Int. J. Engng Ed. Vol. 25, No. 4, pp. 701–711, 2009
Printed in Great Britain.

0949-149X/91 $3.00+0.00

Intelligent Visual Reasoning Tutor: an Intelligent Tutoring System for Visual Reasoning in Engineering & Architecture*

YONG SE KIM, ERIC WANG
Creative Design and Intelligent Tutoring Systems (CREDITS) Research Center, Sungkyunkwan University, 300 Chunchun, Jangan, Suwon 440-746 Korea. E-mail: yskim@skku.edu

Visual reasoning is an essential skill for many disciplines in engineering, architecture, and design. The underlying cognitive processes of visual reasoning form a basis in various problem-solving processes. We describe an intelligent tutoring system for visual reasoning that uses the missing view problem. This system, called Intelligent Visual Reasoning Tutor (IVRT), can adaptively support different learners' needs, track learners' progress, and provide active critiquing. IVRT uses a two-level reasoning architecture, combining geometric reasoning and semantic technologies, which enables the development of ITS for 3D geometry domains. We discuss IVRT's system architecture and implementation, which includes a learning contents model based on skills, lessons, and problems, and a learner model that measures domain competence as a set of skills. Learning contents and pedagogical teaching strategy rules are stored in standard OWL ontologies, which can be customized by the teacher.

Keywords: Visual reasoning; missing view problem; geometric reasoning; semantic technologies for learning; intelligent tutoring systems

VISUAL REASONING

THE ABILITY TO VISUALIZE and reason about geometric aspects of 3D objects is critical for success in engineering, architecture and design. A recent experimental study on design creativity using engineering, design, and social science students identified that visual reasoning capability is an underlying cognitive element of design creativity [1].

A traditional exercise for visual reasoning instruction is the missing view problem [2], in which two consistent, principal orthographic projections are given, as shown in Fig. 1. The learner's objective is to give a third orthographic projection such that these three views correspond to a valid 3-D solid object. The actual cognitive task involved is a 2D-to-3D visualization process, in which the learner mentally constructs a 3D solid that is consistent with both given views in 2D. This type of problem requires interactions of visual analysis, visual synthesis, and modeling, which builds the foundations of the visual reasoning processes.

Note that visual analysis, synthesis, and modeling respectively coincide with seeing, imagining, and drawing of the design ideation model of McKim [3]. In fact, the visual reasoning process forms a basis for the problem-solving process. When a person or a group, in design and in engineering, seeks a solution to a new problem, iterations of the following technical and cognitive processes are performed to achieve that aim. First, careful analysis is made of the nature of the problem and the problem-solving environment, including the entity's capability. Second, partial solutions are gradually synthesized. Using proper representations of these and the given problem, the solutions are reanalyzed considering the problem context, and an improved solution is synthesized. Thus, the visual reasoning process composed of iterations of visual analysis, synthesis and modeling serves the basic role in the problem-solving process. Also, visual reasoning tasks could be used in training such processes in a more structured way than typical problem-solving tasks.

The instruction of such visual reasoning skills has presented numerous challenges for traditional classroom methods. Most instruction methods rely on 2D representations, e.g. textbooks, slides, and learners' own drawings and writings, which do not

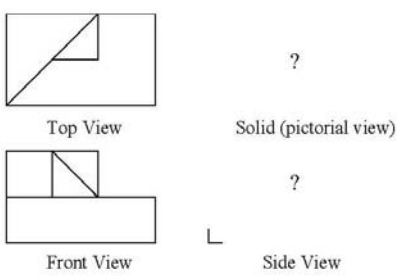


Fig. 1. Example of a missing view problem.

* Accepted 24 February 2009.

adequately prepare learners to visualize and manipulate 3D data. While all learners exhibit some ability to visualize simple shapes in 3D, this is quickly overtaken by the complexity of missing view problems, so the solutions are not so trivial that they can be visualized entirely in the learners' heads. Also, learners show a wide range of capabilities in visual reasoning, so that individualized support for learning is inevitably needed to reflect personal diversities. Hence, we see an ongoing need for intelligent instructional tools that can provide specialized assistance to learners in the development of these skills.

An intelligent tutoring system (ITS) is an instructional software system, typically composed of a domain model, learner model, and pedagogical knowledge [4], which is able to adapt its instruction based on the learner's current state. ITSs have been shown to be effective tools for instructional support. This paper describes an intelligent tutoring system called Intelligent Visual Reasoning Tutor (IVRT), which uses the missing view problem to develop visual reasoning skills. IVRT provides a learning system with which a learner can develop visualization and spatial reasoning capabilities in a self-paced series of exercises. IVRT can adaptively support different learners' needs, track learners' progress, and provide active critiquing.

IVRT uses a two-level architecture for reasoning in complex domains. Within the scope of a single problem instance, it uses 3D geometric reasoning for provable, algorithmic reasoning about 3D geometric entities and relations, which enables it to provide guidance and adaptive critiquing in visual reasoning domains. On a broader scale across multiple learners and problems, it uses knowledge-based reasoning for pedagogical decision-making, which is similar to other ITSs. By maintaining a deliberate separation of reasoning techniques, we can obtain results more reliably, more meaningfully, and more efficiently, for this problem domain, which provides better adaptive support to the learner. This two-level reasoning architecture has enabled us to extend the ITS framework into the domain of visual reasoning using 3D geometry, which has received almost no consideration in previous ITS research.

INTELLIGENT TUTORING SYSTEMS

ITS approaches and implementations

ITSs have proven difficult to implement primarily due to the demands of the domain model. Successful ITS implementations have generally corresponded to learning domains that exhibit certain characteristics.

- Model-tracing ITSs exist for programming [5], algebra [6][7], 2D geometry proofs [8], and physics [9], with widespread usage primarily at the high school level. This approach mostly requires that expert problem-solving knowledge for the domain is known, is fully represented in the domain knowledge, and is executable by a solver module. This implies that candidate solutions in these domains can be evaluated exactly via algorithms.
- Constraint-based ITSs have been developed for SQL [10] and database entity-relationship modeling [11][12] at the undergraduate level, and English punctuation [13]. Solutions in these domains have a formal structure of elements and relationships that must satisfy a set of constraints, which are essentially syntactic patterns of domain elements. This approach works well for domains in which most knowledge is at the syntactic level.
- Some ITSs use natural language processing techniques to assess brief essays [14], which can support domain knowledge at a higher semantic level.
- Relatively few ITSs have considered visual domains. An early ITS for medical diagnosis [15] considered a representation of 2D medical images, annotated with qualitative labels of the sizes of image features as clues for diagnosis. However, it avoided the issue of managing a general representation of 2D image data.

Semantic technologies

Recent research into semantic technologies has provided new capabilities for ITSs. Standard ontological representations such as Web Ontology Language (OWL) permit a more structured organization of knowledge, which supports classification-based reasoning using Description Logic (DL) reasoners. Rule-based reasoning has been enhanced by the emergence of rule language standards, including Semantic Web Rule Language (SWRL), and rule engines such as Jess. While previous ITSs have generally used precursors of these technologies in ad hoc schemes, recent ITSs are incorporating them directly, with expected benefits in knowledge sharing and reuse.

Ontologies and ontology-aware authoring tools have been developed to manage the authoring process of instructional systems [16][17]. Recently, a comprehensive ontology of learning, instruction, and instructional design has been devised that relates teaching strategies to educational learning theories, with its own authoring support tool [18][19].

SYSTEM ARCHITECTURE OF IVRT

Intelligent Visual Reasoning Tutor (IVRT) is an ITS for visual reasoning in a 3D geometric environment. It was originally designed in conjunction with an undergraduate-level course in engineering graphics, as an intelligent instructional tool to assist learners in solving missing view problems, and it has since been integrated successfully into that course.

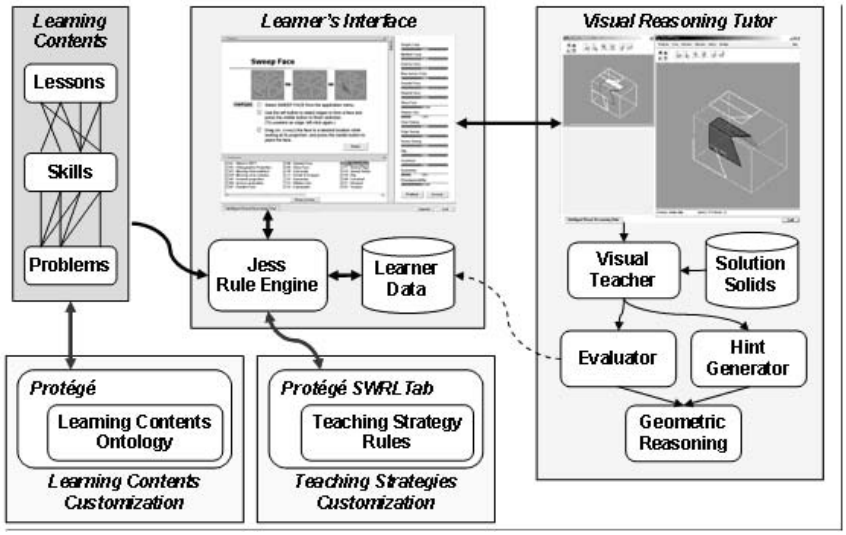


Fig. 2. System architecture of IVRT.

The system architecture of IVRT is shown in Fig. 2. The Visual Reasoning Tutor (VRT) module provides a 3D viewing and modeling environment, allowing interactive selection and sweeping operations of 2D faces in 3D space, with transparency graphics and sound effects to assist visualization. IVRT's premise is that the solution to a missing view problem is a 3D solid which may be constructed one face at a time, in a systematic and incremental manner, using 3D face-sweeping operations; and furthermore, that this approach is generally useful for all missing view problems, i.e. it represents a portion of human expert knowledge.

IVRT directly addresses the issue of explicit representation of 3D geometry. Its VRT module implements a partial boundary representation (B-rep) [20], which is a robust and mathematically correct representation of the learner's 3D solution state. On top of this, it implements geometric reasoning algorithms, which compute rigorous and exact results using 3D geometry. Through judicious use of geometric reasoning, IVRT provides intelligent critiquing and hint generation, without needing to define a traditional domain knowledge base of operations or constraints. Instead, it is the case that 3D geometry, although somewhat more complex in its details, is still a rigorous and well-researched domain, for which exact representations and algorithms are known. In essence, this enables us to encode much of the domain knowledge implicitly through geometric reasoning, without resorting to laborious enumeration of specific knowledge elements.

VISUAL REASONING TUTOR

We have previously developed an instructional software system called Visual Reasoning Tutor (VRT) [21][22], shown in Fig. 3. Two main components of the VRT system have been integrated into the current IVRT system: the Visual Sweeper module, which provides geometric and graphics operations for constructing solution solids from orthographic projections, and the Visual Teacher module, which provides adaptive evaluation and critiquing.

VRT is implemented in Visual C++ and OpenGL on Windows, with Hummingbird Exceed for X Windows emulation. When the learner selects a problem, IVRT invokes a stand-alone instance of VRT to manage that problem-solving session, which persists until the learner exits it.

Visual Sweeper

The Visual Sweeper module, shown in Fig. 3 on the right, implements a boundary representation that can represent a partially constructed solid in terms of its topology, including vertices, edges, and faces, and its geometry, including vertex coordinates and face normal vectors. By convention, the normal vectors of a solid's faces are defined to point toward the outside of the solid; hence, face orientations are significant.

Visual Sweeper provides interactive sweeping operations [23], which are the inverse operation

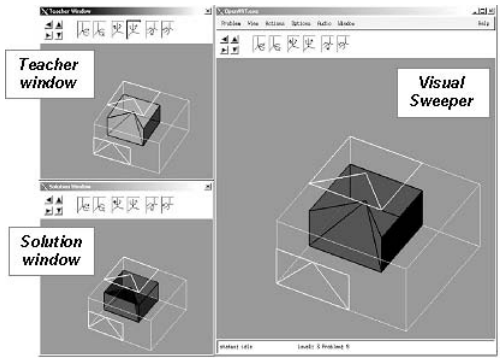
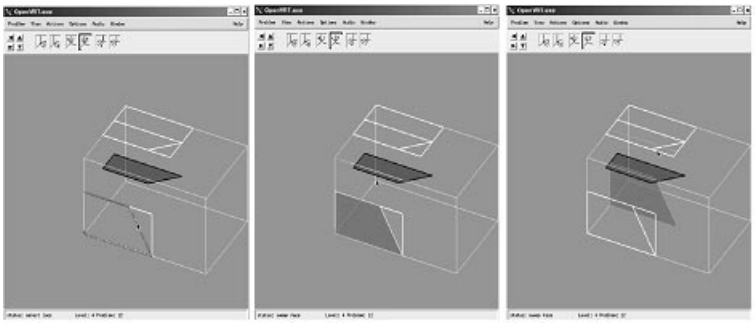


Fig. 3. Visual Reasoning Tutor.



(a) (b) (c)

Fig. 4. Face sweep operation in Visual Sweeper.

of orthographic projection as applied to each face. Sweeping operations are an intuitive, visually-oriented operation to construct 3D solids from orthographic projections. The learner selects a sequence of edges in one orthographic view to form a loop, as shown in Fig. 4(a). Visual Sweeper constructs a 2D face whose boundary is this loop, and highlights it in a transparent green color, shown in Fig. 4(b). This face is then swept interactively by moving the mouse. Throughout each sweeping operation, the projection of the swept face always remains consistent with its originating orthographic view, as shown in Fig. 4(c). By visually comparing the projection of the swept face to the other orthographic view, the learner interactively positions the swept face to satisfy both views. In this way, the learner incrementally constructs a solution solid through a sequence of sweeping operations.

Three kinds of sweeping operations are implemented: *face sweep*, in which all vertices and edges are swept; *edge sweep*, where one selected vertex remains fixed, and one selected edge is swept (which implies that the face "stretches" as necessary to remain consistent with the originating view), and *vertex sweep*, in which one selected edge remains fixed, and a selected vertex is swept. The edge and vertex sweep operations allow the construction of slanted faces that are not orthogonal to one or both orthographic views.

Three additional face-manipulation operations are implemented. The *flip* operation negates the normal vector of a selected face. The *construct* command creates a face from an existing loop of edges in the partially completed solid. This is needed to create faces that are projected as edges in both given views, since such faces cannot be created by sweeping from either given view. The *destruct* command deletes an existing face.

Visual Teacher

The Visual Teacher module captures and critiques the learner's current solution state. It provides the following services: evaluation of a partial solution solid, *hinting* for the next face to be manipulated, display of the nearest *solution solid* (a given problem could have multiple valid solutions), and calculation of the learner's *score*.

- The Teacher window, shown in the upper left of Fig. 3, provides on-demand evaluation of a learner's solution solid. It applies a geometric matching procedure, using geometric reasoning, between the faces of the learner's solid and the faces of each solution solid, using a weighted match function that considers faces' distances and relative angles [24]. This determines the closest solution solid, and the best matching between pairs of faces in the learner's solid and that solution solid. The Teacher window displays the evaluation using face colors: correct (green) for paired faces with perfect match scores, partially correct (yellow) for paired faces with non-perfect match scores, and incorrect (orange) for unpaired faces in the learner's solid only. A partially correct face is one that could be made correct with additional operations.
- The Teacher window can also display a visual hint for the *next face* to be created, by showing that face in blue color, as in Fig. 3, upper left. We have obtained missing view problem-solving knowledge from human domain experts for determining the best *next face* to manipulate, given the learner's current solid and the nearest solution solid. This problem-solving knowledge is implemented as rules in an embedded instance of CLIPS [24]. We arrange these rules into a deterministic procedure for missing view problem solving, as shown in Table 1. Hint generation proceeds by checking this procedure from top down until it identifies a face that is not yet classified as being correct. Subsequent studies have shown that some students abuse the hint mechanism, so we have added a password to restrict access to it.
- The Solution window, shown in the lower left of Fig. 3, displays the solution solid that is closest to the learner's current solid. For each problem, we pre-compute all possible solution solids off-line by applying an exhaustive geometric reasoning algorithm, and store the set of solutions as part of the problem data.

Table 1. Hint-generating rules for missing view problem solving

Selection rules	1. Create visible faces first (top and front faces). 2. Create hidden faces next (bottom and back faces). 3. Create faces using the *construct* command last (right faces, then left faces).
Sorting rules	4. Prefer the face that is adjacent to the most correct faces. 5. Prefer the face with most incident edges. 6. Prefer the face whose normal vector contains the most zero components.

The rationale behind rule 6 could be described as: *prefer the face with the simplest creation process*. A face whose 3D normal vector contains 2 zero components must be parallel to one of the orthographic views. Thus, it can be created using *face sweep* or *construct*, optionally followed by *flip*, which is considered to be the simplest process. If a face's normal vector has only one zero component, then it is a slanted face, which requires a *face sweep* + *edge sweep* (+ optionally *flip*). If a face's normal vector has no zero components, then it requires a *face sweep* + *vertex sweep* (+ optionally *flip*), which is the most challenging process.

- The Teacher module calculates the learner's score for a problem from the ratio of number of correct faces to total number of faces of the closest solution solid, minus penalties for using commands that indicate a non-ideal solution sequence, especially the *destruct* command, or excessive use of *next face* hint generation. The learner can ask for the current score at any time. When the learner finishes the problem, VRT calls the Teacher module to calculate the learner's final score, and returns it to IVRT.

INTELLIGENT VISUAL REASONING TUTOR

IVRT is a successor system that embeds the Visual Reasoning Tutor module within an intelligent tutoring system framework [25]. Hence, results of our earlier studies conducted with previous versions of the VRT module are carried over to IVRT itself.

Learning contents and learner model

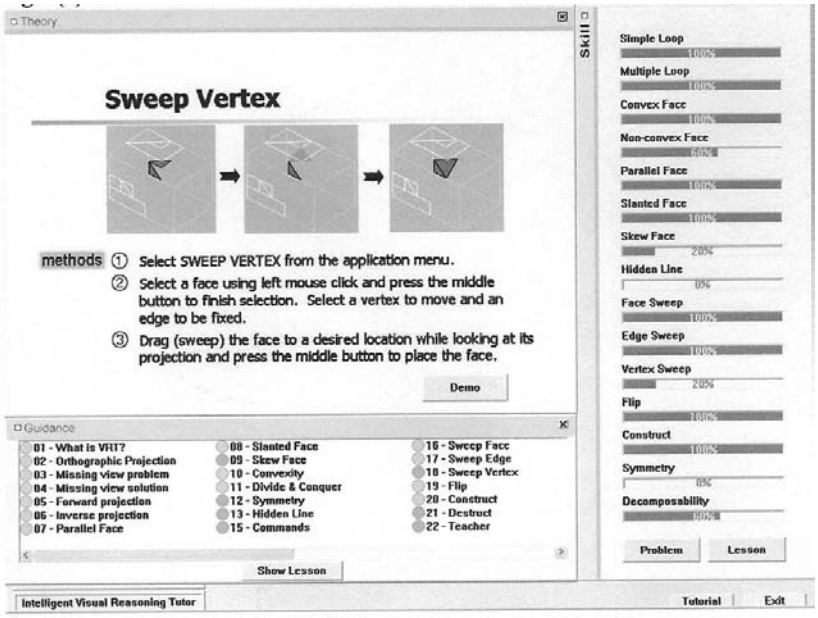
IVRT defines a set of learning contents consisting of skills, lessons, and problems, and a learner model that records learners' skill scores and activity history.

- A lesson is a text or multimedia resource that presents the core concepts for missing view visual reasoning, including the orthographic projection of a solid onto viewing planes, and the inverse operation of converting the orthographic projections back to 3D solid faces.
- A problem is an instance of a missing view problem, consisting of two orthogonal views. The learner's objective is to create a valid 3D object that is consistent with both views, using the Visual Sweeper module.
- A skill is a domain concept or problem-solving process, which has been identified *a priori* by the teacher or domain expert as having significant pedagogical value, requiring explicit instruction and assessable expertise. We have identified a set of 15 skills for the missing view visual reasoning domain, as shown in Table 2.

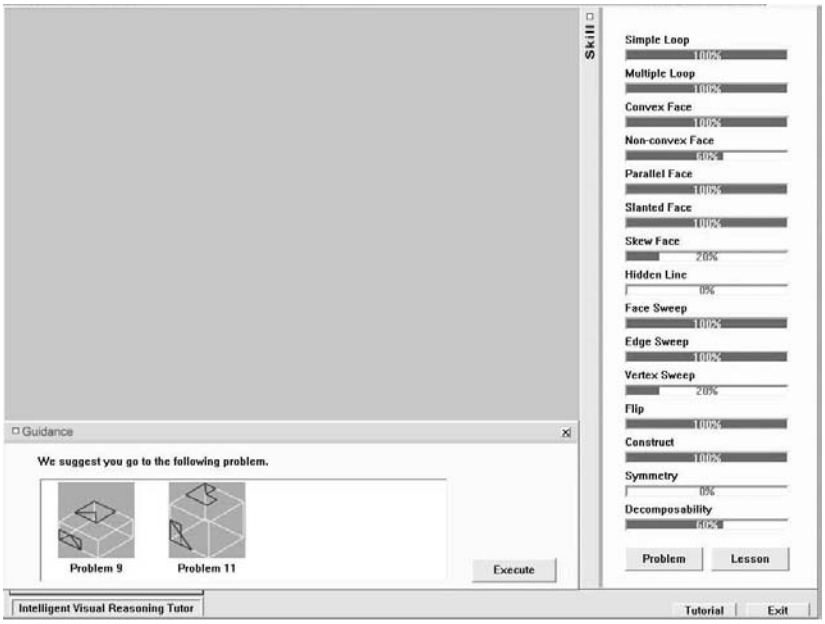
A learner's domain competence is measured by the learner's skill levels, as a set of scores in the interval [0, 100], plus additional data reflecting the learner's command history within the Visual Sweeper module. A new learner begins at 0 score in every skill. As the learner solves problems, the learner earns skill points. This causes more advanced lessons and problems to become available for selection. The learner's objective is to increase every skill score to 100, which completes the tutorial.

Table 2. Skills in Intelligent Visual Reasoning Tutor

#	Skill Name	Description
1	Simple Loop	Handling orthographic views with simple (single) loops only.
2	Multiple Loop	Handling orthographic views with multiple loops.
3	Convex Face	Identifying and creating convex faces.
4	Non-convex Face	Identifying and creating non-convex faces.
5	Parallel Face	Creating faces that are perpendicular to two orthographic views.
6	Slanted Face	Creating faces that are perpendicular to only one orthographic view.
7	Skew Face	Creating faces that are not perpendicular to any orthographic view.
8	Hidden Line	Handling orthographic views with hidden (dashed) lines.
9	Face Sweep	Skill in applying the *face sweep* operation.
10	Edge Sweep	Skill in applying the *edge sweep* operation.
11	Vertex Sweep	Skill in applying the *vertex sweep* operation.
12	Flip	Skill in applying the *flip* operation to reverse a face's normal vector.
13	Construct	Skill in applying the *construct* operation to create a face whose projection in both orthographic views is an edge.
14	Symmetry	Handling problems in which the front and top views are the same.
15	Decomposability	Skill in applying a divide-and-conquer strategy.



(a) Lesson selection and display



(b) Problem selection

Fig. 5. Learner's Interface with skill bars, lesson guidance, and problem selection.

Learner's Interface

The Learner's Interface provides skill level display as a set of skill bars, and interactive selection of lessons and problems based on the learner's current skill scores, shown in Fig. 5.

When the learner requests the current set of lessons, a comprehensive list of all lessons is presented, as shown at the bottom of Fig. 5(a). Within this list, lessons are color-coded to indicate the learner's mastery. Green dots indicate relevant lessons based on the learner's current skill scores, gray dots indicate lessons that the learner has already mastered, and white dots represent advanced lessons whose requirements the learner has not yet fulfilled. For convenience, the comprehensive list of all lessons is always shown, which allows learners to consult earlier lessons at any time.

The problem selection window, in Fig. 5(b), shows the subset of available problems based on the learner's current skill scores. The learner may choose any problem within this subset, which invokes the VRT module.

IVRT's user interface (including VRT's interface during each problem session) provides a learner-initiated activity mode only. Furthermore, all remediation within each VRT problem session is generated via geometric reasoning, is displayed

graphically instead of through text, and is provided on request only. Hence, IVRT's learning contents does not need to include a library of text messages for remediation.

CUSTOMIZATION OF LEARNING CONTENTS

At a high level of abstraction, learning contents are composed of a set of elements, organized into a graph structure that encodes various pedagogical orderings over these elements. We have formalized IVRT's learning contents and learner model as ontologies, using the ontology-based framework shown in Fig. 6. The teacher or domain expert can customize the learning contents by editing the ontology files, using standard ontology tools including Protégé and OWL.

We briefly describe the major portions of the learning contents ontology. First, we define a Node base class to represent a generic learning element, with three subclasses Skill, Lesson, and Problem. We enumerate the 15 skills shown in Table 2 as 15 individuals of the Skill class. Each Lesson and Problem individual is defined by several string properties: title string, internal symbol, and the names of their associated multimedia or problem-data files.

Next, we define an Edge base class to represent one edge of the learning contents graph between two elements. As we must annotate these edges with various attribute data to encode pedagogical knowledge, we cannot represent them using OWL properties, so we have reified them as classes. We define three subclasses of Edge:

- LessonSkillEdge associates one lesson to one skill, with one attribute, which is a numeric interval of scores in that skill. More generally, each lesson shall provide instruction in 1 or more skills, as decided by the domain expert, and we encode these relations as a set of LessonSkillEdges individuals. The domain expert can define many lessons that cover a given skill, to give the learner multiple opportunities to acquire that knowledge. A single lesson can also be used to describe several skills. The score interval attribute encodes a notion of the complexity level of the lesson: a low interval indicates a lesson suitable for beginners, while a high interval restricts a lesson to learners at higher levels of mastery.
- ProblemSkillEdge associates one problem to one skill, with two attributes: a score interval, as for LessonSkillEdge, and a numeric reward, which a learner shall earn in this skill by solving this problem correctly. More generally, each problem requires the use of 1 or more skills to be solved successfully, and the domain expert encodes these as a set of ProblemSkillEdge individuals for each problem.
- SkillSkillEdge defines a prerequisite relationship from one skill A to another skill B, with one attribute, which is a score threshold. If the learner's score in skill A is greater than or equal to this threshold, then this prerequisite relation is satisfied; when all prerequisites to skill B are satisfied, then skill B is "activated".

Using this ontology, the domain expert organizes the learning contents as a lattice structure of elements, by which the expert can impose a partial ordering over the possible sequences of instruction. However, each learner is free to pursue an individualized path through the learning contents. In principle, learners could gravitate toward different subsets of lessons and problems according to their individual preferences, receive problem-solving opportunities involving different subsets of skills, and thereby earn points in those different skills, which advances them into different regions of the learning contents lattice.

IVRT uses a simple learner model consisting of 15 skill scores, a Boolean flag per lesson indicating whether it has been viewed or not, and a score per problem indicating the outcome of the learner's

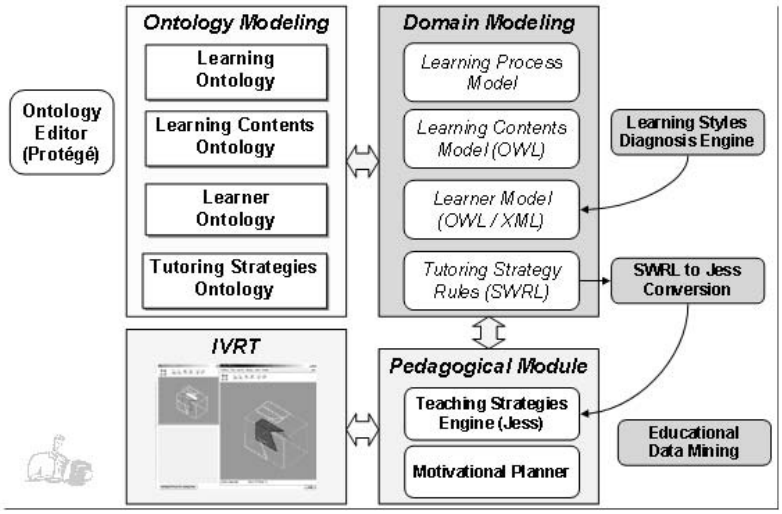


Fig. 6. Framework for ontology-based intelligent tutoring system.

most recent attempt at solving that problem in VRT. We also record various other low-level data, e.g. VRT operations and error counts, and VRT problem score histories, and these could be incorporated into future versions of IVRT's teaching strategy.

PEDAGOGICAL RULES IN IVRT

The pedagogical teaching strategy determines the presentation sequence of lessons and problems based on a learner's skill scores and other history data. IVRT implements one teaching strategy, which shows a subset of lessons and problems based on the learner's current skill scores. Items that the learner has already mastered, and items for which the learner is not yet ready, are not shown. As the learner reads lessons, solves problems, and earns increases in skill scores, new lessons and problems will be "activated", while old lessons and problems may be considered to be "mastered", and are suppressed. In this way, the domain expert can define a default order for the presentation of the course material, while still allowing an adaptive response based on each learner's actual progress.

We implement this pedagogical strategy using inference rules. Selected rules are shown in Table 3, in natural language. Terms used in these rules represent run-time data fields associated with learning content elements *for the current learner*. This presents an intriguing problem in data management using ontologies, namely whether such data fields should be defined within the learning contents ontology, or within the learner model. We will explore this issue more thoroughly in future work.

- A skill has an *activeness* property, which is true or false, and zero or more *required* skills, which are specified by the SkillSkillEdge individuals.
- A lesson or problem has a *visibility* property, which is either true (the lesson or problem is shown to the learner) or false (hidden). It also has one or more *associated* skills, as given by the LessonSkillEdge and ProblemSkillEdge individuals described previously.
- A global property of skill satisfaction is defined by a system predicate, which takes a skill (including its score) and returns true or false. The default test compares the score to a numeric threshold. Each teacher can customize this test.

We have implemented these high-level rules as a set of approximately 40 detailed rules in our customized version of Semantic Web Rule Language (SWRL). Rules are edited using the Protégé SWRLTab rule editor, shown in Fig. 7, and are saved in a standard Protégé OWL ontology file. We have developed an automatic conversion from our customized SWRL with extension keywords in Protégé format [26][27] to the Jess rule engine. The teacher can customize IVRT's teaching strategy by editing this ontology file and reapplying the conversion (even while IVRT is running).

Table 3. Selected teaching strategy rules for IVRT

	#	Description
Rules to Activate Skills	1	For each skill: if it has no required skills, activate this skill.
	2a	For each skill: if **any** required skill is not satisfied, deactivate this skill.
	2b	For each skill: if **all** required skills are satisfied, activate this skill.
Rules to Show and Hide Lessons	3a	(Default strategy) For each lesson: if **all** associated skills are active, show it.
	3b	(Default strategy) For each lesson: if **any** associated skill is inactive, hide it.
	4	(Special strategy) For each lesson: if **any** associated skill is active and within its score interval, show it.
Rules to Show and Hide Problems	5a	For each problem: if **all** associated skills are active and within their score intervals, show it.
	5b	For each problem: if **any** associated skill is inactive or outside its score interval, hide it.

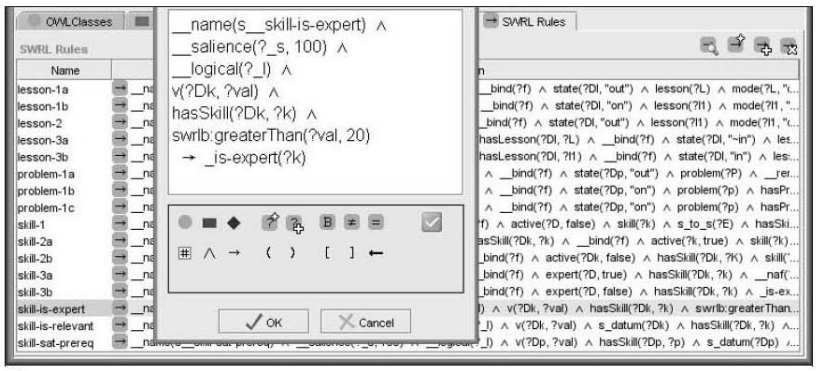


Fig. 7. Editing of teaching strategy rules in Protégé SWRLTab.

The complete IVRT application consists of three concurrent processes. The Learner's Interface is a stand-alone application in Visual C++. On start-up, it executes Jess as a child process, with bidirectional synchronous (blocking) communication using sockets, and instructs Jess to load the learning contents ontology and teaching strategy rules (including any customizations) for missing view visual reasoning, which bootstraps the teaching strategies engine. All pedagogical decision-making is thereafter executed in Jess.

When a learner logs into IVRT, the Learner's Interface loads the learner's record from XML data files, displays its skill scores in the skill bars, and sends the learner's state data to Jess, which applies the teaching strategy rules to compute the learner's available lessons and problems. When the learner requests a list of lessons (problems), the Learner's Interface queries the Jess module for the current list, then displays them. When the learner selects a problem, the Learner's Interface invokes the VRT module, and blocks until the learner finishes that problem session and closes the VRT module. The learner's score from the VRT session is sent to the Jess module, which updates the learner's skill scores, lessons, and problems, and these are written to the learner's XML record.

CONCLUSION AND FUTURE WORK

IVRT is an intelligent tutoring system for visual reasoning, suitable for use at the undergraduate level. We have previously conducted several studies indicating that the VRT module within IVRT measurably improves learners' visual reasoning skills for missing view problems [22], and insofar as VRT is similar across both versions, we expect that these results are carried over to IVRT.

As visual reasoning involves cognitive processes essential in problem solving, learner adaptive and structured supports of IVRT in developing such underlying reasoning skills would contribute in engineering education. It would be desirable that learner adaptive and structured learning support for the underlying skills are provided using computer-based learning systems like IVRT, while many prototyping activities dealing with physical objects in real 3D space are also provided to help engineering students develop required visual reasoning skills.

Advantages of the two-level reasoning architecture

A novel aspect of IVRT is its explicit use of a two-level reasoning architecture, combining geometric reasoning and semantic technologies, while at the same time deliberately maintaining a separation between them. We see this as a promising and necessary development to extend ITSs into domains that are "complex", yet are still tractable. In particular, 3D geometry is a mature and well-understood branch of mathematics, for which rigorous and highly optimized representations and reasoning methods have already been developed. In contrast, intelligent tutoring is a younger field that emphasizes adaptiveness, flexibility, and customizability, and here the state of the art lies toward knowledge engineering-based approaches using semantic technologies for learning.

In developing an ITS involving 3D geometry, we combine the best of both fields of study by using both reasoning approaches in a hybrid manner, where each kind of reasoning is applied where it is most suitable. It would be a disservice to rely on semantic technologies to implement a geometric reasoning engine, and thus we see no drawback in simply maintaining two separate reasoning modules. We obtain a significant benefit from this, as can be seen from IVRT's Visual Teacher module. The specialized power of geometric reasoning allows us to generate assessments and guidance hints with essentially no domain knowledge base, and only a tiny set of rules, which stands in sharp contrast to other ITSs. On the other hand, teaching strategies and learning contents navigation are best handled using semantic technologies, which exploits the proven techniques from ITS research.

Future enhancements

We plan to enhance numerous aspects of IVRT. Foremostly, we will expand IVRT's use of semantic technologies.

- IVRT's teaching strategy rules currently implement a single high-level teaching strategy. We will implement additional teaching strategies through an integration with an emerging standard ontology of learning, which defines many standard teaching strategies [18]. We will also add a layer of meta-strategy rules to choose between multiple teaching strategies.
- Similarly, IVRT's learning contents ontology is perhaps overly optimized in defining precisely the set of properties needed to support the current rule set. We will expand the richness of the learning contents model to support new fine-grained reasoning capabilities.
- When we consider IVRT at the grain size of individual problems, IVRT's guidance and adaptive support are adequately provided by geometric reasoning. However, the current level of feedback is based strictly on the learner's solution state only, and is not adaptive to the learner's preferences or other state information. A plausible approach to overcome this is to expand VRT's data-logging capabilities, obtain the learner's fine-grained actions, and relate these to the eight components of our visual reasoning model, which are: perception, analysis, interpretation, generation, transformation, maintenance, internal representation, and external representation [28].
- At a higher grain size, we face an issue of combining the guidance and adaptive support

across multiple problems, e.g. to implement scaffolding, fading, and other established teaching strategies in which guidance given in previous problems influences the generation of guidance for future ones. We plan to address these issues through application of semantic technologies, e.g. by modeling guidance events explicitly as new elements of our learning contents ontology.

We will also explore ways to diagnose the learner's preferences and learning styles through their user interface behavior [29], which also entails an enhancement of VRT's data-logging facilities.

REFERENCES

1. Y. S. Kim, M. H. Kim, and S. T. Jin, Cognitive Characteristics and Design Creativity: An Experimental Study, *Proc. ASME Int'l. Conf. Design Theory and Methodology*, Long Beach (2005).
2. D. J. Wilde, The geometry of spatial visualization: two problems, *8th IFTOM World Congress*, Prague (1991).
3. R. McKim, *Experiences in Visual Thinking*, Brooks & Cole Publishing Co., Monterey (1972).
4. T. Murray, Principles for Pedagogy-Oriented Knowledge Based Tutor Authoring Systems, in T. Murray, S. Blessing, and S. Ainsworth (Eds), *Authoring Tools for Advanced Technology Learning Environments*, Chapter 15, Kluwer, Boston (2003).
5. A. T. Corbett and J. R. Anderson, Student Modeling In An Intelligent Programming Tutor, in E. Lemut, B. du Boulay, and G. Dettori (Eds), *Cognitive Models and Intelligent Environments for Learning Programming*, Springer-Verlag, Berlin (1993).
6. K. Koedinger, J. Anderson, W. Hadley, and M. Mark, Intelligent Tutoring Goes To School in the Big City, *Int. J. Artificial Intelligence in Educ.* **8**, (1997) pp. 30–43.
7. O. Scheuer, M. Mühlenbrock, and E. Melis, Results from Action Analysis in an Interactive Learning Environment, *J. Interactive Learning Research,* (2007).
8. N. Matsuda and K. VanLehn, Advanced Geometry Tutor: An intelligent tutor that teaches proof-writing with construction, in C.-K. Looi, G. McCalla, B. Bredeweg, and J. Breuker (Eds.), *Proc. 12th Int'l. Conf. on Artificial Intelligence in Education (AIED)*, IOS Press, Amsterdam (2005) pp. 443–450.
9. K. G. Schulze, R. N. Shelby, D. J. Treacy, M. C. Wintersgill, K. VanLehn, and A. Gertner, Andes: An active learning, intelligent tutoring system for Newtonian physics, *Themes in Educ.* **1**(2), (2000) pp. 115–136.
10. A. Mitrovic, An Intelligent SQL Tutor on the Web, *Int. J. Artificial Intelligence in Educ.*, **13**, (2003), pp. 171–195.
11. P. Suraweera and A. Mitrovic, An Intelligent Tutoring System for Entity Relationship Modelling, *Int. J. Artificial Intelligence in Educ.*, **14**, (2004) pp. 375–417.
12. K. Zakharov, S. Ohlsson, and A. Mitrovic, Feedback Micro-engineering in EER-Tutor, in C.-K. Looi, G. McCalla, B. Bredeweg, and J. Breuker (Eds), *Proc. Artificial Intelligence in Education (AIED)*, IOS Press (2005) pp. 718–725.
13. M. Mayo and A. Mitrovic, Optimising ITS Behaviour with Bayesian Networks and Decision Theory, *Int. J. Artificial Intelligence in Educ.*, **12**(2), (2001) pp. 124–153.
14. A. C. Graesser, P. Chipman, B. C. Haynes, and A. Olney, AutoTutor: An intelligent tutoring system with mixed-initiative dialogue, *IEEE Transactions in Educ.* **48**, (2005) pp. 612–618.
15. A. I. Direne, Designing Intelligent Systems for Teaching Visual Concepts, *Int. J. Artificial Intelligence in Educ.*, **8**, (1997), pp. 44–70.
16. L. Aroyo and R. Mizoguchi, Process-aware Authoring of Web-based Educational Systems, Proc. 1st Int'l. Workshop on Semantic Web for Web-based Learning (SW-WL), in J. Eder, R. Mittermeir, and B. Pernici (Eds), *The 15th Conf. on Advanced Information Systems Engineering (CAiSE)*, Klagenfurt/Velden, Austria (2003).
17. L. Aroyo, A. Inaba, L. Soldatova, and R. Mizoguchi, EASE: Evolutional Authoring Support Environment, in J. C. Lester, R. M. Vicari, and F. Paraguaçu (Eds), *Lecture Notes in Computer Science*, **3220**, pp. 140–149, Springer (2004): *Proc. 7th Int'l. Conf. Intelligent Tutoring Systems (ITS)*, Maceió, Brazil (2004).
18. R. Mizoguchi and J. Bourdeau, "Using Ontological Engineering to Overcome AI-ED Problems", *Int. J. of Artificial Intelligence in Educ.*, (2000) **11**(2), 107–121.
19. R. Mizoguchi, Y. Hayashi, and J. Bourdeau, "Inside Theory-Aware and Standards-Compliant Authoring System", *Workshop on Ontologies and Semantic Web Services for Intelligent Distributed Educational Systems (SWEL)*, *13th Int'l. Conf. Artificial Intelligence in Education (AIED)*, Marina Del Rey (2007) pp. 5–18.
20. M. Mäntylä, *An Introduction to Solid Modeling*, Computer Science Press (1988).
21. Y. S. Kim, C. Moon, S. Chauhan, C. Hubbard, O. J. Mengshoel, and H. Zhao, Visual Reasoning Tutor (VRT): Instructional Software System for Missing View Problem, *Proc. ASEE Engineering Design Graphics Division Conf.*, Ames, IA (1995).
22. C. Hubbard, O. J. Mengshoel, C. Moon, and Y. S. Kim, Visual reasoning instructional software system, *Educ.*, **28**(4), (1997) pp. 237–250.
23. H. Zhao and Y. S. Kim, Geometric Operations for Visual Reasoning of a Solid from Orthographic Projections, *Advances in Eng. Software*, **30**(7), (1999).
24. O. J. Mengshoel, S. Chauhan, and Y. S. Kim, Intelligent Critiquing and Tutoring of Spatial Reasoning Skills, *AI in Eng. Design, Analysis, Manufacturing*, **10, (**1996).
25. E. Wang and Y. S. Kim, Intelligent Visual Reasoning Tutor, *Proc. 5th IEEE Int'l. Conf. on Advanced Learning Technologies (ICALT)*, Kaohsiung, Taiwan (2005).

26. E. Wang and Y. S. Kim, A Teaching Strategies Engine using Translation from SWRL to Jess, in M. Ikeda, K. Ashley, T.-W. Chan (Eds), *Lecture Notes in Computer Science*, **4053**, Springer (2006): *Proc. 8th Int'l. Conf. Intelligent Tutoring Systems (ITS)*, Jhongli (2006).
27. E. Wang and Y. S. Kim, Using SWRL for ITS through Keyword Extensions and Rewrite Meta-Rules, *Workshop on Ontologies and Semantic Web Services for Intelligent Distributed Educational Systems (SWEL), 13th Int'l. Conf. Artificial Intelligence in Education (AIED)*, Marina Del Rey (2007) pp. 101–105.
28. J. A. Park and Y. S. Kim, Visual Reasoning and Design Processes, *Int. Conf. Eng. Design (ICED),* Paris (2007).
29. H. J. Cha, Y. S. Kim, S. H. Park, T. B. Yoon, Y. M. Jung, and J. H. Lee, Learning Styles Diagnosis based on User Interface Behaviors for the Customization of Learning Interfaces in an Intelligent Tutoring System, in M. Ikeda, K. Ashley, T.-W. Chan (Eds), *Lecture Notes in Computer Science*, **4053**, Springer (2006): *Proc. 8th Int'l. Conf. Intelligent Tutoring Systems (ITS)*, Jhongli (2006).

Yong Se Kim is the Director of the Creative Design Institute at Sungkyunkwan University, and a Professor of Mechanical Engineering at Sungkyunkwan University. He received his Ph.D. degree in Mechanical Engineering from the Design Division of Stanford in 1990. His research interest is *Design Cognition and Informatics*, which investigates the fundamental processes in design, and provides methodologies and computer-based tools for design and design learning. Specific research topics include cognitive processes of design, design creativity, design for affordance and customization, product-service systems design methodology, feature-based design reasoning, visual and geometric reasoning, and semantic technology for learning.

Eric Wang is a Senior Researcher in the Creative Design Institute at Sungkyunkwan University. His research interests include ontology modeling, rule-based reasoning, intelligent tutoring systems, geometric reasoning, feature-based computer-aided process planning, and systems development and implementation.