

The `spath3` package: code

Andrew Stacey

loopspace@mathforge.org

v2.7 from 2022/08/24

1 Introduction

The `spath3` package is intended as a library for manipulating PGF's *soft paths*. In between defining a path and using it, PGF stores a path as a *soft path* where all the defining structure has been resolved into the basic operations but these have not yet been written to the output file. They can therefore still be manipulated by `TEX`, and as they have a very rigid form (and limited vocabulary), they are relatively easy to modify. This package provides some methods for working with these paths. It was originally not really intended for use by end users but as a foundation on which other packages can be built. However, over the years I've found myself using it at ever higher levels and so a set of interfaces has been designed using TikZ keys.

It also provides the engine that drives a few other packages, such as the `calligraphy`, `knot`, and `penrose` packages. The first two of these are subpackages of this one. The `calligraphy` package simulates a calligraphic pen stroking a path. The `knots` package can be used to draw knot (and similar) diagrams.

For usage, see the documentation of the following packages (`texdoc <package>`):

- `calligraphy`
- `knots`
- `penrose`
- `spath3` (*this* document is the code, there's another which focusses on usage)

2 Technical Details

The format of a soft path is a sequence of triples of the form `\macro {dimension}{dimension}`. The macro is one of a short list, the dimensions are coordinates in points. There are certain further restrictions, particularly that every path must begin with a `move to`, and Bézier curves consist of three triples.

In the original implementation, I wrapped this token list in a `prop` to store useful information along with the path. Over time, this additional structure has proved a little unwieldy and I've pared it back to working primarily with the original soft path as a token list.

A frequent use of this package is to break a path into pieces and do something with each of those pieces. To that end, there are various words that I use to describe the levels of the structure of a path.

At the top level is the path itself. At the bottom level are the triples of the form `\macro{dim}{dim}`, as described above. In between these are the *segments* and *components*.

A *segment* is a minimal drawing piece. Thus it might be a straight line or a Bézier curve. When a path is broken into segments then each segment is a complete path so it isn't simply a selection of triples from the original path.

A *component* is a minimal connected section of the path. So every component starts with a move command and continues until the next move command. For ease of implementation (and to enable a copperplate pen in the calligraphy package!), an isolated move is considered as a component. Thus the following path consists of three components:

```
\path (0,0) -- (1,0) (2,0) (3,0) to[out=0,in=90] (4,0);
```

3 Implementation

3.1 Initialisation

```
1 <@=@=spath>
```

Load the L^AT_EX3 foundation and register us as a L^AT_EX3 package.

```
2 \NeedsTeXFormat{LaTeX2e}
```

```
3 \RequirePackage{expl3}
```

```
4 \RequirePackage{pgf}
```

```
5 \ProvidesExplPackage {spath3} {2022/08/24} {2.7} {Functions for  
6 manipulating PGF soft paths}
```

```
7 \RequirePackage{xparse}
```

Utilities copied from <https://github.com/loopspace/LaTeX3-Utilities> for adding something in braces to a token list. I find I use this quite a lot in my packages.

```
8 \cs_new_protected:Nn \__spath_tl_put_right_braced:Nn
```

```
9 {
```

```
10 \tl_put_right:Nn #1 { { #2 } }
```

```
11 }
```

```
12 \cs_generate_variant:Nn \__spath_tl_put_right_braced:Nn { NV, cV, cv, Nx, cx }
```

```
13
```

```
14 \cs_new_protected:Nn \__spath_tl_gput_right_braced:Nn
```

```
15 {
```

```
16 \tl_gput_right:Nn #1 { { #2 } }
```

```
17 }
```

```
18 \cs_generate_variant:Nn \__spath_tl_gput_right_braced:Nn { NV, cV, cv, Nx, cx }
```

```
19 \cs_new_protected:Nn \__spath_tl_put_left_braced:Nn
```

```
20 {
```

```
21 \tl_put_left:Nn #1 { { #2 } }
```

```
22 }
```

```
23 \cs_generate_variant:Nn \__spath_tl_put_left_braced:Nn { NV, cV, cv, Nx, cx }
```

```
24
```

```
25 \cs_new_protected:Nn \__spath_tl_gput_left_braced:Nn
```

```
26 {
```

```
27 \tl_gput_left:Nn #1 { { #2 } }
```

```
28 }
```

```
29 \cs_generate_variant:Nn \__spath_tl_gput_left_braced:Nn { NV, cV, cv, Nx, cx }
```

I had to think a bit about how to get \TeX to work the way I wanted. I'm really defining *functions* but \TeX doesn't really have that concept, even with all the amazing \LaTeX 3 stuff. The main issue I had was with scoping and return values. By default, \TeX functions aren't scoped – they work on the same level as the calling functions. To protect the internals from being overwritten, each core function works inside a group. But then I have to work to get the answer out of it. So each of my core functions finishes by storing its return value in an appropriate `output` variable. The core functions are then wrapped in a more user friendly interface that will take that output and assign it to a variable. This also means that I can deal with local and global versions without duplicating code.

```

30 \tl_new:N \g__spath_output_tl
31 \int_new:N \g__spath_output_int
32 \seq_new:N \g__spath_output_seq
33 \bool_new:N \g__spath_output_bool

```

To avoid creating vast numbers of variables, we provide ourselves with a few that we reuse frequently. For that reason, most of them don't have very exciting names.

These are general purpose variables.

```

34 \tl_new:N \l__spath_tmpa_tl
35 \tl_new:N \l__spath_tmpb_tl
36 \tl_new:N \l__spath_tmpc_tl
37 \tl_new:N \l__spath_tmpd_tl
38 \tl_new:N \l__spath_tmpe_tl
39 \tl_new:N \l__spath_tmpf_tl
40 \tl_new:N \l__spath_tmpg_tl
41 \tl_new:N \l__spath_tmph_tl
42 \tl_new:N \l__spath_tmphi_tl
43
44 \seq_new:N \l__spath_tmpa_seq
45 \seq_new:N \l__spath_tmpb_seq
46 \seq_new:N \l__spath_tmpc_seq
47
48 \dim_new:N \l__spath_tmpa_dim
49 \dim_new:N \l__spath_tmpb_dim
50
51 \fp_new:N \l__spath_tmpa_fp
52 \fp_new:N \l__spath_tmpb_fp
53 \fp_new:N \l__spath_tmpc_fp
54 \fp_new:N \l__spath_tmpd_fp
55 \fp_new:N \l__spath_tmpe_fp
56 \fp_new:N \l__spath_tmpf_fp
57
58 \int_new:N \l__spath_tmpa_int
59 \int_new:N \l__spath_tmpb_int
60
61 \bool_new:N \l__spath_tmpa_bool

```

Whenever I need more than two `dim` variables it is because I need to remember the position of a move.

```

62 \dim_new:N \l__spath_move_x_dim
63 \dim_new:N \l__spath_move_y_dim

```

Closed paths often need special handling. When it's needed, this will say whether the path is closed or not.

```

64 \bool_new:N \l__spath_closed_bool

```

True rectangles are rare, but need special handling. They are specified by two tokens, the first specifies the lower left corner which can be handled pretty much as other tokens but the second specifies the width and height meaning that it transforms differently. So when encountering one, the coordinates of the lower left corner are useful to remember.

```

65 \dim_new:N \l__spath_rectx_dim
66 \dim_new:N \l__spath_recty_dim
67
68 \bool_new:N \l__spath_rect_bool

```

When restoring a path we need to know whether to update the stored moveto.

```

69 \bool_new:N \l__spath_movetorelevant_bool

```

The intersection routine can't happen inside a group so we need two token lists to hold the paths that we'll intersect.

```

70 \tl_new:N \l__spath_intersecta_tl
71 \tl_new:N \l__spath_intersectb_tl

```

We need to be able to compare against the macros that can occur in a soft path so these token lists contain them. These are global constants so that they can be used in other packages.

```

72 \tl_const:Nn \c_spath_moveto_tl {\pgfsyssoftpath@movetotoken}
73 \tl_const:Nn \c_spath_lineto_tl {\pgfsyssoftpath@linetotoken}
74 \tl_const:Nn \c_spath_curveto_tl {\pgfsyssoftpath@curvetotoken}
75 \tl_const:Nn \c_spath_curvetoa_tl {\pgfsyssoftpath@curvetosupportatoken}
76 \tl_const:Nn \c_spath_curvetob_tl {\pgfsyssoftpath@curvetosupportbtoken}
77 \tl_const:Nn \c_spath_closepath_tl {\pgfsyssoftpath@closepathtoken}
78 \tl_const:Nn \c_spath_rectcorner_tl {\pgfsyssoftpath@rectcornertoken}
79 \tl_const:Nn \c_spath_rectsize_tl {\pgfsyssoftpath@rectsizetoken}

```

We will want to be able to use anonymous spaths internally, so we create a global counter that we can use to refer to them.

```

80 \int_new:N \g__spath_anon_int
81 \int_gzero:N \g__spath_anon_int

```

`\spath_anonymous:N` Set the token list to the next anonymous name.

```

\spath_ganonymous:N
82 \cs_new_protected_nopar:Npn \spath_anonymous:N #1
83 {
84   \tl_set:Nx #1 {anonymous_\int_use:N \g__spath_anon_int}
85   \int_gincr:N \g__spath_anon_int
86 }
87 \cs_new_protected_nopar:Npn \spath_ganonymous:N #1
88 {
89   \tl_gset:Nx #1 {anonymous_\int_use:N \g__spath_anon_int}
90   \int_gincr:N \g__spath_anon_int
91 }
92 \cs_generate_variant:Nn \spath_anonymous:N {c}
93 \cs_generate_variant:Nn \spath_ganonymous:N {c}

```

(End definition for `\spath_anonymous:N` and `\spath_ganonymous:N`.)

And some error messages

```

94 \msg_new:nnn { spath3 } { unknown path construction }
95 { The~ path~ construction~ element~ #1~ is~ not~ currently~ supported.}

```

3.2 Functional Implementation

In the functional approach, we start with a token list containing a soft path and do something to it (either calculate some information or manipulate it in some fashion). We then store that information, or the manipulated path, in an appropriate macro. The macro to store it in is the first argument. These functions occur in two versions, the one with the `g` makes the assignment global.

```

\spath_segments_to_seq:Nn Splits a soft path into segments, storing the result in a sequence.
\spath_segments_gto_seq:Nn
96 \cs_new_protected_nopar:Npn \__spath_segments_to_seq:n #1
97 {
98   \group_begin:
99   \tl_set:Nn \l__spath_tmpa_tl {#1}
100  \tl_clear:N \l__spath_tmpb_tl
101  \seq_clear:N \l__spath_tmpa_seq
102  \dim_zero:N \l__spath_tmpa_dim
103  \dim_zero:N \l__spath_tmpb_dim
104
105  \bool_until_do:nn {
106    \tl_if_empty_p:N \l__spath_tmpa_tl
107  }
108  {
109    \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
110    \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
111    \tl_case:NnF \l__spath_tmpc_tl
112    {
113      \c_spath_moveto_tl
114      {
115        \tl_set_eq:NN \l__spath_tmpb_tl \c_spath_moveto_tl
116        \tl_put_right:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
117        \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
118        \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
119
120        \tl_put_right:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
121        \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
122        \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
123
124        \tl_set:Nx \l__spath_tmpd_tl {\tl_head:N \l__spath_tmpa_tl}
125        \tl_if_eq:NNF \l__spath_tmpd_tl \c_spath_moveto_tl
126        {
127          \tl_clear:N \l__spath_tmpb_tl
128        }
129
130      }
131
132      \c_spath_lineto_tl
133      {
134        \tl_set_eq:NN \l__spath_tmpb_tl \c_spath_moveto_tl
135        \tl_put_right:Nx \l__spath_tmpb_tl
136        {
137          {\dim_use:N \l__spath_tmpa_dim}
138          {\dim_use:N \l__spath_tmpb_dim}
139        }
140        \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl

```

```

141
142 \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
143 \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
144 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
145
146 \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
147 \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
148 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
149
150 }
151
152 \c_spath_curvetoa_tl
153 {
154 \tl_set_eq:NN \l__spath_tmpb_tl \c_spath_movetoa_tl
155 \tl_put_right:Nx \l__spath_tmpb_tl
156 {
157   {\dim_use:N \l__spath_tmpa_dim}
158   {\dim_use:N \l__spath_tmpb_dim}
159 }
160 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_curvetoa_tl
161
162 \prg_replicate:nn {2} {
163   \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
164   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
165   \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
166   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
167   \tl_put_right:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
168   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
169 }
170
171 \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
172 \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
173 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
174
175 \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
176 \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
177 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
178
179 }
180
181 \c_spath_rectcorner_tl
182 {
183   \tl_set_eq:NN \l__spath_tmpb_tl \c_spath_rectcorner_tl
184
185   \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
186   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
187   \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
188   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
189   \tl_put_right:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
190   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
191
192   \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
193   \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
194   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}

```

```

195
196     \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
197     \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
198     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
199
200   }
201
202   \c_spath_closepath_tl
203   {
204     \tl_set_eq:NN \l__spath_tmpb_tl \c_spath_moveto_tl
205     \tl_put_right:Nx \l__spath_tmpb_tl
206     {
207       {\dim_use:N \l__spath_tmpa_dim}
208       {\dim_use:N \l__spath_tmpb_dim}
209     }
210     \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
211
212     \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
213     \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
214     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
215
216     \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
217     \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
218     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
219
220   }
221
222 }
223 {
224
225   \tl_set_eq:NN \l__spath_tmpb_tl \l__spath_tmpc_tl
226   \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
227   \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
228   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
229
230   \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
231   \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
232   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
233
234 }
235
236 \tl_if_empty:NF \l__spath_tmpb_tl
237 {
238   \seq_put_right:NV \l__spath_tmpa_seq \l__spath_tmpb_tl
239 }
240 \tl_clear:N \l__spath_tmpb_tl
241 }
242
243 \seq_gclear:N \g__spath_output_seq
244 \seq_gset_eq:NN \g__spath_output_seq \l__spath_tmpa_seq
245 \group_end:
246 }
247 \cs_new_protected_nopar:Npn \spath_segments_to_seq:Nn #1#2
248 {

```

```

249 \__spath_segments_to_seq:n {#2}
250 \seq_clear_new:N #1
251 \seq_set_eq:NN #1 \g__spath_output_seq
252 \seq_gclear:N \g__spath_output_seq
253 }
254 \cs_generate_variant:Nn \spath_segments_to_seq:Nn {NV, cn, cV, Nv, cv}
255 \cs_new_protected_nopar:Npn \spath_segments_gto_seq:Nn #1#2
256 {
257 \__spath_segments_to_seq:n {#2}
258 \seq_clear_new:N #1
259 \seq_gset_eq:NN #1 \g__spath_output_seq
260 \seq_gclear:N \g__spath_output_seq
261 }
262 \cs_generate_variant:Nn \spath_segments_gto_seq:Nn {NV, cn, cV, Nv, cv}

```

(End definition for `\spath_segments_to_seq:Nn` and `\spath_segments_gto_seq:Nn`.)

`\spath_components_to_seq:Nn` Splits a soft path into *components*, storing the result in a sequence or a clist.

```

\spath_components_gto_seq:Nn
\spath_components_to_clist:Nn
\spath_components_gto_clist:Nn
263 \cs_new_protected_nopar:Npn \__spath_components_to_seq:n #1
264 {
265 \group_begin:
266 \tl_set:Nn \l__spath_tmpa_tl {#1}
267 \seq_clear:N \l__spath_tmpa_seq
268 \tl_set:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
269 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
270
271 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_moveto_tl
272
273 \bool_do_until:nm {
274 \tl_if_empty_p:N \l__spath_tmpa_tl
275 }
276 {
277 \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
278 \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_moveto_tl
279 {
280 \seq_put_right:NV \l__spath_tmpa_seq \l__spath_tmpb_tl
281 \tl_clear:N \l__spath_tmpb_tl
282 }
283 \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl
284 {
285 \seq_put_right:NV \l__spath_tmpa_seq \l__spath_tmpb_tl
286 \tl_clear:N \l__spath_tmpb_tl
287 }
288 \tl_if_single:NTF \l__spath_tmpc_tl
289 {
290 \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpc_tl
291 }
292 {
293 \tl_put_right:Nx \l__spath_tmpb_tl {\l__spath_tmpc_tl}
294 }
295 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
296 }
297
298 \seq_gclear:N \g__spath_output_seq

```



```

299 \seq_gset_eq:NN \g__spath_output_seq \l__spath_tmpa_seq
300 \group_end:
301 }
302 \cs_new_protected_nopar:Npn \spath_components_to_seq:Nn #1#2
303 {
304   \__spath_components_to_seq:n {#2}
305   \seq_clear_new:N #1
306   \seq_set_eq:NN #1 \g__spath_output_seq
307   \seq_gclear:N \g__spath_output_seq
308 }
309 \cs_generate_variant:Nn \spath_components_to_seq:Nn {NV, cn, cV, cv, Nv}
310 \cs_new_protected_nopar:Npn \spath_components_gto_seq:Nn #1#2
311 {
312   \__spath_components_to_seq:n {#2}
313   \seq_clear_new:N #1
314   \seq_gset_eq:NN #1 \g__spath_output_seq
315   \seq_gclear:N \g__spath_output_seq
316 }
317 \cs_generate_variant:Nn \spath_components_gto_seq:Nn {NV, cn, cV, cv, Nv}
318 \cs_new_protected_nopar:Npn \spath_components_to_clist:Nn #1#2
319 {
320   \__spath_components_to_seq:n {#2}
321   \clist_clear_new:N #1
322   \clist_set_from_seq:NN #1 \g__spath_output_seq
323   \seq_gclear:N \g__spath_output_seq
324 }
325 \cs_generate_variant:Nn \spath_components_to_clist:Nn {NV, cn, cV, cv, Nv}
326 \cs_new_protected_nopar:Npn \spath_components_gto_clist:Nn #1#2
327 {
328   \__spath_components_to_seq:n {#2}
329   \clist_clear_new:N #1
330   \clist_gset_from_seq:NN #1 \g__spath_output_seq
331   \seq_gclear:N \g__spath_output_seq
332 }
333 \cs_generate_variant:Nn \spath_components_gto_clist:Nn {NV, cn, cV, cv, Nv}

```

(End definition for \spath_components_to_seq:Nn and others.)

`\spath_length:n` Counts the number of triples in the path.

```

334 \cs_new_protected_nopar:Npn \spath_length:n #1
335 {
336   \int_eval:n {\tl_count:n {#1} / 3}
337 }
338 \cs_generate_variant:Nn \spath_length:n {V}

```

(End definition for \spath_length:n.)

`\spath_reallength:Nn` The real length of a path is the number of triples that actually draw something (that is, the number of lines, curves, rectangles, and closepaths).

```

339 \cs_new_protected_nopar:Npn \__spath_reallength:n #1
340 {
341   \group_begin:
342   \int_set:Nn \l__spath_tmpa_int {0}
343   \tl_map_inline:nn {#1} {

```

```

344 \tl_set:Nn \l__spath_tmpa_tl {##1}
345 \tl_case:NnT \l__spath_tmpa_tl
346 {
347   \c_spath_lineto_tl {}
348   \c_spath_curveto_tl {}
349   \c_spath_closepath_tl {}
350   \c_spath_rectsize_tl {}
351 }
352 {
353   \int_incr:N \l__spath_tmpa_int
354 }
355 }
356 \int_gzero:N \g__spath_output_int
357 \int_gset_eq:NN \g__spath_output_int \l__spath_tmpa_int
358 \group_end:
359 }
360 \cs_new_protected_nopar:Npn \spath_reallength:Nn #1#2
361 {
362   \__spath_reallength:n {#2}
363   \int_set_eq:NN #1 \g__spath_output_int
364   \int_gzero:N \g__spath_output_int
365 }
366 \cs_generate_variant:Nn \spath_reallength:Nn {NV, cn, cV, Nv, cv}
367 \cs_new_protected_nopar:Npn \spath_greallength:Nn #1#2
368 {
369   \__spath_reallength:n {#2}
370   \int_gset_eq:NN #1 \g__spath_output_int
371   \int_gzero:N \g__spath_output_int
372 }
373 \cs_generate_variant:Nn \spath_greallength:Nn {NV, cn, cV}

```

(End definition for `\spath_reallength:Nn` and `\spath_greallength:Nn`.)

`\spath_numberofcomponents:Nn` A component is a continuous segment of the path, separated by moves. Successive moves are not collapsed, and zero length moves count.

```

\spath_numberofcomponents:Nn
374 \cs_new_protected_nopar:Npn \__spath_numberofcomponents:n #1
375 {
376   \group_begin:
377   \int_set:Nn \l__spath_tmpa_int {0}
378   \tl_map_inline:nn {#1} {
379     \tl_set:Nn \l__spath_tmpa_tl {##1}
380     \tl_case:Nn \l__spath_tmpa_tl
381     {
382       \c_spath_moveto_tl
383       {
384         \int_incr:N \l__spath_tmpa_int
385       }
386       \c_spath_rectcorner_tl
387       {
388         \int_incr:N \l__spath_tmpa_int
389       }
390     }
391   }
392   \int_gzero:N \g__spath_output_int

```

```

393 \int_gset_eq:NN \g__spath_output_int \l__spath_tmpa_int
394 \group_end:
395 }
396 \cs_new_protected_nopar:Npn \spath_numberofcomponents:Nn #1#2
397 {
398   \__spath_numberofcomponents:n {#2}
399   \int_set_eq:NN #1 \g__spath_output_int
400   \int_gzero:N \g__spath_output_int
401 }
402 \cs_generate_variant:Nn \spath_numberofcomponents:Nn {NV, cn, cV, Nv}
403 \cs_new_protected_nopar:Npn \spath_gnumberofcomponents:Nn #1#2
404 {
405   \__spath_numberofcomponents:n {#2}
406   \int_gset_eq:NN #1 \g__spath_output_int
407   \int_gzero:N \g__spath_output_int
408 }
409 \cs_generate_variant:Nn \spath_gnumberofcomponents:Nn {NV, cn, cV, Nv}

```

(End definition for \spath_numberofcomponents:Nn and \spath_gnumberofcomponents:Nn.)

`\spath_initialpoint:Nn` The starting point of the path.

```

\spath_ginitialpoint:Nn
410 \cs_new_protected_nopar:Npn \__spath_initialpoint:n #1
411 {
412   \group_begin:
413   \tl_clear:N \l__spath_tmpa_tl
414   \tl_set:Nx \l__spath_tmpa_tl
415   {
416     { \tl_item:nn {#1} {2} }
417     { \tl_item:nn {#1} {3} }
418   }
419   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
420   \group_end:
421 }
422 \cs_new_protected_nopar:Npn \spath_initialpoint:Nn #1#2
423 {
424   \__spath_initialpoint:n {#2}
425   \tl_set_eq:NN #1 \g__spath_output_tl
426   \tl_gclear:N \g__spath_output_tl
427 }
428 \cs_generate_variant:Nn \spath_initialpoint:Nn {NV, cn, cV, Nv}
429 \cs_new_protected_nopar:Npn \spath_ginitialpoint:Nn #1#2
430 {
431   \__spath_initialpoint:n {#2}
432   \tl_gset_eq:NN #1 \g__spath_output_tl
433   \tl_gclear:N \g__spath_output_tl
434 }
435 \cs_generate_variant:Nn \spath_ginitialpoint:Nn {NV, cn, cV, Nv}

```

(End definition for \spath_initialpoint:Nn and \spath_ginitialpoint:Nn.)

`\spath_finalpoint:Nn` The final point of the path.

```

\spath_gfinalpoint:Nn
436 \cs_new_protected_nopar:Npn \__spath_finalpoint:n #1
437 {
438   \group_begin:
439   \tl_set:Nn \l__spath_tmpa_tl {#1}

```

```

440 \tl_reverse:N \l__spath_tmpa_tl
441 \tl_clear:N \l__spath_tmpb_tl
442 \tl_set:Nx \l__spath_tmpc_tl {\tl_item:Nn \l__spath_tmpa_tl {3}}
443 \tl_if_eq:NNTF \l__spath_tmpc_tl \c_spath_rectsize_tl
444 {
445   \tl_set:Nx \l__spath_tmpb_tl
446   {
447     {
448       \dim_eval:n
449       {
450         \tl_item:Nn \l__spath_tmpa_tl {2}
451         +
452         \tl_item:Nn \l__spath_tmpa_tl {5}
453       }
454     }
455     {
456       \dim_eval:n
457       {
458         \tl_item:Nn \l__spath_tmpa_tl {1}
459         +
460         \tl_item:Nn \l__spath_tmpa_tl {4}
461       }
462     }
463   }
464 }
465 {
466   \tl_set:Nx \l__spath_tmpb_tl
467   {
468     { \tl_item:Nn \l__spath_tmpa_tl {2} }
469     { \tl_item:Nn \l__spath_tmpa_tl {1} }
470   }
471 }
472 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
473 \group_end:
474 }
475 \cs_new_protected_nopar:Npn \spath_finalpoint:Nn #1#2
476 {
477   \__spath_finalpoint:n {#2}
478   \tl_set_eq:NN #1 \g__spath_output_tl
479   \tl_gclear:N \g__spath_output_tl
480 }
481 \cs_generate_variant:Nn \spath_finalpoint:Nn {NV, cn, cV, Nv}
482 \cs_new_protected_nopar:Npn \spath_gfinalpoint:Nn #1#2
483 {
484   \__spath_finalpoint:n {#2}
485   \tl_gset_eq:NN #1 \g__spath_output_tl
486   \tl_gclear:N \g__spath_output_tl
487 }
488 \cs_generate_variant:Nn \spath_gfinalpoint:Nn {NV, cn, cV, Nv}

```

(End definition for \spath_finalpoint:Nn and \spath_gfinalpoint:Nn.)

```

\spath_finalmovepoint:Nn Get the last move on the path.
\spath_gfinalmovepoint:Nn
489 \cs_new_protected_nopar:Npn \__spath_finalmovepoint:n #1

```

```

490 {
491   \group_begin:
492   \tl_set:Nn \l__spath_tmpc_tl { {Opt} {Opt} }
493   \tl_set:Nn \l__spath_tmpa_tl {#1}
494   \bool_do_until:nn
495   {
496     \tl_if_empty_p:N \l__spath_tmpa_tl
497   }
498   {
499     \tl_set:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
500     \tl_case:Nn \l__spath_tmpb_tl
501     {
502       \c_spath_moveto_tl
503       {
504         \tl_set:Nx \l__spath_tmpc_tl
505         {
506           { \tl_item:Nn \l__spath_tmpa_tl {2} }
507           { \tl_item:Nn \l__spath_tmpa_tl {3} }
508         }
509       }
510
511       \c_spath_rectcorner_tl
512       {
513         \tl_set:Nx \l__spath_tmpc_tl
514         {
515           { \tl_item:Nn \l__spath_tmpa_tl {2} }
516           { \tl_item:Nn \l__spath_tmpa_tl {3} }
517         }
518       }
519
520     }
521     \prg_replicate:nn {3}
522     {
523       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
524     }
525   }
526   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpc_tl
527   \group_end:
528 }
529 \cs_new_protected_nopar:Npn \spath_finalmovepoint:Nn #1#2
530 {
531   \__spath_finalmovepoint:n {#2}
532   \tl_set_eq:NN #1 \g__spath_output_tl
533   \tl_gclear:N \g__spath_output_tl
534 }
535 \cs_generate_variant:Nn \spath_finalmovepoint:Nn {NV, cn, cV}
536 \cs_new_protected_nopar:Npn \spath_gfinalmovepoint:Nn #1#2
537 {
538   \__spath_finalmovepoint:n {#2}
539   \tl_gset_eq:NN #1 \g__spath_output_tl
540   \tl_gclear:N \g__spath_output_tl
541 }
542 \cs_generate_variant:Nn \spath_gfinalmovepoint:Nn {NV, cn, cV}

```

(End definition for \spath_finalmovepoint:Nn and \spath_gfinalmovepoint:Nn.)

\spath_initialtangent:Nn
\spath_ginitialtangent:Nn

The starting tangent of the path.

```

543 \cs_new_protected_nopar:Npn \__spath_initialtangent:n #1
544 {
545   \group_begin:
546   \tl_set:Nx \l__spath_tmpb_tl {\tl_item:nn {#1} {4}}
547   \tl_if_eq:NNTF \l__spath_tmpb_tl \c_spath_curvetoa_tl
548   {
549     \fp_set:Nn \l__spath_tmpa_fp {3}
550   }
551   {
552     \fp_set:Nn \l__spath_tmpa_fp {1}
553   }
554   \tl_clear:N \l__spath_tmpa_tl
555   \tl_set:Nx \l__spath_tmpa_tl
556   {
557     {
558       \fp_to_dim:n {
559         \l__spath_tmpa_fp
560         *
561         (
562           \tl_item:nn {#1} {5}
563           -
564           \tl_item:nn {#1} {2}
565         )
566       }
567     }
568     {
569       \fp_to_dim:n {
570         \l__spath_tmpa_fp
571         *
572         (
573           \tl_item:nn {#1} {6}
574           -
575           \tl_item:nn {#1} {3}
576         )
577       }
578     }
579   }
580   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
581   \group_end:
582 }
583 \cs_new_protected_nopar:Npn \spath_initialtangent:Nn #1#2
584 {
585   \__spath_initialtangent:n {#2}
586   \tl_set_eq:NN #1 \g__spath_output_tl
587   \tl_gclear:N \g__spath_output_tl
588 }
589 \cs_generate_variant:Nn \spath_initialtangent:Nn {NV, cn, cV, Nv}
590 \cs_new_protected_nopar:Npn \spath_ginitialtangent:Nn #1#2
591 {
592   \__spath_initialtangent:n {#2}
593   \tl_gset_eq:NN #1 \g__spath_output_tl
594   \tl_gclear:N \g__spath_output_tl
595 }

```

```

596 \cs_generate_variant:Nn \spath_ginitialtangent:Nn {NV, cn, cV, Nv}
(End definition for \spath_initialtangent:Nn and \spath_ginitialtangent:Nn.)

```

```

\spath_finalltangent:Nn
\spath_gfinalltangent:Nn

```

The final tangent of the path.

```

597 \cs_new_protected_nopar:Npn \__spath_finalltangent:n #1
598 {
599   \group_begin:
600   \tl_set:Nn \l__spath_tmpa_tl {#1}
601   \tl_reverse:N \l__spath_tmpa_tl
602   \tl_set:Nx \l__spath_tmpb_tl {\tl_item:nn {#1} {6}}
603   \tl_if_eq:NNTF \l__spath_tmpb_tl \c_spath_curveto_tl
604   {
605     \fp_set:Nn \l__spath_tmpa_fp {3}
606   }
607   {
608     \fp_set:Nn \l__spath_tmpa_fp {1}
609   }
610   \tl_clear:N \l__spath_tmpb_tl
611   \tl_set:Nx \l__spath_tmpb_tl
612   {
613     {
614       \fp_to_dim:n {
615         \l__spath_tmpa_fp
616         *
617         (
618           \tl_item:Nn \l__spath_tmpa_tl {2}
619           -
620           \tl_item:Nn \l__spath_tmpa_tl {5}
621         )
622       }
623     }
624     {
625       \fp_to_dim:n {
626         \l__spath_tmpa_fp
627         *
628         (
629           \tl_item:Nn \l__spath_tmpa_tl {1}
630           -
631           \tl_item:Nn \l__spath_tmpa_tl {4}
632         )
633       }
634     }
635   }
636   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
637   \group_end:
638 }
639 \cs_new_protected_nopar:Npn \spath_finalltangent:Nn #1#2
640 {
641   \__spath_finalltangent:n {#2}
642   \tl_set_eq:NN #1 \g__spath_output_tl
643   \tl_gclear:N \g__spath_output_tl
644 }
645 \cs_generate_variant:Nn \spath_finalltangent:Nn {NV, cn, cV, Nv}

```

```

646 \cs_new_protected_nopar:Npn \spath_gfinaltangent:Nn #1#2
647 {
648   \__spath_finaltangent:n {#2}
649   \tl_gset_eq:NN #1 \g__spath_output_tl
650   \tl_gclear:N \g__spath_output_tl
651 }
652 \cs_generate_variant:Nn \spath_gfinaltangent:Nn {NV, cn, cV, Nv}

```

(End definition for `\spath_finaltangent:Nn` and `\spath_gfinaltangent:Nn`.)

`\spath_finalmovetangent:Nn`
`\spath_gfinalmovetangent:Nn`

Get the last move on the path.

```

653 \cs_new_protected_nopar:Npn \__spath_finalmovetangent:n #1
654 {
655   \group_begin:
656   \tl_set:Nn \l__spath_tmpc_tl { {0pt} {0pt} }
657   \tl_set:Nn \l__spath_tmpa_tl {#1}
658   \bool_do_until:n
659   {
660     \tl_if_empty_p:N \l__spath_tmpa_tl
661   }
662   {
663     \tl_set:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
664     \tl_case:Nn \l__spath_tmpb_tl
665     {
666       \c_spath_moveto_tl
667       {
668         \tl_set:Nx \l__spath_tmpd_tl { \tl_item:Nn \l__spath_tmpa_tl {4} }
669         \tl_if_eq:NNTF \l__spath_tmpd_tl \c_spath_curveto_tl
670         {
671           \fp_set:Nn \l__spath_tmpa_fp {3}
672         }
673         {
674           \fp_set:Nn \l__spath_tmpa_fp {1}
675         }
676         \tl_set:Nx \l__spath_tmpc_tl
677         {
678           {
679             \fp_to_dim:n
680             {
681               \l__spath_tmpa_fp
682               *
683               (
684                 \tl_item:Nn \l__spath_tmpa_tl {5}
685                 -
686                 \tl_item:Nn \l__spath_tmpa_tl {2}
687               )
688             }
689           }
690           {
691             \fp_to_dim:n
692             {
693               \l__spath_tmpa_fp
694               *
695               (

```



```

696             \tl_item:Nn \l__spath_tmpa_tl {6}
697             -
698             \tl_item:Nn \l__spath_tmpa_tl {3}
699             )
700         }
701     }
702 }
703 }
704 }
705 \prg_replicate:nn {3}
706 {
707     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
708 }
709 }
710 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpc_tl
711 \group_end:
712 }
713 \cs_new_protected_nopar:Npn \spath_finalmovetangent:Nn #1#2
714 {
715     \__spath_finalmovetangent:n {#2}
716     \tl_set_eq:NN #1 \g__spath_output_tl
717     \tl_gclear:N \g__spath_output_tl
718 }
719 \cs_generate_variant:Nn \spath_finalmovetangent:Nn {NV, cn, cV}
720 \cs_new_protected_nopar:Npn \spath_gfinalmovetangent:Nn #1#2
721 {
722     \__spath_finalmovetangent:n {#2}
723     \tl_gset_eq:NN #1 \g__spath_output_tl
724     \tl_gclear:N \g__spath_output_tl
725 }
726 \cs_generate_variant:Nn \spath_gfinalmovetangent:Nn {NV, cn, cV}

```

(End definition for \spath_finalmovetangent:Nn and \spath_gfinalmovetangent:Nn.)

`\spath_reverse:Nn` This computes the reverse of the path.

```

\spath_greverse:Nn
727 \cs_new_protected_nopar:Npn \__spath_reverse:n #1
728 {
729     \group_begin:
730     \tl_set:Nn \l__spath_tmpa_tl {#1}
731
732     \tl_clear:N \l__spath_tmpb_tl
733     \tl_clear:N \l__spath_tmpd_tl
734
735     \bool_set_false:N \l__spath_rect_bool
736
737     \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
738     \bool_while_do:nn {
739         \tl_if_eq_p:NN \l__spath_tmpc_tl \c_spath_rectcorner_tl
740     }
741     {
742         \tl_put_left:Nx \l__spath_tmpd_tl
743         {
744             \tl_item:Nn \l__spath_tmpa_tl {1}
745             {\tl_item:Nn \l__spath_tmpa_tl {2}}

```

```

746     {\tl_item:Nn \l__spath_tmpa_tl {3}}
747     \tl_item:Nn \l__spath_tmpa_tl {4}
748     {\tl_item:Nn \l__spath_tmpa_tl {5}}
749     {\tl_item:Nn \l__spath_tmpa_tl {6}}
750   }
751   \prg_replicate:nn {6}
752   {
753     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
754   }
755   \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpa_tl}
756   \bool_set_true:N \l__spath_rect_bool
757 }
758
759 \tl_if_empty:NF \l__spath_tmpa_tl
760 {
761   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
762   \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
763   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
764   \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
765   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpa_tl}
766
767   \tl_put_left:Nx \l__spath_tmpd_tl
768   {
769     {\dim_use:N \l__spath_tmpa_dim}
770     {\dim_use:N \l__spath_tmpb_dim}
771   }
772
773   \bool_set_false:N \l__spath_closed_bool
774
775   \bool_until_do:nn {
776     \tl_if_empty_p:N \l__spath_tmpa_tl
777   }
778   {
779     \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpa_tl}
780     \bool_set_false:N \l__spath_rect_bool
781
782     \tl_case:NnTF \l__spath_tmpe_tl
783     {
784       \c_spath_moveto_tl {
785
786         \bool_if:NT \l__spath_closed_bool
787         {
788           \tl_put_right:NV \l__spath_tmpd_tl \c_spath_closepath_tl
789           \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpd_tl}
790           \tl_put_right:Nx \l__spath_tmpd_tl
791           {
792             { \tl_head:N \l__spath_tmpd_tl }
793             { \tl_head:N \l__spath_tmpe_tl }
794           }
795         }
796         \bool_set_false:N \l__spath_closed_bool
797         \tl_put_left:NV \l__spath_tmpd_tl \c_spath_moveto_tl
798         \tl_put_left:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
799         \tl_clear:N \l__spath_tmpd_tl

```

```

800     }
801     \c_spath_rectcorner_tl {
802
803         \bool_if:NT \l__spath_closed_bool
804         {
805             \tl_put_right:NV \l__spath_tmpd_tl \c_spath_closepath_tl
806             \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpd_tl}
807             \tl_put_right:Nx \l__spath_tmpd_tl
808             {
809                 { \tl_head:N \l__spath_tmpd_tl }
810                 { \tl_head:N \l__spath_tmpe_tl }
811             }
812         }
813         \bool_set_false:N \l__spath_closed_bool
814         \tl_put_left:NV \l__spath_tmpd_tl \c_spath_moveto_tl
815         \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpd_tl
816         \tl_clear:N \l__spath_tmpd_tl
817
818         \bool_while_do:nn {
819             \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_rectcorner_tl
820         }
821         {
822             \tl_put_left:Nx \l__spath_tmpe_tl
823             {
824                 \tl_item:Nn \l__spath_tmpe_tl {1}
825                 {\tl_item:Nn \l__spath_tmpe_tl {2}}
826                 {\tl_item:Nn \l__spath_tmpe_tl {3}}
827                 \tl_item:Nn \l__spath_tmpe_tl {4}
828                 {\tl_item:Nn \l__spath_tmpe_tl {5}}
829                 {\tl_item:Nn \l__spath_tmpe_tl {6}}
830             }
831             \prg_replicate:nn {6}
832             {
833                 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
834             }
835             \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpe_tl}
836         }
837         \bool_set_true:N \l__spath_rect_bool
838
839     }
840     \c_spath_lineto_tl {
841         \tl_put_left:NV \l__spath_tmpd_tl \c_spath_lineto_tl
842     }
843     \c_spath_curveto_tl {
844         \tl_put_left:NV \l__spath_tmpd_tl \c_spath_curvetoa_tl
845     }
846     \c_spath_curvetoa_tl {
847         \tl_put_left:NV \l__spath_tmpd_tl \c_spath_curveto_tl
848     }
849     \c_spath_curvetob_tl {
850         \tl_put_left:NV \l__spath_tmpd_tl \c_spath_curvetob_tl
851     }
852 }
853 {

```

```

854     \tl_if_empty:NF \l__spath_tmpa_tl
855     {
856     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
857
858     \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
859     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
860     \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
861     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
862
863     \tl_put_left:Nx \l__spath_tmpd_tl
864     {
865     {\dim_use:N \l__spath_tmpa_dim}
866     {\dim_use:N \l__spath_tmpb_dim}
867     }
868     }
869   }
870   {
871     \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_closepath_tl
872     {
873     \bool_set_true:N \l__spath_closed_bool
874     }
875     {
876     \msg_warning:nxx
877     { spath3 }
878     { unknown path construction }
879     { \l__spath_tmpe_tl }
880     }
881
882     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
883     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
884     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
885
886     }
887   }
888
889   \bool_if:NT \l__spath_closed_bool
890   {
891     \tl_put_right:NV \l__spath_tmpd_tl \c_spath_closepath_tl
892     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpd_tl}
893     \tl_put_right:Nx \l__spath_tmpd_tl
894     {
895     { \tl_head:N \l__spath_tmpd_tl }
896     { \tl_head:N \l__spath_tmpe_tl }
897     }
898   }
899
900   \bool_set_false:N \l__spath_closed_bool
901   \bool_if:NF \l__spath_rect_bool
902   {
903     \tl_put_left:NV \l__spath_tmpd_tl \c_spath_moveto_tl
904   }
905 }
906
907 \tl_put_left:NV \l__spath_tmpb_tl \l__spath_tmpd_tl

```

```

908 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
909 \group_end:
910 }
911 \cs_new_protected_nopar:Npn \spath_reverse:Nn #1#2
912 {
913   \__spath_reverse:n {#2}
914   \tl_set_eq:NN #1 \g__spath_output_tl
915   \tl_gclear:N \g__spath_output_tl
916 }
917 \cs_generate_variant:Nn \spath_reverse:Nn {NV, cn, cV, Nv}
918 \cs_new_protected_nopar:Npn \spath_reverse:N #1
919 {
920   \spath_reverse:NV #1#1
921 }
922 \cs_generate_variant:Nn \spath_reverse:N {c}
923 \cs_new_protected_nopar:Npn \spath_greverse:Nn #1#2
924 {
925   \__spath_reverse:n {#2}
926   \tl_gset_eq:NN #1 \g__spath_output_tl
927   \tl_gclear:N \g__spath_output_tl
928 }
929 \cs_generate_variant:Nn \spath_greverse:Nn {NV, cn, cV, Nv}
930 \cs_new_protected_nopar:Npn \spath_greverse:N #1
931 {
932   \spath_greverse:NV #1#1
933 }
934 \cs_generate_variant:Nn \spath_greverse:N {c}

```

(End definition for \spath_reverse:Nn and \spath_greverse:Nn.)

\spath_initialaction:Nn
\spath_ginitialaction:Nn

This is the first thing that the path does (after the initial move).

```

935 \cs_new_protected_nopar:Npn \__spath_initialaction:n #1
936 {
937   \group_begin:
938   \tl_clear:N \l__spath_tmpa_tl
939   \tl_set:Nx \l__spath_tmpb_tl {\tl_head:n {#1}}
940   \tl_if_eq:NNTF \l__spath_tmpb_tl \c_spath_rectcorner_tl
941   {
942     \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_rectcorner_tl
943   }
944   {
945     \int_compare:nT
946     {
947       \tl_count:n {#1} > 3
948     }
949     {
950       \tl_set:Nx \l__spath_tmpa_tl
951       {
952         \tl_item:Nn {#1} {4}
953       }
954     }
955   }
956   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
957   \group_end:

```

```

958 }
959 \cs_new_protected_nopar:Npn \spath_initialaction:Nn #1#2
960 {
961   \__spath_initialaction:n {#2}
962   \tl_set_eq:NN #1 \g__spath_output_tl
963   \tl_gclear:N \g__spath_output_tl
964 }
965 \cs_generate_variant:Nn \spath_initialaction:Nn {NV}
966 \cs_new_protected_nopar:Npn \spath_ginitialaction:Nn #1#2
967 {
968   \__spath_initialaction:n {#2}
969   \tl_gset_eq:NN #1 \g__spath_output_tl
970   \tl_gclear:N \g__spath_output_tl
971 }
972 \cs_generate_variant:Nn \spath_ginitialaction:Nn {NV}

```

(End definition for \spath_initialaction:Nn and \spath_ginitialaction:Nn.)

\spath_finalaction:Nn
\spath_gfinalaction:Nn

This is the last thing that the path does.

```

973 \cs_new_protected_nopar:Npn \__spath_finalaction:n #1
974 {
975   \group_begin:
976   \tl_clear:N \l__spath_tmpb_tl
977   \int_compare:nT
978   {
979     \tl_count:n {#1} > 3
980   }
981   {
982     \tl_set:Nn \l__spath_tmpa_tl {#1}
983     \tl_reverse:N \l__spath_tmpa_tl
984     \tl_set:Nx \l__spath_tmpb_tl
985     {
986       \tl_item:Nn \l__spath_tmpa_tl {3}
987     }
988     \tl_if_eq:NNT \l__spath_tmpb_tl \c_spath_curvetoa_tl
989     {
990       \tl_set_eq:NN \l__spath_tmpb_tl \c_spath_curveto_tl
991     }
992     \tl_if_eq:NNT \l__spath_tmpb_tl \c_spath_rectsize_tl
993     {
994       \tl_set_eq:NN \l__spath_tmpb_tl \c_spath_rectcorner_tl
995     }
996   }
997   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
998   \group_end:
999 }
1000 \cs_new_protected_nopar:Npn \spath_finalaction:Nn #1#2
1001 {
1002   \__spath_finalaction:n {#2}
1003   \tl_set_eq:NN #1 \g__spath_output_tl
1004   \tl_gclear:N \g__spath_output_tl
1005 }
1006 \cs_generate_variant:Nn \spath_finalaction:Nn {NV}
1007 \cs_new_protected_nopar:Npn \spath_gfinalaction:Nn #1#2

```

```

1008 {
1009   \__spath_finalaction:n {#2}
1010   \tl_gset_eq:NN #1 \g__spath_output_tl
1011   \tl_gclear:N \g__spath_output_tl
1012 }
1013 \cs_generate_variant:Nn \spath_gfinalaction:Nn {NV}

```

(End definition for \spath_finalaction:Nn and \spath_gfinalaction:Nn.)

\spath_minbb:Nn This computes the minimum (bottom left) of the bounding box of the path.

```

\spath_gminbb:Nn
1014 \cs_new_protected_nopar:Npn \__spath_minbb:n #1
1015 {
1016   \group_begin:
1017   \tl_set:Nn \l__spath_tmpa_tl {#1}
1018
1019   \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
1020   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1021
1022   \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
1023   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1024
1025   \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
1026   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1027
1028   \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl
1029   {
1030     \dim_set_eq:NN \l__spath_rectx_dim \l__spath_tmpa_dim
1031     \dim_set_eq:NN \l__spath_recty_dim \l__spath_tmpb_dim
1032   }
1033   \bool_until_do:nn {
1034     \tl_if_empty_p:N \l__spath_tmpa_tl
1035   }
1036   {
1037     \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
1038     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1039
1040     \tl_if_eq:NNTF \l__spath_tmpc_tl \c_spath_rectsize_tl
1041     {
1042       \dim_set:Nn \l__spath_tmpa_dim
1043       {\dim_min:nn
1044         {\tl_head:N \l__spath_tmpa_tl + \l__spath_rectx_dim} {\l__spath_tmpa_dim}
1045       }
1046       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1047
1048       \dim_set:Nn \l__spath_tmpb_dim
1049       {\dim_min:nn
1050         {\tl_head:N \l__spath_tmpa_tl + \l__spath_recty_dim} {\l__spath_tmpb_dim}
1051       }
1052       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1053     }
1054     {
1055       \dim_set:Nn \l__spath_tmpa_dim
1056       {\dim_min:nn {\tl_head:N \l__spath_tmpa_tl} {\l__spath_tmpa_dim}}
1057       \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl

```

```

1058     {
1059       \dim_set:Nn \l__spath_rectx_dim {\tl_head:N \l__spath_tmpa_tl}
1060     }
1061     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1062
1063     \dim_set:Nn \l__spath_tmpb_dim
1064     {\dim_min:nn {\tl_head:N \l__spath_tmpa_tl} {\l__spath_tmpb_dim}}
1065     \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl
1066     {
1067       \dim_set:Nn \l__spath_recty_dim {\tl_head:N \l__spath_tmpa_tl}
1068     }
1069     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1070   }
1071
1072 }
1073 \tl_clear:N \l__spath_tmpb_tl
1074 \tl_put_right:Nx \l__spath_tmpb_tl
1075 {
1076   {\dim_use:N \l__spath_tmpa_dim}
1077   {\dim_use:N \l__spath_tmpb_dim}
1078 }
1079 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
1080 \group_end:
1081 }
1082 \cs_new_protected_nopar:Npn \spath_minbb:Nn #1#2
1083 {
1084   \__spath_minbb:n {#2}
1085   \tl_set_eq:NN #1 \g__spath_output_tl
1086   \tl_gclear:N \g__spath_output_tl
1087 }
1088 \cs_generate_variant:Nn \spath_minbb:Nn {NV, cn, cV}
1089 \cs_new_protected_nopar:Npn \spath_gminbb:Nn #1#2
1090 {
1091   \__spath_minbb:n {#2}
1092   \tl_gset_eq:NN #1 \g__spath_output_tl
1093   \tl_gclear:N \g__spath_output_tl
1094 }
1095 \cs_generate_variant:Nn \spath_gminbb:Nn {NV, cn, cV}

```

(End definition for \spath_minbb:Nn and \spath_gminbb:Nn.)

`\spath_maxbb:Nn` This computes the maximum (top right) of the bounding box of the path.

```

\spath_gmaxbb:Nn
1096 \cs_new_protected_nopar:Npn \__spath_maxbb:n #1
1097 {
1098   \group_begin:
1099   \tl_set:Nn \l__spath_tmpa_tl {#1}
1100
1101   \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
1102   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1103
1104   \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
1105   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1106
1107   \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}

```



```

1108 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1109
1110 \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl
1111 {
1112   \dim_set_eq:NN \l__spath_rectx_dim \l__spath_tmpa_dim
1113   \dim_set_eq:NN \l__spath_recty_dim \l__spath_tmpb_dim
1114 }
1115 \bool_until_do:nn {
1116   \tl_if_empty_p:N \l__spath_tmpa_tl
1117 }
1118 {
1119   \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
1120   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1121
1122   \tl_if_eq:NNTF \l__spath_tmpc_tl \c_spath_rectsize_tl
1123   {
1124     \dim_set:Nn \l__spath_tmpa_dim
1125     {\dim_max:nn
1126      {\tl_head:N \l__spath_tmpa_tl + \l__spath_rectx_dim} {\l__spath_tmpa_dim}}
1127   }
1128   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1129
1130   \dim_set:Nn \l__spath_tmpb_dim
1131   {\dim_max:nn
1132    {\tl_head:N \l__spath_tmpa_tl + \l__spath_recty_dim} {\l__spath_tmpb_dim}}
1133   }
1134   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1135 }
1136 {
1137   \dim_set:Nn \l__spath_tmpa_dim
1138   {\dim_max:nn {\tl_head:N \l__spath_tmpa_tl} {\l__spath_tmpa_dim}}
1139   \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl
1140   {
1141     \dim_set:Nn \l__spath_rectx_dim {\tl_head:N \l__spath_tmpa_tl}
1142   }
1143   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1144
1145   \dim_set:Nn \l__spath_tmpb_dim
1146   {\dim_max:nn {\tl_head:N \l__spath_tmpa_tl} {\l__spath_tmpb_dim}}
1147   \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl
1148   {
1149     \dim_set:Nn \l__spath_recty_dim {\tl_head:N \l__spath_tmpa_tl}
1150   }
1151   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1152 }
1153
1154 }
1155 \tl_clear:N \l__spath_tmpb_tl
1156 \tl_set:Nx \l__spath_tmpb_tl
1157 {
1158   {\dim_use:N \l__spath_tmpa_dim}
1159   {\dim_use:N \l__spath_tmpb_dim}
1160 }
1161 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl

```

```

1162 \group_end:
1163 }
1164 \cs_new_protected_nopar:Npn \spath_maxbb:Nn #1#2
1165 {
1166   \__spath_maxbb:n {#2}
1167   \tl_set_eq:NN #1 \g__spath_output_tl
1168   \tl_gclear:N \g__spath_output_tl
1169 }
1170 \cs_generate_variant:Nn \spath_maxbb:Nn {NV, cn, cV}
1171 \cs_new_protected_nopar:Npn \spath_gmaxbb:Nn #1#2
1172 {
1173   \__spath_maxbb:n {#2}
1174   \tl_gset_eq:NN #1 \g__spath_output_tl
1175   \tl_gclear:N \g__spath_output_tl
1176 }
1177 \cs_generate_variant:Nn \spath_gmaxbb:Nn {NV, cn, cV}

```

(End definition for \spath_maxbb:Nn and \spath_gmaxbb:Nn.)

`\spath_save_to_aux:Nn` This saves a soft path to the auxfile. The first argument is the macro that will be assigned to the soft path when the aux file is read back in.

```

\spath_save_to_aux:N
1178 \int_set:Nn \l__spath_tmpa_int {\char_value_catcode:n {'@}}
1179 \char_set_catcode_letter:N @
1180 \cs_new_protected_nopar:Npn \spath_save_to_aux:Nn #1#2 {
1181   \tl_if_empty:nF {#2}
1182   {
1183     \tl_clear:N \l__spath_tmpa_tl
1184     \tl_put_right:Nn \l__spath_tmpa_tl {
1185       \ExplSyntaxOn
1186       \tl_clear_new:N #1
1187       \tl_set:Nn #1 {#2}
1188       \ExplSyntaxOff
1189     }
1190     \protected@write\@auxout{}{
1191       \tl_to_str:N \l__spath_tmpa_tl
1192     }
1193   }
1194 }
1195 \char_set_catcode:nn {'@} {\l__spath_tmpa_int}
1196 \cs_generate_variant:Nn \spath_save_to_aux:Nn {cn, cV, NV}
1197 \cs_new_protected_nopar:Npn \spath_save_to_aux:N #1
1198 {
1199   \tl_if_exist:NT #1
1200   {
1201     \spath_save_to_aux:NV #1#1
1202   }
1203 }
1204 \cs_generate_variant:Nn \spath_save_to_aux:N {c}

```

(End definition for \spath_save_to_aux:Nn and \spath_save_to_aux:N.)

3.3 Path Manipulation

These functions all manipulate a soft path. They come with a variety of different argument specifications. As a general rule, the first argument is the macro in which to store

the modified path, the second is the path to manipulate, and the rest are the information about what to do. There is always a variant in which the path is specified by a macro and restored back in that same macro.

```

\spath_translate:Nnnn Translates a path by an amount.
\spath_translate:Nnn 1205 \cs_new_protected_nopar:Npn \__spath_translate:nnn #1#2#3
\spath_gtranslate:Nnnn 1206 {
\spath_gtranslate:Nnn 1207   \group_begin:
1208   \tl_set:Nn \l__spath_tmpa_tl {#1}
1209   \tl_clear:N \l__spath_tmpb_tl
1210   \bool_until_do:nn {
1211     \tl_if_empty_p:N \l__spath_tmpa_tl
1212   }
1213   {
1214     \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
1215
1216     \tl_put_right:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
1217     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1218
1219     \tl_if_eq:NNTF \l__spath_tmpc_tl \c_spath_rectsize_tl
1220     {
1221       \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
1222     }
1223     {
1224       \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl + #2}
1225     }
1226     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1227
1228     \tl_if_eq:NNTF \l__spath_tmpc_tl \c_spath_rectsize_tl
1229     {
1230       \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
1231     }
1232     {
1233       \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl + #3}
1234     }
1235     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1236
1237     \tl_put_right:Nx \l__spath_tmpb_tl
1238     {
1239       {\dim_use:N \l__spath_tmpa_dim}
1240       {\dim_use:N \l__spath_tmpb_dim}
1241     }
1242   }
1243   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
1244   \group_end:
1245 }
1246 \cs_generate_variant:Nn \__spath_translate:nnn {nVV}
1247 \cs_new_protected_nopar:Npn \spath_translate:Nnnn #1#2#3#4
1248 {
1249   \__spath_translate:nnn {#2}{#3}{#4}
1250   \tl_set_eq:NN #1 \g__spath_output_tl
1251   \tl_gclear:N \g__spath_output_tl
1252 }
1253 \cs_generate_variant:Nn \spath_translate:Nnnn {NVxx, NVVV, NVnn}

```

```

1254 \cs_new_protected_nopar:Npn \spath_translate:Nnn #1#2#3
1255 {
1256   \spath_translate:NVnn #1#1{#2}{#3}
1257 }
1258 \cs_generate_variant:Nn \spath_translate:Nnn {NVV, cnn, cVV}
1259 \cs_new_protected_nopar:Npn \spath_gtranslate:Nnnn #1#2#3#4
1260 {
1261   \__spath_translate:nnn {#2}{#3}{#4}
1262   \tl_gset_eq:NN #1 \g__spath_output_tl
1263   \tl_gclear:N \g__spath_output_tl
1264 }
1265 \cs_generate_variant:Nn \spath_gtranslate:Nnnn {NVxx, NVVV, NVnn}
1266 \cs_new_protected_nopar:Npn \spath_gtranslate:Nnn #1#2#3
1267 {
1268   \spath_gtranslate:NVnn #1#1{#2}{#3}
1269 }
1270 \cs_generate_variant:Nn \spath_gtranslate:Nnn {NVV, cnn, cVV}

```

This variant allows for passing the coordinates as a single braced group as it strips off the outer braces of the second argument.

```

1271 \cs_new_protected_nopar:Npn \spath_translate:Nn #1#2
1272 {
1273   \spath_translate:Nnn #1 #2
1274 }
1275 \cs_generate_variant:Nn \spath_translate:Nn {NV}
1276 \cs_new_protected_nopar:Npn \spath_gtranslate:Nn #1#2
1277 {
1278   \spath_gtranslate:Nnn #1 #2
1279 }
1280 \cs_generate_variant:Nn \spath_gtranslate:Nn {NV}

```

(End definition for \spath_translate:Nnnn and others.)

```

\spath_translate_to:Nnnn Translates a path so that it starts at a point.
\spath_translate_to:Nnn
\spath_gtranslate_to:Nnnn
\spath_gtranslate_to:Nnn
1281 \cs_new_protected_nopar:Npn \__spath_translate_to:nnn #1#2#3
1282 {
1283   \group_begin:
1284   \spath_initialpoint:Nn \l__spath_tmpa_tl {#1}
1285
1286   \dim_set:Nn \l__spath_tmpa_dim
1287   {
1288     #2
1289     -
1290     \tl_item:Nn \l__spath_tmpa_tl {1}
1291   }
1292   \dim_set:Nn \l__spath_tmpb_dim
1293   {
1294     #3
1295     -
1296     \tl_item:Nn \l__spath_tmpa_tl {2}
1297   }
1298
1299   \__spath_translate:nVV {#1} \l__spath_tmpa_dim \l__spath_tmpb_dim
1300   \group_end:
1301 }

```

```

1302 \cs_new_protected_nopar:Npn \spath_translate_to:Nnnn #1#2#3#4
1303 {
1304   \__spath_translate_to:nnn {#2}{#3}{#4}
1305   \tl_set_eq:NN #1 \g__spath_output_tl
1306   \tl_gclear:N \g__spath_output_tl
1307 }
1308 \cs_generate_variant:Nn \spath_translate_to:Nnnn {NVxx, NVVV, NVnn}
1309 \cs_new_protected_nopar:Npn \spath_translate_to:Nnn #1#2#3
1310 {
1311   \spath_translate_to:NVnn #1#1{#2}{#3}
1312 }
1313 \cs_generate_variant:Nn \spath_translate_to:Nnn {NVV, cnn, cVV}
1314 \cs_new_protected_nopar:Npn \spath_gtranslate_to:Nnnn #1#2#3#4
1315 {
1316   \__spath_translate_to:nnn {#2}{#3}{#4}
1317   \tl_gset_eq:NN #1 \g__spath_output_tl
1318   \tl_gclear:N \g__spath_output_tl
1319 }
1320 \cs_generate_variant:Nn \spath_gtranslate_to:Nnnn {NVxx, NVVV, NVnn}
1321 \cs_new_protected_nopar:Npn \spath_gtranslate_to:Nnn #1#2#3
1322 {
1323   \spath_gtranslate_to:NVnn #1#1{#2}{#3}
1324 }
1325 \cs_generate_variant:Nn \spath_gtranslate_to:Nnn {NVV, cnn, cVV}

```

This variant allows for passing the coordinates as a single braced group as it strips off the outer braces of the second argument.

```

1326 \cs_new_protected_nopar:Npn \spath_translate_to:Nn #1#2
1327 {
1328   \spath_translate_to:Nnn #1 #2
1329 }
1330 \cs_generate_variant:Nn \spath_translate_to:Nn {NV}
1331 \cs_new_protected_nopar:Npn \spath_gtranslate_to:Nn #1#2
1332 {
1333   \spath_gtranslate_to:Nnn #1 #2
1334 }
1335 \cs_generate_variant:Nn \spath_gtranslate_to:Nn {NV}

```

(End definition for \spath_translate_to:Nnnn and others.)

```

\spath_scale:Nnnn Scale a path.
\spath_scale:Nnn 1336 \cs_new_protected_nopar:Npn \__spath_scale:nnn #1#2#3
\spath_gscale:Nnnn 1337 {
\spath_gscale:Nnn 1338   \group_begin:
1339   \tl_set:Nn \l__spath_tmpa_tl {#1}
1340   \tl_clear:N \l__spath_tmpb_tl
1341   \bool_until_do:nn {
1342     \tl_if_empty_p:N \l__spath_tmpa_tl
1343   }
1344   {
1345     \tl_put_right:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
1346     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1347
1348     \fp_set:Nn \l__spath_tmpa_fp {\tl_head:N \l__spath_tmpa_tl * #2}
1349     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}

```

```

1350
1351     \fp_set:Nn \l__spath_tmpb_fp {\tl_head:N \l__spath_tmpa_tl * #3}
1352     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1353
1354     \tl_put_right:Nx \l__spath_tmpb_tl
1355     {
1356         {\fp_to_dim:N \l__spath_tmpa_fp}
1357         {\fp_to_dim:N \l__spath_tmpb_fp}
1358     }
1359 }
1360 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
1361 \group_end:
1362 }
1363 \cs_new_protected_nopar:Npn \spath_scale:Nnnn #1#2#3#4
1364 {
1365     \__spath_scale:nnn {#2}{#3}{#4}
1366     \tl_set_eq:NN #1 \g__spath_output_tl
1367     \tl_gclear:N \g__spath_output_tl
1368 }
1369 \cs_generate_variant:Nn \spath_scale:Nnnn {NVnn, Nnxx}
1370 \cs_new_protected_nopar:Npn \spath_scale:Nnn #1#2#3
1371 {
1372     \spath_scale:NVnn #1#1{#2}{#3}
1373 }
1374 \cs_generate_variant:Nn \spath_scale:Nnn {cnn, cVV, NVV}
1375 \cs_new_protected_nopar:Npn \spath_gscale:Nnnn #1#2#3#4
1376 {
1377     \__spath_scale:nnn {#2}{#3}{#4}
1378     \tl_gset_eq:NN #1 \g__spath_output_tl
1379     \tl_gclear:N \g__spath_output_tl
1380 }
1381 \cs_generate_variant:Nn \spath_gscale:Nnnn {NVnn, Nnxx}
1382 \cs_new_protected_nopar:Npn \spath_gscale:Nnn #1#2#3
1383 {
1384     \spath_gscale:NVnn #1#1{#2}{#3}
1385 }
1386 \cs_generate_variant:Nn \spath_gscale:Nnn {cnn, cVV, NVV}

```

This variant allows for passing the coordinates as a single braced group as it strips off the outer braces of the second argument.

```

1387 \cs_new_protected_nopar:Npn \spath_scale:Nn #1#2
1388 {
1389     \spath_scale:Nnn #1 #2
1390 }
1391
1392 \cs_generate_variant:Nn \spath_scale:Nn {NV}
1393 \cs_new_protected_nopar:Npn \spath_gscale:Nn #1#2
1394 {
1395     \spath_gscale:Nnn #1 #2
1396 }
1397
1398 \cs_generate_variant:Nn \spath_gscale:Nn {NV}

```

(End definition for `\spath_scale:Nnnn` and others.)

`\spath_transform:Nnnnnnnn` Applies an affine (matrix and vector) transformation to path. The matrix is specified in rows first.

```

\spath_transform:Nnnnnnnn
\spath_gtransform:Nnnnnnnn
\spath_gtransform:Nnnnnnnn
1399 \cs_new_protected_nopar:Npn \__spath_transform:nnnnnnn #1#2#3#4#5#6#7
1400 {
1401   \group_begin:
1402   \tl_set:Nn \l__spath_tmpa_tl {#1}
1403   \tl_clear:N \l__spath_tmpb_tl
1404   \bool_until_do:nn {
1405     \tl_if_empty_p:N \l__spath_tmpa_tl
1406   }
1407   {
1408     \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpa_tl}
1409
1410     \tl_put_right:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
1411     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpa_tl}
1412     \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpa_tl}
1413     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpa_tl}
1414     \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpa_tl}
1415     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpa_tl}
1416
1417     \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_rectsize_tl
1418     {
1419       \fp_set:Nn \l__spath_tmpe_fp
1420       {\l__spath_tmpe_tl * #2 + \l__spath_tmpe_tl * #4}
1421       \fp_set:Nn \l__spath_tmpe_fp
1422       {\l__spath_tmpe_tl * #3 + \l__spath_tmpe_tl * #5}
1423     }
1424     {
1425       \fp_set:Nn \l__spath_tmpe_fp
1426       {\l__spath_tmpe_tl * #2 + \l__spath_tmpe_tl * #4 + #6}
1427       \fp_set:Nn \l__spath_tmpe_fp
1428       {\l__spath_tmpe_tl * #3 + \l__spath_tmpe_tl * #5 + #7}
1429     }
1430
1431     \tl_put_right:Nx \l__spath_tmpb_tl
1432     {
1433       {\fp_to_dim:N \l__spath_tmpe_fp}
1434       {\fp_to_dim:N \l__spath_tmpe_fp}
1435     }
1436   }
1437
1438   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
1439   \group_end:
1440 }
1441 \cs_new_protected_nopar:Npn \spath_transform:Nnnnnnnn #1#2#3#4#5#6#7#8
1442 {
1443   \__spath_transform:nnnnnnn {#2}{#3}{#4}{#5}{#6}{#7}{#8}
1444   \tl_set_eq:NN #1 \g__spath_output_tl
1445   \tl_gclear:N \g__spath_output_tl
1446 }
1447 \cs_generate_variant:Nn \spath_transform:Nnnnnnnn
1448 {NVnnnnnnn, Nxxxxxxx, cnnnnnnn}
1449 \cs_new_protected_nopar:Npn \spath_transform:Nnnnnnnn #1#2#3#4#5#6#7
1450 {

```

```

1451 \spath_transform:NVnnnnnn #1#1{#2}{#3}{#4}{#5}{#6}{#7}
1452 }
1453 \cs_generate_variant:Nn \spath_transform:Nnnnnnn {cnnnnnn}
1454 \cs_new_protected_nopar:Npn \spath_transform:Nnn #1#2#3
1455 {
1456 \spath_transform:Nnnnnnn #1{#2}#3
1457 }
1458 \cs_generate_variant:Nn \spath_transform:Nnn {cnn, cVn, NVn, NnV}
1459 \cs_new_protected_nopar:Npn \spath_transform:Nn #1#2
1460 {
1461 \spath_transform:NVnnnnnn #1#1#2
1462 }
1463 \cs_generate_variant:Nn \spath_transform:Nn {cn, cV, NV}
1464
1465 \cs_new_protected_nopar:Npn \spath_gtransform:Nnnnnnn #1#2#3#4#5#6#7#8
1466 {
1467 \__spath_transform:nnnnnn {#2}{#3}{#4}{#5}{#6}{#7}{#8}
1468 \tl_gset_eq:NN #1 \g__spath_output_tl
1469 \tl_gclear:N \g__spath_output_tl
1470 }
1471 \cs_generate_variant:Nn \spath_gtransform:Nnnnnnn {NVnnnnnn, Nnxxxxxx, cnnnnnn}
1472 \cs_new_protected_nopar:Npn \spath_gtransform:Nnnnnnn #1#2#3#4#5#6#7
1473 {
1474 \spath_gtransform:NVnnnnnn #1#1{#2}{#3}{#4}{#5}{#6}{#7}
1475 }
1476 \cs_generate_variant:Nn \spath_gtransform:Nnnnnnn {cnnnnnn}
1477 \cs_new_protected_nopar:Npn \spath_gtransform:Nnn #1#2#3
1478 {
1479 \spath_gtransform:Nnnnnnn #1{#2}#3
1480 }
1481 \cs_generate_variant:Nn \spath_gtransform:Nnn {cnn, cVn, NVn, NnV}
1482 \cs_new_protected_nopar:Npn \spath_gtransform:Nn #1#2
1483 {
1484 \spath_gtransform:NVnnnnnn #1#1#2
1485 }
1486 \cs_generate_variant:Nn \spath_gtransform:Nn {cn, cV, NV}

```

(End definition for \spath_transform:Nnnnnnn and others.)

<pre> \spath_span:Nnnn \spath_span:Nnn \spath_gspan:Nnnn \spath_gspan:Nnn \spath_normalise:Nn \spath_normalise:N \spath_gnormalise:Nn \spath_gnormalise:N </pre>	<p>The <code>span</code> functions transform a path to start and end at specified points. The <code>normalise</code> functions transform it to start at the origin and end at (1pt,0pt).</p> <p>If the path starts and ends at the same point then it is translated to the specified point (or origin) but not otherwise changed.</p> <pre> 1487 \cs_new_protected_nopar:Npn __spath_span:nnn #1#2#3 1488 { 1489 \group_begin: 1490 \spath_initialpoint:Nn \l__spath_tmpa_tl {#1} 1491 \spath_finalpoint:Nn \l__spath_tmpb_tl {#1} 1492 1493 \fp_set:Nn \l__spath_tmpa_fp 1494 { 1495 (\tl_item:Nn \l__spath_tmpb_tl {1}) - 1496 (\tl_item:Nn \l__spath_tmpa_tl {1}) 1497 } </pre>
--	---


```

1498 \fp_set:Nn \l__spath_tmpb_fp
1499 {
1500   (\tl_item:Nn \l__spath_tmpb_tl {2}) -
1501   (\tl_item:Nn \l__spath_tmpa_tl {2})
1502 }
1503 \fp_set:Nn \l__spath_tmpc_fp
1504 {
1505   (\l__spath_tmpa_fp) * (\l__spath_tmpa_fp)
1506   +
1507   (\l__spath_tmpb_fp * \l__spath_tmpb_fp)
1508 }
1509
1510 \fp_compare:nTF
1511 {
1512   \l__spath_tmpc_fp < 0.001
1513 }
1514 {
1515   \spath_translate_to:Nnnn \l__spath_tmpd_tl {#1} #2
1516 }
1517 {
1518   \fp_set:Nn \l__spath_tmpa_fp
1519   {
1520     (
1521       ((\tl_item:nn {#3} {1})
1522       -
1523       (\tl_item:nn {#2} {1}))
1524     *
1525     ((\tl_item:Nn \l__spath_tmpb_tl {1})
1526     -
1527     (\tl_item:Nn \l__spath_tmpa_tl {1}))
1528     +
1529     ((\tl_item:nn {#3} {2})
1530     -
1531     (\tl_item:nn {#2} {2}))
1532     *
1533     ((\tl_item:Nn \l__spath_tmpb_tl {2})
1534     -
1535     (\tl_item:Nn \l__spath_tmpa_tl {2}))
1536     )
1537     /
1538     \l__spath_tmpc_fp
1539   }
1540   \fp_set:Nn \l__spath_tmpb_fp
1541   {
1542     (
1543       ((\tl_item:nn {#3} {2})
1544       -
1545       (\tl_item:nn {#2} {2}))
1546     *
1547     ((\tl_item:Nn \l__spath_tmpb_tl {1})
1548     -
1549     (\tl_item:Nn \l__spath_tmpa_tl {1}))
1550     -
1551     ((\tl_item:nn {#3} {1})

```

```

1552     -
1553     (\tl_item:nn {#2} {1}))
1554     *
1555     ((\tl_item:Nn \l__spath_tmpb_tl {2})
1556     -
1557     (\tl_item:Nn \l__spath_tmpa_tl {2}))
1558     )
1559     /
1560     \l__spath_tmpc_fp
1561 }
1562
1563 \tl_set:Nx \l__spath_tmpc_tl
1564 {
1565   {
1566     \fp_to_decimal:N \l__spath_tmpa_fp
1567   }
1568   {
1569     \fp_to_decimal:N \l__spath_tmpb_fp
1570   }
1571   {
1572     \fp_eval:n { - \l__spath_tmpb_fp }
1573   }
1574   {
1575     \fp_to_decimal:N \l__spath_tmpa_fp
1576   }
1577   {
1578     \fp_to_dim:n
1579     {
1580       \tl_item:nn {#2} {1}
1581       -
1582       \l__spath_tmpa_fp * (\tl_item:Nn \l__spath_tmpa_tl {1})
1583       +
1584       \l__spath_tmpb_fp * (\tl_item:Nn \l__spath_tmpa_tl {2})
1585     }
1586   }
1587   {
1588     \fp_to_dim:n
1589     {
1590       \tl_item:nn {#2} {2}
1591       -
1592       \l__spath_tmpb_fp * (\tl_item:Nn \l__spath_tmpa_tl {1})
1593       -
1594       \l__spath_tmpa_fp * (\tl_item:Nn \l__spath_tmpa_tl {2})
1595     }
1596   }
1597 }
1598 \spath_transform:NnV \l__spath_tmpd_tl {#1} \l__spath_tmpc_tl
1599 }
1600 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpd_tl
1601 \group_end:
1602 }
1603 \cs_new_protected_nopar:Npn \spath_span:Nnnn #1#2#3#4
1604 {
1605   \__spath_span:nnn {#2}{#3}{#4}

```

```

1606 \tl_set_eq:NN #1 \g__spath_output_tl
1607 \tl_gclear:N \g__spath_output_tl
1608 }
1609 \cs_generate_variant:Nn \spath_span:Nnnn {NVnn, NVVV, NnVV}
1610 \cs_new_protected_nopar:Npn \spath_span:Nnn #1#2#3
1611 {
1612   \spath_span:NVnn #1#1{#2}{#3}
1613 }
1614 \cs_generate_variant:Nn \spath_span:Nnn {NVV, cnn, cvv, cVV}
1615 \cs_new_protected_nopar:Npn \spath_gspan:Nnnn #1#2#3#4
1616 {
1617   \__spath_span:nnn {#2}{#3}{#4}
1618   \tl_gset_eq:NN #1 \g__spath_output_tl
1619   \tl_gclear:N \g__spath_output_tl
1620 }
1621 \cs_generate_variant:Nn \spath_gspan:Nnnn {NVnn, NVVV}
1622 \cs_new_protected_nopar:Npn \spath_gspan:Nnn #1#2#3
1623 {
1624   \spath_gspan:NVnn #1#1{#2}{#3}
1625 }
1626 \cs_generate_variant:Nn \spath_gspan:Nnn {NVV, cnn, cvv, cVV}
1627 \cs_new_protected_nopar:Npn \__spath_normalise:n #1
1628 {
1629   \__spath_span:nnn {#1}{0pt}{0pt}{{1pt}{0pt}}
1630 }
1631 \cs_new_protected_nopar:Npn \spath_normalise:Nn #1#2
1632 {
1633   \__spath_normalise:n {#2}
1634   \tl_set_eq:NN #1 \g__spath_output_tl
1635   \tl_gclear:N \g__spath_output_tl
1636 }
1637 \cs_generate_variant:Nn \spath_normalise:Nn {cn,NV, cV, cv}
1638 \cs_new_protected_nopar:Npn \spath_normalise:N #1
1639 {
1640   \spath_normalise:NV #1#1
1641 }
1642 \cs_generate_variant:Nn \spath_normalise:N {c}
1643 \cs_new_protected_nopar:Npn \spath_gnormalise:Nn #1#2
1644 {
1645   \__spath_normalise:n {#2}
1646   \tl_gset_eq:NN #1 \g__spath_output_tl
1647   \tl_gclear:N \g__spath_output_tl
1648 }
1649 \cs_generate_variant:Nn \spath_gnormalise:Nn {cn,NV, cV, cv}
1650 \cs_new_protected_nopar:Npn \spath_gnormalise:N #1
1651 {
1652   \spath_gnormalise:NV #1#1
1653 }
1654 \cs_generate_variant:Nn \spath_gnormalise:N {c}

```

(End definition for \spath_span:Nnnn and others.)

\spath_splice_between:Nnnn This takes three paths and returns a single path in which the middle one is adjusted (and welded) so that it joins the first path to the third.
\spath_splice_between:Nnn
\spath_gsplice_between:Nnnn
\spath_gsplice_between:Nnn

```

1655 \cs_new_protected_nopar:Npn \__spath_splice_between:nnn #1#2#3
1656 {
1657   \group_begin:
1658   \spath_finalpoint:Nn \l__spath_tmpd_tl {#1}
1659   \spath_initialpoint:Nn \l__spath_tmpe_tl {#3}
1660   \spath_span:NnVV \l__spath_tmpb_tl {#2} \l__spath_tmpd_tl \l__spath_tmpe_tl
1661   \spath_append_no_move:NnV \l__spath_tmpe_tl {#1} \l__spath_tmpb_tl
1662   \spath_append_no_move:Nn \l__spath_tmpe_tl {#3}
1663   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
1664   \group_end:
1665 }
1666 \cs_new_protected_nopar:Npn \spath_splice_between:Nnnn #1#2#3#4
1667 {
1668   \__spath_splice_between:nnn {#2}{#3}{#4}
1669   \tl_set_eq:NN #1 \g__spath_output_tl
1670   \tl_gclear:N \g__spath_output_tl
1671 }
1672 \cs_generate_variant:Nn \spath_splice_between:Nnnn {NVnn, NVVV}
1673 \cs_new_protected_nopar:Npn \spath_splice_between:Nnn #1#2#3
1674 {
1675   \spath_splice_between:NVnn #1#1{#2}{#3}
1676 }
1677 \cs_generate_variant:Nn \spath_splice_between:Nnn {NVV, cnn, cvv, Nvn, NVn}
1678 \cs_new_protected_nopar:Npn \spath_gsplice_between:Nnnn #1#2#3#4
1679 {
1680   \__spath_splice_between:nnn {#2}{#3}{#4}
1681   \tl_gset_eq:NN #1 \g__spath_output_tl
1682   \tl_gclear:N \g__spath_output_tl
1683 }
1684 \cs_generate_variant:Nn \spath_gsplice_between:Nnnn {NVnn, NVVV}
1685 \cs_new_protected_nopar:Npn \spath_gsplice_between:Nnn #1#2#3
1686 {
1687   \spath_gsplice_between:NVnn #1#1{#2}{#3}
1688 }
1689 \cs_generate_variant:Nn \spath_gsplice_between:Nnn {NVV, cnn, cvv, Nvn, NVn}

```

(End definition for \spath_splice_between:Nnnn and others.)

`\spath_hobby_curve:Nnnnn` Construct the curve from Hobby's algorithm given the start, end, and tangent directions.

```

1690 \cs_new_protected_nopar:Npn \__spath_hobby_curve:nnnn #1#2#3#4
1691 {
1692   \group_begin:

```

First tangent vector projected onto vector between endpoints
 Something a bit weird here as the components are opposite to how I thought they should be ...

```

1693   \fp_set:Nn \l__spath_tmpe_fp
1694   {
1695     (
1696       (\tl_item:nn {#2} {1})
1697       *
1698       (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})
1699       +
1700       (\tl_item:nn {#2} {2})
1701       *

```

```

1702     (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})
1703     )
1704     /
1705     sqrt
1706     (
1707     (
1708     (\tl_item:nn {#2} {1})^2
1709     +
1710     (\tl_item:nn {#2} {2})^2
1711     )
1712     *
1713     (
1714     (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})^2
1715     +
1716     (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})^2
1717     )
1718     )
1719     }
1720 \fp_set:Nn \l__spath_tmpb_fp
1721 {
1722     (
1723     -
1724     (\tl_item:nn {#2} {1})
1725     *
1726     (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})
1727     +
1728     (\tl_item:nn {#2} {2})
1729     *
1730     (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})
1731     )
1732     /
1733     sqrt
1734     (
1735     (
1736     (\tl_item:nn {#2} {1})^2
1737     +
1738     (\tl_item:nn {#2} {2})^2
1739     )
1740     *
1741     (
1742     (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})^2
1743     +
1744     (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})^2
1745     )
1746     )
1747     }

```

Second tangent vector projected onto vector between endpoints

```

1748 \fp_set:Nn \l__spath_tmpe_fp
1749 {
1750     (
1751     (\tl_item:nn {#3} {1})
1752     *
1753     (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})
1754     +

```

```

1755 (\tl_item:nn {#3} {2})
1756 *
1757 (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})
1758 )
1759 /
1760 sqrt
1761 (
1762 (
1763 (\tl_item:nn {#3} {1})^2
1764 +
1765 (\tl_item:nn {#3} {2})^2
1766 )
1767 *
1768 (
1769 (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})^2
1770 +
1771 (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})^2
1772 )
1773 )
1774 }
1775 \fp_set:Nn \l__spath_tmpd_fp
1776 {
1777 (
1778 (\tl_item:nn {#3} {1})
1779 *
1780 (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})
1781 -
1782 (\tl_item:nn {#3} {2})
1783 *
1784 (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})
1785 )
1786 /
1787 sqrt
1788 (
1789 (
1790 (\tl_item:nn {#3} {1})^2
1791 +
1792 (\tl_item:nn {#3} {2})^2
1793 )
1794 *
1795 (
1796 (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})^2
1797 +
1798 (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})^2
1799 )
1800 )
1801 }
1802
1803 \fp_set:Nn \l__spath_tmpe_fp
1804 {
1805 (
1806 2 + sqrt(2) *
1807 (\l__spath_tmpe_fp - 1/16 * \l__spath_tmpd_fp)
1808 *

```

```

1809 (\l__spath_tmpd_fp - 1/16 * \l__spath_tmpb_fp)
1810 *
1811 (\l__spath_tmpa_fp - \l__spath_tmpc_fp)
1812 )
1813 /
1814 (
1815 1
1816 +
1817 (1 - (3 - sqrt(5))/2)
1818 *
1819 \l__spath_tmpa_fp
1820 +
1821 (3 - sqrt(5))/2
1822 *
1823 \l__spath_tmpc_fp
1824 )
1825 *
1826 sqrt
1827 (
1828 (
1829 (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})^2
1830 +
1831 (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})^2
1832 )
1833 /
1834 (
1835 (\tl_item:nn {#2} {1})^2
1836 +
1837 (\tl_item:nn {#2} {2})^2
1838 )
1839 )
1840 /3
1841 }
1842 \fp_set:Nn \l__spath_tmpf_fp
1843 {
1844 (
1845 2 - sqrt(2) *
1846 (\l__spath_tmpb_fp - 1/16 * \l__spath_tmpd_fp)
1847 *
1848 (\l__spath_tmpd_fp - 1/16 * \l__spath_tmpb_fp)
1849 *
1850 (\l__spath_tmpa_fp - \l__spath_tmpc_fp)
1851 )
1852 /
1853 (
1854 1
1855 +
1856 (1 - (3 - sqrt(5))/2)
1857 *
1858 \l__spath_tmpc_fp
1859 +
1860 (3 - sqrt(5))/2
1861 *
1862 \l__spath_tmpa_fp

```

```

1863 )
1864 *
1865 sqrt
1866 (
1867 (
1868 (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})^2
1869 +
1870 (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})^2
1871 )
1872 /
1873 (
1874 (\tl_item:nn {#3} {1})^2
1875 +
1876 (\tl_item:nn {#3} {2})^2
1877 )
1878 )
1879 /3
1880 }
1881
1882 \tl_set:Nx \l__spath_tmpa_tl
1883 {
1884 {
1885 \fp_to_dim:n
1886 {
1887 \tl_item:nn {#1} {1}
1888 +
1889 \l__spath_tmpe_fp
1890 *
1891 (\tl_item:nn {#2} {1})
1892 }
1893 }
1894 {
1895 \fp_to_dim:n
1896 {
1897 \tl_item:nn {#1} {2}
1898 +
1899 \l__spath_tmpe_fp
1900 *
1901 (\tl_item:nn {#2} {2})
1902 }
1903 }
1904 }
1905 \tl_set:Nx \l__spath_tmpb_tl
1906 {
1907 {
1908 \fp_to_dim:n
1909 {
1910 \tl_item:nn {#4} {1}
1911 -
1912 \l__spath_tmpf_fp
1913 *
1914 (\tl_item:nn {#3} {1})
1915 }
1916 }

```



```

1917 {
1918   \fp_to_dim:n
1919   {
1920     \tl_item:nn {#4} {2}
1921     -
1922     \l__spath_tmpf_fp
1923     *
1924     (\tl_item:nn {#3} {2})
1925   }
1926 }
1927 }
1928
1929 \tl_clear:N \l__spath_tmpc_tl
1930 \tl_set:NV \l__spath_tmpc_tl \c_spath_moveto_tl
1931 \tl_put_right:Nn \l__spath_tmpc_tl {#1}
1932 \tl_put_right:NV \l__spath_tmpc_tl \c_spath_curvetoa_tl
1933 \tl_put_right:NV \l__spath_tmpc_tl \l__spath_tmpa_tl
1934 \tl_put_right:NV \l__spath_tmpc_tl \c_spath_curvetob_tl
1935 \tl_put_right:NV \l__spath_tmpc_tl \l__spath_tmpb_tl
1936 \tl_put_right:NV \l__spath_tmpc_tl \c_spath_curveto_tl
1937 \tl_put_right:Nn \l__spath_tmpc_tl {#4}
1938 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpc_tl
1939 \group_end:
1940 }
1941 \cs_new_protected_nopar:Npn \spath_hobby_curve:Nnnnn #1#2#3#4#5
1942 {
1943   \__spath_hobby_curve:nnnn {#2}{#3}{#4}{#5}
1944   \tl_set_eq:NN #1 \g__spath_output_tl
1945   \tl_gclear:N \g__spath_output_tl
1946 }
1947 \cs_generate_variant:Nn \spath_hobby_curve:Nnnnn {NVVVV}
1948 \cs_new_protected_nopar:Npn \spath_ghobby_curve:Nnnnn #1#2#3#4#5
1949 {
1950   \__spath_hobby_curve:nnnn {#2}{#3}{#4}{#5}
1951   \tl_gset_eq:NN #1 \g__spath_output_tl
1952   \tl_gclear:N \g__spath_output_tl
1953 }
1954 \cs_generate_variant:Nn \spath_ghobby_curve:Nnnnn {NVVVV}

```

(End definition for \spath_hobby_curve:Nnnnn.)

\spath_curve_between:Nnn
\spath_curve_between:Nn
\spath_gcurve_between:Nnn
\spath_gcurve_between:Nn

This takes two paths and returns a single path formed by joining the two paths by a curve.

```

1955 \cs_new_protected_nopar:Npn \__spath_curve_between:nn #1#2
1956 {
1957   \group_begin:
1958   \spath_finalpoint:Nn \l__spath_tmpa_tl {#1}
1959   \spath_finalgangent:Nn \l__spath_tmpb_tl {#1}
1960   \spath_initialpoint:Nn \l__spath_tmpc_tl {#2}
1961   \spath_initialtangent:Nn \l__spath_tmpd_tl {#2}
1962
1963   \spath_hobby_curve:NVVVV \l__spath_tmpe_tl
1964   \l__spath_tmpa_tl \l__spath_tmpb_tl \l__spath_tmpd_tl \l__spath_tmpe_tl
1965 }

```

```

1966 \tl_set:Nn \l__spath_tmpa_tl {#1}
1967 \spath_append_no_move:NV \l__spath_tmpa_tl \l__spath_tmpe_tl
1968 \spath_append_no_move:Nn \l__spath_tmpa_tl {#2}
1969 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
1970 \group_end:
1971 }
1972 \cs_new_protected_nopar:Npn \spath_curve_between:Nnn #1#2#3
1973 {
1974   \__spath_curve_between:nn {#2}{#3}
1975   \tl_set_eq:NN #1 \g__spath_output_tl
1976   \tl_gclear:N \g__spath_output_tl
1977 }
1978 \cs_generate_variant:Nn \spath_curve_between:Nnn {NVn, NVV}
1979 \cs_new_protected_nopar:Npn \spath_curve_between:Nn #1#2
1980 {
1981   \spath_curve_between:NVn #1#1{#2}
1982 }
1983 \cs_generate_variant:Nn \spath_curve_between:Nn {NV, cn, cv}
1984 \cs_new_protected_nopar:Npn \spath_gcurve_between:Nnn #1#2#3
1985 {
1986   \__spath_curve_between:nn {#2}
1987   \tl_gset_eq:NN #1 \g__spath_output_tl
1988   \tl_gclear:N \g__spath_output_tl
1989 }
1990 \cs_generate_variant:Nn \spath_gcurve_between:Nnn {NVn, NVV}
1991 \cs_new_protected_nopar:Npn \spath_gcurve_between:Nn #1#2
1992 {
1993   \spath_gcurve_between:NVnn #1#1{#2}
1994 }
1995 \cs_generate_variant:Nn \spath_gcurve_between:Nn {NV, cn, cv}

```

(End definition for `\spath_curve_between:Nnn` and others.)

`\spath_close_with:Nn` Closes the first path by splicing in the second.

```

\spath_gclose_with:Nn
1996 \cs_new_protected_nopar:Npn \__spath_close_with:nn #1#2
1997 {
1998   \group_begin:
1999   \spath_finalmovepoint:Nn \l__spath_tmpa_tl {#1}
2000   \spath_finalpoint:Nn \l__spath_tmpb_tl {#1}
2001   \dim_compare:nTF
2002   {
2003     \dim_abs:n
2004     {
2005       \tl_item:Nn \l__spath_tmpa_tl {1}
2006       -
2007       \tl_item:Nn \l__spath_tmpb_tl {1}
2008     }
2009     +
2010     \dim_abs:n
2011     {
2012       \tl_item:Nn \l__spath_tmpa_tl {2}
2013       -
2014       \tl_item:Nn \l__spath_tmpb_tl {2}
2015     }

```

```

2016 < 0.01pt
2017 }
2018 {
2019   \__spath_close:n {#1}
2020 }
2021 {
2022   \spath_span:NnVV \l__spath_tmpc_tl {#2} \l__spath_tmpb_tl \l__spath_tmpa_tl
2023   \spath_append_no_move:NnV \l__spath_tmpd_tl {#1} \l__spath_tmpc_tl
2024   \__spath_close:V \l__spath_tmpd_tl
2025 }
2026 \group_end:
2027 }
2028 \cs_new_protected_nopar:Npn \spath_close_with:Nnn #1#2#3
2029 {
2030   \__spath_close_with:nn {#2}{#3}
2031   \tl_set_eq:NN #1 \g__spath_output_tl
2032   \tl_gclear:N \g__spath_output_tl
2033 }
2034 \cs_generate_variant:Nn \spath_close_with:Nnn {cnn, cVV, cvv, NVn}
2035 \cs_new_protected_nopar:Npn \spath_close_with:Nn #1#2
2036 {
2037   \spath_close_with:NVn #1#1{#2}
2038 }
2039 \cs_generate_variant:Nn \spath_close_with:Nn {cn, cV, cv, NV}
2040 \cs_new_protected_nopar:Npn \spath_gclose_with:Nnn #1#2#3
2041 {
2042   \__spath_close_with:nn {#2}{#3}
2043   \tl_gset_eq:NN #1 \g__spath_output_tl
2044   \tl_gclear:N \g__spath_output_tl
2045 }
2046 \cs_generate_variant:Nn \spath_gclose_with:Nnn {cnn, cVV, cvv, NVn}
2047 \cs_new_protected_nopar:Npn \spath_gclose_with:Nn #1#2
2048 {
2049   \spath_gclose_with:NVn #1#1{#2}
2050 }
2051 \cs_generate_variant:Nn \spath_gclose_with:Nn {cn, cV, cv, NV}

```

(End definition for \spath_close_with:Nn and \spath_gclose_with:Nn.)

\spath_close_with_curve:N Closes the path with a curve.

```

\spath_gclose_with_curve:N
2052 \cs_new_protected_nopar:Npn \__spath_close_with_curve:n #1
2053 {
2054   \group_begin:
2055   \spath_finalpoint:Nn \l__spath_tmpa_tl {#1}
2056   \spath_finaltangent:Nn \l__spath_tmpb_tl {#1}
2057   \spath_finalmovepoint:Nn \l__spath_tmpc_tl {#1}
2058   \spath_finalmovetangent:Nn \l__spath_tmpd_tl {#1}
2059   \dim_compare:nTF
2060   {
2061     \dim_abs:n
2062     {
2063       \tl_item:Nn \l__spath_tmpa_tl {1}
2064       -
2065       \tl_item:Nn \l__spath_tmpc_tl {1}

```

```

2066     }
2067     +
2068     \dim_abs:n
2069     {
2070       \tl_item:Nn \l__spath_tmpa_tl {2}
2071       -
2072       \tl_item:Nn \l__spath_tmpc_tl {2}
2073     }
2074     < 0.01pt
2075   }
2076   {
2077     \__spath_close:n {#1}
2078   }
2079   {
2080
2081     \spath_hobby_curve:NVVVV \l__spath_tmpe_tl
2082     \l__spath_tmpa_tl \l__spath_tmpb_tl \l__spath_tmpd_tl \l__spath_tmpc_tl
2083
2084     \tl_set:Nn \l__spath_tmpa_tl {#1}
2085     \spath_append_no_move:NV \l__spath_tmpa_tl \l__spath_tmpe_tl
2086     \__spath_close:V \l__spath_tmpa_tl
2087   }
2088   \group_end:
2089 }
2090 \cs_new_protected_nopar:Npn \spath_close_with_curve:Nn #1#2
2091 {
2092   \__spath_close_with_curve:n {#2}
2093   \tl_set_eq:NN #1 \g__spath_output_tl
2094   \tl_gclear:N \g__spath_output_tl
2095 }
2096 \cs_generate_variant:Nn \spath_close_with_curve:Nn {cn, cV, cv, NV}
2097 \cs_new_protected_nopar:Npn \spath_close_with_curve:N #1
2098 {
2099   \spath_close_with_curve:NV #1#1
2100 }
2101 \cs_generate_variant:Nn \spath_close_with_curve:N {c}
2102 \cs_new_protected_nopar:Npn \spath_gclose_with_curve:Nn #1#2
2103 {
2104   \__spath_close_with_curve:n {#2}
2105   \tl_gset_eq:NN #1 \g__spath_output_tl
2106   \tl_gclear:N \g__spath_output_tl
2107 }
2108 \cs_generate_variant:Nn \spath_gclose_with_curve:Nn {cn, cV, cv, NV}
2109 \cs_new_protected_nopar:Npn \spath_gclose_with_curve:N #1
2110 {
2111   \spath_gclose_with_curve:NV #1#1
2112 }
2113 \cs_generate_variant:Nn \spath_gclose_with_curve:N {c}

```

(End definition for `\spath_close_with_curve:N` and `\spath_gclose_with_curve:N`.)

`\spath_weld:Nnn` This welds one path to another, moving the second so that its initial point coincides with the first's final point. It is called a *weld* because the initial move of the second path is removed.

`\spath_weld:Nn`

`\spath_gweld:Nnn`

`\spath_gweld:Nn`

```

2114 \cs_new_protected_nopar:Npn \__spath_weld:nn #1#2
2115 {
2116   \group_begin:
2117   \tl_set:Nn \l__spath_tmpa_tl {#1}
2118   \tl_set:Nn \l__spath_tmpb_tl {#2}
2119   \spath_finalpoint:Nn \l__spath_tmpe_tl {#1}
2120   \spath_translate_to:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2121
2122   \__spath_append_no_move:VV \l__spath_tmpa_tl \l__spath_tmpb_tl
2123   \group_end:
2124 }
2125 \cs_new_protected_nopar:Npn \spath_weld:Nnn #1#2#3
2126 {
2127   \__spath_weld:nn {#2}{#3}
2128   \tl_set_eq:NN #1 \g__spath_output_tl
2129   \tl_gclear:N \g__spath_output_tl
2130 }
2131 \cs_generate_variant:Nn \spath_weld:Nnn {NVV,NVn}
2132 \cs_new_protected_nopar:Npn \spath_weld:Nn #1#2
2133 {
2134   \spath_weld:NVn #1#1{#2}
2135 }
2136 \cs_generate_variant:Nn \spath_weld:Nn {NV, Nv, cV, cv}
2137 \cs_new_protected_nopar:Npn \spath_gweld:Nnn #1#2#3
2138 {
2139   \__spath_weld:nn {#2}{#3}
2140   \tl_gset_eq:NN #1 \g__spath_output_tl
2141   \tl_gclear:N \g__spath_output_tl
2142 }
2143 \cs_generate_variant:Nn \spath_gweld:Nnn {NVV, NVn}
2144 \cs_new_protected_nopar:Npn \spath_gweld:Nn #1#2
2145 {
2146   \spath_gweld:NVn #1#1{#2}
2147 }
2148 \cs_generate_variant:Nn \spath_gweld:Nn {NV, Nv, cV, cv}

```

(End definition for `\spath_weld:Nnn` and others.)

`\spath_append_no_move:Nnn` Append the path from the second `spath` to the first, removing the adjoining move if neither path has a rectangle either side of the join or if the first path isn't closed.

```

\spath_append_no_move:Nn
\spath_gappend_no_move:Nnn
\spath_gappend_no_move:Nn
2149 \cs_new_protected_nopar:Npn \__spath_append_no_move:nn #1#2
2150 {
2151   \group_begin:
2152   \tl_set:Nn \l__spath_tmpa_tl {#1}
2153   \tl_set:Nn \l__spath_tmpb_tl {#2}
2154   \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpb_tl}
2155   \spath_finalaction:Nn \l__spath_tmpe_tl {#1}
2156   \bool_if:nT {
2157     ! \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_closepath_tl
2158     &&
2159     ! \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_rectcorner_tl
2160     &&
2161     \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_moveto_tl
2162   }

```

```

2163 {
2164   \tl_set:Nx \l__spath_tmpb_tl {\tl_tail:N \l__spath_tmpb_tl}
2165   \tl_set:Nx \l__spath_tmpb_tl {\tl_tail:N \l__spath_tmpb_tl}
2166   \tl_set:Nx \l__spath_tmpb_tl {\tl_tail:N \l__spath_tmpb_tl}
2167 }
2168
2169 \tl_put_right:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
2170 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
2171 \group_end:
2172 }
2173 \cs_generate_variant:Nn \__spath_append_no_move:nn {VV}
2174 \cs_new_protected_nopar:Npn \spath_append_no_move:Nnn #1#2#3
2175 {
2176   \__spath_append_no_move:nn {#2}{#3}
2177   \tl_set_eq:NN #1 \g__spath_output_tl
2178   \tl_gclear:N \g__spath_output_tl
2179 }
2180 \cs_generate_variant:Nn \spath_append_no_move:Nnn {NVV, NVn, NnV}
2181 \cs_new_protected_nopar:Npn \spath_append_no_move:Nn #1#2
2182 {
2183   \spath_append_no_move:NVn #1#1{#2}
2184 }
2185 \cs_generate_variant:Nn \spath_append_no_move:Nn {NV, cv, Nv, cV}
2186 \cs_new_protected_nopar:Npn \spath_gappend_no_move:Nnn #1#2#3
2187 {
2188   \__spath_append_no_move:nn {#2}{#3}
2189   \tl_gset_eq:NN #1 \g__spath_output_tl
2190   \tl_gclear:N \g__spath_output_tl
2191 }
2192 \cs_generate_variant:Nn \spath_gappend_no_move:Nnn {NVV, NVn}
2193 \cs_new_protected_nopar:Npn \spath_gappend_no_move:Nn #1#2
2194 {
2195   \spath_gappend_no_move:NVn #1#1{#2}
2196 }
2197 \cs_generate_variant:Nn \spath_gappend_no_move:Nn {NV, cv, Nv, cV}

```

(End definition for `\spath_append_no_move:Nnn` and others.)

```

\spath_append:Nnn Prepend the path from the second spath to the first.
\spath_append:Nn 2198 \cs_new_protected_nopar:Npn \spath_append:Nnn #1#2#3
\spath_gappend:Nnn 2199 {
\spath_gappend:Nn 2200   \tl_set:Nn #1 {#2}
2201   \tl_put_right:Nn #1 {#3}
2202 }
2203 \cs_generate_variant:Nn \spath_append:Nnn {NVV, NVn}
2204 \cs_new_protected_nopar:Npn \spath_append:Nn #1#2
2205 {
2206   \spath_append:NVn #1#1{#2}
2207 }
2208 \cs_generate_variant:Nn \spath_append:Nn {NV, Nv, cv, cV}
2209 \cs_new_protected_nopar:Npn \spath_gappend:Nnn #1#2#3
2210 {
2211   \tl_gset:Nn #1 {#2}
2212   \tl_gput_right:Nn #1 {#3}

```

```

2213 }
2214 \cs_generate_variant:Nn \spath_gappend:Nnn {NVV, NVn}
2215 \cs_new_protected_nopar:Npn \spath_gappend:Nn #1#2
2216 {
2217   \spath_gappend:NVn #1#1{#2}
2218 }
2219 \cs_generate_variant:Nn \spath_gappend:Nn {NV, Nv, cv, cV}

```

(End definition for `\spath_append:Nnn` and others.)

```

\spath_prepend_no_move:Nnn Prepend the path from the second spath to the first, removing the adjoining move.
\spath_prepend_no_move:Nn 2220 \cs_new_protected_nopar:Npn \spath_prepend_no_move:Nnn #1#2#3
\spath_gprepend_no_move:Nnn 2221 {
\spath_gprepend_no_move:Nn 2222   \spath_append_no_move:Nnn #1{#3}{#2}

```

```

2223 }
2224 \cs_generate_variant:Nn \spath_prepend_no_move:Nnn {NVV, NVn}
2225 \cs_new_protected_nopar:Npn \spath_prepend_no_move:Nn #1#2
2226 {
2227   \spath_prepend_no_move:NVn #1#1{#2}
2228 }
2229 \cs_generate_variant:Nn \spath_prepend_no_move:Nn {NV, cv}
2230 \cs_new_protected_nopar:Npn \spath_gprepend_no_move:Nnn #1#2#3
2231 {
2232   \spath_gappend_no_move:Nnn #1{#3}{#2}
2233 }
2234 \cs_generate_variant:Nn \spath_gprepend_no_move:Nnn {NVV, NVn}
2235 \cs_new_protected_nopar:Npn \spath_gprepend_no_move:Nn #1#2
2236 {
2237   \spath_gprepend_no_move:NVn #1#1{#2}
2238 }
2239 \cs_generate_variant:Nn \spath_gprepend_no_move:Nn {NV, cv}

```

(End definition for `\spath_prepend_no_move:Nnn` and others.)

```

\spath_prepend:Nnn Prepend the path from the second spath to the first.
\spath_prepend:Nn 2240 \cs_new_protected_nopar:Npn \spath_prepend:Nnn #1#2#3
\spath_gprepend:Nnn 2241 {
\spath_gprepend:Nn 2242   \spath_append:Nnn #1{#3}{#2}
2243 }
2244 \cs_generate_variant:Nn \spath_prepend:Nnn {NVV, NVn}
2245 \cs_new_protected_nopar:Npn \spath_prepend:Nn #1#2
2246 {
2247   \spath_prepend:NVn #1#1{#2}
2248 }
2249 \cs_generate_variant:Nn \spath_prepend:Nn {NV}
2250 \cs_new_protected_nopar:Npn \spath_gprepend:Nnn #1#2#3
2251 {
2252   \spath_gappend:Nnn #1{#3}{#2}
2253 }
2254 \cs_generate_variant:Nn \spath_gprepend:Nnn {NVV, NVn}
2255 \cs_new_protected_nopar:Npn \spath_gprepend:Nn #1#2
2256 {
2257   \spath_gprepend:NVn #1#1{#2}
2258 }
2259 \cs_generate_variant:Nn \spath_gprepend:Nn {NV}

```

(End definition for `\spath_prepend:Nnn` and others.)

`\spath_bake_round:Nn` The corner rounding routine is applied quite late in the process of building a soft path so this ensures that it is done.

```

\spath_bake_round:N
\spath_gbake_round:Nn
\spath_gbake_round:N
2260 \cs_new_protected_nopar:Npn \__spath_bake_round:n #1
2261 {
2262   \group_begin:
2263   \tl_set:Nn \l__spath_tmpa_tl {#1}
2264   \pgf@processround \l__spath_tmpa_tl\l__spath_tmpb_tl
2265   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
2266   \group_end:
2267 }
2268 \cs_new_protected_nopar:Npn \spath_bake_round:Nn #1#2
2269 {
2270   \__spath_bake_round:n {#2}
2271   \tl_set_eq:NN #1 \g__spath_output_tl
2272   \tl_gclear:N \g__spath_output_tl
2273 }
2274 \cs_generate_variant:Nn \spath_bake_round:Nn {NV}
2275 \cs_new_protected_nopar:Npn \spath_bake_round:N #1
2276 {
2277   \spath_bake_round:NV #1#1
2278 }
2279 \cs_generate_variant:Nn \spath_bake_round:N {c}
2280 \cs_new_protected_nopar:Npn \spath_gbake_round:Nn #1#2
2281 {
2282   \__spath_bake_round:n {#2}
2283   \tl_gset_eq:NN #1 \g__spath_output_tl
2284   \tl_gclear:N \g__spath_output_tl
2285 }
2286 \cs_generate_variant:Nn \spath_gbake_round:Nn {NV}
2287 \cs_new_protected_nopar:Npn \spath_gbake_round:N #1
2288 {
2289   \spath_gbake_round:NV #1#1
2290 }
2291 \cs_generate_variant:Nn \spath_gbake_round:N {c}

```

(End definition for `\spath_bake_round:Nn` and others.)

`\spath_bake_shorten:Nn` The shortening routine is applied quite late in the process of building a soft path so this ensures that it is done.

```

\spath_bake_shorten:N
\spath_gbake_shorten:Nn
\spath_gbake_shorten:N
2292 \cs_new_protected_nopar:Npn \__spath_bake_shorten:n #1
2293 {
2294   \group_begin:
2295   \tl_set:Nn \l__spath_tmpa_tl {#1}
2296   \pgfsyssoftpath@getcurrentpath\l__spath_tmpb_tl
2297   \pgfsyssoftpath@setcurrentpath\l__spath_tmpa_tl
2298   \pgf@prepare@end@of@path
2299   \pgf@prepare@start@of@path
2300   \pgfsyssoftpath@getcurrentpath\l__spath_tmpa_tl
2301   \pgfsyssoftpath@setcurrentpath\l__spath_tmpb_tl
2302   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
2303   \group_end:
2304 }

```



```

2305 \cs_new_protected_nopar:Npn \spath_bake_shorten:Nn #1#2
2306 {
2307   \__spath_bake_shorten:n {#2}
2308   \tl_set_eq:NN #1 \g__spath_output_tl
2309   \tl_gclear:N \g__spath_output_tl
2310 }
2311 \cs_generate_variant:Nn \spath_bake_shorten:Nn {NV}
2312 \cs_new_protected_nopar:Npn \spath_bake_shorten:N #1
2313 {
2314   \spath_bake_shorten:NV #1#1
2315 }
2316 \cs_generate_variant:Nn \spath_bake_shorten:N {c}
2317 \cs_new_protected_nopar:Npn \spath_gbake_shorten:Nn #1#2
2318 {
2319   \__spath_bake_shorten:n {#2}
2320   \tl_gset_eq:NN #1 \g__spath_output_tl
2321   \tl_gclear:N \g__spath_output_tl
2322 }
2323 \cs_generate_variant:Nn \spath_gbake_shorten:Nn {NV}
2324 \cs_new_protected_nopar:Npn \spath_gbake_shorten:N #1
2325 {
2326   \spath_gbake_shorten:NV #1#1
2327 }
2328 \cs_generate_variant:Nn \spath_gbake_shorten:N {c}

```

(End definition for \spath_bake_shorten:Nn and others.)

```

\spath_close:Nn Appends a close path to the end of the path.
\spath_close:N
\spath_gclose:Nn
\spath_gclose:N
2329 \cs_new_protected_nopar:Npn \__spath_close:n #1
2330 {
2331   \group_begin:
2332   \tl_set:Nn \l__spath_tmpa_tl {#1}
2333   \spath_finalmovepoint:NV \l__spath_tmpb_tl \l__spath_tmpa_tl
2334   \tl_put_right:NV \l__spath_tmpa_tl \c_spath_closepath_tl
2335   \tl_put_right:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
2336   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
2337   \group_end:
2338 }
2339 \cs_generate_variant:Nn \__spath_close:n {V}
2340 \cs_new_protected_nopar:Npn \spath_close:Nn #1#2
2341 {
2342   \__spath_close:n {#2}
2343   \tl_set_eq:NN #1 \g__spath_output_tl
2344   \tl_gclear:N \g__spath_output_tl
2345 }
2346 \cs_generate_variant:Nn \spath_close:Nn {NV}
2347 \cs_new_protected_nopar:Npn \spath_close:N #1
2348 {
2349   \spath_close:NV #1#1
2350 }
2351 \cs_generate_variant:Nn \spath_close:N {c}
2352 \cs_new_protected_nopar:Npn \spath_gclose:Nn #1#2
2353 {
2354   \__spath_close:n {#2}

```

```

2355 \tl_gset_eq:NN #1 \g__spath_output_tl
2356 \tl_gclear:N \g__spath_output_tl
2357 }
2358 \cs_generate_variant:Nn \spath_gclose:Nn {NV}
2359 \cs_new_protected_nopar:Npn \spath_gclose:N #1
2360 {
2361   \spath_gclose:NV #1#1
2362 }
2363 \cs_generate_variant:Nn \spath_gclose:N {c}

```

(End definition for `\spath_close:Nn` and others.)

`\spath_adjust_close:Nn` This closes a path and adjusts the end point to be where the final move point (so where the close points to) is. The intention is that this should be used if the two points are visually the same point but mathematically different.

```

\spath_adjust_close:Nn
\spath_adjust_close:N
\spath_adjust_gclose:Nn
\spath_adjust_gclose:N
2364 \cs_new_protected_nopar:Npn \__spath_adjust_close:n #1
2365 {
2366   \group_begin:
2367   \tl_set:Nn \l__spath_tmpa_tl {#1}
2368   \spath_finalmovepoint:NV \l__spath_tmpb_tl \l__spath_tmpa_tl
2369   \spath_finalpoint:NV \l__spath_tmpe_tl \l__spath_tmpa_tl
2370   \tl_reverse:N \l__spath_tmpa_tl
2371   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2372   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2373   \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpa_tl}
2374   \tl_if_eq:NNT \l__spath_tmpe_tl \c_spath_curveto_tl
2375   {
2376     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2377     \tl_clear:N \l__spath_tmpe_tl
2378     \tl_set:Nx \l__spath_tmpe_tl {
2379       {
2380         \dim_eval:n
2381         {
2382           \tl_item:Nn \l__spath_tmpa_tl {1}
2383           -
2384           \tl_item:Nn \l__spath_tmpe_tl {2}
2385           +
2386           \tl_item:Nn \l__spath_tmpe_tl {2}
2387         }
2388       }
2389     {
2390       \dim_eval:n
2391       {
2392         \tl_item:Nn \l__spath_tmpa_tl {2}
2393         -
2394         \tl_item:Nn \l__spath_tmpe_tl {1}
2395         +
2396         \tl_item:Nn \l__spath_tmpe_tl {1}
2397       }
2398     }
2399   }
2400   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2401   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2402   \tl_put_left:NV \l__spath_tmpa_tl \l__spath_tmpe_tl

```

```

2403     \tl_put_left:NV \l__spath_tmpa_tl \l__spath_tmpd_tl
2404   }
2405   \tl_reverse:N \l__spath_tmpa_tl
2406   \tl_put_right:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
2407   \tl_put_right:NV \l__spath_tmpa_tl \c_spath_closepath_tl
2408   \tl_put_right:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
2409   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
2410   \group_end:
2411 }
2412 \cs_generate_variant:Nn \__spath_adjust_close:n {V}
2413 \cs_new_protected_nopar:Npn \spath_adjust_close:Nn #1#2
2414 {
2415   \__spath_adjust_close:n {#2}
2416   \tl_set_eq:NN #1 \g__spath_output_tl
2417   \tl_gclear:N \g__spath_output_tl
2418 }
2419 \cs_generate_variant:Nn \spath_adjust_close:Nn {NV}
2420 \cs_new_protected_nopar:Npn \spath_adjust_close:N #1
2421 {
2422   \spath_adjust_close:NV #1#1
2423 }
2424 \cs_generate_variant:Nn \spath_adjust_close:N {c}
2425 \cs_new_protected_nopar:Npn \spath_adjust_gclosen:Nn #1#2
2426 {
2427   \__spath_adjust_close:n {#2}
2428   \tl_gset_eq:NN #1 \g__spath_output_tl
2429   \tl_gclear:N \g__spath_output_tl
2430 }
2431 \cs_generate_variant:Nn \spath_adjust_gclosen:Nn {NV}
2432 \cs_new_protected_nopar:Npn \spath_adjust_gclosen:N #1
2433 {
2434   \spath_adjust_gclosen:NV #1#1
2435 }
2436 \cs_generate_variant:Nn \spath_adjust_gclosen:N {c}

```

(End definition for \spath_adjust_close:Nn and others.)

`\spath_open:Nn` Removes all close paths from the path, replacing them by `lineto` if they move any distance. Rectangles are replaced by lines with the start/end at the lower left corner.

```

\spath_open:N 2437 \cs_new_protected_nopar:Npn \__spath_open:n #1
\spath_gopen:Nn 2438 {
\spath_gopen:N 2439   \group_begin:
2440   \spath_replace_rectangles:Nn \l__spath_tmpa_tl {#1}
2441   \tl_clear:N \l__spath_tmpb_tl
2442   \bool_until_do:nm {
2443     \tl_if_empty_p:N \l__spath_tmpa_tl
2444   }
2445   {
2446     \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
2447     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2448
2449     \tl_case:NnF \l__spath_tmpc_tl
2450     {
2451       \c_spath_closepath_tl {

```

```

2452
2453 \bool_if:nF
2454 {
2455   \dim_compare_p:n
2456   {
2457     \l__spath_move_x_dim == \l__spath_tmpa_dim
2458   }
2459   &&
2460   \dim_compare_p:n
2461   {
2462     \l__spath_move_y_dim == \l__spath_tmpb_dim
2463   }
2464 }
2465 {
2466   \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
2467
2468   \tl_put_right:Nx \l__spath_tmpb_tl {
2469     { \dim_use:N \l__spath_move_x_dim }
2470     { \dim_use:N \l__spath_move_y_dim }
2471   }
2472 }
2473
2474 \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
2475 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2476 \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
2477 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2478 }
2479
2480 \c_spath_moveto_tl {
2481   \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpc_tl
2482
2483   \dim_set:Nn \l__spath_move_x_dim {\tl_head:N \l__spath_tmpa_tl}
2484   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2485   \dim_set:Nn \l__spath_move_y_dim {\tl_head:N \l__spath_tmpa_tl}
2486   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2487
2488   \tl_put_right:Nx \l__spath_tmpb_tl {
2489     { \dim_use:N \l__spath_move_x_dim }
2490     { \dim_use:N \l__spath_move_y_dim }
2491   }
2492
2493   \dim_set_eq:NN \l__spath_tmpa_dim \l__spath_move_x_dim
2494   \dim_set_eq:NN \l__spath_tmpb_dim \l__spath_move_y_dim
2495 }
2496 }
2497 {
2498   \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpc_tl
2499
2500   \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
2501   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2502   \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
2503   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2504
2505   \tl_put_right:Nx \l__spath_tmpb_tl {

```

```

2506         { \dim_use:N \l__spath_tmpa_dim }
2507         { \dim_use:N \l__spath_tmpb_dim }
2508     }
2509 }
2510 }
2511 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
2512 \group_end:
2513 }
2514 \cs_generate_variant:Nn \__spath_open:n {V}
2515 \cs_new_protected_nopar:Npn \spath_open:Nn #1#2
2516 {
2517     \__spath_open:n {#2}
2518     \tl_set_eq:NN #1 \g__spath_output_tl
2519     \tl_gclear:N \g__spath_output_tl
2520 }
2521 \cs_generate_variant:Nn \spath_open:Nn {NV}
2522 \cs_new_protected_nopar:Npn \spath_open:N #1
2523 {
2524     \spath_open:NV #1#1
2525 }
2526 \cs_new_protected_nopar:Npn \spath_gopen:Nn #1#2
2527 {
2528     \__spath_open:n {#2}
2529     \tl_gset_eq:NN #1 \g__spath_output_tl
2530     \tl_gclear:N \g__spath_output_tl
2531 }
2532 \cs_generate_variant:Nn \spath_gopen:Nn {NV}
2533 \cs_new_protected_nopar:Npn \spath_gopen:N #1
2534 {
2535     \spath_gopen:NV #1#1
2536 }

```

(End definition for `\spath_open:Nn` and others.)

```

\spath_replace_lines:Nn Replace any line segments by Bézier curves.
\spath_replace_lines:Nn 2537 \cs_new_protected_nopar:Npn \__spath_replace_lines:n #1
\spath_replace_lines:Nn 2538 {
\spath_replace_lines:Nn 2539     \group_begin:
2540     \tl_set:Nn \l__spath_tmpa_tl {#1}
2541     \tl_clear:N \l__spath_tmpb_tl
2542     \dim_set:Nn \l__spath_tmpa_dim {Opt}
2543     \dim_set:Nn \l__spath_tmpb_dim {Opt}
2544
2545     \bool_do_until:nn
2546     {
2547         \tl_if_empty_p:N \l__spath_tmpa_tl
2548     }
2549     {
2550         \tl_set:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpa_tl {1}}
2551         \tl_set:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpa_tl {2}}
2552         \tl_set:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpa_tl {3}}
2553
2554         \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_lineto_tl
2555         {

```

```

2556 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_curvetoa_tl
2557 \tl_put_right:Nx \l__spath_tmpb_tl
2558 {
2559   {
2560     \fp_to_dim:n
2561     {
2562       2/3 * (\l__spath_tmpa_dim)
2563       +
2564       1/3 * (\l__spath_tmpd_tl)
2565     }
2566   }
2567 }
2568 \tl_put_right:Nx \l__spath_tmpb_tl
2569 {
2570   {
2571     \fp_to_dim:n
2572     {
2573       2/3 * (\l__spath_tmpb_dim)
2574       +
2575       1/3 * (\l__spath_tmpe_tl)
2576     }
2577   }
2578 }
2579 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_curvetob_tl
2580 \tl_put_right:Nx \l__spath_tmpb_tl
2581 {
2582   {
2583     \fp_to_dim:n
2584     {
2585       1/3 * (\l__spath_tmpa_dim)
2586       +
2587       2/3 * (\l__spath_tmpd_tl)
2588     }
2589   }
2590 }
2591 \tl_put_right:Nx \l__spath_tmpb_tl
2592 {
2593   {
2594     \fp_to_dim:n
2595     {
2596       1/3 * (\l__spath_tmpb_dim)
2597       +
2598       2/3 * (\l__spath_tmpe_tl)
2599     }
2600   }
2601 }
2602 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_curveto_tl
2603 \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2604 \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2605 }
2606 {
2607   \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2608   \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2609   \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl

```

```

2610     }
2611
2612     \dim_set:Nn \l__spath_tmpa_dim {\l__spath_tmpd_tl}
2613     \dim_set:Nn \l__spath_tmpb_dim {\l__spath_tmpe_tl}
2614
2615     \prg_replicate:nn {3}
2616     {
2617         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2618     }
2619 }
2620 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
2621 \group_end:
2622 }
2623 \cs_generate_variant:Nn \__spath_replace_lines:n {V}
2624 \cs_new_protected_nopar:Npn \spath_replace_lines:Nn #1#2
2625 {
2626     \__spath_replace_lines:n {#2}
2627     \tl_set_eq:NN #1 \g__spath_output_tl
2628     \tl_gclear:N \g__spath_output_tl
2629 }
2630 \cs_generate_variant:Nn \spath_replace_lines:Nn {NV, cV, cv, Nv}
2631 \cs_new_protected_nopar:Npn \spath_replace_lines:N #1
2632 {
2633     \spath_replace_lines:NV #1#1
2634 }
2635 \cs_generate_variant:Nn \spath_replace_lines:N {c}
2636 \cs_new_protected_nopar:Npn \spath_greplace_lines:Nn #1#2
2637 {
2638     \__spath_replace_lines:n {#2}
2639     \tl_gset_eq:NN #1 \g__spath_output_tl
2640     \tl_gclear:N \g__spath_output_tl
2641 }
2642 \cs_generate_variant:Nn \spath_greplace_lines:Nn {NV, cV, cv, Nv}
2643 \cs_new_protected_nopar:Npn \spath_greplace_lines:N #1
2644 {
2645     \spath_greplace_lines:NV #1#1
2646 }
2647 \cs_generate_variant:Nn \spath_greplace_lines:N {c}

```

(End definition for \spath_replace_lines:Nn.)

```

\spath_replace_rectangles:Nn Replace any rectangle components by lines.
\spath_replace_rectangles:Nn
\spath_replace_rectangles:Nn
\spath_replace_rectangles:Nn
2648 \cs_new_protected_nopar:Npn \__spath_replace_rectangles:n #1
2649 {
2650     \group_begin:
2651     \tl_set:Nn \l__spath_tmpa_tl {#1}
2652     \tl_clear:N \l__spath_tmpb_tl
2653
2654     \bool_do_until:nn
2655     {
2656         \tl_if_empty_p:N \l__spath_tmpa_tl
2657     }
2658     {
2659         \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpa_tl }

```

```

2660 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2661 \tl_set:Nx \l__spath_tmpd_tl {\tl_head:N \l__spath_tmpa_tl }
2662 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2663 \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpa_tl }
2664 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2665
2666 \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_rectcorner_tl
2667 {
2668
2669   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2670   \dim_set:Nn \l__spath_tmpa_dim
2671   {
2672     \tl_item:Nn \l__spath_tmpa_tl {1}
2673   }
2674   \dim_set:Nn \l__spath_tmpb_dim
2675   {
2676     \tl_item:Nn \l__spath_tmpa_tl {2}
2677   }
2678
2679   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2680   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2681
2682   \tl_put_right:NV \l__spath_tmpb_tl \c_spath_moveto_tl
2683   \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2684   \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2685
2686   \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
2687   \tl_put_right:Nx \l__spath_tmpb_tl
2688   {
2689     {
2690       \fp_to_dim:n { \l__spath_tmpd_tl + \l__spath_tmpa_dim }
2691     }
2692   }
2693   \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2694
2695   \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
2696   \tl_put_right:Nx \l__spath_tmpb_tl
2697   {
2698     {
2699       \fp_to_dim:n { \l__spath_tmpd_tl + \l__spath_tmpa_dim }
2700     }
2701   }
2702   \tl_put_right:Nx \l__spath_tmpb_tl
2703   {
2704     {
2705       \fp_to_dim:n { \l__spath_tmpe_tl + \l__spath_tmpb_dim }
2706     }
2707   }
2708
2709   \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
2710   \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2711   \tl_put_right:Nx \l__spath_tmpb_tl
2712   {
2713     {

```



```

2714         \fp_to_dim:n { \l__spath_tmpe_tl + \l__spath_tmpe_dim }
2715     }
2716 }
2717
2718 \tl_put_right:NV \l__spath_tmpe_tl \c_spath_closepath_tl
2719 \__spath_tl_put_right_braced:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2720 \__spath_tl_put_right_braced:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2721
2722 }
2723 {
2724     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2725     \__spath_tl_put_right_braced:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2726     \__spath_tl_put_right_braced:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2727 }
2728 }
2729 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
2730 \group_end:
2731 }
2732 \cs_generate_variant:Nn \__spath_replace_rectangles:n {V}
2733 \cs_new_protected_nopar:Npn \spath_replace_rectangles:Nn #1#2
2734 {
2735     \__spath_replace_rectangles:n {#2}
2736     \tl_set_eq:NN #1 \g__spath_output_tl
2737     \tl_gclear:N \g__spath_output_tl
2738 }
2739 \cs_generate_variant:Nn \spath_replace_rectangles:Nn {NV, cV, cv, Nv}
2740 \cs_new_protected_nopar:Npn \spath_replace_rectangles:N #1
2741 {
2742     \spath_replace_rectangles:NV #1#1
2743 }
2744 \cs_generate_variant:Nn \spath_replace_rectangles:N {c}
2745 \cs_new_protected_nopar:Npn \spath_greplace_rectangles:Nn #1#2
2746 {
2747     \__spath_replace_rectangles:n {#2}
2748     \tl_gset_eq:NN #1 \g__spath_output_tl
2749     \tl_gclear:N \g__spath_output_tl
2750 }
2751 \cs_generate_variant:Nn \spath_greplace_rectangles:Nn {NV, cV, cv, Nv}
2752 \cs_new_protected_nopar:Npn \spath_greplace_rectangles:N #1
2753 {
2754     \spath_greplace_rectangles:NV #1#1
2755 }
2756 \cs_generate_variant:Nn \spath_greplace_rectangles:N {c}

```

(End definition for `\spath_replace_rectangles:Nn`.)

```

\spath_remove_empty_components:Nn Remove any component that is simply a moveto.
\spath_remove_empty_components:N 2757 \cs_new_protected_nopar:Npn \__spath_remove_empty_components:n #1
\spath_gremove_empty_components:Nn 2758 {
\spath_gremove_empty_components:N 2759     \group_begin:
2760     \spath_components_to_seq:Nn \l__spath_tmpe_seq {#1}
2761     \tl_clear:N \l__spath_tmpe_tl
2762     \seq_map_inline:Nn \l__spath_tmpe_seq
2763     {

```

```

2764 \int_compare:nF
2765 {
2766   \tl_count:n {##1} == 3
2767 }
2768 {
2769   \tl_put_right:Nn \l__spath_tmpa_tl {##1}
2770 }
2771 }
2772 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
2773 \group_end:
2774 }
2775 \cs_new_protected_nopar:Npn \spath_remove_empty_components:Nn #1#2
2776 {
2777   \__spath_remove_empty_components:n {#2}
2778   \tl_set_eq:NN #1 \g__spath_output_tl
2779   \tl_gclear:N \g__spath_output_tl
2780 }
2781 \cs_generate_variant:Nn \spath_remove_empty_components:Nn {NV}
2782 \cs_new_protected_nopar:Npn \spath_remove_empty_components:N #1
2783 {
2784   \spath_remove_empty_components:NV #1#1
2785 }
2786 \cs_generate_variant:Nn \spath_remove_empty_components:N {c}
2787 \cs_new_protected_nopar:Npn \spath_gremove_empty_components:Nn #1#2
2788 {
2789   \__spath_remove_empty_components:n {#2}
2790   \tl_gset_eq:NN #1 \g__spath_output_tl
2791   \tl_gclear:N \g__spath_output_tl
2792 }
2793 \cs_generate_variant:Nn \spath_gremove_empty_components:Nn {NV}
2794 \cs_new_protected_nopar:Npn \spath_gremove_empty_components:N #1
2795 {
2796   \spath_gremove_empty_components:NV #1#1
2797 }
2798 \cs_generate_variant:Nn \spath_gremove_empty_components:N {c}

```

(End definition for `\spath_remove_empty_components:Nn` and others.)

`\spath_if_eq:nn` Test if two soft paths are equal, we allow a little tolerance on the calculations.

```

2799 \prg_new_protected_conditional:Npnn \spath_if_eq:nn #1#2 { T, F, TF }
2800 {
2801   \group_begin:
2802   \tl_set:Nn \l__spath_tmpa_tl {#1}
2803   \tl_set:Nn \l__spath_tmpb_tl {#2}
2804   \bool_gset_true:N \g__spath_tmpa_bool
2805   \int_compare:nNnTF
2806   {\tl_count:N \l__spath_tmpa_tl}
2807   =
2808   {\tl_count:N \l__spath_tmpb_tl}
2809   {
2810     \int_step_inline:nnnn {1} {3} {\tl_count:N \l__spath_tmpa_tl}
2811     {
2812       \tl_set:Nx \l__spath_tmpc_tl {\tl_item:Nn \l__spath_tmpa_tl {##1}}
2813       \tl_set:Nx \l__spath_tmpd_tl {\tl_item:Nn \l__spath_tmpb_tl {##1}}

```

```

2814 \tl_if_eq:N NF \l__spath_tmpc_tl \l__spath_tmpd_tl
2815 {
2816   \bool_gset_false:N \g__spath_tmpa_bool
2817 }
2818 \dim_set:Nn \l__spath_tmpa_dim {\tl_item:Nn \l__spath_tmpa_tl {##1+1}}
2819 \dim_set:Nn \l__spath_tmpb_dim {\tl_item:Nn \l__spath_tmpb_tl {##1+1}}
2820 \dim_compare:nF
2821 {
2822   \dim_abs:n
2823   {
2824     \l__spath_tmpa_dim - \l__spath_tmpb_dim
2825   }
2826   < 0.001pt
2827 }
2828 {
2829   \bool_gset_false:N \g__spath_tmpa_bool
2830 }
2831 \dim_set:Nn \l__spath_tmpa_dim {\tl_item:Nn \l__spath_tmpa_tl {##1+2}}
2832 \dim_set:Nn \l__spath_tmpb_dim {\tl_item:Nn \l__spath_tmpb_tl {##1+2}}
2833 \dim_compare:nF
2834 {
2835   \dim_abs:n
2836   {
2837     \l__spath_tmpa_dim - \l__spath_tmpb_dim
2838   }
2839   < 0.001pt
2840 }
2841 {
2842   \bool_gset_false:N \g__spath_tmpa_bool
2843 }
2844 }
2845 }
2846 {
2847   \bool_gset_false:N \g__spath_tmpa_bool
2848 }
2849 \group_end:
2850 \bool_if:N TF \g__spath_tmpa_bool
2851 {
2852   \prg_return_true:
2853 }
2854 {
2855   \prg_return_false:
2856 }
2857 }
2858 \prg_generate_conditional_variant:Nnn \spath_if_eq:nn {VV, Vn, nV, vv} {TF, T, F}

```

(End definition for \spath_if_eq:nn.)

3.4 Splitting Commands

\spath_split_curve:NNnn Splits a Bezier cubic into pieces, storing the pieces in the first two arguments.

```

\spath_gsplit_curve:NNnn
2859 \cs_new_protected_nopar:Npn \__spath_split_curve:nn #1#2
2860 {
2861   \group_begin:

```

```

2862 \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_moveto_tl
2863 \tl_put_right:Nx \l__spath_tmpa_tl {
2864   {\tl_item:nn {#1} {2}}
2865   {\tl_item:nn {#1} {3}}
2866 }
2867 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curvetoa_tl
2868 \tl_put_right:Nx \l__spath_tmpa_tl
2869 {
2870   {\fp_to_dim:n
2871     {
2872       (1 - #2) * \tl_item:nn {#1} {2} + (#2) * \tl_item:nn {#1} {5}
2873     }}
2874   {\fp_to_dim:n
2875     {
2876       (1 - #2) * \tl_item:nn {#1} {3} + (#2) * \tl_item:nn {#1} {6}
2877     }}
2878 }
2879
2880 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curvetob_tl
2881 \tl_put_right:Nx \l__spath_tmpa_tl
2882 {
2883   {\fp_to_dim:n
2884     {
2885       (1 - #2)^2 * \tl_item:nn {#1} {2}
2886       + 2 * (1 - #2) * (#2) * \tl_item:nn {#1} {5}
2887       + (#2)^2 * \tl_item:nn {#1} {8}
2888     }}
2889   {\fp_to_dim:n
2890     {
2891       (1 - #2)^2 * \tl_item:nn {#1} {3}
2892       + 2 * (1 - #2) * (#2) * \tl_item:nn {#1} {6}
2893       + (#2)^2 * \tl_item:nn {#1} {9}
2894     }}
2895 }
2896
2897 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curveto_tl
2898 \tl_put_right:Nx \l__spath_tmpa_tl
2899 {
2900   {\fp_to_dim:n
2901     {
2902       (1 - #2)^3 * \tl_item:nn {#1} {2}
2903       + 3 * (1 - #2)^2 * (#2) * \tl_item:nn {#1} {5}
2904       + 3 * (1 - #2) * (#2)^2 * \tl_item:nn {#1} {8}
2905       + (#2)^3 * \tl_item:nn {#1} {11}
2906     }}
2907   {\fp_to_dim:n
2908     {
2909       (1 - #2)^3 * \tl_item:nn {#1} {3}
2910       + 3 * (1 - #2)^2 * (#2) * \tl_item:nn {#1} {6}
2911       + 3 * (1 - #2) * (#2)^2 * \tl_item:nn {#1} {9}
2912       + (#2)^3 * \tl_item:nn {#1} {12}
2913     }}
2914 }
2915

```

```

2916 \tl_gclear:N \g__spath_output_tl
2917 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpa_tl
2918
2919 \tl_clear:N \l__spath_tmpa_tl
2920 \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_moveto_tl
2921 \tl_put_right:Nx \l__spath_tmpa_tl
2922 {
2923   {\fp_to_dim:n
2924     {
2925       (1 - #2)^3 * \tl_item:nn {#1} {2}
2926       + 3 * (1 - #2)^2 * (#2) * \tl_item:nn {#1} {5}
2927       + 3 * (1 - #2) * (#2)^2 * \tl_item:nn {#1} {8}
2928       + (#2)^3 * \tl_item:nn {#1} {11}
2929     }}
2930   {\fp_to_dim:n
2931     {
2932       (1 - #2)^3 * \tl_item:nn {#1} {3}
2933       + 3 * (1 - #2)^2 * (#2) * \tl_item:nn {#1} {6}
2934       + 3 * (1 - #2) * (#2)^2 * \tl_item:nn {#1} {9}
2935       + (#2)^3 * \tl_item:nn {#1} {12}
2936     }}
2937 }
2938
2939 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curvetoa_tl
2940 \tl_put_right:Nx \l__spath_tmpa_tl
2941 {
2942   {\fp_to_dim:n
2943     {
2944       (1 - #2)^2 * \tl_item:nn {#1} {5}
2945       + 2 * (1 - #2) * (#2) * \tl_item:nn {#1} {8}
2946       + (#2)^2 * \tl_item:nn {#1} {11}
2947     }}
2948   {\fp_to_dim:n
2949     {
2950       (1 - #2)^2 * \tl_item:nn {#1} {6}
2951       + 2 * (1 - #2) * (#2) * \tl_item:nn {#1} {9}
2952       + (#2)^2 * \tl_item:nn {#1} {12}
2953     }}
2954 }
2955 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curvetob_tl
2956 \tl_put_right:Nx \l__spath_tmpa_tl
2957 {
2958   {\fp_to_dim:n
2959     {
2960       (1 - #2) * \tl_item:nn {#1} {8} + (#2) * \tl_item:nn {#1} {11}
2961     }}
2962   {\fp_to_dim:n
2963     {
2964       (1 - #2) * \tl_item:nn {#1} {9} + (#2) * \tl_item:nn {#1} {12}
2965     }}
2966 }
2967 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curveto_tl
2968 \tl_put_right:Nx \l__spath_tmpa_tl {
2969   {\tl_item:nn {#1} {11}}

```

```

2970   {\tl_item:nn {#1} {12}}
2971 }
2972
2973 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpa_tl
2974 \group_end:
2975 }
2976 \cs_generate_variant:Nn \__spath_split_curve:nn {nv, nV}
2977 \cs_new_protected_nopar:Npn \spath_split_curve:NNnn #1#2#3#4
2978 {
2979   \__spath_split_curve:nn {#3}{#4}
2980   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
2981   \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
2982   \tl_gclear:N \g__spath_output_tl
2983 }
2984 \cs_generate_variant:Nn \spath_split_curve:NNnn {NNnV, NNVn, NNVV}
2985 \cs_new_protected_nopar:Npn \spath_gsplit_curve:NNnn #1#2#3#4
2986 {
2987   \__spath_split_curve:nn {#3}{#4}
2988   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
2989   \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
2990   \tl_gclear:N \g__spath_output_tl
2991 }
2992 \cs_generate_variant:Nn \spath_gsplit_curve:NNnn {NNnV, NNVn, NNVV}

```

(End definition for \spath_split_curve:NNnn and \spath_gsplit_curve:NNnn.)

\spath_maybe_split_curve:Nn Possibly splits a bezier curve to ensure that the pieces don't self-intersect. Figuring out whether a Bezier cubic self intersects is apparently a difficult problem so we don't bother. We compute a point such that if there is an intersection then it lies on either side of the point. I don't recall where the formula came from!

\spath_maybe_gsplit_curve:Nn

```

2993 \cs_new_protected_nopar:Npn \__spath_maybe_split_curve:n #1
2994 {
2995   \group_begin:
2996   \fp_set:Nn \l__spath_tmpa_fp
2997   {
2998     (
2999       \tl_item:nn {#1} {3}
3000       - 3 * \tl_item:nn {#1} {6}
3001       + 3 * \tl_item:nn {#1} {9}
3002       - \tl_item:nn {#1} {12}
3003     )
3004     *
3005     (3 * \tl_item:nn {#1} {8} - 3 * \tl_item:nn {#1} {11})
3006     -
3007     (
3008       \tl_item:nn {#1} {2}
3009       - 3 * \tl_item:nn {#1} {5}
3010       + 3 * \tl_item:nn {#1} {8}
3011       - \tl_item:nn {#1} {11}
3012     )
3013     *
3014     (3 * \tl_item:nn {#1} {9} - 3 * \tl_item:nn {#1} {12})
3015   }
3016   \fp_set:Nn \l__spath_tmpb_fp

```

```

3017 {
3018 (
3019 \tl_item:nn {#1} {2}
3020 - 3 * \tl_item:nn {#1} {5}
3021 + 3 * \tl_item:nn {#1} {8}
3022 - \tl_item:nn {#1} {11}
3023 )
3024 *
3025 (
3026 3 * \tl_item:nn {#1} {6}
3027 - 6 * \tl_item:nn {#1} {9}
3028 + 3 * \tl_item:nn {#1} {12}
3029 )
3030 -
3031 (
3032 \tl_item:nn {#1} {3}
3033 - 3 * \tl_item:nn {#1} {6}
3034 + 3 * \tl_item:nn {#1} {9}
3035 - \tl_item:nn {#1} {12}
3036 )
3037 *
3038 (
3039 3 * \tl_item:nn {#1} {5}
3040 - 6 * \tl_item:nn {#1} {8}
3041 + 3 * \tl_item:nn {#1} {11}
3042 )
3043 }
3044 \fp_compare:nTF
3045 {
3046 \l__spath_tmpb_fp != 0
3047 }
3048 {
3049 \fp_set:Nn \l__spath_tmpa_fp {.5 * \l__spath_tmpa_fp / \l__spath_tmpb_fp}
3050 \fp_compare:nTF
3051 {
3052 0 < \l__spath_tmpa_fp && \l__spath_tmpa_fp < 1
3053 }
3054 {
3055 \__spath_split_curve:nV {#1} \l__spath_tmpa_fp
3056 }
3057 {
3058 \tl_gset:Nn \g__spath_output_tl { {#1} {} }
3059 }
3060 }
3061 {
3062 \tl_gset:Nn \g__spath_output_tl { {#1} {} }
3063 }
3064 \group_end:
3065 }
3066 \cs_new_protected_nopar:Npn \spath_maybe_split_curve:NNn #1#2#3
3067 {
3068 \__spath_maybe_split_curve:n {#3}
3069 \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3070 \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}

```

```

3071 \tl_gclear:N \g__spath_output_tl
3072 }
3073 \cs_generate_variant:Nn \spath_maybe_split_curve:NNn {NNn, NNV }
3074 \cs_new_protected_nopar:Npn \spath_maybe_gsplit_curve:NNn #1#2#3
3075 {
3076   \__spath_maybe_split_curve:n {#3}
3077   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3078   \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3079   \tl_gclear:N \g__spath_output_tl
3080 }
3081 \cs_generate_variant:Nn \spath_maybe_gsplit_curve:NNn {NNn, NNV}

```

(End definition for \spath_maybe_split_curve:Nn and \spath_maybe_gsplit_curve:Nn.)

\spath_split_curves:Nn Slurp through the path ensuring that beziers don't self-intersect.
\spath_gsplit_curves:Nn

```

3082 \cs_new_protected_nopar:Npn \__spath_split_curves:n #1
3083 {
3084   \group_begin:
3085   \tl_set:Nn \l__spath_tmpa_tl {#1}
3086   \tl_clear:N \l__spath_tmpb_tl
3087   \tl_clear:N \l__spath_tmpe_tl
3088   \bool_do_until:nn
3089   {
3090     \tl_if_empty_p:N \l__spath_tmpa_tl
3091   }
3092   {
3093     \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpa_tl}
3094     \tl_case:NnF \l__spath_tmpe_tl
3095     {
3096       \c_spath_curvetoa_tl
3097       {
3098         \tl_clear:N \l__spath_tmpe_tl
3099         \tl_set_eq:NN \l__spath_tmpe_tl \c_spath_moveto_tl
3100         \tl_put_right:Nx \l__spath_tmpe_tl
3101         {
3102           { \dim_use:N \l__spath_tmpa_dim }
3103           { \dim_use:N \l__spath_tmpe_dim }
3104         }
3105         \dim_set:Nn \l__spath_tmpa_dim
3106         {
3107           \tl_item:Nn \l__spath_tmpa_tl {8}
3108         }
3109         \dim_set:Nn \l__spath_tmpe_dim
3110         {
3111           \tl_item:Nn \l__spath_tmpe_tl {9}
3112         }
3113         \prg_replicate:nn {3}
3114         {
3115           \tl_put_right:Nx \l__spath_tmpe_tl
3116           {
3117             \tl_item:Nn \l__spath_tmpa_tl {1}
3118             {\tl_item:Nn \l__spath_tmpe_tl {2}}
3119             {\tl_item:Nn \l__spath_tmpe_tl {3}}
3120           }

```



```

3121     \prg_replicate:nn {3}
3122     {
3123         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3124     }
3125 }
3126
3127 \spath_maybe_split_curve:NNV
3128 \l__spath_tmpd_tl
3129 \l__spath_tmpe_tl
3130 \l__spath_tmpd_tl
3131 \prg_replicate:nn {3}
3132 {
3133     \tl_set:Nx \l__spath_tmpd_tl {\tl_tail:N \l__spath_tmpd_tl}
3134     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3135 }
3136 \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
3137 \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
3138 }
3139 }
3140 {
3141     \dim_set:Nn \l__spath_tmpa_dim
3142     {
3143         \tl_item:Nn \l__spath_tmpa_tl {2}
3144     }
3145     \dim_set:Nn \l__spath_tmpb_dim
3146     {
3147         \tl_item:Nn \l__spath_tmpa_tl {3}
3148     }
3149     \tl_put_right:Nx \l__spath_tmpb_tl
3150     {
3151         \tl_item:Nn \l__spath_tmpa_tl {1}
3152         {\tl_item:Nn \l__spath_tmpa_tl {2}}
3153         {\tl_item:Nn \l__spath_tmpa_tl {3}}
3154     }
3155     \prg_replicate:nn {3}
3156     {
3157         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3158     }
3159 }
3160 }
3161 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
3162 \group_end:
3163 }
3164 \cs_new_protected_nopar:Npn \spath_split_curves:Nn #1#2
3165 {
3166     \__spath_split_curves:n {#2}
3167     \tl_set_eq:NN #1 \g__spath_output_tl
3168     \tl_gclear:N \g__spath_output_tl
3169 }
3170 \cs_generate_variant:Nn \spath_split_curves:Nn {NV, cV, cn, cv }
3171 \cs_new_protected_nopar:Npn \spath_split_curves:N #1
3172 {
3173     \spath_split_curves:NV #1#1
3174 }

```

```

3175 \cs_generate_variant:Nn \spath_split_curves:N {c}
3176 \cs_new_protected_nopar:Npn \spath_gsplit_curves:Nn #1#2
3177 {
3178   \__spath_split_curves:n {#2}
3179   \tl_gset_eq:NN #1 \g__spath_output_tl
3180   \tl_gclear:N \g__spath_output_tl
3181 }
3182 \cs_generate_variant:Nn \spath_gsplit_curves:Nn {NV, cV, cn, cv }
3183 \cs_new_protected_nopar:Npn \spath_gsplit_curves:N #1
3184 {
3185   \spath_gsplit_curves:NV #1#1
3186 }
3187 \cs_generate_variant:Nn \spath_gsplit_curves:N {c}

```

(End definition for \spath_split_curves:Nn and \spath_gsplit_curves:Nn.)

\spath_split_line:NNnn Splits a line segment.

```

\spath_gsplit_line:NNnn
3188 \cs_new_protected_nopar:Npn \__spath_split_line:nn #1#2
3189 {
3190   \group_begin:
3191   \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_moveto_tl
3192   \tl_put_right:Nx \l__spath_tmpa_tl {
3193     {\tl_item:nn {#1} {2}}
3194     {\tl_item:nn {#1} {3}}
3195   }
3196   \tl_put_right:NV \l__spath_tmpa_tl \c_spath_lineto_tl
3197   \tl_put_right:Nx \l__spath_tmpa_tl
3198   {
3199     {\fp_to_dim:n
3200     {
3201       (1 - #2) * \tl_item:nn {#1} {2} + (#2) * \tl_item:nn {#1} {5}
3202     }}
3203     {\fp_to_dim:n
3204     {
3205       (1 - #2) * \tl_item:nn {#1} {3} + (#2) * \tl_item:nn {#1} {6}
3206     }}
3207   }
3208   \tl_gclear:N \g__spath_output_tl
3209   \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpa_tl
3210
3211   \tl_clear:N \l__spath_tmpa_tl
3212   \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_moveto_tl
3213   \tl_put_right:Nx \l__spath_tmpa_tl
3214   {
3215     {\fp_to_dim:n
3216     {
3217       (1 - #2) * \tl_item:nn {#1} {2} + (#2) * \tl_item:nn {#1} {5}
3218     }}
3219     {\fp_to_dim:n
3220     {
3221       (1 - #2) * \tl_item:nn {#1} {3} + (#2) * \tl_item:nn {#1} {6}
3222     }}
3223   }
3224   \tl_put_right:NV \l__spath_tmpa_tl \c_spath_lineto_tl

```

```

3225 \tl_put_right:Nx \l__spath_tmpa_tl {
3226   {\tl_item:nn {#1} {5}}
3227   {\tl_item:nn {#1} {6}}
3228 }
3229
3230 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpa_tl
3231 \group_end:
3232 }
3233 \cs_new_protected_nopar:Npn \spath_split_line:NNnn #1#2#3#4
3234 {
3235   \__spath_split_line:nn {#3}{#4}
3236   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3237   \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3238   \tl_gclear:N \g__spath_output_tl
3239 }
3240 \cs_generate_variant:Nn \spath_split_line:NNnn {NNnV, NNVn, NNVV}
3241 \cs_new_protected_nopar:Npn \spath_gsplit_line:NNnn #1#2#3#4
3242 {
3243   \__spath_split_line:nn {#3}{#4}
3244   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3245   \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3246   \tl_gclear:N \g__spath_output_tl
3247 }
3248 \cs_generate_variant:Nn \spath_gsplit_line:NNnn {NNnV, NNVn, NNVV}

```

(End definition for \spath_split_line:NNnn and \spath_gsplit_line:NNnn.)

\spath_split_rectangle:Nnn
\spath_gsplit_rectangle:Nnn

Cuts a rectangle at a point.

```

3249 \cs_new_protected_nopar:Npn \__spath_split_rectangle:nn #1#2
3250 {
3251   \group_begin:
3252   \spath_open:Nn \l__spath_tmpa_tl {#1}
3253   \fp_set:Nn \l__spath_tmpa_fp {4*(#2)}
3254   \spath_split_at:NNVV
3255   \l__spath_tmpa_tl \l__spath_tmpb_tl \l__spath_tmpa_tl \l__spath_tmpa_fp
3256   \__spath_append_no_move:VV \l__spath_tmpb_tl \l__spath_tmpa_tl
3257   \group_end:
3258 }
3259 \cs_new_protected_nopar:Npn \spath_split_rectangle:Nnn #1#2#3
3260 {
3261   \__spath_split_rectangle:nn {#2}{#3}
3262   \tl_set_eq:NN #1 \g__spath_output_tl
3263   \tl_gclear:N \g__spath_output_tl
3264 }
3265 \cs_generate_variant:Nn \spath_split_rectangle:Nnn {NnV, NVn, NVV}
3266 \cs_new_protected_nopar:Npn \spath_gsplit_rectangle:Nnn #1#2#3
3267 {
3268   \__spath_split_rectangle:nn {#2}{#3}
3269   \tl_gset_eq:NN #1 \g__spath_output_tl
3270   \tl_gclear:N \g__spath_output_tl
3271 }
3272 \cs_generate_variant:Nn \spath_gsplit_rectangle:Nnn {NnV, NVn, NVV}

```

(End definition for \spath_split_rectangle:Nnn and \spath_gsplit_rectangle:Nnn.)

```

\spath_split_at:NNnn Split a path according to the parameter generated by the intersection routine. The
\spath_split_at:Nnn versions with two N arguments stores the two parts in two macros, the version with a
\spath_split_at:Nn single N joins them back into a single path (as separate components). The keep versions
\spath_gsplit_at:NNnn\spath_gsplit_at:Nnn throw away the other part of the curve.
\spath_gsplit_at:Nn
\spath_split_at_keep_start:Nnn 3273 \cs_new_protected_nopar:Npn \__spath_split_at:nn #1#2
\spath_split_at_keep_start:Nn 3274 {
\spath_gsplit_at_keep_start:Nnn 3275 \group_begin:
\spath_gsplit_at_keep_start:Nn 3276 \int_set:Nn \l__spath_tmpa_int {\fp_to_int:n {floor(#2) + 1}}
\spath_gsplit_at_keep_start:Nn 3277 \fp_set:Nn \l__spath_tmpa_fp {#2 - floor(#2)}
\spath_split_at_keep_end:Nnn 3278
\spath_split_at_keep_end:Nn 3279 % Is split point near one end or other of a component?
\spath_gsplit_at_keep_end:Nnn 3280 \fp_compare:nT
\spath_gsplit_at_keep_end:Nn 3281 {
\spath_split_at_normalised:NNnn 3282 \l__spath_tmpa_fp < 0.01
\spath_split_at_normalised:Nnn 3283 }
\spath_split_at_normalised:Nn 3284 {
\spath_gsplit_at_normalised:Nnn 3285 % Near the start, so we'll place it at the start
\spath_gsplit_at_normalised:Nn 3286 \fp_set:Nn \l__spath_tmpa_fp {0}
\spath_split_at_normalised_keep_start:Nnn 3287 }
\spath_split_at_normalised_keep_start:Nn 3288 \fp_compare:nT
\spath_gsplit_at_normalised_keep_start:Nnn 3289 {
\spath_gsplit_at_normalised_keep_start:Nn 3290 \l__spath_tmpa_fp > 0.99
\spath_split_at_normalised_keep_end:Nnn 3291 }
\spath_split_at_normalised_keep_end:Nn 3292 {
\spath_split_at_normalised_keep_end:Nn 3293 % Near the end, so we'll place it at the end
\spath_split_at_normalised_keep_end:Nn 3294 \fp_set:Nn \l__spath_tmpa_fp {0}
\spath_gsplit_at_normalised_keep_end:Nnn 3295 \int_incr:N \l__spath_tmpa_int
\spath_gsplit_at_normalised_keep_end:Nn 3296 }
3297
\int_zero:N \l__spath_tmpb_int
\bool_set_true:N \l__spath_tmpa_bool
3300
\tl_set:Nn \l__spath_tmpe_tl {#1}
\tl_clear:N \l__spath_tmpe_tl
3303
\dim_zero:N \l__spath_tmpe_dim
\dim_zero:N \l__spath_tmpe_dim
3306
\bool_until_do:nn {
3307 \tl_if_empty_p:N \l__spath_tmpe_tl
3308 ||
3309 \int_compare_p:n { \l__spath_tmpa_int == \l__spath_tmpb_int }
3310 }
3311 {
3312 \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpe_tl}
3313 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3314 \tl_case:Nn \l__spath_tmpe_tl
3315 {
3316 \c_spath_lineto_tl
3317 {
3318 \int_incr:N \l__spath_tmpb_int
3319 }
3320 }
3321 \c_spath_curvetoa_tl
3322 {

```

```

3323     \int_incr:N \l__spath_tmpb_int
3324   }
3325   \c_spath_rectcorner_tl
3326   {
3327     \int_incr:N \l__spath_tmpb_int
3328   }
3329 }
3330 \int_compare:nT { \l__spath_tmpb_int < \l__spath_tmpe_int }
3331 {
3332   \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3333
3334   \tl_put_right:Nx \l__spath_tmpe_tl
3335   {{ \tl_head:N \l__spath_tmpe_tl }}
3336   \dim_set:Nn \l__spath_tmpe_dim {\tl_head:N \l__spath_tmpe_tl}
3337   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3338
3339   \tl_put_right:Nx \l__spath_tmpe_tl
3340   {{ \tl_head:N \l__spath_tmpe_tl }}
3341   \dim_set:Nn \l__spath_tmpe_dim {\tl_head:N \l__spath_tmpe_tl}
3342   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3343
3344 }
3345 }
3346
3347 \tl_clear:N \l__spath_tmpe_tl
3348 \tl_put_right:NV \l__spath_tmpe_tl \c_spath_moveto_tl
3349 \tl_put_right:Nx \l__spath_tmpe_tl
3350 {
3351   {\dim_use:N \l__spath_tmpe_dim}
3352   {\dim_use:N \l__spath_tmpe_dim}
3353 }
3354
3355 \fp_compare:nTF
3356 {
3357   \l__spath_tmpe_fp == 0
3358 }
3359 {
3360   \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpe_tl
3361   \tl_if_empty:NF \l__spath_tmpe_tl
3362   {
3363     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3364     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3365   }
3366 }
3367 {
3368
3369   \tl_case:Nn \l__spath_tmpe_tl
3370   {
3371     \c_spath_lineto_tl
3372     {
3373       \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3374       \tl_put_right:Nx \l__spath_tmpe_tl
3375       {{ \tl_head:N \l__spath_tmpe_tl }}
3376       \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }

```

```

3377
3378 \tl_put_right:Nx \l__spath_tmpd_tl
3379 {{ \tl_head:N \l__spath_tmpe_tl }}
3380 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3381
3382 \spath_split_line:NNVV
3383 \l__spath_tmpa_tl
3384 \l__spath_tmpe_tl
3385 \l__spath_tmpe_tl
3386 \l__spath_tmpe_tl
3387
3388 \prg_replicate:nn {3} {
3389   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3390 }
3391
3392 \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3393 \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3394 }
3395 \c_spath_curvetoa_tl
3396 {
3397   \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3398   \tl_put_right:Nx \l__spath_tmpe_tl
3399   {{ \tl_head:N \l__spath_tmpe_tl }}
3400   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3401
3402   \tl_put_right:Nx \l__spath_tmpe_tl
3403   {{ \tl_head:N \l__spath_tmpe_tl }}
3404   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3405
3406   \prg_replicate:nn {2} {
3407
3408     \tl_put_right:Nx \l__spath_tmpe_tl
3409     { \tl_head:N \l__spath_tmpe_tl }
3410     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3411
3412     \tl_put_right:Nx \l__spath_tmpe_tl
3413     {{ \tl_head:N \l__spath_tmpe_tl }}
3414     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3415
3416     \tl_put_right:Nx \l__spath_tmpe_tl
3417     {{ \tl_head:N \l__spath_tmpe_tl }}
3418     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3419   }
3420
3421   \spath_split_curve:NNVV
3422   \l__spath_tmpe_tl
3423   \l__spath_tmpe_tl
3424   \l__spath_tmpe_tl \l__spath_tmpe_tl
3425
3426   \prg_replicate:nn {3} {
3427     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3428   }
3429
3430   \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl

```

```

3431     \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
3432   }
3433
3434   \c_spath_rectcorner_tl
3435   {
3436     \tl_clear:N \l__spath_tmpd_tl
3437     \tl_put_right:NV \l__spath_tmpd_tl \l__spath_tmpf_tl
3438
3439     \tl_put_right:Nx \l__spath_tmpd_tl {{\tl_head:N \l__spath_tmpe_tl}}
3440     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3441     \tl_put_right:Nx \l__spath_tmpd_tl {{\tl_head:N \l__spath_tmpe_tl}}
3442     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3443
3444     \tl_put_right:Nx \l__spath_tmpd_tl {\tl_head:N \l__spath_tmpe_tl}
3445     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3446
3447     \tl_put_right:Nx \l__spath_tmpd_tl {{\tl_head:N \l__spath_tmpe_tl}}
3448     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3449     \tl_put_right:Nx \l__spath_tmpd_tl {{\tl_head:N \l__spath_tmpe_tl}}
3450     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3451
3452     \spath_split_rectangle:NVV
3453     \l__spath_tmpa_tl
3454     \l__spath_tmpd_tl
3455     \l__spath_tmpa_fp
3456
3457     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpa_tl
3458     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3459   }
3460
3461 }
3462 }
3463
3464 \tl_gclear:N \g__spath_output_tl
3465 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpe_tl
3466 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpe_tl
3467 \group_end:
3468 }
3469 \cs_generate_variant:Nn \__spath_split_at:nn {nV, VV}
3470 \cs_new_protected_nopar:Npn \__spath_split_at_normalised:nn #1#2
3471 {
3472   \group_begin:
3473   \spath_reallength:Nn \l__spath_tmpa_int {#1}
3474
3475   \tl_set:Nx \l__spath_tmpa_tl
3476   {\fp_to_decimal:n {(#2) * (\l__spath_tmpa_int)}}
3477   \__spath_split_at:nV {#1} \l__spath_tmpa_tl
3478   \group_end:
3479 }
3480 \cs_generate_variant:Nn \__spath_split_at_normalised:nn {nV}
3481 \cs_new_protected_nopar:Npn \spath_split_at:NNnn #1#2#3#4
3482 {
3483   \__spath_split_at:nn {#3}{#4}
3484   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}

```

```

3485 \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3486 \tl_gclear:N \g__spath_output_tl
3487 }
3488 \cs_generate_variant:Nn \spath_split_at:NNnn {NNVn, NNVV, NNnV}
3489 \cs_new_protected_nopar:Npn \spath_gsplit_at:NNnn #1#2#3#4
3490 {
3491   \__spath_split_at:nn {#3}{#4}
3492   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3493   \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3494   \tl_gclear:N \g__spath_output_tl
3495 }
3496 \cs_generate_variant:Nn \spath_gsplit_at:NNnn {NNVn, NNVV, NNnV}
3497 \cs_new_protected_nopar:Npn \spath_split_at_keep_start:Nnn #1#2#3
3498 {
3499   \__spath_split_at:nn {#2}{#3}
3500   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3501   \tl_gclear:N \g__spath_output_tl
3502 }
3503 \cs_generate_variant:Nn \spath_split_at_keep_start:Nnn {NVn}
3504 \cs_new_protected_nopar:Npn \spath_split_at_keep_start:Nn #1#2
3505 {
3506   \spath_split_at_keep_start:NVn #1#1{#2}
3507 }
3508 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_start:Nnn #1#2#3
3509 {
3510   \__spath_split_at:nn {#2}{#3}
3511   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3512   \tl_gclear:N \g__spath_output_tl
3513 }
3514 \cs_generate_variant:Nn \spath_gsplit_at_keep_start:Nnn {NVn}
3515 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_start:Nn #1#2
3516 {
3517   \spath_gsplit_at_keep_start:NVn #1#1{#2}
3518 }
3519 \cs_new_protected_nopar:Npn \spath_split_at_keep_end:Nnn #1#2#3
3520 {
3521   \__spath_split_at:nn {#2}{#3}
3522   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3523   \tl_gclear:N \g__spath_output_tl
3524 }
3525 \cs_generate_variant:Nn \spath_split_at_keep_end:Nnn {NVn}
3526 \cs_new_protected_nopar:Npn \spath_split_at_keep_end:Nn #1#2
3527 {
3528   \spath_split_at_keep_end:NVn #1#1{#2}
3529 }
3530 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_end:Nnn #1#2#3
3531 {
3532   \__spath_split_at:nn {#2}{#3}
3533   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3534   \tl_gclear:N \g__spath_output_tl
3535 }
3536 \cs_generate_variant:Nn \spath_gsplit_at_keep_end:Nnn {NVn}
3537 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_end:Nn #1#2
3538 {

```



```

3539 \spath_gsplit_at_keep_end:NVn #1#1{#2}
3540 }
3541 \cs_new_protected_nopar:Npn \spath_split_at_normalised:NNnn #1#2#3#4
3542 {
3543   \__spath_split_at_normalised:nn {#3}{#4}
3544   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3545   \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3546   \tl_gclear:N \g__spath_output_tl
3547 }
3548 \cs_generate_variant:Nn \spath_split_at_normalised:NNnn {NNVn, NNVV, NNnV, ccvn}
3549 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised:NNnn #1#2#3#4
3550 {
3551   \__spath_split_at_normalised:nn {#3}{#4}
3552   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3553   \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3554   \tl_gclear:N \g__spath_output_tl
3555 }
3556 \cs_generate_variant:Nn \spath_gsplit_at_normalised:NNnn {NNVn, NNVV, NNnV, ccvn}
3557 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_start:Nnn #1#2#3
3558 {
3559   \__spath_split_at_normalised:nn {#2}{#3}
3560   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3561   \tl_gclear:N \g__spath_output_tl
3562 }
3563 \cs_generate_variant:Nn \spath_split_at_normalised_keep_start:Nnn {NVn}
3564 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_start:Nn #1#2
3565 {
3566   \spath_split_at_normalised_keep_start:NVn #1#1{#2}
3567 }
3568 \cs_generate_variant:Nn \spath_split_at_normalised_keep_start:Nn {cn}
3569 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_start:Nnn #1#2#3
3570 {
3571   \__spath_split_at_normalised:nn {#2}{#3}
3572   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3573   \tl_gclear:N \g__spath_output_tl
3574 }
3575 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_start:Nnn {NVn}
3576 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_start:Nn #1#2
3577 {
3578   \spath_gsplit_at_normalised_keep_start:NVn #1#1{#2}
3579 }
3580 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_start:Nn {cn}
3581 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_end:Nnn #1#2#3
3582 {
3583   \__spath_split_at_normalised:nn {#2}{#3}
3584   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3585   \tl_gclear:N \g__spath_output_tl
3586 }
3587 \cs_generate_variant:Nn \spath_split_at_normalised_keep_end:Nnn {NVn}
3588 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_end:Nn #1#2
3589 {
3590   \spath_split_at_normalised_keep_end:NVn #1#1{#2}
3591 }
3592 \cs_generate_variant:Nn \spath_split_at_normalised_keep_end:Nn {cn}

```

```

3593 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_end:Nnn #1#2#3
3594 {
3595   \__spath_split_at_normalised:nn {#2}{#3}
3596   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3597   \tl_gclear:N \g__spath_output_tl
3598 }
3599 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_end:Nnn {NVn}
3600 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_end:Nn #1#2
3601 {
3602   \spath_gsplit_at_normalised_keep_end:NVn #1#1{#2}
3603 }
3604 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_end:Nn {cn}
3605 \cs_new_protected_nopar:Npn \spath_split_at:Nnn #1#2#3
3606 {
3607   \__spath_split_at:nn {#2}{#3}
3608   \tl_set:Nx #1
3609   {
3610     \tl_item:Nn \g__spath_output_tl {1}
3611     \tl_item:Nn \g__spath_output_tl {2}
3612   }
3613   \tl_gclear:N \g__spath_output_tl
3614 }
3615 \cs_generate_variant:Nn \spath_split_at:Nnn {NVn, NVV}
3616 \cs_new_protected_nopar:Npn \spath_split_at:Nn #1#2
3617 {
3618   \spath_split_at:NVn #1#1{#2}
3619 }
3620 \cs_new_protected_nopar:Npn \spath_gsplit_at:Nnn #1#2#3
3621 {
3622   \__spath_split_at:nn {#2}{#3}
3623   \tl_gset:Nx #1
3624   {
3625     \tl_item:Nn \g__spath_output_tl {1}
3626     \tl_item:Nn \g__spath_output_tl {2}
3627   }
3628   \tl_gclear:N \g__spath_output_tl
3629 }
3630 \cs_generate_variant:Nn \spath_gsplit_at:Nnn {NVn, NVV}
3631 \cs_new_protected_nopar:Npn \spath_gsplit_at:Nn #1#2
3632 {
3633   \spath_gsplit_at:NVn #1#1{#2}
3634 }
3635 \cs_new_protected_nopar:Npn \spath_split_at_normalised:Nnn #1#2#3
3636 {
3637   \__spath_split_at_normalised:nn {#2}{#3}
3638   \tl_set:Nx #1
3639   {
3640     \tl_item:Nn \g__spath_output_tl {1}
3641     \tl_item:Nn \g__spath_output_tl {2}
3642   }
3643   \tl_gclear:N \g__spath_output_tl
3644 }
3645 \cs_generate_variant:Nn \spath_split_at_normalised:Nnn {NVn, NVV}
3646 \cs_new_protected_nopar:Npn \spath_split_at_normalised:Nn #1#2

```

```

3647 {
3648   \spath_split_at_normalised:NVn #1#1{#2}
3649 }
3650 \cs_generate_variant:Nn \spath_split_at_normalised:Nn {cn}
3651 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised:Nnn #1#2#3
3652 {
3653   \__spath_split_at_normalised:nn {#2}{#3}
3654   \tl_gset:Nx #1
3655   {
3656     \tl_item:Nn \g__spath_output_tl {1}
3657     \tl_item:Nn \g__spath_output_tl {2}
3658   }
3659   \tl_gclear:N \g__spath_output_tl
3660 }
3661 \cs_generate_variant:Nn \spath_gsplit_at_normalised:Nnn {NVn, NVV}
3662 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised:Nn #1#2
3663 {
3664   \spath_gsplit_at_normalised:NVn #1#1{#2}
3665 }
3666 \cs_generate_variant:Nn \spath_gsplit_at_normalised:Nn {cn}

```

(End definition for `\spath_split_at:NNnn` and others.)

3.5 Shortening Paths

This code relates to shortening paths. For curved paths, the routine uses the derivative at the end to figure out how far back to shorten. This means that the actual length that it shortens by is approximate, but it is guaranteed to be along its length.

As in the previous section, there are various versions. In particular, there are versions where the path can be specified by a macro and is saved back into that macro.

`\spath_shorten_at_end:Nnn` This macro shortens a path from the end by a dimension.

```

3667 \cs_new_protected_nopar:Npn \__spath_shorten_at_end:nn #1#2
3668 {
3669   \int_compare:nTF
3670   {
3671     \tl_count:n {#1} > 3
3672   }
3673   {
3674     \group_begin:
3675     \tl_set:Nn \l__spath_tmpa_tl {#1}
3676     \tl_reverse:N \l__spath_tmpa_tl
3677
3678     \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {3}}
3679
3680     \tl_clear:N \l__spath_tmpe_tl
3681     \tl_if_eq:NNTF \l__spath_tmpb_tl \c_spath_curveto_tl
3682     {
3683       \int_set:Nn \l__spath_tmpa_int {3}
3684     }
3685     {
3686       \int_set:Nn \l__spath_tmpa_int {1}
3687     }
3688

```

```

3689 \prg_replicate:nn { \l__spath_tmpa_int }
3690 {
3691   \tl_put_right:Nx \l__spath_tmpe_tl
3692   {
3693     {\tl_head:N \l__spath_tmpa_tl}
3694   }
3695   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3696   \tl_put_right:Nx \l__spath_tmpe_tl
3697   {
3698     {\tl_head:N \l__spath_tmpa_tl}
3699   }
3700   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3701   \tl_put_right:Nx \l__spath_tmpe_tl
3702   {
3703     \tl_head:N \l__spath_tmpa_tl
3704   }
3705   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3706 }
3707
3708 \tl_put_right:Nx \l__spath_tmpe_tl
3709 {
3710   {\tl_item:Nn \l__spath_tmpa_tl {1}}
3711   {\tl_item:Nn \l__spath_tmpa_tl {2}}
3712 }
3713 \tl_put_right:NV \l__spath_tmpe_tl \c_spath_moveto_tl
3714
3715 \tl_reverse:N \l__spath_tmpa_tl
3716
3717 \fp_set:Nn \l__spath_tmpa_fp
3718 {
3719   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {4}}
3720   -
3721   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {1}}
3722 }
3723
3724 \fp_set:Nn \l__spath_tmpb_fp
3725 {
3726   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {5}}
3727   -
3728   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {2}}
3729 }
3730
3731 \fp_set:Nn \l__spath_tmpe_fp
3732 {
3733   sqrt(
3734     \l__spath_tmpa_fp * \l__spath_tmpa_fp
3735     +
3736     \l__spath_tmpb_fp * \l__spath_tmpb_fp
3737   ) * \l__spath_tmpa_int
3738 }
3739
3740 \fp_compare:nTF
3741 {
3742   \l__spath_tmpe_fp > #2

```

```

3743 }
3744 {
3745
3746 \fp_set:Nn \l__spath_tmpc_fp
3747 {
3748 (\l__spath_tmpc_fp - #2)/ \l__spath_tmpc_fp
3749 }
3750
3751 \tl_reverse:N \l__spath_tmpe_tl
3752
3753 \tl_if_eq:NNTF \l__spath_tmpb_tl \c_spath_curveto_tl
3754 {
3755 \spath_split_curve:NNVV
3756 \l__spath_tmpc_tl
3757 \l__spath_tmpd_tl
3758 \l__spath_tmpe_tl
3759 \l__spath_tmpc_fp
3760 }
3761 {
3762 \spath_split_line:NNVV
3763 \l__spath_tmpc_tl
3764 \l__spath_tmpd_tl
3765 \l__spath_tmpe_tl
3766 \l__spath_tmpc_fp
3767 }
3768
3769 \prg_replicate:nn {3}
3770 {
3771 \tl_set:Nx \l__spath_tmpc_tl {\tl_tail:N \l__spath_tmpc_tl}
3772 }
3773
3774 \tl_put_right:NV \l__spath_tmpa_tl \l__spath_tmpc_tl
3775
3776 }
3777 {
3778
3779 \int_compare:nT
3780 {
3781 \tl_count:N \l__spath_tmpa_tl > 3
3782 }
3783 {
3784 \dim_set:Nn \l__spath_tmpa_dim {\fp_to_dim:n {#2 - \l__spath_tmpc_fp } }
3785 \spath_shorten_at_end:NV \l__spath_tmpa_tl \l__spath_tmpa_dim
3786 }
3787 }
3788
3789 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
3790 \group_end:
3791 }
3792 {
3793 \tl_gset:Nn \g__spath_output_tl {#1}
3794 }
3795 }
3796 \cs_new_protected_nopar:Npn \spath_shorten_at_end:Nnn #1#2#3

```

```

3797 {
3798   \_spath_shorten_at_end:nn {#2}{#3}
3799   \tl_set_eq:NN #1 \g__spath_output_tl
3800   \tl_gclear:N \g__spath_output_tl
3801 }
3802 \cs_generate_variant:Nn \spath_shorten_at_end:Nnn {NVV, cnn, cVV, NVn}
3803 \cs_new_protected_nopar:Npn \spath_shorten_at_end:Nn #1#2
3804 {
3805   \spath_shorten_at_end:NVn #1#1{#2}
3806 }
3807 \cs_generate_variant:Nn \spath_shorten_at_end:Nn {cn, cV, NV}
3808 \cs_new_protected_nopar:Npn \spath_gshorten_at_end:Nnn #1#2#3
3809 {
3810   \_spath_shorten_at_end:nn {#2}{#3}
3811   \tl_gset_eq:NN #1 \g__spath_output_tl
3812   \tl_gclear:N \g__spath_output_tl
3813 }
3814 \cs_generate_variant:Nn \spath_gshorten_at_end:Nnn {NVV, cnn, cVV, NVn}
3815 \cs_new_protected_nopar:Npn \spath_gshorten_at_end:Nn #1#2
3816 {
3817   \spath_gshorten_at_end:NVn #1#1{#2}
3818 }
3819 \cs_generate_variant:Nn \spath_gshorten_at_end:Nn {cn, cV, NV}

```

(End definition for \spath_shorten_at_end:Nnn.)

```

\spath_shorten_at_start:Nnn
\spath_shorten_at_start:Nn
\spath_gshorten_at_start:Nnn
\spath_gshorten_at_start:Nn

```

This macro shortens a path from the start by a dimension.

```

3820 \cs_new_protected_nopar:Npn \_spath_shorten_at_start:nn #1#2
3821 {
3822   \int_compare:nTF
3823   {
3824     \tl_count:n {#1} > 3
3825   }
3826   {
3827     \group_begin:
3828     \tl_set:Nn \l__spath_tmpa_tl {#1}
3829
3830     \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {4}}
3831
3832     \tl_clear:N \l__spath_tmpe_tl
3833
3834     \tl_if_eq:NNTF \l__spath_tmpb_tl \c_spath_curvetoa_tl
3835     {
3836       \int_set:Nn \l__spath_tmpa_int {3}
3837     }
3838     {
3839       \int_set:Nn \l__spath_tmpa_int {1}
3840     }
3841
3842     \tl_set_eq:NN \l__spath_tmpe_tl \c_spath_moveto_tl
3843     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
3844
3845     \prg_replicate:nn { \l__spath_tmpa_int }
3846     {

```

```

3847 \__spath_tl_put_right_braced:Nx
3848 \l__spath_tmpe_tl
3849 {\tl_item:Nn \l__spath_tmpa_tl {1}}
3850 \__spath_tl_put_right_braced:Nx
3851 \l__spath_tmpe_tl
3852 {\tl_item:Nn \l__spath_tmpa_tl {2}}
3853 \tl_put_right:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpa_tl {3}}
3854
3855 \prg_replicate:nn {3}
3856 {
3857   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
3858 }
3859 }
3860 \__spath_tl_put_right_braced:Nx
3861 \l__spath_tmpe_tl
3862 {\tl_item:Nn \l__spath_tmpa_tl {1}}
3863 \__spath_tl_put_right_braced:Nx
3864 \l__spath_tmpe_tl
3865 {\tl_item:Nn \l__spath_tmpa_tl {2}}
3866
3867 \fp_set:Nn \l__spath_tmpa_fp
3868 {
3869   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {5}}
3870   -
3871   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {2}}
3872 }
3873
3874 \fp_set:Nn \l__spath_tmpb_fp
3875 {
3876   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {6}}
3877   -
3878   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {3}}
3879 }
3880
3881 \fp_set:Nn \l__spath_tmpc_fp
3882 {
3883   sqrt(
3884     \l__spath_tmpa_fp * \l__spath_tmpa_fp
3885     +
3886     \l__spath_tmpb_fp * \l__spath_tmpb_fp
3887   )
3888   *
3889   \l__spath_tmpa_int
3890 }
3891
3892 \fp_compare:nTF
3893 {
3894   \l__spath_tmpc_fp > #2
3895 }
3896 {
3897
3898   \fp_set:Nn \l__spath_tmpc_fp
3899   {
3900     #2/ \l__spath_tmpc_fp

```

```

3901     }
3902
3903     \tl_if_eq:NNTF \l__spath_tmpb_tl \c_spath_curvetoa_tl
3904     {
3905         \spath_split_curve:NNVV
3906         \l__spath_tmpe_tl
3907         \l__spath_tmpe_tl
3908         \l__spath_tmpe_tl
3909         \l__spath_tmpe_tl
3910     }
3911     {
3912         \spath_split_line:NNVV
3913         \l__spath_tmpe_tl
3914         \l__spath_tmpe_tl
3915         \l__spath_tmpe_tl
3916         \l__spath_tmpe_tl
3917     }
3918
3919     \prg_replicate:nn {2}
3920     {
3921         \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3922     }
3923
3924     \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3925
3926     }
3927     {
3928
3929         \tl_put_left:NV \l__spath_tmpe_tl \c_spath_moveto_tl
3930
3931         \int_compare:nT
3932         {
3933             \tl_count:N \l__spath_tmpe_tl > 3
3934         }
3935         {
3936             \dim_set:Nn \l__spath_tmpe_dim {\fp_to_dim:n {#2 - \l__spath_tmpe_tl}}
3937             \spath_shorten_at_start:NV \l__spath_tmpe_tl \l__spath_tmpe_dim
3938         }
3939     }
3940
3941     \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
3942     \group_end:
3943     }
3944     {
3945         \tl_gset:Nn \g__spath_output_tl {#1}
3946     }
3947 }
3948 \cs_new_protected_nopar:Npn \spath_shorten_at_start:Nnn #1#2#3
3949 {
3950     \__spath_shorten_at_start:nn {#2}{#3}
3951     \tl_set_eq:NN #1 \g__spath_output_tl
3952     \tl_gclear:N \g__spath_output_tl
3953 }
3954 \cs_generate_variant:Nn \spath_shorten_at_start:Nnn {NVV, cnn, cVV, NVn}

```



```

3955 \cs_new_protected_nopar:Npn \spath_shorten_at_start:Nn #1#2
3956 {
3957   \spath_shorten_at_start:NVn #1#1{#2}
3958 }
3959 \cs_generate_variant:Nn \spath_shorten_at_start:Nn {cn, cV, NV}
3960 \cs_new_protected_nopar:Npn \spath_gshorten_at_start:Nnn #1#2#3
3961 {
3962   \__spath_shorten_at_start:nn {#2}{#3}
3963   \tl_gset_eq:NN #1 \g__spath_output_tl
3964   \tl_gclear:N \g__spath_output_tl
3965 }
3966 \cs_generate_variant:Nn \spath_gshorten_at_start:Nnn {NVV, cnn, cVV, NVn}
3967 \cs_new_protected_nopar:Npn \spath_gshorten_at_start:Nn #1#2
3968 {
3969   \spath_gshorten_at_start:NVn #1#1{#2}
3970 }
3971 \cs_generate_variant:Nn \spath_gshorten_at_start:Nn {cn, cV, NV}

```

(End definition for \spath_shorten_at_start:Nnn and others.)

```

\spath_shorten_at_both_ends:Nnn
\spath_shorten_at_both_ends:Nn
\spath_gshorten_at_both_ends:Nnn
\spath_gshorten_at_both_ends:Nn

```

This macro shortens a path from the start by a dimension.

```

3972 \cs_new_protected_nopar:Npn \spath_shorten_at_both_ends:Nnn #1#2#3
3973 {
3974   \spath_shorten_at_start:Nnn #1{#2}{#3}
3975   \spath_shorten_at_end:Nnn #1{#2}{#3}
3976 }
3977 \cs_new_protected_nopar:Npn \spath_shorten_at_both_ends:Nn #1#2
3978 {
3979   \spath_shorten_at_start:Nn #1{#2}
3980   \spath_shorten_at_end:Nn #1{#2}
3981 }
3982 \cs_generate_variant:Nn \spath_shorten_at_both_ends:Nn {cn, cV, NV}
3983 \cs_new_protected_nopar:Npn \spath_gshorten_at_both_ends:Nnn #1#2#3
3984 {
3985   \spath_gshorten_at_start:Nnn #1{#2}{#3}
3986   \spath_gshorten_at_end:Nnn #1{#2}{#3}
3987 }
3988 \cs_new_protected_nopar:Npn \spath_gshorten_at_both_ends:Nn #1#2
3989 {
3990   \spath_gshorten_at_start:Nn #1{#2}
3991   \spath_gshorten_at_end:Nn #1{#2}
3992 }
3993 \cs_generate_variant:Nn \spath_gshorten_at_both_ends:Nn {cn, cV, NV}

```

(End definition for \spath_shorten_at_both_ends:Nnn and others.)

3.6 Points on a Path

```

\spath_point_at:Nnn
\spath_gpoint_at:Nnn

```

Get the location of a point on a path, using the same location specification as the intersection library.

```

3994 \cs_new_protected_nopar:Npn \__spath_point_at:nn #1#2
3995 {
3996   \group_begin:
3997   \int_set:Nn \l__spath_tmpa_int {\fp_to_int:n {floor(#2) + 1}}

```

```

3998 \fp_set:Nn \l__spath_tmpa_fp {#2 - floor(#2)}
3999
4000 \spath_segments_to_seq:Nn \l__spath_tmpa_seq {#1}
4001
4002 \int_compare:nTF
4003 {
4004   \l__spath_tmpa_int < 1
4005 }
4006 {
4007   \spath_initialpoint:Nn \l__spath_tmpc_tl {#1}
4008 }
4009 {
4010   \int_compare:nTF
4011   {
4012     \l__spath_tmpa_int > \seq_count:N \l__spath_tmpa_seq
4013   }
4014   {
4015     \spath_finalpoint:Nn \l__spath_tmpc_tl {#1}
4016   }
4017   {
4018
4019     \tl_set:Nx
4020     \l__spath_tmpa_tl
4021     {\seq_item:Nn \l__spath_tmpa_seq { \l__spath_tmpa_int } }
4022
4023     \int_compare:nTF
4024     {
4025       \tl_count:N \l__spath_tmpa_tl > 3
4026     }
4027     {
4028       \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {4}}
4029     }
4030     {
4031       \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {1}}
4032     }
4033
4034     \tl_clear:N \l__spath_tmpc_tl
4035
4036     \tl_case:Nn \l__spath_tmpb_tl
4037     {
4038       \c_spath_moveto_tl
4039       {
4040         \tl_set:Nx \l__spath_tmpc_tl
4041         {
4042           {
4043             \tl_item:Nn \l__spath_tmpa_tl {2}
4044           }
4045           {
4046             \tl_item:Nn \l__spath_tmpa_tl {3}
4047           }
4048         }
4049       }
4050
4051     \c_spath_lineto_tl

```

```

4052 {
4053   \tl_set:Nx \l__spath_tmpc_tl
4054   {
4055     {\fp_to_dim:n
4056       {
4057         (1 - \l__spath_tmpa_fp) * ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4058         +
4059         \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4060       }
4061     }
4062     {\fp_to_dim:n
4063       {
4064         (1 - \l__spath_tmpa_fp) * ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4065         +
4066         \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4067       }
4068     }
4069   }
4070 }
4071
4072 \c_spath_rectsize_tl
4073 {
4074   \fp_compare:nTF
4075   {
4076     \l__spath_tmpa_fp <= .25
4077   }
4078   {
4079     \tl_set:Nx \l__spath_tmpc_tl
4080     {
4081       {\fp_to_dim:n
4082         {
4083           ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4084           +
4085           4 * \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4086         }
4087       }
4088       {\fp_to_dim:n { \tl_item:Nn \l__spath_tmpa_tl {3} } }
4089     }
4090   }
4091   {
4092     \fp_compare:nTF
4093     {
4094       \l__spath_tmpa_fp <= .5
4095     }
4096     {
4097       \tl_set:Nx \l__spath_tmpc_tl
4098       {
4099         {\fp_to_dim:n
4100           {
4101             ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4102             +
4103             ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4104           }
4105         }

```

```

4106         {\fp_to_dim:n
4107         {
4108             ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4109             +
4110             (4 * (\l__spath_tmpa_fp) - 1) * ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4111         }
4112     }
4113 }
4114 }
4115 {
4116     \fp_compare:nTF
4117     {
4118         \l__spath_tmpa_fp <= .75
4119     }
4120     {
4121         \tl_set:Nx \l__spath_tmpc_tl
4122         {
4123             {\fp_to_dim:n
4124             {
4125                 ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4126                 +
4127                 (3 - 4 * (\l__spath_tmpa_fp)) * ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4128             }
4129             }
4130             {\fp_to_dim:n
4131             {
4132                 ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4133                 +
4134                 ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4135             }
4136             }
4137         }
4138     }
4139 }
4140 {
4141     \tl_set:Nx \l__spath_tmpc_tl
4142     {
4143         {\fp_to_dim:n
4144         {
4145             ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4146         }
4147         }
4148         {\fp_to_dim:n
4149         {
4150             ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4151             +
4152             (4 - 4 * (\l__spath_tmpa_fp)) * ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4153         }
4154         }
4155     }
4156 }
4157 }
4158 }
4159 }

```

```

4160
4161 \c_spath_closepath_tl
4162 {
4163   \tl_set:Nx \l__spath_tmpc_tl
4164   {
4165     {\fp_to_dim:n
4166       {
4167         (1 - \l__spath_tmpa_fp) * ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4168         +
4169         \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4170       }
4171     }
4172     {\fp_to_dim:n
4173       {
4174         (1 - \l__spath_tmpa_fp) * ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4175         +
4176         \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4177       }
4178     }
4179   }
4180 }
4181
4182 \c_spath_curvetoa_tl
4183 {
4184   \tl_set:Nx \l__spath_tmpc_tl
4185   {
4186     {\fp_to_dim:n
4187       {
4188         (1 - \l__spath_tmpa_fp)^3 * \tl_item:Nn \l__spath_tmpa_tl {2}
4189         + 3 * (1 - \l__spath_tmpa_fp)^2 * (\l__spath_tmpa_fp)
4190         * \tl_item:Nn \l__spath_tmpa_tl {5}
4191         + 3 * (1 - \l__spath_tmpa_fp) * (\l__spath_tmpa_fp)^2
4192         * \tl_item:Nn \l__spath_tmpa_tl {8}
4193         + (\l__spath_tmpa_fp)^3 * \tl_item:Nn \l__spath_tmpa_tl {11}
4194       }}
4195     {\fp_to_dim:n
4196       {
4197         (1 - \l__spath_tmpa_fp)^3 * \tl_item:Nn \l__spath_tmpa_tl {3}
4198         + 3 * (1 - \l__spath_tmpa_fp)^2 * (\l__spath_tmpa_fp)
4199         * \tl_item:Nn \l__spath_tmpa_tl {6}
4200         + 3 * (1 - \l__spath_tmpa_fp) * (\l__spath_tmpa_fp)^2
4201         * \tl_item:Nn \l__spath_tmpa_tl {9}
4202         + (\l__spath_tmpa_fp)^3 * \tl_item:Nn \l__spath_tmpa_tl {12}
4203       }}
4204     }
4205   }
4206 }
4207 }
4208 }
4209
4210 \tl_gclear:N \g__spath_output_tl
4211 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpc_tl
4212 \group_end:
4213 }

```

```

4214 \cs_new_protected_nopar:Npn \spath_point_at:Nnn #1#2#3
4215 {
4216   \__spath_point_at:nn {#2}{#3}
4217   \tl_set_eq:NN #1 \g__spath_output_tl
4218   \tl_gclear:N \g__spath_output_tl
4219 }
4220 \cs_generate_variant:Nn \spath_point_at:Nnn {NVn, NVV, NnV}
4221 \cs_new_protected_nopar:Npn \spath_gpoint_at:Nnn #1#2#3
4222 {
4223   \__spath_point_at:nn {#2}{#3}
4224   \tl_gset_eq:NN #1 \g__spath_output_tl
4225   \tl_gclear:N \g__spath_output_tl
4226 }
4227 \cs_generate_variant:Nn \spath_gpoint_at:Nnn {NVn, NVV, NnV}

```

(End definition for \spath_point_at:Nnn and \spath_gpoint_at:Nnn.)

`\spath_tangent_at:Nnn` Get the tangent at a point on a path, using the same location specification as the intersection library.
`\spath_gtangent_at:Nnn`

```

4228 \cs_new_protected_nopar:Npn \__spath_tangent_at:nn #1#2
4229 {
4230   \group_begin:
4231   \int_set:Nn \l__spath_tmpa_int {\fp_to_int:n {floor(#2) + 1}}
4232   \fp_set:Nn \l__spath_tmpa_fp {#2 - floor(#2)}
4233
4234   \spath_segments_to_seq:Nn \l__spath_tmpa_seq {#1}
4235
4236   \int_compare:nTF
4237   {
4238     \l__spath_tmpa_int < 1
4239   }
4240   {
4241     \spath_initialpoint:Nn \l__spath_tmpc_tl {#1}
4242   }
4243   {
4244     \int_compare:nTF
4245     {
4246       \l__spath_tmpa_int > \seq_count:N \l__spath_tmpa_seq
4247     }
4248     {
4249       \spath_finalpoint:Nn \l__spath_tmpc_tl {#1}
4250     }
4251     {
4252
4253       \tl_set:Nx
4254       \l__spath_tmpa_tl
4255       {\seq_item:Nn \l__spath_tmpa_seq { \l__spath_tmpa_int } }
4256
4257       \int_compare:nTF
4258       {
4259         \tl_count:N \l__spath_tmpa_tl > 3
4260       }
4261       {
4262         \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {4}}

```

```

4263 }
4264 {
4265   \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {1}}
4266 }
4267
4268 \tl_clear:N \l__spath_tmpc_tl
4269
4270 \tl_case:Nn \l__spath_tmpb_tl
4271 {
4272   \c_spath_moveto_tl
4273   {
4274     \tl_set:Nx \l__spath_tmpc_tl
4275     {
4276       {
4277         \tl_item:Nn \l__spath_tmpa_tl {2}
4278       }
4279       {
4280         \tl_item:Nn \l__spath_tmpa_tl {3}
4281       }
4282     }
4283   }
4284
4285   \c_spath_lineto_tl
4286   {
4287     \tl_set:Nx \l__spath_tmpc_tl
4288     {
4289       {\fp_to_dim:n
4290         {
4291           ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4292           -
4293           ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4294         }
4295       }
4296       {\fp_to_dim:n
4297         {
4298           ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4299           -
4300           ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4301         }
4302       }
4303     }
4304   }
4305
4306   \c_spath_rectsize_tl
4307   {
4308     \fp_compare:nTF
4309     {
4310       \l__spath_tmpa_fp <= .25
4311     }
4312     {
4313       \tl_set:Nx \l__spath_tmpc_tl
4314       {
4315         {\fp_to_dim:n
4316         {

```

```

4317         \tl_item:Nn \l__spath_tmpa_tl {5}
4318     }
4319 }
4320 {Opt}
4321 }
4322 }
4323 {
4324 \fp_compare:nTF
4325 {
4326     \l__spath_tmpa_fp <= .5
4327 }
4328 {
4329     \tl_set:Nx \l__spath_tmpc_tl
4330     {
4331         {Opt}
4332         {\fp_to_dim:n
4333             {
4334                 ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4335             }
4336         }
4337     }
4338 }
4339 {
4340 \fp_compare:nTF
4341 {
4342     \l__spath_tmpa_fp <= .75
4343 }
4344 {
4345     \tl_set:Nx \l__spath_tmpc_tl
4346     {
4347         {\fp_to_dim:n
4348             {
4349                 - ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4350             }
4351         }
4352         {Opt}
4353     }
4354 }
4355 }
4356 {
4357     \tl_set:Nx \l__spath_tmpc_tl
4358     {
4359         {Opt}
4360         {\fp_to_dim:n
4361             {
4362                 - ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4363             }
4364         }
4365     }
4366 }
4367 }
4368 }
4369 }
4370

```



```

4371 \c_spath_closepath_tl
4372 {
4373   \tl_set:Nx \l__spath_tmpc_tl
4374   {
4375     {\fp_to_dim:n
4376       {
4377         ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4378         -
4379         ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4380       }
4381     }
4382     {\fp_to_dim:n
4383       {
4384         ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4385         -
4386         ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4387       }
4388     }
4389   }
4390 }
4391
4392 \c_spath_curvetoa_tl
4393 {
4394   \tl_set:Nx \l__spath_tmpc_tl
4395   {
4396     {\fp_to_dim:n
4397       {
4398         3*(1 - \l__spath_tmpa_fp)^2 * (\tl_item:Nn \l__spath_tmpa_tl {5}
4399         - \tl_item:Nn \l__spath_tmpa_tl {2})
4400         + 6 * (1 - \l__spath_tmpa_fp) * (\l__spath_tmpa_fp) *
4401         (\tl_item:Nn \l__spath_tmpa_tl {8}
4402         - \tl_item:Nn \l__spath_tmpa_tl {5})
4403         + 3*(\l__spath_tmpa_fp)^2 * (\tl_item:Nn \l__spath_tmpa_tl {11}
4404         - \tl_item:Nn \l__spath_tmpa_tl {8})
4405       }
4406     }
4407     {\fp_to_dim:n
4408       {
4409         3*(1 - \l__spath_tmpa_fp)^2 * (\tl_item:Nn \l__spath_tmpa_tl {6}
4410         - \tl_item:Nn \l__spath_tmpa_tl {3})
4411         + 6 * (1 - \l__spath_tmpa_fp) * (\l__spath_tmpa_fp) *
4412         (\tl_item:Nn \l__spath_tmpa_tl {9}
4413         - \tl_item:Nn \l__spath_tmpa_tl {6})
4414         + 3*(\l__spath_tmpa_fp)^2 * (\tl_item:Nn \l__spath_tmpa_tl {12}
4415         - \tl_item:Nn \l__spath_tmpa_tl {9})
4416       }
4417     }
4418   }
4419 }
4420 }
4421 }
4422
4423 \tl_gclear:N \g__spath_output_tl
4424 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpc_tl

```

```

4425 \group_end:
4426 }
4427 \cs_new_protected_nopar:Npn \spath_tangent_at:Nnn #1#2#3
4428 {
4429   \__spath_tangent_at:nn {#2}{#3}
4430   \tl_set_eq:NN #1 \g__spath_output_tl
4431   \tl_gclear:N \g__spath_output_tl
4432 }
4433 \cs_generate_variant:Nn \spath_tangent_at:Nnn {NVn, NVV, NnV}
4434 \cs_new_protected_nopar:Npn \spath_gtangent_at:Nnn #1#2#3
4435 {
4436   \__spath_tangent_at:nn {#2}{#3}
4437   \tl_gset_eq:NN #1 \g__spath_output_tl
4438   \tl_gclear:N \g__spath_output_tl
4439 }
4440 \cs_generate_variant:Nn \spath_gtangent_at:Nnn {NVn, NVV, NnV}

```

(End definition for \spath_tangent_at:Nnn and \spath_gtangent_at:Nnn.)

`\spath_transformation_at:Nnn` Gets a transformation that will align to a point on the path with the x-axis along the path.

`\spath_gtransformation_at:Nnn`

```

4441 \cs_new_protected_nopar:Npn \__spath_transformation_at:nn #1#2
4442 {
4443   \group_begin:
4444   \tl_clear:N \l__spath_tmpa_tl
4445   \__spath_tangent_at:nn {#1}{#2}
4446   \tl_set_eq:NN \l__spath_tmpb_tl \g__spath_output_tl
4447   \fp_set:Nn \l__spath_tmpa_fp
4448   {
4449     sqrt(
4450       (\tl_item:Nn \l__spath_tmpb_tl {1})^2
4451       +
4452       (\tl_item:Nn \l__spath_tmpb_tl {2})^2
4453     )
4454   }
4455   \fp_compare:nTF {\l__spath_tmpa_fp = 0}
4456   {
4457     \fp_set:Nn \l__spath_tmpa_fp {1}
4458     \fp_set:Nn \l__spath_tmpb_fp {0}
4459   }
4460   {
4461     \fp_set:Nn \l__spath_tmpb_fp
4462     { (\tl_item:Nn \l__spath_tmpb_tl {2}) / \l__spath_tmpa_fp }
4463     \fp_set:Nn \l__spath_tmpa_fp
4464     { (\tl_item:Nn \l__spath_tmpb_tl {1}) / \l__spath_tmpa_fp }
4465   }
4466   \tl_set:Nx \l__spath_tmpa_tl
4467   {
4468     { \fp_to_decimal:n { \l__spath_tmpa_fp } }
4469     { \fp_to_decimal:n { \l__spath_tmpb_fp } }
4470     { \fp_to_decimal:n { - \l__spath_tmpb_fp } }
4471     { \fp_to_decimal:n { \l__spath_tmpa_fp } }
4472   }
4473   \__spath_point_at:nn {#1}{#2}

```

```

4474 \tl_put_right:NV \l__spath_tmpa_tl \g__spath_output_tl
4475 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
4476 \group_end:
4477 }
4478 \cs_new_protected_nopar:Npn \spath_transformation_at:Nnn #1#2#3
4479 {
4480   \__spath_transformation_at:nn {#2}{#3}
4481   \tl_set_eq:NN #1 \g__spath_output_tl
4482   \tl_gclear:N \g__spath_output_tl
4483 }
4484 \cs_generate_variant:Nn \spath_transformation_at:Nnn {NVn, NVV, NnV, NvV}
4485 \cs_new_protected_nopar:Npn \spath_gtransformation_at:Nnn #1#2#3
4486 {
4487   \__spath_transformation_at:nn {#2}{#3}
4488   \tl_gset_eq:NN #1 \g__spath_output_tl
4489   \tl_gclear:N \g__spath_output_tl
4490 }
4491 \cs_generate_variant:Nn \spath_gtransformation_at:Nnn {NVn, NVV, NnV}

```

(End definition for \spath_transformation_at:Nnn and \spath_gtransformation_at:Nnn.)

3.7 Intersection Routines

Note: I'm not consistent with number schemes. The intersection library is 0-based, but the user interface is 1-based (since if we “count” in a \foreach then it starts at 1). This should be more consistent.

```

\spath_intersect:NN Pass two spaths to pgf's intersection routine.
\spath_intersect:nn
4492 \cs_new_protected_nopar:Npn \spath_intersect:NN #1#2
4493 {
4494   \pgfintersectionofpaths%
4495   {%
4496     \pgfsetpath #1
4497   }{%
4498     \pgfsetpath #2
4499   }
4500 }
4501 \cs_new_protected_nopar:Npn \spath_intersect:nn #1#2
4502 {
4503   \tl_set:Nn \l__spath_intersecta_tl {#1}
4504   \tl_set:Nn \l__spath_intersectb_tl {#2}
4505   \spath_intersect:NN \l__spath_intersecta_tl \l__spath_intersectb_tl
4506 }

```

(End definition for \spath_intersect:NN and \spath_intersect:nn.)

\spath_split_component_at_intersections:Nnn Split a component where it intersects a path. Key assumption is that the first path is a single component, so if it is closed then the end joins up to the beginning. The component is modified but the path is not.

```

4507 \cs_new_protected_nopar:Npn \__spath_split_component_at_intersections:nn #1#2
4508 {
4509   \group_begin:
4510   \tl_clear:N \l__spath_tmpe_tl
4511   \seq_clear:N \l__spath_tmpb_seq

```

```

4512
4513 % Find the intersections of these segments
4514 \tl_set:Nn \l__spath_tmpb_tl {#1}
4515 \tl_set:Nn \l__spath_tmpc_tl {#2}
4516
4517 % Remember if the component is closed
4518 \spath_finalaction:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
4519
4520 \bool_set:Nn \l__spath_closed_bool
4521 {
4522   \tl_if_eq_p:NN \l__spath_tmpa_tl \c_spath_closepath_tl
4523   ||
4524   \tl_if_eq_p:NN \l__spath_tmpa_tl \c_spath_rectcorner_tl
4525 }
4526
4527 % Open it
4528 \spath_open:N \l__spath_tmpb_tl
4529
4530 \spath_reallength:NV \l__spath_tmpa_int \l__spath_tmpb_tl
4531
4532 % Sort intersections along the component
4533 \pgfintersectionsorthbyfirstpath
4534 \spath_intersect:NN \l__spath_tmpb_tl \l__spath_tmpc_tl
4535
4536 % If we get intersections
4537 \int_compare:nT {\pgfintersectionsolutions > 0}
4538 {
4539   % Find the times of the intersections on the component
4540   \int_step_inline:nnnn {1} {1} {\pgfintersectionsolutions}
4541   {
4542     \pgfintersectiongetsolutiontimes{##1}{\l__spath_tmph_tl}{\l__spath_tmpl_tl}
4543     \seq_put_left:NV \l__spath_tmph_seq \l__spath_tmph_tl
4544   }
4545
4546   \seq_get_left:NN \l__spath_tmph_seq \l__spath_tmpa_tl
4547   \fp_compare:nT
4548   {
4549     \l__spath_tmph_tl > \l__spath_tmpa_int - .01
4550   }
4551   {
4552     \bool_set_false:N \l__spath_closed_bool
4553   }
4554
4555   \seq_get_right:NN \l__spath_tmph_seq \l__spath_tmpa_tl
4556   \fp_compare:nT
4557   {
4558     \l__spath_tmph_tl < .01
4559   }
4560   {
4561     \bool_set_false:N \l__spath_closed_bool
4562   }
4563
4564   \tl_set:Nn \l__spath_tmpg_tl {-1}
4565

```

```

4566 \seq_map_inline:Nn \l__spath_tmpb_seq
4567 {
4568   \tl_set:Nn \l__spath_tmph_tl {##1}
4569
4570   \tl_set_eq:NN \l__spath_tmpha_tl \l__spath_tmph_tl
4571   \int_compare:nT
4572   {
4573     \fp_to_int:n {floor( \l__spath_tmph_tl ) }
4574     =
4575     \fp_to_int:n {floor( \l__spath_tmpg_tl ) }
4576   }
4577   {
4578     \tl_set:Nx \l__spath_tmph_tl
4579     {
4580       \fp_eval:n {
4581         floor( \l__spath_tmph_tl )
4582         +
4583         ( \l__spath_tmph_tl - floor( \l__spath_tmph_tl ) )
4584         /
4585         ( \l__spath_tmpg_tl - floor( \l__spath_tmpg_tl ) )
4586       }
4587     }
4588   }
4589   \tl_set_eq:NN \l__spath_tmpg_tl \l__spath_tmpha_tl
4590
4591   \spath_split_at:NNVV
4592   \l__spath_tmpd_tl
4593   \l__spath_tmpf_tl
4594   \l__spath_tmpb_tl
4595   \l__spath_tmph_tl
4596
4597   \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpf_tl
4598   \tl_set_eq:NN \l__spath_tmpb_tl \l__spath_tmpd_tl
4599
4600 }
4601
4602
4603 \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpb_tl
4604
4605 \spath_remove_empty_components:N \l__spath_tmpe_tl
4606
4607 \tl_set_eq:NN \l__spath_tmpb_tl \l__spath_tmpe_tl
4608 }
4609
4610
4611 \bool_if:NT \l__spath_closed_bool
4612 {
4613   \spath_join_component:Nn \l__spath_tmpb_tl {1}
4614 }
4615
4616 \tl_gclear:N \g__spath_output_tl
4617 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
4618
4619 \group_end:

```

```

4620 }
4621 \cs_new_protected_nopar:Npn \spath_split_component_at_intersections:Nnn #1#2#3
4622 {
4623   \__spath_split_component_at_intersections:nn {#2}{#3}
4624   \tl_set_eq:NN #1 \g__spath_output_tl
4625   \tl_gclear:N \g__spath_output_tl
4626 }
4627 \cs_generate_variant:Nn \spath_split_component_at_intersections:Nnn {NVn, NVV}
4628 \cs_new_protected_nopar:Npn \spath_split_component_at_intersections:Nn #1#2
4629 {
4630   \spath_split_component_at_intersections:NVn #1#1{#2}
4631 }
4632 \cs_generate_variant:Nn \spath_split_component_at_intersections:Nn {cn, cv}
4633 \cs_new_protected_nopar:Npn \spath_gsplit_component_at_intersections:Nnn #1#2#3
4634 {
4635   \__spath_split_component_at_intersections:nn {#2}{#3}
4636   \tl_gset_eq:NN #1 \g__spath_output_tl
4637   \tl_gclear:N \g__spath_output_tl
4638 }
4639 \cs_generate_variant:Nn \spath_gsplit_component_at_intersections:Nnn {NVn, NVV}
4640 \cs_new_protected_nopar:Npn \spath_gsplit_component_at_intersections:Nn #1#2
4641 {
4642   \spath_gsplit_component_at_intersections:NVn #1#1{#2}
4643 }
4644 \cs_generate_variant:Nn \spath_gsplit_component_at_intersections:Nn {cn, cv}

```

(End definition for \spath_split_component_at_intersections:Nnn.)

\spath_split_path_at_intersections:Nnn Split paths at their intersections. The path versions only split the first path. The others split both paths.

```

\spath_split_path_at_intersections:Nn
\spath_gsplit_path_at_intersections:Nnn
\spath_split_at_intersections:NNnn
\spath_split_at_intersections:NN
\spath_gsplit_at_intersections:NNnn
\spath_gsplit_at_intersections:NN
4645 \cs_new_protected_nopar:Npn \__spath_split_path_at_intersections:nn #1#2
4646 {
4647   \group_begin:
4648
4649   \seq_clear:N \l__spath_tmpa_seq
4650   \seq_clear:N \l__spath_tmpb_seq
4651
4652   \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
4653   \seq_map_inline:Nn \l__spath_tmpa_seq
4654   {
4655     \spath_split_component_at_intersections:Nnn \l__spath_tmpa_tl {##1} {#2}
4656     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
4657   }
4658
4659   \tl_gclear:N \g__spath_output_tl
4660   \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpb_seq {} }
4661   \group_end:
4662 }
4663 \cs_new_protected_nopar:Npn \spath_split_path_at_intersections:Nnn #1#2#3
4664 {
4665   \__spath_split_path_at_intersections:nn {#2}{#3}
4666   \tl_set_eq:NN #1 \g__spath_output_tl
4667   \tl_gclear:N \g__spath_output_tl
4668 }

```

```

4669 \cs_generate_variant:Nn \spath_split_path_at_intersections:Nnn
4670 {NVn, NVV, cVn, cVV, cvn, cvv}
4671 \cs_new_protected_nopar:Npn \spath_split_path_at_intersections:Nn #1#2
4672 {
4673   \spath_split_path_at_intersections:NVn #1#1{#2}
4674 }
4675 \cs_generate_variant:Nn \spath_split_path_at_intersections:Nn {cv, NV}
4676 \cs_new_protected_nopar:Npn \spath_gsplit_path_at_intersections:Nnn #1#2#3
4677 {
4678   \__spath_split_path_at_intersections:nm {#2}{#3}
4679   \tl_gset_eq:NN #1 \g__spath_output_tl
4680   \tl_gclear:N \g__spath_output_tl
4681 }
4682 \cs_generate_variant:Nn \spath_gsplit_path_at_intersections:Nnn
4683 {NVn, NVV, cVn, cVV, cvn, cvv}
4684 \cs_new_protected_nopar:Npn \spath_gsplit_path_at_intersections:Nn #1#2
4685 {
4686   \spath_gsplit_path_at_intersections:NVn #1#1{#2}
4687 }
4688 \cs_generate_variant:Nn \spath_gsplit_path_at_intersections:Nn {cv, NV}
4689 \cs_new_protected_nopar:Npn \spath_split_at_intersections:NNnn #1#2#3#4
4690 {
4691   \__spath_split_path_at_intersections:nm {#3}{#4}
4692   \tl_set_eq:NN #1 \g__spath_output_tl
4693   \__spath_split_path_at_intersections:nm {#4}{#3}
4694   \tl_set_eq:NN #2 \g__spath_output_tl
4695   \tl_gclear:N \g__spath_output_tl
4696 }
4697 \cs_generate_variant:Nn \spath_split_at_intersections:NNnn
4698 {NNVn, NNVV, ccVn, ccVV, ccvn, ccvv}
4699 \cs_new_protected_nopar:Npn \spath_split_at_intersections:NN #1#2
4700 {
4701   \spath_split_at_intersections:NNVV #1#2#1#2
4702 }
4703 \cs_generate_variant:Nn \spath_split_at_intersections:NN {cc}
4704 \cs_new_protected_nopar:Npn \spath_gsplit_at_intersections:NNnn #1#2#3#4
4705 {
4706   \__spath_split_path_at_intersections:nm {#3}{#4}
4707   \tl_gset_eq:NN #1 \g__spath_output_tl
4708   \__spath_split_path_at_intersections:nm {#4}{#3}
4709   \tl_gset_eq:NN #2 \g__spath_output_tl
4710   \tl_gclear:N \g__spath_output_tl
4711 }
4712 \cs_generate_variant:Nn \spath_gsplit_at_intersections:NNnn
4713 {NNVn, NNVV, ccVn, ccVV, ccvn, ccvv}
4714 \cs_new_protected_nopar:Npn \spath_gsplit_at_intersections:NN #1#2
4715 {
4716   \spath_gsplit_at_intersections:NNVV #1#2#1#2
4717 }
4718 \cs_generate_variant:Nn \spath_gsplit_at_intersections:NN {cc}

```

(End definition for \spath_split_path_at_intersections:Nnn and others.)

`th_split_component_at_self_intersections:Nn`
`ath_split_component_at_self_intersections:N`
`h_gsplit_component_at_self_intersections:Nn`
`th_gsplit_component_at_self_intersections:N`

Given a component of a path, split it at points where it self-intersects.

```

4719 \cs_new_protected_nopar:Npn \__spath_split_component_at_self_intersections:n #1
4720 {
4721   \group_begin:
4722   \tl_set:Nn \l__spath_tmpe_tl {#1}
4723
4724   % Remember if the component is closed
4725   \spath_finalaction:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
4726
4727   \bool_set:Nn \l__spath_closed_bool
4728   {
4729     \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_closepath_tl
4730   }
4731
4732   % Copy the path
4733   \tl_set:Nn \l__spath_tmpe_tl {#1}
4734
4735   % Open the path
4736   \spath_open:N \l__spath_tmpe_tl
4737   % Ensure beziers don't self-intersect
4738   \spath_split_curves:N \l__spath_tmpe_tl
4739
4740   % Make a copy for later
4741   \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpe_tl
4742
4743   % Clear some token lists and sequences
4744   \tl_clear:N \l__spath_tmpd_tl
4745   \seq_clear:N \l__spath_tmpe_seq
4746   \int_zero:N \l__spath_tmpe_int
4747
4748   \pgfintersectionsorderbyfirstpath
4749
4750   % Split the path into a sequence of segments
4751   \spath_segments_to_seq:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
4752
4753   \seq_map_indexed_inline:Nn \l__spath_tmpe_seq
4754   {
4755     \seq_map_indexed_inline:Nn \l__spath_tmpe_seq
4756     {
4757       % Don't intersect a segment with itself
4758       \int_compare:nF
4759       {
4760         ##1 == ####1
4761       }
4762       {
4763         \spath_intersect:nn {##2} {####2}
4764
4765         \int_compare:nT {\pgfintersectionsolutions > 0}
4766         {
4767           % Find the times of the intersections on each path
4768           \int_step_inline:mnnn {1} {1} {\pgfintersectionsolutions}
4769           {
4770             \pgfintersectiongetsolutiontimes
4771             {#####1}{\l__spath_tmpe_tl}{\l__spath_tmpe_tl}
4772

```



```

4773     \bool_if:nT
4774     {
4775         !(
4776         \fp_compare_p:n { \l__spath_tmpb_tl > .99 }
4777         &&
4778         \int_compare_p:n {##1 + 1 == ###1}
4779         )
4780         &&
4781         !(
4782         \fp_compare_p:n { \l__spath_tmpb_tl < .01 }
4783         &&
4784         \int_compare_p:n {##1 - 1 == ###1}
4785         )
4786         &&
4787         !(
4788         \l__spath_closed_bool
4789         &&
4790         \fp_compare_p:n { \l__spath_tmpb_tl < .01 }
4791         &&
4792         \int_compare_p:n {##1 == 1}
4793         &&
4794         \int_compare_p:n {\seq_count:N \l__spath_tmpa_seq == ###1}
4795         )
4796         &&
4797         !(
4798         \l__spath_closed_bool
4799         &&
4800         \fp_compare_p:n { \l__spath_tmpb_tl > .99 }
4801         &&
4802         \int_compare_p:n {####1 == 1}
4803         &&
4804         \int_compare_p:n {\seq_count:N \l__spath_tmpa_seq == #1}
4805         )
4806     }
4807     {
4808     \tl_set:Nx \l__spath_tmpa_tl
4809     {\fp_to_decimal:n {\l__spath_tmpb_tl + ##1 - 1}}
4810     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
4811     }
4812   }
4813 }
4814 }
4815 }
4816 }
4817
4818 % Sort the sequence by reverse order along the path
4819 \seq_sort:Nn \l__spath_tmpb_seq
4820 {
4821   \fp_compare:nNnTF { ##1 } < { ##2 }
4822   { \sort_return_swapped: }
4823   { \sort_return_same: }
4824 }
4825
4826 \seq_get_left:NN \l__spath_tmpb_seq \l__spath_tmpa_tl

```

```

4827 \fp_compare:nT
4828 {
4829   \l__spath_tmpa_tl > \seq_count:N \l__spath_tmpa_seq - .01
4830 }
4831 {
4832   \bool_set_false:N \l__spath_closed_bool
4833 }
4834 \seq_get_right:NN \l__spath_tmpb_seq \l__spath_tmpa_tl
4835 \fp_compare:nT
4836 {
4837   \l__spath_tmpa_tl < .01
4838 }
4839 {
4840   \bool_set_false:N \l__spath_closed_bool
4841 }
4842
4843 % Restore the original copy of the path
4844 \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpg_tl
4845
4846 % Clear the token lists
4847 \tl_clear:N \l__spath_tmpf_tl
4848 \tl_clear:N \l__spath_tmph_tl
4849 \tl_clear:N \l__spath_tmpg_tl
4850
4851 \tl_set:Nn \l__spath_tmpe_tl {-1}
4852
4853 \seq_map_inline:Nn \l__spath_tmpb_seq
4854 {
4855   \tl_set:Nn \l__spath_tmpb_tl {##1}
4856   \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpb_tl
4857   \int_compare:nT
4858   {
4859     \fp_to_int:n {floor( \l__spath_tmpb_tl ) }
4860     =
4861     \fp_to_int:n {floor( \l__spath_tmpe_tl ) }
4862   }
4863   {
4864     \tl_set:Nx \l__spath_tmpb_tl
4865     {
4866       \fp_eval:n {
4867         floor( \l__spath_tmpb_tl )
4868         +
4869         ( \l__spath_tmpb_tl - floor( \l__spath_tmpb_tl ) )
4870         /
4871         ( \l__spath_tmpe_tl - floor( \l__spath_tmpe_tl ) )
4872       }
4873     }
4874   }
4875   \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpe_tl
4876
4877   \spath_split_at:NNVV
4878   \l__spath_tmpf_tl
4879   \l__spath_tmph_tl
4880   \l__spath_tmpe_tl

```

```

4881     \l__spath_tmpb_tl
4882
4883     \tl_put_left:NV \l__spath_tmpg_tl \l__spath_tmph_tl
4884     \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmph_tl
4885
4886   }
4887
4888   \tl_put_left:NV \l__spath_tmpg_tl \l__spath_tmpe_tl
4889
4890   \tl_if_empty:NT \l__spath_tmpg_tl
4891   {
4892     \tl_set_eq:NN \l__spath_tmpg_tl \l__spath_tmpe_tl
4893   }
4894
4895   \spath_remove_empty_components:N \l__spath_tmpg_tl
4896
4897   % Do something with closed
4898   \bool_if:NT \l__spath_closed_bool
4899   {
4900     \spath_join_component:Nn \l__spath_tmpg_tl {1}
4901   }
4902
4903   \tl_gclear:N \g__spath_output_tl
4904   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpg_tl
4905   \group_end:
4906 }
4907 \cs_new_protected_nopar:Npn \spath_split_component_at_self_intersections:Nn #1#2
4908 {
4909   \__spath_split_component_at_self_intersections:n {#2}
4910   \tl_set_eq:NN #1 \g__spath_output_tl
4911   \tl_gclear:N \g__spath_output_tl
4912 }
4913 \cs_generate_variant:Nn \spath_split_component_at_self_intersections:Nn {NV}
4914 \cs_new_protected_nopar:Npn \spath_split_component_at_self_intersections:N #1
4915 {
4916   \spath_split_component_at_self_intersections:NV #1#1
4917 }
4918 \cs_generate_variant:Nn \spath_split_component_at_self_intersections:N {c}
4919 \cs_new_protected_nopar:Npn \spath_gsplit_component_at_self_intersections:Nn #1#2
4920 {
4921   \__spath_split_component_at_self_intersections:n {#2}
4922   \tl_gset_eq:NN #1 \g__spath_output_tl
4923   \tl_gclear:N \g__spath_output_tl
4924 }
4925 \cs_generate_variant:Nn \spath_gsplit_component_at_self_intersections:Nn {NV}
4926 \cs_new_protected_nopar:Npn \spath_gsplit_component_at_self_intersections:N #1
4927 {
4928   \spath_gsplit_component_at_self_intersections:NV #1#1
4929 }
4930 \cs_generate_variant:Nn \spath_gsplit_component_at_self_intersections:N {c}

```

(End definition for `\spath_split_component_at_self_intersections:Nn` and others.)

`\spath_split_at_self_intersections:Nn` Split a path at its self intersections. We iterate over the components, splitting each where it meets all the others and itself. To make this more efficient, we split against the

`\spath_split_at_self_intersections:N`

`\spath_gsplit_at_self_intersections:Nn`

`\spath_gsplit_at_self_intersections:N`

components of the original path rather than updating each time.

```

4931 \cs_new_protected_nopar:Npn \__spath_split_at_self_intersections:n #1
4932 {
4933   \group_begin:
4934   \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
4935   \seq_clear:N \l__spath_tmpb_seq
4936   \seq_clear:N \l__spath_tmpc_seq
4937
4938   % Iterate over the components of the original path.
4939   \bool_do_until:nn
4940   {
4941     \seq_if_empty_p:N \l__spath_tmpa_seq
4942   }
4943   {
4944     % Get the next component
4945     \seq_pop_left:NN \l__spath_tmpa_seq \l__spath_tmpa_tl
4946     % Copy for later
4947     \tl_set_eq:NN \l__spath_tmpc_tl \l__spath_tmpa_tl
4948     \int_compare:nT
4949     {
4950       \tl_count:N \l__spath_tmpa_tl > 3
4951     }
4952     {
4953
4954       % Split against itself
4955       \spath_split_component_at_self_intersections:N \l__spath_tmpa_tl
4956       % Grab the rest of the path
4957       \tl_set:Nx \l__spath_tmpb_tl
4958       {
4959         \seq_use:Nn \l__spath_tmpb_seq {}
4960         \seq_use:Nn \l__spath_tmpa_seq {}
4961       }
4962       % Split against the rest of the path
4963       \spath_split_path_at_intersections:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
4964     }
4965     % Save the split path
4966     \seq_put_right:NV \l__spath_tmpc_seq \l__spath_tmpa_tl
4967     % Add the original copy to the sequence of processed components
4968     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpc_tl
4969   }
4970
4971   \tl_gclear:N \g__spath_output_tl
4972   \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpc_seq {} }
4973   \group_end:
4974 }
4975 \cs_generate_variant:Nn \__spath_split_at_self_intersections:n {V, v}
4976 \cs_new_protected_nopar:Npn \spath_split_at_self_intersections:Nn #1#2
4977 {
4978   \__spath_split_at_self_intersections:n {#2}
4979   \tl_set_eq:NN #1 \g__spath_output_tl
4980   \tl_gclear:N \g__spath_output_tl
4981 }
4982 \cs_generate_variant:Nn \spath_split_at_self_intersections:Nn {NV, cn, cV, cv}
4983 \cs_new_protected_nopar:Npn \spath_split_at_self_intersections:N #1

```

```

4984 {
4985   \spath_split_at_self_intersections:NV #1#1
4986 }
4987 \cs_generate_variant:Nn \spath_split_at_self_intersections:N {c}
4988 \cs_new_protected_nopar:Npn \spath_gsplit_at_self_intersections:Nn #1#2
4989 {
4990   \__spath_split_at_self_intersections:n {#2}
4991   \tl_gset_eq:NN #1 \g__spath_output_tl
4992   \tl_gclear:N \g__spath_output_tl
4993 }
4994 \cs_generate_variant:Nn \spath_gsplit_at_self_intersections:Nn {NV, cn, cV, cv}
4995 \cs_new_protected_nopar:Npn \spath_gsplit_at_self_intersections:N #1
4996 {
4997   \spath_gsplit_at_self_intersections:NV #1#1
4998 }
4999 \cs_generate_variant:Nn \spath_gsplit_at_self_intersections:N {c}

```

(End definition for \spath_split_at_self_intersections:Nn and others.)

```

\spath_join_component:Nnn Join the specified component of the spath to its predecessor.
\spath_join_component:Nn
\spath_gjoin_component:Nnn
\spath_gjoin_component:Nn
5000 \cs_new_protected_nopar:Npn \__spath_join_component:nn #1#2
5001 {
5002   \group_begin:
5003   \spath_numberofcomponents:Nn \l__spath_tmpa_int {#1}
5004
5005   \bool_if:nTF
5006   {
5007     \int_compare_p:n { #2 >= 1 }
5008     &&
5009     \int_compare_p:n { #2 <= \l__spath_tmpa_int }
5010   }
5011   {
5012     \int_compare:nTF
5013     {
5014       #2 == 1
5015     }
5016     {
5017       \int_compare:nTF
5018       {
5019         \l__spath_tmpa_int == 1
5020       }
5021       {
5022         \tl_set:Nn \l__spath_tmpa_tl {#1}
5023         \spath_initialpoint:Nn \l__spath_tmpb_tl {#1}
5024         \tl_put_right:NV \l__spath_tmpa_tl \c_spath_closepath_tl
5025         \tl_put_right:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
5026         \tl_gclear:N \g__spath_output_tl
5027         \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
5028       }
5029     }
5030     \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
5031     \seq_pop_left:NN \l__spath_tmpa_seq \l__spath_tmpa_tl
5032
5033     \prg_replicate:nn {3}

```

```

5034     {
5035         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
5036     }
5037
5038     \seq_put_right:NV \l__spath_tmpa_seq \l__spath_tmpa_tl
5039
5040     \tl_gclear:N \g__spath_output_tl
5041     \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpa_seq {}}
5042 }
5043 }
5044 {
5045     \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
5046
5047     \seq_clear:N \l__spath_tmpb_seq
5048     \seq_map_indexed_inline:Nn \l__spath_tmpa_seq
5049     {
5050         \tl_set:Nn \l__spath_tmpa_tl {##2}
5051         \int_compare:nT {##1 = #2}
5052         {
5053             \prg_replicate:nn {3}
5054             {
5055                 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
5056             }
5057         }
5058         \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
5059     }
5060
5061     \tl_gclear:N \g__spath_output_tl
5062     \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpb_seq {}}
5063 }
5064 }
5065 {
5066     \tl_gclear:N \g__spath_output_tl
5067     \tl_gset:Nn \g__spath_output_tl {#1}
5068 }
5069
5070 \group_end:
5071 }
5072 \cs_new_protected_nopar:Npn \spath_join_component:Nnn #1#2#3
5073 {
5074     \__spath_join_component:nn {#2}{#3}
5075     \tl_set_eq:NN #1 \g__spath_output_tl
5076     \tl_gclear:N \g__spath_output_tl
5077 }
5078 \cs_generate_variant:Nn \spath_join_component:Nnn {NVn, NVV}
5079 \cs_new_protected_nopar:Npn \spath_join_component:Nn #1#2
5080 {
5081     \spath_join_component:NVn #1#1{#2}
5082 }
5083 \cs_generate_variant:Nn \spath_join_component:Nn {cn, NV, cV}
5084 \cs_new_protected_nopar:Npn \spath_gjoin_component:Nnn #1#2#3
5085 {
5086     \__spath_join_component:nn {#2}{#3}
5087     \tl_gset_eq:NN #1 \g__spath_output_tl

```

```

5088 \tl_gclear:N \g__spath_output_tl
5089 }
5090 \cs_generate_variant:Nn \spath_gjoin_component:Nnn {NVn, NVV}
5091 \cs_new_protected_nopar:Npn \spath_gjoin_component:Nn #1#2
5092 {
5093   \spath_gjoin_component:NVn #1#1{#2}
5094 }
5095 \cs_generate_variant:Nn \spath_gjoin_component:Nn {cn, NV, cV}

```

(End definition for \spath_join_component:Nnn and others.)

\spath_spot_weld_components:Nn Weld together any components where the last point of one is at the start point of the next (within a tolerance).

```

\spath_spot_weld_components:N
\spath_spot_gweld_components:Nn
\spath_spot_gweld_components:N
5096 \cs_new_protected_nopar:Npn \__spath_spot_weld_components:n #1
5097 {
5098   \group_begin:
5099   \dim_zero:N \l__spath_move_x_dim
5100   \dim_zero:N \l__spath_move_y_dim
5101
5102   \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
5103   \seq_clear:N \l__spath_tmpb_seq
5104   \dim_set:Nn \l__spath_move_x_dim {\tl_item:nn {#1} {2} + 10 pt}
5105   \dim_set:Nn \l__spath_move_y_dim {\tl_item:nn {#1} {3} + 10 pt}
5106
5107   \int_set:Nn \l__spath_tmpa_int {\seq_count:N \l__spath_tmpa_seq}
5108
5109   \seq_map_inline:Nn \l__spath_tmpa_seq
5110   {
5111     \tl_set:Nn \l__spath_tmpa_tl {##1}
5112     \bool_if:nT
5113     {
5114       \dim_compare_p:n
5115       {
5116         \dim_abs:n
5117         {\l__spath_move_x_dim - \tl_item:Nn \l__spath_tmpa_tl {2} }
5118         < 0.01pt
5119       }
5120       &&
5121       \dim_compare_p:n
5122       {
5123         \dim_abs:n
5124         {\l__spath_move_y_dim - \tl_item:Nn \l__spath_tmpa_tl {3} }
5125         < 0.01pt
5126       }
5127     }
5128     {
5129       \prg_replicate:nn {3}
5130       {
5131         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
5132       }
5133       \int_decr:N \l__spath_tmpa_int
5134     }
5135     \tl_reverse:N \l__spath_tmpa_tl
5136     \dim_set:Nn \l__spath_move_x_dim {\tl_item:Nn \l__spath_tmpa_tl {2}}

```

```

5137   \dim_set:Nn \l__spath_move_y_dim {\tl_item:Nn \l__spath_tmpa_tl {1}}
5138   \tl_reverse:N \l__spath_tmpa_tl
5139   \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
5140 }
5141
5142 \tl_set:Nx \l__spath_tmpa_tl {\seq_use:Nn \l__spath_tmpb_seq {} }
5143 \spath_components_to_seq:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
5144
5145
5146 \spath_initialpoint:Nn \l__spath_tmpa_tl {#1}
5147 \spath_finalpoint:Nn \l__spath_tmpb_tl {#1}
5148
5149 \bool_if:nT
5150 {
5151   \dim_compare_p:n
5152   {
5153     \dim_abs:n
5154     {
5155       \tl_item:Nn \l__spath_tmpa_tl {1} - \tl_item:Nn \l__spath_tmpb_tl {1}
5156     }
5157     <
5158     0.01pt
5159   }
5160   &&
5161   \dim_compare_p:n
5162   {
5163     \dim_abs:n
5164     {
5165       \tl_item:Nn \l__spath_tmpa_tl {2} - \tl_item:Nn \l__spath_tmpb_tl {2}
5166     }
5167     <
5168     0.01pt
5169   }
5170 }
5171 {
5172   \int_compare:nTF
5173   {
5174     \seq_count:N \l__spath_tmpb_seq > 1
5175   }
5176   {
5177     \seq_pop_left:NN \l__spath_tmpb_seq \l__spath_tmpb_tl
5178
5179     \prg_replicate:nn {3}
5180     {
5181       \tl_set:Nx \l__spath_tmpb_tl {\tl_tail:N \l__spath_tmpb_tl}
5182     }
5183     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpb_tl
5184   }
5185   {
5186     \tl_set:NV \l__spath_tmpb_tl \c_spath_closepath_tl
5187     \tl_put_right:Nx \l__spath_tmpb_tl
5188     {
5189       { \tl_item:Nn \l__spath_tmpa_tl {1} }
5190       { \tl_item:Nn \l__spath_tmpa_tl {2} }

```



```

5191     }
5192     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpb_tl
5193   }
5194 }
5195
5196 \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpb_seq {}}
5197 \group_end:
5198 }
5199 \cs_new_protected_nopar:Npn \spath_spot_weld_components:Nn #1#2
5200 {
5201   \__spath_spot_weld_components:n {#2}
5202   \tl_set_eq:NN #1 \g__spath_output_tl
5203   \tl_gclear:N \g__spath_output_tl
5204 }
5205 \cs_generate_variant:Nn \spath_spot_weld_components:Nn {NV, cV, cn}
5206 \cs_new_protected_nopar:Npn \spath_spot_weld_components:N #1
5207 {
5208   \spath_spot_weld_components:NV #1#1
5209 }
5210 \cs_generate_variant:Nn \spath_spot_weld_components:N {c}
5211 \cs_new_protected_nopar:Npn \spath_spot_gweld_components:Nn #1#2
5212 {
5213   \__spath_spot_weld_components:n {#2}
5214   \tl_gset_eq:NN #1 \g__spath_output_tl
5215   \tl_gclear:N \g__spath_output_tl
5216 }
5217 \cs_generate_variant:Nn \spath_spot_gweld_components:Nn {NV, cV, cn}
5218 \cs_new_protected_nopar:Npn \spath_spot_gweld_components:N #1
5219 {
5220   \spath_spot_gweld_components:NV #1#1
5221 }
5222 \cs_generate_variant:Nn \spath_spot_gweld_components:N {c}

```

(End definition for `\spath_spot_weld_components:Nn` and others.)

3.8 Exporting Commands

`\spath_convert_to_svg:Nn` Convert the soft path to an SVG document.

```

\spath_gconvert_to_svg:Nn 5223 \cs_new_protected_nopar:Npn \__spath_convert_to_svg:n #1
5224 {
5225   \group_begin:
5226   \tl_clear:N \l__spath_tmpa_tl
5227   \tl_put_right:Nn \l__spath_tmpa_tl
5228   {
5229     <?xml~ version="1.0"~ standalone="no"?>
5230     \iow_newline:
5231     <!DOCTYPE~ svg~ PUBLIC~ "-//W3C//DTD SVG 1.1//EN"~
5232     "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
5233     \iow_newline:
5234     <svg~ xmlns="http://www.w3.org/2000/svg"~ version="1.1"~viewBox="
5235   }
5236
5237   \spath_minbb:Nn \l__spath_tmpb_tl {#1}
5238   \spath_maxbb:Nn \l__spath_tmpc_tl {#1}

```

```

5239 \tl_put_right:Nx \l__spath_tmpa_tl
5240 {
5241   \dim_to_decimal:n
5242   {
5243     \tl_item:Nn \l__spath_tmpb_tl {1} - 10pt
5244   }
5245   \exp_not:n {~}
5246   \dim_to_decimal:n
5247   {
5248     \tl_item:Nn \l__spath_tmpb_tl {2} - 10pt
5249   }
5250   \exp_not:n {~}
5251   \dim_to_decimal:n
5252   {
5253     \tl_item:Nn \l__spath_tmpc_tl {1}
5254     -
5255     \tl_item:Nn \l__spath_tmpb_tl {1}
5256     + 20pt
5257   }
5258   \exp_not:n {~}
5259   \dim_to_decimal:n
5260   {
5261     \tl_item:Nn \l__spath_tmpc_tl {2}
5262     -
5263     \tl_item:Nn \l__spath_tmpb_tl {2}
5264     + 20pt
5265   }
5266 }
5267
5268 \tl_put_right:Nn \l__spath_tmpa_tl
5269 {
5270   ">
5271   \iow_newline:
5272   <path~ d="
5273 }
5274 \tl_set:Nn \l__spath_tmpc_tl {use:n}
5275 \tl_map_inline:nn {#1}
5276 {
5277   \tl_set:Nn \l__spath_tmpb_tl {##1}
5278   \tl_case:NnF \l__spath_tmpb_tl
5279   {
5280     \c_spath_moveto_tl
5281     {
5282       \tl_put_right:Nn \l__spath_tmpa_tl {M~}
5283       \tl_set:Nn \l__spath_tmpc_tl {use:n}
5284     }
5285     \c_spath_lineto_tl
5286     {
5287       \tl_put_right:Nn \l__spath_tmpa_tl {L~}
5288       \tl_set:Nn \l__spath_tmpc_tl {use:n}
5289     }
5290     \c_spath_closepath_tl
5291     {
5292       \tl_put_right:Nn \l__spath_tmpa_tl {Z~}

```

```

5293     \tl_set:Nn \l__spath_tmpc_tl {use:none}
5294   }
5295   \c_spath_curvetoa_tl
5296   {
5297     \tl_put_right:Nn \l__spath_tmpa_tl {C~}
5298     \tl_set:Nn \l__spath_tmpc_tl {use:n}
5299   }
5300   \c_spath_curvetob_tl {
5301     \tl_set:Nn \l__spath_tmpc_tl {use:n}
5302   }
5303   \c_spath_curveto_tl {
5304     \tl_set:Nn \l__spath_tmpc_tl {use:n}
5305   }
5306   }
5307   {
5308     \tl_put_right:Nx
5309     \l__spath_tmpa_tl
5310     {\use:c { \l__spath_tmpc_tl } {\dim_to_decimal:n {##1}} ~}
5311   }
5312 }
5313 \tl_put_right:Nn \l__spath_tmpa_tl
5314 {
5315   "~ fill="none"~ stroke="black"~ />
5316   \iow_newline:
5317   </svg>
5318   \iow_newline:
5319 }
5320 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
5321 \group_end:
5322 }
5323 \cs_new_protected_nopar:Npn \spath_convert_to_svg:Nn #1#2
5324 {
5325   \__spath_convert_to_svg:n {#2}
5326   \tl_set_eq:NN #1 \g__spath_output_tl
5327   \tl_gclear:N \g__spath_output_tl
5328 }
5329 \cs_new_protected_nopar:Npn \spath_gconvert_to_svg:Nn #1#2
5330 {
5331   \__spath_convert_to_svg:n {#2}
5332   \tl_gset_eq:NN #1 \g__spath_output_tl
5333   \tl_gclear:N \g__spath_output_tl
5334 }

```

(End definition for \spath_convert_to_svg:Nn and \spath_gconvert_to_svg:Nn.)

`\spath_export_to_svg:nn` Save a soft path to an SVG file.

```

5335 \iow_new:N \g__spath_stream
5336 \cs_new_protected_nopar:Npn \spath_export_to_svg:nn #1#2
5337 {
5338   \group_begin:
5339   \spath_convert_to_svg:Nn \l__spath_tmpa_tl {#2}
5340   \iow_open:Nn \g__spath_stream {#1 .svg}
5341   \iow_now:Nx \g__spath_stream
5342   {

```

```

5343     \tl_use:N \l__spath_tmpa_tl
5344   }
5345   \iow_close:N \g__spath_stream
5346   \group_end:
5347 }
5348 \cs_generate_variant:Nn \spath_export_to_svg:nn {nv, nV}

```

(End definition for \spath_export_to_svg:nn.)

`\spath_show:n` Displays the soft path on the terminal.

```

5349 \cs_new_protected_nopar:Npn \spath_show:n #1
5350 {
5351   \int_step_inline:nnnn {1} {3} {\tl_count:n {#1}}
5352   {
5353     \iow_term:x {
5354       \tl_item:nn {#1} {##1}
5355       {\tl_item:nn {#1} {##1+1}}
5356       {\tl_item:nn {#1} {##1+2}}
5357     }
5358   }
5359 }
5360 \cs_generate_variant:Nn \spath_show:n {V, v}

```

(End definition for \spath_show:n.)

3.9 PGF and TikZ Interface Functions

Spaths come from PGF so we need some functions that get and set spaths from the pgf system.

`\spath_get_current_path:N` Grab the current soft path from PGF.

```

\spath_gget_current_path:N
5361 \cs_new_protected_nopar:Npn \spath_get_current_path:N #1
5362 {
5363   \pgfsyssoftpath@getcurrentpath #1
5364 }
5365 \cs_generate_variant:Nn \spath_get_current_path:N {c}
5366 \cs_new_protected_nopar:Npn \spath_gget_current_path:N #1
5367 {
5368   \pgfsyssoftpath@getcurrentpath #1
5369   \tl_gset_eq:NN #1 #1
5370 }
5371 \cs_generate_variant:Nn \spath_gget_current_path:N {c}

```

(End definition for \spath_get_current_path:N and \spath_gget_current_path:N.)

`\spath_protocol_path:n` This feeds the bounding box of the soft path to PGF to ensure that its current bounding box contains the soft path.

```

5372 \cs_new_protected_nopar:Npn \spath_protocol_path:n #1
5373 {
5374   \spath_minbb:Nn \l__spath_tmpa_tl {#1}
5375   \exp_last_unbraced:NV \pgf@protocolsizes\l__spath_tmpa_tl
5376
5377   \spath_maxbb:Nn \l__spath_tmpa_tl {#1}
5378   \exp_last_unbraced:NV \pgf@protocolsizes\l__spath_tmpa_tl

```

```

5379 }
5380 \cs_generate_variant:Nn \spath_protocol_path:n {V}

```

(End definition for \spath_protocol_path:n.)

`\spath_set_current_path:n` Sets the current path to the specified soft path.

```

\spath_set_current_path:N
5381 \cs_new_protected_nopar:Npn \spath_set_current_path:n #1
5382 {
5383   \spath_protocol_path:n {#1}
5384   \tl_set:Nn \l__spath_tmpa_tl {#1}
5385   \pgfsyssoftpath@setcurrentpath\l__spath_tmpa_tl
5386 }
5387 \cs_new_protected_nopar:Npn \spath_set_current_path:N #1
5388 {
5389   \spath_protocol_path:V #1
5390   \pgfsyssoftpath@setcurrentpath #1
5391 }
5392 \cs_generate_variant:Nn \spath_set_current_path:N {c}

```

(End definition for \spath_set_current_path:n and \spath_set_current_path:N.)

`\spath_use_path:nn` Uses the given soft path at the PGF level.

```

5393 \cs_new_protected_nopar:Npn \spath_use_path:nn #1#2
5394 {
5395   \spath_set_current_path:n {#1}
5396   \pgfusepath{#2}
5397 }

```

(End definition for \spath_use_path:nn.)

`\spath_tikz_path:nn` Uses the given soft path at the TikZ level.

```

5398 \cs_new_protected_nopar:Npn \spath_tikz_path:nn #1#2
5399 {
5400   \tl_if_empty:nF {#2}
5401   {
5402     \path[#1] \pgfextra{
5403       \spath_set_current_path:n {#2}
5404       \tl_put_left:Nn \tikz@preactions {\def\tikz@actions@path{#2}}
5405     };
5406   }
5407 }
5408 \cs_generate_variant:Nn \spath_tikz_path:nn {Vn, VV, nv, Vv, nV}

```

(End definition for \spath_tikz_path:nn.)

`\spath_set_tikz_data:n` Sets the `\tikz@lastx` and other coordinates from the soft path.

```

5409 \cs_new_protected_nopar:Npn \spath_set_tikz_data:n #1
5410 {
5411   \spath_finalpoint:Nn \l__spath_tmpa_tl {#1}
5412   \tl_set:Nx \l__spath_tmpa_tl
5413   {
5414     \exp_not:c {pgf@x}=\tl_item:Nn \l__spath_tmpa_tl {1} \relax
5415     \exp_not:c {pgf@y}=\tl_item:Nn \l__spath_tmpa_tl {2} \relax
5416   }
5417   \use:c {pgf@process}{%

```

```

5418 \tl_use:N \l__spath_tmpa_tl
5419 \pgftransforminvert
5420 \use:c {pgf@pos@transform@glob}
5421 }
5422 \tl_set:Nx \l__spath_tmpa_tl
5423 {
5424 \exp_not:c {tikz@lastx}=\exp_not:c {pgf@x} \relax
5425 \exp_not:c {tikz@lasty}=\exp_not:c {pgf@y} \relax
5426 \exp_not:c {tikz@lastxsaved}=\exp_not:c {pgf@x} \relax
5427 \exp_not:c {tikz@lastysaved}=\exp_not:c {pgf@y} \relax
5428 }
5429 \tl_use:N \l__spath_tmpa_tl
5430 \spath_finalmovepoint:Nn \l__spath_tmpa_tl {#1}
5431 \bool_if:NT \l_spath_movetorelevant_bool
5432 {
5433 \ifpgfsyssoftpathmovetorelevant%
5434 \tl_gset_eq:cN {pgfsyssoftpath@lastmoveto} \l__spath_tmpa_tl
5435 \fi
5436 }
5437 \tl_set:Nx \l__spath_tmpa_tl
5438 {
5439 \exp_not:c {pgf@x}=\tl_item:Nn \l__spath_tmpa_tl {1} \relax
5440 \exp_not:c {pgf@y}=\tl_item:Nn \l__spath_tmpa_tl {2} \relax
5441 }
5442 \use:c {pgf@process}{%
5443 \tl_use:N \l__spath_tmpa_tl
5444 \pgftransforminvert
5445 \use:c {pgf@pos@transform@glob}
5446 }
5447 \bool_if:NT \l_spath_movetorelevant_bool
5448 {
5449 \dim_if_exist:cT {tikz@lastmovetox}
5450 {
5451 \tl_set:Nx \l__spath_tmpa_tl
5452 {
5453 \exp_not:c {tikz@lastmovetox}=\exp_not:c {pgf@x} \relax
5454 \exp_not:c {tikz@lastmovetoy}=\exp_not:c {pgf@y} \relax
5455 }
5456 \tl_use:N \l__spath_tmpa_tl
5457 }
5458 }
5459 \tl_clear_new:c {tikz@timer}
5460 \tl_set:cn {tikz@timer}
5461 {
5462 \pgftransformreset
5463 \spath_reallength:Nn \l__spath_tmpa_int {#1}
5464 \tl_set_eq:Nc \l__spath_tmpb_tl {tikz@time}
5465 \tl_set:Nx \l__spath_tmpb_tl
5466 {\fp_to_decimal:n {(\l__spath_tmpb_tl) * (\l__spath_tmpa_int)}}
5467 \spath_transformation_at:NnV \l__spath_tmpc_tl {#1} \l__spath_tmpb_tl
5468
5469 \tl_set:Nx \l__spath_tmpa_tl
5470 {
5471 \exp_not:N \pgfpoint

```

```

5472     { \tl_item:Nn \l__spath_tmpc_tl {5} }
5473     { \tl_item:Nn \l__spath_tmpc_tl {6} }
5474   }
5475   \exp_args:NV \pgftransformshift \l__spath_tmpa_tl
5476
5477   \ifpgfresetnontranslationatime
5478   \pgftransformresetnontranslations
5479   \fi
5480
5481   \ifpgfslopedatime
5482
5483   \tl_set:Nx \l__spath_tmpa_tl
5484   {
5485     { \tl_item:Nn \l__spath_tmpc_tl {1} }
5486     { \tl_item:Nn \l__spath_tmpc_tl {2} }
5487     { \tl_item:Nn \l__spath_tmpc_tl {3} }
5488     { \tl_item:Nn \l__spath_tmpc_tl {4} }
5489   }
5490   \ifpgfallowupsidedownatime
5491   \else
5492   \fp_compare:nT { \tl_item:Nn \l__spath_tmpc_tl {4} < 0 }
5493   {
5494     \tl_set:Nx \l__spath_tmpa_tl
5495     {
5496       { \fp_eval:n { - (\tl_item:Nn \l__spath_tmpc_tl {1}) } }
5497       { \fp_eval:n { - (\tl_item:Nn \l__spath_tmpc_tl {2}) } }
5498       { \fp_eval:n { - (\tl_item:Nn \l__spath_tmpc_tl {3}) } }
5499       { \fp_eval:n { - (\tl_item:Nn \l__spath_tmpc_tl {4}) } }
5500     }
5501   }
5502   \fi
5503   \tl_put_right:Nn \l__spath_tmpa_tl { {\pgfpointorigin} }
5504   \exp_last_unbraced:NV \pgftransformcm \l__spath_tmpa_tl
5505   \fi
5506   }
5507 }
5508 \cs_generate_variant:Nn \spath_set_tikz_data:n {V, v}

```

(End definition for \spath_set_tikz_data:n.)

4 The TikZ interface

```
5509 <@@=tikzspath>
```

This provides an interface to the soft path manipulation routines via a series of TikZ keys. They all live in the `spath` family.

```

5510 \RequirePackage{spath3}
5511 \RequirePackage{expl3}
5512 \ExplSyntaxOn
5513
5514 \tl_new:N \l__tikzspath_tmpa_tl
5515 \tl_new:N \l__tikzspath_tmpb_tl
5516 \tl_new:N \l__tikzspath_tmpc_tl
5517 \tl_new:N \l__tikzspath_tmpd_tl

```

```

5518 \tl_new:N \l__tikzspath_tmpe_tl
5519 \tl_new:N \l__tikzspath_tmpf_tl
5520
5521 \int_new:N \l__tikzspath_tmpe_int
5522 \seq_new:N \l__tikzspath_tmpe_seq
5523 \seq_new:N \l__tikzspath_tmpeb_seq
5524 \seq_new:N \l__tikzspath_tmpec_seq
5525 \seq_new:N \l__tikzspath_tmpe_d_seq
5526
5527 \tl_new:N \l__tikzspath_current_tl
5528 \tl_new:N \l__tikzspath_reverse_tl
5529 \tl_new:N \l__tikzspath_prefix_tl
5530 \tl_new:N \l__tikzspath_suffix_tl
5531 \tl_new:N \g__tikzspath_smuggle_tl
5532 \tl_new:N \g__tikzspath_output_tl
5533 \tl_new:N \l__tikzspath_check_tl
5534 \clist_new:N \g__tikzspath_output_clist
5535 \seq_new:N \g__tikzspath_tmpe_seq
5536 \seq_new:N \g__tikzspath_tmpeb_seq
5537 \seq_new:N \g__tikzspath_output_seq
5538 \bool_new:N \l__tikzspath_draft_bool

```

We surround all the keys with checks to ensure that the soft path under consideration does actually exist, but if it doesn't we should warn the user.

```

5539 \msg_new:nnn { spath3 } { missing soft path }
5540 { Soft~ path~ #1~ doesn't~ exist~ \msg_line_context:}
5541 \msg_new:nnnn { spath3 } { empty soft path }
5542 { Soft~ path~ #1~ is~ empty~ \msg_line_context:}
5543 {If~ it~ was~ defined~ inside~ a~ group,~ try~ using~ "save~ global". }
5544 \msg_new:nnn { spath3 } { load intersections }
5545 { You~ need~ to~ load~ the~ "intersections"~ library~
5546   to~ work~ with~ intersections }

```

When saving a soft path, by default we use a naming convention that is compatible with the intersections library so that paths saved here and paths saved by the `name path` facility of the intersections library are mutually exchangeable.

```

5547 \tl_set:Nn \l__tikzspath_prefix_tl {tikz@intersect@path@name@}
5548 \tl_set:Nn \l__tikzspath_suffix_tl {}

```

When a soft path is grabbed from TikZ we're usually deep in a group so I've adapted the code from the intersections library to dig the definition out of the group without making everything global.

Interestingly, the intersections library doesn't clear its naming code once it is used meaning that it keeps resetting the definition of a path back to its original one every time a path command is called.

Also, when the hook is restored outside a scope then no check is made to ensure that the inner one was actually invoked. This can cause issues when the syntax `\tikz ...` is used since the end of the path coincides with the end of the picture.

```

5549 \tl_new:N \g__tikzspath_tikzfinish_tl
5550 \tl_new:N \l__tikzspath_tikzfinish_outside_tl
5551 \cs_new_protected_nopar:Npn \spath_at_end_of_path:
5552 {
5553   \tl_use:N \g__tikzspath_tikzfinish_tl
5554   \tl_gclear:N \g__tikzspath_tikzfinish_tl

```



```

5555 }
5556 \tl_put_right:Nn \tikz@finish {\spath_at_end_of_path:}
5557
5558 \tikzset{
5559   every~ scope/.append~ style={
5560     execute~ at~ begin~ scope={
5561       \tl_set_eq:NN \l__tikzspath_tikzfinish_outside_tl \g__tikzspath_tikzfinish_tl
5562     },
5563     execute~ at~ end~ scope={
5564       \tl_use:N \g__tikzspath_tikzfinish_tl
5565       \tl_gclear:N \g__tikzspath_tikzfinish_tl
5566       \tl_gset_eq:NN \g__tikzspath_tikzfinish_tl \l__tikzspath_tikzfinish_outside_tl
5567     },
5568   },
5569 }

```

This is for delaying something until the path is fully constructed (but no later), sometimes useful to be able to specify this in the path options rather than directly at the end of the path.

```

5570 \tl_new:N \l__tikzspath_tikzpath_finish_tl
5571
5572 \cs_new_protected_nopar:Npn \__tikzspath_at_end_of_path_construction:
5573 {
5574   \tl_use:N \l__tikzspath_tikzpath_finish_tl
5575   \tl_clear:N \l__tikzspath_tikzpath_finish_tl
5576 }
5577
5578 \tl_put_left:Nn \tikz@finish {\__tikzspath_at_end_of_path_construction:}

```

Code for saving a path

```

5579 \cs_new_protected_nopar:Npn \spath_save_path:Nn #1#2
5580 {
5581   \tl_if_empty:NF \g__tikzspath_tikzfinish_tl
5582   {
5583     \tl_use:N \g__tikzspath_tikzfinish_tl
5584   }
5585   \tl_gput_right:Nn \g__tikzspath_tikzfinish_tl
5586   {
5587     \tl_clear_new:N #1
5588     \tl_set:Nn #1 {#2}
5589   }
5590 }
5591 \cs_generate_variant:Nn \spath_save_path:Nn {cn, NV, cV}
5592
5593 \cs_new_protected_nopar:Npn \spath_gsave_path:Nn #1#2
5594 {
5595   \tl_gput_right:Nn \g__tikzspath_tikzfinish_tl
5596   {
5597     \tl_gclear_new:N #1
5598     \tl_gset:Nn #1 {#2}
5599   }
5600 }
5601 \cs_generate_variant:Nn \spath_gsave_path:Nn {cn, NV, cV}

```

`__tikzspath_process_tikz_point:Nn` Process a point via TikZ and store the resulting dimensions.

```

5602 \cs_new_protected_nopar:Npn \__tikzspath_process_tikz_point:Nn #1#2
5603 {
5604   \group_begin:
5605   \use:c {tikz@scan@one@point} \use:n #2 \scan_stop:
5606   \tl_gset:Nx \g__tikzspath_output_tl
5607   {
5608     { \dim_use:c {pgf@x} }
5609     { \dim_use:c {pgf@y} }
5610   }
5611   \group_end:
5612   \tl_set_eq:NN #1 \g__tikzspath_output_tl
5613   \tl_gclear:N \g__tikzspath_output_tl
5614 }

```

(End definition for __tikzspath_process_tikz_point:Nn.)

__tikzspath_tikzset:n Wrapper around \tikzset for expansion.

```

5615 \cs_set_eq:NN \__tikzspath_tikzset:n \tikzset
5616 \cs_generate_variant:Nn \__tikzspath_tikzset:n {V, v}

```

(End definition for __tikzspath_tikzset:n.)

s:nnnn___tikzspath_check_three_paths:nnnn

Given a path name as the second argument, check if it exists and is not empty, and if so reinsert it after the first argument. The third argument is code to be executed in case of a missing or empty path.

```

5617 \cs_new_protected_nopar:Npn \__tikzspath_check_path:nnn #1#2#3
5618 {
5619   \tl_set:Nn \l__tikzspath_check_tl {#3}
5620   \tl_if_exist:cTF {\__tikzspath_path_name:n {#2}}
5621   {
5622     \tl_if_empty:cTF {\__tikzspath_path_name:n {#2}}
5623     {
5624       \msg_warning:nnn { spath3 } { empty soft path } { #2 }
5625     }
5626     {
5627       \tl_set:Nn \l__tikzspath_check_tl {
5628         #1 {\__tikzspath_path_name:n {#2}}
5629       }
5630     }
5631   }
5632   {
5633     \msg_warning:nnn { spath3 } { missing soft path } { #2 }
5634   }
5635   \tl_use:N \l__tikzspath_check_tl
5636 }
5637 \cs_new_protected_nopar:Npn \__tikzspath_check_two_paths:nnnn #1#2#3#4
5638 {
5639   \__tikzspath_check_path:nnn {
5640     \__tikzspath_check_path:nnn {#1}{#2}{#4}
5641   }{#3}{#4}
5642 }
5643 \cs_new_protected_nopar:Npn \__tikzspath_check_three_paths:nnnnn #1#2#3#4#5
5644 {
5645   \__tikzspath_check_path:nnn {

```

```

5646     \tikzspath_check_path:nnn {
5647         \tikzspath_check_path:nnn {#1}{#2}{#5}
5648     }{#3}{#5}
5649 }{#4}{#5}
5650 }
5651 \cs_generate_variant:Nn \tikzspath_check_path:nnn {nVn}
5652 \cs_generate_variant:Nn \tikzspath_check_two_paths:nnnn {nnVn}

```

(End definition for \tikzspath_check_path:nnn \tikzspath_check_two_paths:nnnn \tikzspath_check_three_paths:nnnn.)

h_maybe_current_two_paths_reuse_second:nnnn

If the named path is “current” then get the current path and use that. The second version puts the resulting path back as the current path.

```

5653 \cs_new_protected_nopar:Npn \tikzspath_maybe_current_path:nn #1#2
5654 {
5655     \tl_if_eq:nnT {#2} {current}
5656     {
5657         \spath_get_current_path:c {\tikzspath_path_name:n {#2}}
5658     }
5659     #1 {#2}
5660 }
5661 \cs_new_protected_nopar:Npn \tikzspath_maybe_current_path_reuse:nnn #1#2#3
5662 {
5663     \bool_set_true:N \l_spath_movetorelevant_bool
5664     \tl_if_eq:nnT {#2} {current}
5665     {
5666         \spath_get_current_path:c {\tikzspath_path_name:n {#2}}
5667     }
5668     #1 {#2} #3
5669     \tl_if_eq:nnT {#2} {current}
5670     {
5671         \tl_if_empty:cF {\tikzspath_path_name:n {#2}}
5672         {
5673             \spath_set_current_path:c {\tikzspath_path_name:n {#2}}
5674             \spath_set_tikz_data:v {\tikzspath_path_name:n {#2}}
5675         }
5676     }
5677 }
5678 \cs_new_protected_nopar:Npn \tikzspath_maybe_current_two_paths_reuse_both:nnnn #1#2#3#4
5679 {
5680     \bool_set_true:N \l_spath_movetorelevant_bool
5681     \tl_if_eq:nnT {#2} {current}
5682     {
5683         \spath_get_current_path:c {\tikzspath_path_name:n {#2}}
5684     }
5685     \tl_if_eq:nnT {#3} {current}
5686     {
5687         \spath_get_current_path:c {\tikzspath_path_name:n {#3}}
5688     }
5689     #1 {#2} {#3} #4
5690     \tl_if_eq:nnT {#2} {current}
5691     {
5692         \tl_if_empty:cF {\tikzspath_path_name:n {#2}}
5693         {

```

```

5694     \spath_set_current_path:c {\__tikzspath_path_name:n {#2}}
5695     \spath_set_tikz_data:v {\__tikzspath_path_name:n {#2}}
5696   }
5697 }
5698 \tl_if_eq:nnT {#3} {current}
5699 {
5700   \tl_if_empty:cF {\__tikzspath_path_name:n {#3}}
5701   {
5702     \spath_set_current_path:c {\__tikzspath_path_name:n {#3}}
5703     \spath_set_tikz_data:v {\__tikzspath_path_name:n {#3}}
5704   }
5705 }
5706 }
5707 \cs_new_protected_nopar:Npn \__tikzspath_maybe_current_two_paths_reuse_first:nnnn #1#2#3#4
5708 {
5709   \bool_set_true:N \l_spath_movetorelevant_bool
5710   \tl_if_eq:nnT {#2} {current}
5711   {
5712     \spath_get_current_path:c {\__tikzspath_path_name:n {#2}}
5713   }
5714   \tl_if_eq:nnT {#3} {current}
5715   {
5716     \spath_get_current_path:c {\__tikzspath_path_name:n {#3}}
5717   }
5718   #1 {#2} {#3} #4
5719   \tl_if_eq:nnT {#2} {current}
5720   {
5721     \tl_if_empty:cF {\__tikzspath_path_name:n {#2}}
5722     {
5723       \spath_set_current_path:c {\__tikzspath_path_name:n {#2}}
5724       \spath_set_tikz_data:v {\__tikzspath_path_name:n {#2}}
5725     }
5726   }
5727 }
5728 \cs_new_protected_nopar:Npn \__tikzspath_maybe_current_two_paths_reuse_second:nnnn #1#2#3#4
5729 {
5730   \bool_set_true:N \l_spath_movetorelevant_bool
5731   \tl_if_eq:nnT {#2} {current}
5732   {
5733     \spath_get_current_path:c {\__tikzspath_path_name:n {#2}}
5734   }
5735   \tl_if_eq:nnT {#3} {current}
5736   {
5737     \spath_get_current_path:c {\__tikzspath_path_name:n {#3}}
5738   }
5739   #1 {#2} {#3} #4
5740   \tl_if_eq:nnT {#3} {current}
5741   {
5742     \tl_if_empty:cF {\__tikzspath_path_name:n {#3}}
5743     {
5744       \spath_set_current_path:c {\__tikzspath_path_name:n {#3}}
5745       \spath_set_tikz_data:v {\__tikzspath_path_name:n {#3}}
5746     }
5747   }

```

```

5748 }
5749 \cs_generate_variant:Nn \__tikzspath_maybe_current_path:nn {nV}
5750 \cs_generate_variant:Nn \__tikzspath_maybe_current_path_reuse:nnn {nVn}

(End definition for \__tikzspath_maybe_current_path:nn \__tikzspath_maybe_current_path_reuse:nnn
\__tikzspath_maybe_current_two_paths_reuse_both:nnnn \__tikzspath_maybe_current_two_paths_reuse_
first:nnnn \__tikzspath_maybe_current_two_paths_reuse_second:nnnn.)

```

`__tikzspath_seq_from_foreach:NNn` Convert a PGF foreach list, as the third argument, to a sequence. The second argument is the maximum number on the list.

```

5751 \cs_new_protected_nopar:Npn \__tikzspath_seq_from_foreach:NNn #1#2#3
5752 {
5753   \group_begin:
5754   \seq_gclear:N \g__tikzspath_output_seq
5755
5756   \tl_if_empty:nTF {#3}
5757   {
5758     \int_step_inline:nnnn {1}{1} {#2}
5759     {
5760       \seq_gput_right:Nn \g__tikzspath_output_seq {##1}
5761     }
5762   }
5763   {
5764     \foreach \l__tikzspath_tmpa_tl in {#3}
5765     {
5766       \int_compare:nTF { \l__tikzspath_tmpa_tl > 0 }
5767       {
5768         \seq_gput_right:NV \g__tikzspath_output_seq \l__tikzspath_tmpa_tl
5769       }
5770       {
5771         \seq_gput_right:Nx \g__tikzspath_output_seq
5772         {\int_eval:n {#2 - \l__tikzspath_tmpa_tl}}
5773       }
5774     }
5775     \seq_gsort:Nn \g__tikzspath_output_seq
5776     {
5777       \int_compare:nNnTF {##1} < {##2}
5778       { \sort_return_same: }
5779       { \sort_return_swapped: }
5780     }
5781   }
5782   \group_end:
5783   \seq_set_eq:NN #1 \g__tikzspath_output_seq
5784   \seq_gclear:N \g__tikzspath_output_seq
5785 }
5786 \cs_generate_variant:Nn \__tikzspath_seq_from_foreach:NNn {NVV, NVn}
5787 %

```

(End definition for `__tikzspath_seq_from_foreach:NNn`.)

`__tikzspath_path_name:n` Wrap the argument in the prefix and suffix to generate the proper name.

```

5788 \cs_new:Npn \__tikzspath_path_name:n #1
5789 {
5790   \tl_use:N \l__tikzspath_prefix_tl
5791   #1

```

```

5792 \tl_use:N \l__tikzspath_suffix_tl
5793 }
5794 \cs_generate_variant:Nn \__tikzspath_path_name:n {V}

```

(End definition for `__tikzspath_path_name:n`.)

When joining two paths we provide a set of options for how to process the second path.

```

5795 \bool_new:N \l__tikzspath_reverse_bool
5796 \bool_new:N \l__tikzspath_weld_bool
5797 \bool_new:N \l__tikzspath_move_bool
5798 \bool_new:N \l__tikzspath_global_bool
5799 \bool_new:N \l__tikzspath_current_transformation_bool
5800 \tl_new:N \l__tikzspath_joinpath_tl
5801 \tl_new:N \l__tikzspath_transformation_tl
5802
5803 \cs_new_protected_nopar:Npn \__tikzspath_set_bool:Nn #1#2
5804 {
5805   \tl_if_eq:nnTF {#2}{false}
5806   {
5807     \bool_set_false:N #1
5808   }
5809   {
5810     \bool_set_true:N #1
5811   }
5812 }
5813 \tikzset {
5814   spath/join/.is~ family,
5815   spath/join/.cd,
5816   reverse/.code = {
5817     \__tikzspath_set_bool:Nn \l__tikzspath_reverse_bool {#1}
5818   },
5819   reverse/.default = true,
5820   weld/.code = {
5821     \__tikzspath_set_bool:Nn \l__tikzspath_weld_bool {#1}
5822   },
5823   weld/.default = true,
5824   no~ weld/.code = {
5825     \__tikzspath_set_bool:Nn \l__tikzspath_weld_bool {#1}
5826     \bool_set:Nn \l__tikzspath_weld_bool {!\l__tikzspath_weld_bool}
5827   },
5828   no~ weld/.default = true,
5829   move/.code = {
5830     \__tikzspath_set_bool:Nn \l__tikzspath_move_bool {#1}
5831   },
5832   move/.default = true,
5833   no~ move/.code = {
5834     \__tikzspath_set_bool:Nn \l__tikzspath_move_bool {#1}
5835     \bool_set:Nn \l__tikzspath_move_bool {!\l__tikzspath_move_bool}
5836   },
5837   no~ move/.default = true,
5838   global/.code = {
5839     \__tikzspath_set_bool:Nn \l__tikzspath_global_bool {#1}
5840   },
5841   global/.default = true,

```

```

5842 use~ current~ transformation/.code={
5843   \__tikzspath_set_bool:Nn \l__tikzspath_current_transformation_bool {#1}
5844 },
5845 use~ current~ transformation/.default = true,
5846 transform/.store-in=\l__tikzspath_transformation_tl,
5847 .unknown/.code = {
5848   \tl_set_eq:NN \l__tikzspath_joinpath_tl \pgfkeyscurrentname
5849 }
5850 }

```

When we split a soft path into components, we make it a comma separated list so that it can be fed into a `\foreach` loop. This can also make it possible to extract a single component, but to do this we need a wrapper around `\clist_item:Nn` (there doesn't appear to be a PGF way of getting an item of a CS list).

```

5851 \cs_set_eq:NN \getComponentOf \clist_item:Nn

```

4.1 Helper Functions

`__tikzspath_use_path:n` Use a path, possibly manipulating it first.

```

5852 \cs_new_protected_nopar:Npn \__tikzspath_use_path:n #1
5853 {
5854   \tl_set:Nn \l__tikzspath_joinpath_tl {#1}
5855   \spath_get_current_path:N \l__tikzspath_current_tl
5856
5857   \bool_if:NT \l__tikzspath_reverse_bool
5858   {
5859     \spath_reverse:N \l__tikzspath_joinpath_tl
5860   }
5861
5862   \bool_if:NT \l__tikzspath_current_transformation_bool
5863   {
5864     \pgfgettransform \l__tikzspath_tmpa_tl
5865     \spath_transform:NV
5866     \l__tikzspath_joinpath_tl
5867     \l__tikzspath_tmpa_tl
5868   }
5869
5870   \tl_if_empty:NF \l__tikzspath_transformation_tl
5871   {
5872     \group_begin:
5873     \pgftransformreset
5874     \__tikzspath_tikzset:V \l__tikzspath_transformation_tl
5875     \pgfgettransform \l__tikzspath_tmpa_tl
5876     \tl_gset:Nn \g__tikzspath_smuggle_tl
5877     {
5878       \spath_transform:Nnnnnnn
5879       \l__tikzspath_joinpath_tl
5880     }
5881     \tl_gput_right:NV \g__tikzspath_smuggle_tl \l__tikzspath_tmpa_tl
5882     \group_end:
5883     \tl_use:N \g__tikzspath_smuggle_tl
5884   }
5885

```

```

5886 \bool_if:NT \l__tikzspath_move_bool
5887 {
5888   \tl_if_empty:NTF \l__tikzspath_current_tl
5889   {
5890     \tl_set:Nn \l__tikzspath_tmpc_tl { {Opt} {Opt} }
5891   }
5892   {
5893     \spath_finalpoint:NV
5894     \l__tikzspath_tmpc_tl
5895     \l__tikzspath_current_tl
5896   }
5897   \spath_translate_to:NV \l__tikzspath_joinpath_tl \l__tikzspath_tmpc_tl
5898 }
5899
5900 \tl_if_empty:NTF \l__tikzspath_current_tl
5901 {
5902   \tl_if_empty:NTF \l__tikzspath_joinpath_tl
5903   {
5904     \tl_set_eq:NN \l__tikzspath_current_tl \c_spath_moveto_tl
5905     \tl_put_right:Nn \l__tikzspath_current_tl {{Opt}{Opt}}
5906   }
5907   {
5908     \tl_set_eq:NN \l__tikzspath_current_tl \l__tikzspath_joinpath_tl
5909   }
5910 }
5911 {
5912
5913   \tl_clear:N \l__tikzspath_tmpa_tl
5914   \tl_set:Nn \l__tikzspath_tmpa_tl {spath_}
5915
5916   \tl_put_right:Nn \l__tikzspath_tmpa_tl {append}
5917
5918   \bool_if:NT \l__tikzspath_weld_bool
5919   {
5920     \tl_put_right:Nn \l__tikzspath_tmpa_tl {_no_move}
5921     \spath_numberofcomponents:NV \l__tikzspath_tmpa_int \l__tikzspath_joinpath_tl
5922     \int_compare:nT {\l__tikzspath_tmpa_int == 1}
5923     {
5924       \bool_set_false:N \l_spath_movetorelevant_bool
5925     }
5926   }
5927   \tl_put_right:Nn \l__tikzspath_tmpa_tl {:NV}
5928
5929   \use:c {\tl_use:N \l__tikzspath_tmpa_tl }
5930   \l__tikzspath_current_tl
5931   \l__tikzspath_joinpath_tl
5932 }
5933
5934 \spath_set_current_path:N \l__tikzspath_current_tl
5935 \spath_set_tikz_data:V \l__tikzspath_joinpath_tl
5936 }
5937 \cs_generate_variant:Nn \_tikzspath_use_path:n {V, v}

```

(End definition for _tikzspath_use_path:n.)

__tikzspath_join_with:nn

```
5938 \cs_new_protected_nopar:Npn \__tikzspath_join_with:Nn #1#2
5939 {
5940   \tl_set:Nn \l__tikzspath_joinpath_tl {#2}
5941
5942   \bool_if:NT \l__tikzspath_reverse_bool
5943   {
5944     \spath_reverse:N \l__tikzspath_joinpath_tl
5945   }
5946
5947   \tl_if_empty:NF \l__tikzspath_transformation_tl
5948   {
5949     \group_begin:
5950     \pgftransformreset
5951     \__tikzspath_tikzset:V \l__tikzspath_transformation_tl
5952     \pgfgettransform \l__tikzspath_tmpa_tl
5953     \tl_gset:Nn \g__tikzspath_smuggle_tl
5954     {
5955       \spath_transform:Nnnnnnn
5956       \l__tikzspath_joinpath_tl
5957     }
5958     \tl_gput_right:NV \g__tikzspath_smuggle_tl \l__tikzspath_tmpa_tl
5959     \group_end:
5960     \tl_use:N \g__tikzspath_smuggle_tl
5961   }
5962
5963   \bool_if:NT \l__tikzspath_move_bool
5964   {
5965     \spath_finalpoint:NV
5966     \l__tikzspath_tmpc_tl
5967     #1
5968     \spath_translate_to:NV \l__tikzspath_joinpath_tl \l__tikzspath_tmpc_tl
5969   }
5970
5971   \tl_clear:N \l__tikzspath_tmpa_tl
5972   \tl_set:Nn \l__tikzspath_tmpa_tl {spath_}
5973
5974   \bool_if:NT \l__tikzspath_global_bool
5975   {
5976     \tl_put_right:Nn \l__tikzspath_tmpa_tl {g}
5977   }
5978
5979   \tl_put_right:Nn \l__tikzspath_tmpa_tl {append}
5980
5981   \bool_if:NT \l__tikzspath_weld_bool
5982   {
5983     \tl_put_right:Nn \l__tikzspath_tmpa_tl {_no_move}
5984   }
5985   \tl_put_right:Nn \l__tikzspath_tmpa_tl {:NV}
5986
5987   \cs_if_exist:cF {\tl_use:N \l__tikzspath_tmpa_tl}
5988   {
5989     \tl_show:N \l__tikzspath_tmpa_tl
5990   }
```

```

5991
5992 \use:c {\tl_use:N \l__tikzspath_tmpa_tl } #1
5993 \l__tikzspath_joinpath_tl
5994 }
5995 \cs_generate_variant:Nn \__tikzspath_join_with:Nn {cv, cn}

```

(End definition for __tikzspath_join_with:nn.)

\tikzspath_join_components_upright_with:Nnn

Join the specified components of the first path by splicing in the second.

```

5996 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with_aux:nnn #1#2#3
5997 {
5998 \group_begin:
5999 \tl_set:Nn \l__tikzspath_tmpc_tl {#1}
6000 \tl_if_empty:nT {#3}
6001 {
6002 \spath_spot_weld_components:N \l__tikzspath_tmpc_tl
6003 }
6004
6005 \spath_numberofcomponents:NV \l__tikzspath_tmpa_int \l__tikzspath_tmpc_tl
6006 \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpb_seq \l__tikzspath_tmpa_int {#3}
6007
6008 \spath_components_to_seq:NV \l__tikzspath_tmpa_seq \l__tikzspath_tmpc_tl
6009
6010 \seq_pop_left:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6011 \seq_pop_left:NN \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6012
6013 \seq_map_indexed_inline:Nn \l__tikzspath_tmpa_seq
6014 {
6015 \int_compare:nTF
6016 {
6017 ##1 == \l__tikzspath_tmpb_tl
6018 }
6019 {
6020 \seq_pop_left:NNF \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6021 {
6022 \tl_set:Nn \l__tikzspath_tmpb_tl {-1}
6023 }
6024 \spath_splice_between:Nnn \l__tikzspath_tmpa_tl {#2} {##2}
6025 }
6026 {
6027 \tl_put_right:Nn \l__tikzspath_tmpa_tl {##2}
6028 }
6029 }
6030 \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpa_tl
6031 \group_end:
6032 }
6033 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with:Nnnn #1#2#3#4
6034 {
6035 \__tikzspath_join_components_with_aux:nnn {#2}{#3}{#4}
6036 \tl_set_eq:NN #1 \g__tikzspath_output_tl
6037 \tl_gclear:N \g__tikzspath_output_tl
6038 }
6039 \cs_generate_variant:Nn \__tikzspath_join_components_with:Nnnn {NVnn}
6040 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with:Nnn #1#2#3

```

```

6041 {
6042   \tikzspath_join_components_with:NVnn #1#1{#2}{#3}
6043 }
6044 \cs_generate_variant:Nn \tikzspath_join_components_with:Nnn {cvV}
6045 \cs_new_protected_nopar:Npn \tikzspath_gjoin_components_with:Nnnn #1#2#3#4
6046 {
6047   \tikzspath_join_components_with_aux:nnn {#2}{#3}{#4}
6048   \tl_gset_eq:NN #1 \g_tikzspath_output_tl
6049   \tl_gclear:N \g_tikzspath_output_tl
6050 }
6051 \cs_generate_variant:Nn \tikzspath_gjoin_components_with:Nnnn {NVnn}
6052 \cs_new_protected_nopar:Npn \tikzspath_gjoin_components_with:Nnn #1#2#3
6053 {
6054   \tikzspath_gjoin_components_with:NVnn #1#1{#2}{#3}
6055 }
6056 \cs_generate_variant:Nn \tikzspath_gjoin_components_with:Nnn {cvV}
6057 \cs_new_protected_nopar:Npn \tikzspath_join_components_upright_with_aux:nnn #1#2#3
6058 {
6059   \group_begin:
6060   \tl_set:Nn \l_tikzspath_tmpe_tl {#1}
6061   \tl_if_empty:nT {#3}
6062   {
6063     \spath_spot_weld_components:N \l_tikzspath_tmpe_tl
6064   }
6065
6066   \spath_numberofcomponents:NV \l_tikzspath_tmpe_int \l_tikzspath_tmpe_tl
6067   \tikzspath_seq_from_foreach:NVn \l_tikzspath_tmpe_seq \l_tikzspath_tmpe_int {#3}
6068
6069   \spath_components_to_seq:NV \l_tikzspath_tmpe_seq \l_tikzspath_tmpe_tl
6070
6071   \seq_pop_left:NN \l_tikzspath_tmpe_seq \l_tikzspath_tmpe_tl
6072   \seq_pop_left:NN \l_tikzspath_tmpe_seq \l_tikzspath_tmpe_tl
6073
6074   \tl_set:Nn \l_tikzspath_tmpe_tl {#2}
6075   \spath_transform:NVnnnnn \l_tikzspath_tmpe_tl \l_tikzspath_tmpe_tl {1}{0}{0}{-
1}{Opt}{Opt}
6076
6077   \seq_map_indexed_inline:Nn \l_tikzspath_tmpe_seq
6078   {
6079     \int_compare:nTF
6080     {
6081       ##1 == \l_tikzspath_tmpe_tl
6082     }
6083     {
6084       \seq_pop_left:NNF \l_tikzspath_tmpe_seq \l_tikzspath_tmpe_tl
6085       {
6086         \tl_set:Nn \l_tikzspath_tmpe_tl {-1}
6087       }
6088     }
6089
6090     \spath_finalpoint:NV \l_tikzspath_tmpe_tl \l_tikzspath_tmpe_tl
6091     \spath_initialpoint:Nn \l_tikzspath_tmpe_tl {##2}
6092
6093     \dim_compare:nTF
6094     {

```

```

6094     \tl_item:Nn \l__tikzspath_tmpe_tl {1}
6095     >
6096     \tl_item:Nn \l__tikzspath_tmpf_tl {1}
6097   }
6098   {
6099     \spath_splice_between:NVn
6100     \l__tikzspath_tmpe_tl
6101     \l__tikzspath_tmpd_tl
6102     {##2}
6103   }
6104   {
6105     \spath_splice_between:NVn
6106     \l__tikzspath_tmpe_tl
6107     \l__tikzspath_tmpe_tl
6108     {##2}
6109   }
6110 }
6111 {
6112   \tl_put_right:Nn \l__tikzspath_tmpe_tl {##2}
6113 }
6114 }
6115 \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpe_tl
6116 \group_end:
6117 }
6118 \cs_new_protected_nopar:Npn \__tikzspath_join_components_upright_with:Nnnn #1#2#3#4
6119 {
6120   \__tikzspath_join_components_upright_with_aux:nnn {#2}{#3}{#4}
6121   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6122   \tl_gclear:N \g__tikzspath_output_tl
6123 }
6124 \cs_generate_variant:Nn \__tikzspath_join_components_upright_with:Nnnn {NVnn}
6125 \cs_new_protected_nopar:Npn \__tikzspath_join_components_upright_with:Nnn #1#2#3
6126 {
6127   \__tikzspath_join_components_upright_with:NVnn #1#1{#2}{#3}
6128 }
6129 \cs_generate_variant:Nn \__tikzspath_join_components_upright_with:Nnn {cvV}
6130 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_upright_with:Nnnn #1#2#3#4
6131 {
6132   \__tikzspath_join_components_upright_with_aux:nnn {#2}{#3}{#4}
6133   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6134   \tl_gclear:N \g__tikzspath_output_tl
6135 }
6136 \cs_generate_variant:Nn \__tikzspath_gjoin_components_upright_with:Nnnn {NVnn}
6137 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_upright_with:Nnn #1#2#3
6138 {
6139   \__tikzspath_gjoin_components_upright_with:NVnn #1#1{#2}{#3}
6140 }
6141 \cs_generate_variant:Nn \__tikzspath_gjoin_components_upright_with:Nnn {cvV}

```

(End definition for __tikzspath_join_components_with:Nnn __tikzspath_join_components_upright_-with:Nnn.)

__tikzspath_get_components:Nn Get the components of the named path to the token list.

```

6142 \cs_new_protected_nopar:Npn \__tikzspath_get_components_aux:n #1

```

```

6143 {
6144   \clist_gclear_new:N \g__tikzspath_output_clist
6145   \spath_components_to_seq:Nn \l__tikzspath_tmpa_seq {#1}
6146
6147   \seq_map_inline:Nn \l__tikzspath_tmpa_seq
6148   {
6149     \spath_anonymous:N \l__tikzspath_tmpa_tl
6150     \tl_new:c {\__tikzspath_path_name:V \l__tikzspath_tmpa_tl}
6151     \tl_set:cn {\__tikzspath_path_name:V \l__tikzspath_tmpa_tl} {##1}
6152     \clist_gput_right:NV \g__tikzspath_output_clist \l__tikzspath_tmpa_tl
6153   }
6154 }
6155 \cs_new_protected_nopar:Npn \__tikzspath_get_components:Nn #1#2
6156 {
6157   \clist_clear_new:N #1
6158   \__tikzspath_get_components_aux:n {#2}
6159   \clist_set_eq:NN #1 \g__tikzspath_output_clist
6160   \clist_gclear:N \g__tikzspath_output_clist
6161 }
6162 \cs_generate_variant:Nn \__tikzspath_get_components:Nn {NV, Nv}
6163 \cs_new_protected_nopar:Npn \__tikzspath_gget_components:Nn #1#2
6164 {
6165   \clist_gclear_new:N #1
6166   \__tikzspath_get_components_aux:n {#2}
6167   \clist_gset_eq:NN #1 \g__tikzspath_output_clist
6168   \clist_gclear:N \g__tikzspath_output_clist
6169 }
6170 \cs_generate_variant:Nn \__tikzspath_gget_components:Nn {NV, Nv}

```

(End definition for __tikzspath_get_components:Nn.)

__tikzspath_render_components:n

```

6171 \cs_new_protected_nopar:Npn \__tikzspath_render_components:nn #1#2
6172 {
6173   \group_begin:
6174   \spath_components_to_seq:Nn \l__tikzspath_tmpa_seq {#2}
6175   \seq_map_indexed_inline:Nn \l__tikzspath_tmpa_seq
6176   {
6177     \spath_tikz_path:nn
6178     {
6179       every~ spath~ component/.try,
6180       spath ~component~ ##1/.try,
6181       spath ~component/.try={##1},
6182       every~ #1~ component/.try,
6183       #1 ~component~ ##1/.try,
6184       #1 ~component/.try={##1},
6185     }
6186     {
6187       ##2
6188     }
6189   }
6190   \group_end:
6191 }
6192 \cs_generate_variant:Nn \__tikzspath_render_components:nn {nv}

```

(End definition for `_tikzspath_render_components:n`.)

`\tikzspath_insert_gaps_after_components:nn`

```
6193 \cs_new_protected_nopar:Npn \_tikzspath_insert_gaps_after_components_aux:nnn #1#2#3
6194 {
6195   \group_begin:
6196   \spath_numberofcomponents:Nn \l__tikzspath_tmpa_int {#1}
6197   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_int {#3}
6198
6199   \tl_if_empty:nT {#3}
6200   {
6201     \seq_pop_right:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6202   }
6203
6204   \seq_clear:N \l__tikzspath_tmpb_seq
6205   \seq_map_inline:Nn \l__tikzspath_tmpa_seq {
6206     \seq_put_right:Nx
6207     \l__tikzspath_tmpb_seq
6208     {\int_eval:n
6209      {
6210        \int_mod:nn { ##1 }{\l__tikzspath_tmpa_int } + 1
6211      }
6212    }
6213  }
6214
6215  \spath_components_to_seq:Nn \l__tikzspath_tmpc_seq {#1}
6216
6217  \seq_clear:N \l__tikzspath_tmpd_seq
6218  \seq_map_indexed_inline:Nn \l__tikzspath_tmpc_seq
6219  {
6220    \tl_set:Nn \l__tikzspath_tmpa_tl {##2}
6221    \seq_if_in:NnT \l__tikzspath_tmpa_seq {##1}
6222    {
6223      \spath_shorten_at_end:Nn \l__tikzspath_tmpa_tl {(##2)/2}
6224    }
6225    \seq_if_in:NnT \l__tikzspath_tmpb_seq {##1}
6226    {
6227      \spath_shorten_at_start:Nn \l__tikzspath_tmpa_tl {(##2)/2}
6228    }
6229    \seq_put_right:NV \l__tikzspath_tmpd_seq \l__tikzspath_tmpa_tl
6230  }
6231  \tl_gset:Nx \g__tikzspath_output_tl {\seq_use:Nn \l__tikzspath_tmpd_seq {} }
6232  \group_end:
6233 }
6234 \cs_new_protected_nopar:Npn \_tikzspath_insert_gaps_after_components:Nnnn #1#2#3#4
6235 {
6236   \_tikzspath_insert_gaps_after_components_aux:nnn {#2}{#3}{#4}
6237   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6238   \tl_gclear:N \g__tikzspath_output_tl
6239 }
6240 \cs_generate_variant:Nn \_tikzspath_insert_gaps_after_components:Nnnn {NVnn}
6241 \cs_new_protected_nopar:Npn \_tikzspath_insert_gaps_after_components:Nnn #1#2#3
6242 {
6243   \_tikzspath_insert_gaps_after_components:NVnn #1#1{#2}{#3}
```

```

6244 }
6245 \cs_generate_variant:Nn \__tikzspath_insert_gaps_after_components:Nnn {cnn, cVV}
6246 \cs_new_protected_nopar:Npn \__tikzspath_ginsert_gaps_after_components:Nnnn #1#2#3#4
6247 {
6248   \__tikzspath_insert_gaps_after_components_aux:nnn {#2}{#3}{#4}
6249   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6250   \tl_gclear:N \g__tikzspath_output_tl
6251 }
6252 \cs_generate_variant:Nn \__tikzspath_ginsert_gaps_after_components:Nnnn {NVnn}
6253 \cs_new_protected_nopar:Npn \__tikzspath_ginsert_gaps_after_components:Nnn #1#2#3
6254 {
6255   \__tikzspath_ginsert_gaps_after_components:NVnn #1#1{#2}{#3}
6256 }
6257 \cs_generate_variant:Nn \__tikzspath_ginsert_gaps_after_components:Nnn {cnn, cVV}

```

(End definition for __tikzspath_insert_gaps_after_components:nn.)

__tikzspath_insert_gaps_after_segments:Nn

```

6258 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_segments_aux:nnn #1#2#3
6259 {
6260   \group_begin:
6261   \spath_reallength:Nn \l__tikzspath_tmpa_int {#1}
6262   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_int {#3}
6263
6264   \tl_if_empty:nT {#3}
6265   {
6266     \seq_pop_right:NN \l__tikzspath_tmpb_seq \l__tikzspath_tmpa_tl
6267   }
6268
6269   \seq_clear:N \l__tikzspath_tmpb_seq
6270   \seq_map_inline:Nn \l__tikzspath_tmpa_seq {
6271     \seq_put_right:Nx
6272     \l__tikzspath_tmpb_seq
6273     {\int_eval:n
6274       {
6275         \int_mod:nn { ##1 }{\l__tikzspath_tmpa_int } + 1
6276       }
6277     }
6278   }
6279
6280   \spath_segments_to_seq:Nn \l__tikzspath_tmpc_seq {#1}
6281
6282   \seq_clear:N \l__tikzspath_tmpe_seq
6283   \seq_map_indexed_inline:Nn \l__tikzspath_tmpc_seq
6284   {
6285     \tl_set:Nn \l__tikzspath_tmpe_tl {##2}
6286     \seq_if_in:NnT \l__tikzspath_tmpe_seq {##1}
6287     {
6288       \spath_shorten_at_end:Nn \l__tikzspath_tmpe_tl {(#2)/2}
6289     }
6290     \seq_if_in:NnT \l__tikzspath_tmpe_seq {##1}
6291     {
6292       \spath_shorten_at_start:Nn \l__tikzspath_tmpe_tl {(#2)/2}
6293     }

```

```

6294     \seq_put_right:NV \l__tikzspath_tmpd_seq \l__tikzspath_tmpa_tl
6295   }
6296   \tl_gset:Nx \g__tikzspath_output_tl {\seq_use:Nn \l__tikzspath_tmpd_seq {}}
6297   \group_end:
6298 }
6299 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_segments:Nnnn #1#2#3#4
6300 {
6301   \__tikzspath_insert_gaps_after_segments_aux:nnn {#2}{#3}{#4}
6302   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6303   \tl_gclear:N \g__tikzspath_output_tl
6304 }
6305 \cs_generate_variant:Nn \__tikzspath_insert_gaps_after_segments:Nnnn {NVnn}
6306 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_segments:Nnn #1#2#3
6307 {
6308   \__tikzspath_insert_gaps_after_segments:NVnn #1#1{#2}{#3}
6309 }
6310 \cs_generate_variant:Nn \__tikzspath_insert_gaps_after_segments:Nnn {cnn, cVV}
6311 \cs_new_protected_nopar:Npn \__tikzspath_ginsert_gaps_after_segments:Nnnn #1#2#3#4
6312 {
6313   \__tikzspath_insert_gaps_after_segments_aux:nnn {#2}{#3}{#4}
6314   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6315   \tl_gclear:N \g__tikzspath_output_tl
6316 }
6317 \cs_generate_variant:Nn \__tikzspath_ginsert_gaps_after_segments:Nnnn {NVnn}
6318 \cs_new_protected_nopar:Npn \__tikzspath_ginsert_gaps_after_segments:Nnn #1#2#3
6319 {
6320   \__tikzspath_ginsert_gaps_after_segments:NVnn #1#1{#2}{#3}
6321 }
6322 \cs_generate_variant:Nn \__tikzspath_ginsert_gaps_after_segments:Nnn {cnn, cVV}

```

(End definition for __tikzspath_insert_gaps_after_segments:Nn.)

__tikzspath_join_components:Nn

```

6323 \cs_new_protected_nopar:Npn \__tikzspath_join_components_aux:nn #1#2
6324 {
6325   \group_begin:
6326
6327   \tl_set:Nn \l__tikzspath_tmpa_tl {#1}
6328   \spath_numberofcomponents:NV \l__tikzspath_tmpa_int \l__tikzspath_tmpa_tl
6329   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_int {#2}
6330
6331   \seq_reverse:N \l__tikzspath_tmpa_seq
6332
6333   \seq_map_inline:Nn \l__tikzspath_tmpa_seq
6334   {
6335     \spath_join_component:Nn \l__tikzspath_tmpa_tl {##1}
6336   }
6337   \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpa_tl
6338   \group_end:
6339 }
6340 \cs_new_protected_nopar:Npn \__tikzspath_join_components:Nnn #1#2#3
6341 {
6342   \__tikzspath_join_components_aux:nn {#2}{#3}
6343   \tl_set_eq:NN #1 \g__tikzspath_output_tl

```



```

6344 \tl_gclear:N \g__tikzspath_output_tl
6345 }
6346 \cs_generate_variant:Nn \__tikzspath_join_components:Nnn {NVn}
6347 \cs_new_protected_nopar:Npn \__tikzspath_join_components:Nn #1#2
6348 {
6349   \__tikzspath_join_components:NVn #1#1{#2}
6350 }
6351 \cs_generate_variant:Nn \__tikzspath_join_components:Nn {cn}
6352 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components:Nnn #1#2#3
6353 {
6354   \__tikzspath_join_components_aux:nn {#2}{#3}
6355   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6356   \tl_gclear:N \g__tikzspath_output_tl
6357 }
6358 \cs_generate_variant:Nn \__tikzspath_gjoin_components:Nnn {NVn}
6359 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components:Nn #1#2
6360 {
6361   \__tikzspath_gjoin_components:NVn #1#1{#2}
6362 }
6363 \cs_generate_variant:Nn \__tikzspath_gjoin_components:Nn {cn}

```

(End definition for __tikzspath_join_components:Nn.)

__tikzspath_join_components_with_bezier:Nn

```

6364 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with_bezier_aux:nn #1#2
6365 {
6366   \group_begin:
6367   \tl_set:Nn \l__tikzspath_tmpc_tl {#1}
6368   \tl_if_empty:nT {#2}
6369   {
6370     \spath_spot_weld_components:N \l__tikzspath_tmpc_tl
6371   }
6372
6373   \spath_numberofcomponents:NV \l__tikzspath_tmpa_int \l__tikzspath_tmpc_tl
6374   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpb_seq \l__tikzspath_tmpa_int {#2}
6375
6376   \spath_components_to_seq:NV \l__tikzspath_tmpa_seq \l__tikzspath_tmpc_tl
6377
6378   \seq_pop_left:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6379   \seq_pop_left:NN \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6380
6381   \seq_map_indexed_inline:Nn \l__tikzspath_tmpa_seq
6382   {
6383     \int_compare:nTF
6384     {
6385       ##1 == \l__tikzspath_tmpb_tl
6386     }
6387     {
6388       \seq_pop_left:NNF \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6389       {
6390         \tl_set:Nn \l__tikzspath_tmpb_tl {-1}
6391       }
6392       \spath_curve_between:Nn \l__tikzspath_tmpa_tl {##2}
6393     }

```

```

6394     {
6395     \tl_put_right:Nn \l__tikzspath_tmpa_tl {##2}
6396     }
6397   }
6398   \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpa_tl
6399   \group_end:
6400 }
6401 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with_bezier:Nnn #1#2#3
6402 {
6403   \__tikzspath_join_components_with_bezier_aux:nn {#2}{#3}
6404   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6405   \tl_gclear:N \g__tikzspath_output_tl
6406 }
6407 \cs_generate_variant:Nn \__tikzspath_join_components_with_bezier:Nnn {NVn}
6408 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with_bezier:Nn #1#2
6409 {
6410   \__tikzspath_join_components_with_bezier:NVn #1#1{#2}
6411 }
6412 \cs_generate_variant:Nn \__tikzspath_join_components_with_bezier:Nn {cV}
6413 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_with_bezier:Nnn #1#2#3
6414 {
6415   \__tikzspath_join_components_with_bezier_aux:nn {#2}{#3}
6416   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6417   \tl_gclear:N \g__tikzspath_output_tl
6418 }
6419 \cs_generate_variant:Nn \__tikzspath_gjoin_components_with_bezier:Nnn {NVn}
6420 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_with_bezier:Nn #1#2
6421 {
6422   \__tikzspath_gjoin_components_with_bezier:NVn #1#1{#2}
6423 }
6424 \cs_generate_variant:Nn \__tikzspath_gjoin_components_with_bezier:Nn {cV}

```

(End definition for __tikzspath_join_components_with_bezier:Nn.)

__tikzspath_remove_components:nn

```

6425 \cs_new_protected_nopar:Npn \__tikzspath_remove_components_aux:nn #1#2
6426 {
6427   \group_begin:
6428
6429   \spath_numberofcomponents:Nn \l__tikzspath_tmpa_int {#1}
6430   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_int {#2}
6431
6432   \spath_components_to_seq:Nn \l__tikzspath_tmpb_seq {#1}
6433
6434   \seq_pop_left:NNF \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6435   {
6436     \tl_clear:N \l__tikzspath_tmpa_tl
6437   }
6438
6439   \seq_clear:N \l__tikzspath_tmpe_seq
6440   \seq_map_indexed_inline:Nn \l__tikzspath_tmpe_seq
6441   {
6442     \tl_set:Nn \l__tikzspath_tmpe_tl {##1}
6443     \tl_if_eq:NNTF \l__tikzspath_tmpe_tl \l__tikzspath_tmpe_tl

```

```

6444     {
6445       \seq_pop_left:NNF \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6446       {
6447         \tl_clear:N \l__tikzspath_tmpa_tl
6448       }
6449     }
6450     {
6451       \seq_put_right:Nn \l__tikzspath_tmpe_seq {##2}
6452     }
6453   }
6454
6455   \tl_gset:Nx \g__tikzspath_output_tl {\seq_use:Nn \l__tikzspath_tmpe_seq {} }
6456   \group_end:
6457 }
6458 \cs_new_protected_nopar:Npn \__tikzspath_remove_components:Nnn #1#2#3
6459 {
6460   \__tikzspath_remove_components_aux:nn {#2}{#3}
6461   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6462   \tl_gclear:N \g__tikzspath_output_tl
6463 }
6464 \cs_generate_variant:Nn \__tikzspath_remove_components:Nnn {NVn}
6465 \cs_new_protected_nopar:Npn \__tikzspath_remove_components:Nn #1#2
6466 {
6467   \__tikzspath_remove_components:NVn #1#1{#2}
6468 }
6469 \cs_generate_variant:Nn \__tikzspath_remove_components:Nn {cn}
6470 \cs_new_protected_nopar:Npn \__tikzspath_gremove_components:Nnn #1#2#3
6471 {
6472   \__tikzspath_remove_components_aux:nn {#2}{#3}
6473   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6474   \tl_gclear:N \g__tikzspath_output_tl
6475 }
6476 \cs_generate_variant:Nn \__tikzspath_gremove_components:Nnn {NVn}
6477 \cs_new_protected_nopar:Npn \__tikzspath_gremove_components:Nn #1#2
6478 {
6479   \__tikzspath_gremove_components:NVn #1#1{#2}
6480 }
6481 \cs_generate_variant:Nn \__tikzspath_gremove_components:Nn {cn}

```

(End definition for __tikzspath_remove_components:nn.)

__tikzspath_transform_to:nn

__tikzspath_transform_upright_to:nn

```

6482 \cs_new_protected_nopar:Npn \__tikzspath_transform_to_aux:nn #1#2
6483 {
6484   \group_begin:
6485   \spath_reallength:Nn \l__tikzspath_tmpa_int {#2}
6486
6487   \tl_set:Nx \l__tikzspath_tmpb_tl
6488   {\fp_to_decimal:n {(#1) * (\l__tikzspath_tmpa_int)}}
6489   \spath_transformation_at:NnV \l__tikzspath_tmpe_tl {#2} \l__tikzspath_tmpb_tl
6490   \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpe_tl
6491   \group_end:
6492 }
6493 \cs_new_protected_nopar:Npn \__tikzspath_transform_to:nn #1#2

```

```

6494 {
6495   \__tikzspath_transform_to_aux:nn {#1}{#2}
6496   \exp_last_unbraced:NV \pgfsettransformentries \g__tikzspath_output_tl
6497   \tl_gclear:N \g__tikzspath_output_tl
6498 }
6499 \cs_generate_variant:Nn \__tikzspath_transform_to:nn {nv}
6500 \cs_new_protected_nopar:Npn \__tikzspath_transform_upright_to:nn #1#2
6501 {
6502   \__tikzspath_transform_to_aux:nn {#1}{#2}
6503   \fp_compare:nT { \tl_item:Nn \g__tikzspath_output_tl {4} < 0 }
6504   {
6505     \tl_gset:Nx \g__tikzspath_output_tl
6506     {
6507       { \fp_eval:n { - (\tl_item:Nn \g__tikzspath_output_tl {1}) } } }
6508       { \fp_eval:n { - (\tl_item:Nn \g__tikzspath_output_tl {2}) } } }
6509       { \fp_eval:n { - (\tl_item:Nn \g__tikzspath_output_tl {3}) } } }
6510       { \fp_eval:n { - (\tl_item:Nn \g__tikzspath_output_tl {4}) } } }
6511       { \tl_item:Nn \g__tikzspath_output_tl {5} } }
6512       { \tl_item:Nn \g__tikzspath_output_tl {6} } }
6513     }
6514   }
6515   \exp_last_unbraced:NV \pgfsettransformentries \g__tikzspath_output_tl
6516   \tl_gclear:N \g__tikzspath_output_tl
6517 }
6518 \cs_generate_variant:Nn \__tikzspath_transform_upright_to:nn {nv}

```

(End definition for __tikzspath_transform_to:nn and __tikzspath_transform_upright_to:nn.)

4.2 Keys

Now we define all of our keys.

```

6519 \tikzset{

```

We're in the `spath` key family.

```

6520   spath/.is~family,
6521   spath/.cd,

```

We provide for saving soft paths with a specific prefix and suffix in the name. The default is to make it compatible with the intersections library.

```

6522   set~ prefix/.store~ in=\l__tikzspath_prefix_tl,
6523   prefix/.is~choice,
6524   prefix/default/.style={
6525     /tikz/spath/set~ prefix=tikz@intersect@path@name@
6526   },
6527   set~ suffix/.store~ in=\l__tikzspath_suffix_tl,
6528   suffix/.is~choice,
6529   suffix/default/.style={
6530     /tikz/spath/set~ suffix={}
6531   },
6532   set~ name/.style={
6533     /tikz/spath/prefix=#1,
6534     /tikz/spath/suffix=#1
6535   },

```

Hook in to the end of the path construction

```
6536 at~ end~ path~ construction/.code={
6537   \tl_put_right:Nn \l__tikzspath_tikzpath_finish_tl {#1}
6538 },
```

Keys for saving and cloning a soft path.

```
6539 save/.code={
6540   \tikz@addmode{
6541     \spath_get_current_path:N \l__tikzspath_tmpa_tl
6542     \spath_bake_round:NV \l__tikzspath_tmpa_tl \l__tikzspath_tmpa_tl
6543     \spath_bake_shorten:NV \l__tikzspath_tmpa_tl \l__tikzspath_tmpa_tl
6544     \spath_save_path:cV {\__tikzspath_path_name:n {#1}}
6545     \l__tikzspath_tmpa_tl
6546   }
6547 },
6548 save~ global/.code={
6549   \tikz@addmode{
6550     \spath_get_current_path:N \l__tikzspath_tmpa_tl
6551     \spath_bake_round:NV \l__tikzspath_tmpa_tl \l__tikzspath_tmpa_tl
6552     \spath_bake_shorten:NV \l__tikzspath_tmpa_tl \l__tikzspath_tmpa_tl
6553     \spath_gsave_path:cV {\__tikzspath_path_name:n {#1}}
6554     \l__tikzspath_tmpa_tl
6555   }
6556 },
6557 clone/.code~ 2~ args={
6558   \__tikzspath_maybe_current_path:nn
6559   {
6560     \__tikzspath_check_path:nnn {
6561       \tl_clear_new:c {\__tikzspath_path_name:n {#1}}
6562       \tl_set_eq:cc {\__tikzspath_path_name:n {#1}}
6563     }
6564   }
6565   {#2}{}
6566 },
6567 clone~ global/.code~ 2~ args={
6568   \__tikzspath_maybe_current_path:nn
6569   {
6570     \__tikzspath_check_path:nnn {
6571       \tl_gclear_new:c {\__tikzspath_path_name:n {#1}}
6572       \tl_gset_eq:cc {\__tikzspath_path_name:n {#1}}
6573     }
6574   }
6575   {#2}{}
6576 },
```

Saves a soft path to the aux file.

```
6577 save~ to~ aux/.code={
6578   \__tikzspath_maybe_current_path:nn
6579   {
6580     \__tikzspath_check_path:nnn {
6581       \spath_save_to_aux:c
6582     }
6583   }
6584   {#1}
```

```

6585     {}
6586   },

```

Exports the path as an SVG file.

```

6587   export~ to~ svg/.code={
6588     \__tikzspath_maybe_current_path:nn
6589     {
6590       \__tikzspath_check_path:nnn {
6591         \spath_export_to_svg:nv {#1}
6592       }
6593     }
6594     {#1}
6595     {}
6596   },

```

Inserts the named path at the current point in the path, with options for how this is accomplished. The inserted path can be transformed, reversed, moved to the current point, and welded to the current path. If this is used before the path has been started then it becomes the start of the path (and the “current point” is taken as the origin).

```

6597   use/.code={
6598     \bool_set_false:N \l__tikzspath_reverse_bool
6599     \bool_set_false:N \l__tikzspath_weld_bool
6600     \bool_set_false:N \l__tikzspath_move_bool
6601     \bool_set_false:N \l__tikzspath_current_transformation_bool
6602     \bool_set_true:N \l_spath_movetorelevant_bool
6603     \tl_clear:N \l__tikzspath_joinpath_tl
6604     \tl_clear:N \l__tikzspath_transformation_tl
6605     \tikzset{
6606       spath/join/.cd,
6607       #1
6608     }
6609
6610     \__tikzspath_check_path:nVn
6611     {
6612       \__tikzspath_use_path:v
6613     } \l__tikzspath_joinpath_tl {}
6614
6615   },

```

Some aliases for the above.

```

6616   restore/.style={/tikz/spath/use={#1}},
6617   restore~ reverse/.style={/tikz/spath/use={reverse, #1}},
6618   append/.style={/tikz/spath/use={move, weld, #1}},
6619   append~ no~ move/.style={/tikz/spath/use={weld, #1}},
6620   append~ reverse/.style={/tikz/spath/use={move, weld, reverse, #1}},
6621   append~ reverse~ no~ move/.style={/tikz/spath/use={weld, reverse, #1}},
6622   insert/.style={/tikz/spath/use={#1}},
6623   insert~ reverse/.style={/tikz/spath/use={reverse, #1}},

```

Diagnostic, show the current path in the terminal and log.

```

6624   show~current~path/.code={
6625     \tikz@addmode{
6626       \pgfsyssoftpath@getcurrentpath\l__tikzspath_tmpa_tl
6627       \iow_term:n {---- current~ soft~ path~ ---}

```

```

6628     \spath_show:V \l__tikzspath_tmpa_tl
6629   }
6630 },

```

Diagnostic, show the named soft path in the terminal and log.

```

6631 show/.code={
6632   \__tikzspath_check_path:nnn {
6633     \iow_term:n {----~ soft~ path~ #1~ ---}
6634     \spath_show:v
6635   } {#1} {}
6636 },

```

This joins a path on to an existing path, possibly modifying it first. The possible options are the same as those for `use`. It is possible to specify the same path both for the initial and the joining path as a copy is made internally first.

```

6637 join~ with/.code~ 2~ args={
6638   \bool_set_false:N \l__tikzspath_reverse_bool
6639   \bool_set_false:N \l__tikzspath_weld_bool
6640   \bool_set_false:N \l__tikzspath_move_bool
6641   \bool_set_false:N \l__tikzspath_global_bool
6642   \bool_set_false:N \l__tikzspath_current_transformation_bool
6643   \tl_clear:N \l__tikzspath_joinpath_tl
6644   \tl_clear:N \l__tikzspath_transformation_tl
6645   \tikzset{
6646     spath/join/.cd,
6647     #2
6648   }
6649
6650   \__tikzspath_maybe_current_path_reuse:nnn
6651   {
6652     \__tikzspath_check_two_paths:nnVn
6653     {
6654       \__tikzspath_join_with:cv
6655     }
6656   } {#1} { \l__tikzspath_joinpath_tl {} }
6657 },

```

Does a “spot weld” on a soft path, which means that any components that start where the previous component ends are welded together.

```

6658 spot~ weld/.code={
6659   \__tikzspath_maybe_current_path_reuse:nnn
6660   {
6661     \__tikzspath_check_path:nnn
6662     {
6663       \spath_spot_weld_components:c
6664     }
6665   } {#1} { {} }
6666 },
6667 spot~ weld~ globally/.code={
6668   \__tikzspath_maybe_current_path_reuse:nnn
6669   {
6670     \__tikzspath_check_path:nnn
6671     {
6672       \spath_spot_gweld_components:c

```

```

6673     }
6674   } {#1} { {} }
6675 },

```

Reverses the named path.

```

6676 reverse/.code={
6677   \__tikzspath_maybe_current_path_reuse:nmn
6678   {
6679     \__tikzspath_check_path:nmn
6680     {
6681       \spath_reverse:c
6682     }
6683   } {#1} { {} }
6684 },
6685 reverse~ globally/.code={
6686   \__tikzspath_maybe_current_path_reuse:nmn
6687   {
6688     \__tikzspath_check_path:nmn
6689     {
6690       \spath_greverse:c
6691     }
6692   } {#1} { {} }
6693 },

```

Adjust a path to span between two points.

```

6694 span/.code ~n~ args={3}{
6695   \__tikzspath_maybe_current_path_reuse:nmn
6696   {
6697     \__tikzspath_check_path:nmn
6698     {
6699       \__tikzspath_process_tikz_point:Nn \l__tikzspath_tmpa_tl {#2}
6700       \__tikzspath_process_tikz_point:Nn \l__tikzspath_tmpb_tl {#3}
6701       \spath_span:cVV
6702     }
6703   } {#1} { {} } \l__tikzspath_tmpa_tl \l__tikzspath_tmpb_tl }
6704 },
6705 span~ global/.code ~n~ args={3}{
6706   \__tikzspath_maybe_current_path_reuse:nmn
6707   {
6708     \__tikzspath_check_path:nmn
6709     {
6710       \__tikzspath_process_tikz_point:Nn \l__tikzspath_tmpa_tl {#2}
6711       \__tikzspath_process_tikz_point:Nn \l__tikzspath_tmpb_tl {#3}
6712       \spath_span:cVV
6713     }
6714   } {#1} { {} } \l__tikzspath_tmpa_tl \l__tikzspath_tmpb_tl }
6715 },

```

Defines a to path

```

6716 to/.style={
6717   to~path={
6718     [
6719       spath/span={#1}{(\tikztostart)}{(\tikztotarget)},
6720       spath/use={#1,weld},

```



```

6721     ]
6722     \tikztonodes
6723   }
6724 },

```

Splice three paths together, transforming the middle one so that it exactly fits between the first and third.

```

6725 splice/.code ~n~ args={3}{
6726   \__tikzspath_maybe_current_path_reuse:nnn
6727   {
6728     \__tikzspath_check_three_paths:nnnnn
6729     {
6730       \spath_splice_between:cvv
6731     }
6732   } {#1} { {#2} {#3} {} }
6733 },
6734 splice~ global/.code ~n~ args={3}{
6735   \__tikzspath_maybe_current_path_reuse:nnn
6736   {
6737     \__tikzspath_check_three_paths:nnnnn
6738     {
6739       \spath_gsplice_between:cvv
6740     }
6741   } {#1} { {#2} {#3} {} }
6742 },

```

Join the components of a path by splicing in the second path whenever the components are sufficiently far apart. The third argument is a list of components to splice after, if it is empty then all components are used and a spot weld is done first so that the splicing only happens if there is an actual gap.

The upright versions will join with the reflection of the splice path if it detects that the gap is “upside-down”.

```

6743 join~ components~ with/.code~2~args={
6744   \tl_if_head_is_group:nTF {#2}
6745   {
6746     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
6747     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
6748   }
6749   {
6750     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6751     \tl_clear:N \l__tikzspath_tmpd_tl
6752   }
6753 },
6754 \__tikzspath_maybe_current_path_reuse:nnn
6755 {
6756   \__tikzspath_check_two_paths:nnVn
6757   {
6758     \__tikzspath_join_components_with:cvV
6759   }
6760 } {#1} { \l__tikzspath_tmpc_tl {} \l__tikzspath_tmpd_tl }
6761 },
6762 join~ components~ globally~ with/.code~2~args={
6763   \tl_if_head_is_group:nTF {#2}
6764   {

```

```

6765     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
6766     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
6767   }
6768   {
6769     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6770     \tl_clear:N \l__tikzspath_tmpd_tl
6771   }
6772
6773   \__tikzspath_maybe_current_path_reuse:nnn
6774   {
6775     \__tikzspath_check_two_paths:nnVn
6776     {
6777       \__tikzspath_gjoin_components_with:cvV
6778     }
6779   } {#1} { \l__tikzspath_tmpc_tl {} \l__tikzspath_tmpd_tl }
6780 },
6781 join~ components~ upright~ with/.code~2~args={
6782   \tl_if_head_is_group:nTF {#2}
6783   {
6784     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
6785     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
6786   }
6787   {
6788     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6789     \tl_clear:N \l__tikzspath_tmpd_tl
6790   }
6791
6792   \__tikzspath_maybe_current_path_reuse:nnn
6793   {
6794     \__tikzspath_check_two_paths:nnVn
6795     {
6796       \__tikzspath_join_components_upright_with:cvV
6797     }
6798   } {#1} { \l__tikzspath_tmpc_tl {} \l__tikzspath_tmpd_tl }
6799 },
6800 join~ components~ globally~ upright~ with/.code~2~args={
6801   \tl_if_head_is_group:nTF {#2}
6802   {
6803     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
6804     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
6805   }
6806   {
6807     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6808     \tl_clear:N \l__tikzspath_tmpd_tl
6809   }
6810
6811   \__tikzspath_maybe_current_path_reuse:nnn
6812   {
6813     \__tikzspath_check_two_paths:nnVn
6814     {
6815       \__tikzspath_gjoin_components_upright_with:cvV
6816     }
6817   } {#1} { \l__tikzspath_tmpc_tl {} \l__tikzspath_tmpd_tl }
6818 },

```

```

6819 join~ components~ with~ bezier/.code={
6820   \tl_if_head_is_group:nTF {#1}
6821   {
6822     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#1} {1} }
6823     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#1} {2} }
6824   }
6825   {
6826     \tl_set:Nn \l__tikzspath_tmpc_tl {#1}
6827     \tl_clear:N \l__tikzspath_tmpd_tl
6828   }
6829
6830   \__tikzspath_maybe_current_path_reuse:nVn
6831   {
6832     \__tikzspath_check_path:nnn
6833     {
6834       \__tikzspath_join_components_with_bezier:cV
6835     }
6836   } \l__tikzspath_tmpc_tl { {} } \l__tikzspath_tmpd_tl }
6837 },
6838 join~ components~ globally~ with~ bezier/.code~2~args={
6839   \__tikzspath_maybe_current_path_reuse:nnn
6840   {
6841     \__tikzspath_check_path:nnn
6842     {
6843       \__tikzspath_gjoin_components_with_bezier:cn
6844     }
6845   } {#1} { {} } {#2} }
6846 },

```

Close a path.

```

6847 close/.code={
6848   \__tikzspath_maybe_current_path_reuse:nnn
6849   {
6850     \__tikzspath_check_path:nnn
6851     {
6852       \spath_close:c
6853     }
6854   } {#1} { {} } }
6855 },
6856 close~ globally/.code={
6857   \__tikzspath_maybe_current_path_reuse:nnn
6858   {
6859     \__tikzspath_check_path:nnn
6860     {
6861       \spath_gclose:c
6862     }
6863   } {#1} { {} } }
6864 },

```

Close a path, ensuring that the end point is exactly where it will close up to.

```

6865 adjust~ and~ close/.code={
6866   \__tikzspath_maybe_current_path_reuse:nnn
6867   {
6868     \__tikzspath_check_path:nnn

```

```

6869     {
6870     \spath_adjust_close:c
6871     }
6872   } {#1} { {} }
6873 },
6874 adjust~ and~ close~ globally/.code={
6875   \__tikzspath_maybe_current_path_reuse:nnn
6876   {
6877     \__tikzspath_check_path:nnn
6878     {
6879       \spath_adjust_gclose:c
6880     }
6881   } {#1} { {} }
6882 },

```

Close a path with another path.

```

6883 close~ with/.code~ 2~ args={
6884   \__tikzspath_maybe_current_path_reuse:nnn
6885   {
6886     \__tikzspath_check_two_paths:nnnn
6887     {
6888       \spath_close_with:cv
6889     }
6890   } {#1} { {#2} {} }
6891 },
6892 close~ globally~ with/.code~ 2~ args={
6893   \__tikzspath_maybe_current_path_reuse:nnn
6894   {
6895     \__tikzspath_check_two_paths:nnnn
6896     {
6897       \spath_gclose_with:cv
6898     }
6899   } {#1} { {#2} {} }
6900 },

```

Close a path with a curve.

```

6901 close~ with~ curve/.code={
6902   \__tikzspath_maybe_current_path_reuse:nnn
6903   {
6904     \__tikzspath_check_path:nnn
6905     {
6906       \spath_close_with_curve:c
6907     }
6908   } {#1} { {} }
6909 },
6910 close~ globally~ with~ curve/.code={
6911   \__tikzspath_maybe_current_path_reuse:nnn
6912   {
6913     \__tikzspath_check_path:nnn
6914     {
6915       \spath_gclose_with_curve:c
6916     }
6917   } {#1} { {} }
6918 },

```

These keys shorten the path by a dimension.

```
6919 shorten~ at~ end/.code~ 2~ args={
6920   \__tikzspath_maybe_current_path_reuse:nnn
6921   {
6922     \__tikzspath_check_path:nnn
6923     {
6924       \spath_shorten_at_end:cn
6925     }
6926   } {#1} { {} } {#2} }
6927 },
6928 shorten~ at~ start/.code~ 2~ args ={
6929   \__tikzspath_maybe_current_path_reuse:nnn
6930   {
6931     \__tikzspath_check_path:nnn
6932     {
6933       \spath_shorten_at_start:cn
6934     }
6935   } {#1} { {} } {#2} }
6936 },
6937 shorten~ at~ both~ ends/.code~ 2~ args={
6938   \__tikzspath_maybe_current_path_reuse:nnn
6939   {
6940     \__tikzspath_check_path:nnn
6941     {
6942       \spath_shorten_at_both_ends:cn
6943     }
6944   } {#1} { {} } {#2} }
6945 },
6946 shorten~ globally~ at~ end/.code~ 2~ args={
6947   \__tikzspath_maybe_current_path_reuse:nnn
6948   {
6949     \__tikzspath_check_path:nnn
6950     {
6951       \spath_gshorten_at_end:cn
6952     }
6953   } {#1} { {} } {#2} }
6954 },
6955 shorten~ globally~ at~ start/.code~ 2~ args ={
6956   \__tikzspath_maybe_current_path_reuse:nnn
6957   {
6958     \__tikzspath_check_path:nnn
6959     {
6960       \spath_gshorten_at_start:cn
6961     }
6962   } {#1} { {} } {#2} }
6963 },
6964 shorten~ globally~ at~ both~ ends/.code~ 2~ args={
6965   \__tikzspath_maybe_current_path_reuse:nnn
6966   {
6967     \__tikzspath_check_path:nnn
6968     {
6969       \spath_gshorten_at_both_ends:cn
6970     }
6971   } {#1} { {} } {#2} }
```

```
6972 },
```

These keys split a path at a parameter, the `keep` versions only keep one part of the resultant path.

```
6973 split~ at/.code~ 2~ args={
6974   \__tikzspath_maybe_current_path_reuse:nnn
6975   {
6976     \__tikzspath_check_path:nnn
6977     {
6978       \spath_split_at_normalised:cn
6979     }
6980   } {#1} { {} } {#2} }
6981 },
6982 split~ globally~ at/.code~ 2~ args={
6983   \__tikzspath_maybe_current_path_reuse:nnn
6984   {
6985     \__tikzspath_check_path:nnn
6986     {
6987       \spath_gsplit_at_normalised:cn
6988     }
6989   } {#1} { {} } {#2} }
6990 },
6991 split~ at~ into/.code~ n~ args={4}{
6992   \__tikzspath_maybe_current_path_reuse:nnn
6993   {
6994     \__tikzspath_check_path:nnn
6995     {
6996       \spath_split_at_normalised:ccvn {\__tikzspath_path_name:n {#1}}
6997       {\__tikzspath_path_name:n {#2}}
6998     }
6999   } {#3} { {} } {#4} }
7000 },
7001 split~ globally~ at~ into/.code~ n~ args={4}{
7002   \__tikzspath_maybe_current_path_reuse:nnn
7003   {
7004     \__tikzspath_check_path:nnn
7005     {
7006       \spath_gsplit_at_normalised:ccvn {\__tikzspath_path_name:n {#1}}
7007       {\__tikzspath_path_name:n {#2}}
7008     }
7009   } {#3} { {} } {#4} }
7010 },
7011 split~ at~ keep~ start/.code~ 2~ args={
7012   \__tikzspath_maybe_current_path_reuse:nnn
7013   {
7014     \__tikzspath_check_path:nnn
7015     {
7016       \spath_split_at_normalised_keep_start:cn
7017     }
7018   } {#1} { {} } {#2} }
7019 },
7020 split~ globally~ at~ keep~ start/.code~ 2~ args={
7021   \__tikzspath_maybe_current_path_reuse:nnn
7022   {
```

```

7023     \tikzspath_check_path:nnn
7024     {
7025     \spath_gsplit_at_normalised_keep_start:cn
7026     }
7027   } {#1} { {} {#2} }
7028 },
7029 split~ at~ keep~ end/.code~ 2~ args={
7030   \tikzspath_maybe_current_path_reuse:nnn
7031   {
7032     \tikzspath_check_path:nnn
7033     {
7034       \spath_split_at_normalised_keep_end:cn
7035     }
7036   } {#1} { {} {#2} }
7037 },
7038 split~ globally~ at~ keep~ end/.code~ 2~ args={
7039   \tikzspath_maybe_current_path_reuse:nnn
7040   {
7041     \tikzspath_check_path:nnn
7042     {
7043       \spath_gsplit_at_normalised_keep_end:cn
7044     }
7045   } {#1} { {} {#2} }
7046 },
7047 split~ at~ keep~ middle/.style~ n~ args={3}{
7048   /tikz/spath/split~ at~ keep~ start={#1}{#3},
7049   /tikz/spath/split~ at~ keep~ end={#1}{(#2)/(#3)},
7050 },
7051 split~ globally~ at~ keep~ middle/.style~ n~ args={3}{
7052   /tikz/spath/split~ globally~ at~ keep~ start={#1}{#3},
7053   /tikz/spath/split~ globally~ at~ keep~ end={#1}{(#2)/(#3)},
7054 },

```

This translates the named path.

```

7055 translate/.code~ n~ args={3}{
7056   \tikzspath_maybe_current_path_reuse:nnn
7057   {
7058     \tikzspath_check_path:nnn
7059     {
7060       \spath_translate:cn
7061     }
7062   } {#1} { {} {#2}{#3} }
7063 },
7064 translate~ globally/.code~ n~ args={3}{
7065   \tikzspath_maybe_current_path_reuse:nnn
7066   {
7067     \tikzspath_check_path:nnn
7068     {
7069       \spath_gtranslate:cn
7070     }
7071   } {#1} { {} {#2}{#3} }
7072 },

```

This normalises the named path.

```

7073 normalise/.code={
7074   \__tikzspath_maybe_current_path_reuse:nnn
7075   {
7076     \__tikzspath_check_path:nnn
7077     {
7078       \spath_normalise:c
7079     }
7080   } {#1} { {} }
7081 },
7082 normalise~ globally/.code={
7083   \__tikzspath_maybe_current_path_reuse:nnn
7084   {
7085     \__tikzspath_check_path:nnn
7086     {
7087       \spath_gnormalise:c
7088     }
7089   } {#1} { {} }
7090 },

```

Transforms the named path using TikZ transformation specifications.

```

7091 transform/.code~ 2~ args={
7092   \group_begin:
7093   \pgftransformreset
7094   \tikzset{#2}
7095   \pgfgettransform \l__tikzspath_tmpa_tl
7096   \tl_gset_eq:NN \g__tikzspath_smuggle_tl \l__tikzspath_tmpa_tl
7097   \group_end:
7098
7099   \__tikzspath_maybe_current_path_reuse:nnn
7100   {
7101     \__tikzspath_check_path:nnn
7102     {
7103       \spath_transform:cV
7104     }
7105   } {#1} { {} } \g__tikzspath_smuggle_tl }
7106 },
7107 transform~globally/.code~ 2~ args={
7108   \group_begin:
7109   \pgftransformreset
7110   \tikzset{#2}
7111   \pgfgettransform \l__tikzspath_tmpa_tl
7112   \tl_gset_eq:NN \g__tikzspath_smuggle_tl \l__tikzspath_tmpa_tl
7113   \group_end:
7114
7115   \__tikzspath_maybe_current_path_reuse:nnn
7116   {
7117     \__tikzspath_check_path:nnn
7118     {
7119       \spath_gtransform:cV
7120     }
7121   } {#1} { {} } \g__tikzspath_smuggle_tl }
7122 },

```

Splits first path where it intersects with the second.


```

7123 split~ at~ intersections~ with/.code~ 2~ args={
7124   \tl_if_exist:cTF
7125   {
7126     tikz@library@intersections@loaded
7127   }
7128   {
7129     \__tikzspath_maybe_current_two_paths_reuse_first:nmmm
7130     {
7131       \__tikzspath_check_two_paths:nmmm
7132       {
7133         \spath_split_path_at_intersections:cv
7134       }
7135     } {#1} {#2} { {} }
7136   }
7137   {
7138     \msg_warning:nn { spath3 } { load intersections }
7139   }
7140 },
7141 split~ globally~ at~ intersections~ with/.code~ n~ args={2}{
7142   \tl_if_exist:cTF
7143   {
7144     tikz@library@intersections@loaded
7145   }
7146   {
7147     \__tikzspath_maybe_current_two_paths_reuse_first:nmmm
7148     {
7149       \__tikzspath_check_two_paths:nmmm
7150       {
7151         \spath_gsplit_path_at_intersections:cv
7152       }
7153     } {#1} {#2} { {} }
7154   }
7155   {
7156     \msg_warning:nn { spath3 } { load intersections }
7157   }
7158 },

```

Splits two paths at their mutual intersections.

```

7159 split~ at~ intersections/.code~ n~ args={2}{
7160   \tl_if_exist:cTF
7161   {
7162     tikz@library@intersections@loaded
7163   }
7164   {
7165     \__tikzspath_maybe_current_two_paths_reuse_both:nmmm
7166     {
7167       \__tikzspath_check_two_paths:nmmm
7168       {
7169         \spath_split_at_intersections:cc
7170       }
7171     } {#1} {#2} { {} }
7172   }
7173   {
7174     \msg_warning:nn { spath3 } { load intersections }

```

```

7175     }
7176   },
7177   split~ globally~ at~ intersections/.code~ n~ args={2}{
7178     \tl_if_exist:cTF
7179     {
7180       tikz@library@intersections@loaded
7181     }
7182     {
7183       \__tikzspath_maybe_current_two_paths_reuse_both:nnnn
7184       {
7185         \__tikzspath_check_two_paths:nnnn
7186         {
7187           \spath_gsplit_at_intersections:cc
7188         }
7189       } {#1} {#2} { {} }
7190     }
7191     {
7192       \msg_warning:nn { spath3 } { load intersections }
7193     }
7194   },

```

Splits a path at its self-intersections.

```

7195   split~ at~ self~ intersections/.code={
7196     \tl_if_exist:cTF
7197     {
7198       tikz@library@intersections@loaded
7199     }
7200     {
7201       \__tikzspath_maybe_current_path_reuse:nnn
7202       {
7203         \__tikzspath_check_path:nnn
7204         {
7205           \spath_split_at_self_intersections:c
7206         }
7207       } {#1} { {} }
7208     }
7209     {
7210       \msg_warning:nn { spath3 } { load intersections }
7211     }
7212   },
7213   split~ globally~ at~ self~ intersections/.code={
7214     \tl_if_exist:cTF
7215     {
7216       tikz@library@intersections@loaded
7217     }
7218     {
7219       \__tikzspath_maybe_current_path_reuse:nnn
7220       {
7221         \__tikzspath_check_path:nnn
7222         {
7223           \spath_gsplit_at_self_intersections:c
7224         }
7225       } {#1} { {} }
7226     }

```

```

7227   {
7228     \msg_warning:nn { spath3 } { load intersections }
7229   }
7230 },

```

Extract the components of a path into a comma separated list (suitable for using in a `\foreach` loop).

```

7231 get~ components~ of/.code~ 2~ args={
7232   \__tikzspath_maybe_current_path:nn
7233   {
7234     \__tikzspath_check_path:nnn {
7235       \__tikzspath_get_components:Nv #2
7236     }
7237   }
7238   {#1}
7239   {}
7240 },
7241 get~ components~ of~ globally/.code~ 2~ args={
7242   \__tikzspath_maybe_current_path:nn
7243   {
7244     \__tikzspath_check_path:nnn {
7245       \__tikzspath_gget_components:Nv #2
7246     }
7247   }
7248   {#1}
7249   {}
7250 },

```

Loop through the components of a soft path and render each as a separate TikZ path so that they can be individually styled.

```

7251 render~ components/.code={
7252   \__tikzspath_maybe_current_path:nn
7253   {
7254     \__tikzspath_check_path:nnn {
7255       \__tikzspath_render_components:nv {#1}
7256     }
7257   }
7258   {#1}
7259   {}
7260 },

```

This puts gaps between components of a soft path. The list of components is passed through a `\foreach` loop so can use the shortcut syntax from those loops.

```

7261 insert~ gaps~ after~ components/.code~ 2~ args={
7262   \tl_if_head_is_group:nTF {#2}
7263   {
7264     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
7265     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
7266   }
7267   {
7268     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
7269     \tl_clear:N \l__tikzspath_tmpd_tl
7270   }
7271

```

```

7272 \tikzspath_maybe_current_path_reuse:nnn
7273 {
7274   \tikzspath_check_path:nnn
7275   {
7276     \tikzspath_insert_gaps_after_components:cVV
7277   }
7278 } {#1} { {} \l__tikzspath_tmpc_tl \l__tikzspath_tmpd_tl }
7279 },
7280 insert~ gaps~ globally~ after~ components/.code~ 2~ args={
7281   \tl_if_head_is_group:nTF {#2}
7282   {
7283     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
7284     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
7285   }
7286   {
7287     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
7288     \tl_clear:N \l__tikzspath_tmpd_tl
7289   }
7290 }
7291 \tikzspath_maybe_current_path_reuse:nnn
7292 {
7293   \tikzspath_check_path:nnn
7294   {
7295     \tikzspath_ginsert_gaps_after_components:cVV
7296   }
7297 } {#1} { {} \l__tikzspath_tmpc_tl \l__tikzspath_tmpd_tl }
7298 },

```

This puts gaps between segments of a soft path. The list of segments is passed through a `\foreach` loop so can use the shortcut syntax from those loops.

```

7299 insert~ gaps~ after~ segments/.code~ 2~ args={
7300   \tl_if_head_is_group:nTF {#2}
7301   {
7302     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
7303     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
7304   }
7305   {
7306     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
7307     \tl_clear:N \l__tikzspath_tmpd_tl
7308   }
7309 }
7310 \tikzspath_maybe_current_path_reuse:nnn
7311 {
7312   \tikzspath_check_path:nnn
7313   {
7314     \tikzspath_insert_gaps_after_segments:cVV
7315   }
7316 } {#1} { {} \l__tikzspath_tmpc_tl \l__tikzspath_tmpd_tl }
7317 },
7318 insert~ gaps~ globally~ after~ segments/.code~ 2~ args={
7319   \tl_if_head_is_group:nTF {#2}
7320   {
7321     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
7322     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }

```

```

7323     }
7324     {
7325     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
7326     \tl_clear:N \l__tikzspath_tmpd_tl
7327     }
7328
7329     \__tikzspath_maybe_current_path_reuse:nnn
7330     {
7331     \__tikzspath_check_path:nnn
7332     {
7333     \__tikzspath_ginsert_gaps_after_segments:cVV
7334     }
7335     } {#1} { {} \l__tikzspath_tmpc_tl \l__tikzspath_tmpd_tl }
7336 },

```

Join the specified components together, joining each to its previous one.

```

7337 join~ components/.code~ 2~ args={
7338     \__tikzspath_maybe_current_path_reuse:nnn
7339     {
7340     \__tikzspath_check_path:nnn
7341     {
7342     \__tikzspath_join_components:cn
7343     }
7344     } {#1} { {} {#2} }
7345 },
7346 join~ components~ globally/.code~ 2~ args={
7347     \__tikzspath_maybe_current_path_reuse:nnn
7348     {
7349     \__tikzspath_check_path:nnn
7350     {
7351     \__tikzspath_gjoin_components:cn
7352     }
7353     } {#1} { {} {#2} }
7354 },

```

Remove all components of the path that don't actually draw anything.

```

7355 remove~ empty~ components/.code={
7356     \__tikzspath_maybe_current_path_reuse:nnn
7357     {
7358     \__tikzspath_check_path:nnn
7359     {
7360     \spath_remove_empty_components:c
7361     }
7362     } {#1} { {} }
7363 },
7364 remove~ empty~ components~ globally/.code={
7365     \__tikzspath_maybe_current_path_reuse:nnn
7366     {
7367     \__tikzspath_check_path:nnn
7368     {
7369     \spath_gremove_empty_components:c
7370     }
7371     } {#1} { {} }
7372 },

```

Replace all line segments by Bézier curves.

```
7373 replace~ lines/.code={
7374   \__tikzspath_maybe_current_path_reuse:nnn
7375   {
7376     \__tikzspath_check_path:nnn
7377     {
7378       \spath_replace_lines:c
7379     }
7380   } {#1} { {} }
7381 },
7382 replace~ lines~ globally/.code={
7383   \__tikzspath_maybe_current_path_reuse:nnn
7384   {
7385     \__tikzspath_check_path:nnn
7386     {
7387       \spath_greplace_lines:c
7388     }
7389   } {#1} { {} }
7390 },
```

Remove the specified components.

```
7391 remove~ components/.code~ 2~ args={
7392   \__tikzspath_maybe_current_path_reuse:nnn
7393   {
7394     \__tikzspath_check_path:nnn
7395     {
7396       \__tikzspath_remove_components:cn
7397     }
7398   } {#1} { {} } {#2} }
7399 },
7400 remove~ components~ globally/.code~ 2~ args={
7401   \__tikzspath_maybe_current_path_reuse:nnn
7402   {
7403     \__tikzspath_check_path:nnn
7404     {
7405       \__tikzspath_gremove_components:cn
7406     }
7407   } {#1} { {} } {#2} }
7408 },
```

This puts a conditional around the `spot weld` key because when figuring out a knot drawing then we will initially want to render it without the spot weld to keep the number of components constant.

```
7409 draft~ mode/.is~ choice,
7410 draft~ mode/true/.code={
7411   \bool_set_true:N \l__tikzspath_draft_bool
7412 },
7413 draft~ mode/false/.code={
7414   \bool_set_false:N \l__tikzspath_draft_bool
7415 },
7416 maybe~ spot~ weld/.code={
7417   \bool_if:NF \l__tikzspath_draft_bool
7418   {
7419     \__tikzspath_maybe_current_path_reuse:nnn
```

```

7420     {
7421       \__tikzspath_check_path:nnn
7422       {
7423         \spath_spot_weld_components:c
7424       }
7425     } {#1} { {} }
7426   }
7427 },
7428 maybe~ spot~ weld~ globally/.code={
7429   \bool_if:NF \l__tikzspath_draft_bool
7430   {
7431     \__tikzspath_maybe_current_path_reuse:nnn
7432     {
7433       \__tikzspath_check_path:nnn
7434       {
7435         \spath_spot_gweld_components:c
7436       }
7437     } {#1} { {} }
7438   }
7439 },

```

Set the transformation to lie along a path.

```

7440 transform~ to/.code~ 2~ args={
7441   \__tikzspath_maybe_current_path:nn
7442   {
7443     \__tikzspath_check_path:nnn {
7444       \__tikzspath_transform_to:nv {#2}
7445     }
7446   }
7447   {#1}
7448   {
7449     \pgfsettransformentries {1}{0}{0}{1}{0pt}{0pt}
7450   }
7451 },

```

As above, but with a possible extra 180° rotation if needed to ensure that the new y -axis points vaguely upwards.

```

7452 upright~ transform~ to/.code~ 2~ args={
7453   \__tikzspath_maybe_current_path:nn
7454   {
7455     \__tikzspath_check_path:nnn {
7456       \__tikzspath_transform_upright_to:nv {#2}
7457     }
7458   }
7459   {#1}
7460   {
7461     \pgfsettransformentries {1}{0}{0}{1}{0pt}{0pt}
7462   }
7463 },

```

This is a useful set of styles for drawing a knot diagram.

```

7464 knot/.style~ n~ args={3}{
7465   /tikz/spath/split~ at~ self~ intersections=#1,
7466   /tikz/spath/remove~ empty~ components=#1,

```

```

7467 /tikz/spath/insert~ gaps~ after~ components={#1}{#2}{#3},
7468 /tikz/spath/maybe~ spot~ weld=#1,
7469 /tikz/spath/render~ components=#1
7470 },
7471 global~ knot/.style~ n~ args={3}{
7472 /tikz/spath/split~ globally~ at~ self~ intersections=#1,
7473 /tikz/spath/remove~ empty~ components~ globally=#1,
7474 /tikz/spath/insert~ gaps~ globally~ after~ components={#1}{#2}{#3},
7475 /tikz/spath/maybe~ spot~ weld~ globally=#1,
7476 /tikz/spath/render~ components=#1
7477 },
7478 }

```

This defines a coordinate system that finds a position on a soft path.

```

7479 \cs_new_protected_nopar:Npn \__tikzspath_get_point_at:nn #1#2
7480 {
7481   \group_begin:
7482   \spath_reallength:Nn \l__tikzspath_tmpa_int {#2}
7483   \tl_set:Nx \l__tikzspath_tmpb_tl
7484   {\fp_to_decimal:n {(#1) * (\l__tikzspath_tmpa_int)}}
7485   \spath_point_at:NnV \l__tikzspath_tmpc_tl {#2} \l__tikzspath_tmpb_tl
7486
7487   \tl_clear:N \l__tikzspath_tmpd_tl
7488   \tl_put_right:Nn \l__tikzspath_tmpd_tl {\pgf@x=}
7489   \tl_put_right:Nx \l__tikzspath_tmpd_tl {\tl_item:Nn \l__tikzspath_tmpc_tl {1}}
7490   \tl_put_right:Nn \l__tikzspath_tmpd_tl {\relax}
7491   \tl_put_right:Nn \l__tikzspath_tmpd_tl {\pgf@y=}
7492   \tl_put_right:Nx \l__tikzspath_tmpd_tl {\tl_item:Nn \l__tikzspath_tmpc_tl {2}}
7493   \tl_put_right:Nn \l__tikzspath_tmpd_tl {\relax}
7494
7495   \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpd_tl
7496   \group_end:
7497 }
7498 \cs_generate_variant:Nn \__tikzspath_get_point_at:nn {VV, Vn, Vv}
7499
7500 \tikzdeclarecoordinatesystem{spath}{%
7501   \group_begin:
7502   \tl_set:Nn \l__tikzspath_tmpa_tl {#1}
7503   \tl_trim_spaces:N \l__tikzspath_tmpa_tl
7504
7505   \seq_set_split:NnV \l__tikzspath_tmpa_seq {-} \l__tikzspath_tmpa_tl
7506   \seq_pop_right:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpb_tl
7507
7508   \tl_set:Nx \l__tikzspath_tmpa_tl { \seq_use:Nn \l__tikzspath_tmpa_seq {-} }
7509
7510   \__tikzspath_maybe_current_path:nV
7511   {
7512     \__tikzspath_check_path:nnn {
7513       \__tikzspath_get_point_at:Vv \l__tikzspath_tmpb_tl
7514     }
7515   }
7516   \l__tikzspath_tmpa_tl
7517   {
7518     \tl_gset_eq:NN \g__tikzspath_output_tl \pgfpointorigin

```



```

7519 }
7520
7521 \group_end:
7522 \use:c {pgf@process}{%
7523   \tl_use:N \g__tikzspath_output_tl
7524   \pgftransforminvert
7525   \use:c {pgf@pos@transform@glob}
7526 }
7527 \tl_gclear:N \g__tikzspath_output_tl
7528 }
7529
7530 \ExplSyntaxOff

```

5 The Calligraphy Package

```

7531 <@=@=cal>

```

5.1 Initialisation

```

7532 \RequirePackage{spath3}
7533 \ExplSyntaxOn
7534
7535 \tl_new:N \l__cal_tmpa_tl
7536 \tl_new:N \l__cal_tmpb_tl
7537 \tl_new:N \l__cal_tmp_path_tl
7538 \tl_new:N \l__cal_tmp_rpath_tl
7539 \tl_new:N \l__cal_tmp_rpathb_tl
7540 \tl_new:N \l__cal_tmp_patha_tl
7541
7542 \seq_new:N \l__cal_tmpa_seq
7543
7544 \int_new:N \l__cal_tmpa_int
7545 \int_new:N \l__cal_tmpb_int
7546 \int_new:N \g__cal_path_component_int
7547 \int_new:N \g__cal_label_int
7548
7549 \fp_new:N \l__cal_tmpa_fp
7550 \fp_new:N \l__cal_tmpb_fp
7551 \fp_new:N \l__cal_tmpe_fp
7552 \fp_new:N \l__cal_tmpe_fp
7553 \fp_new:N \l__cal_tmpe_fp
7554
7555 \dim_new:N \l__cal_tmpa_dim
7556 \dim_new:N \l__cal_tmpb_dim
7557 \dim_new:N \l__cal_tmpe_dim
7558 \dim_new:N \l__cal_tmpe_dim
7559 \dim_new:N \l__cal_tmpe_dim
7560 \dim_new:N \l__cal_tmpe_dim
7561 \dim_new:N \l__cal_tmpe_dim
7562 \dim_new:N \l__cal_tmpe_dim
7563
7564 \bool_new:N \l__cal_annotate_bool
7565 \bool_new:N \l__cal_taper_start_bool
7566 \bool_new:N \l__cal_taper_end_bool

```

```

7567 \bool_new:N \l__cal_taperable_bool
7568
7569 \dim_new:N \l__cal_taper_width_dim
7570 \dim_new:N \l__cal_line_width_dim
7571
7572 \bool_set_true:N \l__cal_taper_start_bool
7573 \bool_set_true:N \l__cal_taper_end_bool
7574
7575 \cs_generate_variant:Nn \tl_put_right:Nn {Nv}
7576
7577 \msg_new:nnn { calligraphy } { undefined pen } { The~ pen~ "#1"~ is~ not~ defined. }

```

5.2 TikZ Keys

The public interface to this package is through TikZ keys and styles.

```

7578 \tikzset{
7579   define-pen/.code={
7580     \tikzset{pen-name=#1}
7581     \pgf@relevantforpicturesizefalse
7582     \tikz@addmode{
7583       \pgfsyssoftpath@getcurrentpath\l__cal_tmpa_tl
7584       \spath_components_to_seq:Nv \l__cal_tmpa_seq \l__cal_tmpa_tl
7585       \seq_gclear_new:c {g__cal_pen_\pgfkeysvalueof{/tikz/pen-name}_seq}
7586       \seq_gset_eq:cN
7587         {g__cal_pen_\pgfkeysvalueof{/tikz/pen-name}_seq} \l__cal_tmpa_seq
7588       \pgfusepath{discard}%
7589     }
7590   },
7591   define-pen/.default={default},
7592   use-pen/.code={
7593     \tikzset{pen-name=#1}
7594     \int_gzero:N \g__cal_path_component_int
7595     \cs_set_eq:NN \pgfpathmoveto \cal_moveto:n
7596     \tikz@addmode{
7597       \pgfsyssoftpath@getcurrentpath\l__cal_tmpa_tl
7598       \spath_components_to_seq:Nv \l__cal_tmpa_seq \l__cal_tmpa_tl
7599       \tl_if_exist:cTF {g__cal_pen_\pgfkeysvalueof{/tikz/pen-name}_seq}
7600       {
7601         \cal_path_create:Nc \l__cal_tmpa_seq
7602         {g__cal_pen_\pgfkeysvalueof{/tikz/pen-name}_seq}
7603       }
7604       {
7605         \msg_warning:nx { calligraphy } { undefined pen }
7606         { \pgfkeysvalueof{/tikz/pen-name} }
7607       }
7608     }
7609   },
7610   use-pen/.default={default},
7611   pen-name/.initial={default},
7612   copperplate/.style={pen-name=copperplate},
7613   pen-colour/.initial={black},
7614   weight/.is-choice,
7615   weight/heavy/.style={
7616     line-width=\pgfkeysvalueof{/tikz/heavy-line-width},

```

```

7617     taper~width=\pgfkeysvalueof{/tikz/light~line~width},
7618   },
7619   weight/light/.style={
7620     line~width=\pgfkeysvalueof{/tikz/light~line~width},
7621     taper~width=0pt,
7622   },
7623   heavy/.style={
7624     weight=heavy
7625   },
7626   light/.style={
7627     weight=light
7628   },
7629   heavy~line~width/.initial=2pt,
7630   light~line~width/.initial=1pt,
7631   taper/.is~choice,
7632   taper/.default=both,
7633   taper/none/.style={
7634     taper~start=false,
7635     taper~end=false,
7636   },
7637   taper/both/.style={
7638     taper~start=true,
7639     taper~end=true,
7640   },
7641   taper/start/.style={
7642     taper~start=true,
7643     taper~end=false,
7644   },
7645   taper/end/.style={
7646     taper~start=false,
7647     taper~end=true,
7648   },
7649   taper~start/.code={
7650     \tl_if_eq:nnTF {#1} {true}
7651     {
7652       \bool_set_true:N \l__cal_taper_start_bool
7653     }
7654     {
7655       \bool_set_false:N \l__cal_taper_start_bool
7656     }
7657   },
7658   taper~start/.default={true},
7659   taper~end/.code={
7660     \tl_if_eq:nnTF {#1} {true}
7661     {
7662       \bool_set_true:N \l__cal_taper_end_bool
7663     }
7664     {
7665       \bool_set_false:N \l__cal_taper_end_bool
7666     }
7667   },
7668   taper~end/.default={true},
7669   taper~width/.code={\dim_set:Nn \l__cal_taper_width_dim {#1}},
7670   nib~style/.code~2~args={

```

```

7671 \tl_clear_new:c {l__cal_nib_style_#1}
7672 \tl_set:cn {l__cal_nib_style_#1} {#2}
7673 },
7674 stroke~style/.code~2~args={
7675 \tl_clear_new:c {l__cal_stroke_style_#1}
7676 \tl_set:cn {l__cal_stroke_style_#1} {#2}
7677 },
7678 this~stroke~style/.code={
7679 \tl_clear_new:c
7680 {l__cal_stroke_inline_style_ \int_use:N \g__cal_path_component_int}
7681 \tl_set:cn
7682 {l__cal_stroke_inline_style_ \int_use:N \g__cal_path_component_int} {#1}
7683 },
7684 annotate/.style={
7685 annotate~if,
7686 annotate~reset,
7687 annotation~style/.update~value={#1},
7688 },
7689 annotate~if/.default={true},
7690 annotate~if/.code={
7691 \tl_if_eq:nnTF {#1} {true}
7692 {
7693 \bool_set_true:N \l__cal_annotate_bool
7694 }
7695 {
7696 \bool_set_false:N \l__cal_annotate_bool
7697 }
7698 },
7699 annotate~reset/.code={
7700 \int_gzero:N \g__cal_label_int
7701 },
7702 annotation~style/.initial={draw,->},
7703 annotation~shift/.initial={{0,lex}},
7704 every~annotation~node/.initial={anchor=south-west},
7705 annotation~node~style/.code~2~args={
7706 \tl_clear_new:c {l__cal_annotation_style_ #1 _tl}
7707 \tl_set:cn {l__cal_annotation_style_ #1 _tl}{#2}
7708 },
7709 tl~use:N/.code={
7710 \exp_args:Nv \pgfkeysalso #1
7711 },
7712 tl~use:c/.code={
7713 \tl_if_exist:cT {#1}
7714 {
7715 \exp_args:Nv \pgfkeysalso {#1}
7716 }
7717 },
7718 /handlers/.update~style/.code={
7719 \tl_if_eq:nnF {#1} {\pgfkeysnovalue}
7720 {
7721 \pgfkeys{\pgfkeyscurrentpath/.code=\pgfkeysalso{#1}}
7722 }
7723 },
7724 /handlers/.update~value/.code={

```

```

7725 \tl_if_eq:nnF {#1} {\pgfkeysnovalue}
7726 {
7727   \pgfkeyssetvalue{\pgfkeyscurrentpath}{#1}
7728 }
7729 },
7730 }

```

Some wrappers around the TikZ keys.

```

7731 \NewDocumentCommand \pen { 0{} }
7732 {
7733   \path[define~ pen,every~ calligraphy~ pen/.try,#1]
7734 }
7735
7736 \NewDocumentCommand \definepen { 0{} }
7737 {
7738   \tikz \path[define~ pen,every~ calligraphy~ pen/.try,#1]
7739 }
7740
7741 \NewDocumentCommand \calligraphy { 0{} }
7742 {
7743   \path[use~ pen,every~ calligraphy/.try,#1]
7744 }

```

5.3 The Path Creation

`\cal_path_create:NN` This is the main command for creating the calligraphic paths. First argument is the given path Second argument is the pen path

```

7745 \cs_new_protected_nopar:Npn \cal_path_create:NN #1#2
7746 {
7747   \int_zero:N \l__cal_tmpa_int
7748   \seq_map_inline:Nn #1
7749   {
7750     \int_compare:nT {\tl_count:n {##1} > 3}
7751     {
7752
7753       \int_incr:N \l__cal_tmpa_int
7754       \int_zero:N \l__cal_tmpb_int
7755
7756       \tl_set:Nn \l__cal_tmp_path_tl {##1}
7757       \spath_open:N \l__cal_tmp_path_tl
7758       \spath_reverse:NV \l__cal_tmp_rpath_tl \l__cal_tmp_path_tl
7759
7760       \seq_map_inline:Nn #2
7761       {
7762         \int_incr:N \l__cal_tmpb_int
7763         \group_begin:
7764         \pgfsys@beginscope
7765         \cal_apply_style:c {l__cal_stroke_style_ \int_use:N \l__cal_tmpa_int}
7766         \cal_apply_style:c {l__cal_stroke_inline_style_ \int_use:N \l__cal_tmpa_int}
7767         \cal_apply_style:c {l__cal_nib_style_ \int_use:N \l__cal_tmpb_int}
7768
7769         \spath_initialpoint:Nn \l__cal_tmpa_tl {###1}
7770         \tl_set_eq:NN \l__cal_tmp_patha_tl \l__cal_tmp_path_tl
7771         \spath_translate:NV \l__cal_tmp_patha_tl \l__cal_tmpa_tl

```

```

7772
7773 \int_compare:nTF {\tl_count:n {####1} == 3}
7774 {
7775   \cal_at_least_three:N \l__cal_tmp_patha_tl
7776   \spath_protocol_path:V \l__cal_tmp_patha_tl
7777
7778   \tikz@options
7779   \dim_set:Nn \l__cal_line_width_dim {\pgflinewidth}
7780   \cal_maybe_taper:N \l__cal_tmp_patha_tl
7781 }
7782 {
7783   \spath_weld:Nn \l__cal_tmp_patha_tl {####1}
7784   \spath_weld:NV \l__cal_tmp_patha_tl \l__cal_tmp_rpath_tl
7785   \spath_reverse:Nn \l__cal_tmp_rpathb_tl {####1}
7786   \spath_weld:NV \l__cal_tmp_patha_tl \l__cal_tmp_rpathb_tl
7787
7788   \tl_clear:N \l__cal_tmpa_tl
7789   \tl_set:Nn \l__cal_tmpa_tl
7790   {
7791     fill=\pgfkeysvalueof{/tikz/pen-colour},
7792     draw=none
7793   }
7794   \tl_if_exist:cT {l__cal_stroke_style_ \int_use:N \l__cal_tmpa_int}
7795   {
7796     \tl_put_right:Nv \l__cal_tmpa_tl
7797     {l__cal_stroke_style_ \int_use:N \l__cal_tmpa_int}
7798   }
7799   \tl_if_exist:cT {l__cal_stroke_inline_style_ \int_use:N \l__cal_tmpa_int}
7800   {
7801     \tl_put_right:Nn \l__cal_tmpa_tl {,}
7802     \tl_put_right:Nv \l__cal_tmpa_tl
7803     {l__cal_stroke_inline_style_ \int_use:N \l__cal_tmpa_int}
7804   }
7805   \tl_if_exist:cT {l__cal_nib_style_ \int_use:N \l__cal_tmpb_int}
7806   {
7807     \tl_put_right:Nn \l__cal_tmpa_tl {,}
7808     \tl_put_right:Nv \l__cal_tmpa_tl
7809     {l__cal_nib_style_ \int_use:N \l__cal_tmpb_int}
7810   }
7811   \spath_tikz_path:VV \l__cal_tmpa_tl \l__cal_tmp_patha_tl
7812 }
7813 \pgfsys@endscope
7814 \group_end:
7815 }
7816
7817 \bool_if:NT \l__cal_annotate_bool
7818 {
7819   \seq_get_right:NN #2 \l__cal_tmpa_tl
7820   \spath_finalpoint:NV \l__cal_tmpa_tl \l__cal_tmpa_tl
7821   \spath_translate:NV \l__cal_tmp_patha_tl \l__cal_tmpa_tl
7822   \tikz@scan@one@point
7823   \pgfutil@firstofone
7824   \pgfkeysvalueof{/tikz/annotation-shift}
7825

```

```

7826     \spath_translate:Nnn \l__cal_tmp_path_tl {\pgf@x} {\pgf@y}
7827
7828     \pgfkeysgetvalue{/tikz/annotation~style}{\l__cal_tmpa_tl}
7829     \spath_tikz_path:VV \l__cal_tmpa_tl \l__cal_tmp_path_tl
7830
7831     \spath_finalpoint:NV \l__cal_tmpa_tl \l__cal_tmp_path_tl
7832
7833     \exp_last_unbraced:NV \pgfqpoint \l__cal_tmpa_tl
7834     \begin{scope}[reset~ cm]
7835     \node[
7836         every~annotation~node/.try,
7837         tl~use:c = {\l__cal_annotation_style_ \int_use:N \l__cal_tmpa_int _tl}
7838     ] at (\pgf@x,\pgf@y) {\int_use:N \l__cal_tmpa_int};
7839     \end{scope}
7840     }
7841 }
7842 }
7843 }
7844 \cs_generate_variant:Nn \cal_path_create:NN {Nc}

```

(End definition for `\cal_path_create:NN`.)

`\cal_moveto:n` When creating the path, we need to keep track of the number of components so that we can apply styles accordingly.

```

7845 \cs_new_eq:NN \cal_orig_moveto:n \pgfpathmoveto
7846 \cs_new_nopar:Npn \cal_moveto:n #1
7847 {
7848     \int_gincr:N \g__cal_path_component_int
7849     \cal_orig_moveto:n {#1}
7850 }

```

(End definition for `\cal_moveto:n`.)

`\cal_apply_style:N` Interface for applying `\tikzset` to a token list.

```

7851 \cs_new_nopar:Npn \cal_apply_style:N #1
7852 {
7853     \tl_if_exist:NT #1 {
7854         \exp_args:NV \tikzset #1
7855     }
7856 }
7857 \cs_generate_variant:Nn \cal_apply_style:N {c}

```

(End definition for `\cal_apply_style:N`.)

`\cal_at_least_three:Nn` A tapered path has to have at least three components. This figures out if it is necessary and sets up the splitting.

```

7858 \cs_new_protected_nopar:Npn \cal_at_least_three:Nn #1#2
7859 {
7860     \spath_reallength:Nn \l__cal_tmpa_int {#2}
7861     \tl_clear:N \l__cal_tmpb_tl
7862     \tl_set:Nn \l__cal_tmpb_tl {#2}
7863     \int_compare:nTF {\l__cal_tmpa_int = 1}
7864     {
7865         \spath_split_at:Nn \l__cal_tmpb_tl {2/3}
7866         \spath_split_at:Nn \l__cal_tmpb_tl {1/2}

```

```

7867 }
7868 {
7869   \int_compare:nT {\l__cal_tmpa_int = 2}
7870   {
7871     \spath_split_at:Nn \l__cal_tmpb_tl {1.5}
7872     \spath_split_at:Nn \l__cal_tmpb_tl {.5}
7873   }
7874 }
7875 \tl_set_eq:NN #1 \l__cal_tmpb_tl
7876 }
7877 \cs_generate_variant:Nn \cal_at_least_three:Nn {NV}
7878 \cs_new_protected_nopar:Npn \cal_at_least_three:N #1
7879 {
7880   \cal_at_least_three:NV #1#1
7881 }
7882 \cs_generate_variant:Nn \cal_at_least_three:N {c}

```

(End definition for \cal_at_least_three:Nn.)

\cal_maybe_taper:N Possibly tapers the path, depending on the booleans.

```

7883 \cs_new_protected_nopar:Npn \cal_maybe_taper:N #1
7884 {
7885   \tl_set_eq:NN \l__cal_tmpa_tl #1
7886
7887   \bool_if:NT \l__cal_taper_start_bool
7888   {
7889
7890     \dim_set:Nn \l__cal_tmpa_dim {\tl_item:Nn \l__cal_tmpa_tl {2}}
7891     \dim_set:Nn \l__cal_tmpb_dim {\tl_item:Nn \l__cal_tmpa_tl {3}}
7892     \tl_set:Nx \l__cal_tmpb_tl {\tl_item:Nn \l__cal_tmpa_tl {4}}
7893
7894     \tl_case:NnF \l__cal_tmpb_tl
7895     {
7896       \c_spath_lineto_tl
7897       {
7898
7899         \bool_set_true:N \l__cal_taperable_bool
7900         \dim_set:Nn \l__cal_tmpg_dim {\tl_item:Nn \l__cal_tmpa_tl {5}}
7901         \dim_set:Nn \l__cal_tmph_dim {\tl_item:Nn \l__cal_tmpa_tl {6}}
7902         \dim_set:Nn \l__cal_tmpe_dim {(2\l__cal_tmpa_dim + \l__cal_tmph_dim)/3}
7903         \dim_set:Nn \l__cal_tmpe_dim {(2\l__cal_tmph_dim + \l__cal_tmpe_dim)/3}
7904         \dim_set:Nn \l__cal_tmpe_dim {(\l__cal_tmpa_dim + 2\l__cal_tmpe_dim)/3}
7905         \dim_set:Nn \l__cal_tmpe_dim {(\l__cal_tmpe_dim + 2\l__cal_tmph_dim)/3}
7906         \prg_replicate:nn {4}
7907         {
7908           \tl_set:Nx \l__cal_tmpa_tl {\tl_tail:N \l__cal_tmpa_tl}
7909         }
7910         \tl_put_left:NV \l__cal_tmpa_tl \c_spath_moveto_tl
7911       }
7912     }
7913     \c_spath_curvetoa_tl
7914     {
7915       \bool_set_true:N \l__cal_taperable_bool
7916       \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {5}}
7917       \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {6}}

```



```

7917     \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {8}}
7918     \dim_set:Nn \l__cal_tmpf_dim {\tl_item:Nn \l__cal_tmpa_tl {9}}
7919     \dim_set:Nn \l__cal_tmpg_dim {\tl_item:Nn \l__cal_tmpa_tl {11}}
7920     \dim_set:Nn \l__cal_tmph_dim {\tl_item:Nn \l__cal_tmpa_tl {12}}
7921     \prg_replicate:nn {10}
7922     {
7923         \tl_set:Nx \l__cal_tmpa_tl {\tl_tail:N \l__cal_tmpa_tl}
7924     }
7925     \tl_put_left:NV \l__cal_tmpa_tl \c_spath_moveto_tl
7926 }
7927 }
7928 {
7929     \bool_set_false:N \l__cal_taperable_bool
7930 }
7931
7932 \bool_if:NT \l__cal_taperable_bool
7933 {
7934     \__cal_taper_aux:
7935 }
7936
7937 }
7938
7939 \bool_if:NT \l__cal_taper_end_bool
7940 {
7941
7942     \dim_set:Nn \l__cal_tmpa_dim {\tl_item:Nn \l__cal_tmpa_tl {-2}}
7943     \dim_set:Nn \l__cal_tmpb_dim {\tl_item:Nn \l__cal_tmpa_tl {-1}}
7944     \tl_set:Nx \l__cal_tmpb_tl {\tl_item:Nn \l__cal_tmpa_tl {-3}}
7945
7946     \tl_case:NnF \l__cal_tmpb_tl
7947     {
7948         \c_spath_lineto_tl
7949     {
7950
7951         \bool_set_true:N \l__cal_taperable_bool
7952         \dim_set:Nn \l__cal_tmpg_dim {\tl_item:Nn \l__cal_tmpa_tl {-5}}
7953         \dim_set:Nn \l__cal_tmph_dim {\tl_item:Nn \l__cal_tmpa_tl {-4}}
7954         \dim_set:Nn \l__cal_tmpe_dim {(2\l__cal_tmpa_dim + \l__cal_tmpe_dim)/3}
7955         \dim_set:Nn \l__cal_tmpe_dim {(2\l__cal_tmpb_dim + \l__cal_tmph_dim)/3}
7956         \dim_set:Nn \l__cal_tmpe_dim {(\l__cal_tmpa_dim + 2\l__cal_tmpe_dim)/3}
7957         \dim_set:Nn \l__cal_tmpe_dim {(\l__cal_tmpe_dim + 2\l__cal_tmph_dim)/3}
7958         \tl_reverse:N \l__cal_tmpa_tl
7959         \prg_replicate:nn {3}
7960         {
7961             \tl_set:Nx \l__cal_tmpa_tl {\tl_tail:N \l__cal_tmpa_tl}
7962         }
7963         \tl_reverse:N \l__cal_tmpa_tl
7964     }
7965     \c_spath_curveto_tl
7966     {
7967         \bool_set_true:N \l__cal_taperable_bool
7968         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-5}}
7969         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-4}}
7970         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-8}}

```

```

7971     \dim_set:Nn \l__cal_tmpf_dim {\tl_item:Nn \l__cal_tmpa_tl {-7}}
7972     \dim_set:Nn \l__cal_tmpg_dim {\tl_item:Nn \l__cal_tmpa_tl {-11}}
7973     \dim_set:Nn \l__cal_tmph_dim {\tl_item:Nn \l__cal_tmpa_tl {-10}}
7974     \tl_reverse:N \l__cal_tmpa_tl
7975     \prg_replicate:nn {9}
7976     {
7977         \tl_set:Nx \l__cal_tmpa_tl {\tl_tail:N \l__cal_tmpa_tl}
7978     }
7979     \tl_reverse:N \l__cal_tmpa_tl
7980 }
7981 }
7982 {
7983     \bool_set_false:N \l__cal_taperable_bool
7984 }
7985
7986 \bool_if:NT \l__cal_taperable_bool
7987 {
7988     \__cal_taper_aux:
7989 }
7990
7991 }
7992
7993 \pgfsyssoftpath@setcurrentpath\l__cal_tmpa_tl
7994 \pgfsetstrokecolor{\pgfkeysvalueof{/tikz/pen~colour}}
7995 \pgfusepath{stroke}
7996
7997 }

```

(End definition for \cal_maybe_taper:N.)

`__cal_taper_aux:` Auxiliary macro to avoid unnecessary code duplication.

```

7998 \cs_new_protected_nopar:Npn \__cal_taper_aux:
7999 {
8000     \tl_clear:N \l__cal_tmpe_tl
8001     \tl_put_right:NV \l__cal_tmpe_tl \c_spath_moveto_tl
8002
8003     \fp_set:Nn \l__cal_tmpe_fp
8004     {
8005         \l__cal_tmpe_dim - \l__cal_tmpe_dim
8006     }
8007     \fp_set:Nn \l__cal_tmpe_fp
8008     {
8009         \l__cal_tmpe_dim - \l__cal_tmpe_dim
8010     }
8011     \fp_set:Nn \l__cal_tmpe_fp
8012     {
8013         (\l__cal_tmpe_fp^2 + \l__cal_tmpe_fp^2)^.5
8014     }
8015
8016     \fp_set:Nn \l__cal_tmpe_fp
8017     {
8018         .5*\l__cal_taper_width_dim
8019         *
8020         \l__cal_tmpe_fp / \l__cal_tmpe_fp

```

```

8021 }
8022 \fp_set:Nn \l__cal_tmpb_fp
8023 {
8024   .5*\l__cal_taper_width_dim
8025   *
8026   \l__cal_tmpb_fp / \l__cal_tmpe_fp
8027 }
8028
8029 \fp_set:Nn \l__cal_tmpe_fp
8030 {
8031   \l__cal_tmpe_dim - \l__cal_tmpe_dim
8032 }
8033 \fp_set:Nn \l__cal_tmpe_dim
8034 {
8035   \l__cal_tmpe_dim - \l__cal_tmpe_dim
8036 }
8037 \fp_set:Nn \l__cal_tmpe_dim
8038 {
8039   (\l__cal_tmpe_dim^2 + \l__cal_tmpe_dim^2)^.5
8040 }
8041
8042 \fp_set:Nn \l__cal_tmpe_dim
8043 {
8044   .5*\l__cal_line_width_dim
8045   *
8046   \l__cal_tmpe_dim / \l__cal_tmpe_dim
8047 }
8048 \fp_set:Nn \l__cal_tmpe_dim
8049 {
8050   .5*\l__cal_line_width_dim
8051   *
8052   \l__cal_tmpe_dim / \l__cal_tmpe_dim
8053 }
8054
8055 \tl_put_right:Nx \l__cal_tmpe_dim
8056 {
8057   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8058   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8059 }
8060
8061 \tl_put_right:NV \l__cal_tmpe_dim \c_spath_curvetoa_tl
8062
8063 \tl_put_right:Nx \l__cal_tmpe_dim
8064 {
8065   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8066   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8067 }
8068
8069 \tl_put_right:NV \l__cal_tmpe_dim \c_spath_curvetob_tl
8070
8071 \tl_put_right:Nx \l__cal_tmpe_dim
8072 {
8073   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8074   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}

```

```

8075 }
8076
8077 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curveto_tl
8078
8079 \tl_put_right:Nx \l__cal_tmpb_tl
8080 {
8081   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim}}
8082   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim}}
8083 }
8084
8085 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curvetoa_tl
8086
8087 \tl_put_right:Nx \l__cal_tmpb_tl
8088 {
8089   {
8090     \dim_eval:n
8091     {
8092       \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim
8093       - \fp_to_dim:n{ 1.32 * \l__cal_tmpe_fp
8094     }
8095   }
8096 }
8097 {
8098   \dim_eval:n
8099   {
8100     \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim
8101     + \fp_to_dim:n {1.32* \l__cal_tmpe_fp
8102   }
8103 }
8104 }
8105 }
8106
8107 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curvetob_tl
8108
8109 \tl_put_right:Nx \l__cal_tmpb_tl
8110 {
8111   {
8112     \dim_eval:n
8113     {
8114       -\fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim
8115       - \fp_to_dim:n {1.32 * \l__cal_tmpe_fp
8116     }
8117   }
8118 }
8119 {
8120   \dim_eval:n
8121   {
8122     -\fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim
8123     + \fp_to_dim:n {1.32 * \l__cal_tmpe_fp
8124   }
8125 }
8126 }
8127 }
8128

```

```

8129 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curveto_tl
8130
8131 \tl_put_right:Nx \l__cal_tmpb_tl
8132 {
8133   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpc_fp + \l__cal_tmpg_dim}}
8134   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpd_fp + \l__cal_tmph_dim}}
8135 }
8136
8137 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curvetoa_tl
8138
8139 \tl_put_right:Nx \l__cal_tmpb_tl
8140 {
8141   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpc_fp + \l__cal_tmpe_dim}}
8142   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpd_fp + \l__cal_tmpf_dim}}
8143 }
8144
8145 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curvetob_tl
8146
8147 \tl_put_right:Nx \l__cal_tmpb_tl
8148 {
8149   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8150   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8151 }
8152
8153 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curveto_tl
8154
8155 \tl_put_right:Nx \l__cal_tmpb_tl
8156 {
8157   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8158   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8159 }
8160
8161 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curvetoa_tl
8162
8163 \tl_put_right:Nx \l__cal_tmpb_tl
8164 {
8165   {
8166     \dim_eval:n
8167     {
8168       -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim
8169       + \fp_to_dim:n{ 1.32 * \l__cal_tmpe_dim}
8170     }
8171   }
8172   {
8173     \dim_eval:n
8174     {
8175       -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim
8176       - \fp_to_dim:n {1.32* \l__cal_tmpe_dim}
8177     }
8178   }
8179 }
8180
8181 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curvetob_tl
8182

```

```

8183 \tl_put_right:Nx \l__cal_tmpb_tl
8184 {
8185   {
8186     \dim_eval:n
8187     {
8188       \fp_to_dim:N \l__cal_tmpa_fp + \l__cal_tmpa_dim
8189       + \fp_to_dim:n {1.32 * \l__cal_tmpb_fp}
8190     }
8191   }
8192   {
8193     \dim_eval:n
8194     {
8195       \fp_to_dim:N \l__cal_tmpb_fp + \l__cal_tmpb_dim
8196       - \fp_to_dim:n {1.32 * \l__cal_tmpa_fp}
8197     }
8198   }
8199 }
8200
8201 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curveto_tl
8202
8203 \tl_put_right:Nx \l__cal_tmpb_tl
8204 {
8205   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpa_fp + \l__cal_tmpa_dim}}
8206   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpb_fp + \l__cal_tmpb_dim}}
8207 }
8208
8209 \pgfsyssoftpath@setcurrentpath\l__cal_tmpb_tl
8210 \pgfsetfillcolor{\pgfkeysvalueof{/tikz/pen~colour}}
8211 \pgfusepath{fill}
8212 }

```

(End definition for `_cal_taper_aux:.`)

Defines a copperplate pen.

```

8213 \tl_set:Nn \l__cal_tmpa_tl {\pgfsyssoftpath@movetotoken{0pt}{0pt}}
8214 \spath_components_to_seq:NV \l__cal_tmpa_seq \l__cal_tmpa_tl
8215 \seq_gclear_new:N \g__cal_pen_copperplate_seq
8216 \seq_gset_eq:NN \g__cal_pen_copperplate_seq \l__cal_tmpa_seq

```

`\CopperplatePath` This is used in the decorations section to convert a path to a copperplate path.

```

8217 \DeclareDocumentCommand \CopperplatePath { m }
8218 {
8219   \spath_components_to_seq:NV \l__cal_tmpa_seq #1
8220   \cal_path_create:NN \l__cal_tmpa_seq \g__cal_pen_copperplate_seq
8221 }

```

(End definition for `\CopperplatePath`.)

```
8222 \ExplSyntaxOff
```

5.4 Decorations

If a decoration library is loaded we define some decorations that use the calligraphy library, specifically the copperplate pen with its tapering.

First, a brace decoration.

```

8223 \expandafter\ifx\cname pgfdeclaredecoration\endcname\relax
8224 \else
8225 \pgfdeclaredecoration{calligraphic brace}{brace}%
8226 {%
8227   \state{brace}[width=+\pgfdecoratedremainingdistance,next state=final]%
8228   {%
8229     \pgfsyssoftpath@setcurrentpath{\pgfutil@empty}%
8230     \pgfpathmoveto{\pgfpointorigin}%
8231     \pgfpathcurveto%
8232     {%
8233       \pgfqpoint%
8234       {.15\pgfdecorationsegmentamplitude}%
8235       {.3\pgfdecorationsegmentamplitude}%
8236     }%
8237     {%
8238       \pgfqpoint%
8239       {.5\pgfdecorationsegmentamplitude}%
8240       {.5\pgfdecorationsegmentamplitude}%
8241     }%
8242     {%
8243       \pgfqpoint%
8244       {\pgfdecorationsegmentamplitude}%
8245       {.5\pgfdecorationsegmentamplitude}%
8246     }%
8247     {%
8248       \pgftransformxshift%
8249       {+\pgfdecorationsegmentaspect\pgfdecoratedremainingdistance}%
8250       \pgfpathlineto%
8251       {%
8252         \pgfqpoint%
8253         {-\pgfdecorationsegmentamplitude}%
8254         {.5\pgfdecorationsegmentamplitude}%
8255       }%
8256       \pgfpathcurveto
8257       {%
8258         \pgfqpoint%
8259         {- .5\pgfdecorationsegmentamplitude}%
8260         {.5\pgfdecorationsegmentamplitude}%
8261       }%
8262       {%
8263         \pgfqpoint%
8264         {- .15\pgfdecorationsegmentamplitude}%
8265         {.7\pgfdecorationsegmentamplitude}%
8266       }%
8267       {%
8268         \pgfqpoint%
8269         {0\pgfdecorationsegmentamplitude}%
8270         {1\pgfdecorationsegmentamplitude}%
8271       }%
8272       \pgfpathmoveto%
8273       {%
8274         \pgfqpoint%
8275         {0\pgfdecorationsegmentamplitude}%
8276         {1\pgfdecorationsegmentamplitude}%

```

```

8277 }%
8278 \pgfpathcurveto%
8279 {%
8280 \pgfqpoint%
8281 {.15\pgfdecorationsegmentamplitude}%
8282 {.7\pgfdecorationsegmentamplitude}%
8283 }%
8284 {%
8285 \pgfqpoint%
8286 {.5\pgfdecorationsegmentamplitude}%
8287 {.5\pgfdecorationsegmentamplitude}%
8288 }%
8289 {%
8290 \pgfqpoint%
8291 {\pgfdecorationsegmentamplitude}%
8292 {.5\pgfdecorationsegmentamplitude}%
8293 }%
8294 }%
8295 {%
8296 \pgftransformxshift{+\pgfdecoratedremainingdistance}%
8297 \pgfpathlineto%
8298 {%
8299 \pgfqpoint%
8300 {-\pgfdecorationsegmentamplitude}%
8301 {.5\pgfdecorationsegmentamplitude}%
8302 }%
8303 \pgfpathcurveto%
8304 {%
8305 \pgfqpoint%
8306 {-.5\pgfdecorationsegmentamplitude}%
8307 {.5\pgfdecorationsegmentamplitude}%
8308 }%
8309 {%
8310 \pgfqpoint%
8311 {-.15\pgfdecorationsegmentamplitude}%
8312 {.3\pgfdecorationsegmentamplitude}%
8313 }%
8314 {\pgfqpoint{Opt}{Opt}}%
8315 }%
8316 \tikzset{%
8317 taper width=.5\pgflinewidth,%
8318 taper%
8319 }%
8320 \pgfsyssoftpath@getcurrentpath\cal@tmp@path%
8321 \CopperplatePath{\cal@tmp@path}%
8322 }%
8323 \state{final}{}%
8324 }%

```

The second is a straightened parenthesis (so that when very large it doesn't bow out too far).

```

8325 \pgfdeclaredecoration{calligraphic straight parenthesis}{brace}
8326 {
8327 \state{brace}[width=+\pgfdecoratedremainingdistance,next state=final]%

```



```

8328 {%
8329 \pgfsyssoftpath@setcurrentpath{\pgfutil@empty}%
8330 \pgfpathmoveto{\pgfpointorigin}%
8331 \pgfpathcurveto%
8332 {%
8333 \pgfqpoint%
8334 {.76604\pgfdecorationsegmentamplitude}%
8335 {.64279\pgfdecorationsegmentamplitude}%
8336 }%
8337 {%
8338 \pgfqpoint%
8339 {2.3333\pgfdecorationsegmentamplitude}%
8340 {\pgfdecorationsegmentamplitude}%
8341 }%
8342 {%
8343 \pgfqpoint%
8344 {3.3333\pgfdecorationsegmentamplitude}%
8345 {\pgfdecorationsegmentamplitude}%
8346 }%
8347 {%
8348 \pgftransformxshift{+\pgfdecoratedremainingdistance}%
8349 \pgfpathlineto%
8350 {%
8351 \pgfqpoint%
8352 {-3.3333\pgfdecorationsegmentamplitude}%
8353 {\pgfdecorationsegmentamplitude}%
8354 }%
8355 \pgfpathcurveto%
8356 {%
8357 \pgfqpoint%
8358 {-2.3333\pgfdecorationsegmentamplitude}%
8359 {\pgfdecorationsegmentamplitude}%
8360 }%
8361 {%
8362 \pgfqpoint%
8363 {- .76604\pgfdecorationsegmentamplitude}%
8364 {.64279\pgfdecorationsegmentamplitude}%
8365 }%
8366 {\pgfqpoint{0pt}{0pt}}%
8367 }%
8368 \tikzset{%
8369 taper width=.5\pgflinewidth,%
8370 taper%
8371 }%
8372 \pgfsyssoftpath@getcurrentpath\cal@tmp@path%
8373 \CopperplatePath{\cal@tmp@path}%
8374 }%
8375 \state{final}{}%
8376 }

```

The third is a curved parenthesis.

```

8377 \pgfdeclaredecoration{calligraphic curved parenthesis}{brace}
8378 {
8379 \state{brace}[width=+\pgfdecoratedremainingdistance,next state=final]%
8380 {%

```

```

8381 \pgfsyssoftpath@setcurrentpath{\pgfutil@empty}%
8382 \pgfpathmoveto{\pgfpointorigin}%
8383 \pgf@xa=\pgfdecoratedremainingdistance\relax%
8384 \advance\pgf@xa by -1.5890\pgfdecorationsegmentamplitude\relax%
8385 \edef\cgrphy@xa{\the\pgf@xa}%
8386 \pgfpathcurveto%
8387 {%
8388   \pgfqpoint%
8389   {1.5890\pgfdecorationsegmentamplitude}%
8390   {1.3333\pgfdecorationsegmentamplitude}%
8391 }%
8392 {\pgfqpoint{\cgrphy@xa}{1.3333\pgfdecorationsegmentamplitude}}%
8393 {\pgfqpoint{\pgfdecoratedremainingdistance}{0pt}}%
8394 \tikzset{%
8395   taper width=.5\pgflinewidth,%
8396   taper%
8397 }%
8398 \pgfsyssoftpath@getcurrentpath\cal@tmp@path%
8399 \CopperplatePath{\cal@tmp@path}%
8400 }%
8401 \state{final}{}%
8402 }

```

End the conditional for if pgfdecoration module is loaded

```

8403 \fi

```

6 Drawing Knots

```

8404 <@@=knot>

```

6.1 Initialisation

We load the `spath3` library and the `intersections` TikZ library. Then we get going.

```

8405 \RequirePackage{spath3}
8406 \usetikzlibrary{intersections,spath3}
8407
8408 \ExplSyntaxOn
8409
8410 \tl_new:N \l__knot_tmpa_tl
8411 \tl_new:N \l__knot_tmpb_tl
8412 \tl_new:N \l__knot_tmpc_tl
8413 \tl_new:N \l__knot_tmpd_tl
8414 \tl_new:N \l__knot_tmpg_tl
8415 \tl_new:N \l__knot_redraws_tl
8416 \tl_new:N \l__knot_clip_width_tl
8417 \tl_new:N \l__knot_name_tl
8418 \tl_new:N \l__knot_node_tl
8419 \tl_new:N \l__knot_aux_tl
8420 \tl_new:N \l__knot_auxa_tl
8421 \tl_new:N \l__knot_prefix_tl
8422
8423 \seq_new:N \l__knot_segments_seq
8424
8425 \int_new:N \l__knot_tmpa_int

```

```

8426 \int_new:N \l__knot_strands_int
8427 \int_new:N \g__knot_intersections_int
8428 \int_new:N \g__knot_filaments_int
8429 \int_new:N \l__knot_component_start_int
8430
8431 \fp_new:N \l__knot_tmpa_fp
8432 \fp_new:N \l__knot_tmpb_fp
8433
8434 \dim_new:N \l__knot_tmpa_dim
8435 \dim_new:N \l__knot_tmpb_dim
8436 \dim_new:N \l__knot_tolerance_dim
8437 \dim_new:N \l__knot_redraw_tolerance_dim
8438 \dim_new:N \l__knot_clip_bg_radius_dim
8439 \dim_new:N \l__knot_clip_draw_radius_dim
8440
8441 \bool_new:N \l__knot_draft_bool
8442 \bool_new:N \l__knot_ignore_ends_bool
8443 \bool_new:N \l__knot_self_intersections_bool
8444 \bool_new:N \l__knot_splits_bool
8445 \bool_new:N \l__knot_super_draft_bool
8446
8447 \bool_new:N \l__knot_prepend_prev_bool
8448 \bool_new:N \l__knot_append_next_bool
8449 \bool_new:N \l__knot_skip_bool
8450 \bool_new:N \l__knot_save_bool
8451 \bool_new:N \l__knot_debugging_bool
8452
8453 \seq_new:N \g__knot_nodes_seq
8454
8455 \bool_set_true:N \l__knot_ignore_ends_bool
Configuration is via TikZ keys and styles.
8456 \tikzset{
8457   spath/prefix/knot/.style={
8458     spath/set~ prefix=knot strand,
8459   },
8460   spath/suffix/knot/.style={
8461     spath/set~ suffix={},
8462   },
8463   knot/.code={
8464     \tl_if_eq:nnTF {#1} {none}
8465     {
8466       \tikz@addmode{\tikz@mode@doublefalse}
8467     }
8468     {
8469       \tikz@addmode{\tikz@mode@doubletrue}
8470       \tl_if_eq:nnTF {\pgfkeysnovalue} {#1}
8471       {
8472         \tikz@addoption{\pgfsetinnerstrokecolor{.}}
8473       }
8474       {
8475         \pgfsetinnerstrokecolor{#1}
8476       }
8477       \tikz@addoption{
8478         \pgfsetstrokecolor{knotbg}

```

```

8479     }
8480     \tl_set:Nn \tikz@double@setup{
8481       \pgfsetinnerlinewidth{\pgflinewidth}
8482       \pgfsetlinewidth{\dim_eval:n {\tl_use:N \l__knot_gap_tl \pgflinewidth}}
8483     }
8484   }
8485 },
8486 knot~ gap/.store~ in=\l__knot_gap_tl,
8487 knot~ gap=3,
8488 knot~ diagram/.is~family,
8489 knot~ diagram/.unknown/.code={
8490   \tl_set_eq:NN \l__knot_tmpa_tl \pgfkeyscurrentname
8491   \pgfkeysalso{
8492     /tikz/\l__knot_tmpa_tl=#1
8493   }
8494 },
8495 background~ colour/.code={%
8496   \colorlet{knotbg}{#1}%
8497 },
8498 background~ color/.code={%
8499   \colorlet{knotbg}{#1}%
8500 },
8501 background~ colour=white,
8502 knot~ diagram,
8503 name/.store~ in=\l__knot_name_tl,
8504 name={knot},
8505 save~ intersections/.is~ choice,
8506 save~ intersections/.default=true,
8507 save~ intersections/true/.code={
8508   \bool_set_true:N \l__knot_save_bool
8509 },
8510 save~ intersections/false/.code={
8511   \bool_set_false:N \l__knot_save_bool
8512 },
8513 every~ strand/.style={draw},
8514 ignore~ endpoint~ intersections/.code={
8515   \tl_if_eq:nnTF {#1} {true}
8516   {
8517     \bool_set_true:N \l__knot_ignore_ends_bool
8518   }
8519   {
8520     \bool_set_false:N \l__knot_ignore_ends_bool
8521   }
8522 },
8523 ignore~ endpoint~ intersections/.default=true,
8524 consider~ self~ intersections/.is~choice,
8525 consider~ self~ intersections/true/.code={
8526   \bool_set_true:N \l__knot_self_intersections_bool
8527   \bool_set_true:N \l__knot_splits_bool
8528 },
8529 consider~ self~ intersections/false/.code={
8530   \bool_set_false:N \l__knot_self_intersections_bool
8531   \bool_set_false:N \l__knot_splits_bool
8532 },

```

```

8533 consider~ self~ intersections/no~ splits/.code={
8534   \bool_set_true:N \l__knot_self_intersections_bool
8535   \bool_set_false:N \l__knot_splits_bool
8536 },
8537 consider~ self~ intersections/.default={true},
8538 clip~ radius/.code={
8539   \dim_set:Nn \l__knot_clip_bg_radius_dim {#1}
8540   \dim_set:Nn \l__knot_clip_draw_radius_dim {#1+2pt}
8541 },
8542 clip~ draw~ radius/.code={
8543   \dim_set:Nn \l__knot_clip_draw_radius_dim {#1}
8544 },
8545 clip~ background~ radius/.code={
8546   \dim_set:Nn \l__knot_clip_bg_radius_dim {#1}
8547 },
8548 clip~ radius=10pt,
8549 end~ tolerance/.code={
8550   \dim_set:Nn \l__knot_tolerance_dim {#1}
8551 },
8552 end~ tolerance=14pt,
8553 clip/.style={
8554   clip
8555 },
8556 background~ clip/.style={
8557   clip
8558 },
8559 clip~ width/.code={
8560   \tl_set:Nn \l__knot_clip_width_tl {#1}
8561 },
8562 clip~ width=3,
8563 flip~ crossing/.code={%
8564   \tl_clear_new:c {l__knot_crossing_#1}
8565   \tl_set:cn {l__knot_crossing_#1} {x}
8566 },
8567 ignore~ crossing/.code={%
8568   \tl_clear_new:c {l__knot_ignore_crossing_#1}
8569   \tl_set:cn {l__knot_ignore_crossing_#1} {x}
8570 },
8571 draft~ mode/.is~ choice,
8572 draft~ mode/off/.code={%
8573   \bool_set_false:N \l__knot_draft_bool
8574   \bool_set_false:N \l__knot_super_draft_bool
8575 },
8576 draft~ mode/crossings/.code={%
8577   \bool_set_true:N \l__knot_draft_bool
8578   \bool_set_false:N \l__knot_super_draft_bool
8579 },
8580 draft~ mode/strands/.code={%
8581   \bool_set_true:N \l__knot_draft_bool
8582   \bool_set_true:N \l__knot_super_draft_bool
8583 },
8584 debug/.is~ choice,
8585 debug/true/.code={
8586   \bool_set_true:N \l__knot_debugging_bool

```

```

8587 },
8588 debug/false/.code={
8589   \bool_set_false:N \l__knot_debugging_bool
8590 },
8591 debug/.default=true,
8592 draft/.is~ family,
8593 draft,
8594 crossing~ label/.style={
8595   overlay,
8596   fill=white,
8597   fill~ opacity=.5,
8598   text~ opacity=1,
8599   text=blue,
8600   pin~ edge={blue,<-}
8601 },
8602 strand~ label/.style={
8603   overlay,
8604   circle,
8605   draw=purple,
8606   fill=white,
8607   fill~ opacity=.5,
8608   text~ opacity=1,
8609   text=purple,
8610   inner~ sep=0pt
8611 },
8612 }

```

`\knot_debug:n` Debugging

```

8613 \cs_new_nopar:Npn \knot_debug:n #1
8614 {
8615   \bool_if:NT \l__knot_debugging_bool
8616   {
8617     \iow_term:n {===Knot~ debug: #1===}
8618   }
8619 }
8620
8621 \cs_generate_variant:Nn \knot_debug:n {x}

```

(End definition for \knot_debug:n.)

Wrapper around `\tikzset` for applying keys from a token list, checking for if the given token list exists.

```

8622 \cs_new_nopar:Npn \knot_apply_style:N #1
8623 {
8624   \knot_debug:n {knot~ apply~ style}
8625   \tl_if_exist:NT #1 {
8626     \exp_args:NV \tikzset #1
8627   }
8628 }
8629 \cs_generate_variant:Nn \knot_apply_style:N {c}

```

`\flipcrossings` The user can specify a comma separated list of crossings to flip.

```

8630 \NewDocumentCommand \flipcrossings {m}
8631 {
8632   \tikzset{knot~ diagram/flip~ crossing/.list={#1}}%

```

```
8633 }
```

(End definition for `\flipcrossings`.)

`\strand` This is how the user specifies a strand of the knot.

```
8634 \NewDocumentCommand \strand { 0{} }
8635 {
8636   \int_incr:N \l__knot_strands_int
8637   \tl_clear_new:c {l__knot_options_strand \int_use:N \l__knot_strands_int}
8638   \tl_set:cn {l__knot_options_strand \int_use:N \l__knot_strands_int} {#1}
8639   \path[#1,spath/set~ name=knot,spath/save=\int_use:N \l__knot_strands_int]
8640 }
```

(End definition for `\strand`.)

`knot` This is the wrapper environment that calls the knot generation code.

```
8641 \NewDocumentEnvironment{knot} { 0{} }
8642 {
8643   \knot_initialise:n {#1}
8644 }
8645 {
8646   \knot_render:
8647 }
```

(End definition for `knot`.)

`\knot_initialise:n` Set up some stuff before loading in the strands.

```
8648 \cs_new_protected_nopar:Npn \knot_initialise:n #1
8649 {
8650   \knot_debug:n {knot~ initialise}
8651   \tikzset{knot~ diagram/.cd,every~ knot~ diagram/.try,#1}
8652   \int_zero:N \l__knot_strands_int
8653   \tl_clear:N \l__knot_redraws_tl
8654   \seq_gclear:N \g__knot_nodes_seq
8655 }
```

(End definition for `\knot_initialise:n`.)

`\knot_render:` This is the code that starts the work of rendering the knot.

```
8656 \cs_new_protected_nopar:Npn \knot_render:
8657 {
8658   \knot_debug:n {knot~ render}
```

Start a scope and reset the transformation (since all transformations have already been taken into account when defining the strands).

```
8659   \pgfscope
8660   \pgftransformreset
```

Set the dimension for deciding when to include neighbouring strands

```
8661   \dim_set:Nn \l__knot_redraw_tolerance_dim {\fp_to_dim:n
8662     {
8663       sqrt(2) * max(\l__knot_clip_bg_radius_dim, \l__knot_clip_draw_radius_dim)
8664     }
8665 }
```

Loop through the strands drawing each one for the first time.

```
8666 \int_step_function:nnnN {1} {1} {\l__knot_strands_int} \knot_draw_strand:n
```

In super draft mode we don't do anything else.

```
8667 \bool_if:NF \l__knot_super_draft_bool
8668 {
```

In draft mode we draw labels at the ends of the strands; this also handles splitting curves to avoid self-intersections of Bezier curves if that's requested.

```
8669 \int_step_function:nnnN {1} {1} {\l__knot_strands_int} \knot_draw_labels:n
```

If we're considering self intersections we need to split the strands into filaments.

```
8670 \bool_if:NTF \l__knot_self_intersections_bool
8671 {
8672   \knot_split_strands:
8673   \int_set_eq:NN \l__knot_tmpa_int \g__knot_filaments_int
8674   \tl_set:Nn \l__knot_prefix_tl {filament}
8675 }
8676 {
8677   \int_set_eq:NN \l__knot_tmpa_int \l__knot_strands_int
8678   \tl_set:Nn \l__knot_prefix_tl {strand}
8679 }
```

Initialise the intersection count.

```
8680 \int_gzero:N \g__knot_intersections_int
```

If in draft mode we label the intersections, otherwise we just stick a coordinate at each one.

```
8681 \tl_clear:N \l__knot_node_tl
8682 \bool_if:NT \l__knot_draft_bool
8683 {
8684   \tl_set:Nn \l__knot_node_tl {
8685     \exp_not:N \node[coordinate,
8686       pin={
8687         node~ contents={\int_use:N \g__knot_intersections_int},
8688         knot~ diagram/draft/crossing~ label,
8689         knot~ diagram/draft/crossing~
8690         \int_use:N \g__knot_intersections_int \c_space_tl label/.try
8691       ]
8692     }]}
8693 }
8694 }
```

This double loop steps through the pieces (strands or filaments) and computes the intersections and does stuff with those.

```
8695 \int_step_variable:nnnNn {1} {1} {\l__knot_tmpa_int - 1} \l__knot_tmpa_tl
8696 {
8697   \int_step_variable:nnnNn
8698   {\tl_use:N \l__knot_tmpa_tl + 1}
8699   {1}
8700   {\l__knot_tmpa_int} \l__knot_tmpb_tl
8701   {
8702     \knot_intersections:VV \l__knot_tmpa_tl \l__knot_tmpb_tl
8703   }
8704 }
```


If any redraws were requested, do them here.

```
8705 \tl_use:N \l__knot_redraws_tl
```

Draw the crossing nodes

```
8706 \seq_use:Nn \g__knot_nodes_seq {}
8707 }
```

Close the scope

```
8708 \endpgfscope
8709 \knot_debug:x {knot~rendered,
8710 ~found~\int_use:N \g__knot_intersections_int \c_space_tl~intersections}
8711 }
```

(End definition for \knot_render:.)

`\knot_draw_strand:n` This renders a strand using the options originally specified.

```
8712 \cs_new_protected_nopar:Npn \knot_draw_strand:n #1
8713 {
8714 \knot_debug:n {knot~ draw~ strand~ #1}
8715 \pgfscope
8716 \group_begin:
8717 \spath_bake_round:c {knot strand #1}
8718 \tl_set:Nn \l__knot_tmpa_tl {knot~ diagram/every~ strand/.try,}
8719 \tl_put_right:Nv \l__knot_tmpa_tl {l__knot_options_strand #1}
8720 \tl_put_right:Nn \l__knot_tmpa_tl
8721 {
8722 ,
8723 knot~ diagram/only~ when~ rendering/.try,
8724 only~ when~ rendering/.try,
8725 }
8726 \spath_tikz_path:Vv \l__knot_tmpa_tl {knot strand #1}
8727 \group_end:
8728 \endpgfscope
8729 }
8730 \cs_generate_variant:Nn \tl_put_right:Nn {Nv}
```

(End definition for \knot_draw_strand:n.)

`\knot_draw_labels:n` Draw a label at each end of each strand, if in draft mode. Also, if requested, split potentially self intersecting Bezier curves.

```
8731 \cs_new_protected_nopar:Npn \knot_draw_labels:n #1
8732 {
8733 \knot_debug:n {knot~ draw~ labels}
8734 \bool_if:NT \l__knot_draft_bool
8735 {
8736 \spath_finalpoint:Nv \l__knot_tmpb_tl {knot strand #1}
8737 \dim_set:Nn \l__knot_tmpa_dim {\tl_item:Nn \l__knot_tmpb_tl {1}}
8738 \dim_set:Nn \l__knot_tmpb_dim {\tl_item:Nn \l__knot_tmpb_tl {2}}
8739 \node[
8740 knot~ diagram/draft/strand-label
8741 ] at (\l__knot_tmpa_dim,\l__knot_tmpb_dim) {#1};
8742 \spath_initialpoint:Nv \l__knot_tmpb_tl {knot strand #1}
8743 \dim_set:Nn \l__knot_tmpa_dim {\tl_item:Nn \l__knot_tmpb_tl {1}}
8744 \dim_set:Nn \l__knot_tmpb_dim {\tl_item:Nn \l__knot_tmpb_tl {2}}
8745 \node[
```

```

8746     knot~ diagram/draft/strand-label
8747   ] at (\l__knot_tmpa_dim,\l__knot_tmpb_dim) {#1};
8748 }
8749 \bool_if:nT {
8750   \l__knot_self_intersections_bool
8751   &&
8752   \l__knot_splits_bool
8753 }
8754 {
8755   \tl_clear:N \l__knot_tmpa_tl
8756   \spath_initialpoint:Nv \l__knot_tmpa_tl {knot strand #1}
8757   \tl_put_left:Nv \l__knot_tmpa_tl \c_spath_moveto_tl
8758   \spath_segments_to_seq:Nv \l__knot_segments_seq {knot strand #1}
8759   \seq_map_function:NN \l__knot_segments_seq \knot_split_self_intersects:N
8760   \tl_set_eq:cN {knot strand #1} \l__knot_tmpa_tl
8761 }
8762 }

```

(End definition for \knot_draw_labels:n.)

\knot_split_self_intersects:N

This is the macro that does the split. Figuring out whether a Bezier cubic self intersects is apparently a difficult problem so we don't bother. We compute a point such that if there is an intersection then it lies on either side of the point. I don't recall where the formula came from!

```

8763 \cs_new_protected_nopar:Npn \knot_split_self_intersects:N #1
8764 {
8765   \knot_debug:n {knot~ split~ self~ intersects}
8766   \tl_set:Nx \l__knot_tmpc_tl {\tl_item:nn {#1} {4}}
8767   \tl_case:NnF \l__knot_tmpc_tl
8768   {
8769     \c_spath_curvetoa_tl
8770     {
8771       \fp_set:Nn \l__knot_tmpa_fp
8772       {
8773         (\tl_item:nn {#1} {3} - 3 * \tl_item:nn {#1} {6}
8774         + 3 * \tl_item:nn {#1} {9} - \tl_item:nn {#1} {12})
8775         *
8776         (3 * \tl_item:nn {#1} {8} - 3 * \tl_item:nn {#1} {11})
8777         -
8778         (\tl_item:nn {#1} {2} - 3 * \tl_item:nn {#1} {5}
8779         + 3 * \tl_item:nn {#1} {8} - \tl_item:nn {#1} {11})
8780         *
8781         (3 * \tl_item:nn {#1} {9} - 3 * \tl_item:nn {#1} {12})
8782       }
8783       \fp_set:Nn \l__knot_tmpb_fp
8784       {
8785         (\tl_item:nn {#1} {2} - 3 * \tl_item:nn {#1} {5}
8786         + 3 * \tl_item:nn {#1} {8} - \tl_item:nn {#1} {11})
8787         *
8788         (3 * \tl_item:nn {#1} {6} - 6 * \tl_item:nn {#1} {9}
8789         + 3 * \tl_item:nn {#1} {12})
8790         -
8791         (\tl_item:nn {#1} {3} - 3 * \tl_item:nn {#1} {6}
8792         + 3 * \tl_item:nn {#1} {9} - \tl_item:nn {#1} {12})

```

```

8793      *
8794      (3 * \tl_item:nn {#1} {5} - 6 * \tl_item:nn {#1} {8}
8795      + 3 * \tl_item:nn {#1} {11})
8796    }
8797    \fp_compare:nTF
8798    {
8799      \l__knot_tmpb_fp != 0
8800    }
8801    {
8802      \fp_set:Nn \l__knot_tmpa_fp {5 * \l__knot_tmpa_fp / \l__knot_tmpb_fp}
8803      \fp_compare:nTF
8804      {
8805        0 < \l__knot_tmpa_fp && \l__knot_tmpa_fp < 1
8806      }
8807      {
8808        \spath_split_curve:NNV
8809        \l__knot_tmpc_tl
8810        \l__knot_tmpd_tl
8811        {#1}
8812        \l__knot_tmpa_fp
8813        \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8814        \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8815        \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8816        \tl_set:Nx \l__knot_tmpd_tl {\tl_tail:N \l__knot_tmpd_tl}
8817        \tl_set:Nx \l__knot_tmpd_tl {\tl_tail:N \l__knot_tmpd_tl}
8818        \tl_set:Nx \l__knot_tmpd_tl {\tl_tail:N \l__knot_tmpd_tl}
8819        \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpc_tl
8820        \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpd_tl
8821      }
8822      {
8823        \tl_set:Nn \l__knot_tmpc_tl {#1}
8824        \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8825        \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8826        \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8827        \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpc_tl
8828      }
8829    }
8830    {
8831      \tl_set:Nn \l__knot_tmpc_tl {#1}
8832      \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8833      \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8834      \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8835      \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpc_tl
8836    }
8837  }
8838  \c_spath_lineto_tl
8839  {
8840    \tl_set:Nn \l__knot_tmpc_tl {#1}
8841    \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8842    \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8843    \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8844    \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpc_tl
8845  }
8846 }

```

```

8847 {
8848   \tl_put_right:Nn \l__knot_tmpa_tl {#1}
8849 }
8850 }

```

(End definition for \knot_split_self_intersects:N.)

\knot_intersections:nn This computes the intersections of two pieces and steps through them.

```

8851 \cs_new_protected_nopar:Npn \knot_intersections:nn #1#2
8852 {
8853   \knot_debug:x {knot~ intersections~ between~
8854     \l__knot_prefix_tl \c_space_tl #1~ and~ #2}
8855   \group_begin:
8856   \tl_set_eq:NN \l__knot_tmpa_tl \l__knot_prefix_tl
8857   \tl_put_right:Nn \l__knot_tmpa_tl {#1}
8858   \tl_set_eq:NN \l__knot_tmpb_tl \l__knot_prefix_tl
8859   \tl_put_right:Nn \l__knot_tmpb_tl {#2}
8860   \tl_set_eq:Nc \l__knot_tmpc_tl {knot \tl_use:N \l__knot_tmpa_tl}
8861   \tl_set_eq:Nc \l__knot_tmpd_tl {knot \tl_use:N \l__knot_tmpb_tl}
8862
8863   \bool_if:nTF {
8864     \l__knot_save_bool
8865     &&
8866     \tl_if_exist_p:c {
8867       knot~ intersections~
8868       \tl_use:N \l__knot_name_tl -
8869       \tl_use:N \l__knot_tmpa_tl -
8870       \tl_use:N \l__knot_tmpb_tl
8871     }
8872   }
8873   {
8874     \tl_use:c
8875     {
8876       knot~ intersections~ \tl_use:N \l__knot_name_tl -
8877       \tl_use:N \l__knot_tmpa_tl -
8878       \tl_use:N \l__knot_tmpb_tl
8879     }
8880   }
8881   {
8882     \pgfintersectionofpaths{\pgfsetpath\l__knot_tmpc_tl}{\pgfsetpath\l__knot_tmpd_tl}
8883
8884   }
8885
8886   \knot_debug:x {found~\pgfintersectionsolutions\c_space_tl~ intersections}
8887   \int_compare:nT {\pgfintersectionsolutions > 0}
8888   {
8889     \int_step_function:nnnN
8890     {1}
8891     {1}
8892     {\pgfintersectionsolutions}
8893     \knot_do_intersection:n
8894   }
8895
8896   \knot_save_intersections:VV \l__knot_tmpa_tl \l__knot_tmpb_tl

```

```

8897 \group_end:
8898 }

```

(End definition for \knot_intersections:nn.)

```
\knot_save_intersections:nn
```

```

8899 \cs_new_protected_nopar:Npn \knot_save_intersections:nn #1#2
8900 {
8901 \knot_debug:n {knot~ save~ intersections}
8902 \bool_if:NT \l__knot_save_bool
8903 {
8904 \tl_clear:N \l__knot_aux_tl
8905 \tl_put_right:Nn \l__knot_aux_tl
8906 {
8907 \def\pgfintersectionsolutions
8908 }
8909 \tl_put_right:Nx \l__knot_aux_tl
8910 {
8911 {\int_eval:n {\pgfintersectionsolutions}}
8912 }
8913 \int_compare:nT {\pgfintersectionsolutions > 0}
8914 {
8915 \int_step_inline:nnnn {1} {1} {\pgfintersectionsolutions}
8916 {
8917 \pgfpointintersectionsolution{##1}
8918 \dim_set:Nn \l__knot_tmpa_dim {\pgf@x}
8919 \dim_set:Nn \l__knot_tmpb_dim {\pgf@y}
8920 \tl_put_right:Nn \l__knot_aux_tl
8921 {
8922 \expandafter\def\csname pgfpoint@intersect@solution@##1\endcsname
8923 }
8924 \tl_put_right:Nx \l__knot_aux_tl
8925 {
8926 {
8927 \exp_not:N \pgf@x
8928 =
8929 \dim_use:N \l__knot_tmpa_dim
8930 \exp_not:N \relax
8931 \exp_not:N \pgf@y
8932 =
8933 \dim_use:N \l__knot_tmpb_dim
8934 \exp_not:N \relax
8935 }
8936 }
8937 }
8938 \tl_set:Nn \l__knot_auxa_tl {\expandafter \gdef \csname knot~ intersections~}
8939 \tl_put_right:Nx \l__knot_auxa_tl {\tl_use:N \l__knot_name_tl - #1 - #2}
8940 \tl_put_right:Nn \l__knot_auxa_tl {\endcsname}
8941 \tl_put_right:Nx \l__knot_auxa_tl {\tl_to_str:N \l__knot_aux_tl}
8942 \protected@write\auxout{}\tl_to_str:N \l__knot_auxa_tl}
8943 }
8944 }
8945 }
8946 \cs_generate_variant:Nn \knot_save_intersections:nn {VV}

```

(End definition for `\knot_save_intersections:nn`.)

`\knot_do_intersection:n` This handles a specific intersection.

```

8947 \cs_new_protected_nopar:Npn \knot_do_intersection:n #1
8948 {
8949   \knot_debug:n {knot~ do~ intersection~ #1}

```

Get the intersection coordinates.

```

8950   \pgfpointintersectionsolution{#1}
8951   \dim_set:Nn \l__knot_tmpa_dim {\pgf@x}
8952   \dim_set:Nn \l__knot_tmpb_dim {\pgf@y}
8953   \knot_debug:x {intersection~at~
8954     (\dim_use:N \l__knot_tmpa_dim,\dim_use:N \l__knot_tmpb_dim)}

```

If we're dealing with filaments, we can get false positives from the end points.

```

8955   \bool_set_false:N \l__knot_skip_bool
8956   \bool_if:NT \l__knot_self_intersections_bool
8957   {

```

If one filament preceded the other, test for the intersection being at the relevant end point.

```

8958     \tl_set:Nn \l__knot_tmpc_tl {knot previous}
8959     \tl_put_right:NV \l__knot_tmpc_tl \l__knot_tmpa_tl
8960     \tl_set:Nv \l__knot_tmpc_tl \l__knot_tmpc_tl
8961     \tl_if_eq:NNT \l__knot_tmpc_tl \l__knot_tmpb_tl
8962     {
8963       \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpb_tl {final point}
8964       {
8965         \bool_set_true:N \l__knot_skip_bool
8966       }
8967     }
8968
8969     \tl_set:Nn \l__knot_tmpc_tl {knot previous}
8970     \tl_put_right:NV \l__knot_tmpc_tl \l__knot_tmpb_tl
8971     \tl_set:Nv \l__knot_tmpc_tl \l__knot_tmpc_tl
8972     \tl_if_eq:NNT \l__knot_tmpc_tl \l__knot_tmpa_tl
8973     {
8974       \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpa_tl {final point}
8975       {
8976         \bool_set_true:N \l__knot_skip_bool
8977       }
8978     }
8979   }

```

The user can also say that end points of filaments (or strands) should simply be ignored anyway.

```

8980   \bool_if:NT \l__knot_ignore_ends_bool
8981   {
8982     \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpa_tl {initial point}
8983     {
8984       \bool_set_true:N \l__knot_skip_bool
8985     }
8986     \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpa_tl {final point}
8987     {
8988       \bool_set_true:N \l__knot_skip_bool

```

```

8989     }
8990     \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpb_tl {initial point}
8991     {
8992         \bool_set_true:N \l__knot_skip_bool
8993     }
8994     \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpb_tl {final point}
8995     {
8996         \bool_set_true:N \l__knot_skip_bool
8997     }
8998 }

```

Assuming that we passed all the above tests, we render the crossing.

```

8999     \bool_if:NF \l__knot_skip_bool
9000     {
9001
9002         \int_gincr:N \g__knot_intersections_int
9003         \knot_debug:x {Processing~intersection~\int_use:N \g__knot_intersections_int}

```

This is the intersection test. If the intersection finder finds too many, it might be useful to ignore some.

```

9004     \bool_if:nF
9005     {
9006         \tl_if_exist_p:c {l__knot_ignore_crossing_ \int_use:N
9007             \g__knot_intersections_int}
9008         &&
9009         ! \tl_if_empty_p:c {l__knot_ignore_crossing_ \int_use:N
9010             \g__knot_intersections_int}
9011     }
9012     {

```

This is the flip test. We only render one of the paths. The “flip” swaps which one we render.

```

9013     \bool_if:nTF
9014     {
9015         \tl_if_exist_p:c {l__knot_crossing_ \int_use:N
9016             \g__knot_intersections_int}
9017         &&
9018         ! \tl_if_empty_p:c {l__knot_crossing_ \int_use:N
9019             \g__knot_intersections_int}
9020     }
9021     {
9022         \tl_set_eq:NN \l__knot_tmpg_tl \l__knot_tmpb_tl
9023     }
9024     {
9025         \tl_set_eq:NN \l__knot_tmpg_tl \l__knot_tmpa_tl
9026     }

```

Now we know which one we’re rendering, we test to see if we should also render its predecessor or successor to ensure that we render a path through the entire crossing region.

```

9027     \bool_if:NT \l__knot_self_intersections_bool
9028     {
9029         \knot_test_endpoint:NVnT
9030         \l__knot_redraw_tolerance_dim \l__knot_tmpg_tl {initial point}
9031     }

```

```

9032     \bool_set_true:N \l__knot_prepend_prev_bool
9033   }
9034   {
9035     \bool_set_false:N \l__knot_prepend_prev_bool
9036   }
9037   \knot_test_endpoint:NVT
9038   \l__knot_redraw_tolerance_dim \l__knot_tmpg_tl {final point}
9039   {
9040     \bool_set_true:N \l__knot_append_next_bool
9041   }
9042   {
9043     \bool_set_false:N \l__knot_append_next_bool
9044   }

```

If either of those tests succeeded, do the appending or prepending.

```

9045   \bool_if:nT
9046   {
9047     \l__knot_prepend_prev_bool || \l__knot_append_next_bool
9048   }
9049   {
9050     \tl_clear_new:c {knot \tl_use:N \l__knot_prefix_tl -1}
9051     \tl_set_eq:cc
9052     {knot \tl_use:N \l__knot_prefix_tl -1}
9053     {knot \tl_use:N \l__knot_tmpg_tl}
9054
9055     \tl_clear_new:c {l__knot_options_ \tl_use:N \l__knot_prefix_tl -1}
9056     \tl_set_eq:cc
9057     {l__knot_options_ \tl_use:N \l__knot_prefix_tl -1}
9058     {l__knot_options_ \tl_use:N \l__knot_tmpg_tl}
9059
9060   \bool_if:nT
9061   {
9062     \l__knot_prepend_prev_bool
9063     &&
9064     \tl_if_exist_p:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9065     &&
9066     !\tl_if_empty_p:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9067   }
9068   {
9069     \knot_debug:x {Prepending~
9070       \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}}
9071     \spath_prepend_no_move:cv
9072     {knot \tl_use:N \l__knot_prefix_tl -1}
9073     {knot \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}}

```

If we split potentially self intersecting curves, we test to see if we should prepend yet another segment.

```

9074   \bool_if:nT
9075   {
9076     \l__knot_splits_bool
9077     &&
9078     \tl_if_exist_p:c {knot previous
9079       \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9080   }
9081   &&

```



```

9082         !\tl_if_empty_p:c {knot previous
9083         \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9084     }
9085 }
9086 {
9087     \knot_test_endpoint:NvNT
9088     \l__knot_redraw_tolerance_dim
9089     {knot previous \tl_use:N \l__knot_tmpg_tl}
9090     {initial point}
9091     {
9092         \knot_debug:x {Prepending~
9093         \tl_use:c {knot previous
9094         \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9095         }
9096     }
9097     \spath_prepend_no_move:cv
9098     {knot \tl_use:N \l__knot_prefix_tl -1}
9099     {knot \tl_use:c
9100     {knot previous \tl_use:c
9101     {knot previous \tl_use:N \l__knot_tmpg_tl}
9102     }
9103     }
9104     \tl_set_eq:Nc \l__knot_tmpa_tl
9105     {knot \tl_use:N \l__knot_prefix_tl -1}
9106     }
9107 }
9108 }
9109

```

Now the same for appending.

```

9110     \bool_if:nT
9111     {
9112         \l__knot_append_next_bool
9113         &&
9114         \tl_if_exist_p:c {knot next \tl_use:N \l__knot_tmpg_tl}
9115         &&
9116         !\tl_if_empty_p:c {knot next \tl_use:N \l__knot_tmpg_tl}
9117     }
9118     {
9119         \knot_debug:x {Appending~
9120         \tl_use:c {knot next \tl_use:N \l__knot_tmpg_tl}}
9121         \spath_append_no_move:cv
9122         {knot \tl_use:N \l__knot_prefix_tl -1}
9123         {knot \tl_use:c {knot next \tl_use:N \l__knot_tmpg_tl}}
9124         \bool_if:nT
9125         {
9126             \l__knot_splits_bool
9127             &&
9128             \tl_if_exist_p:c {knot next \tl_use:c { knot next \tl_use:N
9129             \l__knot_tmpg_tl}}
9130             &&
9131             !\tl_if_empty_p:c {knot next
9132             \tl_use:c { knot next \tl_use:N \l__knot_tmpg_tl}
9133             }
9134         }

```

```

9135     {
9136       \knot_debug:x {Testing~ whether~ to~ append~
9137         {knot next \tl_use:c { knot next \tl_use:N \l__knot_tmpg_tl}}
9138       }
9139       \knot_test_endpoint:NvNT
9140       \l__knot_redraw_tolerance_dim
9141       {knot next \tl_use:N \l__knot_tmpg_tl}
9142       {final point}
9143       {
9144         \knot_debug:x {Appending~
9145           {knot next \tl_use:c { knot next \tl_use:N \l__knot_tmpg_tl}}
9146         }
9147         \spath_append_no_move:cv
9148         {knot \tl_use:N \l__knot_prefix_tl -1}
9149         {knot \tl_use:c
9150           {knot next \tl_use:c
9151             {knot next \tl_use:N \l__knot_tmpg_tl}
9152           }
9153         }
9154       }
9155     }
9156   }
9157   \tl_set:Nn \l__knot_tmpg_tl {\tl_use:N \l__knot_prefix_tl -1}
9158 }
9159 }

```

Now we render the crossing.

```

9160   \pgfscope
9161   \group_begin:
9162   \tikzset{
9163     knot~ diagram/every~ intersection/.try,
9164     every~ intersection/.try,
9165     knot~ diagram/intersection~ \int_use:N \g__knot_intersections_int/.try
9166   }
9167   \knot_draw_crossing:VVV \l__knot_tmpg_tl \l__knot_tmpa_dim \l__knot_tmpb_dim
9168   \coordinate
9169   (\l__knot_name_tl \c_space_tl \int_use:N \g__knot_intersections_int)
9170   at (\dim_use:N \l__knot_tmpa_dim, \dim_use:N \l__knot_tmpb_dim);
9171   \group_end:
9172   \endpgfscope

```

This ends the boolean as to whether to consider the intersection at all

```

9173   }

```

And possibly stick a coordinate with a label at the crossing.

```

9174   \tl_if_empty:NF \l__knot_node_tl
9175   {
9176     \seq_gpush:Nx
9177     \g__knot_nodes_seq
9178     {
9179       \l__knot_node_tl
9180       at
9181       (\dim_use:N \l__knot_tmpa_dim, \dim_use:N \l__knot_tmpb_dim) {};
9182     }
9183   }

```

```

9184 }
9185 }
9186
9187 \cs_generate_variant:Nn \knot_intersections:nn {VV}

```

(End definition for \knot_do_intersection:n.)

```

\knot_test_endpoint:N Test whether the point is near the intersection point.
9188 \prg_new_conditional:Npnn \knot_test_endpoint:NN #1#2 {p,T,F,TF}
9189 {
9190   \dim_compare:nTF
9191   {
9192     \dim_abs:n { \l__knot_tmpa_dim - \tl_item:Nn #2 {1}}
9193     +
9194     \dim_abs:n { \l__knot_tmpb_dim - \tl_item:Nn #2 {2}}
9195     <
9196     #1
9197   }
9198   {
9199     \prg_return_true:
9200   }
9201   {
9202     \prg_return_false:
9203   }
9204 }

```

(End definition for \knot_test_endpoint:N.)

```

\knot_test_endpoint:nn Wrapper around the above.
9205 \prg_new_protected_conditional:Npnn \knot_test_endpoint:Nnn #1#2#3 {T,F,TF}
9206 {
9207   \use:c {spath_#3:Nv} \l__knot_tmpd_tl {knot #2}
9208   \knot_test_endpoint:NNTF #1 \l__knot_tmpd_tl
9209   {
9210     \prg_return_true:
9211   }
9212   {
9213     \prg_return_false:
9214   }
9215 }
9216
9217 \cs_generate_variant:Nn \knot_test_endpoint:NnnT {NVnT,NvnT}
9218 \cs_generate_variant:Nn \knot_test_endpoint:NnnF {NVnF,NvnF}
9219 \cs_generate_variant:Nn \knot_test_endpoint:NnnTF {NVnTF,NvnTF}

```

(End definition for \knot_test_endpoint:nn.)

```

\knot_draw_crossing:nnn This is the code that actually renders a crossing.
9220 \cs_new_protected_nopar:Npn \knot_draw_crossing:nnn #1#2#3
9221 {
9222   \knot_debug:n {knot~ draw~ crossing}
9223   \group_begin:
9224   \pgfscope
9225   \path[knot~ diagram/background~ clip] (#2, #3)
9226   circle[radius=\l__knot_clip_bg_radius_dim];

```

```

9227
9228 \tl_set:Nn \l__knot_tmpa_tl {knot~ diagram/every~ strand/.try,}
9229 \tl_if_exist:cT {l__knot_options_ #1}
9230 {
9231 \tl_put_right:Nv \l__knot_tmpa_tl {l__knot_options_ #1}
9232 }
9233 \tl_put_right:Nn \l__knot_tmpa_tl
9234 {
9235 ,knotbg
9236 ,line~ width= \tl_use:N \l__knot_clip_width_tl * \pgflinewidth
9237 }
9238 \spath_tikz_path:Vv \l__knot_tmpa_tl {knot #1}
9239
9240 \endpgfscope
9241
9242 \pgfscope
9243 \path[knot~ diagram/clip] (#2, #3)
9244 circle[radius=\l__knot_clip_draw_radius_dim];
9245
9246 \tl_set:Nn \l__knot_tmpa_tl {knot~ diagram/every~ strand/.try,}
9247 \tl_if_exist:cT {l__knot_options_ #1}
9248 {
9249 \tl_put_right:Nv \l__knot_tmpa_tl {l__knot_options_ #1}
9250 }
9251 \tl_put_right:Nn \l__knot_tmpa_tl
9252 {
9253 ,knot~ diagram/only~ when~ rendering/.try
9254 ,only~ when~ rendering/.try
9255 }
9256 \spath_tikz_path:Vv \l__knot_tmpa_tl {knot #1}
9257
9258 \endpgfscope
9259 \group_end:
9260 }
9261
9262 \cs_generate_variant:Nn \knot_draw_crossing:nnn {nVV, VVV}
9263
9264 \cs_new_protected_nopar:Npn \knot_draw_crossing:nn #1#2
9265 {
9266 \tikz@scan@one@point\pgfutil@firstofone #2 \relax
9267 \knot_draw_crossing:nVV {#1} \pgf@x \pgf@y
9268 }

```

(End definition for \knot_draw_crossing:nnn.)

\knot_split_strands: This, and the following macros, are for splitting strands into filaments.

```

9269 \cs_new_protected_nopar:Npn \knot_split_strands:
9270 {
9271 \knot_debug:n {knot~ split~ strands}
9272 \int_gzero:N \g__knot_filaments_int
9273 \int_step_function:nnnN {1} {1} {\l__knot_strands_int} \knot_split_strand:n
9274 \int_step_function:nnnN {1} {1} {\g__knot_filaments_int} \knot_compute_nexts:n
9275 }

```

(End definition for \knot_split_strands:.)

`\knot_compute_nexts:n` Each filament needs to know its predecessor and successor. We work out the predecessors as we go along, this fills in the successors.

```

9276 \cs_new_protected_nopar:Npn \knot_compute_nexts:n #1
9277 {
9278   \knot_debug:n {knot~ compute~ nexts}
9279   \tl_clear_new:c {knot next \tl_use:c {knot previous filament #1}}
9280   \tl_set:cn {knot next \tl_use:c {knot previous filament #1}} {filament #1}
9281 }

```

(End definition for \knot_compute_nexts:n.)

`\knot_split_strand:n` Sets up the split for a single strand.

```

9282 \cs_new_protected_nopar:Npn \knot_split_strand:n #1
9283 {
9284   \knot_debug:n {knot~ split~ strand}
9285   \int_set_eq:NN \l__knot_component_start_int \g__knot_filaments_int
9286   \int_incr:N \l__knot_component_start_int
9287   \tl_set_eq:Nc \l__knot_tmpa_tl {l__knot_options_strand #1}
9288   \spath_segments_to_seq:Nv \l__knot_segments_seq {knot strand #1}
9289   \seq_map_function:NN \l__knot_segments_seq \knot_save_filament:N
9290 }

```

(End definition for \knot_split_strand:n.)

`\knot_save_filament:N` Saves a filament as a new `spath` object.

```

9291 \cs_new_protected_nopar:Npn \knot_save_filament:N #1
9292 {
9293   \knot_debug:n {knot~ save~ filament}
9294   \tl_set:Nx \l__knot_tmpb_tl {\tl_item:nn {#1} {4}}
9295   \tl_case:NnF \l__knot_tmpb_tl
9296   {
9297     \c_spath_moveto_tl
9298     {
9299       \int_compare:nT {\l__knot_component_start_int < \g__knot_filaments_int}
9300       {
9301         \int_set_eq:NN \l__knot_component_start_int \g__knot_filaments_int
9302       }
9303     }
9304     \c_spath_lineto_tl
9305     {
9306       \int_gincr:N \g__knot_filaments_int
9307       \tl_clear_new:c {knot filament \int_use:N \g__knot_filaments_int}
9308       \tl_set:cn {knot filament \int_use:N \g__knot_filaments_int} {#1}
9309
9310       \tl_clear_new:c {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9311       \tl_set_eq:cN {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9312       \l__knot_tmpa_tl
9313
9314       \tl_clear_new:c {knot previous filament \int_use:N \g__knot_filaments_int}
9315       \int_compare:nF {\l__knot_component_start_int == \g__knot_filaments_int}
9316       {
9317         \tl_set:cx {knot previous filament \int_use:N \g__knot_filaments_int}
9318         {filament \int_eval:n {\g__knot_filaments_int - 1}}
9319       }
9319     }

```

```

9320 }
9321 \c_spath_curvetoa_tl
9322 {
9323   \int_gincr:N \g__knot_filaments_int
9324   \tl_clear_new:c {knot filament \int_use:N \g__knot_filaments_int}
9325   \tl_set:cn {knot filament \int_use:N \g__knot_filaments_int} {#1}
9326   \tl_clear_new:c {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9327   \tl_set_eq:cN {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9328   \l__knot_tmpa_tl
9329
9330   \tl_clear_new:c {knot previous filament \int_use:N \g__knot_filaments_int}
9331   \int_compare:nF {\l__knot_component_start_int == \g__knot_filaments_int}
9332   {
9333     \tl_set:cx
9334     {knot previous filament \int_use:N \g__knot_filaments_int}
9335     {filament \int_eval:n {\g__knot_filaments_int - 1}}
9336   }
9337 }
9338 \c_spath_closepath_tl
9339 {
9340   \int_gincr:N \g__knot_filaments_int
9341   \tl_clear_new:c {knot filament \int_use:N \g__knot_filaments_int}
9342   \tl_clear:N \l__knot_tmpa_tl
9343   \tl_put_right:Nx
9344   {
9345     \tl_item:nn {#1} {1}\tl_item:nn {#1} {2}\tl_item:nn {#1} {3}
9346   }
9347   \tl_put_right:NV \l__knot_tmpa_tl \c_spath_lineto_tl
9348   \tl_put_right:Nx {\tl_item:nn {#1} {5}\tl_item:nn {#1} {6}}
9349
9350   \tl_set:cV {knot filament \int_use:N \g__knot_filaments_int} \l__knot_tmpa_tl
9351   \tl_set_eq:cN {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9352   \l__knot_tmpa_tl
9353   \tl_clear_new:c {knot previous filament \int_use:N \g__knot_filaments_int}
9354   \int_compare:nF {\l__knot_component_start_int == \g__knot_filaments_int}
9355   {
9356     \tl_set:cx
9357     {knot previous filament \int_use:N \g__knot_filaments_int}
9358     {filament \int_eval:n {\g__knot_filaments_int - 1}}
9359   }
9360   \tl_set:cx
9361   {knot previous filament \int_use:N \l__knot_component_start_int}
9362   {filament \int_use:N \g__knot_filaments_int}
9363 }
9364 }
9365 {
9366 }
9367 }

```

(End definition for \knot_save_filament:N.)

\redraw The user can redraw segments of the strands at specific locations.

```

9368 \NewDocumentCommand \redraw { m m }
9369 {

```

```

9370 % \tikz@scan@one@point\pgfutil@firstofone #2 \relax
9371 \tl_put_right:Nn \l__knot_redraws_tl {\knot_draw_crossing:nm}
9372 \tl_put_right:Nx \l__knot_redraws_tl {
9373   {strand #1} {#2}% {\dim_use:N \pgf@x} {\dim_use:N \pgf@y}
9374 }
9375 }

```

(End definition for \redraw.)

```
9376 \ExplSyntaxOff
```

```
<@@=>
```

\pgf@sh__knotknotanchor Add the extra anchors for the knot crossing nodes.

```

9377 \def\pgf@sh__knotknotanchor#1#2{%
9378   \anchor{#2 north west}{%
9379     \csname pgf@anchor@knot #1@north west\endcsname%
9380     \pgf@x=#2\pgf@x%
9381     \pgf@y=#2\pgf@y%
9382   }%
9383   \anchor{#2 north east}{%
9384     \csname pgf@anchor@knot #1@north east\endcsname%
9385     \pgf@x=#2\pgf@x%
9386     \pgf@y=#2\pgf@y%
9387   }%
9388   \anchor{#2 south west}{%
9389     \csname pgf@anchor@knot #1@south west\endcsname%
9390     \pgf@x=#2\pgf@x%
9391     \pgf@y=#2\pgf@y%
9392   }%
9393   \anchor{#2 south east}{%
9394     \csname pgf@anchor@knot #1@south east\endcsname%
9395     \pgf@x=#2\pgf@x%
9396     \pgf@y=#2\pgf@y%
9397   }%
9398   \anchor{#2 north}{%
9399     \csname pgf@anchor@knot #1@north\endcsname%
9400     \pgf@x=#2\pgf@x%
9401     \pgf@y=#2\pgf@y%
9402   }%
9403   \anchor{#2 east}{%
9404     \csname pgf@anchor@knot #1@east\endcsname%
9405     \pgf@x=#2\pgf@x%
9406     \pgf@y=#2\pgf@y%
9407   }%
9408   \anchor{#2 west}{%
9409     \csname pgf@anchor@knot #1@west\endcsname%
9410     \pgf@x=#2\pgf@x%
9411     \pgf@y=#2\pgf@y%
9412   }%
9413   \anchor{#2 south}{%
9414     \csname pgf@anchor@knot #1@south\endcsname%
9415     \pgf@x=#2\pgf@x%
9416     \pgf@y=#2\pgf@y%
9417   }%
9418 }

```

(End definition for \pgf@sh__knotknotanchor.)

knot_crossing

```
9419 \pgfdeclareshape{knot crossing}
9420 {
9421   \inheritsavedanchors[from=circle] % this is nearly a circle
9422   \inheritanchorborder[from=circle]
9423   \inheritanchor[from=circle]{north}
9424   \inheritanchor[from=circle]{north west}
9425   \inheritanchor[from=circle]{north east}
9426   \inheritanchor[from=circle]{center}
9427   \inheritanchor[from=circle]{west}
9428   \inheritanchor[from=circle]{east}
9429   \inheritanchor[from=circle]{mid}
9430   \inheritanchor[from=circle]{mid west}
9431   \inheritanchor[from=circle]{mid east}
9432   \inheritanchor[from=circle]{base}
9433   \inheritanchor[from=circle]{base west}
9434   \inheritanchor[from=circle]{base east}
9435   \inheritanchor[from=circle]{south}
9436   \inheritanchor[from=circle]{south west}
9437   \inheritanchor[from=circle]{south east}
9438   \inheritanchorborder[from=circle]
9439   \pgf@sh__knotknotanchor{crossing}{2}
9440   \pgf@sh__knotknotanchor{crossing}{3}
9441   \pgf@sh__knotknotanchor{crossing}{4}
9442   \pgf@sh__knotknotanchor{crossing}{8}
9443   \pgf@sh__knotknotanchor{crossing}{16}
9444   \pgf@sh__knotknotanchor{crossing}{32}
9445   \backgroundpath{
9446     \pgfutil@tempdima=\radius%
9447     \pgfmathsetlength{\pgf@xb}{\pgfkeysvalueof{/pgf/outer xsep}}%
9448     \pgfmathsetlength{\pgf@yb}{\pgfkeysvalueof{/pgf/outer ysep}}%
9449     \ifdim\pgf@xb<\pgf@yb%
9450       \advance\pgfutil@tempdima by-\pgf@yb%
9451     \else%
9452       \advance\pgfutil@tempdima by-\pgf@xb%
9453     \fi%
9454   }
9455 }
```

(End definition for knot crossing.)

knot_over_cross

```
9456 \pgfdeclareshape{knot over cross}
9457 {
9458   \inheritsavedanchors[from=rectangle] % this is nearly a circle
9459   \inheritanchorborder[from=rectangle]
9460   \inheritanchor[from=rectangle]{north}
9461   \inheritanchor[from=rectangle]{north west}
9462   \inheritanchor[from=rectangle]{north east}
9463   \inheritanchor[from=rectangle]{center}
9464   \inheritanchor[from=rectangle]{west}
9465   \inheritanchor[from=rectangle]{east}
```



```

9466 \inheritanchor[from=rectangle]{mid}
9467 \inheritanchor[from=rectangle]{mid west}
9468 \inheritanchor[from=rectangle]{mid east}
9469 \inheritanchor[from=rectangle]{base}
9470 \inheritanchor[from=rectangle]{base west}
9471 \inheritanchor[from=rectangle]{base east}
9472 \inheritanchor[from=rectangle]{south}
9473 \inheritanchor[from=rectangle]{south west}
9474 \inheritanchor[from=rectangle]{south east}
9475 \inheritanchorborder[from=rectangle]
9476 \backgroundpath{
9477   \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9478   \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9479   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@ya}}
9480   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9481 }
9482 \foregroundpath{
9483 % store lower right in xa/ya and upper right in xb/yb
9484   \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9485   \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9486   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9487   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@ya}}
9488 }
9489 }

```

(End definition for knot over cross.)

knot_under_cross

```

9490 \pgfdeclareshape{knot under cross}
9491 {
9492   \inheritavedanchors[from=rectangle] % this is nearly a circle
9493   \inheritanchorborder[from=rectangle]
9494   \inheritanchor[from=rectangle]{north}
9495   \inheritanchor[from=rectangle]{north west}
9496   \inheritanchor[from=rectangle]{north east}
9497   \inheritanchor[from=rectangle]{center}
9498   \inheritanchor[from=rectangle]{west}
9499   \inheritanchor[from=rectangle]{east}
9500   \inheritanchor[from=rectangle]{mid}
9501   \inheritanchor[from=rectangle]{mid west}
9502   \inheritanchor[from=rectangle]{mid east}
9503   \inheritanchor[from=rectangle]{base}
9504   \inheritanchor[from=rectangle]{base west}
9505   \inheritanchor[from=rectangle]{base east}
9506   \inheritanchor[from=rectangle]{south}
9507   \inheritanchor[from=rectangle]{south west}
9508   \inheritanchor[from=rectangle]{south east}
9509   \inheritanchorborder[from=rectangle]
9510   \backgroundpath{
9511     \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9512     \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9513     \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9514     \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@ya}}
9515   }

```

```

9516 \foregroundpath{
9517 % store lower right in xa/ya and upper right in xb/yb
9518   \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9519   \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9520   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@ya}}
9521   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9522 }
9523 }

```

(End definition for knot under cross.)

knot_vert

```

9524 \pgfdeclareshape{knot vert}
9525 {
9526   \inheritsavedanchors[from=rectangle] % this is nearly a circle
9527   \inheritanchorborder[from=rectangle]
9528   \inheritanchor[from=rectangle]{north}
9529   \inheritanchor[from=rectangle]{north west}
9530   \inheritanchor[from=rectangle]{north east}
9531   \inheritanchor[from=rectangle]{center}
9532   \inheritanchor[from=rectangle]{west}
9533   \inheritanchor[from=rectangle]{east}
9534   \inheritanchor[from=rectangle]{mid}
9535   \inheritanchor[from=rectangle]{mid west}
9536   \inheritanchor[from=rectangle]{mid east}
9537   \inheritanchor[from=rectangle]{base}
9538   \inheritanchor[from=rectangle]{base west}
9539   \inheritanchor[from=rectangle]{base east}
9540   \inheritanchor[from=rectangle]{south}
9541   \inheritanchor[from=rectangle]{south west}
9542   \inheritanchor[from=rectangle]{south east}
9543   \inheritanchorborder[from=rectangle]
9544   \backgroundpath{
9545     % store lower right in xa/ya and upper right in xb/yb
9546     \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9547     \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9548     \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@ya}}
9549     \pgfpathlineto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9550     \pgfpathmoveto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9551     \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@ya}}
9552   }
9553 }

```

(End definition for knot vert.)

knot_horiz

```

9554 \pgfdeclareshape{knot horiz}
9555 {
9556   \inheritsavedanchors[from=rectangle] % this is nearly a circle
9557   \inheritanchorborder[from=rectangle]
9558   \inheritanchor[from=rectangle]{north}
9559   \inheritanchor[from=rectangle]{north west}
9560   \inheritanchor[from=rectangle]{north east}
9561   \inheritanchor[from=rectangle]{center}
9562   \inheritanchor[from=rectangle]{west}

```

```

9563 \inheritanchor[from=rectangle]{east}
9564 \inheritanchor[from=rectangle]{mid}
9565 \inheritanchor[from=rectangle]{mid west}
9566 \inheritanchor[from=rectangle]{mid east}
9567 \inheritanchor[from=rectangle]{base}
9568 \inheritanchor[from=rectangle]{base west}
9569 \inheritanchor[from=rectangle]{base east}
9570 \inheritanchor[from=rectangle]{south}
9571 \inheritanchor[from=rectangle]{south west}
9572 \inheritanchor[from=rectangle]{south east}
9573 \inheritanchorborder[from=rectangle]
9574 \foregroundpath{
9575 % store lower right in xa/ya and upper right in xb/yb
9576   \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9577   \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9578   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@ya}}
9579   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@ya}}
9580   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9581   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9582 }
9583 }

```

(End definition for knot horiz.)