Logical and Algebraic Formulation of Origami Axioms

Fadoua Ghourabi, Tetsuo Ida, Hidekazu Takahashi, Mircea Marin and Asem Kasem
Department of Computer Science
University of Tsukuba, Tsukuba 305-8573, Japan
{ghourabi, ida, hidekazu, mmarin, kasem}@score.cs.tsukuba.ac.jp

We describe Huzita's origami axioms in logical and algebraic point of view. Observing that Huzita's axioms are statements about the existence of certain origami constructions, we can generate basic origami constructions from those axioms. We give the logical specification of Huzita's axioms as constraints among geometric objects of origami in the language of the first-order predicate logic. The logical specification is then translated into logical combinations of algebraic forms, i.e. polynomial equalities, disequalities and inequalities, and further into polynomial ideals (if inequalities are not involved). Origami construction is performed by repeated application of Huzita's axioms. By constraint solving, we obtain solutions that satisfy the logical specification of the origami construction problem. The solutions include fold lines along which origami has to be folded. The obtained solutions both in numerical and symbolic forms make origami computationally tractable for further treatment, such as visualization and automated theorem proving of the correctness of the origami construction.

1 Introduction

Computational origami is a scientific discipline to study computational aspects of origami [3, 6]. One of the foundational studies of the computational origami is the axiomatic definition of origami foldability inspired by Huzita's axioms [2]. Huzita's axioms state the foldability of origami by asserting the existence of fold lines along which we can make a fold. Huzita's axioms are constructions, as Euclid's postulates (5 out of 4) are constructions [7].

We are interested in the mathematical foundation of origami construction. We will show in this paper how Huzita's axioms are used to computerize origami constructions and to automate reasoning about origami.

From the early history of mathematics, the correspondence between geometry and algebra has been noted and exploited. It is natural to hold algebraic views of origami and relate them to geometric ones. We observe that logic is a glue to combine algebra and geometry views in computational origami. We therefore formulate origami construction in the first-order predicate logic. Then we give the origami fold operations in terms of algebraic equations by transforming the logical representation into algebraic one. The numeric solutions of these equations allow the realization of folds on computer. Moreover, the algebraic representation is given to an automated theorem prover in order to perform a proof of the correctness of the origami construction.

As part of our research efforts in computational origami, we are developing a computational origami system called *Eos* (e-origami system) [8]. It has capabilities of visualizing origami constructions based on Huzita's axioms, algebraic analysis of origami folds, and automated theorem proving of correctness of origami constructions. We use *Eos* to illustrate the result of our study in this paper.

The rest of the paper is organized as follows. In section 2, we present the six Huzita's origami axioms. In section 3, the logical specification of Huzita's axioms is detailed. In section 4, we explain the algebraic equations generated from the logic formulas. In section 5, the method for implementing Huzita's axioms by *Eos* is presented. For this, an example of constraint solving of the problem of trisecting an angle is given. In section 6, we illustrate briefly how algebraic forms of axioms are used to prove geometric properties of origami.

2 Huzita's Origami Axioms

Huzita's axioms set is described by the following statements.

(**O1**) Given two points P and Q, we can make a fold along the fold line that passes through P and Q.

(**O2**) Given two points P and Q, we can make a fold to bring P onto Q.

(**O3**) Given two lines m and n, we can make a fold to superpose m and n.

(**O4**) Given a point *P* and a line *m*, we can make a fold along the fold line that is perpendicular to *m* and passes through *P*.

(**O5**) Given two points *P* and *Q* and a line *m*, we can make a fold to superpose *P* and *m* along the fold line that passes through *Q*.

(**O6**) Given two points *P* and *Q* and two lines *m* and *n*, we can make a fold to superpose *P* and *m*, and *Q* and *n*, simultaneously.

These axioms are more powerful than the straightedge and compass method in Euclidean plane geometry [1]. For example, using Huzita's axiom set, by origami we can construct a trisector of an angle, whereas by the straightedge and compass we cannot [4].

3 Modeling Huzita's Axioms

An origami is modeled as a set of faces. A face is not completely independent from the other faces since when we fold one face, others adjacent faces may be moved. Basic elements such as points (face vertices) and lines (face edges) compose each face. When we perform an origami fold operation, new points and lines are created and others are moved. New lines are fold lines generated by folding operations. New points are intersection of faces edges and the fold line. In order to determine the geometric properties of these associated elements, we will introduce several predicates which will be described in conjunction with the logical formulation of each axiom.

We will now describe Huzita's axioms in the first-order predicate logic. In the following, let `Point` and `Line` be the set of points and lines respectively.

Axiom (O1)

In the case of (O1), the fold line k is PQ[1]. Below, the atomic formula $\texttt{OnLine}[X,\ l]$ specifies that point X is on line l.

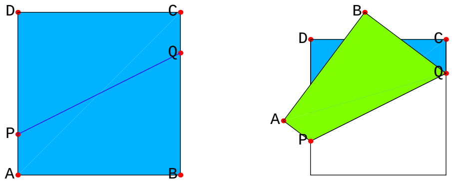


Figure 1: Axiom (O1) by *Eos* system

$$\forall\ P,\ \ Q \in \texttt{Point}\ \exists\ k \in \texttt{Line}$$
$$\texttt{OnLine}[P,\ \ k] \wedge \texttt{OnLine}[Q,\ \ k]$$

Axiom (O2)

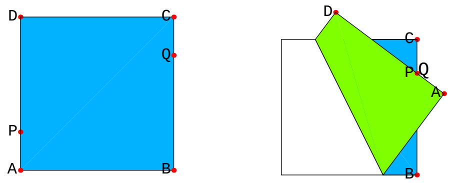


Figure 2: Axiom (O2) by *Eos* system

Given points P and Q, we need to find a fold line k such that the fold along this line brings P onto Q. The line k is simply a symmetric axis. Thus, the image of P by the fold along k is Q.

$$\forall P,\ \ Q\ \in\ \texttt{Point}\ \exists k \in \texttt{Line}$$
$$\texttt{SymmetricPoint}[P,\ \ k]\ ==\ Q$$

The term $\texttt{SymmetricPoint}[X,\ \ l]$ denotes the symmetric point of X with respect to line l. We define the equality between points and denote it by ==.

Axiom (O3)

Given two lines m and n, we need to find a fold line that brings m onto n. In other words, we need to find a fold line k such that for any point P on k the distances

[1] XY means the line that passes through point X and point Y.

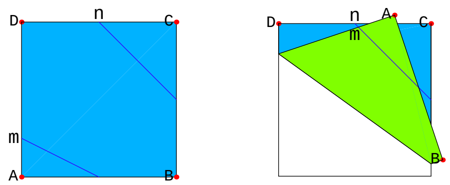


Figure 3: Axiom (O3) by *Eos* system

from P to m and from P to n are the same.

$$\forall\, m,\ n\ \in\ \texttt{Line}\ \exists\, k\ \in\ \texttt{Line}\ \forall\, P\ \in\ \texttt{Point}$$
$$\texttt{OnLine}[P,\ \ k]\ \Longrightarrow$$
$$\texttt{Distance}[P,\ \ n]\texttt{==}\texttt{Distance}[P,\ \ m]$$

In the above formula, k is the set of points P that are equidistant to m and n. Term $\texttt{Distance}[X,\ l]$ computes the distance between point X and line l.

Axiom (O4)

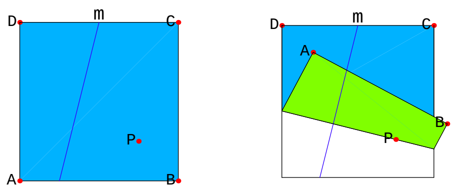


Figure 4: Axiom (O4) by *Eos* system

Given point P and line m, we need to find a fold line k passing through P and the perpendicular to m.

$$\forall\, P\ \in\ \texttt{Point}\ \forall\, m\ \in\ \texttt{Line}\ \exists\, k\ \in\ \texttt{Line}$$
$$\texttt{OnLine}[P,\ \ k] \wedge \texttt{Perpendicular}[k,\ \ m]$$

The predicate $\texttt{Perpendicular}[l,\ t]$ is true if line l is perpendicular to line t, otherwise is false.

Axiom (O5)

Given two points P and Q and a line m we can find a fold line k passing through Q such that the fold along k brings point P onto line m. Thus, Q is on line k and the

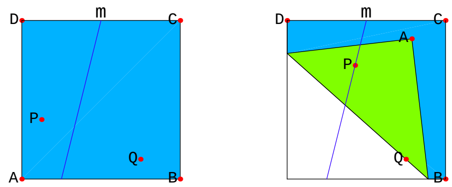


Figure 5: Axiom (O5) by *Eos* system

symmetric point of P with respect to k is on line m.

$$\forall\, P,\ \ Q\ \in\ \texttt{Point}\ \forall\, m\ \in\ \texttt{Line}\ \exists\, k\ \in\ \texttt{Line}$$
$$\texttt{OnLine}[Q,\ \ k] \wedge \texttt{OnLine}[\texttt{SymmetricPoint}[P,\ \ k],\ \ m]$$

Axiom (O6)

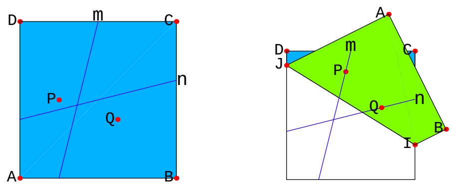


Figure 6: Axiom (O6) by *Eos* system

Given two points P and Q and two lines m and n we need to find a fold line k such that the fold along k brings P onto m and Q onto n. The symmetric points of P and Q with respect to k are respectively on lines m and n.

$$\forall\, P,\ \ Q\ \in\ \texttt{Point}\ \forall\, m,\ \ n\ \in\ \texttt{Line}\ \exists\, k\ \in\ \texttt{Line}$$
$$\texttt{OnLine}[\texttt{SymmetricPoint}[P,\ \ k],\ \ m] \wedge$$
$$\texttt{OnLine}[\texttt{SymmetricPoint}[Q,\ \ k],\ \ n]$$

4 Algebraic Interpretation

For origami construction, we have to transform the logical specifications into algebraic forms. The logical formulas in section 3 are given straightforward algebraic interpretation by the following transformation rule
Let $\mathcal{A} : \mathcal{F} \rightarrow \mathcal{R}$, where $\mathcal{F}$ be the set of logical formulas and $\mathcal{R}$ be the powerset of polynomials over real, i.e., R[x].

We apply $\mathcal{A}$ to get the algebraic meaning of formulas representing Huzita's axioms. First, we define a point by its coordinates *x* and *y*. Line is defined by the following equation $a\,x + b\,y + c = 0$, where the coefficients a and b should not be equal to 0 simultaneously. To ensure this, a suitable constraint on a and b have to be added for each line. Here, without lost of generalization we set a^2+b^2=1.

The transformation of `OnLine`$[P,\ k]$ is given by $\mathcal{A}[\![$`OnLine`$[P,\ k]]\!]$=$\{a\,x_1 + b\,y_1 + c\}$, where P is positioned at (x_1, y_1) and k is the equation $a\,x + b\,y + c = 0$.

The transformation of `Perpendicular`$[m,\ n]$ is given by
$\mathcal{A}[\![$`Perpendicular`$[m,\ n]]\!]$=$\{a_1\ a_2 + b_1\ b_2\}$, where m is specified by a_1, b_1 and c_1 and n is specified by a_2, b_2 and c_2.

In the case of axiom (O3), we deal with equality of distances: `Distance`$[P,\ m]$==`Distance`$[P,\ n]$ which is defined by the equation:

$$\frac{|a_1\ x_1 + b_1\ y_1 + c_1|}{\sqrt{a_1^2 + b_1^2}} = \frac{|a_2\ x_1 + b_2\ y_1 + c_2|}{\sqrt{a_2^2 + b_2^2}}$$

Since we set a_1^2+b_1^2=1 and a_2^2+b_2^2=1, the equation is simplified to
$(a_1\ x_1 + b_1\ y_1 + c_1)^2$=$(a_2\ x_1 + b_2\ y_1 + c_2)^2$
which is equivalent to the formula

$$((a_1\ x_1 + b_1\ y_1 + c_1) + (a_2\ x_1 + b_2\ y_1 + c_2))\times$$
$$((a_1\ x_1 + b_1\ y_1 + c_1) - (a_2\ x_1 + b_2\ y_1 + c_2)) = 0$$

Therefore, any point on the line k is given by
$a\,x_1 + b\,y_1 + c = 0$ is either on the line
$(a_1 - a_2)\ x + (b_1 - b_2)\ y + (c_1 - c_2) = 0$
or on the line
$(a_1 + a_2)\ x + (b_1 + b_2)\ y + (c_1 + c_2) = 0.$

In addition, conjunctions, disjunctions and negations of atomic formulas are used to express Huzita's axioms. We define the translation rules of those logical operators.

The conjunctions of formulas is interpreted as the union of sets of polynomials translated from each formula.

$$\mathcal{A}[\![\bigwedge_{i\in\{1,\ldots,n\}} \phi_i]\!] = \bigcup_{i\in\{1,\ldots,n\}} \mathcal{A}[\![\phi_i]\!]$$

The disjunction of formulas is interpreted as the product of the polynomials that are the elements of the cross product of the set of the polynomials, translated from each formula. Namely,

$$\mathcal{A}[\![\bigvee_{i\in\{1,\ldots,n\}} \phi_i]\!] = \{p_1 \cdots p_n \mid \langle p_1, \ldots, p_n\rangle \in \prod_{i\in\{1,\ldots,n\}} \mathcal{A}[\![\phi_i]\!]\}$$

To deal with negations, we introduce slack variable to turn disequality into equality.

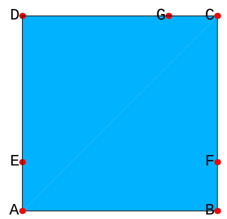
$$\mathcal{A}[\![\neg\phi]\!] = \{\prod_{p\in\mathcal{A}[\![\phi]\!]} p\,\xi_p - 1\}$$

Here, ξ_p is the slack variable introduced for each polynomial p.

5 Origami Constraint Solving

We discuss trisecting an angle as an example of origami constraint solving. Origami construction by *Eos* proceed stepwise, where each step indicates a fold operation that satisfies one of the axioms defined as geometric constraints.

In the following, we will trisect the angle $\angle FEG$.



For display purposes, first we construct the edges of $\angle FEG$ by applying axiom (O1).

Eos provides the function `Constraint` to record the geometric constraints that characterize the fold step. Here, *k* is the fold line. `ThruQ`$[E,\ G,\ k]$ is the logical constraint that the crease *k* passes through *E* and *G*.

c = `Constraint`$[k \in$ `Line`, `ThruQ`$[E,\ G,\ k]]$

The function `Constraint` gives the following formula :

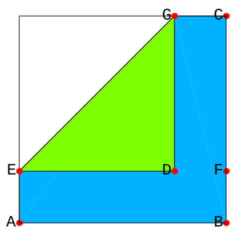
$\exists\, k\in$ `Line` `ThruQ`$[E,\ G,\ k]$

By calling `SolveConstraint`, we solve numerically the constraint generated by function `Constraint` and therefore we compute the fold line.

s = `SolveConstraint[c]`

$\{\{k \to$ `Line`$[-1.,\ 1.,\ -1.]\}\}$

Then, the fold step is visualized by `BFold`. The following call of `BFold` performs axiom (O1)
`BFold`[k/.s, $\{D\}$];



Mathematica notation k /. s denotes the result of the application of substitution s to k.

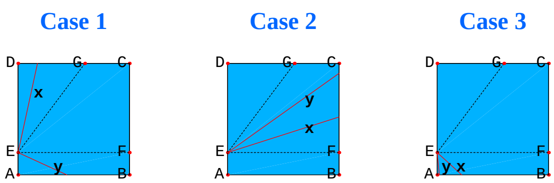
We proceed in the same way to construct the second edge *EF*.

Now, to trisect $\angle FEG$, we perform simultaneously two (O3) folds.

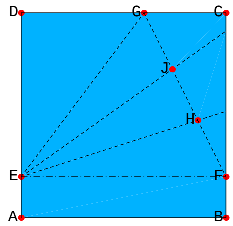
```
flines = {x,y}/.SolveConstraint[
    Constraint[{x ∈ Line,  y ∈ Line},
y==BringLineQ[EF, x]∧x==BringLineQ[EG, y]]]
```

We note that there are three numeric solutions of constraints.
{{Line[−3.73205, 1., −1.],Line[0.57735, 1., −1.]},
{Line[−0.267949, 1., −1.],Line[−0.57735, 1., −1.]},
{Line[1., 1., −1.],Line[1., 0., 0.]}}

Only the second case gives a trisection of the internal angle $\angle FEG$.



After performing the fold operations that make the second case, we obtain the following trisection of $\angle FEG$.



Eos provides also `Fold` function that implements the six Huzita's axioms.

6 Theorem Proving

Eos not only simulates origami folds, but also proves geometric properties of the construction. *Eos* keeps track of the geometrical properties of all the points and fold lines during the construction as symbolic constraints. From such constraints, polynomials are generated. They will become premises of the theorem to be proved. The first step of the proof is to collect the necessary geometrical properties in symbolic form. Then, by choosing the coordinate system, we translate the symbolic representation of the geometrical properties into polynomials. The next step is to transform the conclusion that we want to prove into algebraic form. *Eos* has an interface with *Theorema* which provides Gröbner bases method for theorem proving [5]. Premises are saved in *Theorema* format and then are sent to the prover *Theorema*. Thus, we will obtain the proof as a proof object that can be displayed in a *Mathematica* notebook.

7 Conclusion

In this paper, we formalized the computational origami construction. First, we described a formulation of Huzita's axioms into the first-order predicate logic formulas. The advantage of this formulation is abstraction by the first-order logic. Then, we explained the algebraic interpretation of this logic formulation. Based on the algebraic formulation, *Eos*, on one hand provides methods of constraint solving to achieve origami constructions, and on another hand, supports proof of a geometric properties of the constructed shapes. As future work, we would like to generalize this model to cope with the full syntax of the first-order predicate logic.

In addition, since reasoning with a huge size of constraints is a challenging task for the geometric provers, we would like to investigate optimization and simplification of polynomials and variables generated while transforming formulas into algebraic equations.

Acknowledgements

This research is supported in part by the JSPS Grants-in-Aid for Scientific Research No. 17300004 and No. 17700025, and by MEXT Grant-in-Aid for Exploratory Research No. 17650003.

References

[1] K. Hatori. K's origami - fractional library - origami construction. 2005. http://origami.ousaan.com/library/conste.html.

[2] H. Huzita. Axiomatic Development of Origami Geometry. In H. Huzita ed, editor, *Proceedings of the First International Meeting of Origami Science and Technology*, pages 143–158.

[3] Erik D.Demaine and Martin L.Demaine. Recent results in computational origami. In Thomas Hull, editor, *Origami^3: Third International Meeting of Origami Science, Mathematics and Education*, pages 3–16, Natick, Massachusetts, 2002. A K Peters, Ltd.

[4] P. L. Wantzel. Recherches sur les moyens de connaitre si un Problème de Géométrie peut se résoudre avec la règle et le compas (http://visualiseur.bnf.fr/StatutConsulter?N=magnard3-114561253&B=1&E=PDF&O=NUMM-16381). *Journal de Mathématiques Pures et Appliquees*. 1.(2):366-372

[5] B. Buchberger, C. Dupre, T. Jebelean, F. Kriftner, K. Nakagawa, D. Văsaru, and W. Windsteiger. The Theorema Project: A Progress Report. In M. Kerber and M. Kohlhase, editors, *Symbolic Computation and Automated Reasoning : The Calculemus-2000 Symposium*, pages 98–113, St Andrews, Scotland, August 6-7, 2000.

[6] T. Ida, B. Buchberger. Proving and Solving in Computational Origami. Analele Universitatii din Timisoara, XLI, Fasc. Special, 2003, Mathematica-Informatica, pages = 247–263.

[7] D. E. Joyce. Euclid's Elements. ©1996-1997, June 1997, http://aleph0.clarku.edu/ djoyce/java/elements/toc.html

[8] A. Kasem, T. Ida, H. Takahashi, M. Marin and F. Ghourabi. E-Origami System *Eos*. Proceedings of the Annual Symposium of Japan Society for Software Science and Technology, September 2006.