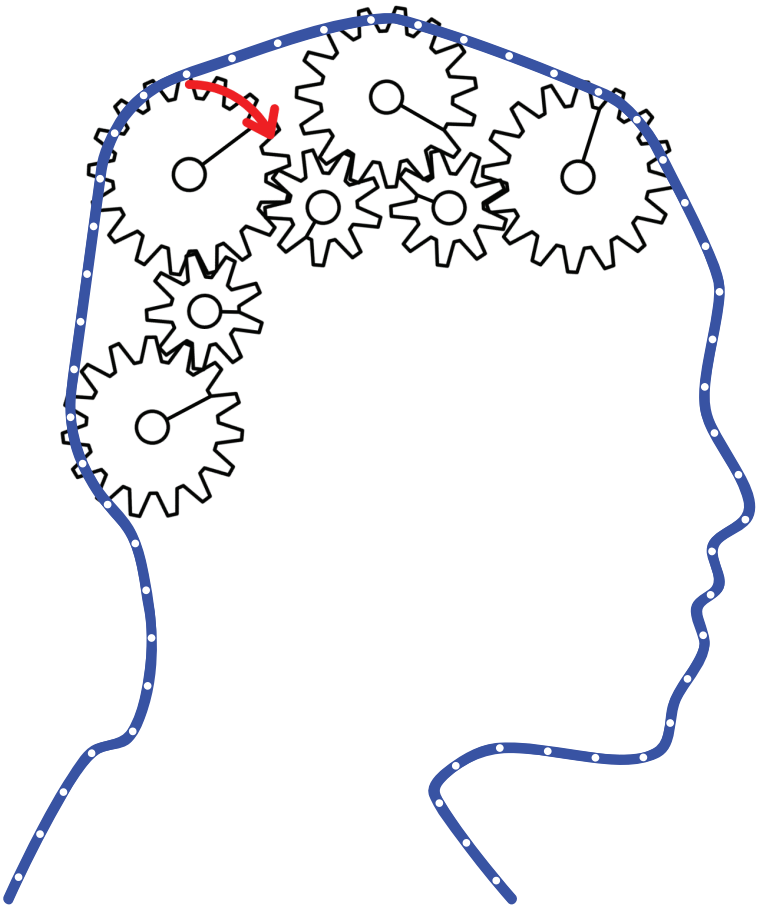
Drawing gears
and
chains of reasoning



Frank Leenaars

DRAWING GEARS

AND

CHAINS OF REASONING

F.A.J. LEENAARS

Doctoral committee

Chair:	prof. dr. ir. A.J. Mouthaan
Promotor:	prof. dr. W.R. van Joolingen
Assistent promotors:	dr. L. Bollen
	dr. A.H. Gijlers
Members:	prof. dr. K.D. Forbus
	prof. dr. D.K.J. Heylen
	prof. dr. J.T. Jeuring
	prof. dr. A.J.M. de Jong
	prof. dr. J.H. Walma van der Molen

CTIT

ISBN: 978-90-3653804-6

DOI: 10.3990/1.9789036538046

Print: Ipskamp Drukkers, Enschede, The Netherlands

DRAWING GEARS AND CHAINS OF REASONING

PROEFSCHRIFT

ter verkrijging van

de graad van doctor aan de Universiteit Twente,

op gezag van de rector magnificus,

prof. dr. H. Brinksma,

volgens besluit van het College voor Promoties

in het openbaar te verdedigen

op woensdag 10 december 2014 om 14:45 uur

door

Franciscus Adriaan Johannes Leenaars

geboren op 6 juni 1984

te Hengelo

Dit proefschrift is goedgekeurd door de promotor:

prof. dr. W.R. van Joolingen

en de assistent-promotoren:

dr. L. Bollen

dr. A.H. Gijlers

DANKWOORD

Hoewel mijn naam eenzaam op de kaft van dit proefschrift staat, is dit boekje tot stand gekomen dankzij samenwerking met en hulp van vele anderen. Een aantal mensen wil ik hier graag in het bijzonder bedanken.

Wouter, Hannie en Lars, jullie hebben mij begeleid tijdens de bachelorthese, de masterthese en nu tijdens dit promotietraject. Wouter, je hebt me altijd veel vertrouwen en vrijheid gegeven. Regelmatig kwam ik enthousiast met één of ander idee bij je binnenlopen en verliet dan even later vol energie je kantoor weer, omdat je constructief mee dacht en me de kans gaf om uit te zoeken waar deze ideeën tot leidden. Hannie en Lars, jullie konden mij helpen met alle vragen die ik had, of ze nou gingen over het vinden van de juiste statistische analyse of het beste algoritme om te bepalen of een constructie van tandwielen en kettingen nog wel kon bewegen. Wanneer ik weer eens vlak voor een deadline met dit soort vragen naar jullie toe kwam, kon ik dankzij jullie expertise snel weer verder.

Al mijn collega's bij IST, bedankt voor de stimulerende werkomgeving. Marjolein, jij was mijn kamergenote tijdens het grootste deel van mijn promotietraject en ik heb genoten van al onze concrete, inhoudelijke discussies en onze gesprekken over wensen en dromen voor de toekomst. ProIST, bedankt dat jullie mij iedere week achter mijn bureau vandaan sleurden om ook nog een beetje sociaal te doen! En natuurlijk bedankt aan iedereen die me geholpen heeft bij mijn experimenten.

To everybody at QRG at Northwestern University, thank you for making my three-month visit with your group an amazing experience. I learned so much from all of you and was very happy with how welcoming you were by inviting me to board game nights, barbecues and bars from the first week I got there.

De onderzoeken die beschreven zijn in dit proefschrift zouden niet mogelijk geweest zijn zonder de medewerking van de docenten en leerlingen van de Bonifatiusschool, de Mare, de Prinseschool, de Rank en de Telgenkamp. In het bijzonder wil ik Jan Goorhuis bedanken, voor de perfecte organisatie van de deelname van leerlingen van de Telgenkamp aan alle onderzoeken die beschreven zijn in dit proefschrift.

Arjan, Peter en Rob, bedankt voor alle leuke en ontspannen momenten die we gehad hebben. Ook al wonen we nu niet meer in dezelfde stad, toch vinden we nog genoeg tijd om samen te chillen.

Mijn familie, dankzij jullie weet ik dat er altijd een plek is waar ik mezelf kan zijn, waar ik wijze raad en een tosti kan krijgen en waar ik een spelletje kan winnen. Pap en Ron, bedankt dat jullie mijn ervaren paranimfen willen zijn.

Annika, al meer dan zes jaar maak jij mijn leven mooier dan het was. Hopelijk kan ik je tijdens jouw eigen promotietraject half zo veel steunen als jij mij gesteund hebt.

TABLE OF CONTENTS

1

GENERAL INTRODUCTION

This dissertation discusses the implementation and evaluation of GearSketch, a learning environment for the gears domain aimed at students in the final years of primary school. This learning environment has a drawing-based interface and lets learners explore the gears domain through simulations which are visualized with animations. Later versions of GearSketch incorporate ideas from cognitive tutors by adaptively selecting practice items for individual learners based on a student model and supporting step-by-step problem solving. This introductory chapter provides a theoretical background by discussing research on simulations (1.1), visualizing results through animations (1.2), drawing in science education (1.3), student modeling and cognitive tutors (1.4) and learning modes and problem solving (1.5). The final section (1.6) introduces the research questions addressed in this dissertation.

1.1 Simulations in education

Simulations are programs that contain a computational model of a system or a process (De Jong & Van Joolingen, 1998). Learners who use simulations can interact with them by changing parameters of the model and examining how this affects the model's behavior. Compared to other approaches to education, such as textbooks and lectures, simulations offer several advantages. They give learners the opportunity to actively explore both realistic and hypothetical situations, to examine events that occur at very small or very large time scales and to interact with idealized versions of the system being simulated (Van Berkum & De Jong, 1991). By exploring hypothetical situations, learners can gather information about the behavior of a system that is not available through non-interactive lesson materials. Changing the time-scale of events lets learners experience events in a way that would not be possible using real demonstrations. For instance, the PhET computer simulations (Wieman, Adams, & Perkins, 2008) let students examine atomic interactions by slowing time down so interactions between individual atoms can be seen and let students explore plate tectonics by speeding time up so that millions of years pass in seconds. The idealized nature of simulations helps students focus on the important aspects of a system. Additionally, students find simulations fun and engaging, which can positively affect their motivation (Khan, 2011; Wieman et al., 2008).

Do these advantages of simulations result in improved learning outcomes? A meta-analysis by Vogel et al. (2006) found evidence that students using interactive simulations have a better attitude towards learning and show higher cognitive gains than those using traditional methods. However, they concluded that it was not possible to draw this conclusion with much confidence because many studies about the use of simulations did not have clearly defined control groups, did not report statistical data, left out important demographic details or did not describe the intervention in sufficient detail. A more recent literature review by Rutten, Van Joolingen and Van der Veen (2012) also found positive effects of enhancing traditional education with simulations, such as facilitating conceptual understanding, improving the ability to predict results of experiments and improving students' cognitive focus, but cautions that most of the reviewed studies only reported short-term results and did not examine long-term effects.

When simulations are used instead of physical labs, the learning experience loses its physicality. Some researchers consider these simulated experiences with the natural environment not to be hands-on activities (Flick, 1993). Others do consider direct manipulation in virtual environments to be hands-on and argue that it can be just as effective as physical manipulation. Zacharia and Olympiou (2011) compared physical manipulative experimentation (PME) with virtual manipulative experimentation (VME) in a module on heat and temperature for university students. They found that the

understanding of students who learned with PME or VME was equally enhanced and that students in both conditions learned more than those following traditional instruction. Simulations can also be used to prepare learners for experiments in a physical laboratory. Zacharia and Anderson (2003) found that students who read a text and used simulations before doing physical experiments made better predictions about these experiments than a control group that prepared by reading the same text and solving practice problems. When using simulations instead of physical labs, the experience is more idealized, because simulations are necessarily based on simplified models of reality. Goldstone and Son (2005) discuss the relative advantages of concrete and idealized representations. Idealized representations are more transferable to other domains and the essence of a phenomenon is highlighted. However, concrete information is easier to remember than abstract information, concrete materials can be more engaging than abstractions and are more clearly connected to real-world situations. Fortunately, simulations are not locked-in to just one representation of reality and can vary the idealization of representations as the student progresses through the learning environment. Goldstone and Son (2005) found that starting out with concrete representations and changing to idealized representations as learners progressed, resulted in student performance that was better than the opposite transition or sticking with a single representation style. Such transitions would not be possible in physical labs.

A consistent finding in research on simulations is that just giving learners a simulation and asking them to figure out how the underlying model works is not efficient (Kirschner, Sweller, & Clark, 2006; Mayer, 2004). Exploring a simulation to find out the rules of the underlying model is an example of discovery learning and requires learners to think of testable hypotheses, design experiments and interpret the results of these experiments. De Jong and Van Joolingen (1998) discuss the problems that learners encounter during each of these steps as well as the problems they encounter during the regulation of this process. Many tools to support discovery learning have been developed and examined. Examples of such tools are an hypothesis scratchpad to support hypothesis generation (Van Joolingen & De Jong, 1991), scaffolding software for designing experiments (Morgan & Brooks, 2012), a curve fitting tool to analyze experimental data (Van Joolingen, De Jong, Lazonder, Savelsbergh, & Manlove, 2005) and a concept mapping tool to support regulation of the learning process (Hagemans, Van der Meij, & De Jong, 2013). Two meta-analyses by Alfieri, Brooks, Aldrich and Tenenbaum (2011) show that while explicit instruction compares favorably to unassisted discovery learning, supported discovery learning compares favorably to other forms of instruction.

1.2 Visualization and animation

Rutten et al. (2012) reviewed studies that examined different representations in simulations and found that adding representations sometimes affected learning outcomes, but that most studies found no effects. For example, Ploetzner, Lippitsch, Galmbacher, Heuer and Scherrer (2009) found that adding dynamic representations to a line graph did not affect learning outcomes. In their study in the domain of kinematics, they compared three conditions within a simulation-based learning environment. The first condition used an image of a runner and a line graph representing the distance traveled by this runner over time, that could be played back and paused at will. The second condition added a vector that dynamically represented the distance traveled by the runner at each point in time. A third condition included both the vector and a stamp diagram that displayed a series of vectors at different points in time to show the runner's progress. However, a posttest that assessed learners' ability to interpret and construct time-position, time-velocity, time-acceleration and time-force graphs showed no difference in learning outcomes between the conditions. On the other hand, Trey and Khan (2008) found that adding a dynamic representation of a weigh scale as an analogy of Le Chatelier's principle to a simulation did lead to improved learning outcomes.

Van der Meij and De Jong (2006) also studied the effects of having multiple representations, but they examined effects of the way in which these representations were integrated and linked to each other instead of effects of adding more representations. When multiple representations are integrated, they occupy the same physical area and appear to be one representation that shows different aspects of the domain. Dynamically linking representations means that when the value of one representation is changed, all representations that are linked to it automatically change as well. For example, when a student changes the size of a force in a numerical representation, the length of a vector representing this force in a picture changes at the same time. The researchers found that students who worked with representations that were both integrated and dynamically linked learned more than students who used separate, non-linked representations. Students also found the integrated, dynamically linked representations easier to work with than unintegrated or non-linked representations.

Animations are commonly used as a form of representation in simulations. Research into the question whether animations are better than static representations has resulted in mixed findings. Hegarty, Kriz and Cate (2003) describe three studies in which static representations are compared to animations of mechanical systems. Their overall finding is that students learn no more from animated representations than they do from series of static representations. One explanation for this finding is that in animations, many parts of the system move at the same time. In contrast, when people attempt to

understand how a system works they reason about the motion of its components sequentially, following a chain of causes and effects. A series of static representations may be just as effective as an animation, if these static representations include the main causes and effects in the chain of events represented by the animation. A second explanation offered by the researchers is that just viewing an animation is a passive progress. When animations are interactive, as they are when used as part of a simulation, they may be more effective. A review study by Tversky, Morrison and Betrancourt (2002) also found that animations often did not facilitate understanding better than static representations. However, they explicitly excluded animations that include interactivity from their review, because "[interactivity] is known to benefit learners on its own" (p. 250). A meta-analysis by Höffler and Leutner (2007) paints a more positive picture for non-interactive animations. The main outcome of this analysis was that animations offer a medium-sized advantage over static pictures, but only when the role of the animation is representational rather than decorational. Together, these findings indicate that using animations to represent simulation results will likely be effective.

1.3 Drawing in science education

Ainsworth, Prain and Tytler (2011) list five reasons why drawing should be recognized as a key element of science education. First, drawing engages students more than conventional teaching. Second, creating drawings helps students understand conventions of representations and their purposes. Third, drawing can help students reason about multiple representations. As students create their own drawings, they make choices about what to represent and what to leave out, which can give them insight into the function of multiple representations of the same phenomenon. Fourth, reading a text and then creating a drawing to represent their understanding of this text makes students' mental models explicit. This can help students identify key features of the subject under study. Fifth, drawings help students communicate their understanding to both their peers and their teacher.

A literature study by Van Meter and Garner (2005) investigated what is known about learning by creating drawings and reached three tentative conclusions. Their first finding was that drawing accuracy when creating drawings based on text was significantly correlated with posttest performance in every study that scored drawings for accuracy. This positive correlation was found in studies during which participants could use their drawings during the posttest, but also in studies in which participants did not have their drawings available during the test. Two different mechanisms could explain this result. One possibility is that learners who better understood the text created more accurate drawings and then performed better on the posttest without benefitting

from creating their more accurate drawing. A second possibility is that creating more accurate drawings improves the effectiveness of drawing as a learning strategy and therefore improves posttest performance. An earlier study by Van Meter (2001) provided some evidence for the second interpretation, as it found that prompting students to compare their own drawings with example drawings after they had read the text, could improve the accuracy of their drawings as well as their results on a free-recall posttest. Van Meter and Garner's second finding was that support is necessary for effective drawing strategy use. For example, learners can be supported by pictures to compare with their own drawings (Van Meter, 2001) or by instructions to attend to certain aspects of their representation (Alesandrini, 1981) and these forms of support lead to improved posttest performance. Van Meter and Garner's third finding was that the benefits of drawing construction are found mostly with higher-order assessments. For example, Alesandrini's (1981) study found that creating a drawing after reading a text lead to better posttest results than writing a summary, whereas a study by Snowman and Cunningham (1975) found no significant difference between these strategies. These different results can be explained by the types of posttest used. Whereas Alesandrini used a posttest that contained application questions, Snowman and Cunningham's posttest contained only factual recognition items. More recent studies have also found positive effects for using a drawing strategy to learn from a text when posttests focus on higher-order comprehension and problem solving (Leopold & Leutner, 2012; Van Meter, Aleksic, Schwartz, & Garner, 2006).

In addition to creating drawings to learn from texts, students can also create drawings to facilitate problem solving. Van Essen and Hamaker (1990) examined the effect of teaching fifth grade students to create drawings while solving word problems, such as "Along one side of a road there are 8 trees in a line. The distance between 2 trees is 10 meters. What is the distance between the first tree and the last one?" (p. 307). Students in the experimental group received two half hour lessons during which they were instructed in the creation of drawings, while students in a control group followed the regular lessons. Students in the experimental group outperformed students in the control group on a subsequent word problem test.

With the increasing availability of technology in schools, drawing construction is no longer restricted to paper and pencil, but can be done using computers and touchscreens as well. Jee et al. (in press) used CogSketch (Forbus, Usher, Lovett, Lockwood, & Wetzel, 2011) to study how domain knowledge was reflected in sketches of scientific structures and processes. In a series of experiments researchers asked geoscience students (relative experts) and students in other fields (relative novices) to create sketches based on geoscience-related photographs and diagrams. They found that the relative experts included more spatial structures that reflected geologic activity in their sketches of photographs and more relational information in their sketches of diagrams

than the relative novices. In addition to these differences in the final sketches, the researchers found differences in sketching order between the groups. When the geoscience students sketched events in diagrams, they did so in an order that matched the causal order of these events as identified by an expert. The relative novices did not sketch the events in this order. Using CogSketch to collect and analyze sketches made it possible to identify this difference in drawing order between the two groups, which would have been difficult to do with pencil-and-paper drawings. This kind of extra information that is available when using digital drawings may be used to get a better picture of students' level of understanding of a domain.

A different example of using learner-created drawings with technology is SimSketch (Bollen & Van Joolingen, 2013), which is based on ideas from modeling and simulation. When working with SimSketch, learners can annotate their drawings with behavior labels. These labels define actions of and interactions between different objects in the learner's drawing. For example, a sketch of the sun, the earth and the moon in which circle behaviors are added to the earth (to indicate that it circles the sun) and the moon (to indicate that it circles the earth), lets learners explore the trajectory of the moon relative to the sun. Adding multiplication behavior to drawings of bacteria can give learners insight into the effects of exponential growth. Such an implementation of modeling and simulation, based on drawings instead of equations or a modeling language, can make this approach to learning usable by learners at an early age.

1.4 Student modeling and cognitive tutors

Students learning with an expert human tutor who adapts her instruction to each individual learn much more effectively than those learning in a classroom with about 30 students per teacher. Bloom (1984) found that on average individually tutored students' scores on subsequent tests were two standard deviations above those of students who received conventional instruction. A more recent meta-analysis by VanLehn (2011) found a smaller but still large effect ($d = 0.79$) of human tutoring on learning outcomes. These results provide motivation for developing interactive learning environments that adapt to individual learners as well. This adaptation is often based on a student model that captures aspects of individual students' knowledge and skills.

Chrysafiadi and Virvou (2013) reviewed the literature on student modeling and discuss a number of different approaches that have been studied. One of the most popular approaches is the overlay model. The overlay model assumes that learners' knowledge is a subset of a domain model which is based on expert-level knowledge of the subject. The domain model consists of a number of elements that represent knowledge of facts and achievement of skills. The learner's objective is then to learn these facts and develop these skills to achieve mastery in the domain. An overlay model is often

represented by a collection of probabilities estimating whether the learner knows each element of the domain model. These probabilities are updated after each practice problem to model learners' progress over time. Because an overlay model only represents students' correct domain knowledge, it cannot be used to model common misconceptions that a student may have developed. A perturbation student model is an extension of the overlay model that does include such misconceptions. A learning environment can use such a perturbation model to identify exactly which misconceptions a student has developed and offer practice problems or explanations to resolve these misconceptions. The disadvantage of perturbation models is that in addition to an expert view of the domain, insight into common misconceptions in this domain is required to implement the model. Additionally, the model becomes more complex, which generally means that students must complete more practice problems before the model correctly represents the students' current knowledge of the domain.

Cognitive or intelligent tutors are interactive learning environments based on a theory of human cognition and a student model. Many tutors are based on ACT* theory (Anderson, Boyle, Corbett, & Lewis, 1990) and ACT-R theory (Anderson et al., 2004) which have at their core two distinct types of knowledge: declarative knowledge and procedural knowledge. Declarative knowledge refers to facts that are stored in human memory and can be acquired by listening to a lecture or reading a text. Procedural knowledge is knowledge of how to do something and is acquired by applying declarative knowledge in new situations, for example in practice problems. Cognitive tutors can model the acquisition of procedural knowledge by using model tracing and knowledge tracing (Corbett & Anderson, 1995). Model tracing occurs at the level of individual problems that learners attempt to solve. Each step a student takes to solve the problem is compared to applicable rules in the domain model. If the student's step is incorrect or unproductive, the step is not allowed and the student may receive feedback explaining why this step should not be used. This ensures that the student stays on recognized solution paths and that the tutor can always find the next steps leading to a solution. Knowledge tracing happens at a higher level and is used to keep track of a student's current domain knowledge. It uses an overlay model to represent the domain knowledge that is to be learned and keeps track of the current probabilities that the student knows each element in the domain model. Based on this overlay model, it is possible to adaptively select practice problems that lead to the greatest expected learning gains.

Cognitive tutors have been shown to be effective, are now actively used in education and are still an active field of research (Desmarais & Baker, 2012). Aleven, McLaren, Sewall and Koedinger (2009) have developed the Cognitive Tutor Authoring Tools (CTAT), which allow teachers with no programming experience to create cognitive tutors through a drag-and-drop interface. After an interface has been created with

CTAT, the author uses this interface to demonstrate how problems can be solved. The author's problem-solving steps are then saved in a behavior graph that can be further edited, annotated and generalized. This behavior graph can then be used to support students working with the cognitive tutor to solve similar problems. Developments like these help make cognitive tutors increasingly accessible and available in education.

1.5 Learning modes and problem solving

According to Hayes and Broadbent (1988) two different modes of learning exist, a selective mode (s-mode) and an unselective mode (u-mode). S-mode learning is a conscious process in which the learner actively creates a verbalizable mental representation of the system or process she is studying. This type of learning will likely be used in situations where a small number of salient variables can explain a system's behavior. In contrast, u-mode learning is unconscious and relies on analysis of the frequency with which certain events co-occur, such as particular actions leading to successful outcomes. Use of u-mode learning is probable in situations in which a lot of information is available and the important variables and their interrelations are not clear. Whereas using s-mode learning leads to knowledge that can be both applied and verbalized, u-mode learning leads to implicit knowledge that can be demonstrated by improved task performance, but is difficult to communicate.

The learning mode used by students depends not only on the domain and task, but is also affected by the interface of a learning environment. Svendsen (1991) discusses an experiment in which participants learn to solve the tower of Hanoi puzzle through practice in a learning environment with either a direct manipulation interface (in which disks could be dragged with the mouse) or a command-driven interface (in which disks were moved by typing e.g. "from 1 to 3"). Students practiced with this problem until they solved the puzzle twice without making any errors or had completed 20 trials. After practicing until this criterion was reached, participants were asked if they had discovered any rules that could be used to solve the problem. Additionally, they were asked how they would explain how to solve the problem to someone who was unfamiliar with the puzzle. Based on their answers to these questions, participants were classified as either being able or unable to verbalize the relevant rules. It was found that participants who used the direct manipulation interface were significantly worse at verbalizing their knowledge of how to solve the puzzle than those in the command-driven interface. This finding indicates that learning environments with direct manipulation interfaces are more likely to induce u-mode learning than learning environments using command-driven interfaces.

Learning environments based on ACT* theory assume that students use a means-ends approach to problem solving when applying their declarative knowledge to solve

practice problems (Anderson, 1993). Such a means-ends approach is a conscious process that resembles s-mode learning. Therefore, care must be taken that a learning environment based on ACT* theory does not unintentionally induce u-mode learning. This could happen when the practice problems it provides are too complex or when it uses a direct manipulation interface. Research on self-explanations provides additional evidence for the importance of stimulating s-mode learning in such learnings environments. Students who often verbally explain solution steps to themselves while studying worked-out examples or solving problems learn more than students who self-explain less often (Chi, Bassok, Lewis, Reimann, & Glaser, 1989). Prompting students to self-explain while learning improves their understanding (Chi, De Leeuw, Chiu, & Lavancher, 1994) and this technique has been successfully applied in the context of cognitive tutors by Aleven and Koedinger (2002). Eliciting self-explanations during problem solving seems to be an effective way of ensuring that students use s-mode learning instead of u-mode learning.

1.6 Dissertation overview

This dissertation discusses GearSketch, a learning environment for the gears domain aimed at students in the final years of primary school. At GearSketch's core is a domain model that is used to transform students' pen strokes into gear and chain systems, ensure the validity of these systems and animate their turning behavior. This means that GearSketch uses ideas from the research on simulations discussed in section 1.1, the research on animations discussed in section 1.2 and the research on learning by drawing discussed in section 1.3. Although GearSketch uses simulations, it is not an inquiry learning environment, but is based on ideas from ACT* theory. Information about the rules governing the gears domain, such as the fact that two meshing gears will turn in opposite directions, is explicitly introduced in a series of tutorials that learners progress through when they start working with GearSketch. After completing these tutorials, students apply their declarative knowledge of the domain rules in practice problems to acquire procedural knowledge. Two types of practice problems are used: questions and puzzles. Questions present a gear and chain system and ask the learner to make a prediction about the behavior of the gears in this system, e.g. "Will gear A turn faster than gear B?" Puzzles present a gear and chain system that is incomplete, and ask the learner to add or move gears or chains to accomplish a goal, e.g. "Add a chain so that gear A will turn in the opposite direction of gear B." Chapters 2, 3 and 4 each discuss an experiment done with a different iteration of GearSketch.

Chapter 2 examines the effects of adding simulation-based support to a drawing-based learning environment. Two versions of GearSketch were created, a static version and a simulation-based version. The static version offered an experience comparable to

learning about gears with a drawing-based strategy using pencil-and-paper. The simulation-based version added support for exploring the behavior of learner-created gear systems with animations. The research questions addressed in Chapter 2 are:

- Do students in the simulation-based condition perform better on practice problems than students in the static condition?
- Do students in the simulation-based condition learn more from working with GearSketch than students in the static condition?

Chapter 3 examines the effects of adding a student model to GearSketch and using this model to adaptively select practice problems for individual students. This student model was developed based on the research discussed in section 1.4. An experiment was done in which participants in the adaptive condition practiced with problems that were selected for them based on a continuously updated student model's estimation of their current domain knowledge, whereas students in the control condition all practiced with the same sequence of practice problems. The research questions addressed in Chapter 3 are:

- Do students in the adaptive condition learn more from working with GearSketch than those in the control condition?
- Can the student model accurately predict students' performance on a posttest?

Chapter 4 examines whether the introduction of a reasoning support tool helps learners during problem solving. This support tool was designed based on ideas about different learning modes discussed in section 1.5. The support tool required students to indicate each reasoning step when answering practice questions and gave feedback on incorrect steps. The support tool only offered support during practice questions, not during practice puzzles. The behavior and performance of participants in a supported condition and participants in a control condition who did not have access to the reasoning support tool were compared. The research questions addressed in Chapter 4 are:

- Are students in the supported condition more successful in answering practice questions than those in the control condition?
- Do students in the supported condition behave differently when attempting to solve practice puzzles than those in the control condition and are they more successful at solving these puzzles?
- Do students in the supported and the control condition learn from working with GearSketch and do those in the supported condition learn more than those in the control condition?

Chapter 5 discusses the results from these experimental studies and offers suggestions for future research.

References

Ainsworth, S., Prain, V., & Tytler, R. (2011). Drawing to learn in science. *Science*, *333*(6046), 1096–1097. doi:10.1126/science.1204153

Alesandrini, K. L. (1981). Pictorial–verbal and analytic–holistic learning strategies in science learning. *Journal of Educational Psychology*, *73*(3), 358–368. doi:10.1037/0022-0663.73.3.358

Aleven, V., & Koedinger, K. R. (2002). An effective metacognitive strategy: Learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science*, *26*(2), 147–179. doi:10.1016/S0364-0213(02)00061-7

Aleven, V., McLaren, B. M., Sewall, J., & Koedinger, K. R. (2009). A new paradigm for intelligent tutoring systems: Example-tracing tutors. *International Journal of Artificial Intelligence in Education*, *19*(2), 105–154.

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. doi:10.1037/a0021017

Anderson, J. R. (1993). Problem solving and learning. *American Psychologist*, *48*(1), 35–44. doi:10.1037/0003-066X.48.1.35

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, *111*(4), 1036–1060. doi:10.1037/0033-295X.111.4.1036

Anderson, J. R., Boyle, C. F., Corbett, A. T., & Lewis, M. W. (1990). Cognitive modeling and intelligent tutoring. *Artificial Intelligence*, *42*(1), 7–49. doi:10.1016/0004-3702(90)90093-F

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. doi:10.3102/0013189X013006004

Bollen, L., & Van Joolingen, W. R. (2013). SimSketch: Multiagent simulations based on learner-created sketches for early science education. *IEEE Transactions on Learning Technologies*, *6*(3), 208–216. doi:10.1109/TLT.2013.9

Chi, M. T. H., Bassok, M., Lewis, M. W., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science*, *13*(2), 145–182. doi:10.1207/s15516709 cog1302_1

Chi, M. T. H., De Leeuw, N., Chiu, M.-H., & Lavancher, C. (1994). Eliciting self-explanations improves understanding. *Cognitive Science*, *18*(3), 439–477. doi:10.1207/s15516709cog1803_3

Chrysafiadi, K., & Virvou, M. (2013). Student modeling approaches: A literature review for the last decade. *Expert Systems with Applications*, *40*(11), 4715–4729. doi:10.1016/j.eswa.2013.02.007

Corbett, A. T., & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modelling and User-Adapted Interaction*, *4*(4), 253–278. doi:10.1007/BF01099821

De Jong, T., & Van Joolingen, W. R. (1998). Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research*, *68*(2), 179–201. doi:10.3102/00346543068002179

Desmarais, M. C., & Baker, R. S. J. d. (2012). A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction*, *22*(1-2), 9–38. doi:10.1007/s11257-011-9106-8

Flick, D. L. B. (1993). The meanings of hands-on science. *Journal of Science Teacher Education*, *4*(1), 1–8. doi:10.1007/BF02628851

Forbus, K., Usher, J., Lovett, A., Lockwood, K., & Wetzel, J. (2011). CogSketch: Sketch understanding for cognitive science research and for education. *Topics in Cognitive Science*, *3*(4), 648–666. doi:10.1111/j.1756-8765.2011.01149.x

Goldstone, R. L., & Son, J. Y. (2005). The transfer of scientific principles using concrete and idealized simulations. *Journal of the Learning Sciences*, *14*(1), 69–110. doi:10.1207/s15327809jls1401_4

Hagemans, M. G., Van der Meij, H., & De Jong, T. (2013). The effects of a concept map-based support tool on simulation-based inquiry learning. *Journal of Educational Psychology*, *105*(1), 1–24. doi:10.1037/a0029433

Hayes, N. A., & Broadbent, D. E. (1988). Two modes of learning for interactive tasks. *Cognition*, *28*(3), 249–276. doi:10.1016/0010-0277(88)90015-7

Hegarty, M., Kriz, S., & Cate, C. (2003). The roles of mental animations and external animations in understanding mechanical systems. *Cognition & Instruction*, *21*(4), 325–360. doi:10.1207/s1532690xci2104_1

Höffler, T. N., & Leutner, D. (2007). Instructional animation versus static pictures: A meta-analysis. *Learning and Instruction*, *17*(6), 722–738. doi:10.1016/j.learninstruc.2007.09.013

Jee, B. D., Gentner, D., Uttal, D. H., Sageman, B., Forbus, K., Manduca, C. A., … Tikoff, B. (in press). Drawing on experience: How domain knowledge is reflected in sketches of scientific structures and processes. *Research in Science Education*. doi:10.1007/s11165-014-9405-2

Khan, S. (2011). New pedagogies on teaching science with computer simulations. *Journal of Science Education and Technology*, *20*(3), 215–232. doi:10.1007/s10956-010-9247-2

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. doi:10.1207/s15326985ep4102_1

Leopold, C., & Leutner, D. (2012). Science text comprehension: Drawing, main idea selection, and summarizing as learning strategies. *Learning and Instruction*, *22*(1), 16–26. doi:10.1016/j.learninstruc.2011.05.005

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. doi:10.1037/0003-066X.59.1.14

Morgan, K., & Brooks, D. W. (2012). Investigating a method of scaffolding student-designed experiments. *Journal of Science Education and Technology*, *21*(4), 513–522. doi:10.1007/s10956-011-9343-y

Ploetzner, R., Lippitsch, S., Galmbacher, M., Heuer, D., & Scherrer, S. (2009). Students' difficulties in learning from dynamic visualisations and how they may be overcome. *Computers in Human Behavior*, *25*(1), 56–65. doi:10.1016/j.chb.2008.06.006

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. doi:10.1016/j.compedu.2011.07.017

Snowman, J., & Cunningham, D. J. (1975). A comparison of pictorial and written adjunct aids in learning from text. *Journal of Educational Psychology*, *67*(2), 307–311. doi:10.1037/h0076934

Svendsen, G. B. (1991). The influence of interface style on problem solving. *International Journal of Man-Machine Studies*, *35*(3), 379–397. doi:10.1016/S0020-7373(05)80134-8

Trey, L., & Khan, S. (2008). How science students can learn about unobservable phenomena using computer-based analogies. *Computers and Education*, *51*(2), 519–529. doi:10.1016/j.compedu.2007.05.019

Tversky, B., Morrison, J. B., & Betrancourt, M. (2002). Animation: Can it facilitate? *International Journal of Human-Computer Studies*, *57*(4), 247–262. doi:10.1006/ijhc.2002.1017

Van Berkum, J. A., & De Jong, T. (1991). Instructional environments for simulations. *Education and Computing*, *6*(3-4), 305–358.

Van der Meij, J., & De Jong, T. (2006). Supporting students' learning with multiple representations in a dynamic simulation-based learning environment. *Learning and Instruction*, *16*(3), 199–212. doi:10.1016/j.learninstruc.2006.03.007

Van Essen, G., & Hamaker, C. (1990). Using self-generated drawings to solve arithmetic word problems. *Journal of Educational Research*, *83*(6), 301–312.

Van Joolingen, W. R., & De Jong, T. (1991). Supporting hypothesis generation by learners exploring an interactive computer simulation. *Instructional Science*, *20*(5-6), 389–404. doi:10.1007/BF00116355

Van Joolingen, W. R., De Jong, T., Lazonder, A. W., Savelsbergh, E. R., & Manlove, S. (2005). Co-Lab: Research and development of an online learning environment

for collaborative scientific discovery learning. *Computers in Human Behavior*, *21*(4), 671–688. doi:10.1016/j.chb.2004.10.039

Van Meter, P. (2001). Drawing construction as a strategy for learning from text. *Journal of Educational Psychology*, *93*(1), 129–140. doi:10.1037/0022-0663.93.1.129

Van Meter, P., Aleksic, M., Schwartz, A., & Garner, J. (2006). Learner-generated drawing as a strategy for learning from content area text. *Contemporary Educational Psychology*, *31*(2), 142–166. doi:10.1016/j.cedpsych.2005.04.001

Van Meter, P., & Garner, J. (2005). The promise and practice of learner-generated drawing: Literature review and synthesis. *Educational Psychology Review*, *17*(4), 285–325. doi:10.1007/s10648-005-8136-3

VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, *46*(4), 197–221. doi:10.1080/00461520.2011.611369

Vogel, J. J., Vogel, D. S., Cannon-Bowers, J., Bowers, G. A., Muse, K., & Wright, M. (2006). Computer gaming and interactive simulations for learning: A meta-analysis. *Journal of Educational Computing Research*, *34*(3), 229–243. doi:10.2190/FLHV-K4WA-WPVQ-H0YM

Wieman, C. E., Adams, W. K., & Perkins, K. K. (2008). PhET: Simulations that enhance learning. *Science*, *322*(5902), 682–683. doi:10.1126/science.1161948

Zacharia, Z. C., & Anderson, O. R. (2003). The effects of an interactive computer-based simulation prior to performing a laboratory inquiry-based experiment on students' conceptual understanding of physics. *American Journal of Physics*, *71*(6), 618–629. doi:10.1119/1.1566427

Zacharia, Z. C., & Olympiou, G. (2011). Physical versus virtual manipulative experimentation in physics learning. *Learning and Instruction*, *21*(3), 317–331. doi:10.1016/j.learninstruc.2010.03.001

2

DRAWING-BASED SIMULATION FOR PRIMARY SCHOOL SCIENCE EDUCATION[1]

Touch screen computers are rapidly becoming available to millions of students. These devices make the implementation of drawing-based simulation environments like GearSketch possible. This study shows that primary school students who received simulation-based support in a drawing-based learning environment performed better than students who did not receive this support. Furthermore, the students who received this support did better on both direct and delayed posttests. These findings indicate that touch screen devices can be effectively used with drawing-based simulation environments to improve drawing-based primary school science education.

[1] This chapter is based on Leenaars, F., Van Joolingen, W., Gijlers, H., & Bollen, L. (2012). Drawing-based simulation for primary school science education: An experimental study of the GearSketch learning environment. In *2012 IEEE Fourth International Conference on Digital Game and Intelligent Toy Enhanced Learning (DIGITEL)* (pp. 1-8). doi:10.1109/ DIGITEL.2012.9

2.1 Introduction

In recent years, touch screen computers have become available to millions of people, in both private and educational settings. More than fifteen million tablets were sold in 2010 and over three times as many are expected to be sold in 2011 (The Economist, 2011). Over a million classrooms worldwide are currently equipped with interactive whiteboards (Lee, 2010). Schools are starting to buy tablets for their students and to explore possibilities for their use in the classroom (Hu, 2011). The availability of these touch screen computers offers opportunities for education that were not previously available.

This paper discusses an empirical study done with GearSketch, a drawing-based simulation environment for the gears domain, designed for use with a touch screen by primary school students. GearSketch is based on ideas from research on simulation-based learning and problem solving with drawings. Research on these topics, the choice for the gears domain, and our research questions and hypotheses are briefly discussed in the next subsections.

2.1.1 Simulation-based learning

A lot of research has been done on the use of simulations in education and a recent overview of this research shows that use of simulations can lead to improved learning outcomes (Rutten, Van Joolingen, & Van der Veen, 2012). At the core of a computer simulation is a model of a system or a process (De Jong & Van Joolingen, 1998). This model is used by the simulation to predict what will happen for given input parameters and allows students to explore hypothetical situations. Active exploration of many different situations allows students to gather information about the domain in a way that is not possible through traditional approaches such as learning with textbooks and lectures.

An important lesson from the research on simulation-based learning is that it is generally not effective if learners are not guided during their interaction with the learning environment (Mayer, 2004). An effective simulation-based learning environment offers learners questions to answer and goals to achieve instead of leaving them to discover concepts and rules on their own.

To work with simulations, learners often have to provide numerical input parameters and interpret simulation results in the form of tables and graphs. These activities require mathematical knowledge and abilities that primary school students do not yet possess.

During the design of GearSketch, we focused on creating a simulation-based learning environment that is easy to use and in which results can be interpreted without advanced

mathematical insight. The goal was to allow students to focus on learning about the domain instead of learning how to work with simulations.

2.1.2 Learning and problem solving with drawings

The creation of a drawing is a learning strategy that is suitable for learners from an early age. By making a drawing, learners externalize their knowledge and ideas, which can help them in multiple ways. For instance, it can help by making abstract ideas more concrete, stimulating self-explanation and facilitating mental animation (Cox, 1999). Creating a drawing can help students during both learning (Van Meter, Aleksic, Schwartz, & Garner, 2006) and problem solving (Van Essen & Hamaker, 1990), but is a strategy that has to be used carefully to be effective (Van Meter & Garner, 2005).

Paper-and-pencil drawings offer only static representations. Without review and feedback from teachers or peers, learners may not notice when their drawings contain incorrect assumptions or ideas. When objects are difficult to draw, more attention may be given to accurately representing these objects than to learning or reasoning about them. With touch screen computers becoming available in educational settings, these problems may be alleviated by supporting learners while they are drawing. For instance, pilot tests showed that students found it difficult to draw symmetric gears with paper and pencil. To prevent students spending too much time and energy precisely drawing each gear, GearSketch uses simple shape recognition to transform circles drawn by learners into gears of the same size. This means learners can spend more time on the important aspects of the gears domain and less time learning to draw nice gears.

2.1.3 Selection of the gears domain

The gears domain is well suited for learning about with a drawing-based simulation environment by primary school students for three main reasons.

First, the domain is inherently interesting for primary school education. Gears are (part of) everyday objects with which learners have experience from an early age. This means gears can be effectively used as reference objects for early mathematics education (Bartolini Bussi, Boni, Ferri, & Garuti, 1999). For instance, the turning direction of meshing gears can be used to introduce the concept of parity (Dixon & Bangert, 2004), and gear ratios can be used as a meaningful framework for learning about fractions (Andrade, 2009). Furthermore, gears can be used in early physics education to introduce such difficult concepts as mechanical advantage to young students (Chambers, Carbonaro, & Murray, 2008).

Second, the nature of the gears domain is both spatial and dynamic. The spatial nature of the domain means that drawings are an excellent way to represent different

configurations of gears and chains. Drawings of gears are representational, which means "the drawing is intended to look-like, or share a physical resemblance with the object(s) that the drawing depicts" (Van Meter & Garner, 2005, p. 288). Therefore, no extra transformational step is needed to draw a gear based on a mental image of a gear; a step which would be required for a non-spatial domain. The gears domain is also dynamic in nature. This means that time and motion play important roles and simulation based on the learner-generated drawings can give valuable insight into the behavior of gears and chains.

Third, even with only two types of objects (gears and chains) and relatively simple rules describing their interaction, complex systems can be created. This means that the goal is not for students to learn to reproduce the rules governing the domain, but to be able to apply these rules in both simple and complex systems. Students can be asked to apply these rules in two different ways: answering questions and solving puzzles. For example, students could be shown five meshing gears in a row and asked in which direction the rightmost gear would turn if the leftmost gear was turning clockwise. To answer this question successfully, students could repeatedly apply the rule that says that meshing gears will spin in opposite directions. Such a question could also be asked and answered with paper and pencil and students could check whether they answered correctly with a simple answer key. But solving and checking puzzles with paper and pencil is more difficult. If students were asked to connect two separate gears in such a way that they would spin in the same direction without moving them closer to each other, they could use the same knowledge they used to answer the previous question and add three meshing gears in a row to connect the outer gears. But this is not the only possible solution. Adding one large gear or a chain would also work. All these solutions can be explored and checked by learners in a drawing-based simulation environment.

2.1.4 Research questions and hypotheses

We expect that a drawing-based approach to learning about gears can benefit from the opportunities offered by computer simulation. Specifically, a simulation-based environment with an internal model of the gears domain can:

- Check the validity of gear and chain systems.
- Tighten hand-drawn chains around their supporting gears.
- Update chains when supporting gears are moved.
- Snap gears into place when they are moved close together, aligning their teeth correctly.
- Animate the turning of gears and chains.

The main question this study attempts to answer is: can simulation-based support in a

drawing-based environment lead to improved learning outcomes compared to a drawing-based approach without this support?

We expect simulation-based support will enable learners to both perform better in the learning environment and learn more during their use of this environment. However, they may learn less from answering the question items than from solving the puzzle items. Learners could use the simulation to see the behavior of systems to which questions refer instead of reasoning about the systems' behavior themselves, which may not be beneficial to learning. Using the simulation in this way is similar to using external animations to learn about mechanical systems. Prior research shows that animations often do not improve learning in this context, perhaps because they replace the need for active mental animation (Hegarty, Kriz, & Cate, 2003). Therefore, simulation-based support may not lead to improved performance on posttest questions.

For the puzzle items, however, the situation is different. Trying to solve puzzles in a simulation-based learning environment is a form of game-based learning rather than animation-based learning. Effective educational games give the player clear goals, control over the situation and immediate feedback (Kiili, 2005). Various kinds of support that a simulation-based learning environment can offer the learner, such as automatically updating chains when gears are moved in the case of GearSketch, give the learner more control by improving the user interface. Immediate feedback, which in the case of GearSketch consists of seeing the gears and chains turn, is believed to play a crucial role in supporting children's cognitive processes (Bottino, Ferlino, Ott, & Tavella, 2007). We expect learners in the simulation condition to be able to solve puzzles that learners in the static condition cannot, because with the simulation they can see why their proposed solution does or does not work. Without this feedback, learners may not notice when their solutions are incorrect and fail to correct their misconceptions. For these reasons, we expect that simulation-based support will lead to both improved performance on the puzzle items during instruction and improved learning outcomes.

These considerations lead to the following six hypotheses. Compared to students working in a static drawing-based learning environment, students who are working in a drawing-based learning environment with simulation-based support:

1. Perform better on questions during instruction.
2. Perform better on puzzles during instruction.
3. Perform equally on questions during a direct posttest.
4. Perform better on puzzles during a direct posttest.
5. Perform equally on questions during a delayed posttest.
6. Perform better on puzzles during a delayed posttest.

2.2 Method

2.2.1 Participants

Seventy-eight fifth grade students from a Dutch primary school initially participated in this study, but four of these students did not participate in the delayed posttest and were not included in the analysis. Seventy-four students (33 girls) remained, with a mean age of 11.31 years (SD = 0.37). Participants were randomly assigned to either the supported condition (N = 36, 14 girls, M_{age} = 11.33, SD_{age} = 0.38) or the static condition (N = 38, 19 girls, M_{age} = 11.30, SD_{age} = 0.37).

2.2.2 Material

Participants worked with a stylus-based touch screen in the GearSketch learning environment, which was developed specifically for this study. Ten Wacom Cintiq 12WX touch screens were connected to PCs in the participating school's computer room. The GearSketch software was run from USB drives plugged into each of the ten PCs. Logs of all the students' actions in this environment were automatically saved to these USB drives. Figure 1 shows a screenshot of the GearSketch learning environment.

Two versions of the GearSketch learning environment were created: a simulation-based and a static version. The static version of the GearSketch environment was created so

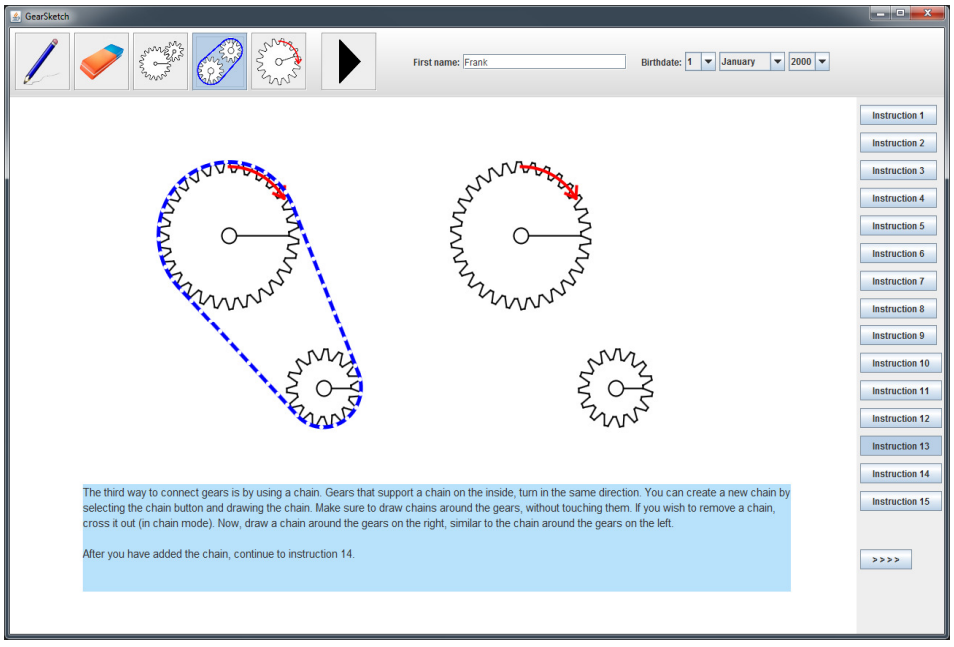


Figure 1. Screenshot of the learning environment in which participants worked.

that participants in the control group could also work with a touch screen computer instead of with paper and pencil. This was done to prevent finding differences between the groups only due to a novelty effect of the new learning environment. The shared properties of these two versions will be discussed first, followed by a description of the differences between these versions. Finally, the paper-and-pencil tests will be described.

2.2.2.1 General properties of the GearSketch environment

The GearSketch environment introduced a number of concepts and rules related to gears, chains and their connections. Specifically, the concepts of a gear's turning speed (rotational speed, in rotations per second) and tooth speed (linear speed of the gear's teeth, in meters per second) were introduced first. Next, three ways of connecting gears were discussed: meshing (teeth are connected), on top of each other (shared axes) and with a chain. For each type of connection the properties of the gears (turning direction, turning speed and tooth speed) were explained. These concepts and relations are summarized in Table 1.

To learn about these concepts, connections and rules, participants progressed linearly through the three phases of the learning environment: the instruction phase, the questions phase and the puzzles phase.

The instruction phase had two goals. The first goal was to introduce the gears domain to the students by defining relevant concepts and explaining the effects of connecting gears in different ways. The second goal was to familiarize students with the GearSketch environment. As each new concept was explained, participants actively used this concept by creating and connecting gears and chains.

In the second phase, participants answered thirteen multiple choice questions about the domain. Participants needed knowledge of the concepts and rules discussed in the first

Table 1
Concepts, connections and rules of the gears domain

Connection type	Turning direction	Turning speed	Tooth speed
Meshing gears	Opposite	Negatively related to size	Equal
Gears on top of each other (sharing an axis)	Equal	Equal	Positively related to size
Gears supporting a chain (same side)	Equal	Negatively related to size	Equal
Gears supporting a chain (opposite side)	Opposite	Negatively related to size	Equal

phase to answer these questions. The questions were accompanied by pointers to relevant parts of the instructions, to encourage students that were unsure about their knowledge to look at the instructions again.

In the final phase, participants were asked to solve nine puzzles of progressing difficulty. To solve these puzzles, participants had to correctly use gears and chains to satisfy specific objectives, such as making two gears turn in opposite directions or with different speeds.

Both versions of the GearSketch environment had the same graphical user interface, except that the static version did not have a play button. The interface was designed to be as intuitive as possible. Using buttons at the top of the screen, participants could choose from five different modes: pencil, eraser, gear, chain and arrow. Participants could draw and remove free pencil strokes in the pencil and eraser modes to, for instance, predict turning directions of gears and chains. In gear mode, participants could add, remove and move gears. Adding gears was done by drawing a circle, which GearSketch automatically transformed into a gear in the specified location with the specified size. Removing gears was done by crossing them out. Gears could be moved by touching and dragging them with the stylus. In chain mode, chains could be added by drawing them and removed by crossing them out. In arrow mode, arrows could be added to gears to indicate that they should turn in the specified direction, with the indicated speed. Arrows could be removed by tapping the top of the gear with the stylus.

2.2.2.2 Differences between the simulation-based and static version of GearSketch

The most salient feature of the simulation-based version was that it could animate the turning of the gears and chains, whereas the static version could not. However, this distinction is a result of a more fundamental difference between the versions, an understanding of which will explain all other differences.

Both versions of GearSketch allowed the user to create gears by simply drawing a circle, but whereas that was approximately the extent of the static version's 'knowledge' of the domain, the simulation-based version's 'understanding' was deeper. The simulation-based version kept track of an elaborate internal model of the system of gears and chains that was displayed. It modeled which gears were connected to each other and in which ways. This model allowed it to support the user in various ways. For instance, gears that were moved close to each other automatically snapped together, their teeth aligned correctly. A chain that was loosely drawn around two or more gears, was automatically tightened around them, as if it were an elastic string (see Figure 2). Because gears could also be connected by placing them on top of each other, gears and chains could overlap without actually touching, because of their location in different

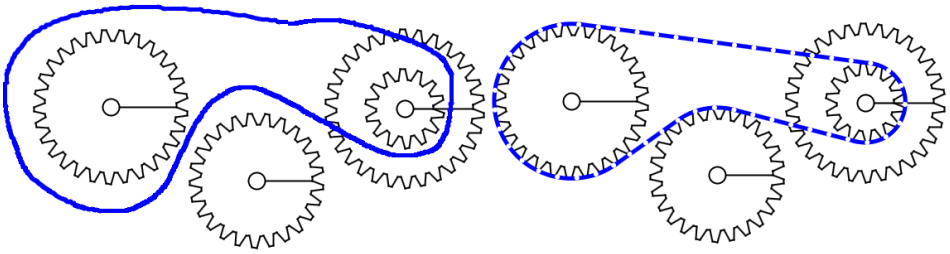


Figure 2. A hand-drawn chain before and after it is tightened by the software.

layers. The internal model of the simulation-based version kept track of these layers, which allowed users to create and animate different kinds of transmission systems.

The simulation-based version always kept its internal model in a valid state, which means that it didn't allow immovable configurations to be created. For instance, it was not possible to connect three gears together in such a way that they could not turn. The gear that the participant was trying to move would turn red, with a gray outline showing the last valid location of this gear. If the participant lifted the stylus to place the gear in its new, invalid, location, the gear would automatically be put back in the last valid location, as indicated by the gray outline.

All these ways of supporting the user are only possible if the learning environment keeps track of a precise, domain specific, internal model.

2.2.2.3 Paper-and-pencil tests

Two paper-and-pencil tests were created to test the participants' knowledge about the concepts, connections and rules treated in the GearSketch environment. These tests each consisted of fifteen multiple choice items and five puzzle items. The tests were constructed by designing two similar but different versions of each item and randomly assigning one version of the item to each test. The goal of creating these pairs of items instead of using the same test twice was to attenuate retest effects.

2.2.3 Procedure

Participants were randomly assigned to either the simulation-based or the static condition. Participants worked individually in the computer room, ten at a time, under the experimenter's supervision. Before they started the students were instructed that they were participating in a study of a new way of learning about gears. They were told not to talk to each other during the experiment, but that they could ask questions by raising their hand at any time. They also typed in their first name and selected their date of birth in the GearSketch software, so that this information could be saved in the automatically created log files.

After this short introduction, participants worked with the GearSketch software for 60 minutes. Questions that participants asked during this time were answered by pointing them in the direction of relevant parts of the instruction. Participants who completed all phases of the GearSketch environment before the 60 minutes had passed, were told to quietly work for themselves in a final 'puzzle'. This puzzle had no specific objective and allowed participants to create whatever configuration of gears, chains and free pencil strokes they wished.

When participants had worked in the GearSketch environment for 60 minutes, their touch screens were switched off and they were given a paper-and-pencil posttest. They were told this test would not count towards their grade, but that it was important for the study that they did their best on the test. They had 20 minutes to complete the test.

Two weeks after working in the GearSketch environment and completing the direct posttest, participants made the second, delayed posttest. They were again informed that this test would not count towards their grade, but that it was important for the study that they did their best.

2.2.4 Analysis

Six sets of items, related to the six hypotheses, were scored for each participant:

1. Their answers to the questions during the instruction.
2. Their solutions to the puzzles during the instruction.
3. Their answers to the questions on the direct posttest.
4. Their solutions to the puzzles on the direct posttest.
5. Their answers to the questions on the delayed posttest.
6. Their solutions to the puzzles on the delayed posttest.

All questions were multiple choice and were simply scored 1 point for a correct answer and 0 points for an incorrect answer. The solutions to the puzzles in the GearSketch environment were scored with the help of a log file replayer. With this tool the log files of each participant for each of the puzzles were played back. If the participant had correctly solved the puzzle at any point in time, the participant was awarded 1 point for this item and 0 points otherwise. The puzzles on the posttests were also scored 1 point for a correct solution and 0 points for an incorrect solution. The only exception was the final puzzle of each posttest, which in effect integrated two different puzzles and which was scored either 0, 1 or 2 points depending on the correctness of the solution.

To find out if there was a difference between the overall scores of participants in the animated and the static condition, these six scores for each participant were entered into a MANOVA (Stevens, 2002). If the MANOVA indicated a significant main effect of

condition, t-tests would be used to examine differences on particular scores. Because we hypothesized that the scores on the questions and puzzles during the instruction and the puzzles during the two posttests would be higher for the participants in the simulation-based condition, these scores were compared with one-tailed t-tests. We had no directed hypotheses about a difference in the question scores on the posttests, so these were compared with two-tailed t-tests.

2.3 Results

There was a significant effect of condition, Wilk's $\lambda = .016$, $F(6, 67) = 678$, $p < .001$. Because the MANOVA showed the effect of condition was significant, we also compared specific scores.

Table 2 lists the average scores and standard deviations for participants in both groups on each set of items. Cohen's *d* (Cohen, 1988), a measure of the size of the difference between the groups' means is also given. Participants in the simulation-based condition scored significantly higher than participants in the static condition on the instruction questions ($t = 4.78$, $d = 1.10$, $p < .001$), the instruction puzzles ($t = 7.63$, $d = 1.75$, $p < .000$), the direct posttest questions ($t = 2.93$, $d = 0.67$, $p = .004$), the direct posttest puzzles ($t = 2.49$, $d = 0.57$, $p = .008$), the delayed posttest questions ($t = 4.20$, $d = 0.99$, $p < .001$) and the delayed posttest puzzles ($t = 2.36$, $d = 0.56$, $p = .011$).

Table 2

Mean scores and standard deviations for participants in both groups on the six sets of items

Items	Max. score	Simulation-based condition		Static condition		Cohen's *d*
		M	*SD*	*M*	*SD*	
Instruction questions	14	8.49	(1.70)	6.38	(2.16)	1.10
Instruction puzzles	9	6.00	(1.73)	3.26	(1.43)	1.75
Direct posttest questions	15	8.87	(1.78)	7.74	(1.62)	0.67
Direct posttest puzzles	6	2.54	(1.32)	1.74	(1.50)	0.57
Delayed posttest questions	15	8.69	(1.51)	7.21	(1.53)	0.99
Delayed posttest puzzles	6	2.78	(1.59)	1.92	(1.53)	0.56

2.4 Discussion

The results confirm the first two hypotheses stated in the introduction: participants in the simulation-based condition performed better on both the questions and puzzles during the instruction than those in the static condition. The large effect size for the instruction items shows that the support that the participants in the simulation-based condition received made it easier to solve these items. To see whether the support also led to higher learning gains, we examined the mean score differences on the two posttests.

The fourth and sixth hypotheses were also confirmed: participants in the simulation-based condition scored better on the puzzle items on both posttests. The effect size for these items is smaller than for the instruction items, which can be explained by the fact that participants in the simulation-based condition received no extra support during the posttests. Therefore, the difference in the mean scores cannot be explained by the items being easier; it is only explained by higher learning gains.

As can be seen from the third and fifth hypotheses, we had no directed expectation about a difference in the results on the question items of both posttests. However, the results show that participants in the simulation-based condition did better on the question items in both posttests than participants in the static condition. The effect sizes fall between the effect sizes of the posttest puzzles and those of the instruction items.

The main research question can be answered affirmatively: simulation-based support can lead to improved learning outcomes in a drawing-based learning environment. Furthermore, the relative difference between the simulation-based and the static condition, as indicated by Cohen's *d*, has not diminished in the two weeks after the instruction. According to Cohen's criteria (Cohen, 1988), the effect size is medium or large for all six comparisons.

Due to previous findings regarding the ineffectiveness of animations in science education (Hegarty et al., 2003), we did not predict participants in the simulation-based condition would score higher on the posttest question items. However, there are plausible explanations for this finding. The simulation, and thus the animation, was interactive even when the system could not be modified. Learners could start and stop the animation, allowing them to predict the system's behavior before watching the animation to see what would happen. This interactivity may make the animation more effective (Tversky, Morrison, & Betrancourt, 2002). Another possibility is that the knowledge that was tested by the question items on the posttest was gained not only from answering the question items during the instruction, but from solving the puzzle items as well. In fact this is quite likely, since both questions and puzzles test participants' abilities to apply the same rules, albeit in different ways. Because the

participants in the simulation-based condition did much better on the instruction puzzles than those in the static condition (d = 1.75), it is not unlikely that they have learned more from solving these puzzles.

Future research could focus on offering further support for learners based on the simulation environment's internal model. For instance, it is possible to automatically deduce for a given puzzle, which concepts and rules a learner has to understand to solve it. If students are unable to solve a puzzle, the learning environment could select or generate a simpler puzzle based on the same concepts and rules. In this way the learning environment could automatically adapt to learners' strengths and weaknesses.

Another possible direction for future research would be to develop drawing-based simulation environments for different domains. Our current knowledge suggests that this type of learning environment would be suitable for many dynamic spatial domains. Perhaps ideas from software such as CogSketch (Forbus, Usher, Lovett, Lockwood, & Wetzel, 2011), a domain general sketch understanding system, could be used to create a drawing-based simulation environment that supports multiple domains.

This study has shown that the new possibilities offered by the availability of touch screen computers can be effectively used to improve drawing-based primary school education about the gears domain. Simulation-based support improves students' performance during instruction and increases learning outcomes both directly after the instruction and two weeks later. Future research may examine how learners can be even better supported and whether simulation-based learning environments can also be effectively used in other primary school science domains.

References

Andrade, A. (2009). The clock project: Gears as visual-tangible representations for mathematical concepts. *International Journal of Technology and Design Education*, *21*(1), 93–110. doi:10.1007/s10798-009-9104-x

Bartolini Bussi, M. G., Boni, M., Ferri, F., & Garuti, R. (1999). Early approach to theoretical thinking: Gears in primary school. *Educational Studies in Mathematics*, *39*(1-3), 67–87. doi:10.1023/A:1003707727896

Bottino, R. M., Ferlino, L., Ott, M., & Tavella, M. (2007). Developing strategic and reasoning abilities with computer games at primary school level. *Computers & Education*, *49*(4), 1272–1286. doi:10.1016/j.compedu.2006.02.003

Chambers, J. M., Carbonaro, M., & Murray, H. (2008). Developing conceptual understanding of mechanical advantage through the use of Lego robotic technology. *Australasian Journal of Educational Technology*, *24*(4), 387–401.

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences*. Hillsdale, NJ:

Psychology Press.

Cox, R. (1999). Representation construction, externalised cognition and individual differences. *Learning and Instruction*, *9*(4), 343–363. doi:10.1016/S0959-4752(98)00051-6

De Jong, T., & Van Joolingen, W. R. (1998). Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research*, *68*(2), 179–201. doi:10.3102/00346543068002179

Dixon, J. A., & Bangert, A. S. (2004). On the spontaneous discovery of a mathematical relation during problem solving. *Cognitive Science*, *28*(3), 433–449. doi:10.1207/s15516709cog2803_6

Forbus, K., Usher, J., Lovett, A., Lockwood, K., & Wetzel, J. (2011). CogSketch: Sketch understanding for cognitive science research and for education. *Topics in Cognitive Science*, *3*(4), 648–666. doi:10.1111/j.1756-8765.2011.01149.x

Hegarty, M., Kriz, S., & Cate, C. (2003). The roles of mental animations and external animations in understanding mechanical systems. *Cognition & Instruction*, *21*(4), 325–360. doi:10.1207/s1532690xci2104_1

Hu, W. (2011, January 4). More schools embrace the iPad as a learning tool. *The New York Times*. Retrieved from http://www.nytimes.com/2011/01/05/education/05tablets .html

Kiili, K. (2005). Digital game-based learning: Towards an experiential gaming model. *The Internet and Higher Education*, *8*(1), 13–24. doi:10.1016/j.iheduc.2004.12.001

Lee, M. (2010). Interactive whiteboards and schooling: The context. *Technology, Pedagogy and Education*, *19*(2), 133–141. doi:10.1080/1475939X.2010.491215

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. doi:10.1037/0003-066X.59.1.14

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. doi:10.1016/j.compedu.2011.07.017

Stevens, J. (2002). *Applied multivariate statistics for the social sciences*. Mahwah, NJ: Lawrence Erlbaum.

The Economist. (2011, March 2). Tablet computers: Taking the tablets. Retrieved September 2, 2011, from http://www.economist.com/blogs/dailychart/2011/03/tablet_computers

Tversky, B., Morrison, J. B., & Betrancourt, M. (2002). Animation: Can it facilitate? *International Journal of Human-Computer Studies*, *57*(4), 247–262. doi:10.1006/ijhc.2002.1017

Van Essen, G., & Hamaker, C. (1990). Using self-generated drawings to solve arithmetic word problems. *The Journal of Educational Research*, *83*(6), 301–312.

Van Meter, P., Aleksic, M., Schwartz, A., & Garner, J. (2006). Learner-generated drawing as a strategy for learning from content area text. *Contemporary Educational Psychology*, *31*(2), 142–166. doi:10.1016/j.cedpsych.2005.04.001

Van Meter, P., & Garner, J. (2005). The promise and practice of learner-generated drawing: Literature review and synthesis. *Educational Psychology Review*, *17*(4), 285–325. doi:10.1007/s10648-005-8136-3

3

GEARSKETCH: AN ADAPTIVE DRAWING-BASED LEARNING ENVIRONMENT FOR THE GEARS DOMAIN[1]

GearSketch is a learning environment for the gears domain, aimed at students in the final years of primary school. It is designed for use with a touchscreen device and is based on ideas from drawing-based learning and research on cognitive tutors. At the heart of GearSketch is a domain model that is used to transform learners' pen strokes into gears and chains, animate the turning of the gears and check whether learners' solutions to practice problems satisfy the given constraints. Additionally, this domain model is the basis for GearSketch's learner model and item generation and selection mechanisms. The learner model is used to track learners' knowledge and adaptively select items as they progress through the practice problems. An experiment fails to show a significant effect of adaptive item selection on learning outcome, possibly because the learner model's predictions of student performance are not sufficiently accurate.

[1] This chapter is based on Leenaars, F. A. J., Van Joolingen, W. R., Gijlers, H., & Bollen, L. (2014). GearSketch: An adaptive drawing-based learning environment for the gears domain. *Educational Technology Research and Development, 62*(5), 555–570. doi:10.1007/s11423-014-9345-6

3.1 Introduction

Ever since computers found their way into education, they have been used to support learning how to solve problems. Among the famous examples of problem-solving tutors are cognitive tutors based on ACT* and ACT-R theory (Anderson, Boyle, Corbett, & Lewis, 1990; Anderson et al., 2004), such as the LISP tutor, and algebra and geometry tutors (Anderson, Corbett, Koedinger, & Pelletier, 1995), as well as Andes, for learning to solve physics problems (VanLehn et al., 2005). Such tutors have in common their modeling of a domain with a set of well-defined rules, such as geometry proofs, in terms of declarative and procedural knowledge. These cognitive tutors have been quite successful in increasing performance on knowledge tests. In a different strategy, called constraint-based tutoring (Mitrovic, 2003; Mitrovic, Mayo, Suraweera, & Martin, 2001), the focus is not on arriving at one single, perfect solution to a problem, but on fulfilling the domain's constraints, which may lead to multiple different, equally correct solutions. Here, the domain description, learner model and feedback mechanism are based on constraint matching, error recognition and error correction. By modeling both the domain and the learner, cognitive tutors can adaptively select practice problems for each individual learner (Mitrovic & Martin, 2004). Instead of offering the same static sequence of problems to all students, the cognitive tutor can select appropriately challenging problems for each learner individually, based on its model of their current knowledge. Mitrovic and Martin (2004) found that this can have a positive effect on learning performance, especially for students who are below or above average ability.

Another approach to supporting complex problem-solving that until recently fell outside the scope of computer-based tutoring is the creation of drawings. Creating a drawing helps learners self-explain problems while working on them (Cox, 1999), can help when solving arithmetic word problems (Van Essen & Hamaker, 1990), and supports scientific reasoning (Ainsworth, Prain, & Tytler, 2011) and the modeling process (Leenaars, van Joolingen, & Bollen, 2013). The increasing availability of pen-based and touchscreen devices in (primary) schools (Hu, 2011; Lee, 2010), makes it possible to combine ideas from technology-enhanced learning and drawing-based learning. Two examples of digital learning environments that interpret students' drawings are CogSketch and SimSketch. CogSketch (Forbus, Usher, Lovett, Lockwood, & Wetzel, 2011) uses open-domain sketch understanding to reason about the drawings that learners make and can give feedback based on this understanding. SimSketch (Bollen & Van Joolingen, 2013) allows learners to explore the behavior of models that they construct themselves by drawing. The increasing availability of touch-based computers also allows new learning environments to combine ideas from cognitive tutors and learning by drawing. Such an environment adopts from drawing-based learning the idea of offering an expressive interface in which students can easily represent the systems they

are reasoning about by making a quick sketch. From cognitive tutoring, this environment adopts modeling individual learners' knowledge of the domain and using these learner models to adapt the tutoring process to each individual learner.

Such a learning environment is well suited for supporting learning to solve problems in a domain with well-defined rules and a natural graphical representation. The gears domain has both. Furthermore, because young learners are familiar with gears from everyday experience (e.g. bicycles, wind-up toys, clocks), gears are suitable as reference objects during the introduction of abstract concepts in mathematics and physics (Bartolini Bussi, Boni, Ferri, & Garuti, 1999). In previous studies, gears have been used to introduce parity (Dixon & Bangert, 2004), fractions (Andrade, 2009) and mechanical advantage (Chambers, Carbonaro, & Murray, 2008). Although the rules governing the gears domain are simple, it is easy to create gear configurations in which gears interact in complex ways. Additionally, when the goal for students is to arrange gears in such a way that given constraints are satisfied, as well as to predict how a gear configuration will behave, the task becomes even more challenging. Besides having to know the rules governing interactions between gears, students must now also recognize which rules are potentially useful and figure out how to deal with the spatial constraints inherent in any concrete configuration of gears. Taken together, these challenges involved in creating a gear configuration that satisfies certain constraints constitute a complex problem that does not have a single, exact solution.

This chapter discusses the development of GearSketch, an adaptive drawing-based learning environment for the gears domain, aimed at primary school students and designed to be used with a pen-based touchscreen. The development questions addressed are:

- DQ1. What kind of problems should students learn to solve by working with GearSketch and how can the required domain knowledge be modeled?
- DQ2. How can ideas from drawing-based problem solving be used to design a simple interface for practicing with these problems?
- DQ3. How can ideas from cognitive tutors be used to design a learner model that tracks students' knowledge as they progress through practice problems?
- DQ4. How can suitable practice problems be automatically generated and selected?

Additionally, an experimental study was conducted to address the following evaluation questions:

- EQ1. Does tracking individual students' knowledge with a learner model and using this model to select appropriate practice problems help students learn more from working with GearSketch?

- EQ2. Does the learner model make accurate predictions about students' performance on a posttest?

The next section introduces GearSketch and addresses the development questions. Subsequent sections discuss an experimental evaluation of GearSketch's learning model and its item selection features.

3.2 GearSketch

GearSketch is software that supports students in the final years of primary school in learning to solve problems in the gears domain. When learners work with GearSketch, they begin by working through a series of integrated tutorials that explain the relevant domain theory and teach them to work with the software. For example, when students are learning about the turning directions of gears connected via chains, the tutorial first explains that gears on the same side of a chain will turn in the same direction, while gears on opposite sides will turn in opposite directions. Next, students draw a chain that connects gears to one another. Then they are asked to predict the directions in which the gears connected by the chain will turn, if one of them is turning in a given direction. Finally, they can check whether their predictions were correct by watching an animation of the gears turning. After finishing the tutorial, students apply and strengthen their domain knowledge by answering questions and solving puzzles.

The next two sections discuss GearSketch's domain model and how this is used by the interface. Two subsequent sections discuss GearSketch's learner model and item generation features.

3.2.1 Domain model

The domain of gears and chains is governed by rules that together make up GearSketch's domain model. These rules define the interaction between gears and chains in two ways. First, they define the relative speeds with which connected gears turn given their size ratio and connection type. Qualitative versions of these rules are listed in Tables 1 and 2. Second, they define constraints on possible spatial layouts. For instance, two meshing gears may not also be connected by a chain, because then they would be unable to turn.

Table 1 summarizes how gears transmit motion for four different connection types. In this table "turning speed" refers to the angular velocity of the gear and "tooth speed" refers to the linear velocity of the gear's teeth. For each connection type, the connected gears will have either equal turning speed or equal tooth speed. When two connected gears' equal speed type is known and the relative size of the gears is also known,

Table 1

Rules relating connection type with turning direction and speed

Connection type	Turning direction	Equal speed
Meshing gears	Opposite	Tooth speed
Gears on top of each other (sharing an axis)	Equal	Turning speed
Gears supporting a chain (same side)	Equal	Tooth speed
Gears supporting a chain (opposite side)	Opposite	Tooth speed

Table 2

Rules relating gear size, turning speed and tooth speed

Relative sizes	TuS(X)[a] = TuS(Y)	ToS(X)[b] = ToS(Y)
Size(X)[c] = Size(Y)	ToS(X) = ToS(Y)	TuS(X) = TuS(Y)
Size(X) > Size(Y)	ToS(X) > ToS(Y)	TuS(X) < TuS(Y)

[a] The turning speed of gear X
[b] The tooth speed of gear X
[c] The size of gear X

conclusions can be drawn about the other type of speed for the two gears. Table 2 summarizes the relevant rules. Students who can reproduce the rules in Tables 1 and 2 from memory have declarative knowledge of these rules. However, using this information to solve problems requires procedural knowledge, in the form of production rules (Anderson et al., 2004). For each declarative rule, multiple production rules can be created. For instance, two production rules for the turning directions of meshing gears are:

1. IF gear A is turning clockwise and gears A and B mesh

   THEN gear B will turn counterclockwise.

2. IF gear A is turning clockwise and gear B should turn counterclockwise

   THEN this can be achieved by connecting gears A and B so that they mesh.

The first of these rules is the type used to explain the behavior of a gear configuration and to predict what will happen if changes are made. The second rule is the type used to create configurations that satisfy given constraints. Figure 1 shows a question item that can be solved by applying the first type of production rule, which follows quite directly from the declarative rules. The steps a student must take to answer this question can be summarized as follows, where TuD(X) means “turning direction of X”:

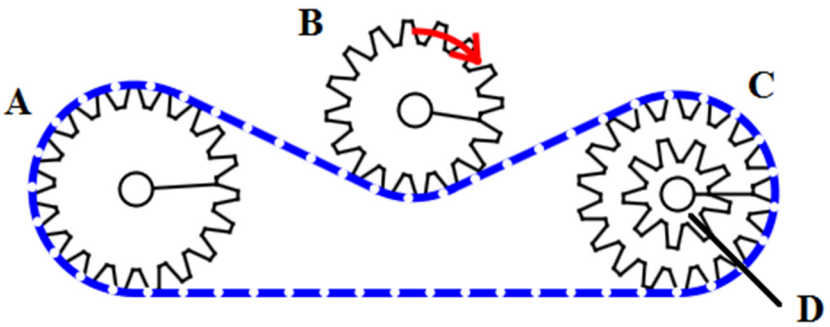


Figure 1. Example question: "Will gear D turn clockwise or counterclockwise?"

1. TuD(B) = clockwise [given]
2. TuD(B) ≠ TuD(C) [turning direction of gears on opposite sides of a chain]
3. TuD(C) = counterclockwise [follows from step 1 and 2]
4. TuD(C) = TuD(D) [turning direction of gears connected via their axes]
5. TuD(D) = counterclockwise [follows from step 3 and 4]

If the student knows the two relevant declarative rules and applies them correctly, finding the answer is straightforward. This is not the case for puzzle items, where recognizing which rules are relevant is more difficult and insight into the spatial properties of the task is necessary to find a solution. To solve the puzzle shown in Figure 2, students need to know the same two rules required for answering the question in Figure 1. Additionally, they need to recognize that these are the relevant rules for solving this problem, place gears B and C in correct locations and correctly connect the gears with a chain. GearSketch needs to offer students who are learning to solve puzzles like this a simple way to represent and examine multiple possible solutions. To achieve this, GearSketch has a computational model of the domain rules as summarized in Table 1 and Table 2, as well as of the ways in which gears can be configured spatially. This computational domain model is the basis for GearSketch's drawing-based interface that interprets and animates learners' drawings.

3.2.2 Interface

Figure 3 shows a screenshot of the GearSketch interface. In the upper left corner there are five large buttons for the different tools the learner can use to add and remove annotations, gears, chains and turning arrows. The sixth button is the play button, which starts the simulation. The three buttons in the upper right corner are used to reset a puzzle to its original state, check the learner's solution and go to the help menu, in order

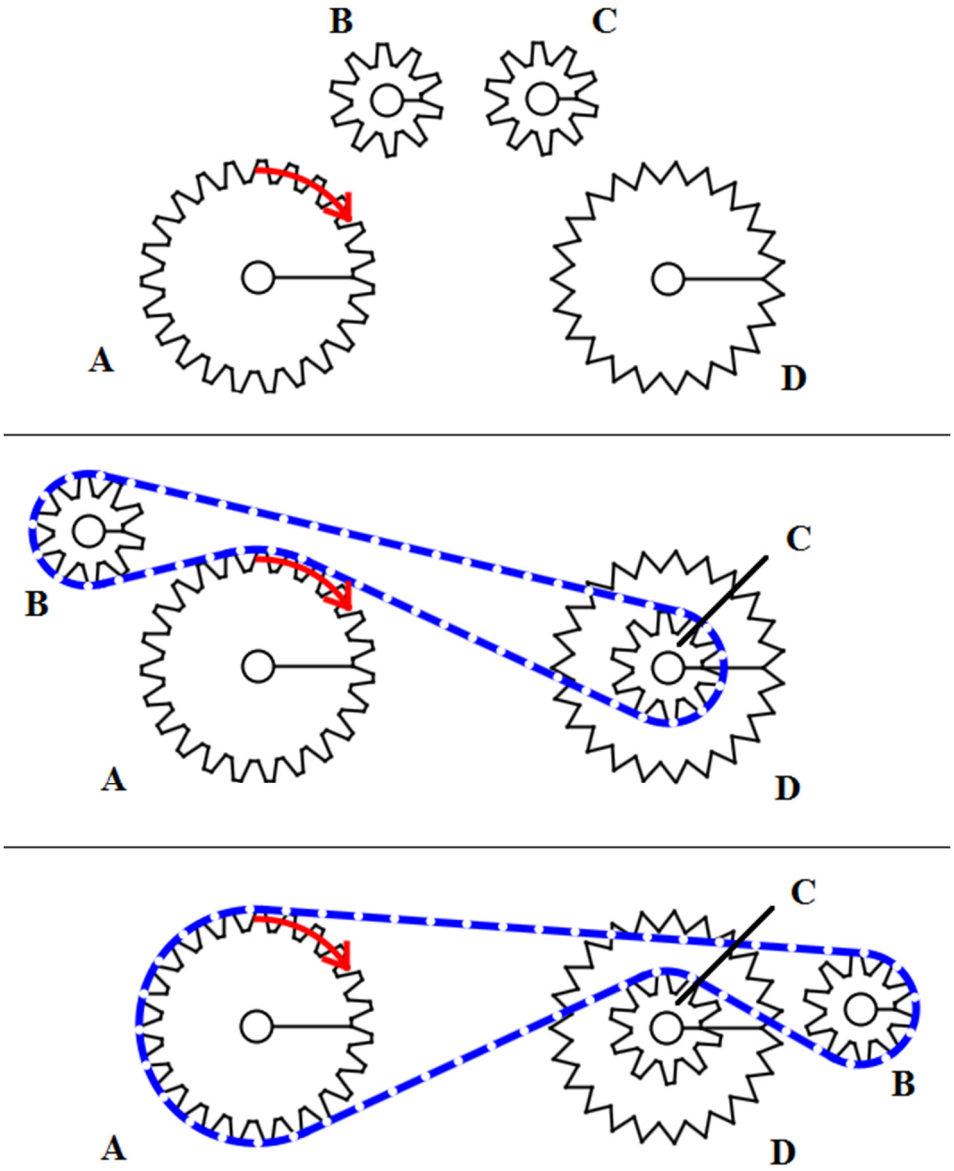


Figure 2. Example puzzle: "Add a chain so that gear D will turn counterclockwise. Only gear B and C can be moved. Gear D's sharp teeth mean that they cannot be connected to a chain or another gear's teeth." Two possible solutions are shown.

from left to right. The help menu gives an overview of the declarative rules in Table 1 and Table 2. The large area in the middle is for drawing the gears and chains, and the blue box at the bottom of the screen can contain questions or instructions for the puzzles.

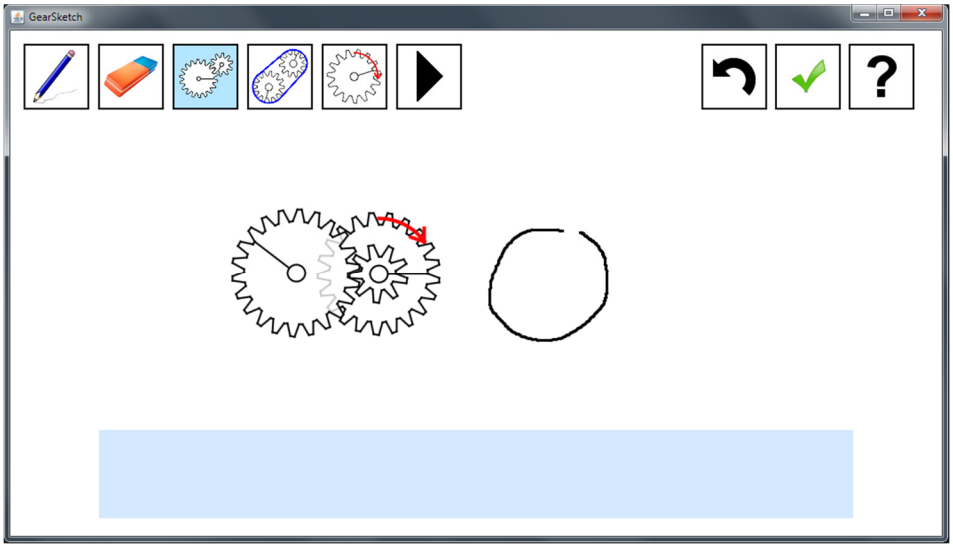


Figure 3. Screenshot of the GearSketch interface showing a new gear being drawn.

The interface is designed to be intuitive to use with a pen-based touchscreen. Gears are added by simply drawing a circle while the gears button is selected. A gear the same size as the circle will appear in its place when the stylus is lifted from the screen. Gears can be removed by crossing them out and be moved by dragging them. When a gear is dragged close to another gear it will snap to connect to the other gear, with its teeth aligned correctly. This also works when a gear is used to connect two other gears to each other. A smaller gear can be dragged on top of a larger gear and will then snap to connect to its axis. This means that gears can overlap when they are on different levels, as can be seen in the configuration on the left side of Figure 3. Because the gears are slightly transparent, underlying gears can still be seen.

A chain is added by selecting the chain tool and drawing its outline. GearSketch will automatically tighten the chain around its supporting gears. Automatically tightening the outline of a chain is a problem of finding the shortest homotopic path (Bespamyatnikh, 2003), with some added complexity due to the different levels of gears. When a chain is drawn through a gear, the chain could be invalid, but the learner could also intend the chain to be on a different level than the gear it is drawn through. Figure 4 shows how GearSketch interprets and tightens a chain in a situation with multiple levels of gears. Once a chain has been added, it can be removed by crossing it out. When supporting gears are moved or other gears are moved into the chain, the chain will react elastically, growing and shrinking as necessary.

When learners are moving and adding gears and chains, GearSketch uses its knowledge of the spatial constraints in the domain model to ensure that no invalid configurations

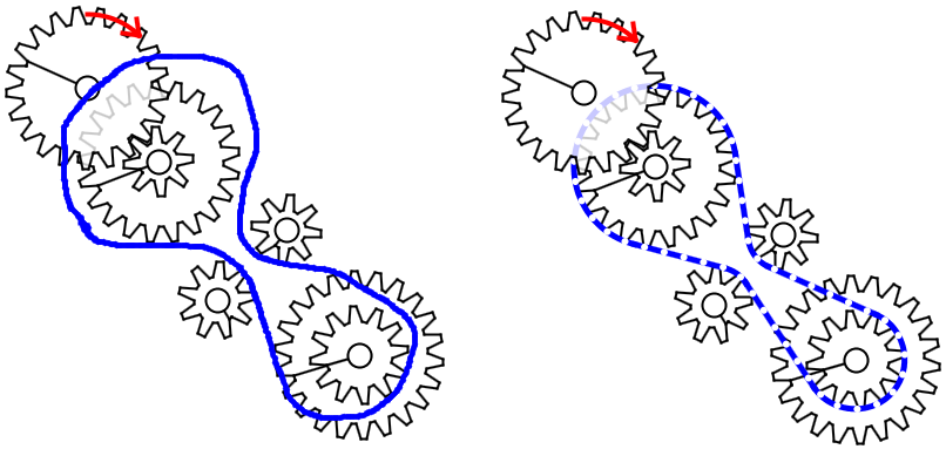


Figure 4. A sketched chain outline on the left and the result of applying GearSketch's tightening algorithm on the right.

are created. A configuration is considered invalid when its gears cannot turn, such as when three meshing gears are connected circularly or when two meshing gears are connected by a chain. GearSketch also makes sure that gears and chains that are on the same level do not overlap, while gears and chains on different levels can. The level objects are on is determined by analyzing the connections between them. The complexity of the algorithms that ensure the validity of configurations created by learners is directly related to the complexity in the domain model. The rules that determine the validity of a given gear configuration are not found in Table 1 or Table 2, but are also part of the domain model; learners need to know these rules to create their own gear and chain systems to solve puzzles in this domain.

When the play button is selected, GearSketch uses its knowledge of the domain model and of how the gears are interconnected to animate the model. Gears with a red arrow in them, which learners can add with the arrow tool, will start to turn. All gears and chains connected to these moving gears will also simultaneously start turning, according to the rules of the domain model. This allows the learner to explore the behavior of the current gear configuration and to see what happens when changes are made. The first time learners attempt to answer a question or solve a puzzle, the play button will be disabled. This means that they must reason about the behavior of the system to predict what will happen rather than being able to just press the play button and see it. If learners do not solve the problem on their first try, the play button is then enabled, so they can explore why their solution was incorrect.

These interface features provide the learner with different types of feedback. By aligning gears' teeth and tightening chains, GearSketch ensures that the configurations

learners create are valid and represented unambiguously. Creating a gear configuration in this way is a form of externalization (Cox, 1999) during which learners are assisted with the help of GearSketch's domain model. The second type of feedback is the animation of the gears when the play button is pressed. A recent meta-analysis (Höffler & Leutner, 2007) shows that this type of dynamic representational visualization leads to higher learning outcomes than static visualizations.

3.2.3 Learner model

GearSketch continuously updates a Bayesian network model (Koller & Friedman, 2009) of each student's domain knowledge as they answer questions and attempt to solve puzzles. Knowledge tracing (Corbett & Anderson, 1995) is used to update the estimates of individual learners' knowledge of each of the rules and of their ability to recognize each rule's applicability in solving puzzles. Figure 5 shows the structure of a small part of the Bayesian network that is used to model learners' knowledge. The learner model contains three nodes: Q, R and P, for each declarative rule in Table 1 and Table 2. The Q node indicates the learner's ability to apply the rule in a question context, which follows in a straightforward way from having declarative knowledge of the rule. The R node indicates the learner's ability to recognize the relevance of this rule when solving a puzzle. The P node, the value of which depends on both the Q and the R nodes, indicates the learner's ability to apply the rule in a puzzle context. Each node is assigned a value between 0 and 1, which represents the probability that a learner has the ability to which this node refers. When the values of the parent nodes (those nodes with arrows pointing to the current node) are known, the value of the current node can be calculated automatically. For instance, the probability that a student is able to apply a rule in a

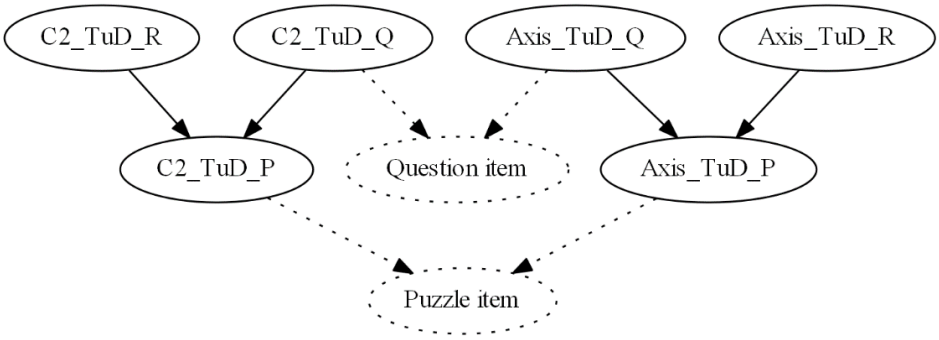


Figure 5. The structure of a small part of the learner model. C2 and Axis refer to connection types between gears, connected by a chain and through their axes, respectively. The R suffix refers to the learner's ability to recognize the relevance of a rule in a puzzle context and the Q and P suffixes refer to the learner's ability to use the rule in a question and a puzzle context, respectively.

puzzle context (P node), can be calculated based on the probability that the learner is able to apply this rule in a question context (Q node) and the probability that the learner is able to recognize the relevance of this rule (R node). The initial values for the parent nodes were based on results from the experiment discussed in Chapter 2 that indicated which rules students found it easy or difficult to grasp.

When a learner completes an item, this item is temporarily added to the network with the appropriate connections. The Question item and Puzzle item nodes in Figure 5 represent the question and puzzle shown in Figure 1 and Figure 2. Although the same declarative rules are needed to solve these items, Figure 5 shows that the items are not connected to the same nodes. As discussed earlier, to solve the puzzle item, learners need to know the relevant rule and to recognize its relevance for solving the puzzle. Therefore, the P nodes connected to puzzle items take both of these abilities into account. When the student submits an answer, GearSketch checks whether it is correct and updates the value of the item in the network. Using Bayesian inference (Koller & Friedman, 2009), the other nodes in the graph are then updated to reflect this new information about the student's abilities. Then, the new state of the model is saved by copying the new values of the rule nodes and removing the temporary item node, as discussed by Conati, Gertner, and VanLehn (2002, pp. 388–389). Students do not attempt to solve items just so that the learner model can be updated; they also learn from this activity. Therefore, the final step consists of calculating the probability that the student learned the relevant rules from attempting this item, following the principles discussed by Corbett and Anderson (1995), and then updating the values of the rule nodes.

For example, a learner is asked a simple question about the turning direction of a gear that meshes with another gear which is turning clockwise. Prior to the selection of this question the learner model assigned a probability of 0.6 to this learner knowing the relevant rule. The probability that the learner answers the question correctly is not simply equal to the probability that the learner knows the rule, because the learner could guess the correct answer even when he or she does not know the rule or make a mistake even though he or she does know the rule. These probabilities are referred to as the *guess* and *slip* parameters. Because there are two possible answers (the gear turns clockwise or counterclockwise), it is reasonable to choose 0.5 as the guess parameter. The chance of slipping on this simple question is small, so 0.1 seems a reasonable value for the slip parameter. Using these values, we can now calculate that the predicted probability of the learner answering the question correctly is $0.9 \times 0.6 + 0.5 \times 0.4 = 0.74$. When the learner actually answers the question we have more information, which we can use to update our previous estimate of the probability that the learner knows the relevant rule. For instance, if the learner would answer the question incorrectly, it follows by Bayesian inference that the probability that the learner knew the rule when

answering the question was $\frac{0.1 \times 0.6}{0.1 \times 0.6 + 0.5 \times 0.4} = 0.23$. Finally, there is a chance that although the learner did not know this rule prior to answering this question, he or she learned it from attempting this problem and receiving feedback. We call this the *transition* probability. For a transition probability of 0.3 the updated estimated probability that the student knows the rule after (incorrectly) answering this question is equal to $0.23 + 0.3 \times (1 - 0.23) = 0.46$.

GearSketch then uses the updated learner model for the individual student to select his or her next item. Because GearSketch has information about the rules required to solve each item, an item of appropriate difficulty can be selected. The probability of the learner correctly solving a candidate item can be determined by temporarily inserting this item into the learner model and calculating its value using Bayesian inference. Students of below average and above average ability can benefit from this type of adaptive problem selection compared to static problem selection, because it allows them to learn more efficiently (Mitrovic & Martin, 2004).

Questions and puzzles can either be simple or complex. An item is defined as simple if its solution involves application of just one rule and defined as complex when it requires application of multiple rules. To offer students a variety of items, GearSketch alternates between the four item types in this order: simple question, simple puzzle, complex question, complex puzzle. After completing a complex puzzle, the student is given a new simple question. If no suitable item in one of these categories can be found, an item from the next category is selected. The criteria for an item's suitability are straightforward. Simple puzzle items are only suitable when the learner model indicates that there is a high probability that the student can apply the relevant rule in the context of a question. For instance, a student will only be given a puzzle that requires use of the rule about the turning direction of meshing gears, if the probability that they can apply this rule is at least 0.75 according to the learner model. Complex question and puzzle items are only suitable when the learner model indicates that the probability that the student can apply each of the relevant rules individually in either a question or puzzle context is high. In practice, this means that students must first correctly answer questions about a rule before they are given a puzzle that requires using this rule. Additionally, before students are given complex items, they must show that they can solve simple items for each of the rules needed to solve the complex item. This leads to a gradual, individualized progression of item difficulty.

3.2.4 Abstract and concrete items

Letting students practice applying each of the rules in Table 1 and Table 2 in both question and puzzle contexts requires a lot of different items. Instead of exactly specifying each of these items, GearSketch uses abstract item descriptions that are

converted into concrete items when they are presented to learners. This means that learners cannot use surface features to recognize items they have attempted multiple times, but must identify the underlying abstract structure of a given gear configuration. An abstract question item contains a question with a set of possible answers and information about the correct answer. An abstract puzzle item contains instructions for the learner regarding the goal of the puzzle and sufficient information for GearSketch to evaluate whether that goal was reached. Both types of abstract items contain an abstract description of the concrete gear configuration that will be shown to the learner. This abstract description is used to generate a new concrete gear configuration each time a student attempts this item. Figure 6 shows an example of an abstract description and two concrete configurations created by GearSketch based on this description. The abstract description specifies information about the gears, their properties and the connections between them. The "unmeshable" and "unstackable" properties are used to create puzzles where certain gears cannot be connected via their teeth or axes. These constraints can be used to guide students in a certain direction or to remove obvious solutions, making puzzles either easier or more difficult.

3.3 Method

3.3.1 Participants

Forty-four fifth grade students from a Dutch primary school (26 girls) participated in the experiment. Participants were randomly assigned to either the adaptive condition (N = 22, 12 girls) or a control condition (N = 22, 14 girls). Participants were not asked to provide their birthdate, but given that the experiment was conducted near the end of the school year, their mean age was approximately 11.5 years.

3.3.2 Material

Participants worked with GearSketch using a Wacom Cintiq 12WX pen display connected to a desktop PC. This tablet has a 1280x800 resolution and is controlled with

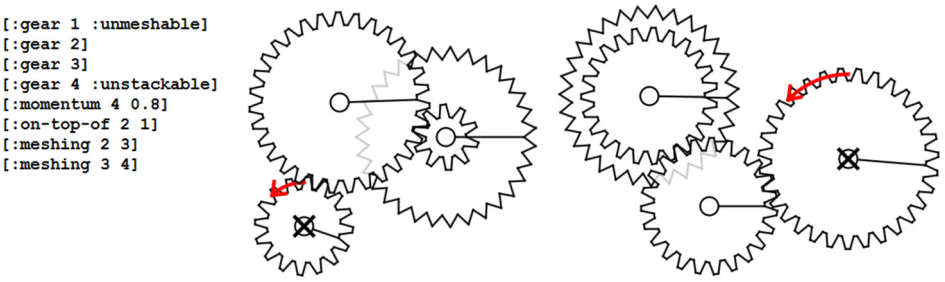


Figure 6. An abstract description and two concrete instantiations of a gear configuration.

a digitizer pen. It does not respond to touches by fingers or hand palm, so participants could rest their hand on it while drawing. The desktop PC's keyboard was used by participants to type their name at the start of the experiment, but otherwise all interaction with GearSketch happened through the tablet.

Two versions of GearSketch were created, a version in which practice problems were selected based on a continuously updated learner model for the adaptive condition and a version with a fixed sequence of practice problems for the control condition. The practice items for the adaptive condition were selected according to the principles discussed in section 3.2.3. The learner model and item selection mechanisms described in section 3.2.3 were also used to simulate a virtual student working with the adaptive version of GearSketch. The sequence of practice items created for this virtual student was used as the fixed sequence of practice items for the control condition. Apart from the mechanism by which the practice items were selected, the version of GearSketch used by participants in the adaptive condition and the version used by those in the control condition were identical.

The learning environment contained 19 short tutorials, which introduced the domain rules in Table 1 and Table 2. These tutorials also explained how to use GearSketch's interface and discussed the spatial constraints on valid gear configurations, such as not being able to connect meshing gears with a chain. The learning environment contained 25 practice questions and 19 practice puzzles. Of these practice items 12 questions and 2 puzzles required some gears to be positioned in such a way that they could be connected by a chain that did not cross the other gears in the configuration. Because the algorithm that created concrete gear configurations based on abstract descriptions of their structure could not guarantee that this constraint would be satisfied, the configuration in these questions and puzzles was defined absolutely. The configuration in the other 13 questions and 15 puzzles was defined abstractly as described in section 3.2.4. The practice questions and puzzles covered all domain rules listed in Table 1 and Table 2. Lastly, a posttest consisting of 9 questions and 6 puzzles was integrated into the learning environment.

3.3.3 Procedure

Participants worked with GearSketch in their school's computer room in groups of up to ten at a time. When participants entered the computer room they were told to take a seat at one of the ten switched on PCs, on which either the adaptive version or the control version of GearSketch was loaded to randomly assign them to one of the two conditions. After a brief verbal explanation about the software by the researcher, participants started individually working with GearSketch. Participants were asked not to talk to each other and to raise their hand if they had any questions for the researcher.

Questions about using the interface were answered directly, but questions about how to solve one of the practice problems were answered by pointing students to the help section which explained the domain's rules.

Participants practiced with GearSketch over a two day period in two 40 minute sessions. Log files were used to track students' progress and at the start of the second session students selected their name from a list to continue at the point where they ended the first session. Participants continued practicing with GearSketch until 20 minutes into the second session. The final 20 minutes of the second session were used to complete the integrated posttest. During this posttest animations were disabled and participants did not receive the feedback about the correctness of their answers that they received when working with the practice problems. After students had completed the posttest they could spend the remaining time freely drawing and animating gear and chains in a final 'puzzle' with no specific goals.

3.3.4 Analysis

The posttest was scored by assigning 1 point to each correct answer and 0 points to each incorrect answer and then summing these points to get a single posttest score for each participant. No distinction was made between the questions and puzzles, because the difference between the two conditions applied equally to both types of problems. Because participants in the adaptive condition were expected to learn more efficiently from working with GearSketch than those in the control condition, a one-tailed *t*-test was used to compare the posttest scores of the two groups.

To examine the accuracy of the learner model's predictions, the final learner model of each participant in the adaptive condition was used to predict their performance on each posttest item. This resulted in a list of 15 probabilities (9 for the questions, 6 for the puzzles) that the student would correctly answer each posttest item for each student in the adaptive condition. To determine the quality of the learner model's predictions, the correlation between these probabilities and the participants' posttest results was calculated. Additionally, an expected posttest score was calculated for each participant in the adaptive condition by summing the 15 probabilities that this participant would answer the items correctly. To determine whether the learner model's prediction displayed any bias toward overestimation or underestimation of the students' capabilities, these predicted scores were compared to the actual posttest scores using a two-tailed paired samples *t*-test.

3.4 Results

There was no significant difference between the posttest scores of participants in the adaptive condition (M = 7.50, SD = 1.92) and participants in the control condition (M = 7.36, SD = 7.36), $t(42) = 0.23$, $p = .408$.

The correlation between the learner models' predictions and actual test performance of participants in the adaptive condition was significant, $r = .168$, $p = .002$. The predicted posttest score for participants in the adaptive condition based on the learner model was 7.49. There was no significant bias towards overestimation or underestimation in the learner models' predictions of these students' performance on the test, $t(21) = 0.035$, $p = .972$.

3.5 Discussion

This chapter discussed choices made in the design of GearSketch to answer four development questions and an experimental study to answer two evaluation questions. The learning goal for students working with GearSketch is to be able to solve qualitative gear problems, such as connecting gears in such a way that they turn with a given relative speed or in a given direction. To solve these problems, students need to acquire procedural knowledge of the declarative rules found in Tables 1 and 2 and an understanding of relevant spatial constraints (DQ1). To help students acquire this knowledge, GearSketch lets students draw and explore gear and chain configurations by interpreting their pen strokes and animating the resulting model (DQ2). The declarative rules of the domain model are introduced in tutorials and students attempt to solve practice problems to compile these rules into procedural knowledge. The rules needed to solve each of these practice problems are represented explicitly in their abstract descriptions, so that a Bayesian learner model can track each student's progress as he or she works through these questions and puzzles (DQ3). Most of the gear configurations used in the practice questions and puzzles are described abstractly, in terms of structure instead of absolute position. This lets GearSketch create a new concrete problem each time a student attempts to solve the same abstract problem. Therefore students cannot learn how to solve these problems through memorizing their surface features, but have to pay attention to their structure. Suitable practice items are then selected based on learners' current knowledge of the domain as represented by the learner model (DQ4).

Contrary to our expectations, the adaptive condition's performance on the posttest was not significantly better than the control condition's performance (EQ1). Additionally, we found that the learner model's predictions of students' test results were not very accurate, although there was no bias towards overestimation or underestimation (EQ2). The low accuracy of the learner model's predictions could be due to inaccurate

estimates for the initial parameters of the learner model or because the students did not work with GearSketch long enough to provide the learner model with sufficient data to adequately capture their knowledge. Another possibility is that the structure of the learner model did not fit students' reasoning process. This could happen if participants did not try to solve the practice problems by deliberately applying their declarative knowledge, but used their own heuristics or trial and error. If this is the case, an overlay model like the one used by GearSketch cannot adequately model the students' problem-solving behavior.

The low correlation between the learner model's predictions and participants' actual test performance offers a possible explanation for the finding that tailoring practice item selection did not improve learning outcomes. Because the learner model's predictions were also the basis for the selection of practice items, it is likely that the selection of practice items was not optimal. Therefore, improving the learner model may be a fruitful approach to making the adaptive item selection of GearSketch more effective in supporting the students' learning process. Another possible explanation could be that students needed more support from the learning environment when they failed to solve a practice problem in order to really learn from this experience.

Whereas Chapter 2 showed that using GearSketch's domain model to interpret and animate students' gear systems facilitates learning, this chapter's findings indicate that using the domain model to track learner's knowledge and select practice items does not improve learning outcomes. This leads to a number of possible directions for future research. One such direction would be to take the ideas in GearSketch that have been shown to work and apply them in other domains. Suitable domains would be those that are governed by a limited number of rules and contain only a types few elements which can interact in complex ways and that have a natural pictorial representation. A good example is the ropes and pulleys domain. A different direction for future research would be to further examine the relation between GearSketch's learner model and students' actual problem-solving behavior. If the goal is to better describe learners' behavior, learner models based on stereotypes or common misconceptions could be explored (Chrysafiadi & Virvou, 2013). If the step-by-step approach that is expected by the current learner model is seen as prescriptive rather than descriptive, which is not unreasonable for a domain that is governed by simple rules, forms of support that encourage the learner to apply their declarative knowledge more deliberately may be examined.

References

Ainsworth, S., Prain, V., & Tytler, R. (2011). Drawing to learn in science. *Science*, *333*(6046), 1096–1097. doi:10.1126/science.1204153

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, *111*(4), 1036–1060. doi:10.1037/0033-295X.111.4.1036

Anderson, J. R., Boyle, C. F., Corbett, A. T., & Lewis, M. W. (1990). Cognitive modeling and intelligent tutoring. *Artificial Intelligence*, *42*(1), 7–49. doi:10.1016/0004-3702(90)90093-F

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *Journal of the Learning Sciences*, *4*(2), 167–207. doi:10.1207/s15327809jls0402_2

Andrade, A. (2009). The clock project: Gears as visual-tangible representations for mathematical concepts. *International Journal of Technology and Design Education*, *21*, 93–110. doi:10.1007/s10798-009-9104-x

Bartolini Bussi, M. G., Boni, M., Ferri, F., & Garuti, R. (1999). Early approach to theoretical thinking: Gears in primary school. *Educational Studies in Mathematics*, *39*(1-3), 67–87. doi:10.1023/A:1003707727896

Bespamyatnikh, S. (2003). Computing homotopic shortest paths in the plane. *Journal of Algorithms*, *49*(2), 284–303. doi:10.1016/S0196-6774(03)00090-7

Bollen, L., & Van Joolingen, W. R. (2013). SimSketch: Multiagent simulations based on learner-created sketches for early science education. *IEEE Transactions on Learning Technologies*, *6*(3), 208–216. doi:10.1109/TLT.2013.9

Chambers, J. M., Carbonaro, M., & Murray, H. (2008). Developing conceptual understanding of mechanical advantage through the use of Lego robotic technology. *Australasian Journal of Educational Technology*, *24*(4), 387–401.

Chrysafiadi, K., & Virvou, M. (2013). Student modeling approaches: A literature review for the last decade. *Expert Systems with Applications*, *40*(11), 4715–4729. doi:10.1016/j.eswa.2013.02.007

Conati, C., Gertner, A., & VanLehn, K. (2002). Using Bayesian networks to manage uncertainty in student modeling. *User Modeling and User-Adapted Interaction*, *12*(4), 371–417. doi:10.1023/A:1021258506583

Corbett, A. T., & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modelling and User-Adapted Interaction*, *4*(4), 253–278. doi:10.1007/BF01099821

Cox, R. (1999). Representation construction, externalised cognition and individual differences. *Learning and Instruction*, *9*(4), 343–363. doi:10.1016/S0959-4752(98)00051-6

Dixon, J. A., & Bangert, A. S. (2004). On the spontaneous discovery of a mathematical relation during problem solving. *Cognitive Science*, *28*(3), 433–449. doi:10.1207/s15516709cog2803_6

Forbus, K., Usher, J., Lovett, A., Lockwood, K., & Wetzel, J. (2011). CogSketch:

Sketch understanding for cognitive science research and for education. *Topics in Cognitive Science*, *3*(4), 648–666. doi:10.1111/j.1756-8765.2011.01149.x

Höffler, T. N., & Leutner, D. (2007). Instructional animation versus static pictures: A meta-analysis. *Learning and Instruction*, *17*(6), 722–738. doi:10.1016/j.learninstruc.2007.09.013

Hu, W. (2011, January 4). More schools embrace the iPad as a learning tool. *The New York Times*. Retrieved from http://www.nytimes.com/2011/01/05/education/05tablets.html

Koller, D., & Friedman, N. (2009). *Probabilistic graphical models: Principles and techniques* (1st ed.). Cambridge, MA: The MIT Press.

Lee, M. (2010). Interactive whiteboards and schooling: The context. *Technology, Pedagogy and Education*, *19*(2), 133–141. doi:10.1080/1475939X.2010.491215

Leenaars, F., Van Joolingen, W. R., & Bollen, L. (2013). Using self-made drawings to support modelling in science education. *British Journal of Educational Technology*, *44*(1), 82–94. doi:10.1111/j.1467-8535.2011.01272.x

Mitrovic, A. (2003). An intelligent SQL tutor on the web. *International Journal of Artificial Intelligence in Education*, *13*(2-4), 173–197.

Mitrovic, A., & Martin, B. (2004). Evaluating adaptive problem selection. In P. M. E. D. Bra & W. Nejdl (Eds.), *Adaptive Hypermedia and Adaptive Web-Based Systems* (pp. 185–194). Springer Berlin / Heidelberg.

Mitrovic, A., Mayo, M., Suraweera, P., & Martin, B. (2001). Constraint-based tutors: A success story. In L. Monostori, J. Váncza, & M. Ali (Eds.), *Engineering of Intelligent Systems* (Vol. 2070, pp. 931–940). Springer Berlin / Heidelberg.

Van Essen, G., & Hamaker, C. (1990). Using self-generated drawings to solve arithmetic word problems. *The Journal of Educational Research*, *83*(6), 301–312.

VanLehn, K., Lynch, C., Schulze, K., Shapiro, J. A., Shelby, R., Taylor, L., … Wintersgill, M. (2005). The Andes physics tutoring system: Lessons learned. *International Journal of Artificial Intelligence in Education*, *15*(3), 147–204.

4

ENCOURAGING DELIBERATE REASONING DURING PROBLEM SOLVING IN THE GEARS DOMAIN

Students working with GearSketch, a drawing-based learning environment for the gears domain, can use simulations to explore the behavior of gears and chains. The learning environment's direct manipulation interface makes it easy to try out different solutions to problems which can speed up learning, but may also lead to trial-and-error behavior. To encourage a deliberate style of problem solving, a reasoning support tool is introduced that requires students to explicitly show their reasoning step while they answer practice questions. A first experiment ($N = 52$) with this tool shows that using it actually leads to worse performance on the practice questions. Changes were made to the tool to better integrate it with the questions and a second experiment ($N = 78$) shows that the improved tool leads to better performance on the practice questions. However, no effects on students' problem-solving behavior during unsupported practice problems is found and using the tool does not lead to improved learning gains. Both experiments show significant learning gains in the gears domain from practicing with GearSketch.

4.1 Introduction

Ever since digital learning environments have been used in education, there has been a tension between giving learners freedom to explore new domains and providing structure. Two types of learning environments that are based on a computational domain model which are on opposite ends of the freedom-structure scale are inquiry learning environments and cognitive tutors. Learning environments based on computer simulations often give students a lot of freedom to discover the workings of an underlying model through self-directed experimentation. However, many studies have confirmed that giving learners complete freedom when working with simulations is not efficient and causes them to flounder (De Jong & Van Joolingen, 1998; Kirschner, Sweller, & Clark, 2006; Mayer, 2004). Simulation-based learning environments that do guide learners' inquiry can improve student motivation and learning gains compared to control classrooms (Hmelo-Silver, Duncan, & Chinn, 2007) and leads to more engagement than other approaches, which can lead to improved understanding (Eysink et al., 2009; Eysink & De Jong, 2012). Two meta-analyses by Alfieri, Brooks, Aldrich and Tenenbaum (2011) also show the importance of guidance during the discovery process. They found that while explicit instruction compares favorably to unassisted discovery learning, supported discovery learning compares favorably to other forms of instruction.

On the other end of the freedom-structure scale, we find learning environments based on cognitive tutors. These learning environments are based on ACT* and ACT-R theory (Anderson, Boyle, Corbett, & Lewis, 1990; Anderson et al., 2004) and help students develop cognitive skills by supporting students' compilation of declarative knowledge into procedural knowledge through problem solving. Like simulation-based inquiry learning environments, cognitive tutors are built around a computational model. However, in the case of cognitive tutors, this model represents not only the domain that students are exploring, but also students' understanding of this domain (Anderson et al., 1990). This lets cognitive tutors track and give feedback on each problem solving step taken by students and select further practice problems based on each individual student's current understanding (Corbett & Anderson, 1995; Corbett, 2001). Learning with cognitive tutors has proven to be very effective compared to traditional approaches (Desmarais & Baker, 2012). However, providing too much structure and feedback can also lead to problems. Students can game the system by requesting a lot of feedback and by using systematic trial-and-error to find correct answers to progress through the learning environment without learning a lot (Baker, Corbett, & Koedinger, 2004; Baker, Corbett, Roll, & Koedinger, 2008). When to give and when to withhold support to achieve optimal student learning is an open problem (Koedinger & Aleven, 2007).

Some of the problems learners encounter during unguided inquiry or unsupported

problem solving can be explained by learners relying on trial-and-error instead of using deliberate problem solving based on their prior knowledge. Klahr and Dunbar (1988) found that students who were trying to figure out how a programmable car responded to different commands could be described as either experimenters or theorists. Experimenters explored experiment space by programming the car with different commands without an explicit hypothesis about how the car would respond to these commands. In contrast, theorists focused on exploring hypothesis space, by searching their memory for the results of earlier experiments with the car and then tested their hypothesis by doing new experiments. There are similarities between Klahr and Dunbar's experimenters and theorists and Hayes and Broadbent's (1988) distinction between two different learning modes: u-mode and s-mode. U-mode learning is unselective, relatively automatic and its results are hard to verbalize. It is most often used in situations in which a complex system's behavior is hard to predict. S-mode learning is selective and characterized by the active construction of a model of the system that is being explored that can be reported verbally. This mode of learning is most often used when the system that is explored is simpler and a smaller number of variables have to be considered at a time. Guidance in learning environments can help learners behave like theorists and use s-mode learning.

Learners' preferred problem solving strategy is not only affected by properties of the system under study and their prior domain knowledge, but also by the learning environment's interface. Svendsen (1991) found that direct manipulation interfaces may not induce s-mode learning as well as command interfaces do. Participants who practiced solving the tower of Hanoi puzzle using a direct manipulation interface made more errors and were less able to explain the principles behind the solution than those who used a command interface. However, there is also evidence that the improved speed and efficiency of direct interfaces can help during problem solving, as Donahue et al. (2013) found in an experiment with an abstract problem-solving task based on the Mastermind game.

GearSketch is a drawing-based learning environment for the gears domain. It features an interface that interprets students' pen strokes and converts them to gears and chains and allows students to explore the behavior of these gear systems through simulations. Learners do not use GearSketch to discover the rules that govern gear and chain interactions from scratch, but progress through a series of tutorials that explicitly explain these rules. Next, they use their declarative knowledge of these rules to answer questions and solve puzzles to acquire procedural knowledge. The results from Chapter 2 show that GearSketch's interface and simulation features help students learn about the domain. On the other hand, the results from Chapter 3 show that tracking students' knowledge with a learner model and selecting practice problems based on this model did not improve learning outcomes. This lack of an effect was likely related to the low

accuracy of the learner model's predictions of students' performance. An explanation for this finding could be that that the learner model expects students to deliberately use their declarative knowledge during problem solving, whereas students may sometimes rely on a trial-and-error approach instead. This explanation is consistent with Svendsen's (1991) finding that direct manipulations interfaces, such as GearSketch's drawing-based interface, can lead to u-mode learning behavior.

This kind of trial-and-error behavior could be used by students during both the practice questions and the practice puzzles. During the questions, students could systematically try all the answers and they would eventually end up at the correct one. During the puzzles, students could use simulations to request feedback about many different gear configurations instead of trying to find a solution based on their knowledge of the domain rules. To discourage this behavior, students are only given access to a simulation of the gear configuration's behavior after they have tried an incorrect solution. However, this could still lead to trial-and-error behavior after an initial mistake has been made. Students may also quickly try a first solution without much thought to gain access to the simulation.

This chapter examines the effects of providing students with a reasoning support tool during the practice questions. In previous versions of GearSketch, these questions were static in the sense that students could not manipulate the gear configuration and only saw an animation of this configuration after they had found the correct answer. Observations during earlier studies with GearSketch suggested that students often did not reflect on their incorrect answers, but quickly checked one of the other answers and proceeded to the next practice item. Given the importance of self-explanations in learning from problem-solving (Chi, Bassok, Lewis, Reimann, & Glaser, 1989), there seems to be room for improvement for these practice items. The goal of the reasoning support tool discussed in this chapter is to encourage students to deliberately use their prior knowledge when answering the practice questions and to reflect on their mistakes. This is done by asking students to explicitly indicate each reasoning step necessary to answer the questions.

Two experimental studies are conducted in which participants in one group are supported with the reasoning support tool and a control group does not have access to this support. The most direct expected effect of the support tool is that it helps learners find the correct answer to the practice questions, which leads to our first research question:

- RQ1. Are students in the supported group more successful in answering the practice questions than those in the control group?

We also expect that working with the reasoning support tool during practice questions

affects the way students reason during the practice puzzles. This leads to the next two research questions:

- RQ2. Are students in the supported group more successful in solving the practice puzzles than those in the control group?
- RQ3. Do students in the supported group behave differently when attempting to solve the practice puzzles than those in the control group?

Finally, we expect both groups of students to learn from working with GearSketch, but expect those in the supported condition to learn more. This leads to the last two research questions:

- RQ4. Do students in the supported and the control condition learn from working with GearSketch?
- RQ5. Do students in the supported group learn more from working with GearSketch than those in the control group?

The reasoning support tool and the two studies conducted to evaluate the effects of using it are discussed in the next two sections.

4.2 Study 1

4.2.1 Method

4.2.1.1 Participants

Fifty-two sixth grade students from a Dutch primary school (27 girls), with a mean age of 12.33 years (SD = 0.57) participated in the experiment. Participants were randomly assigned to either the supported condition (N = 27, 13 girls, M_{age} = 12.23, SD_{age} = 0.50) or a control condition (N = 25, 14 girls, M_{age} = 12.45, SD_{age} = 0.61).

4.2.1.2 Material

Participants worked with the GearSketch learning environment using a laptop and pen tablet. Ten laptops, eight Wacom Cintiq 12WX tablets and two Wacom Bamboo One tablets were used in the study. The Cintiq tablet features a 1280x800 resolution pen-based touchscreen and the lids of the laptops connected to these devices were closed so that participants interacted only with the tablet. The Bamboo tablet does not have a built-in display and participants who worked with these tablets used the laptop's screen (1280x720 resolution) to view the GearSketch learning environment. GearSketch ran in full screen mode on both the Cintiq tablets and the laptop screens.

The version of GearSketch used by the supported group provided a reasoning support tool during practice questions and contained instructions on using this support tool, but

was otherwise identical to the GearSketch version used by the control group. Features of GearSketch common to both conditions will be described first, followed by a detailed description of the reasoning support tool.

Common GearSketch features

GearSketch is a drawing-based learning environment for the gears domain, described in detail in Chapter 3. GearSketch contains tutorials and a fixed sequence of concrete practice problems. The 15 tutorials serve to introduce the rules of the domain model and let learners practice using the interface. The 30 practice problems help learners compile their declarative knowledge of the domain model into procedural knowledge. There are two types of practice problems: questions and puzzles. Question items provide a gear configuration and ask learners to predict in which direction or with which relative speed a given gear will start to turn. Students cannot run a simulation during the practice questions until they have given the correct answer. Answering these questions helps students learn and directly apply the declarative rules introduces in the tutorials. Puzzle items ask learners to accomplish a task for a given gear configuration, such as making a gear turn faster than another gear, by adding or moving gears and chains. Participants cannot simulate the turning of the gears while trying to solve a puzzle for the first time to encourage deliberate problem solving over a trial-and-error approach. If an incorrect solution is checked, the play button becomes available so that students can examine why their solution does not work. At that point, students can use the play button as often as they want for this practice puzzle, but they can only check whether their solution is correct three times. When checking their answer students receive written feedback about their solution, stating either that the puzzle is solved or explaining which of the puzzle's goals has not been achieved. After checking three incorrect solutions, they must continue to the next practice problem. From practicing with these puzzles students learn to recognize which of the rules that were explained in the tutorials can be used to make gear systems behave in a given way. The version of Gearsketch used in this study includes 15 questions and 15 puzzles that are alternately presented, starting with a question. Compared to previous versions of GearSketch, the version used in this study is based on a different domain model and has an updated interface.

The rules of the domain model used in this study are shown in Table 1. This domain model is simplified compared to the domain model used in the previous studies. Because the tooth speed concept is not used, only eight rules are necessary to qualitatively describe the domain instead of the twelve rules used in Chapters 2 and 3. This simplified domain model makes it possible to use a similar pretest and posttest without having to explain the novel tooth speed concept to participants before the pretest.

Table 1

Domain model rules

Connection type	Turning direction	Turning speed
Meshing gears	Opposite	Smaller gear has higher turning speed
Gears on top of each other (sharing an axis)	Equal	Equal turning speed
Gears supporting a chain (same side)	Equal	Smaller gear has higher turning speed
Gears supporting a chain (opposite side)	Opposite	Smaller gear has higher turning speed

To solve the puzzle items, learners can draw, drag and cross out gears and chains to create, move and remove them. In previous studies, learners selected gear or chain buttons to indicate which of these two objects they wanted to create, move or remove. In the version of GearSketch used for this study these different modes have been removed and the learning environment automatically interprets learners' pen strokes as either gears or chains without them having to indicate which of these objects they wish to draw. Circles drawn by the learner are interpreted as gears and replaced by gears of equal size. Pen strokes drawn around multiple gears are replaced by chains that are tightened around the supporting gears. This leads to the simplified interface displayed in Figure 1.

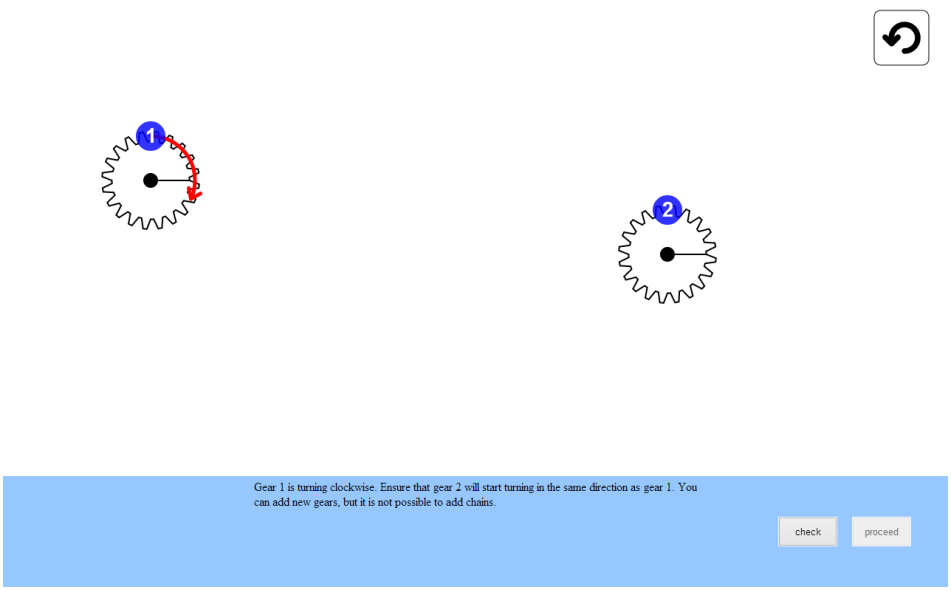


Figure 1. Screenshot of the GearSketch learning environment showing a practice problem.

The reasoning support tool

A reasoning support tool is integrated in the version of GearSketch used by participants in the supported condition and offers support during practice questions. Additionally, three extra tutorials are included to explain how to use the support tool. Participants in the control condition can answer the practice questions by immediately selecting one of the possible answers and checking whether it is correct. In contrast, the reasoning support tool requires participants in the supported condition to indicate each of the reasoning steps that are necessary to answer the practice question before they can select an answer. Figure 2 shows the interface used for selecting these reasoning steps. Indicating these reasoning steps is done using the following procedure:

To indicate a relation between two gears, learners draw a line between the two gear numbers. If the question is about turning direction, learners can indicate the turning directions of the two gears. If the question is about turning speed, learners can indicate the turning speed of the second gear relative to the first gear. For instance, Figure 2 shows the interface for a question about turning speed after a learner has drawn a line between gear 2 and gear 3. Now the learner must select whether gear 3 turns slower than, at the same speed as, or faster than gear 2.

Learners can add a reasoning step about the relation between two gears, if either the turning direction or the turning speed of the first gear is known. This information can be

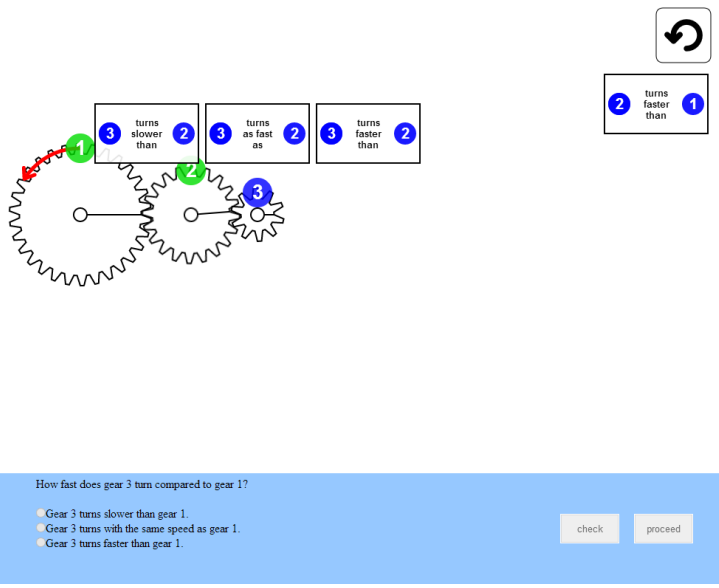


Figure 2. Screenshot of the GearSketch learning environment showing the reasoning support tool. In this question the learner is asked whether gear 3 will turn slower or faster than gear 1.

known for one of two reasons. The first and simplest reason is that there is a turning arrow in the first gear, giving information about its turning direction and speed. The second reason is that one of the learner's previous reasoning steps leads to this information. For instance, in Figure 2 the learner is able to add a reasoning step for the relation between gear 2 and gear 3 because information about the turning speed of gear 2 follows from the first reasoning step, which is about the relation between gear 1 and gear 2. Gears about which information is available are marked with a green circle around their gear number, whereas gears about which no information is available have a blue circle around their gear number.

Each question is about either the turning direction of a specific gear or the relative turning speed of a given gear compared to another gear. Learners are only able to select one of the multiple choice answers and check whether their answer is correct when one of their reasoning steps provides information about this gear.

This procedure ensures that the learner has to explicitly indicate reasoning steps starting from the gear about which information is given and leading to the gear about which the question is asked. When learners select incorrect reasoning steps, they can start over by removing their steps by selecting the undo button in the upper right corner of Figure 2.

Other materials

After participants had completed the tutorials, they received a summary of the domain rules printed on a two-sided sheet of A4 paper. This summary can be found in Appendix A. Participants could consult this summary while attempting to solve the practice problems.

In addition to working with the GearSketch learning environment, participants completed a paper-and-pencil pretest and posttest. These tests both consisted of sixteen multiple-choice questions and four puzzle questions. The items on these tests were created to cover all of the rules in the domain model and ranged from simple items that were based on just one of these rules to complex items that depended on four different rules. To ensure that the tests were of equal difficulty, two versions of each item were created that had the same structure, but different surface features. One version of each item was then randomly assigned to the pretest and the other version to the posttest. The pretest can be found in Appendix B.

4.2.1.3 Procedure

Participants completed the pretest, practiced with GearSketch and completed the posttest in a single session. Groups of students participated in the study in groups of ten at a time, but all students worked individually. Participants were randomly assigned to either the supported condition or the control condition, with five students in each group

of ten assigned to each condition. This assignment was done by loading one of the two versions of GearSketch on each laptop that the students worked with and students were not explicitly told about the existence of the different conditions. For each group of ten students, eight Wacom Cintiq 12WX touchscreens and two Wacom Bamboo One tablets were available. Participants then received a brief oral explanation about the topic of the study and were informed that the tests that they completed would not affect their school grade. They were also told that it was important for the study that they worked on the tests and with GearSketch individually, without discussing it with other students in the room. If they had questions they could raise their hand at any point and the researcher would try to help them. After this explanation, participants completed the pretest for which 15 minutes were available. Next, they worked with GearSketch for 45 minutes. Participants who completed all practice problems before 45 minutes had passed, could spend the rest of this time freely drawing and animating gear and chain systems in a sandbox version of GearSketch without tasks. Finally, participants completed the posttest for which 15 minutes were again available.

4.2.1.4 Analysis

Participants' performance on both practice items and test items was analyzed. Practice performance was analyzed by comparing the proportion of questions and puzzles that participants in both conditions answered correctly on their first try. This comparison was made by comparing the proportions of correct question and puzzle items using t-tests. The average number of practice items completed by each group was also calculated and compared using a t-test.

Performance on the test items was analyzed by calculating two improvement scores for each participant, one for the question items and one for the puzzle items. To examine if participants in both conditions learned from working with GearSketch, t-tests were used to see if these improvement scores were significantly greater than 0. To examine whether the supported group learned more from working with GearSketch than the control group, average improvement scores of the two groups were compared to each other using t-tests.

4.2.2 Results

Participants in the supported condition correctly answered 51% of the practice questions on their first attempt. Participants in the control condition correctly answered 64% of the practice questions on their first attempt. Participants in the control condition correctly answered questions on their first attempt significantly more often than those in the supported condition, $t(625) = 3.11$, $p = .002$.

Participants in the supported condition solved 40% of the practice puzzles on their first

attempt. Participants in the control condition solved 42% of the practice puzzles on their first attempt. There was no significant difference between proportions of puzzles solved on the first attempt between the supported and control group, $t(611) = 0.43$, $p = .67$.

Out of a maximum of 30 practice items, students in the supported condition on average completed 22.30 (SD = 7.98). Students in the control condition on average completed 25.52 (SD = 5.83) practice items. There was no significant difference between the average number of items completed by each group, $t(50) = 1.65$, $p = .11$.

The average scores for the question and puzzle items on the pretest and posttest are listed in Table 2. There was a significant average improvement for participants in the supported condition on both the question items, $t(26) = 3.62$, $p = .001$, $d = 0.70$ and the puzzle items, $t(26) = 4.27$, $p < .001$, $d = 0.82$. For participants in the control condition there was also a significant average improvement on both the question items, $t(24) = 3.88$, $p = .001$, $d = 0.78$ and the puzzle items, $t(24) = 3.50$, $p = .002$, $d = 0.70$. There was no significant difference between the average improvement scores on the question items of the supported group and the control group, $t(50) = 0.67$, $p = .51$. There was also no significant difference between the two groups for the puzzle items, $t(50) = 0.36$, $p = .72$.

4.2.3 Discussion

Participants in the supported condition performed worse on the practice questions than those in the control condition. This finding strongly suggests that the reasoning support tool did not help participants reason about gear systems more deliberately. Therefore we did not further examine differences in problem-solving behavior between the groups when they attempted to solve the practice puzzles. There was no significant difference between the groups in their success on the practice puzzles. Because participants in the experimental condition had to do more work during the practice questions and go through three extra tutorials, they were expected to complete fewer practice items. However, the number of practice items completed did not differ significantly between the two conditions.

Table 2

Pretest and posttest results and standard deviations for study 1

	Pretest questions	Posttest questions	Questions improvement	Pretest puzzles	Posttest puzzles	Puzzles improvement
Supported condition	8.30 (2.40)	10.74 (3.23)	2.44 (3.51)	0.52 (0.70)	1.70 (1.27)	1.19 (1.44)
Control condition	9.16 (1.86)	11.04 (2.49)	1.88 (2.42)	0.72 (1.02)	1.76 (1.54)	1.04 (1.49)

The scores of participants in both groups significantly improved from pretest to posttest and the effect sizes are medium to large for both questions and puzzles. However, there was no significant difference between the average improvements of the groups. This shows that using the support tool did not lead to greater learning gains.

Observations made during the experiment suggested three reasons for the finding that the reasoning support tool did not help participants better answer the practice questions. First, participants in the supported condition seemed to treat selecting the reasoning steps as something that was unrelated to answering the question. Instead of focusing on how the reasoning steps could help them reach a conclusion about the gears that were referred to in the question, participants often started adding steps before they had even read the question. Second, participants seemed to focus on the gear numbers at the expense of looking at the structure of the gear configuration. For example, several students raised their hands to ask why they could not connect two numbers to each other in a question where those numbers were close to each other, but the gears they identified were not connected to each other. Third, when participants answered questions incorrectly, they did not seem to reflect on their reasoning steps, but immediately selected one of the other answers and checked whether that one was correct.

Observations also suggested two smaller issues with the experiment. Participants sometimes noticed when other participants had completed all the practice problems and could freely create their own gear and chain systems. This caused some students who had not yet completed the practice items to become distracted. Finally, after completing the pretest and practicing with GearSketch quietly for an hour, participants seemed less focused during the posttest than they were during the pretest.

Based on these findings and observations, a second study was done in which these issues were addressed. This second study is described in the next section.

4.3 Study 2

In the second study the same design was used as in the first study, but changes were made to the material and procedure to avoid the problems found in the first study. Additionally, a more extensive analysis was done to examine the effect of the reasoning support tool on participants' puzzle-solving behavior. These changes are discussed in the material, procedure and analysis sections.

4.3.1 Method

4.3.1.1 Participants

Seventy-eight fifth and sixth grade students from two Dutch primary schools (38 girls, M_{age} = 11.81, SD_{age} = 0.72) were randomly assigned to either the supported condition (N = 40, 19 girls, M_{age} = 11.76, SD_{age} = 0.68) or a control condition (N = 38, 19 girls, M_{age} = 11.86, SD_{age} = 0.77).

4.3.1.2 Material

Five changes were made to the reasoning support tool used in the first study. First, instead of connecting gears' numbers to indicate a relation between them, participants could now connect the gears themselves directly. This change was made to focus students' attention on the structure of the gear configuration instead of the numbers' locations. A second change to the interface made it possible to remove individual reasoning steps instead of having to start over by pressing the reset button in the top right corner. At each moment only the last reasoning step could be removed, because removing earlier steps would break the chain of steps, possibly leaving later steps without justification. This behavior was explained in an extra tutorial. Third, the answers participants could select were linked to the chain of reasoning steps they created. Only answers that followed from these reasoning steps were selectable. For example, if a student indicated that gear 3 turned faster than gear 2 and gear 2 turned at the same speed as gear 1, it would be possible to select the answer "gear 3 turns faster than gear 1", but not "gear 3 turns slower than gear 1". When students gave an incorrect answer, they must have made a mistake in one or more of their reasoning steps and any such mistakes had to be corrected before the correct answer could be selected. Fourth, when students checked an incorrect answer, the first incorrect reasoning step leading to this answer was marked in red. This helped students backtrack through their steps and correct their mistake, which was now necessary before they could change their answer. Fifth, students' reasoning steps were now explicitly checked. When students selected a correct answer which was based on incorrect reasoning steps, these steps had to be corrected before students could proceed to the next question.

Six new practice items were added to GearSketch, three questions and three puzzles. These items were added because participants in this study were given slightly more time to work with GearSketch and many participants in the first study completed all of the practice items.

Finally, two items on the pretest and posttest were changed. The first item was a question that contained a redundancy that allowed students to answer it correctly even if they misunderstood one of the rules covered by this question. The question was slightly

changed to remove this redundancy. The second item that was changed was a puzzle that was solved correctly by a very small percentage of students on both the pretest and the posttest. This puzzle was changed to make it easier.

4.3.1.3 Procedure

Two changes were made to the procedure. First, students were now given 50 minutes to work with GearSketch's tutorial and practice items. After this they were give 10 minutes during which they could use GearSketch in sandbox mode to freely create and experiment with gear and chain systems. This change was made so that participants would not try to rush to complete all the items within the first 50 minutes when they noticed that some of their classmates were using GearSketch in sandbox mode. A second change was made to the time at which participants worked with GearSketch and made the tests. Participants made the pretest one or two days before working with GearSketch and completed the posttest one to four days after working with GearSketch so that they could focus on the posttest as well as they did on the pretest.

4.3.1.4 Analysis

The same analyses that were done in the first study were used in this second study. In addition students' puzzle-solving behavior was analyzed to determine how using the reasoning support tool during the practice questions affected students' behavior during the practice puzzles. All students' interactions with GearSketch were saved in log files and these log files were used to find the different gear configurations participants created while trying to solve each practice puzzle. Each gear configuration is a state in the state space of all possible gear configurations. Figure 3 shows a number of states visited by a fictional student who attempts to solve one of the practice problems, in which the goal is to make gear 2 turn clockwise. The following states are shown:

A. The initial state.
B. The student tries to drag gear 2, but the gear turns red and a grey outline remains in the gear's previous location to indicate that it cannot be moved. When the student releases the gear it snaps back to its previous position.
C. The student adds a chain, supported by gear 1 and 2 on the inside and gear 3 on the outside. Then the student checks this solution (which is incorrect).
D. The student moves gear 3 to the other side of the chain, presses play and notices that gear 2 is still turning in the wrong direction.
E. The student removes the chain.
F. The student adds a chain around gear 1 and 3.
G. The student moves gear 3 to the right so that gear 2 is now connected to the outside of the chain. Then she presses play and afterwards checks her solution (which is correct).

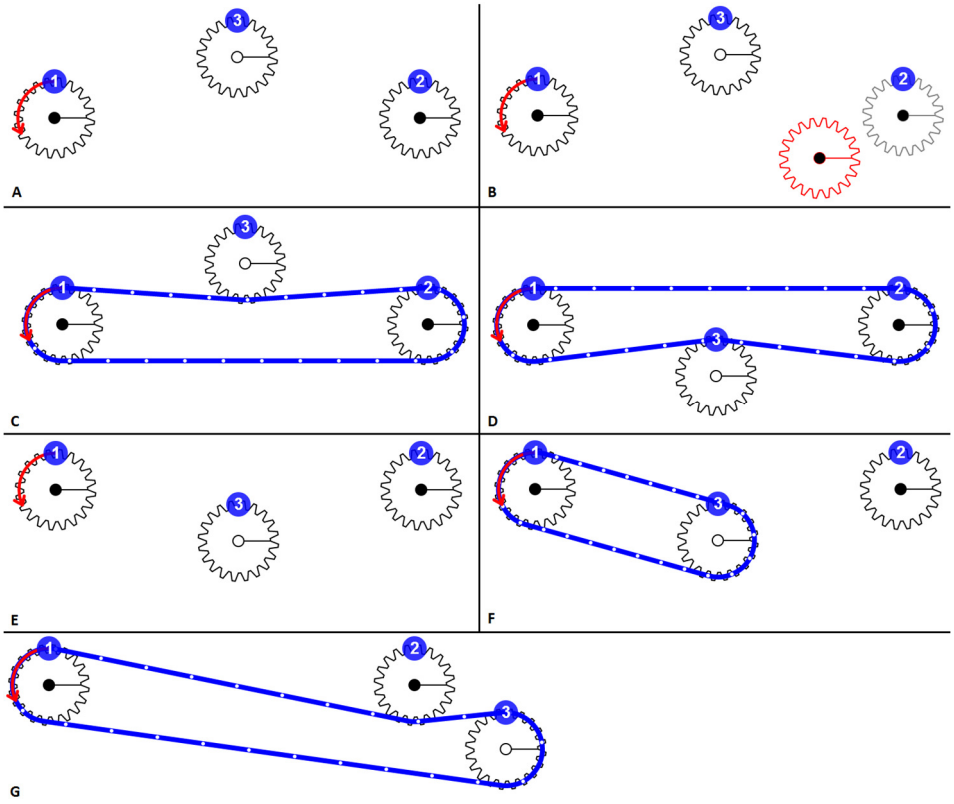


Figure 3. A fictional student's attempt to solve one of the practice puzzles. The goal of this puzzle is to make gear 2 turn clockwise. Gear 1 is turning counterclockwise, only gear 3 can be moved and only chains can be added.

To create the sequence of states from a log file, the gear configuration at the end of each action (i.e. adding, moving or removing a gear or chain) is compared to the gear configuration at the beginning of that action. If the action leads to a change in the configuration, the new state is added to the sequence. Actions that did not affect the gear configuration, such as trying to drag the unmovable gear in B, do not lead to a new state and no new state is added to the state sequence.

Next, the series of concrete states can be translated to a series of abstract states. An abstract state defines the structure of the gear configuration, but ignores the exact position and rotation of the gears and chains. For instance, concrete states C and D have the same structure and are therefore represented by the same abstract state. A graph representation of the abstract states that are visited can then be created by mapping each concrete state to an abstract state and creating a node for each distinct abstract state and edges between nodes that represent transitions between states. Figure 4 shows such a

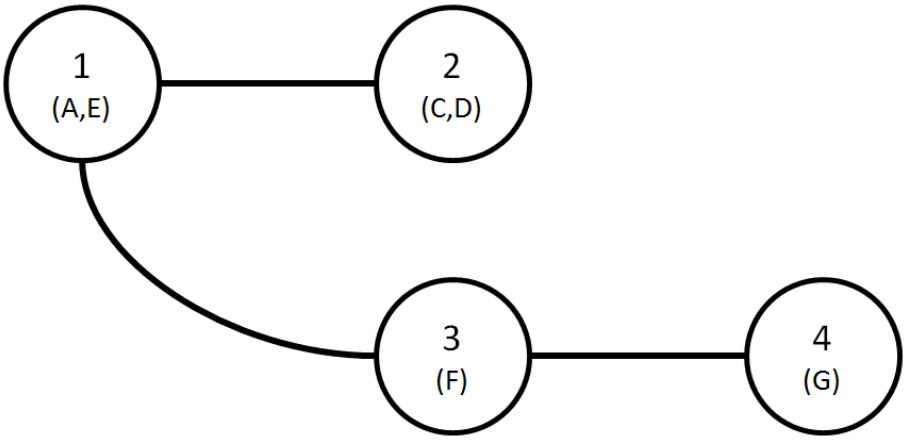


Figure 4. A graph of the abstract states visited by a student trying to solve a practice puzzle. The concrete states in Figure 3 (A, C, D, E, F, G) have been mapped to abstract states (1, 2, 3, 4).

graph of the states visited in Figure 3. Note that state B has not been added to the graph, because it is identical to state A and does not represent a step in the concrete state space. A walk through this graph, which is a sequence of the abstract states visited, represents the student's problem-solving behavior in this puzzle.

Each step in this walk represents a change in the concrete state, but not necessarily in the abstract state. This results in the following walk:

1, 2, 2, 1, 3, 4

which abstractly represents concrete states A, C, D, E, F, G. Finally, information about checks and plays can be added to create an annotated walk:

1, 2, check (incorrect), 2, play, 1, 3, 4, play, check (correct)

Such an annotated walk was created for every participant's attempt to solve each puzzle. Four measures, based on these annotated walks, were then calculated for each participant to capture different aspects of their problem-solving behavior. The first two measures look at the states a student visits before the first time she checks her solution and give an indication of how focused the student works towards this solution. The second two measures look at the states about which a student requests feedback and give an indication of how much the student relies on feedback instead of reasoning about the behavior of the gear configuration herself. In the following definitions 'states' refers to abstract states such as those shown in Figure 4, not the concrete states shown in Figure 3.

SBC (states before check) is a measure of the number of states a student visits before the first time she checks her answer. For the example annotated walk this number is two. One possibility for calculating this measure would be to use the average number of states visited over all the puzzles a student attempted. However, some students complete more puzzles than others and later puzzles require more steps to solve, which would affect the average. Therefore the number of states visited during each puzzle was normalized by calculating a z-score based on the number of states visited by all students who attempted this puzzle. Then the average z-score over all puzzles was calculated for each participant. This is a measure of how focused the student works towards an initial solution.

RBC (revisits before check) refers to the number of times a student visits a state that is identical to a state she has already visited, before the first time the student checks her answer. For the example annotated walk this number is zero, because the student only visits states 1 and 2 once before the first check. This is a second measure of how focused the student works towards a solution, that is not affected by the student using suboptimal solutions that require more steps than necessary. When a student revisits earlier states before checking whether her solution is correct, this may indicate that the student is not sure of how to solve the puzzle and is playing around with different gear configurations. This measure is calculated by averaging the fraction of visits that are revisits, before the first check, over all puzzles each participant completes.

SCWW (states checked when wrong) refers to the total number of states about which feedback is requested, by either checking the solution or pressing play, when an incorrect answer is given. In the example annotated walk this is done for three states. The reason that this statistic is only calculated for those puzzles where a student's initial answer is wrong, is that it would otherwise be heavily skewed by the fraction of practice puzzles the student solves on her first attempt, since by definition they only request feedback about one state for those puzzles. The number of states about which students request feedback after an initially incorrect solution is an indication of trial-and-error behavior. This measure is calculated by averaging the number of states about which a student requests feedback over all the puzzles for which she initially provided an incorrect solution.

RCWW (revisits checked when wrong) refers to the times feedback is requested about a state about which feedback has been requested before, when an incorrect answer is given. When feedback is requested about a state multiple times without changes in between, this is counted as requesting feedback once. In the example annotated walk, feedback is requested about state 2 two times and about state 4 once. This measure is expressed as a fraction, which for this attempt would be 1/3, because one out of three requests for feedback happens in a state about which the student has requested feedback

before. The measure is calculated by averaging this fraction over all the puzzles for which a student initially provided an incorrect solution.

A MANOVA was used to examine whether there was a difference in puzzle-solving behavior, as defined by these four measures, between the supported group and the control group. Stepwise regression was used to examine how well scores on these four measures predicted posttest puzzle scores and increases in puzzle scores from pretest to posttest.

4.3.2 Results

Participants in the supported condition correctly answered 76% of the practice questions on their first attempt. Participants in the control condition correctly answered 65% of the practice questions on their first attempt. Participants in the supported condition correctly answered questions on their first attempt significantly more often than those in the control condition, $t(1021) = 3.52$, $p < .001$. Figure 5 shows the proportion of participants in each condition that answered each question correctly on their first attempt.

Participants in the supported condition solved 47% of the practice puzzles on their first attempt. Participants in the control condition solved 39% of the practice puzzles on their first attempt. Participants in the supported condition solved puzzles on their first attempt significantly more often than those in the control condition, $t(989) = 2.50$, $p = .013$.

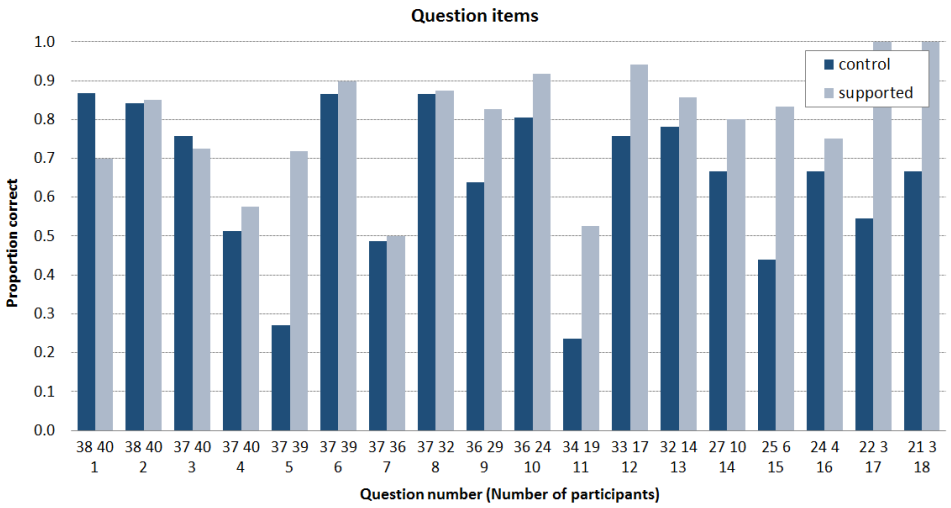


Figure 5. Proportion of students in each group that answered each practice question correctly on their first attempt. The number of participants in each group who completed each item is shown directly below the bars.

Figure 6 shows the proportion of participants in each condition that solved each puzzle correctly on their first attempt.

Out of a maximum of 36 practice items, students in the supported condition on average completed 21.33 (*SD* = 7.38). Students in the control condition on average completed 30.55 (*SD* = 7.66) practice items. Students in the control condition completed significantly more items than those in the supported condition, $t(76) = 5.42$, $p < .001$. Figures 5 and 6 also show the number of participants in both groups who completed each practice item.

The average scores for the question and puzzle items are listed in Table 3. There was a significant average improvement for participants in the supported condition on both the question items, $t(39) = 5.03$, $p < .001$, $d = 0.80$ and the puzzle items, $t(39) = 4.00$, $p < .001$, $d = 0.63$. For participants in the control condition there was also a significant average improvement on both the question items, $t(37) = 3.88$, $p < .001$, $d = 0.63$ and the puzzle items, $t(37) = 5.57$, $p < .001$, $d = 0.90$. There was no significant difference between the average improvement scores on the question items of the supported group and the control group, $t(76) = 0.74$, $p = .23$. There was also no significant difference between the two groups for the puzzle items, $t(76) = -0.418$, $p = .68$.

The average scores on the four puzzle-solving behavior measures for both groups are listed in Table 4. No significant multivariate effect of condition on the four puzzle-solving behavior measures was found, Hotelling's $T(4, 73) = 0.01$, $p = .93$.

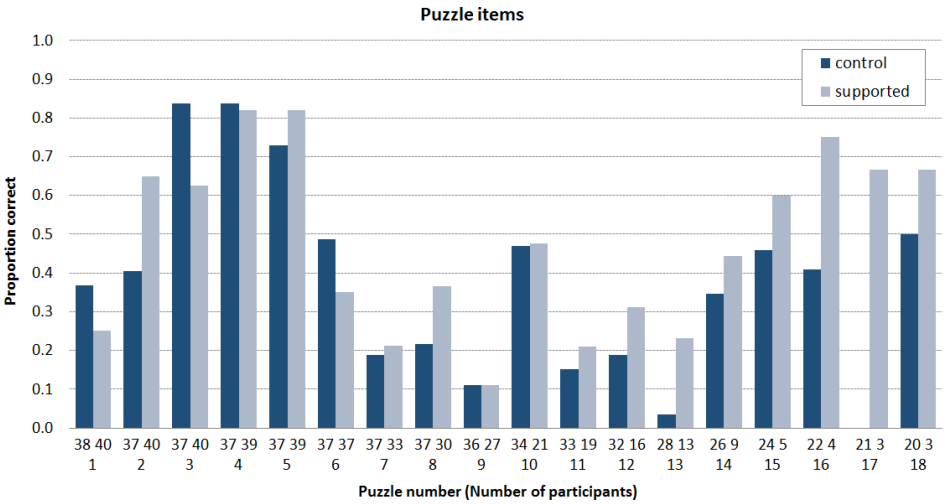


Figure 6. Proportions of students in each group that solved each practice puzzle correctly on their first attempt. The number of participants in each group who completed each item is shown directly below the bars.

Table 3
Pretest and posttest results and standard deviations for study 2

	Pretest questions	Posttest questions	Questions improvement	Pretest puzzles	Posttest puzzles	Puzzles improvement
Supported condition	9.83 (2.32)	12.02 (2.84)	2.20 (2.77)	0.80 (0.91)	1.75 (1.41)	0.95 (1.50)
Control condition	10.05 (2.47)	11.79 (2.87)	1.74 (2.76)	0.76 (0.88)	1.84 (1.15)	1.08 (1.19)

Table 4
Means and standard deviations on the four puzzle-solving behavior measures

	SBC	RBC	SCWW	RCWW
Supported condition	-0.03 (0.32)	0.24 (0.09)	2.87 (1.13)	0.10 (0.13)
Control condition	0.01 (0.38)	0.26 (0.09)	3.00 (0.77)	0.10 (0.11)

Table 5 shows the correlations between the four puzzle-solving behavior measures, the posttest puzzle score and the puzzle improvement score. A stepwise regression analysis with the four puzzle-solving behavior measures as independent variables and posttest puzzle score as a dependent variable, selected only RCWW as a significant predictor, b = -4.688, R^2 = .179, $p < .001$. A second stepwise regression analysis with the same independent variables and puzzle improvement score as the dependent variable did not select any of the independent variables as significant predictors.

Table 5
Pearson correlations between the four puzzle-solving behavior measures, posttest puzzle score (PPS) and puzzle improvement score (PIS)

	SBC	RBC	SCWW	RCWW	PPS	PIS
SBC						
RBC	.774*					
SCWW	.012	.088				
RCWW	-.032	.214	.486*			
PPS	.101	.038	-.143	-.423*		
PIS	.047	.096	.005	-.176	.772*	

* $p < .001$. None of the other correlations are significant at the .05 level.

4.3.3 Discussion

The changes to the support tool seem to have had the intended effect on the participants' performance on the practice items. The supported group significantly outperformed the control group on both the practice questions and practice puzzles. However, comparing the total proportions of correct questions and puzzles does not tell the whole story. Figures 5 and 6 show that the largest differences in proportion occur in the later items, which are generally more complex than the earlier ones. An optimistic interpretation would be that the reasoning support tool used by the experimental group helped participants most when they were attempting to answer complex questions. The structured reasoning that they applied to answer these questions could then also have helped them solve the more complex puzzles. However, the supported group on average completed fewer items than the control group. This may have led to a selection effect, where only the better students in the supported group reached the later, more complex items, whereas relatively more of the average students in the control group reached these items. This selection effect makes sense as an explanation for the difference in performance on the practice puzzles, but it seems unlikely to be the full explanation for the practice questions. The supported group outperformed the control group on 16 of the 18 questions and the proportions of questions which were answered correctly indicate that the later questions were not much more difficult than the early questions.

Like the first study, the results also show that both groups significantly improved their test scores from pretest to posttest. However, there was no significant difference between the improvement of the two groups on either the question items or the puzzle items. The supported group's improved performance on the practice items seems to not have significantly affected their learning outcomes. This finding is further supported by the analysis of four measures of puzzle-solving behavior. There are no indications that using the reasoning support tool affected participants' approach to solving the practice puzzles.

The negative correlation found between RCWW (revisits checked when wrong) and success on the posttest puzzles indicates that better students recheck solutions with the same underlying structure less often. Because posttest puzzle score did not significantly correlate with SCWW (states checked when wrong), it is unlikely that this finding is explained by higher performing students just checking fewer solutions altogether. This could mean that these higher performing students are better able to identify which gear configurations share the same underlying structure, which is an important skill for problem-solving in the gears domain.

4.4 General discussion

In our first study, the reasoning support tool caused students in the supported group to do worse on the practice questions than participants in the control group. After changing this tool to better integrate it with the practice questions students were trying to answer, based on observations during the first study, the second study showed that the reasoning support tool led to improved performance on the practice questions (RQ1). Using this improved tool also may have helped students do better on the practice puzzles (RQ2), although no differences were detected in their puzzle-solving behavior (RQ3). Both studies clearly show that students in both conditions learned from working with GearSketch (RQ4), but no evidence was found that students in the supported condition learned more than those in the control condition (RQ5).

The results of the two studies show that relatively minor changes to the reasoning support tool had a large impact on its effectiveness in helping students solve the practice questions. However, these improved results on the practice questions, did not lead to improved posttest scores. One possible explanation for this finding is that the version of the reasoning support tool used in the second study only allowed students to select the answer that followed from the reasoning steps they had selected. Although this feature forced students to change their incorrect reasoning steps when they made a mistake instead of being able to immediately select a different answer and continue, it also makes the practice items easier by not requiring students to think through the consequences of the reasoning steps on their own. On the posttest, when the support tool was not available, students had to conclude by themselves that the facts that gear 2 turns faster than gear 1 and gear 3 turns with the same speed as gear 2 imply that gear 3 turns faster than gear 1.

Another possible explanation for the finding that the supported group did not learn more despite doing better on the practice questions, is that students in the control group completed more practice problems. This extra practice with different questions and puzzles may have compensated for their poorer practice performance. This explanation could be tested by examining whether letting students in both conditions complete all practice items instead of fixing their time-on-task leads to improved learning outcomes for the supported group compared to the control group.

This chapter introduced four quantitative measures of students' problem-solving behavior to examine if students in the supported condition behaved differently when solving practice puzzles than those in the control condition. One of these measures, RCWW (revisits checked when wrong), correlated negatively with posttest performance on the puzzle questions. This finding indicates that above-average students check fewer isomorphic incorrect solutions than below-average students, suggesting that they may

have more insight into the abstract structure of gear configurations. These measures showed no evidence for a difference in problem-solving behavior between the conditions, which raises the question of whether there was in fact no difference or whether these four measures just didn't capture it. The interpretation that there was in fact no difference in the problem-solving behavior of the groups is consistent with two other findings. First, the supported group's improved practice puzzle performance can be explained by a selection effect that led to only the better students in the supported group attempting to solve the later, more difficult puzzles. Second, there was no difference in improvement on the test puzzles between the groups, which also indicates that the supported group did not use a more deliberate approach to solving these puzzles.

Future research may further investigate what kind of support is needed to encourage students to deliberately apply their declarative knowledge during problem solving. Asking students to indicate their reasoning steps during problem solving may not sufficiently stimulate them to explain to themselves why they choose to use these particular steps. Requiring students to explain why they choose to use their reasoning steps may lead to better results (Aleven & Koedinger, 2002; Chi, De Leeuw, Chiu, & Lavancher, 1994). Although the studies in this chapter do not show that using the reasoning support tool has a positive effect on learning, they do consistently demonstrate that students learn from working with GearSketch.

References

Aleven, V., & Koedinger, K. R. (2002). An effective metacognitive strategy: Learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science*, *26*(2), 147–179. doi:10.1016/S0364-0213(02)00061-7

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. doi:10.1037/a0021017

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, *111*(4), 1036–1060. doi:10.1037/0033-295X.111.4.1036

Anderson, J. R., Boyle, C. F., Corbett, A. T., & Lewis, M. W. (1990). Cognitive modeling and intelligent tutoring. *Artificial Intelligence*, *42*(1), 7–49. doi:10.1016/0004-3702(90)90093-F

Baker, R. S. J. D., Corbett, A. T., & Koedinger, K. R. (2004). Detecting Student Misuse of Intelligent Tutoring Systems. In J. C. Lester, R. M. Vicari, & F. Paraguaçu (Eds.), *Intelligent Tutoring Systems* (pp. 531–540). Springer Berlin Heidelberg.

Baker, R. S. J. D., Corbett, A. T., Roll, I., & Koedinger, K. R. (2008). Developing a

generalizable detector of when students game the system. *User Modeling and User-Adapted Interaction*, *18*(3), 287–314. doi:10.1007/s11257-007-9045-6

Chi, M. T. H., Bassok, M., Lewis, M. W., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science*, *13*(2), 145–182. doi:10.1207/s15516709cog1302_1

Chi, M. T. H., De Leeuw, N., Chiu, M.-H., & Lavancher, C. (1994). Eliciting self-explanations improves understanding. *Cognitive Science*, *18*(3), 439–477. doi:10.1207/s15516709cog1803_3

Corbett, A. T. (2001). Cognitive computer tutors: Solving the two-sigma problem. In M. Bauer, P. J. Gmytrasiewicz, & J. Vassileva (Eds.), *User Modeling 2001* (pp. 137–147). Springer Berlin Heidelberg.

Corbett, A. T., & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modelling and User-Adapted Interaction*, *4*(4), 253–278. doi:10.1007/BF01099821

De Jong, T., & Van Joolingen, W. R. (1998). Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research*, *68*(2), 179–201. doi:10.3102/00346543068002179

Desmarais, M. C., & Baker, R. S. J. d. (2012). A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction*, *22*(1-2), 9–38. doi:10.1007/s11257-011-9106-8

Donahue, T. J., Poor, G. M., Mott, M. E., Leventhal, L. M., Zimmerman, G., & Klopfer, D. (2013). On interface closeness and problem solving (pp. 139–146). Presented at the TEI 2013 - Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction. doi:10.1145/2460625.2460647

Eysink, T. H. S., & De Jong, T. (2012). Does instructional approach matter? How elaboration plays a crucial role in multimedia learning. *Journal of the Learning Sciences*, *21*(4), 583–625. doi:10.1080/10508406.2011.611776

Eysink, T. H. S., De Jong, T., Berthold, K., Kolloffel, B., Opfermann, M., & Wouters, P. (2009). Learner performance in multimedia learning arrangements: An analysis across instructional approaches. *American Educational Research Journal*, *46*(4), 1107–1149. doi:10.3102/0002831209340235

Hayes, N. A., & Broadbent, D. E. (1988). Two modes of learning for interactive tasks. *Cognition*, *28*(3), 249–276. doi:10.1016/0010-0277(88)90015-7

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. doi:10.1080/00461520701263368

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. doi:10.1207/s15326985ep4102_1

Klahr, D., & Dunbar, K. (1988). Dual space search during scientific reasoning. *Cognitive Science*, *12*(1), 1–48. doi:10.1207/s15516709cog1201_1

Koedinger, K. R., & Aleven, V. (2007). Exploring the assistance dilemma in experiments with cognitive tutors. *Educational Psychology Review*, *19*(3), 239–264. doi:10.1007/s10648-007-9049-0

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. doi:10.1037/0003-066X.59.1.14

Svendsen, G. B. (1991). The influence of interface style on problem solving. *International Journal of Man-Machine Studies*, *35*(3), 379–397. doi:10.1016/S0020-7373(05)80134-8

5

GENERAL DISCUSSION

This final chapter summarizes the findings of the studies that were conducted with GearSketch and discusses possible directions for future research. It concludes with some final thoughts on the work discussed in this dissertation.

5.1 Summary of experimental findings

The core chapters of this dissertation are Chapters 2, 3 and 4. Each of these chapters discusses different features of GearSketch and experimental studies that evaluate the educational value of these features.

Chapter 2 discusses GearSketch's simulation-based support. This support is based on a model of the gears domain that lets GearSketch interpret students' drawings of gears and chains, ensure that these drawings constitute valid gear systems and animate the behavior of these systems. To examine the value of this support, an experimental study with 74 fifth grade students was conducted in which a supported group worked with a version of GearSketch with simulation-based support and a control group worked with a static version of the learning environment. This study showed that the supported group significantly outperformed the control group during practice, on a direct posttest and on a delayed posttest. These results indicate that the kind of simulation-based support offered by GearSketch facilitates learning.

Chapter 3 discusses GearSketch's domain model and drawing-based interface in further detail, introduces a learner model based on the domain model and describes how this learner model can be used to select practice problems tailored to individual learner's understanding of the domain. The learner model is a Bayesian overlay model (Chrysafiadi & Virvou, 2013; Corbett & Anderson, 1995) that represents learners' understanding of the rules which govern the gears domain and their ability to recognize when these rules are relevant during problem solving. This model is updated for each individual student after each practice problem she attempts, based on the rules that are relevant for solving this problem and whether the student's solution to the problem is correct. The learner model can make predictions about learners' abilities to solve a particular practice problem, and these predictions are used to adaptively select practice problems from which students can benefit the most. Most practice problems are defined abstractly and each time a learner attempts to solve such a problem, a new concrete representation is created that is structurally similar to other versions of this problem, but has different surface features. This prevents students from learning the solutions to these problems by rote instead of learning how to solve this type of problem in general. An experimental study with 44 fifth grade students showed no effect on learning outcome of adaptive practice problem selection compared to fixed problem selection. A possible explanation for this lack of effect, is the low accuracy of the learner model's predictions of students' performance on individual items.

Chapter 4 discusses a reasoning support tool for GearSketch's practice questions. This reasoning support tool requires students to make each step in their reasoning process explicit before they can select and check their answer to a question. The goal of this

support tool was to ensure that learners deliberately apply their declarative domain knowledge during the practice questions, so that they can acquire related procedural knowledge. A first experiment with 52 sixth grade students showed that using this reasoning support tool actually had a negative effect on learner's performance on the practice questions. Observations indicated that this was due to the fact that the tool was not sufficiently integrated with the questions, which caused learners to go through the motions of indicating each reasoning step without considering how these steps related to the problem they were solving. Changes were made to the support tool to better integrate it with the practice questions and a second experiment with 78 fifth and sixth grade students was conducted. This experiment showed that participants who used the support tool performed better on the practice questions than students in a control group that could not use this tool. Students in the supported group also solved puzzles relatively more often on their first attempt than those in the control group, but this finding could be explained by the fact that the supported group used more time to answer the questions, which meant that they did not get to the later, more difficult practice puzzles. To examine whether using the reasoning support tool during the practice questions affected students' puzzle-solving behavior, four measures were defined to quantify puzzle-solving behavior. These measures showed no evidence for differences in puzzle-solving behavior between the two groups. One of these measures did give insight into behavior of successful students: they checked different isomorphic solutions significantly less often than less successful students. This indicates that recognizing whether two gear systems with different surface features are isomorphic is an important skill in the gear domain. Finally, the second experiment showed that the problem solving ability in the gears domain of both groups improved significantly from a pretest to a posttest. However, the supported group did not show greater improvement than the control group.

The findings from these studies show that students learn from practicing with GearSketch's drawing-based simulations, but that so far attempts to model and support this learning process have not been effective.

5.2 Directions for future research

Three different directions for future research are discussed: improving support for learners working with GearSketch, using other instructional approaches with GearSketch's drawing-based simulation and different digital drawing-based learning environments.

5.2.1 Improving support

The experimental studies discussed in Chapters 2, 3 and 4 show that simulation-based support facilitated learning, but that adaptive practice problem selection and a reasoning support tool had no effect on learning outcome. To better understand why adaptive problem selection and supporting step-by-step reasoning did not improve learning outcomes, more insight into learners' reasoning processes is needed. An effective way to gain this insight is to ask learners to think aloud while they are solving problems in the gears domain, but this has to be done carefully since asking learners to verbalize their thoughts could affect their behavior (Ericsson & Simon, 1980, 1998). To avoid effects on problem-solving behavior, a retrospective think-aloud approach could also be used (Van Den Haak, De Jong, & Schellens, 2003). This could be achieved by using GearSketch's logging facilities to record students' problem-solving behavior and then asking students to explain their thought process while replaying these logs.

Insights gained from such a study could then be used to improve the learner model. A promising approach would be to extend the current overlay model with nodes that represent common misconceptions to create a perturbation model (Chrysafiadi & Virvou, 2013; Desmarais & Baker, 2012). This extended model could use the same structure as the current Bayesian network model, but updating such a model will require more information than updating an overlay model (Goguadze, Sosnovsky, Isotani, & McLaren, 2011). To efficiently identify which misconception caused a student to make a mistake on a particular problem, information about whether a problem was solved correctly or not is not enough. Information about which specific solution the student provided is also needed. Log files collected during the experiments described in Chapters 2, 3 and 4 include this information and could be used to estimate initial values for the nodes in such a learner model. This model can then be used to give targeted feedback to learners when they make a mistake that has likely been caused by a specific misconception.

The reasoning support tool discussed in Chapter 4 aimed to encourage learners to reason more deliberately while answering practice questions. The finding that this support tool did not seem to affect learners' reasoning during the (unsupported) practice puzzles and did not result in a greater learning effect indicates that this goal was not achieved. Although the reasoning support tool required learners to indicate which reasoning steps were necessary to find the answer, it did not ask learners to explain why they used these steps. Self-explanation, a possible mechanism through which such a reasoning support tool might affect learning (Aleven & Koedinger, 2002; Chi, Bassok, Lewis, Reimann, & Glaser, 1989; Chi, De Leeuw, Chiu, & Lavancher, 1994), may therefore not have been sufficiently encouraged. Explicitly asking students to explain why they chose each of their steps may be more effective. Additionally, students may be shown worked-out

examples to study before starting on the practice questions themselves (Renkl, 2002). Introducing step-by-step support for the practice puzzles may also be effective if a well-thought out implementation is found that can track students' solution steps without making the drawing-based interaction more cumbersome.

5.2.2 Other instructional approaches

Instead of building on the work discussed in this dissertation by trying to improve elements of GearSketch that have not been shown to work, elements that do work could be used in different contexts. Chapter 2 showed that working with GearSketch's drawing-based simulation facilitates learning. Instead of using tutorials to explain the declarative domain rules and then letting students practice with those rules in questions and puzzles, the drawing-based simulation could be used as the core of an inquiry learning environment or an educational game. Both of these instructional approaches have been shown to lead to better attitudes towards learning and higher cognitive gains than traditional teaching methods (Vogel et al., 2006).

An effective inquiry learning environment based on GearSketch would not just provide learners with the simulation and let them figure out how gears and chains interact by themselves, but would guide them in their exploration of the domain (De Jong & Van Joolingen, 1998; Mayer, 2004). It is easy to imagine simple tasks that help students discover the basic rules of the domain and more complex tasks that assist students in figuring out how to build systems that achieve specific goals, like gear chains that greatly increase rotational velocity. A meta-analysis by Alfieri, Brooks, Aldrich and Tenenbaum (2011) showed the effectiveness of such a guided discovery approach.

GearSketch's puzzles already contain a number of elements that are commonly seen in educational games. They present a clear goal, set restrictions to make achieving this goal challenging and provide both implicit feedback through animations and explicit feedback by telling students whether their solutions are correct (Kiili, 2005). Embedding these puzzles in interesting settings, such as trying to build a gear box so that your car is faster than an opponent's car, or aligning multiple gears so that a safe can be opened may make their dynamics and aesthetics more interesting (Aleven, Myers, Easterday, & Ogan, 2010). Additional gaming elements, like giving learners badges (Denny, 2013) for achieving goals such as quickly and correctly solving multiple puzzles in a row may further enhance learner's motivation to work with the learning environment.

5.2.3 Digital drawing-based approaches to education

The increasing availability of pen-based and touchscreen devices in schools makes it

possible to use digital drawing-based learning environments in education. GearSketch is based on a model of a single domain, which allows the interface to interpret learners' pen strokes without any additional labeling being necessary. Such an approach is suitable for domains that have a limited number of objects that interact in sufficiently complex ways that learners benefit from being able to simulate this interaction. Ropes and pulleys are another example of such a domain. Of course the drawback of such an approach to digital drawing-based learning environments, is that a new environment will have to be designed and implemented for each domain. Two examples of drawing-based learning environments that are more domain general are CogSketch and SimSketch.

CogSketch (Forbus, Usher, Lovett, Lockwood, & Wetzel, 2011) represents every object that students draw as a glyph. These glyphs consist of one or more pen strokes and a symbolic token that represents what the glyph depicts. Students interact with CogSketch by creating a sketch, segmenting this sketch into glyphs and then labeling these glyphs with concepts from a knowledge base. CogSketch can then integrate information about the spatial representation of glyphs (e.g. whether a glyph is to the left of another glyph or is contained by it) and its knowledge of the objects represented by these glyphs to create a model of students' drawings. By comparing such a model to a model created by an instructor, CogSketch is able to recognize whether elements are missing from the student's drawing or relations are depicted incorrectly. For example, a worksheet for a physical geology class contains a photograph of a wall with a fault, which students have to annotate by adding glyphs. CogSketch can then automatically compare the glyphs added by a student to those added by an instructor and give feedback when parts of the photograph are incorrectly marked. A big advantage of CogSketch is that it can allow teachers to create such assignments for many different domains without the need for advanced programming skills.

Students also interact with SimSketch (Bollen & Van Joolingen, 2013) by segmenting their drawing into objects and labeling these objects. However, in this case the labels provide the objects with dynamic behavior that can be seen by running a simulation in which these objects are animated. For example, by drawing a fox, a rabbit and a patch of grass, and adding labels to these objects to define how they multiply, move and interact with each other, learners can create a predator-prey model. This model can be animated to explore the effects of different starting positions or different parameters for their behavior. Another example is making a sketch of a road and drawing some cars on it. Labels can then be used to indicate that these cars should stay on the road and move in a given direction to explore how traffic jams emerge and clear up. A relatively small number of different labels can allow learners to create complex models in a multitude of different domains.

These examples of drawing-based learning environments show some of the possibilities that arise with the increasing popularity of pen-based and touchscreen devices in educational settings. While GearSketch's approach works well in domains that contain a limited number of different objects that behave according to well-defined rules, the approaches used by CogSketch and SimSketch are more suitable for domains with larger numbers of different interacting objects. These three learning environments demonstrate a few of the ways in which ideas from drawing-based learning can be extended by turning students' drawings into computational models and providing students with different types of feedback based on these models.

5.3 Concluding thoughts

During the development of a new learning environment many design decisions are made which are not directly related to the research questions addressed by experimental studies with this learning environment. These design decisions are about simple things like colors and font sizes and about more complex things like the way in which students add a chain to their system of gears. Additionally, choices are made about the learning environment's domain and target group. When an experiment shows that a certain feature of a learning environment facilitates learning, it is not always clear why this is the case. Did this feature happen to suit the specific domain or did it fit particularly well with the knowledge and skills of the experiment's participants? On the other hand, when a feature does not improve learning, is it because this feature was based on unsound principles or because the learning environment's implementation of the feature was lacking? It is impossible to conclusively demonstrate or disprove the value of instructional theories, principles and approaches with small-scale experiments in a single domain like the ones discussed in this dissertation. The best result one can hope to achieve with such studies is to provide a few new ideas and data points. The research discussed in this dissertation has shown that a learning environment based on drawing and simulation can facilitate learning in the gears domain, but attempts to further improve learning outcomes by modeling and supporting students' learning processes have not yet been successful.

References

Aleven, V., & Koedinger, K. R. (2002). An effective metacognitive strategy: Learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science*, *26*(2), 147–179. doi:10.1016/S0364-0213(02)00061-7

Aleven, V., Myers, E., Easterday, M., & Ogan, A. (2010). Toward a framework for the analysis and design of educational games (pp. 69–76). Presented at the DIGITEL 2010 - The 3rd IEEE International Conference on Digital Game and

Intelligent Toy Enhanced Learning. doi:10.1109/DIGITEL.2010.55

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. doi:10.1037/a0021017

Bollen, L., & Van Joolingen, W. R. (2013). SimSketch: Multiagent simulations based on learner-created sketches for early science education. *IEEE Transactions on Learning Technologies*, *6*(3), 208–216. doi:10.1109/TLT.2013.9

Chi, M. T. H., Bassok, M., Lewis, M. W., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science*, *13*(2), 145–182. doi:10.1207/s15516709cog1302_1

Chi, M. T. H., De Leeuw, N., Chiu, M.-H., & Lavancher, C. (1994). Eliciting self-explanations improves understanding. *Cognitive Science*, *18*(3), 439–477. doi:10.1207/s15516709cog1803_3

Chrysafiadi, K., & Virvou, M. (2013). Student modeling approaches: A literature review for the last decade. *Expert Systems with Applications*, *40*(11), 4715–4729. doi:10.1016/j.eswa.2013.02.007

Corbett, A. T., & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modelling and User-Adapted Interaction*, *4*(4), 253–278. doi:10.1007/BF01099821

De Jong, T., & Van Joolingen, W. R. (1998). Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research*, *68*(2), 179–201. doi:10.3102/00346543068002179

Denny, P. (2013). The effect of virtual achievements on student engagement (pp. 763–772). Presented at the Conference on Human Factors in Computing Systems - Proceedings. doi:10.1145/2470654.2470763

Desmarais, M. C., & Baker, R. S. J. d. (2012). A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction*, *22*(1-2), 9–38. doi:10.1007/s11257-011-9106-8

Ericsson, K. A., & Simon, H. A. (1980). Verbal reports as data. *Psychological Review*, *87*(3), 215–251. doi:10.1037/0033-295X.87.3.215

Ericsson, K. A., & Simon, H. A. (1998). How to study thinking in everyday life: Contrasting think-aloud protocols with descriptions and explanations of thinking. *Mind, Culture, and Activity*, *5*(3), 178–186. doi:10.1207/s15327884mca0503-3

Forbus, K., Usher, J., Lovett, A., Lockwood, K., & Wetzel, J. (2011). CogSketch: Sketch understanding for cognitive science research and for education. *Topics in Cognitive Science*, *3*(4), 648–666. doi:10.1111/j.1756-8765.2011.01149.x

Goguadze, G., Sosnovsky, S., Isotani, S., & McLaren, B. M. (2011). Evaluating a Bayesian student model of decimal misconceptions (pp. 301–305). Presented at

the EDM 2011 - Proceedings of the 4th International Conference on Educational Data Mining.

Kiili, K. (2005). Digital game-based learning: Towards an experiential gaming model. *The Internet and Higher Education*, *8*(1), 13–24. doi:10.1016/j.iheduc.2004.12.001

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. doi:10.1037/0003-066X.59.1.14

Renkl, A. (2002). Worked-out examples: Instructional explanations support learning by self-explanations. *Learning and Instruction*, *12*(5), 529–556. doi:10.1016/S0959-4752(01)00030-5

Van Den Haak, M. J., De Jong, M. D. T., & Schellens, P. J. (2003). Retrospective vs. concurrent think-aloud protocols: Testing the usability of an online library catalogue. *Behaviour and Information Technology*, *22*(5), 339–351. doi:10.1080/0044929031000

Vogel, J. J., Vogel, D. S., Cannon-Bowers, J., Bowers, G. A., Muse, K., & Wright, M. (2006). Computer gaming and interactive simulations for learning: A meta-analysis. *Journal of Educational Computing Research*, *34*(3), 229–243. doi:10.2190/FLHV-K4WA-WPVQ-H0YM

SUMMARY OF THE RESEARCH

ENGLISH SUMMARY

This dissertation discusses the implementation and evaluation of GearSketch, a learning environment for the gears domain aimed at students in the final years of primary school. GearSketch is built around a model of the gears domain that represents how gears and chains can be connected to each other and how these connections transmit motion between objects. When working with GearSketch students use a pen-based or touchscreen computer to draw their own gear and chain systems. The learning environment ensures that only valid systems (i.e. systems in which the gears can turn) can be created. Once students have created an interesting configuration, they can explore its behavior by starting a simulation and watching an animation of the gears and chains in motion.

GearSketch provides students with guidance during their exploration of the gears domain by incorporating tutorials and practice problems. When students start working with GearSketch, they first go through the tutorials. These tutorials introduce the rules which govern the gears domain and explain how to use the interface to draw, move and remove gears and chains. For example, when chains are introduced the tutorials will explain that two gears that are on the inside of a chain will turn in the same direction and students will practice connecting gears by drawing chains around them. After completing the tutorials, students apply their new knowledge about the domain by practicing with questions and puzzles. Questions ask students to make predictions about a given configuration of gears and/or chains, such as "will gear A turn faster, with the same speed or slower than gear B?" Puzzles present students with an incomplete gear system and ask them to add or move gears or chains to achieve a particular goal. For example, a small gear and a large gear may be shown and the student may be given the task of adding extra gears in such a way that the large gear will turn faster than the small gear. To solve such puzzles, students need to know the relevant domain rules and recognize that these rules can be used to achieve their goal. A number of experiments were done to examine how different features of GearSketch affected students' learning outcomes.

Chapter 2 discusses GearSketch's simulation-based support. This support interprets students' pen strokes to convert them to gears and chains, ensures that the gear configuration remains valid and can show students animations of the gears and chains in motion. An experimental study with 74 fifth grade students was conducted in which participants were randomly assigned to either the supported condition or a control condition. The control condition used a version of GearSketch that was mostly static to simulate working with paper and pencil. Participants worked with the software for 60

minutes and afterwards completed a paper-and-pencil posttest. Two weeks later participants completed a second paper-and-pencil posttest. This study showed that participants in the supported condition outperformed those in the control condition on the practice problems in GearSketch and scored higher on both the direct posttest and on the delayed posttest. These findings demonstrate that the simulation-based support offered by GearSketch facilitates learning in the gears domain.

Chapter 3 introduces a learner model based on GearSketch's domain model and discusses how this learner model can be used to adaptively select practice problems for students. The learner model is an overlay model that represents students' knowledge of the domain rules and their ability to recognize when and how these rules can be used to solve practice problems. After each practice problem a student completes, the learner model is updated to represent this student's new knowledge state. This is done by keeping track of the rules that are relevant for each practice problem and using Bayesian inference to calculate the new probability that the student knows these rules, given the learner model's estimate of whether she knew the rules before she attempted this practice problem and whether her solution was correct. Bayesian inference can also be used to predict the probability that a student will solve a given practice problem given her current knowledge state, which allows GearSketch to adaptively select practice problems of appropriate difficulty for individual students. An experimental study with 44 fifth grade students was conducted to examine whether adaptively selecting practice problems led to a greater learning effect than providing a fixed sequence of practice problems. Participants were randomly assigned to either the adaptive or the fixed condition and worked with GearSketch for 60 minutes. Afterwards they completed a posttest. The results showed that there was no significant difference between the posttest scores of the adaptive and the fixed condition. A possible explanation for this lack of effect is the low accuracy of the learner model's predictions of students' success on the posttest items. If the learner model cannot accurately predict students' performance, selecting practice problems based on this learner model is probably not much better than using a reasonable fixed sequence of practice problems.

Chapter 4 introduces a reasoning support tool that attempts to encourage students to deliberately use their declarative knowledge during problem solving so that they may acquire related procedural knowledge. This support tool asks students to explicitly show each reasoning step when they are answering the practice questions. Two experimental studies were done to evaluate the effects of using this support tool. The first study, in which 52 sixth grade students participated, showed that using the support tool actually had a negative impact on the supported group's ability to correctly answer the practice questions. Observations made during the experiment indicated that this was likely due to the fact that the tool was not well integrated with the practice questions, which caused learners to go through the motions of selecting reasoning steps, without thinking about

the question they were trying to answer. Changes were made to the support tool to better integrate it with the questions and a second experiment with 78 fifth and sixth grade students was conducted to see if these changes improved the tool. In this experiment participants were again randomly assigned to either the supported group or a control group that could not use the tool and this time the supported group outperformed the control group on the practice items. However, no differences between the groups were found in puzzle-solving behavior. Scores of both groups increased significantly from pretest to posttest, but the supported group did not improve more than the unsupported group. This indicates that the reasoning support tool did not facilitate learning.

Together these results show that a learning environment based on drawing and simulation can facilitate learning in the gears domain, but attempts to further improve learning outcomes by modeling and supporting students' learning processes have not yet been successful.

NEDERLANDSE SAMENVATTING (DUTCH SUMMARY)

Dit proefschrift beschrijft de implementatie en evaluatie van GearSketch, een leeromgeving over tandwielen gericht op leerlingen in de bovenbouw van de basisschool. GearSketch is gebouwd rondom een model van het domein van tandwielen dat representeert hoe tandwielen en kettingen met elkaar verbonden kunnen worden en hoe deze verbindingen beweging overbrengen. Leerlingen die met GearSketch werken, gebruiken een touchscreen computer of pentablet om hun eigen systeem van tandwielen en kettingen te tekenen. De leeromgeving zorgt ervoor dat alleen valide systemen (systemen waarin de tandwielen en kettingen kunnen draaien) kunnen worden gecreëerd. Wanneer leerlingen een interessant systeem hebben gemaakt, kunnen ze het gedrag van dit systeem verkennen door een simulatie te starten en te kijken hoe de tandwielen en kettingen draaien.

GearSketch biedt leerlingen ondersteuning terwijl ze het domein van tandwielen verkennen met behulp van tutorials en oefenopgaven. Wanneer leerlingen met GearSketch beginnen te werken, gebruiken ze eerst de tutorials. Deze introduceren de regels die het domein van tandwielen beschrijven en leggen uit hoe leerlingen de interface kunnen gebruiken om tandwielen en kettingen te tekenen, verplaatsen en verwijderen. Wanneer bijvoorbeeld kettingen worden geïntroduceerd, leggen de tutorials uit dat twee tandwielen aan de binnenkant van een ketting dezelfde kant op draaien en laten ze leerlingen oefenen met het verbinden van tandwielen door er kettingen omheen te tekenen. Nadat leerlingen alle tutorials hebben gevolgd, passen ze hun nieuwe kennis toe door te oefenen met vragen en puzzels. Vragen laten leerlingen voorspellingen maken over een systeem van tandwielen en kettingen, zoals “zal tandwiel A sneller dan, even snel als of langzamer dan tandwiel B draaien?” Puzzels geven leerlingen een incompleet systeem van tandwielen en vragen leerlingen om een bepaalde taak uit te voeren. Er kunnen bijvoorbeeld een klein en een groot tandwiel getoond worden en de leerling kan dan de taak krijgen om tandwielen toe te voegen zodat het grote tandwiel sneller gaat draaien dan het kleine tandwiel. Om dit soort puzzels op te lossen, hebben leerlingen kennis nodig van de relevante domeinregels en moeten ze inzien dat deze regels gebruikt kunnen worden om de taak te volbrengen. Er is een aantal experimenten uitgevoerd om te onderzoeken hoe verschillende eigenschappen van GearSketch de leeropbrengst van het werken met GearSketch beïnvloeden.

Hoofdstuk 2 beschrijft de op simulaties gebaseerde ondersteuning die GearSketch biedt. Deze ondersteuning interpreteert de tekeningen van leerlingen en vertaalt ze naar systemen van tandwielen en kettingen, zorgt ervoor dat deze systemen valide blijven en kan leerlingen simulaties laten zien waarin de tandwielen en kettingen draaien. Een experimentele studie met 74 leerlingen uit groep 7 werd uitgevoerd waarin deelnemers willekeurig werden toegewezen aan de ondersteunde conditie of aan een controleconditie. De controleconditie gebruikte een versie van GearSketch die grotendeels statisch was, zodat deze vergelijkbaar was met werken met papier en potlood. Deelnemers werkten gedurende 60 minuten met de software en maakte daarna een papieren natoets. Twee weken later maakten deelnemers een tweede natoets. Deze studie liet zien dan deelnemers in de ondersteunde conditie beter presteerden dan deelnemers in de controleconditie op de oefenopgaven in GearSketch en hoger scoorden op zowel de directe natoets als de uitgestelde natoets. Deze bevindingen tonen aan dat de op simulaties gebaseerde ondersteuning die GearSketch biedt het leren over tandwielen effectief kan ondersteunen.

Hoofdstuk 3 introduceert een leerlingmodel dat gebaseerd is op het domeinmodel van GearSketch en bespreekt hoe dit leerlingmodel gebruikt kan worden om adaptief oefenopgaven te selecteren voor leerlingen. Het leerlingmodel is een zogeheten ‘overlay model’ dat weergeeft welke kennis leerlingen hebben van de domeinregels en hoe goed ze inzien wanneer en hoe deze regels gebruikt kunnen worden om oefenopgaven op te lossen. Na iedere oefenopgave die een leerling maakt, wordt het leerlingmodel bijgewerkt om de nieuwe kennisstatus van de leerling te representeren. Dit wordt gedaan door bij te houden welke domeinregels relevant zijn voor iedere oefenopgave en Bayesiaanse inferentie te gebruiken om de nieuwe kans te berekenen dat de leerling deze regels kent, gegeven de vorige inschatting van het model van de kennis van de leerling en of de oplossing van de leerling correct was. Bayesiaanse inferentie kan ook gebruikt worden om met behulp van het leerlingmodel de kans te voorspellen dan een leerling een bepaalde oefenopgave correct zal oplossen, wat GearSketch helpt bij het selecteren van oefenopgaven met een geschikte moeilijkheidsgraad. Een experimentele studie met 44 leerlingen uit groep 7 werd uitgevoerd om te onderzoeken of leerlingen effectiever leerden wanneer ze oefenden met opgaven die voor hen werden gekozen op basis van de inschatting van het leerlingmodel van hun huidige kennis in plaats van met een vaste lijst van oefenopgaven. Leerlingen werden willekeurig toegewezen aan de adaptieve conditie of aan een controleconditie met een vaste volgorde van oefenopgaven en oefenden gedurende 60 minuten met GearSketch. Na het oefenen maakten ze een natoets. De resultaten van dit onderzoek lieten zien dat er geen verschil was tussen scores op de natoets van de twee groepen. Een mogelijke verklaring voor dit resultaat is dat de voorspellingen van het leerlingmodel over hoe succesvol leerlingen zouden zijn in het oplossen van opgaven onnauwkeurig waren. Wanneer het

leerlingmodel de prestaties van leerlingen niet goed kan voorspellen, zal het selecteren van oefenopgaven op basis van de voorspellingen van dit leerlingenmodel waarschijnlijk niet veel effectiever zijn dan het aanbieden van een vaste reeks van oefenopgaven.

Hoofdstuk 4 introduceert een instrument dat leerlingen aanmoedigt om bewust hun declaratieve kennis van de domeinregels te gebruiken tijdens het oplossen van de oefenopgaven om zo procedurele domeinkennis op te doen. Dit instrument vraagt leerlingen om expliciet te laten zien welke redeneerstappen ze gebruiken tijdens het beantwoorden van de oefenvragen. Twee experimentele studies werden uitgevoerd om de effecten van het gebruik van dit instrument te evalueren. De eerste studie, waaraan 52 leerlingen uit groep 8 meededen, liet zien dat het gebruik van dit instrument een negatieve impact had op de prestaties op de oefenvragen van de ondersteunde groep. Observaties die gedaan waren tijdens deze studie gaven aan dat dit waarschijnlijk veroorzaakt werd door het feit dat het instrument niet goed geïntegreerd was met de oefenvragen, wat ervoor zorgde dat leerlingen de redeneerstappen aangaven met het instrument zonder na te denken over de vraag die ze moesten beantwoorden. Er werden aanpassingen gemaakt aan het instrument om het beter met de vragen te integreren en een tweede studie met 78 leerlingen uit groep 7 en groep 8 werd uitgevoerd om te onderzoeken of deze aanpassingen het instrument verbeterden. In dit experiment werden leerlingen opnieuw willekeurig toegewezen aan een ondersteunde conditie of aan een controleconditie die geen gebruik kon maken van het instrument en dit keer presteerde de ondersteunde groep beter op de oefenvragen dan de controlegroep. Er werden echter geen verschillen gevonden tussen de groepen qua probleemoplossend gedrag tijdens het oplossen van de oefenpuzzels. De scores van beide groepen namen significant toe van een voortoets naar een natoets, maar de ondersteunde groep maakte niet meer vooruitgang dan de controlegroep. Dit geeft aan de het instrument niet tot betere leerprestaties leidde.

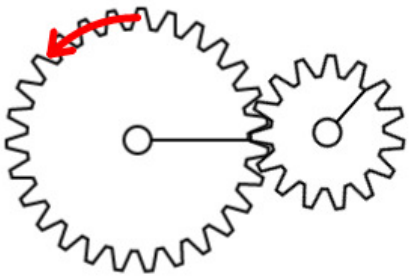
Samen laten deze studies zien dat een leeromgeving die gebaseerd is op tekenen en simulatie, leren in het domein van tandwielen effectief kan ondersteunen, maar pogingen om de leeropbrengst te vergroten door het leerproces te modelleren en ondersteunen zijn nog niet succesvol geweest.

APPENDIX

APPENDIX A
SUMMARY OF THE EXPLANATION

Meshing gears

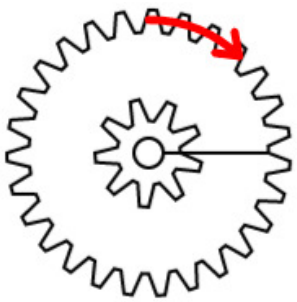
Meshing gears turn in opposite directions.



Meshing gears of the same size turn with the same speed. When two meshing gears have different sizes, the small gear will turn faster than the large gear.

Gears sharing a common axis

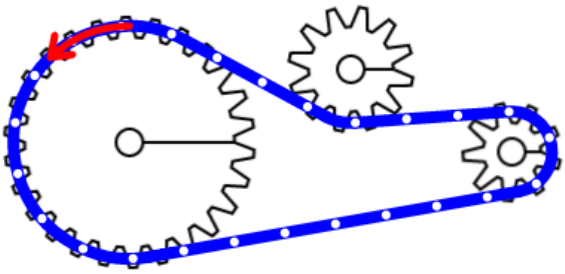
Gears that share a common axis turn in the same direction.



Gear that share a common axis always have the same turning speed.

Gears connected by a chain

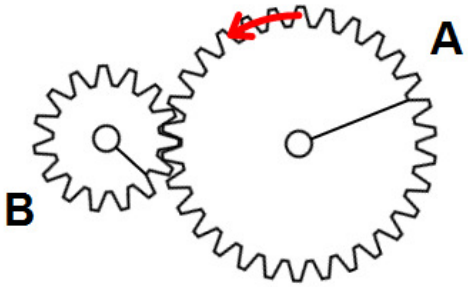
When gears are connected by a chain, all gears on the inside of the chain turn in the same direction. All gears on the outside of the chain turn in the opposite direction.



Gears of the same size turn with the same speed when they are connected by a chain. When gears of different sizes are connected by a chain, the small gears will turn faster than the large gears.

APPENDIX B
PRETEST

Question 1

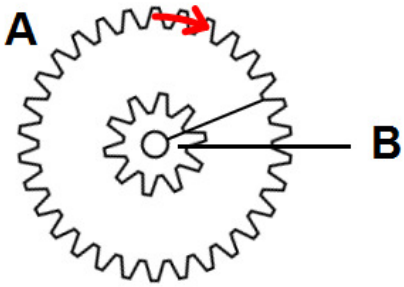


Gear A turns to the left (counterclockwise).

1.1 In which direction does gear B turn? (circle the correct answer)
   a. To the left (counterclockwise).
   b. To the right (clockwise).

1.2 How fast does gear B turn compared to gear A? (circle the correct answer)
   a. Gear B turns slower than gear A.
   b. Gear B turns with the same speed as gear A.
   c. Gear B turns faster than gear A.

Question 2



Gear A turns to the right (clockwise).

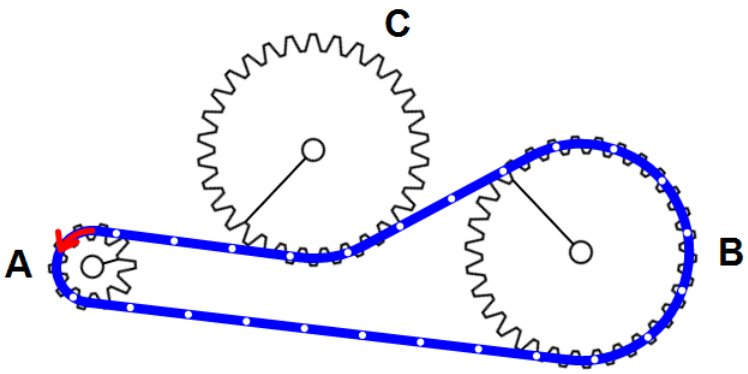
2.1 In which direction does gear B turn?

a. To the left (counterclockwise).
b. To the right (clockwise).

2.2 How fast does gear B turn compared to gear A?

a. Gear B turns slower than gear A.
b. Gear B turns with the same speed as gear A.
c. Gear B turns faster than gear A.

Question 3



Gear A turns to the left (counterclockwise).

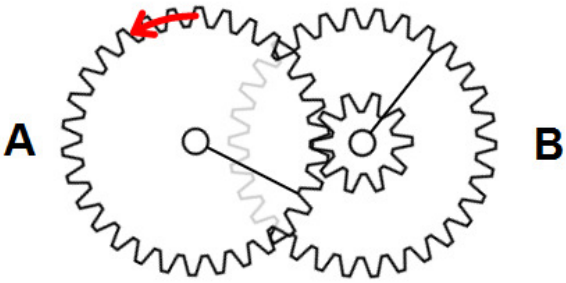
3.1 In which direction does **gear B** turn?
- a. To the left (counterclockwise).
- b. To the right (clockwise).

3.2 How fast does **gear B** turn compared to gear A?
- a. Gear B turns slower than gear A.
- b. Gear B turns with the same speed as gear A.
- c. Gear B turns faster than gear A.

3.3 In which direction does **gear C** turn?
- a. To the left (counterclockwise).
- b. To the right (clockwise).

3.4 How fast does **gear C** turn compared to gear A?
- a. Gear B turns slower than gear A.
- b. Gear B turns with the same speed as gear A.
- c. Gear B turns faster than gear A.

Question 4



Gear A turns to the left (counterclockwise).

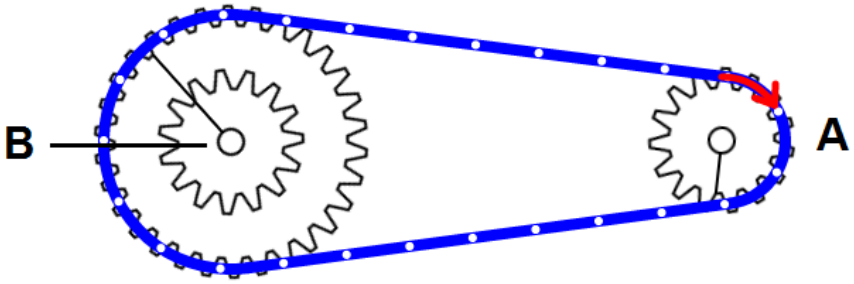
4.1 In which direction does gear B turn?

a. To the left (counterclockwise).
b. To the right (clockwise).

4.2 How fast does gear B turn compared to gear A?

a. Gear B turns slower than gear A.
b. Gear B turns with the same speed as gear A.
c. Gear B turns faster than gear A.

Question 5



Gear A turns to the right (clockwise).

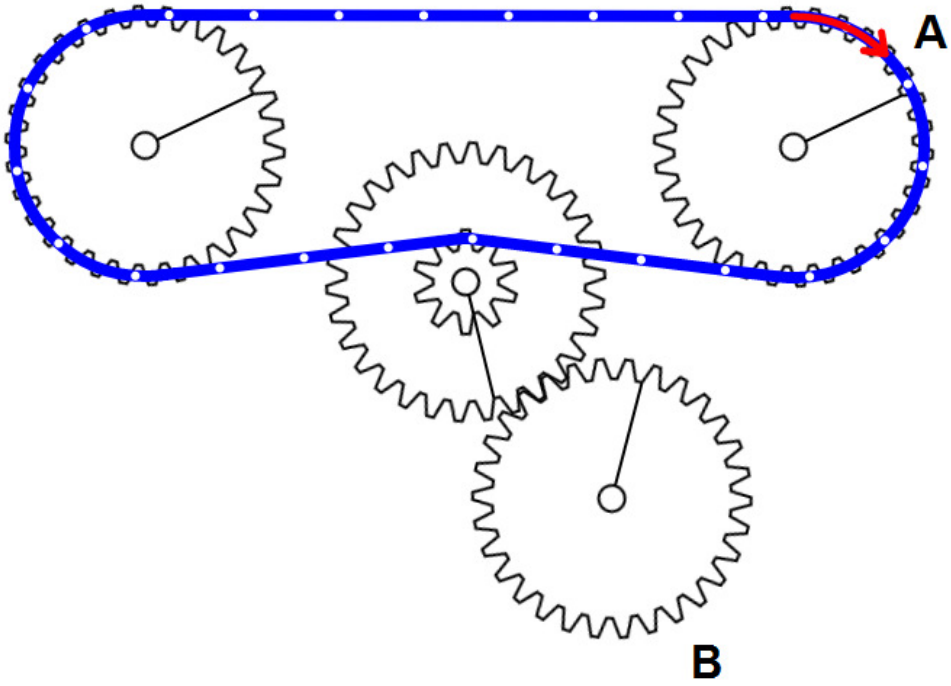
5.1 In which direction does gear B turn?

a. To the left (counterclockwise).
b. To the right (clockwise).

5.2 How fast does gear B turn compared to gear A?

a. Gear B turns slower than gear A.
b. Gear B turns with the same speed as gear A.
c. Gear B turns faster than gear A.

Question 6



Gear A turns to the right (clockwise).

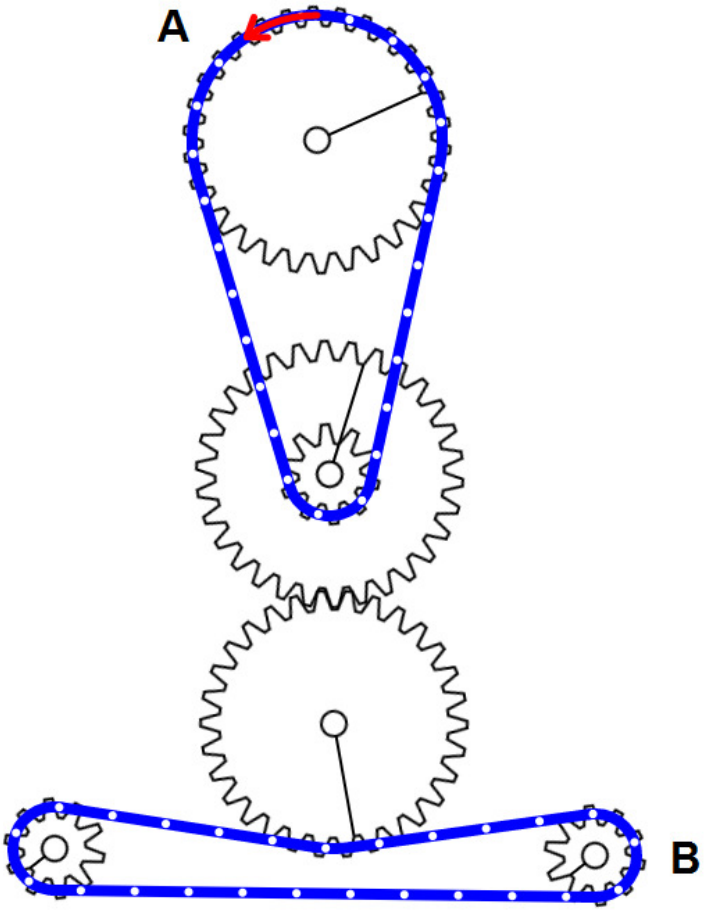
6.1 In which direction does gear B turn?

a. To the left (counterclockwise).
b. To the right (clockwise).

6.2 How fast does gear B turn compared to gear A?

a. Gear B turns slower than gear A.
b. Gear B turns with the same speed as gear A.
c. Gear B turns faster than gear A.

Question 7 (study 1)

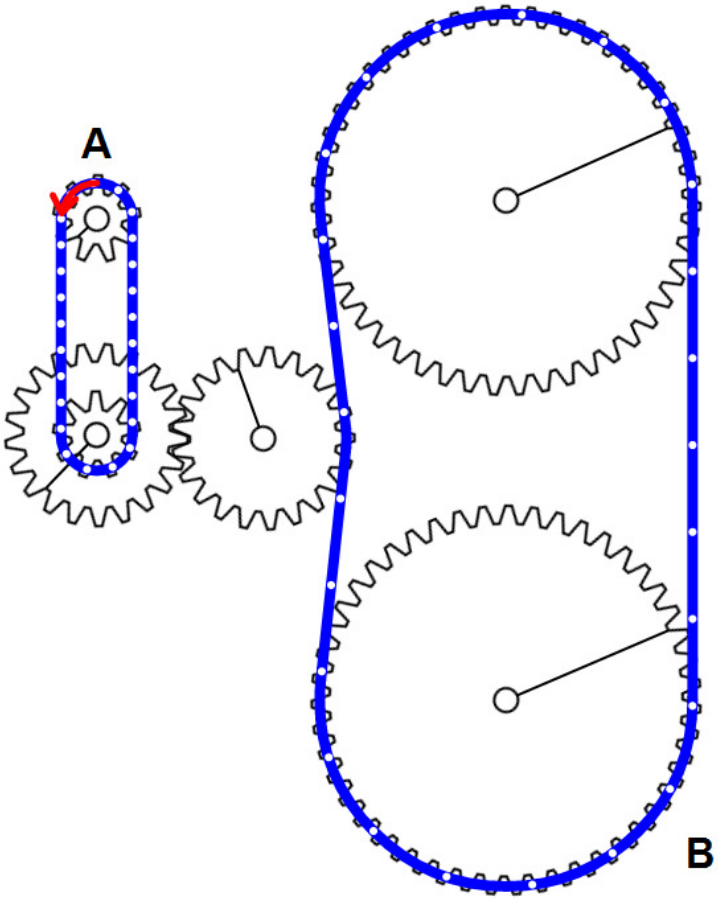


Gear A turns to the left (counterclockwise).

7.1 In which direction does gear B turn?
- a. To the left (counterclockwise).
- b. To the right (clockwise).

7.2 How fast does gear B turn compared to gear A?
- a. Gear B turns slower than gear A.
- b. Gear B turns with the same speed as gear A.
- c. Gear B turns faster than gear A.

Question 7 (study 2)



Gear A turns to the left (counterclockwise).

7.1 In which direction does gear B turn?

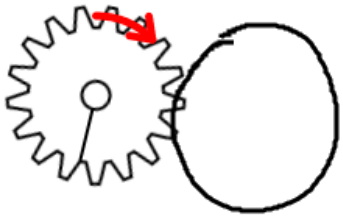
a. To the left (counterclockwise).
b. To the right (clockwise).

7.2 How fast does gear B turn compared to gear A?

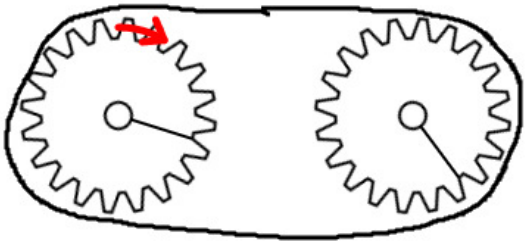
a. Gear B turns slower than gear A.
b. Gear B turns with the same speed as gear A.
c. Gear B turns faster than gear A.

Explanation questions 8 to 11

Questions 8 to 11 are drawing questions. This means that you answer these questions by drawing gears or chains yourself. You don't have to make your drawings very precise, because it takes too much time to draw all those small teeth. Draw a gear by drawing a circle, like this:

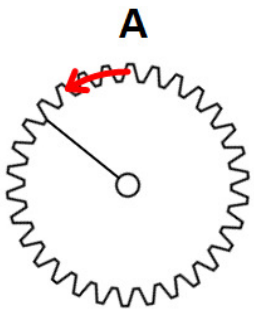


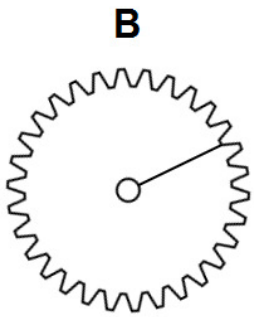
Draw a chain by drawing a stroke around gears. Like this:



Question 8

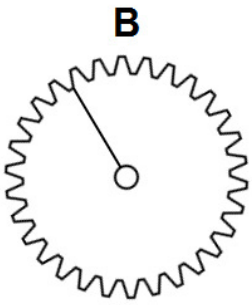
Gear A turns to the left (counterclockwise). Draw one or more **gears** so that gear B will turn to the right (clockwise).

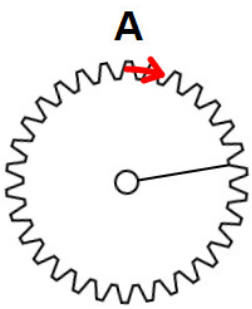




Question 9

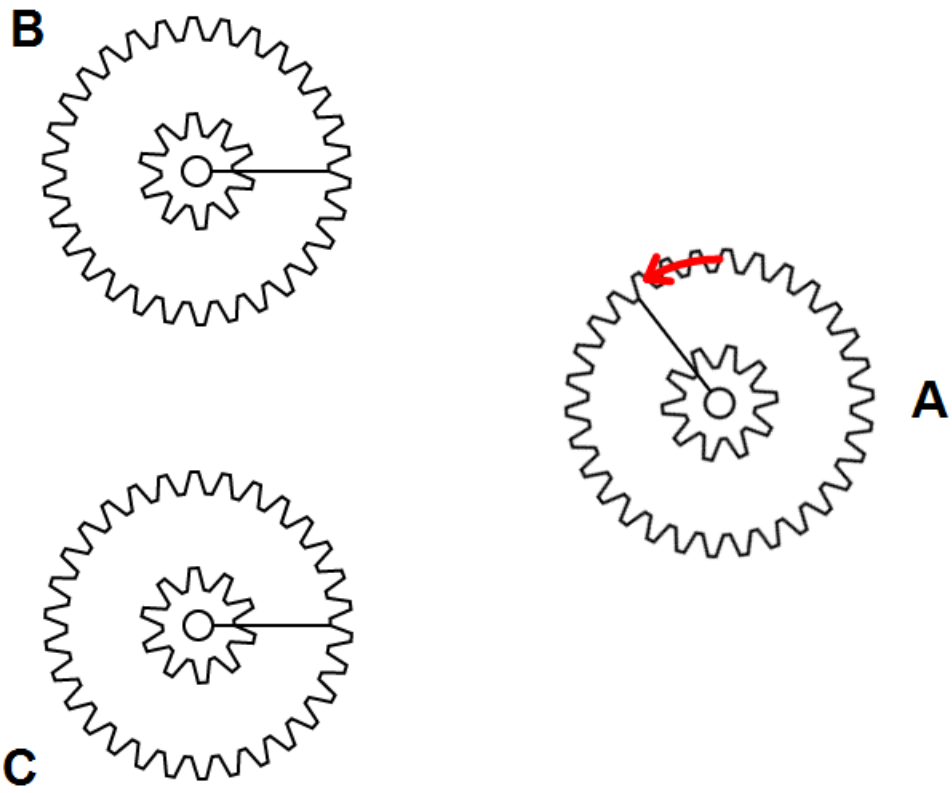
Gear A turns to the right (clockwise). Draw **one gear and one chain** so that gear B will **turn faster than** gear A.





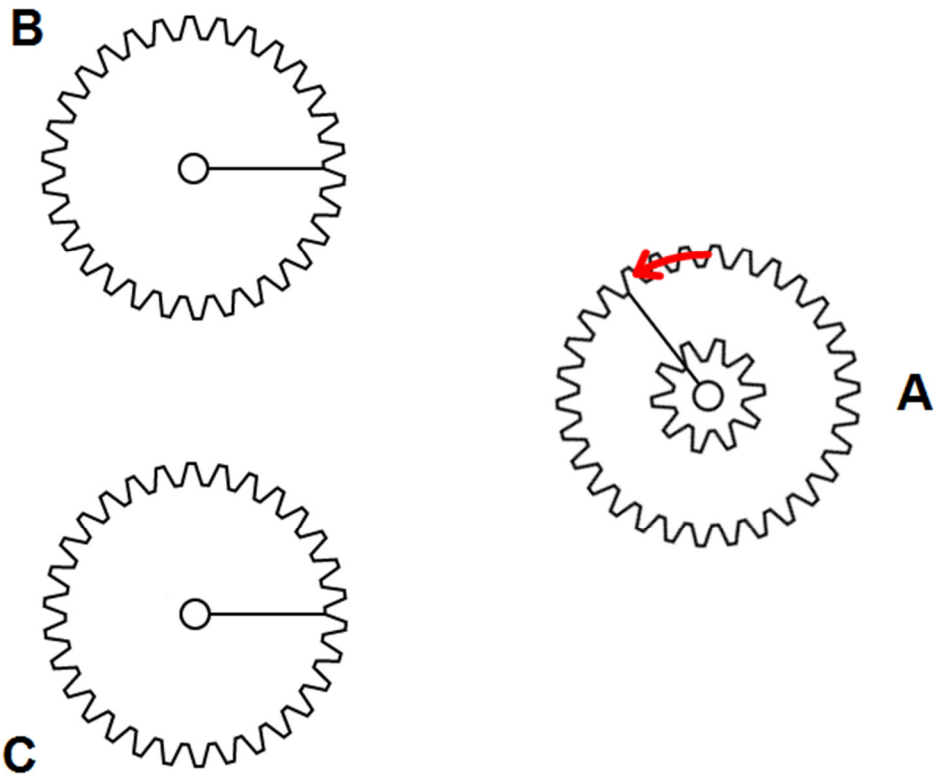
Question 10 (study 1)

Gear A turns to the left (counterclockwise). Draw one or more **chains** so that gear B and C also start to turn. Ensure that gear B will **turn faster than** gear C.



Question 10 (study 2)

Gear A turns to the left (counterclockwise). Draw one or more **chains** so that gear B and C also start to turn. Ensure that gear B will **turn faster than** gear C.



Question 11

Gear A turns to the right (clockwise). Draw one or more **gears** so that gear B will turn **to the left (counterclockwise)** and **faster** than gear A.