

FUJIMOTO, NUMBER THEORY AND A NEW FOLDING TECHNIQUE?

TAMARA B. VEENSTRA

The standard method to fold a piece of paper into 5ths, 7ths, or generally, n ths was developed independently by Fujimoto [2] and Brunton [1] in the 1970's. This method, usually referred to as the Fujimoto approximation technique, is a recursive process that finds a better and better approximation to $1/n$ th of the paper eventually reaching this as precisely as the dynamics of folding paper will allow. During this process, numerous crease marks are made along the paper. In addition to eventually finding $1/n$ th of the paper sometimes these crease lines will also mark all multiples of $1/n$ and sometimes they will mark only some multiples. This paper discusses results that classify which n completely divide the paper into n ths in this manner from two different viewpoints: one using modular arithmetic and another using binary representations. As a consequence there are some interesting connections between these two methods of classifying n which leads to a generalization of the Fujimoto technique providing an alternate way to find $1/n$ th of the paper.

The general algorithm for the Fujimoto approximation technique for any odd n works by first placing a pinch mark at a rough approximation for $1/n$ th of the paper. A sequence of folds is then made recursively using the current pinch mark to locate the next pinch mark. More precisely, at any stage in the process the location of the crease line can be viewed as a fraction of the paper, either from the left hand side or the right hand side. Since n is odd, exactly one of these fractions will have an even numerator. To get the next crease line, fold in half from the current crease line to the edge of the paper corresponding to the even numerator. Eventually, there will be a pinch mark which provides a new, more accurate approximation for $\frac{1}{n}$ th of the paper since the error is reduced by half every time the paper is folded in half. One can repeat the process until there is no noticeable difference in your pinch marks for $1/n$ so that $1/n$ th is found as accurately as possible. At this

To appear in Origami⁴: Proceedings of the Fourth International Meeting of Origami Science, Mathematics, and Education, AK Peters, Natick, MA. Do not photocopy without permission.

time, the pinch marks are extended to crease lines all the way through the paper.

For example, if to fold a piece of paper into 5ths, the first pinch mark guesses $\frac{1}{5}$ from the left and $\frac{4}{5}$ from the right. Since the numerator of $\frac{4}{5}$ is even, we fold halfway from the right hand side to the 1st pinch mark. This gives us a new pinch mark which is $\frac{3}{5}$ from the left and $\frac{2}{5}$ from the right. Again, the numerator on the right hand side is even, so we fold that side in half. We proceed similarly until we produce a new pinch mark for $\frac{1}{5}$ from the left. We see that by the time we return to the pinch mark for $\frac{1}{5}$ from the left we have made pinch marks at $\frac{1}{5}$, $\frac{3}{5}$, $\frac{4}{5}$, and $\frac{2}{5}$ where these represent, in order, the pinch marks made as a fraction of the paper from the left side. Thus, for this value of n we have produced pinch marks at all multiples of $\frac{1}{5}$. Thus, after we have found $\frac{1}{5}$ th accurately and made all the crease lines we will have completely divided the paper into 5ths. This property does not hold for all n . For example, if we apply the Fujimoto approximation technique to $n = 7$ then there are only 3 crease lines produced at $\frac{1}{7}$, $\frac{2}{7}$, and $\frac{4}{7}$ and the paper is not completely divided into 7ths. To study this property we introduce the following definition.

Definition 0.1. For odd n , n has a complete Fujimoto division if the Fujimoto algorithm for finding $1/n$ th of the paper also produces crease lines at all multiples of $1/n$.

There are many examples of n 's which have a complete Fujimoto division and many examples of n 's which do not. For example, $n = 3, 5, 11, 13$ and 19 all have a complete Fujimoto division while $n = 7, 9, 15, 17, 21$ and 23 do not. In the next section we summarize results which answer the question about which n have a complete Fujimoto division by finding formulas to compute the number of crease lines for any n in two different ways.

1. THE NUMBER OF CREASE LINES

Since we are only interested in where the crease lines occur, we will assume that we have already found $1/n$ th exactly and are just going through the algorithm the last time to make the creases all the way down the paper. To keep track of the location of these crease lines we introduce the following notation.

Definition 1.1. Let $l_k(n)$ denote the fraction of the paper that is to the left of the pinch mark at the k th step in the Fujimoto approximation for finding $\frac{1}{n}$ th of the paper.

We will always assume that we start with a pinch mark $1/n$ th from the left so $l_1(n) = \frac{1}{n}$. In most instances we will simply write l_k unless it is necessary to specify the n .

To understand where the crease lines fall we first construct algebraic formulas for the l_k 's. Because the instructions for Fujimoto use the current crease line, l_k , to determine the placement of the next crease line, l_{k+1} , we end up with a recursive function. There are two cases depending on whether we are folding the left or the right side in half. Note that with our notation the left hand side of the paper corresponds to 0 and the right hand side of the paper correspond to 1. If we are folding the left side of the paper in half we have $l_{k+1} = \frac{l_k}{2}$ and folding the right side in half gives $l_{k+1} = 1 - \frac{1-l_k}{2} = \frac{l_k+1}{2}$ as in Figure 1.

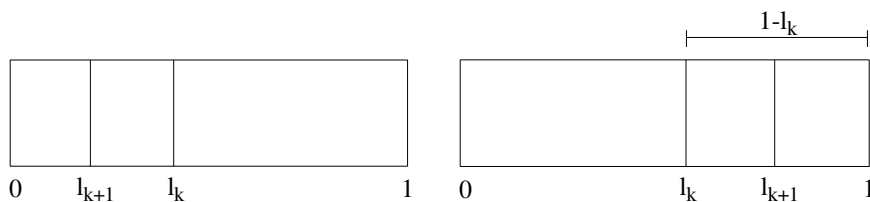


FIGURE 1. Finding l_{k+1} in terms of l_k

We summarize in the the equation below. For l_k as in Definition 1.1 we have $l_1 = \frac{1}{n}$ and

$$(1.1) \quad l_{k+1} = \frac{1}{2}(l_k + c_k) \text{ where } c_k = \begin{cases} 0 & \text{if } nl_k \text{ is even} \\ 1 & \text{if } nl_k \text{ is odd} \end{cases} .$$

To verify that nl_k really is the numerator of l_k we state some facts which are proved in a generalized case in Corollary 2.4. First, $l_k = \frac{a}{n}$ for some integer a with $1 \leq a \leq n - 1$. Second, $\gcd(a, n) = 1$ so the fraction $\frac{a}{n}$ cannot be reduced. Thus, this equation does correspond to the directions for the Fujimoto approximation technique.

These facts already gives us some insight into which n can have a complete Fujimoto division. For example, if $n = 9$ then $l_k \neq \frac{3}{9}$ for any k since $\gcd(3, 9) = 3$. This means there will be no crease mark at $\frac{3}{9}$, so that 9 does not have a complete Fujimoto division. Thus, if n is a composite number it will not have a complete Fujimoto division. The converse is not true, as many prime n , such as $n = 7$, also do not have a complete Fujimoto division. To answer the question about which primes have a complete Fujimoto division we state the following theorem from [6].

Theorem 1.2. *Let n be odd. The number of crease marks produced in the Fujimoto approximation technique is the (multiplicative) order of 2 mod n . Moreover, n has a complete Fujimoto division if and only if n is a prime where $n \equiv \pm 3 \pmod{8}$ and $|2| = n - 1$.*

From this theorem we see that it is relatively rare for n to have a complete Fujimoto division. We next describe a different way to find a formula for the number of crease lines produced in the Fujimoto technique. This uses the binary representation of $\frac{1}{n}$ and is based on independent work by Robert Lang in [4] and [5] and James Brunton in [1]. Their ideas can be stated as the following theorem.

Theorem 1.3. *Let n be odd. The Fujimoto approximation technique for $\frac{1}{n}$ produces r crease lines if and only if $\frac{1}{n} = (\overline{a_1 \cdots a_r})_2$. Moreover, n has a complete Fujimoto division iff $r = n - 1$.*

As an example of applying this theorem we examine the binary representations for $\frac{1}{5} = (\overline{0011})_2$ and $\frac{1}{7} = (\overline{001})_2$. We see that Theorem 1.3 confirms that the Fujimoto algorithm produces 4 crease lines for $n = 5$ and only 3 crease lines for $n = 7$.

The idea underlying the proof of this theorem is that $a_i = 0$ or 1 corresponds to folding the left or right hand side (respectively) in half. However, it does not refer to what happens at the i th step as the correspondence uses the coefficients in reverse order. To clarify how this works (and to help us generalize later) we use the Fujimoto approximation technique for $n = 7$ to construct the binary representation for $\frac{1}{7}$. Applying Equation 1.1 converts the folding algorithm into the algebraic algorithm with $l_1 = \frac{1}{7}$, $l_2 = \frac{1}{2}(l_1 + 1)$, $l_3 = \frac{1}{2}(l_2)$, and $l_4 = \frac{1}{2}(l_3)$. To produce a binary expansion we start with l_4 and recursively substitute in the above formulas.

$$l_4 = \frac{1}{2}(l_3) = \frac{1}{2} \left(\frac{1}{2}(l_2) \right) = \frac{1}{2^2}(l_2) = \frac{1}{2^2} \left(\frac{1}{2}(l_1 + 1) \right) = \frac{l_1}{2^3} + \frac{1}{2^3}.$$

If we plug in $l_1 = \frac{1}{7}$ we see that $l_4 = \frac{1}{7}$. This is where the Fujimoto algorithm starts repeating since it only produces 3 distinct crease lines for $n = 7$. Plugging in $1/7$ for both l_1 and l_4 in the above equation and then solving for it yields

$$\frac{1}{7} = \frac{1}{2^3} \left(\frac{1}{1 - \frac{1}{2^3}} \right) = \frac{1}{2^3} \left(1 + \frac{1}{2^3} + \frac{1}{2^6} + \cdots \right) = (\overline{001})_2!$$

Because we substituted the l_k 's in reverse we see why the coefficients of the binary expansion determine the folding in reverse order. We will prove this always happens for a generalized case in the next section.

2. CONSEQUENCES AND GENERALIZATIONS

We now have two ways to determine the number of crease lines produced in the Fujimoto approximation technique. Thus, we must have the following connection.

Theorem 2.1. *Let n be odd. The fraction $\frac{1}{n}$ has a binary expansion $\frac{1}{n} = (\overline{.a_1a_2\cdots a_r})_2$ if and only if r is the order of 2 mod n .*

We know this theorem is true by combining Theorems 1.3 and 1.2, hence we can just say proof by origami! While Theorem 2.1 emerged naturally out of connections to origami, it does not actually depend on using a base 2 representation; that is, we have the following well-known generalization: (See, for example, [3, p.112]).

Theorem 2.2. *The fraction $\frac{m}{n}$ has a base b representation as $\frac{m}{n} = (\overline{.a_1a_2\cdots a_r})_b$ if and only if the $\gcd(n, b) = 1$ and r equals the order of b mod n .*

A natural question arising from this generalization is whether expressing $\frac{1}{n}$ in other bases corresponds to an alternate folding technique for finding $\frac{1}{n}$ th of the paper. The theorem forces $\gcd(n, b) = 1$ but with that condition we can, in fact, use a base b representation to produce an interesting folding algorithm albeit with some added complications.

For motivation on how to turn a base b representation for $\frac{1}{n}$ into a folding algorithm we recall the steps for the binary case. We first translated the folding algorithm into an algebraic algorithm with the l_k 's. Then we used these recursive formulas to construct the binary representation. For the base b case, we will go through these steps in the opposite order. That is, we will use the base b representation to construct an algebraic algorithm; then we will translate the algebraic algorithm into a folding algorithm.

To convert the base b representation into a recursive algebraic algorithm we have the following theorem.

Theorem 2.3. *Let l_k be a recursively defined set of formulas with $l_{k+1} = \frac{1}{b}(l_k + c_k)$ where the c_k 's are integers such that $0 \leq c_k < b$. Then $l_{r+1} = l_1$ if and only if $l_1 = (\overline{.c_r\cdots c_1})_b$.*

Proof. We first assume that $l_{r+1} = l_1$. Applying the definition of the l_k 's recursively we have:

$$\begin{aligned}
l_{r+1} &= \frac{1}{b} (l_r + c_r) = \frac{1}{b} l_r + \frac{c_r}{b} \\
&= \frac{1}{b} \left(\frac{1}{b} (l_{r-1} + c_{r-1}) \right) + \frac{c_r}{b} = \frac{1}{b^2} l_{r-1} + \frac{c_{r-1}}{b^2} + \frac{c_r}{b} \\
&= \quad \vdots \\
&= \frac{l_k}{b^{r-k+1}} + \frac{c_k}{b^{r-k+1}} + \cdots + \frac{c_r}{b} \\
&= \quad \vdots \\
&= \frac{1}{b^r} l_1 + \frac{c_1}{b^r} + \cdots + \frac{c_{r-1}}{b^2} + \frac{c_r}{b}.
\end{aligned}$$

Now substituting $l_{r+1} = l_1$ and solving for l_1 yields

$$l_1 = \left(\frac{c_1}{b^r} + \cdots + \frac{c_r}{b} \right) \left(\frac{1}{1 - \frac{1}{b^r}} \right)$$

To convert l_1 into an infinite repeating base b representation we recognize $\left(\frac{1}{1 - \frac{1}{b^r}} \right)$ as the sum of a geometric series. Thus, we have

$$\begin{aligned}
l_1 &= \left(\frac{c_1}{b^r} + \cdots + \frac{c_r}{b} \right) \left(1 + \frac{1}{b^r} + \frac{1}{b^{2r}} + \cdots \right) \\
&= \left(\frac{c_r}{b} + \cdots + \frac{c_1}{b^r} \right) + \left(\frac{c_r}{b^{r+1}} + \cdots + \frac{c_1}{b^{2r}} \right) + \cdots = (\overline{.c_r \cdots c_1})_b.
\end{aligned}$$

For the other direction, essentially we work through all the above steps in reverse. \square

To see how Theorem 2.3 translates a ternary representation into an algebraic algorithm we examine $\frac{1}{5} = (\overline{.0121})_3$. We first define $l_1 = (\overline{.0121})_3 = \frac{1}{5}$. Now, since the length of the repeated pattern is 4 the formulas will start repeating with $l_5 = l_1$. Applying the coefficients of $(\overline{.0121})_3$ in the reverse order we have

$$\begin{aligned}
l_2 &= \frac{1}{3} (l_1 + 1) = \frac{1}{3} \left(\frac{1}{5} + 1 \right) = \frac{2}{5} \\
l_3 &= \frac{1}{3} (l_2 + 2) = \frac{1}{3} \left(\frac{2}{5} + 2 \right) = \frac{4}{5} \\
l_4 &= \frac{1}{3} (l_3 + 1) = \left(\frac{4}{5} + 1 \right) = \frac{3}{5} \\
l_5 &= \frac{1}{3} (l_4 + 0) = \frac{1}{5}
\end{aligned}$$

From this example we see where the crease lines should fall in a Fujimoto-like algorithm, but it is not at all obvious how to construct a folding method for obtaining these crease lines in general. We do notice that all the crease lines are multiples of $\frac{1}{n}$. This will always be true as we see in the following corollary.

Corollary 2.4. *Let $\gcd(n, b) = 1$, $l_1 = (\overline{.c_r \cdots c_1})_b$ and $l_{k+1} = \frac{1}{b}(l_k + c_k)$. If $l_1 = \frac{1}{n}$ then for all k , $l_k = \frac{L_k}{n}$ where L_k is an integer such that $0 \leq L_k < n$ and $\gcd(L_k, n) = 1$. Moreover, $L_k \equiv (b^{-1})^{k-1} \pmod{n}$ so it can be completely specified in terms of powers of $b \pmod{n}$.*

Proof. Let the l_k 's be defined as above with $l_1 = \frac{1}{n}$. By Theorem 2.3, $l_{r+1} = l_1 = \frac{1}{n}$. Combining this with an intermediate step in the proof of Theorem 2.3 for $k \leq r$ we have

$$\frac{1}{n} = l_{r+1} = \frac{l_k}{b^{r-k+1}} + \frac{c_k}{b^{r-k+1}} + \cdots + \frac{c_r}{b}.$$

Multiplying by nb^{r-k+1} and solving for nl_k gives

$$nl_k = b^{r-k+1} - n(c_k + \cdots + b^{r-k}c_r).$$

All terms except l_k are integers, so nl_k is an integer. Thus, we let $L_k = nl_k$ and prove the remaining properties of L_k . Reducing mod n yields $L_k \equiv b^{r-k+1} \equiv (b^{-1})^{k-1} \pmod{n}$ since r is the order of $b \pmod{n}$ by Theorem 2.2.

To see that $0 \leq L_k < n$, let $M_k = \frac{L_k}{nb^{r-k+1}} = \frac{1}{n} - \left(\frac{c_r}{b} + \cdots + \frac{c_k}{b^{r-k+1}}\right)$. Thus, $M_k = (\overline{.c_r \cdots c_1})_b - \left(\frac{c_r}{b} + \cdots + \frac{c_k}{b^{r-k+1}}\right) = \frac{c_{k-1}}{b^{r-k+2}} + \frac{c_{k-2}}{b^{r-k+3}} + \cdots$. Clearly $M_k \geq 0$. By construction of binary coefficients we must have $M_k < \frac{1}{b^{r-k+1}}$. Thus $0 \leq M_k < \frac{1}{b^{r-k+1}}$ and $0 \leq L_k < n$.

To show that $\gcd(L_k, n) = 1$ we use induction. Since $l_1 = \frac{1}{n}$ we have $L_1 = 1$ so clearly $\gcd(L_1, n) = 1$. We next assume that $\gcd(L_k, n) = 1$. From equation 1.1 we have $L_{k+1} = L_k + nc_k$. From properties of gcd's, $\gcd(L_k + nc_k, n) = \gcd(L_k, n) = 1$. Thus by induction, $\gcd(L_k, n) = 1$ for all k . \square

Now that we have the base b representation for $\frac{1}{n}$ translated into an algebraic equation we are ready to turn this into a folding algorithm. There are two parts to this process: we first need to understand what the formula $l_{k+1} = \frac{1}{b}(l_k + c_k)$ corresponds to in terms of a folding action. Then we need to specify the folding algorithm independently from the base b representation for $\frac{1}{n}$.

We first examine how $l_{k+1} = \frac{1}{b}(l_k + c_k)$ corresponds to a folding action. Multiplying by $1/b$ corresponds to folding $1/b$ th of the way between two points, just like multiplying by $1/2$ corresponded to folding halfway between two points in the binary case. Thus, we already see

one difficulty of this method as, in general, folding into b ths already requires some work. We must also figure out which two points to fold between. As in the binary case, one of the points is always the current crease line l_k . The other is determined by the coefficient c_k . In binary the c_k 's take only two values, 0 or 1, and these correspond to points at the left and right ends of the paper. For base b we will need b different points since there are b different values for the c_k .

For an example, consider $b = 3$. The formulas for the l_k 's are of the form $l_{k+1} = \frac{1}{3}(l_k + c_k)$ with $c_k = 0, 1$ or 2 . Two of these values for c_k correspond to straightforward generalizations of the binary case where we folded the left side or the right side in half. If $c_k = 0$ then folding $l_{k+1} = l_k/3$ corresponds to folding $1/3$ of the way from the left edge of the paper (denoted as 0) to the crease line l_k . If $l_{k+1} = \frac{1}{3}(l_k + 2) = 1 - \frac{1-l_k}{3}$ then we fold $1/3$ of the way from the right edge of the paper (denoted as 1) to the crease line l_k .

The case where $c_k = 1$ is not at all similar to the binary case. The formula $l_{k+1} = \frac{1}{3}(l_k + 1)$ still corresponds to folding $1/3$ of the way from a point A to l_k , but A does not fall at either end of the paper. To see what the point A corresponds to we examine the action of folding $1/3$ of the way from an arbitrary point A to the current crease line l_k as in Figure 2.

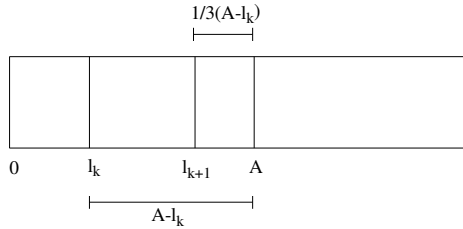


FIGURE 2. Folding the second crease line

This folding corresponds to $l_{k+1} = A - \frac{1}{3}(A - l_k)$. While l_k need not be to the left of A , the formula will be the same. To find A in the case where $l_{k+1} = \frac{1}{3}(l_k + c_k)$ we solve $\frac{1}{3}(l_k + 1) = A - \frac{1}{3}(A - l_k)$ yielding $A = 1/2$. Alternatively, one can think of $A = 1/2$ as the fixed point of the transformation $T(x) = \frac{1}{3}(x + 1)$. Thus this case represents folding $1/3$ of the way from the point at $1/2$ of the paper to the crease line, l_k .

For an arbitrary base b , if $l_{k+1} = \frac{1}{b}(l_k + c_k) = \frac{c_k}{b-1} - \frac{1}{b}(\frac{c_k}{b-1} - l_k)$ the folding algorithm is to fold $\frac{1}{b}$ th of the way from $A_k = \frac{c_k}{b-1}$ to l_k . As a consequence, using this method requires being able to fold $\frac{1}{b}$ and $\frac{1}{b-1}$.

Our last step is to construct a folding algorithm that does not require the base b representation of $\frac{1}{n}$ to determine the c_k 's. Recall that the instructions for the Fujimoto approximation technique specify which way to fold based on the parity of nl_k , or equivalently, $c_k \equiv nl_k \pmod{2}$. In the general case, we're hoping for a similar condition on the numerator $L_k = nl_k$, presumably mod b , to specify the folding action. From Equation 1.1 we have $bL_{k+1} = L_k + nc_k$. All terms in this equation are integers, so we may reduce mod b to obtain $L_k \equiv -nc_k \pmod{b}$ and $c_k \equiv (-n)^{-1}L_k \pmod{b}$ since $\gcd(n, b) = 1$. Moreover, since $0 \leq c_k < b$, c_k is exactly equal to $(-n)^{-1}L_k$ reduced mod b .

Thus, if $\gcd(n, b) = 1$, the generalized algorithm for folding $\frac{1}{n}$ th of the paper using a base b technique is as follows:

- (1) Guess where $\frac{1}{n}$ th of the paper is and make a pinch mark there.
- (2) Find the fold point by computing $c_k \equiv (-n)^{-1}L_k \pmod{b}$ and mark $A_k = \frac{c_k}{b-1}$.
- (3) Fold $\frac{1}{b}$ th of the way from A_k to l_k .
- (4) Repeat steps 2 and 3 until return to $l_k = \frac{1}{n}$.
- (5) Repeat steps 2-4 until there is no noticeable difference in the approximation for $\frac{1}{n}$.

Note that Step 2 and Step 3 may require an additional folding algorithm such as the Fujimoto approximation technique. We illustrate with $\frac{1}{5}$ to clarify. Since $(-n)^{-1} \pmod{b} = (-5)^{-1} \pmod{3} = 1$, we have $c_k \equiv L_k \pmod{b}$. We first, as in the binary case, make a guess pinch mark for $\frac{1}{5} = l_1$. In Table 2 we show the necessary calculations to compute the fold points for the rest of the algorithm. To get the sec-

k	$L_k = nl_k$	$c_k = L_k \pmod{3}$	$A_k = \frac{c_k}{2}$	$l_{k+1} = \frac{1}{b}(l_k + c_k)$
1	1	$1 \pmod{3} = 1$	$1/2$	$\frac{2}{5}$
2	2	$2 \pmod{3} = 2$	$2/2=1=\text{RHS}$	$\frac{4}{5}$
3	4	$4 \pmod{3} = 1$	$1/2$	$\frac{3}{5}$
4	3	$3 \pmod{3} = 0$	$0/2=0=\text{LHS}$	$\frac{1}{5}$

TABLE 1. Finding the fold points

ond crease we fold $1/3$ of the way from a point at $1/2$ of the paper to $\frac{1}{5}$. This produces a crease line at $l_2 = \frac{2}{5}$. To form the third crease line we fold $1/3$ of the way between the right side of the paper and l_2 since $A_2 = 1$. For the 4th crease line we fold one-third of the way from $A_3 = \frac{1}{2}$ to the third crease line l_3 . For the 5th crease line we fold $1/3$ of the way from the left side of the paper ($A_4 = 0$) to l_4 . At this point we will have a new more accurate pinch mark for $\frac{1}{n} = l_1$, and the

process can be repeated as necessary. The error decreases even faster than in the Fujimoto approximation technique since, assuming we can fold into b ths accurately, the error will decrease by a factor of $\frac{1}{b}$ each time we fold into b ths.

3. CONCLUSION

It is quite a bit easier to fully divide a piece of paper into n ths when the Fujimoto approximation technique produces crease lines at all multiples of n . Many origami fans have probably wondered why this does or doesn't work for their favorite n . We now have several different viewpoints from which to examine n to see if there will be a complete Fujimoto division. The modular arithmetic method may be an easier way for many people to understand how this algorithm works mathematically and to determine the number of crease lines that will be produced for any odd n . The connection between this new modular arithmetic analysis and previous binary analysis led to a generalized Fujimoto technique. There are cases where this new technique will work better for completely dividing a paper into n ths. For example, if $n = 7$ the standard Fujimoto technique did not produce crease lines at all multiples of n . However, if we use the base 3 method instead it will produce crease lines at all multiples of n since powers of 3 generate all elements mod 7. It requires adding a pinch mark at $1/2$ of the paper, folding in thirds instead of halves, and performing some mod 3 calculations along the way, but it is still possible this may be a useful folding algorithm in some cases. At the very least the mathematical algorithm is quite intriguing and provides an interesting application to expressing numbers in alternate bases.

REFERENCES

- [1] J. Brunton, "Mathematical exercises in paper folding: I", *Mathematics in School*, vol. 2, no. 4, July 1973, pp. 25-26.
- [2] S. Fujimoto and M. Nishiwaki, *Sojo Suru Origami Asobi Eno Shotai (Invitation to Creative Origami Playing)*, Asahi Culture Center, 1982.
- [3] G. H. Hardy and E. M. Wright, *The Theory of Numbers*, 3rd Ed., Oxford University Press, (1956).
- [4] R. J. Lang, "Four Problems III", *British Origami*, no. 132, pp 7-11, Oct. 1988.
- [5] R. J. Lang, "Origami Approximate Geometric Constructions," in *Tribute to a Mathematician*, Barry Cipra, Erik D. Demaine, Martin L. Demaine, and Tom Rodgers, eds., A K Peters Ltd., 2005.
- [6] T. Veenstra, "Can origami compute the order of elements mod n ?", in submission.