

The `spath3` package: code

Andrew Stacey

loopspace@mathforge.org

v2.8 from 2024/05/31

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 `tilings` (formerly, `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`
- `tilings`
- `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} {2024/05/31} {2.8} {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 on, 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
```

When manipulating soft paths, we might need to separate the shortening due to an arrow from when the path is rendered.

```
70 \bool_new:N \l_spath_arrow_shortening_bool
71 \bool_set_true:N \l_spath_arrow_shortening_bool
```

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

```
72 \tl_new:N \l__spath_intersecta_tl
73 \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.

```
74 \tl_const:Nn \c_spath_moveto_tl {\pgfsyssoftpath@movetotoken}
75 \tl_const:Nn \c_spath_lineto_tl {\pgfsyssoftpath@linetotoken}
76 \tl_const:Nn \c_spath_curveto_tl {\pgfsyssoftpath@curvetotoken}
77 \tl_const:Nn \c_spath_curvetoa_tl {\pgfsyssoftpath@curvetosupportatoken}
78 \tl_const:Nn \c_spath_curvetob_tl {\pgfsyssoftpath@curvetosupportbtoken}
79 \tl_const:Nn \c_spath_closepath_tl {\pgfsyssoftpath@closepathtoken}
80 \tl_const:Nn \c_spath_rectcorner_tl {\pgfsyssoftpath@rectcornertoken}
81 \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.

```
82 \int_new:N \g__spath_anon_int
83 \int_gzero:N \g__spath_anon_int
```

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

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

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

And some error messages

```
96 \msg_new:nnn { spath3 } { unknown path construction }
97 { 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
  98 \cs_new_protected_nopar:Npn \__spath_segments_to_seq:n #1
  99 {
 100   \group_begin:
 101   \tl_set:Nn \l__spath_tmpa_tl {#1}
 102   \tl_clear:N \l__spath_tmpb_tl
 103   \seq_clear:N \l__spath_tmpa_seq
 104   \dim_zero:N \l__spath_tmpa_dim
 105   \dim_zero:N \l__spath_tmpb_dim
 106
 107   \bool_until_do:nn {
 108     \tl_if_empty_p:N \l__spath_tmpa_tl
 109   }
 110   {
 111     \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
 112     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
 113     \token_case_meaning:NnF \l__spath_tmpc_tl
 114     {
 115       \c_spath_moveto_tl
 116       {
 117         \tl_set_eq:NN \l__spath_tmpb_tl \c_spath_moveto_tl
 118         \tl_put_right:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
 119         \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
 120         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
 121
 122         \tl_put_right:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
 123         \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
 124         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
 125
 126         \tl_set:Nx \l__spath_tmpd_tl {\tl_head:N \l__spath_tmpa_tl}
 127         \tl_if_eq:NNF \l__spath_tmpd_tl \c_spath_moveto_tl
 128         {
 129           \tl_clear:N \l__spath_tmpb_tl
 130         }
 131
 132       }
 133
 134       \c_spath_lineto_tl
 135       {
 136         \tl_set_eq:NN \l__spath_tmpb_tl \c_spath_moveto_tl
 137         \tl_put_right:Nx \l__spath_tmpb_tl
 138         {
 139           {\dim_use:N \l__spath_tmpa_dim}
 140           {\dim_use:N \l__spath_tmpb_dim}
 141         }
 142         \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl

```

```

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

```

```

197
198     \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
199     \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
200     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
201
202   }
203
204   \c_spath_closepath_tl
205   {
206     \tl_set_eq:NN \l__spath_tmpb_tl \c_spath_moveto_tl
207     \tl_put_right:Nx \l__spath_tmpb_tl
208     {
209       {\dim_use:N \l__spath_tmpa_dim}
210       {\dim_use:N \l__spath_tmpb_dim}
211     }
212     \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
213
214     \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
215     \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
216     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
217
218     \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
219     \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
220     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
221
222   }
223
224 }
225 {
226
227   \tl_set_eq:NN \l__spath_tmpb_tl \l__spath_tmpc_tl
228   \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
229   \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
230   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
231
232   \tl_put_right:Nx \l__spath_tmpb_tl {{\tl_head:N \l__spath_tmpa_tl}}
233   \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
234   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
235
236 }
237
238 \tl_if_empty:NF \l__spath_tmpb_tl
239 {
240   \seq_put_right:NV \l__spath_tmpa_seq \l__spath_tmpb_tl
241 }
242 \tl_clear:N \l__spath_tmpb_tl
243 }
244
245 \seq_gclear:N \g__spath_output_seq
246 \seq_gset_eq:NN \g__spath_output_seq \l__spath_tmpa_seq
247 \group_end:
248 }
249 \cs_new_protected_nopar:Npn \spath_segments_to_seq:Nn #1#2
250 {

```

```

251 \__spath_segments_to_seq:n {#2}
252 \seq_clear_new:N #1
253 \seq_set_eq:NN #1 \g__spath_output_seq
254 \seq_gclear:N \g__spath_output_seq
255 }
256 \cs_generate_variant:Nn \spath_segments_to_seq:Nn {NV, cn, cV, Nv, cv}
257 \cs_new_protected_nopar:Npn \spath_segments_gto_seq:Nn #1#2
258 {
259 \__spath_segments_to_seq:n {#2}
260 \seq_clear_new:N #1
261 \seq_gset_eq:NN #1 \g__spath_output_seq
262 \seq_gclear:N \g__spath_output_seq
263 }
264 \cs_generate_variant:Nn \spath_segments_gto_seq:Nn {NV, cn, cV, Nv, cv}

```

(End of 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_to_seq:Nn
\spath_components_gto_seq:Nn
\spath_components_to_clist:Nn
\spath_components_gto_clist:Nn
265 \cs_new_protected_nopar:Npn \__spath_components_to_seq:n #1
266 {
267 \group_begin:
268 \tl_set:Nn \l__spath_tmpa_tl {#1}
269 \seq_clear:N \l__spath_tmpa_seq
270 \tl_set:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
271 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
272
273 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_moveto_tl
274
275 \bool_do_until:nm {
276 \tl_if_empty_p:N \l__spath_tmpa_tl
277 }
278 {
279 \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
280 \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_moveto_tl
281 {
282 \seq_put_right:NV \l__spath_tmpa_seq \l__spath_tmpb_tl
283 \tl_clear:N \l__spath_tmpb_tl
284 }
285 \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl
286 {
287 \seq_put_right:NV \l__spath_tmpa_seq \l__spath_tmpb_tl
288 \tl_clear:N \l__spath_tmpb_tl
289 }
290 \tl_if_single:NTF \l__spath_tmpc_tl
291 {
292 \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpc_tl
293 }
294 {
295 \tl_put_right:Nx \l__spath_tmpb_tl {\l__spath_tmpc_tl}
296 }
297 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
298 }
299
300 \seq_gclear:N \g__spath_output_seq

```



```

301 \seq_gset_eq:NN \g__spath_output_seq \l__spath_tmpa_seq
302 \group_end:
303 }
304 \cs_new_protected_nopar:Npn \spath_components_to_seq:Nn #1#2
305 {
306   \__spath_components_to_seq:n {#2}
307   \seq_clear_new:N #1
308   \seq_set_eq:NN #1 \g__spath_output_seq
309   \seq_gclear:N \g__spath_output_seq
310 }
311 \cs_generate_variant:Nn \spath_components_to_seq:Nn {NV, cn, cV, cv, Nv}
312 \cs_new_protected_nopar:Npn \spath_components_gto_seq:Nn #1#2
313 {
314   \__spath_components_to_seq:n {#2}
315   \seq_clear_new:N #1
316   \seq_gset_eq:NN #1 \g__spath_output_seq
317   \seq_gclear:N \g__spath_output_seq
318 }
319 \cs_generate_variant:Nn \spath_components_gto_seq:Nn {NV, cn, cV, cv, Nv}
320 \cs_new_protected_nopar:Npn \spath_components_to_clist:Nn #1#2
321 {
322   \__spath_components_to_seq:n {#2}
323   \clist_clear_new:N #1
324   \clist_set_from_seq:NN #1 \g__spath_output_seq
325   \seq_gclear:N \g__spath_output_seq
326 }
327 \cs_generate_variant:Nn \spath_components_to_clist:Nn {NV, cn, cV, cv, Nv}
328 \cs_new_protected_nopar:Npn \spath_components_gto_clist:Nn #1#2
329 {
330   \__spath_components_to_seq:n {#2}
331   \clist_clear_new:N #1
332   \clist_gset_from_seq:NN #1 \g__spath_output_seq
333   \seq_gclear:N \g__spath_output_seq
334 }
335 \cs_generate_variant:Nn \spath_components_gto_clist:Nn {NV, cn, cV, cv, Nv}

```

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

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

```

336 \cs_new_protected_nopar:Npn \spath_length:n #1
337 {
338   \int_eval:n {\tl_count:n {#1} / 3}
339 }
340 \cs_generate_variant:Nn \spath_length:n {V}

```

(End of 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).

```

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

```

```

346 \tl_set:Nn \l__spath_tmpa_tl {##1}
347 \token_case_meaning:NnT \l__spath_tmpa_tl
348 {
349   \c_spath_lineto_tl {}
350   \c_spath_curveto_tl {}
351   \c_spath_closepath_tl {}
352   \c_spath_rectsize_tl {}
353 }
354 {
355   \int_incr:N \l__spath_tmpa_int
356 }
357 }
358 \int_gzero:N \g__spath_output_int
359 \int_gset_eq:NN \g__spath_output_int \l__spath_tmpa_int
360 \group_end:
361 }
362 \cs_new_protected_nopar:Npn \spath_reallength:Nn #1#2
363 {
364   \__spath_reallength:n {#2}
365   \int_set_eq:NN #1 \g__spath_output_int
366   \int_gzero:N \g__spath_output_int
367 }
368 \cs_generate_variant:Nn \spath_reallength:Nn {NV, cn, cV, Nv, cv}
369 \cs_new_protected_nopar:Npn \spath_greallength:Nn #1#2
370 {
371   \__spath_reallength:n {#2}
372   \int_gset_eq:NN #1 \g__spath_output_int
373   \int_gzero:N \g__spath_output_int
374 }
375 \cs_generate_variant:Nn \spath_greallength:Nn {NV, cn, cV}

```

(End of 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
376 \cs_new_protected_nopar:Npn \__spath_numberofcomponents:n #1
377 {
378   \group_begin:
379   \int_set:Nn \l__spath_tmpa_int {0}
380   \tl_map_inline:nn {#1} {
381     \tl_set:Nn \l__spath_tmpa_tl {##1}
382     \token_case_meaning:Nn \l__spath_tmpa_tl
383     {
384       \c_spath_moveto_tl
385       {
386         \int_incr:N \l__spath_tmpa_int
387       }
388       \c_spath_rectcorner_tl
389       {
390         \int_incr:N \l__spath_tmpa_int
391       }
392     }
393   }
394   \int_gzero:N \g__spath_output_int

```

```

395 \int_gset_eq:NN \g__spath_output_int \l__spath_tmpa_int
396 \group_end:
397 }
398 \cs_new_protected_nopar:Npn \spath_numberofcomponents:Nn #1#2
399 {
400   \__spath_numberofcomponents:n {#2}
401   \int_set_eq:NN #1 \g__spath_output_int
402   \int_gzero:N \g__spath_output_int
403 }
404 \cs_generate_variant:Nn \spath_numberofcomponents:Nn {NV, cn, cV, Nv}
405 \cs_new_protected_nopar:Npn \spath_gnumberofcomponents:Nn #1#2
406 {
407   \__spath_numberofcomponents:n {#2}
408   \int_gset_eq:NN #1 \g__spath_output_int
409   \int_gzero:N \g__spath_output_int
410 }
411 \cs_generate_variant:Nn \spath_gnumberofcomponents:Nn {NV, cn, cV, Nv}

```

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

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

```

\spath_ginitialpoint:Nn
412 \cs_new_protected_nopar:Npn \__spath_initialpoint:n #1
413 {
414   \group_begin:
415   \tl_clear:N \l__spath_tmpa_tl
416   \tl_set:Nx \l__spath_tmpa_tl
417   {
418     { \tl_item:nn {#1} {2} }
419     { \tl_item:nn {#1} {3} }
420   }
421   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
422   \group_end:
423 }
424 \cs_new_protected_nopar:Npn \spath_initialpoint:Nn #1#2
425 {
426   \__spath_initialpoint:n {#2}
427   \tl_set_eq:NN #1 \g__spath_output_tl
428   \tl_gclear:N \g__spath_output_tl
429 }
430 \cs_generate_variant:Nn \spath_initialpoint:Nn {NV, cn, cV, Nv}
431 \cs_new_protected_nopar:Npn \spath_ginitialpoint:Nn #1#2
432 {
433   \__spath_initialpoint:n {#2}
434   \tl_gset_eq:NN #1 \g__spath_output_tl
435   \tl_gclear:N \g__spath_output_tl
436 }
437 \cs_generate_variant:Nn \spath_ginitialpoint:Nn {NV, cn, cV, Nv}

```

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

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

```

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

```

```

442 \tl_reverse:N \l__spath_tmpa_tl
443 \tl_clear:N \l__spath_tmpb_tl
444 \tl_set:Nx \l__spath_tmpc_tl {\tl_item:Nn \l__spath_tmpa_tl {3}}
445 \tl_if_eq:NNTF \l__spath_tmpc_tl \c_spath_rectsize_tl
446 {
447   \tl_set:Nx \l__spath_tmpb_tl
448   {
449     {
450       \dim_eval:n
451       {
452         \tl_item:Nn \l__spath_tmpa_tl {2}
453         +
454         \tl_item:Nn \l__spath_tmpa_tl {5}
455       }
456     }
457     {
458       \dim_eval:n
459       {
460         \tl_item:Nn \l__spath_tmpa_tl {1}
461         +
462         \tl_item:Nn \l__spath_tmpa_tl {4}
463       }
464     }
465   }
466 }
467 {
468   \tl_set:Nx \l__spath_tmpb_tl
469   {
470     { \tl_item:Nn \l__spath_tmpa_tl {2} }
471     { \tl_item:Nn \l__spath_tmpa_tl {1} }
472   }
473 }
474 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
475 \group_end:
476 }
477 \cs_new_protected_nopar:Npn \spath_finalpoint:Nn #1#2
478 {
479   \__spath_finalpoint:n {#2}
480   \tl_set_eq:NN #1 \g__spath_output_tl
481   \tl_gclear:N \g__spath_output_tl
482 }
483 \cs_generate_variant:Nn \spath_finalpoint:Nn {NV, cn, cV, Nv}
484 \cs_new_protected_nopar:Npn \spath_gfinalpoint:Nn #1#2
485 {
486   \__spath_finalpoint:n {#2}
487   \tl_gset_eq:NN #1 \g__spath_output_tl
488   \tl_gclear:N \g__spath_output_tl
489 }
490 \cs_generate_variant:Nn \spath_gfinalpoint:Nn {NV, cn, cV, Nv}

```

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

```

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

```

```

492 {
493   \group_begin:
494   \tl_set:Nn \l__spath_tmpc_tl { {Opt} {Opt} }
495   \tl_set:Nn \l__spath_tmpa_tl {#1}
496   \bool_do_until:nn
497   {
498     \tl_if_empty_p:N \l__spath_tmpa_tl
499   }
500   {
501     \tl_set:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
502     \token_case_meaning:Nn \l__spath_tmpb_tl
503     {
504       \c_spath_moveto_tl
505       {
506         \tl_set:Nx \l__spath_tmpc_tl
507         {
508           { \tl_item:Nn \l__spath_tmpa_tl {2} }
509           { \tl_item:Nn \l__spath_tmpa_tl {3} }
510         }
511       }
512
513       \c_spath_rectcorner_tl
514       {
515         \tl_set:Nx \l__spath_tmpc_tl
516         {
517           { \tl_item:Nn \l__spath_tmpa_tl {2} }
518           { \tl_item:Nn \l__spath_tmpa_tl {3} }
519         }
520       }
521
522     }
523     \prg_replicate:nn {3}
524     {
525       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
526     }
527   }
528   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpc_tl
529   \group_end:
530 }
531 \cs_new_protected_nopar:Npn \spath_finalmovepoint:Nn #1#2
532 {
533   \__spath_finalmovepoint:n {#2}
534   \tl_set_eq:NN #1 \g__spath_output_tl
535   \tl_gclear:N \g__spath_output_tl
536 }
537 \cs_generate_variant:Nn \spath_finalmovepoint:Nn {NV, cn, cV}
538 \cs_new_protected_nopar:Npn \spath_gfinalmovepoint:Nn #1#2
539 {
540   \__spath_finalmovepoint:n {#2}
541   \tl_gset_eq:NN #1 \g__spath_output_tl
542   \tl_gclear:N \g__spath_output_tl
543 }
544 \cs_generate_variant:Nn \spath_gfinalmovepoint:Nn {NV, cn, cV}

```

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

\spath_initialtangent:Nn
\spath_ginitialtangent:Nn

The starting tangent of the path.

```

545 \cs_new_protected_nopar:Npn \__spath_initialtangent:n #1
546 {
547   \group_begin:
548   \tl_set:Nx \l__spath_tmpb_tl {\tl_item:nn {#1} {4}}
549   \tl_if_eq:NNTF \l__spath_tmpb_tl \c_spath_curvetoa_tl
550   {
551     \fp_set:Nn \l__spath_tmpa_fp {3}
552   }
553   {
554     \fp_set:Nn \l__spath_tmpa_fp {1}
555   }
556   \tl_clear:N \l__spath_tmpa_tl
557   \tl_set:Nx \l__spath_tmpa_tl
558   {
559     {
560       \fp_to_dim:n {
561         \l__spath_tmpa_fp
562         *
563         (
564           \tl_item:nn {#1} {5}
565           -
566           \tl_item:nn {#1} {2}
567         )
568       }
569     }
570     {
571       \fp_to_dim:n {
572         \l__spath_tmpa_fp
573         *
574         (
575           \tl_item:nn {#1} {6}
576           -
577           \tl_item:nn {#1} {3}
578         )
579       }
580     }
581   }
582   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
583   \group_end:
584 }
585 \cs_new_protected_nopar:Npn \spath_initialtangent:Nn #1#2
586 {
587   \__spath_initialtangent:n {#2}
588   \tl_set_eq:NN #1 \g__spath_output_tl
589   \tl_gclear:N \g__spath_output_tl
590 }
591 \cs_generate_variant:Nn \spath_initialtangent:Nn {NV, cn, cV, Nv}
592 \cs_new_protected_nopar:Npn \spath_ginitialtangent:Nn #1#2
593 {
594   \__spath_initialtangent:n {#2}
595   \tl_gset_eq:NN #1 \g__spath_output_tl
596   \tl_gclear:N \g__spath_output_tl
597 }

```

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

```
\spath_finalltangent:Nn
\spath_gfinalltangent:Nn
```

The final tangent of the path.

```
599 \cs_new_protected_nopar:Npn \__spath_finalltangent:n #1
600 {
601   \group_begin:
602   \tl_set:Nn \l__spath_tmpa_tl {#1}
603   \tl_reverse:N \l__spath_tmpa_tl
604   \tl_set:Nx \l__spath_tmpb_tl {\tl_item:nn {#1} {6}}
605   \tl_if_eq:NNTF \l__spath_tmpb_tl \c_spath_curveto_tl
606   {
607     \fp_set:Nn \l__spath_tmpa_fp {3}
608   }
609   {
610     \fp_set:Nn \l__spath_tmpa_fp {1}
611   }
612   \tl_clear:N \l__spath_tmpb_tl
613   \tl_set:Nx \l__spath_tmpb_tl
614   {
615     {
616       \fp_to_dim:n {
617         \l__spath_tmpa_fp
618         *
619         (
620           \tl_item:Nn \l__spath_tmpa_tl {2}
621           -
622           \tl_item:Nn \l__spath_tmpa_tl {5}
623         )
624       }
625     }
626     {
627       \fp_to_dim:n {
628         \l__spath_tmpa_fp
629         *
630         (
631           \tl_item:Nn \l__spath_tmpa_tl {1}
632           -
633           \tl_item:Nn \l__spath_tmpa_tl {4}
634         )
635       }
636     }
637   }
638   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
639   \group_end:
640 }
641 \cs_new_protected_nopar:Npn \spath_finalltangent:Nn #1#2
642 {
643   \__spath_finalltangent:n {#2}
644   \tl_set_eq:NN #1 \g__spath_output_tl
645   \tl_gclear:N \g__spath_output_tl
646 }
647 \cs_generate_variant:Nn \spath_finalltangent:Nn {NV, cn, cV, Nv}
```

```

648 \cs_new_protected_nopar:Npn \spath_gfinaltangent:Nn #1#2
649 {
650   \__spath_finaltangent:n {#2}
651   \tl_gset_eq:NN #1 \g__spath_output_tl
652   \tl_gclear:N \g__spath_output_tl
653 }
654 \cs_generate_variant:Nn \spath_gfinaltangent:Nn {NV, cn, cV, Nv}

```

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

\spath_finalmovetangent:Nn Get the last move on the path.

\spath_gfinalmovetangent:Nn

```

655 \cs_new_protected_nopar:Npn \__spath_finalmovetangent:n #1
656 {
657   \group_begin:
658   \tl_set:Nn \l__spath_tmpc_tl { {0pt} {0pt} }
659   \tl_set:Nn \l__spath_tmpa_tl {#1}
660   \bool_do_until:n
661   {
662     \tl_if_empty_p:N \l__spath_tmpa_tl
663   }
664   {
665     \tl_set:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
666     \token_case_meaning:Nn \l__spath_tmpb_tl
667     {
668       \c_spath_moveto_tl
669       {
670         \tl_set:Nx \l__spath_tmpd_tl { \tl_item:Nn \l__spath_tmpa_tl {4} }
671         \tl_if_eq:NNTF \l__spath_tmpd_tl \c_spath_curveto_tl
672         {
673           \fp_set:Nn \l__spath_tmpa_fp {3}
674         }
675         {
676           \fp_set:Nn \l__spath_tmpa_fp {1}
677         }
678         \tl_set:Nx \l__spath_tmpc_tl
679         {
680           {
681             \fp_to_dim:n
682             {
683               \l__spath_tmpa_fp
684               *
685               (
686                 \tl_item:Nn \l__spath_tmpa_tl {5}
687                 -
688                 \tl_item:Nn \l__spath_tmpa_tl {2}
689               )
690             }
691           }
692           {
693             \fp_to_dim:n
694             {
695               \l__spath_tmpa_fp
696               *
697               (

```



```

698         \tl_item:Nn \l__spath_tmpa_tl {6}
699         -
700         \tl_item:Nn \l__spath_tmpa_tl {3}
701     )
702 }
703 }
704 }
705 }
706 }
707 \prg_replicate:nn {3}
708 {
709     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
710 }
711 }
712 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpc_tl
713 \group_end:
714 }
715 \cs_new_protected_nopar:Npn \spath_finalmovetangent:Nn #1#2
716 {
717     \__spath_finalmovetangent:n {#2}
718     \tl_set_eq:NN #1 \g__spath_output_tl
719     \tl_gclear:N \g__spath_output_tl
720 }
721 \cs_generate_variant:Nn \spath_finalmovetangent:Nn {NV, cn, cV}
722 \cs_new_protected_nopar:Npn \spath_gfinalmovetangent:Nn #1#2
723 {
724     \__spath_finalmovetangent:n {#2}
725     \tl_gset_eq:NN #1 \g__spath_output_tl
726     \tl_gclear:N \g__spath_output_tl
727 }
728 \cs_generate_variant:Nn \spath_gfinalmovetangent:Nn {NV, cn, cV}

```

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

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

```

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

```

```

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

```

```

802     }
803     \c_spath_rectcorner_tl {
804
805         \bool_if:NT \l__spath_closed_bool
806         {
807             \tl_put_right:NV \l__spath_tmpd_tl \c_spath_closepath_tl
808             \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpd_tl}
809             \tl_put_right:Nx \l__spath_tmpd_tl
810             {
811                 { \tl_head:N \l__spath_tmpd_tl }
812                 { \tl_head:N \l__spath_tmpe_tl }
813             }
814         }
815         \bool_set_false:N \l__spath_closed_bool
816         \tl_put_left:NV \l__spath_tmpd_tl \c_spath_moveto_tl
817         \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpd_tl
818         \tl_clear:N \l__spath_tmpd_tl
819
820         \bool_while_do:nn {
821             \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_rectcorner_tl
822         }
823         {
824             \tl_put_left:Nx \l__spath_tmpe_tl
825             {
826                 \tl_item:Nn \l__spath_tmpe_tl {1}
827                 {\tl_item:Nn \l__spath_tmpe_tl {2}}
828                 {\tl_item:Nn \l__spath_tmpe_tl {3}}
829                 \tl_item:Nn \l__spath_tmpe_tl {4}
830                 {\tl_item:Nn \l__spath_tmpe_tl {5}}
831                 {\tl_item:Nn \l__spath_tmpe_tl {6}}
832             }
833             \prg_replicate:nn {6}
834             {
835                 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
836             }
837             \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpe_tl}
838         }
839         \bool_set_true:N \l__spath_rect_bool
840
841     }
842     \c_spath_lineto_tl {
843         \tl_put_left:NV \l__spath_tmpd_tl \c_spath_lineto_tl
844     }
845     \c_spath_curveto_tl {
846         \tl_put_left:NV \l__spath_tmpd_tl \c_spath_curvetoa_tl
847     }
848     \c_spath_curvetoa_tl {
849         \tl_put_left:NV \l__spath_tmpd_tl \c_spath_curveto_tl
850     }
851     \c_spath_curvetob_tl {
852         \tl_put_left:NV \l__spath_tmpd_tl \c_spath_curvetob_tl
853     }
854 }
855 {

```

```

856     \tl_if_empty:NF \l__spath_tmpa_tl
857     {
858     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
859
860     \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
861     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
862     \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
863     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
864
865     \tl_put_left:Nx \l__spath_tmpd_tl
866     {
867     {\dim_use:N \l__spath_tmpa_dim}
868     {\dim_use:N \l__spath_tmpb_dim}
869     }
870     }
871   }
872   {
873   \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_closepath_tl
874   {
875     \bool_set_true:N \l__spath_closed_bool
876   }
877   {
878     \msg_warning:nxx
879     { spath3 }
880     { unknown path construction }
881     { \l__spath_tmpe_tl }
882   }
883
884   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
885   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
886   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
887
888   }
889 }
890
891 \bool_if:NT \l__spath_closed_bool
892 {
893   \tl_put_right:NV \l__spath_tmpd_tl \c_spath_closepath_tl
894   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpd_tl}
895   \tl_put_right:Nx \l__spath_tmpd_tl
896   {
897     { \tl_head:N \l__spath_tmpd_tl }
898     { \tl_head:N \l__spath_tmpe_tl }
899   }
900 }
901
902 \bool_set_false:N \l__spath_closed_bool
903 \bool_if:NF \l__spath_rect_bool
904 {
905   \tl_put_left:NV \l__spath_tmpd_tl \c_spath_moveto_tl
906 }
907 }
908
909 \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpd_tl

```

```

910 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
911 \group_end:
912 }
913 \cs_new_protected_nopar:Npn \spath_reverse:Nn #1#2
914 {
915   \__spath_reverse:n {#2}
916   \tl_set_eq:NN #1 \g__spath_output_tl
917   \tl_gclear:N \g__spath_output_tl
918 }
919 \cs_generate_variant:Nn \spath_reverse:Nn {NV, cn, cV, Nv}
920 \cs_new_protected_nopar:Npn \spath_reverse:N #1
921 {
922   \spath_reverse:NV #1#1
923 }
924 \cs_generate_variant:Nn \spath_reverse:N {c}
925 \cs_new_protected_nopar:Npn \spath_greverse:Nn #1#2
926 {
927   \__spath_reverse:n {#2}
928   \tl_gset_eq:NN #1 \g__spath_output_tl
929   \tl_gclear:N \g__spath_output_tl
930 }
931 \cs_generate_variant:Nn \spath_greverse:Nn {NV, cn, cV, Nv}
932 \cs_new_protected_nopar:Npn \spath_greverse:N #1
933 {
934   \spath_greverse:NV #1#1
935 }
936 \cs_generate_variant:Nn \spath_greverse:N {c}

```

(End of 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).

```

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

```

```

960 }
961 \cs_new_protected_nopar:Npn \spath_initialaction:Nn #1#2
962 {
963   \__spath_initialaction:n {#2}
964   \tl_set_eq:NN #1 \g__spath_output_tl
965   \tl_gclear:N \g__spath_output_tl
966 }
967 \cs_generate_variant:Nn \spath_initialaction:Nn {NV}
968 \cs_new_protected_nopar:Npn \spath_ginitialaction:Nn #1#2
969 {
970   \__spath_initialaction:n {#2}
971   \tl_gset_eq:NN #1 \g__spath_output_tl
972   \tl_gclear:N \g__spath_output_tl
973 }
974 \cs_generate_variant:Nn \spath_ginitialaction:Nn {NV}

```

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

\spath_finalaction:Nn This is the last thing that the path does.

```

\spath_gfinalaction:Nn
975 \cs_new_protected_nopar:Npn \__spath_finalaction:n #1
976 {
977   \group_begin:
978   \tl_clear:N \l__spath_tmpb_tl
979   \int_compare:nT
980   {
981     \tl_count:n {#1} > 3
982   }
983   {
984     \tl_set:Nn \l__spath_tmpa_tl {#1}
985     \tl_reverse:N \l__spath_tmpa_tl
986     \tl_set:Nx \l__spath_tmpb_tl
987     {
988       \tl_item:Nn \l__spath_tmpa_tl {3}
989     }
990     \tl_if_eq:NNT \l__spath_tmpb_tl \c_spath_curvetoa_tl
991     {
992       \tl_set_eq:NN \l__spath_tmpb_tl \c_spath_curveto_tl
993     }
994     \tl_if_eq:NNT \l__spath_tmpb_tl \c_spath_rectsize_tl
995     {
996       \tl_set_eq:NN \l__spath_tmpb_tl \c_spath_rectcorner_tl
997     }
998   }
999   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
1000   \group_end:
1001 }
1002 \cs_new_protected_nopar:Npn \spath_finalaction:Nn #1#2
1003 {
1004   \__spath_finalaction:n {#2}
1005   \tl_set_eq:NN #1 \g__spath_output_tl
1006   \tl_gclear:N \g__spath_output_tl
1007 }
1008 \cs_generate_variant:Nn \spath_finalaction:Nn {NV}
1009 \cs_new_protected_nopar:Npn \spath_gfinalaction:Nn #1#2

```

```

1010 {
1011   \_spath_finalaction:n {#2}
1012   \tl_gset_eq:NN #1 \g__spath_output_tl
1013   \tl_gclear:N \g__spath_output_tl
1014 }
1015 \cs_generate_variant:Nn \spath_gfinalaction:Nn {NV}

```

(End of 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
1016 \cs_new_protected_nopar:Npn \_spath_minbb:n #1
1017 {
1018   \group_begin:
1019   \tl_set:Nn \l__spath_tmpa_tl {#1}
1020
1021   \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
1022   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1023
1024   \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
1025   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1026
1027   \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
1028   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1029
1030   \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl
1031   {
1032     \dim_set_eq:NN \l__spath_rectx_dim \l__spath_tmpa_dim
1033     \dim_set_eq:NN \l__spath_recty_dim \l__spath_tmpb_dim
1034   }
1035   \bool_until_do:nn {
1036     \tl_if_empty_p:N \l__spath_tmpa_tl
1037   }
1038   {
1039     \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
1040     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1041
1042     \tl_if_eq:NNTF \l__spath_tmpc_tl \c_spath_rectsize_tl
1043     {
1044       \dim_set:Nn \l__spath_tmpa_dim
1045       {\dim_min:nn
1046         {\tl_head:N \l__spath_tmpa_tl + \l__spath_rectx_dim} {\l__spath_tmpa_dim}
1047       }
1048       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1049
1050       \dim_set:Nn \l__spath_tmpb_dim
1051       {\dim_min:nn
1052         {\tl_head:N \l__spath_tmpa_tl + \l__spath_recty_dim} {\l__spath_tmpb_dim}
1053       }
1054       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1055     }
1056     {
1057       \dim_set:Nn \l__spath_tmpa_dim
1058       {\dim_min:nn {\tl_head:N \l__spath_tmpa_tl} {\l__spath_tmpa_dim}}
1059       \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl

```

```

1060     {
1061     \dim_set:Nn \l__spath_rectx_dim {\tl_head:N \l__spath_tmpa_tl}
1062     }
1063     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1064
1065     \dim_set:Nn \l__spath_tmpb_dim
1066     {\dim_min:nn {\tl_head:N \l__spath_tmpa_tl} {\l__spath_tmpb_dim}}
1067     \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl
1068     {
1069     \dim_set:Nn \l__spath_recty_dim {\tl_head:N \l__spath_tmpa_tl}
1070     }
1071     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1072     }
1073
1074   }
1075   \tl_clear:N \l__spath_tmpb_tl
1076   \tl_put_right:Nx \l__spath_tmpb_tl
1077   {
1078   {\dim_use:N \l__spath_tmpa_dim}
1079   {\dim_use:N \l__spath_tmpb_dim}
1080   }
1081   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
1082   \group_end:
1083 }
1084 \cs_new_protected_nopar:Npn \spath_minbb:Nn #1#2
1085 {
1086   \__spath_minbb:n {#2}
1087   \tl_set_eq:NN #1 \g__spath_output_tl
1088   \tl_gclear:N \g__spath_output_tl
1089 }
1090 \cs_generate_variant:Nn \spath_minbb:Nn {NV, cn, cV}
1091 \cs_new_protected_nopar:Npn \spath_gminbb:Nn #1#2
1092 {
1093   \__spath_minbb:n {#2}
1094   \tl_gset_eq:NN #1 \g__spath_output_tl
1095   \tl_gclear:N \g__spath_output_tl
1096 }
1097 \cs_generate_variant:Nn \spath_gminbb:Nn {NV, cn, cV}

```

(End of 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
1098 \cs_new_protected_nopar:Npn \__spath_maxbb:n #1
1099 {
1100   \group_begin:
1101   \tl_set:Nn \l__spath_tmpa_tl {#1}
1102
1103   \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
1104   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1105
1106   \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
1107   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1108
1109   \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}

```



```

1110 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1111
1112 \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl
1113 {
1114   \dim_set_eq:NN \l__spath_rectx_dim \l__spath_tmpa_dim
1115   \dim_set_eq:NN \l__spath_recty_dim \l__spath_tmpb_dim
1116 }
1117 \bool_until_do:nn {
1118   \tl_if_empty_p:N \l__spath_tmpa_tl
1119 }
1120 {
1121   \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
1122   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1123
1124   \tl_if_eq:NNTF \l__spath_tmpc_tl \c_spath_rectsize_tl
1125   {
1126     \dim_set:Nn \l__spath_tmpa_dim
1127     {\dim_max:nn
1128      {\tl_head:N \l__spath_tmpa_tl + \l__spath_rectx_dim} {\l__spath_tmpa_dim}}
1129   }
1130   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1131
1132   \dim_set:Nn \l__spath_tmpb_dim
1133   {\dim_max:nn
1134    {\tl_head:N \l__spath_tmpa_tl + \l__spath_recty_dim} {\l__spath_tmpb_dim}}
1135   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1136 }
1137 {
1138   {
1139     \dim_set:Nn \l__spath_tmpa_dim
1140     {\dim_max:nn {\tl_head:N \l__spath_tmpa_tl} {\l__spath_tmpa_dim}}
1141     \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl
1142     {
1143       \dim_set:Nn \l__spath_rectx_dim {\tl_head:N \l__spath_tmpa_tl}
1144     }
1145     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1146
1147     \dim_set:Nn \l__spath_tmpb_dim
1148     {\dim_max:nn {\tl_head:N \l__spath_tmpa_tl} {\l__spath_tmpb_dim}}
1149     \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl
1150     {
1151       \dim_set:Nn \l__spath_recty_dim {\tl_head:N \l__spath_tmpa_tl}
1152     }
1153     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1154   }
1155 }
1156 }
1157 \tl_clear:N \l__spath_tmpb_tl
1158 \tl_set:Nx \l__spath_tmpb_tl
1159 {
1160   {\dim_use:N \l__spath_tmpa_dim}
1161   {\dim_use:N \l__spath_tmpb_dim}
1162 }
1163 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl

```

```

1164 \group_end:
1165 }
1166 \cs_new_protected_nopar:Npn \spath_maxbb:Nn #1#2
1167 {
1168   \__spath_maxbb:n {#2}
1169   \tl_set_eq:NN #1 \g__spath_output_tl
1170   \tl_gclear:N \g__spath_output_tl
1171 }
1172 \cs_generate_variant:Nn \spath_maxbb:Nn {NV, cn, cV}
1173 \cs_new_protected_nopar:Npn \spath_gmaxbb:Nn #1#2
1174 {
1175   \__spath_maxbb:n {#2}
1176   \tl_gset_eq:NN #1 \g__spath_output_tl
1177   \tl_gclear:N \g__spath_output_tl
1178 }
1179 \cs_generate_variant:Nn \spath_gmaxbb:Nn {NV, cn, cV}

```

(End of 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
\spath_save_to_aux:N
1180 \int_set:Nn \l__spath_tmpa_int {\char_value_catcode:n {'@}}
1181 \char_set_catcode_letter:N @
1182 \cs_new_protected_nopar:Npn \spath_save_to_aux:Nn #1#2 {
1183   \tl_if_empty:nF {#2}
1184   {
1185     \tl_clear:N \l__spath_tmpa_tl
1186     \tl_put_right:Nn \l__spath_tmpa_tl {
1187       \ExplSyntaxOn
1188       \tl_gclear_new:N #1
1189       \tl_gset:Nn #1 {#2}
1190       \ExplSyntaxOff
1191     }
1192     \protected@write\@auxout{}\{
1193       \tl_to_str:N \l__spath_tmpa_tl
1194     }
1195   }
1196 }
1197 \char_set_catcode:nn {'@} {\l__spath_tmpa_int}
1198 \cs_generate_variant:Nn \spath_save_to_aux:Nn {cn, cV, NV}
1199 \cs_new_protected_nopar:Npn \spath_save_to_aux:N #1
1200 {
1201   \tl_if_exist:NT #1
1202   {
1203     \spath_save_to_aux:NV #1#1
1204   }
1205 }
1206 \cs_generate_variant:Nn \spath_save_to_aux:N {c}

```

(End of 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 1207 \cs_new_protected_nopar:Npn \__spath_translate:nnn #1#2#3
\spath_gtranslate:Nnnn 1208 {
\spath_gtranslate:Nnn 1209   \group_begin:
1210   \tl_set:Nn \l__spath_tmpa_tl {#1}
1211   \tl_clear:N \l__spath_tmpb_tl
1212   \bool_until_do:nn {
1213     \tl_if_empty_p:N \l__spath_tmpa_tl
1214   }
1215   {
1216     \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
1217
1218     \tl_put_right:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
1219     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1220
1221     \tl_if_eq:NNTF \l__spath_tmpc_tl \c_spath_rectsize_tl
1222     {
1223       \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
1224     }
1225     {
1226       \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl + #2}
1227     }
1228     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1229
1230     \tl_if_eq:NNTF \l__spath_tmpc_tl \c_spath_rectsize_tl
1231     {
1232       \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
1233     }
1234     {
1235       \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl + #3}
1236     }
1237     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1238
1239     \tl_put_right:Nx \l__spath_tmpb_tl
1240     {
1241       {\dim_use:N \l__spath_tmpa_dim}
1242       {\dim_use:N \l__spath_tmpb_dim}
1243     }
1244   }
1245   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
1246   \group_end:
1247 }
1248 \cs_generate_variant:Nn \__spath_translate:nnn {nVV}
1249 \cs_new_protected_nopar:Npn \spath_translate:Nnnn #1#2#3#4
1250 {
1251   \__spath_translate:nnn {#2}{#3}{#4}
1252   \tl_set_eq:NN #1 \g__spath_output_tl
1253   \tl_gclear:N \g__spath_output_tl
1254 }
1255 \cs_generate_variant:Nn \spath_translate:Nnnn {NVxx, NVVV, NVnn}

```

```

1256 \cs_new_protected_nopar:Npn \spath_translate:Nnn #1#2#3
1257 {
1258   \spath_translate:NVnn #1#1{#2}{#3}
1259 }
1260 \cs_generate_variant:Nn \spath_translate:Nnn {NVV, cnn, cVV}
1261 \cs_new_protected_nopar:Npn \spath_gtranslate:Nnnn #1#2#3#4
1262 {
1263   \__spath_translate:nnn {#2}{#3}{#4}
1264   \tl_gset_eq:NN #1 \g__spath_output_tl
1265   \tl_gclear:N \g__spath_output_tl
1266 }
1267 \cs_generate_variant:Nn \spath_gtranslate:Nnnn {NVxx, NVVV, NVnn}
1268 \cs_new_protected_nopar:Npn \spath_gtranslate:Nnn #1#2#3
1269 {
1270   \spath_gtranslate:NVnn #1#1{#2}{#3}
1271 }
1272 \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.

```

1273 \cs_new_protected_nopar:Npn \spath_translate:Nn #1#2
1274 {
1275   \spath_translate:Nnn #1 #2
1276 }
1277 \cs_generate_variant:Nn \spath_translate:Nn {NV}
1278 \cs_new_protected_nopar:Npn \spath_gtranslate:Nn #1#2
1279 {
1280   \spath_gtranslate:Nnn #1 #2
1281 }
1282 \cs_generate_variant:Nn \spath_gtranslate:Nn {NV}

```

(End of 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 1283 \cs_new_protected_nopar:Npn \__spath_translate_to:nnn #1#2#3
\spath_gtranslate_to:Nnnn 1284 {
\spath_gtranslate_to:Nnn 1285   \group_begin:
1286   \spath_initialpoint:Nn \l__spath_tmpa_tl {#1}
1287
1288   \dim_set:Nn \l__spath_tmpa_dim
1289   {
1290     #2
1291     -
1292     \tl_item:Nn \l__spath_tmpa_tl {1}
1293   }
1294   \dim_set:Nn \l__spath_tmpb_dim
1295   {
1296     #3
1297     -
1298     \tl_item:Nn \l__spath_tmpa_tl {2}
1299   }
1300
1301   \__spath_translate:nVV {#1} \l__spath_tmpa_dim \l__spath_tmpb_dim
1302   \group_end:
1303 }

```

```

1304 \cs_new_protected_nopar:Npn \spath_translate_to:Nnnn #1#2#3#4
1305 {
1306   \__spath_translate_to:nnn {#2}{#3}{#4}
1307   \tl_set_eq:NN #1 \g__spath_output_tl
1308   \tl_gclear:N \g__spath_output_tl
1309 }
1310 \cs_generate_variant:Nn \spath_translate_to:Nnnn {NVxx, NVVV, NVnn}
1311 \cs_new_protected_nopar:Npn \spath_translate_to:Nnn #1#2#3
1312 {
1313   \spath_translate_to:NVnn #1#1{#2}{#3}
1314 }
1315 \cs_generate_variant:Nn \spath_translate_to:Nnn {NVV, cnn, cVV}
1316 \cs_new_protected_nopar:Npn \spath_gtranslate_to:Nnnn #1#2#3#4
1317 {
1318   \__spath_translate_to:nnn {#2}{#3}{#4}
1319   \tl_gset_eq:NN #1 \g__spath_output_tl
1320   \tl_gclear:N \g__spath_output_tl
1321 }
1322 \cs_generate_variant:Nn \spath_gtranslate_to:Nnnn {NVxx, NVVV, NVnn}
1323 \cs_new_protected_nopar:Npn \spath_gtranslate_to:Nnn #1#2#3
1324 {
1325   \spath_gtranslate_to:NVnn #1#1{#2}{#3}
1326 }
1327 \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.

```

1328 \cs_new_protected_nopar:Npn \spath_translate_to:Nn #1#2
1329 {
1330   \spath_translate_to:Nnn #1 #2
1331 }
1332 \cs_generate_variant:Nn \spath_translate_to:Nn {NV}
1333 \cs_new_protected_nopar:Npn \spath_gtranslate_to:Nn #1#2
1334 {
1335   \spath_gtranslate_to:Nnn #1 #2
1336 }
1337 \cs_generate_variant:Nn \spath_gtranslate_to:Nn {NV}

```

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

```

\spath_scale:Nnnn Scale a path.
\spath_scale:Nnn 1338 \cs_new_protected_nopar:Npn \__spath_scale:nnn #1#2#3
\spath_gscale:Nnnn 1339 {
\spath_gscale:Nnn 1340   \group_begin:
1341   \tl_set:Nn \l__spath_tmpa_tl {#1}
1342   \tl_clear:N \l__spath_tmpb_tl
1343   \bool_until_do:nn {
1344     \tl_if_empty_p:N \l__spath_tmpa_tl
1345   }
1346   {
1347     \tl_put_right:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl}
1348     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1349
1350     \fp_set:Nn \l__spath_tmpa_fp {\tl_head:N \l__spath_tmpa_tl * #2}
1351     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}

```

```

1352
1353     \fp_set:Nn \l__spath_tmpb_fp {\tl_head:N \l__spath_tmpa_tl * #3}
1354     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
1355
1356     \tl_put_right:Nx \l__spath_tmpb_tl
1357     {
1358         {\fp_to_dim:N \l__spath_tmpa_fp}
1359         {\fp_to_dim:N \l__spath_tmpb_fp}
1360     }
1361 }
1362 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
1363 \group_end:
1364 }
1365 \cs_new_protected_nopar:Npn \spath_scale:Nnnn #1#2#3#4
1366 {
1367     \__spath_scale:nnn {#2}{#3}{#4}
1368     \tl_set_eq:NN #1 \g__spath_output_tl
1369     \tl_gclear:N \g__spath_output_tl
1370 }
1371 \cs_generate_variant:Nn \spath_scale:Nnnn {NVnn, Nnxx}
1372 \cs_new_protected_nopar:Npn \spath_scale:Nnn #1#2#3
1373 {
1374     \spath_scale:NVnn #1#1{#2}{#3}
1375 }
1376 \cs_generate_variant:Nn \spath_scale:Nnn {cnn, cVV, NVV}
1377 \cs_new_protected_nopar:Npn \spath_gscale:Nnnn #1#2#3#4
1378 {
1379     \__spath_scale:nnn {#2}{#3}{#4}
1380     \tl_gset_eq:NN #1 \g__spath_output_tl
1381     \tl_gclear:N \g__spath_output_tl
1382 }
1383 \cs_generate_variant:Nn \spath_gscale:Nnnn {NVnn, Nnxx}
1384 \cs_new_protected_nopar:Npn \spath_gscale:Nnn #1#2#3
1385 {
1386     \spath_gscale:NVnn #1#1{#2}{#3}
1387 }
1388 \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.

```

1389 \cs_new_protected_nopar:Npn \spath_scale:Nn #1#2
1390 {
1391     \spath_scale:Nnn #1 #2
1392 }
1393
1394 \cs_generate_variant:Nn \spath_scale:Nn {NV}
1395 \cs_new_protected_nopar:Npn \spath_gscale:Nn #1#2
1396 {
1397     \spath_gscale:Nnn #1 #2
1398 }
1399
1400 \cs_generate_variant:Nn \spath_gscale:Nn {NV}

```

(End of 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
1401 \cs_new_protected_nopar:Npn \__spath_transform:nnnnnnn #1#2#3#4#5#6#7
1402 {
1403   \group_begin:
1404   \tl_set:Nn \l__spath_tmpa_tl {#1}
1405   \tl_clear:N \l__spath_tmpb_tl
1406   \bool_until_do:nn {
1407     \tl_if_empty_p:N \l__spath_tmpa_tl
1408   }
1409   {
1410     \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpa_tl}
1411
1412     \tl_put_right:Nx \l__spath_tmpb_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     \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpa_tl}
1417     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpa_tl}
1418
1419     \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_rectsize_tl
1420     {
1421       \fp_set:Nn \l__spath_tmpe_fp
1422       {\l__spath_tmpe_tl * #2 + \l__spath_tmpe_tl * #4}
1423       \fp_set:Nn \l__spath_tmpe_fp
1424       {\l__spath_tmpe_tl * #3 + \l__spath_tmpe_tl * #5}
1425     }
1426     {
1427       \fp_set:Nn \l__spath_tmpe_fp
1428       {\l__spath_tmpe_tl * #2 + \l__spath_tmpe_tl * #4 + #6}
1429       \fp_set:Nn \l__spath_tmpe_fp
1430       {\l__spath_tmpe_tl * #3 + \l__spath_tmpe_tl * #5 + #7}
1431     }
1432
1433     \tl_put_right:Nx \l__spath_tmpb_tl
1434     {
1435       {\fp_to_dim:N \l__spath_tmpe_fp}
1436       {\fp_to_dim:N \l__spath_tmpe_fp}
1437     }
1438   }
1439
1440   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
1441   \group_end:
1442 }
1443 \cs_new_protected_nopar:Npn \spath_transform:Nnnnnnnn #1#2#3#4#5#6#7#8
1444 {
1445   \__spath_transform:nnnnnnn {#2}{#3}{#4}{#5}{#6}{#7}{#8}
1446   \tl_set_eq:NN #1 \g__spath_output_tl
1447   \tl_gclear:N \g__spath_output_tl
1448 }
1449 \cs_generate_variant:Nn \spath_transform:Nnnnnnnn
1450 {NVnnnnnnn, Nxxxxxxx, cnnnnnnn}
1451 \cs_new_protected_nopar:Npn \spath_transform:Nnnnnnnn #1#2#3#4#5#6#7
1452 {

```

```

1453 \spath_transform:NVnnnnnn #1#1{#2}{#3}{#4}{#5}{#6}{#7}
1454 }
1455 \cs_generate_variant:Nn \spath_transform:Nnnnnnn {cnnnnnn}
1456 \cs_new_protected_nopar:Npn \spath_transform:Nnn #1#2#3
1457 {
1458   \spath_transform:Nnnnnnn #1{#2}#3
1459 }
1460 \cs_generate_variant:Nn \spath_transform:Nnn {cnn, cVn, NVn, NnV}
1461 \cs_new_protected_nopar:Npn \spath_transform:Nn #1#2
1462 {
1463   \spath_transform:NVnnnnnn #1#1#2
1464 }
1465 \cs_generate_variant:Nn \spath_transform:Nn {cn, cV, NV}
1466
1467 \cs_new_protected_nopar:Npn \spath_gtransform:Nnnnnnn #1#2#3#4#5#6#7#8
1468 {
1469   \__spath_transform:nnnnnn {#2}{#3}{#4}{#5}{#6}{#7}{#8}
1470   \tl_gset_eq:NN #1 \g__spath_output_tl
1471   \tl_gclear:N \g__spath_output_tl
1472 }
1473 \cs_generate_variant:Nn \spath_gtransform:Nnnnnnn {NVnnnnnn, Nnxxxxxx, cnnnnnn}
1474 \cs_new_protected_nopar:Npn \spath_gtransform:Nnnnnnn #1#2#3#4#5#6#7
1475 {
1476   \spath_gtransform:NVnnnnnn #1#1{#2}{#3}{#4}{#5}{#6}{#7}
1477 }
1478 \cs_generate_variant:Nn \spath_gtransform:Nnnnnnn {cnnnnnn}
1479 \cs_new_protected_nopar:Npn \spath_gtransform:Nnn #1#2#3
1480 {
1481   \spath_gtransform:Nnnnnnn #1{#2}#3
1482 }
1483 \cs_generate_variant:Nn \spath_gtransform:Nnn {cnn, cVn, NVn, NnV}
1484 \cs_new_protected_nopar:Npn \spath_gtransform:Nn #1#2
1485 {
1486   \spath_gtransform:NVnnnnnn #1#1#2
1487 }
1488 \cs_generate_variant:Nn \spath_gtransform:Nn {cn, cV, NV}

```

(End of 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> 1489 \cs_new_protected_nopar:Npn __spath_span:nnn #1#2#3 1490 { 1491 \group_begin: 1492 \spath_initialpoint:Nn \l__spath_tmpa_tl {#1} 1493 \spath_finalpoint:Nn \l__spath_tmpb_tl {#1} 1494 1495 \fp_set:Nn \l__spath_tmpa_fp 1496 { 1497 (\tl_item:Nn \l__spath_tmpb_tl {1}) - 1498 (\tl_item:Nn \l__spath_tmpa_tl {1}) 1499 } </pre>
--	---


```

1500 \fp_set:Nn \l__spath_tmpb_fp
1501 {
1502   (\tl_item:Nn \l__spath_tmpb_tl {2}) -
1503   (\tl_item:Nn \l__spath_tmpa_tl {2})
1504 }
1505 \fp_set:Nn \l__spath_tmpc_fp
1506 {
1507   (\l__spath_tmpa_fp) * (\l__spath_tmpa_fp)
1508   +
1509   (\l__spath_tmpb_fp * \l__spath_tmpb_fp)
1510 }
1511
1512 \fp_compare:nTF
1513 {
1514   \l__spath_tmpc_fp < 0.001
1515 }
1516 {
1517   \spath_translate_to:Nnnn \l__spath_tmpd_tl {#1} #2
1518 }
1519 {
1520   \fp_set:Nn \l__spath_tmpa_fp
1521   {
1522     (
1523       ((\tl_item:nn {#3} {1})
1524       -
1525       (\tl_item:nn {#2} {1}))
1526     *
1527     ((\tl_item:Nn \l__spath_tmpb_tl {1})
1528     -
1529     (\tl_item:Nn \l__spath_tmpa_tl {1}))
1530     +
1531     ((\tl_item:nn {#3} {2})
1532     -
1533     (\tl_item:nn {#2} {2}))
1534     *
1535     ((\tl_item:Nn \l__spath_tmpb_tl {2})
1536     -
1537     (\tl_item:Nn \l__spath_tmpa_tl {2}))
1538     )
1539     /
1540     \l__spath_tmpc_fp
1541   }
1542   \fp_set:Nn \l__spath_tmpb_fp
1543   {
1544     (
1545       ((\tl_item:nn {#3} {2})
1546       -
1547       (\tl_item:nn {#2} {2}))
1548     *
1549     ((\tl_item:Nn \l__spath_tmpb_tl {1})
1550     -
1551     (\tl_item:Nn \l__spath_tmpa_tl {1}))
1552     -
1553     ((\tl_item:nn {#3} {1})

```

```

1554     -
1555     (\tl_item:nn {#2} {1}))
1556     *
1557     ((\tl_item:Nn \l__spath_tmpb_tl {2})
1558     -
1559     (\tl_item:Nn \l__spath_tmpa_tl {2}))
1560     )
1561     /
1562     \l__spath_tmpc_fp
1563 }
1564
1565 \tl_set:Nx \l__spath_tmpc_tl
1566 {
1567   {
1568     \fp_to_decimal:N \l__spath_tmpa_fp
1569   }
1570   {
1571     \fp_to_decimal:N \l__spath_tmpb_fp
1572   }
1573   {
1574     \fp_eval:n { - \l__spath_tmpb_fp }
1575   }
1576   {
1577     \fp_to_decimal:N \l__spath_tmpa_fp
1578   }
1579   {
1580     \fp_to_dim:n
1581     {
1582       \tl_item:nn {#2} {1}
1583       -
1584       \l__spath_tmpa_fp * (\tl_item:Nn \l__spath_tmpa_tl {1})
1585       +
1586       \l__spath_tmpb_fp * (\tl_item:Nn \l__spath_tmpa_tl {2})
1587     }
1588   }
1589   {
1590     \fp_to_dim:n
1591     {
1592       \tl_item:nn {#2} {2}
1593       -
1594       \l__spath_tmpb_fp * (\tl_item:Nn \l__spath_tmpa_tl {1})
1595       -
1596       \l__spath_tmpa_fp * (\tl_item:Nn \l__spath_tmpa_tl {2})
1597     }
1598   }
1599 }
1600 \spath_transform:NnV \l__spath_tmpd_tl {#1} \l__spath_tmpc_tl
1601 }
1602 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpd_tl
1603 \group_end:
1604 }
1605 \cs_new_protected_nopar:Npn \spath_span:Nnnn #1#2#3#4
1606 {
1607   \__spath_span:nnn {#2}{#3}{#4}

```

```

1608 \tl_set_eq:NN #1 \g__spath_output_tl
1609 \tl_gclear:N \g__spath_output_tl
1610 }
1611 \cs_generate_variant:Nn \spath_span:Nnnn {NVnn, NVVV, NnVV}
1612 \cs_new_protected_nopar:Npn \spath_span:Nnn #1#2#3
1613 {
1614   \spath_span:NVnn #1#1{#2}{#3}
1615 }
1616 \cs_generate_variant:Nn \spath_span:Nnn {NVV, cnn, cvv, cVV}
1617 \cs_new_protected_nopar:Npn \spath_gspan:Nnnn #1#2#3#4
1618 {
1619   \__spath_span:nnn {#2}{#3}{#4}
1620   \tl_gset_eq:NN #1 \g__spath_output_tl
1621   \tl_gclear:N \g__spath_output_tl
1622 }
1623 \cs_generate_variant:Nn \spath_gspan:Nnnn {NVnn, NVVV}
1624 \cs_new_protected_nopar:Npn \spath_gspan:Nnn #1#2#3
1625 {
1626   \spath_gspan:NVnn #1#1{#2}{#3}
1627 }
1628 \cs_generate_variant:Nn \spath_gspan:Nnn {NVV, cnn, cvv, cVV}
1629 \cs_new_protected_nopar:Npn \__spath_normalise:n #1
1630 {
1631   \__spath_span:nnn {#1}{0pt}{0pt}{{1pt}{0pt}}
1632 }
1633 \cs_new_protected_nopar:Npn \spath_normalise:Nn #1#2
1634 {
1635   \__spath_normalise:n {#2}
1636   \tl_set_eq:NN #1 \g__spath_output_tl
1637   \tl_gclear:N \g__spath_output_tl
1638 }
1639 \cs_generate_variant:Nn \spath_normalise:Nn {cn,NV, cV, cv}
1640 \cs_new_protected_nopar:Npn \spath_normalise:N #1
1641 {
1642   \spath_normalise:NV #1#1
1643 }
1644 \cs_generate_variant:Nn \spath_normalise:N {c}
1645 \cs_new_protected_nopar:Npn \spath_gnormalise:Nn #1#2
1646 {
1647   \__spath_normalise:n {#2}
1648   \tl_gset_eq:NN #1 \g__spath_output_tl
1649   \tl_gclear:N \g__spath_output_tl
1650 }
1651 \cs_generate_variant:Nn \spath_gnormalise:Nn {cn,NV, cV, cv}
1652 \cs_new_protected_nopar:Npn \spath_gnormalise:N #1
1653 {
1654   \spath_gnormalise:NV #1#1
1655 }
1656 \cs_generate_variant:Nn \spath_gnormalise:N {c}

```

(End of 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

```

1657 \cs_new_protected_nopar:Npn \__spath_splice_between:nnn #1#2#3
1658 {
1659   \group_begin:
1660   \spath_finalpoint:Nn \l__spath_tmpd_tl {#1}
1661   \spath_initialpoint:Nn \l__spath_tmpe_tl {#3}
1662   \spath_span:NnVV \l__spath_tmpb_tl {#2} \l__spath_tmpd_tl \l__spath_tmpe_tl
1663   \spath_append_no_move:NnV \l__spath_tmpe_tl {#1} \l__spath_tmpb_tl
1664   \spath_append_no_move:Nn \l__spath_tmpe_tl {#3}
1665   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
1666   \group_end:
1667 }
1668 \cs_new_protected_nopar:Npn \spath_splice_between:Nnnn #1#2#3#4
1669 {
1670   \__spath_splice_between:nnn {#2}{#3}{#4}
1671   \tl_set_eq:NN #1 \g__spath_output_tl
1672   \tl_gclear:N \g__spath_output_tl
1673 }
1674 \cs_generate_variant:Nn \spath_splice_between:Nnnn {NVnn, NVVV}
1675 \cs_new_protected_nopar:Npn \spath_splice_between:Nnn #1#2#3
1676 {
1677   \spath_splice_between:NVnn #1#1{#2}{#3}
1678 }
1679 \cs_generate_variant:Nn \spath_splice_between:Nnn {NVV, cnn, cvv, Nvn, NVn}
1680 \cs_new_protected_nopar:Npn \spath_gsplice_between:Nnnn #1#2#3#4
1681 {
1682   \__spath_splice_between:nnn {#2}{#3}{#4}
1683   \tl_gset_eq:NN #1 \g__spath_output_tl
1684   \tl_gclear:N \g__spath_output_tl
1685 }
1686 \cs_generate_variant:Nn \spath_gsplice_between:Nnnn {NVnn, NVVV}
1687 \cs_new_protected_nopar:Npn \spath_gsplice_between:Nnn #1#2#3
1688 {
1689   \spath_gsplice_between:NVnn #1#1{#2}{#3}
1690 }
1691 \cs_generate_variant:Nn \spath_gsplice_between:Nnn {NVV, cnn, cvv, Nvn, NVn}

```

(End of 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.

```

1692 \cs_new_protected_nopar:Npn \__spath_hobby_curve:nnnn #1#2#3#4
1693 {
1694   \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 ...

```

1695   \fp_set:Nn \l__spath_tmpe_fp
1696   {
1697     (
1698       (\tl_item:nn {#2} {1})
1699       *
1700       (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})
1701       +
1702       (\tl_item:nn {#2} {2})
1703       *

```

```

1704      (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})
1705      )
1706      /
1707      sqrt
1708      (
1709      (
1710      (\tl_item:nn {#2} {1})^2
1711      +
1712      (\tl_item:nn {#2} {2})^2
1713      )
1714      *
1715      (
1716      (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})^2
1717      +
1718      (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})^2
1719      )
1720      )
1721      }
1722      \fp_set:Nn \l__spath_tmpb_fp
1723      {
1724      (
1725      -
1726      (\tl_item:nn {#2} {1})
1727      *
1728      (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})
1729      +
1730      (\tl_item:nn {#2} {2})
1731      *
1732      (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})
1733      )
1734      /
1735      sqrt
1736      (
1737      (
1738      (\tl_item:nn {#2} {1})^2
1739      +
1740      (\tl_item:nn {#2} {2})^2
1741      )
1742      *
1743      (
1744      (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})^2
1745      +
1746      (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})^2
1747      )
1748      )
1749      }

```

Second tangent vector projected onto vector between endpoints

```

1750      \fp_set:Nn \l__spath_tmpe_fp
1751      {
1752      (
1753      (\tl_item:nn {#3} {1})
1754      *
1755      (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})
1756      +

```

```

1757 (\tl_item:nn {#3} {2})
1758 *
1759 (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})
1760 )
1761 /
1762 sqrt
1763 (
1764 (
1765 (\tl_item:nn {#3} {1})^2
1766 +
1767 (\tl_item:nn {#3} {2})^2
1768 )
1769 *
1770 (
1771 (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})^2
1772 +
1773 (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})^2
1774 )
1775 )
1776 }
1777 \fp_set:Nn \l__spath_tmpd_fp
1778 {
1779 (
1780 (\tl_item:nn {#3} {1})
1781 *
1782 (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})
1783 -
1784 (\tl_item:nn {#3} {2})
1785 *
1786 (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})
1787 )
1788 /
1789 sqrt
1790 (
1791 (
1792 (\tl_item:nn {#3} {1})^2
1793 +
1794 (\tl_item:nn {#3} {2})^2
1795 )
1796 *
1797 (
1798 (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})^2
1799 +
1800 (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})^2
1801 )
1802 )
1803 }
1804
1805 \fp_set:Nn \l__spath_tmpe_fp
1806 {
1807 (
1808 2 + sqrt(2) *
1809 (\l__spath_tmpe_fp - 1/16 * \l__spath_tmpd_fp)
1810 *

```

```

1811 (\l__spath_tmpd_fp - 1/16 * \l__spath_tmpb_fp)
1812 *
1813 (\l__spath_tmpa_fp - \l__spath_tmpc_fp)
1814 )
1815 /
1816 (
1817 1
1818 +
1819 (1 - (3 - sqrt(5))/2)
1820 *
1821 \l__spath_tmpa_fp
1822 +
1823 (3 - sqrt(5))/2
1824 *
1825 \l__spath_tmpc_fp
1826 )
1827 *
1828 sqrt
1829 (
1830 (
1831 (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})^2
1832 +
1833 (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})^2
1834 )
1835 /
1836 (
1837 (\tl_item:nn {#2} {1})^2
1838 +
1839 (\tl_item:nn {#2} {2})^2
1840 )
1841 )
1842 /3
1843 }
1844 \fp_set:Nn \l__spath_tmpf_fp
1845 {
1846 (
1847 2 - sqrt(2) *
1848 (\l__spath_tmpb_fp - 1/16 * \l__spath_tmpd_fp)
1849 *
1850 (\l__spath_tmpd_fp - 1/16 * \l__spath_tmpb_fp)
1851 *
1852 (\l__spath_tmpa_fp - \l__spath_tmpc_fp)
1853 )
1854 /
1855 (
1856 1
1857 +
1858 (1 - (3 - sqrt(5))/2)
1859 *
1860 \l__spath_tmpc_fp
1861 +
1862 (3 - sqrt(5))/2
1863 *
1864 \l__spath_tmpa_fp

```

```

1865 )
1866 *
1867 sqrt
1868 (
1869 (
1870 (\tl_item:nn {#4} {1} - \tl_item:nn {#1} {1})^2
1871 +
1872 (\tl_item:nn {#4} {2} - \tl_item:nn {#1} {2})^2
1873 )
1874 /
1875 (
1876 (\tl_item:nn {#3} {1})^2
1877 +
1878 (\tl_item:nn {#3} {2})^2
1879 )
1880 )
1881 /3
1882 }
1883
1884 \tl_set:Nx \l__spath_tmpa_tl
1885 {
1886 {
1887 \fp_to_dim:n
1888 {
1889 \tl_item:nn {#1} {1}
1890 +
1891 \l__spath_tmpe_fp
1892 *
1893 (\tl_item:nn {#2} {1})
1894 }
1895 }
1896 {
1897 \fp_to_dim:n
1898 {
1899 \tl_item:nn {#1} {2}
1900 +
1901 \l__spath_tmpe_fp
1902 *
1903 (\tl_item:nn {#2} {2})
1904 }
1905 }
1906 }
1907 \tl_set:Nx \l__spath_tmpb_tl
1908 {
1909 {
1910 \fp_to_dim:n
1911 {
1912 \tl_item:nn {#4} {1}
1913 -
1914 \l__spath_tmpf_fp
1915 *
1916 (\tl_item:nn {#3} {1})
1917 }
1918 }

```



```

1919     {
1920       \fp_to_dim:n
1921       {
1922         \tl_item:nn {#4} {2}
1923         -
1924         \l__spath_tmpf_fp
1925         *
1926         (\tl_item:nn {#3} {2})
1927       }
1928     }
1929   }
1930
1931   \tl_clear:N \l__spath_tmpe_tl
1932   \tl_set:NV \l__spath_tmpe_tl \c_spath_moveto_tl
1933   \tl_put_right:Nn \l__spath_tmpe_tl {#1}
1934   \tl_put_right:NV \l__spath_tmpe_tl \c_spath_curvetoa_tl
1935   \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpea_tl
1936   \tl_put_right:NV \l__spath_tmpe_tl \c_spath_curvetob_tl
1937   \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpeb_tl
1938   \tl_put_right:NV \l__spath_tmpe_tl \c_spath_curveto_tl
1939   \tl_put_right:Nn \l__spath_tmpe_tl {#4}
1940   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
1941   \group_end:
1942 }
1943 \cs_new_protected_nopar:Npn \spath_hobby_curve:Nnnnn #1#2#3#4#5
1944 {
1945   \__spath_hobby_curve:nnnn {#2}{#3}{#4}{#5}
1946   \tl_set_eq:NN #1 \g__spath_output_tl
1947   \tl_gclear:N \g__spath_output_tl
1948 }
1949 \cs_generate_variant:Nn \spath_hobby_curve:Nnnnn {NVVVV}
1950 \cs_new_protected_nopar:Npn \spath_ghobby_curve:Nnnnn #1#2#3#4#5
1951 {
1952   \__spath_hobby_curve:nnnn {#2}{#3}{#4}{#5}
1953   \tl_gset_eq:NN #1 \g__spath_output_tl
1954   \tl_gclear:N \g__spath_output_tl
1955 }
1956 \cs_generate_variant:Nn \spath_ghobby_curve:Nnnnn {NVVVV}

```

(End of 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.

```

1957 \cs_new_protected_nopar:Npn \__spath_curve_between:nn #1#2
1958 {
1959   \group_begin:
1960   \spath_finalpoint:Nn \l__spath_tmpea_tl {#1}
1961   \spath_finalgangent:Nn \l__spath_tmpeb_tl {#1}
1962   \spath_initialpoint:Nn \l__spath_tmpec_tl {#2}
1963   \spath_initialtangent:Nn \l__spath_tmped_tl {#2}
1964
1965   \spath_hobby_curve:NVVVV \l__spath_tmpea_tl
1966   \l__spath_tmpea_tl \l__spath_tmpeb_tl \l__spath_tmpec_tl \l__spath_tmpec_tl
1967

```

```

1968 \tl_set:Nn \l__spath_tmpa_tl {#1}
1969 \spath_append_no_move:NV \l__spath_tmpa_tl \l__spath_tmpe_tl
1970 \spath_append_no_move:Nn \l__spath_tmpa_tl {#2}
1971 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
1972 \group_end:
1973 }
1974 \cs_new_protected_nopar:Npn \spath_curve_between:Nnn #1#2#3
1975 {
1976   \__spath_curve_between:nn {#2}{#3}
1977   \tl_set_eq:NN #1 \g__spath_output_tl
1978   \tl_gclear:N \g__spath_output_tl
1979 }
1980 \cs_generate_variant:Nn \spath_curve_between:Nnn {NVn, NVV}
1981 \cs_new_protected_nopar:Npn \spath_curve_between:Nn #1#2
1982 {
1983   \spath_curve_between:NVn #1#1{#2}
1984 }
1985 \cs_generate_variant:Nn \spath_curve_between:Nn {NV, cn, cv}
1986 \cs_new_protected_nopar:Npn \spath_gcurve_between:Nnn #1#2#3
1987 {
1988   \__spath_curve_between:nn {#2}
1989   \tl_gset_eq:NN #1 \g__spath_output_tl
1990   \tl_gclear:N \g__spath_output_tl
1991 }
1992 \cs_generate_variant:Nn \spath_gcurve_between:Nnn {NVn, NVV}
1993 \cs_new_protected_nopar:Npn \spath_gcurve_between:Nn #1#2
1994 {
1995   \spath_gcurve_between:NVnn #1#1{#2}
1996 }
1997 \cs_generate_variant:Nn \spath_gcurve_between:Nn {NV, cn, cv}

```

(End of 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
1998 \cs_new_protected_nopar:Npn \__spath_close_with:nn #1#2
1999 {
2000   \group_begin:
2001   \spath_finalmovepoint:Nn \l__spath_tmpa_tl {#1}
2002   \spath_finalpoint:Nn \l__spath_tmpb_tl {#1}
2003   \dim_compare:nTF
2004   {
2005     \dim_abs:n
2006     {
2007       \tl_item:Nn \l__spath_tmpa_tl {1}
2008       -
2009       \tl_item:Nn \l__spath_tmpb_tl {1}
2010     }
2011     +
2012     \dim_abs:n
2013     {
2014       \tl_item:Nn \l__spath_tmpa_tl {2}
2015       -
2016       \tl_item:Nn \l__spath_tmpb_tl {2}
2017     }

```

```

2018 < 0.01pt
2019 }
2020 {
2021   \__spath_close:n {#1}
2022 }
2023 {
2024   \spath_span:NnVV \l__spath_tmpc_tl {#2} \l__spath_tmpb_tl \l__spath_tmpa_tl
2025   \spath_append_no_move:NnV \l__spath_tmpd_tl {#1} \l__spath_tmpc_tl
2026   \__spath_close:V \l__spath_tmpd_tl
2027 }
2028 \group_end:
2029 }
2030 \cs_new_protected_nopar:Npn \spath_close_with:Nnn #1#2#3
2031 {
2032   \__spath_close_with:nn {#2}{#3}
2033   \tl_set_eq:NN #1 \g__spath_output_tl
2034   \tl_gclear:N \g__spath_output_tl
2035 }
2036 \cs_generate_variant:Nn \spath_close_with:Nnn {cnn, cVV, cvv, NVn}
2037 \cs_new_protected_nopar:Npn \spath_close_with:Nn #1#2
2038 {
2039   \spath_close_with:NVn #1#1{#2}
2040 }
2041 \cs_generate_variant:Nn \spath_close_with:Nn {cn, cV, cv, NV}
2042 \cs_new_protected_nopar:Npn \spath_gclose_with:Nnn #1#2#3
2043 {
2044   \__spath_close_with:nn {#2}{#3}
2045   \tl_gset_eq:NN #1 \g__spath_output_tl
2046   \tl_gclear:N \g__spath_output_tl
2047 }
2048 \cs_generate_variant:Nn \spath_gclose_with:Nnn {cnn, cVV, cvv, NVn}
2049 \cs_new_protected_nopar:Npn \spath_gclose_with:Nn #1#2
2050 {
2051   \spath_gclose_with:NVn #1#1{#2}
2052 }
2053 \cs_generate_variant:Nn \spath_gclose_with:Nn {cn, cV, cv, NV}

```

(End of 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
2054 \cs_new_protected_nopar:Npn \__spath_close_with_curve:n #1
2055 {
2056   \group_begin:
2057   \spath_finalpoint:Nn \l__spath_tmpa_tl {#1}
2058   \spath_finaltangent:Nn \l__spath_tmpb_tl {#1}
2059   \spath_finalmovepoint:Nn \l__spath_tmpc_tl {#1}
2060   \spath_finalmovetangent:Nn \l__spath_tmpd_tl {#1}
2061   \dim_compare:nTF
2062   {
2063     \dim_abs:n
2064     {
2065       \tl_item:Nn \l__spath_tmpa_tl {1}
2066       -
2067       \tl_item:Nn \l__spath_tmpc_tl {1}

```

```

2068     }
2069     +
2070     \dim_abs:n
2071     {
2072       \tl_item:Nn \l__spath_tmpa_tl {2}
2073       -
2074       \tl_item:Nn \l__spath_tmpc_tl {2}
2075     }
2076     < 0.01pt
2077   }
2078   {
2079     \__spath_close:n {#1}
2080   }
2081   {
2082
2083     \spath_hobby_curve:NVVVV \l__spath_tmpe_tl
2084     \l__spath_tmpa_tl \l__spath_tmpb_tl \l__spath_tmpd_tl \l__spath_tmpc_tl
2085
2086     \tl_set:Nn \l__spath_tmpa_tl {#1}
2087     \spath_append_no_move:NV \l__spath_tmpa_tl \l__spath_tmpe_tl
2088     \__spath_close:V \l__spath_tmpa_tl
2089   }
2090   \group_end:
2091 }
2092 \cs_new_protected_nopar:Npn \spath_close_with_curve:Nn #1#2
2093 {
2094   \__spath_close_with_curve:n {#2}
2095   \tl_set_eq:NN #1 \g__spath_output_tl
2096   \tl_gclear:N \g__spath_output_tl
2097 }
2098 \cs_generate_variant:Nn \spath_close_with_curve:Nn {cn, cV, cv, NV}
2099 \cs_new_protected_nopar:Npn \spath_close_with_curve:N #1
2100 {
2101   \spath_close_with_curve:NV #1#1
2102 }
2103 \cs_generate_variant:Nn \spath_close_with_curve:N {c}
2104 \cs_new_protected_nopar:Npn \spath_gclose_with_curve:Nn #1#2
2105 {
2106   \__spath_close_with_curve:n {#2}
2107   \tl_gset_eq:NN #1 \g__spath_output_tl
2108   \tl_gclear:N \g__spath_output_tl
2109 }
2110 \cs_generate_variant:Nn \spath_gclose_with_curve:Nn {cn, cV, cv, NV}
2111 \cs_new_protected_nopar:Npn \spath_gclose_with_curve:N #1
2112 {
2113   \spath_gclose_with_curve:NV #1#1
2114 }
2115 \cs_generate_variant:Nn \spath_gclose_with_curve:N {c}

```

(End of 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`

```

2116 \cs_new_protected_nopar:Npn \__spath_weld:nn #1#2
2117 {
2118   \group_begin:
2119   \tl_set:Nn \l__spath_tmpa_tl {#1}
2120   \tl_set:Nn \l__spath_tmpb_tl {#2}
2121   \spath_finalpoint:Nn \l__spath_tmpc_tl {#1}
2122   \spath_translate_to:NV \l__spath_tmpb_tl \l__spath_tmpc_tl
2123
2124   \__spath_append_no_move:VV \l__spath_tmpa_tl \l__spath_tmpb_tl
2125   \group_end:
2126 }
2127 \cs_new_protected_nopar:Npn \spath_weld:Nnn #1#2#3
2128 {
2129   \__spath_weld:nn {#2}{#3}
2130   \tl_set_eq:NN #1 \g__spath_output_tl
2131   \tl_gclear:N \g__spath_output_tl
2132 }
2133 \cs_generate_variant:Nn \spath_weld:Nnn {NVV,NVn}
2134 \cs_new_protected_nopar:Npn \spath_weld:Nn #1#2
2135 {
2136   \spath_weld:NVn #1#1{#2}
2137 }
2138 \cs_generate_variant:Nn \spath_weld:Nn {NV, Nv, cV, cv}
2139 \cs_new_protected_nopar:Npn \spath_gweld:Nnn #1#2#3
2140 {
2141   \__spath_weld:nn {#2}{#3}
2142   \tl_gset_eq:NN #1 \g__spath_output_tl
2143   \tl_gclear:N \g__spath_output_tl
2144 }
2145 \cs_generate_variant:Nn \spath_gweld:Nnn {NVV, NVn}
2146 \cs_new_protected_nopar:Npn \spath_gweld:Nn #1#2
2147 {
2148   \spath_gweld:NVn #1#1{#2}
2149 }
2150 \cs_generate_variant:Nn \spath_gweld:Nn {NV, Nv, cV, cv}

```

(End of 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
 \spath_append_no_move:Nn neither path has a rectangle either side of the join or if the first path isn't closed.

```

\spath_gappend_no_move:Nnn 2151 \cs_new_protected_nopar:Npn \__spath_append_no_move:nn #1#2
\spath_gappend_no_move:Nn 2152 {
2153   \group_begin:
2154   \tl_set:Nn \l__spath_tmpa_tl {#1}
2155   \tl_set:Nn \l__spath_tmpb_tl {#2}
2156   \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpb_tl}
2157   \spath_finalaction:Nn \l__spath_tmpd_tl {#1}
2158   \bool_if:nT {
2159     ! \tl_if_eq_p:NN \l__spath_tmpd_tl \c_spath_closepath_tl
2160     &&
2161     ! \tl_if_eq_p:NN \l__spath_tmpd_tl \c_spath_rectcorner_tl
2162     &&
2163     \tl_if_eq_p:NN \l__spath_tmpc_tl \c_spath_moveto_tl
2164   }

```

```

2165 {
2166   \tl_set:Nx \l__spath_tmpb_tl {\tl_tail:N \l__spath_tmpb_tl}
2167   \tl_set:Nx \l__spath_tmpb_tl {\tl_tail:N \l__spath_tmpb_tl}
2168   \tl_set:Nx \l__spath_tmpb_tl {\tl_tail:N \l__spath_tmpb_tl}
2169 }
2170
2171 \tl_put_right:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
2172 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
2173 \group_end:
2174 }
2175 \cs_generate_variant:Nn \__spath_append_no_move:nn {VV}
2176 \cs_new_protected_nopar:Npn \spath_append_no_move:Nnn #1#2#3
2177 {
2178   \__spath_append_no_move:nn {#2}{#3}
2179   \tl_set_eq:NN #1 \g__spath_output_tl
2180   \tl_gclear:N \g__spath_output_tl
2181 }
2182 \cs_generate_variant:Nn \spath_append_no_move:Nnn {NVV, NVn, NnV}
2183 \cs_new_protected_nopar:Npn \spath_append_no_move:Nn #1#2
2184 {
2185   \spath_append_no_move:NVn #1#1{#2}
2186 }
2187 \cs_generate_variant:Nn \spath_append_no_move:Nn {NV, cv, Nv, cV}
2188 \cs_new_protected_nopar:Npn \spath_gappend_no_move:Nnn #1#2#3
2189 {
2190   \__spath_append_no_move:nn {#2}{#3}
2191   \tl_gset_eq:NN #1 \g__spath_output_tl
2192   \tl_gclear:N \g__spath_output_tl
2193 }
2194 \cs_generate_variant:Nn \spath_gappend_no_move:Nnn {NVV, NVn}
2195 \cs_new_protected_nopar:Npn \spath_gappend_no_move:Nn #1#2
2196 {
2197   \spath_gappend_no_move:NVn #1#1{#2}
2198 }
2199 \cs_generate_variant:Nn \spath_gappend_no_move:Nn {NV, cv, Nv, cV}

```

(End of 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 2200 \cs_new_protected_nopar:Npn \spath_append:Nnn #1#2#3
\spath_gappend:Nnn 2201 {
\spath_gappend:Nn 2202   \tl_set:Nn #1 {#2}
2203   \tl_put_right:Nn #1 {#3}
2204 }
2205 \cs_generate_variant:Nn \spath_append:Nnn {NVV, NVn}
2206 \cs_new_protected_nopar:Npn \spath_append:Nn #1#2
2207 {
2208   \spath_append:NVn #1#1{#2}
2209 }
2210 \cs_generate_variant:Nn \spath_append:Nn {NV, Nv, cv, cV}
2211 \cs_new_protected_nopar:Npn \spath_gappend:Nnn #1#2#3
2212 {
2213   \tl_gset:Nn #1 {#2}
2214   \tl_gput_right:Nn #1 {#3}

```

```

2215 }
2216 \cs_generate_variant:Nn \spath_gappend:Nnn {NVV, NVn}
2217 \cs_new_protected_nopar:Npn \spath_gappend:Nn #1#2
2218 {
2219   \spath_gappend:NVn #1#1{#2}
2220 }
2221 \cs_generate_variant:Nn \spath_gappend:Nn {NV, Nv, cv, cV}

```

(End of 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 2222 \cs_new_protected_nopar:Npn \spath_prepend_no_move:Nnn #1#2#3
 \spath_gprepend_no_move:Nnn 2223 {
 \spath_gprepend_no_move:Nn 2224 \spath_append_no_move:Nnn #1{#3}{#2}

```

2225 }
2226 \cs_generate_variant:Nn \spath_prepend_no_move:Nnn {NVV, NVn}
2227 \cs_new_protected_nopar:Npn \spath_prepend_no_move:Nn #1#2
2228 {
2229   \spath_prepend_no_move:NVn #1#1{#2}
2230 }
2231 \cs_generate_variant:Nn \spath_prepend_no_move:Nn {NV, cv}
2232 \cs_new_protected_nopar:Npn \spath_gprepend_no_move:Nnn #1#2#3
2233 {
2234   \spath_gappend_no_move:Nnn #1{#3}{#2}
2235 }
2236 \cs_generate_variant:Nn \spath_gprepend_no_move:Nnn {NVV, NVn}
2237 \cs_new_protected_nopar:Npn \spath_gprepend_no_move:Nn #1#2
2238 {
2239   \spath_gprepend_no_move:NVn #1#1{#2}
2240 }
2241 \cs_generate_variant:Nn \spath_gprepend_no_move:Nn {NV, cv}

```

(End of 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 2242 \cs_new_protected_nopar:Npn \spath_prepend:Nnn #1#2#3
 \spath_gprepend:Nnn 2243 {
 \spath_gprepend:Nn 2244 \spath_append:Nnn #1{#3}{#2}

```

2245 }
2246 \cs_generate_variant:Nn \spath_prepend:Nnn {NVV, NVn}
2247 \cs_new_protected_nopar:Npn \spath_prepend:Nn #1#2
2248 {
2249   \spath_prepend:NVn #1#1{#2}
2250 }
2251 \cs_generate_variant:Nn \spath_prepend:Nn {NV}
2252 \cs_new_protected_nopar:Npn \spath_gprepend:Nnn #1#2#3
2253 {
2254   \spath_gappend:Nnn #1{#3}{#2}
2255 }
2256 \cs_generate_variant:Nn \spath_gprepend:Nnn {NVV, NVn}
2257 \cs_new_protected_nopar:Npn \spath_gprepend:Nn #1#2
2258 {
2259   \spath_gprepend:NVn #1#1{#2}
2260 }
2261 \cs_generate_variant:Nn \spath_gprepend:Nn {NV}

```

(End of 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
2262 \cs_new_protected_nopar:Npn \__spath_bake_round:n #1
2263 {
2264   \group_begin:
2265   \tl_set:Nn \l__spath_tmpa_tl {#1}
2266   \pgf@processround \l__spath_tmpa_tl\l__spath_tmpb_tl
2267   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
2268   \group_end:
2269 }
2270 \cs_new_protected_nopar:Npn \spath_bake_round:Nn #1#2
2271 {
2272   \__spath_bake_round:n {#2}
2273   \tl_set_eq:NN #1 \g__spath_output_tl
2274   \tl_gclear:N \g__spath_output_tl
2275 }
2276 \cs_generate_variant:Nn \spath_bake_round:Nn {NV}
2277 \cs_new_protected_nopar:Npn \spath_bake_round:N #1
2278 {
2279   \spath_bake_round:NV #1#1
2280 }
2281 \cs_generate_variant:Nn \spath_bake_round:N {c}
2282 \cs_new_protected_nopar:Npn \spath_gbake_round:Nn #1#2
2283 {
2284   \__spath_bake_round:n {#2}
2285   \tl_gset_eq:NN #1 \g__spath_output_tl
2286   \tl_gclear:N \g__spath_output_tl
2287 }
2288 \cs_generate_variant:Nn \spath_gbake_round:Nn {NV}
2289 \cs_new_protected_nopar:Npn \spath_gbake_round:N #1
2290 {
2291   \spath_gbake_round:NV #1#1
2292 }
2293 \cs_generate_variant:Nn \spath_gbake_round:N {c}

```

(End of 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
2294 \cs_new_protected_nopar:Npn \__spath_bake_shorten:n #1
2295 {
2296   \group_begin:
2297   \tl_set:Nn \l__spath_tmpa_tl {#1}
2298   \pgfsyssoftpath@getcurrentpath\l__spath_tmpb_tl
2299   \pgfsyssoftpath@setcurrentpath\l__spath_tmpa_tl
2300   \pgf@prepare@end@of@path
2301   \pgf@prepare@start@of@path
2302   \pgfsyssoftpath@getcurrentpath\l__spath_tmpa_tl
2303   \pgfsyssoftpath@setcurrentpath\l__spath_tmpb_tl
2304   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
2305   \group_end:
2306 }

```



```

2307 \cs_new_protected_nopar:Npn \spath_bake_shorten:Nn #1#2
2308 {
2309   \__spath_bake_shorten:n {#2}
2310   \tl_set_eq:NN #1 \g__spath_output_tl
2311   \tl_gclear:N \g__spath_output_tl
2312 }
2313 \cs_generate_variant:Nn \spath_bake_shorten:Nn {NV}
2314 \cs_new_protected_nopar:Npn \spath_bake_shorten:N #1
2315 {
2316   \spath_bake_shorten:NV #1#1
2317 }
2318 \cs_generate_variant:Nn \spath_bake_shorten:N {c}
2319 \cs_new_protected_nopar:Npn \spath_gbake_shorten:Nn #1#2
2320 {
2321   \__spath_bake_shorten:n {#2}
2322   \tl_gset_eq:NN #1 \g__spath_output_tl
2323   \tl_gclear:N \g__spath_output_tl
2324 }
2325 \cs_generate_variant:Nn \spath_gbake_shorten:Nn {NV}
2326 \cs_new_protected_nopar:Npn \spath_gbake_shorten:N #1
2327 {
2328   \spath_gbake_shorten:NV #1#1
2329 }
2330 \cs_generate_variant:Nn \spath_gbake_shorten:N {c}

```

Shortening the path when it is baked can cause issues with arrows. Putting an arrow in a path definition affects the path because the path gets shortened so that the arrow ends where the path was meant to end. So an arrow affects the path definition, but the arrow is not itself part of the path so if an arrow is used when the path is defined and again when the path is used, the path will be shortened twice which might not be what is intended. Therefore it is useful to have a way to disable the shortening and place an arrow tip at the actual end of the line. The following code achieves that.

Save the original command that computes the arrow shortening.

```

2331 \cs_set_eq:Nc \__spath_pgf_arrow_compute_shortening:n {pgf@arrow@compute@shortening}

```

After `\pgf@arrow@compute@shortening` then `\pgf@xa` is the amount to shorten the line by, so we will be setting that to 0pt. Then `\pgf@xb` is the length of the arrow head which is used to position the arrow and so before zeroing `\pgf@xa` we subtract it from `\pgf@xb` so that the arrow is placed so that its back point is at the current position.

```

2332 \cs_new_nopar:Npn \__spath_arrow_compute_shortening:n #1
2333 {
2334   \__spath_pgf_arrow_compute_shortening:n {#1}
2335   \bool_if:NF \l_spath_arrow_shortening_bool
2336   {
2337     \dim_sub:cn {pgf@xb} {\dim_use:c {pgf@xa}}
2338     \dim_zero:c {pgf@xa}
2339   }
2340 }
2341
2342 \cs_set_eq:cN {pgf@arrow@compute@shortening} \__spath_arrow_compute_shortening:n

```

(End of 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

```

```

2343 \cs_new_protected_nopar:Npn \__spath_close:n #1
2344 {
2345   \group_begin:
2346   \tl_set:Nn \l__spath_tmpa_tl {#1}
2347   \spath_finalmovepoint:NV \l__spath_tmpb_tl \l__spath_tmpa_tl
2348   \tl_put_right:NV \l__spath_tmpa_tl \c_spath_closepath_tl
2349   \tl_put_right:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
2350   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
2351   \group_end:
2352 }
2353 \cs_generate_variant:Nn \__spath_close:n {V}
2354 \cs_new_protected_nopar:Npn \spath_close:Nn #1#2
2355 {
2356   \__spath_close:n {#2}
2357   \tl_set_eq:NN #1 \g__spath_output_tl
2358   \tl_gclear:N \g__spath_output_tl
2359 }
2360 \cs_generate_variant:Nn \spath_close:Nn {NV}
2361 \cs_new_protected_nopar:Npn \spath_close:N #1
2362 {
2363   \spath_close:NV #1#1
2364 }
2365 \cs_generate_variant:Nn \spath_close:N {c}
2366 \cs_new_protected_nopar:Npn \spath_gclose:Nn #1#2
2367 {
2368   \__spath_close:n {#2}
2369   \tl_gset_eq:NN #1 \g__spath_output_tl
2370   \tl_gclear:N \g__spath_output_tl
2371 }
2372 \cs_generate_variant:Nn \spath_gclose:Nn {NV}
2373 \cs_new_protected_nopar:Npn \spath_gclose:N #1
2374 {
2375   \spath_gclose:NV #1#1
2376 }
2377 \cs_generate_variant:Nn \spath_gclose:N {c}

```

(End of 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:N
\spath_adjust_gclose:Nn
\spath_adjust_gclose:N
2378 \cs_new_protected_nopar:Npn \__spath_adjust_close:n #1
2379 {
2380   \group_begin:
2381   \tl_set:Nn \l__spath_tmpa_tl {#1}
2382   \spath_finalmovepoint:NV \l__spath_tmpb_tl \l__spath_tmpa_tl
2383   \spath_finalpoint:NV \l__spath_tmpc_tl \l__spath_tmpa_tl
2384   \tl_reverse:N \l__spath_tmpa_tl
2385   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2386   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2387   \tl_set:Nx \l__spath_tmpd_tl {\tl_head:N \l__spath_tmpa_tl}
2388   \tl_if_eq:NNT \l__spath_tmpd_tl \c_spath_curveto_tl
2389   {
2390     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}

```

```

2391 \tl_clear:N \l__spath_tmpe_tl
2392 \tl_set:Nx \l__spath_tmpe_tl {
2393   {
2394     \dim_eval:n
2395     {
2396       \tl_item:Nn \l__spath_tmpe_tl {1}
2397       -
2398       \tl_item:Nn \l__spath_tmpe_tl {2}
2399       +
2400       \tl_item:Nn \l__spath_tmpe_tl {3}
2401     }
2402   }
2403   {
2404     \dim_eval:n
2405     {
2406       \tl_item:Nn \l__spath_tmpe_tl {2}
2407       -
2408       \tl_item:Nn \l__spath_tmpe_tl {1}
2409       +
2410       \tl_item:Nn \l__spath_tmpe_tl {3}
2411     }
2412   }
2413 }
2414 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
2415 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
2416 \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2417 \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2418 }
2419 \tl_reverse:N \l__spath_tmpe_tl
2420 \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2421 \tl_put_right:NV \l__spath_tmpe_tl \c_spath_closepath_tl
2422 \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2423 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
2424 \group_end:
2425 }
2426 \cs_generate_variant:Nn \__spath_adjust_close:n {V}
2427 \cs_new_protected_nopar:Npn \spath_adjust_close:Nn #1#2
2428 {
2429   \__spath_adjust_close:n {#2}
2430   \tl_set_eq:NN #1 \g__spath_output_tl
2431   \tl_gclear:N \g__spath_output_tl
2432 }
2433 \cs_generate_variant:Nn \spath_adjust_close:Nn {NV}
2434 \cs_new_protected_nopar:Npn \spath_adjust_close:N #1
2435 {
2436   \spath_adjust_close:NV #1#1
2437 }
2438 \cs_generate_variant:Nn \spath_adjust_close:N {c}
2439 \cs_new_protected_nopar:Npn \spath_adjust_gclosen:Nn #1#2
2440 {
2441   \__spath_adjust_close:n {#2}
2442   \tl_gset_eq:NN #1 \g__spath_output_tl
2443   \tl_gclear:N \g__spath_output_tl
2444 }

```

```

2445 \cs_generate_variant:Nn \spath_adjust_gclose:Nn {NV}
2446 \cs_new_protected_nopar:Npn \spath_adjust_gclose:N #1
2447 {
2448   \spath_adjust_gclose:NV #1#1
2449 }
2450 \cs_generate_variant:Nn \spath_adjust_gclose:N {c}

```

(End of 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
\spath_gopen:Nn
\spath_gopen:N
2451 \cs_new_protected_nopar:Npn \__spath_open:n #1
2452 {
2453   \group_begin:
2454   \spath_replace_rectangles:Nn \l__spath_tmpa_tl {#1}
2455   \tl_clear:N \l__spath_tmpb_tl
2456   \bool_until_do:nm {
2457     \tl_if_empty_p:N \l__spath_tmpa_tl
2458   }
2459   {
2460     \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
2461     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2462
2463     \token_case_meaning:NnF \l__spath_tmpc_tl
2464     {
2465       \c_spath_closepath_tl {
2466
2467         \bool_if:nF
2468         {
2469           \dim_compare_p:n
2470           {
2471             \l__spath_move_x_dim == \l__spath_tmpa_dim
2472           }
2473           &&
2474           \dim_compare_p:n
2475           {
2476             \l__spath_move_y_dim == \l__spath_tmpb_dim
2477           }
2478         }
2479         {
2480           \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
2481
2482           \tl_put_right:Nx \l__spath_tmpb_tl {
2483             { \dim_use:N \l__spath_move_x_dim }
2484             { \dim_use:N \l__spath_move_y_dim }
2485           }
2486         }
2487
2488         \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
2489         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2490         \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
2491         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2492       }
2493

```

```

2494 \c_spath_moveto_tl {
2495   \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpc_tl
2496
2497   \dim_set:Nn \l__spath_move_x_dim {\tl_head:N \l__spath_tmpa_tl}
2498   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2499   \dim_set:Nn \l__spath_move_y_dim {\tl_head:N \l__spath_tmpa_tl}
2500   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2501
2502   \tl_put_right:Nx \l__spath_tmpb_tl {
2503     { \dim_use:N \l__spath_move_x_dim }
2504     { \dim_use:N \l__spath_move_y_dim }
2505   }
2506
2507   \dim_set_eq:NN \l__spath_tmpa_dim \l__spath_move_x_dim
2508   \dim_set_eq:NN \l__spath_tmpb_dim \l__spath_move_y_dim
2509 }
2510 }
2511 {
2512   \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpc_tl
2513
2514   \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
2515   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2516   \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
2517   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2518
2519   \tl_put_right:Nx \l__spath_tmpb_tl {
2520     { \dim_use:N \l__spath_tmpa_dim }
2521     { \dim_use:N \l__spath_tmpb_dim }
2522   }
2523 }
2524 }
2525 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
2526 \group_end:
2527 }
2528 \cs_generate_variant:Nn \__spath_open:n {V}
2529 \cs_new_protected_nopar:Npn \spath_open:Nn #1#2
2530 {
2531   \__spath_open:n {#2}
2532   \tl_set_eq:NN #1 \g__spath_output_tl
2533   \tl_gclear:N \g__spath_output_tl
2534 }
2535 \cs_generate_variant:Nn \spath_open:Nn {NV}
2536 \cs_new_protected_nopar:Npn \spath_open:N #1
2537 {
2538   \spath_open:NV #1#1
2539 }
2540 \cs_new_protected_nopar:Npn \spath_gopen:Nn #1#2
2541 {
2542   \__spath_open:n {#2}
2543   \tl_gset_eq:NN #1 \g__spath_output_tl
2544   \tl_gclear:N \g__spath_output_tl
2545 }
2546 \cs_generate_variant:Nn \spath_gopen:Nn {NV}
2547 \cs_new_protected_nopar:Npn \spath_gopen:N #1

```

```

2548 {
2549   \spath_gopen:NV #1#1
2550 }
2551 \cs_generate_variant:Nn \spath_open:N {c}
2552 \cs_generate_variant:Nn \spath_gopen:N {c}

```

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

\spath_replace_lines:Nn Replace any line segments by Bézier curves.

```

\spath_replace_lines:Nn 2553 \cs_new_protected_nopar:Npn \__spath_replace_lines:n #1
\spath_replace_lines:Nn 2554 {
\spath_replace_lines:Nn 2555   \group_begin:
2556   \tl_set:Nn \l__spath_tmpa_tl {#1}
2557   \tl_clear:N \l__spath_tmpb_tl
2558   \dim_set:Nn \l__spath_tmpa_dim {0pt}
2559   \dim_set:Nn \l__spath_tmpb_dim {0pt}
2560
2561   \bool_do_until:nn
2562   {
2563     \tl_if_empty_p:N \l__spath_tmpa_tl
2564   }
2565   {
2566     \tl_set:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpa_tl {1}}
2567     \tl_set:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpa_tl {2}}
2568     \tl_set:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpa_tl {3}}
2569
2570     \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_lineto_tl
2571     {
2572       \tl_put_right:NV \l__spath_tmpb_tl \c_spath_curvetoa_tl
2573       \tl_put_right:Nx \l__spath_tmpb_tl
2574       {
2575         {
2576           \fp_to_dim:n
2577           {
2578              $2/3 * (\l__spath_tmpa_dim)$ 
2579             +
2580              $1/3 * (\l__spath_tmpe_tl)$ 
2581           }
2582         }
2583       }
2584       \tl_put_right:Nx \l__spath_tmpb_tl
2585       {
2586         {
2587           \fp_to_dim:n
2588           {
2589              $2/3 * (\l__spath_tmpb_dim)$ 
2590             +
2591              $1/3 * (\l__spath_tmpe_tl)$ 
2592           }
2593         }
2594       }
2595       \tl_put_right:NV \l__spath_tmpb_tl \c_spath_curvetob_tl
2596       \tl_put_right:Nx \l__spath_tmpb_tl
2597       {

```

```

2598     {
2599         \fp_to_dim:n
2600         {
2601             1/3 * (\l__spath_tmpa_dim)
2602             +
2603             2/3 * (\l__spath_tmpd_tl)
2604         }
2605     }
2606 }
2607 \tl_put_right:Nx \l__spath_tmpb_tl
2608 {
2609     {
2610         \fp_to_dim:n
2611         {
2612             1/3 * (\l__spath_tmpb_dim)
2613             +
2614             2/3 * (\l__spath_tmpe_tl)
2615         }
2616     }
2617 }
2618 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_curveto_tl
2619 \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2620 \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2621 }
2622 {
2623     \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2624     \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2625     \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2626 }
2627 }
2628 \dim_set:Nn \l__spath_tmpa_dim {\l__spath_tmpd_tl}
2629 \dim_set:Nn \l__spath_tmpb_dim {\l__spath_tmpe_tl}
2630 }
2631 \prg_replicate:nn {3}
2632 {
2633     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2634 }
2635 }
2636 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
2637 \group_end:
2638 }
2639 \cs_generate_variant:Nn \__spath_replace_lines:n {V}
2640 \cs_new_protected_nopar:Npn \spath_replace_lines:Nn #1#2
2641 {
2642     \__spath_replace_lines:n {#2}
2643     \tl_set_eq:NN #1 \g__spath_output_tl
2644     \tl_gclear:N \g__spath_output_tl
2645 }
2646 \cs_generate_variant:Nn \spath_replace_lines:Nn {NV, cV, cv, Nv}
2647 \cs_new_protected_nopar:Npn \spath_replace_lines:N #1
2648 {
2649     \spath_replace_lines:NV #1#1
2650 }
2651 \cs_generate_variant:Nn \spath_replace_lines:N {c}

```

```

2652 \cs_new_protected_nopar:Npn \spath_greplace_lines:Nn #1#2
2653 {
2654   \__spath_replace_lines:n {#2}
2655   \tl_gset_eq:NN #1 \g__spath_output_tl
2656   \tl_gclear:N \g__spath_output_tl
2657 }
2658 \cs_generate_variant:Nn \spath_greplace_lines:Nn {NV, cV, cv, Nv}
2659 \cs_new_protected_nopar:Npn \spath_greplace_lines:N #1
2660 {
2661   \spath_greplace_lines:NV #1#1
2662 }
2663 \cs_generate_variant:Nn \spath_greplace_lines:N {c}

```

(End of definition for \spath_replace_lines:Nn.)

\spath_replace_rectangles:Nn Replace any rectangle components by lines.

```

\spath_replace_rectangles:Nn 2664 \cs_new_protected_nopar:Npn \__spath_replace_rectangles:n #1
\spath_replace_rectangles:Nn 2665 {
\spath_replace_rectangles:Nn 2666   \group_begin:
\spath_replace_rectangles:Nn 2667   \tl_set:Nn \l__spath_tmpa_tl {#1}
\spath_replace_rectangles:Nn 2668   \tl_clear:N \l__spath_tmpb_tl
2669
2670   \bool_do_until:nn
2671   {
2672     \tl_if_empty_p:N \l__spath_tmpa_tl
2673   }
2674   {
2675     \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl }
2676     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2677     \tl_set:Nx \l__spath_tmpd_tl {\tl_head:N \l__spath_tmpa_tl }
2678     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2679     \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpa_tl }
2680     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2681
2682     \tl_if_eq:NNTF \l__spath_tmpc_tl \c_spath_rectcorner_tl
2683     {
2684
2685       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2686       \dim_set:Nn \l__spath_tmpa_dim
2687       {
2688         \tl_item:Nn \l__spath_tmpa_tl {1}
2689       }
2690       \dim_set:Nn \l__spath_tmpb_dim
2691       {
2692         \tl_item:Nn \l__spath_tmpa_tl {2}
2693       }
2694
2695       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2696       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2697
2698       \tl_put_right:NV \l__spath_tmpb_tl \c_spath_moveto_tl
2699       \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2700       \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2701

```



```

2702 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
2703 \tl_put_right:Nx \l__spath_tmpb_tl
2704 {
2705   {
2706     \fp_to_dim:n { \l__spath_tmpd_tl + \l__spath_tmpe_dim }
2707   }
2708 }
2709 \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2710
2711 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
2712 \tl_put_right:Nx \l__spath_tmpb_tl
2713 {
2714   {
2715     \fp_to_dim:n { \l__spath_tmpd_tl + \l__spath_tmpe_dim }
2716   }
2717 }
2718 \tl_put_right:Nx \l__spath_tmpb_tl
2719 {
2720   {
2721     \fp_to_dim:n { \l__spath_tmpe_tl + \l__spath_tmpe_dim }
2722   }
2723 }
2724
2725 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
2726 \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2727 \tl_put_right:Nx \l__spath_tmpb_tl
2728 {
2729   {
2730     \fp_to_dim:n { \l__spath_tmpe_tl + \l__spath_tmpe_dim }
2731   }
2732 }
2733
2734 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_closepath_tl
2735 \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2736 \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2737
2738 }
2739 {
2740   \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2741   \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2742   \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2743 }
2744 }
2745 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
2746 \group_end:
2747 }
2748 \cs_generate_variant:Nn \__spath_replace_rectangles:n {V}
2749 \cs_new_protected_nopar:Npn \spath_replace_rectangles:Nn #1#2
2750 {
2751   \__spath_replace_rectangles:n {#2}
2752   \tl_set_eq:NN #1 \g__spath_output_tl
2753   \tl_gclear:N \g__spath_output_tl
2754 }
2755 \cs_generate_variant:Nn \spath_replace_rectangles:Nn {NV, cV, cv, Nv}

```

```

2756 \cs_new_protected_nopar:Npn \spath_replace_rectangles:N #1
2757 {
2758   \spath_replace_rectangles:NV #1#1
2759 }
2760 \cs_generate_variant:Nn \spath_replace_rectangles:N {c}
2761 \cs_new_protected_nopar:Npn \spath_greplace_rectangles:Nn #1#2
2762 {
2763   \__spath_replace_rectangles:n {#2}
2764   \tl_gset_eq:NN #1 \g__spath_output_tl
2765   \tl_gclear:N \g__spath_output_tl
2766 }
2767 \cs_generate_variant:Nn \spath_greplace_rectangles:Nn {NV, cV, cv, Nv}
2768 \cs_new_protected_nopar:Npn \spath_greplace_rectangles:N #1
2769 {
2770   \spath_greplace_rectangles:NV #1#1
2771 }
2772 \cs_generate_variant:Nn \spath_greplace_rectangles:N {c}

```

(End of definition for \spath_replace_rectangles:Nn.)

```

\spath_remove_empty_components:Nn
\spath_remove_empty_components:N
\spath_gremove_empty_components:Nn
\spath_gremove_empty_components:N

```

Remove any component that is simply a moveto.

```

2773 \cs_new_protected_nopar:Npn \__spath_remove_empty_components:n #1
2774 {
2775   \group_begin:
2776   \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
2777   \tl_clear:N \l__spath_tmpa_tl
2778   \seq_map_inline:Nn \l__spath_tmpa_seq
2779   {
2780     \int_compare:nF
2781     {
2782       \tl_count:n {##1} == 3
2783     }
2784     {
2785       \tl_put_right:Nn \l__spath_tmpa_tl {##1}
2786     }
2787   }
2788   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
2789   \group_end:
2790 }
2791 \cs_new_protected_nopar:Npn \spath_remove_empty_components:Nn #1#2
2792 {
2793   \__spath_remove_empty_components:n {#2}
2794   \tl_set_eq:NN #1 \g__spath_output_tl
2795   \tl_gclear:N \g__spath_output_tl
2796 }
2797 \cs_generate_variant:Nn \spath_remove_empty_components:Nn {NV}
2798 \cs_new_protected_nopar:Npn \spath_remove_empty_components:N #1
2799 {
2800   \spath_remove_empty_components:NV #1#1
2801 }
2802 \cs_generate_variant:Nn \spath_remove_empty_components:N {c}
2803 \cs_new_protected_nopar:Npn \spath_gremove_empty_components:Nn #1#2
2804 {
2805   \__spath_remove_empty_components:n {#2}

```

```

2806 \tl_gset_eq:NN #1 \g__spath_output_tl
2807 \tl_gclear:N \g__spath_output_tl
2808 }
2809 \cs_generate_variant:Nn \spath_gremove_empty_components:Nn {NV}
2810 \cs_new_protected_nopar:Npn \spath_gremove_empty_components:N #1
2811 {
2812 \spath_gremove_empty_components:NV #1#1
2813 }
2814 \cs_generate_variant:Nn \spath_gremove_empty_components:N {c}

```

(End of 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.

```

2815 \prg_new_protected_conditional:Npnn \spath_if_eq:nn #1#2 { T, F, TF }
2816 {
2817 \group_begin:
2818 \tl_set:Nn \l__spath_tmpa_tl {#1}
2819 \tl_set:Nn \l__spath_tmpb_tl {#2}
2820 \bool_gset_true:N \g__spath_tmpa_bool
2821 \int_compare:nNnTF
2822 {\tl_count:N \l__spath_tmpa_tl}
2823 =
2824 {\tl_count:N \l__spath_tmpb_tl}
2825 {
2826 \int_step_inline:nmm {1} {3} {\tl_count:N \l__spath_tmpa_tl}
2827 {
2828 \tl_set:Nx \l__spath_tmpc_tl {\tl_item:Nn \l__spath_tmpa_tl {##1}}
2829 \tl_set:Nx \l__spath_tmpd_tl {\tl_item:Nn \l__spath_tmpb_tl {##1}}
2830 \tl_if_eq:NNF \l__spath_tmpc_tl \l__spath_tmpd_tl
2831 {
2832 \bool_gset_false:N \g__spath_tmpa_bool
2833 }
2834 \dim_set:Nn \l__spath_tmpa_dim {\tl_item:Nn \l__spath_tmpa_tl {##1+1}}
2835 \dim_set:Nn \l__spath_tmpb_dim {\tl_item:Nn \l__spath_tmpb_tl {##1+1}}
2836 \dim_compare:nF
2837 {
2838 \dim_abs:n
2839 {
2840 \l__spath_tmpa_dim - \l__spath_tmpb_dim
2841 }
2842 < 0.001pt
2843 }
2844 {
2845 \bool_gset_false:N \g__spath_tmpa_bool
2846 }
2847 \dim_set:Nn \l__spath_tmpa_dim {\tl_item:Nn \l__spath_tmpa_tl {##1+2}}
2848 \dim_set:Nn \l__spath_tmpb_dim {\tl_item:Nn \l__spath_tmpb_tl {##1+2}}
2849 \dim_compare:nF
2850 {
2851 \dim_abs:n
2852 {
2853 \l__spath_tmpa_dim - \l__spath_tmpb_dim
2854 }
2855 < 0.001pt

```

```

2856     }
2857     {
2858     \bool_gset_false:N \g__spath_tmpa_bool
2859     }
2860   }
2861 }
2862 {
2863   \bool_gset_false:N \g__spath_tmpa_bool
2864 }
2865 \group_end:
2866 \bool_if:NTF \g__spath_tmpa_bool
2867 {
2868   \prg_return_true:
2869 }
2870 {
2871   \prg_return_false:
2872 }
2873 }
2874 \prg_generate_conditional_variant:Nnn \spath_if_eq:nn {VV, Vn, nV, vv} {TF, T, F}

```

(End of 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
2875 \cs_new_protected_nopar:Npn \__spath_split_curve:nn #1#2
2876 {
2877   \group_begin:
2878   \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_moveto_tl
2879   \tl_put_right:Nx \l__spath_tmpa_tl {
2880     {\tl_item:nn {#1} {2}}
2881     {\tl_item:nn {#1} {3}}
2882   }
2883   \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curvetoa_tl
2884   \tl_put_right:Nx \l__spath_tmpa_tl
2885   {
2886     {\fp_to_dim:n
2887       {
2888         (1 - #2) * \tl_item:nn {#1} {2} + (#2) * \tl_item:nn {#1} {5}
2889       }}
2890     {\fp_to_dim:n
2891       {
2892         (1 - #2) * \tl_item:nn {#1} {3} + (#2) * \tl_item:nn {#1} {6}
2893       }}
2894   }
2895
2896   \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curvetob_tl
2897   \tl_put_right:Nx \l__spath_tmpa_tl
2898   {
2899     {\fp_to_dim:n
2900       {
2901         (1 - #2)^2 * \tl_item:nn {#1} {2}
2902         + 2 * (1 - #2) * (#2) * \tl_item:nn {#1} {5}
2903         + (#2)^2 * \tl_item:nn {#1} {8}

```

```

2904     }}
2905     {\fp_to_dim:n
2906     {
2907         (1 - #2)^2 * \tl_item:nn {#1} {3}
2908         + 2 * (1 - #2) * (#2) * \tl_item:nn {#1} {6}
2909         + (#2)^2 * \tl_item:nn {#1} {9}
2910     }}
2911 }
2912
2913 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curveto_tl
2914 \tl_put_right:Nx \l__spath_tmpa_tl
2915 {
2916     {\fp_to_dim:n
2917     {
2918         (1 - #2)^3 * \tl_item:nn {#1} {2}
2919         + 3 * (1 - #2)^2 * (#2) * \tl_item:nn {#1} {5}
2920         + 3 * (1 - #2) * (#2)^2 * \tl_item:nn {#1} {8}
2921         + (#2)^3 * \tl_item:nn {#1} {11}
2922     }}
2923     {\fp_to_dim:n
2924     {
2925         (1 - #2)^3 * \tl_item:nn {#1} {3}
2926         + 3 * (1 - #2)^2 * (#2) * \tl_item:nn {#1} {6}
2927         + 3 * (1 - #2) * (#2)^2 * \tl_item:nn {#1} {9}
2928         + (#2)^3 * \tl_item:nn {#1} {12}
2929     }}
2930 }
2931
2932 \tl_gclear:N \g__spath_output_tl
2933 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpa_tl
2934
2935 \tl_clear:N \l__spath_tmpa_tl
2936 \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_moveto_tl
2937 \tl_put_right:Nx \l__spath_tmpa_tl
2938 {
2939     {\fp_to_dim:n
2940     {
2941         (1 - #2)^3 * \tl_item:nn {#1} {2}
2942         + 3 * (1 - #2)^2 * (#2) * \tl_item:nn {#1} {5}
2943         + 3 * (1 - #2) * (#2)^2 * \tl_item:nn {#1} {8}
2944         + (#2)^3 * \tl_item:nn {#1} {11}
2945     }}
2946     {\fp_to_dim:n
2947     {
2948         (1 - #2)^3 * \tl_item:nn {#1} {3}
2949         + 3 * (1 - #2)^2 * (#2) * \tl_item:nn {#1} {6}
2950         + 3 * (1 - #2) * (#2)^2 * \tl_item:nn {#1} {9}
2951         + (#2)^3 * \tl_item:nn {#1} {12}
2952     }}
2953 }
2954
2955 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curvetoa_tl
2956 \tl_put_right:Nx \l__spath_tmpa_tl
2957 {

```

```

2958   {\fp_to_dim:n
2959   {
2960     (1 - #2)^2 * \tl_item:nn {#1} {5}
2961     + 2 * (1 - #2) * (#2) * \tl_item:nn {#1} {8}
2962     + (#2)^2 * \tl_item:nn {#1} {11}
2963   }}
2964   {\fp_to_dim:n
2965   {
2966     (1 - #2)^2 * \tl_item:nn {#1} {6}
2967     + 2 * (1 - #2) * (#2) * \tl_item:nn {#1} {9}
2968     + (#2)^2 * \tl_item:nn {#1} {12}
2969   }}
2970   }
2971   \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curvetob_tl
2972   \tl_put_right:Nx \l__spath_tmpa_tl
2973   {
2974     {\fp_to_dim:n
2975     {
2976       (1 - #2) * \tl_item:nn {#1} {8} + (#2) * \tl_item:nn {#1} {11}
2977     }}
2978     {\fp_to_dim:n
2979     {
2980       (1 - #2) * \tl_item:nn {#1} {9} + (#2) * \tl_item:nn {#1} {12}
2981     }}
2982   }
2983   \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curveto_tl
2984   \tl_put_right:Nx \l__spath_tmpa_tl {
2985     {\tl_item:nn {#1} {11}}
2986     {\tl_item:nn {#1} {12}}
2987   }
2988
2989   \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpa_tl
2990   \group_end:
2991   }
2992   \cs_generate_variant:Nn \__spath_split_curve:nn {nv, nV}
2993   \cs_new_protected_nopar:Npn \spath_split_curve:NNnn #1#2#3#4
2994   {
2995     \__spath_split_curve:nn {#3}{#4}
2996     \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
2997     \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
2998     \tl_gclear:N \g__spath_output_tl
2999   }
3000   \cs_generate_variant:Nn \spath_split_curve:NNnn {NNnV, NNVn, NNVV}
3001   \cs_new_protected_nopar:Npn \spath_gsplit_curve:NNnn #1#2#3#4
3002   {
3003     \__spath_split_curve:nn {#3}{#4}
3004     \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3005     \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3006     \tl_gclear:N \g__spath_output_tl
3007   }
3008   \cs_generate_variant:Nn \spath_gsplit_curve:NNnn {NNnV, NNVn, NNVV}

```

(End of 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
\spath_maybe_gsplit_curve:Nn

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!

```

3009 \cs_new_protected_nopar:Npn \__spath_maybe_split_curve:n #1
3010 {
3011   \group_begin:
3012   \fp_set:Nn \l__spath_tmpa_fp
3013   {
3014     (
3015       \tl_item:nn {#1} {3}
3016       - 3 * \tl_item:nn {#1} {6}
3017       + 3 * \tl_item:nn {#1} {9}
3018       - \tl_item:nn {#1} {12}
3019     )
3020     *
3021     (3 * \tl_item:nn {#1} {8} - 3 * \tl_item:nn {#1} {11})
3022     -
3023     (
3024       \tl_item:nn {#1} {2}
3025       - 3 * \tl_item:nn {#1} {5}
3026       + 3 * \tl_item:nn {#1} {8}
3027       - \tl_item:nn {#1} {11}
3028     )
3029     *
3030     (3 * \tl_item:nn {#1} {9} - 3 * \tl_item:nn {#1} {12})
3031   }
3032   \fp_set:Nn \l__spath_tmpb_fp
3033   {
3034     (
3035       \tl_item:nn {#1} {2}
3036       - 3 * \tl_item:nn {#1} {5}
3037       + 3 * \tl_item:nn {#1} {8}
3038       - \tl_item:nn {#1} {11}
3039     )
3040     *
3041     (
3042       3 * \tl_item:nn {#1} {6}
3043       - 6 * \tl_item:nn {#1} {9}
3044       + 3 * \tl_item:nn {#1} {12}
3045     )
3046     -
3047     (
3048       \tl_item:nn {#1} {3}
3049       - 3 * \tl_item:nn {#1} {6}
3050       + 3 * \tl_item:nn {#1} {9}
3051       - \tl_item:nn {#1} {12}
3052     )
3053     *
3054     (
3055       3 * \tl_item:nn {#1} {5}
3056       - 6 * \tl_item:nn {#1} {8}
3057       + 3 * \tl_item:nn {#1} {11}
3058     )
3059   }

```

```

3060 \fp_compare:nTF
3061 {
3062   \l__spath_tmpb_fp != 0
3063 }
3064 {
3065   \fp_set:Nn \l__spath_tmpa_fp {.5 * \l__spath_tmpa_fp / \l__spath_tmpb_fp}
3066   \bool_if:nTF
3067   {
3068     \fp_compare_p:n {0 < \l__spath_tmpa_fp}
3069     &&
3070     \fp_compare_p:n {\l__spath_tmpa_fp < 1}
3071   }
3072   {
3073     \__spath_split_curve:nV {#1} \l__spath_tmpa_fp
3074   }
3075   {
3076     \tl_gset:Nn \g__spath_output_tl { {#1} {} }
3077   }
3078 }
3079 {
3080   \tl_gset:Nn \g__spath_output_tl { {#1} {} }
3081 }
3082 \group_end:
3083 }
3084 \cs_new_protected_nopar:Npn \spath_maybe_split_curve:NNn #1#2#3
3085 {
3086   \__spath_maybe_split_curve:n {#3}
3087   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3088   \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3089   \tl_gclear:N \g__spath_output_tl
3090 }
3091 \cs_generate_variant:Nn \spath_maybe_split_curve:NNn {NNn, NNv }
3092 \cs_new_protected_nopar:Npn \spath_maybe_gsplit_curve:NNn #1#2#3
3093 {
3094   \__spath_maybe_split_curve:n {#3}
3095   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3096   \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3097   \tl_gclear:N \g__spath_output_tl
3098 }
3099 \cs_generate_variant:Nn \spath_maybe_gsplit_curve:NNn {NNn, NNv}

```

(End of 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
3100 \cs_new_protected_nopar:Npn \__spath_split_curves:n #1
3101 {
3102   \group_begin:
3103   \tl_set:Nn \l__spath_tmpa_tl {#1}
3104   \tl_clear:N \l__spath_tmpb_tl
3105   \tl_clear:N \l__spath_tmpc_tl
3106   \bool_do_until:nm
3107   {
3108     \tl_if_empty_p:N \l__spath_tmpa_tl
3109   }

```



```

3110 {
3111   \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpe_tl}
3112   \token_case_meaning:NnF \l__spath_tmpe_tl
3113   {
3114     \c_spath_curvetoa_tl
3115     {
3116       \tl_clear:N \l__spath_tmpe_tl
3117       \tl_set_eq:NN \l__spath_tmpe_tl \c_spath_movetoa_tl
3118       \tl_put_right:Nx \l__spath_tmpe_tl
3119       {
3120         { \dim_use:N \l__spath_tmpe_dim }
3121         { \dim_use:N \l__spath_tmpe_dim }
3122       }
3123       \dim_set:Nn \l__spath_tmpe_dim
3124       {
3125         \tl_item:Nn \l__spath_tmpe_tl {8}
3126       }
3127       \dim_set:Nn \l__spath_tmpe_dim
3128       {
3129         \tl_item:Nn \l__spath_tmpe_tl {9}
3130       }
3131       \prg_replicate:nn {3}
3132       {
3133         \tl_put_right:Nx \l__spath_tmpe_tl
3134         {
3135           \tl_item:Nn \l__spath_tmpe_tl {1}
3136           {\tl_item:Nn \l__spath_tmpe_tl {2}}
3137           {\tl_item:Nn \l__spath_tmpe_tl {3}}
3138         }
3139         \prg_replicate:nn {3}
3140         {
3141           \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3142         }
3143       }
3144     }
3145     \spath_maybe_split_curve:NNV
3146     \l__spath_tmpe_tl
3147     \l__spath_tmpe_tl
3148     \l__spath_tmpe_tl
3149     \prg_replicate:nn {3}
3150     {
3151       \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3152       \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3153     }
3154     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3155     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3156   }
3157 }
3158 {
3159   \dim_set:Nn \l__spath_tmpe_dim
3160   {
3161     \tl_item:Nn \l__spath_tmpe_tl {2}
3162   }
3163   \dim_set:Nn \l__spath_tmpe_dim

```

```

3164     {
3165     \tl_item:Nn \l__spath_tmpa_tl {3}
3166     }
3167     \tl_put_right:Nx \l__spath_tmpb_tl
3168     {
3169     \tl_item:Nn \l__spath_tmpa_tl {1}
3170     {\tl_item:Nn \l__spath_tmpa_tl {2}}
3171     {\tl_item:Nn \l__spath_tmpa_tl {3}}
3172     }
3173     \prg_replicate:nn {3}
3174     {
3175     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3176     }
3177   }
3178 }
3179 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
3180 \group_end:
3181 }
3182 \cs_new_protected_nopar:Npn \spath_split_curves:Nn #1#2
3183 {
3184   \__spath_split_curves:n {#2}
3185   \tl_set_eq:NN #1 \g__spath_output_tl
3186   \tl_gclear:N \g__spath_output_tl
3187 }
3188 \cs_generate_variant:Nn \spath_split_curves:Nn {NV, cV, cn, cv }
3189 \cs_new_protected_nopar:Npn \spath_split_curves:N #1
3190 {
3191   \spath_split_curves:NV #1#1
3192 }
3193 \cs_generate_variant:Nn \spath_split_curves:N {c}
3194 \cs_new_protected_nopar:Npn \spath_gsplit_curves:Nn #1#2
3195 {
3196   \__spath_split_curves:n {#2}
3197   \tl_gset_eq:NN #1 \g__spath_output_tl
3198   \tl_gclear:N \g__spath_output_tl
3199 }
3200 \cs_generate_variant:Nn \spath_gsplit_curves:Nn {NV, cV, cn, cv }
3201 \cs_new_protected_nopar:Npn \spath_gsplit_curves:N #1
3202 {
3203   \spath_gsplit_curves:NV #1#1
3204 }
3205 \cs_generate_variant:Nn \spath_gsplit_curves:N {c}

```

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

\spath_split_line:NNnn Splits a line segment.

```

\spath_gsplit_line:NNnn
3206 \cs_new_protected_nopar:Npn \__spath_split_line:nn #1#2
3207 {
3208   \group_begin:
3209   \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_moveto_tl
3210   \tl_put_right:Nx \l__spath_tmpa_tl {
3211     {\tl_item:nn {#1} {2}}
3212     {\tl_item:nn {#1} {3}}
3213   }

```

```

3214 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_lineto_tl
3215 \tl_put_right:Nx \l__spath_tmpa_tl
3216 {
3217   {\fp_to_dim:n
3218     {
3219       (1 - #2) * \tl_item:nn {#1} {2} + (#2) * \tl_item:nn {#1} {5}
3220     }}
3221   {\fp_to_dim:n
3222     {
3223       (1 - #2) * \tl_item:nn {#1} {3} + (#2) * \tl_item:nn {#1} {6}
3224     }}
3225 }
3226 \tl_gclear:N \g__spath_output_tl
3227 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpa_tl
3228
3229 \tl_clear:N \l__spath_tmpa_tl
3230 \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_moveto_tl
3231 \tl_put_right:Nx \l__spath_tmpa_tl
3232 {
3233   {\fp_to_dim:n
3234     {
3235       (1 - #2) * \tl_item:nn {#1} {2} + (#2) * \tl_item:nn {#1} {5}
3236     }}
3237   {\fp_to_dim:n
3238     {
3239       (1 - #2) * \tl_item:nn {#1} {3} + (#2) * \tl_item:nn {#1} {6}
3240     }}
3241 }
3242 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_lineto_tl
3243 \tl_put_right:Nx \l__spath_tmpa_tl {
3244   {\tl_item:nn {#1} {5}}
3245   {\tl_item:nn {#1} {6}}
3246 }
3247
3248 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpa_tl
3249 \group_end:
3250 }
3251 \cs_new_protected_nopar:Npn \spath_split_line:NNnn #1#2#3#4
3252 {
3253   \__spath_split_line:nn {#3}{#4}
3254   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3255   \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3256   \tl_gclear:N \g__spath_output_tl
3257 }
3258 \cs_generate_variant:Nn \spath_split_line:NNnn {NNnV, NNVn, NNVV}
3259 \cs_new_protected_nopar:Npn \spath_gsplit_line:NNnn #1#2#3#4
3260 {
3261   \__spath_split_line:nn {#3}{#4}
3262   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3263   \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3264   \tl_gclear:N \g__spath_output_tl
3265 }
3266 \cs_generate_variant:Nn \spath_gsplit_line:NNnn {NNnV, NNVn, NNVV}

```

(End of 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.

```

3267 \cs_new_protected_nopar:Npn \__spath_split_rectangle:nn #1#2
3268 {
3269   \group_begin:
3270   \spath_open:Nn \l__spath_tmpa_tl {#1}
3271   \fp_set:Nn \l__spath_tmpa_fp {4*{#2}}
3272   \spath_split_at:NNVV
3273   \l__spath_tmpa_tl \l__spath_tmpb_tl \l__spath_tmpa_tl \l__spath_tmpa_fp
3274   \__spath_append_no_move:VV \l__spath_tmpb_tl \l__spath_tmpa_tl
3275   \group_end:
3276 }
3277 \cs_new_protected_nopar:Npn \spath_split_rectangle:Nnn #1#2#3
3278 {
3279   \__spath_split_rectangle:nn {#2}{#3}
3280   \tl_set_eq:NN #1 \g__spath_output_tl
3281   \tl_gclear:N \g__spath_output_tl
3282 }
3283 \cs_generate_variant:Nn \spath_split_rectangle:Nnn {NnV, NVn, NVV}
3284 \cs_new_protected_nopar:Npn \spath_gsplit_rectangle:Nnn #1#2#3
3285 {
3286   \__spath_split_rectangle:nn {#2}{#3}
3287   \tl_gset_eq:NN #1 \g__spath_output_tl
3288   \tl_gclear:N \g__spath_output_tl
3289 }
3290 \cs_generate_variant:Nn \spath_gsplit_rectangle:Nnn {NnV, NVn, NVV}

```

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

\spath_split_at:NNnn
\spath_split_at:Nnn
\spath_split_at:Nn
\spath_gsplit_at:NNnn
\spath_gsplit_at:Nnn
\spath_gsplit_at:Nn
\spath_split_at_keep_start:Nnn
\spath_split_at_keep_start:Nn
\spath_gsplit_at_keep_start:Nnn
\spath_gsplit_at_keep_start:Nn
\spath_split_at_keep_end:Nnn
\spath_split_at_keep_end:Nn
\spath_gsplit_at_keep_end:Nnn
\spath_gsplit_at_keep_end:Nn
\spath_split_at_normalised:NNnn
\spath_split_at_normalised:Nnn
\spath_split_at_normalised:Nn
\spath_gsplit_at_normalised:NNnn
\spath_gsplit_at_normalised:Nnn
\spath_gsplit_at_normalised:Nn
\spath_split_at_normalised_keep_start:Nnn
\spath_split_at_normalised_keep_start:Nn
\spath_gsplit_at_normalised_keep_start:Nnn
\spath_gsplit_at_normalised_keep_start:Nn
\spath_split_at_normalised_keep_end:Nnn
\spath_split_at_normalised_keep_end:Nn
\spath_gsplit_at_normalised_keep_end:Nnn
\spath_gsplit_at_normalised_keep_end:Nn

Split a path according to the parameter generated by the intersection routine. The versions with two N arguments stores the two parts in two macros, the version with a single N joins them back into a single path (as separate components). The keep versions throw away the other part of the curve.

```

3291 \cs_new_protected_nopar:Npn \__spath_split_at:nn #1#2
3292 {
3293   \group_begin:
3294   \int_set:Nn \l__spath_tmpa_int {\fp_to_int:n {floor{#2} + 1}}
3295   \fp_set:Nn \l__spath_tmpa_fp {#2 - floor{#2}}
3296
3297   % Is split point near one end or other of a component?
3298   \fp_compare:nT
3299   {
3300     \l__spath_tmpa_fp < 0.01
3301   }
3302   {
3303     % Near the start, so we'll place it at the start
3304     \fp_set:Nn \l__spath_tmpa_fp {0}
3305   }
3306   \fp_compare:nT
3307   {
3308     \l__spath_tmpa_fp > 0.99
3309   }
3310   {
3311     % Near the end, so we'll place it at the end
3312     \fp_set:Nn \l__spath_tmpa_fp {0}

```

```

3313     \int_incr:N \l__spath_tmpe_int
3314 }
3315
3316 \int_zero:N \l__spath_tmpe_int
3317 \bool_set_true:N \l__spath_tmpe_bool
3318
3319 \tl_set:Nn \l__spath_tmpe_tl {#1}
3320
3321 \dim_zero:N \l__spath_tmpe_dim
3322 \dim_zero:N \l__spath_tmpe_dim
3323
3324 % Remember if the component is closed
3325 \spath_finalaction:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3326
3327 \bool_set:Nn \l__spath_tmpe_closed_bool
3328 {
3329   \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_closepath_tl
3330   ||
3331   \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_rectcorner_tl
3332 }
3333
3334 % Open it
3335 \spath_open:N \l__spath_tmpe_tl
3336
3337 \tl_clear:N \l__spath_tmpe_tl
3338 \tl_clear:N \l__spath_tmpe_tl
3339 \tl_clear:N \l__spath_tmpe_tl
3340 \tl_clear:N \l__spath_tmpe_tl
3341
3342 \bool_until_do:nn {
3343   \tl_if_empty_p:N \l__spath_tmpe_tl
3344   ||
3345   \int_compare_p:n { \l__spath_tmpe_int == \l__spath_tmpe_int }
3346 }
3347 {
3348   \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpe_tl}
3349   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3350   \token_case_meaning:Nn \l__spath_tmpe_tl
3351   {
3352     \c_spath_lineto_tl
3353     {
3354       \int_incr:N \l__spath_tmpe_int
3355     }
3356     \c_spath_curvetoa_tl
3357     {
3358       \int_incr:N \l__spath_tmpe_int
3359     }
3360     \c_spath_rectcorner_tl
3361     {
3362       \int_incr:N \l__spath_tmpe_int
3363     }
3364   }
3365   \int_compare:nT { \l__spath_tmpe_int < \l__spath_tmpe_int }
3366   {

```

```

3367 \tl_put_right:NV \l__spath_tmpc_tl \l__spath_tmpf_tl
3368
3369 \tl_put_right:Nx \l__spath_tmpc_tl
3370 {{ \tl_head:N \l__spath_tmpe_tl }}
3371 \dim_set:Nn \l__spath_tmpe_dim {\tl_head:N \l__spath_tmpe_tl}
3372 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3373
3374 \tl_put_right:Nx \l__spath_tmpc_tl
3375 {{ \tl_head:N \l__spath_tmpe_tl }}
3376 \dim_set:Nn \l__spath_tmpe_dim {\tl_head:N \l__spath_tmpe_tl}
3377 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3378
3379 }
3380 }
3381
3382 \tl_clear:N \l__spath_tmpd_tl
3383 \tl_put_right:NV \l__spath_tmpd_tl \c_spath_moveto_tl
3384 \tl_put_right:Nx \l__spath_tmpd_tl
3385 {
3386   {\dim_use:N \l__spath_tmpe_dim}
3387   {\dim_use:N \l__spath_tmpe_dim}
3388 }
3389
3390 \fp_compare:nTF
3391 {
3392   \l__spath_tmpe_fp == 0
3393 }
3394 {
3395   \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpd_tl
3396   \tl_if_empty:NF \l__spath_tmpe_tl
3397   {
3398     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpf_tl
3399     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3400   }
3401 }
3402 {
3403
3404   \token_case_meaning:Nn \l__spath_tmpe_tl
3405   {
3406     \c_spath_lineto_tl
3407     {
3408       \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpf_tl
3409       \tl_put_right:Nx \l__spath_tmpe_tl
3410       {{ \tl_head:N \l__spath_tmpe_tl }}
3411       \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3412
3413       \tl_put_right:Nx \l__spath_tmpe_tl
3414       {{ \tl_head:N \l__spath_tmpe_tl }}
3415       \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3416
3417       \spath_split_line:NNVV
3418       \l__spath_tmpe_tl
3419       \l__spath_tmpe_tl
3420       \l__spath_tmpe_tl

```

```

3421     \l__spath_tmpa_fp
3422
3423     \prg_replicate:nn {3} {
3424         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3425     }
3426
3427     \tl_put_right:NV \l__spath_tmpc_tl \l__spath_tmpa_tl
3428     \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
3429 }
3430 \c_spath_curvetoa_tl
3431 {
3432     \tl_put_right:NV \l__spath_tmpd_tl \l__spath_tmpf_tl
3433     \tl_put_right:Nx \l__spath_tmpd_tl
3434     {{ \tl_head:N \l__spath_tmpe_tl }}
3435     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3436
3437     \tl_put_right:Nx \l__spath_tmpd_tl
3438     {{ \tl_head:N \l__spath_tmpe_tl }}
3439     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3440
3441     \prg_replicate:nn {2} {
3442
3443         \tl_put_right:Nx \l__spath_tmpd_tl
3444         { \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
3448         {{ \tl_head:N \l__spath_tmpe_tl }}
3449         \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3450
3451         \tl_put_right:Nx \l__spath_tmpd_tl
3452         {{ \tl_head:N \l__spath_tmpe_tl }}
3453         \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3454     }
3455
3456     \spath_split_curve:NNVV
3457     \l__spath_tmpa_tl
3458     \l__spath_tmpb_tl
3459     \l__spath_tmpd_tl \l__spath_tmpa_fp
3460
3461     \prg_replicate:nn {3} {
3462         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3463     }
3464
3465     \tl_put_right:NV \l__spath_tmpc_tl \l__spath_tmpa_tl
3466     \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
3467 }
3468
3469 \c_spath_rectcorner_tl
3470 {
3471     \tl_clear:N \l__spath_tmpd_tl
3472     \tl_put_right:NV \l__spath_tmpd_tl \l__spath_tmpf_tl
3473
3474     \tl_put_right:Nx \l__spath_tmpd_tl {{\tl_head:N \l__spath_tmpe_tl}}

```

```

3475     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3476     \tl_put_right:Nx \l__spath_tmpd_tl {{\tl_head:N \l__spath_tmpe_tl}}
3477     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3478
3479     \tl_put_right:Nx \l__spath_tmpd_tl {\tl_head:N \l__spath_tmpe_tl}
3480     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3481
3482     \tl_put_right:Nx \l__spath_tmpd_tl {{\tl_head:N \l__spath_tmpe_tl}}
3483     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3484     \tl_put_right:Nx \l__spath_tmpd_tl {{\tl_head:N \l__spath_tmpe_tl}}
3485     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3486
3487     \spath_split_rectangle:NVV
3488     \l__spath_tmpe_tl
3489     \l__spath_tmpd_tl
3490     \l__spath_tmpe_tl
3491
3492     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3493     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3494   }
3495
3496 }
3497 }
3498
3499 \bool_if:NT \l__spath_closed_bool
3500 {
3501   \prg_replicate:nn {3}
3502   {
3503     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3504   }
3505   \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3506   \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpe_tl
3507   \tl_clear:N \l__spath_tmpe_tl
3508 }
3509
3510 \tl_gclear:N \g__spath_output_tl
3511 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpe_tl
3512 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpe_tl
3513 \group_end:
3514 }
3515 \cs_generate_variant:Nn \__spath_split_at:nn {nV, VV}
3516 \cs_new_protected_nopar:Npn \__spath_split_at_normalised:nn #1#2
3517 {
3518   \group_begin:
3519   \spath_reallength:Nn \l__spath_tmpe_int {#1}
3520
3521   \tl_set:Nx \l__spath_tmpe_tl
3522   {\fp_to_decimal:n {(#2) * (\l__spath_tmpe_int)}}
3523   \__spath_split_at:nV {#1} \l__spath_tmpe_tl
3524   \group_end:
3525 }
3526 \cs_generate_variant:Nn \__spath_split_at_normalised:nn {nV}
3527 \cs_new_protected_nopar:Npn \spath_split_at:NNnn #1#2#3#4
3528 {

```



```

3529 \_spath_split_at:nn {#3}{#4}
3530 \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3531 \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3532 \tl_gclear:N \g__spath_output_tl
3533 }
3534 \cs_generate_variant:Nn \spath_split_at:NNnn {NNVn, NNVV, NNnV}
3535 \cs_new_protected_nopar:Npn \spath_gsplit_at:NNnn #1#2#3#4
3536 {
3537 \_spath_split_at:nn {#3}{#4}
3538 \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3539 \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3540 \tl_gclear:N \g__spath_output_tl
3541 }
3542 \cs_generate_variant:Nn \spath_gsplit_at:NNnn {NNVn, NNVV, NNnV}
3543 \cs_new_protected_nopar:Npn \spath_split_at_keep_start:Nnn #1#2#3
3544 {
3545 \_spath_split_at:nn {#2}{#3}
3546 \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3547 \tl_gclear:N \g__spath_output_tl
3548 }
3549 \cs_generate_variant:Nn \spath_split_at_keep_start:Nnn {NVn}
3550 \cs_new_protected_nopar:Npn \spath_split_at_keep_start:Nn #1#2
3551 {
3552 \spath_split_at_keep_start:NVn #1#1{#2}
3553 }
3554 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_start:Nnn #1#2#3
3555 {
3556 \_spath_split_at:nn {#2}{#3}
3557 \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3558 \tl_gclear:N \g__spath_output_tl
3559 }
3560 \cs_generate_variant:Nn \spath_gsplit_at_keep_start:Nnn {NVn}
3561 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_start:Nn #1#2
3562 {
3563 \spath_gsplit_at_keep_start:NVn #1#1{#2}
3564 }
3565 \cs_new_protected_nopar:Npn \spath_split_at_keep_end:Nnn #1#2#3
3566 {
3567 \_spath_split_at:nn {#2}{#3}
3568 \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3569 \tl_gclear:N \g__spath_output_tl
3570 }
3571 \cs_generate_variant:Nn \spath_split_at_keep_end:Nnn {NVn}
3572 \cs_new_protected_nopar:Npn \spath_split_at_keep_end:Nn #1#2
3573 {
3574 \spath_split_at_keep_end:NVn #1#1{#2}
3575 }
3576 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_end:Nnn #1#2#3
3577 {
3578 \_spath_split_at:nn {#2}{#3}
3579 \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3580 \tl_gclear:N \g__spath_output_tl
3581 }
3582 \cs_generate_variant:Nn \spath_gsplit_at_keep_end:Nnn {NVn}

```

```

3583 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_end:Nn #1#2
3584 {
3585   \spath_gsplit_at_keep_end:NVn #1#1{#2}
3586 }
3587 \cs_new_protected_nopar:Npn \spath_split_at_normalised:NNnn #1#2#3#4
3588 {
3589   \__spath_split_at_normalised:nn {#3}{#4}
3590   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3591   \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3592   \tl_gclear:N \g__spath_output_tl
3593 }
3594 \cs_generate_variant:Nn \spath_split_at_normalised:NNnn {NNVn, NNVV, NNnV, ccvn}
3595 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised:NNnn #1#2#3#4
3596 {
3597   \__spath_split_at_normalised:nn {#3}{#4}
3598   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3599   \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3600   \tl_gclear:N \g__spath_output_tl
3601 }
3602 \cs_generate_variant:Nn \spath_gsplit_at_normalised:NNnn {NNVn, NNVV, NNnV, ccvn}
3603 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_start:Nnn #1#2#3
3604 {
3605   \__spath_split_at_normalised:nn {#2}{#3}
3606   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3607   \tl_gclear:N \g__spath_output_tl
3608 }
3609 \cs_generate_variant:Nn \spath_split_at_normalised_keep_start:Nnn {NVn}
3610 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_start:Nn #1#2
3611 {
3612   \spath_split_at_normalised_keep_start:NVn #1#1{#2}
3613 }
3614 \cs_generate_variant:Nn \spath_split_at_normalised_keep_start:Nn {cn}
3615 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_start:Nnn #1#2#3
3616 {
3617   \__spath_split_at_normalised:nn {#2}{#3}
3618   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3619   \tl_gclear:N \g__spath_output_tl
3620 }
3621 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_start:Nnn {NVn}
3622 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_start:Nn #1#2
3623 {
3624   \spath_gsplit_at_normalised_keep_start:NVn #1#1{#2}
3625 }
3626 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_start:Nn {cn}
3627 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_end:Nnn #1#2#3
3628 {
3629   \__spath_split_at_normalised:nn {#2}{#3}
3630   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3631   \tl_gclear:N \g__spath_output_tl
3632 }
3633 \cs_generate_variant:Nn \spath_split_at_normalised_keep_end:Nnn {NVn}
3634 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_end:Nn #1#2
3635 {
3636   \spath_split_at_normalised_keep_end:NVn #1#1{#2}

```

```

3637 }
3638 \cs_generate_variant:Nn \spath_split_at_normalised_keep_end:Nn {cn}
3639 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_end:Nnn #1#2#3
3640 {
3641   \__spath_split_at_normalised:nn {#2}{#3}
3642   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3643   \tl_gclear:N \g__spath_output_tl
3644 }
3645 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_end:Nnn {NVn}
3646 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_end:Nn #1#2
3647 {
3648   \spath_gsplit_at_normalised_keep_end:NVn #1#1{#2}
3649 }
3650 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_end:Nn {cn}
3651 \cs_new_protected_nopar:Npn \spath_split_at:Nnn #1#2#3
3652 {
3653   \__spath_split_at:nn {#2}{#3}
3654   \tl_set: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_split_at:Nnn {NVn, NVV}
3662 \cs_new_protected_nopar:Npn \spath_split_at:Nn #1#2
3663 {
3664   \spath_split_at:NVn #1#1{#2}
3665 }
3666 \cs_new_protected_nopar:Npn \spath_gsplit_at:Nnn #1#2#3
3667 {
3668   \__spath_split_at:nn {#2}{#3}
3669   \tl_gset:Nx #1
3670   {
3671     \tl_item:Nn \g__spath_output_tl {1}
3672     \tl_item:Nn \g__spath_output_tl {2}
3673   }
3674   \tl_gclear:N \g__spath_output_tl
3675 }
3676 \cs_generate_variant:Nn \spath_gsplit_at:Nnn {NVn, NVV}
3677 \cs_new_protected_nopar:Npn \spath_gsplit_at:Nn #1#2
3678 {
3679   \spath_gsplit_at:NVn #1#1{#2}
3680 }
3681 \cs_new_protected_nopar:Npn \spath_split_at_normalised:Nnn #1#2#3
3682 {
3683   \__spath_split_at_normalised:nn {#2}{#3}
3684   \tl_set:Nx #1
3685   {
3686     \tl_item:Nn \g__spath_output_tl {1}
3687     \tl_item:Nn \g__spath_output_tl {2}
3688   }
3689   \tl_gclear:N \g__spath_output_tl
3690 }

```

```

3691 \cs_generate_variant:Nn \spath_split_at_normalised:Nnn {NVn, NVV}
3692 \cs_new_protected_nopar:Npn \spath_split_at_normalised:Nn #1#2
3693 {
3694   \spath_split_at_normalised:NVn #1#1{#2}
3695 }
3696 \cs_generate_variant:Nn \spath_split_at_normalised:Nn {cn}
3697 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised:Nnn #1#2#3
3698 {
3699   \__spath_split_at_normalised:nn {#2}{#3}
3700   \tl_gset:Nx #1
3701   {
3702     \tl_item:Nn \g__spath_output_tl {1}
3703     \tl_item:Nn \g__spath_output_tl {2}
3704   }
3705   \tl_gclear:N \g__spath_output_tl
3706 }
3707 \cs_generate_variant:Nn \spath_gsplit_at_normalised:Nnn {NVn, NVV}
3708 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised:Nn #1#2
3709 {
3710   \spath_gsplit_at_normalised:NVn #1#1{#2}
3711 }
3712 \cs_generate_variant:Nn \spath_gsplit_at_normalised:Nn {cn}

```

(End of 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.

```

3713 \cs_new_protected_nopar:Npn \__spath_shorten_at_end:nn #1#2
3714 {
3715   \int_compare:nTF
3716   {
3717     \tl_count:n {#1} > 3
3718   }
3719   {
3720     \group_begin:
3721     \tl_set:Nn \l__spath_tmpa_tl {#1}
3722     \tl_reverse:N \l__spath_tmpa_tl
3723
3724     \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {3}}
3725
3726     \tl_clear:N \l__spath_tmpe_tl
3727     \tl_if_eq:NNTF \l__spath_tmpb_tl \c_spath_curveto_tl
3728     {
3729       \int_set:Nn \l__spath_tmpa_int {3}
3730     }
3731     {
3732       \int_set:Nn \l__spath_tmpa_int {1}

```

```

3733 }
3734
3735 \prg_replicate:nn { \l__spath_tmpa_int }
3736 {
3737   \tl_put_right:Nx \l__spath_tmpe_tl
3738   {
3739     {\tl_head:N \l__spath_tmpa_tl}
3740   }
3741   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3742   \tl_put_right:Nx \l__spath_tmpe_tl
3743   {
3744     {\tl_head:N \l__spath_tmpa_tl}
3745   }
3746   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3747   \tl_put_right:Nx \l__spath_tmpe_tl
3748   {
3749     \tl_head:N \l__spath_tmpa_tl
3750   }
3751   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3752 }
3753
3754 \tl_put_right:Nx \l__spath_tmpe_tl
3755 {
3756   {\tl_item:Nn \l__spath_tmpa_tl {1}}
3757   {\tl_item:Nn \l__spath_tmpa_tl {2}}
3758 }
3759 \tl_put_right:NV \l__spath_tmpe_tl \c_spath_moveto_tl
3760
3761 \tl_reverse:N \l__spath_tmpa_tl
3762
3763 \fp_set:Nn \l__spath_tmpa_fp
3764 {
3765   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {4}}
3766   -
3767   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {1}}
3768 }
3769
3770 \fp_set:Nn \l__spath_tmpb_fp
3771 {
3772   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {5}}
3773   -
3774   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {2}}
3775 }
3776
3777 \fp_set:Nn \l__spath_tmpe_fp
3778 {
3779   sqrt(
3780     \l__spath_tmpa_fp * \l__spath_tmpa_fp
3781     +
3782     \l__spath_tmpb_fp * \l__spath_tmpb_fp
3783   ) * \l__spath_tmpa_int
3784 }
3785
3786 \fp_compare:nTF

```

```

3787 {
3788   \l__spath_tmpc_fp > #2
3789 }
3790 {
3791
3792   \fp_set:Nn \l__spath_tmpc_fp
3793   {
3794     (\l__spath_tmpc_fp - #2)/ \l__spath_tmpc_fp
3795   }
3796
3797   \tl_reverse:N \l__spath_tmpe_tl
3798
3799   \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_curveto_tl
3800   {
3801     \spath_split_curve:NNVV
3802     \l__spath_tmpe_tl
3803     \l__spath_tmpe_tl
3804     \l__spath_tmpe_tl
3805     \l__spath_tmpe_tl
3806   }
3807   {
3808     \spath_split_line:NNVV
3809     \l__spath_tmpe_tl
3810     \l__spath_tmpe_tl
3811     \l__spath_tmpe_tl
3812     \l__spath_tmpe_tl
3813   }
3814
3815   \prg_replicate:nn {3}
3816   {
3817     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3818   }
3819
3820   \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3821 }
3822 {
3823
3824
3825   \int_compare:nT
3826   {
3827     \tl_count:N \l__spath_tmpe_tl > 3
3828   }
3829   {
3830     \dim_set:Nn \l__spath_tmpe_dim {\fp_to_dim:n {#2 - \l__spath_tmpe_fp } }
3831     \spath_shorten_at_end:NV \l__spath_tmpe_tl \l__spath_tmpe_dim
3832   }
3833 }
3834
3835 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
3836 \group_end:
3837 }
3838 {
3839   \tl_gset:Nn \g__spath_output_tl {#1}
3840 }

```

```

3841 }
3842 \cs_new_protected_nopar:Npn \spath_shorten_at_end:Nnn #1#2#3
3843 {
3844   \__spath_shorten_at_end:nn {#2}{#3}
3845   \tl_set_eq:NN #1 \g__spath_output_tl
3846   \tl_gclear:N \g__spath_output_tl
3847 }
3848 \cs_generate_variant:Nn \spath_shorten_at_end:Nnn {NVV, cnn, cVV, NVn}
3849 \cs_new_protected_nopar:Npn \spath_shorten_at_end:Nn #1#2
3850 {
3851   \spath_shorten_at_end:NVn #1#1{#2}
3852 }
3853 \cs_generate_variant:Nn \spath_shorten_at_end:Nn {cn, cV, NV}
3854 \cs_new_protected_nopar:Npn \spath_gshorten_at_end:Nnn #1#2#3
3855 {
3856   \__spath_shorten_at_end:nn {#2}{#3}
3857   \tl_gset_eq:NN #1 \g__spath_output_tl
3858   \tl_gclear:N \g__spath_output_tl
3859 }
3860 \cs_generate_variant:Nn \spath_gshorten_at_end:Nnn {NVV, cnn, cVV, NVn}
3861 \cs_new_protected_nopar:Npn \spath_gshorten_at_end:Nn #1#2
3862 {
3863   \spath_gshorten_at_end:NVn #1#1{#2}
3864 }
3865 \cs_generate_variant:Nn \spath_gshorten_at_end:Nn {cn, cV, NV}

```

(End of 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.

```

3866 \cs_new_protected_nopar:Npn \__spath_shorten_at_start:nn #1#2
3867 {
3868   \int_compare:nTF
3869   {
3870     \tl_count:n {#1} > 3
3871   }
3872   {
3873     \group_begin:
3874     \tl_set:Nn \l__spath_tmpa_tl {#1}
3875
3876     \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {4}}
3877
3878     \tl_clear:N \l__spath_tmpe_tl
3879
3880     \tl_if_eq:NNTF \l__spath_tmpb_tl \c_spath_curvetoa_tl
3881     {
3882       \int_set:Nn \l__spath_tmpe_int {3}
3883     }
3884     {
3885       \int_set:Nn \l__spath_tmpe_int {1}
3886     }
3887
3888     \tl_set_eq:NN \l__spath_tmpe_tl \c_spath_moveto_tl
3889     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3890

```

```

3891 \prg_replicate:nn { \l__spath_tmpa_int }
3892 {
3893   \__spath_tl_put_right_braced:Nx
3894   \l__spath_tmpe_tl
3895   {\tl_item:Nn \l__spath_tmpa_tl {1}}
3896   \__spath_tl_put_right_braced:Nx
3897   \l__spath_tmpe_tl
3898   {\tl_item:Nn \l__spath_tmpa_tl {2}}
3899   \tl_put_right:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpa_tl {3}}
3900
3901   \prg_replicate:nn {3}
3902   {
3903     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
3904   }
3905 }
3906 \__spath_tl_put_right_braced:Nx
3907 \l__spath_tmpe_tl
3908 {\tl_item:Nn \l__spath_tmpa_tl {1}}
3909 \__spath_tl_put_right_braced:Nx
3910 \l__spath_tmpe_tl
3911 {\tl_item:Nn \l__spath_tmpa_tl {2}}
3912
3913 \fp_set:Nn \l__spath_tmpa_fp
3914 {
3915   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {5}}
3916   -
3917   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {2}}
3918 }
3919
3920 \fp_set:Nn \l__spath_tmpb_fp
3921 {
3922   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {6}}
3923   -
3924   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {3}}
3925 }
3926
3927 \fp_set:Nn \l__spath_tmpe_fp
3928 {
3929   sqrt(
3930     \l__spath_tmpa_fp * \l__spath_tmpa_fp
3931     +
3932     \l__spath_tmpb_fp * \l__spath_tmpb_fp
3933   )
3934   *
3935   \l__spath_tmpa_int
3936 }
3937
3938 \fp_compare:nTF
3939 {
3940   \l__spath_tmpe_fp > #2
3941 }
3942 {
3943
3944   \fp_set:Nn \l__spath_tmpe_fp

```



```

3945     {
3946     #2/ \l__spath_tmpe_fp
3947     }
3948
3949     \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_curvetoa_tl
3950     {
3951     \spath_split_curve:NNVV
3952     \l__spath_tmpe_tl
3953     \l__spath_tmpe_tl
3954     \l__spath_tmpe_tl
3955     \l__spath_tmpe_fp
3956     }
3957     {
3958     \spath_split_line:NNVV
3959     \l__spath_tmpe_tl
3960     \l__spath_tmpe_tl
3961     \l__spath_tmpe_tl
3962     \l__spath_tmpe_fp
3963     }
3964
3965     \prg_replicate:nn {2}
3966     {
3967     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3968     }
3969
3970     \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3971
3972     }
3973     {
3974
3975     \tl_put_left:NV \l__spath_tmpe_tl \c_spath_moveto_tl
3976
3977     \int_compare:nT
3978     {
3979     \tl_count:N \l__spath_tmpe_tl > 3
3980     }
3981     {
3982     \dim_set:Nn \l__spath_tmpe_dim {\fp_to_dim:n {#2 - \l__spath_tmpe_fp } }
3983     \spath_shorten_at_start:NV \l__spath_tmpe_tl \l__spath_tmpe_dim
3984     }
3985     }
3986
3987     \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
3988     \group_end:
3989     }
3990     {
3991     \tl_gset:Nn \g__spath_output_tl {#1}
3992     }
3993     }
3994     \cs_new_protected_nopar:Npn \spath_shorten_at_start:Nnn #1#2#3
3995     {
3996     \__spath_shorten_at_start:nn {#2}{#3}
3997     \tl_set_eq:NN #1 \g__spath_output_tl
3998     \tl_gclear:N \g__spath_output_tl

```

```

3999 }
4000 \cs_generate_variant:Nn \spath_shorten_at_start:Nnn {NVV, cnn, cVV, NVn}
4001 \cs_new_protected_nopar:Npn \spath_shorten_at_start:Nn #1#2
4002 {
4003   \spath_shorten_at_start:NVn #1#1{#2}
4004 }
4005 \cs_generate_variant:Nn \spath_shorten_at_start:Nn {cn, cV, NV}
4006 \cs_new_protected_nopar:Npn \spath_gshorten_at_start:Nnn #1#2#3
4007 {
4008   \__spath_shorten_at_start:nn {#2}{#3}
4009   \tl_gset_eq:NN #1 \g__spath_output_tl
4010   \tl_gclear:N \g__spath_output_tl
4011 }
4012 \cs_generate_variant:Nn \spath_gshorten_at_start:Nnn {NVV, cnn, cVV, NVn}
4013 \cs_new_protected_nopar:Npn \spath_gshorten_at_start:Nn #1#2
4014 {
4015   \spath_gshorten_at_start:NVn #1#1{#2}
4016 }
4017 \cs_generate_variant:Nn \spath_gshorten_at_start:Nn {cn, cV, NV}

```

(End of 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.

```

4018 \cs_new_protected_nopar:Npn \spath_shorten_at_both_ends:Nnn #1#2#3
4019 {
4020   \spath_shorten_at_start:Nnn #1{#2}{#3}
4021   \spath_shorten_at_end:Nnn #1{#2}{#3}
4022 }
4023 \cs_new_protected_nopar:Npn \spath_shorten_at_both_ends:Nn #1#2
4024 {
4025   \spath_shorten_at_start:Nn #1{#2}
4026   \spath_shorten_at_end:Nn #1{#2}
4027 }
4028 \cs_generate_variant:Nn \spath_shorten_at_both_ends:Nn {cn, cV, NV}
4029 \cs_new_protected_nopar:Npn \spath_gshorten_at_both_ends:Nnn #1#2#3
4030 {
4031   \spath_gshorten_at_start:Nnn #1{#2}{#3}
4032   \spath_gshorten_at_end:Nnn #1{#2}{#3}
4033 }
4034 \cs_new_protected_nopar:Npn \spath_gshorten_at_both_ends:Nn #1#2
4035 {
4036   \spath_gshorten_at_start:Nn #1{#2}
4037   \spath_gshorten_at_end:Nn #1{#2}
4038 }
4039 \cs_generate_variant:Nn \spath_gshorten_at_both_ends:Nn {cn, cV, NV}

```

(End of 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.

```

4040 \cs_new_protected_nopar:Npn \__spath_point_at:nn #1#2
4041 {

```

```

4042 \group_begin:
4043 \int_set:Nn \l__spath_tmpa_int {\fp_to_int:n {floor(#2) + 1}}
4044 \fp_set:Nn \l__spath_tmpa_fp {#2 - floor(#2)}
4045
4046 \spath_segments_to_seq:Nn \l__spath_tmpa_seq {#1}
4047
4048 \int_compare:nTF
4049 {
4050   \l__spath_tmpa_int < 1
4051 }
4052 {
4053   \spath_initialpoint:Nn \l__spath_tmpc_tl {#1}
4054 }
4055 {
4056   \int_compare:nTF
4057   {
4058     \l__spath_tmpa_int > \seq_count:N \l__spath_tmpa_seq
4059   }
4060   {
4061     \spath_finalpoint:Nn \l__spath_tmpc_tl {#1}
4062   }
4063   {
4064
4065     \tl_set:Nx
4066     \l__spath_tmpa_tl
4067     {\seq_item:Nn \l__spath_tmpa_seq { \l__spath_tmpa_int } }
4068
4069     \int_compare:nTF
4070     {
4071       \tl_count:N \l__spath_tmpa_tl > 3
4072     }
4073     {
4074       \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {4}}
4075     }
4076     {
4077       \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {1}}
4078     }
4079
4080     \tl_clear:N \l__spath_tmpc_tl
4081
4082     \token_case_meaning:Nn \l__spath_tmpb_tl
4083     {
4084       \c_spath_moveto_tl
4085       {
4086         \tl_set:Nx \l__spath_tmpc_tl
4087         {
4088           {
4089             \tl_item:Nn \l__spath_tmpa_tl {2}
4090           }
4091           {
4092             \tl_item:Nn \l__spath_tmpa_tl {3}
4093           }
4094         }
4095       }

```

```

4096
4097 \c_spath_lineto_tl
4098 {
4099   \tl_set:Nx \l__spath_tmpc_tl
4100   {
4101     {\fp_to_dim:n
4102       {
4103         (1 - \l__spath_tmpa_fp) * ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4104         +
4105         \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4106       }
4107     }
4108     {\fp_to_dim:n
4109       {
4110         (1 - \l__spath_tmpa_fp) * ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4111         +
4112         \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4113       }
4114     }
4115   }
4116 }
4117
4118 \c_spath_rectsize_tl
4119 {
4120   \fp_compare:nTF
4121   {
4122     \l__spath_tmpa_fp <= .25
4123   }
4124   {
4125     \tl_set:Nx \l__spath_tmpc_tl
4126     {
4127       {\fp_to_dim:n
4128         {
4129           ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4130           +
4131           4 * \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4132         }
4133       }
4134       {\fp_to_dim:n {\tl_item:Nn \l__spath_tmpa_tl {3} } }
4135     }
4136   }
4137   {
4138     \fp_compare:nTF
4139     {
4140       \l__spath_tmpa_fp <= .5
4141     }
4142     {
4143       \tl_set:Nx \l__spath_tmpc_tl
4144       {
4145         {\fp_to_dim:n
4146           {
4147             ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4148             +
4149             ( \tl_item:Nn \l__spath_tmpa_tl {5} )

```

```

4150     }
4151   }
4152   {\fp_to_dim:n
4153     {
4154       ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4155       +
4156       (4 * (\l__spath_tmpa_fp) - 1) * ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4157     }
4158   }
4159 }
4160 }
4161 {
4162   \fp_compare:nTF
4163   {
4164     \l__spath_tmpa_fp <= .75
4165   }
4166   {
4167     \tl_set:Nx \l__spath_tmpc_tl
4168     {
4169       {\fp_to_dim:n
4170         {
4171           ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4172           +
4173           (3 - 4 * (\l__spath_tmpa_fp)) * ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4174         }
4175       }
4176       {\fp_to_dim:n
4177         {
4178           ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4179           +
4180           ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4181         }
4182       }
4183     }
4184   }
4185 }
4186 {
4187   \tl_set:Nx \l__spath_tmpc_tl
4188   {
4189     {\fp_to_dim:n
4190       {
4191         ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4192       }
4193     }
4194     {\fp_to_dim:n
4195       {
4196         ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4197         +
4198         (4 - 4 * (\l__spath_tmpa_fp)) * ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4199       }
4200     }
4201   }
4202 }
4203 }

```

```

4204     }
4205   }
4206
4207   \c_spath_closepath_tl
4208   {
4209     \tl_set:Nx \l__spath_tmpc_tl
4210     {
4211       {\fp_to_dim:n
4212         {
4213           (1 - \l__spath_tmpa_fp) * ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4214           +
4215           \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4216         }
4217       }
4218       {\fp_to_dim:n
4219         {
4220           (1 - \l__spath_tmpa_fp) * ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4221           +
4222           \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4223         }
4224       }
4225     }
4226   }
4227
4228   \c_spath_curvetoa_tl
4229   {
4230     \tl_set:Nx \l__spath_tmpc_tl
4231     {
4232       {\fp_to_dim:n
4233         {
4234           (1 - \l__spath_tmpa_fp)^3 * \tl_item:Nn \l__spath_tmpa_tl {2}
4235           + 3 * (1 - \l__spath_tmpa_fp)^2 * (\l__spath_tmpa_fp)
4236           * \tl_item:Nn \l__spath_tmpa_tl {5}
4237           + 3 * (1 - \l__spath_tmpa_fp) * (\l__spath_tmpa_fp)^2
4238           * \tl_item:Nn \l__spath_tmpa_tl {8}
4239           + (\l__spath_tmpa_fp)^3 * \tl_item:Nn \l__spath_tmpa_tl {11}
4240         }}
4241       {\fp_to_dim:n
4242         {
4243           (1 - \l__spath_tmpa_fp)^3 * \tl_item:Nn \l__spath_tmpa_tl {3}
4244           + 3 * (1 - \l__spath_tmpa_fp)^2 * (\l__spath_tmpa_fp)
4245           * \tl_item:Nn \l__spath_tmpa_tl {6}
4246           + 3 * (1 - \l__spath_tmpa_fp) * (\l__spath_tmpa_fp)^2
4247           * \tl_item:Nn \l__spath_tmpa_tl {9}
4248           + (\l__spath_tmpa_fp)^3 * \tl_item:Nn \l__spath_tmpa_tl {12}
4249         }}
4250     }
4251   }
4252 }
4253 }
4254 }
4255
4256 \tl_gclear:N \g__spath_output_tl
4257 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpc_tl

```

```

4258 \group_end:
4259 }
4260 \cs_new_protected_nopar:Npn \spath_point_at:Nnn #1#2#3
4261 {
4262   \__spath_point_at:nn {#2}{#3}
4263   \tl_set_eq:NN #1 \g__spath_output_tl
4264   \tl_gclear:N \g__spath_output_tl
4265 }
4266 \cs_generate_variant:Nn \spath_point_at:Nnn {NVn, NVV, NnV}
4267 \cs_new_protected_nopar:Npn \spath_gpoint_at:Nnn #1#2#3
4268 {
4269   \__spath_point_at:nn {#2}{#3}
4270   \tl_gset_eq:NN #1 \g__spath_output_tl
4271   \tl_gclear:N \g__spath_output_tl
4272 }
4273 \cs_generate_variant:Nn \spath_gpoint_at:Nnn {NVn, NVV, NnV}

```

(End of 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`

```

4274 \cs_new_protected_nopar:Npn \__spath_tangent_at:nn #1#2
4275 {
4276   \group_begin:
4277   \int_set:Nn \l__spath_tmpa_int {\fp_to_int:n {floor(#2) + 1}}
4278   \fp_set:Nn \l__spath_tmpa_fp {#2 - floor(#2)}
4279
4280   \spath_segments_to_seq:Nn \l__spath_tmpa_seq {#1}
4281
4282   \int_compare:nTF
4283   {
4284     \l__spath_tmpa_int < 1
4285   }
4286   {
4287     \spath_initialpoint:Nn \l__spath_tmpe_tl {#1}
4288   }
4289   {
4290     \int_compare:nTF
4291     {
4292       \l__spath_tmpa_int > \seq_count:N \l__spath_tmpa_seq
4293     }
4294     {
4295       \spath_finalpoint:Nn \l__spath_tmpe_tl {#1}
4296     }
4297     {
4298
4299       \tl_set:Nx
4300       \l__spath_tmpe_tl
4301       {\seq_item:Nn \l__spath_tmpa_seq { \l__spath_tmpe_int } }
4302
4303       \int_compare:nTF
4304       {
4305         \tl_count:N \l__spath_tmpe_tl > 3
4306       }

```

```

4307 {
4308   \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {4}}
4309 }
4310 {
4311   \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {1}}
4312 }
4313
4314 \tl_clear:N \l__spath_tmpc_tl
4315
4316 \token_case_meaning:Nn \l__spath_tmpb_tl
4317 {
4318   \c_spath_moveto_tl
4319   {
4320     \tl_set:Nx \l__spath_tmpc_tl
4321     {
4322       {
4323         \tl_item:Nn \l__spath_tmpa_tl {2}
4324       }
4325       {
4326         \tl_item:Nn \l__spath_tmpa_tl {3}
4327       }
4328     }
4329   }
4330
4331 \c_spath_lineto_tl
4332 {
4333   \tl_set:Nx \l__spath_tmpc_tl
4334   {
4335     {\fp_to_dim:n
4336      {
4337        ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4338        -
4339        ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4340      }
4341     }
4342     {\fp_to_dim:n
4343      {
4344        ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4345        -
4346        ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4347      }
4348     }
4349   }
4350 }
4351
4352 \c_spath_rectsize_tl
4353 {
4354   \fp_compare:nTF
4355   {
4356     \l__spath_tmpa_fp <= .25
4357   }
4358   {
4359     \tl_set:Nx \l__spath_tmpc_tl
4360     {

```



```

4361         {\fp_to_dim:n
4362         {
4363             \tl_item:Nn \l__spath_tmpa_tl {5}
4364         }
4365         }
4366         {Opt}
4367     }
4368 }
4369 {
4370     \fp_compare:nTF
4371     {
4372         \l__spath_tmpa_fp <= .5
4373     }
4374     {
4375         \tl_set:Nx \l__spath_tmpc_tl
4376         {
4377             {Opt}
4378             {\fp_to_dim:n
4379             {
4380                 ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4381             }
4382         }
4383     }
4384 }
4385 {
4386     \fp_compare:nTF
4387     {
4388         \l__spath_tmpa_fp <= .75
4389     }
4390     {
4391         \tl_set:Nx \l__spath_tmpc_tl
4392         {
4393             {\fp_to_dim:n
4394             {
4395                 -( \tl_item:Nn \l__spath_tmpa_tl {5} )
4396             }
4397             }
4398             {Opt}
4399         }
4400     }
4401 }
4402 {
4403     \tl_set:Nx \l__spath_tmpc_tl
4404     {
4405         {Opt}
4406         {\fp_to_dim:n
4407         {
4408             - ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4409         }
4410     }
4411 }
4412 }
4413 }
4414 }

```

```

4415     }
4416
4417 \c_spath_closepath_tl
4418 {
4419   \tl_set:Nx \l__spath_tmpc_tl
4420   {
4421     {\fp_to_dim:n
4422       {
4423         ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4424         -
4425         ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4426       }
4427     }
4428     {\fp_to_dim:n
4429       {
4430         ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4431         -
4432         ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4433       }
4434     }
4435   }
4436 }
4437
4438 \c_spath_curvetoa_tl
4439 {
4440   \tl_set:Nx \l__spath_tmpc_tl
4441   {
4442     {\fp_to_dim:n
4443       {
4444         3*(1 - \l__spath_tmpa_fp)^2 * (\tl_item:Nn \l__spath_tmpa_tl {5}
4445         - \tl_item:Nn \l__spath_tmpa_tl {2})
4446         + 6 * (1 - \l__spath_tmpa_fp) * (\l__spath_tmpa_fp) *
4447         (\tl_item:Nn \l__spath_tmpa_tl {8}
4448         - \tl_item:Nn \l__spath_tmpa_tl {5})
4449         + 3*(\l__spath_tmpa_fp)^2 * (\tl_item:Nn \l__spath_tmpa_tl {11}
4450         - \tl_item:Nn \l__spath_tmpa_tl {8})
4451       }
4452     }
4453     {\fp_to_dim:n
4454       {
4455         3*(1 - \l__spath_tmpa_fp)^2 * (\tl_item:Nn \l__spath_tmpa_tl {6}
4456         - \tl_item:Nn \l__spath_tmpa_tl {3})
4457         + 6 * (1 - \l__spath_tmpa_fp) * (\l__spath_tmpa_fp) *
4458         (\tl_item:Nn \l__spath_tmpa_tl {9}
4459         - \tl_item:Nn \l__spath_tmpa_tl {6})
4460         + 3*(\l__spath_tmpa_fp)^2 * (\tl_item:Nn \l__spath_tmpa_tl {12}
4461         - \tl_item:Nn \l__spath_tmpa_tl {9})
4462       }}
4463     }
4464   }
4465 }
4466 }
4467 }
4468

```

```

4469 \tl_gclear:N \g__spath_output_tl
4470 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpc_tl
4471 \group_end:
4472 }
4473 \cs_new_protected_nopar:Npn \spath_tangent_at:Nnn #1#2#3
4474 {
4475   \__spath_tangent_at:nn {#2}{#3}
4476   \tl_set_eq:NN #1 \g__spath_output_tl
4477   \tl_gclear:N \g__spath_output_tl
4478 }
4479 \cs_generate_variant:Nn \spath_tangent_at:Nnn {NVn, NVV, NnV}
4480 \cs_new_protected_nopar:Npn \spath_gtangent_at:Nnn #1#2#3
4481 {
4482   \__spath_tangent_at:nn {#2}{#3}
4483   \tl_gset_eq:NN #1 \g__spath_output_tl
4484   \tl_gclear:N \g__spath_output_tl
4485 }
4486 \cs_generate_variant:Nn \spath_gtangent_at:Nnn {NVn, NVV, NnV}

```

(End of 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`

```

4487 \cs_new_protected_nopar:Npn \__spath_transformation_at:nn #1#2
4488 {
4489   \group_begin:
4490   \tl_clear:N \l__spath_tmpa_tl
4491   \__spath_tangent_at:nn {#1}{#2}
4492   \tl_set_eq:NN \l__spath_tmpb_tl \g__spath_output_tl
4493   \fp_set:Nn \l__spath_tmpa_fp
4494   {
4495     sqrt(
4496       (\tl_item:Nn \l__spath_tmpb_tl {1})^2
4497       +
4498       (\tl_item:Nn \l__spath_tmpb_tl {2})^2
4499     )
4500   }
4501   \fp_compare:nTF {\l__spath_tmpa_fp = 0}
4502   {
4503     \fp_set:Nn \l__spath_tmpa_fp {1}
4504     \fp_set:Nn \l__spath_tmpb_fp {0}
4505   }
4506   {
4507     \fp_set:Nn \l__spath_tmpb_fp
4508     { (\tl_item:Nn \l__spath_tmpb_tl {2}) / \l__spath_tmpa_fp }
4509     \fp_set:Nn \l__spath_tmpa_fp
4510     { (\tl_item:Nn \l__spath_tmpb_tl {1}) / \l__spath_tmpa_fp }
4511   }
4512   \tl_set:Nx \l__spath_tmpa_tl
4513   {
4514     { \fp_to_decimal:n { \l__spath_tmpa_fp } }
4515     { \fp_to_decimal:n { \l__spath_tmpb_fp } }
4516     { \fp_to_decimal:n {- \l__spath_tmpb_fp } }
4517     { \fp_to_decimal:n { \l__spath_tmpa_fp } }

```

```

4518 }
4519 \__spath_point_at:nn {#1}{#2}
4520 \tl_put_right:NV \l__spath_tmpa_tl \g__spath_output_tl
4521 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
4522 \group_end:
4523 }
4524 \cs_new_protected_nopar:Npn \spath_transformation_at:Nnn #1#2#3
4525 {
4526 \__spath_transformation_at:nn {#2}{#3}
4527 \tl_set_eq:NN #1 \g__spath_output_tl
4528 \tl_gclear:N \g__spath_output_tl
4529 }
4530 \cs_generate_variant:Nn \spath_transformation_at:Nnn {NVn, NVV, NnV, NvV}
4531 \cs_new_protected_nopar:Npn \spath_gtransformation_at:Nnn #1#2#3
4532 {
4533 \__spath_transformation_at:nn {#2}{#3}
4534 \tl_gset_eq:NN #1 \g__spath_output_tl
4535 \tl_gclear:N \g__spath_output_tl
4536 }
4537 \cs_generate_variant:Nn \spath_gtransformation_at:Nnn {NVn, NVV, NnV}

```

(End of 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
4538 \cs_new_protected_nopar:Npn \spath_intersect:NN #1#2
4539 {
4540 \pgfintersectionofpaths%
4541 {%
4542 \pgfsetpath #1
4543 }{%
4544 \pgfsetpath #2
4545 }
4546 }
4547 \cs_new_protected_nopar:Npn \spath_intersect:nn #1#2
4548 {
4549 \tl_set:Nn \l__spath_intersecta_tl {#1}
4550 \tl_set:Nn \l__spath_intersectb_tl {#2}
4551 \spath_intersect:NN \l__spath_intersecta_tl \l__spath_intersectb_tl
4552 }

```

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

`\spath_split_component_at_intersections:Nmn` 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.

```

4553 \cs_new_protected_nopar:Npn \__spath_split_component_at_intersections:nn #1#2
4554 {
4555 \group_begin:

```

```

4556 \tl_clear:N \l__spath_tmpe_tl
4557 \seq_clear:N \l__spath_tmpb_seq
4558
4559 % Find the intersections of these segments
4560 \tl_set:Nn \l__spath_tmpb_tl {#1}
4561 \tl_set:Nn \l__spath_tmpe_tl {#2}
4562
4563 % Remember if the component is closed
4564 \spath_finalaction:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
4565
4566 \bool_set:Nn \l__spath_closed_bool
4567 {
4568   \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_closepath_tl
4569   ||
4570   \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_rectcorner_tl
4571 }
4572
4573 % Open it
4574 \spath_open:N \l__spath_tmpe_tl
4575
4576 \spath_reallength:NV \l__spath_tmpe_int \l__spath_tmpe_tl
4577
4578 % Sort intersections along the component
4579 \pgfintersectionsorthbyfirstpath
4580 \spath_intersect:NN \l__spath_tmpe_tl \l__spath_tmpe_tl
4581
4582 % If we get intersections
4583 \int_compare:nT {\pgfintersectionsolutions > 0}
4584 {
4585   % Find the times of the intersections on the component
4586   \int_step_inline:nnnn {1} {1} {\pgfintersectionsolutions}
4587   {
4588     \pgfintersectiongetsolutiontimes{##1}{\l__spath_tmpe_tl}{\l__spath_tmpe_tl}
4589     \seq_put_left:NV \l__spath_tmpb_seq \l__spath_tmpe_tl
4590   }
4591
4592   \seq_get_left:NN \l__spath_tmpb_seq \l__spath_tmpe_tl
4593   \fp_compare:nT
4594   {
4595     \l__spath_tmpe_tl > \l__spath_tmpe_int - .01
4596   }
4597   {
4598     \bool_set_false:N \l__spath_closed_bool
4599   }
4600
4601   \seq_get_right:NN \l__spath_tmpb_seq \l__spath_tmpe_tl
4602   \fp_compare:nT
4603   {
4604     \l__spath_tmpe_tl < .01
4605   }
4606   {
4607     \bool_set_false:N \l__spath_closed_bool
4608   }
4609

```

```

4610 \tl_set:Nn \l__spath_tmpg_tl {-1}
4611
4612 \seq_map_inline:Nn \l__spath_tmpb_seq
4613 {
4614   \tl_set:Nn \l__spath_tmph_tl {##1}
4615
4616   \tl_set_eq:NN \l__spath_tmpha_tl \l__spath_tmph_tl
4617   \int_compare:nT
4618   {
4619     \fp_to_int:n {floor( \l__spath_tmph_tl ) }
4620     =
4621     \fp_to_int:n {floor( \l__spath_tmpg_tl ) }
4622   }
4623   {
4624     \tl_set:Nx \l__spath_tmph_tl
4625     {
4626       \fp_eval:n {
4627         floor( \l__spath_tmph_tl )
4628         +
4629         ( \l__spath_tmph_tl - floor( \l__spath_tmph_tl ) )
4630         /
4631         ( \l__spath_tmpg_tl - floor( \l__spath_tmpg_tl ) )
4632       }
4633     }
4634   }
4635   \tl_set_eq:NN \l__spath_tmpg_tl \l__spath_tmpha_tl
4636
4637   \spath_split_at:NNVV
4638   \l__spath_tmpd_tl
4639   \l__spath_tmpha_tl
4640   \l__spath_tmph_tl
4641   \l__spath_tmph_tl
4642
4643   \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpha_tl
4644   \tl_set_eq:NN \l__spath_tmph_tl \l__spath_tmpe_tl
4645
4646 }
4647
4648
4649 \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmph_tl
4650
4651 \spath_remove_empty_components:N \l__spath_tmpe_tl
4652
4653 \tl_set_eq:NN \l__spath_tmph_tl \l__spath_tmpe_tl
4654 }
4655
4656
4657 \bool_if:NT \l__spath_closed_bool
4658 {
4659   \spath_join_component:Nn \l__spath_tmph_tl {1}
4660 }
4661
4662 \tl_gclear:N \g__spath_output_tl
4663 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmph_tl

```

```

4664
4665 \group_end:
4666 }
4667 \cs_new_protected_nopar:Npn \spath_split_component_at_intersections:Nnn #1#2#3
4668 {
4669 \__spath_split_component_at_intersections:nn {#2}{#3}
4670 \tl_set_eq:NN #1 \g__spath_output_tl
4671 \tl_gclear:N \g__spath_output_tl
4672 }
4673 \cs_generate_variant:Nn \spath_split_component_at_intersections:Nnn {NVn, NVV}
4674 \cs_new_protected_nopar:Npn \spath_split_component_at_intersections:Nn #1#2
4675 {
4676 \spath_split_component_at_intersections:NVn #1#1{#2}
4677 }
4678 \cs_generate_variant:Nn \spath_split_component_at_intersections:Nn {cn, cv}
4679 \cs_new_protected_nopar:Npn \spath_gsplit_component_at_intersections:Nnn #1#2#3
4680 {
4681 \__spath_split_component_at_intersections:nn {#2}{#3}
4682 \tl_gset_eq:NN #1 \g__spath_output_tl
4683 \tl_gclear:N \g__spath_output_tl
4684 }
4685 \cs_generate_variant:Nn \spath_gsplit_component_at_intersections:Nnn {NVn, NVV}
4686 \cs_new_protected_nopar:Npn \spath_gsplit_component_at_intersections:Nn #1#2
4687 {
4688 \spath_gsplit_component_at_intersections:NVn #1#1{#2}
4689 }
4690 \cs_generate_variant:Nn \spath_gsplit_component_at_intersections:Nn {cn, cv}

```

(End of 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_gsplit_path_at_intersections:Nn
\spath_split_at_intersections:NNnn
\spath_split_at_intersections:NN
\spath_gsplit_at_intersections:NNnn
\spath_gsplit_at_intersections:NN
4691 \cs_new_protected_nopar:Npn \__spath_split_path_at_intersections:nn #1#2
4692 {
4693 \group_begin:
4694
4695 \seq_clear:N \l__spath_tmpa_seq
4696 \seq_clear:N \l__spath_tmpb_seq
4697
4698 \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
4699 \seq_map_inline:Nn \l__spath_tmpa_seq
4700 {
4701 \spath_split_component_at_intersections:Nnn \l__spath_tmpa_tl {##1} {#2}
4702 \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
4703 }
4704
4705 \tl_gclear:N \g__spath_output_tl
4706 \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpb_seq {}}
4707 \group_end:
4708 }
4709 \cs_new_protected_nopar:Npn \spath_split_path_at_intersections:Nnn #1#2#3
4710 {
4711 \__spath_split_path_at_intersections:nn {#2}{#3}
4712 \tl_set_eq:NN #1 \g__spath_output_tl

```

```

4713 \tl_gclear:N \g__spath_output_tl
4714 }
4715 \cs_generate_variant:Nn \spath_split_path_at_intersections:Nnn
4716 {NVn, NVV, cVn, cVV, cvn, cvv}
4717 \cs_new_protected_nopar:Npn \spath_split_path_at_intersections:Nn #1#2
4718 {
4719 \spath_split_path_at_intersections:NVn #1#1{#2}
4720 }
4721 \cs_generate_variant:Nn \spath_split_path_at_intersections:Nn {cv, NV}
4722 \cs_new_protected_nopar:Npn \spath_gsplit_path_at_intersections:Nnn #1#2#3
4723 {
4724 \__spath_split_path_at_intersections:nn {#2}{#3}
4725 \tl_gset_eq:NN #1 \g__spath_output_tl
4726 \tl_gclear:N \g__spath_output_tl
4727 }
4728 \cs_generate_variant:Nn \spath_gsplit_path_at_intersections:Nnn
4729 {NVn, NVV, cVn, cVV, cvn, cvv}
4730 \cs_new_protected_nopar:Npn \spath_gsplit_path_at_intersections:Nn #1#2
4731 {
4732 \spath_gsplit_path_at_intersections:NVn #1#1{#2}
4733 }
4734 \cs_generate_variant:Nn \spath_gsplit_path_at_intersections:Nn {cv, NV}
4735 \cs_new_protected_nopar:Npn \spath_split_at_intersections:NNnn #1#2#3#4
4736 {
4737 \__spath_split_path_at_intersections:nn {#3}{#4}
4738 \tl_set_eq:NN #1 \g__spath_output_tl
4739 \__spath_split_path_at_intersections:nn {#4}{#3}
4740 \tl_set_eq:NN #2 \g__spath_output_tl
4741 \tl_gclear:N \g__spath_output_tl
4742 }
4743 \cs_generate_variant:Nn \spath_split_at_intersections:NNnn
4744 {NNVn, NNVV, ccVn, ccVV, ccvn, ccvv}
4745 \cs_new_protected_nopar:Npn \spath_split_at_intersections:NN #1#2
4746 {
4747 \spath_split_at_intersections:NNVV #1#2#1#2
4748 }
4749 \cs_generate_variant:Nn \spath_split_at_intersections:NN {cc}
4750 \cs_new_protected_nopar:Npn \spath_gsplit_at_intersections:NNnn #1#2#3#4
4751 {
4752 \__spath_split_path_at_intersections:nn {#3}{#4}
4753 \tl_gset_eq:NN #1 \g__spath_output_tl
4754 \__spath_split_path_at_intersections:nn {#4}{#3}
4755 \tl_gset_eq:NN #2 \g__spath_output_tl
4756 \tl_gclear:N \g__spath_output_tl
4757 }
4758 \cs_generate_variant:Nn \spath_gsplit_at_intersections:NNnn
4759 {NNVn, NNVV, ccVn, ccVV, ccvn, ccvv}
4760 \cs_new_protected_nopar:Npn \spath_gsplit_at_intersections:NN #1#2
4761 {
4762 \spath_gsplit_at_intersections:NNVV #1#2#1#2
4763 }
4764 \cs_generate_variant:Nn \spath_gsplit_at_intersections:NN {cc}

```

(End of 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.

```
4765 \cs_new_protected_nopar:Npn \__spath_split_component_at_self_intersections:n #1
4766 {
4767   \group_begin:
4768   \tl_set:Nn \l__spath_tmpe_tl {#1}
4769
4770   % Remember if the component is closed
4771   \spath_finalaction:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
4772
4773   \bool_set:Nn \l__spath_closed_bool
4774   {
4775     \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_closepath_tl
4776   }
4777
4778   % Copy the path
4779   \tl_set:Nn \l__spath_tmpe_tl {#1}
4780
4781   % Open the path
4782   \spath_open:N \l__spath_tmpe_tl
4783   % Ensure beziers don't self-intersect
4784   \spath_split_curves:N \l__spath_tmpe_tl
4785
4786   % Make a copy for later
4787   \tl_set_eq:NN \l__spath_tmpg_tl \l__spath_tmpe_tl
4788
4789   % Clear some token lists and sequences
4790   \tl_clear:N \l__spath_tmpd_tl
4791   \seq_clear:N \l__spath_tmpb_seq
4792   \int_zero:N \l__spath_tmpe_int
4793
4794   \pgfintersectionsortbyfirstpath
4795
4796   % Split the path into a sequence of segments
4797   \spath_segments_to_seq:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
4798
4799   \seq_map_indexed_inline:Nn \l__spath_tmpe_tl
4800   {
4801     \seq_map_indexed_inline:Nn \l__spath_tmpe_tl
4802     {
4803       % Don't intersect a segment with itself
4804       \int_compare:nF
4805       {
4806         ##1 == #####1
4807       }
4808       {
4809         \spath_intersect:nn {##2} {#####2}
4810
4811         \int_compare:nT {\pgfintersectionsolutions > 0}
4812         {
4813           % Find the times of the intersections on each path
4814           \int_step_inline:nnnn {1} {1} {\pgfintersectionsolutions}
4815           {
4816             \pgfintersectiongetsolutiontimes
4817             {#####1}{\l__spath_tmpb_tl}{\l__spath_tmpe_tl}
```

```

4818
4819 \bool_if:nT
4820 {
4821   !(
4822     \fp_compare_p:n { \l__spath_tmpb_tl > .99 }
4823     &&
4824     \int_compare_p:n {##1 + 1 == ###1}
4825   )
4826   &&
4827   !(
4828     \fp_compare_p:n { \l__spath_tmpb_tl < .01 }
4829     &&
4830     \int_compare_p:n {##1 - 1 == ###1}
4831   )
4832   &&
4833   !(
4834     \l__spath_closed_bool
4835     &&
4836     \fp_compare_p:n { \l__spath_tmpb_tl < .01 }
4837     &&
4838     \int_compare_p:n {##1 == 1}
4839     &&
4840     \int_compare_p:n {\seq_count:N \l__spath_tmpa_seq == ###1}
4841   )
4842   &&
4843   !(
4844     \l__spath_closed_bool
4845     &&
4846     \fp_compare_p:n { \l__spath_tmpb_tl > .99 }
4847     &&
4848     \int_compare_p:n {####1 == 1}
4849     &&
4850     \int_compare_p:n {\seq_count:N \l__spath_tmpa_seq == ##1}
4851   )
4852 }
4853 {
4854   \tl_set:Nx \l__spath_tmpa_tl
4855   {\fp_to_decimal:n {\l__spath_tmpb_tl + ##1 - 1}}
4856   \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
4857 }
4858 }
4859 }
4860 }
4861 }
4862 }
4863
4864 % Sort the sequence by reverse order along the path
4865 \seq_sort:Nn \l__spath_tmpb_seq
4866 {
4867   \fp_compare:nNnTF { ##1 } < { ##2 }
4868   { \sort_return_swapped: }
4869   { \sort_return_same: }
4870 }
4871

```

```

4872 \seq_get_left:NN \l__spath_tmpb_seq \l__spath_tmpa_tl
4873 \fp_compare:nT
4874 {
4875   \l__spath_tmpa_tl > \seq_count:N \l__spath_tmpa_seq - .01
4876 }
4877 {
4878   \bool_set_false:N \l__spath_closed_bool
4879 }
4880 \seq_get_right:NN \l__spath_tmpb_seq \l__spath_tmpa_tl
4881 \fp_compare:nT
4882 {
4883   \l__spath_tmpa_tl < .01
4884 }
4885 {
4886   \bool_set_false:N \l__spath_closed_bool
4887 }
4888
4889 % Restore the original copy of the path
4890 \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpg_tl
4891
4892 % Clear the token lists
4893 \tl_clear:N \l__spath_tmpf_tl
4894 \tl_clear:N \l__spath_tmph_tl
4895 \tl_clear:N \l__spath_tmpg_tl
4896
4897 \tl_set:Nn \l__spath_tmpe_tl {-1}
4898
4899 \seq_map_inline:Nn \l__spath_tmpb_seq
4900 {
4901   \tl_set:Nn \l__spath_tmpe_tl {##1}
4902   \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpe_tl
4903   \int_compare:nT
4904   {
4905     \fp_to_int:n {floor( \l__spath_tmpe_tl ) }
4906     =
4907     \fp_to_int:n {floor( \l__spath_tmpe_tl ) }
4908   }
4909   {
4910     \tl_set:Nx \l__spath_tmpe_tl
4911     {
4912       \fp_eval:n {
4913         floor( \l__spath_tmpe_tl )
4914         +
4915         ( \l__spath_tmpe_tl - floor( \l__spath_tmpe_tl ) )
4916         /
4917         ( \l__spath_tmpe_tl - floor( \l__spath_tmpe_tl ) )
4918       }
4919     }
4920   }
4921   \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpe_tl
4922
4923   \spath_split_at:NNVV
4924   \l__spath_tmpe_tl
4925   \l__spath_tmph_tl

```

```

4926     \l__spath_tmpe_tl
4927     \l__spath_tmpb_tl
4928
4929     \tl_put_left:NV \l__spath_tmpg_tl \l__spath_tmph_tl
4930     \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpf_tl
4931
4932   }
4933
4934   \tl_put_left:NV \l__spath_tmpg_tl \l__spath_tmpe_tl
4935
4936   \tl_if_empty:NT \l__spath_tmpg_tl
4937   {
4938     \tl_set_eq:NN \l__spath_tmpg_tl \l__spath_tmpe_tl
4939   }
4940
4941   \spath_remove_empty_components:N \l__spath_tmpg_tl
4942
4943   % Do something with closed
4944   \bool_if:NT \l__spath_closed_bool
4945   {
4946     \spath_join_component:Nn \l__spath_tmpg_tl {1}
4947   }
4948
4949   \tl_gclear:N \g__spath_output_tl
4950   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpg_tl
4951   \group_end:
4952 }
4953 \cs_new_protected_nopar:Npn \spath_split_component_at_self_intersections:Nn #1#2
4954 {
4955   \__spath_split_component_at_self_intersections:n {#2}
4956   \tl_set_eq:NN #1 \g__spath_output_tl
4957   \tl_gclear:N \g__spath_output_tl
4958 }
4959 \cs_generate_variant:Nn \spath_split_component_at_self_intersections:Nn {NV}
4960 \cs_new_protected_nopar:Npn \spath_split_component_at_self_intersections:N #1
4961 {
4962   \spath_split_component_at_self_intersections:NV #1#1
4963 }
4964 \cs_generate_variant:Nn \spath_split_component_at_self_intersections:N {c}
4965 \cs_new_protected_nopar:Npn \spath_gsplit_component_at_self_intersections:Nn #1#2
4966 {
4967   \__spath_split_component_at_self_intersections:n {#2}
4968   \tl_gset_eq:NN #1 \g__spath_output_tl
4969   \tl_gclear:N \g__spath_output_tl
4970 }
4971 \cs_generate_variant:Nn \spath_gsplit_component_at_self_intersections:Nn {NV}
4972 \cs_new_protected_nopar:Npn \spath_gsplit_component_at_self_intersections:N #1
4973 {
4974   \spath_gsplit_component_at_self_intersections:NV #1#1
4975 }
4976 \cs_generate_variant:Nn \spath_gsplit_component_at_self_intersections:N {c}

```

(End of 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
`\spath_split_at_self_intersections:N`
`\spath_gsplit_at_self_intersections:Nn`
`\spath_gsplit_at_self_intersections:N`

where it meets all the others and itself. To make this more efficient, we split against the components of the original path rather than updating each time.

```

4977 \cs_new_protected_nopar:Npn \__spath_split_at_self_intersections:n #1
4978 {
4979   \group_begin:
4980   \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
4981   \seq_clear:N \l__spath_tmpb_seq
4982   \seq_clear:N \l__spath_tmpc_seq
4983
4984   % Iterate over the components of the original path.
4985   \bool_do_until:nn
4986   {
4987     \seq_if_empty_p:N \l__spath_tmpa_seq
4988   }
4989   {
4990     % Get the next component
4991     \seq_pop_left:NN \l__spath_tmpa_seq \l__spath_tmpa_tl
4992     % Copy for later
4993     \tl_set_eq:NN \l__spath_tmpc_tl \l__spath_tmpa_tl
4994     \int_compare:nT
4995     {
4996       \tl_count:N \l__spath_tmpa_tl > 3
4997     }
4998     {
4999
5000       % Split against itself
5001       \spath_split_component_at_self_intersections:N \l__spath_tmpa_tl
5002       % Grab the rest of the path
5003       \tl_set:Nx \l__spath_tmpb_tl
5004       {
5005         \seq_use:Nn \l__spath_tmpb_seq {}
5006         \seq_use:Nn \l__spath_tmpa_seq {}
5007       }
5008       % Split against the rest of the path
5009       \spath_split_path_at_intersections:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
5010     }
5011     % Save the split path
5012     \seq_put_right:NV \l__spath_tmpc_seq \l__spath_tmpa_tl
5013     % Add the original copy to the sequence of processed components
5014     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpc_tl
5015   }
5016
5017   \tl_gclear:N \g__spath_output_tl
5018   \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpc_seq {} }
5019   \group_end:
5020 }
5021 \cs_generate_variant:Nn \__spath_split_at_self_intersections:n {V, v}
5022 \cs_new_protected_nopar:Npn \spath_split_at_self_intersections:Nn #1#2
5023 {
5024   \__spath_split_at_self_intersections:n {#2}
5025   \tl_set_eq:NN #1 \g__spath_output_tl
5026   \tl_gclear:N \g__spath_output_tl
5027 }
5028 \cs_generate_variant:Nn \spath_split_at_self_intersections:Nn {NV, cn, cV, cv}

```

```

5029 \cs_new_protected_nopar:Npn \spath_split_at_self_intersections:N #1
5030 {
5031   \spath_split_at_self_intersections:NV #1#1
5032 }
5033 \cs_generate_variant:Nn \spath_split_at_self_intersections:N {c}
5034 \cs_new_protected_nopar:Npn \spath_gsplit_at_self_intersections:Nn #1#2
5035 {
5036   \__spath_split_at_self_intersections:n {#2}
5037   \tl_gset_eq:NN #1 \g__spath_output_tl
5038   \tl_gclear:N \g__spath_output_tl
5039 }
5040 \cs_generate_variant:Nn \spath_gsplit_at_self_intersections:Nn {NV, cn, cV, cv}
5041 \cs_new_protected_nopar:Npn \spath_gsplit_at_self_intersections:N #1
5042 {
5043   \spath_gsplit_at_self_intersections:NV #1#1
5044 }
5045 \cs_generate_variant:Nn \spath_gsplit_at_self_intersections:N {c}

```

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

\spath_join_component:Nnn
\spath_join_component:Nn
\spath_gjoin_component:Nnn
\spath_gjoin_component:Nn

Join the specified component of the spath to its predecessor.

```

5046 \cs_new_protected_nopar:Npn \__spath_join_component:nn #1#2
5047 {
5048   \group_begin:
5049   \spath_numberofcomponents:Nn \l__spath_tmpa_int {#1}
5050
5051   \bool_if:nTF
5052   {
5053     \int_compare_p:n { #2 >= 1 }
5054     &&
5055     \int_compare_p:n { #2 <= \l__spath_tmpa_int }
5056   }
5057   {
5058     \int_compare:nTF
5059     {
5060       #2 == 1
5061     }
5062     {
5063       \int_compare:nTF
5064       {
5065         \l__spath_tmpa_int == 1
5066       }
5067       {
5068         \tl_set:Nn \l__spath_tmpa_tl {#1}
5069         \spath_initialpoint:Nn \l__spath_tmpb_tl {#1}
5070         \tl_put_right:NV \l__spath_tmpa_tl \c_spath_closepath_tl
5071         \tl_put_right:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
5072         \tl_gclear:N \g__spath_output_tl
5073         \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
5074       }
5075     }
5076     \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
5077     \seq_pop_left:NN \l__spath_tmpa_seq \l__spath_tmpa_tl
5078

```

```

5079     \prg_replicate:nn {3}
5080     {
5081       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
5082     }
5083
5084     \seq_put_right:NV \l__spath_tmpa_seq \l__spath_tmpa_tl
5085
5086     \tl_gclear:N \g__spath_output_tl
5087     \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpa_seq {}}
5088   }
5089 }
5090 {
5091   \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
5092
5093   \seq_clear:N \l__spath_tmpb_seq
5094   \seq_map_indexed_inline:Nn \l__spath_tmpa_seq
5095   {
5096     \tl_set:Nn \l__spath_tmpa_tl {##2}
5097     \int_compare:nT {##1 = #2}
5098     {
5099       \prg_replicate:nn {3}
5100       {
5101         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
5102       }
5103     }
5104     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
5105   }
5106
5107   \tl_gclear:N \g__spath_output_tl
5108   \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpb_seq {}}
5109 }
5110 }
5111 {
5112   \tl_gclear:N \g__spath_output_tl
5113   \tl_gset:Nn \g__spath_output_tl {#1}
5114 }
5115
5116 \group_end:
5117 }
5118 \cs_new_protected_nopar:Npn \spath_join_component:Nnn #1#2#3
5119 {
5120   \__spath_join_component:nn {#2}{#3}
5121   \tl_set_eq:NN #1 \g__spath_output_tl
5122   \tl_gclear:N \g__spath_output_tl
5123 }
5124 \cs_generate_variant:Nn \spath_join_component:Nnn {NVn, NVV}
5125 \cs_new_protected_nopar:Npn \spath_join_component:Nn #1#2
5126 {
5127   \spath_join_component:NVn #1#1{#2}
5128 }
5129 \cs_generate_variant:Nn \spath_join_component:Nn {cn, NV, cV}
5130 \cs_new_protected_nopar:Npn \spath_gjoin_component:Nnn #1#2#3
5131 {
5132   \__spath_join_component:nn {#2}{#3}

```

```

5133 \tl_gset_eq:NN #1 \g__spath_output_tl
5134 \tl_gclear:N \g__spath_output_tl
5135 }
5136 \cs_generate_variant:Nn \spath_gjoin_component:Nnn {NVn, NVV}
5137 \cs_new_protected_nopar:Npn \spath_gjoin_component:Nn #1#2
5138 {
5139   \spath_gjoin_component:NVn #1#1{#2}
5140 }
5141 \cs_generate_variant:Nn \spath_gjoin_component:Nn {cn, NV, cV}

```

(End of 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
5142 \cs_new_protected_nopar:Npn \__spath_spot_weld_components:n #1
5143 {
5144   \group_begin:
5145   \dim_zero:N \l__spath_move_x_dim
5146   \dim_zero:N \l__spath_move_y_dim
5147
5148   \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
5149   \seq_clear:N \l__spath_tmpb_seq
5150   \dim_set:Nn \l__spath_move_x_dim {\tl_item:nn {#1} {2} + 10 pt}
5151   \dim_set:Nn \l__spath_move_y_dim {\tl_item:nn {#1} {3} + 10 pt}
5152
5153   \int_set:Nn \l__spath_tmpa_int {\seq_count:N \l__spath_tmpa_seq}
5154
5155   \seq_map_inline:Nn \l__spath_tmpa_seq
5156   {
5157     \tl_set:Nn \l__spath_tmpa_tl {##1}
5158     \bool_if:nT
5159     {
5160       \dim_compare_p:n
5161       {
5162         \dim_abs:n
5163         {\l__spath_move_x_dim - \tl_item:Nn \l__spath_tmpa_tl {2} }
5164         < 0.01pt
5165       }
5166       &&
5167       \dim_compare_p:n
5168       {
5169         \dim_abs:n
5170         {\l__spath_move_y_dim - \tl_item:Nn \l__spath_tmpa_tl {3} }
5171         < 0.01pt
5172       }
5173     }
5174     {
5175       \prg_replicate:nn {3}
5176       {
5177         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
5178       }
5179       \int_decr:N \l__spath_tmpa_int
5180     }
5181     \tl_reverse:N \l__spath_tmpa_tl

```



```

5182 \dim_set:Nn \l__spath_move_x_dim {\tl_item:Nn \l__spath_tmpa_tl {2}}
5183 \dim_set:Nn \l__spath_move_y_dim {\tl_item:Nn \l__spath_tmpa_tl {1}}
5184 \tl_reverse:N \l__spath_tmpa_tl
5185 \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
5186 }
5187
5188 \tl_set:Nx \l__spath_tmpa_tl {\seq_use:Nn \l__spath_tmpb_seq {}}
5189 \spath_components_to_seq:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
5190
5191
5192 \spath_initialpoint:Nn \l__spath_tmpa_tl {#1}
5193 \spath_finalpoint:Nn \l__spath_tmpb_tl {#1}
5194
5195 \bool_if:nT
5196 {
5197   \dim_compare_p:n
5198   {
5199     \dim_abs:n
5200     {
5201       \tl_item:Nn \l__spath_tmpa_tl {1} - \tl_item:Nn \l__spath_tmpb_tl {1}
5202     }
5203     <
5204     0.01pt
5205   }
5206   &&
5207   \dim_compare_p:n
5208   {
5209     \dim_abs:n
5210     {
5211       \tl_item:Nn \l__spath_tmpa_tl {2} - \tl_item:Nn \l__spath_tmpb_tl {2}
5212     }
5213     <
5214     0.01pt
5215   }
5216 }
5217 {
5218   \int_compare:nTF
5219   {
5220     \seq_count:N \l__spath_tmpb_seq > 1
5221   }
5222   {
5223     \seq_pop_left:NN \l__spath_tmpb_seq \l__spath_tmpb_tl
5224
5225     \prg_replicate:nn {3}
5226     {
5227       \tl_set:Nx \l__spath_tmpb_tl {\tl_tail:N \l__spath_tmpb_tl}
5228     }
5229     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpb_tl
5230   }
5231   {
5232     \tl_set:NV \l__spath_tmpb_tl \c_spath_closepath_tl
5233     \tl_put_right:Nx \l__spath_tmpb_tl
5234     {
5235       { \tl_item:Nn \l__spath_tmpa_tl {1} }

```

```

5236     { \tl_item:Nn \l__spath_tmpa_tl {2} }
5237   }
5238   \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpb_tl
5239 }
5240 }
5241
5242 \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpb_seq {}}
5243 \group_end:
5244 }
5245 \cs_new_protected_nopar:Npn \spath_spot_weld_components:Nn #1#2
5246 {
5247   \__spath_spot_weld_components:n {#2}
5248   \tl_set_eq:NN #1 \g__spath_output_tl
5249   \tl_gclear:N \g__spath_output_tl
5250 }
5251 \cs_generate_variant:Nn \spath_spot_weld_components:Nn {NV, cV, cn}
5252 \cs_new_protected_nopar:Npn \spath_spot_weld_components:N #1
5253 {
5254   \spath_spot_weld_components:NV #1#1
5255 }
5256 \cs_generate_variant:Nn \spath_spot_weld_components:N {c}
5257 \cs_new_protected_nopar:Npn \spath_spot_gweld_components:Nn #1#2
5258 {
5259   \__spath_spot_weld_components:n {#2}
5260   \tl_gset_eq:NN #1 \g__spath_output_tl
5261   \tl_gclear:N \g__spath_output_tl
5262 }
5263 \cs_generate_variant:Nn \spath_spot_gweld_components:Nn {NV, cV, cn}
5264 \cs_new_protected_nopar:Npn \spath_spot_gweld_components:N #1
5265 {
5266   \spath_spot_gweld_components:NV #1#1
5267 }
5268 \cs_generate_variant:Nn \spath_spot_gweld_components:N {c}

```

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

3.8 Exporting Commands

\spath_convert_to_svg:Nn
\spath_gconvert_to_svg:Nn

Convert the soft path to an SVG document.

```

5269 \cs_new_protected_nopar:Npn \__spath_convert_to_svg:n #1
5270 {
5271   \group_begin:
5272   \tl_clear:N \l__spath_tmpa_tl
5273   \tl_put_right:Nn \l__spath_tmpa_tl
5274   {
5275     <?xml~ version="1.0"~ standalone="no"?>
5276     \iow_newline:
5277     <!DOCTYPE~ svg~ PUBLIC~ "-//W3C//DTD SVG 1.1//EN"~
5278     "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
5279     \iow_newline:
5280     <svg~ xmlns="http://www.w3.org/2000/svg"~ version="1.1"~viewBox="
5281   }
5282
5283   \spath_minbb:Nn \l__spath_tmpb_tl {#1}

```

```

5284 \spath_maxbb:Nn \l__spath_tmpc_tl {#1}
5285 \tl_put_right:Nx \l__spath_tmpa_tl
5286 {
5287   \dim_to_decimal:n
5288   {
5289     \tl_item:Nn \l__spath_tmpb_tl {1} - 10pt
5290   }
5291   \exp_not:n {~}
5292   \dim_to_decimal:n
5293   {
5294     \tl_item:Nn \l__spath_tmpb_tl {2} - 10pt
5295   }
5296   \exp_not:n {~}
5297   \dim_to_decimal:n
5298   {
5299     \tl_item:Nn \l__spath_tmpc_tl {1}
5300     -
5301     \tl_item:Nn \l__spath_tmpb_tl {1}
5302     + 20pt
5303   }
5304   \exp_not:n {~}
5305   \dim_to_decimal:n
5306   {
5307     \tl_item:Nn \l__spath_tmpc_tl {2}
5308     -
5309     \tl_item:Nn \l__spath_tmpb_tl {2}
5310     + 20pt
5311   }
5312 }
5313
5314 \tl_put_right:Nn \l__spath_tmpa_tl
5315 {
5316   ">
5317   \iow_newline:
5318   <path~ d="
5319 }
5320 \tl_set:Nn \l__spath_tmpc_tl {use:n}
5321 \tl_map_inline:nn {#1}
5322 {
5323   \tl_set:Nn \l__spath_tmpb_tl {##1}
5324   \token_case_meaning:NnF \l__spath_tmpb_tl
5325   {
5326     \c_spath_moveto_tl
5327     {
5328       \tl_put_right:Nn \l__spath_tmpa_tl {M~}
5329       \tl_set:Nn \l__spath_tmpc_tl {use:n}
5330     }
5331     \c_spath_lineto_tl
5332     {
5333       \tl_put_right:Nn \l__spath_tmpa_tl {L~}
5334       \tl_set:Nn \l__spath_tmpc_tl {use:n}
5335     }
5336     \c_spath_closepath_tl
5337     {

```

```

5338     \tl_put_right:Nn \l__spath_tmpa_tl {Z~}
5339     \tl_set:Nn \l__spath_tmpc_tl {use_none:n}
5340   }
5341   \c_spath_curvetoa_tl
5342   {
5343     \tl_put_right:Nn \l__spath_tmpa_tl {C~}
5344     \tl_set:Nn \l__spath_tmpc_tl {use:n}
5345   }
5346   \c_spath_curvetob_tl {
5347     \tl_set:Nn \l__spath_tmpc_tl {use:n}
5348   }
5349   \c_spath_curveto_tl {
5350     \tl_set:Nn \l__spath_tmpc_tl {use:n}
5351   }
5352 }
5353 {
5354   \tl_put_right:Nx
5355   \l__spath_tmpa_tl
5356   {\use:c { \l__spath_tmpc_tl } {\dim_to_decimal:n {##1}} ~}
5357 }
5358 }
5359 \tl_put_right:Nn \l__spath_tmpa_tl
5360 {
5361   "~ fill="none"~ stroke="black"~ />
5362   \iow_newline:
5363   </svg>
5364   \iow_newline:
5365 }
5366 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
5367 \group_end:
5368 }
5369 \cs_new_protected_nopar:Npn \spath_convert_to_svg:Nn #1#2
5370 {
5371   \__spath_convert_to_svg:n {#2}
5372   \tl_set_eq:NN #1 \g__spath_output_tl
5373   \tl_gclear:N \g__spath_output_tl
5374 }
5375 \cs_new_protected_nopar:Npn \spath_gconvert_to_svg:Nn #1#2
5376 {
5377   \__spath_convert_to_svg:n {#2}
5378   \tl_gset_eq:NN #1 \g__spath_output_tl
5379   \tl_gclear:N \g__spath_output_tl
5380 }

```

(End of 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.

```

5381 \iow_new:N \g__spath_stream
5382 \cs_new_protected_nopar:Npn \spath_export_to_svg:nn #1#2
5383 {
5384   \group_begin:
5385   \spath_convert_to_svg:Nn \l__spath_tmpa_tl {#2}
5386   \iow_open:Nn \g__spath_stream {#1 .svg}
5387   \iow_now:Nx \g__spath_stream

```

```

5388 {
5389   \tl_use:N \l__spath_tmpa_tl
5390 }
5391 \iow_close:N \g__spath_stream
5392 \group_end:
5393 }
5394 \cs_generate_variant:Nn \spath_export_to_svg:nn {nv, nV}

```

(End of definition for \spath_export_to_svg:nn.)

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

```

5395 \cs_new_protected_nopar:Npn \spath_show:n #1
5396 {
5397   \int_step_inline:nnnn {1} {3} {\tl_count:n {#1}}
5398   {
5399     \iow_term:x {
5400       \tl_item:nn {#1} {##1}
5401       {\tl_item:nn {#1} {##1+1}}
5402       {\tl_item:nn {#1} {##1+2}}
5403     }
5404   }
5405 }
5406 \cs_generate_variant:Nn \spath_show:n {V, v}

```

(End of 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`

```

5407 \cs_new_protected_nopar:Npn \spath_get_current_path:N #1
5408 {
5409   \pgfsyssoftpath@getcurrentpath #1
5410 }
5411 \cs_generate_variant:Nn \spath_get_current_path:N {c}
5412 \cs_new_protected_nopar:Npn \spath_gget_current_path:N #1
5413 {
5414   \pgfsyssoftpath@getcurrentpath #1
5415   \tl_gset_eq:NN #1 #1
5416 }
5417 \cs_generate_variant:Nn \spath_gget_current_path:N {c}

```

(End of 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.

```

5418 \cs_new_protected_nopar:Npn \spath_protocol_path:n #1
5419 {
5420   \spath_minbb:Nn \l__spath_tmpa_tl {#1}
5421   \dim_set:Nn \l__spath_tmpa_dim {\tl_item:Nn \l__spath_tmpa_tl {1}}
5422   \dim_set:Nn \l__spath_tmpb_dim {\tl_item:Nn \l__spath_tmpa_tl {2}}
5423   \pgf@protocolsizes\l__spath_tmpa_dim\l__spath_tmpb_dim

```

```

5424
5425 \spath_maxbb:Nn \l__spath_tmpa_tl {#1}
5426 \dim_set:Nn \l__spath_tmpa_dim {\tl_item:Nn \l__spath_tmpa_tl {1}}
5427 \dim_set:Nn \l__spath_tmpb_dim {\tl_item:Nn \l__spath_tmpa_tl {2}}
5428 \pgf@protocolsizes\l__spath_tmpa_dim\l__spath_tmpb_dim
5429 }
5430 \cs_generate_variant:Nn \spath_protocol_path:n {V}

```

(End of 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
5431 \cs_new_protected_nopar:Npn \spath_set_current_path:n #1
5432 {
5433   \spath_protocol_path:n {#1}
5434   \tl_set:Nn \l__spath_tmpa_tl {#1}
5435   \pgfsyssoftpath@setcurrentpath\l__spath_tmpa_tl
5436 }
5437 \cs_new_protected_nopar:Npn \spath_set_current_path:N #1
5438 {
5439   \spath_protocol_path:V #1
5440   \pgfsyssoftpath@setcurrentpath #1
5441 }
5442 \cs_generate_variant:Nn \spath_set_current_path:N {c}

```

(End of 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.

```

5443 \cs_new_protected_nopar:Npn \spath_use_path:nn #1#2
5444 {
5445   \spath_set_current_path:n {#1}
5446   \pgfusepath{#2}
5447 }

```

(End of definition for \spath_use_path:nn.)

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

```

5448 \cs_new_protected_nopar:Npn \spath_tikz_path:nn #1#2
5449 {
5450   \tl_if_empty:nF {#2}
5451   {
5452     \path[#1] \pgfextra{
5453       \spath_set_current_path:n {#2}
5454       \tl_put_left:Nn \tikz@preactions {\def\tikz@actions@path{#2}}
5455     };
5456   }
5457 }
5458 \cs_generate_variant:Nn \spath_tikz_path:nn {Vn, VV, nv, Vv, nV}

```

(End of definition for \spath_tikz_path:nn.)

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

```

5459 \cs_new_protected_nopar:Npn \spath_set_tikz_data:n #1
5460 {
5461   \spath_finalpoint:Nn \l__spath_tmpa_tl {#1}
5462   \tl_set:Nx \l__spath_tmpa_tl

```

```

5463 {
5464   \exp_not:c {pgf@x}=\tl_item:Nn \l__spath_tmpa_tl {1} \relax
5465   \exp_not:c {pgf@y}=\tl_item:Nn \l__spath_tmpa_tl {2} \relax
5466 }
5467 \use:c {pgf@process}{%
5468   \tl_use:N \l__spath_tmpa_tl
5469   \pgftransforminvert
5470   \use:c {pgf@pos@transform@glob}
5471 }
5472 \tl_set:Nx \l__spath_tmpa_tl
5473 {
5474   \exp_not:c {tikz@lastx}=\exp_not:c {pgf@x} \relax
5475   \exp_not:c {tikz@lasty}=\exp_not:c {pgf@y} \relax
5476   \exp_not:c {tikz@lastxsaved}=\exp_not:c {pgf@x} \relax
5477   \exp_not:c {tikz@lastysaved}=\exp_not:c {pgf@y} \relax
5478 }
5479 \tl_use:N \l__spath_tmpa_tl
5480 \spath_finalmovepoint:Nn \l__spath_tmpa_tl {#1}
5481 \bool_if:NT \l_spath_movetorelevant_bool
5482 {
5483   \ifpgfsyssoftpathmovetorelevant%
5484   \tl_gset_eq:cN {pgfsyssoftpath@lastmoveto} \l__spath_tmpa_tl
5485   \fi
5486 }
5487 \tl_set:Nx \l__spath_tmpa_tl
5488 {
5489   \exp_not:c {pgf@x}=\tl_item:Nn \l__spath_tmpa_tl {1} \relax
5490   \exp_not:c {pgf@y}=\tl_item:Nn \l__spath_tmpa_tl {2} \relax
5491 }
5492 \use:c {pgf@process}{%
5493   \tl_use:N \l__spath_tmpa_tl
5494   \pgftransforminvert
5495   \use:c {pgf@pos@transform@glob}
5496 }
5497 \bool_if:NT \l_spath_movetorelevant_bool
5498 {
5499   \dim_if_exist:cT {tikz@lastmovetox}
5500   {
5501     \tl_set:Nx \l__spath_tmpa_tl
5502     {
5503       \exp_not:c {tikz@lastmovetox}=\exp_not:c {pgf@x} \relax
5504       \exp_not:c {tikz@lastmovetoy}=\exp_not:c {pgf@y} \relax
5505     }
5506     \tl_use:N \l__spath_tmpa_tl
5507   }
5508 }
5509 \tl_clear_new:c {tikz@timer}
5510 \tl_set:cn {tikz@timer}
5511 {
5512   \pgftransformreset
5513   \spath_reallength:Nn \l__spath_tmpa_int {#1}
5514   \tl_set_eq:Nc \l__spath_tmpb_tl {tikz@time}
5515   \tl_set:Nx \l__spath_tmpb_tl
5516   {\fp_to_decimal:n {(\l__spath_tmpb_tl) * (\l__spath_tmpa_int)}}

```

```

5517 \spath_transformation_at:NnV \l__spath_tmpc_tl {#1} \l__spath_tmpb_tl
5518
5519 \tl_set:Nx \l__spath_tmpa_tl
5520 {
5521   \exp_not:N \pgfpoint
5522   { \tl_item:Nn \l__spath_tmpc_tl {5} }
5523   { \tl_item:Nn \l__spath_tmpc_tl {6} }
5524 }
5525 \exp_args:NV \pgftransformshift \l__spath_tmpa_tl
5526
5527 \ifpgfresetnontranslationatime
5528 \pgftransformresetnontranslations
5529 \fi
5530
5531 \ifpgfslopedatime
5532
5533 \tl_set:Nx \l__spath_tmpa_tl
5534 {
5535   { \tl_item:Nn \l__spath_tmpc_tl {1} }
5536   { \tl_item:Nn \l__spath_tmpc_tl {2} }
5537   { \tl_item:Nn \l__spath_tmpc_tl {3} }
5538   { \tl_item:Nn \l__spath_tmpc_tl {4} }
5539 }
5540 \ifpgfallowupsidedownatime
5541 \else
5542 \fp_compare:nT { \tl_item:Nn \l__spath_tmpc_tl {4} < 0 }
5543 {
5544   \tl_set:Nx \l__spath_tmpa_tl
5545   {
5546     { \fp_eval:n { - (\tl_item:Nn \l__spath_tmpc_tl {1}) } }
5547     { \fp_eval:n { - (\tl_item:Nn \l__spath_tmpc_tl {2}) } }
5548     { \fp_eval:n { - (\tl_item:Nn \l__spath_tmpc_tl {3}) } }
5549     { \fp_eval:n { - (\tl_item:Nn \l__spath_tmpc_tl {4}) } }
5550   }
5551 }
5552 \fi
5553 \tl_put_right:Nn \l__spath_tmpa_tl {{\pgfpointorigin}}
5554 \exp_last_unbraced:NV \pgftransformcm \l__spath_tmpa_tl
5555 \fi
5556 }
5557 }
5558 \cs_generate_variant:Nn \spath_set_tikz_data:n {V, v}

```

(End of definition for `\spath_set_tikz_data:n`.)

4 The TikZ interface

```
5559 <@=tikzspath>
```

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

```

5560 \RequirePackage{spath3}
5561 \RequirePackage{expl3}
5562 \ExplSyntaxOn

```



```

5563
5564 \tl_new:N \l__tikzspath_tmpa_tl
5565 \tl_new:N \l__tikzspath_tmpb_tl
5566 \tl_new:N \l__tikzspath_tmpc_tl
5567 \tl_new:N \l__tikzspath_tmpd_tl
5568 \tl_new:N \l__tikzspath_tmpe_tl
5569 \tl_new:N \l__tikzspath_tmpf_tl
5570
5571 \int_new:N \l__tikzspath_tmpa_int
5572 \seq_new:N \l__tikzspath_tmpa_seq
5573 \seq_new:N \l__tikzspath_tmpb_seq
5574 \seq_new:N \l__tikzspath_tmpc_seq
5575 \seq_new:N \l__tikzspath_tmpd_seq
5576
5577 \tl_new:N \l__tikzspath_current_tl
5578 \tl_new:N \l__tikzspath_reverse_tl
5579 \tl_new:N \l__tikzspath_prefix_tl
5580 \tl_new:N \l__tikzspath_suffix_tl
5581 \tl_new:N \g__tikzspath_smuggle_tl
5582 \tl_new:N \g__tikzspath_output_tl
5583 \tl_new:N \l__tikzspath_check_tl
5584 \clist_new:N \g__tikzspath_output_clist
5585 \seq_new:N \g__tikzspath_tmpa_seq
5586 \seq_new:N \g__tikzspath_tmpb_seq
5587 \seq_new:N \g__tikzspath_output_seq
5588 \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.

```

5589 \msg_new:nnn { spath3 } { missing soft path }
5590 { Soft~ path~ #1~ doesn't~ exist~ \msg_line_context:}
5591 \msg_new:nnnn { spath3 } { empty soft path }
5592 { Soft~ path~ #1~ is~ empty~ \msg_line_context:}
5593 {If~ it~ was~ defined~ inside~ a~ group,~ try~ using~ "save~ global". }
5594 \msg_new:nnn { spath3 } { load intersections }
5595 { You~ need~ to~ load~ the~ "intersections"~ library~
5596   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.

```

5597 \tl_set:Nn \l__tikzspath_prefix_tl {tikz@intersect@path@name@}
5598 \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.

```

5599 \tl_new:N \g__tikzspath_tikzfinish_tl

```

```

5600 \tl_new:N \l__tikzspath_tikzfinish_outside_tl
5601 \cs_new_protected_nopar:Npn \spath_at_end_of_path:
5602 {
5603   \tl_use:N \g__tikzspath_tikzfinish_tl
5604   \tl_gclear:N \g__tikzspath_tikzfinish_tl
5605 }
5606 \tl_put_right:Nn \tikz@finish {\spath_at_end_of_path:}
5607
5608 \tikzset{
5609   every~ scope/.append~ style={
5610     execute~ at~ begin~ scope={
5611       \tl_set_eq:NN \l__tikzspath_tikzfinish_outside_tl \g__tikzspath_tikzfinish_tl
5612     },
5613     execute~ at~ end~ scope={
5614       \tl_use:N \g__tikzspath_tikzfinish_tl
5615       \tl_gclear:N \g__tikzspath_tikzfinish_tl
5616       \tl_gset_eq:NN \g__tikzspath_tikzfinish_tl \l__tikzspath_tikzfinish_outside_tl
5617     },
5618   },
5619 }

```

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.

```

5620 \tl_new:N \l__tikzspath_tikzpath_finish_tl
5621
5622 \cs_new_protected_nopar:Npn \__tikzspath_at_end_of_path_construction:
5623 {
5624   \tl_use:N \l__tikzspath_tikzpath_finish_tl
5625   \tl_clear:N \l__tikzspath_tikzpath_finish_tl
5626 }
5627
5628 \tl_put_left:Nn \tikz@finish {\__tikzspath_at_end_of_path_construction:}

```

Code for saving a path

```

5629 \cs_new_protected_nopar:Npn \spath_save_path:Nn #1#2
5630 {
5631   \tl_if_empty:NF \g__tikzspath_tikzfinish_tl
5632   {
5633     \tl_use:N \g__tikzspath_tikzfinish_tl
5634   }
5635   \tl_gput_right:Nn \g__tikzspath_tikzfinish_tl
5636   {
5637     \tl_clear_new:N #1
5638     \tl_set:Nn #1 {#2}
5639   }
5640 }
5641 \cs_generate_variant:Nn \spath_save_path:Nn {cn, NV, cV}
5642
5643 \cs_new_protected_nopar:Npn \spath_gsave_path:Nn #1#2
5644 {
5645   \tl_gput_right:Nn \g__tikzspath_tikzfinish_tl
5646   {
5647     \tl_gclear_new:N #1
5648     \tl_gset:Nn #1 {#2}

```

```

5649 }
5650 }
5651 \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.

```

5652 \cs_new_protected_nopar:Npn \__tikzspath_process_tikz_point:Nn #1#2
5653 {
5654   \group_begin:
5655   \use:c {tikz@scan@one@point} \use:n #2 \scan_stop:
5656   \tl_gset:Nx \g__tikzspath_output_tl
5657   {
5658     { \dim_use:c {pgf@x} }
5659     { \dim_use:c {pgf@y} }
5660   }
5661   \group_end:
5662   \tl_set_eq:NN #1 \g__tikzspath_output_tl
5663   \tl_gclear:N \g__tikzspath_output_tl
5664 }

```

(End of definition for __tikzspath_process_tikz_point:Nn.)

`__tikzspath_tikzset:n`

Wrapper around `\tikzset` for expansion.

```

5665 \cs_set_eq:NN \__tikzspath_tikzset:n \tikzset
5666 \cs_generate_variant:Nn \__tikzspath_tikzset:n {V, v}

```

(End of definition for __tikzspath_tikzset:n.)

`s:nmmn___tikzspath_check_three_paths:nmmn`

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.

```

5667 \cs_new_protected_nopar:Npn \__tikzspath_check_path:nnn #1#2#3
5668 {
5669   \tl_set:Nn \l__tikzspath_check_tl {#3}
5670   \tl_if_exist:cTF {\__tikzspath_path_name:n {#2}}
5671   {
5672     \tl_if_empty:cTF {\__tikzspath_path_name:n {#2}}
5673     {
5674       \msg_warning:nnn { spath3 } { empty soft path } { #2 }
5675     }
5676     {
5677       \tl_set:Nn \l__tikzspath_check_tl {
5678         #1 {\__tikzspath_path_name:n {#2}}
5679       }
5680     }
5681   }
5682   {
5683     \msg_warning:nnx { spath3 } { missing soft path } { #2 }
5684   }
5685   \tl_use:N \l__tikzspath_check_tl
5686 }
5687 \cs_new_protected_nopar:Npn \__tikzspath_check_two_paths:nmmn #1#2#3#4
5688 {
5689   \__tikzspath_check_path:nnn {
5690     \__tikzspath_check_path:nnn {#1}{#2}{#4}

```

```

5691   }{#3}{#4}
5692 }
5693 \cs_new_protected_nopar:Npn \__tikzspath_check_three_paths:nnnnn #1#2#3#4#5
5694 {
5695   \__tikzspath_check_path:nnn {
5696     \__tikzspath_check_path:nnn {
5697       \__tikzspath_check_path:nnn {#1}{#2}{#5}
5698     }{#3}{#5}
5699   }{#4}{#5}
5700 }
5701 \cs_generate_variant:Nn \__tikzspath_check_path:nnn {nVn}
5702 \cs_generate_variant:Nn \__tikzspath_check_two_paths:nnnn {nnVn}

```

(End of 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.

```

5703 \cs_new_protected_nopar:Npn \__tikzspath_maybe_current_path:nn #1#2
5704 {
5705   \tl_if_eq:nnT {#2} {current}
5706   {
5707     \spath_get_current_path:c {\__tikzspath_path_name:n {#2}}
5708   }
5709   #1 {#2}
5710 }
5711 \cs_new_protected_nopar:Npn \__tikzspath_maybe_current_path_reuse:nnn #1#2#3
5712 {
5713   \bool_set_true:N \l_spath_movetorelevant_bool
5714   \tl_if_eq:nnT {#2} {current}
5715   {
5716     \spath_get_current_path:c {\__tikzspath_path_name:n {#2}}
5717   }
5718   #1 {#2} #3
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_both: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 {#2} {current}
5741 {
5742   \tl_if_empty:cF {\__tikzspath_path_name:n {#2}}
5743   {
5744     \spath_set_current_path:c {\__tikzspath_path_name:n {#2}}
5745     \spath_set_tikz_data:v {\__tikzspath_path_name:n {#2}}
5746   }
5747 }
5748 \tl_if_eq:nnT {#3} {current}
5749 {
5750   \tl_if_empty:cF {\__tikzspath_path_name:n {#3}}
5751   {
5752     \spath_set_current_path:c {\__tikzspath_path_name:n {#3}}
5753     \spath_set_tikz_data:v {\__tikzspath_path_name:n {#3}}
5754   }
5755 }
5756 }
5757 \cs_new_protected_nopar:Npn \__tikzspath_maybe_current_two_paths_reuse_first:nnnn #1#2#3#4
5758 {
5759   \bool_set_true:N \l_spath_movetorelevant_bool
5760   \tl_if_eq:nnT {#2} {current}
5761   {
5762     \spath_get_current_path:c {\__tikzspath_path_name:n {#2}}
5763   }
5764   \tl_if_eq:nnT {#3} {current}
5765   {
5766     \spath_get_current_path:c {\__tikzspath_path_name:n {#3}}
5767   }
5768   #1 {#2} {#3} #4
5769   \tl_if_eq:nnT {#2} {current}
5770   {
5771     \tl_if_empty:cF {\__tikzspath_path_name:n {#2}}
5772     {
5773       \spath_set_current_path:c {\__tikzspath_path_name:n {#2}}
5774       \spath_set_tikz_data:v {\__tikzspath_path_name:n {#2}}
5775     }
5776   }
5777 }
5778 \cs_new_protected_nopar:Npn \__tikzspath_maybe_current_two_paths_reuse_second:nnnn #1#2#3#4
5779 {
5780   \bool_set_true:N \l_spath_movetorelevant_bool
5781   \tl_if_eq:nnT {#2} {current}
5782   {
5783     \spath_get_current_path:c {\__tikzspath_path_name:n {#2}}
5784   }
5785   \tl_if_eq:nnT {#3} {current}
5786   {
5787     \spath_get_current_path:c {\__tikzspath_path_name:n {#3}}
5788   }
5789   #1 {#2} {#3} #4
5790   \tl_if_eq:nnT {#3} {current}
5791   {
5792     \tl_if_empty:cF {\__tikzspath_path_name:n {#3}}

```

```

5793     {
5794       \spath_set_current_path:c {\_tikzspath_path_name:n {#3}}
5795       \spath_set_tikz_data:v {\_tikzspath_path_name:n {#3}}
5796     }
5797   }
5798 }
5799 \cs_generate_variant:Nn \_tikzspath_maybe_current_path:nn {nV}
5800 \cs_generate_variant:Nn \_tikzspath_maybe_current_path_reuse:nnn {nVn}

```

(End of 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.

```

5801 \cs_new_protected_nopar:Npn \_tikzspath_seq_from_foreach:NNn #1#2#3
5802 {
5803   \group_begin:
5804   \seq_gclear:N \g__tikzspath_output_seq
5805
5806   \tl_if_empty:nTF {#3}
5807   {
5808     \int_step_inline:nnnn {1}{1} {#2}
5809     {
5810       \seq_gput_right:Nn \g__tikzspath_output_seq {##1}
5811     }
5812   }
5813   {
5814     \foreach \l__tikzspath_tmpa_tl in {#3}
5815     {
5816       \int_compare:nTF { \l__tikzspath_tmpa_tl > 0 }
5817       {
5818         \seq_gput_right:NV \g__tikzspath_output_seq \l__tikzspath_tmpa_tl
5819       }
5820       {
5821         \seq_gput_right:Nx \g__tikzspath_output_seq
5822         {\int_eval:n {#2 - \l__tikzspath_tmpa_tl}}
5823       }
5824     }
5825     \seq_gsort:Nn \g__tikzspath_output_seq
5826     {
5827       \int_compare:nNnTF {##1} < {##2}
5828       { \sort_return_same: }
5829       { \sort_return_swapped: }
5830     }
5831   }
5832   \group_end:
5833   \seq_set_eq:NN #1 \g__tikzspath_output_seq
5834   \seq_gclear:N \g__tikzspath_output_seq
5835 }
5836 \cs_generate_variant:Nn \_tikzspath_seq_from_foreach:NNn {NVV, NVn}
5837 %

```

(End of definition for _tikzspath_seq_from_foreach:NNn.)

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

```
5838 \cs_new:Npn \__tikzspath_path_name:n #1
5839 {
5840   \tl_use:N \l__tikzspath_prefix_tl
5841   #1
5842   \tl_use:N \l__tikzspath_suffix_tl
5843 }
5844 \cs_generate_variant:Nn \__tikzspath_path_name:n {V}
```

(End of definition for `__tikzspath_path_name:n`.)

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

```
5845 \bool_new:N \l__tikzspath_reverse_bool
5846 \bool_new:N \l__tikzspath_weld_bool
5847 \bool_new:N \l__tikzspath_move_bool
5848 \bool_new:N \l__tikzspath_global_bool
5849 \bool_new:N \l__tikzspath_current_transformation_bool
5850 \tl_new:N \l__tikzspath_joinpath_tl
5851 \tl_new:N \l__tikzspath_transformation_tl
5852
5853 \cs_new_protected_nopar:Npn \__tikzspath_set_bool:Nn #1#2
5854 {
5855   \tl_if_eq:nnTF {#2}{false}
5856   {
5857     \bool_set_false:N #1
5858   }
5859   {
5860     \bool_set_true:N #1
5861   }
5862 }
5863 \tikzset {
5864   spath/join/.is~ family,
5865   spath/join/.cd,
5866   reverse/.code = {
5867     \__tikzspath_set_bool:Nn \l__tikzspath_reverse_bool {#1}
5868   },
5869   reverse/.default = true,
5870   weld/.code = {
5871     \__tikzspath_set_bool:Nn \l__tikzspath_weld_bool {#1}
5872   },
5873   weld/.default = true,
5874   no~ weld/.code = {
5875     \__tikzspath_set_bool:Nn \l__tikzspath_weld_bool {#1}
5876     \bool_set:Nn \l__tikzspath_weld_bool {! \l__tikzspath_weld_bool}
5877   },
5878   no~ weld/.default = true,
5879   move/.code = {
5880     \__tikzspath_set_bool:Nn \l__tikzspath_move_bool {#1}
5881   },
5882   move/.default = true,
5883   no~ move/.code = {
5884     \__tikzspath_set_bool:Nn \l__tikzspath_move_bool {#1}
5885     \bool_set:Nn \l__tikzspath_move_bool {! \l__tikzspath_move_bool}
5886   },
```

```

5887 no~ move/.default = true,
5888 global/.code = {
5889   \tikzspath_set_bool:Nn \tikzspath_global_bool {#1}
5890 },
5891 global/.default = true,
5892 use~ current~ transformation/.code={
5893   \tikzspath_set_bool:Nn \tikzspath_current_transformation_bool {#1}
5894 },
5895 use~ current~ transformation/.default = true,
5896 transform/.store-in=\tikzspath_transformation_tl,
5897 .unknown/.code = {
5898   \tl_set_eq:NN \tikzspath_joinpath_tl \pgfkeyscurrentname
5899 }
5900 }

```

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).

```

5901 \cs_set_eq:NN \GetComponentOf \clist_item:Nn

```

4.1 Helper Functions

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

```

5902 \cs_new_protected_nopar:Npn \tikzspath_use_path:n #1
5903 {
5904   \tl_set:Nn \tikzspath_joinpath_tl {#1}
5905   \spath_get_current_path:N \tikzspath_current_tl
5906
5907   \bool_if:NT \tikzspath_reverse_bool
5908   {
5909     \spath_reverse:N \tikzspath_joinpath_tl
5910   }
5911
5912   \bool_if:NT \tikzspath_current_transformation_bool
5913   {
5914     \pgfgettransform \tikzspath_tmpa_tl
5915     \spath_transform:NV
5916     \tikzspath_joinpath_tl
5917     \tikzspath_tmpa_tl
5918   }
5919
5920   \tl_if_empty:NF \tikzspath_transformation_tl
5921   {
5922     \group_begin:
5923     \pgftransformreset
5924     \tikzspath_tikzset:V \tikzspath_transformation_tl
5925     \pgfgettransform \tikzspath_tmpa_tl
5926     \tl_gset:Nn \g_tikzspath_smuggle_tl
5927     {
5928       \spath_transform:Nnnnnn
5929       \tikzspath_joinpath_tl
5930     }

```



```

5931 \tl_gput_right:NV \g__tikzspath_smuggle_tl \l__tikzspath_tmpa_tl
5932 \group_end:
5933 \tl_use:N \g__tikzspath_smuggle_tl
5934 }
5935
5936 \bool_if:NT \l__tikzspath_move_bool
5937 {
5938 \tl_if_empty:NTF \l__tikzspath_current_tl
5939 {
5940 \tl_set:Nn \l__tikzspath_tmpc_tl { {Opt} {Opt} }
5941 }
5942 {
5943 \spath_finalpoint:NV
5944 \l__tikzspath_tmpc_tl
5945 \l__tikzspath_current_tl
5946 }
5947 \spath_translate_to:NV \l__tikzspath_joinpath_tl \l__tikzspath_tmpc_tl
5948 }
5949
5950 \tl_if_empty:NTF \l__tikzspath_current_tl
5951 {
5952 \tl_if_empty:NTF \l__tikzspath_joinpath_tl
5953 {
5954 \tl_set_eq:NN \l__tikzspath_current_tl \c_spath_moveto_tl
5955 \tl_put_right:Nn \l__tikzspath_current_tl {{Opt}}{Opt}}
5956 }
5957 {
5958 \tl_set_eq:NN \l__tikzspath_current_tl \l__tikzspath_joinpath_tl
5959 }
5960 }
5961 {
5962
5963 \tl_clear:N \l__tikzspath_tmpa_tl
5964 \tl_set:Nn \l__tikzspath_tmpa_tl {spath_}
5965
5966 \tl_put_right:Nn \l__tikzspath_tmpa_tl {append}
5967
5968 \bool_if:NT \l__tikzspath_weld_bool
5969 {
5970 \tl_put_right:Nn \l__tikzspath_tmpa_tl {_no_move}
5971 \spath_numberofcomponents:NV \l__tikzspath_tmpa_int \l__tikzspath_joinpath_tl
5972 \int_compare:nT {\l__tikzspath_tmpa_int == 1}
5973 {
5974 \bool_set_false:N \l_spath_movetorelevant_bool
5975 }
5976 }
5977 \tl_put_right:Nn \l__tikzspath_tmpa_tl {:NV}
5978
5979 \use:c {\tl_use:N \l__tikzspath_tmpa_tl }
5980 \l__tikzspath_current_tl
5981 \l__tikzspath_joinpath_tl
5982 }
5983
5984 \spath_set_current_path:N \l__tikzspath_current_tl

```

```

5985 \spath_set_tikz_data:V \l__tikzspath_joinpath_tl
5986 }
5987 \cs_generate_variant:Nn \__tikzspath_use_path:n {V, v}

```

(End of definition for __tikzspath_use_path:n.)

__tikzspath_join_with:nn

```

5988 \cs_new_protected_nopar:Npn \__tikzspath_join_with:Nn #1#2
5989 {
5990 \tl_set:Nn \l__tikzspath_joinpath_tl {#2}
5991
5992 \bool_if:NT \l__tikzspath_reverse_bool
5993 {
5994 \spath_reverse:N \l__tikzspath_joinpath_tl
5995 }
5996
5997 \tl_if_empty:NF \l__tikzspath_transformation_tl
5998 {
5999 \group_begin:
6000 \pgftransformreset
6001 \__tikzspath_tikzset:V \l__tikzspath_transformation_tl
6002 \pgfgettransform \l__tikzspath_tmpa_tl
6003 \tl_gset:Nn \g__tikzspath_smuggle_tl
6004 {
6005 \spath_transform:Nnnnnnn
6006 \l__tikzspath_joinpath_tl
6007 }
6008 \tl_gput_right:NV \g__tikzspath_smuggle_tl \l__tikzspath_tmpa_tl
6009 \group_end:
6010 \tl_use:N \g__tikzspath_smuggle_tl
6011 }
6012
6013 \bool_if:NT \l__tikzspath_move_bool
6014 {
6015 \spath_finalpoint:NV
6016 \l__tikzspath_tmpc_tl
6017 #1
6018 \spath_translate_to:NV \l__tikzspath_joinpath_tl \l__tikzspath_tmpc_tl
6019 }
6020
6021 \tl_clear:N \l__tikzspath_tmpa_tl
6022 \tl_set:Nn \l__tikzspath_tmpa_tl {spath_}
6023
6024 \bool_if:NT \l__tikzspath_global_bool
6025 {
6026 \tl_put_right:Nn \l__tikzspath_tmpa_tl {g}
6027 }
6028
6029 \tl_put_right:Nn \l__tikzspath_tmpa_tl {append}
6030
6031 \bool_if:NT \l__tikzspath_weld_bool
6032 {
6033 \tl_put_right:Nn \l__tikzspath_tmpa_tl {_no_move}
6034 }

```

```

6035 \tl_put_right:Nn \l__tikzspath_tmpa_tl {:NV}
6036
6037 \cs_if_exist:cF {\tl_use:N \l__tikzspath_tmpa_tl}
6038 {
6039   \tl_show:N \l__tikzspath_tmpa_tl
6040 }
6041
6042 \use:c {\tl_use:N \l__tikzspath_tmpa_tl } #1
6043 \l__tikzspath_joinpath_tl
6044 }
6045 \cs_generate_variant:Nn \__tikzspath_join_with:Nn {cv, cn}

```

(End of 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.

```

6046 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with_aux:nnn #1#2#3
6047 {
6048   \group_begin:
6049   \tl_set:Nn \l__tikzspath_tmpc_tl {#1}
6050   \tl_if_empty:nT {#3}
6051   {
6052     \spath_spot_weld_components:N \l__tikzspath_tmpc_tl
6053   }
6054
6055   \spath_numberofcomponents:NV \l__tikzspath_tmpa_int \l__tikzspath_tmpc_tl
6056   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpb_seq \l__tikzspath_tmpa_int {#3}
6057
6058   \spath_components_to_seq:NV \l__tikzspath_tmpa_seq \l__tikzspath_tmpc_tl
6059
6060   \seq_pop_left:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6061   \seq_pop_left:NN \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6062
6063   \seq_map_indexed_inline:Nn \l__tikzspath_tmpa_seq
6064   {
6065     \int_compare:nTF
6066     {
6067       ##1 == \l__tikzspath_tmpb_tl
6068     }
6069     {
6070       \seq_pop_left:NNF \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6071       {
6072         \tl_set:Nn \l__tikzspath_tmpb_tl {-1}
6073       }
6074       \spath_splice_between:Nnn \l__tikzspath_tmpa_tl {#2} {##2}
6075     }
6076     {
6077       \tl_put_right:Nn \l__tikzspath_tmpa_tl {##2}
6078     }
6079   }
6080   \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpa_tl
6081   \group_end:
6082 }
6083 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with:Nnnn #1#2#3#4
6084 {

```

```

6085 \tikzspath_join_components_with_aux:nnn {#2}{#3}{#4}
6086 \tl_set_eq:NN #1 \g__tikzspath_output_tl
6087 \tl_gclear:N \g__tikzspath_output_tl
6088 }
6089 \cs_generate_variant:Nn \tikzspath_join_components_with:Nnnn {NVnn}
6090 \cs_new_protected_nopar:Npn \tikzspath_join_components_with:Nnn #1#2#3
6091 {
6092 \tikzspath_join_components_with:NVnn #1#1{#2}{#3}
6093 }
6094 \cs_generate_variant:Nn \tikzspath_join_components_with:Nnn {cvV}
6095 \cs_new_protected_nopar:Npn \tikzspath_gjoin_components_with:Nnnn #1#2#3#4
6096 {
6097 \tikzspath_join_components_with_aux:nnn {#2}{#3}{#4}
6098 \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6099 \tl_gclear:N \g__tikzspath_output_tl
6100 }
6101 \cs_generate_variant:Nn \tikzspath_gjoin_components_with:Nnnn {NVnn}
6102 \cs_new_protected_nopar:Npn \tikzspath_gjoin_components_with:Nnn #1#2#3
6103 {
6104 \tikzspath_gjoin_components_with:NVnn #1#1{#2}{#3}
6105 }
6106 \cs_generate_variant:Nn \tikzspath_gjoin_components_with:Nnn {cvV}
6107 \cs_new_protected_nopar:Npn \tikzspath_join_components_upright_with_aux:nnn #1#2#3
6108 {
6109 \group_begin:
6110 \tl_set:Nn \l__tikzspath_tmpc_tl {#1}
6111 \tl_if_empty:nT {#3}
6112 {
6113 \spath_spot_weld_components:N \l__tikzspath_tmpc_tl
6114 }
6115
6116 \spath_numberofcomponents:N \l__tikzspath_tmpa_int \l__tikzspath_tmpc_tl
6117 \tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpb_seq \l__tikzspath_tmpa_int {#3}
6118
6119 \spath_components_to_seq:N \l__tikzspath_tmpa_seq \l__tikzspath_tmpc_tl
6120
6121 \seq_pop_left:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6122 \seq_pop_left:NN \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6123
6124 \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6125 \spath_transform:NVnnnnn \l__tikzspath_tmpd_tl \l__tikzspath_tmpc_tl {1}{0}{0}{-
1}{Opt}{Opt}
6126
6127 \seq_map_indexed_inline:Nn \l__tikzspath_tmpa_seq
6128 {
6129 \int_compare:nTF
6130 {
6131 ##1 == \l__tikzspath_tmpb_tl
6132 }
6133 {
6134 \seq_pop_left:NNF \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6135 {
6136 \tl_set:Nn \l__tikzspath_tmpb_tl {-1}
6137 }

```

```

6138
6139 \spath_finalpoint:NV \l__tikzspath_tmpe_tl \l__tikzspath_tmpa_tl
6140 \spath_initialpoint:Nn \l__tikzspath_tmpf_tl {##2}
6141
6142 \dim_compare:nTF
6143 {
6144   \tl_item:Nn \l__tikzspath_tmpe_tl {1}
6145   >
6146   \tl_item:Nn \l__tikzspath_tmpf_tl {1}
6147 }
6148 {
6149   \spath_splice_between:NVn
6150   \l__tikzspath_tmpa_tl
6151   \l__tikzspath_tmpd_tl
6152   {##2}
6153 }
6154 {
6155   \spath_splice_between:NVn
6156   \l__tikzspath_tmpa_tl
6157   \l__tikzspath_tmpc_tl
6158   {##2}
6159 }
6160 }
6161 {
6162   \tl_put_right:Nn \l__tikzspath_tmpa_tl {##2}
6163 }
6164 }
6165 \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpa_tl
6166 \group_end:
6167 }
6168 \cs_new_protected_nopar:Npn \__tikzspath_join_components_upright_with:Nnnn #1#2#3#4
6169 {
6170   \__tikzspath_join_components_upright_with_aux:nnn {#2}{#3}{#4}
6171   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6172   \tl_gclear:N \g__tikzspath_output_tl
6173 }
6174 \cs_generate_variant:Nn \__tikzspath_join_components_upright_with:Nnnn {NVnn}
6175 \cs_new_protected_nopar:Npn \__tikzspath_join_components_upright_with:Nnn #1#2#3
6176 {
6177   \__tikzspath_join_components_upright_with:NVnn #1#1{#2}{#3}
6178 }
6179 \cs_generate_variant:Nn \__tikzspath_join_components_upright_with:Nnn {cvV}
6180 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_upright_with:Nnnn #1#2#3#4
6181 {
6182   \__tikzspath_join_components_upright_with_aux:nnn {#2}{#3}{#4}
6183   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6184   \tl_gclear:N \g__tikzspath_output_tl
6185 }
6186 \cs_generate_variant:Nn \__tikzspath_gjoin_components_upright_with:Nnnn {NVnn}
6187 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_upright_with:Nnn #1#2#3
6188 {
6189   \__tikzspath_gjoin_components_upright_with:NVnn #1#1{#2}{#3}
6190 }
6191 \cs_generate_variant:Nn \__tikzspath_gjoin_components_upright_with:Nnn {cvV}

```

(End of 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.

```

6192 \cs_new_protected_nopar:Npn \_tikzspath_get_components_aux:n #1
6193 {
6194   \clist_gclear_new:N \g__tikzspath_output_clist
6195   \spath_components_to_seq:Nn \l__tikzspath_tmpa_seq {#1}
6196
6197   \seq_map_inline:Nn \l__tikzspath_tmpa_seq
6198   {
6199     \spath_anonymous:N \l__tikzspath_tmpa_tl
6200     \tl_new:c {\_tikzspath_path_name:V \l__tikzspath_tmpa_tl}
6201     \tl_set:cn {\_tikzspath_path_name:V \l__tikzspath_tmpa_tl} {##1}
6202     \clist_gput_right:NV \g__tikzspath_output_clist \l__tikzspath_tmpa_tl
6203   }
6204 }
6205 \cs_new_protected_nopar:Npn \_tikzspath_get_components:Nn #1#2
6206 {
6207   \clist_clear_new:N #1
6208   \_tikzspath_get_components_aux:n {#2}
6209   \clist_set_eq:NN #1 \g__tikzspath_output_clist
6210   \clist_gclear:N \g__tikzspath_output_clist
6211 }
6212 \cs_generate_variant:Nn \_tikzspath_get_components:Nn {NV, Nv}
6213 \cs_new_protected_nopar:Npn \_tikzspath_gget_components:Nn #1#2
6214 {
6215   \clist_gclear_new:N #1
6216   \_tikzspath_get_components_aux:n {#2}
6217   \clist_gset_eq:NN #1 \g__tikzspath_output_clist
6218   \clist_gclear:N \g__tikzspath_output_clist
6219 }
6220 \cs_generate_variant:Nn \_tikzspath_gget_components:Nn {NV, Nv}

```

(End of definition for `_tikzspath_get_components:Nn`.)

`_tikzspath_render_components:n`

```

6221 \cs_new_protected_nopar:Npn \_tikzspath_render_components:nn #1#2
6222 {
6223   \group_begin:
6224   \spath_components_to_seq:Nn \l__tikzspath_tmpa_seq {#2}
6225   \seq_map_indexed_inline:Nn \l__tikzspath_tmpa_seq
6226   {
6227     \spath_tikz_path:nn
6228     {
6229       every~ spath~ component/.try,
6230       spath ~component~ ##1/.try,
6231       spath ~component/.try={##1},
6232       every~ #1~ component/.try,
6233       #1 ~component~ ##1/.try,
6234       #1 ~component/.try={##1},
6235     }
6236     {
6237       ##2

```

```

6238     }
6239   }
6240   \group_end:
6241 }
6242 \cs_generate_variant:Nn \__tikzspath_render_components:nn {nv}

```

(End of definition for __tikzspath_render_components:n.)

__tikzspath_insert_gaps_after_components:nn

```

6243 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_components_aux:nnn #1#2#3
6244 {
6245   \group_begin:
6246   \spath_numberofcomponents:Nn \l__tikzspath_tmpa_int {#1}
6247   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_int {#3}
6248
6249   \tl_if_empty:nT {#3}
6250   {
6251     \seq_pop_right:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6252   }
6253
6254   \seq_clear:N \l__tikzspath_tmpb_seq
6255   \seq_map_inline:Nn \l__tikzspath_tmpa_seq {
6256     \seq_put_right:Nx
6257     \l__tikzspath_tmpb_seq
6258     {\int_eval:n
6259      {
6260        \int_mod:nn { #1 }{\l__tikzspath_tmpa_int } + 1
6261      }
6262     }
6263   }
6264
6265   \spath_components_to_seq:Nn \l__tikzspath_tmpc_seq {#1}
6266
6267   \seq_clear:N \l__tikzspath_tmpd_seq
6268   \seq_map_indexed_inline:Nn \l__tikzspath_tmpc_seq
6269   {
6270     \tl_set:Nn \l__tikzspath_tmpa_tl {##2}
6271     \seq_if_in:NnT \l__tikzspath_tmpa_seq {##1}
6272     {
6273       \spath_shorten_at_end:Nn \l__tikzspath_tmpa_tl {(#2)/2}
6274     }
6275     \seq_if_in:NnT \l__tikzspath_tmpb_seq {##1}
6276     {
6277       \spath_shorten_at_start:Nn \l__tikzspath_tmpa_tl {(#2)/2}
6278     }
6279     \seq_put_right:NV \l__tikzspath_tmpd_seq \l__tikzspath_tmpa_tl
6280   }
6281   \tl_gset:Nx \g__tikzspath_output_tl {\seq_use:Nn \l__tikzspath_tmpd_seq {} }
6282   \group_end:
6283 }
6284 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_components:Nnnn #1#2#3#4
6285 {
6286   \__tikzspath_insert_gaps_after_components_aux:nnn {#2}{#3}{#4}
6287   \tl_set_eq:NN #1 \g__tikzspath_output_tl

```

```

6288 \tl_gclear:N \g__tikzspath_output_tl
6289 }
6290 \cs_generate_variant:Nn \__tikzspath_insert_gaps_after_components:Nnnn {NVnn}
6291 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_components:Nnn #1#2#3
6292 {
6293   \__tikzspath_insert_gaps_after_components:NVnn #1#1{#2}{#3}
6294 }
6295 \cs_generate_variant:Nn \__tikzspath_insert_gaps_after_components:Nnn {cnn, cVV}
6296 \cs_new_protected_nopar:Npn \__tikzspath_ginsert_gaps_after_components:Nnnn #1#2#3#4
6297 {
6298   \__tikzspath_insert_gaps_after_components_aux:nnn {#2}{#3}{#4}
6299   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6300   \tl_gclear:N \g__tikzspath_output_tl
6301 }
6302 \cs_generate_variant:Nn \__tikzspath_ginsert_gaps_after_components:Nnnn {NVnn}
6303 \cs_new_protected_nopar:Npn \__tikzspath_ginsert_gaps_after_components:Nnn #1#2#3
6304 {
6305   \__tikzspath_ginsert_gaps_after_components:NVnn #1#1{#2}{#3}
6306 }
6307 \cs_generate_variant:Nn \__tikzspath_ginsert_gaps_after_components:Nnn {cnn, cVV}

```

(End of definition for __tikzspath_insert_gaps_after_components:nn.)

__tikzspath_insert_gaps_after_segments:Nn

```

6308 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_segments_aux:nnn #1#2#3
6309 {
6310   \group_begin:
6311   \spath_reallength:Nn \l__tikzspath_tmpa_int {#1}
6312   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_int {#3}
6313
6314   \tl_if_empty:nT {#3}
6315   {
6316     \seq_pop_right:NN \l__tikzspath_tmpb_seq \l__tikzspath_tmpa_tl
6317   }
6318
6319   \seq_clear:N \l__tikzspath_tmpb_seq
6320   \seq_map_inline:Nn \l__tikzspath_tmpa_seq {
6321     \seq_put_right:Nx
6322     \l__tikzspath_tmpb_seq
6323     {\int_eval:n
6324       {
6325         \int_mod:nn { ##1 }{\l__tikzspath_tmpa_int } + 1
6326       }
6327     }
6328   }
6329
6330   \spath_segments_to_seq:Nn \l__tikzspath_tmpc_seq {#1}
6331
6332   \seq_clear:N \l__tikzspath_tmpd_seq
6333   \seq_map_indexed_inline:Nn \l__tikzspath_tmpc_seq
6334   {
6335     \tl_set:Nn \l__tikzspath_tmpa_tl {##2}
6336     \seq_if_in:NnT \l__tikzspath_tmpa_seq {##1}
6337     {

```



```

6338     \spath_shorten_at_end:Nn \l__tikzspath_tmpa_tl {(#2)/2}
6339   }
6340   \seq_if_in:NnT \l__tikzspath_tmpb_seq {##1}
6341   {
6342     \spath_shorten_at_start:Nn \l__tikzspath_tmpa_tl {(#2)/2}
6343   }
6344   \seq_put_right:NV \l__tikzspath_tmpd_seq \l__tikzspath_tmpa_tl
6345 }
6346 \tl_gset:Nx \g__tikzspath_output_tl {\seq_use:Nn \l__tikzspath_tmpd_seq {}}
6347 \group_end:
6348 }
6349 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_segments:Nnnn #1#2#3#4
6350 {
6351   \__tikzspath_insert_gaps_after_segments_aux:nnn {#2}{#3}{#4}
6352   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6353   \tl_gclear:N \g__tikzspath_output_tl
6354 }
6355 \cs_generate_variant:Nn \__tikzspath_insert_gaps_after_segments:Nnnn {NVnn}
6356 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_segments:Nnn #1#2#3
6357 {
6358   \__tikzspath_insert_gaps_after_segments:NVnn #1#1{#2}{#3}
6359 }
6360 \cs_generate_variant:Nn \__tikzspath_insert_gaps_after_segments:Nnn {cnn, cVV}
6361 \cs_new_protected_nopar:Npn \__tikzspath_ginsert_gaps_after_segments:Nnnn #1#2#3#4
6362 {
6363   \__tikzspath_insert_gaps_after_segments_aux:nnn {#2}{#3}{#4}
6364   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6365   \tl_gclear:N \g__tikzspath_output_tl
6366 }
6367 \cs_generate_variant:Nn \__tikzspath_ginsert_gaps_after_segments:Nnnn {NVnn}
6368 \cs_new_protected_nopar:Npn \__tikzspath_ginsert_gaps_after_segments:Nnn #1#2#3
6369 {
6370   \__tikzspath_ginsert_gaps_after_segments:NVnn #1#1{#2}{#3}
6371 }
6372 \cs_generate_variant:Nn \__tikzspath_ginsert_gaps_after_segments:Nnn {cnn, cVV}

```

(End of definition for __tikzspath_insert_gaps_after_segments:Nn.)

__tikzspath_join_components:Nn

```

6373 \cs_new_protected_nopar:Npn \__tikzspath_join_components_aux:nn #1#2
6374 {
6375   \group_begin:
6376
6377   \tl_set:Nn \l__tikzspath_tmpa_tl {#1}
6378   \spath_numberofcomponents:NV \l__tikzspath_tmpa_int \l__tikzspath_tmpa_tl
6379   \__tikzspath_seq_from_foreach:NVN \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_int {#2}
6380
6381   \seq_reverse:N \l__tikzspath_tmpa_seq
6382
6383   \seq_map_inline:Nn \l__tikzspath_tmpa_seq
6384   {
6385     \spath_join_component:Nn \l__tikzspath_tmpa_tl {##1}
6386   }
6387   \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpa_tl

```

```

6388 \group_end:
6389 }
6390 \cs_new_protected_nopar:Npn \__tikzspath_join_components:Nnn #1#2#3
6391 {
6392   \__tikzspath_join_components_aux:nn {#2}{#3}
6393   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6394   \tl_gclear:N \g__tikzspath_output_tl
6395 }
6396 \cs_generate_variant:Nn \__tikzspath_join_components:Nnn {NVn}
6397 \cs_new_protected_nopar:Npn \__tikzspath_join_components:Nn #1#2
6398 {
6399   \__tikzspath_join_components:NVn #1#1{#2}
6400 }
6401 \cs_generate_variant:Nn \__tikzspath_join_components:Nn {cn}
6402 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components:Nnn #1#2#3
6403 {
6404   \__tikzspath_join_components_aux:nn {#2}{#3}
6405   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6406   \tl_gclear:N \g__tikzspath_output_tl
6407 }
6408 \cs_generate_variant:Nn \__tikzspath_gjoin_components:Nnn {NVn}
6409 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components:Nn #1#2
6410 {
6411   \__tikzspath_gjoin_components:NVn #1#1{#2}
6412 }
6413 \cs_generate_variant:Nn \__tikzspath_gjoin_components:Nn {cn}

```

(End of definition for __tikzspath_join_components:Nn.)

__tikzspath_join_components_with_bezier:Nn

```

6414 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with_bezier_aux:nn #1#2
6415 {
6416   \group_begin:
6417   \tl_set:Nn \l__tikzspath_tmpc_tl {#1}
6418   \tl_if_empty:nT {#2}
6419   {
6420     \spath_spot_weld_components:N \l__tikzspath_tmpc_tl
6421   }
6422
6423   \spath_numberofcomponents:N \l__tikzspath_tmpa_int \l__tikzspath_tmpc_tl
6424   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpb_seq \l__tikzspath_tmpa_int {#2}
6425
6426   \spath_components_to_seq:N \l__tikzspath_tmpa_seq \l__tikzspath_tmpc_tl
6427
6428   \seq_pop_left:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6429   \seq_pop_left:NN \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6430
6431   \seq_map_indexed_inline:Nn \l__tikzspath_tmpa_seq
6432   {
6433     \int_compare:nTF
6434     {
6435       ##1 == \l__tikzspath_tmpb_tl
6436     }
6437     {

```

```

6438     \seq_pop_left:NNF \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6439     {
6440       \tl_set:Nn \l__tikzspath_tmpb_tl {-1}
6441     }
6442     \spath_curve_between:Nn \l__tikzspath_tmpa_tl {##2}
6443   }
6444   {
6445     \tl_put_right:Nn \l__tikzspath_tmpa_tl {##2}
6446   }
6447 }
6448 \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpa_tl
6449 \group_end:
6450 }
6451 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with_bezier:Nnn #1#2#3
6452 {
6453   \__tikzspath_join_components_with_bezier_aux:nn {#2}{#3}
6454   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6455   \tl_gclear:N \g__tikzspath_output_tl
6456 }
6457 \cs_generate_variant:Nn \__tikzspath_join_components_with_bezier:Nnn {NVn}
6458 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with_bezier:Nn #1#2
6459 {
6460   \__tikzspath_join_components_with_bezier:NVn #1#1{#2}
6461 }
6462 \cs_generate_variant:Nn \__tikzspath_join_components_with_bezier:Nn {cV}
6463 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_with_bezier:Nnn #1#2#3
6464 {
6465   \__tikzspath_join_components_with_bezier_aux:nn {#2}{#3}
6466   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6467   \tl_gclear:N \g__tikzspath_output_tl
6468 }
6469 \cs_generate_variant:Nn \__tikzspath_gjoin_components_with_bezier:Nnn {NVn}
6470 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_with_bezier:Nn #1#2
6471 {
6472   \__tikzspath_gjoin_components_with_bezier:NVn #1#1{#2}
6473 }
6474 \cs_generate_variant:Nn \__tikzspath_gjoin_components_with_bezier:Nn {cV}

```

(End of definition for __tikzspath_join_components_with_bezier:Nn.)

__tikzspath_remove_components:nn

```

6475 \cs_new_protected_nopar:Npn \__tikzspath_remove_components_aux:nn #1#2
6476 {
6477   \group_begin:
6478
6479   \spath_numberofcomponents:Nn \l__tikzspath_tmpa_int {#1}
6480   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_int {#2}
6481
6482   \spath_components_to_seq:Nn \l__tikzspath_tmpb_seq {#1}
6483
6484   \seq_pop_left:NNF \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6485   {
6486     \tl_clear:N \l__tikzspath_tmpa_tl
6487   }

```

```

6488 \seq_clear:N \l__tikzspath_tmpc_seq
6489 \seq_map_indexed_inline:Nn \l__tikzspath_tmpb_seq
6490 {
6491   \tl_set:Nn \l__tikzspath_tmpb_tl {##1}
6492   \tl_if_eq:NNTF \l__tikzspath_tmpb_tl \l__tikzspath_tmpa_tl
6493   {
6494     \seq_pop_left:NNTF \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6495     {
6496       \tl_clear:N \l__tikzspath_tmpa_tl
6497     }
6498   }
6499   {
6500     \seq_put_right:Nn \l__tikzspath_tmpc_seq {##2}
6501   }
6502 }
6503 }
6504
6505 \tl_gset:Nx \g__tikzspath_output_tl {\seq_use:Nn \l__tikzspath_tmpc_seq {} }
6506 \group_end:
6507 }
6508 \cs_new_protected_nopar:Npn \__tikzspath_remove_components:Nnn #1#2#3
6509 {
6510   \__tikzspath_remove_components_aux:nn {#2}{#3}
6511   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6512   \tl_gclear:N \g__tikzspath_output_tl
6513 }
6514 \cs_generate_variant:Nn \__tikzspath_remove_components:Nnn {NVn}
6515 \cs_new_protected_nopar:Npn \__tikzspath_remove_components:Nn #1#2
6516 {
6517   \__tikzspath_remove_components:NVn #1#1{#2}
6518 }
6519 \cs_generate_variant:Nn \__tikzspath_remove_components:Nn {cn}
6520 \cs_new_protected_nopar:Npn \__tikzspath_gremove_components:Nnn #1#2#3
6521 {
6522   \__tikzspath_remove_components_aux:nn {#2}{#3}
6523   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6524   \tl_gclear:N \g__tikzspath_output_tl
6525 }
6526 \cs_generate_variant:Nn \__tikzspath_gremove_components:Nnn {NVn}
6527 \cs_new_protected_nopar:Npn \__tikzspath_gremove_components:Nn #1#2
6528 {
6529   \__tikzspath_gremove_components:NVn #1#1{#2}
6530 }
6531 \cs_generate_variant:Nn \__tikzspath_gremove_components:Nn {cn}

```

(End of definition for __tikzspath_remove_components:nn.)

```

\__tikzspath_transform_to:nn
  \__tikzspath_transform_upright_to:nn

```

```

6532 \cs_new_protected_nopar:Npn \__tikzspath_transform_to_aux:nn #1#2
6533 {
6534   \group_begin:
6535   \spath_reallength:Nn \l__tikzspath_tmpa_int {#2}
6536
6537   \tl_set:Nx \l__tikzspath_tmpb_tl

```

```

6538 {\fp_to_decimal:n {(#1) * (\l__tikzspath_tmpa_int)}}
6539 \spath_transformation_at:NnV \l__tikzspath_tmpc_tl {#2} \l__tikzspath_tmpb_tl
6540 \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpc_tl
6541 \group_end:
6542 }
6543 \cs_new_protected_nopar:Npn \__tikzspath_transform_to:nn #1#2
6544 {
6545   \__tikzspath_transform_to_aux:nn {#1}{#2}
6546   \exp_last_unbraced:NV \pgfsettransformentries \g__tikzspath_output_tl
6547   \tl_gclear:N \g__tikzspath_output_tl
6548 }
6549 \cs_generate_variant:Nn \__tikzspath_transform_to:nn {nv}
6550 \cs_new_protected_nopar:Npn \__tikzspath_transform_upright_to:nn #1#2
6551 {
6552   \__tikzspath_transform_to_aux:nn {#1}{#2}
6553   \fp_compare:nT { \tl_item:Nn \g__tikzspath_output_tl {4} < 0 }
6554   {
6555     \tl_gset:Nx \g__tikzspath_output_tl
6556     {
6557       { \fp_eval:n { - (\tl_item:Nn \g__tikzspath_output_tl {1}) } }
6558       { \fp_eval:n { - (\tl_item:Nn \g__tikzspath_output_tl {2}) } }
6559       { \fp_eval:n { - (\tl_item:Nn \g__tikzspath_output_tl {3}) } }
6560       { \fp_eval:n { - (\tl_item:Nn \g__tikzspath_output_tl {4}) } }
6561       { \tl_item:Nn \g__tikzspath_output_tl {5} }
6562       { \tl_item:Nn \g__tikzspath_output_tl {6} }
6563     }
6564   }
6565   \exp_last_unbraced:NV \pgfsettransformentries \g__tikzspath_output_tl
6566   \tl_gclear:N \g__tikzspath_output_tl
6567 }
6568 \cs_generate_variant:Nn \__tikzspath_transform_upright_to:nn {nv}

```

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

4.2 Keys

Now we define all of our keys.

```

6569 \tikzset{
    We're in the spath key family.
6570   spath/.is~family,
6571   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.

```

6572   set~ prefix/.store~ in=\l__tikzspath_prefix_tl,
6573   prefix/.is~choice,
6574   prefix/default/.style={
6575     /tikz/spath/set~ prefix=tikz@intersect@path@name@
6576   },
6577   set~ suffix/.store~ in=\l__tikzspath_suffix_tl,
6578   suffix/.is~choice,
6579   suffix/default/.style={
6580     /tikz/spath/set~ suffix={ }

```

```

6581 },
6582 set~ name/.style={
6583   /tikz/spath/prefix=#1,
6584   /tikz/spath/suffix=#1
6585 },

```

Hook in to the end of the path construction

```

6586 at~ end~ path~ construction/.code={
6587   \tl_put_right:Nn \l__tikzspath_tikzpath_finish_tl {#1}
6588 },

```

Keys for saving and cloning a soft path.

```

6589 save/.code={
6590   \tikz@addmode{
6591     \spath_get_current_path:N \l__tikzspath_tmpa_tl
6592     \spath_bake_round:NV \l__tikzspath_tmpa_tl \l__tikzspath_tmpa_tl
6593     \spath_bake_shorten:NV \l__tikzspath_tmpa_tl \l__tikzspath_tmpa_tl
6594     \spath_save_path:cV {\__tikzspath_path_name:n {#1}}
6595     \l__tikzspath_tmpa_tl
6596   }
6597 },
6598 save~ global/.code={
6599   \tikz@addmode{
6600     \spath_get_current_path:N \l__tikzspath_tmpa_tl
6601     \spath_bake_round:NV \l__tikzspath_tmpa_tl \l__tikzspath_tmpa_tl
6602     \spath_bake_shorten:NV \l__tikzspath_tmpa_tl \l__tikzspath_tmpa_tl
6603     \spath_gsave_path:cV {\__tikzspath_path_name:n {#1}}
6604     \l__tikzspath_tmpa_tl
6605   }
6606 },
6607 clone/.code~ 2~ args={
6608   \__tikzspath_maybe_current_path:nn
6609   {
6610     \__tikzspath_check_path:nnn {
6611       \tl_clear_new:c {\__tikzspath_path_name:n {#1}}
6612       \tl_set_eq:cc {\__tikzspath_path_name:n {#1}}
6613     }
6614   }
6615   {#2}{}
6616 },
6617 clone~ global/.code~ 2~ args={
6618   \__tikzspath_maybe_current_path:nn
6619   {
6620     \__tikzspath_check_path:nnn {
6621       \tl_gclear_new:c {\__tikzspath_path_name:n {#1}}
6622       \tl_gset_eq:cc {\__tikzspath_path_name:n {#1}}
6623     }
6624   }
6625   {#2}{}
6626 },

```

Saves a soft path to the aux file.

```

6627 save~ to~ aux/.code={
6628   \__tikzspath_maybe_current_path:nn

```

```

6629     {
6630         \__tikzspath_check_path:nmn {
6631             \spath_save_to_aux:c
6632         }
6633     }
6634     {#1}
6635     {}
6636 },

```

Exports the path as an SVG file.

```

6637 export~ to~ svg/.code={
6638     \__tikzspath_maybe_current_path:nm
6639     {
6640         \__tikzspath_check_path:nmn {
6641             \spath_export_to_svg:nv {#1}
6642         }
6643     }
6644     {#1}
6645     {}
6646 },

```

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).

```

6647 use/.code={
6648     \bool_set_false:N \l__tikzspath_reverse_bool
6649     \bool_set_false:N \l__tikzspath_weld_bool
6650     \bool_set_false:N \l__tikzspath_move_bool
6651     \bool_set_false:N \l__tikzspath_current_transformation_bool
6652     \bool_set_true:N \l_spath_movetorelevant_bool
6653     \tl_clear:N \l__tikzspath_joinpath_tl
6654     \tl_clear:N \l__tikzspath_transformation_tl
6655     \tikzset{
6656         spath/join/.cd,
6657         #1
6658     }
6659
6660     \__tikzspath_check_path:nVn
6661     {
6662         \__tikzspath_use_path:v
6663     } \l__tikzspath_joinpath_tl {}
6664
6665 },

```

Some aliases for the above.

```

6666 restore/.style={/tikz/spath/use={#1}},
6667 restore~ reverse/.style={/tikz/spath/use={reverse, #1}},
6668 append/.style={/tikz/spath/use={move, weld, #1}},
6669 append~ no~ move/.style={/tikz/spath/use={weld, #1}},
6670 append~ reverse/.style={/tikz/spath/use={move, weld, reverse, #1}},
6671 append~ reverse~ no~ move/.style={/tikz/spath/use={weld, reverse, #1}},
6672 insert/.style={/tikz/spath/use={#1}},
6673 insert~ reverse/.style={/tikz/spath/use={reverse, #1}},

```

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

```

6674 show~current~path/.code={
6675   \tikz@addmode{
6676     \pgfsyssoftpath@getcurrentpath\l__tikzspath_tmpa_tl
6677     \iow_term:n {---- current~ soft~ path~ ---}
6678     \spath_show:V \l__tikzspath_tmpa_tl
6679   }
6680 },

```

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

```

6681 show/.code={
6682   \__tikzspath_check_path:nnn {
6683     \iow_term:n {---- soft~ path~ #1~ ---}
6684     \spath_show:v
6685   } {#1} {}
6686 },

```

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.

```

6687 join~ with/.code~ 2~ args={
6688   \bool_set_false:N \l__tikzspath_reverse_bool
6689   \bool_set_false:N \l__tikzspath_weld_bool
6690   \bool_set_false:N \l__tikzspath_move_bool
6691   \bool_set_false:N \l__tikzspath_global_bool
6692   \bool_set_false:N \l__tikzspath_current_transformation_bool
6693   \tl_clear:N \l__tikzspath_joinpath_tl
6694   \tl_clear:N \l__tikzspath_transformation_tl
6695   \tikzset{
6696     spath/join/.cd,
6697     #2
6698   }
6699
6700   \__tikzspath_maybe_current_path_reuse:nnn
6701   {
6702     \__tikzspath_check_two_paths:nnVn
6703     {
6704       \__tikzspath_join_with:cv
6705     }
6706   } {#1} { \l__tikzspath_joinpath_tl {} }
6707 },

```

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

```

6708 spot~ weld/.code={
6709   \__tikzspath_maybe_current_path_reuse:nnn
6710   {
6711     \__tikzspath_check_path:nnn
6712     {
6713       \spath_spot_weld_components:c
6714     }
6715   } {#1} { {} }
6716 },
6717 spot~ weld~ globally/.code={

```



```

6718 \tikzspath_maybe_current_path_reuse:nnn
6719 {
6720 \tikzspath_check_path:nnn
6721 {
6722 \spath_spot_gweld_components:c
6723 }
6724 } {#1} { {} }
6725 },

```

Reverses the named path.

```

6726 reverse/.code={
6727 \tikzspath_maybe_current_path_reuse:nnn
6728 {
6729 \tikzspath_check_path:nnn
6730 {
6731 \spath_reverse:c
6732 }
6733 } {#1} { {} }
6734 },
6735 reverse~ globally/.code={
6736 \tikzspath_maybe_current_path_reuse:nnn
6737 {
6738 \tikzspath_check_path:nnn
6739 {
6740 \spath_greverse:c
6741 }
6742 } {#1} { {} }
6743 },

```

Adjust a path to span between two points.

```

6744 span/.code ~n~ args={3}{
6745 \tikzspath_maybe_current_path_reuse:nnn
6746 {
6747 \tikzspath_check_path:nnn
6748 {
6749 \tikzspath_process_tikz_point:Nn \l__tikzspath_tmpa_tl {#2}
6750 \tikzspath_process_tikz_point:Nn \l__tikzspath_tmpb_tl {#3}
6751 \spath_span:cVV
6752 }
6753 } {#1} { {} } \l__tikzspath_tmpa_tl \l__tikzspath_tmpb_tl }
6754 },
6755 span~ global/.code ~n~ args={3}{
6756 \tikzspath_maybe_current_path_reuse:nnn
6757 {
6758 \tikzspath_check_path:nnn
6759 {
6760 \tikzspath_process_tikz_point:Nn \l__tikzspath_tmpa_tl {#2}
6761 \tikzspath_process_tikz_point:Nn \l__tikzspath_tmpb_tl {#3}
6762 \spath_span:cVV
6763 }
6764 } {#1} { {} } \l__tikzspath_tmpa_tl \l__tikzspath_tmpb_tl }
6765 },

```

Defines a to path

```

6766 to/.style={
6767   to~path={
6768     [
6769       spath/span={#1}{(\tikztostart)}{(\tikztotarget)},
6770       spath/use={#1,weld},
6771     ]
6772     \tikztonodes
6773   }
6774 },

```

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

```

6775 splice/.code ~n~ args={3}{
6776   \__tikzspath_maybe_current_path_reuse:nnn
6777   {
6778     \__tikzspath_check_three_paths:nnnnn
6779     {
6780       \spath_splice_between:cvv
6781     }
6782   } {#1} { {#2} {#3} {} }
6783 },
6784 splice~ global/.code ~n~ args={3}{
6785   \__tikzspath_maybe_current_path_reuse:nnn
6786   {
6787     \__tikzspath_check_three_paths:nnnnn
6788     {
6789       \spath_gsplice_between:cvv
6790     }
6791   } {#1} { {#2} {#3} {} }
6792 },

```

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”.

```

6793 join~ components~ with/.code~2~args={
6794   \tl_if_head_is_group:nTF {#2}
6795   {
6796     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
6797     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
6798   }
6799   {
6800     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6801     \tl_clear:N \l__tikzspath_tmpd_tl
6802   }
6803
6804   \__tikzspath_maybe_current_path_reuse:nnn
6805   {
6806     \__tikzspath_check_two_paths:nnVn
6807     {
6808       \__tikzspath_join_components_with:cvV
6809     }

```

```

6810 } {#1} { \l__tikzspath_tmpc_tl {} \l__tikzspath_tmpd_tl }
6811 },
6812 join~ components~ globally~ with/.code~2~args={
6813   \tl_if_head_is_group:nTF {#2}
6814   {
6815     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
6816     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
6817   }
6818   {
6819     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6820     \tl_clear:N \l__tikzspath_tmpd_tl
6821   }
6822
6823   \__tikzspath_maybe_current_path_reuse:nnn
6824   {
6825     \__tikzspath_check_two_paths:nnVn
6826     {
6827       \__tikzspath_gjoin_components_with:cvV
6828     }
6829   } {#1} { \l__tikzspath_tmpc_tl {} \l__tikzspath_tmpd_tl }
6830 },
6831 join~ components~ upright~ with/.code~2~args={
6832   \tl_if_head_is_group:nTF {#2}
6833   {
6834     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
6835     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
6836   }
6837   {
6838     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6839     \tl_clear:N \l__tikzspath_tmpd_tl
6840   }
6841
6842   \__tikzspath_maybe_current_path_reuse:nnn
6843   {
6844     \__tikzspath_check_two_paths:nnVn
6845     {
6846       \__tikzspath_join_components_upright_with:cvV
6847     }
6848   } {#1} { \l__tikzspath_tmpc_tl {} \l__tikzspath_tmpd_tl }
6849 },
6850 join~ components~ globally~ upright~ with/.code~2~args={
6851   \tl_if_head_is_group:nTF {#2}
6852   {
6853     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
6854     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
6855   }
6856   {
6857     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6858     \tl_clear:N \l__tikzspath_tmpd_tl
6859   }
6860
6861   \__tikzspath_maybe_current_path_reuse:nnn
6862   {
6863     \__tikzspath_check_two_paths:nnVn

```

```

6864     {
6865     \__tikzspath_gjoin_components_upright_with:cvV
6866     }
6867   } {#1} { \l__tikzspath_tmpc_tl {} \l__tikzspath_tmpd_tl }
6868 },
6869 join~ components~ with~ bezier/.code={
6870   \tl_if_head_is_group:nTF {#1}
6871   {
6872     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#1} {1} }
6873     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#1} {2} }
6874   }
6875   {
6876     \tl_set:Nn \l__tikzspath_tmpc_tl {#1}
6877     \tl_clear:N \l__tikzspath_tmpd_tl
6878   }
6879
6880   \__tikzspath_maybe_current_path_reuse:nVn
6881   {
6882     \__tikzspath_check_path:nnn
6883     {
6884       \__tikzspath_join_components_with_bezier:cV
6885     }
6886   } \l__tikzspath_tmpc_tl { {} \l__tikzspath_tmpd_tl }
6887 },
6888 join~ components~ globally~ with~ bezier/.code~2~args={
6889   \__tikzspath_maybe_current_path_reuse:nnn
6890   {
6891     \__tikzspath_check_path:nnn
6892     {
6893       \__tikzspath_gjoin_components_with_bezier:cn
6894     }
6895   } {#1} { {} } {#2} }
6896 },

```

Close a path.

```

6897 close/.code={
6898   \__tikzspath_maybe_current_path_reuse:nnn
6899   {
6900     \__tikzspath_check_path:nnn
6901     {
6902       \spath_close:c
6903     }
6904   } {#1} { {} }
6905 },
6906 close~ globally/.code={
6907   \__tikzspath_maybe_current_path_reuse:nnn
6908   {
6909     \__tikzspath_check_path:nnn
6910     {
6911       \spath_gclose:c
6912     }
6913   } {#1} { {} }
6914 },

```

Open a path.

```

6915 open/.code={
6916   \__tikzspath_maybe_current_path_reuse:nnn
6917   {
6918     \__tikzspath_check_path:nmn
6919     {
6920       \spath_open:c
6921     }
6922   } {#1} { {} }
6923 },
6924 open~ globally/.code={
6925   \__tikzspath_maybe_current_path_reuse:nnn
6926   {
6927     \__tikzspath_check_path:nmn
6928     {
6929       \spath_gopen:c
6930     }
6931   } {#1} { {} }
6932 },

```

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

```

6933 adjust~ and~ close/.code={
6934   \__tikzspath_maybe_current_path_reuse:nnn
6935   {
6936     \__tikzspath_check_path:nmn
6937     {
6938       \spath_adjust_close:c
6939     }
6940   } {#1} { {} }
6941 },
6942 adjust~ and~ close~ globally/.code={
6943   \__tikzspath_maybe_current_path_reuse:nnn
6944   {
6945     \__tikzspath_check_path:nmn
6946     {
6947       \spath_adjust_gclose:c
6948     }
6949   } {#1} { {} }
6950 },

```

Close a path with another path.

```

6951 close~ with/.code~ 2~ args={
6952   \__tikzspath_maybe_current_path_reuse:nnn
6953   {
6954     \__tikzspath_check_two_paths:nmmn
6955     {
6956       \spath_close_with:cv
6957     }
6958   } {#1} { {#2} {} }
6959 },
6960 close~ globally~ with/.code~ 2~ args={
6961   \__tikzspath_maybe_current_path_reuse:nnn
6962   {
6963     \__tikzspath_check_two_paths:nmmn
6964     {

```

```

6965     \spath_gclose_with:cv
6966     }
6967   } {#1} { {#2} {} }
6968 },

```

Close a path with a curve.

```

6969 close~ with~ curve/.code={
6970   \__tikzspath_maybe_current_path_reuse:nnn
6971   {
6972     \__tikzspath_check_path:nnn
6973     {
6974       \spath_close_with_curve:c
6975     }
6976   } {#1} { {} }
6977 },
6978 close~ globally~ with~ curve/.code={
6979   \__tikzspath_maybe_current_path_reuse:nnn
6980   {
6981     \__tikzspath_check_path:nnn
6982     {
6983       \spath_gclose_with_curve:c
6984     }
6985   } {#1} { {} }
6986 },

```

These keys shorten the path by a dimension.

```

6987 shorten~ at~ end/.code~ 2~ args={
6988   \__tikzspath_maybe_current_path_reuse:nnn
6989   {
6990     \__tikzspath_check_path:nnn
6991     {
6992       \spath_shorten_at_end:cn
6993     }
6994   } {#1} { {} } {#2} }
6995 },
6996 shorten~ at~ start/.code~ 2~ args={
6997   \__tikzspath_maybe_current_path_reuse:nnn
6998   {
6999     \__tikzspath_check_path:nnn
7000     {
7001       \spath_shorten_at_start:cn
7002     }
7003   } {#1} { {} } {#2} }
7004 },
7005 shorten~ at~ both~ ends/.code~ 2~ args={
7006   \__tikzspath_maybe_current_path_reuse:nnn
7007   {
7008     \__tikzspath_check_path:nnn
7009     {
7010       \spath_shorten_at_both_ends:cn
7011     }
7012   } {#1} { {} } {#2} }
7013 },
7014 shorten~ globally~ at~ end/.code~ 2~ args={

```

```

7015 \tikzspath_maybe_current_path_reuse:nnn
7016 {
7017   \tikzspath_check_path:nnn
7018   {
7019     \spath_gshorten_at_end:cn
7020   }
7021 } {#1} { {} {#2} }
7022 },
7023 shorten~ globally~ at~ start/.code~ 2~ args ={
7024   \tikzspath_maybe_current_path_reuse:nnn
7025   {
7026     \tikzspath_check_path:nnn
7027     {
7028       \spath_gshorten_at_start:cn
7029     }
7030   } {#1} { {} {#2} }
7031 },
7032 shorten~ globally~ at~ both~ ends/.code~ 2~ args={
7033   \tikzspath_maybe_current_path_reuse:nnn
7034   {
7035     \tikzspath_check_path:nnn
7036     {
7037       \spath_gshorten_at_both_ends:cn
7038     }
7039   } {#1} { {} {#2} }
7040 },

```

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

```

7041 split~ at/.code~ 2~ args={
7042   \tikzspath_maybe_current_path_reuse:nnn
7043   {
7044     \tikzspath_check_path:nnn
7045     {
7046       \spath_split_at_normalised:cn
7047     }
7048   } {#1} { {} {#2} }
7049 },
7050 split~ globally~ at/.code~ 2~ args={
7051   \tikzspath_maybe_current_path_reuse:nnn
7052   {
7053     \tikzspath_check_path:nnn
7054     {
7055       \spath_gsplit_at_normalised:cn
7056     }
7057   } {#1} { {} {#2} }
7058 },
7059 split~ at~ into/.code~ n~ args={4}{
7060   \tikzspath_maybe_current_path_reuse:nnn
7061   {
7062     \tikzspath_check_path:nnn
7063     {
7064       \spath_split_at_normalised:ccvn {\tikzspath_path_name:n {#1}}
7065       {\tikzspath_path_name:n {#2}}

```

```

7066     }
7067   } {#3} { {} {#4} }
7068 },
7069 split~ globally~ at~ into/.code~ n~ args={4}{
7070   \__tikzspath_maybe_current_path_reuse:nnn
7071   {
7072     \__tikzspath_check_path:nnn
7073     {
7074       \spath_gsplit_at_normalised:ccvn {\__tikzspath_path_name:n {#1}}
7075       {\__tikzspath_path_name:n {#2}}
7076     }
7077   } {#3} { {} {#4} }
7078 },
7079 split~ at~ keep~ start/.code~ 2~ args={
7080   \__tikzspath_maybe_current_path_reuse:nnn
7081   {
7082     \__tikzspath_check_path:nnn
7083     {
7084       \spath_split_at_normalised_keep_start:cn
7085     }
7086   } {#1} { {} {#2} }
7087 },
7088 split~ globally~ at~ keep~ start/.code~ 2~ args={
7089   \__tikzspath_maybe_current_path_reuse:nnn
7090   {
7091     \__tikzspath_check_path:nnn
7092     {
7093       \spath_gsplit_at_normalised_keep_start:cn
7094     }
7095   } {#1} { {} {#2} }
7096 },
7097 split~ at~ keep~ end/.code~ 2~ args={
7098   \__tikzspath_maybe_current_path_reuse:nnn
7099   {
7100     \__tikzspath_check_path:nnn
7101     {
7102       \spath_split_at_normalised_keep_end:cn
7103     }
7104   } {#1} { {} {#2} }
7105 },
7106 split~ globally~ at~ keep~ end/.code~ 2~ args={
7107   \__tikzspath_maybe_current_path_reuse:nnn
7108   {
7109     \__tikzspath_check_path:nnn
7110     {
7111       \spath_gsplit_at_normalised_keep_end:cn
7112     }
7113   } {#1} { {} {#2} }
7114 },
7115 split~ at~ keep~ middle/.style~ n~ args={3}{
7116   /tikz/spath/split~ at~ keep~ start={#1}{#3},
7117   /tikz/spath/split~ at~ keep~ end={#1}{(#2)/(#3)},
7118 },
7119 split~ globally~ at~ keep~ middle/.style~ n~ args={3}{

```



```

7120 /tikz/spath/split~ globally~ at~ keep~ start={#1}{#3},
7121 /tikz/spath/split~ globally~ at~ keep~ end={#1}{(#2)/(#3)},
7122 },

```

This translates the named path.

```

7123 translate/.code~ n~ args={3}{
7124   \__tikzspath_maybe_current_path_reuse:nnn
7125   {
7126     \__tikzspath_check_path:nnn
7127     {
7128       \spath_translate:cnn
7129     }
7130   } {#1} { {} } {#2}{#3} }
7131 },
7132 translate~ globally/.code~ n~ args={3}{
7133   \__tikzspath_maybe_current_path_reuse:nnn
7134   {
7135     \__tikzspath_check_path:nnn
7136     {
7137       \spath_gtranslate:cnn
7138     }
7139   } {#1} { {} } {#2}{#3} }
7140 },

```

This normalises the named path.

```

7141 normalise/.code={
7142   \__tikzspath_maybe_current_path_reuse:nnn
7143   {
7144     \__tikzspath_check_path:nnn
7145     {
7146       \spath_normalise:c
7147     }
7148   } {#1} { {} } }
7149 },
7150 normalise~ globally/.code={
7151   \__tikzspath_maybe_current_path_reuse:nnn
7152   {
7153     \__tikzspath_check_path:nnn
7154     {
7155       \spath_gnormalise:c
7156     }
7157   } {#1} { {} } }
7158 },

```

Transforms the named path using TikZ transformation specifications.

```

7159 transform/.code~ 2~ args={
7160   \group_begin:
7161   \pgftransformreset
7162   \tikzset{#2}
7163   \pgfgettransform \l__tikzspath_tmpa_tl
7164   \tl_gset_eq:NN \g__tikzspath_smuggle_tl \l__tikzspath_tmpa_tl
7165   \group_end:
7166
7167   \__tikzspath_maybe_current_path_reuse:nnn

```

```

7168     {
7169         \__tikzspath_check_path:nnn
7170         {
7171             \spath_transform:cV
7172         }
7173     } {#1} { {} \g__tikzspath_smuggle_tl }
7174 },
7175 transform~globally/.code~ 2~ args={
7176     \group_begin:
7177     \pgftransformreset
7178     \tikzset{#2}
7179     \pgfgettransform \l__tikzspath_tmpa_tl
7180     \tl_gset_eq:NN \g__tikzspath_smuggle_tl \l__tikzspath_tmpa_tl
7181     \group_end:
7182
7183     \__tikzspath_maybe_current_path_reuse:nnn
7184     {
7185         \__tikzspath_check_path:nnn
7186         {
7187             \spath_gtransform:cV
7188         }
7189     } {#1} { {} \g__tikzspath_smuggle_tl }
7190 },

```

Splits first path where it intersects with the second.

```

7191 split~ at~ intersections~ with/.code~ 2~ args={
7192     \tl_if_exist:cTF
7193     {
7194         tikz@library@intersections@loaded
7195     }
7196     {
7197         \__tikzspath_maybe_current_two_paths_reuse_first:nnnn
7198         {
7199             \__tikzspath_check_two_paths:nnnn
7200             {
7201                 \spath_split_path_at_intersections:cv
7202             }
7203         } {#1} {#2} { {} }
7204     }
7205     {
7206         \msg_warning:nn { spath3 } { load intersections }
7207     }
7208 },
7209 split~ globally~ at~ intersections~ with/.code~ n~ args={2}{
7210     \tl_if_exist:cTF
7211     {
7212         tikz@library@intersections@loaded
7213     }
7214     {
7215         \__tikzspath_maybe_current_two_paths_reuse_first:nnnn
7216         {
7217             \__tikzspath_check_two_paths:nnnn
7218             {
7219                 \spath_gsplit_path_at_intersections:cv

```

```

7220     }
7221     } {#1} {#2} { {} }
7222   }
7223   {
7224     \msg_warning:nn { spath3 } { load intersections }
7225   }
7226 },

```

Splits two paths at their mutual intersections.

```

7227 split~ at~ intersections/.code~ n~ args={2}{
7228   \tl_if_exist:cTF
7229   {
7230     tikz@library@intersections@loaded
7231   }
7232   {
7233     \__tikzspath_maybe_current_two_paths_reuse_both:nmnn
7234     {
7235       \__tikzspath_check_two_paths:nmnn
7236       {
7237         \spath_split_at_intersections:cc
7238       }
7239     } {#1} {#2} { {} }
7240   }
7241   {
7242     \msg_warning:nn { spath3 } { load intersections }
7243   }
7244 },
7245 split~ globally~ at~ intersections/.code~ n~ args={2}{
7246   \tl_if_exist:cTF
7247   {
7248     tikz@library@intersections@loaded
7249   }
7250   {
7251     \__tikzspath_maybe_current_two_paths_reuse_both:nmnn
7252     {
7253       \__tikzspath_check_two_paths:nmnn
7254       {
7255         \spath_gsplit_at_intersections:cc
7256       }
7257     } {#1} {#2} { {} }
7258   }
7259   {
7260     \msg_warning:nn { spath3 } { load intersections }
7261   }
7262 },

```

Splits a path at its self-intersections.

```

7263 split~ at~ self~ intersections/.code={
7264   \tl_if_exist:cTF
7265   {
7266     tikz@library@intersections@loaded
7267   }
7268   {
7269     \__tikzspath_maybe_current_path_reuse:nmnn

```

```

7270     {
7271       \__tikzspath_check_path:nnn
7272       {
7273         \spath_split_at_self_intersections:c
7274       }
7275     } {#1} { {} }
7276   }
7277   {
7278     \msg_warning:nn { spath3 } { load intersections }
7279   }
7280 },
7281 split~ globally~ at~ self~ intersections/.code={
7282   \tl_if_exist:cTF
7283   {
7284     tikz@library@intersections@loaded
7285   }
7286   {
7287     \__tikzspath_maybe_current_path_reuse:nnn
7288     {
7289       \__tikzspath_check_path:nnn
7290       {
7291         \spath_gsplit_at_self_intersections:c
7292       }
7293     } {#1} { {} }
7294   }
7295   {
7296     \msg_warning:nn { spath3 } { load intersections }
7297   }
7298 },

```

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

```

7299 get~ components~ of/.code~ 2~ args={
7300   \__tikzspath_maybe_current_path:nn
7301   {
7302     \__tikzspath_check_path:nnn {
7303       \__tikzspath_get_components:Nv #2
7304     }
7305   }
7306   {#1}
7307   {}
7308 },
7309 get~ components~ of~ globally/.code~ 2~ args={
7310   \__tikzspath_maybe_current_path:nn
7311   {
7312     \__tikzspath_check_path:nnn {
7313       \__tikzspath_gget_components:Nv #2
7314     }
7315   }
7316   {#1}
7317   {}
7318 },

```

Loop through the components of a soft path and render each as a separate TikZ

path so that they can be individually styled.

```

7319 render~ components/.code={
7320   \__tikzspath_maybe_current_path:nn
7321   {
7322     \__tikzspath_check_path:nnn {
7323       \__tikzspath_render_components:nv {#1}
7324     }
7325   }
7326   {#1}
7327   {}
7328 },

```

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.

```

7329 insert~ gaps~ after~ components/.code~ 2~ args={
7330   \tl_if_head_is_group:nTF {#2}
7331   {
7332     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
7333     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
7334   }
7335   {
7336     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
7337     \tl_clear:N \l__tikzspath_tmpd_tl
7338   }
7339
7340   \__tikzspath_maybe_current_path_reuse:nnn
7341   {
7342     \__tikzspath_check_path:nnn
7343     {
7344       \__tikzspath_insert_gaps_after_components:cVV
7345     }
7346   } {#1} { {} } \l__tikzspath_tmpc_tl \l__tikzspath_tmpd_tl }
7347 },
7348 insert~ gaps~ globally~ after~ components/.code~ 2~ args={
7349   \tl_if_head_is_group:nTF {#2}
7350   {
7351     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
7352     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
7353   }
7354   {
7355     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
7356     \tl_clear:N \l__tikzspath_tmpd_tl
7357   }
7358
7359   \__tikzspath_maybe_current_path_reuse:nnn
7360   {
7361     \__tikzspath_check_path:nnn
7362     {
7363       \__tikzspath_ginsert_gaps_after_components:cVV
7364     }
7365   } {#1} { {} } \l__tikzspath_tmpc_tl \l__tikzspath_tmpd_tl }
7366 },

```

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.

```

7367 insert~ gaps~ after~ segments/.code~ 2~ args={
7368   \tl_if_head_is_group:nTF {#2}
7369   {
7370     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
7371     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
7372   }
7373   {
7374     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
7375     \tl_clear:N \l__tikzspath_tmpd_tl
7376   }
7377
7378   \__tikzspath_maybe_current_path_reuse:nnn
7379   {
7380     \__tikzspath_check_path:nnn
7381     {
7382       \__tikzspath_insert_gaps_after_segments:cVV
7383     }
7384   } {#1} { {} } \l__tikzspath_tmpc_tl \l__tikzspath_tmpd_tl }
7385 },
7386 insert~ gaps~ globally~ after~ segments/.code~ 2~ args={
7387   \tl_if_head_is_group:nTF {#2}
7388   {
7389     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
7390     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
7391   }
7392   {
7393     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
7394     \tl_clear:N \l__tikzspath_tmpd_tl
7395   }
7396
7397   \__tikzspath_maybe_current_path_reuse:nnn
7398   {
7399     \__tikzspath_check_path:nnn
7400     {
7401       \__tikzspath_ginsert_gaps_after_segments:cVV
7402     }
7403   } {#1} { {} } \l__tikzspath_tmpc_tl \l__tikzspath_tmpd_tl }
7404 },

```

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

```

7405 join~ components/.code~ 2~ args={
7406   \__tikzspath_maybe_current_path_reuse:nnn
7407   {
7408     \__tikzspath_check_path:nnn
7409     {
7410       \__tikzspath_join_components:cn
7411     }
7412   } {#1} { {} } {#2} }
7413 },
7414 join~ components~ globally/.code~ 2~ args={
7415   \__tikzspath_maybe_current_path_reuse:nnn
7416   {

```

```

7417     \tikzspath_check_path:nnn
7418     {
7419     \tikzspath_gjoin_components:cn
7420     }
7421   } {#1} { {} {#2} }
7422 },

```

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

```

7423 remove~ empty~ components/.code={
7424   \tikzspath_maybe_current_path_reuse:nnn
7425   {
7426     \tikzspath_check_path:nnn
7427     {
7428       \spath_remove_empty_components:c
7429     }
7430   } {#1} { {} }
7431 },
7432 remove~ empty~ components~ globally/.code={
7433   \tikzspath_maybe_current_path_reuse:nnn
7434   {
7435     \tikzspath_check_path:nnn
7436     {
7437       \spath_gremove_empty_components:c
7438     }
7439   } {#1} { {} }
7440 },

```

Replace all line segments by Bézier curves.

```

7441 replace~ lines/.code={
7442   \tikzspath_maybe_current_path_reuse:nnn
7443   {
7444     \tikzspath_check_path:nnn
7445     {
7446       \spath_replace_lines:c
7447     }
7448   } {#1} { {} }
7449 },
7450 replace~ lines~ globally/.code={
7451   \tikzspath_maybe_current_path_reuse:nnn
7452   {
7453     \tikzspath_check_path:nnn
7454     {
7455       \spath_greplace_lines:c
7456     }
7457   } {#1} { {} }
7458 },

```

Remove the specified components.

```

7459 remove~ components/.code~ 2~ args={
7460   \tikzspath_maybe_current_path_reuse:nnn
7461   {
7462     \tikzspath_check_path:nnn
7463     {
7464       \tikzspath_remove_components:cn

```

```

7465     }
7466   } {#1} { {} {#2} }
7467 },
7468 remove~ components~ globally/.code~ 2~ args={
7469   \__tikzspath_maybe_current_path_reuse:nnn
7470   {
7471     \__tikzspath_check_path:nnn
7472     {
7473       \__tikzspath_gremove_components:cn
7474     }
7475   } {#1} { {} {#2} }
7476 },

```

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.

```

7477 draft~ mode/.code={
7478   \__tikzspath_set_bool:Nn \l__tikzspath_draft_bool {#1}
7479 },
7480 maybe~ spot~ weld/.code={
7481   \bool_if:NF \l__tikzspath_draft_bool
7482   {
7483     \__tikzspath_maybe_current_path_reuse:nnn
7484     {
7485       \__tikzspath_check_path:nnn
7486       {
7487         \spath_spot_weld_components:c
7488       }
7489     } {#1} { {} }
7490   }
7491 },
7492 maybe~ spot~ weld~ globally/.code={
7493   \bool_if:NF \l__tikzspath_draft_bool
7494   {
7495     \__tikzspath_maybe_current_path_reuse:nnn
7496     {
7497       \__tikzspath_check_path:nnn
7498       {
7499         \spath_spot_gweld_components:c
7500       }
7501     } {#1} { {} }
7502   }
7503 },

```

Set the transformation to lie along a path.

```

7504 transform~ to/.code~ 2~ args={
7505   \__tikzspath_maybe_current_path:nn
7506   {
7507     \__tikzspath_check_path:nnn {
7508       \__tikzspath_transform_to:nv {#2}
7509     }
7510   }
7511   {#1}
7512   {

```



```

7513     \pgfsettransformentries {1}{0}{0}{1}{0pt}{0pt}
7514   }
7515 },

```

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

```

7516 upright~ transform~ to/.code~ 2~ args={
7517   \__tikzspath_maybe_current_path:nn
7518   {
7519     \__tikzspath_check_path:nmn {
7520       \__tikzspath_transform_upright_to:nv {#2}
7521     }
7522   }
7523   {#1}
7524   {
7525     \pgfsettransformentries {1}{0}{0}{1}{0pt}{0pt}
7526   }
7527 },

```

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

```

7528 knot/.style~ n~ args={3}{
7529   /tikz/spath/split~ at~ self~ intersections=#1,
7530   /tikz/spath/remove~ empty~ components=#1,
7531   /tikz/spath/insert~ gaps~ after~ components={#1}{#2}{#3},
7532   /tikz/spath/maybe~ spot~ weld=#1,
7533   /tikz/spath/render~ components=#1
7534 },
7535 global~ knot/.style~ n~ args={3}{
7536   /tikz/spath/split~ globally~ at~ self~ intersections=#1,
7537   /tikz/spath/remove~ empty~ components~ globally=#1,
7538   /tikz/spath/insert~ gaps~ globally ~after~ components={#1}{#2}{#3},
7539   /tikz/spath/maybe~ spot~ weld~ globally=#1,
7540   /tikz/spath/render~ components=#1
7541 },
7542 arrow~ shortening/.code={
7543   \__tikzspath_set_bool:Nn \l_spath_arrow_shortening_bool {#1}
7544 },

```

For single argument commands which take a path as their argument, set the default to be `current` so that they use the current path.

```

7545 show/.default=current,
7546 spot~ weld/.default=current,
7547 spot~ weld~ globally/.default=current,
7548 reverse/.default=current,
7549 reverse~ globally/.default=current,
7550 close/.default=current,
7551 close~ globally/.default=current,
7552 open/.default=current,
7553 open~ globally/.default=current,
7554 adjust~ and~ close/.default=current,
7555 adjust~ and~ close~ globally/.default=current,
7556 close~ with~ curve/.default=current,
7557 close~ globally~ with~ curve/.default=current,
7558 remove~ empty~ components/.default=current,

```

```

7559 remove~ empty~ components~ globally/.default=current,
7560 replace~ lines/.default=current,
7561 replace~ lines~ globally/.default=current,
7562 maybe~ spot~ weld/.default=current,
7563 maybe~ spot~ weld~ globally/.default=current,
7564 }

```

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

```

7565 \cs_new_protected_nopar:Npn \__tikzspath_get_point_at:nn #1#2
7566 {
7567   \group_begin:
7568   \spath_reallength:Nn \l__tikzspath_tmpa_int {#2}
7569   \tl_set:Nx \l__tikzspath_tmpb_tl
7570   {\fp_to_decimal:n {(#1) * (\l__tikzspath_tmpa_int)}}
7571   \spath_point_at:NnV \l__tikzspath_tmpc_tl {#2} \l__tikzspath_tmpb_tl
7572
7573   \tl_clear:N \l__tikzspath_tmpd_tl
7574   \tl_put_right:Nn \l__tikzspath_tmpd_tl {\pgf@x=}
7575   \tl_put_right:Nx \l__tikzspath_tmpd_tl {\tl_item:Nn \l__tikzspath_tmpc_tl {1}}
7576   \tl_put_right:Nn \l__tikzspath_tmpd_tl {\relax}
7577   \tl_put_right:Nn \l__tikzspath_tmpd_tl {\pgf@y=}
7578   \tl_put_right:Nx \l__tikzspath_tmpd_tl {\tl_item:Nn \l__tikzspath_tmpc_tl {2}}
7579   \tl_put_right:Nn \l__tikzspath_tmpd_tl {\relax}
7580
7581   \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpd_tl
7582   \group_end:
7583 }
7584 \cs_generate_variant:Nn \__tikzspath_get_point_at:nn {VV, Vn, Vv}
7585
7586 \tikzdeclarecoordinatesystem{spath}{%
7587   \group_begin:
7588   \tl_set:Nn \l__tikzspath_tmpa_tl {#1}
7589   \tl_trim_spaces:N \l__tikzspath_tmpa_tl
7590
7591   \seq_set_split:NnV \l__tikzspath_tmpa_seq {~} \l__tikzspath_tmpa_tl
7592   \seq_pop_right:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpb_tl
7593
7594   \tl_set:Nx \l__tikzspath_tmpa_tl { \seq_use:Nn \l__tikzspath_tmpa_seq {~} }
7595
7596   \__tikzspath_maybe_current_path:nV
7597   {
7598     \__tikzspath_check_path:nnn {
7599       \__tikzspath_get_point_at:Vv \l__tikzspath_tmpb_tl
7600     }
7601   }
7602   \l__tikzspath_tmpa_tl
7603   {
7604     \tl_gset_eq:NN \g__tikzspath_output_tl \pgfpointorigin
7605   }
7606
7607   \group_end:
7608   \use:c {pgf@process}{%
7609     \tl_use:N \g__tikzspath_output_tl
7610     \pgftransforminvert

```

```

7611     \use:c {pgf@pos@transform@glob}
7612   }
7613   \tl_gclear:N \g__tikzspath_output_tl
7614 }
7615
7616 \ExplSyntaxOff

```

5 The Calligraphy Package

```

7617 <@@=cal>

```

5.1 Initialisation

```

7618 \RequirePackage{spath3}
7619 \ExplSyntaxOn
7620
7621 \tl_new:N \l__cal_tmpa_tl
7622 \tl_new:N \l__cal_tmpb_tl
7623 \tl_new:N \l__cal_tmp_path_tl
7624 \tl_new:N \l__cal_tmp_rpath_tl
7625 \tl_new:N \l__cal_tmp_rpathb_tl
7626 \tl_new:N \l__cal_tmp_patha_tl
7627
7628 \seq_new:N \l__cal_tmpa_seq
7629
7630 \int_new:N \l__cal_tmpa_int
7631 \int_new:N \l__cal_tmpb_int
7632 \int_new:N \g__cal_path_component_int
7633 \int_new:N \g__cal_label_int
7634
7635 \fp_new:N \l__cal_tmpa_fp
7636 \fp_new:N \l__cal_tmpb_fp
7637 \fp_new:N \l__cal_tmpe_fp
7638 \fp_new:N \l__cal_tmpe_dim
7639 \fp_new:N \l__cal_tmpe_dim
7640
7641 \dim_new:N \l__cal_tmpa_dim
7642 \dim_new:N \l__cal_tmpb_dim
7643 \dim_new:N \l__cal_tmpe_dim
7644 \dim_new:N \l__cal_tmpe_dim
7645 \dim_new:N \l__cal_tmpe_dim
7646 \dim_new:N \l__cal_tmpe_dim
7647 \dim_new:N \l__cal_tmpe_dim
7648 \dim_new:N \l__cal_tmpe_dim
7649
7650 \bool_new:N \l__cal_annotate_bool
7651 \bool_new:N \l__cal_taper_start_bool
7652 \bool_new:N \l__cal_taper_end_bool
7653 \bool_new:N \l__cal_taperable_bool
7654
7655 \dim_new:N \l__cal_taper_width_dim
7656 \dim_new:N \l__cal_line_width_dim
7657
7658 \bool_set_true:N \l__cal_taper_start_bool

```

```

7659 \bool_set_true:N \l__cal_taper_end_bool
7660
7661 \cs_generate_variant:Nn \tl_put_right:Nn {Nv}
7662
7663 \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.

```

7664 \tikzset{
7665   define~pen/.code={
7666     \tikzset{pen~name=#1}
7667     \pgf@relevantforpicturesizefalse
7668     \tikz@addmode{
7669       \pgfsyssoftpath@getcurrentpath\l__cal_tmpa_tl
7670       \spath_components_to_seq:Nv \l__cal_tmpa_seq \l__cal_tmpa_tl
7671       \seq_gclear_new:c {g__cal_pen_\pgfkeysvalueof{/tikz/pen~name}_seq}
7672       \seq_gset_eq:cN
7673       {g__cal_pen_\pgfkeysvalueof{/tikz/pen~name}_seq} \l__cal_tmpa_seq
7674       \pgfusepath{discard}%
7675     }
7676   },
7677   define~pen/.default={default},
7678   use~pen/.code={
7679     \tikzset{pen~name=#1}
7680     \int_gzero:N \g__cal_path_component_int
7681     \cs_set_eq:NN \pgfpathmoveto \cal_moveto:n
7682     \tikz@addmode{
7683       \pgfsyssoftpath@getcurrentpath\l__cal_tmpa_tl
7684       \spath_components_to_seq:Nv \l__cal_tmpa_seq \l__cal_tmpa_tl
7685       \tl_if_exist:cTF {g__cal_pen_\pgfkeysvalueof{/tikz/pen~name}_seq}
7686       {
7687         \cal_path_create:Nc \l__cal_tmpa_seq
7688         {g__cal_pen_\pgfkeysvalueof{/tikz/pen~name}_seq}
7689       }
7690       {
7691         \msg_warning:nnx { calligraphy } { undefined pen }
7692         { \pgfkeysvalueof{/tikz/pen~name} }
7693       }
7694     }
7695   },
7696   use~pen/.default={default},
7697   pen~name/.initial={default},
7698   copperplate/.style={pen~name=copperplate},
7699   pen~colour/.initial={black},
7700   weight/.is~choice,
7701   weight/heavy/.style={
7702     line~width=\pgfkeysvalueof{/tikz/heavy~line~width},
7703     taper~width=\pgfkeysvalueof{/tikz/light~line~width},
7704   },
7705   weight/light/.style={
7706     line~width=\pgfkeysvalueof{/tikz/light~line~width},
7707     taper~width=0pt,
7708   },

```

```

7709 heavy/.style={
7710   weight=heavy
7711 },
7712 light/.style={
7713   weight=light
7714 },
7715 heavy~line~width/.initial=2pt,
7716 light~line~width/.initial=1pt,
7717 taper/.is~choice,
7718 taper/.default=both,
7719 taper/none/.style={
7720   taper~start=false,
7721   taper~end=false,
7722 },
7723 taper/both/.style={
7724   taper~start=true,
7725   taper~end=true,
7726 },
7727 taper/start/.style={
7728   taper~start=true,
7729   taper~end=false,
7730 },
7731 taper/end/.style={
7732   taper~start=false,
7733   taper~end=true,
7734 },
7735 taper~start/.code={
7736   \tl_if_eq:nnTF {#1} {true}
7737   {
7738     \bool_set_true:N \l__cal_taper_start_bool
7739   }
7740   {
7741     \bool_set_false:N \l__cal_taper_start_bool
7742   }
7743 },
7744 taper~start/.default={true},
7745 taper~end/.code={
7746   \tl_if_eq:nnTF {#1} {true}
7747   {
7748     \bool_set_true:N \l__cal_taper_end_bool
7749   }
7750   {
7751     \bool_set_false:N \l__cal_taper_end_bool
7752   }
7753 },
7754 taper~end/.default={true},
7755 taper~width/.code={\dim_set:Nn \l__cal_taper_width_dim {#1}},
7756 nib~style/.code~2~args={
7757   \tl_clear_new:c {l__cal_nib_style_#1}
7758   \tl_set:cn {l__cal_nib_style_#1} {#2}
7759 },
7760 stroke~style/.code~2~args={
7761   \tl_clear_new:c {l__cal_stroke_style_#1}
7762   \tl_set:cn {l__cal_stroke_style_#1} {#2}

```

```

7763 },
7764 this~stroke~style/.code={
7765   \tl_clear_new:c
7766   {l__cal_stroke_inline_style_ \int_use:N \g__cal_path_component_int}
7767   \tl_set:cn
7768   {l__cal_stroke_inline_style_ \int_use:N \g__cal_path_component_int} {#1}
7769 },
7770 annotate/.style={
7771   annotate~if,
7772   annotate~reset,
7773   annotation~style/.update~value={#1},
7774 },
7775 annotate~if/.default={true},
7776 annotate~if/.code={
7777   \tl_if_eq:nnTF {#1} {true}
7778   {
7779     \bool_set_true:N \l__cal_annotate_bool
7780   }
7781   {
7782     \bool_set_false:N \l__cal_annotate_bool
7783   }
7784 },
7785 annotate~reset/.code={
7786   \int_gzero:N \g__cal_label_int
7787 },
7788 annotation~style/.initial={draw,->},
7789 annotation~shift/.initial={{0,lex}},
7790 every~annotation~node/.initial={anchor=south-west},
7791 annotation~node~style/.code~2~args={
7792   \tl_clear_new:c {l__cal_annotation_style_ #1 _tl}
7793   \tl_set:cn {l__cal_annotation_style_ #1 _tl}{#2}
7794 },
7795 tl~use:N/.code={
7796   \exp_args:Nv \pgfkeysalso #1
7797 },
7798 tl~use:c/.code={
7799   \tl_if_exist:cT {#1}
7800   {
7801     \exp_args:Nv \pgfkeysalso {#1}
7802   }
7803 },
7804 /handlers/.update~style/.code={
7805   \tl_if_eq:nnF {#1} {\pgfkeysnovalue}
7806   {
7807     \pgfkeys{\pgfkeyscurrentpath/.code=\pgfkeysalso{#1}}
7808   }
7809 },
7810 /handlers/.update~value/.code={
7811   \tl_if_eq:nnF {#1} {\pgfkeysnovalue}
7812   {
7813     \pgfkeyssetvalue{\pgfkeyscurrentpath}{#1}
7814   }
7815 },
7816 }

```

Some wrappers around the TikZ keys.

```

7817 \NewDocumentCommand \pen { 0{} }
7818 {
7819   \path[define~ pen,every~ calligraphy~ pen/.try,#1]
7820 }
7821
7822 \NewDocumentCommand \definepen { 0{} }
7823 {
7824   \tikz \path[define~ pen,every~ calligraphy~ pen/.try,#1]
7825 }
7826
7827 \NewDocumentCommand \calligraphy { 0{} }
7828 {
7829   \path[use~ pen,every~ calligraphy/.try,#1]
7830 }

```

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

```

7831 \cs_new_protected_nopar:Npn \cal_path_create:NN #1#2
7832 {
7833   \int_zero:N \l__cal_tmpa_int
7834   \seq_map_inline:Nn #1
7835   {
7836     \int_compare:nT {\tl_count:n {##1} > 3}
7837     {
7838
7839       \int_incr:N \l__cal_tmpa_int
7840       \int_zero:N \l__cal_tmpb_int
7841
7842       \tl_set:Nn \l__cal_tmp_path_tl {##1}
7843       \spath_open:N \l__cal_tmp_path_tl
7844       \spath_reverse:NV \l__cal_tmp_rpath_tl \l__cal_tmp_path_tl
7845
7846       \seq_map_inline:Nn #2
7847       {
7848         \int_incr:N \l__cal_tmpb_int
7849         \group_begin:
7850         \pgfsys@beginscope
7851         \cal_apply_style:c {l__cal_stroke_style_ \int_use:N \l__cal_tmpa_int}
7852         \cal_apply_style:c {l__cal_stroke_inline_style_ \int_use:N \l__cal_tmpa_int}
7853         \cal_apply_style:c {l__cal_nib_style_ \int_use:N \l__cal_tmpb_int}
7854
7855         \spath_initialpoint:Nn \l__cal_tmpa_tl {####1}
7856         \tl_set_eq:NN \l__cal_tmp_patha_tl \l__cal_tmp_path_tl
7857         \spath_translate:NV \l__cal_tmp_patha_tl \l__cal_tmpa_tl
7858
7859         \int_compare:nTF {\tl_count:n {####1} == 3}
7860         {
7861           \cal_at_least_three:N \l__cal_tmp_patha_tl
7862           \spath_protocol_path:V \l__cal_tmp_patha_tl
7863

```

```

7864     \tikz@options
7865     \dim_set:Nn \l__cal_line_width_dim {\pgflinewidth}
7866     \cal_maybe_taper:N \l__cal_tmp_patha_tl
7867 }
7868 {
7869     \spath_weld:Nn \l__cal_tmp_patha_tl {####1}
7870     \spath_weld:NV \l__cal_tmp_patha_tl \l__cal_tmp_rpath_tl
7871     \spath_reverse:Nn \l__cal_tmp_rpathb_tl {####1}
7872     \spath_weld:NV \l__cal_tmp_patha_tl \l__cal_tmp_rpathb_tl
7873
7874     \tl_clear:N \l__cal_tmpa_tl
7875     \tl_set:Nn \l__cal_tmpa_tl
7876     {
7877         fill=\pgfkeysvalueof{/tikz/pen-colour},
7878         draw=none
7879     }
7880     \tl_if_exist:cT {l__cal_stroke_style_ \int_use:N \l__cal_tmpa_int}
7881     {
7882         \tl_put_right:Nv \l__cal_tmpa_tl
7883         {l__cal_stroke_style_ \int_use:N \l__cal_tmpa_int}
7884     }
7885     \tl_if_exist:cT {l__cal_stroke_inline_style_ \int_use:N \l__cal_tmpa_int}
7886     {
7887         \tl_put_right:Nn \l__cal_tmpa_tl {,}
7888         \tl_put_right:Nv \l__cal_tmpa_tl
7889         {l__cal_stroke_inline_style_ \int_use:N \l__cal_tmpa_int}
7890     }
7891     \tl_if_exist:cT {l__cal_nib_style_ \int_use:N \l__cal_tmpb_int}
7892     {
7893         \tl_put_right:Nn \l__cal_tmpa_tl {,}
7894         \tl_put_right:Nv \l__cal_tmpa_tl
7895         {l__cal_nib_style_ \int_use:N \l__cal_tmpb_int}
7896     }
7897     \spath_tikz_path:VV \l__cal_tmpa_tl \l__cal_tmp_patha_tl
7898 }
7899 \pgfsys@endscope
7900 \group_end:
7901 }
7902
7903 \bool_if:NT \l__cal_annotate_bool
7904 {
7905     \seq_get_right:NN #2 \l__cal_tmpa_tl
7906     \spath_finalpoint:NV \l__cal_tmpa_tl \l__cal_tmpa_tl
7907     \spath_translate:NV \l__cal_tmp_path_tl \l__cal_tmpa_tl
7908     \tikz@scan@one@point
7909     \pgfutil@firstofone
7910     \pgfkeysvalueof{/tikz/annotation-shift}
7911
7912     \spath_translate:Nnn \l__cal_tmp_path_tl {\pgf@x} {\pgf@y}
7913
7914     \pgfkeysgetvalue{/tikz/annotation-style}{\l__cal_tmpa_tl}
7915     \spath_tikz_path:VV \l__cal_tmpa_tl \l__cal_tmp_path_tl
7916
7917     \spath_finalpoint:NV \l__cal_tmpa_tl \l__cal_tmp_path_tl

```



```

7918
7919 \exp_last_unbraced:NV \pgfqpoint \l__cal_tmpa_tl
7920 \begin{scope}[reset~ cm]
7921 \node[
7922   every~annotation~node/.try,
7923   tl~use:c = {l__cal_annotation_style_ \int_use:N \l__cal_tmpa_int _tl}
7924 ] at (\pgf@x,\pgf@y) {\int_use:N \l__cal_tmpa_int};
7925 \end{scope}
7926 }
7927 }
7928 }
7929 }
7930 \cs_generate_variant:Nn \cal_path_create:NN {Nc}

```

(End of 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.

```

7931 \cs_new_eq:NN \cal_orig_moveto:n \pgfpathmoveto
7932 \cs_new_nopar:Npn \cal_moveto:n #1
7933 {
7934   \int_gincr:N \g__cal_path_component_int
7935   \cal_orig_moveto:n {#1}
7936 }

```

(End of definition for \cal_moveto:n.)

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

```

7937 \cs_new_nopar:Npn \cal_apply_style:N #1
7938 {
7939   \tl_if_exist:NT #1 {
7940     \exp_args:NV \tikzset #1
7941   }
7942 }
7943 \cs_generate_variant:Nn \cal_apply_style:N {c}

```

(End of 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.

```

7944 \cs_new_protected_nopar:Npn \cal_at_least_three:Nn #1#2
7945 {
7946   \spath_reallength:Nn \l__cal_tmpa_int {#2}
7947   \tl_clear:N \l__cal_tmpb_tl
7948   \tl_set:Nn \l__cal_tmpb_tl {#2}
7949   \int_compare:nTF {\l__cal_tmpa_int = 1}
7950   {
7951     \spath_split_at:Nn \l__cal_tmpb_tl {2/3}
7952     \spath_split_at:Nn \l__cal_tmpb_tl {1/2}
7953   }
7954   {
7955     \int_compare:nT {\l__cal_tmpa_int = 2}
7956     {
7957       \spath_split_at:Nn \l__cal_tmpb_tl {1.5}
7958       \spath_split_at:Nn \l__cal_tmpb_tl {.5}

```

```

7959     }
7960   }
7961   \tl_set_eq:NN #1 \l__cal_tmpb_tl
7962 }
7963 \cs_generate_variant:Nn \cal_at_least_three:Nn {NV}
7964 \cs_new_protected_nopar:Npn \cal_at_least_three:N #1
7965 {
7966   \cal_at_least_three:NV #1#1
7967 }
7968 \cs_generate_variant:Nn \cal_at_least_three:N {c}

```

(End of definition for \cal_at_least_three:Nn.)

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

```

7969 \cs_new_protected_nopar:Npn \cal_maybe_taper:N #1
7970 {
7971   \tl_set_eq:NN \l__cal_tmpa_tl #1
7972
7973   \bool_if:NT \l__cal_taper_start_bool
7974   {
7975
7976     \dim_set:Nn \l__cal_tmpa_dim {\tl_item:Nn \l__cal_tmpa_tl {2}}
7977     \dim_set:Nn \l__cal_tmpb_dim {\tl_item:Nn \l__cal_tmpa_tl {3}}
7978     \tl_set:Nx \l__cal_tmpb_tl {\tl_item:Nn \l__cal_tmpa_tl {4}}
7979
7980     \token_case_meaning:NnF \l__cal_tmpb_tl
7981     {
7982       \c_spath_lineto_tl
7983       {
7984
7985         \bool_set_true:N \l__cal_taperable_bool
7986         \dim_set:Nn \l__cal_tmpg_dim {\tl_item:Nn \l__cal_tmpa_tl {5}}
7987         \dim_set:Nn \l__cal_tmph_dim {\tl_item:Nn \l__cal_tmpa_tl {6}}
7988         \dim_set:Nn \l__cal_tmpe_dim {(2\l__cal_tmpa_dim + \l__cal_tmph_dim)/3}
7989         \dim_set:Nn \l__cal_tmpe_dim {(2\l__cal_tmpe_dim + \l__cal_tmph_dim)/3}
7990         \dim_set:Nn \l__cal_tmpe_dim {(\l__cal_tmpe_dim + 2\l__cal_tmpe_dim)/3}
7991         \dim_set:Nn \l__cal_tmpe_dim {(\l__cal_tmpe_dim + 2\l__cal_tmpe_dim)/3}
7992         \prg_replicate:nn {4}
7993         {
7994           \tl_set:Nx \l__cal_tmpa_tl {\tl_tail:N \l__cal_tmpa_tl}
7995         }
7996         \tl_put_left:NV \l__cal_tmpa_tl \c_spath_moveto_tl
7997       }
7998     }
7999   }
8000   \c_spath_curvetoa_tl
8001   {
8002     \bool_set_true:N \l__cal_taperable_bool
8003     \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {5}}
8004     \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {6}}
8005     \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {8}}
8006     \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {9}}
8007     \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {11}}
8008     \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {12}}
8009     \prg_replicate:nn {10}
8010     {

```

```

8009         \tl_set:Nx \l__cal_tmpa_tl {\tl_tail:N \l__cal_tmpa_tl}
8010     }
8011     \tl_put_left:NV \l__cal_tmpa_tl \c_spath_moveto_tl
8012 }
8013 }
8014 {
8015     \bool_set_false:N \l__cal_taperable_bool
8016 }
8017
8018 \bool_if:NT \l__cal_taperable_bool
8019 {
8020     \__cal_taper_aux:
8021 }
8022
8023 }
8024
8025 \bool_if:NT \l__cal_taper_end_bool
8026 {
8027
8028     \dim_set:Nn \l__cal_tmpa_dim {\tl_item:Nn \l__cal_tmpa_tl {-2}}
8029     \dim_set:Nn \l__cal_tmpb_dim {\tl_item:Nn \l__cal_tmpa_tl {-1}}
8030     \tl_set:Nx \l__cal_tmpb_tl {\tl_item:Nn \l__cal_tmpa_tl {-3}}
8031
8032     \token_case_meaning:NnF \l__cal_tmpb_tl
8033     {
8034         \c_spath_lineto_tl
8035     {
8036
8037         \bool_set_true:N \l__cal_taperable_bool
8038         \dim_set:Nn \l__cal_tmpg_dim {\tl_item:Nn \l__cal_tmpa_tl {-5}}
8039         \dim_set:Nn \l__cal_tmph_dim {\tl_item:Nn \l__cal_tmpa_tl {-4}}
8040         \dim_set:Nn \l__cal_tmpe_dim {(2\l__cal_tmpa_dim + \l__cal_tmpg_dim)/3}
8041         \dim_set:Nn \l__cal_tmpe_dim {(2\l__cal_tmpe_dim + \l__cal_tmph_dim)/3}
8042         \dim_set:Nn \l__cal_tmpe_dim {(\l__cal_tmpe_dim + 2\l__cal_tmpe_dim)/3}
8043         \dim_set:Nn \l__cal_tmpe_dim {(\l__cal_tmpe_dim + 2\l__cal_tmpe_dim)/3}
8044         \tl_reverse:N \l__cal_tmpa_tl
8045         \prg_replicate:nn {3}
8046     {
8047         \tl_set:Nx \l__cal_tmpa_tl {\tl_tail:N \l__cal_tmpa_tl}
8048     }
8049         \tl_reverse:N \l__cal_tmpa_tl
8050     }
8051     \c_spath_curveto_tl
8052     {
8053         \bool_set_true:N \l__cal_taperable_bool
8054         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-5}}
8055         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-4}}
8056         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-8}}
8057         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-7}}
8058         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-11}}
8059         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-10}}
8060         \tl_reverse:N \l__cal_tmpa_tl
8061         \prg_replicate:nn {9}
8062     {

```

```

8063         \tl_set:Nx \l__cal_tmpa_tl {\tl_tail:N \l__cal_tmpa_tl}
8064     }
8065     \tl_reverse:N \l__cal_tmpa_tl
8066 }
8067 }
8068 {
8069     \bool_set_false:N \l__cal_taperable_bool
8070 }
8071
8072 \bool_if:NT \l__cal_taperable_bool
8073 {
8074     \__cal_taper_aux:
8075 }
8076
8077 }
8078
8079 \pgfsyssoftpath@setcurrentpath\l__cal_tmpa_tl
8080 \pgfsetstrokecolor{\pgfkeysvalueof{/tikz/pen~colour}}
8081 \pgfusepath{stroke}
8082
8083 }

```

(End of definition for \cal_maybe_taper:N.)

__cal_taper_aux: Auxiliary macro to avoid unnecessary code duplication.

```

8084 \cs_new_protected_nopar:Npn \__cal_taper_aux:
8085 {
8086     \tl_clear:N \l__cal_tmpb_tl
8087     \tl_put_right:NV \l__cal_tmpb_tl \c_spath_moveto_tl
8088
8089     \fp_set:Nn \l__cal_tmpa_fp
8090     {
8091         \l__cal_tmpd_dim - \l__cal_tmpb_dim
8092     }
8093     \fp_set:Nn \l__cal_tmpb_fp
8094     {
8095         \l__cal_tmpa_dim - \l__cal_tmpe_dim
8096     }
8097     \fp_set:Nn \l__cal_tmpe_fp
8098     {
8099          $(\l__cal_tmpa\_fp^2 + \l__cal\_tmpb\_fp^2)^{.5}$ 
8100     }
8101
8102     \fp_set:Nn \l__cal_tmpa_fp
8103     {
8104          $.5 * \l__cal\_taper\_width\_dim$ 
8105         *
8106          $\l__cal\_tmpa\_fp / \l__cal\_tmpe\_fp$ 
8107     }
8108     \fp_set:Nn \l__cal_tmpb_fp
8109     {
8110          $.5 * \l__cal\_taper\_width\_dim$ 
8111         *
8112          $\l__cal\_tmpb\_fp / \l__cal\_tmpe\_fp$ 

```

```

8113 }
8114
8115 \fp_set:Nn \l__cal_tmpe_dim
8116 {
8117   \l__cal_tmpe_dim - \l__cal_tmpe_dim
8118 }
8119 \fp_set:Nn \l__cal_tmpe_dim
8120 {
8121   \l__cal_tmpe_dim - \l__cal_tmpe_dim
8122 }
8123 \fp_set:Nn \l__cal_tmpe_dim
8124 {
8125   (\l__cal_tmpe_dim^2 + \l__cal_tmpe_dim^2)^.5
8126 }
8127
8128 \fp_set:Nn \l__cal_tmpe_dim
8129 {
8130   .5*\l__cal_line_width_dim
8131   *
8132   \l__cal_tmpe_dim / \l__cal_tmpe_dim
8133 }
8134 \fp_set:Nn \l__cal_tmpe_dim
8135 {
8136   .5*\l__cal_line_width_dim
8137   *
8138   \l__cal_tmpe_dim / \l__cal_tmpe_dim
8139 }
8140
8141 \tl_put_right:Nx \l__cal_tmpe_dim
8142 {
8143   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8144   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8145 }
8146
8147 \tl_put_right:NV \l__cal_tmpe_dim \c_spath_curvetoa_tl
8148
8149 \tl_put_right:Nx \l__cal_tmpe_dim
8150 {
8151   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8152   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8153 }
8154
8155 \tl_put_right:NV \l__cal_tmpe_dim \c_spath_curvetob_tl
8156
8157 \tl_put_right:Nx \l__cal_tmpe_dim
8158 {
8159   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8160   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8161 }
8162
8163 \tl_put_right:NV \l__cal_tmpe_dim \c_spath_curveto_tl
8164
8165 \tl_put_right:Nx \l__cal_tmpe_dim
8166 {

```

```

8167     {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8168     {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8169   }
8170
8171   \tl_put_right:NV \l__cal_tmpe_dim \c_spath_curvetoa_tl
8172
8173   \tl_put_right:Nx \l__cal_tmpe_dim
8174   {
8175     {
8176       \dim_eval:n
8177       {
8178         \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim
8179         - \fp_to_dim:n { 1.32 * \l__cal_tmpe_dim }
8180       }
8181     }
8182   }
8183   {
8184     \dim_eval:n
8185     {
8186       \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim
8187       + \fp_to_dim:n { 1.32 * \l__cal_tmpe_dim }
8188     }
8189   }
8190 }
8191 }
8192
8193   \tl_put_right:NV \l__cal_tmpe_dim \c_spath_curvetob_tl
8194
8195   \tl_put_right:Nx \l__cal_tmpe_dim
8196   {
8197     {
8198       \dim_eval:n
8199       {
8200         -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim
8201         - \fp_to_dim:n { 1.32 * \l__cal_tmpe_dim }
8202       }
8203     }
8204   }
8205   {
8206     \dim_eval:n
8207     {
8208       -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim
8209       + \fp_to_dim:n { 1.32 * \l__cal_tmpe_dim }
8210     }
8211   }
8212 }
8213 }
8214
8215   \tl_put_right:NV \l__cal_tmpe_dim \c_spath_curveto_tl
8216
8217   \tl_put_right:Nx \l__cal_tmpe_dim
8218   {
8219     {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}
8220     {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim}}

```

```

8221 }
8222
8223 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curvetoa_tl
8224
8225 \tl_put_right:Nx \l__cal_tmpb_tl
8226 {
8227   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim}}
8228   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim}}
8229 }
8230
8231 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curvetob_tl
8232
8233 \tl_put_right:Nx \l__cal_tmpb_tl
8234 {
8235   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim}}
8236   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim}}
8237 }
8238
8239 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curveto_tl
8240
8241 \tl_put_right:Nx \l__cal_tmpb_tl
8242 {
8243   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim}}
8244   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim}}
8245 }
8246
8247 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curvetoa_tl
8248
8249 \tl_put_right:Nx \l__cal_tmpb_tl
8250 {
8251   {
8252     \dim_eval:n
8253     {
8254       -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim
8255       + \fp_to_dim:n{ 1.32 * \l__cal_tmpe_dim}
8256     }
8257   }
8258   {
8259     \dim_eval:n
8260     {
8261       -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim
8262       - \fp_to_dim:n {1.32* \l__cal_tmpe_dim}
8263     }
8264   }
8265 }
8266
8267 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curvetob_tl
8268
8269 \tl_put_right:Nx \l__cal_tmpb_tl
8270 {
8271   {
8272     \dim_eval:n
8273     {
8274       \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim

```

```

8275     + \fp_to_dim:n {1.32 * \l__cal_tmpb_fp}
8276   }
8277 }
8278 {
8279   \dim_eval:n
8280   {
8281     \fp_to_dim:N \l__cal_tmpb_fp + \l__cal_tmpb_dim
8282     - \fp_to_dim:n {1.32 * \l__cal tmpa_fp}
8283   }
8284 }
8285 }
8286
8287 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curveto_tl
8288
8289 \tl_put_right:Nx \l__cal_tmpb_tl
8290 {
8291   {\dim_eval:n { \fp_to_dim:N \l__cal tmpa_fp + \l__cal tmpa_dim}}
8292   {\dim_eval:n { \fp_to_dim:N \l__cal tmpb_fp + \l__cal tmpb_dim}}
8293 }
8294
8295 \pgfsyssoftpath@setcurrentpath\l__cal_tmpb_tl
8296 \pgfsetfillcolor{\pgfkeysvalueof{/tikz/pen~colour}}
8297 \pgfusepath{fill}
8298 }

```

(End of definition for __cal_taper_aux:.)

Defines a copperplate pen.

```

8299 \tl_set:Nn \l__cal tmpa_tl {\pgfsyssoftpath@movetotoken{Opt}{Opt}}
8300 \spath_components_to_seq:NV \l__cal tmpa_seq \l__cal tmpa_tl
8301 \seq_gclear_new:N \g__cal_pen_copperplate_seq
8302 \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.

```

8303 \DeclareDocumentCommand \CopperplatePath { m }
8304 {
8305   \spath_components_to_seq:NV \l__cal tmpa_seq #1
8306   \cal_path_create:NN \l__cal tmpa_seq \g__cal_pen_copperplate_seq
8307 }

```

(End of definition for \CopperplatePath.)

```
8308 \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.

```

8309 \expandafter\ifx\csname pgfdeclaredcoration\endcsname\relax
8310 \else
8311 \pgfdeclaredcoration{calligraphic brace}{brace}%
8312 {%
8313   \state{brace}[width+=\pgfdecoratedremainingdistance,next state=final]%
8314   {%

```



```

8315 \pgfsyssoftpath@setcurrentpath{\pgfutil@empty}%
8316 \pgfpathmoveto{\pgfpointorigin}%
8317 \pgfpathcurveto%
8318 {%
8319   \pgfqpoint%
8320   {.15\pgfdecorationsegmentamplitude}%
8321   {.3\pgfdecorationsegmentamplitude}%
8322 }%
8323 {%
8324   \pgfqpoint%
8325   {.5\pgfdecorationsegmentamplitude}%
8326   {.5\pgfdecorationsegmentamplitude}%
8327 }%
8328 {%
8329   \pgfqpoint%
8330   {\pgfdecorationsegmentamplitude}%
8331   {.5\pgfdecorationsegmentamplitude}%
8332 }%
8333 {%
8334   \pgftransformxshift%
8335   {+\pgfdecorationsegmentaspect\pgfdecoratedremainingdistance}%
8336   \pgfpathlineto%
8337   {%
8338     \pgfqpoint%
8339     {-\pgfdecorationsegmentamplitude}%
8340     {.5\pgfdecorationsegmentamplitude}%
8341   }%
8342   \pgfpathcurveto
8343   {%
8344     \pgfqpoint%
8345     {- .5\pgfdecorationsegmentamplitude}%
8346     {.5\pgfdecorationsegmentamplitude}%
8347   }%
8348   {%
8349     \pgfqpoint%
8350     {- .15\pgfdecorationsegmentamplitude}%
8351     {.7\pgfdecorationsegmentamplitude}%
8352   }%
8353   {%
8354     \pgfqpoint%
8355     {0\pgfdecorationsegmentamplitude}%
8356     {1\pgfdecorationsegmentamplitude}%
8357   }%
8358   \pgfpathmoveto%
8359   {%
8360     \pgfqpoint%
8361     {0\pgfdecorationsegmentamplitude}%
8362     {1\pgfdecorationsegmentamplitude}%
8363   }%
8364   \pgfpathcurveto%
8365   {%
8366     \pgfqpoint%
8367     {.15\pgfdecorationsegmentamplitude}%
8368     {.7\pgfdecorationsegmentamplitude}%

```

```

8369 }%
8370 {%
8371   \pgfqpoint%
8372   {.5\pgfdecorationsegmentamplitude}%
8373   {.5\pgfdecorationsegmentamplitude}%
8374 }%
8375 {%
8376   \pgfqpoint%
8377   {\pgfdecorationsegmentamplitude}%
8378   {.5\pgfdecorationsegmentamplitude}%
8379 }%
8380 }%
8381 {%
8382   \pgftransformxshift{+\pgfdecoratedremainingdistance}%
8383   \pgfpathlineto%
8384   {%
8385     \pgfqpoint%
8386     {-\pgfdecorationsegmentamplitude}%
8387     {.5\pgfdecorationsegmentamplitude}%
8388   }%
8389   \pgfpathcurveto%
8390   {%
8391     \pgfqpoint%
8392     {-.5\pgfdecorationsegmentamplitude}%
8393     {.5\pgfdecorationsegmentamplitude}%
8394   }%
8395   {%
8396     \pgfqpoint%
8397     {-.15\pgfdecorationsegmentamplitude}%
8398     {.3\pgfdecorationsegmentamplitude}%
8399   }%
8400   {\pgfqpoint{Opt}{Opt}}%
8401 }%
8402 \tikzset{%
8403   taper width=.5\pgflinewidth,%
8404   taper%
8405 }%
8406 \pgfsyssoftpath@getcurrentpath\cal@tmp@path%
8407 \CopperplatePath{\cal@tmp@path}%
8408 }%
8409 \state{final}{}%
8410 }%

```

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

```

8411 \pgfdeclaredecoration{calligraphic straight parenthesis}{brace}
8412 {
8413   \state{brace}[width=+\pgfdecoratedremainingdistance,next state=final]%
8414   {%
8415     \pgfsyssoftpath@setcurrentpath{\pgfutil@empty}%
8416     \pgfpathmoveto{\pgfpointorigin}%
8417     \pgfpathcurveto%
8418     {%
8419       \pgfqpoint%

```

```

8420     {.76604\pgfdecorationsegmentamplitude}%
8421     {.64279\pgfdecorationsegmentamplitude}%
8422   }%
8423   {%
8424     \pgfqpoint%
8425     {2.3333\pgfdecorationsegmentamplitude}%
8426     {\pgfdecorationsegmentamplitude}%
8427   }%
8428   {%
8429     \pgfqpoint%
8430     {3.3333\pgfdecorationsegmentamplitude}%
8431     {\pgfdecorationsegmentamplitude}%
8432   }%
8433   {%
8434     \pgftransformxshift{+\pgfdecoratedremainingdistance}%
8435     \pgfpathlineto%
8436     {%
8437       \pgfqpoint%
8438       {-3.3333\pgfdecorationsegmentamplitude}%
8439       {\pgfdecorationsegmentamplitude}%
8440     }%
8441     \pgfpathcurveto%
8442     {%
8443       \pgfqpoint%
8444       {-2.3333\pgfdecorationsegmentamplitude}%
8445       {\pgfdecorationsegmentamplitude}%
8446     }%
8447     {%
8448       \pgfqpoint%
8449       {-.76604\pgfdecorationsegmentamplitude}%
8450       {.64279\pgfdecorationsegmentamplitude}%
8451     }%
8452     {\pgfqpoint{0pt}{0pt}}%
8453   }%
8454   \tikzset{%
8455     taper width=.5\pgflinewidth,%
8456     taper%
8457   }%
8458   \pgfsyssoftpath@getcurrentpath\cal@tmp@path%
8459   \CopperplatePath{\cal@tmp@path}%
8460 }%
8461 \state{final}{}%
8462 }

```

The third is a curved parenthesis.

```

8463 \pgfdeclaredecoration{calligraphic curved parenthesis}{brace}
8464 {
8465   \state{brace}[width=+\pgfdecoratedremainingdistance,next state=final]%
8466   {%
8467     \pgfsyssoftpath@setcurrentpath{\pgfutil@empty}%
8468     \pgfpathmoveto{\pgfpointorigin}%
8469     \pgf@xa=\pgfdecoratedremainingdistance\relax%
8470     \advance\pgf@xa by -1.5890\pgfdecorationsegmentamplitude\relax%
8471     \edef\cgrphy@xa{\the\pgf@xa}%
8472     \pgfpathcurveto%

```

```

8473   {%
8474     \pgfqpoint%
8475     {1.5890\pgfdecorationsegmentamplitude}%
8476     {1.3333\pgfdecorationsegmentamplitude}%
8477   }%
8478   {\pgfqpoint{\cgrphy@xa}{1.3333\pgfdecorationsegmentamplitude}}%
8479   {\pgfqpoint{\pgfdecoratedremainingdistance}{0pt}}%
8480   \tikzset{%
8481     taper width=.5\pgflinewidth,%
8482     taper%
8483   }%
8484   \pgfsyssoftpath@getcurrentpath\cal@tmp@path%
8485   \CopperplatePath{\cal@tmp@path}%
8486 }%
8487 \state{final}{}%
8488 }

```

End the conditional for if pgfdecoration module is loaded

```
8489 \fi
```

6 Drawing Knots

```
8490 <@=knot>
```

6.1 Initialisation

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

```

8491 \RequirePackage{spath3}
8492 \usetikzlibrary{intersections,spath3}
8493
8494 \ExplSyntaxOn
8495
8496 \tl_new:N \l__knot_tmpa_tl
8497 \tl_new:N \l__knot_tmpb_tl
8498 \tl_new:N \l__knot_tmpc_tl
8499 \tl_new:N \l__knot_tmpd_tl
8500 \tl_new:N \l__knot_tmpe_tl
8501 \tl_new:N \l__knot_redraws_tl
8502 \tl_new:N \l__knot_clip_width_tl
8503 \tl_new:N \l__knot_name_tl
8504 \tl_new:N \l__knot_node_tl
8505 \tl_new:N \l__knot_aux_tl
8506 \tl_new:N \l__knot_auxa_tl
8507 \tl_new:N \l__knot_prefix_tl
8508
8509 \seq_new:N \l__knot_segments_seq
8510
8511 \int_new:N \l__knot_tmpa_int
8512 \int_new:N \l__knot_strands_int
8513 \int_new:N \g__knot_intersections_int
8514 \int_new:N \g__knot_filaments_int
8515 \int_new:N \l__knot_component_start_int
8516
8517 \fp_new:N \l__knot_tmpa_fp

```

```

8518 \fp_new:N \l__knot_tmpb_fp
8519
8520 \dim_new:N \l__knot_tmpa_dim
8521 \dim_new:N \l__knot_tmpb_dim
8522 \dim_new:N \l__knot_tolerance_dim
8523 \dim_new:N \l__knot_redraw_tolerance_dim
8524 \dim_new:N \l__knot_clip_bg_radius_dim
8525 \dim_new:N \l__knot_clip_draw_radius_dim
8526
8527 \bool_new:N \l__knot_draft_bool
8528 \bool_new:N \l__knot_ignore_ends_bool
8529 \bool_new:N \l__knot_self_intersections_bool
8530 \bool_new:N \l__knot_splits_bool
8531 \bool_new:N \l__knot_super_draft_bool
8532
8533 \bool_new:N \l__knot_prepend_prev_bool
8534 \bool_new:N \l__knot_append_next_bool
8535 \bool_new:N \l__knot_skip_bool
8536 \bool_new:N \l__knot_save_bool
8537 \bool_new:N \l__knot_debugging_bool
8538
8539 \seq_new:N \g__knot_nodes_seq
8540
8541 \bool_set_true:N \l__knot_ignore_ends_bool
      Configuration is via TikZ keys and styles.
8542 \tikzset{
8543   spath/prefix/knot/.style={
8544     spath/set~ prefix=knot strand,
8545   },
8546   spath/suffix/knot/.style={
8547     spath/set~ suffix={},
8548   },
8549   knot/.code={
8550     \tl_if_eq:nnTF {#1} {none}
8551     {
8552       \tikz@addmode{\tikz@mode@doublefalse}
8553     }
8554     {
8555       \tikz@addmode{\tikz@mode@doubletrue}
8556       \tl_if_eq:nnTF {\pgfkeysnovalue} {#1}
8557       {
8558         \tikz@addoption{\pgfsetinnerstrokecolor{.}}
8559       }
8560       {
8561         \pgfsetinnerstrokecolor{#1}
8562       }
8563       \tikz@addoption{
8564         \pgfsetstrokecolor{knotbg}
8565       }
8566       \tl_set:Nn \tikz@double@setup{
8567         \pgfsetinnerlinewidth{\pgflinewidth}
8568         \pgfsetlinewidth{\dim_eval:n {\tl_use:N \l__knot_gap_tl \pgflinewidth}}
8569       }
8570     }

```

```

8571 },
8572 knot~ gap/.store~ in=\l__knot_gap_tl,
8573 knot~ gap=3,
8574 knot~ diagram/.is~family,
8575 knot~ diagram/.unknown/.code={
8576   \tl_set_eq:NN \l__knot_tmpa_tl \pgfkeyscurrentname
8577   \pgfkeysalso{
8578     /tikz/\l__knot_tmpa_tl=#1
8579   }
8580 },
8581 background~ colour/.code={%
8582   \colorlet{knotbg}{#1}%
8583 },
8584 background~ color/.code={%
8585   \colorlet{knotbg}{#1}%
8586 },
8587 background~ colour=white,
8588 knot~ diagram,
8589 name/.store~ in=\l__knot_name_tl,
8590 name={knot},
8591 save~ intersections/.is~ choice,
8592 save~ intersections/.default=true,
8593 save~ intersections/true/.code={
8594   \bool_set_true:N \l__knot_save_bool
8595 },
8596 save~ intersections/false/.code={
8597   \bool_set_false:N \l__knot_save_bool
8598 },
8599 every~ strand/.style={draw},
8600 ignore~ endpoint~ intersections/.code={
8601   \tl_if_eq:nnTF {#1} {true}
8602   {
8603     \bool_set_true:N \l__knot_ignore_ends_bool
8604   }
8605   {
8606     \bool_set_false:N \l__knot_ignore_ends_bool
8607   }
8608 },
8609 ignore~ endpoint~ intersections/.default=true,
8610 consider~ self~ intersections/.is~choice,
8611 consider~ self~ intersections/true/.code={
8612   \bool_set_true:N \l__knot_self_intersections_bool
8613   \bool_set_true:N \l__knot_splits_bool
8614 },
8615 consider~ self~ intersections/false/.code={
8616   \bool_set_false:N \l__knot_self_intersections_bool
8617   \bool_set_false:N \l__knot_splits_bool
8618 },
8619 consider~ self~ intersections/no~ splits/.code={
8620   \bool_set_true:N \l__knot_self_intersections_bool
8621   \bool_set_false:N \l__knot_splits_bool
8622 },
8623 consider~ self~ intersections/.default={true},
8624 clip~ radius/.code={

```

```

8625     \dim_set:Nn \l__knot_clip_bg_radius_dim {#1}
8626     \dim_set:Nn \l__knot_clip_draw_radius_dim {#1+2pt}
8627   },
8628   clip~ draw~ radius/.code={
8629     \dim_set:Nn \l__knot_clip_draw_radius_dim {#1}
8630   },
8631   clip~ background~ radius/.code={
8632     \dim_set:Nn \l__knot_clip_bg_radius_dim {#1}
8633   },
8634   clip~ radius=10pt,
8635   end~ tolerance/.code={
8636     \dim_set:Nn \l__knot_tolerance_dim {#1}
8637   },
8638   end~ tolerance=14pt,
8639   clip/.style={
8640     clip
8641   },
8642   background~ clip/.style={
8643     clip
8644   },
8645   clip~ width/.code={
8646     \tl_set:Nn \l__knot_clip_width_tl {#1}
8647   },
8648   clip~ width=3,
8649   flip~ crossing/.code={%
8650     \tl_clear_new:c {l__knot_crossing_#1}
8651     \tl_set:cn {l__knot_crossing_#1} {x}
8652   },
8653   ignore~ crossing/.code={%
8654     \tl_clear_new:c {l__knot_ignore_crossing_#1}
8655     \tl_set:cn {l__knot_ignore_crossing_#1} {x}
8656   },
8657   draft~ mode/.is~ choice,
8658   draft~ mode/off/.code={%
8659     \bool_set_false:N \l__knot_draft_bool
8660     \bool_set_false:N \l__knot_super_draft_bool
8661   },
8662   draft~ mode/crossings/.code={%
8663     \bool_set_true:N \l__knot_draft_bool
8664     \bool_set_false:N \l__knot_super_draft_bool
8665   },
8666   draft~ mode/strands/.code={%
8667     \bool_set_true:N \l__knot_draft_bool
8668     \bool_set_true:N \l__knot_super_draft_bool
8669   },
8670   debug/.is~ choice,
8671   debug/true/.code={
8672     \bool_set_true:N \l__knot_debugging_bool
8673   },
8674   debug/false/.code={
8675     \bool_set_false:N \l__knot_debugging_bool
8676   },
8677   debug/.default=true,
8678   draft/.is~ family,

```

```

8679 draft,
8680 crossing~ label/.style={
8681   overlay,
8682   fill=white,
8683   fill~ opacity=.5,
8684   text~ opacity=1,
8685   text=blue,
8686   pin~ edge={blue,<-}
8687 },
8688 strand~ label/.style={
8689   overlay,
8690   circle,
8691   draw=purple,
8692   fill=white,
8693   fill~ opacity=.5,
8694   text~ opacity=1,
8695   text=purple,
8696   inner~ sep=0pt
8697 },
8698 }

```

`\knot_debug:n` Debugging

```

8699 \cs_new_nopar:Npn \knot_debug:n #1
8700 {
8701   \bool_if:NT \l__knot_debugging_bool
8702   {
8703     \iow_term:n {===Knot~ debug: #1===}
8704   }
8705 }
8706
8707 \cs_new_nopar:Npn \knot_show_strand:n #1
8708 {
8709   \bool_if:NT \l__knot_debugging_bool
8710   {
8711     \iow_term:n {===Knot~ debug: #1===}
8712     \spath_show:v {knot #1}
8713   }
8714 }
8715
8716 \cs_generate_variant:Nn \knot_debug:n {x}

```

(End of definition for \knot_debug:n.)

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

```

8717 \cs_new_nopar:Npn \knot_apply_style:N #1
8718 {
8719   \knot_debug:n {knot~ apply~ style}
8720   \tl_if_exist:NT #1 {
8721     \exp_args:NV \tikzset #1
8722   }
8723 }
8724 \cs_generate_variant:Nn \knot_apply_style:N {c}

```

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


```

8725 \NewDocumentCommand \flipcrossings {m}
8726 {
8727   \tikzset{knot~ diagram/flip~ crossing/.list={#1}}%
8728 }

```

(End of definition for \flipcrossings.)

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

```

8729 \NewDocumentCommand \strand { 0{} }
8730 {
8731   \int_incr:N \l__knot_strands_int
8732   \tl_clear_new:c {l__knot_options_strand \int_use:N \l__knot_strands_int}
8733   \tl_set:cn {l__knot_options_strand \int_use:N \l__knot_strands_int} {#1}
8734   \path[#1,spath/set~ name=knot,spath/save=\int_use:N \l__knot_strands_int]
8735 }

```

(End of definition for \strand.)

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

```

8736 \NewDocumentEnvironment{knot} { 0{} }
8737 {
8738   \knot_initialise:n {#1}
8739 }
8740 {
8741   \knot_render:
8742 }

```

(End of definition for knot.)

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

```

8743 \cs_new_protected_nopar:Npn \knot_initialise:n #1
8744 {
8745   \knot_debug:n {knot~ initialise}
8746   \tikzset{knot~ diagram/.cd,every~ knot~ diagram/.try,#1}
8747   \int_zero:N \l__knot_strands_int
8748   \tl_clear:N \l__knot_redraws_tl
8749   \seq_gclear:N \g__knot_nodes_seq
8750 }

```

(End of definition for \knot_initialise:n.)

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

```

8751 \cs_new_protected_nopar:Npn \knot_render:
8752 {
8753   \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).

```

8754   \pgfscope
8755   \pgftransformreset

```

Set the dimension for deciding when to include neighbouring strands

```

8756   \dim_set:Nn \l__knot_redraw_tolerance_dim {\fp_to_dim:n
8757     {
8758       sqrt(2) * max(\l__knot_clip_bg_radius_dim, \l__knot_clip_draw_radius_dim)
8759     }
8760 }

```

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

```
8761 \int_step_function:nnnN {1} {1} {\l__knot_strands_int} \knot_draw_strand:n
```

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

```
8762 \bool_if:NF \l__knot_super_draft_bool
8763 {
```

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.

```
8764 \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.

```
8765 \bool_if:NTF \l__knot_self_intersections_bool
8766 {
8767   \knot_split_strands:
8768   \int_set_eq:NN \l__knot_tmpa_int \g__knot_filaments_int
8769   \tl_set:Nn \l__knot_prefix_tl {filament}
8770 }
8771 {
8772   \int_set_eq:NN \l__knot_tmpa_int \l__knot_strands_int
8773   \tl_set:Nn \l__knot_prefix_tl {strand}
8774 }
```

Initialise the intersection count.

```
8775 \int_gzero:N \g__knot_intersections_int
```

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

```
8776 \tl_clear:N \l__knot_node_tl
8777 \bool_if:NT \l__knot_draft_bool
8778 {
8779   \tl_set:Nn \l__knot_node_tl {
8780     \exp_not:N \node[coordinate,
8781       pin={
8782         node~ contents={\int_use:N \g__knot_intersections_int},
8783         knot~ diagram/draft/crossing~ label,
8784         knot~ diagram/draft/crossing~
8785         \int_use:N \g__knot_intersections_int \c_space_tl label/.try
8786       ]
8787     }]
8788   }
8789 }
```

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

```
8790 \int_step_variable:nnnNn {1} {1} {\l__knot_tmpa_int - 1} \l__knot_tmpa_tl
8791 {
8792   \int_step_variable:nnnNn
8793   {\tl_use:N \l__knot_tmpa_tl + 1}
8794   {1}
8795   {\l__knot_tmpa_int} \l__knot_tmpb_tl
8796   {
8797     \knot_intersections:VV \l__knot_tmpa_tl \l__knot_tmpb_tl
8798   }
8799 }
```

If any redraws were requested, do them here.

```
8800 \tl_use:N \l__knot_redraws_tl
```

Draw the crossing nodes

```
8801 \seq_use:Nn \g__knot_nodes_seq {}
8802 }
```

Close the scope

```
8803 \endpgfscope
8804 \knot_debug:x {knot~rendered,
8805 ~found~\int_use:N \g__knot_intersections_int \c_space_tl~intersections}
8806 }
```

(End of definition for \knot_render:.)

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

```
8807 \cs_new_protected_nopar:Npn \knot_draw_strand:n #1
8808 {
8809 \knot_debug:n {knot~ draw~ strand~ #1}
8810 \pgfscope
8811 \group_begin:
8812 \spath_bake_round:c {knot strand #1}
8813 \tl_set:Nn \l__knot_tmpa_tl {knot~ diagram/every~ strand/.try,}
8814 \tl_put_right:Nv \l__knot_tmpa_tl {l__knot_options_strand #1}
8815 \tl_put_right:Nn \l__knot_tmpa_tl
8816 {
8817 ,
8818 knot~ diagram/only~ when~ rendering/.try,
8819 only~ when~ rendering/.try,
8820 }
8821 \knot_show_strand:n {strand #1}
8822 \spath_tikz_path:Vv \l__knot_tmpa_tl {knot strand #1}
8823 \group_end:
8824 \endpgfscope
8825 }
8826 \cs_generate_variant:Nn \tl_put_right:Nn {Nv}
```

(End of 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.

```
8827 \cs_new_protected_nopar:Npn \knot_draw_labels:n #1
8828 {
8829 \knot_debug:n {knot~ draw~ labels}
8830 \bool_if:NT \l__knot_draft_bool
8831 {
8832 \spath_finalpoint:Nv \l__knot_tmpb_tl {knot strand #1}
8833 \dim_set:Nn \l__knot_tmpa_dim {\tl_item:Nn \l__knot_tmpb_tl {1}}
8834 \dim_set:Nn \l__knot_tmpb_dim {\tl_item:Nn \l__knot_tmpb_tl {2}}
8835 \node[
8836 knot~ diagram/draft/strand-label
8837 ] at (\l__knot_tmpa_dim,\l__knot_tmpb_dim) {#1};
8838 \spath_initialpoint:Nv \l__knot_tmpb_tl {knot strand #1}
8839 \dim_set:Nn \l__knot_tmpa_dim {\tl_item:Nn \l__knot_tmpb_tl {1}}
8840 \dim_set:Nn \l__knot_tmpb_dim {\tl_item:Nn \l__knot_tmpb_tl {2}}
```

```

8841     \node[
8842       knot~ diagram/draft/strand~label
8843     ] at (\l__knot_tmpa_dim,\l__knot_tmpb_dim) {#1};
8844   }
8845   \bool_if:nT {
8846     \l__knot_self_intersections_bool
8847     &&
8848     \l__knot_splits_bool
8849   }
8850   {
8851     \tl_clear:N \l__knot_tmpa_tl
8852     \spath_remove_empty_components:c {knot strand #1}
8853     \spath_initialpoint:Nv \l__knot_tmpa_tl {knot strand #1}
8854     \tl_put_left:Nv \l__knot_tmpa_tl \c_spath_moveto_tl
8855     \spath_segments_to_seq:Nv \l__knot_segments_seq {knot strand #1}
8856     \seq_map_function:NN \l__knot_segments_seq \knot_split_self_intersects:N
8857     \tl_set_eq:cN {knot strand #1} \l__knot_tmpa_tl
8858   }
8859 }

```

(End of 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!

```

8860 \cs_new_protected_nopar:Npn \knot_split_self_intersects:N #1
8861 {
8862   \knot_debug:n {knot~ split~ self~ intersects}
8863   \tl_set:Nx \l__knot_tmpc_tl {\tl_item:nn {#1} {4}}
8864   \token_case_meaning:NnF \l__knot_tmpc_tl
8865   {
8866     \c_spath_curvetoa_tl
8867     {
8868       \fp_set:Nn \l__knot_tmpa_fp
8869       {
8870         (\tl_item:nn {#1} {3} - 3 * \tl_item:nn {#1} {6}
8871         + 3 * \tl_item:nn {#1} {9} - \tl_item:nn {#1} {12})
8872         *
8873         (3 * \tl_item:nn {#1} {8} - 3 * \tl_item:nn {#1} {11})
8874         -
8875         (\tl_item:nn {#1} {2} - 3 * \tl_item:nn {#1} {5}
8876         + 3 * \tl_item:nn {#1} {8} - \tl_item:nn {#1} {11})
8877         *
8878         (3 * \tl_item:nn {#1} {9} - 3 * \tl_item:nn {#1} {12})
8879       }
8880       \fp_set:Nn \l__knot_tmpb_fp
8881       {
8882         (\tl_item:nn {#1} {2} - 3 * \tl_item:nn {#1} {5}
8883         + 3 * \tl_item:nn {#1} {8} - \tl_item:nn {#1} {11})
8884         *
8885         (3 * \tl_item:nn {#1} {6} - 6 * \tl_item:nn {#1} {9}
8886         + 3 * \tl_item:nn {#1} {12})
8887         -

```

```

8888     (\tl_item:nn {#1} {3} - 3 * \tl_item:nn {#1} {6}
8889     + 3 * \tl_item:nn {#1} {9} - \tl_item:nn {#1} {12})
8890     *
8891     (3 * \tl_item:nn {#1} {5} - 6 * \tl_item:nn {#1} {8}
8892     + 3 * \tl_item:nn {#1} {11})
8893   }
8894   \fp_compare:nTF
8895   {
8896     \l__knot_tmpb_fp != 0
8897   }
8898   {
8899     \fp_set:Nn \l__knot_tmpa_fp { .5 * \l__knot_tmpa_fp / \l__knot_tmpb_fp }
8900     \bool_if:nTF
8901     {
8902       \fp_compare_p:n { 0 < \l__knot_tmpa_fp }
8903       &&
8904       \fp_compare_p:n { \l__knot_tmpa_fp < 1 }
8905     }
8906     {
8907       \spath_split_curve:NNnV
8908       \l__knot_tmpc_tl
8909       \l__knot_tmpd_tl
8910       {#1}
8911       \l__knot_tmpa_fp
8912       \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8913       \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8914       \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8915       \tl_set:Nx \l__knot_tmpd_tl {\tl_tail:N \l__knot_tmpd_tl}
8916       \tl_set:Nx \l__knot_tmpd_tl {\tl_tail:N \l__knot_tmpd_tl}
8917       \tl_set:Nx \l__knot_tmpd_tl {\tl_tail:N \l__knot_tmpd_tl}
8918       \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpc_tl
8919       \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpd_tl
8920     }
8921     {
8922       \tl_set:Nn \l__knot_tmpc_tl {#1}
8923       \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8924       \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8925       \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8926       \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpc_tl
8927     }
8928   }
8929   {
8930     \tl_set:Nn \l__knot_tmpc_tl {#1}
8931     \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8932     \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8933     \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8934     \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpc_tl
8935   }
8936 }
8937 \c_spath_lineto_tl
8938 {
8939   \tl_set:Nn \l__knot_tmpc_tl {#1}
8940   \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8941   \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}

```

```

8942     \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8943     \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpc_tl
8944   }
8945 }
8946 {
8947   \tl_put_right:Nn \l__knot_tmpa_tl {#1}
8948 }
8949 }

```

(End of definition for `\knot_split_self_intersects:N`.)

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

```

8950 \cs_new_protected_nopar:Npn \knot_intersections:nn #1#2
8951 {
8952   \knot_debug:x {knot~ intersections~ between~
8953     \l__knot_prefix_tl \c_space_tl #1~ and~ #2}
8954   \group_begin:
8955   \tl_set_eq:NN \l__knot_tmpa_tl \l__knot_prefix_tl
8956   \tl_put_right:Nn \l__knot_tmpa_tl {#1}
8957   \tl_set_eq:NN \l__knot_tmpb_tl \l__knot_prefix_tl
8958   \tl_put_right:Nn \l__knot_tmpb_tl {#2}
8959   \tl_set_eq:Nc \l__knot_tmpc_tl {knot \tl_use:N \l__knot_tmpa_tl}
8960   \tl_set_eq:Nc \l__knot_tmpd_tl {knot \tl_use:N \l__knot_tmpb_tl}
8961
8962   \bool_if:nTF {
8963     \l__knot_save_bool
8964     &&
8965     \tl_if_exist_p:c {
8966       knot~ intersections~
8967       \tl_use:N \l__knot_name_tl -
8968       \tl_use:N \l__knot_tmpa_tl -
8969       \tl_use:N \l__knot_tmpb_tl
8970     }
8971   }
8972   {
8973     \tl_use:c
8974     {
8975       knot~ intersections~ \tl_use:N \l__knot_name_tl -
8976       \tl_use:N \l__knot_tmpa_tl -
8977       \tl_use:N \l__knot_tmpb_tl
8978     }
8979   }
8980   {
8981     \pgfintersectionofpaths{\pgfsetpath\l__knot_tmpc_tl}{\pgfsetpath\l__knot_tmpd_tl}
8982
8983   }
8984
8985   \knot_debug:x {found~\pgfintersectionsolutions\c_space_tl~ intersections}
8986   \int_compare:nT {\pgfintersectionsolutions > 0}
8987   {
8988     \int_step_function:nnnN
8989     {1}
8990     {1}
8991     {\pgfintersectionsolutions}

```

```

8992     \knot_do_intersection:n
8993   }
8994
8995   \knot_save_intersections:VV \l__knot_tmpa_tl \l__knot_tmpb_tl
8996   \group_end:
8997 }

```

(End of definition for \knot_intersections:nn.)

\knot_save_intersections:nn

```

8998 \cs_new_protected_nopar:Npn \knot_save_intersections:nn #1#2
8999 {
9000   \knot_debug:n {knot~ save~ intersections}
9001   \bool_if:NT \l__knot_save_bool
9002   {
9003     \tl_clear:N \l__knot_aux_tl
9004     \tl_put_right:Nn \l__knot_aux_tl
9005     {
9006       \def\pgfintersectionsolutions
9007     }
9008     \tl_put_right:Nx \l__knot_aux_tl
9009     {
9010       {\int_eval:n {\pgfintersectionsolutions}}
9011     }
9012     \int_compare:nT {\pgfintersectionsolutions > 0}
9013     {
9014       \int_step_inline:nnnn {1} {1} {\pgfintersectionsolutions}
9015       {
9016         \pgfpointintersectionsolution{##1}
9017         \dim_set:Nn \l__knot_tmpa_dim {\pgf@x}
9018         \dim_set:Nn \l__knot_tmpb_dim {\pgf@y}
9019         \tl_put_right:Nn \l__knot_aux_tl
9020         {
9021           \expandafter\def\csname pgfpoint@intersect@solution@##1\endcsname
9022         }
9023         \tl_put_right:Nx \l__knot_aux_tl
9024         {
9025           {
9026             \exp_not:N \pgf@x
9027             =
9028             \dim_use:N \l__knot_tmpa_dim
9029             \exp_not:N \relax
9030             \exp_not:N \pgf@y
9031             =
9032             \dim_use:N \l__knot_tmpb_dim
9033             \exp_not:N \relax
9034           }
9035         }
9036       }
9037       \tl_set:Nn \l__knot_auxa_tl {\expandafter \gdef \csname knot~ intersections~}
9038       \tl_put_right:Nx \l__knot_auxa_tl {\tl_use:N \l__knot_name_tl - #1 - #2}
9039       \tl_put_right:Nn \l__knot_auxa_tl {\endcsname}
9040       \tl_put_right:Nx \l__knot_auxa_tl {\tl_to_str:N \l__knot_aux_tl}}
9041     \protected@write\@auxout{\tl_to_str:N \l__knot_auxa_tl}

```

```

9042     }
9043   }
9044 }
9045 \cs_generate_variant:Nn \knot_save_intersections:nn {VV}

```

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

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

```

9046 \cs_new_protected_nopar:Npn \knot_do_intersection:n #1
9047 {
9048   \knot_debug:n {knot~ do~ intersection~ #1}

```

Get the intersection coordinates.

```

9049   \pgfpointintersectionsolution{#1}
9050   \dim_set:Nn \l__knot_tmpa_dim {\pgf@x}
9051   \dim_set:Nn \l__knot_tmpb_dim {\pgf@y}
9052   \knot_debug:x {intersection~at~
9053     (\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.

```

9054   \bool_set_false:N \l__knot_skip_bool
9055   \bool_if:NT \l__knot_self_intersections_bool
9056   {

```

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

```

9057     \tl_set:Nn \l__knot_tmpc_tl {knot previous}
9058     \tl_put_right:NV \l__knot_tmpc_tl \l__knot_tmpa_tl
9059     \tl_set:Nv \l__knot_tmpc_tl \l__knot_tmpc_tl
9060     \tl_if_eq:NNT \l__knot_tmpc_tl \l__knot_tmpb_tl
9061     {
9062       \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpb_tl {final point}
9063       {
9064         \bool_set_true:N \l__knot_skip_bool
9065       }
9066     }
9067
9068     \tl_set:Nn \l__knot_tmpc_tl {knot previous}
9069     \tl_put_right:NV \l__knot_tmpc_tl \l__knot_tmpb_tl
9070     \tl_set:Nv \l__knot_tmpc_tl \l__knot_tmpc_tl
9071     \tl_if_eq:NNT \l__knot_tmpc_tl \l__knot_tmpa_tl
9072     {
9073       \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpa_tl {final point}
9074       {
9075         \bool_set_true:N \l__knot_skip_bool
9076       }
9077     }
9078   }

```

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

```

9079   \bool_if:NT \l__knot_ignore_ends_bool
9080   {
9081     \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpa_tl {initial point}
9082     {
9083       \bool_set_true:N \l__knot_skip_bool

```



```

9084     }
9085     \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpa_tl {final point}
9086     {
9087         \bool_set_true:N \l__knot_skip_bool
9088     }
9089     \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpb_tl {initial point}
9090     {
9091         \bool_set_true:N \l__knot_skip_bool
9092     }
9093     \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpb_tl {final point}
9094     {
9095         \bool_set_true:N \l__knot_skip_bool
9096     }
9097 }

```

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

```

9098     \bool_if:NF \l__knot_skip_bool
9099     {
9100
9101         \int_gincr:N \g__knot_intersections_int
9102         \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.

```

9103     \bool_if:nF
9104     {
9105         \tl_if_exist_p:c {l__knot_ignore_crossing_ \int_use:N
9106             \g__knot_intersections_int}
9107         &&
9108         ! \tl_if_empty_p:c {l__knot_ignore_crossing_ \int_use:N
9109             \g__knot_intersections_int}
9110     }
9111     {

```

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

```

9112     \bool_if:nTF
9113     {
9114         \tl_if_exist_p:c {l__knot_crossing_ \int_use:N
9115             \g__knot_intersections_int}
9116         &&
9117         ! \tl_if_empty_p:c {l__knot_crossing_ \int_use:N
9118             \g__knot_intersections_int}
9119     }
9120     {
9121         \tl_set_eq:NN \l__knot_tmpg_tl \l__knot_tmpb_tl
9122     }
9123     {
9124         \tl_set_eq:NN \l__knot_tmpg_tl \l__knot_tmpa_tl
9125     }

```

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.

```

9126     \bool_if:NT \l__knot_self_intersections_bool

```

```

9127 {
9128   \knot_test_endpoint:N\Nt
9129   \l__knot_redraw_tolerance_dim \l__knot_tmpg_tl {initial point}
9130   {
9131     \bool_set_true:N \l__knot_prepend_prev_bool
9132   }
9133   {
9134     \bool_set_false:N \l__knot_prepend_prev_bool
9135   }
9136   \knot_test_endpoint:N\Nt
9137   \l__knot_redraw_tolerance_dim \l__knot_tmpg_tl {final point}
9138   {
9139     \bool_set_true:N \l__knot_append_next_bool
9140   }
9141   {
9142     \bool_set_false:N \l__knot_append_next_bool
9143   }

```

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

```

9144   \bool_if:nT
9145   {
9146     \l__knot_prepend_prev_bool || \l__knot_append_next_bool
9147   }
9148   {
9149     \tl_clear_new:c {knot \tl_use:N \l__knot_prefix_tl -1}
9150     \tl_set_eq:cc
9151     {knot \tl_use:N \l__knot_prefix_tl -1}
9152     {knot \tl_use:N \l__knot_tmpg_tl}
9153
9154     \tl_clear_new:c {l__knot_options_ \tl_use:N \l__knot_prefix_tl -1}
9155     \tl_set_eq:cc
9156     {l__knot_options_ \tl_use:N \l__knot_prefix_tl -1}
9157     {l__knot_options_ \tl_use:N \l__knot_tmpg_tl}
9158
9159     \bool_if:nT
9160     {
9161       \l__knot_prepend_prev_bool
9162       &&
9163       \tl_if_exist_p:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9164       &&
9165       !\tl_if_empty_p:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9166     }
9167     {
9168       \knot_debug:x {Prepending~
9169         \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}}
9170       \spath_prepend_no_move:cv
9171       {knot \tl_use:N \l__knot_prefix_tl -1}
9172       {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.

```

9173   \bool_if:nT
9174   {
9175     \l__knot_splits_bool
9176     &&

```

```

9177         \tl_if_exist_p:c {knot previous
9178             \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9179         }
9180         &&
9181         !\tl_if_empty_p:c {knot previous
9182             \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9183         }
9184     }
9185     {
9186         \knot_test_endpoint:Nvnt
9187         \l__knot_redraw_tolerance_dim
9188         {knot previous \tl_use:N \l__knot_tmpg_tl}
9189         {initial point}
9190         {
9191             \knot_debug:x {Prepending~
9192                 \tl_use:c {knot previous
9193                     \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9194                 }
9195             }
9196             \spath_prepend_no_move:cv
9197             {knot \tl_use:N \l__knot_prefix_tl -1}
9198             {knot \tl_use:c
9199                 {knot previous \tl_use:c
9200                     {knot previous \tl_use:N \l__knot_tmpg_tl}
9201                 }
9202             }
9203             \tl_set_eq:Nc \l__knot_tmpa_tl
9204             {knot \tl_use:N \l__knot_prefix_tl -1}
9205         }
9206     }
9207 }
9208

```

Now the same for appending.

```

9209     \bool_if:nT
9210     {
9211         \l__knot_append_next_bool
9212         &&
9213         \tl_if_exist_p:c {knot next \tl_use:N \l__knot_tmpg_tl}
9214         &&
9215         !\tl_if_empty_p:c {knot next \tl_use:N \l__knot_tmpg_tl}
9216     }
9217     {
9218         \knot_debug:x {Appending~
9219             \tl_use:c {knot next \tl_use:N \l__knot_tmpg_tl}}
9220         \spath_append_no_move:cv
9221         {knot \tl_use:N \l__knot_prefix_tl -1}
9222         {knot \tl_use:c {knot next \tl_use:N \l__knot_tmpg_tl}}
9223         \bool_if:nT
9224         {
9225             \l__knot_splits_bool
9226             &&
9227             \tl_if_exist_p:c {knot next \tl_use:c { knot next \tl_use:N
9228                 \l__knot_tmpg_tl}}
9229             &&

```

```

9230         !\tl_if_empty_p:c {knot next
9231         \tl_use:c { knot next \tl_use:N \l__knot_tmpg_tl}
9232     }
9233 }
9234 {
9235     \knot_debug:x {Testing~ whether~ to~ append~
9236     {knot next \tl_use:c { knot next \tl_use:N \l__knot_tmpg_tl}}
9237 }
9238 \knot_test_endpoint:NvNT
9239 \l__knot_redraw_tolerance_dim
9240 {knot next \tl_use:N \l__knot_tmpg_tl}
9241 {final point}
9242 {
9243     \knot_debug:x {Appending~
9244     {knot next \tl_use:c { knot next \tl_use:N \l__knot_tmpg_tl}}
9245 }
9246 \spath_append_no_move:cv
9247 {knot \tl_use:N \l__knot_prefix_tl -1}
9248 {knot \tl_use:c
9249     {knot next \tl_use:c
9250     {knot next \tl_use:N \l__knot_tmpg_tl}
9251     }
9252     }
9253 }
9254 }
9255 }
9256 \tl_set:Nn \l__knot_tmpg_tl {\tl_use:N \l__knot_prefix_tl -1}
9257 }
9258 }

```

Now we render the crossing.

```

9259 \pgfscope
9260 \group_begin:
9261 \tikzset{
9262     knot~ diagram/every~ intersection/.try,
9263     every~ intersection/.try,
9264     knot~ diagram/intersection~ \int_use:N \g__knot_intersections_int/.try
9265 }
9266 \knot_draw_crossing:VVV \l__knot_tmpg_tl \l__knot_tmpa_dim \l__knot_tmpb_dim
9267 \coordinate
9268 (\l__knot_name_tl \c_space_tl \int_use:N \g__knot_intersections_int)
9269 at (\dim_use:N \l__knot_tmpa_dim, \dim_use:N \l__knot_tmpb_dim);
9270 \group_end:
9271 \endpgfscope

```

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

```

9272     }

```

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

```

9273 \tl_if_empty:NF \l__knot_node_tl
9274 {
9275     \seq_gpush:Nx
9276     \g__knot_nodes_seq
9277     {
9278         \l__knot_node_tl

```

```

9279         at
9280         (\dim_use:N \l__knot_tmpa_dim, \dim_use:N \l__knot_tmpb_dim) {};
9281     }
9282 }
9283 }
9284 }
9285
9286 \cs_generate_variant:Nn \knot_intersections:nn {VV}

```

(End of definition for \knot_do_intersection:n.)

`\knot_test_endpoint:N` Test whether the point is near the intersection point.

```

9287 \prg_new_conditional:Npnn \knot_test_endpoint:NN #1#2 {p,T,F,TF}
9288 {
9289   \dim_compare:nTF
9290   {
9291     \dim_abs:n { \l__knot_tmpa_dim - \tl_item:Nn #2 {1}}
9292     +
9293     \dim_abs:n { \l__knot_tmpb_dim - \tl_item:Nn #2 {2}}
9294     <
9295     #1
9296   }
9297   {
9298     \prg_return_true:
9299   }
9300   {
9301     \prg_return_false:
9302   }
9303 }

```

(End of definition for \knot_test_endpoint:N.)

`\knot_test_endpoint:nn` Wrapper around the above.

```

9304 \prg_new_protected_conditional:Npnn \knot_test_endpoint:Nnn #1#2#3 {T,F,TF}
9305 {
9306   \use:c {spath_#3:Nv} \l__knot_tmpd_tl {knot #2}
9307   \knot_test_endpoint:NNTF #1 \l__knot_tmpd_tl
9308   {
9309     \prg_return_true:
9310   }
9311   {
9312     \prg_return_false:
9313   }
9314 }
9315
9316 \cs_generate_variant:Nn \knot_test_endpoint:NnnT {NVnT,NvnT}
9317 \cs_generate_variant:Nn \knot_test_endpoint:NnnF {NVnF,NvnF}
9318 \cs_generate_variant:Nn \knot_test_endpoint:NnnTF {NVnTF,NvnTF}

```

(End of definition for \knot_test_endpoint:nn.)

`\knot_draw_crossing:nnn` This is the code that actually renders a crossing.

```

9319 \cs_new_protected_nopar:Npn \knot_draw_crossing:nnn #1#2#3
9320 {
9321   \knot_debug:n {knot~ draw~ crossing}

```

```

9322 \group_begin:
9323 \pgfscope
9324 \path[knot~ diagram/background~ clip] (#2, #3)
9325 circle[radius=\l__knot_clip_bg_radius_dim];
9326
9327 \tl_set:Nn \l__knot_tmpa_tl {knot~ diagram/every~ strand/.try,}
9328 \tl_if_exist:cT {l__knot_options_ #1}
9329 {
9330 \tl_put_right:Nv \l__knot_tmpa_tl {l__knot_options_ #1}
9331 }
9332 \tl_put_right:Nn \l__knot_tmpa_tl
9333 {
9334 ,knotbg
9335 ,line~ width= \tl_use:N \l__knot_clip_width_tl * \pgflinewidth
9336 }
9337 \spath_tikz_path:Vv \l__knot_tmpa_tl {knot #1}
9338
9339 \endpgfscope
9340
9341 \pgfscope
9342 \path[knot~ diagram/clip] (#2, #3)
9343 circle[radius=\l__knot_clip_draw_radius_dim];
9344
9345 \tl_set:Nn \l__knot_tmpa_tl {knot~ diagram/every~ strand/.try,}
9346 \tl_if_exist:cT {l__knot_options_ #1}
9347 {
9348 \tl_put_right:Nv \l__knot_tmpa_tl {l__knot_options_ #1}
9349 }
9350 \tl_put_right:Nn \l__knot_tmpa_tl
9351 {
9352 ,knot~ diagram/only~ when~ rendering/.try
9353 ,only~ when~ rendering/.try
9354 }
9355 \spath_tikz_path:Vv \l__knot_tmpa_tl {knot #1}
9356
9357 \endpgfscope
9358 \group_end:
9359 }
9360
9361 \cs_generate_variant:Nn \knot_draw_crossing:nnn {nVV, VVV}
9362
9363 \cs_new_protected_nopar:Npn \knot_draw_crossing:nn #1#2
9364 {
9365 \tikz@scan@one@point\pgfutil@firstofone #2 \relax
9366 \knot_draw_crossing:nVV {#1} \pgf@x \pgf@y
9367 }

```

(End of definition for \knot_draw_crossing:nnn.)

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

```

9368 \cs_new_protected_nopar:Npn \knot_split_strands:
9369 {
9370 \knot_debug:n {knot~ split~ strands}
9371 \int_gzero:N \g__knot_filaments_int

```

```

9372 \int_step_function:nnnN {1} {1} {\l__knot_strands_int} \knot_split_strand:n
9373 \int_step_function:nnnN {1} {1} {\g__knot_filaments_int} \knot_compute_nexts:n
9374 }

```

(End of 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.

```

9375 \cs_new_protected_nopar:Npn \knot_compute_nexts:n #1
9376 {
9377   \knot_debug:n {knot~ compute~ nexts}
9378   \tl_clear_new:c {knot next \tl_use:c {knot previous filament #1}}
9379   \tl_set:cn {knot next \tl_use:c {knot previous filament #1}} {filament #1}
9380 }

```

(End of definition for \knot_compute_nexts:n.)

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

```

9381 \cs_new_protected_nopar:Npn \knot_split_strand:n #1
9382 {
9383   \knot_debug:n {knot~ split~ strand}
9384   \int_set_eq:NN \l__knot_component_start_int \g__knot_filaments_int
9385   \int_incr:N \l__knot_component_start_int
9386   \tl_set_eq:Nc \l__knot_tmpa_tl {\l__knot_options_strand #1}
9387   \spath_remove_empty_components:c {knot strand #1}
9388   \spath_segments_to_seq:Nv \l__knot_segments_seq {knot strand #1}
9389   \seq_map_function:NN \l__knot_segments_seq \knot_save_filament:N
9390 }

```

(End of definition for \knot_split_strand:n.)

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

```

9391 \cs_new_protected_nopar:Npn \knot_save_filament:N #1
9392 {
9393   \knot_debug:n {knot~ save~ filament}
9394   \tl_set:Nx \l__knot_tmpb_tl {\tl_item:nn {#1} {4}}
9395   \token_case_meaning:NnF \l__knot_tmpb_tl
9396   {
9397     \c_spath_moveto_tl
9398     {
9399       \int_compare:nT {\l__knot_component_start_int < \g__knot_filaments_int}
9400       {
9401         \int_set_eq:NN \l__knot_component_start_int \g__knot_filaments_int
9402       }
9403     }
9404     \c_spath_lineto_tl
9405     {
9406       \int_gincr:N \g__knot_filaments_int
9407       \tl_clear_new:c {knot filament \int_use:N \g__knot_filaments_int}
9408       \tl_set:cn {knot filament \int_use:N \g__knot_filaments_int} {#1}
9409
9410       \tl_clear_new:c {\l__knot_options_filament \int_use:N \g__knot_filaments_int}
9411       \tl_set_eq:cN {\l__knot_options_filament \int_use:N \g__knot_filaments_int}
9412       \l__knot_tmpa_tl
9413

```

```

9414     \tl_clear_new:c {knot previous filament \int_use:N \g__knot_filaments_int}
9415     \int_compare:nF {\l__knot_component_start_int == \g__knot_filaments_int}
9416     {
9417         \tl_set:cx {knot previous filament \int_use:N \g__knot_filaments_int}
9418         {filament \int_eval:n {\g__knot_filaments_int - 1}}
9419     }
9420 }
9421 \c_spath_curvetoa_tl
9422 {
9423     \int_gincr:N \g__knot_filaments_int
9424     \tl_clear_new:c {knot filament \int_use:N \g__knot_filaments_int}
9425     \tl_set:cn {knot filament \int_use:N \g__knot_filaments_int} {#1}
9426     \tl_clear_new:c {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9427     \tl_set_eq:cN {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9428     \l__knot_tmpa_tl
9429
9430     \tl_clear_new:c {knot previous filament \int_use:N \g__knot_filaments_int}
9431     \int_compare:nF {\l__knot_component_start_int == \g__knot_filaments_int}
9432     {
9433         \tl_set:cx
9434         {knot previous filament \int_use:N \g__knot_filaments_int}
9435         {filament \int_eval:n {\g__knot_filaments_int - 1}}
9436     }
9437 }
9438 \c_spath_closepath_tl
9439 {
9440     \int_gincr:N \g__knot_filaments_int
9441     \tl_clear_new:c {knot filament \int_use:N \g__knot_filaments_int}
9442     \tl_clear:N \l__knot_tmpa_tl
9443     \tl_put_right:Nx
9444     {
9445         \tl_item:nn {#1} {1}\tl_item:nn {#1} {2}\tl_item:nn {#1} {3}
9446     }
9447     \tl_put_right:NV \l__knot_tmpa_tl \c_spath_lineto_tl
9448     \tl_put_right:Nx {\tl_item:nn {#1} {5}\tl_item:nn {#1} {6}}
9449
9450     \tl_set:cV {knot filament \int_use:N \g__knot_filaments_int} \l__knot_tmpa_tl
9451     \tl_set_eq:cN {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9452     \l__knot_tmpa_tl
9453     \tl_clear_new:c {knot previous filament \int_use:N \g__knot_filaments_int}
9454     \int_compare:nF {\l__knot_component_start_int == \g__knot_filaments_int}
9455     {
9456         \tl_set:cx
9457         {knot previous filament \int_use:N \g__knot_filaments_int}
9458         {filament \int_eval:n {\g__knot_filaments_int - 1}}
9459     }
9460     \tl_set:cx
9461     {knot previous filament \int_use:N \l__knot_component_start_int}
9462     {filament \int_use:N \g__knot_filaments_int}
9463 }
9464 }
9465 {
9466 }
9467 }

```


(End of definition for \knot_save_filament:N.)

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

```
9468 \NewDocumentCommand \redraw { m m }
9469 {
9470 % \tikz@scan@one@point\pgfutil@firstofone #2 \relax
9471 \tl_put_right:Nn \l__knot_redraws_tl {\knot_draw_crossing:nm}
9472 \tl_put_right:Nx \l__knot_redraws_tl {
9473   {strand #1} {#2}% {\dim_use:N \pgf@x} {\dim_use:N \pgf@y}
9474 }
9475 }
```

(End of definition for \redraw.)

```
9476 \ExplSyntaxOff
```

<@@=>

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

```
9477 \def\pgf@sh__knotknotanchor#1#2{%
9478   \anchor{#2 north west}{%
9479     \csname pgf@anchor@knot #1@north west\endcsname%
9480     \pgf@x=#2\pgf@x%
9481     \pgf@y=#2\pgf@y%
9482   }%
9483   \anchor{#2 north east}{%
9484     \csname pgf@anchor@knot #1@north east\endcsname%
9485     \pgf@x=#2\pgf@x%
9486     \pgf@y=#2\pgf@y%
9487   }%
9488   \anchor{#2 south west}{%
9489     \csname pgf@anchor@knot #1@south west\endcsname%
9490     \pgf@x=#2\pgf@x%
9491     \pgf@y=#2\pgf@y%
9492   }%
9493   \anchor{#2 south east}{%
9494     \csname pgf@anchor@knot #1@south east\endcsname%
9495     \pgf@x=#2\pgf@x%
9496     \pgf@y=#2\pgf@y%
9497   }%
9498   \anchor{#2 north}{%
9499     \csname pgf@anchor@knot #1@north\endcsname%
9500     \pgf@x=#2\pgf@x%
9501     \pgf@y=#2\pgf@y%
9502   }%
9503   \anchor{#2 east}{%
9504     \csname pgf@anchor@knot #1@east\endcsname%
9505     \pgf@x=#2\pgf@x%
9506     \pgf@y=#2\pgf@y%
9507   }%
9508   \anchor{#2 west}{%
9509     \csname pgf@anchor@knot #1@west\endcsname%
9510     \pgf@x=#2\pgf@x%
9511     \pgf@y=#2\pgf@y%
9512   }%
```

```

9513 \anchor{#2 south}{%
9514 \csname pgf@anchor@knot #1@south\endcsname%
9515 \pgf@x=#2\pgf@x%
9516 \pgf@y=#2\pgf@y%
9517 }%
9518 }

```

(End of definition for \pgf@sh__knotknotanchor.)

knot_crossing

```

9519 \pgfdeclareshape{knot crossing}
9520 {
9521 \inheritsavedanchors[from=circle] % this is nearly a circle
9522 \inheritanchorborder[from=circle]
9523 \inheritanchor[from=circle]{north}
9524 \inheritanchor[from=circle]{north west}
9525 \inheritanchor[from=circle]{north east}
9526 \inheritanchor[from=circle]{center}
9527 \inheritanchor[from=circle]{west}
9528 \inheritanchor[from=circle]{east}
9529 \inheritanchor[from=circle]{mid}
9530 \inheritanchor[from=circle]{mid west}
9531 \inheritanchor[from=circle]{mid east}
9532 \inheritanchor[from=circle]{base}
9533 \inheritanchor[from=circle]{base west}
9534 \inheritanchor[from=circle]{base east}
9535 \inheritanchor[from=circle]{south}
9536 \inheritanchor[from=circle]{south west}
9537 \inheritanchor[from=circle]{south east}
9538 \inheritanchorborder[from=circle]
9539 \pgf@sh__knotknotanchor{crossing}{2}
9540 \pgf@sh__knotknotanchor{crossing}{3}
9541 \pgf@sh__knotknotanchor{crossing}{4}
9542 \pgf@sh__knotknotanchor{crossing}{8}
9543 \pgf@sh__knotknotanchor{crossing}{16}
9544 \pgf@sh__knotknotanchor{crossing}{32}
9545 \backgroundpath{
9546 \pgfutil@tempdima=\radius%
9547 \pgfmathsetlength{\pgf@xb}{\pgfkeysvalueof{/pgf/outer xsep}}%
9548 \pgfmathsetlength{\pgf@yb}{\pgfkeysvalueof{/pgf/outer ysep}}%
9549 \ifdim\pgf@xb<\pgf@yb%
9550 \advance\pgfutil@tempdima by-\pgf@yb%
9551 \else%
9552 \advance\pgfutil@tempdima by-\pgf@xb%
9553 \fi%
9554 }
9555 }

```

(End of definition for knot crossing.)

knot_over_cross

```

9556 \pgfdeclareshape{knot over cross}
9557 {
9558 \inheritsavedanchors[from=rectangle] % this is nearly a circle
9559 \inheritanchorborder[from=rectangle]

```

```

9560 \inheritanchor[from=rectangle]{north}
9561 \inheritanchor[from=rectangle]{north west}
9562 \inheritanchor[from=rectangle]{north east}
9563 \inheritanchor[from=rectangle]{center}
9564 \inheritanchor[from=rectangle]{west}
9565 \inheritanchor[from=rectangle]{east}
9566 \inheritanchor[from=rectangle]{mid}
9567 \inheritanchor[from=rectangle]{mid west}
9568 \inheritanchor[from=rectangle]{mid east}
9569 \inheritanchor[from=rectangle]{base}
9570 \inheritanchor[from=rectangle]{base west}
9571 \inheritanchor[from=rectangle]{base east}
9572 \inheritanchor[from=rectangle]{south}
9573 \inheritanchor[from=rectangle]{south west}
9574 \inheritanchor[from=rectangle]{south east}
9575 \inheritanchorborder[from=rectangle]
9576 \backgroundpath{
9577   \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9578   \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9579   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@ya}}
9580   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9581 }
9582 \foregroundpath{
9583 % store lower right in xa/ya and upper right in xb/yb
9584   \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9585   \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9586   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9587   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@ya}}
9588 }
9589 }

```

(End of definition for knot over cross.)

knot_{under}cross

```

9590 \pgfdeclareshape{knot under cross}
9591 {
9592   \inheritsavedanchors[from=rectangle] % this is nearly a circle
9593   \inheritanchorborder[from=rectangle]
9594   \inheritanchor[from=rectangle]{north}
9595   \inheritanchor[from=rectangle]{north west}
9596   \inheritanchor[from=rectangle]{north east}
9597   \inheritanchor[from=rectangle]{center}
9598   \inheritanchor[from=rectangle]{west}
9599   \inheritanchor[from=rectangle]{east}
9600   \inheritanchor[from=rectangle]{mid}
9601   \inheritanchor[from=rectangle]{mid west}
9602   \inheritanchor[from=rectangle]{mid east}
9603   \inheritanchor[from=rectangle]{base}
9604   \inheritanchor[from=rectangle]{base west}
9605   \inheritanchor[from=rectangle]{base east}
9606   \inheritanchor[from=rectangle]{south}
9607   \inheritanchor[from=rectangle]{south west}
9608   \inheritanchor[from=rectangle]{south east}
9609   \inheritanchorborder[from=rectangle]

```

```

9610 \backgroundpath{
9611   \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9612   \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9613   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9614   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9615 }
9616 \foregroundpath{
9617 % store lower right in xa/ya and upper right in xb/yb
9618   \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9619   \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9620   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9621   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9622 }
9623 }

```

(End of definition for knot under cross.)

knot_vert

```

9624 \pgfdeclareshape{knot vert}
9625 {
9626   \inheritsavedanchors[from=rectangle] % this is nearly a circle
9627   \inheritanchorborder[from=rectangle]
9628   \inheritanchor[from=rectangle]{north}
9629   \inheritanchor[from=rectangle]{north west}
9630   \inheritanchor[from=rectangle]{north east}
9631   \inheritanchor[from=rectangle]{center}
9632   \inheritanchor[from=rectangle]{west}
9633   \inheritanchor[from=rectangle]{east}
9634   \inheritanchor[from=rectangle]{mid}
9635   \inheritanchor[from=rectangle]{mid west}
9636   \inheritanchor[from=rectangle]{mid east}
9637   \inheritanchor[from=rectangle]{base}
9638   \inheritanchor[from=rectangle]{base west}
9639   \inheritanchor[from=rectangle]{base east}
9640   \inheritanchor[from=rectangle]{south}
9641   \inheritanchor[from=rectangle]{south west}
9642   \inheritanchor[from=rectangle]{south east}
9643   \inheritanchorborder[from=rectangle]
9644   \backgroundpath{
9645 % store lower right in xa/ya and upper right in xb/yb
9646   \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9647   \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9648   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9649   \pgfpathlineto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9650   \pgfpathmoveto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9651   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9652 }
9653 }

```

(End of definition for knot vert.)

knot_horiz

```

9654 \pgfdeclareshape{knot horiz}
9655 {
9656   \inheritsavedanchors[from=rectangle] % this is nearly a circle

```

```

9657 \inheritanchorborder[from=rectangle]
9658 \inheritanchor[from=rectangle]{north}
9659 \inheritanchor[from=rectangle]{north west}
9660 \inheritanchor[from=rectangle]{north east}
9661 \inheritanchor[from=rectangle]{center}
9662 \inheritanchor[from=rectangle]{west}
9663 \inheritanchor[from=rectangle]{east}
9664 \inheritanchor[from=rectangle]{mid}
9665 \inheritanchor[from=rectangle]{mid west}
9666 \inheritanchor[from=rectangle]{mid east}
9667 \inheritanchor[from=rectangle]{base}
9668 \inheritanchor[from=rectangle]{base west}
9669 \inheritanchor[from=rectangle]{base east}
9670 \inheritanchor[from=rectangle]{south}
9671 \inheritanchor[from=rectangle]{south west}
9672 \inheritanchor[from=rectangle]{south east}
9673 \inheritanchorborder[from=rectangle]
9674 \foregroundpath{
9675 % store lower right in xa/ya and upper right in xb/yb
9676   \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9677   \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9678   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@ya}}
9679   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@ya}}
9680   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9681   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9682 }
9683 }

```

(End of definition for knot horiz.)