A Computational Approach to String Figures

Yulong Liu

December 2023

Introduction

String figures are intricate designs made from a single loop of string. People have been playing with the string for entertainment, storytelling, and cultural preservation around the world for millennia. Despite being merely a trivial knot from the view of knot theory, string figures are intriguing objects to study mathematically nonetheless.

The first step of studying string figures is to develop a precise language to represent them. With such representations, we can build algorithms to compute the common movements that we see when people make string figures. This computational approach allows not only mathematicians to explore the endless possibilities of string figure constructions without having to physically play the string, but also programmers to build databases to store the enormous string figures corpora and implement computer simulations to visualize the string figure constructions.

Before diving into the study of string figures, let us play around with the string first, So grab a loop of string, about arm length, and we will get started by making some string figures!

Playing with the String

Before holding the string, let us name our fingers first. Using a numbering system similar to piano fingerings, we will name our left hand fingers, from thumb to pinky, as $L1$ to $L5$; and on the right hand we have $R1$ to $R5$ respectively. String figures are usually held by the two hands with the palms facing each other. To illustrate, Figure 1 is a top-down view of the hand positions, called normal position ([Sto99] p. 2).

For instance, we can put a loop of string around $L1$, $L5$, and $R2$ and create a simple string figure as per Figure 2.

Given a simple string figure like the one in Figure 2, we will refer to different parts of the string as string segments. For each part of the string that goes around a finger F, we will use Fn to denote the near segment and Ff for the far segment. In the special case where we have a segment between $L1$ and $L5$ that touches our palm, we call it the left palmar string segment, denoted as Lp. Similarly, Rp will refer to the right palmar string segment which is not part of our current string figure.

Now we will do some string figure moves. Starting with the figure above, we will use $R5$, the right pinky, to pick Lp, the left palmar string, by passing $R5$ over $R2f$. To pick $R2f$ with $R5$, we insert $R5$ between the left palm and Lp

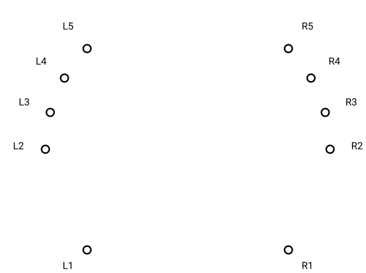

Figure 1: Top-down view of the normal position

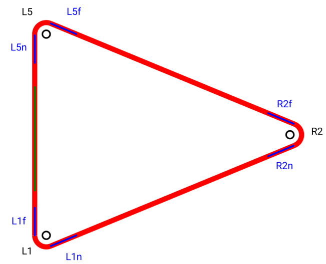

Figure 2: A simple string figure

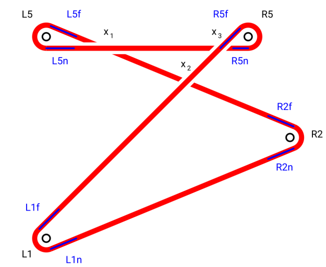

Figure 3: A more complicated string figure

from above, and put our hands back to the normal position while letting $R5$ to hold the string. The result should look like Figure 3.

Notice we now have three crossings on our string figures. We will name them x_1, x_2, x_3 as per Figure 3. Finally, we will pick $R5n$ with $R1$ by passing over the intermediate segments $R2n$ and $R2f$ and inserting $R1$ between $R5f$ and $R5n$ from below. After resetting our hands to normal position by pulling $R5n$ towards us, we get a star like Figure 4!

Playing with the string is intuitive, and the diagrams help us to understand how the moves transform from one string figure to another. In this section, we seek to develop a method that can describe exactly the behavior of these moves to an arbitrary string figure. Being able to describe precisely the transformation from one string figure to another allows us to develop algorithms for computers to do the calculations on a large scale. However, before teaching computers how to play string figures, we need to first represent the string figures in a machine-readable way.

Representing String Figures

String diagrams are visual and very intuitive, but they become unwieldy when we get into much more complex string figures. Instead, we describe a string figure as a sequence of fingers and crossings which we will refer to as linear sequences ([Sto99] p. 6). A linear sequence of a string figure starts with the left finger that is closest to the player. Then, we continue noting down the fingers and crossings by traversing along the string clockwise. For instance, we write the linear sequence for the string figure in Figure 2 as:

$$L1 : L5 : R2$$

For string figures in Figure 3 and 4, we record the crossings with their parity. That is, if we traverse over a crossing x_1, we put $x_i(o)$; likewise, when we traverse under x_i, we put $x_i(u)$. For instance, the linear sequence for

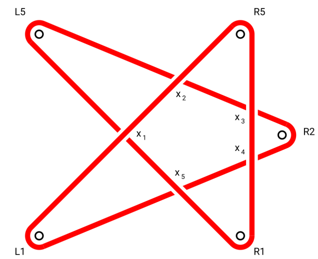

Figure 4: The Star

Figure 3 is denoted as following:

$$L1 : x_2(o) : x_3(o) : R5 : x_3(u) : x_1(o) : L5 : x_1(u) : x_2(u) : R2$$

The notation for linear sequences is similar to the Dowker notation for mathematical knots ([Ada94] p35) with the addition of the fingers, and it provides a machine-readable way of describing all the necessary information about a string figure. With computer simulations that can reconstruct the projection of a string figure from its linear sequence, this format allows string figure collectors to store them in a database.

The String Calculus

Not only can we describe string figures in a precise way with linear sequences, we can also precisely describe the behavior of some common string figure moves in terms of the linear sequences. The idea of computing the movements to linear sequences originates from Storers string figure calculus: an algorithmic approach of applying moves to string figures under their linear sequence form. When we play with string figures, we hold the string in an initial position and perform some finger movements to make the new string figure. String figure calculus works exactly the same way: we take a string figure and apply the calculus of a move to get a new string figure. The only difference is that there is no string.

String Segments from Linear Sequences

The most common movement is to pick a string segment with a finger, which is what we have done to construct the star (Figure 4). When we have the string on our hand, its easy to identify all the near and far segments on the string. However, when we are dealing with a linear sequence, it arises the problem of identifying the string segments from this sequence of just fingers and crossings.

The method we will develop in this section allows us to identify the string segments adjacent to a finger in a linear sequence based on the orientation of each finger. In other words, we are interested in the direction towards which the linear sequence traverses around a finger (either clockwise or counterclockwise) along the string. Given the orientation on a finger, we can identify the near and far segment for this finger. For instance, if the linear sequence traverses L1 clockwise, we know that it will reach the near segment before the far one, with the finger itself in between. Figure 5 describes all possible cases of orientations with respect to a linear sequence.

Figure 5 allows us to reduce the problem of identifying string segments into deriving orientations on each finger for an arbitrary linear sequence. Since the first occurrence of a finger in a linear sequence is, by convention, going clockwise, we can already start identifying the string segments of the first finger using the chart.

To derive the orientation on other fingers in a linear sequence, we do it inductively using the orientation of the previous adjacent finger and the number of crossings in between. More precisely, having an even number of crossings between two adjacent fingers means that the orientation is preserved when the linear sequence traverses from one finger to another. On the other hand, an odd number of crossings in between implies that the orientation is reversed. These changes of orientations are illustrated by Figure 6 and 7.

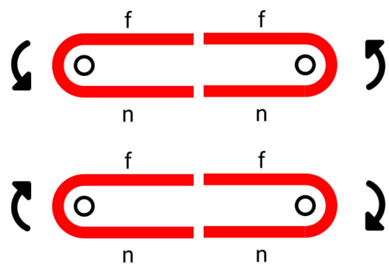

Figure 5: Orientations of String Segments

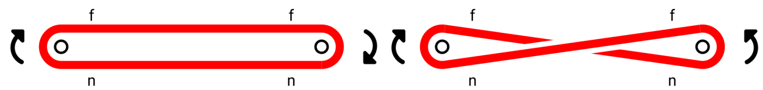

Figure 6: Orientations when traversing to a finger of the opposite hand

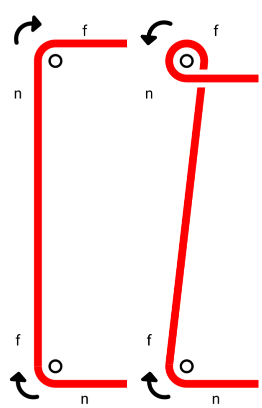

Figure 7: Orientations when traversing to a finger of the opposite hand

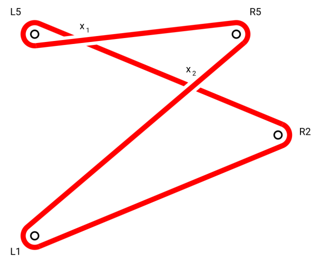

Figure 8: String figure derived from linear sequence 1

Figure 9: The simplest string figure

As an example, consider the following linear sequence:

$$L1 : x_2(o) : R5 : x_1(o) : L5 : x_1(u) : x_2(u) : R2 \tag{1}$$

By convention, $L1$ is clockwise. Therefore $R5$ is counterclockwise as we reverse the orientation when we pass through an odd number of crossings (namely, the one crossing x_2). Similarly, the orientation of $L5$ is clockwise for having only one crossing from the previous finger. Lastly, since $L5$ is clockwise and there are two crossings (i.e. evenly many) between $L5$ and $R2$, the orientation of $R2$ is clockwise. Therefore, we can list the string segments in order and construct an augmented linear sequence, which indeed agrees with its string figure (Figure 8).

$$[n]L1[f] : x_2(o) : [f]R5[n] : x_1(o) : [n]L5[f] : x_1(u) : x_2(u) : [f]R2[n]$$

Picking Segments on Linear Sequences

Now that we can identify string segments from linear sequences, we will now specify the process of picking a segment in a linear sequence.

Let us consider a simple string figure (Figure 9), whose augmented linear sequence is

$$[n]L1[f] : [f]R1[n]$$

We will consider picking the segment $R1n$with the finger $R5$. Since $R1f$ is an intermediate segment located between our finger $R5$ and the target $R1f$, this leads to two possibilities: $R5$ needs to either go over $R1f$ or under $R1f$ in order to reach $R1n$. Once $R5$ reaches $R1n$, here comes another two possibilities of picking this segment: we can either touch the segment with the fingernail of $R5$ (i.e. back of $R5$), pull the string back to normal position; or we can touch the segment with the front of $R5$, and pull the string like a hook. These four possible scenarios produces these string figures, we will also define the notations accordingly as per Figure 10 11 12 13 ([Sto99] p. 15).

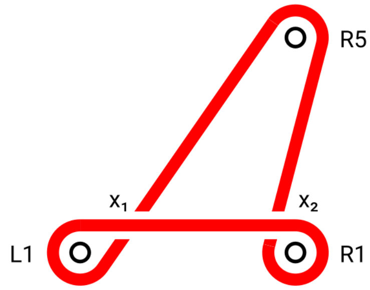

Figure 10: Applying $\underleftarrow{R5}(\underline{R1n})$ gives $L1 : x_1(o) : x_2(o) : R1 : x_2(u) : R5 : x_1(u)$

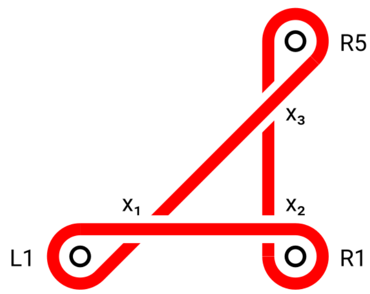

Figure 11: Applying $\underleftarrow{R5}(\overline{R1n})$ gives $L1 : x_1(o) : x_2(o) : R1 : x_2(u) : x_3(u) : R5 : x_3(o) : x_1(u)$

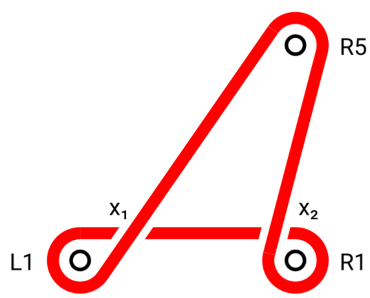

Figure 12: Applying $\overleftarrow{R5}(\underline{R1n})$ gives $L1 : x_1(u) : x_2(u) : R1 : x_2(o) : R5 : x_1(o)$

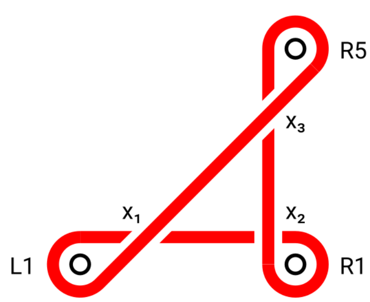

Figure 13: Applying $\overleftarrow{R5}(\overline{R1n})$ gives $L1 : x_1(u) : x_2(u) : R1 : x_2(o) : x_3(u) : R5 : x_3(o) : x_1(o)$

For instance, $\underleftarrow{R5}(\overline{R1n})$ denotes the pick move where $R5$ traverses below (denoted by underarrow) the intermediate string and picks the segment $R1n$ from above (denoted by overline). Notice the similarities between different variants of moves. All four picks create two new crossings on the intermediate segment, and picking from above creates a third one. In addition, passing over or under the intermediate segment $R2n$ only differs by the parity (i.e. undercrossing or overcrossing) of the two crossings on the $R2n$.

The example of picking $R2n$ from $L1 : R1$ provides us an intuition on how a pick move would change a linear sequence in general. We need to first identify the intermediate segments, and create a new pair of crossings for each of them. The parity of the crossings depends on how the finger goes by the intermediate segments (e.g. if the finger passes over them, the crossings that occur on these segments are be undercrossings). The other occurrences of these newly created crossings appear around our finger, which we will call a spike ([Sto99] p367). The spike is then inserted to the place where the target segment is in the linear sequence.

Let us apply the calculus to a more complex example. Consider the string figure we started during the making of the star:

$$[n]L1[f] : [n]L5[f] : [f]R2[n]$$

The first step of making the star was to use $R5$ to pick Lp from above, passing over the intermediate segment $L5f = R2f$ (i.e. the segment between $L5$ and $R2$). We can first identify the segment Lp from our sequence since Lp is a shorthand for the segment $L1f$ or $L5n$. The intermediate strings between Lp and $R5$ are also decidable. To do so, we impose an ordering of the fingers by their number (i.e. order by nearest) and the segments given the fact that the near segment is nearer than the far segment. Then we see that, under our ordering:

$$L1f = L5n = Lp < L5 < L5f = L2f < R5R2n = L1n < L1 < Lp < R5$$

These two inequalities show that while the segment $L5f = L2f$ is intermediate (i.e. between Lp and $R5$), the segment $R2n = L1n$ is not.

Having identified the intermediate segment, we will create a pair of crossings, say x_1 and x_2, to be inserted. Since the move is to pick by passing over the intermediate segment, the crossings on the segment will appear as undercrossings, and we will have overcrossings for the spike. After putting $x_1(o)$ and $x_2(o)$ around $R5$ to make the spike, we will need to surround $R5$ with an additional crossing because $R5$ picks the target segment from above. Therefore, the resulting linear sequence is as follows:

$$L1 : x_2(o) : x_3(o) : R5 : x_3(u) : x_1(o) : L5 : x_1(u) : x_2(u) : R2$$

Notice that the crossings occur as x_2x_1 in the spike, whereas they occur as x_1x_2 on the intermediate segment. This is due to the direction in which the linear sequence traverses across hands.

Conclusion

We made a string figure without a string! Or more precisely, we defined a language that allows us to encode string figures as sequences of symbols, and perform manipulations to these sequences to make new string figures.

References

[Ada94] C. C. Adams. *The knot book*. American Mathematical Soc, 1994.

[Sto99] Tom Storer. *String-figures*. University of Michigan, 1999.