Report on Fold-Based 3D Modelling Prototype

Lewen Yu        Advised by Julie Dorsey and Min Kim

December 15, 2012

1 Introduction

As described in the original proposal, we wished to explore a new geometric modelling paradigm inspired by origami. In this paradigm, the modeller
begins with a flat plane as the primitive, and constructs the model by applying a series of folding transformations, such as mountain and valley folds, reverse folds, and compound folds. An advantage of fold-based modelling over some other 3D modelling paradigms is its extremely accessible physical counterpart. Today, there are many hobbyists who engage in the craft of paper-folding, along with a thriving community of origami enthusiasts and academic researchers.

The prototype is implemented on iPad, over the MentalCanvas codebase maintained by the Yale Graphics Lab. Though early works on origami simulators exist [?], given today’s ubiquity of touch-based interfaces and the expressiveness of multitouch gestures, the iPad is thought to be an ideal target for a modern take on fold-based modelling. This report details my efforts this semester to implement the prototype.

2 Implementation

Keywords

- Canvas: a plane that the user can fold and on which the user can draw.

- Crease: a line projected onto the plane of some canvas. It serves as the edge that also separates two sides of the folded canvas.

- Pinch, Pan, Tap: iOS gestures.

Modifications to mountain and valley folds

The MentalCanvas system implements folding through two steps: first, specifying a crease on some canvas with the pinch gesture; and second, specifying the side and degree of the fold with the drag gesture (the location of the initial press of the drag lands determines which side is folded). In particular, only the x-component of the pan is taken into account when performing the fold. If the canvas kept track of the crease, it would be possible to use the line orthogonal to the crease to gauge the degree of the fold. The Crease class was created for this reason. Given the original crease (which is simply specified by two screen points defining the line), we can project the vector of
the drag gesture onto this line orthogonal to the crease and acquire a more accurate value for the fold operation.

Unlike physical origami, a virtual crease need not constrain itself to a single direction of fold. A virtual crease can be folded in either direction, making mountain and valley folds virtually synonymous.

**Pinching to fold**

The current implementation allows a single side of the canvas to be folded at any given time. It might be useful to allow both sides to be folded simultaneously. Thus, I mapped the pinch gesture to this function. To do this, I added an additional field to the Canvas class to point to its "sibling" canvas. As the user begins folding, the canvas is split into two (the original canvas is deleted), the sibling fields updated, and rotation transforms corresponding to the fold are applied to folding canvases. Further work should be done to keep track of the relationship between various canvases in a complex model.

**Self-intersection and recursive folding**

Self-intersection detection prevents different sides of a folded canvas to travel through each other during folding. Such a feature is important for emulating physical origami. Some bugs remain in the current implementation of this feature. The algorithm to detect intersections between canvases in Mental-Canvas is as follows\(^1\):

```java
boolean intersects (Canvas other):
   let (other_front_pts, other_back_pts) be points of other split along the plane of this Canvas;
   for each point p in other_front_pts:
      if p is on the plane of this Canvas:
         if p’s local coordinates is within outline of this Canvas:
            return true
   return false
```

\(^1\)algorithm derived by Patrick Paczkowski
Additionally, in a functional origami simulator, it is necessary to be able
to fold a canvas recursively in groups. Theoretically, this is as simple as
projecting the crease onto all visible canvases and applying the same fold
operation to every single one of them (provided they are flat). However,
due to the way MentalCanvas handles folding (it deletes and creates a new
copy of the canvas when applying transforms), this technique is difficult to
implement without some way to keep track of and continuously update all
live canvases.

In the next semester, these two features will be my initial focus.

3 Future Work

Further augmentation to data structure

Origami lends itself to tree-based data structures. The follow modifications
should enable simpler implementations of folding operations.

Ideally, a canvas should consist of a binary tree of canvases. Assuming
each fold creates exactly two canvases, each “sub-canvas” should become the
children of the original canvas (which is no longer rendered, but kept for the
undo operation). Each canvas should have access to its creases. By default,
a canvas already has functionality for painting and texture-mapping, but
further adjustments may be necessary if a textured canvas becomes folded.

Creases are also represented by a vector of binary trees. A single fold op-
eration can define new creases and subdivide preexisting creases. Subdivided
creases become children of the crease before the fold, and any new creases
become the root of a new binary tree, which is then appended to the end
of the vector. A canvas has access to this vector. Each crease should have
access to adjacent creases and points. Future plans include allowing recur-
sive folding on top of flatly-folded sides, reverse folds, and modular models
through virtual glue.

This data structure is heavily based on the efforts of Miyazaki [?].

Reverse-folding

Though reverse folds (See Figure 3) is considered a intermediate fold in
origami, it is easier to achieve in virtual space because creases are well-
declared and need not conform to the physical constraints of real paper. To
perform a reverse fold, the system should simply allow the user to specify the new location of an overlapping point on two sibling canvases, and then ensure that the resulting fold (the crease of which can be inferred from the location of the new point) is a reverse fold where both canvases are folded symmetrically in opposite directions toward the inside face of the canvases.

**Gluing of models/modular origami**

Origami models can be created through combining modular components. In particular, early Chinese paper folding and a number of contemporary
origami styles make extensive use of this technique. Resulting models are usually distinctively more three-dimensional than non-modular origami. Simple modular origami (with no need for glue) is trivial in our system since there is no implementation of collision-detection. However, for origamic architecture and paper cutouts that require glue, it becomes necessary to implement virtual glue. To accomplish this within our system, we need to augment the Canvas class with a list of all artificially connected canvases and enable groupings of canvases. In the base MentalCanvas system, there exists a CanvasGroup class. It may be necessary to slightly augment this class to fully implement modular origami.

References