ABSTRACT
Algorithm Visualization (AV) can be analyzed and evaluated through three properties: the symbol system, the interactivity, and the didactic structure. While the relationship between various symbol systems has been researched and efforts to increase learner and AV interaction are being made, the importance and potential of the didactic structure of AV is yet to be explored. In view of this deficiency, this paper proposes Categories of Algorithm Learning Objective (CALO) as a pedagogical framework for designing and structuring AV. Based on seven non-hierarchical learning objectives commonly found in CS education, CALO provides a conceptual framework for a more learner-centered design, as well as a shared language for educators, learners and designers for communication and evaluation.

Categories and Subject Descriptors
K.3.2 [Computers and Education]: Computer & Information Science Education – Computer Science Education

General Terms
Algorithms

Keywords
Algorithm Visualization, Computer Science Education, Learning Objective, CALO, AV System

1. INTRODUCTION
Baecker's Sorting Out Sorting [2] has first sparked wide-spread interests in Algorithm Visualization (AV) [19]. Since then, a sequel [9] has been made, and numerous studies on the pedagogical effectiveness of using AV have been conducted. The results of many such studies, however, fail to agree with the initial enthusiasm educators have held. The quest to explain such discrepancies has yielded many valuable insights. To name just a few examples: Hundhausen et al. [11] conclude that how learners use AVs imposes significantly more impact than what the AV shows them on the computer screen. Rößling and Naps [20] identify nine pedagogical requirements for AV to be effective and, together with their working group, present the Engagement Taxonomy [16], later extended by Lauer [13]. Shaffer et al. [23] report that even pedagogically valuable AV tools can be used in ways that produce no pedagogical effect. Going through papers on AV published in the last couple of years, we observe a trend suggesting that the academic discussion on AV has been changing gears. The previous focus lay on whether AV has any pedagogical value for learners learning about algorithms. Now the focus seems to center on how educators and researchers can improve the pedagogical efficacy of AV and also congruently assess its pedagogical value.

Based on these findings, researchers have been incorporating more functions into AV in order to facilitate better and more learner engagement in addition to the passive viewing of animations offered in the early stages of AV development. Tests and evaluations have been designed and studied using (the revised) Bloom's Taxonomy (abbreviated as (R)BT in the following) [1] or the SOLO Taxonomy (SOLO) [3]. They intended to explain the difference of novice and advanced programmers in their cognitive capacities, and assess whether using AV makes any difference in the learning outcome. In our efforts to design and develop pedagogically beneficial features and evaluate and quantify the pedagogical effectiveness of AV systems, a few concerns have been documented in the process. These include as inadequate pedagogical considerations in support of implemented features [18], arbitrariness of the study design that leads to counter-intuitive results [22], and difficulties in objectively assessing the pedagogical value of AV [28].

As AV grows from mere animated sequences of yesteryear to today's interactive application of considerable complexity, a conceptual framework for designing pedagogically effective AV and congruently assessing its pedagogical value clearly assumes mounting importance.

2. THREE PROPERTIES OF AV
Taking a cue from the field of multimedia learning which deals with all aspects of educational technologies, we suggest that AV should be analyzed by its symbol system (composed of many subsystems such as texts, graphics, sounds and animations), its interactivity (functions that require user input and engagement), and its didactic structure (system design and organization based on pedagogical considerations).

The interface of AV and the information conveyed are coded in and represented by symbol systems such as texts, graphics and sounds, etc. Many theorists from the field of Multimedia Learning and researchers on Educational Technologies have contributed a
wealth of study results we can build upon by examining the interrelationship between these symbol systems, and how their presence, absence or juxtaposition enhances or hinders effective learning. To name only a couple of the better known findings here: there are various principles and findings presented by Clark and Mayer [5, 15], the Symbol Systems Theory by Salomon [21], the Dual Coding Theory by Paivio [17], and the Split Attention Effect by Chandler and Sweller [4]. These various findings on how the interplay and manipulation of texts, graphics, sounds and animations, apart from the content itself, contribute indirectly to the learning outcome by either advantageously alleviate learners' cognitive load or disadvantageously distract learners' attention, offer important insights to be considered by AV designers.

Interaction has long enjoyed a prominent status in multimedia design. It is no surprise that the significance of interactivity has also carried over to AV design and implementation, for studies confirm once and over again the relevance of a learner's active engagement to the learning outcome. Stasko's proposal [25] that users should construct their own visualization has notably seen enthusiastic adoption, and designers are also adding more interactive functions, such as pop-up quizzes. As AV designers try to figure out more ways to shake the user out of any complacent passivity, better design decisions may be possible when remembering that reaction on the part of the learner does not equal interaction, nor does participation [7]. The principles of active learning maintain that the more users directly manipulate and act upon the learning materials, the higher the mental efforts and psychological involvement, and therefore the better the learning outcome. We suggest the following forms of engaging users in their learning with AV systems for considerations when designing new features: writing, speaking, playing and teaching.

Perhaps the most important property of AV is also the most overlooked. The didactic structure substantiates the underlying rationale why we deploy AV to help learners learn algorithms in the first place. It provides an unambiguous frame of reference as to where we want learners to go with us and how we plan to get there. However, the didactic structure has been unintentionally left unexplored, because there are few, if any, specific and objective pedagogical frameworks that serve as AV design guidelines. The arrays of observable symbols are already defined by our cultural tradition and the built-in interactions based on our common experience are intuitively conceivable. However, the didactic structure and organization of AV needs ulterior deliberations, thus remaining veiled and eclipsed by its more extroverted siblings. To make up for the neglect in this regard, we introduce Categories of Algorithm Learning Objective, or CALO.

3. OUR PROPOSAL: CALO

Mager [14] defines a learning objective as a description of expected performance learners are able to exhibit after activities of learning; it specifies the intended result and not the process of instruction. Because a set of well-chosen learning objectives provide a sound basis for the design, implementation and evaluation of instruction [14], we believe it can be employed to make up for the lack of didactic structure in AV systems discussed earlier. The next logical question is: what are the learning objectives for using AV? AV has been adopted in the curriculum in order to help learners "understand" algorithms. “Understanding algorithms” is however too vague a specification to be a useful learning objective. We need to ask ourselves: what are the tasks or behaviors learners are expected to perform or exhibit to demonstrate that they have “understood” an algorithm? Based on studies of the BRACElet Project [29-33] and a collection of CS exam questions typical in the freshman or sophomore year, we have tentatively identified and classified the following seven Categories of Algorithm Learning Objective (CALO).

<table>
<thead>
<tr>
<th>CALO</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Descriptive</td>
<td>discerning and describing algorithms</td>
</tr>
<tr>
<td>Demonstrative</td>
<td>demonstrating algorithms with graphics or objects</td>
</tr>
<tr>
<td>Decoding</td>
<td>following and tracking algorithms</td>
</tr>
<tr>
<td>Coding</td>
<td>reproducing learned algorithms</td>
</tr>
<tr>
<td>Evaluative</td>
<td>analyzing, comparing and evaluating algorithms</td>
</tr>
<tr>
<td></td>
<td>that solve the same set of problems</td>
</tr>
<tr>
<td>Appropriative</td>
<td>writing a complete program evoking, extending</td>
</tr>
<tr>
<td></td>
<td>or modifying learned algorithms to solve a given problem</td>
</tr>
<tr>
<td>Originative</td>
<td>developing own algorithms to solve unfamiliar problems</td>
</tr>
</tbody>
</table>

Each of these categories in CALO is associated with a set of unlimited possible tasks of similar characteristics that learners should be capable of performing as a learning objective. We give some examples for each category in the following:

**Descriptive** - (1) Given the name of an algorithm, learners are capable of verbally describing the work flow. (2) Given a verbal description or graphical representation of how an algorithm works, learners are capable of identifying that algorithm. (3) Presented with two or more algorithms either by verbal description or graphical representation, learners are able to tell them apart and name them accurately.

**Demonstrative** - Given the name or description of an algorithm, learners are able to manually demonstrate in discrete steps how it works, using real objects, drawings, or computer programs.

**Decoding** - (1) Given a piece of pseudo or actual code, learners can identify the algorithm, state its purpose, comment each line of code, and name its output. (2) Given more than 3 lines of code in random order, learners know how to arrange them in correct order (also known as Parson's Problems [6]).

**Coding** - (1) Learners are capable of reproducing an algorithm in pseudo or actual code, with or without external aid, such as verbal description or computer animation. (2) Given a piece of incomplete code, e.g. missing lines or blanks to be filled, learners are able to fill in the missing parts.

**Evaluative** - Given various algorithms that solve a similar type of problems, learners are able to compare them and explain in terms of efficiency and/or accuracy which one may be more suitable for the given problem.

**Appropriative** - (1) Given a problem or task description, learners can decide on the algorithms needed and write a complete piece of program invoking learned algorithms, with or without reference materials. (2) Learners are able to extend or modify learned or known algorithms as needed and incorporate them in their written code.

**Originative** - When confronted with real-world or unfamiliar problems, i.e. outside of the academic settings, learners are able to design or come up with their own algorithms in response.
These given sample tasks are only representative and by no means exhaustive. Some tasks might also fall into more than one category: debugging, for example, involves both tracing skills (decoding) and coding skills (coding).

Viewed macroscopically, the descriptive category is primarily concerned with learners' verbal expression of their algorithmic understanding, whereas the demonstrative category emphasizes non-verbal and exact presentation of the internal workings of algorithms. Neither of these two categories involves any programming. While the decoding category deals with learners' passive ability to trace code, the coding category tackles learners' active coding skills, albeit in isolation and without global context. The last three categories require skills that go beyond isolated coding skills: the evaluative category requires knowledge of not only the mathematical workings of algorithms but also analysis of problems and solutions at hand. We see the appropriate category as problem solving skills still mostly confined in a classroom setting where learners solve carefully designed and controlled test questions. In the originative category, learners will either synthesize everything they have learned in the classroom and apply it to real-world problems, or they will temporarily disregard the confines of prior knowledge and invent something new.

Readers may be tempted to view CALO as a sequential and accumulative development of cognitive abilities that learners acquire linearly. However, we regard CALO as non-hierarchical, and each category as independent from each other. We have met learners who can proficiently script an algorithm on their own (coding) but find it difficult to understand other people's code (decoding). Similarly, it does not necessarily follow that just because someone is capable of using an algorithm to solve a given problem (appropriative), they will also be versed enough in other algorithms to compare and evaluate them (evaluative). It is also not hard to imagine persons who can come up with their own algorithms to solve real-world problems (originative) without knowing any algorithms by their names (descriptive). As one last example, the ability to track and understand an algorithm (decoding) does not guarantee the ability to demonstrate with props exactly how that algorithm works (demonstrative). There is little doubt that some of these skills might be acquired atearlier stages of learning than others, but before sufficient evidence has been gathered in future research, any assumptions at this point would rather defy the purpose of CALO. Given its current status as a proposal, we anticipate discussions and more field studies that will lead to the refinement of CALO.

When we use CALO to examine today's existing AV systems, most of them are conceptualized to support the Decoding and Descriptive objectives, with a few others offering features that support the Evaluative objective. While the focus of AV shifts towards the Demonstrative objective as we witness a plethora of AV systems showcase their functionality to support learner constructed AVs, we look forward to seeing the Coding, Appropriate and Originative objectives start enjoying more explicit support in AV.

CALO can also assist us in AV evaluation. If students have been learning algorithms with an AV system that exclusively supports the Decoding or Demonstrative objectives, and are then asked to perform Coding or Evaluative tasks in a test setting, we cannot be certain that we are congruently assessing the pedagogical effectiveness of that AV system, for the tester may have been tested on learning objectives that are not supported in the used AV system in the first place. This could explain in part why many studies fail to ascertain the pedagogical value of AV.

With CALO as a conceptual framework and didactic structure, designers can work with educators to design and build functionality that supports learners in achieving specific learning objectives. In specifying the intended learning outcomes of using AV, CALO helps researchers design test items that are more likely to congruently reflect on the pedagogical effectiveness of the AV system being accessed. Furthermore, when learners are acquainted with the intended learning outcomes expected of them, they will have a set of tools that help them take a more active role in learning, where they are able to self-evaluate their learning progress and select tools that best assist them to achieve their goals. Lastly, CALO also operates as a shared language for learners and instructors, with which learners can better communicate their needs or suggestions for AV.

4. CALO, (R)BT AND SOLO

How does CALO stack up against (The Revised) Bloom's Taxonomy and the SOLO Taxonomy, two educational objectives popular among many educators and theorists? Can they serve the purpose and fill the void? Do we really need yet another taxonomy?

Fuller et al. [10] have proposed a working group to develop a CS specific learning taxonomy due to the inadequacies of current existing taxonomies. Although both (R)BT and SOLO offer insightful explanations of learners' cognitive development, they are not CS specific, nor do they specify concrete tasks that can be performed and evaluated. SOLO has been used in CS classes to classify and assess learners' exam responses, but the consistency of SOLO ratings has only been moderate [6]. When evaluation criteria based on SOLO were introduced in the classroom to influence learners' judgment about what is expected of them, Thompson [26] observes a clear resistance from the learners.

The cumulative and hierarchical nature of (R)BT especially has attracted many criticisms. When using (R)BT to assess test items, ambiguities and inconsistencies exist in interpreting the correlation of learning objectives and learning outcomes. Naps et al. [16], while proposing a mapping of effectiveness studies to Bloom's Taxonomy, recognize the difficulties in doing so. Using introductory programming courses as a case study, Thompson et al. [27] find that consistent application of BT is difficult. Johnson and Fuller [12] document the disagreement between conveners and assessors on their assessment and understanding of (R)BT. Starr et al. [24] also observe similar phenomenon.

Based on these findings, we venture to conclude that the mapping of learners' exam responses to the (R)BT or SOLO taxonomy still poses an academic dispute among educators. Because (R)BT and SOLO are mainly concerned with the intrinsic cognitive progress in linear stages, educators have yet to specify clear and concise learning outcomes accordingly that can be communicated to learners. Learners may also have a hard time incorporating these taxonomies to evaluate their own learning.

It is in this regard that we have conceptualized CALO, which not only is CS specific, but also AV-centric. By specifying concrete tasks learners are expected to perform, researchers can look at these individual tasks and study what features and functionality implemented in AV may benefit learners in learning these tasks. By grouping the tasks homogeneous in characteristics into a category, we have devised a new taxonomy that will be useful for specification and discussions among all parties involved. By taking advantage of the taxonomy's non-hierarchical structure, educators are able to leave the chicken and egg question to researchers and focus on implementing a more flexible and encompassing AV system, as well as designing less ambiguous
and incongruent evaluations to assess the pedagogical values of the AV system.

5. CONCLUSION AND FUTURE WORK
The symbol system, interactivity and didactic structure constitute three properties of AV that researchers can examine and explore in order to develop and evaluate pedagogically effective AV systems. This paper proposes CALO as:

1) a didactic structure for AV designers as a pedagogical design framework to organize and implement new functions;
2) a set of learning objectives for researchers of AV to write test items that more congruently assess an AV system’s pedagogical effectiveness;
3) a self-evaluating tool for learners to check their own progress and performance against the expected learning outcomes, so that they can select suitable AV tools that best cater to their needs;
4) a shared language for learners, educators, researchers and designers to communicate.

Many research papers have stated that the purpose of AV is to help learners understand algorithms. One of the most apparent advantages of integrating CALO into AV design is to spell out what this rather generic expression understand algorithms entails. CALO classifies algorithm understanding into seven categories, effectively dividing a rather vague concept into concrete subdivisions that can be approached independently by AV designers and researchers.

At the risk of stating the obvious, we would like to point out that the very status of CALO as a proposal invites researchers to adapt these categories based on further researches and studies. One may also consider the following questions:

- What program functions can we build into AV to support learners’ descriptive, demonstrative, decoding, coding, evaluative, appropriative and originative skills respectively?
- Do any of these categories in fact form a causal relationship with one another?
- Using (R)BT or SOLO as a frame of reference, is it possible to find mappings, whether one-to-one, one-to-many, or many-to-one, between CALO, (R)BT or SOLO?

We hope that the introduction of CALO will bring about a more user-centered, pedagogically effective, and comprehensively assessable AV design.

6. ACKNOWLEDGMENTS
We are grateful for the funding provided by the German Research Foundation (DFG) in support of our research project.

7. REFERENCES


