

Verifiable Origami Construction

Geoffrey Ramseyer (geoff.ramseyer@cs.stanford.edu)

keywords: verification, automation, provability

Abstract

Origami folds create complex geometries, and the proportions of origami models can depend heavily on the details of the initial geometries. But any formal reasoning about a fold pattern depends on the details of the pattern, and must be done manually; therefore, any change in the fold pattern often invalidates any prior formal reasoning. An automated proof system about origami folding, therefore, could simplify the process of reasoning about origami folds. In this work, I develop a system of translating origami fold sequences into nonlinear constraint systems on real-valued variables, which modern tools can solve.

For my purposes, I will consider a fold to be one of the seven single-fold Huzita-Justin axioms. Using sequences of these folds, one can construct a set of real values strictly larger than the set one can create with a compass and straightedge [Lang \(1996\)](#). This has proven invaluable for modern origami design processes, as this grants increased flexibility in the reference point construction steps that typically form the first folds of an origami model. For the practical purpose of identifying reference points, Robert Lang’s ReferenceFinder provides approximations of points that are close enough for most origami purposes [Lang \(2004-2017\)](#).

However, any reasoning about more complicated predicates, like “two lines are parallel” or “two angles are equal,” and Boolean combinations thereof, must still be carried out by hand. In practice, one can carry out this kind of reasoning for simple predicates relatively easily. However, the length of time required to prove a predicate increases as the number of folds increases. And should some step of the construction be altered, or some proportion requirement change, the analysis must often be redone. As such, automating proofs would be convenient.

A Satisfiability Modulo Theories (SMT) solver is a program that reasons about constraint systems. Given a declared set of variables and a system of constraints about those variables, such as “ $x < y$ ” and “ $3 * x + y = 5$ ” where x and y are real numbers, the solvers find assignments of values to the variables that satisfy the constraints. The most commonly used general-purpose solver is Z3, developed by Microsoft Research, but there are many other specialized solvers [z3](#). In 2013, Leonardo de Moura and Dejan Jovanović developed the Model Constructing Satisfiability calculus, a new SMT framework for reasoning about nonlinear real arithmetic [De Moura and Jovanović \(2013\)](#). This development was critical for my purposes, as it greatly accelerates automated reasoning about the cubic equations origami folds can generate.

To translate a fold sequence into constraints on real variables, I represent each line as two

points in \mathbb{R}^2 and each point as two real variables. Given a single fold specification, one can symbolically define two distinct points on the resulting fold line. Converting a sequence of folds into constraints, then, means simply composing the symbolic computations of individual folds by plugging in the appropriate real variables. In this work I reason only about single-fold axioms, but the constraint generation is modular and one could also compute symbolically the lines generated by any of the hundreds of multifold axioms as desired [Alperin and Lang \(2009\)](#).

An SMT solver is more flexible than a computer algebra system. In addition to calculating coordinates of points, it can prove that a particular constraint system is unsatisfiable. This flexibility lets the user assert geometric predicates about an origami construction. In particular, a property P must hold on a construction C if and only if an SMT solver can prove that $C \rightarrow \neg P$ is unsatisfiable. Moreover, an SMT solver can support reasoning about constructions with variable inputs. For example, one could prove whether or not a sequence of folds will trisect any angle, not just one particular angle. This capability could allow an origami designer to automatically ensure that certain proportions of the model will be correct, given some freedom for artistic variation, instead of having to recompute after each proportion adjustment.

Conveniently, when SMT solvers produce a positive result, they also produce the assignment of variables satisfying the constraints. I use this assignment to generate visualizations of the fold pattern, which helps users find potential errors in their input.

The satisfiability problem of real polynomial constraints is in NP, and as such, no solver will be able to solve every constraint system quickly. Occasionally, one solver will solve a constraint system that another cannot, so to improve the reliability of my tool, I run both of the leading solvers, Z3 and Yices2, in parallel [yic](#). This does not overcrowd the CPU, as each solver runs primarily in a single thread. As more solvers become able to reason in nonlinear real arithmetic, and as the solvers continue to develop, the reliability of the origami solving will improve.

The project is available at https://github.com/gramseyer/origami_smt

References

- Yices2. URL <http://yices.csl.sri.com/>.
- Z3. URL <https://github.com/Z3Prover/z3/wiki>.
- Roger C Alperin and Robert J Lang. One-, two-, and multi-fold origami axioms. 2009.
- Leonardo De Moura and Dejan Jovanović. A model-constructing satisfiability calculus. In *International Workshop on Verification, Model Checking, and Abstract Interpretation*, pages 1–12. Springer, 2013.
- Robert J Lang. Origami and geometric constructions. *Self Published (1996 2003)*, 1996.
- Robert J Lang. Referencefinder, 2004-2017. URL <http://www.langorigami.com/article/referencefinder>.