# Collision Avoidance in Surface Mines

A server-based approach

H. van de Kamp

Master of Science Thesis

Delft University of Technology

Delft Center for Systems and Control

# Collision Avoidance in Surface Mines

**A server-based approach**

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

H. van de Kamp

May 25, 2020

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

The research for this thesis was performed at Vehicle Dynamics Group, which is part of the University of Pretoria. Their cooperation is hereby gratefully acknowledged.

TUDelft Delft University of Technology

DCSC

# Abstract

Due to new legislation, a lot of research is done in the development of a Collision Avoidance Systems (CAS) for surface mines. Almost all CASs studied in literature are decentralized and developed for passenger vehicles or aviation applications. That makes them unsuitable for the mining environment. A centralized system can ensure optimal control, which means scenarios like intersections can be managed much better. This results in a decreased impact on production. It also increases acceptance, which means operators will be less inclined to turn off the CAS.

For this reason, a centralized CAS for mining vehicles was developed. Literature shows that Mixed Integer Linear Programming (MILP) can create relatively accurate models that can be solved very fast. Since only linear constraints can be used, a new geometric model was formulated using four squares that are equally spaced on the longitudinal axis of the vehicle. The vehicle dynamics are described using a linear model. When the optimization problem becomes infeasible, no control is issued to the vehicle. The construction of three different operation modes for the vehicle allow the algorithm to deal with the non linear event of a collision which would otherwise result in an infeasible problem. Weight can be assigned to individual vehicles to determine which vehicle should be prioritized or to model traffic situations.

Simulations of six scenarios commonly encountered in mines, involving two vehicles, show that the system is able to avoid all collisions. When a third vehicle is added, collisions are still avoided but solution times increase exponentially. As simulation results were promising, the system was implemented on two test vehicles. Quantitative tests show that the optimization runs as fast using noisy real world data as it does using clean, artificially created data.

# Table of Contents

# List of Figures

# List of Tables

# Acknowledgements

I would like to thank my two supervisors, dr.ir. Kim Batselier from the Delft University of Technology for his long distance support and dr. Theunis R. Botha from the University of Pretoria for always having the door of his office open and a great hike. I would also like to thank prof. P.S. Els for being very approachable with questions, giving helpful feedback, and lending me his copy of the book 'Raka' by N.P. van Wyk Louw.

Furthermore I would like to thank my fellow master students in Pretoria. Andries, Megan, Monique and Wian, I had a lot of fun working with you and I can list a specific contribution to my thesis from all four of you. Glenn, without your help the networking issues would probably still be unresolved. All the other staff of VDG, thank you for helping with one or another problem. I will miss the Friday afternoon coffee at 14:00 and the socials at Capital Craft.

My family in South Africa was a great base of support and showed me what true hospitality is. Thanks for letting me spend all those amazing weekends at your houses and caring for me when I was sick. To all my friends, thanks for all those late nights at the dam, the conversations and the memories. Martijn, thank you for some great weekends in Bloemfontein and Cullinan. Last but not least, thanks to my parents for supporting me from the other side of the world and visiting me for a great vacation in Cape Town.

This truly was a journey to remember!

Delft, University of Technology H. van de Kamp
May 25, 2020

# Chapter 1

# Introduction

The mining industry has contributed 8.3% to the Gross Domestic Product (GDP) of South Africa over the last six years [8]. Since it is a significant part of the South African economy, government dedicates a lot of effort to improve safety in the mining sector. The Mine Health and Safety Council (MHSC) is the public entity that provides the minister of Mineral Resources and Energy with advice regarding safety and health legislation. Due to the fact that about 10 to 15 % of fatalities in mines are caused by vehicles, the MHSC decided to enforce employers to have an Collision Avoidance System (CAS) installed on each Trackless Mobile Machine (TMM) that is present at a mine site. The CAS should avoid collisions with other vehicles, but also detect pedestrians and prevent collisions with them. At the time of writing, the legislation is planned to come into force at the end of 2020. Since no CAS existed when the legislation was proposed first, a lot of research is conducted by industry to develop suitable systems.

## 1-1 CAS Requirements

The legal requirements of a CAS are derived from the Mine Health and Safety Act [9]. In Regulation 8.10 of the act, it is specified that each electrically and battery powered TMM must be equipped with a pedestrian detection system. If a potential collision between a pedestrian and TMM is detected, both the vehicle driver and the pedestrian must be warned of each others presence. If no action is taken to prevent the collision, the vehicle must be automatically retarded to 'a safe speed' and it is required that the 'brakes of the TMM are automatically applied without human intervention' [9, p. 185-187]. Diesel powered TMMs must be equipped with a system that detects other diesel powered TMMs. These systems must also warm the vehicle and if no action is taken, automatically apply brakes in order to slow the vehicle down.

There are three distinguishable levels of control in a CAS, as defined by Earth Moving Equipment Safety Roundtable (EMERST). [6]. Level 7 control is described as 'operator awareness'. The driver of the TMM is made aware of potential hazards in the vicinity of the vehicle. The technology is aimed at helping the driver observe and understand potential hazards by supplying information. Level 8 control is more commanding and is described as 'advisory control'. At level 8 the information is accompanied by warnings. The actions needed by the driver to prevent unsafe situations are also provided. When the driver does not follow the instructions provided by the level 8 controller, the level 9 technology can automatically intervene and take over control from the driver. This so called 'intervention control' is the last resort in a CAS and ideally the level 7 and 8 control should assist the driver in such a way that a level 9 intervention is not necessary.

Another important aspect of a CAS is the acceptance. The acceptance indicates how happy the end users are with the product. Since the product is forced on them by legislation, it is very important that the end users are happy with the implementation. A big part of that is the human-machine interface used by the operators. However, the collision detection and prevention also play a crucial role. If the system is too conservative, there will be frequent false alarms and unnecessary warnings. The unnecessary braking action performed by the CAS will also reduce mine efficiency. Since good performance by the driver is often rewarded with a bonus, this reduction in efficiency would give drivers a monetary incentive to unplug the system or look for other ways to disable the system. Furthermore, it might lead to drivers ignoring the warnings which would render the system useless.

There is a clear distinction in the goal of a CAS developed specifically for the mining environment compared to more general systems that are used for normal road vehicles. The philosophy of a general CAS is normally to mitigate the damage that occurs during a collision. In the mining sector, this is not sufficient. Firstly, because of the large equipment an accident at low speed can still be deadly. Secondly, the economic impact of a small accident can still be severe since the mine must shut down (part of) its operations for investigation. For this reason, the goal of a CAS for surface mines is to achieve 'zero harm'.

## 1-2 Research Goal and Structure

The Vehicle Dynamics Group (VDG) has developed a standardized test that is used to evaluate the performance of new CASs that are being developed by other entities [10]. The test platform consists of two Land Rover vehicles, equipped with a range of advanced sensors. A photograph of the two vehicles during one of these tests can be seen in Figure 1-1. The VDG is also interested in developing their own CAS. The first goal of this thesis is to perform an analysis of the current state of the art in CASs. Consequently, the biggest shortcomings of the current state of the art are identified. This is done in Chapter 2.

It becomes clear that the systems discussed in literature are mostly developed for either passenger vehicles or fpr the aviation industry. Due to simplification, they cannot be used in the mining industry. Another major shortcoming identified is that most CASs are decentralized. Centralized systems have some advantages over decentralized systems, for example regarding their impact on mining operations. The main goal of this thesis is to develop a centralized CAS to ensure minimal impact on the efficiency of the mine's operations while ensuring that no accidents occur.

The centralized collision avoidance problem is posed as an optimization problem. In Chapter 3 a Non-Linear Programming formulation of the problem is tested. Mitigating the damage

**Figure 1-1:** The two Land Rovers are used to test CASs for commercial entities

caused by a collision, especially a collision involving multiple vehicles, is very hard when the risk assessment is not accurate. To get a accurate model, a big part of the chapter is dedicated to deriving an accurate cost function. The collision detection system is then formulated as an optimization problem. During simulations it became clear that the Non-Linear Programming (NLP) approach is not a viable solution due to solving times and failure to find a solution in many instances.

An alternative method of formulating the problem, MILP, is described in Chapter 4. Literature shows that Mixed Integer Linear Programming (MILP) has big advantages over NLP, especially regarding computational time. The MILP formulation only supports linear equations, and simplifications made in similar algorithms for the aviation industry make those models invalid for the application studied here. This required that other linear models for both the dynamics and geometry of the vehicle must be formulated, which are discussed in Chapter 5. To avoid infeasible problems when unavoidable collisions occur, three different operation modes are developed. These modes model build in an extra safety measure and also enable the modelling of a collision, which is a highly non linear event. The modes also ensure that the problem is always feasible. The complete model, consisting of the dynamic and geometric model combined with the different modes, is then constructed using constraint modelling. This is described in Chapter 6.

To verify that the CAS is working, simulations are performed. These are discussed in Chapter 7. As the results of the simulations are promising, the CAS is implemented on the two test vehicles that VDG has at its disposal. The test setup and the quantitative results of the tests that are performed are described in Chapter 8. A conclusion and recommendations for further research are discussed in Chapter 9.

# Chapter 2

# State of the art

Legislation in South-Africa enforces that by the end of 2020 each Trackless Mobile Machine (TMM) used in a mine is equipped with a Collision Avoidance System (CAS), as can be read in Section 1-1. To get a better understanding of the shortcomings of current collision detection and management methods, it is best to study the current state of the art in CAS applications. The sensors commonly used in intelligent vehicles for localization and collision detection are examined in Section 2-1. In Section 2-2 some examples of CASs are given in order to give an overview of the current state of the art. Different models used to describe the motion of vehicles are studied in Section 2-3. In Section 2-4, different methods of risk assessment that were found in literature are listed. From this, the shortcomings of current systems are determined in Section 2-5. A conclusion is given in Section 2-6.

## 2-1 Sensors

Since the CAS should be able to prevent collisions, it is important that the vehicle can both interpret its surroundings and localize itself. Sensors commonly used for interpreting the surroundings are LIght Detection And Ranging (LIDAR), RAdio Detection And Ranging (RADAR) and video camera [11]. In [11] and [12] these sensing methods are discussed.

LIDAR uses laser beams, which are reflected from the surface of an object back to the source. The distance to the objects is calculated by measuring how long it takes for the light to be reflected back. This can form either a 2D line scan or a 3D area scan. RADAR works with the same principle, but uses radio waves which have a much lower frequency. One of the main differences between the two is that LIDAR is more accurate than RADAR. However, it is also much more expensive. An advantage of RADAR is that it can use the Doppler frequency test to measure the relative speed of moving object without additional cost. It is also less sensitive to changing light and weather conditions than LIDAR. Advances in the quality of sensors make both RADAR and LIDAR very reliable for collision detection. Equipment in the mining environment is subject to rough treatment, dust and changing light conditions. This makes the sensitive and expensive LIDAR sensor unsuitable for use in surface mines. RADAR is also not very suitable, since it is hard to get a 360° coverage using RADAR sensors. The raw data must also be interpreted using an object detection algorithm, which can be challenging to train for a new environment such as in mines.

These problems of requiring multiple sensors for a 360° coverage and object detection algorithms to analyse the raw data also occur when using video cameras. A video camera uses visible or ultraviolet light to capture images. Combining multiple cameras, a 3D image can be generated. This is called stereo vision [11]. Using a neural network and a 3D algorithm for object detection makes the method fairly slow. The changing light conditions present in mines further complicate the implementation of a reliable CAS. Infrared based systems are generally not robust enough to endure the harsh mining environment. Furthermore, the slow response time limits their effectiveness. Due to direct sunlight and hot equipment, infrared cameras are also prone to issue false alarms [12]. Ultrasound based systems are also relatively fragile and struggle to deal with vibrations caused by mining equipment. Both types of systems are thus unsuitable for our application.

The sensors studied above are mainly used to interpret the surroundings of a vehicle. They can also be used for localization. However, this requires a detailed map of the environment in which the vehicle moves around. Since mines are changing frequently, this is hard to achieve. The sensor most often used for localization is GPS or a more precise variation D-GPS. WiFi-tracking can also be used for localization but requires an infrastructural network throughout the mine. One methods that is often used to improve the precision of a localization algorithm is sensor fusion. In sensor fusion, the GPS data is combined with measurements from steering angles, wheel speed, IMU's and other sensors to correct for inaccurate or missing GPS measurements.

A fourth method of localization, Radio Frequency Identification (RFID), is discussed in [12]. With RFID systems, vehicles and personnel carry a transmitter (or 'tag') that sends out its identification code. Vehicles are additionally equipped with a receiver, and an alarm is issued if a tag is detected in the proximity of the vehicle. The authors of [12] state that RFID tags seem to be a promising future solution to some of these mentioned limitations. However, the state of the art at the time of writing did not allow for development of a cheap and reliable system. Research specifically on mining vehicles was done in [1], however results were not convincing. The authors even used phrases as 'surprisingly poor' to describe performance under certain conditions. Due to the absence of GPS, these sensors are however used in underground mines as their price is far below that of the other methods discussed above.

## 2-2 Collision Detection (Examples)

Since the first collision avoidance system called Forewarn was developed in 1955 by R. D. Olney, a lot of different methods for collision detection have been tested [13]. In [14] a vehicle based cooperative system is described that should prevent collisions at intersections. Using in-vehicle sensors, collision patterns are learned through historical data. This ensures that the system is suitable for each unique intersection. Since the paper only evaluates the system in a simulation, it remains unclear how the system would be implemented on a real vehicle.

The authors of [15] predict the trajectory of a vehicle using a Kalman filter with the positions, speed and acceleration to filter noisy GPS data, and to fill the gap between the measurements.

Due to the probabilistic uncertainty of the vehicle's real position, its position is modeled as an ellipse. To reduce computational cost, the ellipse is then described by a series of aligned circles. Based on the predicted trajectory, the risk of a collision is estimated. For this estimate it considers the Time To Collision (TTC). Because additional parameters are needed to indicate the chance of the crash happening, the number of circles overlapping (indicating severity of the crash), the total time during which circles overlap (indicating robustness of estimation) and the configuration of the crash are used. This last parameter can be very useful to see what maneuver should be made to avoid the collision. Still, the TTC is the main risk indicator used by the authors. Each vehicle receives the positions and speeds of other vehicles via Vehicle To Vehicle (V2V) communication and then perform the collision detection. Since the authors only do collision detection, no method is proposed to avoid a potential collision.

In [16], the authors also uses a cooperative system that is based on V2V communication. It is, according to the authors, one of the most extensive real world tests performed. If predicted trajectories of vehicles intersect, the TTC is calculated. The threshold on the TTC is set to the reaction time of the driver plus the minimum braking time, and if the TTC is lower than the threshold an emergency brake maneuver is needed. This equates to a level 9 action of the CAS. Level 7 and 8 warnings are also given at a fixed interval independent of vehicle characteristics. The authors note that the dynamic adaptation of the warning interval as proposed in [17] has the potential to greatly improve human perception and system acceptance. The risk of collision is calculated using the speed, direction and acceleration. In this application, a Decentralized Environmental Notification Message (DENM) is sent by a vehicle when it brakes hard. This triggers an Emergency Electronic Brake Light in vehicles in its vicinity. Since the DENM is directly transmitted to surrounding vehicles, it is one of the fastest methods of warning other vehicles. It furthermore states that a clear line of sight ensures small packet loss. Mine workers wearing tracking devices on their hard hats do not generally have a power supply that is big enough to produce the high transmission power to overcome packet loss when there is no clear line of sight. Since the system needs multiple packages to trigger a warning, the vehicles need a high transmission power to reliably send out a warning. The authors acknowledge in the conclusions that, if there are limits on transmission power, packet transmission frequency or data rates, performance deteriorates fast when there is no clear line of sight. Due to the V2V communication method, congestion of the channels can also become a serious problem.

In [18] the position of the vehicle is estimated using sensor fusion of GPS, steering angle, wheel speed and yaw rate. According to the authors, this dynamic model is significantly superior to the purely kinematic model, keeping in mind that the goal of the authors is "robustness to GPS outages and corruption, while permitting inaccuracies up to about 90 cm." The authors use a Kalman filter to estimate the position of the vehicle using the state vector $X_k = \begin{bmatrix} x & \dot{x} & y & \dot{y} & \phi \end{bmatrix}^T$ which contains the position and speed in both the West-East direction ($x$ and $\dot{x}$) and the South-North direction ($y$ and $\dot{y}$) and the heading $\phi$. The bicycle model is then used to determine the next position, using the vector with Vehicle Sensor (VS) measurements for velocity, steering angle and yaw rate $X_{VS} = \begin{bmatrix} v & \alpha & \dot{\phi} \end{bmatrix}^T$ and $X_k$. The authors tuned the filter so that the VS measurements, which are fast and accurate, are given a significant weight resulting in a delay of less than 100ms to detect lane changes and U-turns. GPS corrections are done within 500ms. The authors believe the response cannot be significantly enhanced. For collision detection, a model is used that predicts collisions based

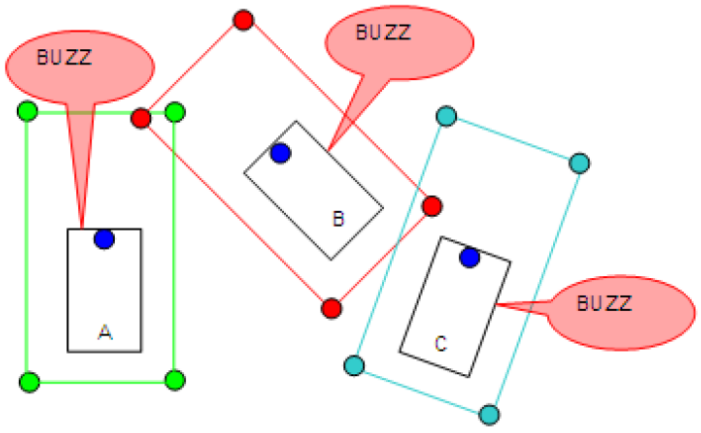


**Figure 2-1:** Schematic diagram for alarm generation in vehicles with overlapping safety zones [1].

on the assumption that a vehicle drives in a straight line. However, in turns this might trigger false alarms or interventions.

The authors of [1] use a dynamic safety zone around vehicles. The safety zone is defined using the vehicle size, direction, speed, acceleration, braking power and the driver reaction time. The vehicle transmits the four coordinates that make up this safety zone to other vehicles in its vicinity. The vehicle then constructs polygons describing its own safety zone and the safety zones of nearby vehicles. If an overlap is detected, an alarm sounds in both vehicles. A schematic describing the method can be seen in 2-1. This method also assumes that vehicles drive in a straight line. Interestingly, the authors mention how wireless broadcasting methods are cheap and suitable for centralized tracking. They do however not explain why their collision warning system is decentralized.

Most systems that are currently available are based on V2V communication. There are however also systems that take a centralized approach. One of them is described in [19]. A cost function that consists of the vehicle's deviation from its original trajectory and the sum of the control input is minimized. Constraints on the distance between the vehicles (modeled as points), constraints on the vehicle position based on a bicycle model and constraints on the input ensure that the vehicle movements are restricted and according to the laws of physics. To ensure collision avoidance, vehicles cannot be in positions in which an obstacle or other vehicles is present. Obstacle avoidance is ensured by introducing the hard constraints $z_i(t) \notin \mathcal{O}(t), \forall i, \forall t$ where $z_i(t)$ is the predicted position of vehicle $i$ at time instance $t$ and $\mathcal{O}(t)$ the area containing an obstacle at time instance $t$. To avoid that two vehicles $i$ and $j$ are in the same position, a constraint of the form $\|z_i(t) - z_j(t)\| \geq d_{safe} \quad \forall t$ is added where $d_{safe}$ is a safety margin that can be chosen by the user. The optimization algorithm used to calculate the optimal control input to the vehicle however uses a reference trajectory. In order to use a similar approach, the reference trajectory has to be determined for each vehicle. While this will be easy for some vehicles driving the same route over and over again, this can become very challenging for other vehicles. No mention is made of the computational time needed by the algorithm. It is thus hard to tell if the implementation of the algorithm had real-time capabilities.

In [20] online real-time computations are done to solve an MPC problem for collision avoidance. The problem is formulated almost identical to the one in [19], but since it is a decentralized system the model assumes that the behaviour of all surrounding vehicles stays the same for future horizon. Solving a 30 second scenario takes 4-5 seconds using a QP formulation. A time step of 4-5 seconds is however too long for a real time application. To ensure feasibility, it is necessary to take into account the out of horizon events. To ensure persistent feasibility, the authors try to ensure that the terminal states are inside the maximum control invariant set $\mathcal{C}_\infty$. A control invariant set is a set $\mathcal{C} \subseteq \mathcal{X}$ for which it holds that:

$$x(t) \in \mathcal{C} \implies \exists u(t) \in \mathcal{U} \text{ such that } f(x(t), u(t)) \in \mathcal{C}, \forall t \in \mathbb{N}_+.$$

The maximum control invariant set $\mathcal{C}_\infty$ is a control invariant set containing all sets $\mathcal{C} \in \mathcal{X}$. Since the control invariant set is a subset of the allowed set $\mathcal{X}$, the problem always remains feasible if the terminal state is in $\mathcal{X}$ [21, p.173-174]. Furthermore, the road friction coefficient is estimated using a consensus algorithm. Integrating road conditions in the model has the potential to greatly improve performance, especially if road conditions change significantly on a regular basis.

## 2-3 Motion Models

As can be read in Section 2-2, there are a lot of different methods for collision detection. One of the main features used to predict collisions is a motion prediction model. In [22], an overview is given of three different classes of motion prediction models currently used in intelligent vehicles. The paper identifies three major motion prediction models. Firstly, there is the physics-based model. In these models, motion is predicted using the laws of physics. The advantage of these models is that these models are relatively simple, but due to inaccuracies their use is limited to short-term motion prediction.

A second class of models is the maneuver-based motion model. In this model, each vehicle's future movements are predicted using a series of actions or a maneuver. Using the current behaviour of the vehicle, the system tries to match it with a maneuver. The trajectory of the vehicle is then predicted using this recognized maneuver. This class of model has a drawback in that is hard to generalize to different road layouts. Furthermore, using fully defined maneuvers, a very large database is needed to store all small variations in execution. Using probabilistic representations solves this problem, but comes at high computational cost. Furthermore, this representation cannot take the physical limitations on the vehicles into account.

The third class of motion prediction models described in the paper are the interaction-aware models. These models are very complete since all dependencies between vehicles are taken into account. Most of these systems are based on Dynamic Bayesian Networks and formulate dependencies between pairs of vehicles. However, in complex traffic situations the amount of dependencies makes the model too complex. For this reason, a common simplification to these models is to assume that each vehicle is influenced by other vehicles but does not influence other vehicles. This asymmetric dependency makes the model suitable for complex traffic situations. Some of these models also incorporate traffic rules. The interaction-aware models are very complete and more reliable than the other two classes of models. However, the computational cost is high and these models are not very well suited for real-time applications.

## 2-4 Risk Assessment

In Section 2-2 examples of different CASs are given. From this we can conclude that a lot of different methods are used to analyze the risk of a collision. In [22], an overview of existing risk assessment methods is presented. Two main classes are identified. The first one makes use of a motion prediction model, as described in Section 2-3. In this approach, a binary is often used to express whether a vehicle will be involved in a future collision or not. Using the full trajectory of the vehicle often results in complex motions equations that are hard or even impossible to solve analytically. For this reason, the trajectory is often discretized to a set of points. A simple threshold distance between the points can be used to iteratively predict a collision at each time step. To get a more accurate model, the vehicle shape can be taken into account. For computational speed, the vehicle is however often represented as a simple point. Another possibility is to detect unavoidable collisions, taking into account vehicle characteristics. As an alternative, uncertainty in the accuracy of the motion prediction model can be taken into account if the future trajectories are represented as a probability distribution. By integrating over all the different trajectories, the probability of collision can be computed. Other risk indicators such as Principal Degree of Force (PDOF), the amount of overlap between vehicle shapes [15] and velocity of the vehicles can be used to give a more accurate estimate of the collision risk.

The second class assumes that dangerous situations are primarily caused by unexpected behaviour of road users. For this reason, these methods detect unusual events and detect conflicting maneuvers. Unusual events can be recognized by manually defining nominal behaviour and detecting deviations from that. Another method is to learn the nominal behaviour from data. By putting a threshold on the probability of a certain event, unusual behaviour can be easily detected. Using the maneuver-based motion detection as described in Section 2-3, one can predict the maneuvers that a vehicle will perform in the future.

By comparing driver behaviour with the expected maneuvers learned from data, it is possible to distinguish between nominal and unusual behaviour. It is also possible to recognize dangerous maneuvers and base the risk assessment on these maneuvers.

## 2-5 Shortcomings

In Section 2-3 a description was given of the different classes of motion prediction models currently used. In Section 2-4 it is explained how these motion predictions are used to detect collisions. Some examples of the motion models, risk assessments and collisions detection methods are applied in CASs can be found in Section 2-2. In order to design an optimal CAS for a surface mine, it is important to identify shortcomings in current systems. Since almost all CASs found in literature are developed for passenger vehicles or for the aviation industry, the assumptions and simplifications used make these systems unsuitable for usage in a mining application.

One of the first shortcomings in most current systems is the V2V communication used. Decentralized control has the advantage that is does not need infrastructure to relay information to the central computer. The local computations make the system significantly faster. A decentralized system is however not capable of optimal control, leading to a loss in mine production. In addition to the revenue lost by unnecessary braking of mine trucks, it also reduces acceptance of the CAS by people who work with it. This could lead to personnel disabling the system, nullifying the potential safety benefits that the system offers to the very people who disable the system. One of the main reason why most systems are decentralized is because an infrastructure network is needed that can communicate commands to the vehicles. Since mines are much smaller than general road networks, it is much easier to set up the infrastructure needed for centralized control in a mine.

Another shortcoming is the risk assessment method used. In [15] the overlap between vehicle shapes is used as an indication of the severity of the collision. Duration of overlap is used as a measure for robustness of the prediction. However, if vehicles have a frontal collision at high speed, it might be that at one iteration only the front sides overlap while at the next iteration only the rear sides overlap. This would lead to a small risk associated with the vehicle pair, while in fact it is a very dangerous situation. A smaller time step might solve this problem, but leads to much higher computational cost. While [22] states that PDOF and speed are good additional risk estimators, little can be found about their use in current systems.

In order to speed up the algorithm, a lot of the authors use simplifications which result in systems that cannot be used in real-life applications. In [20] the author only considers multi-lane one-directional roads such as highways. While for a specific application this might be useful, these simplifications make the algorithm unsuitable for a CAS in a mining environment.

Lastly, it seems that most CASs model the vehicle as a circle. While this greatly simplifies the problem, it also leads to inaccuracies. Due to a large area next to the vehicle being covered unnecessarily, the number of false positives (an unnecessary braking command sent to the vehicle) will increase dramatically. More on this can be read in Section 5-2.

## 2-6 Conclusion

A number of different sensors are currently used to predict collisions. RARAR, LIDAR and vision based systems are used to detect and recognize obstacles in the vehicle's path. A drawback is the high cost to achieve 360° coverage. These systems also need complex algorithms to analyse the raw data, which makes them somewhat slow. Location based systems commonly use GPS and similar systems to measure the location of a vehicle. This location is then compared to the location of obstacles and other vehicles, making it a relative fast algorithm.

Almost all CASs encountered in literature are decentralized. Simplifications made in the model make them unsuitable for direct application in a mining environment. V2V communication is used to exchange information. Collision avoidance is done by braking a vehicle when a collision is predicted. This normally leads to both vehicles braking. Three types of motion models are used to predict the vehicle's future location. The motion model directly influences the risk assessment for a scenario.

Important shortcomings of current systems are the sub-optimal control due to the decentralized nature of most systems. Centralized systems do a global optimization, which becomes invalid when the scenario changes. Risk estimation is only based on the predicted position of the vehicle, while important parameters for the severity of a crash, such as PDOF, mass and speed, are disregarded. Simplifications in both vehicle behaviour and the geometric model of the vehicle make the models often unsuitable for use in a mining environment.

# Chapter 3

# Non-Linear Optimization

In Section 2-5 some of the advantages of centralized over decentralized control are listed. Because of this, the decision was made to develop a centralized Collision Avoidance System (CAS). One of the advantages is that optimal control can be achieved. In order to to this, the problem is formulated as an optimization problem. This optimization is not intended to solve one solution for all vehicles in the mine. Optimization is done for a small subsets of vehicles, that are chosen either by the distance between them or based on another algorithm that predicts likely future collisions at a relatively high speed. A logical option was to use Non-Linear Programming (NLP) to formulate an optimization problem that finds the optimal control strategy. Using a non linear formulations makes it possible to solve some of the shortcomings of the current State Of The Art (SOTA). Most importantly, simplifications that result in inaccurate representations of both the vehicle movement and geometry are not needed in a non-linear problem. An accurate representation of these two important aspects of the simulations will possibly result in a better solution to the optimization problem.

One possible drawback of NLP is the computational cost of solving the problem and the possibility of finding a local optimum instead of the global optimal solution. Since the computational cost is hard to predict, a simple implementation was written to see whether the NLP approach might be viable. In order to accurately formulate the problem using NLP, a cost function must be found that correctly represents the risks posed in a collision scenario. Since the work involves the lives of miners, an ethical reflection on using mathematics in a CAS is presented in Section 3-1. Three factors are found that together give an accurate representation of the cost of an accident. The health cost, monetary cost and logistic cost are discussed in respectively Section 3-2, 3-3 and 3-4. The cost function that combines these three components is given in Section 3-5. The implementation of the problem using NLP and a conclusion regarding the feasibility of using this method are discussed in Section 3-6.

## 3-1 Ethics

As said above, mitigating the consequences of a collision is - after avoiding collisions- the most important function of a CAS. To mitigate the consequences of a collision, a cost function has to be formulated which expresses these consequences as a function of the control input.

In [23] the ethics of using a cost function to describe collisions are investigated. The authors state that it becomes hard to make the right decision by only looking at a mathematical equation. For example, situations can occur in which a collision with a pedestrian generates a lower cost than a collision with an obstacle. This would result in the system preferring a collision with the pedestrian instead of evading the pedestrian and hitting an obstacle. The cost of the two actions (collide and evade) obscure the fact that, from an ethical point of view, the evasive maneuver always has preference over the collision. Another classic example is given in [24]. The author suggests that, purely looking at outcome, an intelligent vehicle would rather crash into a motorcyclist wearing a helmet than one not wearing a helmet since the one wearing the helmet is less likely to sustain brain damage. Since a utilitarian approach wants to maximize overall well-being, the helmeted motorcyclists consistently loses which can be considered unfair. Another consequence could be that more and more motorcyclists stop wearing a helmet, thus decreasing their overall safety.

The authors of [23] furthermore state that placing heavy cost on human life compared to monetary cost can result in poorly defined optimization problems. One of their suggestion is to consider all paths that result in a collision as infeasible. In that way no cost can override the incentive to avoid collisions. This approach is called the deontological view, where decisions are made based on the morality of the action, not on the consequences of the action.

Putting constraints on situations might seem useful in most scenarios, however there can still be situations where a collision cannot be avoided. An optimal control strategy could then mitigate the consequences of the unavoidable collision. In that case, soft constraints might be used. This prevents an infeasible problem while at the same time the constraints are minimally violated. A hierarchy can be made in the constraints by changing weights. A set of rules defining this hierarchy is proposed by the author as follows:

- A vehicle should not collide with a pedestrian (most vulnerable road user);
- A vehicle should not collide with another vehicle, except where avoiding such a collision would conflict with the first law;
- A vehicle should not collide with any other object in the environment, except where avoiding such a collision would conflict with the first or second law.

The authors also mention that humans often use traffic laws as guidelines instead of hard constraint. This can for example mean that a vehicle can drive in an adjacent lane when overtaking or to avoid an obstacle. They conclude by stating that a combination of deontological rules and consequential rules (a cost function) is a reasonable starting point in automating vehicles.

In order to realize a hierarchical approach as proposed, three different cost functions should be formulated. Since the health cost of a collision is the most important, it should be minimized first. Secondly, the monetary cost of a collision should be minimized. Lastly, a function describing the logistic cost of a control action can be used to minimize the impact of the CAS on mining operations.

## 3-2 Health Cost

In order to do a proper risk assessment, it is important to study what the health risk is as a function of the collision characteristics. One of the main measures of injury severity is the Injury Severity Score (ISS). A short explanation of the ISS is given in subsection 3-2-1. Since the parameters mass, Principal Degree of Force (PDOF) and speed are recurring in most literature, they are studied in subsection 3-2-2, 3-2-3 and 3-2-4 respectively. All these components are combined in one cost function in subsection 3-2-5.

### 3-2-1 ISS Score

In order to get a measure for injury severity, it is important that an indicator is used to describe the injury. The ISS is commonly used to estimate trauma severity and is developed by the Association for the Advancement of Automotive Medicine [25]. It rates the most serious injury to six body regions and calculates a score between 0 and 75. Severe trauma is classified as a ISS higher than 15. The ISS is one of the most widely used assessment systems. According to [26] the ISS is a very good indicator of mortality risk. Moreover it is very simple to evaluate, and thus a suitable measure for occupant injury severity.

In [27] three main views on crash severity are mentioned. Firstly, there is the transportation viewpoint that focuses on road and environmental characteristics. The crash viewpoint focuses on the vehicle to vehicle interaction, and the medical viewpoint focuses on the human body characteristics. The authors try to combine the three viewpoints to find a more complete estimate of crash injury severity. The transportation viewpoint is used to estimate the risk and characteristics of the crash by means of a simulation. The crash and medical viewpoints are exploited to estimate the crash injury severity. Some of the information used to estimate the severity of a crash are the speed difference $\Delta v$ between the vehicles involved in the crash, the energy absorption of the colliding vehicles and the PDOF and mass of the target and bullet vehicle. The bullet vehicle is the vehicle for which the main area of contact at impact is the front of the vehicle. The authors of [27] explain that there is a clear difference between crash severity (energy resulting from a crash) and crash injury severity (severity of injury to humans). For that reason, they use a regression model to estimate the ISS using the kinetic energy applied to the subject vehicle, the PDOF, use of seat belt, presence of airbags and the age of the drivers as independent variables. While calculating the ISS might seem a good idea, values obtained in the report are inaccurate due to the small sample size (82 collisions). Additionally, the data comes from a study on passenger vehicles in Australia. This data does not necessarily translate well to the injury severity in mining vehicle accidents. For these reasons, other methods should be studied to get an indication of the health cost of a collision.

### 3-2-2 Mass

The authors of [27] conclude that mass is one of the most important indicator of crash severity. The research done in [7] also seems to confirm that vehicle mass plays an important role in the mortality rate in collisions. In Table 3-1 collisions of passenger vehicles are categorized based on the object with which they collide. It then specifies what percentage of the total number of crashes and what percentage of fatalities belongs to each category. Heavy trucks

| Type of object | % of total crashes | % of total fatalities |
|---|---|---|
| Passenger vehicle | 61.2 | 31.9 |
| Light truck/van | 9.1 | 14.3 |
| Fixed object | 11.0 | 20.1 |
| Pedestrian/cyclist | 8.1 | 0.0 |
| Motorcycle | 1.3 | 1.2 |
| Heavy truck | 2.0 | 21.6 |
| Bus/train | 0.4 | 2.3 |
| Other | 6.9 | 8.6 |

**Table 3-1:** Nature of the object collided with and fatality of passenger vehicle occupants [7].

are disproportionately represented in the fatality statistics considering how many collisions fall into that category, while buses, trains and fixed objects also have a very high fatality rate compared to the number of accidents. Light trucks and vans also pose a big proportional risk, while the risk associated with passenger vehicles is far smaller. The authors give no explanation for the high number of fatalities involving motorcycles. It is highly unlikely that motorcyclists are included in the statistics, since the study focuses on fatalities among restrained passenger vehicle occupants. A possible explanation is that these collisions often occur at high speeds.

In [28] the relation between:

$$\mu = \frac{\text{Mass of heavier vehicle}}{\text{Mass of lighter vehicle}}$$

and:

$$R = \frac{\text{Probability of driver fatality in lighter vehicle}}{\text{Probability of driver fatality in heavier vehicle}}$$

is determined. They find that the mass ratio $\mu$ has a very large influence on the mortality rate. The authors find that the relation can be described by $R = \mu^u$. The values of $u$ for a host of different scenarios are given in [29]. An example used by the authors indicates that if a vehicle crashes into a vehicle that is twice as heavy, the probability that the driver of the lighter vehicle is killed, is 12 times bigger than the probability that the driver of the heavier vehicle gets killed. In mines, where interactions between large haul trucks weighting easily over 500 tonnes when fully loaded and passenger vehicles weighting well under 5 tonnes, this is very significant. This means that any significant collision between a haul truck and another vehicle will almost certainly result in severe injury or a fatality of the occupant(s) of the lighter vehicle.

The research of [28] is based on two main assumptions. The first assumption is that the crash is inelastic. The second assumption is that driver risk primarily depends on the speed difference of the two vehicles $\Delta v$. However, the authors acknowledge that this is a simplification and that for vehicles of the same mass, the severity increases with common mass. One of the main drawbacks of [28] is that it only uses the mass ratio to determine the risk. To get a better understanding of crash dynamics, it would be beneficial to consider multiple variables. Since the model is statistically accurate it is however - with some extensions- very suitable for risk assessment. It is interesting to note that the mass ratio has a far bigger influence on mortality

| PDOF | % change in Pr(ISS +15) compared to rear |
|---|---|
| Front | 192% |
| Right | 600% |
| Left | 1375% |

**Table 3-2:** Impact direction risk comparisons [2].

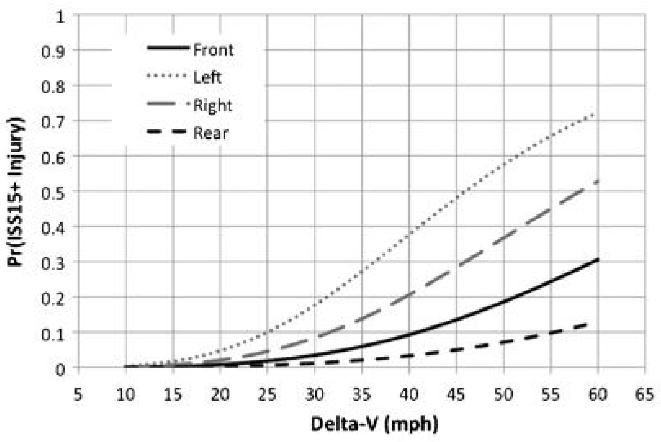


**Figure 3-1:** Predicted risk as a function of $\Delta v$ (mph) by direction of impact. For this example, other variables are fixed at: passenger vehicle, single impact, all occupants belted, no females, and no older occupants [2].

risk than on injury risk. While both are important to a CAS algorithm, the mortality risk is most important. Firstly, this is the main aim of the CAS since a fatality is always worse than an injury. Secondly, the mine will come to a complete standstill if a fatal incident occurs. Only after an investigation of the accident will the mine start up operations again. Because of this procedure, a fatal incident is much more costly than a non-fatal incident.

### 3-2-3 PDOF

The data used in [27] and [7] suggest that the PDOF does have a significant effect on the injury severity risk. However, these studies do not directly quantify this effect. In [2] a study is done that focuses on predicting the ISS for vehicle crashes. The risk of a severe injury (ISS 15+) is compared for different directions of impact. The results of their research can be seen in Table 3-2. These statistics are purely based on the PDOF. More nuanced results, which correct for $\Delta v$, can be seen in Figure 3-1. Comparing these results shows a clear discrepancy between Figure 3-1 and Table 3-2. One of the most likely explanations is that frontal collisions often occur at the highest speed difference $\Delta v$, while side impacts occur at much lower $\Delta v$. The lowest speed difference is seen in rear impacts.

The figure and table found in [2] have helped greatly in determining the injury risk as a function of the PDOF. The authors split up their data base in 4 directions, each covering 90°. That means the risk can be calculated independent from whether a vehicle is the bullet or target vehicle. However, the data is from crashes in the United States of America, where vehicles drive on the right side of the road. In the USA left side impacts are thus closest to

| Impact location | $\mathcal{N}$ | % |
|---|---|---|
| Frontal | 1205 | 37.5 |
| Near side | 248 | 38.8 |
| Far side | 335 | 11.1 |
| Rear | 112 | 3.5 |
| Roof | 132 | 4.1 |
| Other | 145 | 4.5 |
| Unknown | 16 | 0.5 |

**Table 3-3:** Location of the most severe impact to the case vehicle, showing absolute numbers and percentage of total fatalities [7].

the driver, while in South Africa right side impacts are closest to the driver. Since there are significant differences between left and right side collision, the left and right statistics must be swapped.

The researchers in [7] only look at data of fatally injured occupants of passenger vehicle crashes and disregard all non-fatal injuries. From this, they made Table 3-3 which supports the notion that lateral collisions result in a far higher number of fatalities than non-lateral collisions. It also shows that driver side collisions are more dangerous than far side collisions.

Statistical analysis done in [30] indicates that the average injury severity for lateral impact is higher than that for non-lateral impact and is another confirmation of the research mentioned above. The authors find that the ISS for lateral impacts is on average 25 compared to 20 for non-lateral impacts.

In [29] the risk of a fatality is quantified. A function of the form:

$$R = A\mu^u$$

is used, where $A$ is a parameter indicating the driver fatality risk in a vehicle with a given PDOF relative to that of a frontal impact with a vehicle of equal mass. The other parameters are described in 3-2-2. The parameters for different directions can be seen in Table 3-4. The data used by the authors only considers crashing which involve fatalities. The report indicates that there is a bigger risk of a fatality when struck from behind than when involved in a frontal collisions. This can be explained since the report takes into account rear seat passengers while also correcting for speed. In a CAS application however, the risk of a fatality would appear much smaller when a vehicle is struck from behind. This because almost all vehicles in mining environments have loading beds at the rear end of the vehicle, thus creating a big crumple zone which is not present at the front end. The $ISS_{+15}$ parameter comes from Table 3-2. The values of $A_{Kon}$ are alternative values of $A$ derived from a linearization of the curves in Figure 3-1. From [7] we get the $A_{Gre}$ parameters. However, since $\Delta v$ has not been excluded in the data used to determine $A_{Kon}$ and $A_{Gre}$, we can assume that the relative high amount of frontal collision fatalities is a result of the generally higher speed of impact experienced in frontal collisions. The influence of $\Delta v$ was also visible when comparing Table 3-2 with Figure 3-1 as mentioned earlier. The parameters are collected in Table 3-4. Because ideally the parameters should be independent of speed, the parameters $A_{Gre}$ and $A_{Kon}$ are not considered when calculating the average $A_{\Delta v}$ for all four PDOF.

| PDOF | $u$ | $A_{Eva}$ | $ISS_{+15}$ | $A_{Kon}$ | $A_{Gre}$ | $A_{\Delta v}$ |
|---|---|---|---|---|---|---|
| Far | 3.47 | 4.53 | 3.13 | 2.32 | 0.296 | 3.33 |
| Near | 3.24 | 10.05 | 7.16 | 1.74 | 1.03 | 6.32 |
| Rear | 3.71 | 1.09 | 0.52 | 0.45 | 0.093 | 0.69 |
| Frontal | 3.74 | 1 | 1 | 1 | 1 | 1 |

**Table 3-4:** Parameters for the equation describing relative risk of fatality for different PDOF.

### 3-2-4 Speed

One of the characteristics of the crash that we can predict accurately is the vehicle speed on impact. It is reasonable to assume that speed does have an impact on injury and fatality, since the kinetic energy increases with speed. In [3] a literature review is done of 11 studies describing influence of vehicle impact speed on pedestrian injury and fatality. The report makes it clear that speed does indeed have an effect on injury severity. The authors identify that most authors use a biased data set that contains an excessive amount of severe injuries and fatalities, leading to risk estimates that are too high. Some of the researchers use data that does not contain a bias, however these are very small data sets with very large confidence intervals. This means that accurate fatality risk estimates cannot be recovered from literature. However, estimates that are considered too high are available in literature. For this study overestimating the risk is not a big problem and this means we can use the estimates collected in [3]. Using the data from 9 studies analyzed in [3], a curve is made of the average risk versus speed in Vehicle To Pedestrian (V2P) collisions. The results can be seen in Figure 3-2. The data is collected by estimating data points on the different graphs in [3], and is thus not as accurate as retrieving the original data. However, some of the original data is not available and a few percentage points different should not result in significantly different outcomes in this study.

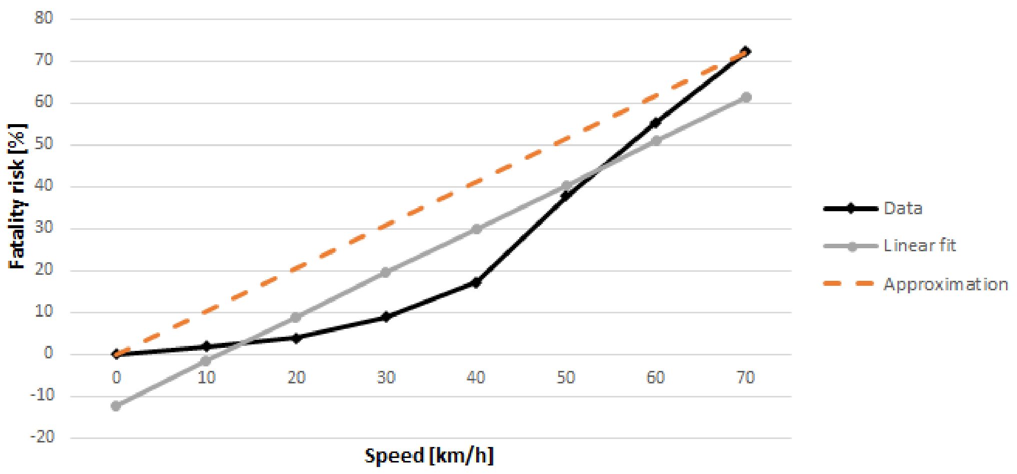


**Figure 3-2:** Average fatality risk versus speed in V2P collisions from [3] and its linear approximation.

For Vehicle To Vehicle (V2V) collisions a statistical analysis is done in [2]. In Figure 3-3 one

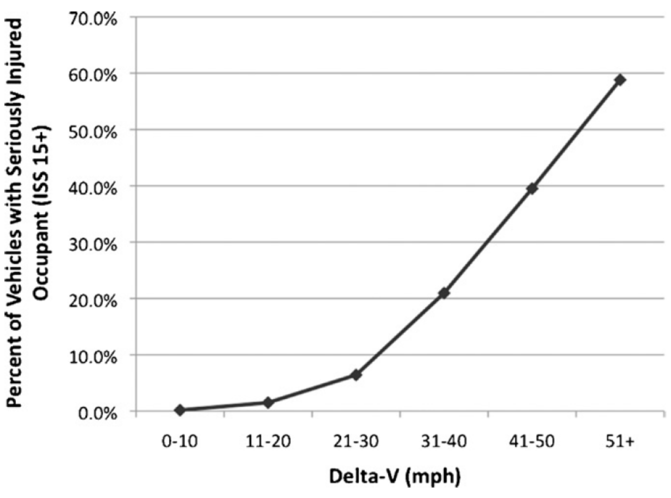


**Figure 3-3:** Proportion of vehicles containing one or more seriously injurted occupants (ISS 15+) as a function of $\Delta v$ [2].

can see that the risk of injury as a function of speed is very similar to that in V2P collisions. The injury risk in V2V collisions is slightly lower than in V2P collisions (note the mph and $\mathrm{km\,h^{-1}}$ difference). Since a higher risk assumption is not detrimental to the performance of the algorithm, the V2P injury risk will also be used for V2V collisions.

From Figure 3-2 we can see that the severity of an injury can be approximated linearly as a function of speed. Using the curve fitting toolbox from `MATLAB`, we find that the least squares fit gives the risk $R_v$ as a function of the speed $v$ as:

$$R_{\Delta v}(v) = 1.052\Delta v - 12.13.$$

However, the risk of an injury cannot be negative. Furthermore, one can see that the risk only starts to pick up at around $11\,\mathrm{km\,h^{-1}}$. One would however expect that $R_{\Delta v}(0) = 0$ and $R_{\Delta v}(v) \geq 0, \forall v > 0$. This can be achieved by simply removing the constant term. This leads to an overestimated risk, but this should not pose a problem in this study. This leaves us with an estimate of the risk of injury due to speed as:

$$R_{\Delta v} = 1.052\Delta v.$$

### 3-2-5 Health cost function

To estimate health cost for collisions, we use the fact that the severity of an injury is related to the speed, mass and direction of an impact as indicated by literature. For this we can use the power relation

$$R_i = A\mu^u R_{\Delta v}$$

found in [29] multiplied with the speed factor $R_{\Delta v}$ to indicate the mortality rate $R_i$ as a function of the parameter $A$ which represents the PDOF, the parameter $u$ which indicates the power relation between mass, mortality rate and PDOF which can be found in [29], the mass ratio between the vehicles $\mu$ and the risk from speed $R_{\Delta v}$. That the power relation $A\mu^u$ can be multiplied with $R_{\Delta v}$ is based on Figure 3-1, where it seems that the almost linear function $R_{\Delta v}$ is multiplied with a constant factor based on the PDOF.

## 3-3 Monetary Cost

In Section 3-2 a function was given that describes the health cost of a collision. As described in Section 3-1, a deontological approach is preferred. This means that separate cost functions must be defined for the health cost and monetary cost. In [31] the cost of crashes in South Africa are analyzed. In the report, the crash cost is divided in three categories. First there is the incident cost, which covers on-scene costs, the costs of towing and the congestion costs. Secondly, there is the cost of human casualty, which included all health-related costs. Thirdly, there is the vehicle damage cost. The portion of the total crash cost divided over the three categories is incident cost 15.8%, human casualty cost 9.3% and vehicle repair cost 14.9%. However, since these are only the costs of crashes happening on public roads, it is hard to translate these statistics to a surface mine situation. One option is to follow the deontological view in [23] where a hierarchical list of constraints is used, as can be read in Section 3-1. In order to make these rules, some assumptions have to be made about the cost of different types of collision. Since we look purely at monetary cost, the following hierarchy was made, ordered from highest to lowest monetary cost:

- Collisions between two mining vehicles;
- Collisions between a mining vehicle and a passenger vehicle;
- Collisions between a mining vehicle and an object;
- Collisions between two passenger vehicles;
- Collisions between a passenger vehicle and an object.

This hierarchy is based on the fact that mining vehicles contribute much more to mining operations than normal passenger vehicles. Furthermore, mining vehicles are much more expensive and it is also much harder to replace a mining vehicle than a normal passenger vehicle. At prices in the $5 million region, a collision involving a mining vehicle will generally be much more expensive than one with only normal vehicles. From the set of rules we can see that the mass of the vehicles can also be used to describe the monetary cost of the crash. Since bigger vehicles are generally more specialized and expensive than lighter vehicles, we can use the mass of the vehicle as a indicator of the cost associated with a collision involving this vehicle.

It is much harder to optimize over a set of discrete rules than over a continuous function. For this reason, the monetary cost of a collision is approximated as the sum of the vehicle masses for all $V$ vehicles involved in the collision. Since more damage is done when the collision occurs at a higher speed, this is also incorporated in the cost function. This leads to:

$$C_m = \Delta v \sum_{i=1}^{V} M_i$$

which is much easier to optimize, and also more complete than the set of rules derived earlier. However, since the masses of the vehicles do not change, this optimization will simply try to decrease the speed $\Delta v$ to 0. Additionally, it could select which vehicles are chosen to collide with each other. Since the health risk $R_i$ already is a linear function of the speed due to the term $R_v$ in the equation, the monetary cost will not add anything to the optimization algorithm. This leaves us with two options. Firstly, we can modify the monetary cost so that it optimizes over a variable that is independent of the health cost in order to get a better optimum, or secondly we can omit this part which will have the benefit of a faster algorithm which is desirable for our real-time application. Since the literature suggests no other variables that have an influence on the monetary cost of a collision, we choose to omit the monetary cost from the optimization problem.

## 3-4 Logistic Cost

The logistic cost describes how mining operations are influenced by the controller. So far no scientific papers are published on this topic, which means that this function must be designed from scratch. The logistic cost function should ensure that the vehicle always functions optimally. If for example a vehicle has to slow down to half the maximum speed to avoid a collision but brakes to a full stop, the collision might be avoided but the performance of the vehicle is sub-optimal. From this we can see that collision avoidance and optimal performance have conflicting goals. Collision avoidance forces vehicles to slow down, while for optimal performance all vehicles should be driving at maximum speed.

When a vehicle with a low maximum speed of $8\,\mathrm{km\,h^{-1}}$ slows down to $6\,\mathrm{km\,h^{-1}}$, the relative cost is much higher than when a faster vehicle driving at a maximum speed of $50\,\mathrm{km\,h^{-1}}$ slows down to $48\,\mathrm{km\,h^{-1}}$. Furthermore, vehicles with higher maximum speeds can generally accelerate faster at a lower cost. This means that the logistic cost of a vehicle should be expressed as a function of the maximum speed of that vehicle. As was stated in Section 3-3, heavy mining vehicles generally contribute much more to mining operations than normal passenger vehicles. Firstly, they do most of the production in surface mines. Additionally, heavy mining vehicles use a lot more energy and time to accelerate than lighter vehicles. This can be incorporated in the cost function by using a coefficient $M_i$ that signifies how important the vehicle is to mining operation. Using these assumptions, the cost $J$ of a braking action by vehicle $i$ over a time horizon $[0, T]$ is given by:

$$J_i = M_i \int_0^T \frac{V_{i,max} - V_i}{V_{i,max}} dt$$

where for now the payload of the vehicle is used as a value of $M_i$. Other methods of defining this parameter, for example by using the empty mass of the vehicle or empirical data, could also be used. Additionally, it is possible to generate these values artificially. In this way, an ambulance can be given a high value to give it high priority. Vehicles in priority lanes could in a similar way get a higher weight so that the weights are a reflection of the traffic situation.

## 3-5 Cost Function

If avoiding a collision is not possible, then a solution is sought that minimizes health cost, monetary cost and logistic cost. To estimate the health cost of collisions, we use the fact that the severity of an injury is related to the speed, mass and direction of impact. We use the power relation:

$$R_i = A\mu^u R_{\Delta v}$$

to indicate the mortality risk $R_i$ as a function of the parameter $A$ representing the PDOF, the mass ratio between the vehicles $\mu$, the parameter $u$ indicating the power relaxtion between mass, mortality rate and PDOF found in [28] and the risk from speed $R_{\Delta v}$. Using Figure 3-1, the severity of an injury as a function of speed is approximated linearly as:

$$R_{\Delta v} = 1.031 \Delta v.$$

Since monetary cost is linearly related to health cost this term can be omitted. For the logistic cost of control we use:

$$J_i = M_i \int_0^T \frac{V_{i,max} - V_i}{V_{i,max}} dt.$$

## 3-6 Implementation and Performance

After a first implementation of this method, it became apparent that a NLP implementation is not feasible in a real-time application. Preliminary results show that these optimization methods could not cope with the highly nonlinear optimization problem that we formulated. The main reason is that the solver took often more than 7 seconds to find a single solution. Since the problem is highly nonlinear, a multi-start approach is preferred to avoid getting stuck in a local minimum. Another problem was that the solution was far from optimal. With a simple hand calculation one could in most cases find a solution for which both the health and logistic cost were lower than the optimal solution found by the solver. This problem did persist, even when the initial guess was set as a manually calculated result that was close to the expected optimal solution.

Furthermore, the solver often returned infeasible solutions even if the initial condition was almost identical to the predicted optimal solution. This meant that the solver did not output any solution, and consequently no control command could be sent to the vehicle. This unreliability is a serious issue in our application.

## 3-7 Conclusion

Since NLP does not need any approximations, it seems like a favourable method for developing a CAS. Based on ethic considerations, a hierarchical approach would be best to formulate a cost function. Three cost were identified which were based on health, material and logistics. The health and material cost overlap, leaving only two cost term. Optimization led to very long computation times and inaccurate solutions, even when an initial guess very close to the optimal solution was given. The decision what thus made that these optimization methods are not suitable for the CAS application. A faster optimization method is Mixed Integer Linear Programming (MILP), which is discussed in Chapter 4.

The authors of [32] compare MILP and NLP with each other. The optimal solutions for both methods were very similar with a similar cost for both MILP and NLP. As the authors find that MILP is faster than NLP for every instance that they solved, MILP should be preferred over NLP. Since real time computation is one of the main features needed in the CAS application, studying [32] earlier might have guided us towards the use of MILP before implementing an NLP method. Given the more accurate model that can be created using NLP, it was however worthwhile to try it out.

# Chapter 4

# MILP

In this chapter a review of the Mixed Integer Linear Programming (MILP) method is given. The choice for MILP was made since preliminary results show that MILP might be more suitable for a real time application than, as can be read in Chapter 3. This means that most of the research described in Chapter 3 will not be directly used in the remainder of the thesis, however some insights can still be gained.

As the name suggests, Mixed Integer Linear Programming is an optimization method that makes use of a linear cost function and linear constraints. The concept of MILP has been used in aviation applications for some time now. MILP is now also being used in vehicle collision avoidance as well, for example in [19], but its main use is still in aviation applications. One of the first authors that used MILP for collision avoidance is Schouwenaars, who uses MILP to safely guide a military Unmanned Aerial Vehicle (UAV) around obstacles in [5]. An important aspect of the author's work is that he ensures the UAV has a guaranteed collision free path, independent of the environment outside the prediction horizon. A review of this and other uses of MILP in literature are given in Section 4-1.

The MILP formulation accommodates for the use of continuous, discrete and binary variables. This opens a lot of possibilities to model nonlinear problems by using binaries and simplified linearized models. The so-called 'Big M' method is a technique often used to construct sets of linear constraints that model a non-linear constraint. The mathematics behind it are discussed in Section 4-2. A method used to modify constraints during the optimization is discussed in Section 4-3. Different solvers that are available are discussed in Section 4-4.

## 4-1 MILP in Literature

A lot of knowledge can be gathered from literature about the uses of MILP. In subsection 4-1-1 an overview is given of studies in the computational speed of MILP in comparison with other techniques. Receding Horizon Control (RHC), as used in [5], is discussed in subsection 4-1-2. A method to detect and avoid collisions when using the MILP framework, is described in subsection 4-1-3. The length of the prediction horizon and the step size are important parameters of the model. An analysis on the subject is given in subsection 4-1-4.

### 4-1-1 Computation speed

In [33] it is mentioned that MILP is generally fast. The authors of [32] explicitly state that MILP is faster than Non-Linear Programming (NLP) for all instances that were tested. In [34] a comparison between different collision avoidance algorithms is done. The author finds that MILP is one of the best algorithms for smaller problems involving four to eight airplanes. For bigger problems with up to 32 airplanes, the time needed to find a solutions increases to the point where the algorithm becomes unsuitable for a real-time application. These results are promising since in a mine, it seems highly unlikely that more than 8 vehicles will be on a collision course with each other.

### 4-1-2 Receding horizon control

When MILP is used in a Collision Avoidance System (CAS) application, a common approach is to use Receding Horizon Control (RHC). With RHC one uses a prediction horizon of a certain length and calculates multiple solutions in succession instead of calculating one global solution based on a time horizon that includes the terminal state. The RHC approach does have a number of advantages over calculating a global trajectory when used in a CAS application. Firstly, computational time is very high. The RHC strategy has been bench marked against the calculation of a global trajectory as early as 2002 [4]. In Figure 4-1 one can see that the the cumulative computational time when using RHC is significantly lower than the median computational time when using a fixed horizon.

Furthermore, the global method cannot deal with changing environments. The authors of [4] test a host of scenario's with different complexity levels. They find that, for every problem that was tested, the cumulative time needed in the RHC approach was less than the time needed to calculate a single global solution. Furthermore, the global solver failed to design a trajectory for some scenarios while a solution was always found for the RHC problem. The increased performance when using RHC is promising in a path planning scenario, since the vehicle can start its trajectory earlier when a RHC approach is used. However, in a CAS application it is crucial to have control available as fast as possible. Furthermore, the changing conditions due to inaccurate prediction of the vehicle movement can render global solutions invalid after a short while. This makes the RHC approach much better than the global approach for the intended application in this thesis.

### 4-1-3 Collision detection

The authors of [35] mention that Schouwenaars [5] uses a discrete set of points at which collision avoidance is enforces and a control input is calculated. Since this set is uniformly distributed over the prediction horizon, this method is called uniform gridding. Uniform gridding is not optimal, since vehicles can still collide between time steps. To prevent this, a smaller time step can be used but this leads to an exponential increase in calculation times. For this reason, the authors of [35] propose an iterative time step selection algorithm. To enforce

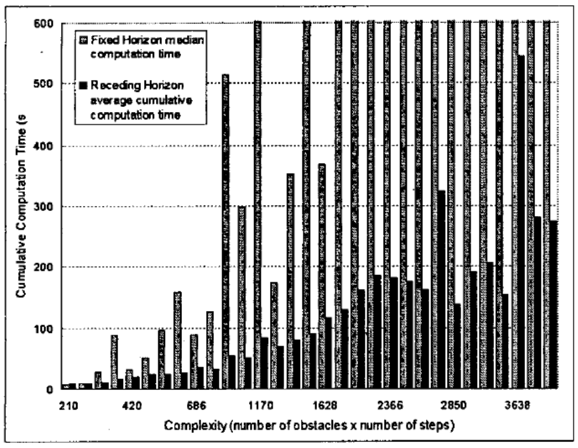


**Figure 4-1:** Cumulative Computation Time vs. Complexity [4]

collision avoidance the authors use a dense set of points with time step $\Delta t$ between points, which guarantees that each collision will be detected. To calculate optimal control, they start with an empty or very coarse set. After an optimal control sequence is calculated, they check again for collisions using the dense set with time step $\Delta t$. If a collision is detected, additional time steps are added to the coarse set and a new optimal control sequence is calculated. The process is repeated until no collisions are detected. One drawback of the method in our application is that it might have very big time steps, resulting in fast computations but unnecessary early braking action. A solution that for example avoids a collision is to brake immediately when a possible collision is detected.

Because unnecessary braking will be very problematic for a CAS's acceptance, this is one of the main areas identified in which an improvement can be made. Braking should only happen right before a collision, and besides that no action should be taken. Using the method proposed in [35], a dense set can be used to detect collisions. The time step is chosen conservatively as $\Delta t = \frac{d_{min}}{v_{max}}$ where $d_{min}$ is the width of the smallest vehicle plus an additional length to ensure that small corner collisions are also detected. $v_{max}$ is the highest maximum velocity of the vehicles. If a collision is detected, an optimal control sequence is calculated over a set of points right before the predicted time of collision. If after that a collision is still detected using the dense set, extra points are added and a new control sequence is calculated. The process is then repeated until a collision free path is found.

### 4-1-4 Step size and horizon length

A drawback of RHC is that a global optimal solution will in most cases not be found, but literature shows that the sub-optimal solution is often very close to the optimal solution, with a maximum difference of 3% in the final cost being mentioned [4]. Another problem that occurs with a finite time horizon is that the program cannot deal with the environment outside the prediction horizon window. A solution to this problem was proposed and successfully tested in [5, p. 71-92]. The author does this by ensuring that the trajectory always terminates in a feasible invariant set, meaning that the vehicle can always divert to a safe alternative trajectory if no solution is found within a certain time. In the aviation application of [5], these alternative trajectories are loiter circles where the UAV can safely keep circling until a new trajectory is found. An example of the UAV's flight path with the loiter circle for each intermediate trajectory is shown in Figure 4-2.

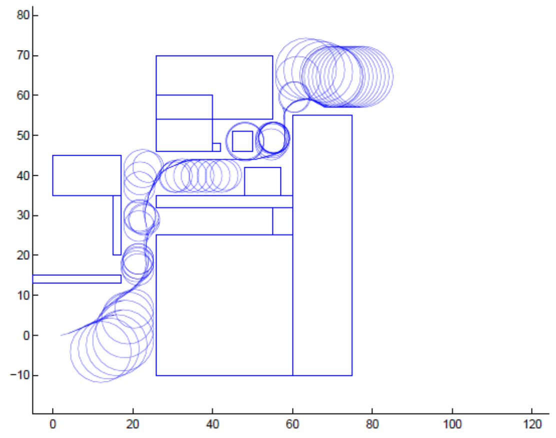


**Figure 4-2:** Sequence of intermediate receding horizon trajectories with loiter constraints consisting of 24 sample points [5].

In the CAS, this safety mechanism is however only necessary if the time horizon is too short. By taking it long enough, one can ensure that all events happening in the horizon can be dealt with in a safe manner. This however leads to a longer calculation time. It is important to keep track of the calculation time $t_c$ because it should preferably not be longer than the time step $\Delta t$ used in the simulation. If $t_c > \Delta t$, the first element of the optimal control sequence is then meant for a moment that is already in the past when the solution is returned. In that case there is no benefit of a small time step, and it can just as well be increased to reduce the computational time.

However, by increasing the time step $\Delta t$ used i one compromises on the optimality of the solution. Furthermore, a big time step gives a coarse resolution. This could lead to undersampling, where vehicles are colliding but the algorithm does not detect the collision. A possible solution is to decrease the time step to a very small size, at a very high computational cost. Another solution, that can also be seen in Figure 4-3, is to enlarge the vehicle with a safety zone. By choosing an appropriate distance $d_{safe}$, the actual collision will always be avoided.

The maximum length of the time step is thus limited by the speed and dimensions of the vehicle. To lower the computational cost, one could use a shorter horizon. However, the length of the prediction horizon cannot be shortened indefinitely. If the length of the prediction horizon is too short, a potential collision will only be detected at the last moment. Since the vehicle has a limited deceleration rate, the prediction horizon must be long enough to ensure that all vehicles can from maximum speed be brought to a stop without colliding.

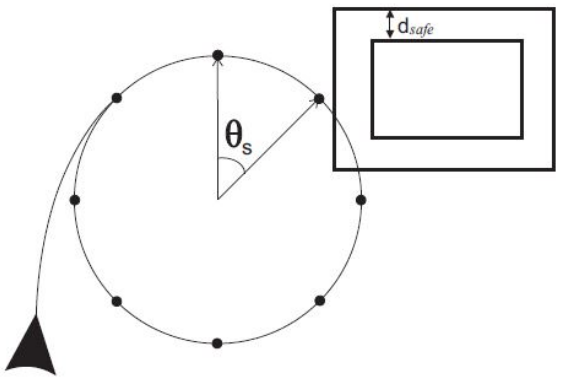


**Figure 4-3:** Undersampling leads to an undetected collision between a vehicle and obstacle [5]

## 4-2 'Big M' Method

One of the main features of MILP that is used in collision avoidance is the selective use of constraints. This is done using the so-called 'Big M' method. The method is used for both constraint relaxation and constraint modification. In this way, some non-linear constraints can be linearized without losing the non-linear properties. How this is done is illustrated in the example below.

In almost all cases encountered in literature, vehicles or UAVs are modelled as a point with a radius. To ensured that no collision occurs between vehicle $i$ an $j$, the two points have to be a certain distance $d_{safe}$ from each other. The author of [5] formulates this as:

$$\|p_i(t+k|t) - p_j(t+k|t)\| \geq d_{safe}$$

for all time instances $k$ in the prediction horizon. To simplify these constraints, one should realize that it does not matter in which direction the points are separated. Additionally, if the vehicles' distance to each other in the $x$ direction is more than $d_{safe}$, the vehicles can be less than $d_{safe}$ from each other in the $y$ direction. These constraints could thus be split up in a $x$ and $y$ direction as:

$$\begin{aligned} |x_i - x_j| &\geq d_{safe} \quad \text{OR} \\ |y_i - y_j| &\geq d_{safe}. \end{aligned}$$

This method requires that the absolute value is used, but since this is a nonlinear operator which cannot be used in the MILP framework. The so-called 'Big M' method is used to change these constraints in a form that is suitable for MILP. In the 'Big M' method, binaries $b_i$ are used to activate and deactivate constraints. The constraints are now formulated as:

$$x_i - x_j \geq d_{safe} - Mb_1 \quad \text{AND} \tag{4-1}$$

$$x_j - x_i \geq d_{safe} - Mb_2 \quad \text{AND} \tag{4-2}$$

$$y_i - y_j \geq d_{safe} - Mb_3 \quad \text{AND} \tag{4-3}$$

$$y_j - y_i \geq d_{safe} - Mb_4 \quad \text{AND} \tag{4-4}$$

$$\sum_{i=1}^{4} b_i \leq 3 \tag{4-5}$$

where $M$ is a sufficient large number so that the constraint is always satisfied if $b_i = 1$. Constraint 4-5 ensures that at least one of the binaries is zero and thus at least one constraint is satisfied. In this way, the reformulated constraint becomes a set of linear equations which is compatible with MILP.

## 4-3 Constraint Modification

Since MILP uses hard constraints, the optimization problem becomes infeasible when vehicles are too close to each other because at least one of the constraints in the set 4-1 to 4-5 will not be satisfied. The MILP method can in such a case be used for constraint modification. This can be done by using a binary $b_c$ that indicates that the vehicles are too close to each other. Constraints 4-5 is then rewritten as:

$$\sum_{i=1}^{4} b_i \quad \leq \quad 3 + b_c \tag{4-6}$$

which means that the set 4-1 to 4-4 and 4-6 are satisfied even when the vehicles are less than $d_{safe}$ from each other. In this example, the benefit would be that it prevents the problem from becoming infeasible. The application of constraint modification in this study is discussed in Chapter 6.

## 4-4 Solver

Multiple solvers are available for problems that are formulated using MILP. Since 2014 MATLAB has a dedicated MILP solver called `intlinprog`. While these and other free solvers such as GLPK are available, most authors in literature use the commercial CPLEX developed by IBM. The algorithm scores high in benchmark tests and comes with a MATLAB interface. The most common solving methods for MILP problems are branch & cut algorithms, LP relaxations and heuristics. The CPLEX solver uses a branch and cut algorithm. The basis of this algorithm is integer relaxation. This means that the algorithm solves the problem without the integer constraint on the binaries. If the values of the binaries found in the optimal solution are not integers, branches are created for both values. For each of the nodes in these branches a lower bound is set equal to the integer solution. The upper bound for each node is equal to the optimal non-integer solution of the problem. These nodes are then pruned to find the optimal solution. If in a node the value of the upper bound is then lower than the value of another existing lower bound, the node is disregarded. In the cutting part of the algorithm, feasible regions are cut away to reduce the search area. After repeating this process, eventually the optimal integer solution is found. The CPLEX algorithm uses 13 different cutting methods to find optimal ways in which to reduce the feasible area [36].

## 4-5 Conclusion

Literature states that MILP is a fast optimization method. Using RHC, it can be adapted very well to solve the centralized control problem studied in a CAS as was shown in [5]. By enlarging the vehicle with a safety zone, a bigger time step can be used. It also prevents that small corner collisions go by undetected. This leads to a lower computational cost, which allows for a longer prediction horizon. In determining the step size, one should also look at the time it takes to solve to optimization problem. If the solving time is longer than the time step, the time step should be increased.

An important technique to achieve an accurate model and effective constraint modelling is the 'Big M' method. Due to the linear nature of the method, new dynamic and geometric models of the vehicle must be formulated. These will be addressed in Chapter 5.

Chapter 5

# Linear Model

As said in Chapter 4, the Mixed Integer Linear Programming (MILP) method uses constraint modelling. This chapter covers the model that is used in the optimization problem. As MILP is used, the model must be linear. The dynamic model of the vehicle's movements is given in Section 5-1. Section 5-2 describes how the geometry of the vehicle is approximated using only linear equations.

## 5-1 Vehicle Dynamics

Since MILP only used linear equations, the vehicle dynamics must also be modelled in a linear fashion. This model is used to predict the future position of the vehicle, so that potential collisions can be detected. The vehicle's dynamic model consists of two parts. These are the forward or longitudinal and sideways or lateral movement, discussed in subsection 5-1-1 and 5-1-2 respectively. The combination of these two forms the dynamic model, which is summarized in subsection 5-1-3.

### 5-1-1 Longitudinal movement

The longitudinal movement is the only direction that is affected by the Collision Avoidance System (CAS). The velocity and acceleration of the vehicle are described in a body fixed frame. The position of the vehicle is described in a space-fixed frame [37, ch. 3]. The speed of the vehicle in global frame as a function of the lateral vehicle speed are:

$$\dot{x}(t) = v(t) \cdot cos(\theta) \tag{5-1}$$

$$\dot{y}(t) = v(t) \cdot sin(\theta) \tag{5-2}$$

respectively, where $v(t)$ is the longitudinal speed of the vehicle and $\theta$ the heading. Integrating (5-1) and 5-2 gives:

$$x(t) = x_0 + cos(\theta) \int_0^t v(t)dt \quad (5\text{-}3)$$

$$y(t) = y_0 + sin(\theta) \int_0^t v(t)dt \quad (5\text{-}4)$$

respectively. By assuming a constant vehicle speed for each time interval $[t, t + \Delta t]$, the position of the vehicle at the next step can be predicted using a left Riemann sum as:

$$x(t + \Delta t) = x(t) + \Delta t \cdot v(t) \cdot cos(\theta) \quad (5\text{-}5)$$

$$y(t + \Delta t) = y(t) + \Delta t \cdot v(t) \cdot sin(\theta) \quad (5\text{-}6)$$

where the accuracy depends on the size of the time step $\Delta t$. The initial position $(x_0, y_0)$ and heading $\theta$ are determined using measurements. These measurements will be obtained from a GPS on each vehicle.

The trapezoidal ruled is another method that can be used to estimate the position of the vehicle. The left Riemann sum is not very accurate as an approximation because of the coarse step size $\Delta t$. The assumption of constant deceleration is also more accurate than that of constant speed, which we know is not possible. However, the trapezoidal rule tends to overestimate the value when used on a concave function and underestimate the value when used on a convex function. In the CAS application, this means that the algorithm will overestimate the distance travelled when accelerating and underestimate the distance travelled when decelerating. In the first case, this will lead to false positives (unnecessary braking), while in the second case it will lead to false negatives (braking too late). Using the left Riemann sum achieves the opposite effect.

The acceleration of the vehicle is the only parameter affected by the control input $u$. The equation for the acceleration in the body fixed frame is equal to:

$$\dot{v}(t) = u(t) \quad (5\text{-}7)$$

where the control input $u(t) \in [-d_{max}, a_{max}]$. The maximum deceleration $d_{max}$ and maximum acceleration $a_{max}$ of the vehicle differ for each vehicle, to ensure that the vehicle's acceleration and deceleration in the model match real-world vehicle characteristics. Integrating (5-7) gives us:

$$v(t) = v_0 + \int_0^t u(t)dt \quad (5\text{-}8)$$

and when approximating this with a left Riemann sum this is equal to:

$$v(t + 1) = v_0 + \Delta t \cdot u(t). \quad (5\text{-}9)$$

The model thus assumes that the CAS is the only factor influencing the acceleration of the vehicle. This is not true, since the system can only brake the vehicle and the driver has to accelerate it. For this reason the control has an upper bound of $U(t) \leq a_{max}$. The algorithm in this way simulates an aggressive driving style. More on this can be found in Section 6-3.

### 5-1-2 Lateral movement

Mining vehicle seldom drive on straight trajectories. For that reason, the CAS should also work on curved trajectories. The most obvious way to achieve this is to make the heading $\theta$ a variable. In (5-5) and 5-6 the change in heading was set as a constant. By making the heading $\theta(t)$ a variable, the equations for the position in the $x$ and $y$ direction become:

$$x(t+1) = x(t) + \Delta t \cdot v(t) \cdot cos(\theta(t)) \tag{5-10}$$

$$y(t+1) = y(t) + \Delta t \cdot v(t) \cdot sin(\theta(t)) \tag{5-11}$$

respectively. The easiest way to model the heading change is by assuming that the yaw rate is known. The equation of the heading as a function of the yaw rate is given as:

$$\dot{\theta}(t) = r \tag{5-12}$$

where $r$ is the yaw angular velocity or yaw rate. Integrating (5-12) gives us:

$$\theta(t) = \theta_0 + \int_0^t r(t)dt \tag{5-13}$$

and when using a left Riemann sum, this becomes:

$$\theta(t+1) = \theta(t) + \Delta t \cdot r(t). \tag{5-14}$$

The yaw rate $r(t)$ is a function of the speed and steer angle and can be measured using a relatively cheap sensor. If the yaw rate is assumed constant, the vehicle will turn even when stationary, showing that this assumption is flawed and leads to huge deviations from the real vehicle dynamics. The steering angle $\delta$ can be derived from the measured speed and steer angle. Assuming the steering angle constant is a valid approximation when the time horizon is not too long. In [38, p. 206] it is described how the the vehicle's yaw rate $r(t)$ and steering angle $\delta$ are related to each other. This relations is given as:

$$\frac{r(t)}{\delta} = \frac{v(t)/L}{1 + \frac{Kv(t)^3}{57.3Lg}} \tag{5-15}$$

where $v(t)$ is the velocity and $L$ the wheelbase of the vehicle, $K$ the understeer gradient and $g$ the gravitational acceleration. A vehicle with neutral steer (no oversteer or understeer) has $K = 0$ and with this assumption the yaw rate $r(t)$ becomes a function of speed $V(t)$, wheelbase $L$ and steering angle $\delta$ and can be written as:

$$r(t) = \frac{v(t)}{L}\delta. \tag{5-16}$$

As said before, it is assumed that the steering angle $\delta$ is constant for the time horizon. Substituting (5-16) in (5-14), one can calculate the heading over time as:

$$\theta(t+1) = \theta(t) + \Delta t \cdot \frac{v(t)}{L}\delta \tag{5-17}$$

which is then substituted in the equations for position . These then become:

$$x(t+1) = x(t) + \Delta t \cdot v(t) \cdot cos\Big(\theta(t-1) + \Delta t \cdot \frac{v(t-1)}{L}\delta\Big) \tag{5-18}$$

$$y(t+1) = y(t) + \Delta t \cdot v(t) \cdot sin\Big(\theta(t-1) + \Delta t \cdot \frac{v(t-1)}{L}\delta\Big) \tag{5-19}$$

which is a non-linear function. Since the MILP solver can only solve models with linear equations, this approach is not suitable. A possible solution is to use a non-linear solver. The model is already highly simplified, however by using quadratic constraints the computational time will most likely increase significantly.

It is however possible to estimate the future yaw rate. By assuming that the steering angle $\delta$ and the acceleration of the vehicle stay constant until either the minimum or maximum speed is reached, the yaw rate can be predicted for the time horizon. The predicted future speed is then calculated using (5-16) and the initial acceleration $a_0$ as:

$$v_p(t) = \min\Big(v_{max}, \quad \max(0, \quad v_0 + t \cdot a_0)\Big) \tag{5-20}$$

where $v_{max}$ is the maximum speed of the vehicle. This predicted speed is then used together with the initial yaw rate $r_0$ to calculate the predicted heading for each time step as:

$$\theta(t+1) \quad = \quad \theta(t) + \Delta t \cdot \frac{v_p(t)}{L}\delta. \tag{5-21}$$

### 5-1-3 Complete model

The model is time-variant, due to the fact that the heading $\theta$ changes over time for $r_0 \neq 0$. It is however still linear since it can be written as $x(t+1) = A(t)x(t) + Bu(t)$ and thus still suitable for the MILP formulation. The set of four linear equations described above are:

$$\begin{aligned} x(t+1) &= x(t) + \Delta t \cdot v(t) \cdot cos(\theta(t)) && (5\text{-}22)\\ y(t+1) &= y(t) + \Delta t \cdot v(t) \cdot sin(\theta(t)) && (5\text{-}23)\\ \theta(t+1) &= \theta(t) + \Delta t \cdot \frac{v_p(t)}{L}\delta && (5\text{-}24)\\ v(t+1) &= v(t) + \Delta t \cdot u(t) && (5\text{-}25) \end{aligned}$$

and predict the dynamic behaviour of a vehicle. Since $\theta(t)$ is a known parameter, $cos(\theta(t))$ and $cos(\theta(t))$ are constants. This makes the system linear but time variant. The minimum speed $v_{min}$ of each vehicle is set at $0\,\mathrm{km\,h^{-1}}$. The maximum speed is based on vehicle characteristics and differs for each vehicle. When the vehicle is reversing, the absolute value of its speed is used. By setting the heading $\theta = \theta_0 + \pi$ rad (where $\theta_0$ is the measured heading) the vehicles direction of travel can be easily adapted for a reversing vehicle. The measured yaw rate is independent of the vehicle's direction and can be used directly.

## 5-2 Geometric Model

As was stated in Section 2-5, the models that are used to describe the shape of the vehicle are not adequate. Two models are commonly used in similar applications. The simplest model that is often encountered in literature uses a point with a radius to describe the vehicle, which is computationally cheap but geometrically inaccurate. A computationally expensive but geometrically accurate model uses a polygon to describe the vehicle shape. The merits of the polygon model are discussed in subsection 5-2-1. A modification of the simple point model using a number of squares, is discussed in subsection 5-2-2. A conclusion on the best model is given in subsection 5-2-3.

### 5-2-1 Polygon

As said above, the polygon represents the vehicle geometry very accurately. However, it is computationally expensive which can be a serious drawback for a real-time application such as a CAS. Some applications do use polygons to model a vehicle and detect collisions. The authors of [1] have developed a system that creates a polygon made of 4 points. These points are broadcast to all vehicles in its vicinity. By checking whether the received polygon of neighbouring vehicles overlaps with the vehicles own polygon, collision detection is done. A buzzer is activated to warn the driver if this is the case. The method does however not use a prediction algorithm to look in the future, but only checks for collisions using current information. The corners of the polygon are defined in a global frame, using the $x$ and $y$ coordinates of the centre of the vehicle. The equations for the 4 corner points are:

$$\begin{aligned}
a_x &= x + \left(\frac{L}{2} + s_f\right) sin(\theta) + \left(\frac{W}{2} + s_s\right) cos(\theta) \\
a_y &= y + \left(\frac{L}{2} + s_f\right) cos(\theta) - \left(\frac{W}{2} + s_s\right) sin(\theta) \\
b_x &= x + \left(\frac{L}{2} + s_f\right) sin(\theta) - \left(\frac{W}{2} + s_s\right) cos(\theta) \\
b_y &= y + \left(\frac{L}{2} + s_f\right) cos(\theta) + \left(\frac{W}{2} + s_s\right) sin(\theta) \\
c_x &= x - \left(\frac{L}{2} + s_b\right) sin(\theta) - \left(\frac{W}{2} + s_s\right) cos(\theta) \\
c_y &= y - \left(\frac{L}{2} + s_b\right) cos(\theta) + \left(\frac{W}{2} + s_s\right) sin(\theta) \\
d_x &= x - \left(\frac{L}{2} + s_b\right) sin(\theta) + \left(\frac{W}{2} + s_s\right) cos(\theta) \\
d_y &= y - \left(\frac{L}{2} + s_b\right) cos(\theta) - \left(\frac{W}{2} + s_s\right) sin(\theta)
\end{aligned}$$

where $\theta$ is the heading of the vehicle, where $\theta = 0$ when the vehicle travels in the direction of the positive $y$ axis, $L$ and $W$ are the length and width of the vehicle and $s_f$, $s_s$ and $s_b$ are the safety distances at the front, side and back respectively. The vehicle is surrounded by a safety zone to make the system more robust. While the safety distances can be chosen to be constant, one can also choose to make them dynamic, for example based on the speed of the vehicle or the Principal Degree of Force (PDOF). The parameters described above are visualized in Figure 5-1.

#### Polygons: Collision detection

To do collision detection, one has to compare the polygons of all vehicles involved in the crash with each other. This is commonly done by seeing whether the corner points of one polygon lie inside the other polygon and vice versa. Since the polygons are all constructed of 4 corner points, one must do four checks. A fast and simple way to check that a point $p$ is inside the convex polygon $abcd$ is making use of the cross product. The cross product of two vectors $\mathbf{ab} = (ab_x, ab_y, ab_z)$ and $\mathbf{ap} = (ap_x, ap_y, ap_z)$ is defined as:

$$\mathbf{ab} \times \mathbf{ap} = (ab_y ap_z - ab_z ap_y)\mathbf{x} + (ab_x ap_z - ab_z ap_x)\mathbf{y} + (ab_x ap_y - ab_y ap_x)\mathbf{z}$$

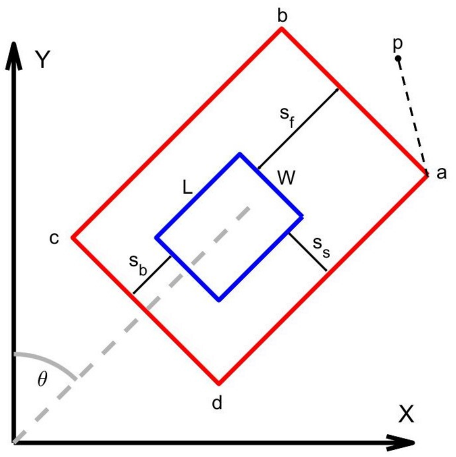


**Figure 5-1:** Diagram describing the vehicle (blue) with the safety zone (red) in a global reference frame.

where $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ are the unit vectors in each direction. When the vectors $\mathbf{ab}$ and $\mathbf{ap}$ are lying in the horizontal planes for the CAS application, it holds that $ab_z = ap_z = 0$ and the cross product can be simplified to:

$$\mathbf{ab} \times \mathbf{ap} \quad = \quad (ab_x ap_y - ab_y ap_x)\mathbf{z}$$

which is a vector parallel to the $z$ axis. Since it is an right-handed orthonormal basis, the sign of $ab_x ap_y - ab_y ap_x$ will change depending on whether the point $p$ lies left or right of the line $ab$. If the edges of the polygon are travelled counterclockwise, the point is inside the polygon if it is to the left of every edge of the polygon. In Figure 5-1 the point $p$ lies right of the edge $ab$ and is thus outside of the polygon. The cross product $\mathbf{ab} \times \mathbf{ap}$ will thus be positive. To ensure that a point $p$ is not inside the polygon $abcd$, the constraints:

$$\begin{aligned}
(x_b - x_a)(y_p - y_a) - (y_b - y_a)(x_p - x_a) > 0 \quad & \text{AND} \\
(x_c - x_b)(y_p - y_b) - (y_c - y_b)(x_p - x_b) > 0 \quad & \text{AND} \\
(x_d - x_c)(y_p - y_c) - (y_d - y_c)(x_p - x_c) > 0 \quad & \text{AND} \\
(x_a - x_d)(y_p - y_d) - (y_a - y_d)(x_p - x_d) > 0 \quad &
\end{aligned}$$

must all be satisfied. To confirm that a vehicle is not colliding with another vehicle, $4 \times 4 = 16$ constraints must be checked. Since this must also be done vice versa, 32 constraints are needed at each time instance to ensure collision avoidance between vehicles, meaning a significant computational burden compared with the simple point method used in [5].

The equations are also non-linear and thus not suitable for a MILP formulation. In [5] the author uses a linear set of equations to achieve obstacle avoidance between a point (the vehicle) and polygon (an obstacle). The obstacle is modeled using a convex polygon. Obstacle avoidance is then done by checking that the vehicle is in one of the outer halfspaces that are formed by the edges of the polygon. A halfspace $i$ is described by:

$$u_i x(t) + v_i y(t) + w_i z(t) + h_i \leq 0$$

with $x(t)$, $y(t)$ and $z(t)$ the position of the vehicle at time step $t$. Using the 'Big M' method described in Section 4-2, the different constraints are combined and for each time step $t$ a set of constraints:

$$\begin{aligned} u_1x(t) + v_1y(t) + w_1z(t) + h_1 \leq Mb_1 \quad &\text{AND} \\ u_2x(t) + v_2y(t) + w_2z(t) + h_2 \leq Mb_2 \quad &\text{AND} \\ &\vdots \\ u_Nx(t) + v_Ny(t) + w_Iz(t) + h_N \leq Mb_N \quad &\text{AND} \\ \sum_{i=1}^{N} b_i \leq N - 1 \end{aligned}$$

is formulated to ensure collision avoidance, where $N$ is the number of edges of the convex polygon. The last constraint ensures that at least one of the constraints is satisfied. Since the airplane in [5] is modelled as a point, this is an efficient way to put hard constraints on collision avoidance when an MILP formulation is used. However, when the vehicle is modelled as a polygon, computational cost is similar to using the cross product as described above. Other methods used to check for point-polygon collisions can be found in [39, p. 201-203]. According to the author, these checks can at best be formulated as an $\mathcal{O}(n)$ problem.

### 5-2-2 Squares

The point model is often encountered in aviation applications such as [5]. One of the biggest advantages of the point model is that only one calculation is required. By calculating the 2-norm distance between the centers of two points and comparing this with the sum of the radii, one can perform collision detection. For an MILP application the 1-norm can be used, which must be evaluated in the $x$ and $y$ direction. In that case the vehicle is effectively modelled as a square with the dimensions of the radius of the circle, as demonstrated by Schouwenaars [5]. Since the distance between airplanes should be orders of magnitude bigger than the vehicle dimensions [40], this is a reasonable approach even though the circle or square covers an area much larger than the airplane.

However, with mining vehicles this method cannot be used since 2-lane roads in mines are commonly about 3.5 times the width of the widest vehicle driving on it [41, p. 55]. When big haul trucks with a length of 20 meters and a width of 10 meter are modelled using the point model, a radius of over 22 meters is required to cover the whole vehicle. This means that, if the road width is indeed 35 meters, the two vehicles can never pass each without a collision being detected or the vehicle appearing to swerve off the road. Making the radius smaller is not an option, since the vehicles can then collide without the CAS detecting a collision. This means that using the point model would significantly increase the number of false positives (a level 9 intervention while the situation is in fact safe), making efficient operation of a mine almost impossible.

A new model studied in this thesis is inspired by the computational advantage of the point model. The rectangular shape of most vehicles can be approached by using a string of squares. By placing equal squares on the longitudinal axis of the vehicle, one can reduce the covered area significantly. The vehicle is in this way still fully covered by the squares, but a significant smaller area next to the vehicle is covered. In Figure 5-2 this is visualized for vehicles in two orientations, where one vehicle's heading is parallel with the $y$ axis and one vehicle is at a 45° angle with the $y$ axis. The figure also illustrates that the area covered by the squares changes for different orientations of the vehicle.

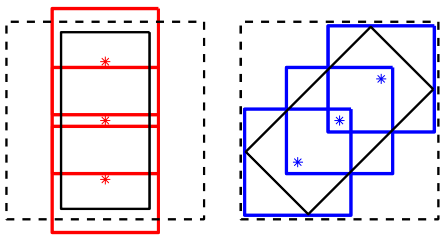
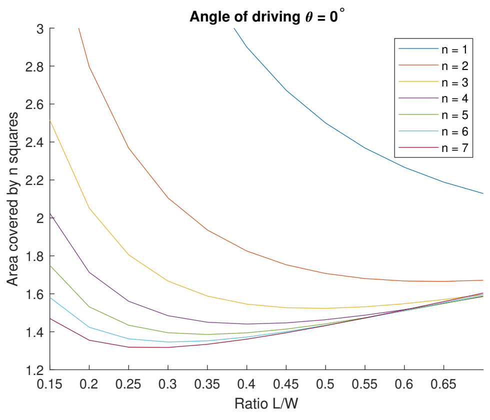


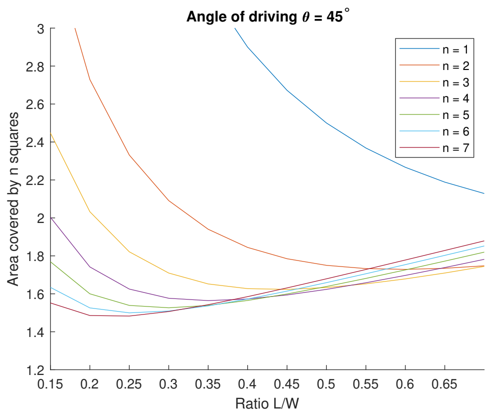
**Figure 5-2:** The geometric vehicle model using three squares (coloured lines) compared to the actual vehicles (solid black line) and a model using only one square (dashed black line).

Two options are available when using the multiple points model. Firstly, a set amount of points can be used. The other option is to change the number of points depending on the shape and/or orientation of the vehicle. In both cases, the points are evenly spaced over the longitudinal axis. If for example three points are used, one point is at the center of the vehicle while the other two points are at $\frac{1}{4}^{th}$ and $\frac{3}{4}^{th}$ of the length. The most important criteria to base this decision on is computational speed, while it should be kept in mind that the vehicle should be fully covered by the points, while a minimal area next to the vehicle should be covered.

For $n$ points used, the area covered by the points $A_n$ is a function of the width $w$ and length $l$ of the vehicle. It is calculated as the sum of the $n$ squares with dimension and subtracting the overlapping areas $(n-1)A_{ov}$ from it. This area $A_{ov}$ depends on the orientation of the vehicle, as can be seen in Figure 5-2. For different width to length ratio's the fraction $\frac{A_n}{l \cdot w}$ was calculated. The two most extreme orientations ($\theta = 0°$ and $\theta = 45°$) are visualized in Figure 5-2, for the case where 3 points are used. These two orientations are also used to calculate the area covered by $n$ squares, in Figure 5-3a when the vehicle is lined up with the $y$ axis and in Figure 5-3b when the vehicle is at a 45° angle with the $y$ axis. In Figure 5-2 one can see that the area that is unnecessarily covered at the sides of the vehicle by the circles is significantly reduced when more squares are used. Using as example a big truck with a length of 20 meters and a width of 10 meter, the width of the squares is 22.9 meter using a single square. Adding a second square reduces the width to 14.3 meters, where a third square reduces it to 12.0 meters. The dimension of the squares for up to 7 squares is given in Table 5-1, where it becomes clear that the reduction in width becomes smaller with an increasing number of squares. As said before, roads are generally 3.5 times the width of the



**(a)** $\theta = 0\,\mathrm{deg}$



**(b)** $\theta = 45\,\mathrm{deg}$

**Figure 5-3:** Area vs. number of squares $n$ for different angles of driving $\theta$

widest vehicle driving on it [41, p. 55]. This means that the width of the squares can be at most 1.75 the width of the vehicle. Using two squares already fulfills this requirement.

| #squares | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $W_s$ [m] | 22.9 | 14.3 | 12.0 | 11.2 | 10.7 | 10.5 | 10.3 |

**Table 5-1:** Width of squares $W_s$ for $n$ squares for a vehicle with width $W = 10$m

Another significant area that is unnecessarily covered is at the front and back of the vehicles. This effectively offers an extra safety buffer at the front and back of the vehicle. This is useful in a CAS since in almost every collision there is at least one target vehicle. A target vehicle is a vehicle for which the initial point of contact with the other vehicle is either the front for a vehicle driving forward or rear for a vehicle reversing.

In Table 5-2 one can see that the ratio $\frac{w}{l}$ is between 0.25 and 0.5 for most mining vehicles. This does not take into account the safety area in front of the vehicle, which is needed to account for both GPS inaccuracy and unexpected manoeuvres. In Chapter 7 one can read that especially in the dove tailing scenario this safety zone must be of considerable length. This means that the model length of the vehicle increases significantly, so that the ratio $\frac{w}{l}$ for most vehicles will be even lower. From Figure 5-3 it becomes clear that in these cases an increase in the number of squares used will lead to a much smaller area being covered and an expected decrease in false positives.

Combining Figure 5-3 with Table 5-1, it seems that using a fixed number of four points is the best trade-off between computational cost and accuracy. To distribute the four points evenly over the vehicle, one point is places at the center of the vehicle, two points are placed in front of the center point and one at the rear of the vehicle. The dynamic model is used to determine the position of the center points, after which the other three points are added in their respective location. More on the location of these points can be read in Section 6-2. Analysing the effect of using a different number of points, both on collision avoidance and computational cost, can make the trade-off between these two performance indicators more clear and might lead to other choices.

| Vehicle | Length $l$ [m] | Width $w$ [m] | $w/l$ |
|---|---|---|---|
| LF-10 Load haul dumper | 9.7 | 2.6 | 0.27 |
| Cat 140H grader | 8.7 | 2.5 | 0.28 |
| CAT 225GC Excavator | 11.2 | 3.2 | 0.29 |
| Bell B30E truck | 10.0 | 3.3 | 0.33 |
| Toyota Hilux Pickup | 4.7 | 1.6 | 0.34 |
| Volvo L350F Loader | 9.6 | 3.6 | 0.38 |
| Belaz 75710 Haul truck | 20.6 | 9.9 | 0.48 |
| Cat 797F Haul truck | 15.1 | 9.5 | 0.63 |
| Cat D11T Dozer | 6.2 | 8.6 | 1.39 |

**Table 5-2:** Ratio $w/l$ for different vehicles

#### Squares: Collision detection

The multiple points model uses the same method for collision detection as the single point model. At each time instance $t$ the distance between points on different vehicles is compared with the width $d$ of the squares. If the distance between two points $i$ and $j$ is bigger than $d_i + d_j$ one knows that the squares are overlapping and the vehicles have collided. The distance $d_{veh}$ is related to the dimensions of the vehicle and the number of squares $n$, and is equal to:

$$d_{veh} = \sqrt{\frac{L}{2n}^2 + \frac{W}{2}^2} \tag{5-26}$$

for a vehicle of length $L$ and width $W$. By giving the squares this width $d_{veh}$, one ensures that that vehicles are fully covered by the squares formed around the $n$ points. For two vehicle $i$ and $j$, collision detection is done by checking that:

$$\begin{aligned} |x_i(t) - x_j(t)| &> d_{veh}(i) + d_{veh}(j) \quad \text{OR} \\ |y_i(t) - y_j(t)| &> d_{veh}(i) + d_{veh}(j) \end{aligned}$$

and if both of the constraints are not satisfied, the vehicles have collided. This method for collision detection is very simple, although a number of checks must be done since all points on vehicle $i$ must compared to all points on vehicle $j$. One can also use predicted future positions of the vehicle to check for future collisions.

### 5-2-3 Best model

To test whether a point is inside a polygon is a $\mathcal{O}(n)$ problem [39, p. 201-203], where $n$ is the number of sides used to construct the polygon. A more detailed description of the collision detection methods can be found in subsection 5-2-1. Normally $n = 4$, as the vehicle is accurately modelled by a rectangle. If there are $V$ vehicles involved in the potential collision, there are $V$ polygons each consisting of $n$ points. As each point of each vehicle must be tested against all other vehicles, there are a total of $V \cdot n \cdot (V - 1)$ tests of $\mathcal{O}(n)$, making it an $\mathcal{O}(V^2 n^2)$ problems where $n$ is the number of circles used and $V$ the number of vehicles involved in a potential collision.

In order to do collision detection with the multiple points model, each point on a vehicle must be compared to all point of the other vehicles involved in the potential collision. This means that $n^2V\frac{(V-1)}{2}$ checks must be done for all $V$ vehicles. The computational cost of the circle model thus increases quadratically with the number of vehicles $V$, where $n = 4$ is a good choice as can be read in subsection 5-2-2.

This makes collision detection for both the multiple points and polygon model $\mathcal{O}(V^2n^2)$ problems. The computational cost is thus equal for both models. As said earlier, the polygon model has a more accurate geometric model. However, using the multiple points model, it is easier to analyse which vehicle is the bullet vehicle. This can be done by analysing the binaries associated with the frontal square. This is essential especially with dovetailing scenarios.

## 5-3 Conclusion

The MILP formulation only supports linear equations, which requires the use of a linear dynamic model in the CAS. The dynamics of the vehicle can be split in lateral and longitudinal movement. Combining these two, there are four equations that together construct the dynamic model. These can be found in subsection 5-1-3.

Geometric models that are used in the current State Of The Art (SOTA), often modelling the vehicle as a single point, cannot be used in a CAS for surface mines. This required the construction of a new geometric model using linear equations. When using polygons, the representation is very accurate, but computationally expensive. Using the multiple points model, a good approximation of the geometry can be achieved. A total of four squares give a sufficiently accurate model.

Chapter 6

# CAS Formulation

The linear model describing vehicle dynamics and geometry are described in Section 5-1 and Section 5-2 respectively. In this chapter these models are translated to linear constraints that are suitable for the Mixed Integer Linear Programming (MILP) implementation. In subsection 5-2-2 a method for collision detection for the preferred geometric model is described. This collision detection method must now be used to realize collision avoidance. This is described in Section 6-1. Since a standard MILP approach cannot deal with unavoidable collisions, a reformulation of the collision avoidance constraints is discussed in Section 6-2. The control input to the vehicle is the solution of the optimization problem and is described in Section 6-3. The cost function used in the optimization problem is discussed in Section 6-4.

## 6-1 Collision Avoidance

Collision avoidance can be split in two objectives. One is the avoidance of collisions with other vehicles and pedestrians, while the other one is obstacle avoidance. While obstacle avoidance in not unimportant, it is not the main scope of this research. This is mainly because legislation is aimed at reducing the number of fatal accidents involving vehicles. These accidents are almost without exception caused by collisions between vehicles or between a vehicle and a pedestrian. Obstacles could also be modelled as stationary vehicles, thus making it easy to extend the possibilities of the Collision Avoidance System (CAS) once other vehicles are successfully avoided. While some interesting methods are proposed in [5] to convexify non-convex obstacles, these will not be discussed since the focus is only on collisions involving vehicles and possibly pedestrians. In this study, pedestrians are modelled as stationary vehicles. This results in a single type of collisions, where only vehicles are involved.

The method used in literature to ensure collision detection between vehicles for the multiple point model is described in Section 5-2. Using a MILP implementation, this collision detection method can be adapted easily to be used for collision avoidance. The MILP model is constructed using constraints. As seen in subsection 5-2-2, collision avoidance between vehicle $i$ and vehicle $j$ at time $t$ is achieved when:

$$\begin{aligned} |x_i(t) - x_j(t)| &\leq d_{veh}(i) + d_{veh}(j) \quad \text{OR} \\ |y_i(t) - y_j(t)| &\leq d_{veh}(i) + d_{veh}(j) \end{aligned}$$

where the distance $d_{veh}$ is based on the vehicle dimensions and according to (5-26). The 'Big M' method that is used to reformulate these collision avoidance constraints as a set of linear equations, as described in Section 4-2. The absolute term is replaced by two constraints, making a total of four constraints. The safety constraints become:

$$\begin{aligned} \forall t, \forall n \in i,j \qquad x_i(t) - x_j(t) &\geq d_{veh}(i) + d_{veh}(j) - Mb_1(t) \quad \text{AND} && (6\text{-}1) \\ x_j(t) - x_i(t) &\geq d_{veh}(i) + d_{veh}(j) - Mb_2(t) \quad \text{AND} && (6\text{-}2) \\ y_i(t) - y_j(t) &\geq d_{veh}(i) + d_{veh}(j) - Mb_3(t) \quad \text{AND} && (6\text{-}3) \\ y_j(t) - y_i(t) &\geq d_{veh}(i) + d_{veh}(j) - Mb_4(t) \quad \text{AND} && (6\text{-}4) \\ \sum_{i=1}^{4} b_i(t) &\leq 3 && (6\text{-}5) \end{aligned}$$

where $M$ is a sufficient large number so that the constraint is always satisfied when $b = 1$. The distance $d_{veh}(i) + d_{veh}(j)$ ensure that the model is suitable for any vehicle independent of its dimensions, while the last constraint ensures that at least one of the constraints is satisfied.

## 6-2 Unavoidable Collisions

A major obstacle in using the MILP method is that the collision avoidance is programmed using hard constraints. This means that the problem becomes infeasible when a collision is unavoidable, in which case the solver will return an error. In similar applications, the safety zone around (aerial) vehicles could be enlarged so much that this problem never occurred [5]. Furthermore, a number of other control commands such as accelerating and steering were available, making it much easier to find a feasible solution to the optimization problem. However, in a mining environment unavoidable collision can occur for a number of reasons. One possible scenario involves two vehicles that are passing each other. Close to each other one or both of vehicles swerve in the direction of the other vehicle. These unexpected manoeuvres are hard to anticipate since these do not happen often. It would thus be excessive to slow vehicles down in every passing scenario. Another scenario where unavoidable collision can occur is when GPS drift occurs. Due to noise on the GPS data it might appear that vehicles are safely driving past each other, which does not trigger the CAS to send a braking command. When correct GPS data is then received, it might be too late to avoid the collision and thus only damage mitigation can be done.

Because an infeasible problem does not have a solution, no control command will be issued to the vehicle if these situations occur. One of the most important tasks of the CAS is however to mitigate the consequences of a collision. This means that, if a collision cannot be avoided, the vehicles should be issued with control commands that ensures minimal damage. If this is not done, the CAS will not be functioning at the moments it is needed most. It is thus important that a system is developed that tries to avoid collisions while also being able to deal with unavoidable collisions.

In [42] the authors use a so called linear time-invariant maneuver automaton (LTI-MA). The LTI-MA is used to describe a host of agile maneuvers of an Unmanned Aerial Vehicle (UAV) that are particularly hard to describe using a single linear time-invariant (LTI) model. The authors use binaries to switch from one mode to another. While these maneuvers have no relevance in a mining environment, the LTI-MA framework is very well suited to describe the non-linear event of a collision.

In Figure 6-1 the LTI-MA as used in the CAS is represented in a graph. The transition from one mode to another is dependent on the distance between the vehicles. If the distance is bigger than a specific distance $d_{safe}$, the vehicle is in the 'cruise' mode. If the distance is between $d_{safe}$ and 0 meter, the vehicle is in the 'close' mode where a penalty applies due to the proximity of the vehicle. If the vehicles are colliding, the vehicle goes to the 'stop' mode. In this mode, the vehicle will come to a stop in one time step and stay there for the remainder of the time horizon. A heavy cost is also associated with the 'stop' mode.

The vehicle has to go from the 'cruise' mode through the 'close' mode to the 'stop' mode. This was done after simulations showed that in some instances the vehicle would use the model in an unexpected way. The vehicle will come to a complete stop in one time step when entering the 'close' mode. It is however not forced to stay there when the vehicles have not actually collided. The cost of entering the 'stop' mode briefly and driving on afterwards led to a lower cost than braking for a few time steps, even though the cost of the 'stop' mode was orders higher than that of the 'close' mode.

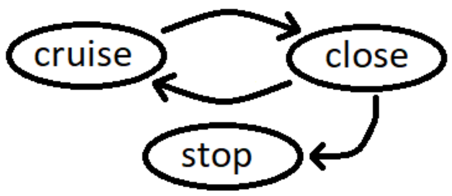


**Figure 6-1:** The LTI-MA used in the CAS

### 6-2-1 'Stop' mode

The 'stop' mode is activated when the vehicles overlap. To detect this activation, the binary $B_{stop}$ is used. By modifying (6-5) to become:

$$\sum_{i=1}^{4} b_i \leq \quad 3 + B_{stop} \tag{6-6}$$

no constraints have to be satisfied if $B_{stop} = 1$. By adding a high enough penalty for reaching this mode, $b_{stop}$ will never be 1 if that can be avoided. When the 'stop' mode is activated, the vehicle must be brought to a stop immediately. A few equations must be modified to ensure that this is possible. Firstly, the maximum deceleration must be increased. This can be done by setting the constraint on acceleration to:

$$-D_{min}(1 - B_{stop}(t+1)) - B_{stop}(t+1)\frac{V_{max}}{\Delta t} \leq u(t) \leq A_{max}$$

such that the vehicle is allowed to stop in one time step. By modifying the constraint on the maximum speed to become

$$0 \leq V(t) \leq V_{max}(1 - B_{stop}(t))$$

one can ensures that the vehicle has stopped at the moment that the constraints cannot be satisfied and $B_{stop}(t) = 1$. Since the vehicles will be overlapping and the vehicles will not be moving due to the constraint on speed, the vehicle will also stay in the 'stop' mode for the remainder of the time horizon. This effectively makes it a terminal state which is perfect for our application.

### 6-2-2 'Close' mode

To ensure some robustness, it is common practice to set up a safety zone around the vehicle. In this way, the system can be used even if GPS measurements are inaccurate. This is visualized earlier in Figure 5-1. The safety zones can however overlap, for example because of jumps caused by bed GPS reception. When the safety zones of vehicles are overlapping, it does however not necessarily mean that a collision has occurred. For this reason, the 'close' mode is added.

By using a binary, one can detect when the vehicle's safety zones are overlapping but have not yet collided, and subsequently switch to the 'close' mode. In the 'close' mode, the safety zone around the vehicle must be decreased in size. The close mode is implemented using a binary $B_{close}$, where $B_{close} = 1$ indicates that the safety zones of vehicles intersect. Since the 'close' mode must be avoided if possible, a cost is associated with entering this mode.

#### Distance constraints

As can be read in Section 4-3, binaries can be used to change constraints. Changing the safety zone around the vehicle is done by changing (6-1) to (6-4) using the $B_{close}$ binary. The new equations become:

$$\begin{aligned}
\forall t, \forall n \in i,j \qquad x_i(t) - x_j(t) &\geq (1 - B_{close}(t))(d_{safe}(i) + d_{safe}(j)) + \\
&\qquad d_{veh}(i) + d_{veh}(j) - Mb_1(t) && \text{(6-7)}\\
x_j(t) - x_i(t) &\geq (1 - B_{close}(t))(d_{safe}(i) + d_{safe}(j)) + \\
&\qquad d_{veh}(i) + d_{veh}(j) - Mb_2(t) && \text{(6-8)}\\
y_i(t) - y_j(t) &\geq (1 - B_{close}(t))(d_{safe}(i) + d_{safe}(j)) + \\
&\qquad d_{veh}(i) + d_{veh}(j) - Mb_3(t) && \text{(6-9)}\\
y_j(t) - y_i(t) &\geq (1 - B_{close}(t))(d_{safe}(i) + d_{safe}(j)) + \\
&\qquad d_{veh}(i) + d_{veh}(j) - Mb_4(t) && \text{(6-10)}
\end{aligned}$$

and by ensuring that the cost for entering the 'close' mode is smaller than for entering the 'stop' mode, the algorithm will have a preference for the 'cruise mode', followed by the 'close' mode and lastly the 'stop' mode. The distance $d_{veh}$ is given in (5-26) and the distance that is added due to the safety zone is given by:

$$d_{safe} = \sqrt{\frac{(L + s_r + s_f)}{2n}^2 + \frac{(W + 2s_s)}{2}^2} - d_{veh} \tag{6-11}$$

where the parameters can be found in (5-1). The parameter $M$ is big enough to ensure that the constraint is always satisfied when $b_i = 1$.

### Speed constraint

Since a dangerous situation is present when two vehicles are this close to each other, it would be wise to set a lower maximum speed for all vehicles when this happens. To achieve this, (6-6) could be modified to become:

$$0 \leq V(t) \leq (1 - B_{stop}(t) - B_{close}(t)) \cdot V_{max} + B_{close}(t) \cdot V_{safety}(v)$$

so that the upper bound on the speed $V$ at time $t$ is linked to the different modes. During simulations it however became clear that this approach leads to unwanted side effects. This became especially clear in the dove tailing scenario. When a vehicle is about to crash in the rear of a vehicle, the vehicles will enter the 'close' mode before colliding. However, if a low minimum speed is enforced in the 'close' mode, it could happen that the front vehicle also suddenly brakes maximally. When this happens and the front vehicle can decelerate faster than the rear vehicle, the speed difference would increase leading to a more severe collision. Simulations show that collisions, which could be prevented when only the rear vehicle decelerates, do occur when a speed limit is implemented for the 'close' mode. For this reason the unmodified constraint found in (6-6) is kept.

### Positions of points

Since the four squares are equally spaced over the vehicle, they are located at $\frac{1}{8}^{th}$, $\frac{3}{8}^{nd}$, $\frac{5}{8}^{nd}$ and $\frac{7}{8}^{th}$ of the vehicle length. However, in the default or 'cruise' mode the vehicle polygon is enlarged by a safety margin $s_f$ in the front and $s_r$ at the rear. This was already illustrated in Figure 5-1. The center point of the vehicle $(x, y)$ is still used to predict future vehicle movement. The four squares used for collision avoidance are evenly divided along the longitudinal axis of the vehicle. The four points are labeled from the front of the vehicle to the rear as 'buffer', 'front', 'mid' and 'rear'. The positions of the 'rear', 'front' and 'buffer' points are dependent on the position of the 'mid' point of the vehicle. The 'mid' point is dependent of the location of the vehicle $(x, y)$. The equations are formulated for all four points as:

$$x_{mid}(t) = x(t) - \frac{L}{8}sin(\phi(t)) + \Big(1 - B_{close}(t)\Big)\Big(\frac{s_f - s_r}{2} - \frac{s_f + s_r}{8}\Big)sin(\phi(t)) \tag{6-12}$$

$$y_{mid}(t) = x(t) - \frac{L}{8}cos(\phi(t)) + \Big(1 - B_{close}(t)\Big)\Big(\frac{s_f - s_r}{2} - \frac{s_f + s_r}{8}\Big)cos(\phi(t)) \tag{6-13}$$

$$x_{rear}(t) = x_{mid}(t) - \frac{L}{4}sin(\phi(t)) - \Big(1 - B_{close}(t)\Big)\frac{s_f + s_r}{4}sin(\phi(t)) \tag{6-14}$$

$$y_{rear}(t) = y_{mid}(t) - \frac{L}{4}cos(\phi(t)) - \Big(1 - B_{close}(t)\Big)\frac{s_f + s_r}{4}cos(\phi(t)) \tag{6-15}$$

$$x_{front}(t) = x_{mid}(t) + \frac{L}{4}sin(\phi(t)) + \Big(1 - B_{close}(t)\Big)\frac{s_f + s_r}{4}sin(\phi(t)) \tag{6-16}$$

$$y_{front}(t) = y_{mid}(t) + \frac{L}{4}cos(\phi(t)) + \Big(1 - B_{close}(t)\Big)\frac{s_f + s_r}{4}cos(\phi(t)) \tag{6-17}$$

$$x_{buffer}(t) = x_{mid}(t) + 2\frac{L}{4}sin(\phi(t)) + 2\Big(1 - B_{close}(t)\Big)\frac{s_f + s_r}{4}sin(\phi(t)) \tag{6-18}$$

$$y_{buffer}(t) = y_{mid}(t) + 2\frac{L}{4}cos(\phi(t)) + 2\Big(1 - B_{close}(t)\Big)\frac{s_f + s_r}{4}cos(\phi(t)) \tag{6-19}$$

$$\tag{6-20}$$

and again is the binary $B_{close}$ is used to modify constraints. Analyzing (6-12), the $x(t) - \frac{L}{8}sin(\phi(t))$ term indicates that the 'mid' point sits at $\frac{3}{8}L$ from the rear of the vehicle. The $\frac{s_f - s_b}{2}$ term corrects the position of the model's geometric center when the model's length is increased due to the safety margins at the front and left. The $-\frac{s_f + s_b}{8}$ term again shifts the 'mid' point to $\frac{3}{8}^{th}$ from the rear of the enlarged vehicle. The other points are constructed in a very similar way.

### 6-2-3 Bullet and target vehicle

During simulations, it became clear that the method described above has shortcomings in some scenarios. An important flaw of the system occurred in the transition from the 'close' to the 'stop' mode. As can be read in Section 6-4, the cost function includes a penalty for the 'close' and 'stop' mode, a reward for a higher speed and a weight to ensure that braking action happens as late as possible. If the algorithm detects an unavoidable collision, the 'stop' mode will be activated somewhere in the prediction horizon. Due to the time related weight in the cost function, the vehicle will brake maximally to ensure that the vehicle enters the 'close' mode as late as possible. When the vehicle has however entered the 'close' mode, a minimal cost is achieved when the vehicle accelerates maximally. Since the 'stop' mode is effectively a terminal state, reward can only be gained before entering the mode. Multiplying the cost of the 'stop' mode with the vehicle speed when entering the mode could potentially solve the problem, however this would lead to a non-linear cost function.

A possible solution is to hard code that the vehicles must brake maximally when the 'stop' mode is detected somewhere in the time horizon. In a dove tailing scenario, this could however lead to more severe collisions as can be read above in subsection 6-2-2. To ensure damage mitigation, it would however be beneficial to hard code maximal braking of the bullet vehicle when an unavoidable collision is detected. The bullet vehicle is the vehicle that frontally collides with the other vehicle. In the case of a head-on collision both vehicles are bullet vehicles. To hard code this brake action, it is necessary to detect which vehicle is the bullet vehicle. This is done using the 'buffer' point on the vehicles. The constraints to detect whether a vehicle $i$ is a bullet vehicle in a collision involving $N$ vehicles are:

$$\begin{array}{llcl}
\forall t, \forall p \in P, \forall n \in N \neq i & x_i(t, front) - x_n(t,p) & \geq & d_{veh}(i) + d_{veh}(n) + d_{safe}(i) \\
& & & \quad + d_{safe}(n) - Mb_{f,1}(t,p) \\
& x_n(t, front) - x_i(t,p) & \geq & d_{veh}(i) + d_{veh}(n) + d_{safe}(i) \\
& & & \quad + d_{safe}(n) - Mb_{f,2}(t,p) \\
& y_i(t, front) - y_n(t,p) & \geq & d_{veh}(i) + d_{veh}(n) + d_{safe}(i) \\
& & & \quad + d_{safe}(n) - Mb_{f,3}(t,p) \\
& y_n(t, front) - y_i(t,p) & \geq & d_{veh}(i) + d_{veh}(n) + d_{safe}(i) \\
& & & \quad + d_{safe}(n) - Mb_{f,4}(t,p) \\
\forall t, \forall p \in P & \sum_{j=1}^{4} b_{f,j}(t,p) - B_b(i,t) & \leq & 3
\end{array}$$

where $P$ denotes the set of points $\{'rear', 'mid', 'front', 'buffer'\}$ that are used to construct the geometric model of the vehicle. The binary $B_b(i,t)$ indicates that vehicle $i$ is the bullet vehicle during a collision that occurs at time $t$ when $B_b(i,t) = 1$. After running the optimization, the binaries $B_{stop}$ and $B_b$ can now be analyzed. If the 'stop' mode is detected, braking commands can be sent to all bullet vehicles involved in the collision.

The model does not allow vehicles to have a negative velocity. It is however important that the algorithm can deal with vehicles that are reversing. For this reason, the absolute value of the velocity is used. When the vehicle is reversing, the data is changed to suggest that the vehicle has rotated 180° and is driving forward. The safety buffer is in this case located at the rear of the vehicle. In this way, the buffer is always the foremost square in the direction of travel.

## 6-3 Control Output

As seen in subsection 5-1-1, the control output, generated as a solution of the optimization problem, influences the speed of the vehicle as:

$$v(t+1) = v(t) + \Delta t \cdot u(t)$$

and that $u(t) \in [-d_{max}, a_{max}]$ which means that the vehicle can both accelerate for positive values of $u(t)$ and decelerate for negative values of $u(t)$. The control command that is sent to the vehicle is thus a reference acceleration. The CAS is not actually used to accelerate the vehicle. However, the system will not perform optimally if acceleration is not modelled. If only deceleration is allowed, braking action will be punished heavily by the cost function. If acceleration is also modelled, the vehicle can 'recover' from the braking action in the modelling environment, thus making it easier to brake the vehicle and making the system more efficient.

Adding acceleration also simulates driver behaviour and acts as an extra safety measure. In Section 6-4 one can read how the cost function is constructed. Since the cost decreases for higher speed, the algorithm prefers to accelerate the vehicle at the maximum acceleration rate. This effectively leads to the algorithm modelling an aggressive driving style. In most scenarios this is preferable, except for the dove tailing scenario. In most cases, the predicted acceleration will be higher than the driver's input to the vehicle, leading to lower speeds than predicted by the algorithm. The brake is the only system that can be actuated using the reference deceleration. For that reason a value of zero is sent as reference when the solution to the optimization problem is not negative. The control output is defined over the whole time horizon. This makes it possible to send relevant control commands to the vehicle even if it takes some time to solve the next problem.

## 6-4 Cost Function

As stated in subsection 4-1-1, one of the methods of reducing computational cost of MILP in autonomous vehicles is Receding Horizon Control (RHC), where the algorithm calculates a solutions over a shorter time horizon instead of a single global solution. The cost function should thus aim to optimize vehicle movement over a prediction horizon while avoiding collisions. Where in Chapter 3 the cost function was focused on minimizing the health cost, for the MILP implementation the cost function tries to minimize logistic cost. The reason for this is that the hard collision avoidance constraints should guarantee that no collision occurs. Putting a high penalty on entering the 'stop' mode should ensure that this only happens when a collision is unavoidable. Looking at literature, the cost function that is used in [5] is given as:

$$J_T = \sum_{i=1}^{V} \sum_{k=0}^{T-1} \mathcal{L}_{i,k} + \mathcal{F}_{T,i} \tag{6-21}$$

where $\mathcal{L}_{i,k}$ indicates the stage cost for vehicle $i$ at the $k^{th}$ time instance and $\mathcal{F}_{T,i}$ is the terminal cost. This is almost identical to the cost function used in [19].

The authors suggest that for the terminal cost some kind of heuristic function is used that guarantees a certain performance and also a guarantee on stability. However, both authors use a terminal cost related to the trajectory of the vehicles. In [5] a separate algorithm calculates a predicted reference path, where in [19] predefined reference trajectories are used. The terminal cost is then determined based on the distance between the vehicle's positions and the reference trajectory. But since the CAS application only supports braking actions, no control is available to correct deviations from the reference trajectory. The author of [43] uses the cost function:

$$J_T = \sum_{i=1}^{N} T_{Fi}$$

which sums over all $N$ vehicles the time $T_{Fi}$ needed for vehicle $i$ to reach its final way point, similar to the terminal cost $\mathcal{F}_{i,k}$ in (6-21). However, for our application there is often no final way point, especially when using RHC. This means that a new cost function has to be designed.

Collision avoidance is guaranteed by the use of hard constraints when perfect GPS data is available and when the optimization problem can be run instantaneous. However, due to delays and GPS drift, the 'close' and 'stop' mode are introduced to ensure that optimization problems are always feasible. Since the hard constraints are formulated to ensure collision avoidance, the main goal of the cost function is to ensure that minimal logistic cost is incurred. For this reason, a reward is given for vehicles driving at higher speed. The term is formulated as:

$$-\frac{\alpha \cdot V(t)}{V_{max}} \tag{6-22}$$

where $V_{max}$ is the maximal speed of the vehicle and $\alpha$ a weighting parameter. The $V_{max}$ term is introduced in order to differentiate between different vehicles, as was discussed in Section 3-4. Slow vehicles normally are heavy, and in general heavier vehicles like haul trucks, graders and excavators are contributing to production more than light vehicles such as civil vehicles and (empty) smaller trucks. The cost function in (6-22) will thus prioritize more important vehicles. An extra parameter could be added to manually indicate the importance of each vehicle, but will be prone to abuse when a system is commercially used.

As can be read in subsection 6-2-2 and 6-2-1, a cost is associated with the 'close' and 'stop' mode. The cost terms $Q_{close}$ and $Q_{stop}$ for each mode are multiplied with the respective binaries $B_{close}$ and $B_{stop}$ so that the penalty only applies when the mode is activated. The final cost function for a collision involving $N$ vehicles is:

$$J = \sum_{t=1}^{T}\sum_{n=1}^{N}\Big(-\frac{\alpha \cdot V(n,t)}{V_{max}(v)} + Q_{close}B_{close}(t) + Q_{stop}B_{stop}(t)\Big)W(t) + \beta \cdot B_{frontal}$$

where the binary $B_{frontal}$ is added with a weight $\beta$ to the function because the algorithm is otherwise indifferent to the value of the binary, while it is in fact used for further analysis of the problem. The term $W(t)$ is a decreasing ramp function over time. The term is added to ensure that braking happens as late as possible. If the term is left out, the algorithm will detect that a brake action is needed and also perform the action, but it can happen at any time step within the time horizon because the final cost $J$ will not be influenced. Due to the dynamic nature of a mine, it is however better to perform a braking action as late as possible, to ensure unnecessary early braking and false positives are kept at a minimum.

## 6-5 Conclusion

The MILP method achieves collision avoidance by using hard constraints on the distance between vehicles. Due to sudden unexpected maneuvers or sensor noise, unavoidable collision can however occur. In the normal formulation, this would result in infeasible problems without a solution, effectively disabling the CAS when it is most needed. To deal with these unavoidable collisions, a algorithm is constructed that allows vehicles to operate in different operation modes.

Constraints are linked to and modified based on these modes. In the 'close' mode, the distance allowed between vehicles is reduced while the 'stop' mode is a terminal state which simulates the collision. The bullet and target vehicle(s) are identified for additional control outside the optimization. The linear cost function is mainly based on the logistic cost, together with the penalties for entering the 'close' and 'stop' modes. Weights can be added to give vehicles a priority or to model traffic situations.

# Chapter 7

# Simulations

In order to get a good idea of how the Collision Avoidance System (CAS) performs, it is necessary to run extensive simulations. In this chapter the different simulations and results are described. The simulations are run on a HP Elitebook 8570w using an Intel Core i7-3630QM processor. The model is written in AMPL, a programming language specifically designed to describe and solve large-scale complex mathematical problems. The solver is the `cplex` solver developed by IBM for linear, mixed integer and quadratic programming optimization problems. Using an API, the script used to occupy the model with data and analyse the results is executed in MATLAB. The `cplex` solver is then tasked with finding the optimal solution to the problem. While it is hard to determine the overheads created by this method, it might have had an influence on the results seen here.

To get a realistic simulation, it is important that randomness in the vehicle's behaviour due to the driver input, is modelled. This is discussed in Section 7-1. Different scenarios that are simulated are described in Section 7-2. The results of the simulations, with regards to both collision avoidance and computational performance, are analyzed in Section 7-3. A conclusion can be read in Section 7-4.

## 7-1 Vehicle Behaviour

In order to get accurate simulations, the vehicle's behaviour cannot be solely dictated by the CAS. It is important to see how the system reacts to unexpected changes in the vehicle's behaviour, for example sudden braking. For this reason, a random control input is generated that simulates the driver's input to the vehicle.

The driver input is randomly selected within the boundaries of the vehicle's characteristics. Due to the randomness of the driver input, a number of simulations did not result in the vehicle's safety zones overlapping at any point during the simulation period. This means that no dangerous situation occurs. To ensure that most simulations are however valid, the initial value is chosen as $u_d(1) \in [-1, 1]$. Since drivers do not often change their input from maximal acceleration to maximal braking, a certain smoothness was required. To ensure this, the driver input $u_d(t+1) \in [u_d(t) - 0.8, u_d(t) + 1]$. This asymmetric interval was chosen because a symmetric interval results in a lot of simulations where the vehicle simply comes to a stop and does not move for the remainder of the simulation. The CAS overrides this driver input only when $u_{CAS}(t) < u_d(t)$. The computational delay was included in the model to ensure that the correct control command was sent to the vehicles.

## 7-2 Scenarios

To get a good estimate of how the system performs, it is important to test a host of different scenarios. The scenarios that are tested in this thesis are taken from a collection of common mining vehicle interaction scenarios [6]. All common driving scenarios except those with reversing vehicles are evaluated, as this is equivalent to a vehicle driving forward for the CAS developed in this study.

In subsection 7-2-1 the simulations involving two vehicles are described. In subsection 7-2-2 simulations with three vehicles are described, while in subsection 7-2-3 simulations with two vehicles and a single pedestrian are described.

### 7-2-1 Two Vehicles

The most basic scenario that was tested involves two vehicles. It is the computationally least expensive scenario, and also a traffic scenario that is most commonly encountered on mines. First scenarios where the vehicles are driving in a straight line are discussed, followed by scenarios where vehicles drive on a curved path.

#### Straight line driving

For vehicles driving in a straight line, five different scenarios are tested. Each scenario features a different angle of impact. In Figure 7-1 the different scenarios are visualized. Two variations of the T1 scenario are tested, where the angle of approach $a$ between the vehicles is either 45° or 135°. Each simulation is started with the vehicles in the correct configuration regarding direction and position, however the distance between the vehicles and the speed of the vehicles is chosen randomly within the boundaries of the vehicle characteristics.

To simulate some unpredictability of the driver's behaviour, a random control sequence was generated before the simulation. If the CAS did not generate a braking control for the vehicle, the control from the generated control was used. In order to do some testing for false positives, the simulation was also run using only the random control sequence. This resulted in a number of simulations where the vehicles did not enter each others safety zone. In Table 7-2, the $\#sim$ variable differs between scenarios. This is due to the fact that some scenarios did indeed not result in a dangerous situation.

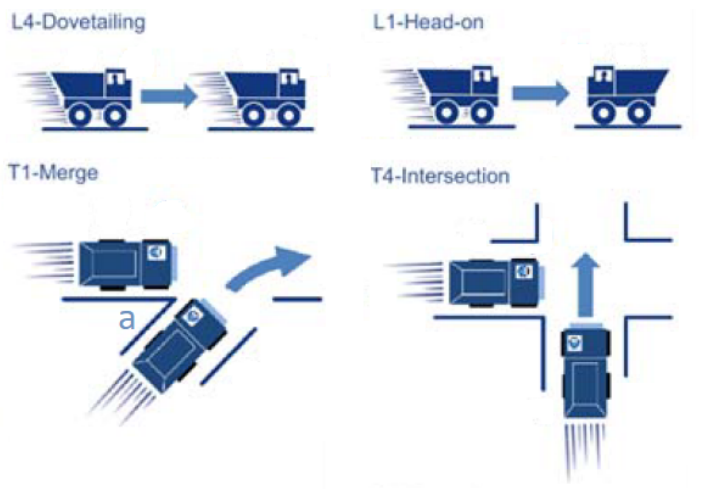


**Figure 7-1:** Scenarios tested for two vehicles when driving straight [6]

#### Curved line driving

Since it is much harder to predict the vehicle's movement when it is driving in a curved line, these curved scenario's where also tested. Two variations are possible, which can be seen in Figure 7-2. In the first scenario the two vehicles are driving towards each other. In the second scenario, one vehicle is driving behind another vehicle. Both of these are tested.

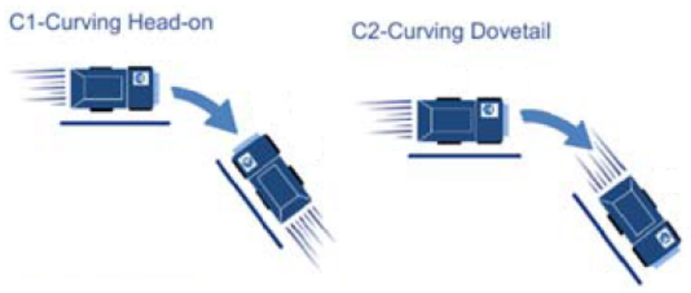


**Figure 7-2:** Scenarios tested for two vehicles on curved paths [6]

### 7-2-2 Three Vehicles

In the previous sections simulations with two vehicles were described. On a mine site, there are however a lot of interactions involving multiple vehicles. For this reason, some scenarios involving three vehicles were also simulated. The scenarios are very similar to the ones for straight driving with two vehicles and can be seen in Figure 7-3.

As can be read in Section 5-2, the Mixed Integer Linear Programming (MILP) formulation is $\mathcal{O}(V^2n^2)$ where $V$ is the number of vehicles. As the number of constraints increases quadratically, the computational time is also expected to increase significantly. By analyzing the differences in performance between scenarios involving either two or three vehicles, more insight can be gained in the scalability of the system.

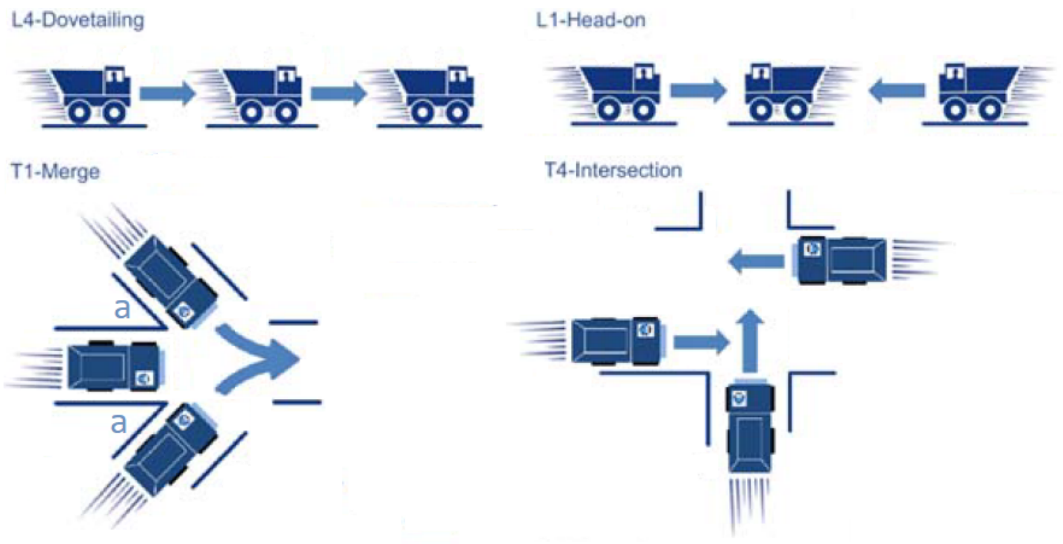


**Figure 7-3:** Scenarios tested for three vehicles when driving straight [6]

### 7-2-3 Two Vehicles and One Pedestrian

An important function of the CAS is to prevent collisions between vehicles and pedestrians. If the safety zone around the pedestrian is big enough, one can assume that a pedestrian remains stationary within its safety parameter. The method has two possible drawbacks for long prediction horizons. Either the safety zone has to be very big or the model becomes inaccurate. While pedestrians cannot be controlled in the same manner as a vehicle, they can be warned by means of a buzzer. It then must be assumed that the pedestrian will stop when the buzzer sounds.

In order to speed up calculations, it would be much better to model a pedestrian as a single square. Due to time constraints, the pedestrian is modelled as a stationary vehicle with characteristics that don't allow it to move. As pedestrians slow compared to vehicles, this assumption is reasonably valid. There is not much difference between these simulations and the simulations with three vehicles, except that one of the vehicles does not move. For these simulations, an adaptation of the scenarios described in Figure 7-1 will be tested. The pedestrian will be positioned at the point where the trajectories of the two vehicles intersect. It will be interesting to see the difference in computational time between these simulations and the ones discussed in Figure 7-1.

## 7-3 Results

In this section the results will be analyzed. The two criteria used to analyse the system are collision avoidance and computational performance. In subsection 7-3-1 the performance of the system with regards to collision avoidance is described. The computational performance is important as this is an indicator on the system's applicability in a commercial application. The computational performance is discussed in subsection 7-3-2.

## 7-3-1 Collision avoidance

One of the most important performance criteria of the CAS is its ability to avoid collisions. This is also what is analysed during the commercial tests done by the Vehicle Dynamics Group (VDG). The results of simulations are discussed here. The exact results are gathered in Table 7-2. For each kind of scenario, an analysis of the results is also done.

#### Two vehicles

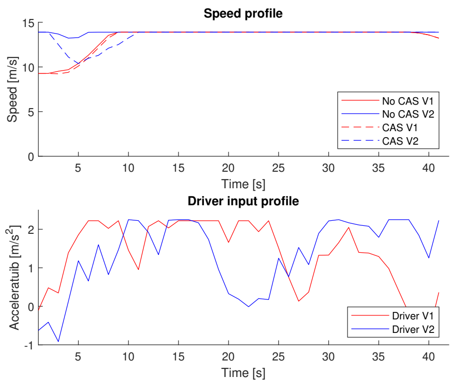
The simulations of scenarios with two vehicles were the most simple ones performed. The safety zone at the front of the vehicles is set at $s_f = 4m$ and at the rear $s_r = 2m$. As can be seen in Table 7-2, the algorithm can reduce the number of collisions significantly. For all straight and curved driving scenarios, except for L4 - Dovetailing, all collisions are prevented. The number of times the vehicles are too close to each other is also reduced significantly. Another important effect of the CAS is that the average speed of the vehicles during these close passings are reduces greatly. Where the average speed in these close encounters for all scenarios was between 10 and $11ms^{-1}$ for the uncontrolled case, it is below $1ms^{-1}$ when the CAS is activated. This provides additional safety by ensuring that vehicles drive at a low speed when they pass close by another vehicle.

In the L4 - Dovetailing scenario the cause for collision is braking of the front vehicle due to the driver input. One might imagine that the driver is stopping for some animal that sits in the road or for another reason that cannot be predicted by the computer. The rear vehicle is in these situations driving too close to the front vehicle. The small distance, big speed differences and computational delays cause the rear vehicle to drive in the front vehicle. To solve this, the size of the safety zone was increased. Performance for different $s_f$ can be seen in Table 7-1. The size of the safety zone did not have any effect on computational performance. If there are no computational delays and the rear vehicle can brake as hard as the front vehicle, these collisions could be prevented without an increased $s_f$.

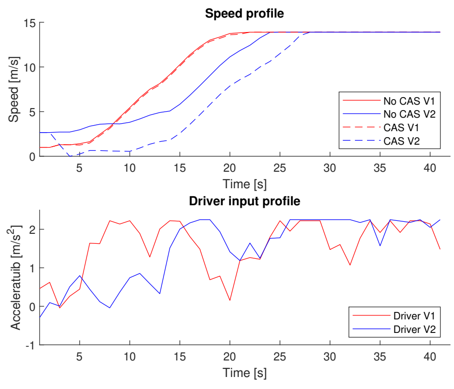
| $s_f$ [m] | #Close prevented | #Collision prevented |
|---|---|---|
| 4 | $\frac{79}{133} = 59\%$ | $\frac{98}{105} = 93\%$ |
| 8 | $\frac{84}{146} = 57\%$ | $\frac{114}{115} = 99\%$ |
| 12 | $\frac{90}{168} = 53\%$ | $\frac{108}{110} = 98\%$ |
| 16 | $\frac{112}{196} = 57\%$ | $\frac{100}{100} = 100\%$ |

**Table 7-1:** Performance in L4 - Dovetailing scenario for different $s_f$

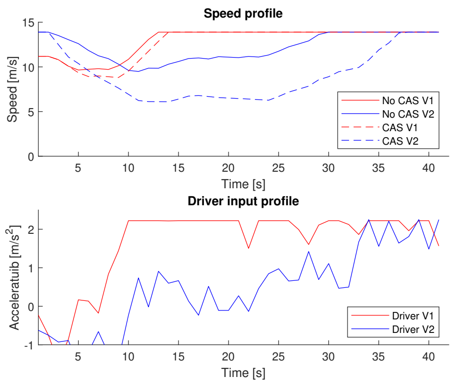
Four speed profiles of the T4 - Intersection scenario can be seen in Figure 7-4. The top plot of each profile shows the velocity of both vehicles during the simulation. The bottom plot shows the driver input for both vehicles. The top plot shows that the CAS overwrites the driver input and slows down the vehicle in each simulation. It also becomes clear that due to the driver's unexpected driving, the algorithm sometimes changes which vehicle must be slowed down. This can result in a braking command issued to both vehicles, as can be seen best in Figure 7-4d.



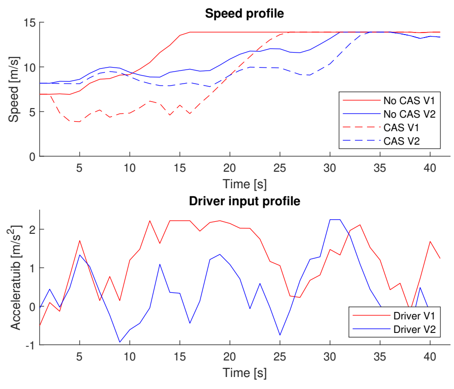
**(a)** Speed profile 1



**(b)** Speed profile 2



**(c)** Speed profile 3



**(d)** Speed profile 4

**Figure 7-4:** Speed profiles for four simulations

### Three vehicles

The four scenarios with three vehicles that are simulated have similar results to those for two vehicles. In Table 7-2 one can see that most collisions are prevented. Only in the L1 - Head-on scenario some collisions still occur. The cause is the same as in the L4 - Dovetailing scenario with two vehicles. In the simulations with three vehicles, the speed differences in the L4 - Dovetailing scenario are a little lower than in the simulations with two vehicles, due to the way in which the random initial conditions for the simulation are generated. This prevents the accidents that occurred in the simulations with two vehicles. An increased safety zone is however also beneficial in this scenario. The speed during a close encounter is decreased by about $10ms^{-1}$, comparable to the scenarios involving two vehicles.

### Two vehicles one pedestrian

These scenarios are different from the previous results, in that the vehicles must both come to a complete stop due to the pedestrian that is placed at the intersection of the two trajectories. Again two collisions are detected in the L4- Dovetailing scenario. These collisions are between vehicles and caused by the safety zone being too small. This was also seen in the scenarios involving two vehicles. The pedestrian is not hit during any of the simulations.

#### Collision avoidance: conclusion

Based on the results from the simulations, it looks like the algorithm is very capable of avoiding collisions. In all scenarios except L4 - Dovetailing all collisions were avoided. When the safety zone was increased, the L4 - Dovetailing was also safely controlled by the CAS. Decreasing the solution time might solve the problem with this scenario, or guarantee a more appropriate size of the safety zone.

Another option is to make the safety zone variable. By having an algorithm determine the orientation during the predicted collision event, the safety zones of all vehicles can be changes so that only bullet vehicles have a higher $s_f$. Additionally, other methods of keeping an appropriate distance between dovetailing vehicles can be studied.

### 7-3-2 Computational performance

To see whether the system can be used in a commercial application, it is important to analyse the computational performance and compare between the different scenarios.

#### Two vehicles

The computational performance for the straight driving scenarios involving two vehicles is promising for a commercial application. The solution time is on average around the 0.5 seconds, as can be seen in Table 7-3. By simply upgrading to better hardware, this can be scaled down an order of magnitude. The curved scenarios however show a solution time that is up to 5 times as large on average. Because of noise and vibrations of the vehicle, almost all real world scenarios where vehicle drive in a straight line, will still have a slight steering component which increases the computational time.

For each simulation, the longest solution time $T_m$ was collected. The second and third column in Table 7-3 show the maximum and minimum value of $T_m$ over all simulations. The maximum value shows that, especially in the curved scenario, there are some problems that take almost an hour to solve. The reason for this is unclear, and it only happens for a handful of problems out of the thousands that were run. The average $T_m$ is determined after excluding the highest 3% of values. This is done since, especially in the curved scenario, a handful of excessive outliers give a very skewed value. These values are an important indicator of the performance of the system.

#### Three vehicles

To analyse the scalability of the system, comparing the solution times of the different scenarios is interesting. As can be seen in Table 7-3, the computational time increases very fast when an extra vehicle is added to the model. Solution times for some time steps blow up to more than half an hour, but these outliers are likely not caused by the difficulty of the optimization

| Scenario | $\#sim$ | $\#col$ - No CAS | $\#col$ - CAS | $\#close$ - CAS |
|---|---|---|---|---|
| **L4 - Dovetailing** | | | | |
| Two vehicles | 197 | 160 | 15 | 87 |
| Three vehicles | 99 | 73 | 0 | 25 |
| Two vehicles & one pedestrian | 100 | 100 | 2 | 90 |
| **T1 - Merge** $a = 45°$ | | | | |
| Two vehicles | 232 | 97 | 0 | 26 |
| Three vehicles | 100 | 65 | 0 | 16 |
| Two vehicles & one pedestrian | 100 | 100 | 0 | 88 |
| **T4 - Intersection** | | | | |
| Two vehicles | 231 | 68 | 0 | 13 |
| Three vehicles | 100 | 46 | 0 | 23 |
| Two vehicles & one pedestrian | 100 | 100 | 0 | 79 |
| **T1 - Merge** $a = 135°$ | | | | |
| Two vehicles | 262 | 127 | 0 | 57 |
| Three vehicles | 100 | 74 | 0 | 14 |
| Two vehicles & one pedestrian | 100 | 100 | 0 | 87 |
| **L1 - Head-on** | | | | |
| Two vehicles | 300 | 282 | 0 | 209 |
| Three vehicles | 100 | 100 | 12 | 90 |
| Two vehicles & one pedestrian | 100 | 100 | 0 | 0 |
| **C1 - Curving Head-on** | | | | |
| Two vehicles | 146 | 58 | 0 | 59 |
| **C2 - Curving Dovetail** | | | | |
| Two vehicles | 139 | 90 | 0 | 36 |

**Table 7-2:** Comparison of performance for different scenarios using $s_f = 4$

problem. The mean $T_m$ is a more accurate indicator for the computational requirements for the system. The top 3% is excluded to remove some of the extreme outliers. Solution times still remain high. Especially in the T1, $\alpha = 135°$ scenario, a reduction by only one order of magnitude will not be enough in a commercial application. It is interesting to see that, as with the scenario involving two vehicles, this specific type of simulation takes a lot of time to solve.

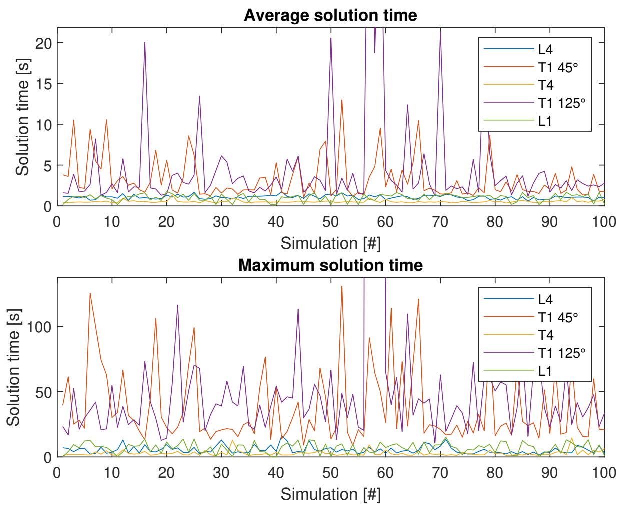


**Figure 7-5:** Average and maximum solution time of each simulation of three vehicles

### Two vehicles one pedestrian

As with the scenarios involving three vehicles, the simulations involving two vehicles and one pedestrian took a lot of time. The reason is unclear, and cannot be explained by the construction of the code. These simulations are almost identical to those with three vehicles, expect one of the vehicles cannot be controlled. One would expect that this decreases the computational load, since less variables are available.

However, in Table 7-3 it becomes clear that these scenarios take longer to solve than it does for the scenarios using three vehicles. It is thus clear that pedestrians should not be modeled as a stationary vehicle. A much better approach will be to introduce geo-fencing, where pedestrians are modeled as obstacles. When this is done, additional obstacles like movable buildings or other equipment could also be added to the model, providing additional benefit.

### Computational performance: conclusion

The results of the simulations with two vehicles are promising. If the solution times can be reduces by an order of magnitude, an acceptable rate of 10-20 Hz can be achieved. A possible way of achieving this might be to only use three instead of four points to model the vehicle geometry. It seems however that the problem is not very scalable. By rewriting the problem in C++ an increase in computational speed can be achieved, as it removes some of the overheads created by using MATLAB. It also gives the option of parallelizing the process.

| Scenario | $\bar{T}_{sim}$ [s] | max $T_m$ [s] | min $T_m$ [s] | mean $T_m$ [s] ex. top 3% |
|---|---|---|---|---|
| **L4 - Dovetailing** | | | | |
| Two vehicle | 0.528 | 2.736 | 0.806 | 1.412 |
| Three vehicles | 1.078 | 16.18 | 1.397 | 4.899 |
| Two vehicles & one pedestrian | 1.253 | 10.49 | 1.27 | 5.072 |
| **T1 - Merge** $a = 45°$ | | | | |
| Two vehicle | 0.421 | 9.483 | 0.575 | 1.387 |
| Three vehicles | 3.260 | 130.7 | 8.765 | 32.90 |
| Two vehicles & one pedestrian | 13.04 | 600.4 | 2.113 | 125.8 |
| **T4 - Intersection** | | | | |
| Two vehicle | 0.329 | 6.373 | 0.509 | 0.774 |
| Three vehicles | 0.528 | 14.53 | 0.874 | 2.336 |
| Two vehicles & one pedestrian | 1.374 | 10.94 | 0.852 | 4.774 |
| **T1 - Merge** $a = 135°$ | | | | |
| Two vehicle | 0.478 | 31.64 | 0.770 | 1.720 |
| Three vehicles | 4.813 | 2204 | 10.60 | 39.20 |
| Two vehicles & one pedestrian | 11.98 | 4808 | 1.841 | 71.172 |
| **L1 - Head-on** | | | | |
| Two vehicle | 0.623 | 4.958 | 0.459 | 1.738 |
| Three vehicles | 1.008 | 15.06 | 0.167 | 6.439 |
| Two vehicles & one pedestrian | 0.951 | 8.070 | 0.610 | 2.931 |
| **C1 - Curving Head-on** | | | | |
| Two vehicles | 2.490 | 2807 | 0.494 | 14.94 |
| **C2 - Curving Dovetail** | | | | |
| Two vehicles | 1.235 | 3448 | 0.479 | 4.147 |

**Table 7-3:** Comparison of computation time for different scenarios using $s_f = 4$

When this is done, a new optimization problem can be initialized before the solution of the previous problem is available. If one of the problems then takes very long to solve, the new problem might have been solved already. This reduces the impact of the high solution times that were sometimes seen in simulations.

### 7-3-3 False positives

In a number of simulations, the vehicles started at such an initial speed and position that they did not enter each others safety zone. These simulations were used to detect false positives. A false positive is defined as activation of the brakes by the CAS while no collision occurs if the CAS does not intervene. As described in Section 7-1, the vehicles were randomly accelerating or decelerating when the CAS did not issue brake commands.

| Scenario | # Tests | # False positives | False positives [%] |
|---|---|---|---|
| **L4 - Dovetailing** | | | |
| Two vehicle | 103 | 23 | 22 |
| **T1 - Merge** $a = 45°$ | | | |
| Two vehicle | 68 | 68 | 100 |
| **T4 - Intersection** | | | |
| Two vehicle | 69 | 69 | 100 |
| **T1 - Merge** $a = 135°$ | | | |
| Two vehicle | 38 | 38 | 100 |
| **L1 - Head-on** | | | |
| Two vehicle | 0 | 0 | - |

**Table 7-4:** False positives for straight diving scenarios involving two vehicles

The simulations show that false positives do occur. Exact results can be seen in Table 7-4. However, these are not comprehensive results. For example, the L1 - Head-on scenario did not have any simulations in which the vehicles did not collide. This means that there was no simulation in which false positives could occur. In scenario L4 - Dovetailing, a false positive occurred in 22% of simulations. In the other three scenarios, all simulations resulted in an intervention of the CAS.

A possible explanation lies in the difference between the collision detection with and without the CAS being activated. Without the CAS, a fast algorithm propagates the vehicle according to the driver input. For each vehicle, a polygon is then constructed at each time step that is an exact models of the vehicle with its safety zone. If the polygons of the vehicles involved in the collision are overlapping, the vehicles are passing very close. This is also done for a polygon of the vehicle without its safety zone, which is used to detect a collision. The area covered by the multiple point model is bigger than that of the polygon, as can be seen in Figure 5-2. This might be an explanation for probably all and at least a number of false positives.

In Figure 7-6 the number of interventions for each scenario are visualized. It becomes clear that most false positives have a very small amount of interventions. Since the time step has a length of 0.5 seconds, this means that for the majority of simulations (42%) these false positives occur only two times. It will be interesting to study how often these interventions happen when the solution time is decreases significantly. If these happen at the same rate, is might be possible to introduce filtering so that the brakes will only be activated when the CAS sends braking commands in for example three consecutive solutions. Extensive research should however be done to guarantee that this filtering does not have an impact on the collision avoidance capabilities of the system.

## 7-4 Conclusion

To get a good idea of the performance of the system, simulations are performed. Three different types of scenarios are tested. They involve two vehicles, three vehicles or two vehicles

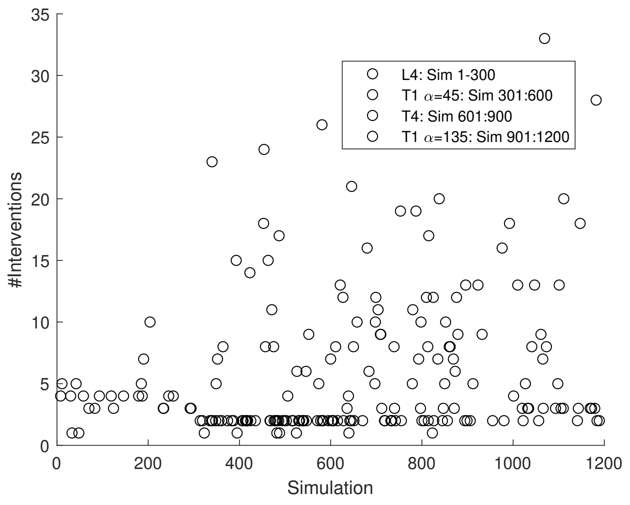


**Figure 7-6:** Number of interventions during a false positive

and a pedestrian. The simulations are run on a HP Elitebook 8570w using an Intel Core i7-3630QM processor. Random input data is generated that simulated driver behaviour. The CAS overrides this only when the braking command sent to the vehicle is stronger than that generated by the driver.

The results of the simulation are promising. If the safety zone is big enough, all collisions are avoided. A drawback is the high solution time, especially for scenarios involving three vehicles or two vehicles and a pedestrian. It might be possible to speed up computational times significantly. Rewriting the algorithm in C++ can potentially achieve a significant reduction while also making it possible to solve optimization problems in parallel.

# Chapter 8

# Testing

After seeing promising results from the simulations, tests were done on two Land Rover 110 4x4 vehicle provided by Vehicle Dynamics Group (VDG). Due to time constraints and a broken the brake robot, it was not possible to do a full test. The goal of the tests was thus not necessarily to see the collision avoidance properties of the system, but mainly to see whether the solution times and results found in the simulations could also be achieved when using real data and communication delays.

In this chapter a description is given of the equipment used for the tests and results. First, the sensors used on the vehicle are described in Section 8-1. A short overview of the hardware used in the network is given in Section 8-2. The results of the tests are given in Section 8-3. A short conclusion can be read in Section 8-4.

## 8-1 Odometry

As the system is GPS based, it is important to get accurate measurements. The hardware used by the VDG is the VBOX 3i Dual Antenna system or VB3iSL, which operates at 100Hz. This antenna tracks both the American GPS and Russian GLONASS satellites. The combination of these two systems make it possible for the antenna to track almost twice as many satellites. This ensures that the satellite connection is more robust, while also increasing accuracy. Because two antennas are used, the for our application important yaw rate can be accurately measured using the VBOX.

To increase accuracy and reliability of the data, Racelogic's Inertial Measurement Unit (RLVBIMU04-V2) is connected with the VB3iSL. The IMU has three accelerometers and three gyroscopes, which ensure that velocity data is more smooth, It also makes measurements of all angular velocities more accurate. Another benefit of the IMU is that an estimated position can be calculated if the satellite data is interrupted.

During testing, the VDG uses the RACELOGIC DGNSS RTK Base Station. The base station enables differential correction of the position estimate. For our testing, this was not used as the testing was more focused on computational performance as on testing the collision avoidance capacity of the Collision Avoidance System (CAS).

## 8-2 Network

The network that is used for testing consists of a central unit and a unit on each vehicle. The central unit is tasked with listening for vehicle data, running the optimization and sending out control commands to the vehicles. The unit on the vehicle is tasked with gathering vehicle data, sending the data to the central unit, listening for control and using the control command to actuate the brake robot.

The network is set up using a laptop as the central hub. The laptop is a HP Elitebook 8570w, using a Intel Core i7-3630QM processor. The position data from each vehicle is sent to the laptop using Wi-Fi. On the laptop side, a TP-Link TD-W9970 wirelesss router was used to communicate with both vehicles. Using the DSP System toolbox provided by MATLAB, the communication was set up using the User Datagram Protocol (UDP). The protocol cannot check that the message is received, which saves a lot of overhead. To ensure some robustness, three confirmation bites are sent at the beginning and end of the message.

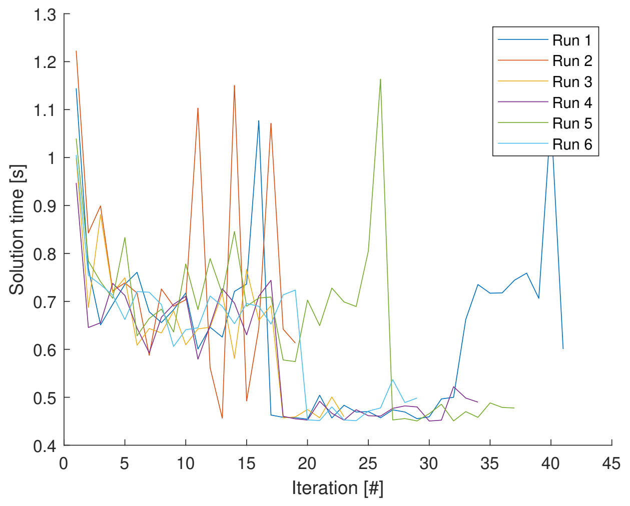


**Figure 8-1:** Solution time for different test runs

## 8-3 Results

As the brake robot was not activated, it is not possible to compare the performance of the CAS during testing with the results from simulations. The first thing that was tested was the communication delay. The vehicles were sending out their odometry data at a rate of 100Hz, a rate much faster than that of the solver. This was done to ensure that very recent data is always available to occupy the model. These odometry messages also carried a time stamp, and by sending the time stamp back from the central unit upon reception, it was established that the time delay was less than a millisecond. Since this is an order of magnitude slower than that of the fastest process in the system, this is insignificant.

A total of six test runs were done to gather data. These were done in the parking lot of the faculty, as can be seen in Figure 8-2. These were all of the L4 - Dovetailing type as discussed in Figure 7-1. The solution time for the different runs can be seen in Figure 8-1. Run 1, 3, 4 and 6 show very similar times. The vehicle is manually stopped between the $15^{th}$ and $20^{th}$ iteration, which can be clearly seen in the solution time decreasing. Although these are short runs, the times look very similar to the times found in Chapter 7.

Furthermore, the algorithm seems to work as expected with braking commands being sent to the vehicle at more or less at the same moment that the driver manually braked the vehicle. However, while 0.5 seconds sounds relatively fast, it became clear during testing that a lot can happen in that period of time. A vehicle driving at $50kmh^{-1}$ or around $14ms^{-1}$ can drive seven meters in that period. While this was prior knowledge, testing made it even clearer how far a vehicle can travel in 0.5 seconds. Combining the aforementioned seven meters with steering, this means that the actual path of the vehicle can differ so much from the predicted path that the control sent to the vehicle might be useless. For a commercial application, it is thus important to lower the computational time with at least an order of magnitude.

**Figure 8-2:** One of the vehicles that was used for testing

## 8-4 Conclusion

Since simulation results were promising, the system was implemented on two test vehicles provided by the the VDG. Due to a broken brake robot, the testing was focused on verifying that the results from the simulations can be reproduced using noisy real world data. The solution times during testing are very comparable to those found during simulations in Chapter 7. The control commands that were issued by the solver also seem reasonable as a braking command was issued before the driver manually braked the vehicle.

# Chapter 9

# Conclusion and Recommendations

## 9-1 Conclusion

A centralized Collision Avoidance System (CAS) has potential significant benefits over a decentralized system. The accurate modelling when using Non-Linear Programming (NLP) results in very complex problems that cannot be solved fast and reliably. Using Mixed Integer Linear Programming (MILP) for collision avoidance has great potential. Simulation times decrease to a rate of between 1 and 2 Hz. Geometric models used in literature are not suitable for this application. This required the formulation of a new geometric model. Describing the geometric shape of the vehicle using four squares shows an accurate and computationally efficient representation. In retrospect, a polygon model would also be very well suited since it is more accurate and comes at a similar computational cost. More research on the effect of using a different amount of points would be interesting.

Similar MILP algorithms for collision avoidance, used for aviation applications, cannot deal with the non linear event of a collision due to the hard constraints on collision avoidance. To ensure that the optimization problem never becomes infeasible, different operation modes were introduced. These modes successfully ensured that collisions were prevented in all scenarios that were tested. Simulations show that the algorithm is also able to deal with the randomness that is inherent to vehicles that are primary controlled by the driver.

One of the limitations of the algorithm become clear in the L4 - Dovetailing scenario. Due to the delay of $\pm 0.5$ s caused by the solution time, the rear vehicle reacts too late and cannot brake in time. This happens especially when the rear vehicle cannot decelerate as fast as the front vehicle. To solve this, the safety zone in front of the vehicle $s_f$ can be extended. If the solution time is decreased significantly, the problem will also disappear. Pedestrians should not be modeled as stationary vehicles since this increases the computational cost of the model. False positives do occur frequently. The exact cause of this is unclear and should be studied further. The brake activation during these false positives happens only for short instances, which might be filtered out when the solution time is reduced significantly.

The algorithm was also implemented on two test vehicles. A variation on the L4 - Dovetailing scenario, where the front vehicle remains stationary, was tested a total of seven times. Results show that the solution time during testing is very similar to the simulations where noise free artificial data is used. The tests shows that the control sent to the vehicle is similar to that seen in the simulations. The method has the potential to be used in a commercial CAS, provided that the solution time is reduced by at least an order of magnitude if a method is derived to reduce the computational cost for the T1 scenario. More research must also be done in the scalability of the optimization problem and how this can be improved.

## 9-2 Recommendations

As with any research, a lot of topics are left untouched. Studying these could potentially greatly improve the performance of the CAS or alternatively give insight in shortcomings that the system has. The two main criteria on which the CAS is tested are the solution time and accuray of the algorithm. This section gives a number of suggestions to improve both of these factors.

### Solution time

As was said in the conclusion, the algorithm must be sped up significantly if a commercial application is considered. In order to speed up computations, rewriting the algorithm in C++ could potentially remove some of the overhead created by using MATLAB. A major benefit of using C++ would be that the optimization can be run parallel. While the Parallel Computing Toolbox for MATLAB has this functionality, it does not work succesfully when the AMPL API is used inside the parallel loop. With C++ one can manually assign processes to different threads, making optimizations much faster. As can be read in Chapter 7, it happens regularly that a solver takes an excessive long time to solve the optimization problem. When the CAS is serialized, this means that no control command is available for the vehicle for an extended amount of time. If it is parallelized, multiple optimization problems run at the same time. Data comes in at a much higher rate than the solving time $t_s$. If a new optimization problem is started every $t_s$ seconds, there will generally be only one problem being solved at any time. However, when a problem takes longer to solve, a new problem with the most recent vehicle data starts being solved. If the first problem then takes longer to be solver than $t_s$ plus the solution time of the seconds problem, the second problem will be solved first. The second problem's solution is based on more recent vehicle data and thus more accurate than the first problem's solution. This means that the solver can stop working on the first problem as soon as the second problem is solved.

A change to the actual algorithm could potentially also reduce the solving time significantly. As can be read in subsection 4-1-3, uniform gridding is used for collision detection. The authors of [35] indicate that computational time decreases significantly using an iterative approach, compared to uniform gridding. The iterative approach does collision detection on a fine grid, and if a collision is detected collision avoidance is done on a very coarse grid. More points are added to the collision avoidance grid as long as a collision is detected using the fine grid. The authors of [35] state that the gain in computational time increases even more if the problem becomes more complex. This is especially interesting since [34] indicates that MILP struggles with bigger problems. It remains to be seen whether a reduction in computational cost is achieved if this method is used in the centralized CAS. Especially with a high number of vehicles the method might lead to higher computational times, because braking of one vehicle might for example lead to a collision with a dove tailing vehicle. In these more complex situations it might happen that the high number of iterations might lead to a higher cumulative computational time than uniform gridding. Experiments should be performed to determine whether the extra solutions that have to be generated with this proposed method are indeed faster than a single calculation using uniform gridding, as the authors suggest.

### Accuracy

A few improvements could be made that could improve accuracy of the algorithm. Using a kinematic bicycle model to model the dynamics of the vehicle is a strong simplification of the vehicle's behaviour. This model assumes no side slip and no skidding, which are inaccurate especially in a dusty mining environment. Using a linearized dynamic bicycle mode, the predictions of the vehicle's future position might become more accurate.

Finding reference trajectories for all vehicles could potentially greatly improve the accuracy of the algorithm. Pedicting vehicle movement becomes much easier when it is known where the vehicle is headed. The steering angle $\delta$ could then be predicted for future times, so that the algorithm could better anticipate vehicle movements like turning.

By extending the functionality of the CAS to include steering, a lot more options are available to avoid collisions. The steering would however have to be coupled with stability control to ensure that the CAS does not cause the driver to loose control over the vehicle.

A topic for further research will be how sensor noise does influence the performance of the system. The odometry sensors, as described in Section 8-1, are very accurate and some of the best sensors available.

# Bibliography

[1] D. Anurag, S. Ghosh, and S. Bandyopadhyay, "GPS based vehicular collision warning system using IEEE 802.15.4 mac/phy standard," in *Proceedings of 2008 8th International Conference on Intelligent Transport System Telecommunications*, 2008.

[2] D. W. Kononen, C. A. C. Flannagan, and S. C. Wang, "Identification and validation of a logistic regression model for predicting serious injuries associated with motor vehicle crashes," *Accident Analysis & Prevention*, vol. 43, no. 1, pp. 112–122, 2011.

[3] E. Rosen, H. Stigson, and U. Sander, "Literature review of pedestrian fatality risk as a function of car impact speed," *Accident Analysis & Prevention*, vol. 43, no. 1, pp. 25–33, 2011.

[4] J. Bellingham, A. Richards, and J. P. How, "Receding horizon control of autonomoud aerial vehicles," in *Proceedings of the 2002 American Control Conference*, 2002.

[5] T. Schouwenaars, *Safe Trajectory Planning of Autonomous Vehicles.* PhD thesis, Katholieke Universiteit Leuven, 2006.

[6] EMESRT, "Pr-5a vehicle interaction systems," tech. rep., 2016.

[7] R. N. Green, A. German, E. S. Nowak, D. Dalmotas, and D. E. Steward, "Fatal injuries to restrained passenger car occupants in Canada: Crash modes and kinematics of injuries," *Accident Analysis & Prevention*, vol. 26, no. 2, pp. 207–214, 1994.

[8] Stats SA, "Statistical Release P0441 - Gross Domestic Product (GDP)," 2019.

[9] Mine Health and Safety Council, "Mine Health and Safety Act 29 of 1996 and Regulations," 2018.

[10] H. A. Hamersma, P. S. Els, and C. E. Doran, "Evaluation of trackless mobile machine collision management systems," in *Mining Goes Digital: Proceedings of the* 39th *International Symposium 'Application of Computers and Operations Research in the Mineral Industry' (APCOM 2019), Wroclaw, Poland, 4-6 June 2019*, pp. 627–635, 2019.

[11] D. M. Gavrila, “Lecture notes in ME41105 Intelligent Vehicles,” October 2018.

[12] T. M. Ruff and D. Hession-Kunz, “Application of radio-frequency identification systems to collision avoidance in metal/nonmetal mines,” *IEEE Transactions on Industry Applications*, vol. 37, no. 1, pp. 112–116, 2001.

[13] R. D. Olney, R. Wragg, R. W. Schumacher, and F. H. Landau, “Collision warning system technology,” in *Steps Forward. Intelligent Transport Systems World Congress, Yokohama*, pp. 1138–1145, 1995.

[14] F. D. Salim, S. W. Loke, A. Rakotonirainy, B. Srinivasan, and S. Khrisnaswamy, “Collision pattern modeling and real-time collision detection at road intersections,” in *Proceedings of 2007 IEEE Intelligent Transportation Systems Conference*, 2007.

[15] S. Ammoun and F. Nashashib, “Real time trajectory prediction for collision risk estimation between vehicles,” in *Proceedings of 2009 IEEE* $5^{th}$ *International Conference on Intelligent Computer Communication and Processing*, 2009.

[16] M. Sepulcre, J. Gonzalvez, and J. Hernandez, “Cooperative vehicle-to-vehicle active safety testing under challenging conditions,” *Transportation Research Part C: Emerging Technologies*, vol. 26, pp. 233–255, 2013.

[17] Y. Zhang, E. K. Antonsson, and K. Grote, “A new threat assessment measure for collision avoidance systems,” in *2006 IEEE Intelligent Transportation Systems Conference*, pp. 968–975, 2006.

[18] R. Sengupta, S. Razaei, S. E. Shladover, D. Cody, S. Dickey, and H. Krishnan, “Cooperative collision warning systems: Concept definition and experimental implementation,” *Journal of Intelligent Transportation Systems*, vol. 11, no. 3, pp. 143–155, 2017.

[19] B. Alrifaee, M. G. Mamaghani, and D. Abel, “Centralized non-convex model predictive control for cooperative collision avoidance of networked vehicles,” in *Proceedings of 2014 IEEE International Symposium on Intelligent Control*, 2014.

[20] M. Jalalmaab, *Model Predictive Control of Highway Emergency Maneuvering and Collision Avoidance.* PhD thesis, University of Waterloo, 2017.

[21] A. Bemporad, F. Borrelli, and M. Morari, *Predictive Control for Linear and Hybrid Systems.* Cambridge University Press, 2017.

[22] S. Lefevre, D. Vasquez, and C. Laugier, “A survey on motion prediction and risk assessment for intelligent vehicles,” *ROBOMECH Journal*, vol. 1, no. 1, pp. 1–15, 2014.

[23] J. C. Gerdes and S. M. Thornton, “Implementable ethics for autonomous cars,” in *Autonomous Driving: Technical, Legal and Social Aspects* (M. Maurer, J. Gerdes, B. Lenz, and H. Winner, eds.), pp. 87–102, New York: Springer, 2015.

[24] N. J. Goodall, “Machine ethics and automated vehicles,” in *Road Vehicle Automation* (G. Meyer and S. Beiker, eds.), pp. 93–102, New York: Springer, 2014.

[25] S. P. Baker, B. O’Neill, W. Haddon, and W. B. Long, “The injury severity score: A method for describing patients with multiple injuries and evaluating emergency care,” *The Journal Of Trauma*, vol. 14, no. 3, pp. 187–196, 1974.

[26] J. S. Sampalis, S. Boukas, and A. N. abd A. Lavoie, "Preventable death classification: Interrater reliability and comparison with ISS-based survival probability estimates," *Accident Analysis & Prevention*, vol. 27, no. 2, pp. 199–206, 19951.

[27] A. Sobhani, W. Young, D. Logan, and S. Bahrololoom, "A kinetic energy model of two-vehicle crash injury severity," *Accident Analysis & Prevention*, vol. 43, no. 3, pp. 741–754, 2011.

[28] L. Evans, "Driver injury and fatality risk in two-car crashes versus mass ratio inferred using Newtonian mechanics," *Accident Analysis & Prevention*, vol. 26, no. 5, pp. 609–616, 1994.

[29] L. Evans and M. C. Frick, "Mass ratio and relative driver fatality risk in two-vehicle crashes," *Accident Analysis & Prevention*, vol. 25, no. 2, pp. 213–224, 1993.

[30] B. A. McLellan, S. B. Rizoli, F. D. Brenneman, and B. R. Boulanger, "Injury pattern and severity in lateral motor vehicle collisions: a Canadian experience," *The Journal of Trauma*, vol. 41, no. 4, pp. 708–713, 1996.

[31] F. Labuschagne, E. de Beer, D. Roux, and K. Venter, "The cost of crashes in South Africa 2016," in *Proceedings of the* $36^{th}$ *Southern African Transport Conference*, 2017.

[32] F. Borrelli, D. Subramanian, A. U. Raghunathan, and L. T. Biegler, "MILP and NLP techniques for centralized trajectory planning of multiple unmanned air vehicles," in *Proceedings of the 2006 American Control Conference*, 2006.

[33] C. Frese and J. Beyerer, "A comparison of motion planning algorithms for cooperative collision avoidance of multiple cognitive automobiles," in *2011 IEEE Intelligent Vehicles Symposium (IV)*, 2011.

[34] J. Holt, S. Biaz, and C. A. Aji, "Comparison of unmanned aerial system collision avoidance algorithms in a simulated environment," *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 3, pp. 881–883, 2013.

[35] M. G. Earl and R. D'Andrea, "Iterative MILP methods for vehicle-control problems," *IEEE Transactions on Robotics*, vol. 21, no. 6, pp. 1158–1167, 2005.

[36] IBM Corp., *IBM ILOG CPLEX Optimization Studio CPLEX User's Manual*, 2015.

[37] M. Abe, *Vehicle Handling Dynamics: Theory and Application.* Butterworth-Heinemann, 2009.

[38] T. D. Gillespie, *Fundamentals of Vehicle Dynamics.* SAE International, 1992.

[39] C. Ericson, *Real-Time Collision Detection.* Elsevier, 2005.

[40] Federal Aviation Administration, *Order JO 7110.65W.* U.S. Department of Transportation, 2015.

[41] R. J. Thompson, "Mine haul road design, construction & maintenance management," tech. rep., ASPASA, 2011.

[42] T. Schouwenaars, B. Mettler, E. Feron, and J. How, “Hybrid model for trajectory planning of agile autonomous vehicles,” *Journal of Aerospace Computing, Information, and Communication*, vol. 1, no. 12, pp. 629–651, 2004.

[43] J. Holt, “Comparison of aerial collision avoidance algorithms in a simulated environment,” Master’s thesis, Auburn University, 2012.

# Glossary

## List of Acronyms

| | |
|---|---|
| **CAS** | Collision Avoidance System |
| **DENM** | Decentralized Environmental Notification Message |
| **EMERST** | Earth Moving Equipment Safety Roundtable |
| **GDP** | Gross Domestic Product |
| **ISS** | Injury Severity Score |
| **LIDAR** | LIght Detection And Ranging |
| **LTI** | linear time-invariant |
| **LTI-MA** | linear time-invariant maneuver automaton |
| **MHSC** | Mine Health and Safety Council |
| **MILP** | Mixed Integer Linear Programming |
| **NLP** | Non-Linear Programming |
| **PDOF** | Principal Degree of Force |
| **RADAR** | RAdio Detection And Ranging |
| **RFID** | Radio Frequency Identification |
| **RHC** | Receding Horizon Control |
| **SOTA** | State Of The Art |
| **TMM** | Trackless Mobile Machine |
| **TTC** | Time To Collision |
| **UDP** | User Datagram Protocol |
| **UAV** | Unmanned Aerial Vehicle |
| **V2P** | Vehicle To Pedestrian |
| **V2V** | Vehicle To Vehicle |
| **VDG** | Vehicle Dynamics Group |
| **VS** | Vehicle Sensor |