Investigating the Effects of Adaptive Guidance in Simulation-Based Inquiry Learning Environments: A Focus on Prior Knowledge, Ability Level, and Self-Regulated Learning

Louis van Maurik

Faculty of Behavioural, Management and Social sciences - University of Twente

M.Sc. Thesis

dr. A. van Dijk

prof. dr. T.H.S. Eysink

April 2025

Abstract

Simulation-based inquiry learning can be less effective when learners lack sufficient guidance. One proposed solution is adaptive guidance, which adjusts the available guidance to individual learners' needs. However, the extent to which personal attributes influence the type and amount of guidance required remains an open question. Therefore, this study investigated the effects of different moderating variables on the effectiveness of adaptive guidance in a simulation-based inquiry learning environment. A total of 72 fifth- and sixth-grade students participated in a pre-test/post-test design, with one condition receiving adaptive guidance and the other receiving no guidance. While both conditions showed significant improvement in absolute scores on the post-test, resulting in a positive normalized knowledge gain, no significant difference was found between the two conditions. Further analysis explored the role of prior knowledge, ability level, and self-regulated learning skills as potential moderators, but no significant effects were found. These findings suggest that adaptive guidance, as implemented in this study, may not be a decisive factor in improving learning outcomes and highlight the need for further research on which factors influence the effectiveness of adaptive guidance.

Contents

1. Introduction

Inquiry learning, a well-known educational method that involves students discovering scientific ideas through participation in inquiry processes, is widely recognized as a valuable instructional approach in science education (Minner et al., 2009; White & Frederiksen, 1998). By engaging in inquiry activities, students can begin to develop proficiency in the scientific reasoning skills of hypothesis generation, experimentation, and evidence evaluation (Lazonder & Kamp, 2012).

The use of computer simulations has emerged as an effective way to further enhance this approach (Blake & Scanlon, 2007; Lai et al., 2018; Rieber et al., 2004). This method, known as simulation-based inquiry learning, offers several advantages over hands-on inquiry learning: it provides a safer environment for exploration (de Jong, 2006), shows phenomena that are usually hidden (Olympiou et al., 2013; Windschitl, 2000), and simplifies and/or highlights key aspects to help with understanding (van Joolingen et al., 2007).

Although simulation-based inquiry learning offers significant potential, one major challenge is that inquiry learning (whether simulation-based or in real life) may not always lead to optimal results without appropriate guidance (Hmelo-Silver et al., 2007; Mayer, 2004). Research has shown that students have difficulties during the inquiry process, such as choosing appropriate variables (Chinn & Brewer, 1993; Klahr & Dunbar, 1988), testing their formulated hypotheses (Kuhn et al., 1992; van Joolingen & de Jong, 1991), and drawing valid conclusions (Klahr & Dunbar, 1988; Kuhn et al., 1992). Without sufficient guidance, these difficulties can impede learning and limit the effectiveness of inquiry-based approaches (Alfieri et al., 2011).

Although guided learning has been shown to be more effective than unassisted learning, its impact can vary based on individual learner attributes, such as prior knowledge, ability level, and self-regulated learning (Faber et al., 2023; van Dijk et al., 2016; S. van Riesen et al., 2018b).

For instance, van Riesen et al. (2018) found that low prior knowledge students gained more from a more scaffolded simulation-based environment than high prior knowledge students. Van Dijk et al. (2016) found a difference in use of prompts within a simulation-based inquiry environment, where high ability students significantly used more prompts when available, compared to low ability students. Finally, Faber et al. (2023) demonstrated that, within inquiry learning, students who scored high on a self-regulated learning questionnaire also had better task performance.

A recent development and advantage of simulation-based environments is that some have built-in adaptive guidance to account for individual differences between learners (Brenner et al., 2017; Graesser & McNamara, 2010). These environments allow for real-time assessments to be conducted, which enable tailored instructional supports (Timms et al., 2012).

However, the extent to which personal attributes influence the effectiveness of adaptive guidance remains largely unknown. To our knowledge, only one study has examined the role of personal attributes, specifically prior knowledge, in an environment with adaptive guidance (Brenner et al., 2017). The results of this study showed that prior knowledge alone is not sufficient to determine when students will differentially benefit from assistance. The authors suggest that additional factors should be explored in future research.

The goal of this research is to investigate the effects of adaptive guidance in a simulation-based inquiry learning environment on students' learning outcomes. Specifically, it examines the effectiveness of adaptive guidance on learning outcomes based on prior knowledge, ability level, and self-regulated learning. The findings of this research could contribute to the development of simulation-based environments that automatically adjust their support based on students' personal attributes, ultimately leading to improved learning outcomes in scientific inquiry.

2. Theoretical Framework

2.1 Inquiry Learning

Inquiry learning is a prominent educational approach in which a learner discovers scientific concepts by taking part in the inquiry cycle (White & Frederiksen, 1998; Wichmann & Leutner, 2009). While scholars have debated the precise definition of inquiry learning (Alfieri et al., 2011; Klahr & Nigam, 2004; Strike, 1975), Lazonder & Harmsen (2016) defined this educational approach as one in which students conduct experiments, make observations or collect information in order to infer the principles underlying a topic or domain.

2.1.1 Students' difficulties within the inquiry cycle

Different scholars have defined the phases of the inquiry learning cycle differently, each with a different approach (de Jong, 2006; Kuhn et al., 2008; Lim, 2004; National Research Council, 1996; White & Frederiksen, 1998). Pedaste et al. (2015) summarised these different approaches in a review study into five core phases: orientation, conceptualization, investigation, conclusion, and discussion.

In the *orientation phase*, the topic of the investigation is explored to build a foundational understanding. This basic understanding is essential, as it would be very difficult or even impossible for students to formulate a meaningful research question or hypothesis and to design a useful experiment without it (Quintana et al., 2004).

In the following phase, the *conceptualization phase*, the formulation of the research question(s) or hypotheses takes place. Within this phase, students struggle with choosing which variables are appropriate to work with (Klahr & Dunbar, 1988), including identifying the correct independent variable (Richardson, 2008). They have difficulties setting goals (Charney et al., 1990), and often do not plan which experiments to run (Glaser et al., 1992). Students also find it challenging to understand the relationship between theoretical variables and the variables they can manipulate in their experiments (Glaser et al., 1992; van Joolingen & de Jong, 1997).

During the *investigation phase,* students struggle to test their formulated hypotheses (Kuhn et al., 1992; van Joolingen & de Jong, 1991) or may fail to acquire sufficient evidence to test their hypotheses (Schauble et al., 1991). This often happens because students either run just a single trial or repeatedly run the same trial without any changes (Kuhn et al., 1992). On the contrary, students also make the mistake of changing too many variables at once (Glaser et al., 1992; Richardson, 2008; Schunn & Anderson, 1999). They tend to try to achieve a certain outcome or may design experiments that are fun to execute or watch (Schauble et al., 1991, 1995), and often forget to monitor and record their process while doing so (De Jong, 2006; Jong et al., 2005).

Based on the performed experiments, students then draw conclusions in the *conclusion phase*. Here, students tend to favor their initial belief/hypothesis, even when confronted with negative results (Dunbar et al., 1993; Klahr & Dunbar, 1988; Klayman & Ha, 1987; Quinn & Alessi, 1994). Students may also draw conclusions from confounded data (Klahr & Dunbar, 1988; Kuhn et al., 1992), fail to connect outcomes of an experiment to theories being tested (Schunn & Anderson, 1999) or have difficulties with linking data back to the hypotheses (Chinn & Brewer, 1993; Klahr & Dunbar, 1988).

Finally, in the *discussion phase*, which according to Pedaste et al. (2015) can take place either after each individual phase or at the cycle's end, students reflect on their inquiry process and communicate their findings. Within this phase, students tend to have difficulties articulating and defending their claims (Sadler, 2004). They may also find it challenging to provide reasoning to describe why evidence supports claims (McNeill & Krajcik, 2007).

2.1.2 Guidance in inquiry learning

As students often lack the inquiry skills needed to complete systematic scientific inquiry, ensuring effective guidance during inquiry learning is of great importance. Research consistently demonstrates that minimally guided or unguided inquiry is less effective and efficient than guided inquiry learning (Hmelo-Silver et al., 2007; Kirschner et al., 2006; Mayer, 2004), which has also been confirmed in a number of overview studies (Lazonder & Harmsen, 2016; Rutten et al., 2012; Smetana & Bell, 2012). Lazonder and Harmsen (2016) defined guidance in inquiry-based learning as any form of assistance offered before and/or during the inquiry learning process that aims to simplify, provide a view on, elicit, supplant, or prescribe the scientific reasoning skills involved.

Early classifications of guidance in inquiry learning focused on inquiry phases (de Jong, 2006) or reasoning processes (Lazonder, 2014) but did not define the type of guidance. Although Jong and Lazonder (2014) proposed a framework based on guidance type, some terminological inconsistencies persisted (Lazonder & Harmsen, 2016). For example, similar guidance strategies had been labeled differently, like "mechanistic cues" (Kaplan & Black, 2003) and "hints" (Rey, 2011), complicating research comparisons (Klahr, 2013).

Sun et al. (2022) integrated these existing typologies (Lazonder & Harmsen, 2016; Zacharia et al., 2015) into six guidance types: process constraints, prompts, heuristics, scaffolds, metacognitive supports, and direct presentation of information. Process constraints simplify the complexity of inquiry tasks by limiting their scope, while prompts and heuristics provide reminders or instructions to guide learners’ actions, with heuristics offering more detailed guidance than prompts. Scaffolds serve as structural supports that help learners manage complex tasks and organize their learning processes. Metacognitive supports help learners monitor and regulate their cognitive processes. Direct presentation of information provides learners with background knowledge or other information related to the learning task.

2.2 Adaptive Guidance within Simulation-Based Environments

Simulation-based environments have emerged as a promising approach to integrating guidance into learning. Simulation-based environments are computational interactive models that allow users to dynamically explore situations or natural phenomena by manipulating or modifying parameters within them (National Research Council, 2011). Ainsworth and Van Labeke (Ainsworth & VanLabeke, 2004) further defined instructional simulations as tools that allow learners to conduct experiments.

One major advantage of simulation-based environments is their ability to enable “on-the-fly” assessments, which in turn allows for adaptive guidance while students progress throughout the environment (Timms et al., 2012). Research shows that one-on-one human tutoring is highly effective, with experienced tutors achieving a two standard deviation improvement over traditional classroom instruction (Bloom, 1984). Simulation-based environments that mimic aspects of human tutors have also been shown to be successful (Brenner et al., 2017; Graesser & McNamara, 2010; Long & Aleven, 2013; Poitras & Lajoie, 2014).

As an example, AutoTutor monitors students' cognitive and metacognitive activities based on their interactions with a computer tutor (Graesser & McNamara, 2010). The computer tutor, also referred to as pedagogical agent, intervenes using several types of dialogue strategies (e.g., prompts, questions, and corrections), which have been shown to lead to positive gains on learning outcomes (Graesser & McNamara, 2010).

Another example would be the environment developed by Brenner et al. (2017), where they explored how the frequency and level of assistance impacted learning in the Voyage to

Galapagos (VTG) environment, where students simulate Darwin's journey through the Galapagos Islands, collecting data and making observations similar to his. The used a Bayesian Network, which consisted of four layers: supportable events layer, diagnosis layer, error evaluation layer, and Knowledge, Skills, and Abilities evaluation layer. Notably, all detectors for both positive and negative actions are integrated within the supportable events layer. For instance, if a student failed to photograph an animal at one location, tried to go to the next location, and had fewer than four images, the system would detect the error. The pedagogical tutor would then give feedback appropriately.

2.3 Personal attributes

As adaptive guidance has the capability to adjust the amount of guidance on-the-fly, one possibility would be to pre-adjust the amount or level of guidance the student would receive during their inquiry learning cycle, based on certain personal attributes. Within this study, we would like to focus on three attributes: prior knowledge, ability level, self-regulated learning skills.

2.3.1 Prior Knowledge

Effect of prior knowledge on learning outcomes. Research on expert-novice differences in the last decades has clearly demonstrated that prior knowledge affects the effectiveness of inquiry learning (Kalyuga, 2007; S. van Riesen et al., 2018a). Students with limited prior domain knowledge, conceptual knowledge about a specific topic within the larger domain, often use less effective inquiry strategies and require more trials to draw conclusions, while their more knowledgeable peers adopt structured, goal-oriented approaches (P. A. Alexander & Judy, 1988; Schauble et al., 1991). Hattie and Donoghue (2016) confirmed these findings through a meta-synthesis of numerous meta-analyses. They stated that active forms of learning, such as inquiry learning, have a significant impact on deep knowledge acquisition, but only when learners have already acquired the necessary surface knowledge. A lack of experience and essential surface knowledge causes students to make more errors in their understanding, incorporate irrelevant information into their explanations, and rely on backward reasoning. In contrast, experts employ forward reasoning (Albanese & Mitchell, 1993).

Effect of prior knowledge on the effectiveness of guidance. In addition to the influence of prior knowledge on learning in general, research also shows that students with different levels of prior knowledge require different forms of guidance (Blayney et al., 2010; Kalyuga, 2007; Lambiotte & Dansereau, 1992; Tuovinen & Sweller, 1999). For example, Blayney et al. (2010) examined the interaction between the isolated–interactive elements effect and learners' prior knowledge. They found that low prior knowledge students benefited from isolated instructional methods, while high prior knowledge students learned more effectively with interactive formats. Similarly, Tuovinen and Sweller (1999) found that low prior knowledge students learn better from worked examples, while high prior knowledge students show no advantage due to their existing schemas. Kalyuga (2007) described this as the "expertise reversal effect," where guidance helpful to novices may be redundant or even hinder experts. However, some research suggests that when domain-specific or strategy knowledge is extremely low, even instructional formats designed to support learning may fail to be effective or could even have detrimental effects (Clark, 1990; Seufert, 2003).

Effect of prior knowledge on the effectiveness of guidance within simulation-based environments. Research also investigated if the expertise reversal effect translated to simulation-based environments, showing mixed-results. Hsu et al. (2015) designed a learning environment with varying levels of written instructional detail for Grade 10 (low prior knowledge) and Grade 11 (high prior knowledge) students. Consistent with the expertise reversal effect, low prior knowledge students benefited more from detailed instructions, while high prior knowledge students performed better with less detailed guidance. Similarly, Van Riesen et al. (2018a, 2018b, 2022) showed that low prior knowledge students gained more from constrained experiment design tools (EDTs), whereas intermediate learners performed better with open-ended designs. Furthermore, Hulshof and de Jong (Hulshof & de Jong, 2006) found that students provided with “just-in-time” information tips during learning with a simulation on geometrical optics outperformed those without them. Notably, the study revealed that participants with low prior knowledge benefited most from the tips, showing substantial learning gains despite their initial lower understanding. Interestingly, these same low prior knowledge participants accessed fewer information tips than participants with higher prior knowledge. Based on these findings, Hulshof and de Jong suggested that for students starting with little background knowledge, accessing even a small number of tips might be sufficient to produce a large performance increase. These studies suggest that the expertise reversal effect translates to simulation-based inquiry learning environments.

Other studies found no moderating effect of prior knowledge within simulation-based environments. For example, Kuang et al. (2020) investigated the effects of providing support for hypothesis generation in the domain of Newton’s first law of motion. Secondary school students received either a set of terms (variables, conditions, and relations) to aid hypothesis generation or the same terms plus a partial hypothesis to build upon. Students who received the partial hypothesis (T + PHy condition) generated more complex hypotheses, collected better data, and acquired more domain knowledge. However, no significant moderating effect of prior knowledge was observed. Similarly, Liu and Chuang (2011) investigated how the format of verbal instruction in a simulation-based environment affected learning. They found no significant interactions between media format and prior knowledge, nor differences in performance between students with high and low prior knowledge. Furthermore, Roll et al. (2018) investigated the impact of directive support in an interactive physics simulation on electric circuits. They found that there were no differences between conditions with regard to prior knowledge alone. A particularly insightful study by Brenner et al. (2017) examined the effect of adaptive guidance in the “Voyage to Galapagos” simulation. The simulation-based environment provided adaptive guidance based on students’ actions. Although the study anticipated an expertise reversal effect, results were mixed: some low-knowledge students benefited from minimal assistance, while others required more support. In contrast to the studies mentioned in the previous paragraph, all these studies suggest that factors beyond prior knowledge influence the optimal level of guidance.

2.3.2 Ability Level

General effect of ability level on learning outcomes. Another personal attribute, and probably closely related to prior knowledge, would be ability level. Studies have shown significant variability in children's cognitive abilities, which in turn affects their academic achievement throughout their schooling years (Weinert & Helmke, 1998). High-ability students, in other literature sometimes referred to as gifted learners, tend to exhibit strong curiosity and eagerness to learn new knowledge (Perleth & Wilde, 2009; Subotnik et al., 2011). They prefer complex, challenging tasks and enjoy open-ended discovery (Baska & VanTassel-Baska, 2021; Watters & Diezmann, 1997), and have strong analytical skills, making them great problem-solvers (Steiner & Carr, 2003). Moreover, Alexander et al. (1995) found that high-ability students consistently demonstrate high levels of declarative metacognitive knowledge, as seen in elevated scores on interviews.

Effect of ability level on the effectiveness of guidance in simulation-based environments. To identify the conditions for successful technology-enhanced inquiry learning for young children of varying ability levels, Wang et al. (2010) emphasized the need to determine the appropriate levels of structure, scaffolding, and guidance. Van Dijk et al. (2016) examined whether prompts improved inquiry processes across ability levels. They found that high-ability children, who were more active and effective in inquiry, used the prompts when available, while average and low-ability children rarely used the prompts. Learning outcomes also differed: high- and average-ability children showed immediate knowledge gains (pretest to posttest), while low-ability children exhibited delayed gains, only showing improvement later. However, the study concluded that the presence of prompts did not significantly influence the overall learning outcomes for any ability group.

Similarly, Homer and Plass (Homer & Plass, 2014) studied high school students' learning outcomes using exploratory simulations (requiring learners to generate and test hypotheses) vs. guided simulations (loosely based on the worked example approach) of the ideal gas law. They also looked at the interaction between instructional format and executive functions, a term closely related to ability level. Executive function is generally defined as a set of high-level cognitive abilities that influence basic functions like attention and memory, while also supporting the planning, monitoring, and regulation of mental activities and behaviors (Meltzer, 2018). In the study of Homer and Plass (2014), learning outcomes were measured at two levels: comprehension of basic concepts presented in the simulation and transfer of the knowledge to similar problems. Although no interaction effects were found between executive function and learning outcomes at the comprehension level, the study did find that students with higher levels of executive functions had better transfer with the exploratory condition and students with lower levels of executive functions had better transfer with the guided simulations.

2.3.3 Self-Regulated Learning

General effect of SRL on learning outcomes. Cognitive scientists have long highlighted the importance of metacognition and SRL as important components of effective learning both within and beyond the classroom (National Research Council, 2000; Zimmerman, 2002). According to Zimmerman (1986), self-regulated learners actively engage in their own learning process through metacognitive (reflecting on their own thought processes), motivational, and behavioral strategies. Metacognitively, self-regulated learners are students who plan, organize, self-instruct, self-monitor, and self-evaluate. Motivationally, self-regulated learners see themselves as competent, self-efficacious, and autonomous. Behaviorally, self-regulated learners choose, structure, and create optimized learning environments.

The effective use of these SRL skills is strongly associated with positive learning outcomes across various domains. Underscoring this connection, Faber et al. (2023) investigated how indicators of performance and self-regulation relate to overall game performance in a medical emergency simulation. They found that scores on the SRL scale of Pintrich's Motivated Strategies for Learning Questionnaire (MSLQ) were a significant predictor of task performance. Students who scored higher on the SRL skill tended to have better learning outcomes. Whether the Pintrich’s MSLQ score would have a moderating effect on the effectiveness of guidance is still open for future research.

2.4 Present study

The present study examined the influence of adaptive guidance on the learning outcomes of grade 5 and 6 students in a simulation-based inquiry learning environment. Furthermore, it investigated the moderating effect of prior knowledge, ability level, and SRL skills. This study used an experimental design, with a pre-test and post-test, conducted using quantitative methods. Participants were randomly assigned to either a control or an experimental condition. Both groups engaged in the same inquiry learning activity within the simulation-based environment, but only the experimental group received adaptive guidance. In contrast, the control group interacted with the environment without receiving real-time feedback during the inquiry learning process.

3. Method

3.1 Participants

The research sample consisted of 72 students in grades 5 and 6 (43 boys, 29 girls; $M_{age} = 9.58$, SD = 0.71) across three different Dutch primary schools. The participants were randomly divided into two conditions: the experimental condition (25 boys, 16 girls; $M_{age} = 9.59$, SD = 0.71) and control condition (18 boys, 13 girls; $M_{age} = 9.58$, SD = 0.72). Although 81 students initially participated, some were absent on one of the test days due to illness, resulting in incomplete data. Specifically, three students were excluded for missing the pre-test, and six were excluded for missing the post-test. These exclusions were handled using listwise deletion, as comparisons between pre-test and post-test scores were not possible.

To obtain a deeper analysis of the moderating variables such as ability level, prior knowledge, and SRL skill, both the experimental group and the control group were further subdivided. For ability level, teachers needed to classify students into three categories: aiming at 25% in the low-ability group, 50% in the average-ability group, and 25% in the high-ability group. For prior knowledge, domain knowledge test scores (see Section 3.2.2) divided the students into three groups: students scoring below the 33rd percentile (low), those scoring between the 33rd and 66th percentiles (medium), and those scoring above the 66th percentile (high). Finally, for SRL skills, the students were classified into four groups based on a SRL questionnaire (see Section 3.2.3). For a detailed overview of the subgroups, see Table 1.

Table 1

Overview division participants within subgroups

Subgroup		Adaptive	Non-Adaptive	Total
Ability Level	Low	10	8	18
	Medium	14	17	31
	High	17	6	23
	Total	**41**	**31**	**72**
Resulting Label	A-U-	11	11	22
	A-U+	1	0	1
	A+U-	14	10	24
	A+U+	15	9	24
	Total	**41**	**30**	**71***
Prior Knowledge	Low	17	13	30
	Medium	16	13	29
	High	8	5	13
	Total	**41**	**31**	**72**

** One student did not complete the SRL questionnaire and was therefore excluded from that particular analysis.*

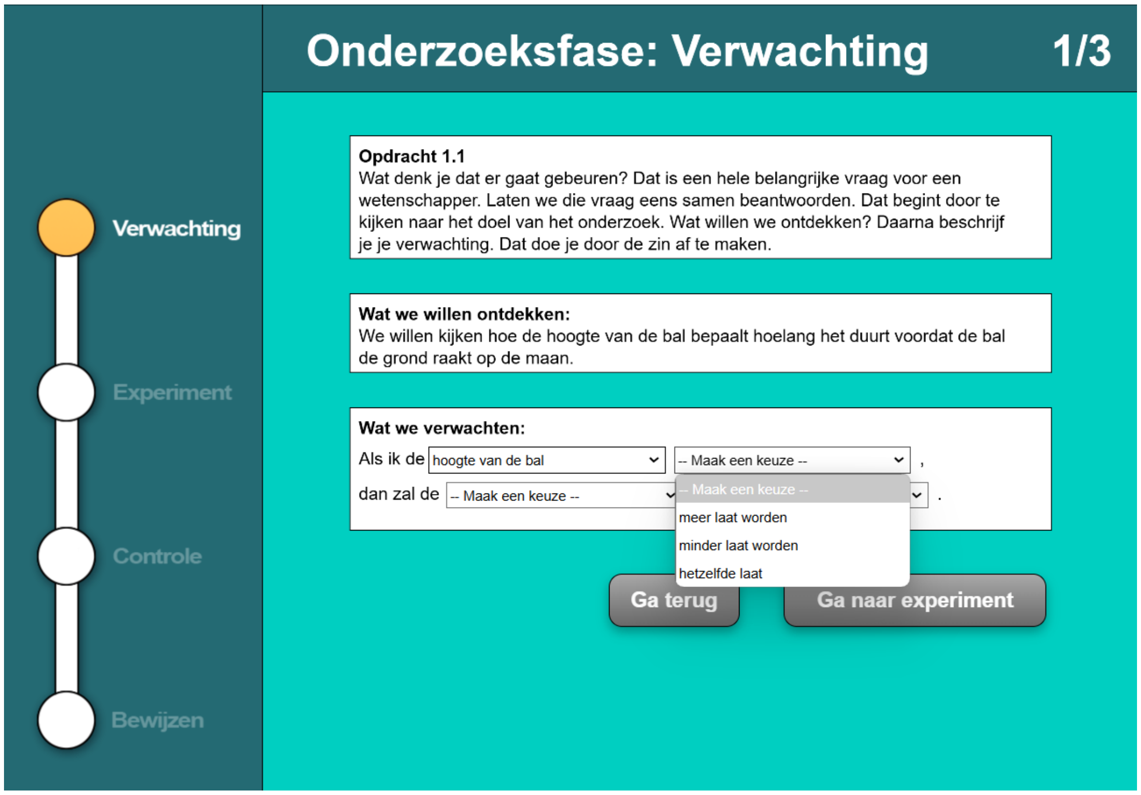
3.2 Instrumentation

3.2.1 The Simulation-based Inquiry Learning Environment

Overview. The simulation-based inquiry learning environment, which was used by both the experimental and control condition, was specifically designed and developed for this study. See Appendix A for a full overview of the code of the environment. The environment was a web-based software that lets students explore the effects of gravity on free-falling objects. It allows them to examine how variables such as a ball's mass, height, and color influence outcomes like the time it takes to reach the ground and its impact speed. The learning environment is similar to and based on the works of Jimoyiannis and Komis (2001). In addition, to avoid that students would already know too much about the topic, this phenomenon was simulated on the moon, as done in previous work like van Dijk et al. (2016).

Figure 1

Screenshot of the hypothesis phase during the first inquiry cycle



Opdracht 1.1
Wat denk je dat er gaat gebeuren? Dat is een hele belangrijke vraag voor een wetenschapper. Laten we die vraag eens samen beantwoorden. Dat begint door te kijken naar het doel van het onderzoek. Wat willen we ontdekken? Daarna beschrijf je je verwachting. Dat doe je door de zin af te maken.

Wat we willen ontdekken:
We willen kijken hoe de hoogte van de bal bepaalt hoelang het duurt voordat de bal de grond raakt op de maan.

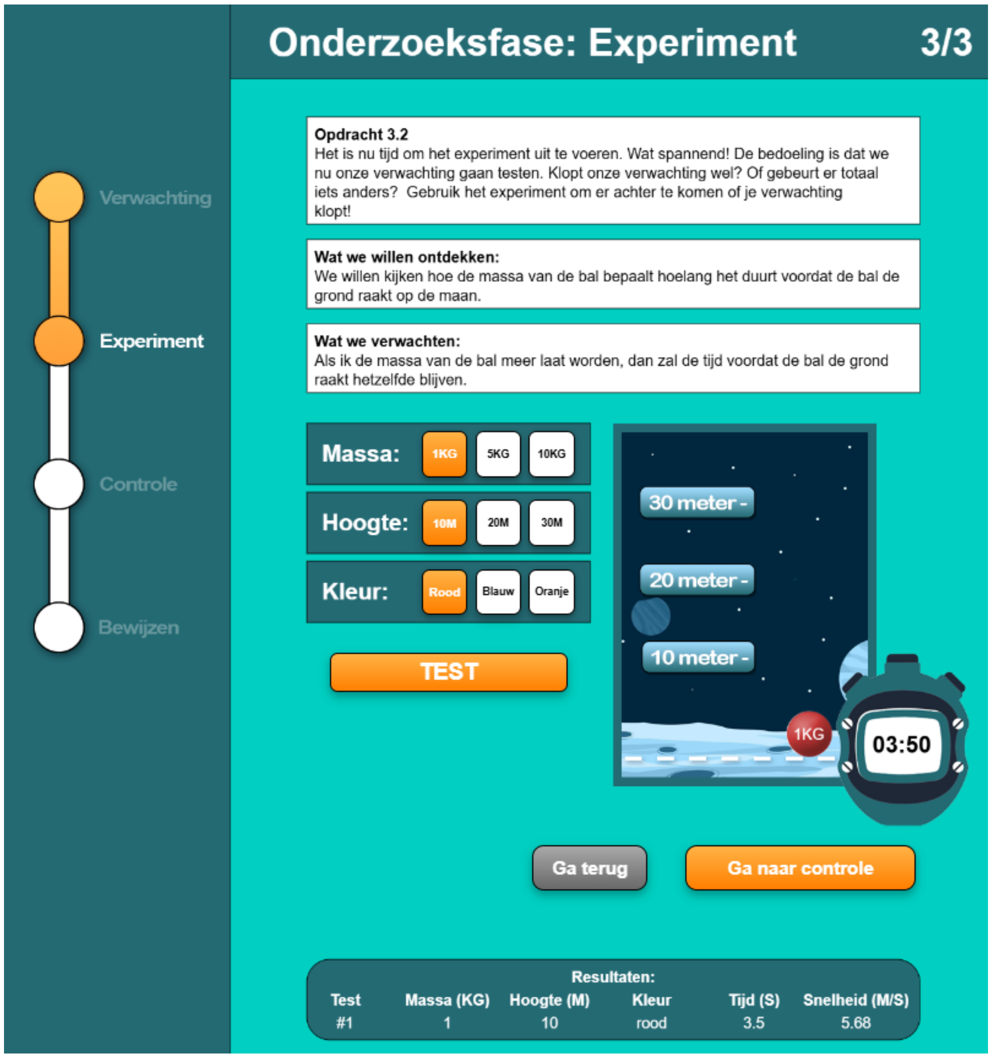
Wat we verwachten:
Als ik de hoogte van de bal -- Maak een keuze -- ,
dan zal de -- Maak een keuze -- -- Maak een keuze -- .

-- Maak een keuze --
meer laat worden
minder laat worden
hetzelfde laat

Both the experimental and the control group made use of the whole environment, where they were guided through the inquiry learning process by an online avatar, depicted as a scientist. The process was divided into four distinct phases: the hypothesis phase, the experiment phase, the analysis phase, and the proof phase. In the *hypothesis phase* (see Figure 1), students were assisted in formulating a hypothesis by constructing sentences using drop-down menus. Students select an independent variable, a dependent variable, and predict the relationship between the two. In the *experiment phase* (see Figure 2), students were able to change all variables and conduct the experiment, where they can also run the experiment as many times as preferred. The results were automatically written down below the simulation. In the *analysis phase*, students described their observations, again using drop-down menus to construct a sentence, and assessed whether their hypothesis aligned with their observations during the experiment. In the *proof phase*, students needed to select the specific results they used to support their claim of what happened.

Figure 2

Elongated screenshot of the experiment during the third inquiry cycle



Opdracht 3.2
Het is nu tijd om het experiment uit te voeren. Wat spannend! De bedoeling is dat we nu onze verwachting gaan testen. Klopt onze verwachting wel? Of gebeurt er totaal iets anders? Gebruik het experiment om er achter te komen of je verwachting klopt!

Wat we willen ontdekken:
We willen kijken hoe de massa van de bal bepaalt hoelang het duurt voordat de bal de grond raakt op de maan.

Wat we verwachten:
Als ik de massa van de bal meer laat worden, dan zal de tijd voordat de bal de grond raakt hetzelfde blijven.

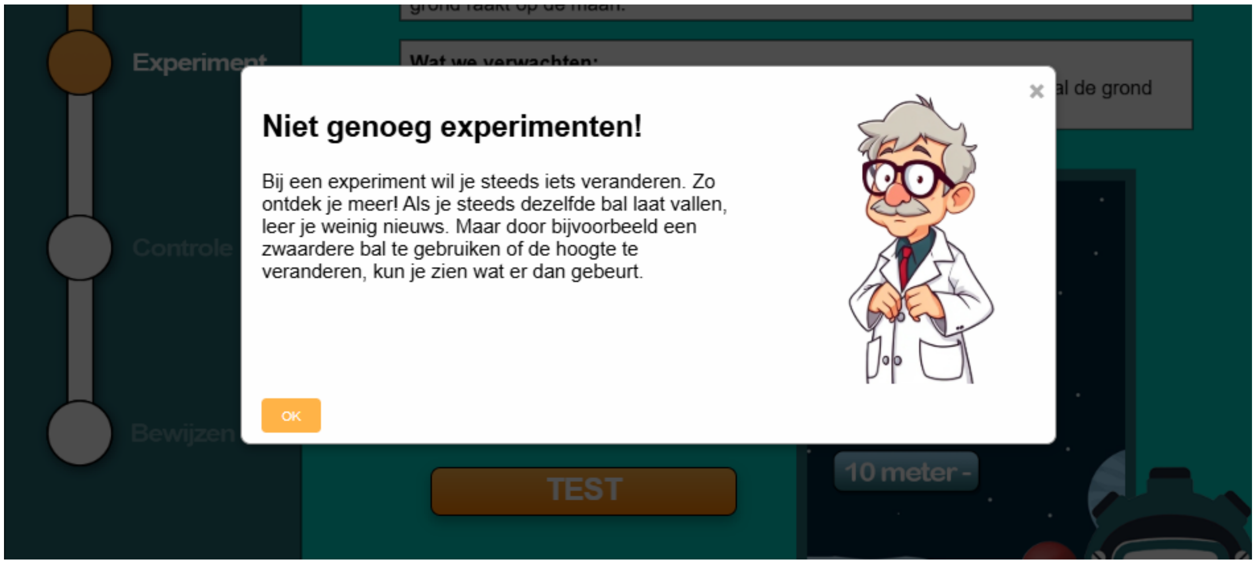
Resultaten:

Test	Massa (KG)	Hoogte (M)	Kleur	Tijd (S)	Snelheid (M/S)
#1	1	10	rood	3.5	5.68

The inquiry process was repeated three times, each focusing on the influence of another variable. This repetition ensures an extended period of engagement with the inquiry environment, which forces the students to consciously choose self-regulation skills during the inquiry learning process (Sins et al., 2024). The first cycle focused on the impact of height on the time before the ball hits the ground. The second cycle focused on the impact of height on the speed at impact as the ball hits the ground. The final cycle focused on the impact of mass on the time before the ball hits the ground.

Figure 3

Adaptive feedback within the inquiry learning environment



Niet genoeg experimenten!

Bij een experiment wil je steeds iets veranderen. Zo ontdek je meer! Als je steeds dezelfde bal laat vallen, leer je weinig nieuws. Maar door bijvoorbeeld een zwaardere bal te gebruiken of de hoogte te veranderen, kun je zien wat er dan gebeurt.

Adaptive Guidance. Throughout the inquiry learning process, the experimental group received real-time feedback (where the control did not receive any feedback) in the form of prompts, heuristics, or direct information whenever a mistake was made (see Figure 3). For example, if a student's hypothesis did not include the variable under investigation, the system would detect this and provide appropriate feedback. In total, the system could identify 14 types of errors (see Table 2 for a detailed overview of the learning environment's adaptive capabilities). Some errors triggered sequential feedback, offering increasing levels of scaffolding. Altogether, students could receive up to 26 different feedback messages (see Appendix B).

Table 2

Overview adaptivity within the system

	Provides adaptive feedback when…
Hypothesis	The student did not place an independent variable in the first drop-down menu when forming the hypothesis.
	The student did not place a dependent variable in the third drop-down menu when forming the hypothesis.
	The student did not choose the same dependent variable that was specified in the goal when forming the hypothesis
	The student did not choose the same independent variable that was specified in the goal when forming the hypothesis.
Experiment	The student only conducted a single experiment instead of multiple.
	The student did not run at least two distinct tests where the independent variable (the one under investigation) was changed between those tests.
	The student did not conduct a controlled experiment – the student did not only change the independent variable between two tests, but also changed another variable, making it uncontrolled.
Analysis	The student did not choose the dependent variable (the one specified in the goal) in the first drop-down menu when describing what happened during the experiment.
	The student did not choose the independent variable (the one specified in the goal) in the third drop-down menu when describing what happened during the experiment.
	The student, while having the correct dependent and independent variable, gives a description of the experiment that does not match the observed experiment, and is therefore factually incorrect (e.g. when I increased the height of the ball, the time before the ball hit the ground decreased).
	The student incorrectly supported or refuted their stated hypothesis (e.g. hypothesis: when I increased the height of the ball, the time before the ball hit the ground decreased; student: my hypothesis is TRUE).
Proof	The student did not select multiple results to support their claim about what happened – the student only clicked on one box.
	The student did not select two results where the value of the independent variable (the one specified in the goal) was different between them.
	The student chose two results with different values for the independent variable (the one specified in the goal), however, other independent variables also varied between those two results, which makes the pair uncontrolled.

3.2.2 Domain knowledge test

A domain knowledge test was developed to assess students' understanding of gravity on the moon. The test consisted of nine questions: three multiple-choice and six open-ended. It began with two open questions aimed at evaluating general assumptions about gravity on the moon. The remaining questions focused on three specific factors: the influence of height, mass, and color on the time it took before the ball hit the ground, and on the velocity of the ball when it hit the ground. Each factor was assessed either through a single, more complex open question (e.g. “Matthijs’s father wonders if anything would change if he replaced his ball with one of a different color but the same mass. What do you think? Would anything change if he dropped this ball? Explain your reasoning.”) or a combination of a multiple-choice question (e.g. “You and your friend Sarah live in an apartment on the moon, where you live on the second floor, and Sarah lives on the third. You both drop a ball at the same time out of the window, what do you expect to happen? Choose one of the following options.", see Appendix C for all options) followed by an open-ended question (e.g., "Explain your answer.").

To assess students' knowledge at both the pre-test and post-test stages, parallel forms of the domain knowledge test were created. The questions in the pre-test and post-test remained structurally identical, but specific variables (e.g., names in the scenarios or numerical values in the problems) were altered to ensure that the tests measured changes in students' understanding without relying on memorization. Both the pre-test and the post-test had acceptable reliabilities for the number of items within the test, reaching Cronbach's alphas of $\alpha = .68$ and $\alpha = .74$, respectively. See Appendix C for an overview of the pre- and post-test.

3.2.3 SRL Questionnaire

To measure SRL skills, an existing student questionnaire developed by Sins et al. (2024) was used (see Appendix D). This questionnaire specifically assessed the availability and use of SRL strategies. Students who struggle with *availability* (A) do not know how to plan or monitor their learning activities. Students who struggle with *use* (U) do know how to plan or monitor these strategies but lack understanding on when to use these strategies.

The questionnaire consists of 17 items divided into four themes: planning, activating prior knowledge, time monitoring, and learning monitoring. For each item, students indicate how well they know how to perform a given strategy (ranging from "not at all" to "completely") and how often they actually use it (ranging from "never" to "always"). Based on their responses, students receive an independent score (either a plus or minus) for both availability and use, categorizing them into one of three groups: A-U-, A+U-, or A+U+. Although the questionnaire allows for the possibility of an A-U+ score, it is disregarded, as a student cannot lack knowledge of planning or monitoring strategies while still knowing when to apply them.

3.3 Procedure

Participation in the study took place during school hours as an extracurricular activity and was divided into two sessions. On the first day, the students completed the pre-test to assess their domain knowledge. They were given 30 minutes to complete the test. They also filled out the SRL questionnaire, for which they had 15 minutes to complete. Two or three days later, the second session took place, which took 75 minutes to complete.

During the second session, the students individually engaged with the created learning environment. Before this session started, students received an explanation that introduced the features of the simulation. They had 45 minutes to work on the inquiry tasks. The researcher reviewed each student's responses individually before closing the simulation to ensure all data was collected. Next, the children had another 30 minutes to complete the domain knowledge post-test.

3.4 Data analysis

Both the quantitative data from domain knowledge test responses, as well as SRL test responses, were recorded in the Go-Lab platform. The exported data from the different primary schools was combined into one Excel file for data management and was anonymized. Consequently, the data was imported into the IBM SPSS software for further analysis.

3.4.1 Scoring Domain Knowledge

The domain knowledge test was scored using a coding scheme (see Appendix C). Students could achieve a maximum score of 10 points. The test began with two general questions, each worth one point. Following this, students could earn up to two points for each of the four sub-focus areas: (1) the effect of height on the time the ball takes to reach the ground, (2) the effect of height on the ball's speed upon impact, (3) the effect of mass on the time the ball takes to reach the ground, and (4) the effect of the ball's color on the time the ball takes to reach the ground.

For sub-focus areas one, two, and three, students answered a multiple-choice question predicting the outcome (worth one point) and provided an explanation for their answer in a subsequent question (worth an additional point). The fourth sub-focus area was assessed with an open-ended question, where students could earn up to two points—one for a correct prediction and one for a well-reasoned explanation.

Both the pre- and post-test were assessed by two independent coders. Answers that were deemed invalid due to reading errors were corrected within the analysis, as such errors were not the focus of this study. Instances where answers were considered reading errors were clearly defined in the coding scheme. Furthermore, the coders were not aware of the condition, ability level, or SRL skill of the students when scoring the answers. The inter-rater reliability of the questions scores ranged from .56 to 1.00 and .46 to 1.00 (Cohen's kappa) for the pre- and post-test, respectively. The lower inter-rater reliability for some questions was further analyzed and can be explained by the correction for chance agreement, as the category usage for these questions was highly skewed.

To account for potential differences in prior knowledge, as indicated by the pretest, normalized knowledge gain scores were calculated. These scores were derived by dividing each child's absolute learning gain (i.e., the post-test score minus the pre-test score) by the possibility for learning gain (i.e., the maximum possible score minus the pre-test score).

3.4.2 Analysis of Adaptive Feedback

The effect of adaptive feedback on students' absolute scores in the pretest and posttest was analyzed using a mixed-design ANOVA. The within-subjects factor was Time (pre-test vs. post-test), and the between-subjects factor was Version (Adaptive vs. Non-Adaptive). Assumptions of normality were assessed through visual inspections of Q-Q plots and boxplots, as well as skewness and kurtosis values, all of which indicated that the data approximated normality.

For normalized knowledge gain scores, where normality could not be assumed, a nonparametric test, the Mann-Whitney U test, was performed to explore the impact of Version on learning outcomes. Furthermore, subgroup analyses were performed using Mann-Whitney U tests to examine the influence of prior knowledge, ability level, and SRL skill on the effect of adaptive feedback. In these analyses, only the normalized knowledge gain scores were included.

4. Results

4.1 Effect of Adaptive Feedback

4.1.1 Absolute Test Scores

To investigate whether students learned from the learning intervention, a mixed-design ANOVA was performed. The results of the mixed design ANOVA revealed a significant main effect of Time, indicating that the absolute scores improved from the pre-test to post-test across all participants ($\mathrm{F}(1,70) = 23.79, \mathrm{p} < 0.001, \ \eta_{\mathrm{p}}^2 = 0.25$). However, the Time x Version interaction was not significant ($\mathrm{F}(1,70) = 0.25, \mathrm{p} = 0.62, \ \eta_{\mathrm{p}}^2 = 0.00$), suggesting that the pattern of improvement of the absolute scores did not differ between the Adaptive and Non-Adaptive groups. Furthermore, the main effect of Version was not significant ($\mathrm{F}(1,70) = 0.56, \mathrm{p} = 0.50, \ \eta_{\mathrm{p}}^2 = 0.01$), indicating that there was no general difference in absolute scores between the experimental and the control condition. For descriptive statistics, see Table 3.

To further explore the impact of Version on learning outcomes, a Mann-Whitney U test was performed to compare the normalized knowledge gain scores. The results did not show significant differences in scores between the two conditions (U = 579.00, Z = -0.65, p = 0.52), indicating comparable normalized knowledge gains. The results suggest that while both conditions showed improvement, the presence of feedback (adaptive vs. non-adaptive) did not differentially influence the rate of improvement from pre-test to post-test.

4.1.2 Normalized Knowledge Gain Scores

To further explore the impact of Version on learning outcomes, a Mann-Whitney U test was performed to compare the normalized knowledge gain scores. The results did not show significant differences in scores between the two conditions (U = 579.00, Z = -0.65, p = 0.52), indicating comparable normalized knowledge gains.

Table 3

Absolute test scores for pre- and post-test

	Absolute test scores				
	Pre-test		**Post-test**		
	Mean	SD	Mean	SD	n
Adaptive	3.44	0.35	5.12	0.39	41
Non-adaptive	3.24	0.40	4.61	0.45	30

4.2 Influence of Subgroups

To explore the impact of subgroups on the effectiveness on the presence of adaptive guidance on normalized knowledge gain, multiple analyses were conducted across the three moderating variables: ability level, prior knowledge, and SRL skills. For a descriptive overview of the mean and the standard deviation of the normalized knowledge gain scores for each subgroup, see Table 4. However, given the nonparametric nature of the Mann-Whitney U test, the primary focus in inferential analyses is on the median and rank-based comparisons. The following sections present the results of Mann-Whitney U tests for each subgroup, highlighting differences between adaptive and non-adaptive guidance conditions.

Table 4

Means and Standard Deviations for Normalized Knowledge Gain Scores by Prior knowledge, Ability Level, and SRL Skill

	Normalized Knowledge Gain					
	Adaptive			**Non-adaptive**		
	Mean	SD	n	Mean	SD	n
Prior knowledge						
Low	.31	.23	17	.40	.28	13
Medium	.19	.35	16	.11	.40	13
High	.28	.48	8	-.70	.83	5
Ability-level						
Low	.24	.28	10	-.26	.78	8
Medium	.29	.32	14	.17	.47	17
High	.24	.38	17	.38	.34	6
SRL skill						
A-U-	.22	.25	11	.23	.27	11
A-U+	.35	n/a	1	n/a	n/a	0
A+U-	.27	.30	14	.22	.71	10
A+U+	.26	.43	15	-.22	.65	9

4.2.1 Prior knowledge

To explore how prior knowledge influences the effectiveness of adaptive feedback, the data were analyzed based on grouping participants with varying levels of prior knowledge (low, medium, high). Three Mann-Whitney U tests were conducted to examine the effect of adaptive feedback on normalized knowledge gain scores within these groups. For students with low prior knowledge, no significant difference was found between the condition that received adaptive guidance (Mdn = .33, IQR = .44) and those that did not (Mdn = .38, IQR = .27), (U = 94.50, Z = -0.67, p = .50). For students with medium prior knowledge, results revealed no significant difference between the adaptive guidance condition (Mdn = .15, IQR = .64) and the non-adaptive guidance condition (Mdn = .11, IQR = .28), (U = 96.00, Z = -0.35, p = .72). Among participants with high prior knowledge, although this difference approached significance, results were not significantly different between the adaptive guidance condition (Mdn = .17, IQR = .84) and the non-adaptive guidance (Mdn = -1.00, IQR = 1.60), (U = 8.00, Z = -1.76, p = .08).

4.2.2 Ability-Level

To better understand how adaptive guidance might support learners with different abilities, the data were analyzed separately for participants with low, medium, and high ability levels. Three Mann-Whitney U tests were performed to examine the effect of the presence of adaptive guidance on normalized knowledge gain scores within these groups. For low-ability students, results showed no significant difference between those who received adaptive guidance (Mdn = .28, SQR = .46) and those who did not (Mdn = -.01, IQR = 1.29), (U = 26.00, Z = -1.25, p = .21). Similarly, for medium-ability students, no significant difference was found between those who received adaptive feedback (Mdn = .32, IQR = .61) and those who did not (Mdn = .17, SQR = .41), (U = 103.00, Z = -0.64, p = .52). For high-ability participants, results also showed no significant difference between the adaptive guidance condition (Mdn = .17, IQR = .57) and the non-adaptive guidance condition (Mdn = .39, SQR = .71), (U = 39.00, Z = -0.85, p = .40).

4.2.3 Self-Regulated Learning Skills

To investigate the impact of SRL skills on the effectiveness of adaptive guidance, participants were grouped based on their SRL skill levels (A–U–, A–U+, A+U–, and A+U+). Three Mann-Whitney U tests were performed to examine the effect of adaptive guidance on normalized knowledge gain scores within these groups. For the A–U– group, results did not significantly differ between participants who received adaptive feedback (Mdn = .20, IQR = .39) and those who received non-adaptive feedback (Mdn = .14, IQR = .36), (U = 59.00, Z = -.10, p =

.92). For the A+U- group, no significant difference was found between the adaptive guidance condition (Mdn = .25, IQR = .51) and the non-adaptive guidance condition (Mdn = .34, IQR = .73), (U = 62.50, Z = -.45, p = .66). Finally, for the A+U+ group, the adaptive guidance condition (Mdn = .30, IQR = .79) and the non-adaptive guidance condition (Mdn = .00, IQR = 1.15), did not significantly differ (U = 40.00, Z = -1.65, p = .10). These results indicate that self-regulated learning skills did not significantly influence normalized knowledge gain scores between the two conditions.

5. Discussion

This study examined the effects of adaptive guidance, in a simulation-based learning environment, compared to the same environment without this real-time adaptive guidance, on students' learning outcomes. It particularly focussed on the moderating effect of students' prior knowledge, ability levels, and SRL skills on the effectiveness of adaptive guidance. Prior to this study, further research was needed to compare the impact of adaptive guidance to the same simulation without it. Additionally, existing studies had not yet thoroughly examined which moderating variables influence its effectiveness.

Our results indicate that adaptive guidance, as implemented in this study, was not a decisive factor in improving learning outcomes. Both the experimental and the control condition showed significant normalized knowledge gains, but no significant difference was found between the two conditions. Additionally, no significant moderating effects were observed for prior knowledge, ability level, or SRL skills.

5.1 Effect of adaptive guidance on learning outcomes

Whether adaptive guidance positively influences learning outcomes within a simulation-based inquiry learning environment is still open for discussion. Our findings are consistent with several studies that investigated the effects of adaptive guidance on student performance, which also did not find an interaction effect between condition (with adaptive guidance or without) and student performance (Brenner et al., 2017; Duffy & Azevedo, 2015; Faber et al., 2024).

One possible explanation is that other personal attributes (than the ones discussed below), influenced the effectiveness of the adaptive guidance within a simulation-based approach. Multiple other attributes have been shown to moderate the effectiveness of guidance or learning outcomes in general. For example, Nelson & Ketelhut (2008) found that students with higher *self-efficacy* engaged more with the embedded guidance, leading to improved learning outcomes. Other potential moderating attributes could be *digital competence* (Q. Wang et al., 2024) and/or *motivation* (Hanano et al., 2024). However, whether these personal attributes have a moderating effect on the effectiveness of guidance, particularly adaptive guidance, remains an area for future research.

Another possible explanation is that adaptive guidance may be a victim of the same problem it aims to solve, and accidentally increased cognitive load. Cognitive load is the amount of information our working memory can process at any given time (Pengelley et al., 2024). When cognitive load is high, the brain struggles to effectively process feedback, limiting its ability to learn from feedback and adjust behavior accordingly (Krigolson et al., 2015). Furthermore, within the design of this study, the direct adaptive guidance encourages the learner to pursue sub-goals that were defined by the activity design (e.g., a hypothesis should contain an independent variable), which may not align with the objectives they need to understand first to do the inquiry learning task. Research on goal specificity suggests that allowing students to set their own goals is more effective than externally assigning them (Miller, 1999).

As a result, requiring students to focus on a sub-goal and preventing them from progressing until they correct an error may have further increased cognitive load. This could also explain why post-test scores, though significantly higher than pre-test scores, remain relatively low for the adaptive guidance condition. However, this remains an assumption. Future research could explore the effectiveness of different guidance approaches, such as adaptive guidance that persists until an error is corrected, a system that allows students to proceed with errors after guidance is given, or one that automatically corrects mistakes, to determine their impact on cognitive load and learning outcomes.

5.2 Moderating effect of prior knowledge on the effectiveness of adaptive guidance on learning outcomes

Contrary to some expectations, our study found prior knowledge was not predictive of students' need for adaptive guidance. This mirrors findings by Brenner et al. (2017) on adaptive guidance, and it aligns with some studies on more general types of guidance in simulation-based environments (Kuang et al., 2020; Liu & Chuang, 2011). However, it conflicts with other work where prior knowledge has been shown to have a moderating effect on the effectiveness of guidance (e.g., Hsu et al., 2015; Van Riesen et al., 2018a).

We argue that the way the student engages with learning materials, in combination with prior knowledge, may be a more crucial factor than prior knowledge alone. For instance, Brenner et al. (2017) observed that active, though initially unproductive, novice learners (students with low levels of prior knowledge) required different guidance levels than less active ones. This activity was measured with log-file analysis, a method which was outside the scope of this study. This difference between novices would explain why in some studies it is particularly demanding for novice learners to make sense of the given guidance (Koedinger & Roll, 2012), while other research shows that guidance is more effective for novices and may be redundant or even hinder experts (Kalyuga, 2007). It would also support the idea that fostering meaning-making through interaction, even if initially incorrect, is key. Concepts like Productive Failure (Loibl et al., 2016) and the benefits of learning from mistakes (Mathan & Koedinger, 2005) further highlight this need for allowing mistakes. Future research could therefore benefit from analyzing student activity patterns (e.g., via log-files) alongside prior knowledge to potentially develop more effective guidance by clustering students based on these combined factors.

5.3 Moderating effect of ability level on the effectiveness of adaptive guidance on learning outcomes

Our finding that the effectiveness of guidance was not significantly moderated by ability level aligns with the results of van Dijk et al. (2016). They also reported no significant interaction between their guidance condition and students' ability levels (low, medium, or high), despite employing a different type of guidance, where students were required to actively access prompts, unlike the adaptive system used in our study. The similarity in findings, despite these methodological differences (adaptive vs. self-accessed guidance), suggests that ability level *alone* may not be a decisive factor in determining the effectiveness of guidance on student performance within simulation-based learning environments. Future research could investigate combinations of ability level with other personal attributes to find a possible predictor of effectiveness of guidance.

5.4 Moderating effect of SRL skills on the effectiveness of adaptive guidance on learning outcomes

Our results indicate that SRL level did not significantly moderate the guidance's effectiveness within this simulation-based environment. The adaptive guidance was not found to be significantly more or less beneficial for students depending on their SRL capabilities, measured by the SRL questionnaire. Comparing this specific finding to previous work is difficult, as research regarding SRL and guidance has often focused on related but distinct questions. For instance, prior studies have concentrated on measuring SRL processes as they occur within the simulation-based environment (Sabourin et al., 2013) or identifying specific SRL-related interactions themselves (Duffy & Azevedo, 2015; Li et al., 2023; Munshi et al., 2023), rather than explicitly testing baseline SRL skills as a variable that impacts the overall effectiveness of guidance.

5.5 Limitations

While this study contributes to the understanding of adaptive guidance in simulation-based environments, several limitations should be acknowledged. The most significant limitation comes from the relatively small sample size. This low statistical power, particularly for comparisons within subgroups (low, medium, high ability), could have obscured subtle interaction effects. Furthermore, the small sample size combined with the use of multiple Mann-Whitney U tests without correction (e.g., Bonferroni) introduced a risk of Type I error. Addressing this limitation requires future studies to use a larger sample size. This would not only improve the statistical power for examining the moderating effect of the personal attributes examined in this study, but would also enable more advanced analyses like two-step clustering to identify clusters based on combined attributes that might better predict the need for guidance.

Furthermore, as this study only focused on learning outcomes, it was limited by the lack of data on how students react to the guidance and interact with the learning environment. Future research could gain significant insights by using log-file analysis, a method where students' actions within a simulation-based environment are saved and further analyzed. This would provide qualititive data on student inquiry activities; shedding light on how different personal attributes influence within-environment behaviours and affect learning outcomes.

Finally, advancements can be made in the adaptive guidance itself. Future iterations could move towards systems using continuous data collection on student performance, predictive modeling of students' struggles, and adaptive guidance based on probability thresholds. Methods such as Bayesian Networks (Brenner et al., 2017) offer innovative real-time assessment and could be areas for future investigation.

5.6 Conclusion

This research investigated the effectiveness of adaptive guidance in a simulation-based environment, focusing on how prior knowledge, ability level, and SRL skills might affect it. While the adaptive guidance did not significantly improve learning outcomes compared to the control group, and the examined personal attributes did not reveal a significant moderating effect, the study still provides valuable insights. It highlights the difficulty of designing effective adaptive guidance for inquiry learning and suggests that other factors might be more important. It also paves the way for further investigation using methods like log-file analysis and more advanced adaptive systems, hopefully leading to a better understanding of personalized adaptive guidance within simulation-based environments, resulting in better learning outcomes among students.

References

Ainsworth, S., & VanLabeke, N. (2004). Multiple forms of dynamic representation. *Learning and Instruction*, *14*(3), 241–255. https://doi.org/10.1016/J.LEARNINSTRUC.2004.06.002

Albanese, M. A., & Mitchell, S. (1993). Problem-based learning: a review of literature on its outcomes and implementation issues. *Academic Medicine, 68*(1), 52–81. https://doi.org/10.1097/00001888-199301000-00012

Alexander, J. M., Carr, M., & Schwanenflugel, P. J. (1995). Development of metacognition in gifted children: Directions for future research. *Developmental Review*, *15*(1), 1–37. https://doi.org/10.1006/DREV.1995.1001

Alexander, P. A., & Judy, J. E. (1988). The Interaction of Domain-Specific and Strategic Knowledge in Academic Performance. *Review of Educational Research*, *58*(4), 375–404. https://doi.org/10.3102/00346543058004375

Alfieri, L., Brooks, P. J., Aldrich, N. J., & Tenenbaum, H. R. (2011). Does Discovery-Based Instruction Enhance Learning? *Journal of Educational Psychology*, *103*(1), 1–18. https://doi.org/10.1037/A0021017

Baska, A., & VanTassel-Baska, J. (2021). *Curriculum Planning and Instructional Design*. Routledge. https://www.routledge.com/Curriculum-Planning-and-Instructional-Design-for-Gifted-Learners/VanTassel-Baska-Baska/p/book/9781618218896

Blake, C., & Scanlon, E. (2007). Reconsidering simulations in science education at a distance: features of effective use. *Journal of Computer Assisted Learning*, *23*(6), 491–502. https://doi.org/10.1111/J.1365-2729.2007.00239.X

Blayney, P., Kalyuga, S., & Sweller, J. (2010). Interactions between the isolated-interactive elements effect and levels of learner expertise: Experimental evidence from an accountancy class. *Instructional Science*, *38*(3), 277–287. https://doi.org/10.1007/S11251-009-9105-X

Bloom, B. S. (1984). The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring. *Educational Researcher*, *13*(6), 4–16. https://doi.org/10.3102/0013189X013006004

Brenner, D. G., Matlen, B. J., Timms, M. J., Gochyyev, P., Grillo-Hill, A., Luttgen, K., & Varfolomeeva, M. (2017). Modeling Student Learning Behavior Patterns in an Online Science Inquiry Environment. *Technology, Knowledge and Learning*, *22*(3), 405–425. https://doi.org/10.1007/S10758-017-9325-0

Charney, D., Reder, L., & Kusbit, G. W. (1990). Goal Setting and Procedure Selection in Acquiring Computer Skills: A Comparison of Tutorials, Problem Solving, and Learner Exploration. *Cognition and Instruction*, *7*(4), 323–342. https://doi.org/10.1207/S1532690XCI0704_3

Chinn, C. A., & Brewer, W. F. (1993). The role of anomalous data in knowledge acquisition: A theoretical framework and implications for science instruction. *Review of Educational Research*, *63*(1), 1–49. https://doi.org/10.3102/00346543063001001

Clark, R. E. (1990). When teaching kills learning: Studies of mathematantics. In H. Mandl, E. De Corte, N. Bennett, & H. F. Friedrich (Eds.), *Learning and instruction: European research in an international context: Vol. 2.2*. Pergamon. https://www.researchgate.net/publication/313407358_When_teaching_kills_learning_Studies_of_mathematantics de Jong, T. (2006). Technological advances in inquiry learning. *Science*, *312*(5773), 532–533. https://doi.org/10.1126/SCIENCE.1127750

de Jong, T., Beishuizen, J. J., Hulshof, C. D., Prins, F., Rijn, H. van, van Someren, M., Veenman, M., & Wilhelm, P. (2014). Determinants of discovery learning in a complex simulation learning environment. In P. Gandenfors & P. Johansson (Eds.), *Cognition, education, and communication technology* (pp. 257–283). Routledge. https://doi.org/10.4324/9781410612892

Duffy, M. C., & Azevedo, R. (2015). Motivation matters: Interactions between achievement goals and agent scaffolding for self-regulated learning within an intelligent tutoring system. *Computers in Human Behavior*, *52*, 338–348. https://doi.org/10.1016/J.CHB.2015.05.041

Dunbar, K., Petitto, L. A., Klahr, D., Simon, H. A., & Klayman, J. (1993). Concept Discovery in a Scientific Domain. *Cognitive Science*, *17*(3), 397–434. https://doi.org/10.1207/S15516709COG1703_3

Faber, T. J. E., Dankbaar, M. E. W., Kickert, R., van den Broek, W. W., & van Merriënboer, J. J. G. (2023). Identifying indicators to guide adaptive scaffolding in games. *Learning and Instruction*, *83*, 101666. https://doi.org/10.1016/J.LEARNINSTRUC.2022.101666

Faber, T. J. E., van den Broek, M. E. W., Bruinink, W. W., Hogeveen, L. J., & van Merriënboer, M. (2024). Effects of adaptive scaffolding on performance, cognitive load and engagement in game-based learning: a randomized controlled trial. *BMC Medical Education*, *24*(1), 934. https://doi.org/10.1186/s12909-024-05698-3

Glaser, R., Schauble, L., Raghavan, K., & Zeitz, C. (1992). Scientific reasoning across different domains. In E. De Corte, M. C. Linn, H. Mandl, & L. Verschaffel (Eds.), *Computer-Based*

Learning Environments and Problem Solving (Vol. 84). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-77228-3_16

Graesser, A., & McNamara, D. (2010). Self-regulated learning in learning environments with pedagogical agents that interact in natural language. *Educational Psychologist*, *45*(4), 234–244. https://doi.org/10.1080/00461520.2010.515933

Hanano, M., Rith-Najarian, L., Gong-Guy, E., & Chavira, D. (2024). Motivational variables as moderating effects of a web-based mental health program for university students: Secondary analysis of a randomized controlled trial. *JMIR Formative Research*, *8*(1), e56118. https://doi.org/10.2196/56118

Hattie, J. A. C., & Donoghue, G. M. (2016). Learning strategies: A synthesis and conceptual model. *Npj Science of Learning*, *1*, 16013. https://doi.org/10.1038/npjscilearn.2016.13

Hmelo-Silver, C. E., Duncan, R. G., & Chinn, C. A. (2007). Scaffolding and achievement in problem-based and inquiry learning: A response to Kirschner, Sweller, and Clark (2006). *Educational Psychologist*, *42*(2), 99–107. https://doi.org/10.1080/00461520701263368

Homer, B. D., & Plass, J. L. (2014). Level of interactivity and executive functions as predictors of learning in computer-based chemistry simulations. *Computers in Human Behavior*, *36*, 365–375. https://doi.org/10.1016/J.CHB.2014.03.041

Hsu, Y.-S., Lai, T.-L., & Hsu, W.-H. (2015). A design model of distributed scaffolding for inquiry-based learning. *Research in Science Education*, *45*(2), 241–273. https://doi.org/10.1007/s11165-014-9421-2

Hulshof, C. D., & de Jong, T. (2006). Using just-in-time information to support discovery learning about geometrical optics in a computer-based simulation. *Interactive Learning Environments*, *14*(1), 79–94. https://doi.org/10.1080/10494820600769171

Jimoyiannis, A., & Komis, V. (2001). Computer simulations in physics teaching and learning: A case study on students' understanding of trajectory motion. *Computers & Education*, *36*(2), 183–204. https://doi.org/10.1016/S0360-1315(00)00059-2

Jong, T. de, & Lazonder, A. W. (2014). The guided discovery principle in multimedia learning. In R. E. Mayer (Ed.), *he Cambridge handbook of multimedia learning 2nd edition* (pp. 371–390). Cambridge University Press. https://research.utwente.nl/en/publications/the-guided-discovery-principle-in-multimedia-learning-2

Kalyuga, S. (2007). Expertise reversal effect and its implications for learner-tailored instruction. *Educational Psychology Review*, *19*(4), 509–539. https://doi.org/10.1007/S10648-007-9054-3

Kaplan, D. E., & Black, J. B. (2003). Mental models and computer-based scientific inquiry learning: Effects of mechanistic cues on adolescent representation and reasoning about causal systems. *Journal of Science Education and Technology*, *12*(4), 483–493. https://doi.org/10.1023/B:JOST.0000006308.01183.85

Kirschner, P. A., Sweller, J., & Clark, R. E. (2006). Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational Psychologist*, *41*(2), 75–86. https://doi.org/10.1207/S15326985EP4102_1

Klahr, D. (2013). What do we mean? On the importance of not abandoning scientific rigor when talking about science education. *Proceedings of the National Academy of Sciences of the United States of America*, *110*(Suppl. 3), 14075–14080. https://doi.org/10.1073/PNAS.1212738110

Klahr, D., & Dunbar, K. (1988). Dual space search during scientific reasoning. *Cognitive Science*, *12*(1), 1–48. https://doi.org/10.1207/S15516709COG1201_1

Klahr, D., & Nigam, M. (2004). The equivalence of learning paths in early science instruction: Effects of direct instruction and discovery learning. *Psychological Science*, *15*(10), 661–667. https://doi.org/10.1111/J.0956-7976.2004.00737.X

Klayman, J., & Ha, Y. W. (1987). Confirmation, disconfirmation, and information in hypothesis testing. *Psychological Review*, *94*(2), 211–228. https://doi.org/10.1037/0033-295X.94.2.211

Koedinger, K. R., & Roll, I. (2012). Learning to think: Cognitive mechanisms of knowledge Transfer. In K. J. Holyoak & R. G. Morrison (Eds.), *The Oxford handbook of thinking and reasoning*. Oxford Library of Psychology. https://doi.org/10.1093/OXFORDHB/9780199734689.013.0040

Krigolson, O. E., Hassall, C. D., Satel, J., & Klein, R. M. (2015). The impact of cognitive load on reward evaluation. *Brain Research*, *1627*, 225–232. https://doi.org/10.1016/J.BRAINRES.2015.09.028

Kuang, X., Eysink, T. H. S., & de Jong, T. (2020). Effects of providing partial hypotheses as a support for simulation-based inquiry learning. *Journal of Computer Assisted Learning*, *36*(4), 487–501. https://doi.org/10.1111/JCAL.12415

Kuhn, D., Iordanou, K., Pease, M., & Wirkala, C. (2008). Beyond control of variables: What needs to develop to achieve skilled scientific thinking? *Cognitive Development*, *23*(4), 435–451. https://doi.org/10.1016/J.COGDEV.2008.09.006

Kuhn, D., Schauble, L., & Garcia-Mila, M. (1992). Cross-domain development of scientific reasoning. *Cognition and Instruction*, *9*(4), 285–327. https://doi.org/10.1207/S1532690XCI0904_1

Lai, C. L., Hwang, G. J., & Tu, Y. H. (2018). The effects of computer-supported self-regulation in science inquiry on learning outcomes, learning processes, and self-efficacy. *Educational Technology Research and Development*, *66*(4), 863–892. https://doi.org/10.1007/S11423-018-9585-Y

Lambiotte, J. G., & Dansereau, D. F. (1992). Effects of knowledge maps and prior knowledge on recall of science lecture content. *Journal of Experimental Education*, *60*(3), 189–201. https://doi.org/10.1080/00220973.1992.9943875

Lazonder, A. W. (2014). Inquiry learning. In J. M. Spector, M. D. Merill, J. Elen, & J. Bishop (Eds.), *Handbook of research on educational communications and technology* (4th ed., pp. 453–464). Springer. https://doi.org/10.1007/978-1-4614-3185-5_36

Lazonder, A. W., & Harmsen, R. (2016). Meta-analysis of inquiry-based learning: Effects of guidance. *Review of Educational Research*, *86*(3), 681–718. https://doi.org/10.3102/0034654315627366

Lazonder, A. W., & Kamp, E. (2012). Bit by bit or all at once? Splitting up the inquiry task to promote children's scientific reasoning. *Learning and Instruction*, *22*(6), 458–464. https://doi.org/10.1016/J.LEARNINSTRUC.2012.05.005

Li, T., Fan, Y., Tan, Y., Wang, Y., Singh, S., Li, X., Raković, M., van der Graaf, J., Lim, L., Yang, B., Molenaar, I., Bannert, M., Moore, J., Swiecki, Z., Tsai, Y. S., Shaffer, D. W., & Gašević, D. (2023). Analytics of self-regulated learning scaffolding: Effects on learning processes. *Frontiers in Psychology*, *14*, 1206696. https://doi.org/10.3389/FPSYG.2023.1206696

Lim, B. R. (2004). Challenges and issues in designing inquiry on the Web. *British Journal of Educational Technology*, *35*(5), 627–643. https://doi.org/10.1111/J.0007-1013.2004.00419.X

Liu, H. C., & Chuang, H. H. (2011). Investigation of the impact of two verbal instruction formats and prior knowledge on student learning in a simulation-based learning environment. *Interactive Learning Environments*, *19*(4), 433–446. https://doi.org/10.1080/10494820903356940

Long, Y., & Aleven, V. (2013). Supporting students' self-regulated learning with an open learner model in a linear equation tutor. In H. C. Lane, K. Yacef, J. Mostow, & P. Pavlik (Eds.), *Proceedings of the 16th International Conference on Artificial Intelligence in Education, AIED 2013* (pp. 219–228). New York: Springer. https://doi.org/10.1007/978-3-642-39112-5_23

Mayer, R. E. (2004). Should there be a three-strikes rule against pure discovery learning? The case for guided methods of instruction. *American Psychologist*, *59*(1), 14–19. https://doi.org/10.1037/0003-066X.59.1.14

McNeill, K. L., & Krajcik, J. (2007). Middle school students' use of appropriate and inappropriate evidence in writing scientific explanations. In M. Lovett & P. Shah (Eds.), *Thinking with data: The proceedings of the 33rd carnegie symposium on cognition* (pp. 233–265). Lawrence Erlbaum Associates, Inc.

Meltzer, L. (2018). *Executive function in education: From theory to practice* (2nd ed.). The Guilford Press.

Miller, D. T. (1999). The norm of self-interest. *American Psychologist*, *54*(12), 1053–1060. https://doi.org/10.1037/0003-066X.54.12.1053

Minner, D. D., Levy, A. J., & Century, J. (2009). Inquiry-based science instruction—what is it and does it matter? Results from a research synthesis years 1984 to 2002. *Journal of Research in Science Teaching*, *47*(4), 474–496. https://doi.org/10.1002/TEA.20347

Munshi, A., Biswas, G., Baker, R., Ocumpaugh, J., Hutt, S., & Paquette, L. (2023). Analysing adaptive scaffolds that help students develop self-regulated learning behaviours. *Journal of Computer Assisted Learning*, *39*(2), 351–368. https://doi.org/10.1111/JCAL.12761

National Research Council. (1996). *National science education standards*. National Academies Press. https://doi.org/10.17226/4962

National Research Council. (2000). *How people learn: Brain, mind, experience, and school (Expanded Edition)*. National Academies Press. https://doi.org/10.17226/9853

National Research Council. (2011). *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas*. The National Academies Press. https://doi.org/10.17226/13165

Nelson, B. C., & Ketelhut, D. J. (2008). Exploring embedded guidance and self-efficacy in educational multi-user virtual environments. *International Journal of Computer-Supported Collaborative Learning, 3*(4), 413–427. https://doi.org/10.1007/S11412-008-9049-1

Olympiou, G., Zacharias, Z., & de Jong, T. (2013). Making the invisible visible: Enhancing students' conceptual understanding by introducing representations of abstract objects in a simulation. *Instructional Science*, *41*(3), 575–596. https://doi.org/10.1007/S11251-012-9245-2

Pedaste, M., Mäeots, M., Siiman, L. A., de Jong, T., van Riesen, S. A. N., Kamp, E. T., Manoli, C. C., Zacharia, Z. C., & Tsourlidaki, E. (2015). Phases of inquiry-based learning: Definitions and the inquiry cycle. *Educational Research Review*, *14*, 47–61. https://doi.org/10.1016/J.EDUREV.2015.02.003

Pengelley, J., Whipp, P. R., & Malpique, A. (2024). A testing load: A review of cognitive load in computer and paper-based learning and assessment. *Technology, Pedagogy and Education*, *34*(1), 1–17. https://doi.org/10.1080/1475939X.2024.2367517

Perleth, C., & Wilde, A. (2009). Developmental trajectories of giftedness in children. In L. V. Shavinina (Ed.), *International handbook on giftedness* (pp. 319–335). Springer, Dordrecht. https://doi.org/10.1007/978-1-4020-6162-2_14

Poitras, E. G., & Lajoie, S. P. (2014). Developing an agent-based adaptive system for scaffolding self-regulated inquiry learning in history education. *Educational Technology Research and Development*, *62*(3), 335–366. https://doi.org/10.1007/S11423-014-9338-5

Quinn, J., & Alessi, S. (1994). The effects of simulation complexity and hypothesis-generation strategy on learning. *Journal of Research on Computing in Education*, *27*(1), 75–91. https://doi.org/10.1080/08886504.1994.10782117

Quintana, C., Reiser, B. J., Davis, E. A., Krajcik, J., Fretz, E., Duncan, R. G., Kyza, E., Edelson, D., & Soloway, E. (2004). A scaffolding design framework for software to support science inquiry. *The Journal of the Learning Sciences*, *13*(3), 337–386. https://doi.org/10.1207/S15327809JLS1303_4

Rey, G. D. (2011). Reset button and instructional advice in computer simulations. *European Psychologist*, *16*(1), 58–67. https://doi.org/10.1027/1016-9040/A000042

Richardson, J. (2008). *Science ASSISTments: Tutoring inquiry skills in middle school students* [Unpublished Interactive Qualifying Project, Worcester Polytechnic Institute]. https://digital.wpi.edu/concern/student_works/8g84mm968?locale=pt-BR

Rieber, L. P., Tzeng, S. C., & Tribble, K. (2004). Discovery learning, representation, and explanation within a computer-based simulation: Finding the right mix. *Learning and Instruction*, *14*(3), 307–323. https://doi.org/10.1016/J.LEARNINSTRUC.2004.06.008

Roll, I., Butler, D., Yee, N., Welsh, A., Perez, S., Briseno, A., Perkins, K., & Bonn, D. (2018). Understanding the impact of guiding inquiry: The relationship between directive support, student attributes, and transfer of knowledge, attitudes, and behaviours in inquiry learning. *Instructional Science*, *46*(1), 77–104. https://doi.org/10.1007/S11251-017-9437-X

Rutten, N., van Joolingen, W. R., & van Der Veen, J. T. (2012). The learning effects of computer simulations in science education. *Computers and Education*, *58*(1), 136–153. https://doi.org/10.1016/J.COMPEDU.2011.07.017

Sabourin, J., Mott, B., & Lester, J. (2013). Utilizing dynamic bayes nets to improve early prediction models of self-regulated learning. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 7899 LNCS*, 228–241. https://doi.org/10.1007/978-3-642-38844-6_19

Sadler, T. D. (2004). Informal reasoning regarding socioscientific issues: A critical review of research. *Journal of Research in Science Teaching*, *41*(5), 513–536. https://doi.org/10.1002/TEA.20009

Schauble, L., Glaser, R., Duschl, R. A., Schulze, S., & John, J. (1995). Students' understanding of the objectives and procedures of experimentation in the science classroom. *Journal of the Learning Sciences*, *4*(2), 131–166. https://doi.org/10.1207/S15327809JLS0402_1

Schauble, L., Klopfer, L. E., & Raghavan, K. (1991). Students' transition from an engineering model to a science model of experimentation. *Journal of Research in Science Teaching*, *28*(9), 859–882. https://doi.org/10.1002/TEA.3660280910

Schunn, C., & Anderson, J. R. (1999). The generality/specificity of expertise in scientific reasoning. *Cognitive Science*, *23*(3), 337–370. https://doi.org/10.1016/S0364-0213(99)00006-3

Seufert, T. (2003). Supporting coherence formation in learning from multiple representations. *Learning and Instruction*, *13*(2), 227–237. https://doi.org/10.1016/S0959-4752(02)00022-1

Sins, P., de Brouwer, J., van Dijk, A., Eysink, T., Morssink-Santing, V., & Klaver, L. (2024). *Zelfregulerend leren in het W&T-basisonderwijs Praktijkboek voor leraren*. TechYourFuture. https://www.techyourfuture.nl/wp-content/uploads/2024/06/401-0091-Publicatie-over-zelfregulerend-leren-DIGITAAL.pdf

Smetana, L. K., & Bell, R. L. (2012). Computer simulations to support science instruction and learning: A critical review of the literature. *International Journal of Science Education*, *34*(9), 1337–1370. https://doi.org/10.1080/09500693.2011.605182

Steiner, H. H., & Carr, M. (2003). Cognitive development in gifted children: Toward a More Precise Understanding of Emerging Differences in Intelligence. *Educational Psychology Review*, *15*(3), 215–246. https://doi.org/10.1023/A:1024636317011

Strike, K. A. (1975). The logic of learning by discovery. *Review of Educational Research*, *45*(3), 461–483. https://doi.org/10.3102/00346543045003461

Subotnik, R. F., Olszewski-Kubilius, P., & Worrell, F. C. (2011). Rethinking giftedness and gifted education: A proposed direction forward based on psychological science. *Psychological Science in the Public Interest, Supplement*, *12*(1), 3–54. https://doi.org/10.1177/1529100611418056

Sun, Y., Yan, Z., & Wu, B. (2022). How differently designed guidance influences simulation-based inquiry learning in science education: A systematic review. *Journal of Computer Assisted Learning*, *38*(4), 960–976. https://doi.org/10.1111/jcal.12667

Timms, M., Clements, D. H., Gobert, J., Ketelhut, D. J., Lester, J., Reese, D. D., & Wiebe, E. (2012). *New measurement paradigms*. https://research.acer.edu.au/ar_misc/9

Tuovinen, J. E., & Sweller, J. (1999). A comparison of cognitive load associated with discovery learning and worked examples. *Journal of Educational Psychology*, *91*(2), 334–341. https://doi.org/10.1037/0022-0663.91.2.334

van Dijk, A. M., Eysink, T. H. S., & de Jong, T. (2016). Ability-related differences in performance of an inquiry task: The added value of prompts. *Learning and Individual Differences*, *47*, 145–155. https://doi.org/10.1016/J.LINDIF.2016.01.008

Van Dijk, A. M., Eysink, T. H. S., & De Jong, T. (2016). Ability-related differences in performance of an inquiry task: The added value of prompts. *Learning and Individual Differences*, *47*, 145–155. https://doi.org/10.1016/J.LINDIF.2016.01.008

van Dijk, A. M., & Lazonder, A. W. (2016). Scaffolding students' use of learner-generated content in a technology-enhanced inquiry learning environment. *Interactive Learning Environments*, *24*(1), 194–204. https://doi.org/10.1080/10494820.2013.834828

van Joolingen, W. R., & de Jong, A. J. M. (1991). Modelling domain knowledge for intelligent simulation learning environments. *Computers and Education*, *18*, 29–37. https://doi.org/10.1016/0360-1315(92)90033-2

van Joolingen, W. R., & de Jong, T. (1997). An extended dual search space model of scientific discovery learning. *Instructional Science*, *25*(5), 307–346. https://doi.org/10.1023/A:1002993406499/METRICS

van Joolingen, W. R., de Jong, T., & Dimitrakopoulou, A. (2007). Issues in computer supported inquiry learning in science. *Journal of Computer Assisted Learning*, *23*(2), 111–119. https://doi.org/10.1111/J.1365-2729.2006.00216.X

van Riesen, S. A. N., Gijlers, H., Anjewierden, A. A., & de Jong, T. (2022). The influence of prior knowledge on the effectiveness of guided experiment design. *Interactive Learning Environments*, *30*(1), 17–33. https://doi.org/10.1080/10494820.2019.1631193 van Riesen, S., Gijlers, H., Anjewierden, A., & de Jong, T. (2018a). Supporting learners' experiment design. *Educational Technology Research and Development*, *66*(2), 475–491. https://doi.org/10.1007/s11423-017-9568-4

van Riesen, S., Gijlers, H., Anjewierden, A., & de Jong, T. (2018b). The influence of prior knowledge on experiment design guidance in a science inquiry context. *International Journal of Science Education*, *40*(11), 1327–1344. https://doi.org/10.1080/09500693.2018.1477263

Wang, F., Kinzie, M. B., McGuire, P., & Pan, E. (2010). Applying technology to inquiry-based learning in early childhood education. *Early Childhood Education Journal*, *37*(5), 381–389. https://doi.org/10.1007/S10643-009-0364-6

Wang, Q., Zhao, G., & Zeng, J. (2024). Examining the mediating role of digital competence and the moderating role of technostress in the effects of facilitating conditions on higher education students' digital informal learning. *Australasian Journal of Educational Technology*, *40*(5), 47–64. https://doi.org/10.14742/AJET.9324

Watters, J. J., & Diezmann, C. M. (1997). Optimising activities to meet the needs of young children gifted in mathematics and science. In P. Rilero & J. Allison (Eds.), *Creative childhood experiences in mathematics and science. Projects, activity series, and centers for early childhood* (pp. 143–170). ERIC Clearinghouse for Science, Mathematics, and Environmental Education.

Weinert, F. E., & Helmke, A. (1998). The neglected role of individual differences in theoretical models of cognitive development. *Learning and Instruction*, *8*(4), 309–323. https://doi.org/10.1016/S0959-4752(97)00024-8

White, B. Y., & Frederiksen, J. R. (1998). Inquiry, modeling, and metacognition: Making science accessible to all students. *Cognition and Instruction*, *16*(1), 3–118. https://doi.org/10.1207/S1532690XCI1601_2

Wichmann, A., & Leutner, D. (2009). Inquiry-learning: Multi-Level support with respect to inquiry, explanations and regulation during an inquiry cycle. *Zeitschrift Fur Padagogische Psychologie*, *23*(2), 117–127. https://doi.org/10.1024/1010-0652.23.2.117

Windschitl, M. (2000). Supporting the development of science inquiry skills with special classes of software. *Educational Technology Research and Development*, *48*(2), 81–95. https://doi.org/10.1007/bf02313402

Zacharia, Z. C., Manoli, C., Xenofontos, N., de Jong, T., Pedaste, M., Van Riesen, S. A. N., Kamp, E. T., Mäeots, M., Siiman, L., & Tsourlidaki, E. (2015). Identifying potential types of guidance for supporting student inquiry when using virtual and remote labs in science: A literature review. *Educational Technology Research and Development, 63*(2), 257–302. https://doi.org/10.1007/s11423-015-9370-0

Zimmerman, B. J. (1986). Becoming a self-regulated learner: Which are the key subprocesses? *Contemporary Educational Psychology*, *11*(4), 307–313. https://doi.org/10.1016/0361-476X(86)90027-5

Zimmerman, B. J. (2002). Becoming a self-regulated learner: An overview. *Theory into Practice*, *41*(2), 64–70. https://doi.org/10.1207/S15430421TIP4102_2

Appendices

Appendix A: Simulation-based environment code

```
/*
 File Name: MasterThesis.js
 Description:
 This code is used for my master thesis about the influence of character traits on the effect of adaptive guidance in a simulation-based environment. It taught kids which factors influence the pace and time a ball falls on the moon. When adaptive is set to true, it also gives real-time feedback based on the user inputs in the program.

 Although I have some programming experience, the code written is not fully optimized and uses some workarounds, as my master thesis is not aimed at writing quality code.

 Author: Louis van Maurik
 Contact: l.vanmaurik@student.utwente.nl
 Date Created: 01 October 2024
 Last Modified: 05/11/2024

 Dependencies:
 - p5.js
 */

let heightTimeSim;
let heightVelocitySim;
let drawBackground;
let explainPhase, hypothesisPhase, experimentPhase, analyzePhase, proofPhase;
let differentSimulationVariables;
let currentPhase = 'explain';
```

```
function preload() {
  //backgroundImg = loadImage('background.jpg');
  experimentImg = loadImage('ExperimentImage.png');
  ballImgRed = loadImage('ballImageRed.png');
  ballImgBlue = loadImage('ballImageBlue.png');
  ballImgOrange = loadImage('ballImageOrange.png');
  stopwatch = loadImage('stopwatch.png');
  scientistWithNameImg = loadImage('clipartProfessorWithName.png');
  progressBarVerwachting = loadImage('ProgressBarVerwachting.png');
  progressBarExperiment = loadImage('ProgressBarExperiment.png');
  progressBarControle = loadImage('ProgressBarControle.png');
  progressBarBewijzen = loadImage('ProgressBarBewijzen.png');
  explanationVideo = createImg("explanationVideo.gif");
}
function setup() {
  createCanvas(1300, 1800);
  let textBoxWidth = width-500; //

  drawBackgroundObjects = new DrawBackgroundObjects(textBoxWidth,
differentSimulationVariables);
  heightTimeSim = new HeightTimeSim(textBoxWidth);
  heightVelocitySim = new HeightVelocitySim(textBoxWidth);
  massTimeSim = new MassTimeSim(textBoxWidth);
  adaptiveFeedback = new AdaptiveFeedback(drawBackgroundObjects);

  currentSim = heightTimeSim;
```

```
}


function draw() {
  //Draws the backbground colors, rectangles and the exercise number in the top right corner
  drawBackgroundObjects.drawBackground(currentSim.exerciseNumber);


  //Draw each phase (phases are all a different 'page' of the program)
  drawCurrentPhase();
}


//Draw the currentPhase (creates them when for the first time or unhides DOM objects when
already created)
function drawCurrentPhase() {
  switch (currentPhase) {
  case 'explain':
    if (!explainPhase) {
      explainPhase = new ExplainPhase(drawBackgroundObjects, currentSim, nextPhase);
    } else {
      explainPhase.unhideAllDomObjects();
    }
    explainPhase.drawExplainPhase();
    break;
  case 'hypothesis':
    if (!hypothesisPhase) {
      hypothesisPhase = new HypothesisPhase(drawBackgroundObjects, currentSim,
adaptiveFeedback, nextPhase, previousPhase);
    } else {
      hypothesisPhase.unhideAllDomObjects();
    }
    hypothesisPhase.drawHypothesisPhase();
    image(progressBarVerwachting, 20, 200);
    break;
  case 'experiment':
    if (!experimentPhase) {
      experimentPhase = new ExperimentPhase(drawBackgroundObjects, currentSim,
adaptiveFeedback, nextPhase, previousPhase);
    } else {
      experimentPhase.unhideAllDomObjects();
    }
    experimentPhase.drawExperimentPhase();
    image(progressBarExperiment, 20, 200);
    break;
  case 'analyze':
    if (!analyzePhase) {
      analyzePhase = new AnalyzePhase(drawBackgroundObjects, currentSim, adaptiveFeedback,
nextPhase, previousPhase);
    } else {
      analyzePhase.unhideAllDomObjects();
    }
    analyzePhase.drawAnalyzePhase();
    image(progressBarControle, 20, 200);
    break;
  case 'proof':
    if (!proofPhase) {
      proofPhase = new ProofPhase(drawBackgroundObjects, currentSim, adaptiveFeedback,
nextPhase, previousPhase);
```

```
    } else {
      proofPhase.unhideAllDomObjects();
    }
    proofPhase.drawProofPhase();
    image(progressBarBewijzen, 20, 200);
    break;
  case 'end':
    fill(255);
    textSize(80);
    textStyle(BOLD);
    text('Je bent klaar!', 500, 400);
    break;
  default:
    console.log('CurrentPhase has a really weird value');
    break;
  }
}

//Sets currentPhase to the nextPhase
function nextPhase() {
  currentPhase = getNextPhase(currentPhase);
}

//Sets currentPhase to the previous phase
function previousPhase() {
  currentPhase = getPreviousPhase(currentPhase);
}
```

```
//Switch the phase to the next one and returns it
//When in the last phase, go the the next exercise or to the end screen
function getNextPhase(phase) {
  switch (phase) {
  case 'explain':
    return 'hypothesis';
  case 'hypothesis':
    return 'experiment';
  case 'experiment':
    return 'analyze';
  case 'analyze':
    return 'proof';
  case 'proof':
    if (currentSim === heightTimeSim) {
      currentSim = heightVelocitySim;
    } else if (currentSim === heightVelocitySim) {
      currentSim = massTimeSim;
    } else if (currentSim === massTimeSim) {
      return 'end';
    }
    explainPhase = undefined;
    hypothesisPhase = undefined;
    experimentPhase = undefined;
    analyzePhase = undefined;
    proofPhase = undefined;
    adaptiveFeedback.resetAllCounts();

    return 'explain';
  default:
    console.error('Unexpected phase:', phase);
    return phase;
  }
}

//Switch the phase to the one previous of it and returns it
//However, you can't go back before a explain, so you can't go the previous exercise
function getPreviousPhase(phase) {
  switch (phase) {
  case 'hypothesis':
    return 'explain';
  case 'experiment':
    return 'hypothesis';
  case 'analyze':
    return 'experiment';
  case 'proof':
    return 'analyze';
  default:
    console.error('Unexpected phase:', phase);
    return phase;
  }
}

/*

 This class is responsible for giving feedback to the user. It only knows about the
drawBackgroundObjects class, where it makes use of the popup-method.
```

```
 It is can be called by each specific phase class.

 */

class AdaptiveFeedback {
  constructor(drawBackgroundObjects) {
    this.drawBackgroundObjects = drawBackgroundObjects;

    //Creates all counts
    this.resetAllCounts();
  }

  //resets all counts when a new experiment is started (or creates them when first called in the
constructor)
  resetAllCounts() {
    this.checkIndCorrectCount = 0;
    this.checkDepCorrectCount = 0;
    this.checkIndGoalHypothesisCount = 0;
    this.checkDepGoalHypothesisCount = 0;
    this.checkVariedIndCount = 0;
    this.checkVariedAndControlledIndCount = 0;
    this.checkIndGoalAnalyzeCount = 0;
    this.checkDepGoalAnalyzeCount = 0;
    this.checkCheckBoxVariedAndControlledIndCount = 0;
    this.checkStatementAnalyzeCount = 0;
  }
  //checks if all adaptive criteria are met by calling each check function in the HYPOTHESIS
phase
  giveAdaptiveFeedbackHypothesisPhase(goal, reqIndVar, reqDepVar, givenIndVar, givenDepVar,
indVariableOptions, depVariableOptions, dropdownIndependentVar, dropdownDependentVar) {
    return this.checkIndCorrect(givenIndVar, indVariableOptions, reqIndVar,
dropdownIndependentVar) &&
      this.checkDepCorrect(givenDepVar, depVariableOptions, reqDepVar,
dropdownDependentVar) &&
      this.indIsGoalHypothesis(goal, givenIndVar, reqIndVar, dropdownIndependentVar) &&
      this.depIsGoalHypothesis(goal, givenDepVar, reqDepVar, dropdownDependentVar);
  }

  //Checks if a independent variable is placed on the independent variable spot, if not: give an
error and return false
  checkIndCorrect(givenIndVar, indVariableOptions, reqIndVar, dropdown) {
    if (!indVariableOptions.includes(givenIndVar)) {
      if (this.checkIndCorrectCount === 0) {
        this.checkIndCorrectCount++;
        const text = "Een experiment heeft altijd een optie nodig die je vooraf kan kiezen. Bij iets
zoals de <b>'" + givenIndVar + "'</b> weet je pas achteraf wat de waarde was. <br><br>Probeer
daarom een optie in het eerste vakje te kiezen die je voordat het experiment doet al kan
aanpassen.";
        this.drawBackgroundObjects.createPopUp('Let op:', text);
      } else if (this.checkIndCorrectCount === 1) {
        this.checkIndCorrectCount++;
        const text = "Laat ik wat duidelijker zijn. Opties die je zou kunnen aanpassen zijn: <b>" +
indVariableOptions[0] + ", " + indVariableOptions[1] + ", " + indVariableOptions[2] + "</b>.
```

```
<br><br>Kies 1 van deze opties, en zorg ervoor dat die overeenkomt met wat we willen
onderzoeken."
          this.drawBackgroundObjects.createPopUp('Nog niet helemaal:', text);
      } else {
        const text = "Hmmm, laat ik je bij deze stap wat meer helpen. De juiste keuze voor het
eerste vakje is: '<b>" + reqIndVar + "</b>'.";
        this.drawBackgroundObjects.createPopUp('Het juiste antwoord:', text);
      }
      dropdown.style('font-size', '16px');
      dropdown.style('border', '4px solid red');
      return false;
    } else {
      return true;
    }
  }

  //Checks if a independent variable is placed on the dependent variable spot, if not: give an
error and return false
  checkDepCorrect(givenDepVar, depVariableOptions, reqDepVar, dropdown) {
    if (!depVariableOptions.includes(givenDepVar)) {
      if (this.checkDepCorrectCount === 0) {
        this.checkDepCorrectCount++;
        const text = "Een experiment heeft altijd een optie nodig die je pas meet aan het einde. Bij
iets zoals '<b>" + givenDepVar + "</b>' is een optie die je aanpast aan het begin.
<br><br>Probeer daarom een optie in het derde vakje te kiezen die je kan meten.";
        this.drawBackgroundObjects.createPopUp('Let op:', text);
      } else if (this.checkDepCorrectCount === 1) {
        this.checkDepCorrectCount++;
        const text = "Laat ik wat duidelijker zijn. Opties die je zou kunnen meten zijn: <b>" +
depVariableOptions[0] + ", " + depVariableOptions[1] + "</b>. <br><br>Kies 1 van deze opties,
en zorg ervoor dat die overeenkomt met wat we willen onderzoeken." ;
        this.drawBackgroundObjects.createPopUp('Nog niet helemaal:', text);
      } else {
        const text = "Hmmm, laat ik je bij deze stap wat meer helpen. De juiste keuze voor het
derde vakje is: '<b>" + reqDepVar + "</b>'.";
        this.drawBackgroundObjects.createPopUp('Het juiste antwoord:', text);
      }
      dropdown.style('font-size', '16px');
      dropdown.style('border', '4px solid red');
      return false;
    } else {
      return true;
    }
  }

  //Checks if the given dependent variable is the one as described in the goal in the hypothesis
phase, if not: give an error and return false
  indIsGoalHypothesis(goal, givenIndVar, reqIndVar, dropdown) {
    if (givenIndVar != reqIndVar) {
      if (this.checkIndGoalHypothesisCount === 0) {
        this.checkIndGoalHypothesisCount++;
        const text = "De keuze in het eerste vakje komt niet overeen met wat we willen ontdekken.
Dus let op: <br><br>" + goal;
        this.drawBackgroundObjects.createPopUp('Wat willen we ontdekken?', text);
      } else {
        const text = "In het eerste vakje staat '<b>" + givenIndVar + "</b>'. Dit is helaas niet wat willen onderzoeken. De juiste keuze is: <b>" + reqIndVar + "</b>.";
        this.drawBackgroundObjects.createPopUp('Het juiste antwoord:', text);
      }
      dropdown.style('font-size', '16px');
      dropdown.style('border', '4px solid red');
      return false;
    } else {
      return true;
    }
  }


  //Checks if the given independent variable is the one as described in the goal in the hypothesis phase, if not: give an error and return false
  depIsGoalHypothesis(goal, givenDepVar, reqDepVar, dropdown) {
    if (givenDepVar != reqDepVar) {
      if (this.checkDepGoalHypothesisCount === 0) {
        this.checkDepGoalHypothesisCount++;
        const text = "De keuze in het derde vakje komt niet overeen met wat we willen ontdekken. Dus let op: <br><br>" + goal;
        this.drawBackgroundObjects.createPopUp('Wat willen we ontdekken?', text);
      } else {
        const text = "In het derde vakje staat '<b>" + givenDepVar + "</b>'. Dit is helaas niet wat willen onderzoeken. De juiste keuze is: <b>" + reqDepVar + "</b>.";
        this.drawBackgroundObjects.createPopUp('Het juiste antwoord:', text);
      }
      dropdown.style('font-size', '16px');
      dropdown.style('border', '4px solid red');
      return false;
    } else {
      return true;
    }
  }

  //checks if all experiment criteria are met by calling each checks function in the EXPERIMENT
phase
  giveAdaptiveFeedbackExperimentPhase(results, reqIndVar, amountOfDepVars) {
    // results is a lists of list with the following data in this particulair order: Weight, Height,
Color, TimeToDrop, Velocity
    let indIndex = this.getIndVarIndex(reqIndVar);
    return  this.checkMoreThenOne(results) &&
      this.checkVariedInd(results, indIndex, reqIndVar) &&
      this.checkVariedIndAndControlled(results, indIndex, amountOfDepVars, reqIndVar);
  }

  //Checks if the user has done more then one experiment, if not: give an error and return false
  checkMoreThenOne(results) {
    if (results.length < 2) {
      const text = "Bij een experiment wil je steeds iets veranderen. Zo ontdek je meer! Als je
steeds dezelfde bal laat vallen, leer je weinig nieuws. Maar door bijvoorbeeld een zwaardere bal
te gebruiken of de hoogte te veranderen, kun je zien wat er dan gebeurt.";
      this.drawBackgroundObjects.createPopUp('Niet genoeg experimenten!', text);
      return false;
    } else {
      return true;
    }
  }

  //Check if the independent variable has been changed between experiments by looking at the
results, if not: give an error and return false
  checkVariedInd(results, index, reqIndVar) {
    if (results.length === 0) {
      return false;  // No data to compare
    }

    // Take the value from the first list at the specified index to compare with others
    // Note: It only has to be the first, because when it is changed once, it already is different
from the first result
    let firstValue = results[0][index];

    // Compare the value at the specified index in all subsequent lists
    for (let i = 1; i < results.length; i++) {
      if (results[i][index] !== firstValue) {
        return true; // Found a difference, return false
      }
    }

    if (this.checkVariedIndCount === 0) {
      this.checkVariedIndCount++;
      const text = "Kijk nog is goed naar wat we willen ontdekken en wat we verwachten. Wat
moeten we aanpassen in het experiment om daar achter te komen?";
      this.drawBackgroundObjects.createPopUp('Wat wilden we ontdekken?', text);
    } else {
      const text = "We willen kijken wat voor effect de '<b>" + reqIndVar + "</b>' heeft. Pas
daarom de '<b>" + reqIndVar + "</b>' aan in het experiment!";
      this.drawBackgroundObjects.createPopUp('Wat wilden we ontdekken?', text);
    }

    // If no differences were found, return false
    return false;
  }

  //Checks if the independent variable has changed AND it is controlled, so the other ones
weren't changed. If not: give an error and return false
  checkVariedIndAndControlled(results, index, amountOfDepVars, reqIndVar) {
    //make a list of independent indexes where the value should be the same, so without the
independent index
    let notIndexes = [];
    for (let i = 0; i < amountOfDepVars; i++) {
      if (i !== index) {
        notIndexes.push(i);
      }
    }

    // Loop through each result and compare with each other
    for (let i = 0; i < results.length; i++) {
      for (let j = i + 1; j < results.length; j++) {

        //Check only when the required independent variables are not the same
        if (results[i][index] !== results[j][index]) {
          let otherIndexesAreTheSame = true;


          //checks is all otherIndexesAre are the same for these particulair results
          //Note: otherIndexes looks only at the other dependent variables
          for (let k = 0; k < notIndexes.length; k++) {
            if (results[i][notIndexes[k]] !== results[j][notIndexes[k]]) {
              otherIndexesAreTheSame = false;
            }
          }
          if (otherIndexesAreTheSame) {
            // If there are no differences between the other independent variables, and the required
independent variables are not the same,
            // return true
            return true;
          }
        }
      }
    }

    if (this.checkVariedAndControlledIndCount === 0) {
      this.checkVariedAndControlledIndCount++;
      const text = "Je bent op de goede weg. Je hebt namelijk wel de '<b>" + reqIndVar + "</b>'
aangepast. Alleen heb je ook ondertussen iets anders aangepast. Probeer ervoor te zorgen dat je
maar 1 ding tegelijk aanpast!";
      this.drawBackgroundObjects.createPopUp('Pas maar 1 ding tegelijk aan:', text);
    } else {
      const text = "Nog niet helemaal. Laat ik het in wat kleinere stapjes uitleggen. Klik eerst op
<i>test</i>. Pas daarna alleen de '<b>" + reqIndVar + "</b>' aan. Klik dan weer op <i>test</i>.
Nu heb je twee experimenten die exact hetzelfde zijn, naast de '<b>" + reqIndVar + "</b>'.";
      this.drawBackgroundObjects.createPopUp('Pas maar 1 ding tegelijk aan:', text);
    }
    return false;
  }

  //Checks if the given dependent variable is the one as described in the goal in the analyze
phase, if not: give an error and return false
  indIsGoalAnalyze(givenIndVar, reqIndVar, dropdown) {
    if (givenIndVar != reqIndVar) {
      if (this.checkIndGoalAnalyzeCount === 0) {
        this.checkIndGoalAnalyzeCount++;
        const text = "De keuze in het eerste vakje komt niet overeen met wat we willen
ontdekken.";
        this.drawBackgroundObjects.createPopUp('Let op:', text);
      } else {
        const text = "In het eerste vakje staat '<b>" + givenIndVar + "</b>'. Dit is helaas niet wat
wilden onderzoeken. De juiste keuze is: <b>" + reqIndVar + "</b>.";
        this.drawBackgroundObjects.createPopUp('Het juiste antwoord:', text);
      }
      dropdown.style('font-size', '16px');
      dropdown.style('border', '4px solid red');
      return false;
    } else {
      return true;
    }
  }



  //Checks if the given independent variable is the one as described in the goal in the analysis
phase, if not: give an error and return false
  depIsGoalAnalyze(givenDepVar, reqDepVar, dropdown) {
    if (givenDepVar != reqDepVar) {
      if (this.checkDepGoalAnalyzeCount === 0) {
        this.checkDepGoalAnalyzeCount++;
        const text = "De keuze in het derde vakje komt niet overeen met wat we willen
ontdekken.";
        this.drawBackgroundObjects.createPopUp('Let op:', text);
      } else {
        const text = "In het derde vakje staat '<b>" + givenDepVar + "</b>'. Dit is helaas niet wat
wilden onderzoeken. De juiste keuze is: <b>" + reqDepVar + "</b>.";
        this.drawBackgroundObjects.createPopUp('Het juiste antwoord:', text);
      }
      dropdown.style('font-size', '16px');
      dropdown.style('border', '4px solid red');
      return false;
    } else {
      return true;
    }
  }

  //checks if all adaptive criteria are met by calling each checks function in the ANALYZE phase
  //Note: DD stands for dropdown-menu
  giveAdaptiveFeedbackAnalyzePhase(results, givenIndVar, givenDepVar, givenIndVarChange,
givenDepVarChange,
    givenIndVarCheck, givenDepVarCheck, givenIndVarChangeCheck, givenDepVarChangeCheck,
reqDepVar, reqIndVar, hypothesisCheck,
    DD_IndVarCheck, DD_IndChangeCheck, DD_DepVarCheck, DD_DepChangeCheck,
DD_HypothesisCheck) {
    return this.indIsGoalAnalyze(givenIndVarCheck, reqIndVar, DD_IndVarCheck) &&
      this.depIsGoalAnalyze(givenDepVarCheck, reqDepVar, DD_DepVarCheck) &&
      this.checkIsEqualToWhatHappend(givenIndVarCheck, givenDepVarCheck,
givenIndVarChangeCheck, givenDepVarChangeCheck, DD_IndChangeCheck,
DD_DepChangeCheck) &&
      this.checkIsHypothesisCorrect(givenIndVar, givenDepVar, givenIndVarChange,
givenDepVarChange, hypothesisCheck, DD_HypothesisCheck);
  }

  //Checks if the description of what happend is true, if not: give an error and return false
  checkIsEqualToWhatHappend(givenIndVarCheck, givenDepVarCheck,
givenIndVarChangeCheck, givenDepVarChangeCheck, dropdownOne, dropdownTwo) {
    const relationshipDropdownInput =
this.getRelationshipDropdownInput(givenIndVarChangeCheck, givenDepVarChangeCheck);
    const realRelationship = this.getRelationshipFromInd(givenIndVarCheck);

    if (relationshipDropdownInput === realRelationship) {
      return true;
    }

    //Give the error pop-up
    if (this.checkStatementAnalyzeCount === 0) {
      this.checkStatementAnalyzeCount++;
      const text = "De '<b>" + givenIndVarCheck + "</b>' en de '<b>" + givenDepVarCheck +
"</b>' zijn beide correct gekozen! Die hoef je dus niet meer aan te passen. Alleen klopt wat je
zegt niet helemaal!  <br><br>Kijk nog is goed wat er gebeurde, als '<b>" + givenIndVarCheck +
"</b>' minder/meer werd, wat gebeurde er dan met '<b>" + givenDepVarCheck + "</b>'?";
      this.drawBackgroundObjects.createPopUp('Wat gebeurde er precies?', text);
    } else {
      if (realRelationship === 'direct') {
        const text = "Het klopt nog steeds niet helemaal.  Het juiste antwoord is: Toen ik '" +
givenIndVarCheck + "' '<b>meer liet worden</b>', ging '" + givenDepVarCheck + "' '<b>meer
worden</b>'. Pas dit aan!";
        this.drawBackgroundObjects.createPopUp('Wat gebeurde er precies?', text);
      } else if (realRelationship === 'independent') {
        const text = "Het klopt nog steeds niet helemaal.  Het juiste antwoord is: Toen ik '" +
givenIndVarCheck + "' '<b>meer of minder liet worden</b>', ging '" + givenDepVarCheck + "'
'<b>hetzelfde blijven</b>'. Pas dit aan!";
        this.drawBackgroundObjects.createPopUp('Wat gebeurde er precies?', text);
      }
    }

    //Make dropdowns orange
    dropdownOne.style('font-size', '16px');
    dropdownOne.style('border', '4px solid orange');
    dropdownTwo.style('font-size', '16px');
    dropdownTwo.style('border', '4px solid orange');

    return false;
  }
  //Checks if the stated hypothesis was correct or not, if not: give an error and return false
  checkIsHypothesisCorrect(givenIndVar, givenDepVar, givenIndVarChange,
givenDepVarChange, hypothesisCheck, dropdown) {
    const relationshipDropdownInput = this.getRelationshipDropdownInput(givenIndVarChange,
givenDepVarChange);
    const realRelationship = this.getRelationshipFromInd(givenIndVar);

    //when the inputted relationship is true and the hypothesis check is 'waar', return true
    //But also, when when the inputted relationship is false and the hypothesis check is 'niet
waar', return also true
    if ((relationshipDropdownInput === realRelationship && hypothesisCheck === 'waar') ||
(relationshipDropdownInput !== realRelationship && hypothesisCheck === 'niet waar')) {
      return true;
    }

    const text = "Kijk nog is goed naar je verwachting. Was je verwachting nou juist WEL of NIET
waar?";
    this.drawBackgroundObjects.createPopUp("Klopt je verwachting?", text);

    //Make dropdown red
    dropdown.style('font-size', '16px');
    dropdown.style('border', '4px solid red');

    return false;
  }

  // Helper function to determine the relationship between variables
  // Note: this function is fully 'hardcoded'. It would be more elegant to actualy check it in the
results
  getRelationshipFromInd(givenIndVar) {
    if (givenIndVar === 'massa van de bal' || givenIndVar === 'kleur van de bal') {
      //either 'massa van de bal' and 'kleur van de bal' don't have any effect on time to drop and
the velocity
      return 'independent';
    } else if (givenIndVar === 'hoogte van de bal') {
      //the height of the ball has a direct relationship with either time to drop and the velocity
      return 'direct';
    }
  }

  // Helper function to determine the relationship between variables from dropdown menu
inputs
  getRelationshipDropdownInput(indVarChange, depVarChange) {
    // Check for 'direct' relationship
    if (
      (
      (indVarChange === 'meer laat worden' || indVarChange === 'meer liet worden') &&
      depVarChange === 'meer worden'
      ) ||
      (
      (indVarChange === 'minder laat worden' || indVarChange === 'minder liet worden') &&
      depVarChange === 'minder worden'
      )
      ) {
      return 'direct';
    }

    // Check for 'inverse' relationship
    if (
      ((indVarChange === 'meer laat worden' || indVarChange === 'meer liet worden') &&
      depVarChange === 'minder worden')
      ||
      ((indVarChange === 'minder laat worden' || indVarChange === 'minder liet worden') &&
      depVarChange === 'meer worden')
      )
    {
      return 'inverse';
    }

    // Check for 'independent' relationship
    if (
      indVarChange === 'hetzelfde laat' ||
      indVarChange === 'hetzelfde liet' ||
      depVarChange === 'hetzelfde blijven'
      ) {
      return 'independent';
    }

    // If no valid relationship is found, give an error message
    console.log('Something is going very wrong: no relationship was found in the
getRelationshipMethod')
  }
  //checks if all criteria are met by calling each checks function in the PROOF phase
  giveAdaptiveFeedbackProofPhase(results, reqIndVar, indVariableOptions, checkboxesList) {
    console.log('I just got in the adaptiveFeedbackProofPhase function. CheckboxesList = ' +
checkboxesList);
    let indIndex = this.getIndVarIndex(reqIndVar);
    let checkedCheckboxesIndexList = this.indexListCheckedBoxes(checkboxesList);
    return  this.moreThenOneOptionChecked(checkedCheckboxesIndexList) &&
      this.independentCheckBosIsChanged (results, indIndex, indVariableOptions,
checkedCheckboxesIndexList) &&
      this.independentCheckBosIsChangedAndControlled(results, indIndex, indVariableOptions,
reqIndVar, checkedCheckboxesIndexList);
  }

  indexListCheckedBoxes(checkboxesList) {
    let checkedCheckboxesIndexList = [];
    console.log('I just got in the indexListCheckedBoxes function. CheckboxesList = ' +
checkboxesList);

    for (let i = 0; i < checkboxesList.length; i++) {
      // Check if the checkbox is checked and add the index to a this list
      if (checkboxesList[i].checked()) {
        checkedCheckboxesIndexList.push(i);
      }
    }
    return checkedCheckboxesIndexList;
  }

  //checks if there are atleast 2 boxes selected, if not: give an error and return false
  moreThenOneOptionChecked(checkedCheckboxesIndexList) {
    if (checkedCheckboxesIndexList.length > 1) {
      return true;
    } else {
      const text = "Momenteel heb je maar 1 resultaat geselecteerd. Om te bewijzen dat er iets wel
of niet veranderd is in het experiment, heb je er <b>minstens twee</b> nodig.";
      this.drawBackgroundObjects.createPopUp('Niet genoeg bewijs!', text);
      return false;
    }
  }

  //checks if the required independent variable is not the same between selected checkboxes, if
not: give an error and return false
  independentCheckBosIsChanged(results, indIndex, indVariableOptions,
checkedCheckboxesIndexList) {
    for (let i = 0; i < checkedCheckboxesIndexList.length; i++) {
      for (let j = i+1; j < checkedCheckboxesIndexList.length; j++) {
        //checks if the independent variable are the same. If so, return false and give an error
message
        if (results[checkedCheckboxesIndexList[i]][indIndex] ===
results[checkedCheckboxesIndexList[j]][indIndex]) {
          const text = "Momenteel heb je resultaten geselecteerd waartussen soms de '<b>" +
indVariableOptions[indIndex] + "</b>' hetzelfde is. Zorg ervoor dat de '<b>" +
indVariableOptions[indIndex] + "</b>' verschilt tussen de aangevinkte resultaten!";
          this.drawBackgroundObjects.createPopUp('De ' + indVariableOptions[indIndex] + ' is
hetzelfde!', text);
          return false;
        }
      }
    }
    return true;
  }

  //checks if the other independent variables are not changed between the selected checkboxes,
, if not: give an error and return false
  independentCheckBosIsChangedAndControlled(results, indIndex, indVariableOptions,
reqIndVar, checkedCheckboxesIndexList) {
    let amountOfDepVars = indVariableOptions.length;
    let checkedResults = [];

    for (let i = 0; i < checkedCheckboxesIndexList.length; i++) {
      checkedResults.push(results[checkedCheckboxesIndexList[i]]);
    }

    let notIndexes = [];
    for (let i = 0; i < amountOfDepVars; i++) {
      if (i !== indIndex) {
        notIndexes.push(i);
      }
    }

    // Loop through each result and compare with each other
    for (let i = 0; i < checkedResults.length; i++) {
      for (let j = i + 1; j < checkedResults.length; j++) {
        if (checkedResults[i][indIndex] !== checkedResults[j][indIndex]) {
          let otherIndexesAreTheSame = true;
          for (let k = 0; k < notIndexes.length; k++) {
            if (checkedResults[i][notIndexes[k]] !== checkedResults[j][notIndexes[k]]) {
              otherIndexesAreTheSame = false;
            }
          }
          if (otherIndexesAreTheSame) {
            return true;
          }
        }
      }
    }

    if (this.checkCheckBoxVariedAndControlledIndCount === 0) {
      this.checkCheckBoxVariedAndControlledIndCount++;
      const text = "Je bent op de goede weg. Je hebt namelijk wel opties gekozen waar de '<b>" +
reqIndVar + "</b>' veranderd. Alleen heb je ook ondertussen opties aangevinkt waar iets anders
is veranderd. <br><br> Kies opties waar de '<b>" + reqIndVar + "</b>' veranderd en de rest
hetzelfde blijft.";
      this.drawBackgroundObjects.createPopUp('Niet 1 ding veranderd', text);
    } else {
      const text = "Nog niet helemaal. Het klopt dat de '<b>" + reqIndVar + "</b>' veranderd bij de
opties die je hebt aangevinkt. <br><br> Alleen moet de '<b>" +
indVariableOptions[notIndexes[0]] + "</b>' en de '<b>" + indVariableOptions[notIndexes[1]] +
"</b>' hetzelfde blijven. Die mogen niet veranderen. <br><br> Kies opties waar de '<b>" +
reqIndVar + "</b>' veranderd en de rest hetzelfde blijft.";
      this.drawBackgroundObjects.createPopUp('Pas maar 1 ding tegelijk aan:', text);
    }
    return false;
```

```
 }

 //Helper method that returns the index depending on what the independent variable is
 //Note: the index is the one needed for get the right column in the results list
 getIndVarIndex(reqIndVar) {
   switch (reqIndVar) {
   case 'massa van de bal':
     return 0;
   case 'hoogte van de bal':
     return 1;
   case 'kleur van de bal':
     return 2;
   default:
     console.log('The getIndVarIndex is broken, it got a unkown reqIndVar string and does not
know which index to return');
     return undefined;
   }
 }

 //Helper method that returns the index depending on what the dependent variable is
 //Note: the index is the one needed for get the right column in the results list
 getDepVarIndex(reqDepVar) {
   switch (reqDepVar) {
   case 'massa van de bal':
     return 3;
   case 'hoogte van de bal':
     return 4;
   default:
     console.log('The getDepVarIndex is broken, it got a unkown reqDepVar string and does not
know which index to return');
     return undefined;
   }
 }
}

class AnalyzePhase {
 constructor(drawBackgroundObjects, currentSim, adaptiveFeedback, nextPhaseMethod,
previousPhaseMethod) {
   this.drawBackgroundObjects = drawBackgroundObjects;
   this.currentSim = currentSim;
   this.adaptiveFeedback = adaptiveFeedback;
   this.nextPhaseMethod = nextPhaseMethod;
   this.previousPhaseMethod = previousPhaseMethod;

   //Options slection menus
   this.variableOptions = this.currentSim.variableOptions;
   this.dependentChanges = ['meer liet worden', 'minder liet worden', 'hetzelfde liet'];
   this.independentChanges = ['meer worden', 'minder worden', 'hetzelfde blijven'];
   this.hypothesisCheck = ['waar', 'niet waar'];

   // Define options for dropdown menus
   this.createDropdowns();
   this.allSelected = false;
   this.nextButton = this.createNextButton();
   this.previousButton = this.createPreviousButton();
 }
 // Create all drow
 createDropdowns() {
   // Create independent and dependent variable dropdowns
   this.dropdownIndependentVarCheck =
this.drawBackgroundObjects.createDropdown(this.variableOptions, 510, 595,
'givenIndVarCheck', this.currentSim);
   this.dropdownIndependentChangeCheck =
this.drawBackgroundObjects.createDropdown(this.dependentChanges, 790, 595,
'givenIndVarChangeCheck', this.currentSim);
   this.dropdownDependentVarCheck =
this.drawBackgroundObjects.createDropdown(this.variableOptions, 485, 635,
'givenDepVarCheck', this.currentSim);
   this.dropdownDependentChangeCheck =
this.drawBackgroundObjects.createDropdown(this.independentChanges, 765, 635,
'givenDepVarChangeCheck', this.currentSim);
   this.dropdownHypothesisCheck =
this.drawBackgroundObjects.createDropdown(this.hypothesisCheck, 665, 685,
'hypothesisCheck', this.currentSim);
 }

 createNextButton() {
   return this.drawBackgroundObjects.createButton('Ga naar bewijzen',
this.doNextButton.bind(this), 895, 770, '300px', '60px');
 }

 createPreviousButton() {
   return this.drawBackgroundObjects.createButton('Ga terug', this.doPreviousButton.bind(this),
695, 770, '150px', '60px');
 }

 // Main method to draw the analyze phase
 drawAnalyzePhase() {
   this.drawTitle();
   this.drawExerciseBox();
   this.drawGoalBox();
   this.drawExpectedOutcomeBox();
   this.drawAnalyzeBox(400, 550);
   this.drawResultBox();

   this.updateAllSelected();
   this.updateNextButtonStyle();
 }

 // Draws the title for the analyze phase
 drawTitle() {
   fill(255);
   textSize(40);
   textStyle(BOLD);
   text('Onderzoeksfase: Verwachting controleren', 350, 70);
 }

 // Draws the exercise box with an explanation
 drawExerciseBox() {
   const titleExercise = 'Opdracht ' + this.currentSim.exerciseNumber + '.3';
   const explanation = this.currentSim.analyzeExp;
   this.drawBackgroundObjects.drawTextBox(titleExercise, explanation, 400, 150);
 }

 // Draws the goal box
 drawGoalBox() {
   const goal = this.currentSim.goal;
   this.drawBackgroundObjects.drawTextBox('Wat we wilden ontdekken:', goal, 400, 300);
 }

 // Draws the expected outcome (hypothesis) box
 drawExpectedOutcomeBox() {
   const hypothesis = this.currentSim.getHypothesis();
   this.drawBackgroundObjects.drawTextBox('Wat je had verwacht:', hypothesis, 400, 420);
 }

 drawAnalyzeBox(xpos, ypos) {
   const textsize = 20;

   // Draw hypothesis background box
   fill(255);
   rect(xpos, ypos, this.drawBackgroundObjects.textBoxWidth, 180);

   // Draw hypothesis title
   fill(0);
   textSize(textsize);
   textStyle(BOLD);
   text('Wat er gebeurde:', xpos + 10, ypos + 30);
```

```
    // Draw hypothesis structure text
    textStyle(NORMAL);
    text('Toen ik de', xpos + 10, ypos + 65);
    text(',', xpos + 670, ypos + 65);
    text('ging de ', xpos + 10, ypos + 105);
    text('.', xpos + 645, ypos + 105);
    text('Dus mijn verwachting was', xpos + 10, ypos + 155);
    text('.', xpos + 550, ypos + 155);
  }

  drawResultBox() {
    const textsize = 20;
    const xpos = 400;
    const ypos = 950;
    const columnTitles = ['Test', 'Massa (KG)', 'Hoogte (M)', 'Kleur', 'Tijd (S)', 'Snelheid (M/S)']; //
Add "Test" column

    // Calculate the length of the array (number of rows)
    const LENGTHOFARRAY = this.currentSim.results.length;

    // Calculate the height of the box dynamically based on array length
    const rowHeight = 30; // Height per row of data
    const boxHeight = 45 + rowHeight * (LENGTHOFARRAY + 1); // +1 for the title row

    // Draw background box for results
    fill(36, 106, 115);
    rect(xpos, ypos, this.drawBackgroundObjects.textBoxWidth, boxHeight, 30);
```

```
// Draw result box title
fill(255);
textSize(textsize);
textStyle(BOLD);
textAlign(CENTER);
text('Resultaten:', xpos + this.drawBackgroundObjects.textBoxWidth / 2, ypos + 30);

// Set up for drawing column titles
textSize(20);
textStyle(BOLD);

// Define the column widths for each of the 6 columns (including "Test")
const colWidth = this.drawBackgroundObjects.textBoxWidth / 6;

// Draw the column titles, including "Test" as the first one
for (let i = 0; i < columnTitles.length; i++) {
  text(columnTitles[i], xpos + colWidth * i + 50, ypos + 60); // Adjusted position for title
}

// Set text to normal style for the results
textStyle(NORMAL);

// Draw each element in its respective column
for (let i = 0; i < LENGTHOFARRAY; i++) {
  const resultRow = this.currentSim.results[i];

  // Draw the test number in the first column (starting at #1)
```

```
      text('#' + (i+1), xpos + 50, ypos + 90 + (i * rowHeight)); // First column for test number

      // Draw the rest of the results in their respective columns
      for (let j = 0; j < resultRow.length; j++) {
        text(resultRow[j], xpos + colWidth * (j + 1) + 50, ypos + 90 + (i * rowHeight)); // Adjust for
the additional column
      }
    }

    textAlign(LEFT);
  }

  // Check if all dropdowns have a selection to enable/disable the next button and change its
color
  updateAllSelected() {
    if (this.currentSim.getGivenIndVarCheck() !== undefined &&
      this.currentSim.getGivenDepVarCheck() !== undefined &&
      this.currentSim.getGivenIndVarChangeCheck() !== undefined &&
      this.currentSim.getGivenDepVarChangeCheck() !== undefined &&
      this.currentSim.getHypothesisCheck() !== undefined) {
      this.allSelected = true;
    } else {
      this.allSelected = false;
    }
  }

  // Updates the color of the nextButton when all dropdownmenus are selected
  updateNextButtonStyle() {
    if (this.allSelected) {
      this.nextButton.style('background', 'linear-gradient(to bottom, #FFB347, #FF8000)');
    }
  }

  //Each time the next button is pressed, both get made black (as both are made orange when
one of them is false;
  updateDropdownStyle() {
    this.dropdownIndependentChangeCheck.style('border', '1px solid black');
    this.dropdownDependentChangeCheck.style('border', '1px solid black');
  }

  //calls the callback function of the main class to go to phase button when allSelected is true,
and no feedback needs to be given
  doNextButton() {
    this.currentSim.setEvidence();
    this.updateDropdownStyle();

    const results = this.currentSim.results;
    const givenIndVar = this.currentSim.getGivenIndVar();
    const givenDepVar = this.currentSim.getGivenDepVar();
    const givenIndVarChange = this.currentSim.getGivenIndVarChange();
    const givenDepVarChange = this.currentSim.getGivenDepVarChange();
    const givenIndVarCheck = this.currentSim.getGivenIndVarCheck();
    const givenDepVarCheck = this.currentSim.getGivenDepVarCheck();
    const givenIndVarChangeCheck = this.currentSim.getGivenIndVarChangeCheck();
    const givenDepVarChangeCheck = this.currentSim.getGivenDepVarChangeCheck();
    const reqDepVar = this.currentSim.getReqDepVar();
    const reqIndVar = this.currentSim.getReqIndVar();
    const hypothesisCheck = this.currentSim.getHypothesisCheck();

    if (this.allSelected) {
      if (!adaptive || this.adaptiveFeedback.giveAdaptiveFeedbackAnalyzePhase(results,
givenIndVar, givenDepVar, givenIndVarChange, givenDepVarChange,
        givenIndVarCheck, givenDepVarCheck, givenIndVarChangeCheck,
givenDepVarChangeCheck, reqDepVar, reqIndVar, hypothesisCheck,
        this.dropdownIndependentVarCheck, this.dropdownIndependentChangeCheck,
this.dropdownDependentVarCheck,
        this.dropdownDependentChangeCheck, this.dropdownHypothesisCheck)) {

        this.hideAllDomObjects();
        this.nextPhaseMethod();
      }
    }
  }

  doPreviousButton() {
    this.hideAllDomObjects();
    this.previousPhaseMethod();
  }

  hideAllDomObjects() {
    this.dropdownIndependentVarCheck.hide();
    this.dropdownIndependentChangeCheck.hide();
    this.dropdownDependentVarCheck.hide();
    this.dropdownDependentChangeCheck.hide();
    this.dropdownHypothesisCheck.hide();
    this.nextButton.hide();
    this.previousButton.hide();
  }

  unhideAllDomObjects() {
    this.dropdownIndependentVarCheck.show();
    this.dropdownIndependentChangeCheck.show();
    this.dropdownDependentVarCheck.show();
    this.dropdownDependentChangeCheck.show();
    this.dropdownHypothesisCheck.show();
    this.nextButton.show();
    this.previousButton.show();
  }
}

/*

  All important variables that are not phase specific are stored within this class. As each
experiment is slightly different for some variables,
  but the rest works fully the same, it uses a parent and child class structure.

*/

class DifferentSimulationVariables {
  constructor(temptTextBoxWidth) {
    this.textBoxWidth = temptTextBoxWidth;
    this.hypothesisExp = this.ArrayOfStrings('Wat denk je dat er gaat gebeuren? Dat is een hele
belangrijke vraag voor een wetenschapper. Laten we die vraag eens samen beantwoorden. Dat
begint door te kijken naar het doel van het onderzoek. Wat willen we ontdekken? Daarna
beschrijf je je verwachting. Dat doe je door de zin af te maken.');
    this.experimentExp = this.ArrayOfStrings('Het is nu tijd om het experiment uit te voeren. Wat
spannend! De bedoeling is dat we nu onze verwachting gaan testen. Klopt onze verwachting
wel? Of gebeurt er totaal iets anders?  Gebruik het experiment om er achter te komen of je
verwachting klopt!');
    this.analyzeExp = this.ArrayOfStrings('We gaan nu kijken of wat er gebeurde hetzelfde is als
wat we dachten dat er ging gebeuren. Wat gebeurde er aan het eind? Was onze verwachting
uteindelijk waar of niet waar? Klik op de juiste antwoorden in de vakjes onderin.');
    this.proofExp = this.ArrayOfStrings('Als je wilt laten zien dat iets echt gebeurt is, moet je de
goede resultaten kiezen die dat laten zien. Gelukkig hebben we alles opgeschreven. Nu moet je
alleen de juiste kiezen die onze beschrijving van er gebeurde bewijzen. Resultaten die niet
helpen, hoef je niet te kiezen!');


    this.indVariableOptions = ['massa van de bal', 'hoogte van de bal', 'kleur van de bal']
    this.depVariableOptions = ['tijd voordat de bal de grond raakt', 'snelheid als de bal de grond
raakt'];

    this.variableOptions = this.indVariableOptions.concat(this.depVariableOptions);

    //the given hyptohesis variables by the user
    this.givenIndVar = undefined;
    this.givenDepVar = undefined;
    this.givenIndVarChange = undefined;
    this.givenDepVarChange = undefined;
    //the given analyze variables by the user
    this.givenIndVarCheck = undefined;
    this.givenDepVarCheck = undefined;
    this.givenIndVarChangeCheck = undefined;
    this.givenDepVarChangeCheck = undefined;
    this.hypothesisCheck = undefined;

    //Strings (hypothesis/evidence) created by the user by putting together there dropdown
menu inputs
    this.givenHypothesis = ['No hypothesis has been set yet', 'this text is a placeholder'];
    this.givenEvidence = ['No evidence has been selected', 'This text is a placeholder'];

    //A lists of list with the following data in each one: Weight, Height, Color, TimeToDrop,
Velocity
    //Each time a experiment get's tested, it will be added to this list
    this.results = [];
  }

  setHypothesis() {
    this.givenHypothesis = this.ArrayOfStrings("Als ik de " + this.givenIndVar + " " +
this.givenIndVarChange + ", dan zal de " + this.givenDepVar + " " + this.givenDepVarChange +
".");
  }

  setEvidence() {
    this.givenEvidence = this.ArrayOfStrings("Toen ik de " + this.givenIndVarCheck + " " +
this.givenIndVarChangeCheck + ", ging de " + this.givenDepVarCheck + " " +
this.givenDepVarChangeCheck + ".");
  }

  getHypothesis() {
    return this.givenHypothesis;
  }

  getEvidence() {
    return this.givenEvidence;
  }

  getGivenIndVar() {
    return this.givenIndVar;
  }

  getGivenDepVar() {
    return this.givenDepVar;
  }

  getGivenDepVarChange() {
    return this.givenDepVarChange;
  }

  getGivenIndVarChange() {
    return this.givenIndVarChange;
  }
```

```
getGivenIndVarCheck() {
  return this.givenIndVarCheck;
}

getGivenDepVarCheck() {
  return this.givenDepVarCheck;
}

getGivenDepVarChangeCheck() {
  return this.givenDepVarChangeCheck;
}

getGivenIndVarChangeCheck() {
  return this.givenIndVarChangeCheck;
}

getHypothesisCheck() {
  return this.hypothesisCheck;
}

getIndVariableOptions() {
  return this.indVariableOptions;
}

getDepVariableOptions() {
  return this.depVariableOptions;
}
```

```
  getVariableOptions() {
    return this.variableOptions;
  }



  //Gets called by the hypothesis and the analyze classes
  //Changes the value of the given variables when a dropdownmenu is changed
  //Note: it would have been better to send a callback to a specific set function as an argument
when creating a dropdown menu
  setDropdownSelection(dropdownString, selectedValue) {
    switch (dropdownString) {
    case 'givenIndVar':
      this.givenIndVar = selectedValue;
      break;

    case 'givenIndVarChange':
      this.givenIndVarChange = selectedValue;
      break;

    case 'givenDepVar':
      this.givenDepVar = selectedValue;
      break;

    case 'givenDepVarChange':
      this.givenDepVarChange = selectedValue;
      break;
    case 'givenIndVarCheck':
      this.givenIndVarCheck = selectedValue;
      break;

    case 'givenIndVarChangeCheck':
      this.givenIndVarChangeCheck = selectedValue;
      break;

    case 'givenDepVarCheck':
      this.givenDepVarCheck = selectedValue;
      break;

    case 'givenDepVarChangeCheck':
      this.givenDepVarChangeCheck = selectedValue;
      break;

    case 'hypothesisCheck':
      this.hypothesisCheck = selectedValue;
      break;

    default:
      console.log('Unknown dropdown type, something is going very wrong');
      break;
    }
  }

  //Create a string of arrays depending on the input string, so it doesn't exceeds the line
  //I later learned that there are in built character width, but oh well
```

```
ArrayOfStrings(inputString) {
  const lines = [];
  const maxLineLength = map(this.textBoxWidth, 0, 1000, 0, 183);
  let currentLine = '';
  let currentLength = 0;
  let currentWord = '';
  let wordLength = 0;

  for (const charr of inputString) {
    // Determine the width of the character
    let charWidth;
    switch (charr) {
    case 'l':
    case 'i':
    case '.':
    case ',':
    case "'":
    case ':':
    case ';':
      charWidth = 1;
      break;
    case 'W':
    case 'M':
    case 'B':
    case 'O':
    case 'D':
      charWidth = 3;
      break;
```

```
case 'A':
case 'H':
case 'K':
case 'R':
case 'S':
case 'T':
case 'V':
case 'Y':
case 'C':
case 'G':
case 'P':
case 'Q':
case 'X':
case 'Z':
case '0':
case '1':
case '2':
case '3':
case '4':
case '5':
case '6':
case '7':
case '8':
case '9':
  charWidth = 2;
  break;
default:
  charWidth = 2;
```

```
}

// If it's a space, decide whether to break the line or continue
if (charr === ' ') {

  // Check if the word fits in the current line
  if (currentLength + wordLength > maxLineLength) {

    // If the word doesn't fit, push the current line and start a new one
    if (currentLine) {
      lines.push(currentLine.trim());
    }
    currentLine = currentWord + ' '; // Start the new line with the word
    currentLength = wordLength + 1; // Reset the current length

  } else {
    // If the word fits, add it to the current line
    currentLine += currentWord + ' ';
    currentLength += wordLength + 1; // Add word length plus space
  }

  // Reset currentWord and wordLength
  currentWord = '';
  wordLength = 0;
} else {
  // Otherwise, keep building the current word
  currentWord += charr;
  wordLength += charWidth;
```

```
      }
    }

    // Add the last word and line to the result
    if (currentWord) {
      // Check if the last word fits in the current line
      if (currentLength + wordLength > maxLineLength) {
        // If it doesn't fit, push the current line and start a new one
        if (currentLine) {
          lines.push(currentLine.trim());
        }
        lines.push(currentWord); // Add the remaining word as a new line
      } else {
        currentLine += currentWord;
        lines.push(currentLine.trim()); // Push the final line
      }
    } else if (currentLine) {
      lines.push(currentLine.trim()); // Push the final line if any
    }

    return lines;
  }
}

/*

  Each of the upcoming classes are child classes of the super class. Each one represents its own
kind of experiment.
```

```
  Explanations, goals, required independent variables etc. are stored here, as they are different for each experiment.

*/

class HeightTimeSim extends DifferentSimulationVariables {
  constructor(textBoxWidth) {
    super(textBoxWidth);
    this.goal = this.ArrayOfStrings('We willen kijken hoe de hoogte van de bal bepaalt hoelang het duurt voordat de bal de grond raakt op de maan.');
    this.reqIndVar = 'hoogte van de bal';
    this.reqDepVar = 'tijd voordat de bal de grond raakt';
    this.exerciseNumber = '1';
    this.simExp = this.setSimExp();
  }

  setSimExp() {
    //As the drawbox function uses array of strings, the explaination text is created by hand, as a line break (empty string in the array) is put
    //in the explanation for readability
    let simExp = super.ArrayOfStrings('Hoi! Ik ben Evan, en ik werk al heel lang met wetenschap en natuur. Vandaag wil ik jullie laten zien hoe het is om een wetenschapper te zijn. Spannend, hè? We gaan samen een experiment doen! We willen uitzoeken of de bal langer of korter onderweg is naar de grond als we hem van verschillende hoogtes laten vallen.');
    simExp.push(' ');
    simExp =  simExp.concat(super.ArrayOfStrings('Eerst gaan we bedenken wat we denken dat er gebeurt. Daarna gaan we het echt uitproberen. Onderin kun je al zien hoe ons experiment eruit gaat zien. Na het experiment bekijken we samen wat we hebben gezien. Laten we beginnen!'));
    return simExp;
  }

  getReqIndVar() {
    return this.reqIndVar;
  }

  getReqDepVar() {
    return this.reqDepVar;
  }

  getSimExp() {
    return this.simExp;
  }
}

class HeightVelocitySim extends DifferentSimulationVariables {
  constructor(textBoxWidth) {
    super(textBoxWidth);
    this.goal = this.ArrayOfStrings('We willen kijken hoe de hoogte van de bal bepaalt hoe snel de
bal gaat wanneer hij de grond raakt op de maan.');
    this.reqIndVar = 'hoogte van de bal';
    this.reqDepVar = 'snelheid als de bal de grond raakt';
    this.exerciseNumber = '2';
    this.simExp = this.setSimExp();
  }

  setSimExp() {
    //See above setSimExp
    let simExp = super.ArrayOfStrings(' Wat ben je lekker bezig. Je lijkt wel een echte
wetenschapper op deze manier. Ga zo door! ');
    simExp.push(' ');
    simExp =  simExp.concat(super.ArrayOfStrings('Voor het volgende experiment willen we niet
alleen kijken hoe lang het duurt voordat de bal de grond raakt, maar juist hoe snel de bal gaat
als hij de grond raakt. We willen ook weten of de bal sneller gaat als we hem van een hogere
plek laten vallen. Dus, laten we het experiment nog een keer doen!'));
    simExp.push(' ');
    return simExp;
  }

  getSimExp() {
    return this.simExp;
  }

  getReqIndVar() {
    return this.reqIndVar;
  }

  getReqDepVar() {
    return this.reqDepVar;
  }
}

class MassTimeSim extends DifferentSimulationVariables {
  constructor(textBoxWidth) {
    super(textBoxWidth);
    this.goal = this.ArrayOfStrings('We willen kijken hoe de massa van de bal bepaalt hoelang het duurt voordat de bal de grond raakt op de maan.');
    this.reqIndVar = 'massa van de bal';
    this.reqDepVar = 'tijd voordat de bal de grond raakt';
    this.exerciseNumber = '3';
    this.simExp = this.setSimExp();
  }

  setSimExp() {
    //See above setSimExp
    let simExp = super.ArrayOfStrings('We gaan het experiment nog 1 keer doen, maar dan kijkend hoe de massa de tijd voordat de bal de grond raakt veranderd. Net zoals bij de andere opdrachten, is dit dus op de maan. Je verwachting van wat er gebeuren is misschien dus heel anders dan wat er misschien gebeurt!');
    simExp.push(' ');
    simExp =  simExp.concat(super.ArrayOfStrings('P.S. Waarschijnlijk was het al redelijk duidelijk door de vorige experimenten, maar massa is een ander woord voor gewicht. Eigenlijk zijn ze ietsjes anders, maar daar hoef je nu niet over na te denken!'));
    return simExp;
  }

  getReqIndVar() {
    return this.reqIndVar;
  }

  getReqDepVar() {
    return this.reqDepVar;
  }
  getSimExp() {
    return this.simExp;
  }
}
class ExperimentPhase {
  constructor(drawBackgroundObjects, currentSim, adaptiveFeedback, nextPhaseMethod,
previousPhaseMethod) {
    this.drawBackgroundObjects = drawBackgroundObjects;
    this.currentSim = currentSim;
    this.adaptiveFeedback = adaptiveFeedback;
    this.nextPhaseMethod = nextPhaseMethod;
    this.previousPhaseMethod = previousPhaseMethod;

    //state of the experiment
    this.isRunning = false;

    //These are the x en y pos of the experiment, so it could be moved if needed
    this.experimentXpos = 800;
    this.experimentYpos = 550;

    //These are the xpos and ypos of the falling ball, which are conncected to the experiment
    this.xpos = this.experimentXpos+225;
    this.ypos = this.experimentYpos+275;

    //Creates all experiment options buttons
    this.weightOneButton = this.createOptionButton('1KG', this.setBallWeight.bind(this, 1),
this.experimentXpos-250, this.experimentYpos+10);
```

```
    this.weightFiveButton = this.createOptionButton('5KG', this.setBallWeight.bind(this, 5),
this.experimentXpos-180, this.experimentYpos+10);
    this.weightTenButton = this.createOptionButton('10KG', this.setBallWeight.bind(this, 10),
this.experimentXpos-110, this.experimentYpos+10);
    this.hightTenButton = this.createOptionButton('10M', this.setBallHeight.bind(this, 10),
this.experimentXpos-250, this.experimentYpos + 100);
    this.hightTwentyButton = this.createOptionButton('20M', this.setBallHeight.bind(this, 20),
this.experimentXpos-180, this.experimentYpos + 100);
    this.hightThirtyButton = this.createOptionButton('30M', this.setBallHeight.bind(this, 30),
this.experimentXpos-110, this.experimentYpos + 100);
    this.redBallButton = this.createOptionButton('Rood', this.setBallcolor.bind(this, 'rood'),
this.experimentXpos-250, this.experimentYpos + 190);
    this.blueBallButton = this.createOptionButton('Blauw', this.setBallcolor.bind(this, 'blauw'),
this.experimentXpos-180, this.experimentYpos + 190);
    this.orangeBallButton = this.createOptionButton('Oranje', this.setBallcolor.bind(this, 'oranje'),
this.experimentXpos-110, this.experimentYpos + 190);

    //Declares current selected values
    this.previousYposHeightSet = undefined;     //This one seems odd, but it is because the
height in the experiment is not actually e.g. 30, but a specific location in the canvas
    this.currentSelectedColor = undefined;
    this.currentSelectedHeight = undefined;
    this.currentSelectedWeight = undefined;

    //Initialize previous variables with their functions
    this.setBallcolor('rood');
    this.setBallHeight(10);
    this.setBallWeight(1);
    this.didOneExperiment = false;
    this.ballOntheGround = false;
    this.experimentButton = this.createExperimentButton(this.doExperimentButton.bind(this));
    this.nextButton = this.createNextButton();
    this.previousButton = this.createPreviousButton();
  }

  createNextButton() {
    return this.drawBackgroundObjects.createButton('Ga naar controle',
this.doNextButton.bind(this), 895, 1100, '300px', '60px');
  }

  createPreviousButton() {
    return this.drawBackgroundObjects.createButton('Ga terug', this.doPreviousButton.bind(this),
695, 1100, '150px', '60px');
  }

  // Main method to draw the experiment phase
  drawExperimentPhase() {
    this.drawTitle();
    this.drawExerciseBox();
    this.drawGoalBox();
    this.drawExpectedOutcomeBox();
    image(experimentImg, this.experimentXpos, this.experimentYpos);
    this.drawExperiment();
    this.drawButtonBoxes();
    this.drawResultBox();

    if (this.didOneExperiment) {
      this.nextButton.style('background', 'linear-gradient(to bottom, #FFB347, #FF8000)'); //
Gradient effect for 3D look
    }
  }

  // Draws the title for the analyze phase
  drawTitle() {
    fill(255);
    textSize(50);
    textStyle(BOLD);
    text('Onderzoeksfase: Experiment', 350, 70);
  }

  // Draws the exercise box with an explanation
  drawExerciseBox() {
    const titleExercise = 'Opdracht ' + this.currentSim.exerciseNumber + '.2';
    const explanation = this.currentSim.experimentExp;
    this.drawBackgroundObjects.drawTextBox(titleExercise, explanation, 400, 150);
  }

  // Draws the goal box
  drawGoalBox() {
    const goal = this.currentSim.goal;
    this.drawBackgroundObjects.drawTextBox('Wat we willen ontdekken:', goal, 400, 310);
  }
```

```
// Draws the expected outcome (hypothesis) box
drawExpectedOutcomeBox() {
  const hypothesis = this.currentSim.getHypothesis();
  this.drawBackgroundObjects.drawTextBox('Wat we verwachten:', hypothesis, 400, 420);
}

drawButtonBoxes() {
  //Draw boxes around the buttons
  fill(36, 106, 115);
  rect(400, this.experimentYpos, 370, 80);
  rect(400, this.experimentYpos + 90, 370, 80);
  rect(400, this.experimentYpos + 180, 370, 80);

  fill(255);
  textSize(30);
  textStyle(BOLD);
  text('Massa:', 420, this.experimentYpos+50);
  text('Hoogte:', 420, this.experimentYpos+140);
  text('Kleur:', 420, this.experimentYpos+230);
}

drawExperiment() {
  image(this.currentBallColor, this.xpos, this.ypos);
  textAlign(CENTER);
  fill(255);

  text(this.currentSelectedWeight + 'KG', this.xpos+30, this.ypos+38);
```

```
    textAlign(LEFT);
    if (this.isRunning) {
      if (this.ypos <= this.experimentYpos+375) {

        this.currentTime = Date.now();
        //the time in seconds
        let t = (this.currentTime-this.startTime)/1000;
        const startHeight = this.currentSelectedHeight == 10 ? 275 : this.currentSelectedHeight ==
20 ? 175 : 75
          this.ypos = this.experimentYpos + startHeight + 1.625*pow(t, 2)*10/2;
      } else {
        this.isRunning = false;
        this.didOneExperiment = true;
        this.ballOntheGround = true;

        this.currentSim.results.push([this.currentSelectedWeight, this.currentSelectedHeight,
this.currentSelectedColor, this.calculateTimeToDrop(), this.calculateVelocityAtDrop()]);
        this.showResultPopup();
      }
    }
    this.drawStopwatch();
  }

  showResultPopup() {
    let resultString = 'Massa: ' + this.currentSelectedWeight + ', Hoogte: ' +
this.currentSelectedHeight +
      ', Kleur: ' + this.currentSelectedColor + '.  <br>Tijd: '+ this.calculateTimeToDrop() +
      ', Snelheid: ' + this.calculateVelocityAtDrop() + '.' + ' <br><br><br> <i>De resultaten zijn
beneden opgeschreven.</i>';
    this.drawBackgroundObjects.createPopUp('Resultaten:', resultString);
  }

  drawStopwatch() {
    image(stopwatch, 1090, 850);
    fill(0);
    textSize(30);
    textStyle(BOLD);
    let timeString;
    if (this.isRunning) {

      //This parts shows the timer going up when the simulation is run
      const timeDifference = this.startTime ? this.currentTime - this.startTime : 0; // Calculate the
time difference
      const time = new Date(timeDifference); // Create a Date object from the time difference
      const seconds = String(time.getSeconds()).padStart(2, '0'); // Get the seconds and format
them to always have two digits
      const milliseconds = String(time.getMilliseconds()).substring(0, 2).padStart(2, '0'); // Get the
first two digits of the milliseconds and format them
      timeString = seconds + ':' + milliseconds // Combine the seconds and milliseconds into a
display string
    } else if (this.ballOntheGround) {

      //When the ball on the ground, the timer shows the the calculated time to drop
      //This is hardcoded, as it should not be different, even if the framerate drops
      if (this.calculateTimeToDrop() === 3.5) {
        timeString = '03:50';
      } else if (this.calculateTimeToDrop() === 4.96) {
        timeString = '04:96';
      } else if (this.calculateTimeToDrop() === 6.07) {
        timeString = '06:07';
      }
    } else {

      //When the ball is not falling or on the ground, it will default to 0
      timeString = '00:00';
    }
    text(timeString, 1138, 980); // Display the time string on the screen at position (1138, 980)
  }

  doExperimentButton() {
    if (!this.isRunning) {
      this.isRunning = true;
      this.startTime = Date.now();
    }
  }

  drawResultBox() {
    const textsize = 20;
    const xpos = 400;
    const ypos = 1250;
    const columnTitles = ['Test', 'Massa (KG)', 'Hoogte (M)', 'Kleur', 'Tijd (S)', 'Snelheid (M/S)']; //
Add "Test" column
```

```
// Calculate the length of the array (number of rows)
const LENGTHOFARRAY = this.currentSim.results.length;

// Calculate the height of the box dynamically based on array length
const rowHeight = 30; // Height per row of data
const boxHeight = 45 + rowHeight * (LENGTHOFARRAY + 1); // +1 for the title row

// Draw background box for results
fill(36, 106, 115);
rect(xpos, ypos, this.drawBackgroundObjects.textBoxWidth, boxHeight, 30);

// Draw hypothesis title
fill(255);
textSize(textsize);
textStyle(BOLD);
textAlign(CENTER);
text('Resultaten:', xpos + this.drawBackgroundObjects.textBoxWidth / 2, ypos + 30);

// Set up for drawing column titles
textSize(20);
textStyle(BOLD);

// Define the column widths for each of the 6 columns (including "Test")
const colWidth = this.drawBackgroundObjects.textBoxWidth / 6;

// Draw the column titles, including "Test" as the first one
for (let i = 0; i < columnTitles.length; i++) {
  text(columnTitles[i], xpos + colWidth * i + 50, ypos + 60); // Adjusted position for title
```

```
    }

    // Set text to normal style for the results
    textStyle(NORMAL);

    // Draw each element in its respective column
    for (let i = 0; i < LENGTHOFARRAY; i++) {
      const resultRow = this.currentSim.results[i];

      // Draw the test number in the first column (starting at #1)
      text('#' + (i+1), xpos + 50, ypos + 90 + (i * rowHeight)); // First column for test number

      // Draw the rest of the results in their respective columns
      for (let j = 0; j < resultRow.length; j++) {
        text(resultRow[j], xpos + colWidth * (j + 1) + 50, ypos + 90 + (i * rowHeight)); // Adjust for
the additional column
      }
    }

    textAlign(LEFT);
  }



  calculateTimeToDrop() {
    return this.roundToTwoDecimals(Math.sqrt(this.currentSelectedHeight*2 / 1.625));
  }

  calculateVelocityAtDrop() {
```

```
  return this.roundToTwoDecimals(1.625 * this.calculateTimeToDrop());
}

roundToTwoDecimals(value) {
  return Math.floor(value * 100) / 100;
}



createExperimentButton(callback) {
  const experimentButton = createButton('TEST'); // Empty title as it holds an image
  experimentButton.position(430, 850);
  experimentButton.mousePressed(callback);

  const backgroundGradient = 'linear-gradient(to bottom, #FFB347, #FF8000)';
  experimentButton.style('background', backgroundGradient);
  this.styleButton(experimentButton, '310px', '50px', '#FFFFFF', '30px');
  this.drawBackgroundObjects.addButtonHoverEffect(experimentButton);

  return experimentButton;
}

// Creates a square button with an image that triggers a callback when clicked
createOptionButton(text, callback, xpos, ypos) {
  const buttonSize = '60px'; // Size of the square button
  const OptionButton = createButton(text); // Empty title as it holds an image
  OptionButton.position(xpos, ypos);
  OptionButton.mousePressed(callback);
```

```
    this.styleButton(OptionButton, buttonSize, buttonSize, '#00000', '14px');
    this.drawBackgroundObjects.addButtonHoverEffect(OptionButton);

    return OptionButton;
  }

  // Helper method to apply styles to the button
  styleButton(button, ButtonWidth, buttonHeight, textColor, textSize) {
    button.style('width', ButtonWidth);
    button.style('height', buttonHeight);
    button.style('background-color', '#FFFFFF');
    button.style('border', '2px solid black');
    button.style('border-radius', '10px');
    button.style('box-shadow', '0px 4px 8px rgba(0, 0, 0, 0.3)');
    button.style('cursor', 'pointer');
    button.style('font-weight', 'bold'); // Makes the text bold
    button.style('color', textColor); //
    button.style('font-size', textSize);
  }

  //calls the callback function of the main class to go to next phase button when
didOneExperiment is true, and no feedback needs to be given
  doNextButton() {
    if (this.didOneExperiment && (!adaptive ||
      this.adaptiveFeedback.giveAdaptiveFeedbackExperimentPhase(
      this.currentSim.results, this.currentSim.reqIndVar,
this.currentSim.indVariableOptions.length))) {
      //Removes all experiment option buttons
      this.hideAllDomObjects();

      // Proceed to the next phase
      this.nextPhaseMethod();
    }
  }

  doPreviousButton() {
    this.hideAllDomObjects();
    this.previousPhaseMethod()
  }

  hideAllDomObjects() {
    this.weightOneButton.hide();
    this.weightFiveButton.hide();
    this.weightTenButton.hide();
    this.hightTenButton.hide();
    this.hightTwentyButton.hide();
    this.hightThirtyButton.hide();
    this.redBallButton.hide();
    this.blueBallButton.hide();
    this.orangeBallButton.hide();

    this.nextButton.hide();
    this.previousButton?.hide();
    this.experimentButton.hide();
  }
```

```
unhideAllDomObjects() {
  this.weightOneButton.show();
  this.weightFiveButton.show();
  this.weightTenButton.show();
  this.hightTenButton.show();
  this.hightTwentyButton.show();
  this.hightThirtyButton.show();
  this.redBallButton.show();
  this.blueBallButton.show();
  this.orangeBallButton.show();

  this.nextButton.show();
  this.previousButton?.show();
  this.experimentButton.show();
}

setBallcolor(colorString) {
  if (this.isRunning === false) {
    this.currentSelectedColor = colorString;

    this.makeUnselectedButton(this.orangeBallButton);
    this.makeUnselectedButton(this.blueBallButton);
    this.makeUnselectedButton(this.redBallButton);

    //Makes the selected button orange (selected) AND change the image of the ball
    if (colorString === 'oranje') {
      this.currentBallColor = ballImgOrange;
      this.makeSelectedButton(this.orangeBallButton);
    } else if (colorString === 'rood') {
      this.currentBallColor = ballImgRed;
      this.makeSelectedButton(this.redBallButton);
    } else if (colorString === 'blauw') {
      this.currentBallColor = ballImgBlue;
      this.makeSelectedButton(this.blueBallButton);
    }

    //Change the height to the previous selected height
    this.ypos = this.previousYposHeightSet;
    this.ballOntheGround = false;
  }
}

setBallHeight(num) {
  if (this.isRunning === false) {
    this.currentSelectedHeight = num;

    //Makes all buttons weight buttons grey (unselected)
    this.makeUnselectedButton(this.hightTenButton);
    this.makeUnselectedButton(this.hightTwentyButton);
    this.makeUnselectedButton(this.hightThirtyButton);

    //Makes the selected button orange (selected) AND change the height (ypos) of the ball
    if (num === 10) {
      this.ypos = this.experimentYpos+275;
      this.makeSelectedButton(this.hightTenButton);
    } else if (num === 20) {
      this.ypos = this.experimentYpos+175;
      this.makeSelectedButton(this.hightTwentyButton);
    } else if (num === 30) {
      this.ypos = this.experimentYpos+75;
      this.makeSelectedButton(this.hightThirtyButton);
    }
    //Save the height so it can be set back to this state when changing anohter variable
    this.previousYposHeightSet = this.ypos;
    this.ballOntheGround = false;
  }
}

setBallWeight(num) {
  if (this.isRunning === false) {
    this.currentSelectedWeight = num;

    //Makes all buttons weight buttons grey (unselected)
    this.makeUnselectedButton(this.weightOneButton);
    this.makeUnselectedButton(this.weightFiveButton);
    this.makeUnselectedButton(this.weightTenButton);

    //Makes the selected button orange (selected)
    if (num === 1) {
      this.makeSelectedButton(this.weightOneButton);
    } else if (num === 5) {
      this.makeSelectedButton(this.weightFiveButton);
    } else if (num === 10) {
      this.makeSelectedButton(this.weightTenButton);
    }


    //Change the height to the previous selected height
    this.ypos = this.previousYposHeightSet;
    this.ballOntheGround = false;
  }
}

//changes the css style of the button that is selected
makeSelectedButton(button) {
  button.style('color', '#FFFFFF'); // Make text white
  button.style('font-size', '16px'); // Adjust text size
  const backgroundGradient = 'linear-gradient(to bottom, #FFB347, #FF8000)';
  button.style('background', backgroundGradient); //Set the background to an orange gradient
}

//changes the css style back to normal of the buttons that are unselected
makeUnselectedButton(button) {
  button.style('color', '#000000'); // Make text black
  button.style('font-size', '14px'); // Adjust text size
  button.style('background', '#FFFFFF'); // Reset background to solid white
}
}
class ExplainPhase {
constructor(drawBackgroundObjects, currentSim, nextPhaseMethod) {
  this.drawBackgroundObjects = drawBackgroundObjects;
  this.currentSim = currentSim;
  this.nextPhaseMethod = nextPhaseMethod;
  this.nextButton = this.createNextButton();
  this.nextButton.style('background', 'linear-gradient(to bottom, #FFB347, #FF8000)'); //
Gradient effect for 3D look
}

createNextButton() {
  return this.drawBackgroundObjects.createButton('Aan de slag', this.doNextButton.bind(this),
895, 450, '300px', '60px');
}

drawExplainPhase() {
  this.drawTitle();
  this.drawExplanation();
  image(scientistWithNameImg, 160, 220);
  explanationVideo.position(400, 475);
  fill(36, 106, 115);
  rect(400-20, 475-20, 440, 280, 30);
}

drawTitle() {
  fill(255);
  textSize(50);
  textStyle(BOLD);
  text('Onderzoek ' + this.currentSim.exerciseNumber, 350, 70);
}

drawExplanation() {
  const explanation = this.currentSim.getSimExp();
  const titleExercise = 'Opdracht ' + this.currentSim.exerciseNumber;
  this.drawBackgroundObjects.drawTextBox(titleExercise, explanation, 400, 150);
}

doNextButton() {
  this.nextButton?.hide();
  explanationVideo.hide();
  this.nextPhaseMethod();
}

unhideAllDomObjects(){
  this.nextButton.show();
  explanationVideo.show();
}
}
class HypothesisPhase {
constructor(drawBackgroundObjects, currentSim, adaptiveFeedback, nextPhaseMethod,
previousPhaseMethod) {
  this.drawBackgroundObjects = drawBackgroundObjects;
  this.currentSim = currentSim;
  this.adaptiveFeedback = adaptiveFeedback;
  this.nextPhaseMethod = nextPhaseMethod;
  this.previousPhaseMethod = previousPhaseMethod;

  // Define options for dropdown menus
  this.variableOptions = this.currentSim.getVariableOptions();
  this.dependentChanges = ['meer laat worden', 'minder laat worden', 'hetzelfde laat'];
  this.independentChanges = ['meer worden', 'minder worden', 'hetzelfde blijven'];

  // Create dropdowns
  this.createDropdowns();

  this.allSelected = false;
  this.nextButton = this.createNextButton();
  this.previousButton = this.createPreviousButton();
}

createDropdowns() {
  // Create independent and dependent variable dropdowns
  this.dropdownIndependentVar =
this.drawBackgroundObjects.createDropdown(this.variableOptions, 490, 505, 'givenIndVar',
this.currentSim);
  this.dropdownIndependentChange =
this.drawBackgroundObjects.createDropdown(this.dependentChanges, 770, 505,
'givenIndVarChange', this.currentSim);
  this.dropdownDependentVar =
this.drawBackgroundObjects.createDropdown(this.variableOptions, 510, 545, 'givenDepVar',
this.currentSim);
  this.dropdownDependentChange =
this.drawBackgroundObjects.createDropdown(this.independentChanges, 790, 545,
'givenDepVarChange', this.currentSim);
}

createNextButton() {
  return this.drawBackgroundObjects.createButton('Ga naar experiment',
this.doNextButton.bind(this), 895, 650, '300px', '60px');
}

createPreviousButton() {
  return this.drawBackgroundObjects.createButton('Ga terug', this.doPreviousButton.bind(this),
695, 650, '150px', '60px');
}

// Main method to draw the hypothesis phase
drawHypothesisPhase() {
  this.drawTitle();
  this.drawExplanation();
  this.drawGoal();
  this.drawHypothesisBox(400, 460);

  this.updateAllSelected();
  this.updateNextButtonStyle();
}

drawTitle() {
  fill(255);
  textSize(50);
  textStyle(BOLD);
  text('Onderzoeksfase: Verwachting', 350, 70);
}

// Draws the exercise box with an explanation
```

```
drawExplanation() {
  const explanation = this.currentSim.hypothesisExp;
  const titleExercise = 'Opdracht ' + this.currentSim.exerciseNumber + '.1';
  this.drawBackgroundObjects.drawTextBox(titleExercise, explanation, 400, 150);
}

drawGoal() {
  const goal = this.currentSim.goal;
  this.drawBackgroundObjects.drawTextBox('Wat we willen ontdekken:', goal, 400, 330);
}

drawHypothesisBox(xpos, ypos) {
  const textsize = 20;

  // Draw hypothesis background box
  fill(255);
  rect(xpos, ypos, this.drawBackgroundObjects.textBoxWidth, 140);

  // Draw hypothesis title
  fill(0);
  textSize(textsize);
  textStyle(BOLD);
  text('Wat we verwachten:', xpos + 10, ypos + 30);

  // Draw hypothesis structure text
  textStyle(NORMAL);
  text('Als ik de', xpos + 10, ypos + 65);
  text(',', xpos + 650, ypos + 65);
```

```
    text('dan zal de ', xpos + 10, ypos + 105);
    text('.', xpos + 670, ypos + 105);
  }

  updateAllSelected() {
    // Check if all dropdowns have a selection to enable/disable the next button
    this.allSelected = this.currentSim.getGivenIndVar() &&
      this.currentSim.getGivenDepVar() &&
      this.currentSim.getGivenIndVarChange() &&
      this.currentSim.getGivenDepVarChange();
  }

  updateNextButtonStyle() {
    // Update next button style when all selections are made
    if (this.allSelected) {
      this.nextButton.style('background', 'linear-gradient(to bottom, #FFB347, #FF8000)'); //
Gradient orange effect
    }
  }

  //calls the callback function of the main class to go to next phase when allSelected is true, and
no feedback needs to be given
  doNextButton() {

    this.currentSim.setHypothesis();

    if (this.allSelected) {
      const goal = this.currentSim.goal;
      const reqIndVar = this.currentSim.getReqIndVar();
      const reqDepVar = this.currentSim.getReqDepVar();
      const givenIndVar = this.currentSim.getGivenIndVar();
      const givenDepVar = this.currentSim.getGivenDepVar();
      const indVariableOptions = this.currentSim.getIndVariableOptions();
      const depVariableOptions = this.currentSim.getDepVariableOptions();
      const dropdownIndependentVar = this.dropdownIndependentVar;
      const dropdownDependentVar = this.dropdownDependentVar;

      if (!adaptive || this.adaptiveFeedback.giveAdaptiveFeedbackHypothesisPhase(goal, reqIndVar,
reqDepVar, givenIndVar, givenDepVar, indVariableOptions, depVariableOptions,
dropdownIndependentVar, dropdownDependentVar)) {
        // Remove dropdowns from DOM
        this.hideAllDomObjects();

        // Proceed to the next phase
        this.nextPhaseMethod();
      }
    }
  }

  doPreviousButton() {
    this.hideAllDomObjects();
    this.previousPhaseMethod();
  }

  hideAllDomObjects() {
    this.dropdownDependentVar.hide();
    this.dropdownDependentChange.hide();
    this.dropdownIndependentVar.hide();
    this.dropdownIndependentChange.hide();
    this.nextButton.hide();
    this.previousButton.hide();
  }

  unhideAllDomObjects() {
    this.dropdownDependentVar.show();
    this.dropdownDependentChange.show();
    this.dropdownIndependentVar.show();
    this.dropdownIndependentChange.show();
    this.nextButton.show();
    this.previousButton.show();
  }
}
class ProofPhase {
  constructor(drawBackgroundObjects, currentSim, adaptiveFeedback, nextPhaseMethod,
previousPhaseMethod) {
    this.drawBackgroundObjects = drawBackgroundObjects;
    this.currentSim = currentSim;
    this.adaptiveFeedback = adaptiveFeedback;
    this.nextPhaseMethod = nextPhaseMethod;
    this.previousPhaseMethod = previousPhaseMethod;

    this.ProofBoxXpos = 400;
    this.ProofBoxYpos = 450;
    // Create an array to store the checkboxes so we can reference them later if needed
    this.checkboxes = [];
    this.createCheckBoxes();


    this.nextButton = this.createNextButton();
    this.nextButtonXpos = 895;
    this.nextButtonYpos = 590;
    this.previousButton = this.createPreviousButton();
    this.previousButtonXpos = 695;
    this.previousButtonYpos = 590;
  }

  createNextButton() {
    return this.drawBackgroundObjects.createButton('Volgende', this.doNextButton.bind(this),
this.nextButtonXpos, this.nextButtonYpos, '300px', '60px');
  }

  createPreviousButton() {
    return this.drawBackgroundObjects.createButton('Ga terug', this.doPreviousButton.bind(this),
this.previousButtonXpos, this.previousButtonYpos, '150px', '60px');
  }

  createCheckBoxes() {
    // Calculate the height of the box dynamically based on array length
    const rowHeight = 30; // Height per row of data
    const boxHeight = 40 + rowHeight * (this.currentSim.results.length + 1); // +1 for the title
row
```

```
    // Define the xpos and ypos of the checkboxes
    const xpos = this.ProofBoxXpos + 30; // Move 5 pixels to the left
    const ypos = this.ProofBoxYpos + 65; // Move 20 pixels down

    // Draw each checkbox and place it inside the canvas
    // NOTE: It lopes from the length of checkboxes. If there are already some checkboxes made,
it should not make new one on top of it
    //       This makes sure you can add results, even though the proof class was already created
    for (let i = this.checkboxes.length; i < this.currentSim.results.length; i++) {
      let checkbox = createCheckbox('', false);  // Use p5.js method to create a checkbox
      checkbox.position(xpos, ypos + (i * rowHeight));  // Position checkbox inside the canvas

      // Make the checkbox twice as big using CSS transform
      checkbox.style('transform', 'scale(2)');
      checkbox.style('transform-origin', 'left top');  // Ensure it scales from the top-left corner

      this.checkboxes.push(checkbox); // Store reference to the checkbox
    }
  }

  // Main method to draw the proof phase
  drawProofPhase() {
    this.drawTitle();
    this.drawExerciseBox();
    this.drawOutcomeBox();
    this.drawProofBox();
    this.updateButtons();
  }
  // Draws the title for the analyze phase
  drawTitle() {
    fill(255);
    textSize(40);
    textStyle(BOLD);
    text('Onderzoeksfase: Verwachting bewijzen', 350, 70);
  }

  // Draws the exercise box with an explanation
  drawExerciseBox() {
    const titleExercise = 'Opdracht ' + this.currentSim.exerciseNumber + '.4';
    const explanation = this.currentSim.proofExp;
    this.drawBackgroundObjects.drawTextBox(titleExercise, explanation, 400, 150);
  }

  drawOutcomeBox() {
    const hypothesis = this.currentSim.getEvidence();
    this.drawBackgroundObjects.drawTextBox('Wat er gebeurde:', hypothesis, 400, 310);
  }

  drawProofBox() {
    const textsize = 20;
    const xpos = this.ProofBoxXpos; //
    const ypos = this.ProofBoxYpos; //Height of the box
    const columnTitles = ['Test', 'Massa (KG)', 'Hoogte (M)', 'Kleur', 'Tijd (S)', 'Snelheid (M/S)']; //
Add "Test" column
```

```
  // Calculate the height of the box dynamically based on array length
  const rowHeight = 30; // Height per row of data
  const boxHeight = 45 + rowHeight * (this.currentSim.results.length + 1); // +1 for the title
row

  // Draw background box for results
  fill(36, 106, 115);
  rect(xpos, ypos, this.drawBackgroundObjects.textBoxWidth, boxHeight, 30);

  // Draw hypothesis title
  fill(255);
  textSize(textsize);
  textStyle(BOLD);
  textAlign(CENTER);
  text('Resultaten:', xpos + this.drawBackgroundObjects.textBoxWidth / 2, ypos + 30);

  // Set up for drawing column titles
  textSize(20);
  textStyle(BOLD);

  // Define the column widths for each of the 6 columns (including "Test")
  const colWidth = this.drawBackgroundObjects.textBoxWidth / 6;

  // Draw the column titles, including "Test" as the first one
  for (let i = 0; i < columnTitles.length; i++) {
    text(columnTitles[i], xpos + colWidth * i + 50, ypos + 60); // Adjusted position for title
  }
```

```
    // Set text to normal style for the results
    textStyle(NORMAL);

    // Draw each element in its respective column
    for (let i = 0; i < this.currentSim.results.length; i++) {
      const resultRow = this.currentSim.results[i];
      for (let j = 0; j < resultRow.length; j++) {
        text(resultRow[j], xpos + colWidth * (j + 1) + 50, ypos + 90 + (i * rowHeight)); // Adjust for
the additional column
      }
    }

    textAlign(LEFT);
  }

  doPreviousButton() {
    this.hideAllDomObjects();
    this.previousPhaseMethod();
  }

  //calls the callback function of the main class to go to next phase button when
checkOneIsChecked is true, and no feedback needs to be given
  doNextButton() {
    if (this.checkOneIsChecked() && (!adaptive ||
this.adaptiveFeedback.giveAdaptiveFeedbackProofPhase(
      this.currentSim.results, this.currentSim.getReqIndVar(),
this.currentSim.getIndVariableOptions(), this.checkboxes))) {
      this.hideAllDomObjects();
      this.nextPhaseMethod();
    }
  }

  hideAllDomObjects() {
    //delete all checkboxes
    for (let i = 0; i < this.checkboxes.length; i++) {
      this.checkboxes[i].hide();
    }

    //delete the buttons
    this.nextButton.hide();
    this.previousButton.hide();
  }

  unhideAllDomObjects() {
    for (let i = 0; i < this.checkboxes.length; i++) {
      this.checkboxes[i].show();
    }

    this.createCheckBoxes();



    //delete the buttons
    this.nextButton.show();
    this.previousButton.show();
  }
```

```
  //The postions and coloring of the button is done over and over, as it can be that there are
more results added by going back or that all boxes get unselected
  updateButtons() {
    //Update the color of the nextbutton
    if (this.checkOneIsChecked()) {
      this.nextButton.style('background', 'linear-gradient(to bottom, #FFB347, #FF8000)'); //Make
orange
    } else {
      this.nextButton.style('background', 'linear-gradient(to bottom, #A9A9A9, #696969)'); //Make
gray
    }

    //Update the position of the previousButton and the nextButton based on the length of the
results
    this.nextButton.position(this.nextButtonXpos, this.nextButtonYpos  +
(30*this.currentSim.results.length));
    this.previousButton.position(this.previousButtonXpos, this.previousButtonYpos  +
(30*this.currentSim.results.length));
  }

  // Method to check if atleast one of the checkboxes is checked, if not return false
  checkOneIsChecked() {
    for (let i = 0; i < this.checkboxes.length; i++) {
      let isChecked = this.checkboxes[i].checked();  // Check if the checkbox is checked
      if (isChecked) {
        return true;
      }
    }
    return false;
  }
}
class DrawBackgroundObjects {
  constructor(tempTextBoxWidth) {
    this.textBoxWidth = tempTextBoxWidth;
  }

  // Draws the background
  drawBackground(exerciseNumber) {
    this.drawBackgroundColor();
    this.drawSidePanels();
    this.drawExerciseNumber(exerciseNumber);
    this.writeVersion();
  }

  // Draws a solid color for the background
  drawBackgroundColor() {
    fill(0, 207, 193);
    rect(0, 0, width, height);
  }

  // Draws the side panels on the canvas
  drawSidePanels() {
    fill(36, 106, 115);
    rect(0, 0, width, 100);
    rect(0, 0, 300, height);
  }
```

```
//Display the current exercise number (e.g. 1/3), depending on the currentSim
drawExerciseNumber(exerciseNumber) {
  fill(255);
  textSize(50);
  textStyle(BOLD);
  text(exerciseNumber + '/3', width-100, 70);
}

//This is purely for during testing, so I know which version a student has
writeVersion(){
  fill(0);
  textSize(10);
  textStyle(BOLD);
  text('adaptive = ' + adaptive, 10, height - 10);
}

// Draws a text box with a title and multiple lines of text
drawTextBox(title, stringArray, xpos, ypos) {
  const textSizeValue = 20;
  const boxHeight = textSizeValue * 2 + stringArray.length * (textSizeValue + 5);

  this.drawTextBoxBackground(xpos, ypos, boxHeight);
  this.drawTitle(title, xpos, ypos, textSizeValue);
  this.drawTextLines(stringArray, xpos, ypos, textSizeValue);
}

// Draw the background of the text box
```

```
drawTextBoxBackground(xpos, ypos, height) {
  fill(255);
  rect(xpos, ypos, this.textBoxWidth, height);
}

// Draws the title in the text box
drawTitle(title, xpos, ypos, textSizeValue) {
  fill(0);
  textSize(textSizeValue);
  textStyle(BOLD);
  text(title, xpos + 10, ypos + 30);
}

// Draws each line of text in the text box
drawTextLines(stringArray, xpos, ypos, textSizeValue) {
  textStyle(NORMAL);

  stringArray.forEach((line, index) => {
    text(line, xpos + 10, ypos + 55 + index * (textSizeValue + 5));
  }
  );
}

// Creates a dropdown with options at a specific position and associates it with a callback
createDropdown(options, xpos, ypos, dropdownString, currentSim) {
  const dropdown = createSelect();
  this.initializeDropdown(dropdown, options, dropdownString, xpos, ypos, currentSim);
  return dropdown;
```

```
}

// Helper method to initialize a dropdown with options and styles
initializeDropdown(dropdown, options, dropdownString, xpos, ypos, currentSim) {
  // Set dropdown position
  dropdown.position(xpos, ypos);

  //Adds a default option but makes sure it is disabled
  const defaultOption = '-- Maak een keuze --';
  dropdown.option(defaultOption);
  dropdown.selected(defaultOption);
  dropdown.elt[0].disabled = true;

  options.forEach(option => dropdown.option(option)); // Add the provided options

  dropdown.changed(() => {
    currentSim.setDropdownSelection(dropdownString, dropdown.value()); // Callback
    dropdown.style('border', '1px solid black'); // Change border color to black
  }
  );
  this.styleDropdown(dropdown); // Apply custom styles
}

// Apply custom styles to the dropdown
styleDropdown(dropdown) {
  dropdown.style('font-size', '16px');
  dropdown.style('width', '270px');
  dropdown.style('height', '30px');
```

```
}

// Creates a button that triggers a callback when clicked
createButton(title, callback, xpos, ypos, buttonWidth, buttonHeight) {
  const button = createButton(title);
  button.position(xpos, ypos);
  button.mousePressed(callback);

  this.styleButton(button, buttonWidth, buttonHeight);
  this.addButtonHoverEffect(button);

  return button;
}

// Helper method to apply styles to the button
styleButton(button, buttonWidth, buttonHeight) {
  const buttonColor = '#808080';
  const backgroundGradient = 'linear-gradient(to bottom, #A9A9A9, #696969)';

  button.style('font-size', '24px');
  button.style('font-weight', 'bold');
  button.style('width', buttonWidth);
  button.style('height', buttonHeight);
  button.style('background', backgroundGradient);
  button.style('color', '#FFFFFF');
  button.style('border', '2px solid black');
  button.style('border-radius', '15px');
  button.style('box-shadow', '0px 6px 12px rgba(0, 0, 0, 0.3)');
```

```
    button.style('cursor', 'pointer');
  }

  // Adds a hover effect to the button for interactivity
  addButtonHoverEffect(button) {
    button.mouseOver(() => {
      button.style('box-shadow', '0px 8px 14px rgba(0, 0, 0, 0.4)'); // Darker shadow on hover
    }
    );

    button.mouseOut(() => {
      button.style('box-shadow', '0px 6px 12px rgba(0, 0, 0, 0.3)'); // Original shadow
    }
    );
  }

  createPopUp(title, text) {
    // Show the modal
    const modal = document.getElementById("myModal");
    const modalTitle = document.getElementById("modalTitle"); // Ensure this variable is
declared
    const modalMessage = document.getElementById("modalMessage");
    const closeButton = document.querySelector(".close-button");
    const okButton = document.getElementById("modalOkButton");

    // Set the modal message
    modalTitle.innerText = title;
    modalMessage.innerHTML = text;
    // Display the modal
    modal.style.display = "block";

    // Close the modal when the user clicks on <span> (x)
    closeButton.onclick = function() {
      modal.style.display = "none";
    };

    // Close the modal when the user clicks on the OK button
    okButton.onclick = function() {
      modal.style.display = "none";
    };

    // Close the modal when the user clicks anywhere outside of the modal
    window.onclick = function(event) {
      if (event.target === modal) {
        modal.style.display = "none";
      }
    };
  }
}
<html>
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <script language="javascript">
  const adaptive = true;
  </script>
  <script language="javascript" type="text/javascript" src="libraries/p5.min.js"></script>
  <script language="javascript" type="text/javascript" src="AdaptiveFeedback.js"></script>
  <script language="javascript" type="text/javascript" src="Anaylyze.js"></script>
  <script language="javascript" type="text/javascript"
src="DifferentSimulationVariables.js"></script>
  <script language="javascript" type="text/javascript" src="Experiment.js"></script>
  <script language="javascript" type="text/javascript" src="ExplainPhase.js"></script>
  <script language="javascript" type="text/javascript" src="Hypothesis.js"></script>
  <script language="javascript" type="text/javascript" src="Proof.js"></script>
  <script language="javascript" type="text/javascript" src="drawBackground.js"></script>
  <script language="javascript" type="text/javascript" src="MasterThesis.js"></script>
  <style>
    body { padding: 0; margin: 0; }

    /* Modal styles */
    .modal-content {
  background-color: #fefefe;
  margin: 5% auto;
  padding: 20px;
  padding-bottom: 60px;
  border: 1px solid #888;
  width: 60%;
  max-width: 800px;
  border-radius: 10px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);
  font-size: 20px;
  font-family: Arial, sans-serif;
  display: flex;
  justify-content: space-between;
  align-items: flex-start;
  position: relative;
  overflow-y: auto; /* Enable scrolling if the content exceeds the window height */
}

.modal {
  display: none;
  position: fixed;
  z-index: 1;
  left: 0;
  top: 0;
  width: 100%;
  height: 100%;
  overflow: auto;
  background-color: rgba(0,0,0,0.7);
}

.modal-image {
  max-width: 300px;
  height: auto;
  border-radius: 10px;
  object-fit: contain;
}

#modalOkButton {
  padding: 10px 20px;
  background-color: #FFB347;
  border: none;
  border-radius: 5px;
  color: white;
  cursor: pointer;
  position: absolute;
  bottom: 10px;
  left: 20px;
  margin-top: 40px;
}

#modalOkButton:hover {
  background-color: #FF8000; /* Darker shade on hover */
}

.close-button {
  position: absolute;
  top: 10px;
  right: 10px;
  color: #aaa;
  font-size: 28px;
  font-weight: bold;
  z-index: 10;
  cursor: pointer;
}

.close-button:hover,
.close-button:focus {
  color: black;
  text-decoration: none;
  cursor: pointer;
}
  </style>
</head>

<body>

  <!-- Modal Structure -->
  <div id="myModal" class="modal">
    <div class="modal-content">
      <div class="text-content">
        <span class="close-button">&times;</span>
        <h2 id="modalTitle">This is the assigned title of the pop-up</h2>
        <p id="modalMessage">This is the assigned message of the pop-up!</p>
      </div>
      <!-- Image on the right -->
      <img src="clipartProfessor.png" alt="Popup Image" class="modal-image">
      <!-- Move the OK button out of the text-content div -->
      <button id="modalOkButton">OK</button>
    </div>
  </div>

</body>
</html>
<html>
```

```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <script language="javascript">
  const adaptive = false;
  </script>

  <script language="javascript" type="text/javascript" src="libraries/p5.min.js"></script>
  <script language="javascript" type="text/javascript" src="AdaptiveFeedback.js"></script>
  <script language="javascript" type="text/javascript" src="Anaylyze.js"></script>
  <script language="javascript" type="text/javascript"
src="DifferentSimulationVariables.js"></script>
  <script language="javascript" type="text/javascript" src="Experiment.js"></script>
  <script language="javascript" type="text/javascript" src="ExplainPhase.js"></script>
  <script language="javascript" type="text/javascript" src="Hypothesis.js"></script>
  <script language="javascript" type="text/javascript" src="Proof.js"></script>
  <script language="javascript" type="text/javascript" src="drawBackground.js"></script>
  <script language="javascript" type="text/javascript" src="MasterThesis.js"></script>

  <style>
    body { padding: 0; margin: 0; }

    /* Modal styles */
    .modal-content {
  background-color: #fefefe;
  margin: 5% auto;
  padding: 20px;
  padding-bottom: 60px;
  border: 1px solid #888;
  width: 60%;
  max-width: 800px;
  border-radius: 10px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);
  font-size: 20px;
  font-family: Arial, sans-serif;
  display: flex;
  justify-content: space-between;
  align-items: flex-start;
  position: relative;
  overflow-y: auto;
}

.modal {
  display: none;
  position: fixed;
  z-index: 1;
  left: 0;
  top: 0;
  width: 100%;
  height: 100%;
  overflow: auto;
  background-color: rgba(0,0,0,0.7);
}

.modal-image {
  max-width: 300px;
  height: auto;
  border-radius: 10px;
  object-fit: contain;
}

#modalOkButton {
  padding: 10px 20px;
  background-color: #FFB347;
  border: none;
  border-radius: 5px;
  color: white;
  cursor: pointer;
  position: absolute;
  bottom: 10px;
  left: 20px;
  margin-top: 40px;
}

#modalOkButton:hover {
  background-color: #FF8000; /* Darker shade on hover */
}

.close-button {
  position: absolute;
  top: 10px;
  right: 10px;
  color: #aaa;
  font-size: 28px;
  font-weight: bold;
  z-index: 10;
  cursor: pointer;
}

.close-button:hover,
.close-button:focus {
  color: black;
  text-decoration: none;
  cursor: pointer;
}
  </style>
</head>

<body>

  <!-- Modal Structure -->
  <div id="myModal" class="modal">
    <div class="modal-content">
      <div class="text-content">
        <span class="close-button">&times;</span>
        <h2 id="modalTitle">This is the assigned title of the pop-up</h2>
        <p id="modalMessage">This is the assigned message of the pop-up!</p>
      </div>
      <!-- Image on the right -->
      <img src="clipartProfessor.png" alt="Popup Image" class="modal-image">
      <!-- Move the OK button out of the text-content div -->
      <button id="modalOkButton">OK</button>
    </div>
  </div>

</body>
</html>
```

Appendix B: Overview adaptive feedback

The specific feedback given to the user is determined by their input and the current stage of the inquiry cycle. The following error messages are examples of error messages that are displayed during the first cycle of inquiry, where the independent variable is the height of the ball, and the dependent variable is the time at which the ball hits the ground. Additionally, the variable components of the error messages that were different in each repetition of inquiry cycle are indicated in *italics*.

Table 5:
Extensive overview adaptive feedback

Phase	**Check**	**Feedback**
Hypothesis	Ensures that an independent variable is placed in the first dropdown menu when forming the hypothesis	- An experiment always needs an option that can be selected in advance. For something like the speed when the ball hits the ground, the value can only be measured at the end. Therefore, try to select an option in the first field that can be adjusted before conducting the experiment. - Let me clarify. Options you could change are mass of the ball, height of the ball, colour of the ball. Select one of these options and make sure this is what we want to investigate. - Hmmm, let me help you a bit more with this step. The correct choice for the first field is: *height of the ball*.
	Ensures that a dependent variable is placed in the third dropdown menu when forming the hypothesis	- An experiment always needs an option that you measure at the end. For something like the *colour of the ball*, it is an option that you change at the beginning. Therefore, try to select an option in the third field that you can measure. - Let me clarify. Options you could measure are: *time when the ball hits the ground* and *speed when the ball hits the ground*. Select one of these options and make sure this is what we want to investigate.

Phase	Check	Feedback
		- Hmmm, let me help you a bit more with this step. The correct choice for the third field is: *time when the ball hits the ground*.
	Ensures that the chosen dependent variable aligns with what was specified in the goal	- The choice in the first field does not align with what we want to investigate. So, pay attention to goal: We want to know how the *height of the ball* influences the *time when the ball hits the ground*. - In the first field it says *the colour of the ball*. Unfortunately, this is not what we want to investigate. The correct answer is: *the height of the ball*.
	Ensures that the chosen independent variable aligns with what was specified in the goal	- The choice in the third field does not align with what we want to investigate. So, pay attention to goal: We want to know how the *height of the ball* influences the *time when the ball hits the ground*. - In the third field it says *speed when the ball hits the ground*. Unfortunately, this is not what we want to investigate. The correct answer is: *time when the ball hits the ground*.
Experiment	Ensures that the student has conducted multiple experiments rather than just one	- In an experiment, you want to change something each time. That way, you discover more! If you drop the same ball every time, you learn little. But by using a heavier ball or changing the height, you can see what happens.
	Ensures that at least two distinct tests were performed where the independent variable was changed	- Look closely at what we want to find out and what we expect. What should we change in the experiment to figure it out? - We want to see what effect the *height of the ball* has. So, change the *height of the ball* in the experiment!
	Ensures the student did a controlled experiment	- You’re on the right track. You have changed the *height of the ball*. However, you also have changed something else when doing so. Try to only change one variable at a time! - Not quite correct. Let me explain it in a bit more detail. First, click on the test button. Now, only change the *height of the ball*. Click again on the test button. Now you have two experiments that are exactly the same, besides

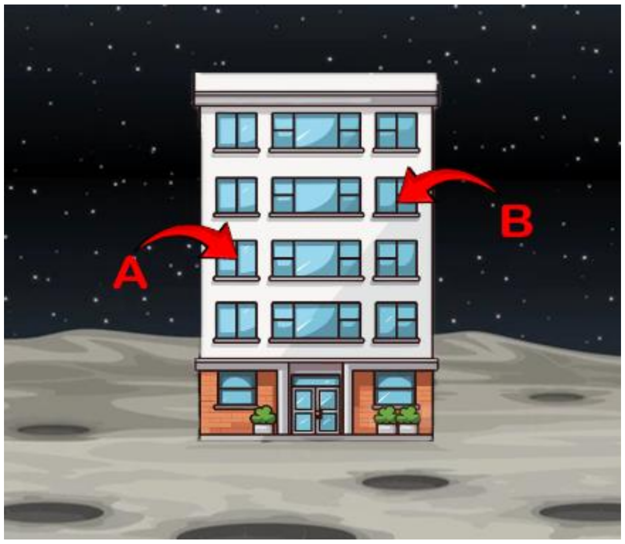
Phase	Check	Feedback
		the *height of the ball*.
Analysis	Ensures that the chosen dependent variable in the first dropdown menu aligns with the goal	- The selected option in the field of the first dropdown menu does not match with what we are investigating. - In the field of the first dropdown menu, it says *colour of the ball*. This is not what we want to investigate. The correct answer is *height of the ball*.
	Ensures that the chosen independent variable in the third dropdown menu aligns with the goal	- The selected option in the field of the third dropdown menu does not match with what we are investigating. - In the field of the first dropdown menu, it says *speed when the ball hits the ground*. This is not what we want to investigate. The correct answer is *time before the ball hits the ground*.
	Ensures that the student's description of results matches observations	- The *height of the ball* and the *time before the ball hits the ground* are both correct! You won't have to change these fields anymore. However, your claim is not fully correct. Look again at what happened. When the *height of the ball* increased/decreased, what happened then with the *time before the ball hits the ground*? - It still is not fully correct. The correct answer is: When I increased the *height of the ball*, the *time before the ball hits the ground* also increased. Please change your answer!
	Ensures the claim about the hypothesis being supported or refuted is correct	- Take another closer look at your hypothesis. Was your hypothesis now TRUE or FALSE?
Proof	Ensures multiple experimental results were selected for analysis	- You only selected one result. To prove if something changed or not in the experiment, you need at least two results from the experiment.
	Ensures the selected results show different values of the independent variable	- At the moment, you only selected results where the *height of the ball* is the same. Make sure you select results where the *height of the ball* has been changed.

Phase	**Check**	**Feedback**
	Ensures the selected results represent a controlled experiment	- You’re on the right track! You have chosen two options where the *height of the ball* has been changed. Only within the selected options, something else has also changed. Please choose options where only the *height of the ball* has been changed and the rest stays the same. - Not quite yet. It is true that the *height of the ball* is different between the two selected options. However, the *colour of the ball* and the *mass of the ball* need to stay the same. Please choose options where only the *height of the ball* has been changed and the rest stays the same.

Appendix C: Pre- and Post-test

This appendix contains a translation of the pre- and post-test conducted as part of this study, where the pre- and post-test are parallel forms of each other. The test layouts and overall design were a lot more kid-friendly than shown below, and they both were incorporated into the Go-Lab online environment. The questions and exercises were used to analyse participants' reasoning and understanding of physics phenomena on the moon. Each assignment includes multiple-choice questions and reasoning-based explanations, with scoring criteria provided for each response.

Pre-test



Assignment 1

Imagine this: You live in an apartment building on the Moon, in Room A. Let's assume you can live on the Moon as if it were normal. You hold a ball outside the window in your hands and then let go of the ball.

Question 1.1:

What happens when you let go of the ball?

(1 point): The ball falls downward / The ball falls toward the Moon / it falls / the ball bounces (bouncing could only happen on the moon surface, which indicates it falls).

(0 points): The ball floats / The ball rises / the ball floats down (contradiction) / Something else.

Question 1.2:

Can you explain why this happens?

(1 point): Gravity / There is less gravity / Gravity is different on the moon (this answers is only a logical answer when the learning indicates that ball falls slower down in question 1.1) / The Moon's gravitational pull.

Note: If the student already says that the ball falls because of gravity in question 1.1, then he/she still gets a point for 1.2, although it is not mentioned in the correct box.

(0 points): In space, things float / There is no gravity in space / Something else.

Note: The answer should be a logical follow-up on question 1.1. So, when a student talks about gravity, but it first says the ball floats, it is still incorrect.

Question 1.3:

Your friend Sarah also lives in the same apartment building on the Moon, in Room B. Now, both of you hold a ball outside your windows at the same time. At the exact same moment, you both let go of your balls. What happens?

a) **Both balls fall to the ground. The ball you let go of (A) hits the ground before the ball of Sarah (B).**

b) Only Sarah's ball (B) falls to the ground; yours floats into space.

c) Both balls fall to the ground and hit the ground at exactly the same time.

d) Both balls float and drift into space.

e) Both balls fall to the ground. Sarah's ball (B) hits the ground before your ball hits the ground (A).

f) Only your ball (A) falls to the ground; Sarah's ball (B) floats into space.

Note: Answer E can be marked as correct, ***provided*** *the answer to 1.4 or 2.2 clearly indicates that the student has switched window A (you) and B (Sarah). For example, the student might explicitly state that Sarah is lower, and you are higher up, even though in the diagram A is lower than B. Make sure to remain consistent with this interpretation for that student.*

Question 1.4:

Why did you choose your answer for Question 1.3? Explain your reasoning.

(1 point): The ball I dropped from Window A is lower than the window Sarah dropped her ball from, Window B. / Window A is lower than B / B is higher than A.

Note: If the student clearly indicates that they accidentally associate window A with Sarah and window B with themselves, the reverse can be marked as correct. These are statements like "because Sarah drops the ball from a lower position than me." Ensure *consistency with this interpretation. Highlight this specific answer in red in Excel when coding.*

(0 points): Window B is lower than A / Another explanation.

Assignment 2

Sarah and you want to conduct another experiment. You are still standing at window A, and Sarah is still standing at room B. You want to find out what happens to the speed of the ball when it hits the ground if you release the balls from different heights. So, you both let go of your balls at the exact same time from your respective rooms.

Question 2.1:

What do you expect to happen to the speed of the balls when they hit the ground?

a) The ball Sarah drops from window B hits the ground with a higher speed, compared to your ball, which is dropped from window A.

b) The ball you drop from window A hits the ground with a higher speed, compared to Sarah's ball, which is dropped from window B.

c) Both balls hit the ground at exactly the same speed.

d) Both balls float and never hit the ground.

e) This cannot be determined because it depends on which way the ball floats.

*Note: Answer B can be marked as correct, **provided** that the answer to 1.4 or 2.2 clearly shows that the student has switched window A (you) and B (Sarah). For example, if the* *student explicitly states that Sarah is lower and you are higher, even though in the diagram B is higher than A. Be consistent with this interpretation for that student.*

Question 2.2:

Why did you choose your answer for Question 2.1? Explain your reasoning.

(1 point) The speed increases if the ball has a longer distance to fall / the farther the ball falls, the faster it arrives. (The student must mention that the speed increases during the descent.)

(1/2 point) The window where Sarah is standing is higher / window A is lower / the distance from window B is longer.

Note: If the student clearly indicates that they accidentally associate window A with Sarah and B with themselves, the reverse can be marked as correct. These are statements like, "because Sarah drops the ball from a lower position than me." Be consistent with this interpretation. Highlight this specific answer in red in Excel.

(0 points) It doesn’t matter how high you start / the speed of the ball does not change based on height / another explanation.

Assignment 3

Imagine this: Tim and his smaller brother want to go bowling on the Moon. Bowling balls are heavy so they can knock down pins more easily without changing direction. However, it’s important to choose a ball that isn’t too heavy, so it can still be lifted. Therefore, Tim chooses the heavier green ball, which weighs 12 kilograms. His little brother chooses a lighter ball: the blue ball, which weighs 6 kilograms.

Question 3.1:

Tim and his little brother want to see what effect the ball's mass has on the Moon. So, they both let go of their balls at the exact same time and from the same height. What happens?

a) Both balls fall to the ground. The green ball Tim dropped (12 kg) hits the ground first.

b) Only Tims green ball (12 kg) falls to the ground; his little brother’s blue ball (6 kg) floats into space.

c) Both balls fall to the ground and hit the ground at exactly the same time.

d) Both balls float and drift into space.

e) Both balls fall to the ground. The blue ball dropped by Tim’s little brother (6 kg) hits the ground first.

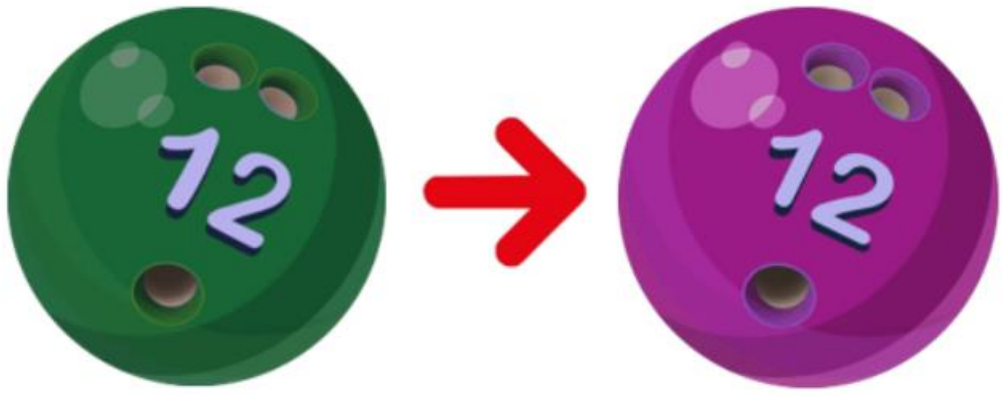
f) Only the blue ball dropped by Tim’s little brother (6 kg) falls to the ground; Tim's green ball (12 kg) floats into space.

Question 3.2:

Why did you choose your answer for Question 3.1? Explain your reasoning.

(1 point): The mass of the balls does not affect how long it takes for them to fall / The weight of the ball doesn’t matter (the phrase "on the time it takes for the ball to hit the ground" is implied from the previous question).

(0 points): 12 is greater than 6 / Because the blue ball is lighter, it floats / Another explanation.



Question 3.3:

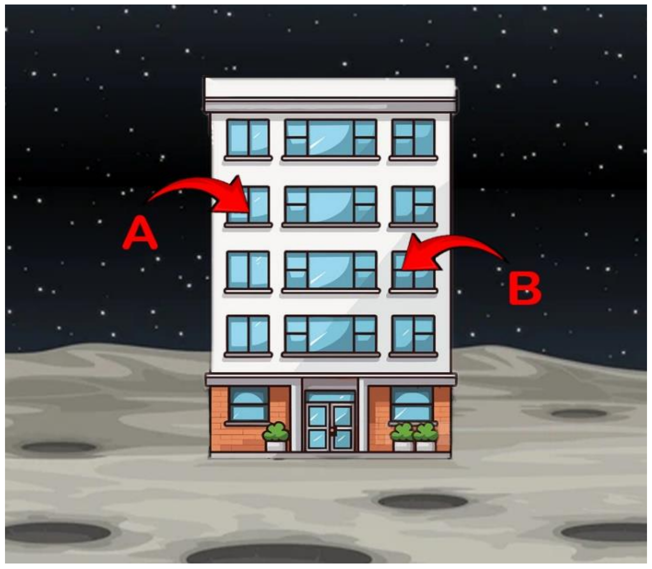
Tim wonders if anything would change if he replaced his ball with one of a different color but the same mass (12 kg). What do you think—would anything change if he dropped this ball? Explain your reasoning.

(2 points) No, the color of the ball does not affect how long it takes to reach the ground. / The color of the ball has no influence (the answer "no" is implied). / No, only the color is different (this implies that the color has no influence).

(1 point) No / The same thing will happen / No, the mass is still the same (this implies that color has no influence but incorrectly suggests that the mass would have an effect). / This doesn’t change the experiment. / They reach the ground at the same time.

(0 points) Yes / The ball falls faster. / Yes, the color is different (but this doesn’t change *what* happens). / No, because the balls stay floating. / Another answer.

Post-test



Assignment 1

Imagine this: You live in an apartment building on the Moon, in Room A. Let’s assume you can live on the Moon as if it were normal. You hold a ball outside the window in your hands and then let go of the ball.

Question 1.1:

What happens when you let go of the ball?

(1 point): The ball falls downward / The ball falls toward the Moon / it falls / the ball bounces (bouncing could only happen on the moon surface, which indicates it falls).

(0 points): The ball floats / The ball rises / the ball floats down (contradiction) / Something else.

Question 1.2:

Can you explain why this happens?

(1 point): Gravity / There is less gravity / Gravity is different on the moon (this answers is only a logical answer when the learning indicates that ball falls <u>slower</u> down in question 1.1) / The Moon's gravitational pull.

Note: If the student already says that the ball falls because of gravity in question 1.1, then he/she still gets a point for 1.2, although it is not mentioned in the correct box.

(0 points): In space, things float / There is no gravity in space / Something else.

Note: The answer should be a logical follow-up on question 1.1. So, when a student talks about gravity, but it first says the ball floats, it is still incorrect.

Question 1.3:

Your friend Thomas also lives in the same apartment building on the Moon, in Room B. Now, both of you hold a ball outside your windows at the same time. At the exact same moment, you both let go of your balls. What happens?

a) Both balls fall to the ground. The ball you let go of (A) hits the ground before the ball of Thomas (B).

b) Only Thomas’s ball (B) falls to the ground; yours floats into space.

c) Both balls fall to the ground and hit the ground at exactly the same time.

d) Both balls float and drift into space.

e) Both balls fall to the ground. Thomas's ball (B) hits the ground before your ball hits the ground (A).

f) Only your ball (A) falls to the ground; Thomas's ball (B) floats into space.

Note: Answer A can be marked as correct, ***provided*** *the answer to 1.4 or 2.2 clearly indicates that the student has switched window A (you) and B (Thomas). For example, the student might explicitly state that Thomas is higher up, and you are lower, even though in the diagram B is lower than A. Make sure to remain consistent with this interpretation for that student.*

Question 1.4:

Why did you choose your answer for Question 1.3? Explain your reasoning.

(1 point): The ball I dropped from Window A is higher than the window Thomas dropped his ball from, Window B. / Window A is higher than B / B is lower than A.

Note: If the student clearly indicates that they accidentally associate window A with Thomas and window B with themselves, the reverse can be marked as correct. These are statements like "because Thomas drops the ball from a higher position than me." Ensure consistency with this interpretation. Highlight this specific answer in red in Excel when coding.

(0 points): Location B is higher than A / Another explanation.

Assignment 2

Thomas and you want to conduct another experiment. You are still standing at room A, and Thomas is still standing at room B. You want to find out what happens to the speed of the ball when it hits the ground if you release the balls from different heights. So, you both let go of your balls at the exact same time from your respective rooms.

Question 2.1:

What do you expect to happen to the speed of the balls when they hit the ground?

a) The ball Thomas drops from window B hits the ground with a higher speed, compared to your ball, which is dropped from window A.

b) The ball you drop from window A hits the ground with a higher speed, compared to Thomas' ball, which is dropped from window B.

c) Both balls hit the ground at exactly the same speed.

d) Both balls float and never hit the ground.

e) This cannot be determined because it depends on which way the ball floats.

Note: Answer A can be marked as correct, ***provided*** *that the answer to 1.4 or 2.2 clearly shows that the student has switched window A (you) and B (Thomas). For example, if the student explicitly states that Thomas is higher and you are lower, even though in the diagram B is lower than A. Be consistent with this interpretation for that student.*

Question 2.2:

Why did you choose your answer for Question 2.1? Explain your reasoning.

(1 point) The speed increases if the ball has a longer distance to fall / the farther the ball falls, the faster it arrives. (The student must mention that the speed increases during the descent.)

(1/2 point) The window where you are standing is higher / window B is lower / the distance from window A is longer.

Note: If the student clearly indicates that they accidentally associate window A with Thomas and B with themselves, the reverse can be marked as correct. These are statements like, "because Thomas drops the ball from a higher position than me." Be consistent with this interpretation. Highlight this specific answer in red in Excel.

(0 points) It doesn't matter how high you start / the speed of the ball does not change based on height / another explanation.

Assignment 3

Imagine this: Matthijs and his father want to go bowling on the Moon. Bowling balls are heavy so they can knock down pins more easily without changing direction. However, it’s important to choose a ball that isn’t too heavy, so it can still be lifted. Therefore, Matthijs chooses the red ball, which weighs 8 kilograms. His father chooses a heavier ball: the blue ball, which weighs 16 kilograms.

Question 3.1:

Matthijs and his father want to see what effect the ball's mass has on the Moon. So, they both let go of their balls at the exact same time and from the same height. What happens?

a) Both balls fall to the ground. The red ball Matthijs dropped (8 kg) hits the ground first.

b) Only Matthijs's red ball (8 kg) falls to the ground; his father’s blue ball (16 kg) floats into space.

c) Both balls fall to the ground and hit the ground at exactly the same time.

d) Both balls float and drift into space.

e) Both balls fall to the ground. The blue ball dropped by Matthijs’s father (16 kg) hits the ground first.

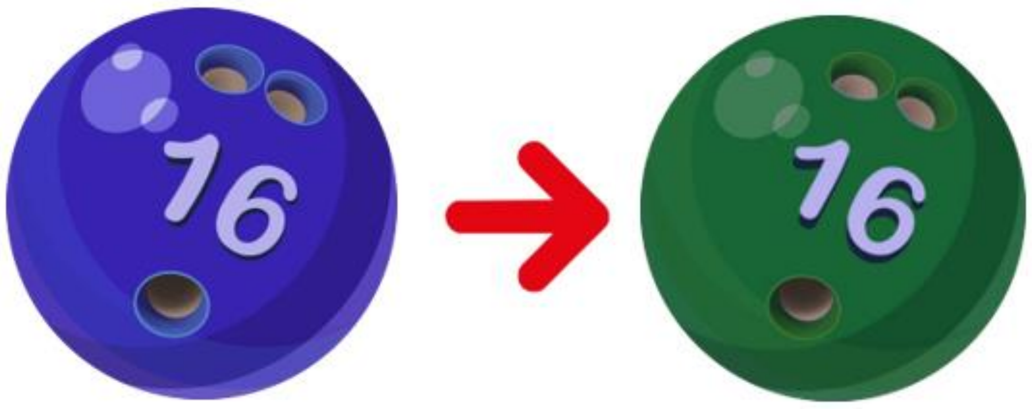
f) Only the blue ball dropped by Matthijs’s father (16 kg) falls to the ground; Matthijs's red ball (8 kg) floats into space.

Question 3.2:

Why did you choose your answer for Question 3.1? Explain your reasoning.

(1 point): The mass of the balls does not affect how long it takes for them to fall / The weight of the ball doesn’t matter (the phrase "on the time it takes for the ball to hit the ground" is implied from the previous question).

(0 points): 16 is greater than 8 / Because the red ball is lighter, it floats / Another explanation.



Question 3.3:

Matthijs's father wonders if anything would change if he replaced his ball with one of a different color but the same mass (16 kg). What do you think—would anything change if he dropped this ball? Explain your reasoning.

(2 points) No, the color of the ball does not affect how long it takes to reach the ground. / The color of the ball has no influence (the answer "no" is implied). / No, only the color is different (this implies that the color has no influence).

(1 point) No / The same thing will happen / No, the mass is still the same (this implies that color has no influence but incorrectly suggests that the mass would have an effect). / This doesn't change the experiment. / They reach the ground at the same time.

(0 points) Yes / The ball falls faster. / Yes, the color is different (but this doesn't change *what* happens). / No, because the balls stay floating. / Another answer.

Appendix D: Self-regulated learning questionnaire

This Appendix contains the self-regulated learning questionnaire (Sins et al., 2024). However, the original text was given in Dutch, and the overall layout was more age-friendly for the target group. The questionnaire was designed to evaluate two key aspects: (1) how often participants engage in specific research-related behaviors, and (2) their confidence or knowledge in performing these behaviors.

Read this text carefully before starting the questionnaire.

You are about to complete a questionnaire. This questionnaire is about how you work when conducting research.

By research, we mean everything you do to figure out the answer to a question.

Research could include activities like searching for information or conducting an experiment. In a few days, you will conduct research on dropping a ball on the Moon.

The questionnaire measures two things:

- How often you do something.

- How well you know how to do something.

Sometimes, you might know how to do something well but not do it often. For example: cleaning your room. Many people know how to do it, but they don’t do it very often. Other times, you might not know how to do something well, especially if it’s new to you or if it’s complicated.

The questionnaire starts with a few general questions. After that, there are 17 questions about how you work when conducting research.

1. What is your first name?

2. What is your last name?

3. What is your age?

8 9 10 11 12 13 14 15

4. What is your gender?

Boy Girl Other

5. How do you work when conducting research? (1 of 4)

| How often do you do this? | Do you know how to do it? |

(options: Never, Almost Never, Sometimes, Almost Always, Always | options: No idea, no, a little, yes, totally)

Before I start research, I look at what I will do first and what I will do next.

If I find a step in the research difficult, I make sure to give it more time.

For a big research project, I spread my work over a longer period of time.

Before I start research, I consider how long I will spend on it.

6. How do you work when conducting research? (2 of 4)

| How often do you do this? | Do you know how to do it? |

(options: Never, Almost Never, Sometimes, Almost Always, Always | options: No idea, no, a little, yes, totally)

Before I start research, I ask myself: ‘What is the research about? What do I already know about this topic?’

Before I start research, I ask myself: ‘Have I read or heard anything about this topic before?’

Before I start research, I ask myself: ‘Have I learned anything about this topic before?’

Before I start a research assignment, I ask myself: ‘Have I done a similar assignment before?’

If I have to do a research assignment that I have done before, I ask myself: ‘How did I do it then? Was that a good method?’

7. How do you work when conducting research? (3 of 4)

| How often do you do this? | Do you know how to do it? |

(options: Never, Almost Never, Sometimes, Almost Always, Always | options: No idea, no, a little, yes, totally)

During research, I ask myself: ‘Do I still have enough time?’

During research, I check what I have already done.

During research, I check how much more I need to do.

During research, I follow the planning I made.

8. How do you work when conducting research? (4 of 4)

| How often do you do this? | Do you know how to do it? |

(options: Never, Almost Never, Sometimes, Almost Always, Always | options: No idea, no, a little, yes, totally)

Before I start research, I think about my goals.

During research, I ask myself: ‘Is this method working well?’

During research, I ask myself: ‘Do I still understand everything?’

During research, I ask myself: ‘What do I find difficult? What should I practice more?’