



Effects of Online Disturbance Tube Adaptation on Conservativeness and the Safety-Performance Trade-off

Adaptive Tube-based MPC for Autonomous Search-and-Rescue Robots

Nicolas Laplagne¹

Supervisors: A. Jamshidnejad¹, S. Schoonebeek¹

¹EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 21, 2026

Name of the student: Nicolas Laplagne

Final project course: CSE3000 Research Project

Thesis committee: A. Jamshidnejad, S. Schoonebeek, L. C. Siebert

An electronic version of this thesis is available at <http://repository.tudelft.nl/>.

Abstract

Robots are increasingly used in search-and-rescue (SaR) to perform dangerous tasks in place of humans, and to free human rescue workers to focus on other tasks. These robots require effective and reliable control methods to ensure that they perform as well as the humans they replace. Model predictive control (MPC) is an optimisation-based method which innately handles hard constraints such as not colliding with victims, and has been well-studied in SaR contexts. Tube-based MPC (TMPC) is an extension of MPC which is robust to the bounded disturbances inevitable in real-life SaR, but due to its fixed tube is often overly conservative. Surma and Jamshidnejad propose state-dependant dynamic TMPC (SDD-TMPC), which adjusts tube size online based on an estimate of the current maximum disturbance, allowing for reduced conservativeness when possible disturbances are small. We develop a grid-based SaR simulation to evaluate how SDD-TMPC’s adaptive tube affects conservativeness and the safety-performance trade-off in a dynamic, partially observable environment. SDD-TMPC, MPC, and TMPC are compared over many simulated search missions, where the agent is tasked with finding all victims in an environment. The results show that while retaining the empirical safety of TMPC, SDD-TMPC behaves less conservatively and outperforms TMPC in mission feasibility and duration.

1 Introduction

As population density increases and the frequency and severity of extreme weather events grow, robots are used more and more often in post-disaster search and rescue (SaR) contexts [1–3]. Robots can be used instead of humans for the most dangerous tasks in an SaR operation, and can explore areas that would have otherwise posed too great a risk to human rescue workers. If autonomous, SaR robots can additionally free rescue workers to focus on other tasks, potentially allowing for increased efficacy and better post-disaster outcomes. However, due to the safety-critical nature of SaR operations, such systems must be as reliable as the humans they replace. Particularly, we often define hard constraints which must be satisfied at all times throughout a mission, such as not colliding with or otherwise injuring victims, or avoiding damage to the robot.

Model predictive control (MPC) is a well-studied control technique with a wide variety of applications for the control of both manned and autonomous systems, including in SaR contexts [4, 5]. MPC uses a model (often physics-based) to predict how different possible control actions will affect the system under control across a time horizon, and can then use these predictions to optimise for multiple objectives simultaneously while innately handling constraints. However, because MPC relies so heavily on its model, it is not robust to unmodelled disturbances, which are inevitable in real-life

situations. Since such disturbances may cause a robot controlled by MPC to inadvertently violate its constraints, the need arises for a control method that is robust to external disturbances.

The literature presents many extensions of MPC that aim to address this issue; of particular interest to this paper is robust tube-based MPC (TMPC). Whereas pure MPC only considers the nominal state trajectory of an action, TMPC instead considers the set of states reachable from the nominal trajectory by a bounded disturbance, known as the invariant tube. Therefore, a TMPC controller is robust to the effects of such disturbances without needing to explicitly model them. A key issue identified in [6], however, is that TMPC can often be overly conservative, as the invariant tube must be large enough to encompass all possible disturbances in a system. This can lead to worse performance than MPC in situations where the tube is larger than necessary for the disturbances in a given state, and can even cause TMPC to decide that an objective is infeasible where a less conservative formulation would not.

In [6], Surma and Jamshidnejad propose state-dependant dynamic TMPC (SDD-TMPC), a novel variant of TMPC which dynamically adjusts the size of the tube based on the current estimated uncertainty. Critically, rather than acting as a pure MPC controller as in standard TMPC, the nominal controller in SDD-TMPC is provided with the predicted tube size adjustments, allowing it to actively factor the dynamic tube size into its control decisions. The system can therefore act less conservatively when uncertainty is low, or even act so as to reduce uncertainty where necessary, all while retaining the safety guarantees of TMPC. The authors demonstrate how SDD-TMPC outperforms both TMPC and MPC in a variety of scenarios, including being able to find solutions to tasks where TMPC could not, but the case studies presented are by the authors’ own admission ‘carefully designed simulations’, tailored to highlight where SDD-TMPC performs better than the other two methods.

A significant gap in the literature exists in the evaluation of state-of-the-art model predictive control methods such as SDD-TMPC in more realistic environments. While new methods are often proposed to address elements such as partial observability [7] or dynamic obstacles [8], they are often tested on toy or purely numerical examples, and there has been little research into how any of these methods maps to more complete simulations with complex goals.

In order to begin evaluating the effectiveness of SDD-TMPC for real-world SaR use, this paper investigates the following question: *How does online adaptation of the disturbance tube influence the conservativeness of safety margins and the trade-off between empirical safety and performance in a dynamic, partially observable search-and-rescue environment?*

To address this question, we consider two related sub-questions:

1. What is the quantifiable impact of an adaptive tube on mission feasibility and completion times compared to standard MPC and fixed-tube TMPC?
2. How does adaptive tube sizing influence the robot’s tra-

jectory and action choices?

2 Related Works

This section explores existing work regarding the use of MPC in SaR scenarios, *recursive feasibility* in the context of partially observable environments, and methods similar to SDD-TMPC.

The use of MPC in SaR has gained traction primarily due to its ability to optimise multiple criteria simultaneously while handling constraints [4]. MPC has been shown to be effective in a variety of SaR-related tasks, including victim-aware path planning [4] and multiple-UAV search missions [5]. However, an inherent difficulty of real-world SaR operations is the combination of partial observability and a dynamic environment, and much of the research regarding MPC in SaR chooses to neglect one or the other for the sake of maintaining recursive feasibility. The recent work in [8] proposes a TMPC-based method for navigation of a partially observable dynamic SaR environment while acknowledging the lack of recursive feasibility, but the robots are given the relatively simple goal of reaching a specified coordinate, and know the full initial state of the environment beforehand.

Other works have proposed various methods to maintain recursive feasibility despite partial observability. Approaches such as those presented in [7, 9] handle sensor uncertainty by optimising control trees where each branch represents a possible measurement, weighted by the belief state probability distribution, but can only guarantee robust satisfaction of chance constraints. Recent hierarchical formulations such as parent-child MPC enable recursive feasibility guarantees without the use of terminal constraints and at much lower computational cost, enabling a longer prediction horizon [10], but under partial observability this also only decreases the risk of recursive infeasibility.

While this paper focuses on SDD-TMPC in particular, dynamic tube-based MPC as a whole has recently enjoyed significant attention for the same promise of decreased conservativeness while retaining safety guarantees [11–13]. As discussed in Section 1, it is this class of formulation where the application to more realistic scenarios is particularly absent. Consequently, this paper aims to address this gap, at least in the case of SDD-TMPC.

3 Finite Input Set SDD-TMPC: Simulation Description and Problem Formulation

This section mathematically describes the SaR simulation, detailing the modelling of the environment, the agent’s movement and sensing mechanics, and the disturbance behaviour. Furthermore, we provide a formal problem definition for finite input set predictive control in the context of the simulation, outline the formulation of the MPC, TMPC, and SDD-TMPC controllers, and discuss their limitations with regards to safety.

3.1 Environment description

The true state of the SaR environment is modelled as a grid $\mathcal{G}(k)$ of dimension $n_h \times n_w$, varying with discrete time index $k \in \mathbb{N}$. The grid is composed of multiple matrices, each

representing a property such as traversability or fire state. We thus define

$$\mathcal{G}_k = [M^{\text{trvs}}(k), M^{\text{vuln}}(k), M^{\text{victim}}(k), M^{\text{fire}}(k)]^\top, \quad (1)$$

with the traversability matrix¹ $M^{\text{trvs}}(k) \in \{0, 1\}^{n_h \times n_w}$:

$$M_{i,j}^{\text{trvs}}(k) = \begin{cases} 0 & \text{if cell is traversable} \\ 1 & \text{if cell is obstructed} \end{cases},$$

the vulnerability matrix $M^{\text{vuln}}(k) \in [0, 1]^{n_h \times n_w}$:

$$M_{i,j}^{\text{vuln}}(k) = \begin{cases} 0 & \text{if cell safe} \\ 0.6 & \text{if cell vulnerable} \\ 1 & \text{if cell high risk} \end{cases},$$

the victim occupancy matrix $M^{\text{victim}}(k) \in \{0, 1\}^{n_h \times n_w}$:

$$M_{i,j}^{\text{victim}}(k) = \begin{cases} 1 & \text{if victim in cell} \\ 0 & \text{otherwise} \end{cases},$$

and the fire state matrix $M^{\text{fire}}(k) \in \llbracket 0, 3 \rrbracket^{n_h \times n_w}$:

$$M_{i,j}^{\text{fire}}(k) = \begin{cases} 0 & \text{if cell safe at step } k \\ 1 & \text{if cell flammable at step } k \\ 2 & \text{if cell burning at step } k \\ 3 & \text{if cell burnt at step } k \end{cases}$$

Initial states for the traversability and vulnerability matrices are procedurally generated from a seed and a global configuration object, based on a room-and-tunnel model. N_v victims and N_f point fires are then distributed randomly across the remaining traversable tiles. For further details on environment generation, see our colleague’s paper [14].

The fire state matrix evolves according to a simple four-stage model: safe cells become flammable if adjacent to a burning cell, flammable cells adjacent to burning cells ignite with probability p_f each step, and burning cells become burnt after duration T_f steps. The fire uses the same seed as the environment generation, and is thus deterministic per-map. While the simulation does allow for dynamic victims and multiple agents, this paper assumes static victims and a single agent so as not to overcomplicate the system.

3.2 Agent description

We define the agent’s state $x_k = [q_k^\top, \mathcal{P}_k^\top]^\top$ as the concatenation of the agent’s current position

$$q_k = (i_k, j_k)^\top \in \mathcal{Q} = \llbracket 0, n_h - 1 \rrbracket \times \llbracket 0, n_w - 1 \rrbracket \quad (2)$$

and the agent’s perception grid

$$\mathcal{P}_k = [P^{\text{trvs}}(k), P^{\text{vuln}}(k), P^{\text{victim}}(k), P^{\text{fire}}(k), P^{\text{conf}}(k), P^{\text{xplr}}(k)]^\top \in \mathcal{M}, \quad (3)$$

where $\mathcal{M}$ is the set of all possible perceived environment states. The agent is initially placed in a designated entry point at the top-left corner of the map, and the agent’s perception is

¹Note that M^{trvs} , M^{vuln} , and M^{victim} do not vary with k , but are defined with parameter k for convenience.

initialised to all zero matrices at a conceptual pre-simulation step $k = -1$.

The agent's perception grid $\mathcal{P}_k$ reflects a scan of its surroundings taken from its currently occupied position q_k . Chronologically, this scan occurs immediately upon the agent's arrival at q_k , and the resulting perception grid $\mathcal{P}_k$ is then used by the controller to plan the current step's action u_k . At step $k = 0$, before the agent executes its first action, an additional initial scan is performed from the starting position q_0 to provide the agent with initial information about its immediate surroundings. A cell (i, j) is considered to have been scanned at step k if both of the following conditions are satisfied:

1. The cell is within scan radius r of the agent's current position, more precisely $\sqrt{(i_k - i)^2 + (j_k - j)^2} \leq r$.
2. The cell is determined to be visible to the agent by the restrictive precise angle shadowcasting algorithm, as implemented by libtcod's `compute_fov` function [15, 16].

The first four matrices in $\mathcal{P}$ represent the agent's perceived knowledge of the environment. For each step $k \geq 0$, the matrices are updated in reference to their ground truth counterpart as follows:

$$P_{i,j}^{\dots}(k) = \begin{cases} M_{i,j}^{\dots}(k) & \text{if cell scanned at step } k \\ P_{i,j}^{\dots}(k-1) & \text{otherwise} \end{cases}. \quad (4)$$

The confidence matrix $P^{\text{conf}}(k) \in [0, 1]^{n_h \times n_w}$ represents the agent's current certainty about the information associated with each cell. It is updated using the following rule:

$$P_{i,j}^{\text{conf}}(k) = \begin{cases} P_{i,j}^{\text{conf}}(k-1) + \eta_{x_k}(i, j) & \text{if cell scanned} \\ \cdot (1 - P_{i,j}^{\text{conf}}(k-1)) & \text{at step } k \\ \max(P_{i,j}^{\text{conf}}(k-1) - \sigma, 0) & \text{otherwise} \end{cases}, \quad (5)$$

where σ is the confidence decay parameter, and η_{x_k} is the scan accuracy function:

$$\eta_{x_k}(i, j) = \eta_0 \cdot \exp \left[-\frac{1}{2} \frac{(i_k - i)^2 + (j_k - j)^2}{(\frac{r}{3})^2} \right]$$

with $\eta_0 \in (0, 1]$ the base scan accuracy.

Finally, the exploration matrix $P^{\text{xplr}}(k) \in \{0, 1\}^{n_h \times n_w}$ tracks which cells have been scanned by the agent so far:

$$P_{i,j}^{\text{xplr}}(k) = \begin{cases} 1 & \text{if cell scanned at step } k \\ P_{i,j}^{\text{xplr}}(k-1) & \text{otherwise} \end{cases}. \quad (6)$$

3.3 Disturbance mechanics

Following each control action, the agent additionally experiences a disturbance $w(x_k) \in \mathbb{W}(x_k)$, dependent on the target cell's vulnerability and the accuracy of the agent's perception of the cell. $w(x_k)$ is determined probabilistically as follows:

$$w(x_k) = \begin{cases} \mathbf{0} & \text{with probability } 1 - p_d(k) \\ \text{Uniform}(\mathbb{W}) & \text{with probability } p_d(k) \end{cases}, \quad (7)$$

with

$$p_d(k) = \max(M_{\tau_k}^{\text{vuln}}(k) - P_{\tau_k}^{\text{conf}}(k), 0)^2. \quad (8)$$

We use τ_k to denote the agent's target cell $q_k + u_k$ at step k , and refer to the case where $w(x_k) \in \mathbb{W}$ as a *slip*. If a slip would cause the agent to enter a cell which is occupied by a victim, obstructed, or out of bounds, the disturbance is negated and a victim or terrain collision is recorded. Slipping into a burning cell does not negate the disturbance, but a fire collision is recorded. The agent is otherwise unaffected by collisions.

The target cell's vulnerability can be thought of as the difficulty of manoeuvring in that cell, e.g. the likelihood that a limb placed in that cell will become unstable, while the confidence encodes how closely the agent's perception of that cell matches reality, with greater confidence conferring a better chance of entering the cell safely despite difficult terrain. The squaring of the difference between vulnerability and confidence ensures that navigating highly vulnerable terrain with little prior knowledge is heavily penalised, while minor perception gaps are relatively forgiving.

3.4 Problem statement

We consider, similarly to [6], a discrete-time system with state vector x_k and control input u_k , affected by additive state-dependent external disturbance $w(x_k)$ at time $k \in \mathbb{N}$, where $x_k \in \mathbb{X}$, $u_k \in \mathbb{U}$, $w(x_k) \in \mathbb{W}(x_k)$. $\mathbb{X}$, $\mathbb{U}$, and $\mathbb{W}(x_k)$ are the admissible sets for the state, control input, and disturbance respectively. Since we are operating in a strict grid-based simulation, $\mathbb{U}$ is exactly the Von Neumann neighbourhood of the origin in $\mathbb{Z}^2$, representing the actions UP, DOWN, LEFT, RIGHT, and WAIT. As suggested by Section 3.3, $\mathbb{W}(x_k)$ is state dependent, such that $\mathbb{W}(x_k) = \{\mathbf{0}\}$ when $p_d(k) = 0$, and $\mathbb{W}(x_k) = \mathbb{W}$ otherwise, where $\mathbb{W} = \mathbb{U} \setminus \{\mathbf{0}\}$, implying a maximum disturbance magnitude of one grid cell per time step.

In order to adapt the system in [6, eq. (1)] to the simulation, the state transition $x_{k+1} = [q_{k+1}^\top, \mathcal{P}_{k+1}^\top]^\top$ must be evaluated sequentially. First, the agent's physical position is updated subject to the control input and disturbance:

$$q_{k+1} = q_k + u_k + w(x_k). \quad (9)$$

Subsequently, the agent updates its perception grid from its new location:

$$\mathcal{P}_{k+1} = f(q_{k+1}, \mathcal{P}_k), \quad (10)$$

where $f : \mathcal{Q} \times \mathcal{M} \rightarrow \mathcal{M}$ updates the perception grid as described in Eqs. (4) to (6).

We assume the agent has complete knowledge of its own state, i.e. that the value of x_k is perfectly known at each time step, whereas the disturbance at each step is unknown and may take any value within $\mathbb{W}(x_k)$. Similarly, the environment is initially unknown, and the agent must inform its perception of the world based on local observation of the environment. The goal of each controller is to select actions such that a given objective function J is maximised across a prediction horizon of size N , while aiming to satisfy hard constraints.

Thus for time step k , the general robust formulation is:

$$J^*(x_k) = \max_{\tilde{x}_k, \tilde{u}_k} J(\tilde{x}_k, \tilde{u}_k) \quad (11)$$

s.t. for $t = k, \dots, k + N - 1$:

$$\begin{aligned} q_{t+1|k} &= q_{t|k} + u_{t|k} + w(x_{t|k}) \\ \mathcal{P}_{t+1|k} &= f(q_{t+1|k}, \mathcal{P}_{t|k}) \\ x_{k|k} &= x_k \\ x_{t+1|k} &\in \mathbb{X} \\ u_{t|k} &\in \mathbb{U} \\ w(x_{t|k}) &\in \mathbb{W}(x_{t|k}), \end{aligned}$$

as in as in [6, eq. (2)], with $\tilde{x}_k = (x_{k+1|k}, \dots, x_{k+N|k})$, $x_{t|k}$ the value of x_t predicted at time k , $\tilde{u}_k = (u_{k|k}, \dots, u_{k+N-1|k})$, $u_{t|k}$ the value of u_t decided at time k , N the prediction horizon, and J^* the optimal value for the objective function.

3.5 Prediction model

The state transition model used by the controller is similar to the state transition defined in the previous section. The agent's position is updated with respect to the predicted control action from the previous time step, as in Eq. (9), but as the disturbance $w(x_{t|k})$ is unknown, it is not taken into account:

$$\begin{aligned} \text{for } t = k, \dots, k + N - 1 : \\ q_{t+1|k} &= q_{t|k} + u_{t|k}. \end{aligned} \quad (12)$$

Similarly to Eq. (10), the agent's predicted perception grid is then updated from the new position via a mock scan:

$$\mathcal{P}_{t+1|k} = g(q_{t+1|k}, \mathcal{P}_{t|k}). \quad (13)$$

However, $g : \mathcal{Q} \times \mathcal{M} \rightarrow \mathcal{M}$ differs somewhat from f in Eq. (10), as the controller cannot predict what it will observe in unexplored tiles. In contrast to Eq. (4), the traversability, vulnerability, and victim matrices are not updated, as there is no way to do so that would add useful information.

Since we consider the agent to have knowledge of the dynamics of the environment, the fire matrix is updated, and by almost the same process as described in Section 3.1. However, as the true fire spread is stochastic, the controller must assume that, at each step, any tile which could ignite does so, or else risk violating constraints due to a misprediction. Because the computational complexity of online MPC restricts us to a finite prediction horizon, this formulation cannot guarantee recursive feasibility with respect to fire spread (discussed in detail in Section 3.7).

Finally, the confidence and exploration matrices are updated identically to Eqs. (5) and (6), as they represent meta-information about the agent's knowledge of the environment, rather than directly reflecting the environment itself.

3.6 Controller formulations

Given the SaR environment defined in Section 3.1, we define four constraints, shared by all three formulations:

- The agent must be within the bounds of the grid
- The agent must not occupy an obstructed cell

- The agent must not occupy a burning cell
- The agent must not occupy the same cell as a victim

We thus define the feasible set $\mathbb{F}$ to be the set of all states which do not break these constraints:

$$\mathbb{F} = \{x_k \in \mathcal{Q} \times \mathcal{M} \mid \mathbf{0} \leq q_k < (n_h, n_w)^\top, P_{q_k}^{\text{trvs}}(k) \neq 1, P_{q_k}^{\text{fire}}(k) \neq 2, P_{q_k}^{\text{victim}}(k) \neq 1\} \quad (14)$$

In the context of the overall SaR operation, we define *mission feasibility* as the agent's ability to successfully locate all victims while strictly remaining within $\mathbb{F}$.

Because the agent does not have direct access to the ground truth, $\mathbb{F}$ must be defined using the agent's perception matrices. We discuss the implications of this regarding safety and planning in Section 3.7.

All three formulations also share the same objective function J , split into a stage objective L and terminal objective V :

$$J(\tilde{x}_k, \tilde{u}_k) = \sum_{i=k+1}^{k+N} L(x_{i|k}) + V(x_{k+N|k}). \quad (15)$$

The stage objective is evaluated once per prediction step and then accumulated across the prediction horizon $\{k + 1, \dots, k + N\}$. The function rewards scanning previously un-scanned cells (*discovery*) and keeping the total global confidence high (*confidence*):

$$\begin{aligned} L(x_{t|k}) &= \omega_{\text{dscv}} \cdot \sum_{i=0}^{n_h-1} \sum_{j=0}^{n_w-1} (P_{i,j}^{\text{xplr}}(t|k) - P_{i,j}^{\text{xplr}}(t-1|k)) \\ &\quad + \omega_{\text{conf}} \cdot \sum_{i=0}^{n_h-1} \sum_{j=0}^{n_w-1} P_{i,j}^{\text{conf}}(t|k). \end{aligned} \quad (16)$$

The terminal objective V is computed only for the final prediction step $k + N$, and effectively penalises the distance from the agent's final position to the closest unexplored cell (*exploration*):

$$V(x_{k+N|k}) = -\omega_{\text{xplr}} \cdot D_{q_{k+N|k}}(x_k) \cdot N. \quad (17)$$

The matrix $D(x_k)$ is constructed at the beginning of the prediction horizon as follows:

$$D_{i,j}(x_k) = \begin{cases} 0 & \text{if } P_{i,j}^{\text{trvs}}(k) = 0 \text{ and } P_{i,j}^{\text{xplr}}(k) = 0 \\ \infty & \text{if } P_{i,j}^{\text{trvs}}(k) = 1 \text{ or } (P_{i,j}^{\text{xplr}}(k) = 1 \\ & \text{but no path to any unexplored cell}) \\ d & \text{otherwise, with } d \text{ the length of the} \\ & \text{shortest path to any unexplored cell} \end{cases}, \quad (18)$$

where a path is considered valid if constraints are satisfied for all cells along the path. This establishes a heuristic for the agent to follow towards unexplored areas, to prevent stagnation when the distance the agent would have to travel to scan new cells is greater than the prediction horizon N . Notably, the reliance on constraints for path validity implies that $D(x_k)$ may vary between controller formulations due to differences in constraint tightening, but only insofar as whether the agent can feasibly reach a given unexplored area.

MPC

The MPC formulation is provided with no knowledge of the disturbances, and its admissible state set is therefore identical to the feasible set $\mathbb{F}$:

$$\mathbb{X}_{\text{MPC}} = \mathbb{F}. \quad (19)$$

TMPC

The TMPC formulation uses a fixed tube, and must therefore assume maximal disturbance at every step in order to ensure safety guarantees. We thus define

$$\mathbb{X}_{\text{TMPC}} = \mathbb{F} \ominus \mathbb{W}, \quad (20)$$

where $\ominus$ denotes the Pontryagin set difference. In this case, this shrinks the feasible set by removing all cells for which a one-step disturbed neighbour violates the constraints.

SDD-TMPC

In contrast, the SDD-TMPC formulation uses an adaptive tube, allowing it to grow or shrink the admissible state set based on its perception. As in the case of fire spread, we assume the agent has knowledge of the mechanics of the disturbance, but does not know whether any individual action will result in a slip. In order to maintain safety, the controller must therefore treat any chance of a slip greater than 0 as guaranteed:

$$\mathbb{X}_{\text{SDD-TMPC}} = \begin{cases} \mathbb{F} & \text{if } p'_d(t|k) = 0 \\ \mathbb{F} \ominus \mathbb{W} & \text{otherwise} \end{cases}. \quad (21)$$

This effectively makes the tube of SDD-TMPC binary, and this formulation is therefore a simplification of the continuously varying SDD-TMPC proposed in [6]. The decision to simplify the tube was informed by the strict grid-based nature of the simulation, as the controller would not be able to correct for a disturbance of more than one cell and would therefore not be able to satisfy constraints for any tube larger than $\mathbb{W}$.

Similarly to Eq. (14), since the agent does not have access to M^{vuln} , the controller cannot directly predict $p_d(k)$ using Eq. (8), and instead predicts $p'_d(k)$ by substituting in P^{vuln} (again discussed in Section 3.7).

Additionally, we assume an artificial global confidence of $1 - \epsilon$ when calculating $D(x_k)$ for SDD-TMPC, to prevent low current confidence from prematurely invalidating paths to distant unexplored regions. As the agent can achieve confidence arbitrarily close to 1 for any traversable cell by repeatedly scanning it, any path deemed valid under this assumption is eventually realisable. Safety is unaffected, as the controller evaluates immediate trajectory feasibility using actual confidence, ensuring the agent only enters cells with sufficiently high confidence.²

3.7 Local safety and limitations on recursive feasibility

While the controllers are designed to satisfy hard constraints, the combination of partial observability, dynamic hazards,

²The same artificial confidence could easily be used for MPC or TMPC, but it would have no impact as their constraints and therefore path validity do not depend on confidence.

and computational limits precludes guarantees of recursive feasibility. Instead, the tube-based controllers enforce *local safety*, acknowledging limitations regarding long-term mission viability. Although pure MPC strictly enforces constraints on the nominal trajectory, it cannot guarantee even local safety in the presence of unknown disturbances.

As described in Section 3.6, we define the constraints in relation to the agent's perception grid $\mathcal{P}_k$, which is naturally susceptible to belief errors regarding unexplored areas. This susceptibility is a recurring challenge in partially observable MPC literature. In our simulation, the agent may plan a path based on its current belief state, only to find an obstruction upon moving closer, causing the controller to plan sub-optimally in the later stages of the prediction horizon. However, because the agent is disturbed by a maximum of one grid cell per time step, a scan radius $r \geq 2$ ensures that any cell will be scanned before it can be physically reached. Therefore, the agent's perception of all reachable cells is guaranteed to be accurate before executing an action, allowing the tube-based controllers to maintain local safety without needing to account for belief errors.

Despite this guarantee of local safety for the robust formulations, the presence of dynamic obstacles introduces computational limits. To guarantee recursive feasibility against a dynamic hazard like fire, the prediction horizon would have to be large enough to contain a complete path to an un-burnable terminal zone, assuming such a path even exists. As the computational complexity of the MPC optimisation problem grows exponentially with the horizon length, enforcing strict recursive feasibility would be computationally intractable for online control. Therefore, all three controllers rely on the finite horizon N to maintain local safety (or nominal safety, for MPC) and must accept a non-zero risk of constraint violation in adversarial edge cases.

Finally, the nature of the simulated fire presents a theoretical limit. In this SaR environment, the true fire spread is stochastic and theoretically unconstrained. While the per-step growth of the fire is bounded, in an extreme adversarial scenario, it is possible for the fire to spread to cover all traversable tiles on the map. In this case, a terminal safe set does not exist. Therefore, regardless of computational capacity or the length of the prediction horizon, strict recursive feasibility remains theoretically impossible for this specific simulation setup.

4 Results

This section presents a comparative analysis of all three controllers in the SaR environment. To thoroughly evaluate the research question, this section is split into two parts, roughly corresponding to the two sub-questions presented in Section 1. The first subsection focuses on qualitative performance and safety metrics recorded across many simulated SaR missions. The second isolates individual trajectory or action choices made by the controllers and compares their behaviour in a qualitative manner, as well as discussing the conservativeness of the controllers.

The experiment was conducted over 500 randomly gener-

ated seeds with the same configuration³, with each agent attempting each seed once. The agents’ goal was to scan every victim in the environment. For each run, we recorded the outcome, number of victims found, mission duration, average safety margin, and the number of times the agent collided with terrain, a victim, or a burning tile. Based on previous testing, it was concluded that the vast majority of runs where the mission is feasible finish in under 300 steps. Therefore, the simulations were constrained to a maximum of 300 steps and were terminated early if all victims were found before then. If an agent reached 300 steps, it was considered to have failed the seed, and the outcome was recorded as a timeout. Because each seed was attempted by all three controllers, statistical analysis was performed under the assumption of completely dependent data, and tests suitable for such data were used.

4.1 Empirical safety and performance

Collision rate

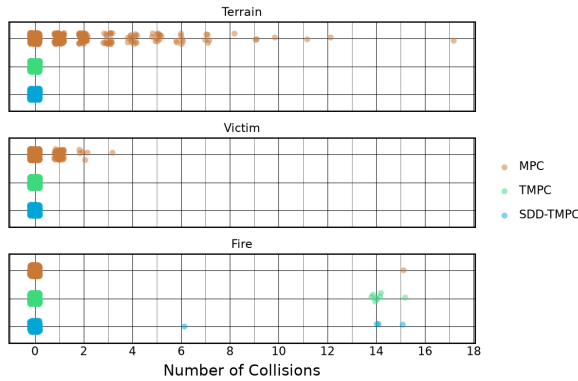


Figure 1: Number of collisions per run separated by hazard type and agent controller. Each point represents a run with the specified controller where x collisions of the specified type occurred. Both tube-based formulations only suffered fire collisions. The pure MPC agent, while avoiding collisions in most runs, did often collide with the terrain and victims due to disturbances.

Fig. 1 shows the number of collisions per run for each formulation. TMPC and SDD-TMPC never collided with terrain or a victim, but the lack of recursive feasibility regarding the dynamic fire did allow them to collide with fire in a small number of runs, as discussed in Section 3.7. MPC, on the other hand, collided in 48.0% of runs, and experienced up to 17 terrain collisions in a single run.

Interestingly, a Friedman test revealed a statistically significant difference in the number of fire collisions per run across the controllers ($\chi^2 = 7.40, p = 0.025$). Although post-hoc comparisons using Wilcoxon signed-rank tests with a Bonferroni correction ($\alpha = 0.016$) found no significant difference between MPC and SDD-TMPC ($p = 0.307$) or between TMPC and SDD-TMPC ($p = 0.323$), these tests confirmed that TMPC collided with fire significantly more than MPC ($p = 0.024$).

³Provided in Appendix A.

We hypothesise that this is due to TMPC being overly conservative, preventing it from finding a way through or around the fire when MPC can. In attempting to satisfy hard constraints, TMPC is furthermore unable to leave a burning cell due to the risk of tripping back into it, instead reporting the optimisation problem as infeasible until the fire burns out. The same is true for SDD-TMPC until it believes there is no longer a risk of tripping, and we believe this is the reason for the clusters present at 14 fire collisions for both robust controllers; this was exactly the fire duration in our testing.

Mission feasibility and duration

Table 1 illustrates the proportion of runs for each formulation where all victims were found before the time limit, for the no-consequence scenario as well as the case where any collision is treated as a failure. Paired exact McNemar’s tests were performed, which confirmed statistically significant differences ($p < 0.001$) between all three controllers in both scenarios.

Table 1: Proportion of runs where all victims were found in 300 steps or fewer, when collisions are ignored or treated as a failure.

Consequence of collision	Success rate (%)	
	None	Failure
MPC	97.2	50.2
TMPC	35.0	34.4
SDD-TMPC	81.6	81.2

In the no-consequence case, we see that MPC has a near-perfect success rate, able to find all victims in 97.2% of missions. This is followed closely by SDD-TMPC, which in turn had more than double the success rate of fixed-tube TMPC. This validates the expectation that SDD-TMPC should encounter fewer infeasible missions than fixed-tube TMPC. In the SaR environment specifically, TMPC cannot pass through any one-wide gaps, while the same is only true for SDD-TMPC if said gap is also high-risk. Since MPC has no tube, it should consequently have no such restrictions, and this is borne out by the data. This data supports the hypothesis that SDD-TMPC should outperform TMPC by virtue of more missions being feasible, but still lag behind the less restricted MPC.

However, to assume that collisions never incur consequences biases the data in favour of MPC, which is able to move more freely than the tube-based formulations by ignoring disturbances. When treating any collision as ‘fatal’ for the robot, we see that the success rates of TMPC and SDD-TMPC are barely affected, as they experience very few collisions overall. MPC’s success rate, on the other hand, sees a significant drop compared to its ideal-case success rate, falling far below that of SDD-TMPC. These findings are interpreted alongside additional context in Section 4.2.

Fig. 2 illustrates the mission durations for each formulation, only including successful runs. While the distributions across all three controllers are similar, pure MPC holds a slight speed advantage. As the data is unbalanced and non-normally distributed, we used a linear mixed-effects model

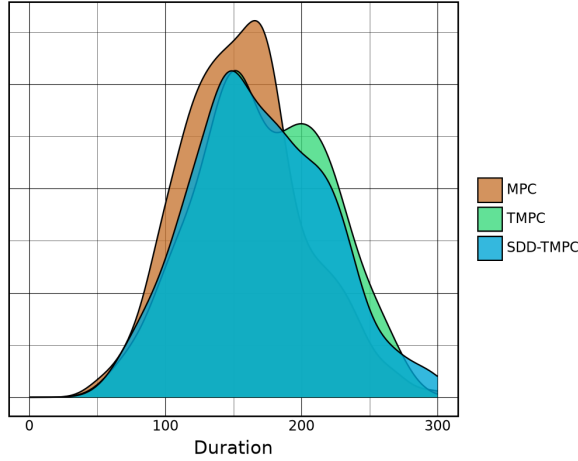


Figure 2: Distribution of mission durations by agent controller, excluding timeouts. MPC was still often the fastest of the three controllers, but all three performed similarly in practice.

on log-transformed durations and found statistically significant differences among the controllers ($p < 0.001$). We then isolated the seeds where both controllers in a pair succeeded, and post-hoc Wilcoxon signed-rank tests with a Bonferroni adjustment confirmed these speed differences ($p < 0.001$). Successful missions were an average of 13 steps shorter for SDD-TMPC than for TMPC, demonstrating that SDD-TMPC outperforms fixed-tube TMPC even when pure feasibility is not considered. However, SDD-TMPC is still slower than pure MPC by an average of 14 steps when both succeed.

4.2 Trajectory and action choices

Manual inspection of the agents’ trajectories for a small subset of runs reveals interesting qualitative differences between the controllers. As expected, MPC acts the least conservatively of the three, but is visibly faster, and TMPC is often unable to enter specific rooms or entire sections of the map due to the entrances being only one or two cells wide, and always leaves a one cell safety margin between itself and hazards except in adversarial edge cases.

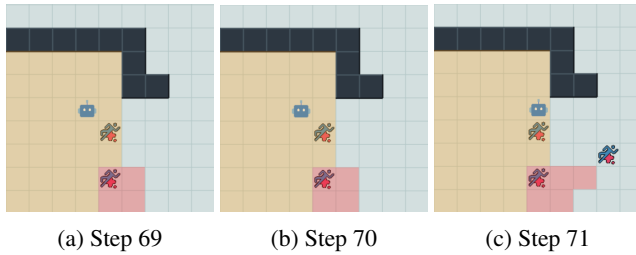


Figure 3: Agent perception at steps 69-71 in an SDD-TMPC run (agent: robot icon, victim: body icon, wall: black, safe: white, vulnerable: yellow, high-risk: red). At step 69, the agent encounters a one-wide gap in a vulnerable zone, but its confidence is too low to guarantee it does not slip into the victim, and it therefore waits. At step 70, the agent’s confidence is high enough, and it can safely pass the victim, seen in step 71.

SDD-TMPC exhibits a number of emergent behaviours following from its adaptive tube. Fig. 3 shows a scenario in which SDD-TMPC waits to accumulate sufficient confidence to safely pass through a thin gap. In this same scenario, MPC proceeds without enough confidence and risks colliding with the victim, and TMPC is entirely unable to enter the room to the right.

Furthermore, in contrast to TMPC, SDD-TMPC is able to pass hazards with no safety margin whenever its confidence exceeds the current area’s vulnerability; this is trivially true in safe areas and often incidentally true in vulnerable areas. When this is not the case, such as in high-risk or unexplored vulnerable areas, SDD-TMPC behaves like fixed-tube TMPC, and indeed seems to prefer doing so over waiting to increase confidence whenever possible.

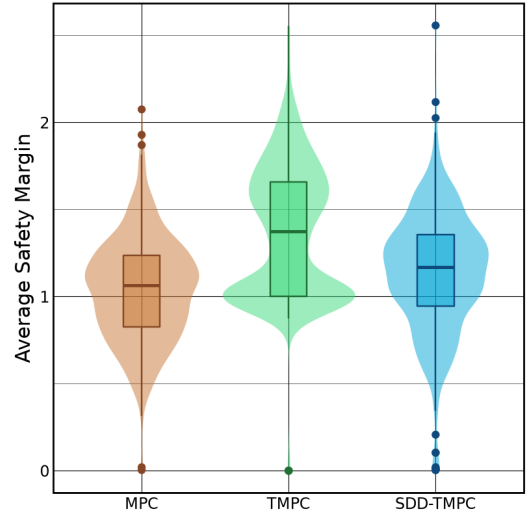


Figure 4: Average safety margin over each run, separated by controller. A clear hierarchy is present, with MPC displaying the lowest median run-average safety margin and TMPC only dropping below a run-average of one in rare adversarial edge cases. SDD-TMPC’s averages were distributed similarly to but slightly higher than MPC’s.

Fig. 4 shows the distribution of run-average safety margins by controller. As expected, TMPC maintains the largest safety margin of the three, only dropping below an average of one where the agent either spawned on top of a victim or was cornered by fire. MPC and SDD-TMPC display similarly shaped distributions, with SDD-TMPC seeming to maintain a larger safety margin than MPC on average. To evaluate the differences in safety margins and therefore conservativeness across the three controllers, a Friedman test was conducted, which revealed a statistically significant difference across the controllers ($\chi^2 = 380.1, p < 0.001$). Subsequent post-hoc Wilcoxon signed-rank tests with Bonferroni correction confirmed that TMPC maintained significantly larger safety margins than SDD-TMPC, and SDD-TMPC maintained significantly larger safety margins than pure MPC ($p < 0.001$).

Returning to the discussion of success rate and feasibility, we should also consider that not all environments are necessarily *safely* completable. Deciding this for every seed tested

would be unreasonable and likely unreliable, but we demonstrate a related effect through the MPC controller’s collision rate. While MPC suffered collisions in nearly half of total runs, this proportion increases to 51.4% in runs where TMPC timed out and even further to 72.8% when only considering runs where SDD-TMPC timed out. A multiple logistic regression was performed to predict MPC collisions based on the occurrences of timeouts for TMPC and SDD-TMPC. The overall model was highly significant ($p < 0.001$), and an SDD-TMPC timeout on a seed was found to be a large, statistically significant predictor of an MPC collision (OR = 3.54, CI = [2.09, 5.97], $p < 0.001$). This strongly supports the hypothesis that SDD-TMPC mostly encounters infeasibility and thus timeouts in map geometries which are truly dangerous, where the less conservative MPC controller attempts to continue but instead collides. In stark contrast, a TMPC timeout was not a significant predictor of an MPC collision ($p = 0.716$), suggesting that TMPC times out even in maps which are safely completable.

5 Responsible Research

This section reflects on the ethical implications of research contributing to the deployment of autonomous robotics in SaR situations, and discusses the reproducibility of this study and the methods used within.

5.1 Ethics

The deployment of autonomous systems in life-or-death scenarios such as SaR carries profound ethical responsibilities concerning safety, accountability, and the allocation of rescue resources. Because this paper uses a simplified grid-based simulation with static victims, complex ethical dilemmas regarding dynamic triage or victim prioritisation fall outside its scope.

Instead, the main ethical considerations of this study relate to system accountability and safety assurances. In SaR environments, it is critical that autonomous decisions are traceable in the event of a failure. A key advantage of model predictive control frameworks is their inherent transparency, as they rely on explicit mathematical formulations for both prediction and decision-making. This avoids the ‘black-box’ issue common in data-driven approaches, which is crucial for post-incident accountability and informed system iteration.

However, an ethical issue remains in the ‘sim-to-real’ gap. Specifically, there is a risk of unwarranted trust in autonomous controllers that have only been tested in simplified environments such as ours. Future physical deployments building upon this work should be wary of the differences between a real SaR scenario and this simulation, especially with regards to victim behaviour and sensor noise.

Finally, as this study relies entirely on simulated environments, no human participants or identifiable data were involved.

5.2 Reproducibility

To ensure that this paper’s findings are easily replicable, precise descriptions of the simulation environment, controller formulations, and experimental procedures were provided.

All simulation parameters which differ to the defaults defined in [14] have been provided in Appendix A, and all code and raw data has been made freely available at <https://github.com/FilipOBK/SDD-TMPC-Research>.

6 Conclusions

6.1 Conclusion

The main objective of this research was to evaluate how the adaptive tube of state-dependent dynamic tube-based MPC (SDD-TMPC) influences conservativeness and the safety-performance trade-off in a partially observable, dynamic search-and-rescue (SaR) environment. To this end, a grid-based SaR simulation was developed, along with MPC, TMPC, and SDD-TMPC controllers for the simulation’s agents. All three controllers were tested over 500 randomly generated search missions, where they were tasked with locating 12 victims spread across the map, and were given no prior knowledge of the environment other than the simulation dynamics. Missions were restricted to 300 steps based on prior testing, and a run was recorded as a failure if it reached this limit.

SDD-TMPC was observed to perform substantially better than fixed-tube TMPC, increasing the success rate of missions from 35.0% to 81.6%, and decreasing the average duration of mutually successful missions by an average of 13 steps. When ignoring collisions, SDD-TMPC still lagged behind pure MPC, which boasted a success rate of 97.2% and an average duration 14 steps lower than that of SDD-TMPC in runs where both succeeded. However, the MPC controller suffered collisions in just under half of runs, and only managed a success rate of 50.2% when treating collisions as ‘fatal’. Conversely, TMPC and SDD-TMPC only violated constraints a very small number of times, and only when cornered by the dynamic fire. Analysis of TMPC and SDD-TMPC failures’ power to predict MPC collisions supports the hypothesis that SDD-TMPC mostly encounters mission infeasibility in truly dangerous geometry, whereas the more conservative TMPC often fails even on safely completable maps.

6.2 Future work

Despite aiming to evaluate SDD-TMPC in an ostensibly more realistic environment, several simplifications were still made. The most impactful of these is the discrete, grid-based nature of the simulation. This required a reformulation of SDD-TMPC to a finite input set framework, removing much of the complexity that comes with handling kinetics, but crucially also required that the tube be binary. While we were still able to show a clear improvement over TMPC, a continuously varying tube is a key aspect of SDD-TMPC which allows for much more nuanced approaches to safety margins, and was not considered in this study. As mentioned in Section 5.1, future research should also involve more complex victim modelling to account for victim movement and mental state as in [17], to better ensure victim safety. Finally, further research should be performed on methods to estimate disturbance magnitude and thus tube size, and how sensor noise or other measurement inaccuracies affect this estimate with respect to safety guarantees.

A Configuration values

Table 2: Experiment configuration values

Variable	Description	Value
n_h	grid height	35
n_w	grid width	35
N_r	number of rooms	7
N_v	number of victims	12
N_f	number of fires	2
$w_{t,\min}$	min. tunnel width	1
$w_{t,\max}$	max. tunnel width	3
p_f	fire spread chance	0.04
T_f	fire duration	14
N	prediction horizon	4
r	scan radius	4
η_0	base scan accuracy	0.35
σ	confidence decay	0.05
ω_{dscv}	discovery weight	$\frac{200}{\pi r^2}$
ω_{conf}	confidence weight	$\frac{3}{n_h \cdot n_w}$
ω_{xplr}	exploration weight	$\frac{300}{n_h + n_w}$

B Generative AI Use

The generative AI tool *Google Gemini* was used in the research and creation of this report, following the guidelines set out by TUDelft. All code and academic writing was produced by the student, and the scope of generative AI use was limited to the following aggregative and editorial roles:

- Assistance finding academic research on specific topics
- Assistance debugging student-created code
- Assistance in structuring and improving clarity of report

References

- [1] H. Chitikena, F. Sanfilippo, and S. Ma, ‘Robotics in search and rescue (sar) operations: An ethical and design perspective framework for response phase,’ *Applied Sciences*, vol. 13, no. 3, 2023, ISSN: 2076-3417. DOI: 10.3390/app13031800. [Online]. Available: <https://www.mdpi.com/2076-3417/13/3/1800>.
- [2] K. Tong, Y. Hu, B. Dikic, S. Solmaz, F. Fraundorfer, and D. Watzenig, ‘Robots saving lives: A literature review about search and rescue (sar) in harsh environments,’ Jun. 2024, pp. 953–960. DOI: 10.1109/IV55156.2024.10588685.
- [3] R. V S and G. Prabhakar, ‘From preparedness to action: Synthesising insights on robot usage in post-earthquake search operations,’ *Resilient Cities and Structures*, vol. 5, no. 1, pp. 60–70, 2026, ISSN: 2772-7416. DOI: <https://doi.org/10.1016/j.rcns.2026.01.001>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2772741626000037>.
- [4] M. Baglioni and A. Jamshidnejad, ‘A novel mpc formulation for dynamic target tracking with increased area coverage for search-and-rescue robots,’ *Journal of Intelligent & Robotic Systems*, vol. 110, no. 4, p. 140, Sep. 2024, ISSN: 1573-0409. DOI: 10.1007/s10846-024-02167-3. [Online]. Available: <https://doi.org/10.1007/s10846-024-02167-3>.
- [5] F. A. de Alcantara Andrade et al., ‘Autonomous unmanned aerial vehicles in search and rescue missions using real-time cooperative model predictive control,’ *Sensors*, vol. 19, no. 19, 2019, ISSN: 1424-8220. DOI: 10.3390/s19194067. [Online]. Available: <https://www.mdpi.com/1424-8220/19/19/4067>.
- [6] F. Surma and A. Jamshidnejad, ‘State-dependent dynamic tube mpc: A novel tube mpc method with a fuzzy model of disturbances,’ *International Journal of Robust and Nonlinear Control*, 2023. eprint: arXiv: 2310.19997.
- [7] U. Rosolia, D. Guastella, G. Muscato, and F. Borrelli, ‘Model predictive control in partially observable multi-modal discrete environments,’ *IEEE Control Systems Letters*, vol. 7, Jan. 2023. DOI: 10.1109/LCSYS.2023.3284807.
- [8] K. Rado, M. Baglioni, and A. Jamshidnejad, ‘Enabling robots to autonomously search dynamic cluttered post-disaster environments,’ *Scientific Reports*, vol. 15, no. 1, p. 34 778, Oct. 2025, ISSN: 2045-2322. DOI: 10.1038/s41598-025-18573-y. [Online]. Available: <https://doi.org/10.1038/s41598-025-18573-y>.
- [9] C. Piquepal and M. Toussaint, ‘Control-tree optimization: An approach to mpc under discrete partial observability,’ in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2021, pp. 9666–9672. DOI: 10.1109/icra48506.2021.9561664. [Online]. Available: <http://dx.doi.org/10.1109/ICRA48506.2021.9561664>.
- [10] F. Surma and A. Jamshidnejad, ‘Recursive feasibility without terminal constraints via parent–child mpc architecture,’ *Results in Control and Optimization*, vol. 22, p. 100 653, 2026, ISSN: 2666-7207. DOI: <https://doi.org/10.1016/j.rico.2025.100653>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2666720725001389>.
- [11] A. Dey and S. Bhasin, *Output feedback mpc with adaptive tubes*, 2026. arXiv: 2605.23661 [eess.SY]. [Online]. Available: <https://arxiv.org/abs/2605.23661>.
- [12] Z. Liu and O. Stursberg, ‘Recursive feasibility and stability of mpc with time-varying and uncertain state constraints,’ in *2019 18th European Control Conference (ECC)*, 2019, pp. 1766–1771. DOI: 10.23919/ECC.2019.8795973.
- [13] X. Dai, Y. Liao, and Y. Chen, ‘Safe swarm navigation in constrained environments: A dynamic tube-based distributed mpc approach,’ *Drones*, vol. 10, no. 3, 2026, ISSN: 2504-446X. [Online]. Available: <https://www.mdpi.com/2504-446X/10/3/177>.

- [14] T. Schutgens, 'Coordination-aware task allocation via a lightweight supervisory layer: An event-driven approach to multi-agent search and rescue,' Bachelor's Thesis, Delft University of Technology, 2026.
- [15] D. Marczuk, *Restrictive precise angle shadowcasting*, 2022. [Online]. Available: https://roguebasin.com/index.php/Restrictive_Precise_Angle_Shadowcasting.
- [16] K. Benesch, *Python-tcod: The python port of libtcod*, 2026. [Online]. Available: <https://python-tcod.readthedocs.io/>.
- [17] M. Grouev, 'Human-robot interaction in search and rescue: Modeling victim behaviour and trust,' Bachelor's Thesis, Delft University of Technology, 2026.