories of dynamic obstacles, enabling proactive collision avoidance.

2. Related work

Recent advances in trajectory planning balance computational efficiency and dynamic handling. MPC enables constraint-aware trajectory optimization but often prioritizes reference tracking over progress maximization. Sampling-based methods like RRTs offer rapid feasible path generation but struggle with real-time execution [7]. Optimization-based approaches, including Heilmeier et al.'s [7] quadratic programming (QP) for minimum-curvature trajectories, enable aggressive cornering through curvature minimization and track boundary constraints. Liniger et al. [8] demonstrated receding horizon controllers achieving near time-optimal performance, while Kong et al. [5] highlighted MPCC's superior lap times despite computational overhead compared to Pure Pursuit.

Liniger et al. [8] introduced convex "slab" constraints for systematic obstacle avoidance, extended via dynamic programming for non-convex scenarios like overtaking. However, such methods primarily address static obstacles or assume perfect prediction. Recent work by Borrello et al. [9] integrates trajectory optimization with linearized separating hyper-plane constraints, allowing MPC programs to sidestep both static and moving objects while maintaining real-time performance. Heilmeier's occupancy grid enables reactive static obstacle avoidance but lacks dynamic integration [7]. Current limitations include fragmented prediction-planning pipelines, where trajectory generation and obstacle forecasting remain decoupled, leading to suboptimal solutions in high-speed interactions.

MPCC bridges path planning and tracking by optimizing progress-contouring trade-offs. Lam et al. [6] validated its industrial applicability through actuator-constrained contouring systems, while Liniger et al. [8] achieved 50 Hz real-time execution on embedded platforms for RC car racing. Kong et al. [5] noted MPCC's lap time advantages over MPC and Pure Pursuit but highlighted sensitivity to sensor noise and computational delays. Brito et al. [10] demonstrated that embedding convex ellipsoidal obstacle constraints directly within MPCC formulations enables smooth avoidance maneuvers while preserving the quadratic structure necessary for fast optimization.

Differential drive UGVs face unique challenges from nonholonomic constraints and terrain variability. The F1Tenth framework, utilizing bicycle kinematic models, enables high-speed testing with Liniger et al. demonstrating 3 m/s drifting under saturated tire forces [8]. While MPCC shows promise for racing, existing implementations neglect dynamic obstacle

prediction. Contemporary research reveals a significant knowledge gap for differential drive systems, where motion characteristics fundamentally differ from Ackermann steering platforms [11].

Current obstacle avoidance approaches typically treat moving objects as static at each control instant, failing to leverage predictive capabilities inherent in optimization-based methods [12]. This reactive paradigm becomes particularly limited in structured racing environments where geometric regularity offers opportunities for enhanced trajectory planning that remain underexplored in dynamic scenarios [13]. This work addresses these gaps by integrating obstacle trajectory forecasts into MPCC's optimization framework for proactive collision avoidance on differential drive platforms.

3. System overview and methodology

3.1 Voyager UGV platform

The Voyager UGV in Fig. 1 is a differential-drive robot designed by the Council for Scientific and Industrial Research (CSIR) [14] for high-performance autonomous navigation research. The platform's hardware specifications can be seen in Table 1.

Fig. 1. Voyager platform.

Table 1. Voyager's hardware specifications [14].

Hardware	**Specification**
Drive Mechanism	Differential drive with castors
Battery	24V 24Ah Lithium Phosphate
LiDAR Sensor	Ouster OS0-32
IMU Sensor	Xsens MTi-630 AHRS
Camera	Logitech C920 HD Pro Webcam
Operating System	Ubuntu 22.04 LTS
Robot Operating System Version	ROS2 Humble
Software speed	Limit of 1.2 m/s implemented for safety. Theoretical maximum speed of 2.7 m/s.
CPU	Intel Core i9 Processor

The vehicle's kinematics can be represented by a differential drive model characterised by the following state-space representation and the frame in Fig. 2:

$$\boldsymbol{\xi} = \begin{bmatrix} x \\ y \\ \psi \\ s \end{bmatrix}, \boldsymbol{u} = \begin{bmatrix} v \\ \boldsymbol{\omega} \\ v_s \end{bmatrix} \tag{1}$$

$$\boldsymbol{A} = \begin{bmatrix} 0 & 0 & -v\sin(\psi) & 0 \\ 0 & 0 & v\cos(\psi) & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} \cos(\psi) & 0 & 0 \\ \sin(\psi) & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

The state vector $\boldsymbol{\xi}$, represents the vehicle's position (x, y), heading angle ψ, and the progress s of a desired position (x_d, y_d) along the reference path s. The control inputs **u** represents the linear velocity v, angular velocity ω, and rate of progress v_s of the desired position along the reference path.

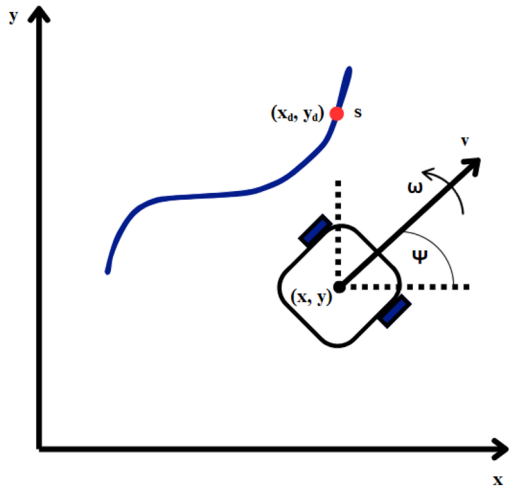

Fig. 2. Differential drive kinematic model diagram.

This state-space representation enables the prediction of future vehicle states based on current states and control inputs, forming the basis for the predictive component of the MPCC formulation.

3.2 High-level overview

The proposed trajectory planning and collision avoidance system for the Voyager UGV uses a modular architecture that integrates sensing, perception, prediction, and control components in a cohesive framework. Fig. 3 shows the system's overall architecture, highlighting the information flow between components.

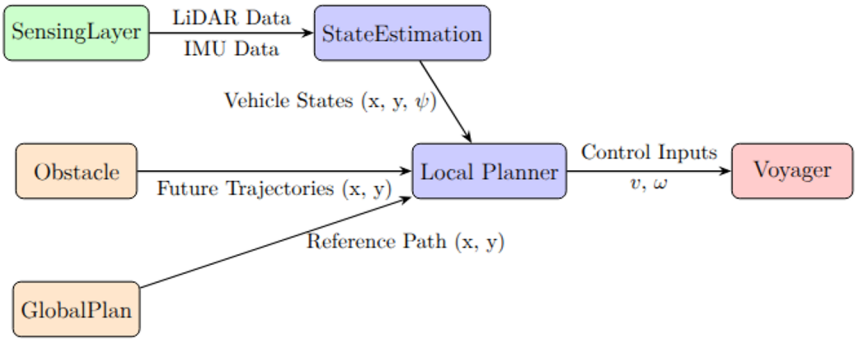

Fig. 3. System architecture diagram showing data flow from sensors through processing modules to MPCC controller.

The system operates within a structured racing environment where Voyager shares the circuit with an independently driven obstacle vehicle that follows the same reference trajectory but with predetermined lateral and longitudinal offsets. The obstacle vehicle operates without collision avoidance capabilities and maintains a maximum speed below Voyager's minimum operational speed, forcing active overtaking maneuvers rather than passive speed reduction strategies. This experimental design creates repeatable, challenging scenarios that test the spatial planning capabilities of the MPCC algorithm.

State estimation leverages RTAB-MAP's localization mode, which provides robust pose estimation by fusing data from the onboard 3D LiDAR, IMU, and wheel encoders. RTAB-MAP employs an incremental appearance-based loop closure detector with bag-of-words methodology to estimate the vehicle's position within pre-existing maps generated through offline SLAM processes. The system receives reference trajectories as parameterized linear splines from the global planning module, which extracts smooth centerlines from SLAM-generated occupancy grids through a systematic four-stage process adapted from medical imaging techniques.

Rather than implementing complex sensor-based obstacle detection algorithms, the system assumes knowledge of obstacle trajectories through ROS2 network communication. The obstacle vehicle is represented geometrically as a circular entity with defined radius and centre point, simplifying collision detection calculations while maintaining realistic spatial requirements. The collision risk assessment evaluates potential conflicts using geometric distance calculations combined with predictive time-to-collision analysis, computing Euclidean distances between vehicle and obstacle centres while accounting for their circular representations.

The risk assessment system employs graduated risk metrics where collision avoidance priority increases dynamically as vehicle-to-obstacle distance decreases below predetermined safety thresholds. Safety margin computation establishes minimum acceptable separation distances that account for vehicle dynamics, sensor uncertainties, and control system response times. This graduated approach ensures smooth transitions between normal trajectory following and emergency avoidance behaviours, providing constraint activation signals to the MPCC optimization framework.

The MPCC module serves as the central decision-maker, integrating vehicle states, obstacle predictions, and reference trajectory information to generate optimal control commands. The framework simultaneously optimizes trajectory tracking accuracy and progress along the desired reference path through a unified optimization problem formulated in a curvilinear coordinate system adapted for differential drive kinematics.

Dynamic obstacle avoidance is integrated into the MPCC framework through either cost function modifications or constraint formulations. The cost function approach introduces vehicle-to-obstacle distance terms with Gaussian-shaped weighting functions that dynamically prioritize collision avoidance as obstacles approach, providing smooth convergence behaviour while maintaining robust avoidance performance. Alternatively, constraint-based implementation treats obstacle avoidance as inequality constraints, maintaining the quadratic structure of the optimization problem while providing hard guarantees for collision-free navigation.

The system ensures real-time performance through careful solver selection and parameter tuning, with prediction horizon lengths and discretization intervals balanced to meet the required 10 Hz control loop frequency. Control commands comprising linear and angular velocities are extracted from the optimal control sequence and transmitted to Voyager's low-level controllers via standardized ROS2 interfaces, maintaining consistency with the physical platform's communication architecture.

3.3 MPCC formulation

The formulation adapts the approach presented by Lam et al., modified to work with the Voyager's kinematic model.

The MPCC formulation consists of three main components. The state equation that predicts future vehicle states based on current states and control inputs.

$$\boldsymbol{\xi_{k+1}} = f(\boldsymbol{\xi_k}, \boldsymbol{u_k}) = \begin{bmatrix} x_k + v_k cos(\psi_k)\Delta t \\ y_k + v_k sin(\psi_k)\Delta t \\ \psi_k + \omega_k \Delta t \\ s_k + v_{s,k}\Delta t \end{bmatrix} \tag{3}$$

The cost function to be minimised, balancing path tracking by minimising contouring error, minimising lag error, and maximising progress along the reference path, is defined as

$$J = \sum_{i=0}^{N_p} \left[e_c^T Q_c e_c + e_l^T Q_l e_l - q_0 s_i + u_i^T R u_i \right] \tag{4}$$

where e_c is the contouring error, which represents the lateral deviation from the reference path calculated by:

$$e_c = \sin(\theta_p) \cdot (x - x_d) - \cos(\theta_p) \cdot (y - y_d) \tag{5}$$

The lag error e_l, which represents the longitudinal deviation along the reference path is calculated by:

$$e_l = -\cos(\theta_p) \cdot (x - x_d) - \sin(\theta_p) \cdot (y - y_d) \tag{6}$$

where (x, y) is the current vehicle's position, and (x_d, y_d) is the desired vehicle position calculated as a function of the progress variable *s* using linear spline interpolation. The path angle is represented by

$$\theta_p = \arctan 2 \left(\frac{dy}{ds}, \frac{dx}{ds} \right) \tag{7}$$

where $\frac{dx}{ds}$ and $\frac{dy}{ds}$ are the derivatives of the path position coordinates with respect to the progress variable. The reference trajectory is represented using linear spline interpolation rather than higher-order approaches such as quintic spline interpolation since racing tracks are typically smooth and well-defined providing sufficient accuracy. This design choice was made to require less computational effort for faster real-time execution due to simplified calculations of path derivatives and properties. The linear spline interpolation represents the path as a sequence of connected line segments, with the path parameter *s* proportional to the distance along the path. The reference path is represented as a piecewise linear spline defined by N trajectory points $(x_k,\ y_k)_{k=0}^{N-1}$. The spline is parameterised by progress variable $s \in [0, s_{max}]$, where s_{max} is the total path length. The cumulative progress s_k at point k, position interpolation for any point in $s \in [s, s_{i+1}]$, and path derivatives is computed as:

$$s_k = \begin{cases} 0, & k < 0 \\ s_{k-1} + \sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2}, & k \geq 1 \end{cases} \tag{8}$$

$$x(s) = x_i + \frac{s - s_i}{s_{i+1} - s_i}(x_{i+1} - x_i) \tag{9}$$

$$y(s) = y_i + \frac{s-s_i}{s_{i+1}-s_i}(y_{i+1} - y_i) \quad (10)$$

$$\frac{dx}{ds} = \frac{x_{i+1}-x_i}{s_{i+1}-s_i} \quad (11)$$

$$\frac{dy}{ds} = \frac{y_{i+1}-y_i}{s_{i+1}-s_i} \quad (12)$$

The formulation includes constraints on the control inputs to ensure they remain within feasible limits, with $0 \leq v \leq v_{max}$, $-\omega_{max} \leq \omega \leq \omega_{max}$, $0 \leq v_s \leq v_{s,max}$.

Dynamic obstacle avoidance is integrated into the MPCC framework through two alternative formulations: a cost function approach and a constraint-based approach. Both methods represent the vehicle and obstacles as circles to simplify collision detection while maintaining computational efficiency.

The vehicle-to-obstacle distance is calculated as

$$D_{V2O} = \sqrt{(X - X_{obs})^2 + (Y - Y_{obs})^2} - r_{obs} - r_{veh} \quad (13)$$

where (X, Y) represents the vehicle's position, (X_{obs}, Y_{obs}) is the obstacle's center position, and r_{obs} and r_{veh} are the obstacle and vehicle radii respectively.

The first approach incorporates obstacle avoidance directly into the cost function by adding a vehicle-to-obstacle distance term. The modified cost function becomes:

$$J = \sum_{i=0}^{N_p} \left[e_c^T Q_c e_c + e_l^T Q_l e_l - q_0 s_i + u_i^T R u_i + \sum_{j=1}^{N_{obs}} q_{V2O} e_{V2O,j,i}^2 \right] \quad (14)$$

The obstacle avoidance error is defined as:

$$e_{V2O} = D_{V2O} - D_{sft,o} \quad (15)$$

The dynamic weighting factor q_{V2O} uses a gaussian-shaped prioritization function that increases collision avoidance priority as the vehicle approaches the obstacles.

$$q_{V2O} = \begin{cases} P_k & if D_{V2O} < 0, \\ P_k e^{-\frac{2D_{V2O}^2}{D_{sft,O}^2}}, & elseif \ \ 0 < D_{V2O} < D_{sft,O} \\ 0, & otherwise \end{cases} \quad (16)$$

where P_k represents the maximum penalty weight and $D_{sft,O}$ is the safety distance threshold. This formulation provides smooth transitions between normal trajectory following and emergency collision avoidance behaviors.

The alternative approach treats obstacle avoidance as hard inequality constraints, ensuring guaranteed collision-free navigation.

$$D_{V2O} > D_{sft,O} \quad (16)$$

This constraint is applied for each obstacle j at each prediction step i over the horizon N. The constraint-based formulation maintains the quadratic structure of the optimization problem while providing absolute guarantees for collision avoidance, though it may result in more conservative behavior compared to the cost function approach.

For both formulations, obstacle positions (X_{obs}, Y_{obs}) at future prediction steps are computed using the known obstacle trajectories received through ROS2 communication. The obstacle vehicle follows the same reference trajectory as Voyager but with predetermined lateral and longitudinal offsets, enabling accurate prediction of future positions over the MPCC horizon. The selection between the cost function and constraint-based approaches depends on the desired balance between performance optimality and safety guarantees, with the constraint approach providing harder safety margins at the potential cost of solution feasibility in tight scenarios.

4. Experimental setup

4.1 Simulation environment

The MPCC framework used in this work was developed and validated in MATLAB. The MATLAB implementation enables comprehensive algorithm validation and parameter tuning into a controlled environment before future integration into the physical Voyager.

The racetrack centreline, for the Austrian F1 circuit, was generated following the methodology described by Betz et al. [15], which is widely adopted in autonomous racing research. The centreline data is used to construct a reference trajectory via linear spline interpolation, parameterised by a progress variable s. Linear splines were selected over quintic splines for this application due to their computational efficiency and because the reference points are close enough to represent a smooth trajectory with linear splines.

To evaluate the validity of the linear spline interpolation, the progress variable s was incremented at three different resolutions: 10, 5, and 0.1 units along the centreline in blue. Each resulting path, in red, can be seen below, and the differences in path smoothness and fidelity were visualised.

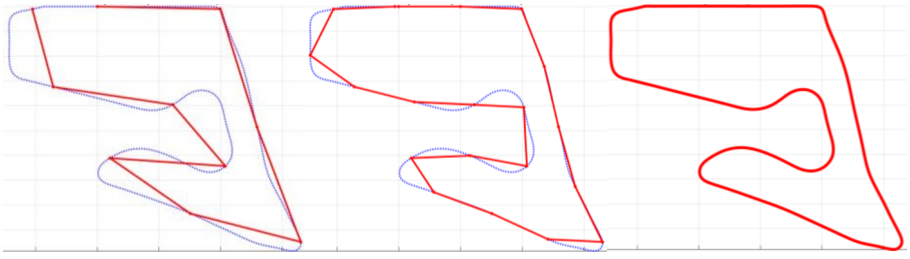

Fig. 4 (a). Spline at resolution of 10 units.

Fig. 4 (b). Spline at resolution of 5 units.

Fig. 4 (c). Spline at resolution of 0.1 units.

4.2 MPCC controller validation

The MPCC controller was developed and validated in MATLAB to enable comprehensive algorithm testing and parameter optimization. The simulation environment models the Voyagers differential drive kinematics and incorporates realistic sensor noise characteristics via

The controller publishes differential drive commands consisting of linear and angular velocities. These commands are constrained with maximum linear velocity limited to 0.2 m/s and maximum angular velocity constrained to 90 degrees/second. The velocity constraints are subsequently enforced at lower control levels through internal motor torque command limitations, ensuring adherence to the platform's mechanical and safety specifications.

Although the Voyager can handle speeds of up to 1.2 m/s, the reduced speed constraints were implemented for several engineering and experimental reasons. The maximum linear velocity of 0.2 m/s enables precise trajectory tracking evaluation without introducing excessive dynamic effects that could obscure the fundamental obstacle avoidance performance. This conservative speed selection ensures that the MPCC algorithm operates within stable convergence regions where solver performance remains consistent and predictable. The minimum operational speed was set to 0.1 m/s to maintain sufficient vehicle dynamics for meaningful differential drive control while avoiding numerical issues associated with near-zero velocities in the kinematic model.

The obstacle vehicle operates at a fixed speed of 0.05 m/s, which is deliberately slower than Voyager's minimum operational velocity. This speed differential necessitates active overtaking maneuvers rather than passive speed-reduction strategies, creating controlled scenarios that specifically test the spatial planning capabilities of the MPCC algorithm under dynamic conditions. This experimental design ensures repeatable, challenging test cases while maintaining computational tractability for real-time simulation execution.

The MPCC cost function weights were selected through systematic engineering analysis combined with empirical tuning to achieve optimal balance between competing control objectives. The weight selection process incorporated both theoretical considerations from MPCC literature and experimental validation within the simulation environment.

Increasing the contouring error weight (q_c) forces tighter path following but reduces flexibility for obstacle avoidance manoeuvres. Decreasing it allows greater lateral deviation, enabling smoother overtaking but potentially compromising path fidelity and increasing lap times.

Increasing the lag error weight (q_l) enforces stricter longitudinal tracking and speed regulation, while decreasing it allows more aggressive speed variations that may improve obstacle avoidance responsiveness but reduce trajectory tracking.

Higher values for the control input weights (r_v, r_ω, r_{vs}) promotes smoother, more conservative control actions, but potentially limiting dynamic manoeuvring capabilities. Lower weights promote more aggressive control but may introduce oscillatory behaviour.

Higher values for the progress weight (q_θ) encourages faster progression along the reference trajectory, improving lap times but potentially compromising safety margins, where lower values prioritize safety and tracking accuracy over speed.

Lastly, increasing the obstacle avoidance penalty (p_k) creates stronger repulsive forces on obstacles but potentially causes overly conservative manoeuvres.

Keeping the above in mind the following weights were selected through systematic engineering analysis and reasoning

$$\boldsymbol{Q} = \begin{bmatrix} q_c & 0 \\ 0 & q_l \end{bmatrix} = \begin{bmatrix} 50 & 0 \\ 0 & 100 \end{bmatrix} \tag{18}$$

$$\mathbf{R} = \begin{bmatrix} r_v & 0 & 0 \\ 0 & r_\omega & 0 \\ 0 & 0 & r_{vs} \end{bmatrix} = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} \tag{19}$$

$$q_\theta = 30 \tag{20}$$

$$p_k = 100 \tag{21}$$

The higher lag error weight relative to the contouring weight prioritizes longitudinal control stability while allowing sufficient lateral flexibility for obstacle avoidance. This 2:1

ration allows speed regulation during overtaking while maintaining path-following capability.

The low control input weights enable responsive dynamic control actions necessary for effective obstacle avoidance without introducing excessive control effort penalties.

The moderate progress weight strikes an optimal balance between forward progression and maintaining safety priorities. This value ensures good lap time performance while allowing collision avoidance and tracking priorities to override when needed.

The high obstacle avoidance penalty provides strong repulsive forces for collision avoidance while maintain solver convergence. This weight allows reliable obstacle avoidance without causing overly conservative manoeuvers that would impact lap times.

4.3 Simulation experiments

The obstacle avoidance experiments were conducted to evaluate the performance differences between the two MPCC formulations: cost function-based and inequality constraint-based approaches. The experimental setup consisted of a systematic overtaking scenario designed to test both methodologies under identical conditions.

An obstacle vehicle was positioned at progress variable $s = 2$ on the reference trajectory, maintaining a constant lateral offset of 0.1 m from the y-axis centerline. The obstacle follows the same reference spline as Voyager but operates at the fixed speed of 0.03 m/s without collision avoidance capabilities. Voyager begins at the origin ($s = 0$) and must execute an overtaking maneuver to progress along the reference trajectory while avoiding collision with the slower-moving obstacle.

The experimental design creates a controlled, repeatable scenario that isolates the performance characteristics of each obstacle avoidance formulation. The obstacle's slower operational speed relative to Voyager's minimum velocity (0.04 m/s) forces active spatial planning rather than passive speed matching, directly testing the MPCC algorithm's ability to generate safe overtaking trajectories.

Fig. 5 (a) and Fig. 5 (b) shows the trajectories of both the vehicle and obstacle, including their respective radii and a selected safety zone of 0.3 m, illustrating the obstacle overtaking maneuver for both methods. Tab. 2 gives key performance indicators collected over 20 laps from tests conducted with prediction and control horizons ranging from 3 to 10. The key performance indicators include the prediction and control horizon ranges (averaged across the range), the corresponding average solver time (averaged over the same range and with both methods), and a comparison of lap times for the inequality-constraint and cost-function methods against the baseline lap time recorded without an obstacle.

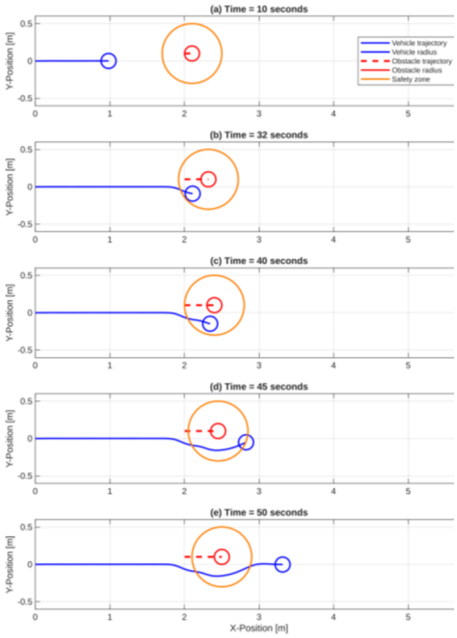

Fig. 5 (a). Dynamic obstacle avoidance using dynamic cost function weighting for a prediction and control horizon of 3 time steps

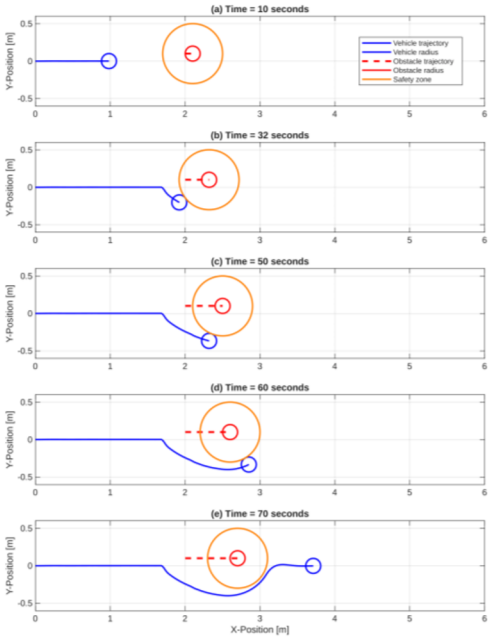

Fig. 5 (b). Dynamic obstacle avoidance using obstacles as inequality constraints for a prediction and control horizon of 3 time steps

Table 2. Performance metrics for various prediction and control horizons in the MPCC framework.

Pred Horizon (steps)	Ctrl Horizon (steps)	Average Solver Time (s)	Baseline Lap Time (s)	Lap Time with Ineq Constraints Method (s)	Lap Time Increase - Inequality (%)	Lap Time with Cost Method (s)	Lap Time Increase - Cost (%)
3	3	0.01734	953.66	984.23	3.10	970.1	1.69
4	3-4	0.02235	953.65	983.83	3.06	969.98	1.68
5	3-5	0.0308	953.6	984.64	3.15	970.08	1.69
6	3-6	0.0427	953.45	984.95	3.19	970.1	1.71
7	3-7	0.05108	953.76	984.87	3.15	969.93	1.66
8	3-8	0.083	953.67	984.56	3.13	969.99	1.68
9	3-9	0.114	953.55	984.36	3.12	969.99	1.69
10	3-10	0.1228	953.65	984.33	3.11	970	1.68

5. Discussion

Both methodologies demonstrated excellent real-time performance characteristics suitable for closed-loop control implementation. With prediction and control horizons set to 3 time steps each, the optimization problem maintained computational tractability while providing sufficient look-ahead for safe obstacle avoidance.

The cost function approach achieved an average solver time of 0.01698 seconds per optimization cycle for a prediction horizon and control horizon of 3, representing highly efficient convergence behavior which increases to an average solver time of 0.1228 seconds per optimization cycle for a prediction of 10. This performance demonstrates that the Gaussian weighting mechanism does not significantly increase computational burden compared to the baseline MPCC formulation.

The inequality constraint approach recorded an average solver time of 0.01734 seconds, exhibiting marginally higher computational requirements due to the additional constraint evaluations. However, this difference of 0.00036 seconds (approximately 2.1% increase) remains well within acceptable real-time control margins for the 10 Hz control frequency requirement.

The cost function approach consistently achieved superior lap time performance, with increases ranging from 1.66% to 1.71% compared to the baseline across all prediction horizon configurations. In contrast, the inequality constraint method demonstrated lap time increases of 3.06% to 3.19%, representing approximately double the performance penalty of the cost function approach. This substantial difference highlights the trade-off between safety conservatism and trajectory efficiency, where the hard constraint formulation's guaranteed safety margins come at the cost of reduced racing performance.

Notably, both methods maintained consistent lap time performance across varying prediction horizons, indicating that the obstacle avoidance behavior remains stable regardless of the look-ahead distance. The cost function method's superior lap time performance (averaging 1.68% increase) compared to the inequality constraint method (averaging 3.13% increase) demonstrates that the Gaussian penalty structure enables more efficient trajectory optimization while maintaining collision-free operation.The inequality constraint approach Fig 5 (b) exhibits conservative trajectory planning with the vehicle maintaining clear separation from the obstacle's safety zone throughout the entire overtaking maneuver. This

behavior reflects the hard constraint formulation that mathematically guarantees minimum separation distances.

Both approaches consistently maintained solver convergence within the 0.1-second control sampling interval and safely avoided the obstacle, validating the computational feasibility of real-time MPCC-based obstacle avoidance for differential drive platforms. The efficient solver performance enables deployment on resource-constrained embedded computing platforms while maintaining safety and performance guarantees.

The simulation results demonstrate distinct behavioral characteristics between the two methodologies. The cost function approach in Fig 5 (a) displays more aggressive trajectory optimization, with the vehicle trajectory approaching closer to the obstacle while still maintaining collision-free operation. The Gaussian penalty structure allows the optimizer to trade safety margins for trajectory efficiency when computationally beneficial, resulting in smoother but potentially less conservative overtaking paths and explaining the superior lap time performance observed in the quantitative results.

The inequality constraint approach Fig 5 (b) exhibits conservative trajectory planning with the vehicle maintaining clear separation from the obstacle's safety zone throughout the entire overtaking maneuver. This behavior reflects the hard constraint formulation that mathematically guarantees minimum separation distances, directly correlating with the higher lap time penalties observed across all prediction horizon configurations.

Both methodologies successfully achieved collision-free navigation while maintaining reasonable progress along the reference trajectory, validating the theoretical framework under simulated conditions.

6. Conclusion

This work presents a Model Predictive Contouring Control framework for the trajectory planning and collision avoidance on the CSIR's Voyager differential-drive UGV within structured racetrack environments. The MPCC formulation successfully unifies path following and short-horizon planning using linear-spline track representation and curvilinear progress dynamics, effectively balancing progress maximization with contouring and lag error minimization.

The systemic validation in MATLAB demonstrates that the framework can achieve smooth and safe overtaking maneuvers under controlled conditions with known obstacle trajectories.

The system shows a fundamental trade-off between planning foresight and computational feasibility, with smaller prediction and control horizons achieving real-time performance while longer horizons that improve lookahead exceed the computational budget. This acts as an important guidance for practical MPCC deployment on the Voyager, where computational resources are constrained.

The study establishes a validated baseline for integrating dynamic obstacle avoidance under perfect trajectory knowledge, demonstrating the framework's potential for autonomous navigation in structured environments.

The next phase involves transitioning from MATLAB validation to ROS2-based system integration to bridge the gap between algorithm validation and practical deployment. The validated MPCC algorithm will be implemented as a ROS2 node with the Voyager's embedded architecture. The ROS2 node will interface with the vehicle's control systems through standardized ROS topics.

Prior to physical implementation, the integrated system will be validated in physics simulators including Gazebo and Rviz. The simulation environment will provide realistic vehicle dynamics, sensor noise, and environmental interactions that cannot be captured in MATLAB's kinematic models.

Reference trajectories will be generated through SLAM-based mapping followed by centerline extraction algorithms. RTAB-MAP's localization package will provide robust pose estimation by fusing data from the onboard 3D LiDAR, IMU, and wheel encoders.

An obstacle vehicle will be introduced that broadcasts its trajectory and position over the ROS2 network. This setup will validate the MPCC framework's collision avoidance capabilities under realistic communication constraints while maintaining controlled experimental conditions.

The integrated system will be evaluated on computational performance, trajectory tracking, obstacle avoidance safety margins, and communication between components.

References

1. S. Bouzoualegh, E.-H. Guechi, R. Kelaiaia, Model Predictive Control of a Differential-Drive Mobile Robot, Acta Universitatis Sapientiae, Electrical and Mechanical Engineering, **10**, 20–41, (2018). https://doi.org/10.2478/auseme-2018-0002

2. B. Evans, Accelerating deep reinforcement learning for autonomous racing, Ph.D. thesis, Stellenbosch: Stellenbosch University, 2023

3. O. D. Keanly, J. A. A. Engelbrecht, Trajectory Planning and Collision Avoidance for an Autonomous Racing Car, (2024)

4. A. Bertipaglia, M. Alirezaei, R. Happee, B. Shyrokau, Model Predictive Contouring Control for Vehicle Obstacle Avoidance at the Limit of Handling, arXiv preprint arXiv:2308.06742, (2023)

5. X. Kong, C. Fisher, B. Evans, A Critical Evaluation of Pure Pursuit, MPC and MPCC Controllers, MATEC Web of Conferences, **406**, 4013, (2019)

6. D. Lam, C. Manzie, M. Good, Model Predictive Contouring Control, in 49th IEEE Conference on Decision and Control (CDC)(2010), 6137–6142

7. A. Heilmeier, A. Wischnewski, L. Hermansdorfer, J. Betz, M. Lienkamp, B. Lohmann, Minimum Curvature Trajectory Planning and Control for an Autonomous Race Car, Vehicle System Dynamics, **58**, 1497–1527, (2020)

8. A. Liniger, A. Domahidi, M. Morari, Optimization-Based Autonomous Racing of 1:43 Scale RC Cars, in IEEE Conference on Control Applications (CCA)(2015), 714–719

9. G. Borrello, L. Lorusso, M. Basso, A. Acernese, Trajectory Planning Based on Model Predictive Control with Dynamic Obstacle Avoidance, IEEE Transactions on Intelligent Transportation Systems, (2021)

10. B. Brito, B. Floor, L. Ferranti, J. Alonso-Mora, Model Predictive Contouring Control for Collision Avoidance in Unstructured Dynamic Environments, IEEE Robot Autom Lett, **4**, 4459–4466, (2019)

11. L. Di, N. Huang, J. He, X. Wu, H. Huang, Y. Su, others, Trajectory tracking and obstacle avoidance in dynamic environments using an improved artificial potential field method, PLoS One, **20**, e0326879, (2025)

12. C.-H. Wu, K.-Y. Chan, Developing a dynamic obstacle avoidance system for autonomous mobile robots using Bayesian optimization and object tracking: Implementation and testing, Proc Inst Mech Eng C J Mech Eng Sci, (2024)

13. M. Shi, G. Chen, Á. Serra Gómez, S. Wu, J. Alonso-Mora, Evaluating Dynamic Environment Difficulty for Obstacle Avoidance Benchmarking, arXiv preprint arXiv:2404.14848, (2024)

14. K. Purdon, J. S. Dickens, W. De Ronde, K. Ramruthan, G. Crafford, Voyager, a ground mobile robotic platform for research development, https://researchspace.csir.co.za/items/efd30aa7-ec13-42b8-81ab-def1773bb3b7, (2023)

15. J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, M. Behl, V. Krovi, R. Mangharam, Autonomous Vehicles on the Edge: A Survey on Autonomous Vehicle Racing, IEEE Open Journal of Intelligent Transportation Systems, **3**, 458–488, (2022). https://doi.org/10.1109/ojits.2022.3181510