An Introduction to Analyzing 7-Line *Single-Step* Images

This chapter has been created to explore more fully when one could obtain an image that is similar to E8.4 *Three Shape-Shifting Triangles*, 3SST, (30,19,163,13). Some of these explorations occurred prior to March 2023 and are included in ***Electronic String Art***. Very few similar images were found, but see sections E8.6, E9.5, and E12.8.

This chapter uses ideas set forth in E10.1 which focuses on functionally linking ***n***, ***S***, ***P***, and ***J***. In this instance, we focus on creating a counting rule, ***P***, that ends up close to the top after 7 lines are drawn. Call this *The 7-Line Generator Function*:

The 7-Line Generator Function: ***P*** = ROUND(***k·n·S***/7,0) for ***k*** = 1, 2, or 3.

An example using ***n*** = 10 and ***k*** = 1

S	*nS*	*nS*/7	*P*	*7P*	*E* = 7P-*nS*	*nS* mod *7*
7	70	10	10	70	0	0
8	80	11.4	11	77	-3	3
9	90	12.9	13	91	1	6
10	100	14.3	14	98	-2	2
11	110	15.7	16	112	2	5
12	120	17.1	17	119	-1	1
13	130	18.6	19	133	3	4

CLAIM: This ***P*** produces a 7th endpoint ***E*** = 7***P***- ***k·n·S*** that is in one of 7 locations on the vertex frame: -3 ≤ ***E*** ≤ 3.

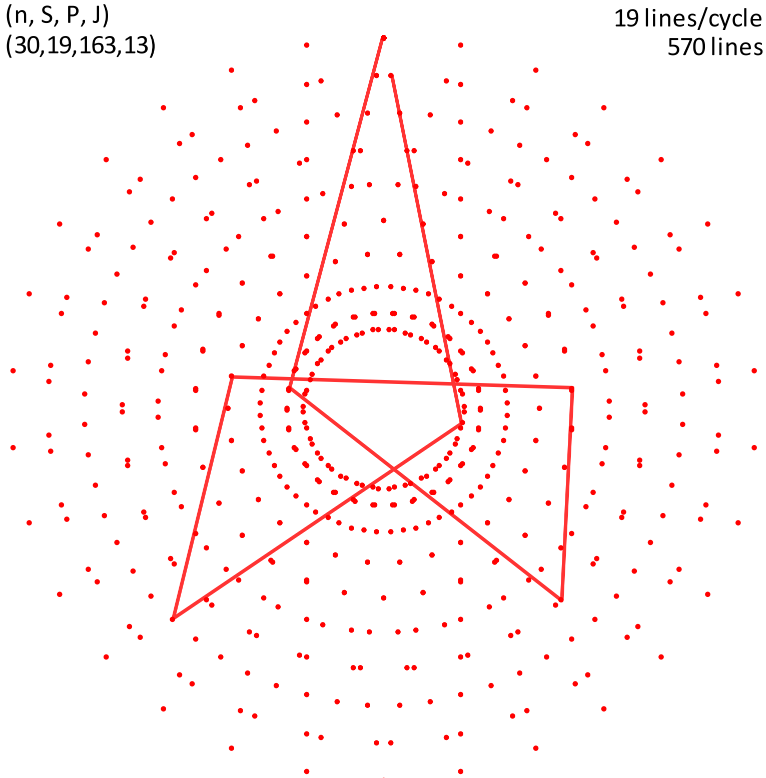
The reason is straightforward. ***P*** is chosen so that after either 1, 2 or 3 times around (***k*** = 1, 2, or 3), the total number of subdivisions counted is as close as possible to a total of ***n·S***, 2·***n·S***, or 3·***n·S***. Seven outcomes for ***E*** are possible:

E = 0 if 0 = ***k·n·S*** mod 7	***E*** = -1 if 1 = ***k·n·S*** mod 7	***E*** = -2 if 2 = ***k·n·S*** mod 7	***E*** = -3 if 3 = ***k·n·S*** mod 7
	E = 1 if 6 = ***k·n·S*** mod 7	***E*** = 2 if 5 = ***k·n·S*** mod 7	***E*** = 3 if 4 = ***k·n·S*** mod 7

An example is shown in tabular form to the right using ***n*** = 10 and ***k*** = 1. The ROUND function takes the closest whole number. When ***n·S***/7 is a bit bigger than the whole number, the number is rounded down (for example, ***P*** is rounded down from 11.4 to 11 when ***S*** = 8, so the 7th endpoint at 77 is just before the top at ***n·S*** = 80, so ***E*** = -3). Conversely, when ***n·S***/7 is a bit smaller than the whole number, the number is rounded up (for example, ***P*** is rounded up from 18.6 to 19 when ***S*** = 13, so the 7th endpoint at 133 is just past the top at ***n·S*** = 130, so ***E*** = 3). In this instance, the 7 images above are all 7-gons or close to 7-gons. If ***k*** = 2, the underlying image is a 7,2-star and if ***k*** = 3, the underlying image is a 7,3-star.

The actual 3SST ***P*** = 163 value is consistent with this ***P*** *Generator Equation*: Given ***n*** = 30, ***S*** = 19 and ***k*** = 2, we have ***P*** = 163 = ROUND(1140/7,0) and 7***P*** = 1141 or 1 past the top of the image since 6 = 2·30·19 mod 7.

The first 7 lines of 3SST are shown here. Notice that the 7th endpoint is 1 past the top, 1 = 7***P*** mod ***nS***.



All images in this chapter use this ***P*** function, or a generalization of this ***P*** function based on 0 remainder images other than 7 lines long, to create images.

In the exploration process, other image types were uncovered that are not part of ESA, but which are worthy counterparts to 3SST.

Other Sections. The discussion proceeds on multiple fronts. Although zero remainder images are only 7 lines long, they provide a template for how the images morph as parameters (typically ***S*** and hence ***P***, for fixed ***n*** and ***J***) change. We therefore start by examining patterns in zero remainder images. How do zero remainder images change as ***n***, ***S***, ***J***, or ***k*** changes? Why do zero remainder images come in triplets? Why do we get different results when ***J*** = 7 or a multiple of 7 (as long as ***n*** > 2***J***)? How can we generalize polygons and stars in a cycle both in terms of those images and more complex variations?