Non-visual Expression Method for Mathematical Documents in Elementary Geometry

Ryoji Fukuda
rfukuda@oita-u.ac.jp
Faculty of Engineering,
Oita University
Japan

Masato Kojo
v13E6012@oita-u.ac.jp
Graduate Scool of
Engineering,
Oita University

Abstract: *We propose a method for expressing mathematical contents in elementary geometry. To realize similar scenarios when giving information elements in elementary geometry documents for people with visual disabilities, we assume that graphical elements are input by a handwriting system, and the recipient obtains several types of documents on demand. We therefore need some functions for making a document graph from several graphical elements. We define the expression rules for translation functions and the output document graphs.*

1. Introduction

Figures in mathematical expressions are useful when visual communication methods are available, but can be barriers for someone who is unable to view them. Consider a situation in which a student with visual disability attends a mathematics class, and the other students are sighted. It may be very difficult, maybe even impossible, to completely avoid the above problem, even when personal support is available. Our future goal is to develop software for expressing figures using non-visual methods in real-time communication.

Note taker is a service for people with hearing impairment. During lectures in some universities, for example, some supporting staff members will write down any verbal interactions if necessary. The students with a hearing impairment can then obtain the messages in voice form. This is not an easy task and may require some training; these days, some training courses exist for this type of service, and some assistive software for this service has been recently developed. Consequently, many students with hearing disabilities take lectures using such services.

Standard explanations are given orally during lectures, and students can comprehend them using a textbook, notebook, or other content. Thus, the support for students with visual impairment usually concentrates on expressions or the creation of non-visual communication contents. In fact, many students with visual disabilities learn with the aid of a supporter, who creates braille translations or tactile figures. In some lectures with mathematical contents, especially in the case of geometrical concepts, several figures containing various information tips are used. Since these figures have many aspects to them, we have to pick up several elements, or close up some parts of them. The teacher then draws a different figure on a black board, or orally explains the new figure.

Students can understand a rough sketch or an imaginary figure if they have normal eyesight, that is, these contents depend on the ability of the human eye-brain system. Thus, some temporal figures that are difficult to understand for students with visual disabilities are used in a lecture. Our software is a graphical input system for elementary geometry [1]. The main aim of this software is to aid the supporters of students with visual disabilities in an elementary geometry class for providing temporal graphical contents, where the output is a set of explanations.

There may be many different elements of information in a single figure, and we use a few of them in one situation. Sighted students select these elements in the figure according to the teachers'

explanations or their own experience. Such figures may be reminders for corresponding properties. The total information given by the figure is a set of these concepts and properties. We call such a graph an "explanation graph," and therefore, our software outputs an explanation graph for the graphical information.

According to our objective, for a real-time use of supply information, the output must be constructed quickly. We prepared some functions to output an explanation graph. These functions have several parameters, which are graphical elements. A software user then inputs several graphical elements, e.g., line segments and circles. Then, they select a function with some graphical elements, which are parameters of the function. Consequently, the system outputs the explanation graph. The output graph is a set of explanations. Each explanation is a character string, and is connected to the other character strings. Using a string voice output, these are available in non-visual communication form.

We have to prepare explanation functions before using our system. The input elements of the function are graphical elements, and the output is an explanation graph. We define the explanation rules for the contents of the graphical elements and explanation outputs. These are expressed using the XML format. We prepared an XML file for explaining the functions. We input some graphical elements using our handwriting input system, select the function and corresponding parameter elements, and then obtain the output expectation graph. In the software, we obtain the text output according to the graph structure. This is also expressed in XML format, and voice or Braille output can be available if we create translation software from our explanation graph.

2. Outline of the Assumed System

Our future goal is to create an assistive software program for a volunteer supporting a student with visual impairment during a mathematics class. The target contents are graphical elements in elementary geometry.

These contents are barriers for people with visual impairment. The human eye-brain system creates many types of information elements from the same figure on demand. This ability accelerates the difficulty of this problem. In this section, we explain the specifications of our assumed system, which are based on our previous system [1]. With this software, we can overcome the above difficulties.

2.1 Input of Information Elements.

The purpose of the system is to construct an explanation graph structure. Nodes in the graph are short sentences (explanations), and are connected to each other based on sentence structures. For each node, we use certain graphical elements that are input using a mouse.

The mutual relations between two elements are important and must be pinned downed when the user inputs the elements. Table 1 lists the graphical elements, and Table 2.2 lists the relations and properties between two elements, which are input using a mouse. First, the user draws a curve. The system then recognizes the curve type. If there are several elements, relations between new input element and each existing element. We can check

Basic Element	Filiations
Point	Origin
Line segment	X and Y axes
Angle	Right angle
Triangle	Regular triangle, Right angled triangle, isosceles triangle
Quadrangle	Regular tetragon, Rectangle, Parallelogram
Circle	
Arc	Half circle

Table 2.1 Basic Elements

the recognition results for the type of graphical element and the relations among other elements; in addition, the input elements and their relations are editable. One important point is that the system obtains these types and relations or properties because they require a non-visual expression. The following shows a standard method for inputting the graphical contents.

1. A curve is drawn using a mouse, and the system recognizes its curve type. The drawn curve is then replaced by a smooth printed curve. If some curves already exist before the user's curve is drawn, corresponding relations are suggested. Incorrect recognition results are corrected.
2. Some existing elements, or certain parts of an element, are selected to create a new element.
3. Some relations are added if necessary.

Ellements' Pair	Relation, Property
Circle/ Point	Center point, On the curve
Circle/Line segment	Radius, Diameter, Arc, Cross, Tangent
Circle/Triangle	Inscription,Circumscription
Circles	Touch, Included,Cross
Line segments	Cross, Orthogonal, Parallel

Table 2.2 Elements' Relation

Property		Detail
Mid point theorem		
Triangular congruence condition	1 2 3	3 sides 2 sides and 1 angle 1 sides and 2 angles
Triangular similarity condition	1 2 3	3 sides 2 side and 1 angle 2 angles
Circumferential angle theorem		
Pythagorean theorem		

Table 2.3 Properties

2.2 System Database

We suppose the real time use of the system, then it may be quite difficult to create sufficient information elements. From a simple figure, a sighted student may grasp several information elements. Then we make a set of information elements using premade definitions or properties. The definitions and properties contains the method to explain them on demand. The expression rules are described in the next section. This database will depend on a field of target document, level of the students and so on. If we try to create global database, its size may be very large. The definitions of all elements in Table 2.1 and all relations (properties) in Table 2.2 are standard necessary elements in a database for a starting course to study elementary geometry. Moreover, we need some properties (theorem, lemmas), Table 2.3 is a list of basic and standard properties to explain contents in elementary geometry.

2.3 Creation of Explanation

Explanations corresponding to graphical elements or relations need to be flexible, that is, our demands for the fineness or the types of the explanation may be variable according to a situation or knowledge. Then, we prepare a several types of explanation for each definition and property, we describe their details or expression methods in the next section. Thus we can make a database for the on demand use of input information elements. In the software these are treated as follows.

1. Several elements are input and the element types and mutual relations are determined.
2. Select some elements among them and chose an explanation function.
3. Output the explanation graph.

Explanation functions are defined in some definitions or properties in the database. Each explanation function has its parameter elements in general. Then, after selecting some elements as its parameters, possible explanation function are selected and the user chose one of them.

2.4 Output of Explanation

The output of the system is a graph structure of explanations. Node of the graph is a small sentences, which is a short explanation of a definitions or properties or other simple message. A definition contains one condition, and a property may contain one condition, one statement and some proofs. Definitions and properties have several parameters and several related items. A parameter is a definition, that is, this is graphical elements described by using a definition. A related item is a definition or a property or a simple message.

Key	Name	Detail
p	previous	Change to the previous node.
n	next	Change to the next node.
m	parent	Change to the parent node.
c	condition	Change to the first node of the condition.
s	statement	Change to the first node of the statement.
r	proof	Change to the first node of the proof.
d	detail	Output the detailed explanation
t	top	Output simple explanation

Table 2.4 Roles of Keys

We suppose that visually impaired persons use these output graphs. First the system focuses on the top node and, with a keyboard input, the user obtain corresponding text output or change the node. Table 3.1 list the roles of the keys.

3. Rules for Explanation Graph

In this section, we explain the basic rules for expressing an explanation graph. These rules are described using the XML format. Before using the system in a new mathematical field, we have to prepare a database of severalt sets of definitions and properties. These are stored as an XML file (a text file). Information elements or function calls are also expressed as an XML file, and may be used only for internal processing.

Tag Name	Attribution	Meaning
EgRealNum	realCtt	real number
EgInteger	intCtt	integer
EgBool	boolCtt	boolean
EgChar	charCtt	character
EgString	strCtt	string
EgVec2	vec2Ctt	2-dim vector
EgVec3	vec3Ctt	3-dim vector

Table 3.1 Basic Elements

3.1 XML Elements of the Document

An XML element is expressed using a tag name. As an example, We define a tag name “Point.” An XML element starts with **<Point [*attr*]>** and ends with **</Point>**, and when there are no sub-elements, it is

expressed as <Point [*attr*] />. The [*attr*] part is replaced by a set of attribute definitions. Each attribute has its own basic data type. First, we describe three attributes of our expression rules as "name," "type," and "refName," which are string attributes (EgString). The "name" attribute is an identifier of the element, and must be unique in the corresponding group.
Consider a case in which the "name" values are equal for two elements. If these are sub-elements of the same parent element, it contradicts our grammar. There is no contradiction when these are sub-elements of a different parent element.

The "type" attribute represents the data type of the element. The value of the "type" is a name of the basic data type or other data type defined according to our rules. The "rename" attribute is the name of another element. If there is a predefined element, we can refer to this element using this attribute. The data type of the corresponding element must then be same as the data type defined using the "type" attribute.

Recalling the <Point> element, when we describe an element,

<Point type="EgVec2" name="pt." v2Ctt="(0.0,0.0)"/>,

which implies that this element is expressed by the name, "pt," and the corresponding value (coordinate) is (0.0, 0.0). Sequential data are expressed using an "Array" element. We can define several "EgArrayElm" and an element "EgGenTerm." However, this grammar does not require a numerical calculation function, and we can describe concrete values or general formulae for general terms.

3.2 Description for Definitions

We describe a new data type using the element "EgDefinition." This element contains the "name" (EgString) and "paraNum"(EgInteger) attributes, one "EgCondition" sub-element, and several other sub-elements. The "name" attribute is used as a tag name in a description of the corresponding elements, i.e., the parameters. Using these parameters, we can describe the condition of the definition, i.e., "EgCondition." These concepts are described through the following definition.

```
<EgDefinition name="EgeRAngle" paraNum="5">
        <EgePoint name="a" />
        <EgePoint name="b"/>
        <EgePoint name="c"/>
        <EgeLSeg name="ab">
                <EgePoint name="st" refName="a"/>
                <EgePoint name="ed" refName="b"/>
        </EgeLSeg >
        <EgeLSeg name="bc">
                <EgePoint name="st" refName="b"/>
                <EgePoint name="ed" refName="c"/>
        </EgeLSeg >
        <EgCondition >
                "@ab and @bc meet at right angles."
        </EgCondition>
</EgDefinition>
```

This element has five parameters, and the first five elements must be the parameters. The "EgCondition" element describes the text output, and its sub-elements are the definitions, properties, and "EgMess" elements (which represents a simple string). If there is only one "EgMess" sub-element, a string is input directly.

For the "EgCondition" element, we use its defined parameters. A parameter is expressed using

“@” followed by the name of the target element, and after the name string, we need at least one space. When we describe a sub-element of a parameter, a period followed by the name of sub-element is added after the parameter name. The description “@a.x > @b.x” implies that the x-coordinate of point “a” is greater than that of point “b.”

3.3 Expression of Properties

Mathematical properties are described using the “EgProperty” element. This element has “name” and “paraNum” attributes, and “EgCondition,” “EgConclusion,” and “EgProof” sub-elements. These sub-elements are sets of XML elements, and may be “EgDefinition,” “EgProperty,” or some other element. Our grammar for these expressions does not require each of these elements to be given. We define an expression method for various aspects of the documents.

The following XML element is an expression of the Pythagorean Theorem. In this element, we describe a condition and a statement. There are no proofs for this element.

3.3 Explanation Function

Mathematical properties are described using the “EgProperty” element. This element has “name” and “paraNum” attributes, and “EgCondition,” “ EgConclusion,” and “EgProof” sub-elements. These sub-elements are sets of XML elements, and may be “ EgDefinition,” “EgProperty,” or some other element. Our grammar for these expressions does not require each of these elements to be given. We define an expression method for various aspects of the documents.

Explanation function	Parent Element	Detail
EgName	EgDefinition EgProperty	Title of an element
EgSurface		Simple explanation. Default explanation.
EgDetail		Detailed explanation
EgCondition		Condition
EgStatement	EgProperty	Statement of Proof (without the condition)
EgProof		Proof of the property.

Table 3.2 Explanation Functions

The following XML element is an expression of the Pythagorean Theorem. In this element, we describe a condition and a statement. There are no proofs for this element.

```
<EgProperty name=”PythagoreanTh” paraNum=”1” >
        <EgTriangle name=”tr” praNum=”7”>
                <EgPoint name=”vtx1”/>
                <EgPoint name=”vtx2”/>
                <EgPoint name=”vtx3”/>
                <EgLSeg name=”side1_2” />
                <EgLSeg name=”side2_3” />
                <EgLSeg name=”side3_1” />
                <EgAngle name=”agl1”/>
        </EgTriangle>
        <EgCondition>
                @tr1.agl1 is right angle.
        </EgCondition>
        <EgStatement>
                ”@side2_3  ^2 = @side1_2 ^2 + @side3_1 ^2”
        </EgStatement>
</EgProperty>
```

4. An Example of an Explanation Graph

In this section, we consider the explanation graph shown in Figure 4.1. This graph explains that the angles for the same arc of a round circle are equal. Using this property, the angle ABC is equal to the angle AB'C.

4.1 A General Property in Database

We need a general property for the theorem of circumference angles. We assume that the following properties are described in a database.

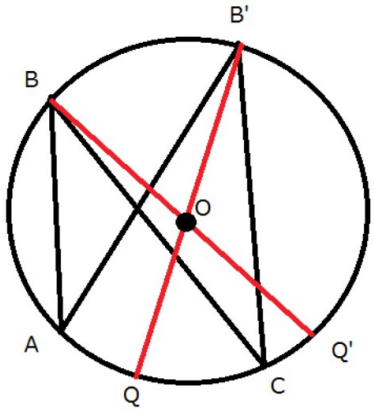

Figure 4.1 Circumference Angles

```
<EgProperty name="CircumTh" paraNum="10">
        <EgPoint name="A" />
        <EgPoint name="B"/>
        ……
        <EgAngle name="b" paraNum="3">
                <EgPoint name="P1" refname="A" />
                ……
        < /EgAngle>
        ……
        <EgName strCtt="Circumference Theorem"/>
        <EgSurface >
                <EgMess strCtt="@b is equal to @b'"/>
        </EgSurface>
        ……
        <EgProof>
                <EgProperty name="CircumAndCenterAngle" paraNum="2">
                        <EgAngle name="a1" refName="b"/>
                        <EgPoint name="O"   refName="O"/>
                </EgProperty >
                <EgProperty name="CircumAndCenterAngle" paraNum="2">
                        <EgAngle name="a2" refName="b'"/>
                        <EgPoint name="O"   refName="O"/>
                </EgProperty >
                <EgMess strCtt="@a1 is equal to @a2 "/>
        </EgProof>
</EgProperty>
<EgProperty name="CircumAndCenterAngle" paraNum="10">
            ……
        <EgSurface >
                <EgMess strCtt="@ac is equal to @ao/2 "/>
        </EgSurface>
            ……
</EgProperty>
```

In the proof for the "CircumTh" property, we use the "CircumAndCentrAngle" property twice. This shows that an angle at the circumference is half of the center angle.

4.2 An Actual Use of the Property

In our software, graphical elements are created using a mouse, and after the selection of the explanation function, that is, the selection of a property, we obtain an output explanation graph.

During this step, we have several properties and definitions in a database. We therefore do not have to describe the explanations in the first property description.

5. Conclusion

We developed expression rules for explanations. Using these rules, we can construct a graph structure of graphical information. From this graph, we can obtain some information elements on demand. This may be an effective method for providing graphical contents using alternative explanations. The effectiveness of this method depends on the structure of the output explanation graphs. Evaluations and improvements of the structure are necessary for the actual use of the system.

This method also requires a large database, which will take a considerably long time to create by directly inputting the characters. We therefore need an easy creation method for such documents, which is an area of future research.

References

[1] Fukuda, R., Kojo, M., Graphic Input system in Elementary Geometry for Non-visual Communication, Proceedings of th 18th ATCM, 2013, pp.261-269

[2] Chik V., Plimmer B., Hosking J., Intelligent mind-mapping Proceedings of the 19th Australasian conference on Computer-Human Interaction: Entertaining User Interfaces ACM New York, NY, USA e2007

[3] Midmap editor: web page: http://freemind.sourceforge.net/wiki/index.php/Main__ Page

[4] Fukuda, R., Expression Rules of Directed Graphs for Non-visual Communication, Lecture Notes in Computer Science, Computers Helping People with Special Needs, 13th International Conference, ICCHP 2012,pp. 182-185