CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Designing Driver Assistance for Racing Games

Guidelines for creating driver assistance for aiding less experienced players

Master's thesis in Computer Science and Engineering

JENNY ORELL
ULRIKA UDDEBORG

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2019

Master's thesis 2019

Designing Driver Assistance for Racing Games

Guidelines for creating driver assistance for aiding less experienced players

JENNY ORELL
ULRIKA UDDEBORG

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2019

Designing Driver Assistance for Racing Games
Guidelines for creating driver assistance for aiding less experienced players
JENNY ORELL
ULRIKA UDDEBORG

Supervisor: Staffan Björk, Department of Computer Science and Engineering
Advisor: Mattias Adolfsson, Ghost Games
Examiner: Olof Torgersson, Department of Computer Science and Engineering

Master's Thesis 2019
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2019

Designing Driver Assistance for Racing Games
Guidelines for creating driver assistance for aiding less experienced players
JENNY ORELL
ULRIKA UDDEBORG
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Mechanics and systems for assisting players can be found in many modern video games. In racing games, examples of assistances are assistance systems for braking or accelerating, that removes or lessens the need for players to perform these actions. This thesis investigates what aspects and factors should be considered when designing driver assistance in a racing video game, particularly aimed at less experienced players.

This project was executed using an iterative design process, which consisted of three iterations. This resulted in two different driver assistance prototypes, as well as a set of guidelines for driver assistance in racing games, based on pre-existing theory and our findings though playtests and other feedback.

Keywords: video game, racing game, assistance, driver assistance, player experience, game design.

Acknowledgements

Firstly, we want to thank our academic supervisor Staffan Björk for guiding us through this thesis. We also want to thank the employees of Ghost for their help, in particular Nicolas Mercier and our advisor Mattias Adolfsson. We also want to thank everyone who participated in our feedback sessions and playtests as well as others who have supported our work.

Jenny Orell & Ulrika Uddeborg, Gothenburg, June 2019

Contents

1

Introduction

Video games today are highly complex in regards to gameplay, with many different mechanics and elements for creating immersion through a deep gaming experience. When playing video games, players have different purposes and skill levels. Some players may only be interested in the story of a game, while others want to master the game. In order to attract a large audience, games therefore need to consider players' different strengths and goals. Many games do this through the help of different assistance systems and mechanics. One example is more passive assistance in the form of difficulty selection where the players select one out of several predefined difficulty levels determining the overall difficulty of the game. A more active assistance system is for example aim assistance [1], where the game instead helps the player aim during the game in real-time.

In racing games, these assistances can be mechanics such as the previously mentioned difficulty selection, visual assistances that indicates where to drive, or more active assistances that lessens or removes the need for the player to do some part of the vehicle handling, such as accelerating or braking. Today, there are some flaws with these systems. Static difficulty adjustment is often not adequate for offering different skill levels as players have difficulty accurately assessing their own skill level [2]. The more active assistance mechanics can instead be too obvious to the player, making the experience feel artificial [3] or unfair [1], and offering little to no challenge for the player if assisting too much.

In this thesis, we want to further explore the possibilities of driver assistances in racing video games, particularly what aspects would be beneficial for players with less experience with racing games.

1.1 Purpose

The purpose of this thesis is to investigate the possible methods for giving players driver assistance in a racing game. This could have a positive impact on the player experience, especially regarding players of lower skill levels. This will be done by developing one or more prototypes in collaboration with the game developer Ghost Games, who have experience in developing racing games to answer the following research question:

"What factors should be considered when implementing driver assistance in a racing video game to aid players with less experience?"

The player experience of the prototypes will be evaluated and compared with the non-assisted experience to see what aspects and methods can give an improved player experience and further player enjoyment. The final result of the thesis will be guidelines on what to consider when implementing driver assistance for racing games, and prototypes demonstrating driver assistances.

1.2 Delimitations

The thesis will not focus on giving the player assistance in an open world environment. The thesis will instead be limited to a more closed race setting, constrained to selected race tracks. In the open world, players' desired actions are far more unpredictable and it is likely that designers want to allow the player large degree of freedom. The thesis will also not look at multiplayer experiences, only single-player settings, as that brings up issues of fairness and balancing between players. Additionally, it will focus on players of age 12 and up, since this seems to be a common age rating for this type of games.

2

Background

This chapter covers previously existing work related to the thesis, such as real-world driver assistance. The chapter also describe driver assistance mechanics present in current racing games. The games and game series covered are *Mario Kart 8 Deluxe*, *Real Racing 3*, the *Need for Speed* franchise, and the *Forza* franchise.

2.1 Real-life vehicle driver assistance

Today's real-world vehicles come with many features for improving both the driving experience and the security of the driver. This is done by reducing or fully eliminating errors caused by the driver [4] or by other types of vehicle assistance such as electronic stability control and anti-lock braking. The latter have the purpose of avoiding accidents during critical situations, in comparison to the former which instead help to avoid accidents by continuously providing assistance to the driver [5].

Another type of driver assistance is autonomous vehicles helping the driver by automating a subset of or all the vehicle's systems. Autonomous vehicles are commonly divided into six different categories ranging from 0 to 5, depending on their level of automation where level 0 is no automation and level 5 is full automation. Level 1 and 2 requires the driver to be fully engaged during the whole task, but the vehicle features partial automation such as partly automated acceleration or steering. For levels 3, 4, and 5, some or all driver tasks are fully automated, where at level 5 all systems are fully automated making the vehicle self-driving without any human driver [6].

For drivers of real-life vehicles, Jeon [7] has identified that driver personality and mental state might influence their behaviour in different driving situations. For example, anxious drivers such as novice drivers risks getting tunnel vision while focusing on single tasks [7, p. 562], while drivers in a negative mood has an increased risk of causing accidents [8, 9].

2.2 Assistance mechanics in racing games

There are several ways that players can be aided during gameplay, taking advantage of video games' digital medium. Examples of this include visual cues on the screen, aim assistance, and difficulty settings. In multiplayer games, balancing can be used to assist players and is often used when players of different skill levels wishes to compete with or against each other. For example, balancing can be added by assisting poorly performing players by giving them an advantage and is an important part when designing a game [10]. When balancing games, the aim is to provide a satisfying experience to all players despite the skill gap [11].

Some assistance mechanics in real-life vehicles are already present in existing video games, such as automatic transmission that removes the need to shift gears. Unlike assisted activities in many other games, such as aim assistance for shooting [1], assisted activities in racing games are mostly continuous instead of discrete. This enables small, less noticeable assistances [11].

2.2.1 Mario Kart 8 Deluxe

Mario Kart 8 Deluxe [12] for Nintendo Switch, released in 2017, is the latest game in the *Mario Kart* franchise. *Mario Kart* is a racing video game series published by Nintendo [13]. The game's races are done on circuit race tracks, with clear limitations of the edges of the tracks as shown in figure 2.1. Going outside the track edges, if not limited by physical obstacles directly by the track, leads to poor terrain, walls or drops, serving as barriers to encourage the player to keep on track.

Figure 2.1: A race in *Mario Kart 8 Deluxe* on a closed track. Source: Adapted from [14]

The game frequently makes use of different types of player assistance, for example player power-ups. Typically, the better power-ups are given to players that are

behind in the race and rarely offered to the players in the lead, balancing player positions and performance. *Mario Kart 8 Deluxe* also features two player assists that directly impacts the player's steering and manoeuvring: *Smart Steering* and *Auto Acceleration*. Smart Steering helps the player keep to the middle of the race track and to avoid bumping into walls when not turning enough. Auto Acceleration allows the player to drive without having to use the acceleration button, albeit slower than if the player accelerates on their own.

2.2.2 Real Racing 3

Real Racing 3 [15] is a free-to-play mobile game developed by Firemonkeys Studios [16] published by Electronic Arts. For PC and console racing games, the player usually steers their vehicle with a handheld controller or with mouse and keyboard. As *Real Racing 3* is a mobile game, the vehicle is instead steered by tilting the device left or right as if the player were holding a steering wheel. Unlike most racing games, the player never have to accelerate. This is probably due to *Real Racing 3* being a mobile game with more limited precision and input modes. The game is shown in figure 2.2.

Figure 2.2: A race in the mobile game *Real Racing 3*. Source: Adapted from [15]

The game features three types of driver assistance for the player: steering, braking and traction control. Steering and braking assistance has three different levels of assistance: high, low, and no assistance at all. The player can choose to have traction control either on or off. The different levels of steering assistance helps the player keep their car on the track. High steering assistance makes it less likely but not impossible, to drive off the track, while low assistance also helps, but to a lesser degree. High braking assistance allows the player to completely ignore braking, while low brake assistance requires some braking from the player to perform well

but still a lot less than with the braking assistance turned off. Traction control helps the player avoid skidding, which slows the car down.

2.2.3 The *Need for Speed* franchise

Need for Speed is a multi-platform racing game franchise published by Electronic Arts [17] with its first release in 1994, and is currently developed by Ghost Games [18]. The series' latest game *Need for Speed Payback* [19] was released in November 2017. The later instalments of the series has been open world games with both single-player and multiplayer elements. *Need for Speed Payback* features a ranked mode, where players compete against each other trying to increase their rank.

Need for Speed Payback contains some features for assisting the player during gameplay. The game has a mini-map, showing the player's surroundings in top-view, where the shortest road towards the player's selected destination is highlighted. When the player is doing races or other activities checkpoints in the form of arches are shown on the road, indicating the way to the player. In races with sharp turns, there are arrows signalling this to the player. These features can be seen in figure 2.3.

Figure 2.3: A race in *Need for Speed Payback* where the player is competing against computer controlled characters. Source: Adapted from [19]

An assistance feature present in *Need for Speed (2015)* is a visual assistance that shows a suggested path to the player during a race, which could not be hidden by the player. This can be seen in figure 2.4. This feature was not present in the succeeding game *Need for Speed Payback*.

Figure 2.4: The visual assistance line found in *Need for Speed (2015)*. Source: Adapted from [20]

2.2.4 The *Forza* franchise

The *Forza* franchise, published by Microsoft Studios [21] for Xbox and Microsoft Windows, is a racing game franchise divided into two series: *Forza Motorsport* [22] and *Forza Horizon* [23]. *Forza Motorsport* is a more realistic racing game, closer to a simulation. *Forza Horizon* has an open-world arcade environment, more similar to *Need for Speed*.

Figure 2.5: Image from *Forza Horizon 4*. Source: Adapted from [24]

Both *Forza* games have an option for enabling a visual assistance line similar to that found in *Need for Speed (2015)* (see fig. 2.5). Unlike in *Need for Speed (2015)*

that line, this line also shows when to accelerate or brake by shifting colour. *Forza Horizon 4* also features visual assistance in the form of a mini-map of the nearby road network, fixed to the player and slightly tilted so that the player can better see what's ahead. The game also features a GPS voice assistant, similar to the ones for real-world vehicles. This system gives the player instructive audio prompts such as the direction to turn in an upcoming junction.

Both *Forza* series let the player choose a difficulty level that regulates several options for assisting the player, for example braking, steering, stability control, and gear shifting. These level of assistances lower the skill and effort required by the player, by performing all or part of the aspect, similarly to those found in *Real Racing 3* but with assistance options that are closer to those found in real-world vehicles. At the maximum assistance level, the car drives entirely after the suggested line and only requires accelerating. With this level of assistance it is very difficult to take over the steering even if desired by the player without decreasing the amount of assistance given. If the player wishes to, they can change these assistance mechanics individually to better suit their needs.

3

Theory

In this chapter, models, concepts and frameworks that are relevant to the study of this thesis are presented.

3.1 Research through design

Research through design is a term for describing how the design process itself can serve as a part of doing research [25, 26]. In the interaction design and HCI research community, research through design has become increasingly common. Through reflecting on the design process and the resulting designs, the researcher can gain various forms of insights on the problem at hand [25].

To do this, the researcher first performs secondary research, by studying research and conclusions made by others [26]. This is then combined with exploratory research in the form of design. A large variety of methods can be used to ideate, experiment and evaluate [26]. The result is often artefacts or systems, but can also be other forms of results such as methods or frameworks [25]. Hanington and Martin [26] argue that the most important artefact from research through design is documentation, which communicates design actions and puts them into context.

3.2 Player motivation and enjoyment

Ryan and Deci [27] splits up motivation into two types, *extrinsic motivation* and *intrinsic motivation*. Intrinsic motivation comes from within, for example a wish to explore new challenges or to learn new things. Extrinsic motivation are instead motivation from external sources, such as a bribe or pressure from someone else [28]. Play is usually considered an intrinsically motivated activity [29].

Every player of a game has a different background in regards to game experience and a different skill level. To ensure that a player enjoys the game and stays motivated to play, the game needs to be difficult enough to be a challenge for the player, but still take into account their skill level to avoiding making the game feel impossible for them [30]. Sweetser and Wyeth [30] claims that player enjoyment can be considered the most important goal of a computer game.

3.2.1 Flow

Flow refers to a mental state that happens when a user is interested and attentive to an intrinsically motivated activity, without being distracted or anxious [29, 31]. Providing the player with an appropriate challenge is often considered a very important part of game design and flow in games, and the game should provide a sufficiently difficult challenge for the player while still matching their level of skill [30]. A lack of this balance may result in players becoming bored and uninterested as well as break the flow [30, 32]. The level of challenge needed to achieve flow can be different for different players, despite the players playing the same game [33].

GameFlow is a game specific model of flow proposed by Sweetser and Wyeth [30], which focuses on flow in games and player enjoyment. They propose eight aspects that promotes player enjoyment and flow: *concentration*, *challenge*, *player skills*, *control*, *clear goals*, *feedback*, *immersion*, and *social interaction* [30]. All of these but social interaction is directly corresponding to an aspect described by the original flow theory.

Concentration is described as a requirement of focus on the game played, and that the game therefore also should allow it. *Challenge* is that the challenge in the game should be appropriate in regards to the player's skill level, making it neither too easy nor too difficult. The *player skills* aspect says that games shall allow for player improvement and mastery of the game. *Control* in GameFlow is that during gameplay, players should feel in control of their own actions. *Clear goals* is described as providing clear and meaningful goals for the player. *Feedback* means giving the player suitable feedback for events. *Immersion* says that players should feel connected and engaged during gameplay. Finally, *social interaction* is described as providing social interaction as well as enabling opportunities for it [30].

3.2.2 Self-determination theory

Self-determination theory is a motivational theory that builds upon this, and claims that motivation is based on three needs: *competence*, *relatedness* and *autonomy* [28]. The need for competence consists of wanting to be in control and to achieve mastery of the environment [29, 34]. Relatedness is the need to feel connected with others [28]. Autonomy is described by Deci and Ryan [35, p. 154] as "the experience of choice". This concerns making a meaningful choice and having free will [29], differing from merely making a decision, as in order to achieve autonomy a choice is required to allow for the freedom to truly consider other options [35]. As these needs are fulfilled, they lead to greater self-motivation [28].

3.3 Artificial intelligence in games

When describing artificial intelligence, AI, in computer science, Russell and Norvig [36] defines it as "the science of agent design". A rational agent is something that

perceives its environment and makes the choice that maximises the possible output [36]. Millington and Funge [37] on the other hand describes artificial intelligence as "making computers able to perform the thinking tasks that humans [...] are capable of". In academia, there are multiple motivations for researching and investigating artificial intelligence. Millington and Funge [37] divides these motivations into three groups: philosophy (understanding thought and intelligence), psychology (understanding the brain) and engineering (making algorithms perform tasks like humans). In game development, the engineering is the main area of interest, as game developers want to build game characters that appear human-like [37].

Millington and Funge [37] also identifies three parts of game AI: movement, decision making and strategy. Movement algorithms turn decisions done by the AI into movement, decision making selects a behaviour to perform, and strategy is a group approach where the decisions of a group of AI characters are influenced by the same algorithm. Unlike the other two aspects, strategy is generally not considered necessary for all games [37]. These parts are then built into its own system, and connected and interfaced to the game world by AI programmers.

These systems does not necessarily need to be complex to be good, but that what players consider good AI is about creating the right behaviours that does not end up appearing silly or stupid. Understanding what the players perceive and expect can be the key to giving AI characters an illusion of intelligence [37, 38].

3.3.1 Steering and movement

To get more complex movement behaviours, simple behaviours can be combined into a larger system by the use of blending or arbitration [37]. Blending performs all behaviours and then combines them to a single result by weighting or prioritising the input from each behaviour. There are some possible problems with this method, for example two behaviours may conflict with each other, which can lead to the character doing nothing [37]. Arbitration selects some behaviours that gets to control the AI character. Unlike blending, this method tries to take the context into account and select the best course of action for this context. In regards to steering, this method often becomes complex and in those cases less efficient.

3.3.2 Pathfinding

Pathfinding in software development is a technique commonly used for finding walkable routes in a digital environment such as in games, both for a player's character and characters controlled by a computer. Pathfinding can also be used for problem solving and decision making, for example how to reach a certain outcome based on available actions and their outcome [37]. There exists several different pathfinding algorithms with different aims, where some result in the optimal solution while others specialise in finding a solution quickly, or paths that are good enough to be convincing [39]. Which algorithm to use can also depend on the context; for example, a 2D or 3D world, or a state based system.

A common pathfinding algorithm is the A* algorithm, which is used for finding the optimal path. It is popular for pathfinding in games as it is efficient and simple to implement [37]. However, the computation time of this algorithm grows significantly with the world complexity. For complex worlds where the pathfinding needs to be fast, the pure A* algorithm is not a feasible solution [40, p. 270]. Often, an optimised version of the A* algorithm is used for pathfinding [37].

3.4 Machine learning

Machine learning is a set of methods for recognising patterns in data, and provides automated analysis of huge quantities of data. Found patterns are then used to predict future data, or can be used as a basis for decision making. Examples of machine learning applications include filtering spam mail, image classification, and face detection and recognition [41]. Machine learning has also been used in games to allow the game's AI to learn about the player and adapt to their playstyle. This has been a quite uncommon occurrence in commercial games, but was used as early as 1997 [42].

There are typically two types of machine learning techniques: *supervised* and *unsupervised* learning. Supervised learning is the most commonly used form of machine learning. It uses a given training set consisting of input and corresponding output and uses the input-output pairs to learn a mapping based on the patterns detected in the training set. Supervised learning can be further divided into two types: *classification* for discrete prediction models, and *regression* for continuous prediction models. A classification model gives outputs within a finite set such as a Boolean value or whether a player will win or lose a game. A regression model give continuous output, for example, vehicle speed. Unsupervised learning, on the other hand, recognises data patterns only from the given input, without a predefined target output. Therefore, finding a certain desired type of patterns could be impossible using unsupervised learning, since there is no particular mapping or pattern to compare results to [41].

3.5 Dynamic difficulty adjustment

A common solution for supporting players of different skill levels in video games is to offer difficulty adjustment in the form of difficulty settings [10], where the player can manually choose on which level of difficulty they wish to play.These are often static, meaning that the level of assistance is set at the start of the game and is never adjusted during the gameplay [10]. This static difficulty adjustment is often not enough for representing different skill levels as players tend to have difficulty with accurately determining their own skill level [2]. This may give an inappropriate amount of assistance, which can result in a poor player experience. If the game is poorly balanced for the player, it can result in frustration [10] and the player's flow may be interrupted [33].

A possible alternative to static difficulty settings is dynamic difficulty adjustment, DDA, also called adaptive difficulty. This means that the agents in the game (often non-player characters) adjust their skill level to give the player an appropriate challenge according to the player's performance. This has been investigated for several different types of games, such as multiplayer online battle arena (MOBA) games and computer roleplaying games [43, 44].

Players often prefer human opponents over computer controlled opponents because the AI controlled opponent seem too unintelligent and un-humanlike in their actions [45]. Additionally, non-human opponents might be too intelligent in that they do not make any mistakes, for example, a computer controlled vehicle in a racing game that never crashes even though it is driving through the track at very high speed [46, p. 325]. By using DDA, an advanced AI agent could adjust by worsening its own performance to give players an enjoyable experience. Liu et al. [2] believes that a dynamic difficulty system should consider player performance and personality as well as the game context, which could potentially make the AI more human-like. They also caution that even though a player might be defeated multiple times, lowering their average performance level, one should be careful with decreasing the difficulty level in such situations, as players might find enjoyment and satisfaction from completing difficult tasks and thus want to keep the challenge [2].

Most DDA systems focus on player performance, but other variants has also been investigated. Liu et al. [2] designed a DDA system for games that analyses the player's anxiety level in real-time and the game's difficulty level adapts based on this. It was found that most of the players participating in the experiment got a better experience when the game featured this affect-based DDA system.

4

Methodology

This section contains several design processes and methods and aims to give a broader perspective on available methods, how to conduct them and why one would use them. The methods listed in this section have been structured into the phases that have been used during this project: analysis, ideation, prototyping, and evaluation. Some of the methods can be used in multiple stages for different purposes in the process, but have been placed into the phase where they were used during our thesis. The methods that are relevant but never used are placed in the phase where they were planned or considered to be used.

4.1 Iterative design

Many problems in interaction design has to do with the user experience and how the user perceives the design. Similarly, problems in game design often regard the player experience. These are often what Rittel and Webber [47] would call wicked problems. A wicked problem is a problem with no definitive problem statement which cannot be solved with a traditional, linear approach as the understanding of the problem changes when possible solutions are discovered. It is the wicked problems' social complexity that makes them so difficult to define, understand and find solutions for [47]. Each problem can be approached in multiple ways leading to different designers coming up with different solutions [37]. Solutions to a wicked problem are never inherently right or wrong; rather they are compared to other solutions and evaluated as better or worse, or whether they are good enough [47].

Many authors suggest an iterative process when working with problems in game and interaction design [46, 48, 49], but there are many different ways of structuring and organising an iterative process.

4.1.1 The Wheel

The Wheel is an iterative process for designing user experiences, defined by Hartson and Pyla [48] in *The UX Book*. It is a very versatile design process as it can be applied to different kinds of design, for example technical design such as hardware design or for coming up with new design concepts for a product. The process is divided into four steps, or activities: *Analyse*, *Design*, *Prototype*, and *Evaluate*.

The Analyse activity consists of understanding the domain and the users' behaviours and needs and the Design activity means creating design concepts and behaviours from the requirements and needs gathered from analysis. Prototype consists of prototype creation, which is often done simultaneously to design, as the designer further reflects on the design while creating the prototype. The Evaluate activity means testing the design against user goals and requirements. This process encourages iterations, both within the four steps, but also between them. The process also encourages going back to the previous activity if needed, without going through the intermediate steps.

4.1.2 Playcentric design process

Fullerton [46] proposes an iterative design process called the *playcentric design process* for creating games. In this process, Fullerton [46] advocates for a strong focus on players and player experience.

The process is divided into seven different steps: brainstorming, physical prototype, presentation (optional), software prototype(s), design documentation, production, and quality assurance. The presentation step is usually performed with the goal of securing funding for the game, if needed, and can be skipped otherwise. In each step of the development of the game, the designers work iteratively: generating and formalising ideas, testing them and then evaluating the results. This loop repeats until the result is considered successful. In the early foundational stages of a prototype, Fullerton [46] suggests performing self-testing by the designers and then moving on to testing the prototype with external participants [46, p. 272].

The first step, brainstorming, starts with ideation of concepts and mechanics and setting up "player experience goals", goals that describe the desired player experience, and ends with evaluation of the product according to these goals. In the next two steps (skipping the presentation step), various kinds of prototypes are created, both physical and digital. The digital prototype development phase can end in several digital prototypes, where each digital prototype demonstrates different gameplay concepts. Each prototype, physical or digital, should then be evaluated against the player experience goals.

After, and possibly during, prototyping and refining the design, Fullerton [46] proposes design documentation: gathering knowledge and insights gained during the prototype development and developing a list of goals for the game from this. Then, the production phase begins, where the main product is developed in sprints. It is suggested to use an agile development method such as Scrum (see sec. 4.4.1). In production, iterative cycles should still be performed in order to keep the playcentric focus.

The final step is quality assurance, where the software prototype(s) are evaluated according to the player experience goals that were set at the beginning of the process.

4.2 Analysis

In this section, a number of methods suitable for the Analysis phase of the Wheel [48] is presented. In the Analysis phase, the goal is to analyse and gain knowledge of the business domain as well as the users and their needs.

4.2.1 Literature review

Literature review is a method of secondary research, in which the researcher uses existing published sources to gather and summarise interesting previous findings and research [26, p. 112]. These sources can be, for example, academic papers and journals, books, news articles, or blog and forum posts. Some caution should be used when selecting references, especially concerning non-peer reviewed sources such as websites and blog posts [26, p. 112]. However, literature reviews for design purposes often include a large range of different types of sources [26, 49].

4.2.2 Competitive testing

Competitive testing is a method where one tests competitors' products in the same way that one tests or plans to test their own product [26, 50]. Nielsen [50] recommends testing three to four competitor designs. This can be used both as background research in the early stages of designing as well as in the very end of the design process, when evaluating a design [26, p. 36]. When used before starting the design development, the researcher can gain insights that can be used later during the development [50].

4.2.3 Grounded theory

Grounded theory, or the grounded theory method, is a data-driven, iterative method for developing new theories [51, 52]. Unlike many other analysis methods, grounded theory doesn't start with a pre-defined hypothesis formed from reviewing existing literature, but instead starts and focuses on data collection and forming a theory through categorising the collected data. After forming a theory, literature is reviewed to ensure that the theory also has theoretical support [51].

The process for grounded theory starts with identifying the area to research, which often contains a target group to research, and then beginning the data collection. Data collection can be done, for example, by observing an activity or talking and discussing with individuals or a group. While gathering the data, the researchers performs "open coding". For each data point, the researchers should note possible meanings and associations. From this, each point of data collected is given a code, where a code is usually a phrase or a word. These codes are then further abstracted into "concepts", "higher categories" and "core categories" gradually during

the data collection process. Each abstraction collects a number of previous groupings by grouping by similarities. While performing this work, the researchers record challenges encountered while working, especially notes about codes and their relationships to each other. The data collection and coding is continued until theoretical saturation is achieved, i.e. when the gathered data stops yielding new codes and concepts. From this data, researchers form a theory. Finally, the researchers take their theory and review relevant literature to ensure that their theory has support in literature as well as in the data [51].

4.2.4 Content analysis

Content analysis is a qualitative method for systematic interpretation of data. The purpose of the method is to compile detailed data, such as interview transcripts and observations, into data points that are easier to analyse. These data points are usually referred to as "codes". These codes are grouped into categories based on affinities. There are two approaches to this: inductive, which is the most common approach, and deductive. In inductive content analysis, the categories are created during the analysis while in the deductive approach, the categories are decided beforehand [26, p. 40]. The categories form more overarching findings from the original data. This process is very similar to the grounded theory method (see sec. 4.2.3), but does not end with the forming of a theory.

4.2.5 Affinity diagramming

Affinity diagramming is a method for visualising and categorising data. After having gathered data, possibly in the form of insights, observations or requirements, the data is typically written on post-it notes and put up on a board. The notes are then discussed and moved around jointly by the designers and grouped based on common patterns and themes, starting off with smaller themes and then grouping them together further to create overarching themes [26, p. 12].

4.2.6 Observation

People are often unable to identify and evaluate their own behaviours, especially when the behaviour reflects poorly on them [49]. By observing and recording the actions and behaviour of people or artefacts the researcher can identify common behaviours, themes or patterns [49]. Observations can be semi-structured or structured [26, p. 120]. Semi-structured observation is more open, and allows for deviation from the guiding questions. This is typically used for the exploratory stage of the design process [26, p. 120]. Structured observation is more formal, with large degrees of pre-prepared structuring such as checklists or other forms of codifying. This is well suited to investigate clear and well defined elements. Existing frameworks can also be used to structure observations, such as *AEIOU* [26, p. 10] which organises ob-

servations into five categories: *activities*, *environments*, *interactions*, *objects* and *users*.

Fly-on-the-wall observation [26, p. 90] is another type of observation, where the researcher is not directly involved with the observed person or artefact. This aims to reduce potential influences caused by the researcher being present. This can be done with *secret outsiders* or *recognised outsiders*. Secret outsiders are distant and unseen by the participants, while recognised outsiders are known by the participants, but attempts to be as unobtrusive and distant as possible.

4.2.7 Interviews

Interviews are a method of survey research [26, p. 102]. Interviews can be structured, following a script, or more unstructured, allowing for more flexible conversation during the interview but still often with prepared guiding questions. With structured interviews, it is often easier to get the participants to give answers to the researcher's underlying design inquiry. When doing more unstructured interviews, the researcher often needs to guide the conversation more to get answers that focus on the initial inquiry. Interviews can be done either in a group setting or individually. Group interviews often facilitate a more natural conversation, but the participants may also influence each others opinions and answers. Interviews can be used for both exploratory and evaluative purposes, and are often used with a complementary method such as observations (see sec. 4.2.6) or questionnaires (see sec. 4.5.2).

4.2.8 Personas

Personas are user archetype descriptions that aims to represent the core user group of the design [26, 29, 49]. To create personas designers should gather information from the target audience and then analyse the data and their needs and look for common patterns and themes. From this, designers can create one or more fictional people that represent the target audience [26, 29]. These personas are then used as a reference throughout the design process, for example, when developing and discussing the design. It is important to note that while the personas are often given names, they are not a real person and designers should be cautious to not develop the product only for the persona, but for the real users [29].

4.3 Ideation

The methods described in this section were considered or used in this study for idea generation and idea selection.

4.3.1 Brainstorming

Brainstorming is a method for group ideation [53]. It consists of three phases: fact-finding, idea-finding and solution finding. In the fact-finding phase the problem is defined and the group prepares by gathering and analysing data and information that may be relevant. The idea-finding phase consists of coming up with ideas, and for this phase four guidelines should be followed: no criticising of ideas, no idea is too crazy, the more ideas the better, and suggest improvements for given ideas [53]. The final phase, solution finding, is to evaluate the ideas and solutions and decide on what to proceed with.

4.3.2 Mind mapping

Mind mapping is a visual method for idea generation that enables non-linear, visual organisation of a problem space [26, p. 118]. To create a mind map, designers should start by identifying a question that will be the focus of the mind map. Extensions are drawn out from this centre, consisting of simple, short words or phrases. These are called primary connections. Secondary connections go out from these, which consist of deeper information related to the primary connection. This free association should be continued until all relevant information is represented in the mind map [26, p. 118].

4.4 Prototyping

In the Prototyping phase, various prototypes are created to serve as external representations of a design. Prototyping is an important aspect of interaction design and can be done in many different ways. Prototypes can be either physical or digital, and low fidelity or high fidelity and different scenarios require different kinds of prototypes. . A low fidelity prototype is a very simple prototype that attempts to show a concept, while a high fidelity prototype is closer to a final, finished product.

4.4.1 Agile software development

There are many different software development methods and frameworks for working with agile software development [54]. The purpose of these methods is to organise the work process and create coherent team work. Another advantage with having an agile development process is that it becomes easier to adapt to changes during the project. Because of this, several agile methods and frameworks can also be considered iterative processes. Fullerton [46] advocates for using an agile workflow when creating software prototypes, and suggests Scrum as a possible option.

Scrum [55] is a commonly used agile framework for software development. A Scrum team has a designated "Scrum Master" and a "product owner", beyond the regular

team members. The role of the Scrum Master is to lead the team and the working process, while the product owner is responsible for verifying that the product meets its requirements. The Scrum workflow is built up of short sprints of at most 1 month each. After each sprint, the product should be complete and deliverable to a customer. When planning each sprint, the team selects a number of tasks, user stories, from their backlog to complete during the sprint. The backlog is a collection of all possible tasks and changes that the team might be interested in doing.

Another common agile framework is Kanban [56]. Unlike Scrum, Kanban does not have sprints during which selected tasks are to be completed. Instead, Kanban takes all tasks of the project into account whenever planning the next step in the project, but still aims to have a deliverable product as tasks are completed.

4.4.2 Pair programming

Pair programming is an agile software development technique where two programmers sit down together and work at one computer [57]. One programmer is the "driver", the ones who writes the code, while the other is the "navigator" who focuses more on the overall perspective of the task. These roles are often switched fairly frequently, as both programmers should be active and engaged in solving the task.

4.4.3 Voll's game AI development process

Developing video game AI is quite different from developing AI in academia. Voll [38] states that game developers have one job: to create an experience, and that every decision taken must support that experience, including AI design. When designing AI for a game the goal is not to make it incredibly clever, but to solve a problem that cannot be solved in a simpler way. Good AI can enhance many aspects of the gameplay, while bad AI risks ruining player flow and immersion [38].

Voll [38] suggests a process for designing an AI for simulating a human player, that she has previously used successfully to make the users believe they were playing against a real, human player. The process was divided into 3 steps.

For the first step, the designer watches people play the game, to know and understand their behaviour and playstyle in order to make an artificial player behave human-like. It is also important to get an impression on how players expect other human players to play, or what unexpected human behaviour would look like. During this step, Voll [38] identified several player personalities.

After researching the players, the second step is to begin the development by starting with small designs or implementations. This means intentionally starting out with AI behaviour that could potentially be seen as stupid due to its lack of complexity, and then iterate over the concept to increase its complexity and capabilities. By starting off with rather simple behaviours developers can better control their systems, lessening the risk of ending up with an AI that behaves too artificially or

unnaturally by being initially complex [38]. Similarly, Millington and Funge [37] also advises against initial complexity of game AI.

The third and final step is to evaluate the AI. The designer is to identify problematic behaviours and try to adjust and improve them. At this point, Voll [38] warns again about implementing complex solutions. As in many other design and development processes, the designer should iterate over the design to refine it.

4.5 Evaluation

There exist many different methods of performing evaluations, which often aim to measure things such as usability and ergonomics [26]. Different methods focus on different aspects of the design, and are typically used in different stages of the overall design process.

Evaluation of a product can roughly be divided into two different types: summative and formative. A summative evaluation is typically performed last in the process on a final design, for example by testing the design against guidelines or standards. Formative evaluation is instead performed during the development process, by iterating over a design until its goals are fulfilled [58].

When evaluating multiple designs for the same issue, Nielsen [50] recommends to alternate the order of testing the different versions for the different participants as users compare the subsequent version with the previous versions. Below, a number of methods for evaluation are described that may be suitable for evaluating the prototypes and their user experience in this thesis. However, some of the methods mentioned earlier in this chapter can also be used as evaluative methods, such as competitive testing (see sec. 4.2.2), observations (see sec. 4.2.6) and interviews (see sec. 4.2.7).

4.5.1 Think-aloud technique

The think-aloud technique is a very common method for evaluating usability of a design [26, p. 180]. It is a data collection method where the participant testing the design talks about what they are doing and thinking as they complete a given task [26, 48]. The focus should be on what the participant is thinking about what they are doing, not on describing what they are doing. This technique is often used in conjunction with observation (see sec. 4.2.6), to get data on not only the participant's actions, but also their thoughts and feelings from performing the task [48].

4.5.2 Questionnaire

Questionnaires are a method for performing survey research, usually in written form [26, p. 140]. They are simple to create and administer, but attention should be paid

to the wording and presentation of questions. It is important to note that how a question is asked can have a large effect on the responses.

4.5.3 Participatory design

Participatory design is a human-centered design approach that proposes actively involving users and other stakeholders to weigh in on the design during the design process [26, p. 128]. Several different methods can be used for participatory design, which all have in common that they directly involve and consult users and other stakeholders, often face-to-face. Participatory design aims to gather insights and information from co-design activities, which is then interpreted and considered by the designers and used as information and inspiration to further the development of the design [26, p. 128]. Cooper et al. [49] advocates for the use of personas (see sec. 4.2.8) over participatory design, as participatory design only takes into account individual user feedback, while personas are created from larger scale observations of the users.

4.5.4 A/B testing

A/B testing is a method that can be used when there are two different versions of a design that the designer wants to compare. Some testers get to test version A, while others test version B. This allows the designer to make changes and see which design is preferred by the users. However, this method does not tell the designer why one design was preferred over another, and can not tell of larger design problems [26, p. 8].

5

Process

This chapter describes the design and development process of the thesis, in chronological order. This thesis was carried out over a period of almost five months, starting off with a research period of four weeks. The research made the foundation of the rest of the project. The prototype development and evaluation was executed in an iterative fashion divided into three iterations.

During the project, two different software prototypes were created. One of the prototypes blended the player's input with AI behaviour while the other was a one visual assistance that highlighted the road and sharp curves. These were evaluated using feedback sessions with employees at Ghost and through playtests. A playtest to analyse the difficulties that players have with manoeuvring was also held during this process, after creating an initial blending prototype. Altogether, analysis from our prototypes, feedback sessions and playtests resulted in eight guidelines on what to consider when creating an assistance system for players in a racing game.

5.1 Planning

During the initial discussions with Ghost, the type of assistance system to be designed was discussed. The plan was to develop this using Ghost's existing codebase as a base for our prototypes. The vision at the time was to give assistance with the player's own specific issues and weaknesses regarding the gameplay. This was envisioned to be done by analysing individual player performance in the game, possibly using some form of machine learning. There were several different ideas on what type of assistance to give to the player, such as direct help in controlling the vehicle, a visual assistance shown to the player or suggestions on upgrading a car in the game. Another area that was suggested to look into was adaptive difficulty, where the game adjusts the required skill level to give the player an appropriate challenge (see sec. 3.5).

The thesis was planned to be split into three major parts: background research, prototyping, and report writing. The initial time plan that was envisioned during this phase can be seen in figure 5.1.

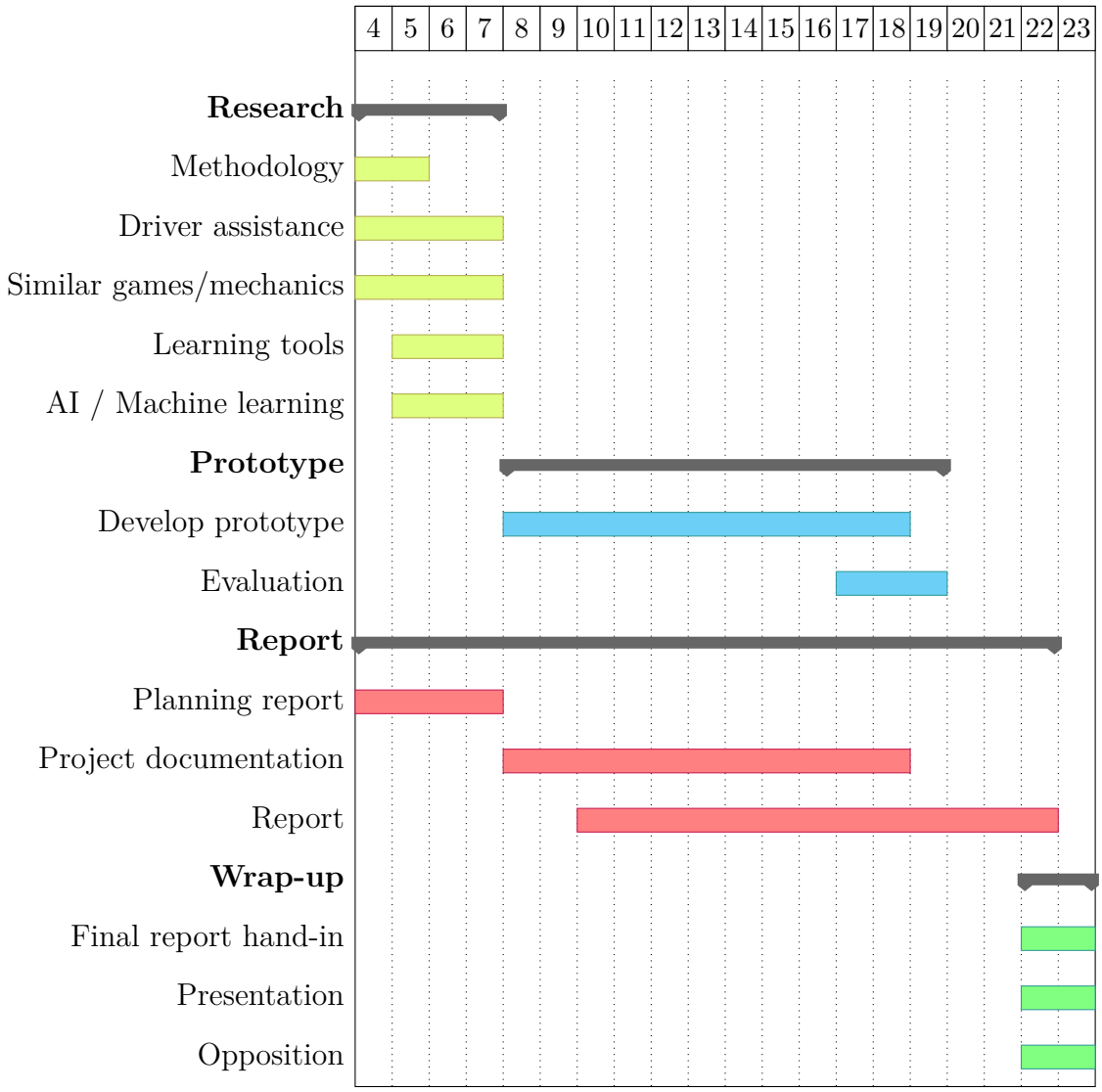


Figure 5.1: The initial time plan of the thesis, as envisioned during the planning phase.

The first four weeks was intended to be dedicated to literature studies and other research to gather knowledge as well as to choose appropriate methodology for the entire design and development process. The aim of this was to have a clear base to work from when creating and evaluating our prototypes, which in turn would serve as the base for our final results. The literature study were to serve as a knowledge base to know what is possible and what to keep in mind for creating prototypes that will give good feedback and progress our research.

As the topic of the thesis and the area of interest can be considered a wicked problem with many possible solutions, it was decided to use an iterative process as suggested by several authors [46, 48, 49]. During the prototyping phase, it was planned to create software prototypes of one or more assistances. These prototypes would be evaluated and analysed to find possible improvements, and updated to incorporate

these improvements. We would also keep a project diary where progress, process, reflections, and any problems was continuously documented. It was decided not to make physical prototypes, as there already existed a codebase that could be built upon and utilised.

The final result of the thesis was envisioned as one or more prototypes demonstrating a driver assistance system and the evaluation of these prototypes.

5.2 Pre-study

The first four weeks was spent researching the domain and possible ways to proceed with our thesis. It consisted of literature review, competitive testing of existing racing games, project planning, and investigating an already existing prototype for driver assistance. From the researching done, the project's scope had to be redefined, and after having completed the initial research we also had some thoughts on guidelines based on existing theories.

5.2.1 Literature review

This study spanned several different topics, including interaction design, game design, artificial intelligence and user experience. To get further understanding of these topics in preparation for the design work, multiple different databases and search engines was used to find relevant literature. These were Google Scholar [59], Chalmers Library [60], ACM Digital Library [61], IEEE Xplore Digital Library [62] and DiGRA Digital Library [63]. In addition to this, regular web searches was also used to find other materials, such as video material, online articles and other online sources. Some of the keywords that was used was: game, video game, artificial intelligence, machine learning, blending, adaptive difficulty, dynamic difficulty, motivation, enjoyment, flow.

Some areas that we wanted to look into were well researched, while other topics was quite difficult to find information about. Generally, there was a lack of research pertaining specifically to racing games.

It was difficult to find information on the use of academic artificial intelligence and machine learning in games. Most of the research found concerned artificial intelligence as in the simulation of non-player characters, that is, the more narrow concept used in game design and development. From this, it would seem that there has not been much research on applying AI and machine learning to video games themselves and we were unable to find any literature that was relevant to our topic. Despite this, it was quite difficult to find materials in regards to game AI and blending. We were introduced to this term by employees at Ghost, but it seems to be an uncommonly used term.

When searching for literature on games and player enjoyment, there was multiple sources that discussed flow and flow in regards to games. Thus, it was decided to

research the topic further. It was easy to find information regarding this. Another topic that we found plenty of information on was adaptive difficulty or dynamic difficulty.

This literature study formed our first thoughts on guidelines, as well as creating a knowledge base that shaped the scope of our thesis. The literature study also served to inform us on possible methodology that we may want to use during the project.

5.2.2 Existing racing games

In order to see what driver assistance systems were already present in racing games, competitive testing was performed on a number of games. The games that was tested and looked at was *Need for Speed (2015)*, *Need for Speed Payback*, *Forza Horizon 4*, *Forza Motorsport 7*, *Real Racing 3* and *Mario Kart 8 Deluxe*.

Examples of visual assistances found are: minimap, checkpoints, sharp turn indications, physical barriers and suggested path. Showing a suggested path was done both by merely visualising the path and some versions that also showed if the player should accelerate or brake. Since visual assistance seemed to be quite explored and common in games already, we chose to focus on vehicle handling assistance instead, such as aiding with steering. An interesting finding was that the suggested path in *Need for Speed (2015)* was always visible in-game, and could not be turned off.

Steering assistance also seems to be common among the games we tried, although their executions are quite different. *Mario Kart 8 Deluxe* only have steering assistance as on or off, while *Real Racing 3* have two different setting levels: high and low. Both *Forza Horizon 4* and *Forza Motorsport 7* had several levels for amount of steering assistance given. Other types of vehicle handling assistances found were automatic acceleration, braking assistance, traction control, stability control, and gear shifting.

5.2.3 Existing code and prototype

As it was intended to build upon Ghost's existing codebase when creating our prototypes, we used Visual Studio 2017 [64] and C++ since that is what was used by Ghost previously. This codebase was very large and it took some time to get an overview of the parts that were pertinent for our work.

A simple driver assistance prototype had already been created previously at Ghost, that uses blending (see sec. 3.3) to mix the player input with the AI's (non-player characters') suggested path. This suggested path is how an AI controlled vehicle would drive and is re-generated after certain distance intervals. The prototype's blending was able to affect the level of throttle, brake, handbrake, and the steering direction. The weight factors for player input versus AI input could be adjusted during run-time with a debug tool, making it easy to modify the amount of assistance given during tests. The debug tool also has a variable for toggling the assistance on

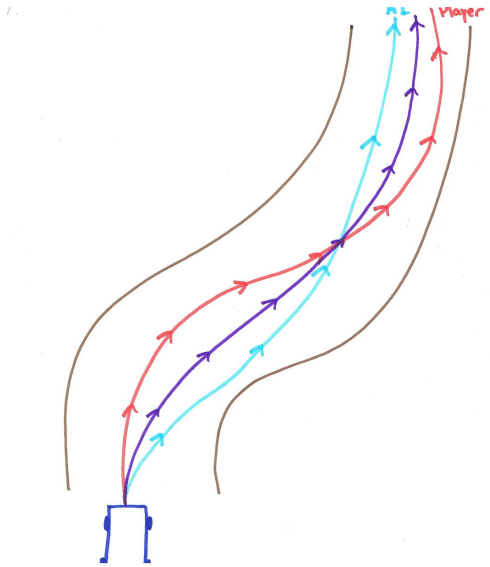


Figure 5.2: Blending of AI and player steering. The purple line is the resulting, blended path, while the red line is the player path and the light blue line is the AI path.

and off. There were not that much time invested in this prototype previously and while it demonstrated some potential, it was a very rough prototype.

5.2.4 Project planning

We aimed to use research through design (see sec. 3.1) to find an answer to our research question. Therefore, we planned the prototyping part of the thesis to be a rather large part. We planned out this work in more detail, to consist of iterative software prototyping with multiple iterations of 2-3 weeks with a smaller formative evaluation at the end of each iteration. A final, larger evaluation was also planned to evaluate the final prototype version. The software prototyping was planned to be structured using Kanban. The prototype versions from each iteration were intended to be used for testing and evaluation to identify how the player performance could be assisted, while still providing an engaging and enjoyable player experience. Our iterative process that we planned to use was based on the Wheel (see sec. 4.1.1), but we chose to instead call the Design step Ideation, as that better described the activities of the step according to our existing mental model.

Before we had access to the pre-existing codebase and tools, we still wanted to have a rough plan regarding on how to start developing our prototypes. Since it was unknown how time-consuming or challenging it would be to make changes and additions to the codebase, the first plan was very simple and rudimentary. The stages that was envisioned at this point was to first simply *render the AI's path*, then to make the player vehicle *follow the AI path* and then to ensure that *the path is continuously and frequently updated*. After these basic steps were successfully accomplished, the aim was to find a way to predict where the player is likely to

want to move.

Each step represented an iteration, where some iterations had additional sub-features that could improve the prototype of that iteration. This process was open for change later when we got to know more about the code.

5.2.5 Scope definition

While weighing each player's performance individually was a very interesting area, it was found to be quite challenging to weigh the individual player performance sufficiently and it would also require a very large amount of data. Additionally, applying machine learning on the available codebase was deemed challenging to integrate into the current state of the game and therefore more difficult to test and evaluate. Thus, both the initially suggested areas of machine learning and DDA were deemed inappropriate for the scope of this thesis. A successful system for dynamic difficulty adjustment would need much knowledge about human behaviour as the same emotions could potentially be expressed in various ways by different individuals [65]. The DDA system would also need to work seamlessly with the gameplay, possibly even be unknown to the player, as the system's efficacy might decrease if the player knows that they are getting assistance [66]. Instead of this, it was decided to continue by focusing on modifying the existing game and creating a general aid for the players.

Without the performance measuring of players, a system for suggesting how to modify or upgrade the player car was less interesting as it would be general and not personalised. It was therefore deemed out of scope. We also decided not to focus on as giving visual assistance to the player, since various forms of visual assistance seemed to be somewhat common in existing racing games. The decision was made to instead focus on directly aiding the player's steering of the car, both due to the area being more uncommon and due to Ghost showing interest in the topic. This could possibly be done by using either game telemetry or the game's existing AI. If possible, this could be enhanced to predict the player's upcoming actions through their current action. However, this was limited to single-player and when the player is in a race, and still not aimed at an open world setting.

Much of the related work in regards to real-life driver assistance focuses on improving safety which is of utmost importance in those settings. For cars in video games, there is far less concern regarding safety, as the player is never put in a real dangerous situation. Crashing the car and vehicle accidents are often instead expected and part of the gameplay. As Bailey commented about games when discussing why we play video games and said that games allow players to "take risks without suffering the negative consequences" and to "experience a dangerous situation without actual danger and to feel some of the same emotions." [67]. This thesis will therefore only investigate and evaluate driver assistance that enhances the enjoyment of the driving experience in a racing game, and not consider vehicle and driver safety. However, some inspiration can be found in existing work about autonomous cars. Vehicle automation in real life is split into six levels (see sec. 2.1). In this project we aim

for an equivalent of automation level 1 and 2, i.e. systems that support the driver but do not fully automate driving tasks.

At the end of the pre-study, the outcome of the thesis was envisioned as one or more final prototypes and a set of guidelines for how to make driver assistance in racing video games in order to help less experienced players.

5.2.6 First thoughts on guidelines

The first ideas about guidelines where in large drawn from the early discussions on the subject with the employees at Ghost and others, but also from the theories of flow and GameFlow (see sec. 3.2.1). We believed that providing an *appropriate challenge*, allowing the player to *learn* and letting the player feel *in control* would be important factors in adding driving assistance to a racing game. The first and third aspects is backed up by both flow and GameFlow, while the learning aspect is very similar to an element of GameFlow called player skills.

5.3 Iteration one

As the pre-study served as an analysis phase for this initial iteration, this iteration was begun with a ideation phase. Because of some unforeseen delays and the unexpected difficulty of getting an overview of the relevant code, this iteration was started later than anticipated.

5.3.1 Ideation

The ideation started with thinking about the possibilities given by the existing prototype (see sec. 5.2.3). This was a quite short ideation session as the factors that could be changed and manipulated were already in place, making the ideation quite narrow. Still, some thought was put into coming up with possible combinations of how the factors of throttle, brake, handbrake, and steering direction could be utilised.

5.3.2 Prototyping

During the prototyping phase several versions of the initial prototype were created, where each version had different values for blending factors. The versions were then used to evaluate by self-testing which blending values most felt like they resulted in an enjoyable player experience. If a factor was not zero, it was always affecting the player's input.

5.3.3 Evaluation

With these prototypes we only performed self-testing, as described by Fullerton [46], and not external testing, as these prototypes were very simple and too many flaws were found.

When looking at different versions, the brake and steering factors were identified as aspects that could be the most potentially helpful. Handbrake is seldom, if ever, used in regular play and is therefore of little help to the player even when blending. Blending throttle did also have a unsatisfactory effect, as the most common case is that the player is accelerating at the maximum level themselves. In that case, the blending does not make a large difference in the player performance. The other somewhat common case is that the player is not accelerating at all which instead makes the car drive itself.

When tested while playing, a lot of the vehicle behaviour from the AI blending was found to be quite intrusive on the experience of the player. When the steering blending factor was high, the controls of the car felt unresponsive and slow if the player steering deviated from the AI's path. The steering blending was the most prominent, as it felt like the car snapped back towards the AI path if the player tried to steer even slightly away from it. This was found to be quite unsatisfactory and not very enjoyable, which is in line with our thoughts on letting the player feel in control (see sec. 5.2.6). At very low blending levels, the steering blending was instead very difficult to discern and had no noticeable impact. At higher brake blending levels (>50%), blending the brake input seemed to be somewhat helpful, particularly in turns. However, particularly the brake blending seemed to make it more difficult to drift during the race.

Another issue that was found with the implementation was that the AI does not take opponents into account when generating the path used to nudge the player's vehicle. This resulted in additional unwanted and intrusive vehicle behaviour since when the player tries to avoid opponents by steering away from the AI's path, typically at the beginning of a race, the blending pulls the player's vehicle back towards the path.

5.4 Iteration two

After having implemented a basic prototype and identified its potential as well as its areas for possible improvements, a second iteration was started which was based on the result of the first one. The idea was to try to remove or at least decrease the intrusiveness, and increase the effectiveness of the assistance given to the player. In the end we managed to create two prototypes, and attempted to create another two that unfortunately were deemed unfeasible for our time frame, but they are still relevant to the concept. Internal feedback sessions concluded this iteration during which we let employees test and evaluate the created prototypes.

During this iteration we discovered self-determination theory (see sec. 3.2.2) and the grounded theory method (see sec. 4.2.3), which was useful during the evaluation

and analysis (see sec. 5.5.1) performed after the feedback sessions.

5.4.1 Analysis

From our first iteration, we found several flaws. Braking assistance performed poorly, as it did not take the player's situation and context into account, for example, if the player is trying to drift. The braking assistance counteracted this by braking differently, leading to unexpected behaviour of the player's vehicle.

The driver assistance was often most intrusive in sharper turns, when wanting to drift, but also in some versions made it difficult to steer and turn in general as wanted. When driving on straight roads, the player might want to keep going forward, while the AI may want to move closer to one of the sides. Then, even though the player is not touching any steering input, the car is moving slightly to the side, creating a very strange user experience.

5.4.2 Ideation

The ideation phase started with a short session mostly consisting of brainstorming and sketching to show and substantiate our ideas on how to remedy the flaws that was found with the initial prototype. The choice was made to do this individually at first to get as many different ideas as possible, and then to jointly discuss and further solidify some of our ideas. This let us come up with ideas on how to proceed with the already existing prototype by fixing its issues regarding intrusiveness and the vehicle feeling unresponsive, as well as improving the assistance it is supposed to give.

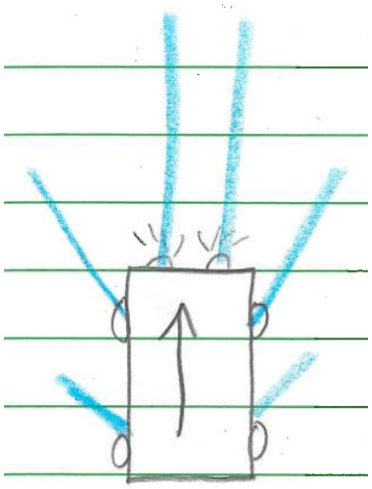
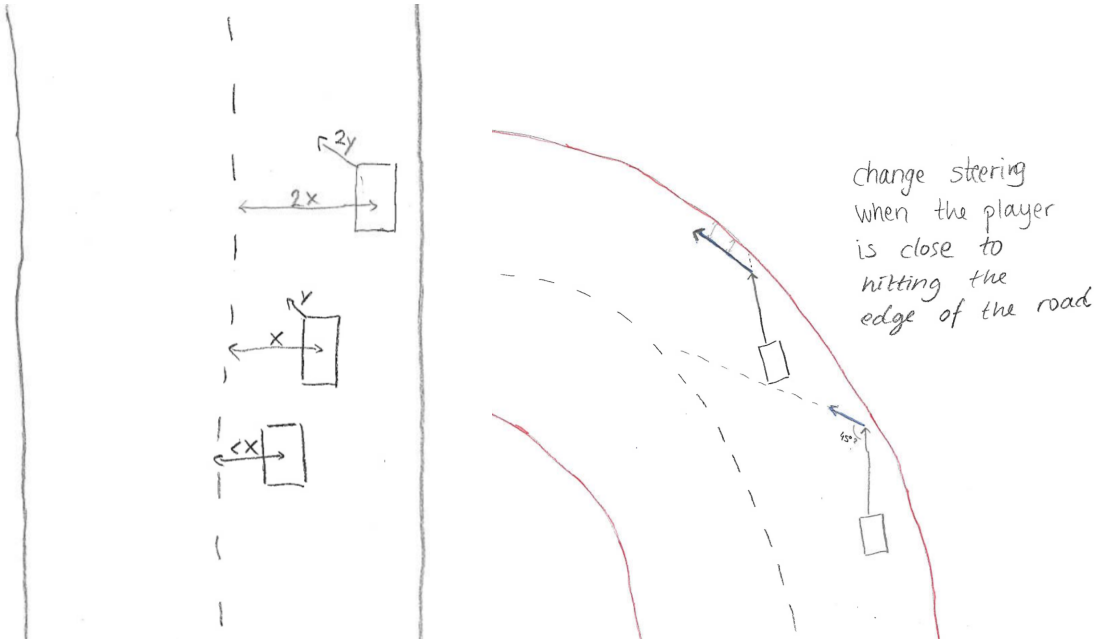


Figure 5.3: Raycasting sketch.

An idea for making the assistance take opponents into account when manoeuvring was to implement ray casting for checking nearby obstacles (see fig. 5.3). This was

the approach that Voll [38] used for fulfilling the same purpose, and we thought that it would probably be the easiest and thus most suitable method for our prototype.



(a) Strength of blending dependent on player deviation from the suggested path.

(b) Stopping the player from going outside the race track.

Figure 5.4: Two ideas on how to adapt the given assistance depending on the player's position on the track.

Another idea was to take the distance between the player vehicle and the AI path into consideration and make the blending effect depend on that relation, instead of always blending the player's input.

5.4.3 Prototyping

Several different prototype versions building on our initial prototype from the first iteration was investigated. Only two of them turned out to be feasible within the time frame: what we called hard blending and soft blending. The first one, hard blending, tried to use the players distance from the AI path to blend in a better way. The second version, soft blending, was created in an attempt to improve on hard blending and make the blending smoother. The other versions were other ways of trying to take the player's context and behaviour into account.

5.4.3.1 Hard blending

One of the ideas from looking at the initial assistance system was that if a player is close to the AI path, they probably require less or no assistance and if the player is far away from the AI path they are more likely to need assistance. The blending

was adjusted to only be active if the player's vehicle was a certain offset distance away from the AI generated path. If the player's vehicle was close enough to the AI path the vehicle would be fully controlled by the player's input. This prototype was called hard blending, and its concept is visualised in figure 5.5. It was possible to adjust this offset distance during run-time. Only the steering changed in this prototype. The other factors were still blended as in the initial version.

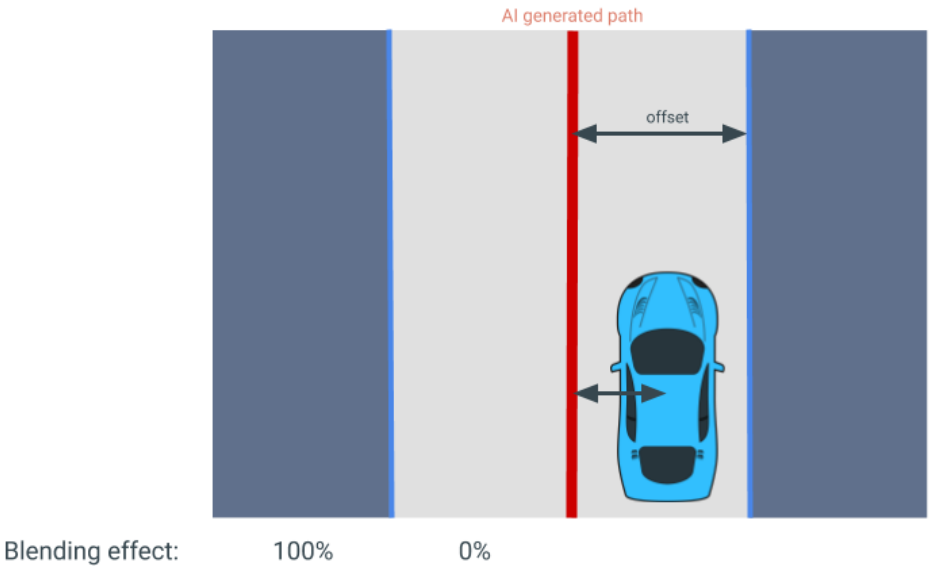


Figure 5.5: Concept image of the hard blending assistance. The red line is the AI generated path.

For analysing the prototype and identify issues that needed to be fixed, we took inspiration from Voll's process [38] for designing an AI opponent. This process was found suitable as we were not designing a whole new game as Fullerton's playcentric process [46] is made for, but rather improving an existing system.

5.4.3.2 Soft blending

Many of the issues found in the initial prototype was still present in the hard blending prototype. These were used as inspiration for developing an improved version of hard blending. We chose to proceed by trying to remove the intrusiveness of the blending snapping back towards the AI path. It was considered to try to take opponents into account when generating the AI path, but we believed it would be more work for likely less gain at this point, as a lot of variables would have to be taken into account. The proposed steps for this prototype was:

1. Apply steering blending when the player is at least a certain distance from the AI path
2. Decrease blending factor when moving away from the offset line

The soft blending prototype was created with the hard blending prototype as a base. Similarly, instead of only looking at the AI generated path, an additional offset line

is taken into account for the blending. This line is parallel to the AI path, but with a variable offset. With hard blending, the blending was fully activated when the player's vehicle had passed the offset second line. For soft blending, this was changed to differentiate the player's position against the second line. The closer the player is to the offset line, the larger the blending factor is set. The blending factor decreases as the player moves further away from the offset line. This concept is visualised in figure 5.6. Both the offset and the fading distance (the distance at which the blending factor is zero) are variables that can be adjusted in real-time when testing using the debug tool.

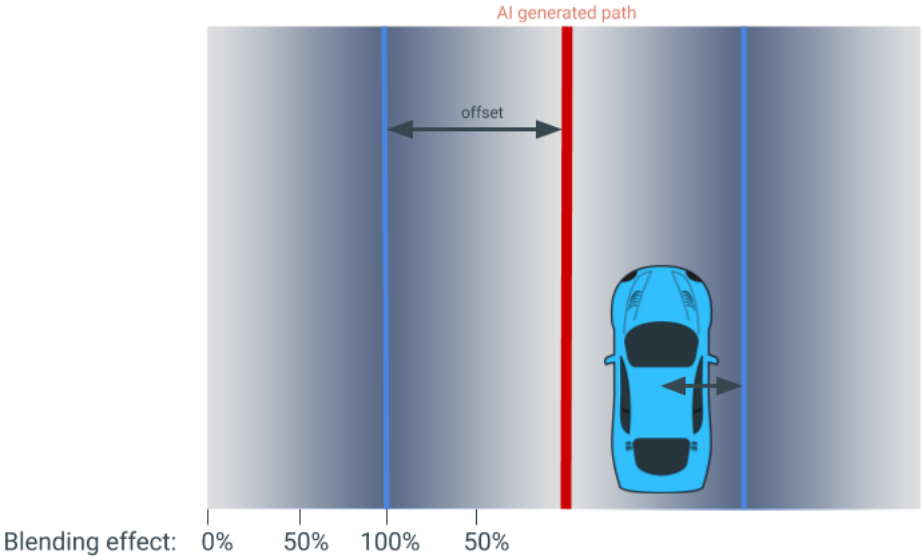
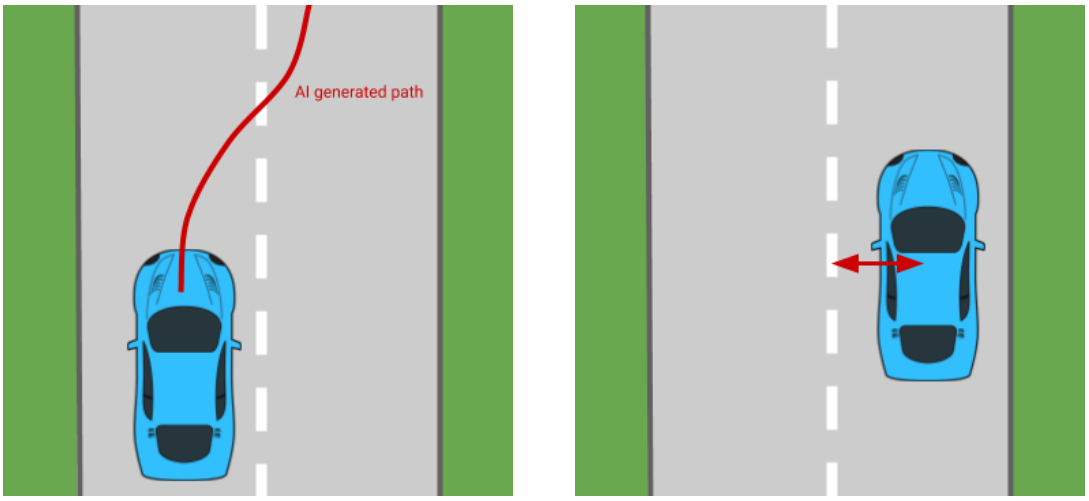


Figure 5.6: Concept image of the soft blending assistance, visualising the fading effect. The red line is the AI generated path.

5.4.3.3 Other prototypes

The two prototypes described previously, hard and soft blending, were the successfully completed prototypes created during iteration two. However, we did look into several other prototype ideas as well.

One idea was to update the player's suggested path more frequently and having it originate from the player's position. This attempt was cancelled due to technical challenges regarding the construction of the AI generated path that would require us to put time into fixing other, secondary issues that are not directly related to our project. Another idea was to compare the player's position against the middle of the race track. It proved challenging to reliably find the centre of the race track, as there were many different configurations of race tracks. These two concepts are visualised in figure 5.7. In another attempted prototype, blending was done depending on how much the player was braking, with full braking meaning no blending at all. These three prototypes were not fully explored due to lack of time, and could have potential if investigated further in the future.



(a) Generate the AI path frequently from the player's position.

(b) Check the player's position against the middle of the race track.

Figure 5.7: Two other prototype concepts that were considered.

An idea that proved unsuccessful in our prototyping was to not blend in AI input when the player was not actively steering. This ended up not working very well, especially in turns on the race track. If the player drove straight forward, not touching the input, but would have needed to turn a little they did not get enough help to make a difference.

5.4.4 Evaluation

When self-testing this version, it was noticed that small changes of the adjustable variables, both the new ones added in this iteration and those present in the previous iteration, had a very noticeable impact on the player experience. For low blending values the prototype's effect was perceived as barely noticeable, but the assistance became too noticeable when these values were increased only slightly. It was proposed that instead of having the blending effect fade linearly, it might be better to have an exponential decrease. However, that would probably increase the difficulty of balancing the blending effect. For this prototype the linear fade was kept, and values that was found to not be very intrusive on the player experience were chosen.

Three prototype versions was selected to evaluate the potential of the concepts. These versions were taken from both the first and second iterations and selected by self-testing.

When self-testing in the first iteration, blending of the braking was found particularly noticeable. Because of this, we chose to test two versions of the prototype from our first iteration. Both versions used 75% of the player input and 25% AI input for blending. One version blended only steering and the second also blended braking. The last selected prototype was soft blending with 75% of the player input and 25% AI input at maximum blending. This allowed us to test the viability and potential

of different versions of the first prototype against the newer soft blending prototype.

For testing our blending prototypes, we first considered using participatory design (see sec. 4.5.3). However, it was not feasible to include all stakeholders at this early stage, due to time constraints. Therefore, we decided to limit the testing sessions to employees at Ghost. Three evaluative feedback sessions were carried out with different Ghost employees, two on the same day and the third session on a later day. We chose to evaluate our work on employees since they were easily accessible, and it was uncertain whether we were allowed to invite external participants to the office to test our prototypes. Each session started with an introduction of the project, session and its goal, and the participant performed one race without any assistance to get familiar with the car and track. Then, each version was tested for 2-3 races without the participant knowing in what way they were being assisted. After having tested all three versions, the participant was told what each version did, and they got to try them out again in the same order as before. Throughout the whole session the participant was encouraged to use the think-aloud technique, that is, talk out loud about their thoughts and any comments that they had simultaneously to testing (see sec. 4.5.1). The order of the versions were randomised for each session, so that no participant had the same order, as suggested by Nielsen [50].

5.5 Iteration three

In conjunction with each evaluative feedback session done in the previous iteration, we also had a brief ideation session with each of the participants on how they would improve the prototypes. However, after concluding the feedback and ideation sessions, they did not end up providing the kind of results that we expected or wanted. The main goal was to evaluate efficacy of the different versions of assistances, but since the prototypes were tested on fairly experienced racing game players the result was that the less assistance a version had the more appreciated it was from the participants.

In an attempt to achieve a more structured way of motivating our design decisions, we decided to take inspiration from the playcentric design process. Fullerton [46] suggests listing player experience goals at the beginning of the development. By considering our own goals as well as previous discussions with Ghost, the following player experience goals were identified for what the assistance system should do:

- Help players improve their driving/racing mechanics.
- Make the races feel fair.
    - The races should neither be dominated by the player(s), or the NPCs.
- Be non-intrusive.
    - The assistance should not take away the immersion from the player's game experience.
- Improve the immersion.
    - Giving poorly performing players assistance to let them focus on the gameplay instead of their performance.

However, after these player experience goals were identified, it still felt as if these player experience goals had very low support in their validity, as they were mostly based of our thoughts from self-testing. This lead to the study taking another approach: examine what players find troublesome regarding in racing games in order to identify what to assist with. Therefore, it was planned to begin with a user study consisting of playtests. After analysing the outcome of these, we would continue with prototype development based on the results of the analysis, and then evaluate the prototypes created with similar playtests, hopefully with both returning and new participants. Having new participants would help to avoid influences from the first playtest.

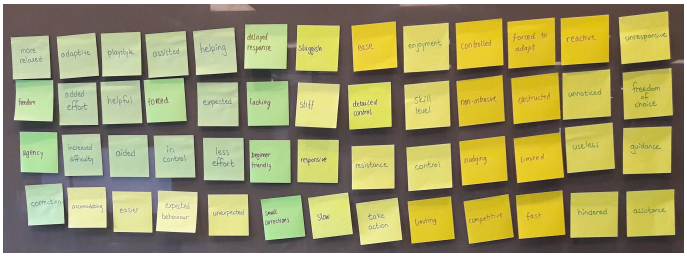
5.5.1 Analysis

From the feedback sessions that were performed, we found that our participants still found many of the same issues that we had previously found in our first iteration. After doing a more thorough analysis of the results, it was decided that we needed to shift gears and do a deeper analysis of the prospective users of the driver assistance was required. To do this, a playtest was performed with external participants and the results of this was analysed.

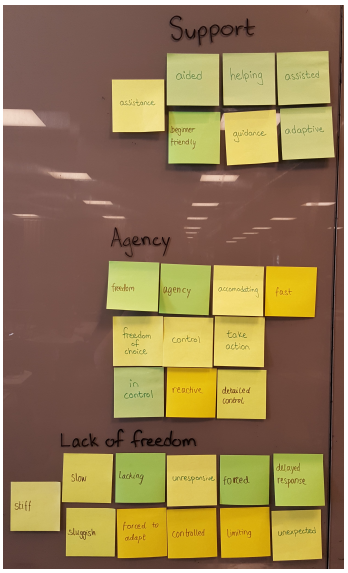
5.5.1.1 Grounded theory analysis

After the two first feedback sessions performed in the second iteration, all results were combined by storing all observations as individual data entries. For compiling the data from the feedback sessions, we intended to use the grounded theory method (see sec. 4.2.3) after a recommendation from our academic supervisor. However, since we never formed a theory out of its result, our approach was more similar to content analysis (see sec. 4.2.4). The method was performed based on the data where we first individually performed data coding, and later discussed together, choosing one code was chosen for each data entry. When the third feedback session had been completed, coding was performed again in the same way as before. However, since we had already created codes for the data from the first two sessions, we were highly influenced by these and found ourselves re-using them a lot.

For coming up with concepts each code was written down on a post-it note, as shown in figure 5.8a. Affinity diagramming (see sec. 4.2.5) was performed in order to create groupings and themes. From this 10 concepts were formed, some of which are shown in figure 5.8b. Categories were then created by combining similar concepts resulting in 4 final categories: *Assistance*, *Agency*, *Enjoyment*, and *Predictability*. Due to rather limited data and thus few categories, and not enough depth leading to what Hook [51] calls saturation, no core categories were created and no theory was formed.



(a) The codes resulting from performing the grounded theory method on the data gathered during the feedback sessions.



(b) Some of the concepts that were created through affinity diagramming on our codes.

Figure 5.8: Using affinity diagramming to perform the grounded theory method.

5.5.1.2 Playtest

It was decided that a playtest of the game without the use of driver assistance would be useful to find out what the participants found difficult about driving in racing games and collect information and data about this. During the playtest session the participants were to play the game as normal, while we observed them and their behaviour.

For finding participants a questionnaire was sent out to some of our acquaintances, which asked how much they play video games, what type of controls they play with (keyboard and mouse, controller, or other) as well as evaluating their previous experience of playing racing games. They were also to fill in at which times they were available. There were 4 responses to this questionnaire. None of the respondents played games very much on average, but all had some experience in playing video games. The initial plan was to invite people based on their racing game experience. Since there were so few responses and all of them were not available during the same time, a time during where 3 respondents were available was chosen. This ended up being a playtest session containing two participants with less racing game experience and one more experienced.

The playtest was setup with a prototype version of a race track for the participants to race on. The participants' screens were recorded and telemetry data was collected. This data was things such as position, speed and brake. During the playtest semi-structured observation was used (see sec. 4.2.6) to examine the behaviours of the participants. Drawing from fly-on-the-wall observation, we attempted to be silent

during our observation and mainly interact with the participants after the playtest was performed. For the playtest, the participants first played the race 2-3 times. Initially they were only told the basic controls of steering, throttle, brake, and speed boost. After these races they were introduced to the controls for drifting. They played same race again, but this time they were encouraged to try to drift. After the participants had played, a semi-structured group interview was held. For the interview, we had a few general questions and went into more detail gradually, as to not limit the responses from the participants. This interview was audio recorded.

5.5.1.3 Playtest analysis

From the playtest and its following group discussion, we concluded that the participants do not want to be controlled or tricked into thinking that they are better at the game than they actually are, by being given invisible assistance. The participants therefore suggested visual assistance instead, since it would be clear that assistance is given and as players they would be given the option to follow it or not. They also stated that they want to be able to learn the game properly which they speculated that assistance might hinder them from, especially if they think their good performance is fully because of their own skill and not that they are being aided.

The participants with less racing game experience also said that they sometimes found it difficult to see where the race track were. However, this was probably influenced by the prototype state of the race course, which resulted in, amongst other things, road barriers and other signs being loaded into the game world too late. If identifying the track still is a problem in a more finished game version, the participants were positive towards having visual cues for showing the stretching of the course, for example a line on the ground that they could follow.

Another issue that was identified was when to time actions, especially regarding turning and drifting in sharp curves. The playtest participants that had less racing game experience seemed to have trouble knowing how to steer to effectively take the curve without braking too much, and were often not able to get the car to drift. This could also be because of the vehicle handling being different from what they are used with from previous racing game experiences. Some aids with timing was suggested, but the participants expressed more interest in getting clear feedback when successfully drifting, as the drift does not start immediately and they found it difficult to know when they did succeed.

5.5.2 Ideation

After our playtest, we held an ideation session where we reflected over and analysed the result from the playtest. The playtest participants seemed very positive towards having visual assistance. We had previously deemed visual assistance out of scope for our thesis, but as the participants were quite negative about being assisted directly we wished to consider it again. It was decided to brainstorm (see sec. 4.3.1) about

the possibility of how to visually aid players. Our thoughts were structured as a mind map (see sec. 4.3.2), shown in figure 5.9.

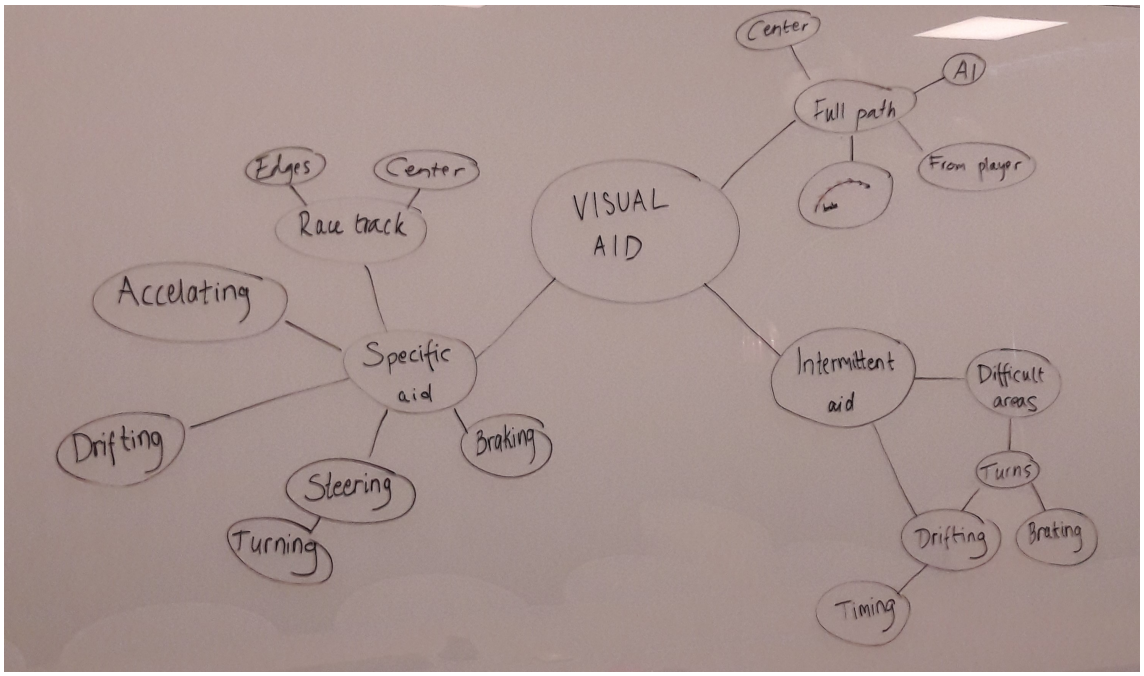


Figure 5.9: The mind map created from the brainstorming session during iteration three.

From all ideas generated with the mind map, we chose the ideas that seemed the most promising. These were:

- **Drawing full path:** draw the AI path similarly, but make it originate from the player's position.
- **Intermittent aid:** help with timing of drifting (helping in sharp turns).

A number of digital sketches was created of how these assistance systems could look, by drawing on top of a screenshot from a previous *Need for Speed* title. These images were then used as inspiration during the prototyping phase, together with the visual assistance present in *Need for Speed (2015)* and *Forza Motorsport 7*.

5.5.3 Prototyping

When we were to implement the visual assistance prototype we needed to put together multiple versions of the code, which turned out to be more complicated and time-consuming than we had expected.

First, we looked at data from the previous playtest for one of the participants that performed well to identify recommended areas for drifting, as seen in figure 5.10. Then, a number of particularly sharp curves along the track were selected where the player is recommended to drift according to the data. For the aesthetics of the visualisation of the sharp turns, we experimented with various designs; different

variations of just drawing the points, just drawing a line between the points, and drawing both the points and lines between them. Some of the designs are shown in figure 5.11. We tried different colouring and sizes of the points and lines. In the end we chose to use a design where both the points are drawn and with thick lines between them. After having chosen a design, the chosen curves were marked in-game.

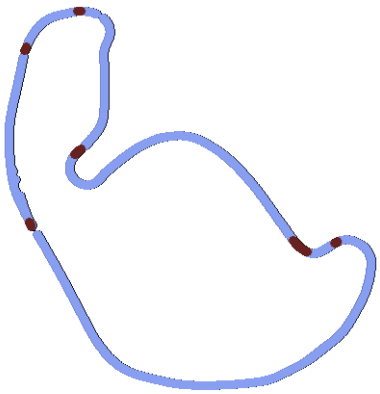


Figure 5.10: Visualised telemetry from one of a playtest participant's race.

After we had placed the lines indicating sharp curves in the race, we held a short ideation session with one of the employees that had previously participated in our feedback sessions. He was asked to try the visual assistance prototype. He seemed positive about the concept, but it was noticed that one of the curves did not benefit from the markings, as it was very short and not helping much since the curve was not sharp enough. He also gave us the idea to add a visual suggested path together with this prototype.

When continuing on this prototype, we decided to take the advice of the employee and add a suggested path. We added a visualisation of the AI path that we used for the blending in our previous prototypes. At first, this line was shown as a white line, but this made it blend in with the already existing white lines on the road. We then coloured the line red instead, but since the sharp turns indicator is red as well, the turns were difficult to differentiate from the suggested path. We therefore coloured it green instead to make it stand out more from the sharp turns.

5.5.4 Evaluation

It was decided to hold another playtest, this time for evaluation. Initially, the idea was to conduct A/B testing (see sec. 4.5.4). However, we realised that we would need a larger number of participants to get a more indicative result. We also wished to test the blending prototype as well as the visual prototype, and they were likely too different to be compared using A/B testing.

Instead, the playtest was set up in the same way as the previous analysis playtest, with an observed playtest (see sec. 4.2.6) and a concluding group interview (see sec. 4.2.7) after the playtest. The same participants as in the previous playtest were

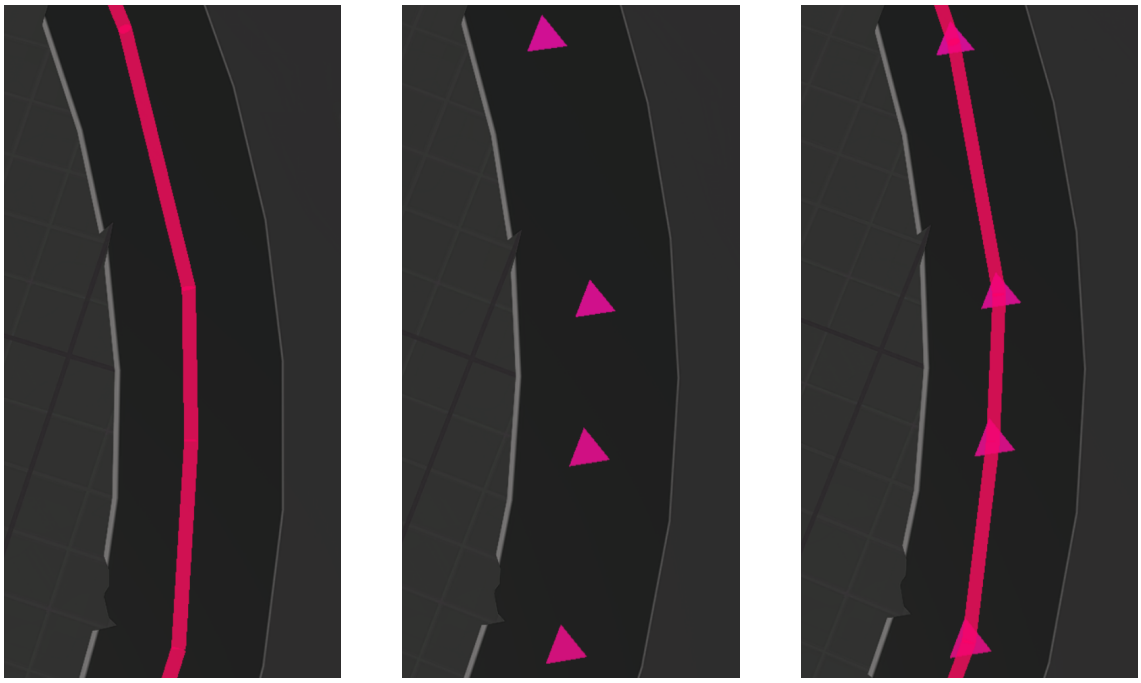


Figure 5.11: Three versions of the visual assistance seen from above: one with a line, one with points, and one as a combination of the first two with both points and lines between them.

invited. An effort was also made to extend the playtest to people who was unable to participate in the previous playtest, but this ultimately proved unsuccessful due to time constraints and scheduling clashes.

In the playtest, two prototypes were tested. The first was the visual assistance created in this iteration, as this is what the participants suggested during the last playtest. The other was the simple blending prototype created in iteration one (see sec. 5.3). The blending prototype was chosen to be included in this evaluation as the testing participants was negatively set towards such an assistance during the previous playtest, to evaluate if it actually had a negative impact on their experience.

5.6 Refining guidelines

The last part of our process was dedicated for refining the guidelines from previous iterations. They are based on results from our feedback sessions and playtests, as well as information gathered during our pre-study. Many of our guidelines can be connected to enjoyment (see sec. 3.2) and especially GameFlow (see sec. 3.2.1).

5.6.1 Analysis

For the blending assistance in the playtest, the participants did not notice that they were getting assistance. They did also not notice any major differences regarding their performance, other than that they might have improved slightly during the first playtest. For such an assistance to be useful, it would probably need to blend AI variables more than the prototype they tested did, especially as the vehicle's speed was quite high resulting in the blending assistance being very subtle.

The playtest participants with less racing game experience seemed to like the concept of having visual assistance highlighting the race track, and were particularly fond of the highlighted sharp curves and that it could be seen through terrain. Having the sharp curves clearly visible early on let them plan in advance, which they liked.

5.6.2 First draft

The guidelines below were the result of compiling player analysis and group discussion data from the playtest evaluation regarding player needs, together with our early thoughts on guidelines.

Planning: players want to be able to plan their (near) future.

Performance: players get higher enjoyment when they perform better.

Consistency: help players' reoccurring errors in the same way each time.

Flexibility: players don't want to be forced into playing in a certain way.

Control: players want to feel in control of their actions.

Learning: players want to be able to learn and improve while playing.

Challenge: players want an appropriate challenge.

Honesty: players don't want to be deceived regarding their actual skill.

Predictability: players don't want false or misleading information.

After having drafted initial guidelines, we felt like they were too generally defined and did not reflect the context in which they were created: racing games. Therefore, we formulated a list of racing game specific factors that we had considered during our project. These factors can be found in the list below:

- Continuous input
- Most players use a game controller
- High speed
- Vehicle steering → expectations regarding vehicle behaviour
- You play against opponents (AI or real players)
    - symmetrical gameplay
- Open world → players might expect more freedom

5.6.3 Further refinement

For our second iteration on the guidelines, we attempted to connect our findings more clearly with the theory and our observations and findings from our project. These are the guidelines formulated from this:

Planning: Beginner players might need more time to think as well as more information in order to compensate for their lack of reflexes, and learned habits and overall game knowledge. This can be done by enabling players to plan ahead by clearly showing information about upcoming, critical moments, for example sharp curves where the player need to slow down or drift to smoothly make through the curve.

In our playtest, we found that the visual assistance being visible through the world's environment was considered helpful, as it let the player prepare further in advance regardless of the environment of the game in that moment. Unexpected events can lead to mistakes which risks breaking the player's flow [31]. It can also work against their need for competence, as seen in self-determination theory [28].

Performance: Players get more enjoyment (see sec. 3.2) and an improved game experience when they perform better in a race, i.e. when they are not too far behind their opponents. New players want to feel like they actually have a chance to win, or else the competition risks feeling one-sided and meaningless.

This was given as feedback in our playtests. Players gained more satisfaction from getting a higher placement or at least feeling a possibility of winning. This is seen as the aspect of challenge in GameFlow [30] and by the need for competence in self-determination theory [28].

Flexibility: Assistance should not be forced on players; it should be optional and only be given to them if and when they want it. Therefore, there should exist a setting in the game for enabling and disabling the assistance. Flexibility may be even more important in open-world games, where the players are likely to expect more freedom.

When holding the playtest interviews, the player with more racing game experience was less interested than the others in using the visual assistance line. An example of always active assistance can be found in *Need for Speed (2015)* (see sec. 2.2.3). This guideline is connected to the need for competence in self-determination theory [28] and the aspect of control in GameFlow [30].

Control: Players want to *feel* in control of their own vehicle, and that their actions lead to meaningful events and choices. They do not want to feel that they are controlled by someone or something, or else it risks breaking their flow (see sec. 3.2.1). If the player is given assistance in any way, it should either be known and obvious (for example visual), or completely unnoticed.

This was found in all evaluations, both the feedback sessions and the playtests. However, assisting the steering is an assistance which is fairly unobtrusive for *Mario Kart 8 Deluxe*, but did not work in our prototypes. This is supported by control in

GameFlow [30] and as part of the need for autonomy in self-determination theory [28].

Learning: Players want to learn during gameplay, which will improve their performance and thus also their enjoyment. This can be done by for example showing specific information about the race, and by letting them learn helpful habits by exposing them to successful experiences such as successful drifting. It is important that players can still improve even though they are getting assistance, so that they eventually will be able to play without it.

During our playtests players have expressed that they want to learn during gameplay and improve at the game. A large amount of assistance, such as the maximum assistance possible in *Forza Motorsport 7*, may remove mistakes entirely. This makes learning and improving almost impossible to achieve. This is represented as the need for competence in self-determination theory [28] and mainly as player skills in GameFlow [30].

Challenge: Players want to feel challenged, but not overwhelmed. Therefore it is important that players are faced with an appropriate challenge; the race should neither feel to easy nor too difficult to win. Assistance should enable the player to perform good in terms of letting them feel that they have a chance to win, but it should not be guaranteed.

This was seen partly in the playtests. This guideline is very close to what GameFlow's [30] concept of challenge and is also connected to the need for competence in self-determination theory [28].

Honesty: When giving a player noticeable assistance such as visual cues, the player must be able to trust the information given. Players also want to know what is occurring, and not feel misled.

During the initial playtest, participants expressed a dislike of being invisibly aided by a driver assistance. They felt that if they had discovered such an assistance, they would have felt tricked and disappointed in their own performance. However, it is worth noting that they didn't notice much of this in the following playtest. This does indeed go against both the aspect of control found in GameFlow [30] and the need for autonomy in self-determination theory [28].

Predictability: Assistance need to help similarly so that the player knows what to expect from it and they should get an even experience from the assistance.

Both in our feedback session and the playtest of the visual assistance, participants expressed confusion and irritation when the visual cues were lacking or misleading. This is similar, in part, to the aspect of clear goals in GameFlow [30].

Consistency: A player should not get assistance in unexpected ways. If a player is given assistance in a certain way in a situation, they should receive assistance in the same way when encountering a similar situation. If a player expect some type of help, its execution or presentation should not suddenly be changed.

This allows for mastery and learning, as supported by the need for competence in self-determination theory [28] and player skills and clear goals in GameFlow [30].

6

Results

The result of our thesis is a set of guidelines for designing driver assistance for racing games and two software prototypes that implements two possible versions of driver assistance.

6.1 Prototypes

The final prototype results consist of two prototypes, the soft blending prototype and the visual assistance prototype.

6.1.1 Soft blending prototype

The soft blending prototype uses blending with the AI path to assist the player. It aims to be flexible for the player, in that it has a scaling amount of blending that depends on the player's distance from the AI path. The prototype uses an offset from the AI path that it scales around, with the most assistance when driving on the offset line and less as the player moves away from the offset line. If the player gets far enough away from the offset line, either towards or away from the AI path, the blending assistance is turned off. If the player is close to the AI path, it is assumed that they do not need assistance and if they are too far away from the AI path it is assumed that they wish to deviate from the race altogether or is performing poorly enough that the assistance would be too obvious and intrusive. This can be seen in figure 5.6 in section 5.4.3.2.

6.1.2 Visual assistance prototype

The visual assistance prototype uses visual elements in the game to assist the player. When turned on, it always shows the AI path as a green line on the race track. As the player approaches sharp curves in the race track, differently shaped red blocks was overlaid over the AI path line. This can be seen in figure 6.1. The placing of these sharp turn indicators was done manually, so these are only present on a specific prototype race track.

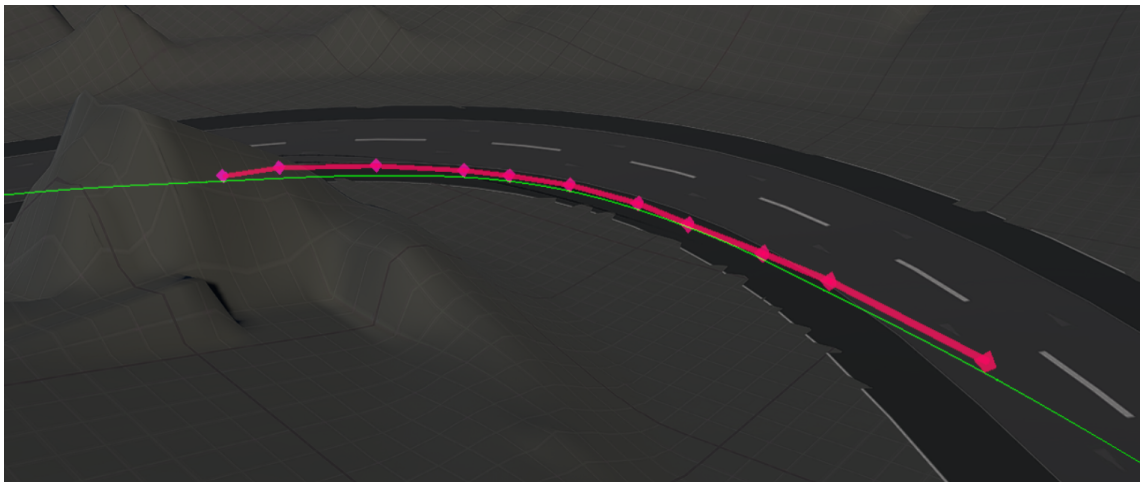


Figure 6.1: The suggested path in green and the markings for sharp turns in red.

6.2 Guidelines

The final result of the thesis are eight different guidelines which serve as recommendations for creating driver assistance systems for racing games. These guidelines aim to give indications on factors that can make a positive impact on the player experience for players of lower skill levels. These are based on both our findings and feedback given to us, and on the theories of flow [31], GameFlow [30] and self-determination theory [28]. The aspects from flow, GameFlow and self-determination theory that has influenced our guidelines the most are *control*, *player skills*, *challenge*, and the need for *competence*.

6.2.1 Let the player feel in control of the car

Don't take control from the player, let them feel like they are choosing what they want to do.

Players want to *feel* in control of their own vehicle, and that their actions lead to meaningful events and choices. They do not want to feel that they are controlled by someone or something, or else it risks breaking their immersion and flow. We propose that if the player is given assistance in any way, it should either be known and obvious to the player, for example visual assistance, or completely unobtrusive and nearly unnoticed.

This was observed in both our feedback sessions with Ghost employees and in our playtests. Participants were very negative towards not being able to have full independence and control of their actions in the game. Additionally, during the initial playtest, participants disliked the thought of finding out that they had previously been assisted without knowing it, and felt that this would give a lack of control. They believed they would feel tricked and deceived regarding their own performance. In some cases, this guideline may be contrary to some of our other guidelines. For example, letting the player be completely in control of their actions may lead to

them performing much worse, which may be in conflict with the guidelines found in 6.2.3 and 6.2.4. However, it seems to be the feeling of control that is wanted, and not necessarily actual control. A counterexample to our observations is *Mario Kart 8 Deluxe* which has a smart steering option that significantly affects the players' steering, but is generally found acceptable. This guideline is also well supported by both the aspect of control found in GameFlow [30], and the needs for competence and autonomy in self-determination theory [28].

6.2.2 The vehicle should behave as expected

Match player expectations of vehicles; both real world and in-game.

Unmet or opposed pre-existing expectations from the players of how the game will act can result in confusion and a break in the immersion and concentration of the player. Such an unexpected event could be that the car suddenly move to the left when the player tries to move it forward, which does not match the player's real-world expectation of how a car should move. Our advice is to ensure that any assistance system helps in a consistent way, so that the player knows what to expect from it. If a player is given assistance in a certain way in one situation, they should receive assistance in the same way when encountering a similar situation. A player should also get an even and stable experience from the assistance.

This was observed mostly in our evaluative playtest, but also in the short feedback session during our third iteration. Participants were confused and annoyed when the cues from the visual assistance was misleading or lacking. Unexpected and discrepant behaviour works against learning and mastery of the game by not conforming to what the player has learnt. This guideline is connected to the aspect of player skills and clear goals in GameFlow [30] and the need for competence in self-determination theory [28].

6.2.3 Even out the playing field for the player

Make the player feel like they have a possibility of winning.

Players get more enjoyment (see sec. 3.2) and an improved game experience when they feel as if they can achieve something, with or without practice. They want to feel that they have a chance of winning a race, since if they are too far behind their opponents, it may feel impossible and meaningless. Evening out the competition should be done in a way that feels natural and non-jarring to the player. For example, a sudden teleportation of the player's vehicle would break the immersion and flow of the player due to unexpected behaviour as well as taking control from the player (see sec. 6.2.1).

This was given as feedback in our playtests and was particularly noticeable in the latter playtest. Players gained more satisfaction from getting a higher placement or at least feeling as if they had a possibility of winning if they were to improve. This guideline is closely connected to the guideline found in 6.2.4. This guideline

is supported by the aspect of challenge in GameFlow [30] and by the need for competence in self-determination theory [28].

6.2.4 Present the player with a challenge

Don't give the player a guaranteed or very easy win from using the assistance.

Players want to feel challenged, but not overwhelmed. Therefore it is important that players are faced with an appropriate challenge; the race should neither feel to easy nor too difficult to win. Assistance should enable the player to perform well in terms of letting them feel that they have a chance to win, but it should not be guaranteed. We suggest giving the player assistance in such a way that it improves their overall performance, but that they still need to try to win.

This was a sentiment supported by the playtests, as the participants expressed a more positive impression of the game when they were perceiving a level of challenge where it was not automatically a win, but they had a possibility of winning or getting a higher placement. This could be done similarly to *Forza* franchise, with many different levels of assistance. This guideline is a very important aspect of the theories studied, presented in GameFlow [30] as challenge and in self-determination theory as the need for competence [28]. The guideline is also connected to the concept of player skills in GameFlow, in that it aims to encourage learning and help the player achieve mastery by pushing them to improve and perform their best.

6.2.5 Allow for planning ahead during races

Allow players longer reaction time in order to compensate for lack of learned habits and game knowledge.

Beginner players might need more time to think and process information in order to compensate for their lack of reflexes, learned habits and overall game knowledge. If they need to react quickly in the heat of a moment, they risk making critical mistakes. These mistakes might not only result in bad placement in the race, but also breaking their immersion and flow [31], and demotivate them from continue playing. We suggest to allow the players to plan ahead by clearly showing information about upcoming, critical moments, such as sharp curves where the player need to slow down or drift to smoothly make through the curve. It is likely beneficial to the player to see this information early, even if it is not completely natural in the game setting.

This was found during our evaluative playtest when testing our visual assistance. The participants were positive towards the assistance being visible through the world's environment as it warned them about upcoming sharp turns, even if the turns were obscured by the environment (see fig. 6.2). This allowed for them to plan their actions in advance. A weaker example of this can be seen in *Need for Speed (2015)* and *Forza Horizon 4* which show a path on the ground for the player to follow. *Forza Horizon 4* also offers colour coding of the line to help players know

when to brake or accelerate. This guideline aids the player in achieving the need for competence in self-determination theory [28] by letting the player be more in control. It is also beneficial for the aspects of control and clear goals found in GameFlow [30] by giving the player more information, but may possibly be detrimental to the aspect of immersion if it's added as an element that is not naturally part of the world.

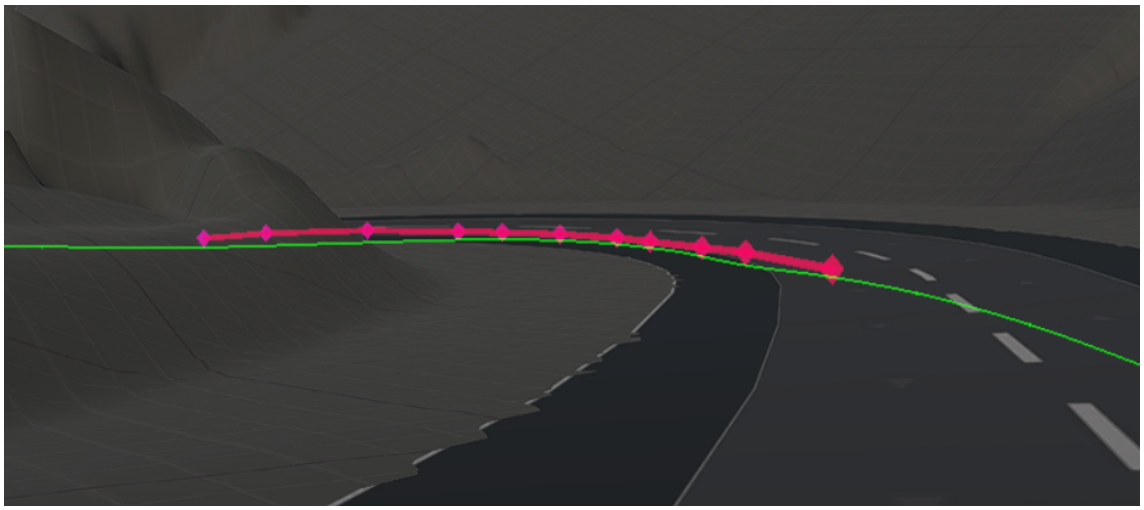


Figure 6.2: An image from our visual driver assistance prototype, allowing the player a longer reaction time as it is visible through the game world's terrain.

6.2.6 Give positive reinforcement for good driving

Let players know when they are performing well.

For players that are new to a game or a game genre it can be difficult knowing how to perform certain actions, for example drifting and overall timing of actions. It can also be difficult for the player to notice successful attempts. From our research, we believe that it would be beneficial for the player experience and enjoyment to help the player gain game knowledge by letting them develop good habits through pointing out successful experiences and successful difficult manoeuvres, for example timing steering in sharper curves leading to successful drifting. This could be done, for example, by providing the player with feedback such as visual indications. Apart from improved overall performance that will come from their learning, it is also likely to motivate the player to continue performing successful actions and increasing their enjoyment of the game.

Participants in our playtests said that they wish to learn and improve during gameplay. By giving positive feedback when players perform well, players know what to attempt to replicate next time. Possibly, rewards could be given to encourage player to repeat their successful actions. This guideline is supported by both the need for competence in self-determination theory [28] and the concepts of player skills and feedback in GameFlow [30]. However, if too intrusive it might instead be harmful to the player experience, since it may break immersion and flow [31] for the player.

6.2.7 Allow for mistakes while driving

Let the player's errors and mistakes in-game be opportunities for learning.

Assisting the player by correcting or reducing the effects of their mistakes might result in improved performance, but also risks hindering them from learning. By allowing the player to make mistakes, they are given the opportunity to learn. Learning during gameplay will in time lead to improved performance and is thus likely to increase their enjoyment. Letting the player make mistakes also makes their actions feel more meaningful as they might have negative impacts on the game, resulting in more challenging gameplay (see also sec. 6.2.4). For assisting a player without hindering them from improving, we recommend correcting the player's errors, or reducing their negative impacts, but still allowing some of the mistakes. An example would be to slightly adjust a player's steering to avoid collisions, without taking control away from them and obstructing their gameplay.

During the playtests, the participants expressed a wish to be able to learn and improve at the game. Assisting very much may remove mistakes entirely, which in turn makes learning and improving very difficult to achieve. An example of a very high level of assistance is the highest setting found in *Forza Motorsport 7*, where the player only have to hold down a single button to play and complete a race. This completely removes the possibility of improving, as it is also very difficult to act differently from how the game suggests and steers for the player. This is connected to our previous guideline concerning allowing the player to feel in control (see sec. 6.2.1). It is important that players can still improve even though they are getting assistance, so that they eventually will be able to play without it if they wish. This guideline is connected to the concept of player skills in GameFlow, in that it allows for "player skill development and mastery" [30] and the need for competence in self-determination theory, "to achieve mastery" [29].

6.2.8 Don't force driver assistance on the player

Allow the player to choose if they want help or not.

More experienced players may not want to use the assistance, especially if the assistance is a very noticeable part of the player experience. If these players are forced to use it, it may become more of annoyance than an aid. Due to the players' different skill levels and attitude towards being given assistance, we suggest that assistance should not be forced on players and that it should instead be optional by only giving it to them if and when they want it. For example, this can be achieved through having a setting in the game for enabling and disabling the assistance.

During our playtest interviews the more experienced participant was less interested than the others in having and using the visual assistance line, and believed that they would be more annoyed than helped by it when playing for longer. An example of lack of assistance flexibility is *Need for Speed (2015)* with its visual assistance line. The line showed a suggested path for a race, but could not be removed or hidden by the player. In contrast, *Forza Motorsport 7* offers an option for enabling or disabling

it. Seemingly, this unflexible assistance was not appreciated by the players, as it was not present in the sequel, *Need for Speed Payback*. Offering flexibility may also be more important in an open-world game, where the players are likely to expect more freedom than in a more limited game. This guideline concerns several aspects of both GameFlow [30], mainly control, but also by not limiting the concepts of player skills and challenge. Similarly, this guideline allows for the need for autonomy found in self-determination theory [28], but also the need for competence by allowing the player feel like they are succeeding with things on their own as they get better. It may also allow for more immersion in the game world if this can be turned off, which encourages flow [31].

7

Discussion

This chapter will discuss our results, as well as the process used during our project. It will also go through ethical issues connected to the thesis and possible future work, containing topics that are relevant to the project, but that we did not have enough time or resources to look further into.

7.1 Results

In this section, the results of the thesis will be discussed. The thesis has two kinds of results, and the section is therefore split into two parts: prototypes, and guidelines.

7.1.1 Prototypes

When developing our visual assistance prototype, we did not consider accessibility. Our visual prototype mainly uses red and green to differentiate between the different visual aids, which is the most common type of colour blindness. This issue was identified through one of the playtest participants that was colourblind. Fortunately, the sharp turns indications had a different shape than the line following the race track, making it possible to still distinguish them from each other.

The visual assistance prototype that was created was visible through the game world. This was not a conscious decision on our part, but a consequence of it being drawn using the existing debugging tools and not properly created in the game with the proper aesthetics due to that being more time-consuming. However, this turned out to be an appreciated feature of the assistance and was the basis for one of our guidelines.

It was noticed during the testing of the visual prototype that that the player tended to try to follow the line closely. In such a case, the blending assistance may be less noticeable as the players are already attempting to follow the path that the assistance is nudging towards. However, during our playtest we also noted that the less experienced players did not necessarily perform better when following the line. The line attempted to show an efficient path, which the players were not used to following, worsening their performance. This was discussed during the interviews and the less experienced players believed that they would have preferred a line that

showed the race course without trying to be as efficient, for example by showing the middle of the track. This could have been a possible further iteration, if we had had more time.

7.1.2 Guidelines

Our guidelines seem to align well with previous work and research findings regarding user interactions such as flow, GameFlow (see sec. 3.2.1) and self-determination theory (see sec. 3.2.2). However, some of our guidelines are otherwise somewhat weakly supported as they are mostly based on results from our playtests, whose issues are discussed in section 7.2.4. More rigorous playtests could have increased the validity of our guidelines.

We ended up formulating the guidelines quite late in the process. This lead to that the first real iteration of the guidelines was not connected enough to our early findings. It would probably have been better to incorporate the guideline thoughts and iterations into each iteration of the rest of our process. This would also have given us further time to reflect during the process, which possibly could have improved our iterative steps of improving the prototype. Some of the guidelines are still a little too vaguely connected to racing games. This lead to them also being somewhat abstract, and might result in that they are sometimes challenging to apply in practice.

7.2 Process

The process we used for structuring our work flow was closely inspired by the Wheel (see sec. 4.1.1). It was chosen because it is an iterative process, which is suitable for working with wicked problems such as our intended task. It was divided into phases that felt suitable for our project, and which let us structure our workflow in an organised way while still allowing for flexibility in our iterations. We chose to call the Design phase found in the Wheel process Ideation, since we felt that it better reflected what we wanted to do and achieve during that phase.

We chose to not follow Fullerton's [46] playcentric design process since it is aimed at structuring the workflow when creating a complete, new game. Since we were aiming to modify an already existing game base, we found it more suitable to instead use the Wheel process.

We initially did not think it necessary to analyse the user needs in great detail before and during the prototype development, and that it would be sufficient to evaluate during the development with expert users that we already had access to and possibly with external participants with the final prototypes. However, it proved difficult to continue building on the pre-existing prototype without knowing what players actually want and need help with. This lead us to performing a playtest for identifying what the players found challenging, but it would probably have been more beneficial if this had been done from the start.

7.2.1 Analysis

It could have been useful to create personas to analyse goals and needs of different kinds of players. Our results could have been compared against these and we might have better known our target audience to design driver assistance for. However, in order for us to be able to create good personas the properly represent the target audience, we would have needed significantly more thorough analysis of the player base. In the time span we had, the risk of having poor personas that did not properly represent the target group properly would have been quite high and may have resulted in an equally poor result.

For analysing players' needs, we intended to use the grounded theory method (see sec. 4.2.3), but ended up with a method more similar to content analysis 4.2.4 as previously mentioned in the process chapter. After having gotten a suggestion of using grounded theory, we investigated the method and thought that it would be a useful way of analysing our observation data. We also initially thought that creating a theory from the data could be a good start with forming our guidelines. It later turned out that the result was not very useful after all, since the categories we came up with seemed too abstract to continue our work on.

7.2.2 Ideation

In the Wheel [48], this phase is called Design. We felt that Ideation better represented what we wanted to achieve and actually did during this phase. Our goal during this phase was to come up with ideas and concepts that we could later create and implement. We also thought that "design" implies more practical work than merely idea generation and creating concepts, for example creating low-fidelity prototypes. In our process, the practical work of creating prototypes is mostly done with higher fidelity in code and on top of pre-existing work, which therefore suits better in a Prototyping phase.

7.2.3 Prototyping

During the development and prototyping phase, we had planned to use Kanban for structuring the software development. We did not end up following this, as the prototypes were too small in scope to benefit from splitting them up into multiple tasks. As our project was exploratory, we did also not wish to limit our ideas and attempts through planning too rigidly.

7.2.4 Evaluation

The prototypes used for our feedback sessions were selected by self-testing. Our evaluations of them were biased by preconceptions where we knew what the system was trying to achieve and knew what behaviours to expect. Since we looked for

certain behaviours, we probably noticed them and their flaws more than a regular player would. However, Fullerton [46] recommend first testing prototypes on yourself and then proceed with testing with others. Still, this may have caused that the prototypes selected for the feedback sessions were not the best ones.

In the feedback sessions, we tested on mostly expert players. They knew what to expect from the game and had a high level of knowledge of the game. This makes it likely that they themselves do not need assistance. This seemed to result in them being the most positive towards assistance systems that tried to help the least, as they felt obstructed when it tried to help more. These people may not have been ideal to test on, but they were chosen because the prototypes were in a very early stage of development and we were afraid that testing on external persons would negatively affect the result with lack of qualitative feedback. Ghost employees were also easily accessible and we needed to test our prototypes rather quickly to resolve issues, whereas organising a playtest with external participants would take quite some time. It was also uncertain if we were allowed to invite external person to the office at all, and we therefore had little or no choice but to test with employees. Our goal with these evaluations was also to guide us to future approaches and design choices, and not purely for user testing. We might have gotten more valuable feedback if the test had been performed in a way that allowed for more ongoing discussion.

Our evaluative playtests were performed with a very low number of participants, and unfortunately both playtests were performed with the same participants. Having the same participants made it more difficult to know if their improvement during the second playtest was due to learning from the first playtest or from the assistance helping them. However, since we performed formative testing it was less important for us to have a higher number of participants as it is the qualitative result from the test that matters, not the quantitative. Additionally, the unaccustomed setting in which the playtest was performed, us being present in the room during the playtest and being aware of that they were being recorded may have influenced the results. Fly-on-the-wall observations with secret outsiders (see sec. 4.2.6) may have given better, more unbiased results.

7.3 Ethical issues

Changing the behaviour of a player's avatar or adapting the game state based on the player's performance, with or without their consciousness, could be critical as it can be both misleading for the player getting assistance as well as it might be considered unfair, especially in an environment involving other human players. This section aims to discuss some of the ethical issues that may arise when implementing player assistance in a game.

7.3.1 Tricking the player

An ethical issue regarding the work of this thesis is if it is appropriate to lead the player to believe that they are performing better than they actually are. Due to the goals of the game world, this is probably not an issue as the game is not a simulation of real world physics and is thus more forgiving for minor vehicle behaviour changes, as long as it is not intrusive for the player's game experience.

7.3.2 Hindering player improvement

Another possible issue with the proposed driver assistance is how it affects a player's improvement and future performance. Linderoth [68] poses scepticism towards the fact that a game being considered good automatically means that players improve their skills while playing it. However, Gutwin, Vicencio-Moreira, and Mandryk [69] examined the difference between improvement in a first person shooter (FPS) game with and without aim assistance, and found that both types of players did in fact improve, and the assisted players found the game more enjoyable. It is worth noting the assistance method used by Gutwin, Vicencio-Moreira, and Mandryk [69] did not take away the need for assisted players to aim, but they did not need to aim quite as accurately as the unassisted players, making the game more forgiving. Thus, this may be something that we should keep in mind and it may be reasonable to create a flexible prototype that can give several different levels of assistance to the player. However, this will not be the main goal of the prototype, as it would probably require more time to find an appropriate level for all types of players.

7.3.3 Taking away player agency

During our project, we had issues regarding the assistance feeling very intrusive when it tried to help the player steer the car. For identifying the underlying problem, we compared our assistance system with the one in *Mario Kart 8 Deluxe* where the steering assistance is often very subtle.

In *Mario Kart 8 Deluxe* there are obstacles around the track, making it impossible to actually drive outside. Trying to escape the track makes you hit an invisible wall, but often the wall does not feel intrusive as there are physical fences visualising this barrier. Even though most track edges have some kind of fence, there are some more open areas which still has this invisible wall to hinder players from exiting the track. Near this wall, you seem to get more and more assistance as you approach it. You do not seem to get much assistance when driving in the middle of the track; the game helps keeping you on the track in aiding your steering instead of helping you drive along the optimal path as our assistance systems tried to do.

The assistance in *Mario Kart 8 Deluxe* is often not very noticeable while you play as expected, i.e. trying to win. However, if you try to fail by driving in the wrong direction or outside the track the assistance actually becomes very intrusive as it forces you to go the right way. A speculation why the assistance is not irritating to

play with is partly that the game has a very cartoon- and arcade-like setting, but mostly due to the barriers at the edges of the track; most of the time you simply cannot drive outside it. Also, there is a lot of action happening all the time. Players pick up coins and power-ups, try to avoid obstacles (both placed on the map by the game, and others thrown by players), hit opponents with projectiles, and therefore players might not always notice the assistance as they are busy with other tasks. Assistance might also therefore be more welcome since it decreases the player's need of multitasking.

7.3.4 Use in competitive environments

For player assistance in single player games or game sessions, the aid only impacts one player. For racing games however, it is not unusual for multiple players to play together and compete against each other. In *Need for Speed Payback*, apart from regular multiplayer there is a game mode where the player can compete against other players of equal skill level, where the players obtain different ranks representing their prestige [70]. In this mode, wins lead to an increase in rank while losses results in a decrease. In such competitive environments, is it still appropriate to boost the performance of an underperforming player? Even though it can seem unfair for the players that are ahead to give an advantage to poorly performing players, it is a mechanic that can already be found in existing games. One such example is *Mario Kart*, where the players in the back in a race get better items for catching up with the other players [1]. However, one could argue if the setting of *Mario Kart 8 Deluxe* is as competitive as the one in ranked matches in *Need for Speed Payback*.

7.4 Future work

During the thesis, several related areas were found that could be interesting to look into further when designing a driver assistance system for a racing game. The topics listed in this section was considered during this project, but deemed out of scope for the thesis itself.

7.4.1 Player prediction

To get better assistance, it would be useful to better predict where the players will and wants to move, as right now the driver assistance assumes that the player want to follow the same path as the game AI would take.

For drivers of real-life vehicles, their mental state and personality may influence their behaviour in different driving situations [7]. This is likely to also apply to players, and since individuals may behave differently in different situations, it might be difficult to adapt vehicle assistance based only on the driver's current actions. Additionally, in games some players are more experienced than others and games

can impact on a player's mood. Therefore, the same solution should probably not be applied to all players.

More individual predictions should be possible to do through using player data from the current player and earlier players [71, 72]. A possible approach is something similar to what was used by Harrison and Roberts [71], where an algorithm is used that examines the actions of the current player to those that has been done by previous players, and groups the players into groups depending on their actions and how they correspond to previous players. This kind of approach could also be used to choose what types of content in the game that the player should be recommended [71]. It might also be possible to use machine learning such as artificial neural networks to categorise players and player types.

7.4.2 Player analysis

To get more focused and targeted driver assistance in racing games it would be beneficial to better identify the target audience of such an assistance system, as that determines the best way to aid the players. At the start of this thesis, we decided that the target audience for our assistance system would be players above the age of 12, since that is the age rating of similar games. The aim was to assist players with a low racing game skill level. This meant that the assistance was not meant for players performing extremely poorly, or players with a high racing game experience. This may not have been the correct target audience. For example, we did not consider players with special needs. Examples of such players are children, people with disabilities, and elders. These players might be the most critical to assist, and therefore future work should consider targeting them when designing driver assistance.

A better conducted analysis of the player base and the target audience would also be beneficial to get more strongly validated results. The participants should be more carefully selected to represent the intended player base and target audience for the driver assistance. Additionally, a larger number of participants is probably needed in order to get a group that is representative of the target audience. The results of a more rigorous analysis could be used throughout the process of designing a driver assistance system. That would likely lead to more helpful and appreciated assistance that be more likely would help with the most important areas. Possibly, personas (see sec. 4.2.8) could be created and used for this, in order to identify areas where different kind of players might need and want assistance.

7.4.3 User testing

To achieve a more conclusive answer on the efficacy and player experience of different driver assistance prototypes and mechanics, more thorough user testing is needed. Firstly, the participants should better represent the player base, as discussed in section 7.4.2. The tests should probably also be performed with more people and multiple times. Nielsen [73] suggests that 5 participants is a suitable number for

performing usability tests, and to test after each iteration of the design. When conducting multiple user tests it is probably acceptable, and perhaps beneficial, to invite some of the same participants again, but there should probably also be new people in order to get fresh insights.

Apart from just improving the user testing method itself, it could also be valuable to test the blending prototypes on the identified target audience. Unsuccessful results from testing of the blending prototypes might not be due to issues with the implementation or concept, but could also arise from that they might have been tested on the wrong audience. While they seemed to be unhelpful for the players they were tested on, they could potentially be better suited for other types of players. For example, the assistances could be helpful for the different groups previously mentioned in section 7.4.2.

7.4.4 Multiplayer

In multiplayer games player assistance can be more difficult to implement as it may detract from an opponent's experience, for example if you aid one player by slowing down opponents. Therefore, for assisting a player in a racing game in a multiplayer environment other solutions would probably be needed. Even though catch-up mechanics exist in multiplayer races, such as *Mario Kart 8 Deluxe* they can be perceived as unfair for the other players. Game developers and designers should therefore be extremely cautious when trying to balance a game according to the players' skill levels, to get a healthy balance of challenge and fair play.

8

Conclusion

During this thesis, we have explored the possibility of driver assistance in a racing game with help from AI. The research question that guided us through our work was:

"What factors should be considered when implementing driver assistance in a racing video game to aid players with less experience?"

The project was performed in collaboration with Ghost Games. All work was done in an iterative fashion, in a total of three different iterations. Two real-time assistance prototypes were created, which both use the "optimal" path that the game's AI generates, by checking the player's position against this path and then applies player input blending with game AI depending on the result. These were called "soft blending" and "hard blending". Other similar solutions were also explored, but they were never completed due to time constraints. A third prototype was also created, which uses visual cues for guiding the player during a race.

Based on findings from playtests performed with our prototypes and theory we encountered during our thesis work, guidelines were created on what to consider when implementing driver assistance in a racing game. The guidelines we came up with in order to answer our research question are:

- **Let the player feel in control of the car:** Don't take control from the player, let them feel like they are choosing what they want to do.
- **The vehicle should behave as expected:** Match player expectations of vehicles; both real world and in-game.
- **Even out the playing field for the player:** Make the player feel like they have a possibility of winning.
- **Present the player with a challenge:** Don't give the player a guaranteed or very easy win from using the assistance.
- **Allow for planning ahead during races:** Allow players longer reaction time in order to compensate for lack of learned habits and game knowledge.
- **Give positive reinforcement for good driving:** Let players know when they are performing well.

- **Allow for mistakes while driving:** Let the player's errors and mistakes in-game be opportunities for learning.
- **Don't force driver assistance on the player:** Allow the player to choose if they want help or not.

These guidelines aim to give indications on factors that could positively impact the player experience for players of lower skill levels. However, these guidelines are tailored towards the type of racing games that have been investigated during this thesis and thus might not be applicable to all types of racing games or other games in general. For creating driver assistance in a racing game, these guidelines should only be used as food for thought since more work and refinement on them is needed before they can be fully followed. For example, a proper player analysis is necessary in order to get an accurate understanding of the goals and needs of the target audience.

Bibliography

[1] Rodrigo Vicencio-Moreira, Regan L Mandryk, and Carl Gutwin. "Balancing multiplayer first-person shooter games using aiming assistance". In: *2014 IEEE Games Media Entertainment*. IEEE. 2014, pp. 1–8.

[2] Changchun Liu, Pramila Agrawal, Nilanjan Sarkar, and Shuo Chen. "Dynamic difficulty adjustment in computer games through real-time anxiety-based affective feedback". In: *International Journal of Human-Computer Interaction* 25.6 (2009), pp. 506–529.

[3] Barbaros Bostan and Sertaç Öğüt. "Game challenges and difficulty levels: lessons learned From RPGs". In: *International simulation and gaming association conference*. 2009.

[4] Karel A Brookhuis, Dick De Waard, and Wiel H Janssen. "Behavioural impacts of advanced driver assistance systems–an overview". In: *European Journal of Transport and Infrastructure Research* 1.3 (2001), pp. 245–253.

[5] Meng Lu, Kees Wevers, and Rob Van Der Heijden. "Technical Feasibility of Advanced Driver Assistance Systems (ADAS) for Road Traffic Safety". In: *Transportation Planning and Technology* 28.3 (2005), pp. 167–187. DOI: 10.1080/03081060500120282.

[6] SAE International. *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*. June 2018.

[7] Myounghoon Jeon. "Towards affect-integrated driving behaviour research". In: *Theoretical Issues in Ergonomics Science* 16.6 (2015), pp. 553–585.

[8] Elisabeth Wells-Parker, Jennifer Ceminsky, Victoria Hallberg, Ronald W. Snow, Gregory Dunaway, Shawn Guiling, Marsha Williams, and Bradley Anderson. "An exploratory study of the relationship between road rage and crash experience in a representative sample of US drivers". In: *Accident Analysis & Prevention* 34.3 (2002), pp. 271–278. ISSN: 0001-4575. DOI: https://doi.org/10.1016/S0001-4575(01)00021-5.

[9] Jerry L Deffenbacher, Eugene R Oetting, and Rebekah S Lynch. "Development of a driving anger scale". In: *Psychological reports* 74.1 (1994), pp. 83–91.

[10] Scott Bateman, Regan L Mandryk, Tadeusz Stach, and Carl Gutwin. "Target assistance for subtly balancing competitive play". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM. 2011, pp. 2355–2364.

[11] Jared E Cechanowicz, Carl Gutwin, Scott Bateman, Regan Mandryk, and Ian Stavness. "Improving player balancing in racing games". In: *Proceedings of*

the first ACM SIGCHI annual symposium on Computer-human interaction in play. ACM. 2014, pp. 47–56.

[12] Nintendo. *Mario Kart 8 Deluxe*. 2017. URL: https://www.nintendo.com/games/detail/mario-kart-8-deluxe-switch (visited on 2019-02-12).

[13] Nintendo. *Nintendo*. 2019. URL: https://www.nintendo.com/ (visited on 2019-02-04).

[14] Russell Holly. *Image of a Mario Kart 8 Deluxe race*. 2017. URL: https://www.imore.com/how-perform-new-super-drift-boost-mario-kart-8-deluxe (visited on 2019-06-20).

[15] Electronic Arts Inc. *Real Racing 3*. 2013. URL: https://www.ea.com/sv-se/games/real-racing/real-racing-3 (visited on 2019-01-31).

[16] Electronic Arts Inc. *Firemonkeys Studios*. 2018. URL: https://www.ea.com/studios/firemonkeys (visited on 2019-01-31).

[17] Electronic Arts Inc. *Electronic Arts*. 2019. URL: https://www.ea.com/ (visited on 2019-01-30).

[18] Electronic Arts Inc. *Ghost Games*. 2019. URL: https://www.ea.com/studios/ghost-games (visited on 2019-01-23).

[19] Electronic Arts Inc. *Need for Speed Payback*. 2017. URL: https://www.ea.com/en-gb/games/need-for-speed/need-for-speed-payback (visited on 2019-01-25).

[20] Electronic Arts Inc. *Need for Speed*. 2015. URL: https://www.ea.com/games/need-for-speed/need-for-speed-2015 (visited on 2019-06-20).

[21] Microsoft. *Microsoft Studios*. 2019. URL: https://www.microsoftstudios.com/ (visited on 2019-01-31).

[22] Microsoft Corporation. *Forza Motorsport*. 2019. URL: https://www.forzamotorsport.net/ (visited on 2019-01-30).

[23] Microsoft Corporation. *Forza Horizon*. 2019. URL: https://www.forzamotorsport.net/en-us/games/fh (visited on 2019-01-30).

[24] Ben Gilbert. *Image of Forza Horizon 4*. 2018. URL: https://www.businessinsider.com/forza-horizon-4-review-2018-10 (visited on 2019-06-20).

[25] William Gaver. "What should we expect from research through design?" In: *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM. 2012, pp. 937–946.

[26] Bruce Hanington and Bella Martin. *Universal methods of design. 100 ways to research complex problems, develop innovative ideas, and design effective solutions*. Rockport Publishers, 2012.

[27] Richard M Ryan and Edward L Deci. "Intrinsic and extrinsic motivations: Classic definitions and new directions". In: *Contemporary educational psychology* 25.1 (2000), pp. 54–67.

[28] Richard M Ryan and Edward L Deci. "Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being." In: *American psychologist* 55.1 (2000), p. 68.

[29] Celia Hodent. *The Gamer's Brain: How Neuroscience and UX Can Impact Video Game Design*. CRC Press, 2017.

[30] Penelope Sweetser and Peta Wyeth. “GameFlow: a model for evaluating player enjoyment in games”. In: *Computers in Entertainment (CIE)* 3.3 (2005), pp. 3–3.

[31] Mihály Csíkszentmihályi. *Flow: The Psychology of Optimal Experience*. Harper and Row, 1990.

[32] Steve Rabin. *Introduction to Game Development, Second Edition*. Cengage Learning, 2010.

[33] Jenova Chen. “Flow in games (and everything else)”. In: *Communications of the ACM* 50.4 (2007), pp. 31–34.

[34] Edward Deci and Richard Ryan. “The "What" and "Why" of Goal Pursuits: Human Needs and the Self-Determination of Behavior”. In: *Psychological Inquiry* 11 (Oct. 2000), pp. 227–268. DOI: 10.1207/S15327965PLI1104_01.

[35] Edward Deci and Richard M Ryan. *Intrinsic Motivation and Self-Determination in Human Behavior*. Perspectives in Social Psychology. Springer US, 1985. ISBN: 9780306420221.

[36] Stuart Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Pearson Education Limited, 2013.

[37] Ian Millington and John Funge. *Artificial intelligence for games. Second edition*. CRC Press, 2009.

[38] Kimberly Voll. *Less is More: Designing Awesome AI for Games*. Game Developers Conference. 2015. URL: https://www.youtube.com/watch?v=1xWg54mdQos.

[39] Vahid Rahmani and Nuria Pelechano. “Improvements to Hierarchical Pathfinding for Navigation Meshes”. In: *Proceedings of the Tenth International Conference on Motion in Games*. MIG ’17. Barcelona, Spain: ACM, 2017, 8:1–8:6. ISBN: 978-1-4503-5541-4. DOI: 10.1145/3136457.3136465. URL: http://doi.acm.org/10.1145/3136457.3136465.

[40] Steve Rabin. *Game AI Pro: Collected Wisdom of Game AI Professionals*. Taylor & Francis, 2013. ISBN: 9781466565968.

[41] Kevin P. Murphy. *Machine Learning : A Probabilistic Perspective*. MIT Press, 2012.

[42] Jonathan Schaeffer, Vadim Bulitko, and Michael Buro. “Bots get smart”. In: *IEEE Spectrum* 45.12 (2008), pp. 48–56.

[43] Mirna Paula Silva, Victor do Nascimento Silva, and Luiz Chaimowicz. “Dynamic difficulty adjustment on MOBA games”. In: *Entertainment Computing* 18 (2017), pp. 103–123.

[44] Pieter Spronck, Marc Ponsen, Ida Sprinkhuizen-Kuyper, and Eric Postma. “Adaptive game AI with dynamic scripting”. In: *Machine Learning* 63.3 (2006), pp. 217–248.

[45] Jonathan Schaeffer. “A gamut of games”. In: *AI Magazine* 22.3 (2001), p. 29.

[46] Tracy Fullerton. *Game Design Workshop. A Playcentric Approach to Creating Innovative Games, Third Edition*. A K Peters/CRC Press, 2014.

[47] Horst WJ Rittel and Melvin M Webber. “Wicked problems”. In: *Man-made Futures* 26.1 (1974), pp. 272–280.

[48] Rex Hartson and Pardha S. Pyla. *The UX Book. Process and Guidelines for Ensuring a Quality User Experience*. Elsevier, 2012. ISBN: 978-0-12-385241-0.

[49] A. Cooper, R. Reimann, D. Cronin, and C. Noessel. *About Face: The Essentials of Interaction Design*. Wiley, 2014. ISBN: 9781118766583.

[50] Jakob Nielsen. *Parallel & Iterative Design + Competitive Testing = High Usability*. Ed. by Nielsen Norman Group. 2011. URL: https://www.nngroup.com/articles/parallel-and-iterative-design/ (visited on 2019-04-16).

[51] Nathan Hook. "Grounded theory". In: *Game Research Methods*. Ed. by Petri Lankoski and Staffan Björk. ETC Press, 2015, pp. 309–320.

[52] Kathy Charmaz. *Constructing grounded theory: A practical guide through qualitative analysis*. Sage, 2006.

[53] Haakon Faste, Nir Rachmel, Russell Essary, and Evan Sheehan. "Brainstorm, Chainstorm, Cheatstorm, Tweetstorm: new ideation strategies for distributed HCI design". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM. 2013, pp. 1343–1352.

[54] Agile Alliance. 2019. URL: https://www.agilealliance.org/ (visited on 2019-01-29).

[55] Agile Alliance. *Scrum*. 2019. URL: https://www.agilealliance.org/glossary/scrum/ (visited on 2019-01-29).

[56] Agile Alliance. *Kanban*. 2019. URL: https://www.agilealliance.org/glossary/kanban/ (visited on 2019-01-29).

[57] Agile Alliance. *Pair Programming*. 2019. URL: https://www.agilealliance.org/glossary/pairing/ (visited on 2019-01-30).

[58] Günther Gediga, Kai-Christoph Hamborg, and Ivo Düntsch. "The IsoMetrics usability inventory: an operationalization of ISO 9241-10 supporting summative and formative evaluation of software systems". In: *Behaviour & Information Technology* 18.3 (1999), pp. 151–164.

[59] Google. *Google Scholar*. 2019. URL: https://scholar.google.se/.

[60] Chalmers University of Technology. *Chalmers Library*. 2019. URL: http://www.lib.chalmers.se/.

[61] ACM. *ACM Digital Library*. 2019. URL: https://dl.acm.org/.

[62] IEEE. *IEEE Xplore Digital Library*. 2019. URL: https://ieeexplore.ieee.org/Xplore/home.jsp.

[63] DiGRA. *DiGRA Digital Library*. 2019. URL: http://www.digra.org/digital-library/.

[64] Microsoft. *Visual Studio*. 2019. URL: https://visualstudio.microsoft.com/ (visited on 2019-06-07).

[65] John I Lacey and Beatrice C Lacey. "Verification and extension of the principle of autonomic response-stereotypy". In: *The American journal of psychology* 71.1 (1958), pp. 50–73.

[66] Alexander Baldwin, Daniel Johnson, and Peta A. Wyeth. "The Effect of Multiplayer Dynamic Difficulty Adjustment on the Player Experience of Video Games". In: *CHI '14 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '14. Toronto, Ontario, Canada: ACM, 2014, pp. 1489–1494. ISBN: 978-1-4503-2474-8. DOI: 10.1145/2559206.2581285. URL: http://doi.acm.org/10.1145/2559206.2581285.

[67] Jonathan Bailey. *Why we play video games*. 2018. URL: https://www.gamasutra.com/blogs/JonathanBailey/20180912/325742/Why_we_play_video_games.php (visited on 2019-01-29).

[68] Jonas Linderoth. "Why gamers don't learn more: An ecological approach to games as learning environments". In: *Journal of Gaming & Virtual Worlds* 4.1 (2012), pp. 45–62.

[69] Carl Gutwin, Rodrigo Vicencio-Moreira, and Regan L Mandryk. "Does Helping Hurt?: Aiming Assistance and Skill Development in a First-Person Shooter Game". In: *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play*. ACM. 2016, pp. 338–349.

[70] Electronic Arts Inc. *Under the Hood: Multiplayer Speedlists - Need for Speed Payback*. n.d. URL: https://www.ea.com/games/need-for-speed/need-for-speed-payback/news/nfs-payback-multiplayer-speedlists (visited on 2019-01-25).

[71] Brent Harrison and David L Roberts. "Using sequential observations to model and predict player behavior". In: *Proceedings of the 6th International Conference on Foundations of Digital Games*. ACM. 2011, pp. 91–98.

[72] Christoffer Holmgård, Antonios Liapis, Julian Togelius, and Georgios N Yannakakis. "Generative agents for player decision modeling in games." In: *FDG*. Citeseer. 2014.

[73] Jakob Nielsen. *Why You Only Need to Test with 5 Users*. Ed. by Nielsen Norman Group. 2000. URL: https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/ (visited on 2019-06-10).