







Computational Thinking: Focus on Pattern Identification

Marielle Léonard^{1,2}(✉) , Yvan Peter¹ , Yann Secq¹ ,
and Cédric Fluckiger² 

¹ University of Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, Lille, France
{marielle.leonard,yvan.peter,yann.secq}@univ-lille.fr

² University of Lille, ULR 4354 - CIREL - Centre Interuniversitaire de Recherche en
Éducation de Lille, 59000 Lille, France
cedric.fluckiger@univ-lille.fr

Abstract. This article focuses on pattern identification in the context of pupils aged 9 to 15 who are learning programming at school. In this context, programming puzzles that involve moving a robot on a 2D grid using a block-based programming language is common. We consider the ability to identify and formally characterize recurring structures within data or processes, to be a fundamental skill of computational thinking. In this article, we study the case where the *motif* (i.e. repeating unit) can be identified visually from the grid (obstacles, target...) for tasks involving the use of a loop. We ask what makes *motif* identification, and thus problem solving, difficult in this context. We provide a quantitative analysis based on the success rates of a hundred tasks from an online programming contest (200,000 participants). We have identified relevant features of the *visual motif*, which led us to specify five categories according to the degree of correspondence between the *visual motif* (2D grid) and the *algorithmic motif* (corresponding loop based program).

Keywords: Computational thinking · Pattern · Pattern identification · Loop · Computer science education · Quantitative analysis · Large-scale study

1 Introduction

Computer Science (CS) education has recently been reintroduced into school curricula in many countries. In France, CS content has been included in compulsory school curricula since 2016. For students aged 9 to 12, programming is part of the mathematics curriculum¹. The prescribed task is to control a robot or a character on a screen using a block-based programming language. For students aged 12 to 15, they are expected to be able to “Write, develop and execute a simple program.”². But what does “a simple program” mean?

¹ Cycle 3 curriculum in effect in 2020, mathematics, space and geometry section.

² Cycle 4 curriculum in effect in 2020, mathematics, theme E - algorithmic and programming.

For this age group, using loops is one of the objectives of the school curriculum, along with sequences of instructions, conditional instructions and variables. At first, it could be considered that a program including a single loop, without nesting, conditional statements or explicit variables, is a simple program for students to write.

In previous studies [11], we set up pedagogical scenarios to explore how primary school students deal with programming tasks whose solution focuses on the use of a loop. The results from these case studies led us to consider the identification of patterns, redundancies, as essential to deal with this type of task. Especially, a recurring difficulty has been identified: the transition from one to several instructions inside the loop.

In this article, we want to improve our analysis of pattern identification when solving loop-focused programming tasks. Our two research questions are:

1. **RQ1** What does pattern recognition consist in, in the context of visual programming puzzles resolution?
2. **RQ2** What are the parameters that make pattern recognition difficult when getting started with solving loop-focused tasks?

To answer these questions, we mobilized elements of the theory of conceptual fields by G. Vergnaud [16] to conduct an *a priori* analysis. This allows us to distinguish several elements involved in pattern recognition and to identify parameters that can explain the difficulty of the problems. Then, we carried out a large-scale statistical analysis based on the success rates of 101 loop-focused programming tasks from the 2018 to 2021 editions of the Algorea french programming contest, which is organized by the France-ioi association. This statistical analysis validates the relevance of the identified parameters.

In the next section, we introduce the context of this research: the concept of *motif* and our analysis framework based on *classes of situations*. We then present the analysis of the programming tasks as well as the experimental setting before presenting the statistical analysis of the results for these tasks. We conclude by suggesting perspectives to go further in our understanding of the process of learning the basics of programming.

2 Theoretical Framework

2.1 From Pattern to Motif

In computer science, the word *pattern* is used in works about *design patterns* in the field of software engineering [4]. It is also associated with a specific skill in the scope of computational thinking, for which we can find various expressions: “looking for patterns” [21], “pattern recognition” [6], “identifying and making use of patterns” [3].

Some works more specifically mention the notion of loop, the focus of this paper. Gouws et al. [5] have defined a framework for describing computational thinking skills based on a literature review. This framework contains a category

called “Patterns and Algorithms”, in which the notion of loop is taken as an example. Rich et al. [12] define learning trajectories, including goals and examples of associated activities, one of which deals with iterative structures. The authors mention the importance of the perception of redundancy because it is intimately linked to the initiation to the notion of loop. Unfortunately, they do not provide any analysis of pattern identification activities.

On the other hand, in mathematics education, some works address this question of pattern identification. For Collins & Laski [2], a pattern is a sequence with a replicable regularity, which can vary along one or more dimensions. Liljedahl [7] proposes to distinguish two categories of patterns: *repeating patterns* and *number/growing patterns* (Fig. 1). The first corresponds to a cyclic structure generated by the repetition of a discernible unit. This definition is used in several works [9, 18, 20]. The second corresponds to a pattern parameterized by one or more pieces of information.

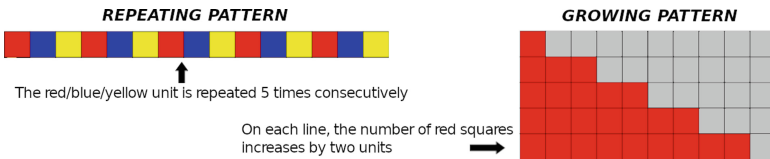


Fig. 1. Two categories of pattern in early mathematical education

For the previous authors, a pattern denotes the whole sequence, while our focus is on the repeating unit. In our work, we choose to use the word *motif* for the unit of repetition. With this meaning, the term *motif* is usually used in the artistic or literary field: “an idea that is used many times in a piece of writing or music”³, “a design which is used as a decoration or as part of an artistic pattern”⁴. Drawing inspiration from the previous definitions, we define a *motif* in our context as **an entity that can be identified within a set, because it is repeated identically or with predictable variations.**

Liljedahl [7] lists different tasks related to the concept of pattern: copying a pattern, continuing a pattern, finding missing elements in a pattern, transferring a pattern from one representation to another, identifying the unit of repetition, i.e., identifying the *motif*. Based on experiments conducted with young children aged 3 to 6, Warren et al. [18] designed a pedagogical sequence and establish a progression in the difficulty of these tasks [19, 20]. In this progression, the identification of the *motif* is the most difficult task and it is the one that reveals the understanding of the structure of the pattern [19]. Indeed, the term-to-term matching strategy, which consists in processing the elements of the pattern one by one without considering it as a whole, is systematically defeated during the activity of *motif* identification [2].

³ Cambridge Dictionary.

⁴ Collins Dictionary.

In our context, we are interested in the activity of *motif* identification in the field of computer science education. More specifically, we study *motif* identification when pupils deal with loop-focused programming tasks. We consider that the distinction proposed by Liljedahl [7] is a beginning of characterization of the forms of complexity of pattern abstraction, in particular the transition from directly observable (visual) patterns to unobservable patterns (changes of state of the environment, even similarity processes in the context of design patterns). We propose to specify what *motif* identification is in our context (**RQ1**) and to study in more detail, the characteristics of the *motifs* to be identified and their relation with the difficulty of the task (**RQ2**).

2.2 Classes of Situations

We aim to characterize and categorize the motifs to be identified when solving loop-focused programming tasks. For this purpose, we rely on the concept of class of situation developed by Vergnaud within the theory of conceptual fields [16]. Vergnaud takes a constructivist and cognitivist approach to learning. He aims to understand conceptualization, especially in the case of complex cognitive tasks, of which computer programming is a part. The unit of analysis is the subject/situation couple, where situation is used in the sense of a task. Vergnaud's hypothesis is that any finalized action is based on a *conceptualization-in-act*, that is to say that the actions of the subject reflect a cognitive activity that remains most often implicit, including for the subject itself. In computer science education, the conceptual field theory was used by Rogalski [13,14] to study *computer literacy* in high school and more recently by Spach [15] to analyze educational robotics situations. In our context, we place ourselves in this theoretical framework to study situations where the goal of the subject is to design a computer program that solves a loop-focused task.

Vergnaud invites us to analyze the situations the subject is confronted with, by grouping them into classes. This categorization can be considered from the point of view of the expert, by an analysis of the characteristics of the situations, and from the point of view of the subject, by studying the way in which he deals with the situations. The expert relies on the identification of situation variables aimed at differentiating close situations. The change in value of a situation variable may affect the structure of the subject's processing of the situation. This makes it possible to define two distinct classes of situations.

Vergnaud also insists on the progressiveness of the conceptualization, which should be considered over a long period of time. In a study on additive structures in the mathematic field, Vergnaud & Durand [17] asked 28 pupils in each level from grades 1 to 5, to solve additive tasks whose answer is strictly the same numerically, but for which the formulation of the task induces a different reasoning. They thus identified classes of situations which correspond to levels of difficulty in the resolution of these additive tasks. Their results show an effect of the age on students' ability to solve these tasks.

In this article, we propose to refine the definition of the concept of *motif* in relation to RQ1 and to characterize difficulties related to the motif identification

activity when dealing with loop-focused type situations in a programming context (RQ2). We rely on the works carried out around the concept of pattern and we mobilize the concept of class of situation to categorize loop-focused tasks. We are also inspired by the study by Vergnaud & Durand [17] which we have transposed into our context. The following section details the methodology and the experimental framework that we used to carry out this study.

3 Methodology and Experimental Setting

Our work is based on the analysis of the characteristics and results from a selection of 101 different loop-focused tasks that come from the 2018 to 2021 editions of the Algorea french programming contest. They consist in programming puzzles [10] involving a virtual robot on a grid, using the Scratch language. Their common point is that the sequence of actions to be performed by the virtual robot includes redundancy, which must be identified to solve the problem. The reference solution therefore involves a loop or several loops in sequence, but no nested loops. For the study of these programming situations, we considered the two points of view indicated by Vergnaud. First, we carried out an analysis of the programming tasks from the point of view of the expert, also called *a priori* analysis, which led us to identify the parameters that have a potential impact on the difficulty. Then we analyzed the activity of the subjects confronted with these situations during their participation in the Algorea competition, through the success rates noted for these problems.

3.1 *A Priori* Analysis: *Visual Motif* and *Algorithmic Motif*

When dealing with a loop-focused problem involving programming a virtual robot on a grid, one has to consider two kinds of *motifs*. The first one is a **visual motif**, which is observable on the grid. It consists of adjacent cells, which may contain visually salient elements (marked cell, or containing an object). This can be related to the concept of data which is one of the core concepts of computer science [1].

The second kind of *motif* is the **algorithmic motif**, which is related to the concepts of algorithm and machine, two other core concepts of computer science [1]. The *algorithmic motif* consists of actions to be executed one after the other by the machine, actions which are induced both by the pattern identified in the data and by the specificities of the machine. A series of actions in a fixed chronological order constitutes this *algorithmic motif* and in our context it is induced by the *visual motif* but it is also dependant on the possible actions (i.e., robot language, orientation system). The *algorithmic motif* is only observable during the actual execution of the actions. For instance, in the relative orientation system, the rotational actions of the robot are not matched with any element of the grid. In a program designed in the Scratch language, the *algorithmic motif* corresponds to the sequence of blocks inside the repeat block.

Solving a loop-focused programming problem in our context therefore requires identifying the *visual motif* on the grid, matching this *visual motif* with the actions to be performed by the virtual robot on this same grid, and expressing this *algorithmic motif* with the Scratch programming language.

For each of these *motifs*, *visual* then *algorithmic*, we identify several parameters or characteristics, which correspond to variables of situation in the sense of Vergnaud [16]. For the *visual motif*, we consider the number of cells it occupies on the grid, the presence of visually salient elements within the *visual motif* and the presence of decorative elements on the grid. For the *algorithmic motif*, we retain the number of actions constituting the *motif* and the presence of actions that are not part of the pattern (corresponding to instructions outside of the loop). As a variable of situation, we also study the degree of correspondence between the *visual motif* and the *algorithmic motif*. These are the parameters that will drive our analysis of the difficulty of the programming problems.

3.2 Experimental Setting

The virtual robot programming situations that we study come from the Algorea online contest, whose programming environment is shown in Fig. 2.

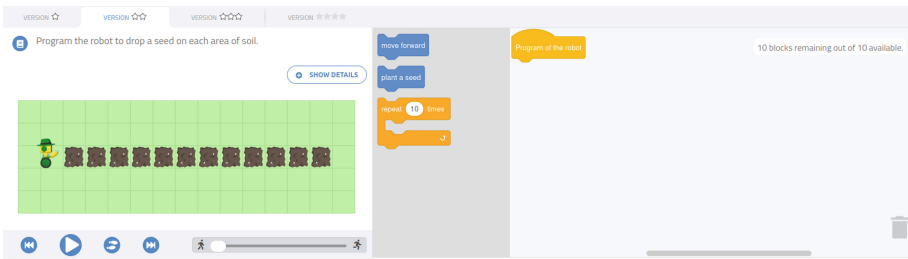


Fig. 2. Algorea contest programming environment (Situation 1, where the *visual motif* covers a single cell)

This environment is suitable for our study on *motif* identification. On the one hand, for loop-focused tasks, the repeat block is the only available control structure block. The subject quickly infers that he is in the situation where he needs to use this repeat block. On the other hand, the total number of blocks that can constitute a program is limited, which forces pupils to make use of this repeat block. However, the number of trials is not limited, which allows a trial and error strategy.

In total, the Algorea competition involves more than 200,000 participants each year, from grade 4 to grade 12 (9 to 18 years old). In the context of this study, we are only interested in the individual results of pupils from grades 4 to 9 (9 to 15 years old), who selected the Scratch language. This represents between 6,000 and 75,000 participants depending on the round of the contest. Studied

participants are distributed over the 6 class levels, with an over-representation of middle school students. Thus, we do not control the size of the sample studied, which varies depending on the round but remains substantial. In addition, the competition takes place in school or at home, so in real life conditions. However, we consider that the large sample size compensates for the variations of the participation context.

4 Analysis and Results

For each situation, we collect the success rate by class level, i.e. the quotient of the number of participants who succeeded in the task over the number of participants who opened the task. As a preliminary to the study on the identification of *motifs*, we proceeded to some analysis of a more general nature. On the one hand, we checked the robustness of our data concerning success rates. When considering all class levels together, a chi-square test of independence allows us to verify that all differences in success rates between two situations are statistically significant with a p-value less than 0.01. For a particular class level, a success rate difference of 5% units between two situations is significant for the middle school levels (p-value < 0.02). Only a few situations for the elementary level, whose numbers are smaller, lead to differences in success rates of 5% units that are less statistically robust.

On the other hand, Fig. 3 confirms that, as expected, the success rate decreases when the number of instructions in the reference solution increases. However, we notice a significant dispersion of values on the vertical axis, sometimes by more than 50% units, which tells us that other situational variables influence the success rate. The identification and study of these variables are the subject of the following sections. To this end, for each characteristic identified in the Sect. 3.1, we calculate the median of the success rates and the interquartile range, as indicators of the distribution of the data. At first, we focus on the *visual motif* which allows a first categorization of the tasks. We then complete and refine the analysis by also considering the *algorithmic pattern*.

4.1 Visual Motif

In this section, it is the visual aspect of the pattern that matters, regardless of the actions that the virtual robot has to perform.

Concerning the number of cells over which the *visual motif* extends, we can very clearly distinguish two classes of situations (Fig. 4). For a first class of situation, the *visual motif* consists of a single cell of the grid (example Fig. 2). The success rate of these tasks is high as early as elementary school. The interquartile range is low, which means that this characteristic is significant in explaining the success rate. On the other hand, the interquartile range is much higher if the *visual motif* extends over several cells (examples Fig. 6). In this case, other variables contribute significantly to the value of the success rate.

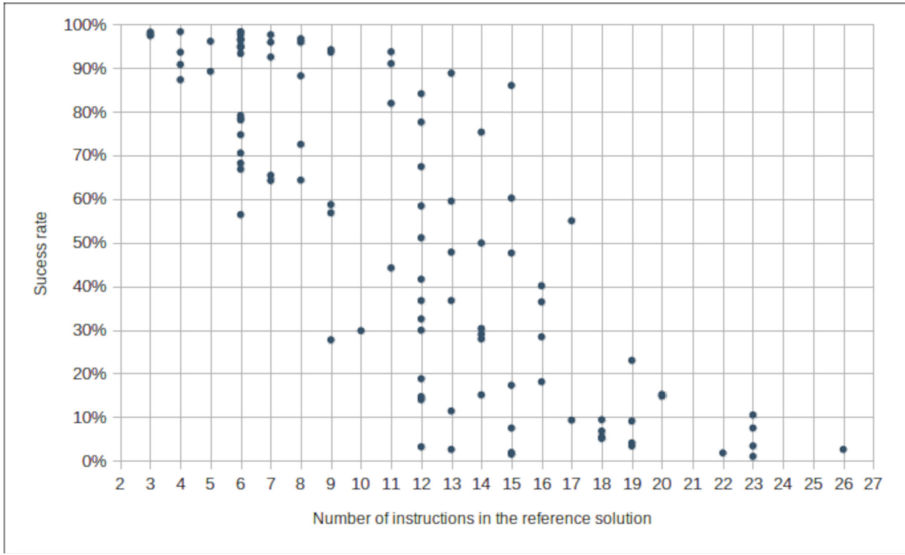


Fig. 3. Success rate scatter plot depending on the number of instructions in the reference solution (linear correlation rate -0.81 ; p -value < 0.05 on the Bravais-Pearson test)

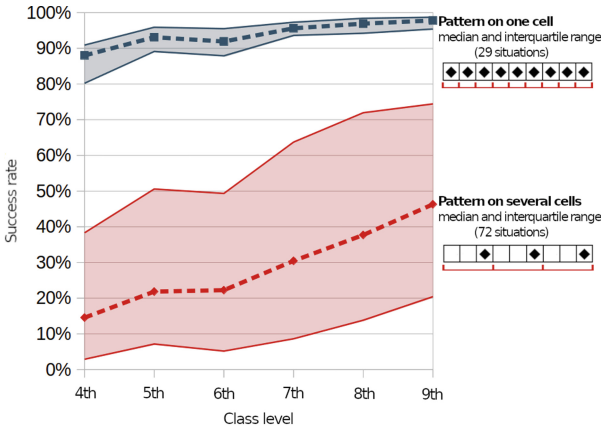


Fig. 4. Two classes of situations: situations where the *motif* is limited to a single cell, and situations for which the *motif* extends over several cells

For 70 situations for which the *visual motif* extends over several cells, we study the adjacent cells which are not part of the same *motif*.

When adjacent identical cells with a visually salient element do not belong to the same *motif* (examples Fig. 6: situations 3 and 4), the success rate is low (Fig. 5: red curve), and this is more pronounced with younger students. However,

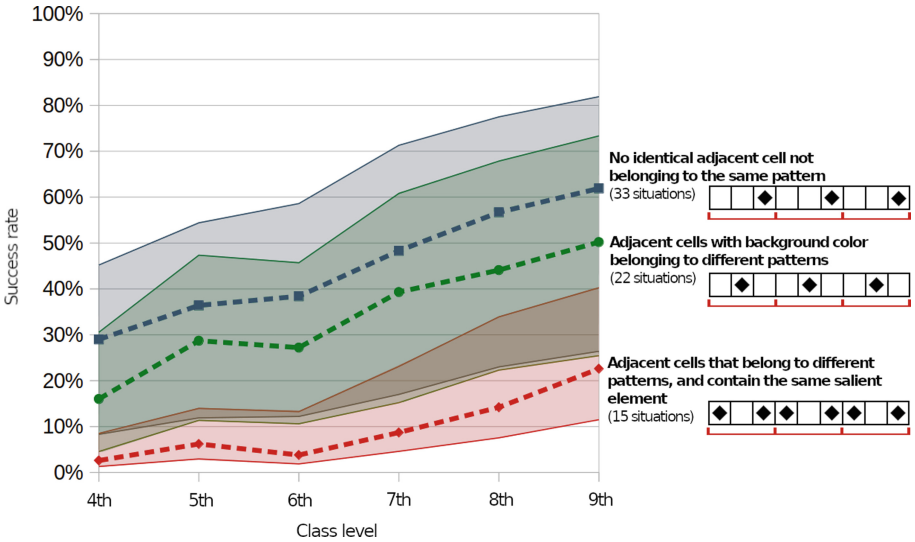


Fig. 5. Study of adjacent cells not belonging to the same *visual motif*

when two adjacent empty cells belong to different *motifs*, the success rate is close to that of situations without identical adjacent cells belonging to different *motifs*. This result leads us to think that the salient elements are taken as privileged reference points during the identification of the *visual motif*. Identical salient elements on adjacent cells are perceived as part of the same visual entity. When they do not belong to the same *motif*, this makes the *motif* less visible and therefore more difficult to identify.

We show in the same way, the effect of the presence of decorative elements on the grid, that is, visual elements that are not on the expected path of the robot, but make the cells look different from regular empty cells, or may be forbidden cells for the robot. For lack of space, we only give for each modality, the value of the median (Q2) and the interquartile range (IQR) for all the class levels taken together, with the unit being the percentage point of the success rate. Depending on how they are arranged, the decorative elements are more of a help or a source of difficulty. When they completely constrain the path of the robot (Q2: 60.0, IQR: 31.3), they constitute an aid compared to situations without decorative elements (Q2: 51.1, IQR: 58.0). If not, they seem to act as distractors and are a source of difficulty (Q2: 27.7, IQR: 49.0). This difficulty becomes massive when these decorative elements make some *motifs* visually different (Q2: 3.2, IQR: 3.3). Thus the study of the characteristics of the *visual motifs* shows that the nature of the elements present on the grid has an effect on the complexity of the situation. The easier the *motif* is to visually isolate, the more likely the situation is resolved by pupils. Conversely, factors that disrupt the visibility of the *motif* negatively impact the success rate of the situation.

4.2 Matches Between *Visual Motif* and *Algorithmic Motif*

Once the *visual motif* on the grid has been identified, it is necessary to deduce the matching *algorithmic motif*. We distinguish 5 classes of situations concerning the correspondence between *visual motif* and *algorithmic motif*. For the first three classes, all the *visual motifs* are the same, which is no longer true for the last two classes.

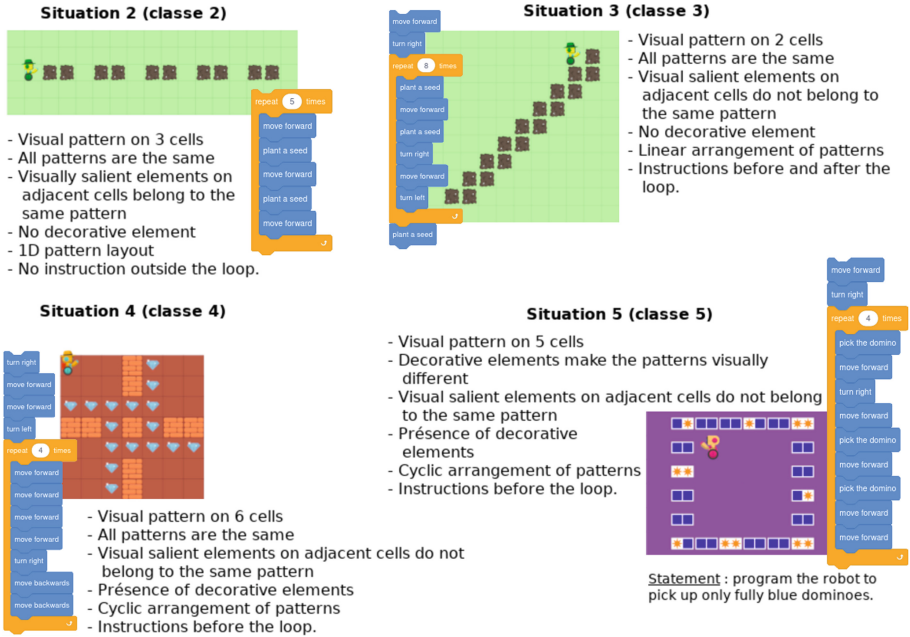


Fig. 6. Prototypical example of a situation for each class defined for the correspondence between *visual motif* and *algorithmic motif*

A first class of situation, very distinct, and which we have already identified in the previous section, concerns situations where the *motif* is limited to a single cell (example in Fig. 2). The other classes are represented in Fig. 6. We put in a second class, situations where the *motif* extends over multiple cells, and for which we have a strict correspondence between *visual motif* and *algorithmic motif*. Each movement action of the robot is identifiable by the boundary between two cells, while the other actions are identifiable by a visually salient element. These are the situations where the movement of the robot is only possible in one direction, and the situations for which the orientation of the robot is absolute (north, south, east, west). A third class corresponds to situations where several states of the robot on the same cell are visually identical, making the correspondence between visual pattern and algorithmic only partial. These are the situations where the robot has a relative orientation, and for which the pivoting

actions of the robot are not observable before the execution of the program. To solve these situations, it is necessary to mentally simulate the pivoting actions of the robot, by representing them on the appropriate cells and by keeping the orientation of the robot in memory. The fourth class concerns situations in relative orientation for which the arrangement of *motifs* is cyclical. The *visual motifs* are identical but each rotated by a quarter turn compared to the previous one. Finally, for the fifth class, the correspondence between *visual motif* and *algorithmic motif* is hindered, and it is necessary to disregard certain visual elements. That is, either visual salient elements or decorative elements are equivalent but visually different, or several *visual motifs* are partially superimposed, disturbing the visibility of each of them.

The 5 classes of situations defined above correspond to a gradation in the difficulty of matching *visual motif* and *algorithmic motif* (Fig. 7). The situations of class 1, for which the correspondence between the two *motifs* is attached to the cell, are solved well by most pupils from elementary school. However, situations of class 5, which require much more abstraction skills, are still difficult for most middle school students. The interquartile zones of classes 1 and 5 do not overlap with those of the other classes of situations. We deduce that the degree of correspondence between *visual motif* and *algorithmic motif* strongly determines the difficulty of these situations. On the other hand, classes 2, 3 and 4 have partially overlapping interquartile areas, which means that other variables also impact the difficulty of these situations in a significant way. These are also classes of situations where we observe the strongest progression during the 6 class levels studied.

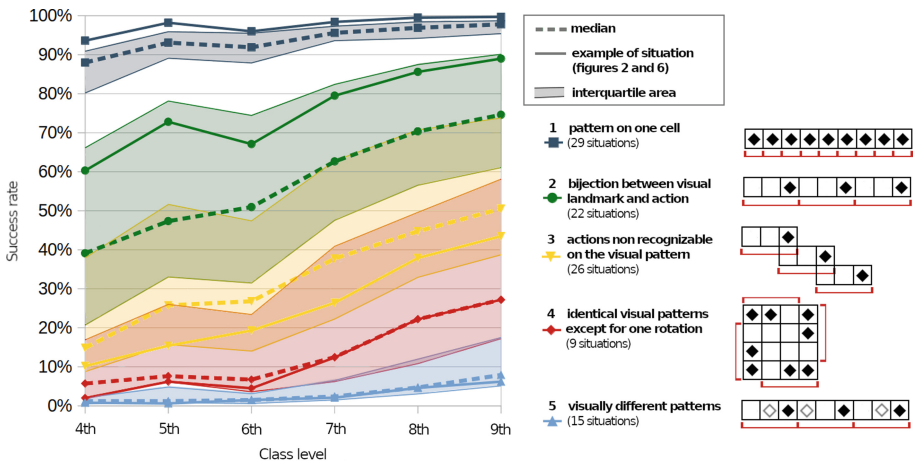


Fig. 7. Study of the correspondence between *visual motif* and *algorithmic motif*

Concerning the *algorithmic motif* expressed in Scratch language, we further show that the success rate is correlated with the number of instructions in the

loop (linear correlation rate of -0.79) and that the situation is significantly less successfully resolved when it is necessary to place instructions outside the loop, especially before the loop. We think that this last difficulty is linked to the identification of the position of the robot to be considered for the beginning of the pattern, i.e. the robot has to move to reach the beginning of the pattern. This position need to be mentally anticipated.

5 Conclusion and Perspectives

We show in this study that a loop-focused programming puzzle, even if the solution includes only one loop, is not necessarily a simple task. When solving this type of problem, pattern identification skill is essential, especially the identification of the repeat unit. We have specified the definition of a *motif* in this context. More precisely for programming puzzles that involve moving a virtual robot on a 2D grid, the identification of a *visual motif* and of the corresponding *algorithmic motif* are required (**RQ1**). Using a quantitative analysis of one hundred loop-focused tasks, we have characterized factors that make it difficult to identify the *visual motif* and we have established a gradation in the difficulties encountered, in particular for the matching of visual and algorithmic *motifs* (**RQ2**). Among the difficulties identified, we find the one, already identified in a previous study [8], related to the association of the programming situation with orientation in space.

Our contribution to knowledge concerns the understanding of what pattern identification covers in the situation of programming a virtual robot on a grid. This contribution makes it possible to better understand the obstacles encountered when starting learning computer science. The practical implication addresses teachers, by helping them to understand the difficulties of their students and to design relevant courses.

Further work is underway to continue this study. On the one hand, can we consider that a student has mastered the notion of loop when he has solved programming problems by trial and error, which is possible in this context? On the other hand, we know that motif identification is not the only issue in the treatment of loop-focused situations. Once the *motif* has been identified, the *motifs* have to be counted, which can lead to other difficulties that remain to be analyzed. To refine our understanding, we need more precise data. This is why we have set up a collection of activity traces at several scales. Apart from the success rates collected at the national level analyzed in this article, we have traces of activities at the class level and video recordings of contest participation at the individual level. Class-wide activity traces should allow us to distinguish between expert solving procedures and trial-and-error successes. As for the analysis of the video recordings, we seek to identify indicators that reflect the reasoning, the conceptualization-in-action [16] of the participant (expert procedure, errors). The objective will then be to match these indicators with the traces of activity in order to scale up, i.e. to make the link between the three collection scales.

Acknowledgements. This work is supported by the Digital Transition for Teaching Interreg project (<https://teachtransition.eu>). We would also like to thank the France-IOI association for providing the success rates for the Algorea contest.

References

1. Berry, G.: *L'Hyperpuissance de l'informatique: algorithmes, données, machines, réseaux*. Odile Jacob (2017)
2. Collins, M.A., Laski, E.V.: Preschoolers' strategies for solving visual pattern tasks. *Early Child. Res. Q.* **32**, 204–214 (2015)
3. Csizmadia, A., et al.: *Computational thinking-a guide for teachers* (2015)
4. Gamma, E., Helm, R., Johnson, R., Johnson, R.E., Vlissides, J.: *Design Patterns: Elements of Reusable Object-oriented Software*. Pearson Deutschland GmbH (1995)
5. Gouws, L.A., Bradshaw, K., Wentworth, P.: Computational thinking in educational activities: an evaluation of the educational game light-bot. In: *Proceedings of the 18th ACM conference on Innovation and technology in computer science education, ITiCSE 2013*, pp. 10–15. Association for Computing Machinery, Canterbury (2013). <https://doi.org/10.1145/2462476.2466518>
6. Hsu, T.C., Chang, S.C., Hung, Y.T.: How to learn and how to teach computational thinking: suggestions based on a review of the literature. *Comput. Educ.* **126**, 296–310 (2018). <https://doi.org/10.1016/j.compedu.2018.07.004>
7. Liljedahl, P.: Repeating pattern or number pattern: the distinction is blurred. *Focus Learn. Probl. Math.* **26**(3), 24–42 (2004)
8. Léonard, M., Peter, Y., Secq, Y.: Reconnaissance et synthèse de motifs redondants avec des élèves de 6–7 ans MOTIFS.MOTIFS.MOTIFS. $\leq 3 \times$ MOTIFS. $3 \times$ MOTIFS (2020). <https://hal.univ-lille.fr/hal-02971775>
9. Papic, M.: Promoting repeating patterns with young children - more than just alternating colours! *Aust. Primary Math. Classroom* **12**(3), 8–13 (2007)
10. Pelánek, R., Effenberger, T.: Design and analysis of microworlds and puzzles for block-based programming. *Comput. Sci. Educ.* 1–39 (2020)
11. Peter, Y., Secq, Y., Léonard, M.: Reconnaissance de motifs redondants et répétitions: introduction à la Pensée Informatique. *STICEF (Sci. Technol. l'Inf. Commun. Pour l'Éduc. Format.)* **27**(2) (2020)
12. Rich, K.M., Strickland, C., Binkowski, T.A., Moran, C., Franklin, D.: K-8 learning trajectories derived from research literature: sequence, repetition, conditionals. In: *Proceedings of the 2017 ACM Conference on International Computing Education Research, ICER 2017*, pp. 182–190. Association for Computing Machinery, Tacoma (2017). <https://doi.org/10.1145/3105726.3106166>
13. Rogalski, J.: Acquisition de savoirs et de savoir-faire en informatique. *Cahiers Didactique Math.* (43) (1987)
14. Rogalski, J., Vergnaud, G.: Didactique de l'informatique et acquisitions cognitives en programmation. *Psychol. Française* **32**(4) (1987)
15. Spach, M.: Activités robotiques à l'école primaire et apprentissage de concepts informatiques: quelle place du scénario pédagogique? Les limites du co-apprentissage. Ph.D. Thesis, Université Sorbonne Paris Cité (2017)
16. Vergnaud, G.: La théorie des champs conceptuels. In: *Recherches en didactique des mathématiques*, vol. 10/2.3. La Pensée Sauvage (1991)
17. Vergnaud, G., Durand, C.: Structures additives et complexité psychogénétique. *Rev. Française Pédag.* 28–43 (1976)

18. Warren, E., Cooper, T.: Using repeating patterns to explore functional thinking. *Aust. Prim. Math. Classr.* **11**(1), 9 (2006)
19. Warren, E., Miller, J.: Exploring four year old indigenous students' ability to pattern. *Int. Res. Early Child. Educ.* **1**(2), 42–56 (2010)
20. Warren, E., Miller, J., Cooper, T.: Repeating patterns: Strategies to assist young students to generalise the mathematical structure. *Australas. J. Early Child.* **37**(3), 111–120 (2012). <https://doi.org/10.1177/183693911203700315>
21. Wing, J.M.: Computational thinking. *Commun. ACM* **49**(3), 33–35 (2006). <https://doi.org/10.1145/1118178.1118215>