

The `spath3` package: code

Andrew Stacey

loopspace@mathforge.org

v2.9 from 2026/05/31

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{pgf}
```

```
4 \ProvidesExplPackage {spath3} {2026/05/31} {2.9} {Functions for  
5 manipulating PGF soft paths}
```

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.

```
6 \cs_new_protected:Nn \__spath_tl_put_right_braced:Nn
```

```
7 {
```

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

```
9 }
```

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

```
11
```

```
12 \cs_new_protected:Nn \__spath_tl_gput_right_braced:Nn
```

```
13 {
```

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

```
15 }
```

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

```
17 \cs_new_protected:Nn \__spath_tl_put_left_braced:Nn
```

```
18 {
```

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

```
20 }
```

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

```
22
```

```
23 \cs_new_protected:Nn \__spath_tl_gput_left_braced:Nn
```

```
24 {
```

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

```
26 }
```

```
27 \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 T_EX to work the way I wanted. I'm really defining *functions* but T_EX doesn't really have that concept, even with all the amazing L^AT_EX3 stuff. The main issue I had was with scoping and return values. By default, T_EX

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.

```

28 \tl_new:N \g__spath_output_tl
29 \int_new:N \g__spath_output_int
30 \seq_new:N \g__spath_output_seq
31 \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.

```

32 \tl_new:N \l__spath_tmpa_tl
33 \tl_new:N \l__spath_tmpb_tl
34 \tl_new:N \l__spath_tmpe_tl
35 \tl_new:N \l__spath_tmpe_tl
36 \tl_new:N \l__spath_tmpe_tl
37 \tl_new:N \l__spath_tmpe_tl
38 \tl_new:N \l__spath_tmpe_tl
39 \tl_new:N \l__spath_tmpe_tl
40 \tl_new:N \l__spath_tmpe_tl
41
42 \seq_new:N \l__spath_tmpe_seq
43 \seq_new:N \l__spath_tmpe_seq
44 \seq_new:N \l__spath_tmpe_seq
45
46 \dim_new:N \l__spath_tmpe_dim
47 \dim_new:N \l__spath_tmpe_dim
48
49 \fp_new:N \l__spath_tmpe_fp
50 \fp_new:N \l__spath_tmpe_fp
51 \fp_new:N \l__spath_tmpe_fp
52 \fp_new:N \l__spath_tmpe_fp
53 \fp_new:N \l__spath_tmpe_fp
54 \fp_new:N \l__spath_tmpe_fp
55
56 \int_new:N \l__spath_tmpe_int
57 \int_new:N \l__spath_tmpe_int
58
59 \bool_new:N \l__spath_tmpe_bool

```

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

```

60 \dim_new:N \l__spath_move_x_dim
61 \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.

```

62 \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.

```
63 \dim_new:N \l__spath_rectx_dim
64 \dim_new:N \l__spath_recty_dim
65
66 \bool_new:N \l__spath_rect_bool
```

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

```
67 \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.

```
68 \bool_new:N \l_spath_arrow_shortening_bool
69 \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.

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

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

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

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

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

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

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

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

And some error messages

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

3.2 Functional Implementation

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

<code>\spath_segments_to_seq:Nn</code> <code>\spath_segments_gto_seq:Nn</code>	Splits a soft path into <i>segments</i> , storing the result in a sequence.
---	---

```

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

```

```

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

```

```

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

```

```

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

```

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

<pre> \spath_components_to_seq:Nn \spath_components_gto_seq:Nn \spath_components_to_clist:Nn \spath_components_gto_clist:Nn </pre>	<p>Splits a soft path into <i>components</i>, storing the result in a sequence or a clist.</p> <pre> 263 \cs_new_protected_nopar:Npn __spath_components_to_seq:n #1 264 { 265 \group_begin: 266 \tl_set:Nn \l__spath_tmpa_tl {#1} 267 \seq_clear:N \l__spath_tmpa_seq 268 \tl_set:Nx \l__spath_tmpb_tl {\tl_head:N \l__spath_tmpa_tl} 269 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl} 270 271 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_moveto_tl 272 273 \bool_do_until:nn { 274 \tl_if_empty_p:N \l__spath_tmpa_tl 275 } 276 { 277 \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl} 278 \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_moveto_tl 279 { 280 \seq_put_right:NV \l__spath_tmpa_seq \l__spath_tmpb_tl 281 \tl_clear:N \l__spath_tmpb_tl 282 } 283 \tl_if_eq:NNT \l__spath_tmpc_tl \c_spath_rectcorner_tl 284 { 285 \seq_put_right:NV \l__spath_tmpa_seq \l__spath_tmpb_tl 286 \tl_clear:N \l__spath_tmpb_tl 287 } 288 \tl_if_single:NTF \l__spath_tmpc_tl 289 { 290 \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpc_tl 291 } 292 { 293 \tl_put_right:Nx \l__spath_tmpb_tl {\l__spath_tmpc_tl} 294 } 295 \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl} 296 } 297 298 \seq_gclear:N \g__spath_output_seq </pre>
--	---


```

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

```

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

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

```

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

```

(End 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).

```

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

```

```

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

```

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

```

```

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

```

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

\spath_initialpoint:Nn
\spath_ginitialpoint:Nn

The starting point of the path.

```

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

```

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

\spath_finalpoint:Nn
\spath_gfinalpoint:Nn

The final point of the path.

```

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

```

```

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

```

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

`\spath_finalmovepoint:Nn` Get the last move on the path.

`\spath_gfinalmovepoint:Nn` 489 `\cs_new_protected_nopar:Npn __spath_finalmovepoint:n #1`

```

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

```

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

\spath_initialtangent:Nn
\spath_ginitialtangent:Nn

The starting tangent of the path.

```

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

```

596 \cs_generate_variant:Nn \spath_ginitia tangent:Nn {NV, cn, cV, Nv}

(End of definition for \spath_initia tangent:Nn and \spath_ginitia tangent:Nn.)

\spath_finia tangent:Nn
\spath_gfinia tangent:Nn

The final tangent of the path.

```

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

```

```

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

```

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

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

```

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

```



```

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

```

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

\spath_reverse:Nn
\spath_greverse:Nn

This computes the reverse of the path.

```

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

```

```

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

```

```

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

```

```

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

```

```

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

```

(End 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).

```

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

```

```

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

```

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

\spath_finalaction:Nn
\spath_gfinalaction:Nn

This is the last thing that the path does.

```

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

```

```

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

```

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

```

```

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

```

(End 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`

```

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

```



```

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

```

```

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

```

(End 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.

```

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

```

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

```

```

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

```

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

```

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

```

(End 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
\spath_gtranslate_to:Nnnn
\spath_gtranslate_to:Nnn
1281 \cs_new_protected_nopar:Npn \__spath_translate_to:nnn #1#2#3
1282 {
1283   \group_begin:
1284   \spath_initialpoint:Nn \l__spath_tmpa_tl {#1}
1285
1286   \dim_set:Nn \l__spath_tmpa_dim
1287   {
1288     #2
1289     -
1290     \tl_item:Nn \l__spath_tmpa_tl {1}
1291   }
1292   \dim_set:Nn \l__spath_tmpb_dim
1293   {
1294     #3
1295     -
1296     \tl_item:Nn \l__spath_tmpa_tl {2}
1297   }
1298
1299   \__spath_translate:nVV {#1} \l__spath_tmpa_dim \l__spath_tmpb_dim
1300   \group_end:
1301 }

```

```

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

```

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

```

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

```

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

```

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

```

```

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

```

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

```

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

```

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

\spath_transform:Nnnnnnnn
\spath_transform:Nnnnnnnn
\spath_gtransform:Nnnnnnnn
\spath_gtransform:Nnnnnnnn

Applies an affine (matrix and vector) transformation to path. The matrix is specified in rows first.

```

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

```

```

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

```

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


```

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

```

```

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

```

```

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

```

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

```

\spath_splice_between:Nnnn
\spath_splice_between:Nnn
\spath_gsplice_between:Nnnn
\spath_gsplice_between:Nnn

```

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.

```

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

```

(End 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.

```

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

```

First tangent vector projected onto vector between endpoints

Something a bit weird here as the components are opposite to how I thought they should be ...

```

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

```

```

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

```

Second tangent vector projected onto vector between endpoints

```

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

```

```

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

```

```

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

```

```

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

```



```

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

```

(End 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.

```

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

```

```

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

```

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

```

```

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

```

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

```

```

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

```

(End 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`

```

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

```

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

```

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

```

```

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

```

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

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

```

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

```

(End 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 2220 \cs_new_protected_nopar:Npn \spath_prepend_no_move:Nnn #1#2#3
\spath_gprepend_no_move:Nnn 2221 {
\spath_gprepend_no_move:Nn 2222   \spath_append_no_move:Nnn #1{#3}{#2}
2223 }
2224 \cs_generate_variant:Nn \spath_prepend_no_move:Nnn {NVV, NVn}
2225 \cs_new_protected_nopar:Npn \spath_prepend_no_move:Nn #1#2
2226 {
2227   \spath_prepend_no_move:NVn #1#1{#2}
2228 }
2229 \cs_generate_variant:Nn \spath_prepend_no_move:Nn {NV, cv}
2230 \cs_new_protected_nopar:Npn \spath_gprepend_no_move:Nnn #1#2#3
2231 {
2232   \spath_gappend_no_move:Nnn #1{#3}{#2}
2233 }
2234 \cs_generate_variant:Nn \spath_gprepend_no_move:Nnn {NVV, NVn}
2235 \cs_new_protected_nopar:Npn \spath_gprepend_no_move:Nn #1#2
2236 {
2237   \spath_gprepend_no_move:NVn #1#1{#2}
2238 }
2239 \cs_generate_variant:Nn \spath_gprepend_no_move:Nn {NV, cv}

```

(End 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 2240 \cs_new_protected_nopar:Npn \spath_prepend:Nnn #1#2#3
\spath_gprepend:Nnn 2241 {
\spath_gprepend:Nn 2242   \spath_append:Nnn #1{#3}{#2}
2243 }
2244 \cs_generate_variant:Nn \spath_prepend:Nnn {NVV, NVn}
2245 \cs_new_protected_nopar:Npn \spath_prepend:Nn #1#2
2246 {
2247   \spath_prepend:NVn #1#1{#2}
2248 }
2249 \cs_generate_variant:Nn \spath_prepend:Nn {NV}
2250 \cs_new_protected_nopar:Npn \spath_gprepend:Nnn #1#2#3
2251 {
2252   \spath_gappend:Nnn #1{#3}{#2}
2253 }
2254 \cs_generate_variant:Nn \spath_gprepend:Nnn {NVV, NVn}
2255 \cs_new_protected_nopar:Npn \spath_gprepend:Nn #1#2
2256 {
2257   \spath_gprepend:NVn #1#1{#2}
2258 }
2259 \cs_generate_variant:Nn \spath_gprepend:Nn {NV}

```

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

```

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

```



```

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

```

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.

```

2329 \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.

```

2330 \cs_new_nopar:Npn \__spath_arrow_compute_shortening:n #1
2331 {
2332   \__spath_pgf_arrow_compute_shortening:n {#1}
2333   \bool_if:NF \l_spath_arrow_shortening_bool
2334   {
2335     \dim_sub:cn {pgf@xb} {\dim_use:c {pgf@xa}}
2336     \dim_zero:c {pgf@xa}
2337   }
2338 }
2339
2340 \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

```

```

2341 \cs_new_protected_nopar:Npn \__spath_close:n #1
2342 {
2343   \group_begin:
2344   \tl_set:Nn \l__spath_tmpa_tl {#1}
2345   \spath_finalmovepoint:NV \l__spath_tmpb_tl \l__spath_tmpa_tl
2346   \tl_put_right:NV \l__spath_tmpa_tl \c_spath_closepath_tl
2347   \tl_put_right:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
2348   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
2349   \group_end:
2350 }
2351 \cs_generate_variant:Nn \__spath_close:n {V}
2352 \cs_new_protected_nopar:Npn \spath_close:Nn #1#2
2353 {
2354   \__spath_close:n {#2}
2355   \tl_set_eq:NN #1 \g__spath_output_tl
2356   \tl_gclear:N \g__spath_output_tl
2357 }
2358 \cs_generate_variant:Nn \spath_close:Nn {NV}
2359 \cs_new_protected_nopar:Npn \spath_close:N #1
2360 {
2361   \spath_close:NV #1#1
2362 }
2363 \cs_generate_variant:Nn \spath_close:N {c}
2364 \cs_new_protected_nopar:Npn \spath_gclose:Nn #1#2
2365 {
2366   \__spath_close:n {#2}
2367   \tl_gset_eq:NN #1 \g__spath_output_tl
2368   \tl_gclear:N \g__spath_output_tl
2369 }
2370 \cs_generate_variant:Nn \spath_gclose:Nn {NV}
2371 \cs_new_protected_nopar:Npn \spath_gclose:N #1
2372 {
2373   \spath_gclose:NV #1#1
2374 }
2375 \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
2376 \cs_new_protected_nopar:Npn \__spath_adjust_close:n #1
2377 {
2378   \group_begin:
2379   \tl_set:Nn \l__spath_tmpa_tl {#1}
2380   \spath_finalmovepoint:NV \l__spath_tmpb_tl \l__spath_tmpa_tl
2381   \spath_finalpoint:NV \l__spath_tmpc_tl \l__spath_tmpa_tl
2382   \tl_reverse:N \l__spath_tmpa_tl
2383   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2384   \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2385   \tl_set:Nx \l__spath_tmpd_tl {\tl_head:N \l__spath_tmpa_tl}
2386   \tl_if_eq:NNT \l__spath_tmpd_tl \c_spath_curveto_tl
2387   {
2388     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}

```

```

2389 \tl_clear:N \l__spath_tmpe_tl
2390 \tl_set:Nx \l__spath_tmpe_tl {
2391   {
2392     \dim_eval:n
2393     {
2394       \tl_item:Nn \l__spath_tmpa_tl {1}
2395       -
2396       \tl_item:Nn \l__spath_tmpe_tl {2}
2397       +
2398       \tl_item:Nn \l__spath_tmpe_tl {2}
2399     }
2400   }
2401   {
2402     \dim_eval:n
2403     {
2404       \tl_item:Nn \l__spath_tmpe_tl {2}
2405       -
2406       \tl_item:Nn \l__spath_tmpe_tl {1}
2407       +
2408       \tl_item:Nn \l__spath_tmpe_tl {1}
2409     }
2410   }
2411 }
2412 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
2413 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
2414 \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2415 \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2416 }
2417 \tl_reverse:N \l__spath_tmpe_tl
2418 \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2419 \tl_put_right:NV \l__spath_tmpe_tl \c_spath_closepath_tl
2420 \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2421 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
2422 \group_end:
2423 }
2424 \cs_generate_variant:Nn \__spath_adjust_close:n {V}
2425 \cs_new_protected_nopar:Npn \spath_adjust_close:Nn #1#2
2426 {
2427   \__spath_adjust_close:n {#2}
2428   \tl_set_eq:NN #1 \g__spath_output_tl
2429   \tl_gclear:N \g__spath_output_tl
2430 }
2431 \cs_generate_variant:Nn \spath_adjust_close:Nn {NV}
2432 \cs_new_protected_nopar:Npn \spath_adjust_close:N #1
2433 {
2434   \spath_adjust_close:NV #1#1
2435 }
2436 \cs_generate_variant:Nn \spath_adjust_close:N {c}
2437 \cs_new_protected_nopar:Npn \spath_adjust_gclose:Nn #1#2
2438 {
2439   \__spath_adjust_close:n {#2}
2440   \tl_gset_eq:NN #1 \g__spath_output_tl
2441   \tl_gclear:N \g__spath_output_tl
2442 }

```

```

2443 \cs_generate_variant:Nn \spath_adjust_gclose:Nn {NV}
2444 \cs_new_protected_nopar:Npn \spath_adjust_gclose:N #1
2445 {
2446   \spath_adjust_gclose:NV #1#1
2447 }
2448 \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
2449 \cs_new_protected_nopar:Npn \__spath_open:n #1
2450 {
2451   \group_begin:
2452   \spath_replace_rectangles:Nn \l__spath_tmpa_tl {#1}
2453   \tl_clear:N \l__spath_tmpb_tl
2454   \bool_until_do:nn {
2455     \tl_if_empty_p:N \l__spath_tmpa_tl
2456   }
2457   {
2458     \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
2459     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2460
2461     \token_case_meaning:NnF \l__spath_tmpc_tl
2462     {
2463       \c_spath_closepath_tl {
2464
2465         \bool_if:nF
2466         {
2467           \dim_compare_p:n
2468           {
2469             \l__spath_move_x_dim == \l__spath_tmpa_dim
2470           }
2471           &&
2472           \dim_compare_p:n
2473           {
2474             \l__spath_move_y_dim == \l__spath_tmpb_dim
2475           }
2476         }
2477         {
2478           \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
2479
2480           \tl_put_right:Nx \l__spath_tmpb_tl {
2481             { \dim_use:N \l__spath_move_x_dim }
2482             { \dim_use:N \l__spath_move_y_dim }
2483           }
2484         }
2485
2486         \dim_set:Nn \l__spath_tmpa_dim {\tl_head:N \l__spath_tmpa_tl}
2487         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2488         \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpa_tl}
2489         \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
2490       }
2491     }

```

```

2492 \c_spath_moveto_tl {
2493   \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2494
2495   \dim_set:Nn \l__spath_move_x_dim {\tl_head:N \l__spath_tmpe_tl}
2496   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
2497   \dim_set:Nn \l__spath_move_y_dim {\tl_head:N \l__spath_tmpe_tl}
2498   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
2499
2500   \tl_put_right:Nx \l__spath_tmpb_tl {
2501     { \dim_use:N \l__spath_move_x_dim }
2502     { \dim_use:N \l__spath_move_y_dim }
2503   }
2504
2505   \dim_set_eq:NN \l__spath_tmpe_dim \l__spath_move_x_dim
2506   \dim_set_eq:NN \l__spath_tmpb_dim \l__spath_move_y_dim
2507 }
2508 }
2509 {
2510   \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2511
2512   \dim_set:Nn \l__spath_tmpe_dim {\tl_head:N \l__spath_tmpe_tl}
2513   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
2514   \dim_set:Nn \l__spath_tmpb_dim {\tl_head:N \l__spath_tmpe_tl}
2515   \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
2516
2517   \tl_put_right:Nx \l__spath_tmpb_tl {
2518     { \dim_use:N \l__spath_tmpe_dim }
2519     { \dim_use:N \l__spath_tmpb_dim }
2520   }
2521 }
2522 }
2523 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
2524 \group_end:
2525 }
2526 \cs_generate_variant:Nn \__spath_open:n {V}
2527 \cs_new_protected_nopar:Npn \spath_open:Nn #1#2
2528 {
2529   \__spath_open:n {#2}
2530   \tl_set_eq:NN #1 \g__spath_output_tl
2531   \tl_gclear:N \g__spath_output_tl
2532 }
2533 \cs_generate_variant:Nn \spath_open:Nn {NV}
2534 \cs_new_protected_nopar:Npn \spath_open:N #1
2535 {
2536   \spath_open:NV #1#1
2537 }
2538 \cs_new_protected_nopar:Npn \spath_gopen:Nn #1#2
2539 {
2540   \__spath_open:n {#2}
2541   \tl_gset_eq:NN #1 \g__spath_output_tl
2542   \tl_gclear:N \g__spath_output_tl
2543 }
2544 \cs_generate_variant:Nn \spath_gopen:Nn {NV}
2545 \cs_new_protected_nopar:Npn \spath_gopen:N #1

```

```

2546 {
2547   \spath_gopen:NV #1#1
2548 }
2549 \cs_generate_variant:Nn \spath_open:N {c}
2550 \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
\spath_replace_lines:Nn
\spath_replace_lines:Nn
2551 \cs_new_protected_nopar:Npn \__spath_replace_lines:n #1
2552 {
2553   \group_begin:
2554   \tl_set:Nn \l__spath_tmpa_tl {#1}
2555   \tl_clear:N \l__spath_tmpb_tl
2556   \dim_set:Nn \l__spath_tmpa_dim {0pt}
2557   \dim_set:Nn \l__spath_tmpb_dim {0pt}
2558
2559   \bool_do_until:nn
2560   {
2561     \tl_if_empty_p:N \l__spath_tmpa_tl
2562   }
2563   {
2564     \tl_set:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpa_tl {1}}
2565     \tl_set:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpa_tl {2}}
2566     \tl_set:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpa_tl {3}}
2567
2568     \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_lineto_tl
2569     {
2570       \tl_put_right:NV \l__spath_tmpb_tl \c_spath_curvetoa_tl
2571       \tl_put_right:Nx \l__spath_tmpb_tl
2572       {
2573         {
2574           \fp_to_dim:n
2575           {
2576             
$$\frac{2}{3} * (\l__spath_tmpa\_dim)$$

2577             +
2578             
$$\frac{1}{3} * (\l__spath_tmpe\_tl)$$

2579           }
2580         }
2581       }
2582       \tl_put_right:Nx \l__spath_tmpb_tl
2583       {
2584         {
2585           \fp_to_dim:n
2586           {
2587             
$$\frac{2}{3} * (\l__spath_tmpe\_dim)$$

2588             +
2589             
$$\frac{1}{3} * (\l__spath_tmpe\_tl)$$

2590           }
2591         }
2592       }
2593       \tl_put_right:NV \l__spath_tmpb_tl \c_spath_curvetob_tl
2594       \tl_put_right:Nx \l__spath_tmpb_tl
2595       {

```

```

2596     {
2597         \fp_to_dim:n
2598         {
2599             1/3 * (\l__spath_tmpa_dim)
2600             +
2601             2/3 * (\l__spath_tmpe_dim)
2602         }
2603     }
2604 }
2605 \tl_put_right:Nx \l__spath_tmpe_tl
2606 {
2607     {
2608         \fp_to_dim:n
2609         {
2610             1/3 * (\l__spath_tmpe_dim)
2611             +
2612             2/3 * (\l__spath_tmpe_dim)
2613         }
2614     }
2615 }
2616 \tl_put_right:NV \l__spath_tmpe_tl \c_spath_curveto_tl
2617 \__spath_tl_put_right_braced:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2618 \__spath_tl_put_right_braced:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2619 }
2620 {
2621     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2622     \__spath_tl_put_right_braced:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2623     \__spath_tl_put_right_braced:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
2624 }
2625
2626 \dim_set:Nn \l__spath_tmpe_dim {\l__spath_tmpe_dim}
2627 \dim_set:Nn \l__spath_tmpe_dim {\l__spath_tmpe_dim}
2628
2629 \prg_replicate:nn {3}
2630 {
2631     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
2632 }
2633 }
2634 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
2635 \group_end:
2636 }
2637 \cs_generate_variant:Nn \__spath_replace_lines:n {V}
2638 \cs_new_protected_nopar:Npn \spath_replace_lines:Nn #1#2
2639 {
2640     \__spath_replace_lines:n {#2}
2641     \tl_set_eq:NN #1 \g__spath_output_tl
2642     \tl_gclear:N \g__spath_output_tl
2643 }
2644 \cs_generate_variant:Nn \spath_replace_lines:Nn {NV, cV, cv, Nv}
2645 \cs_new_protected_nopar:Npn \spath_replace_lines:N #1
2646 {
2647     \spath_replace_lines:NV #1#1
2648 }
2649 \cs_generate_variant:Nn \spath_replace_lines:N {c}

```

```

2650 \cs_new_protected_nopar:Npn \spath_greplace_lines:Nn #1#2
2651 {
2652   \__spath_replace_lines:n {#2}
2653   \tl_gset_eq:NN #1 \g__spath_output_tl
2654   \tl_gclear:N \g__spath_output_tl
2655 }
2656 \cs_generate_variant:Nn \spath_greplace_lines:Nn {NV, cV, cv, Nv}
2657 \cs_new_protected_nopar:Npn \spath_greplace_lines:N #1
2658 {
2659   \spath_greplace_lines:NV #1#1
2660 }
2661 \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
\spath_replace_rectangles:Nn
\spath_replace_rectangles:Nn
2662 \cs_new_protected_nopar:Npn \__spath_replace_rectangles:n #1
2663 {
2664   \group_begin:
2665   \tl_set:Nn \l__spath_tmpa_tl {#1}
2666   \tl_clear:N \l__spath_tmpb_tl
2667
2668   \bool_do_until:nn
2669   {
2670     \tl_if_empty_p:N \l__spath_tmpa_tl
2671   }
2672   {
2673     \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl }
2674     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2675     \tl_set:Nx \l__spath_tmpd_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_tmpe_tl {\tl_head:N \l__spath_tmpa_tl }
2678     \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2679
2680     \tl_if_eq:NNTF \l__spath_tmpc_tl \c_spath_rectcorner_tl
2681     {
2682
2683       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2684       \dim_set:Nn \l__spath_tmpa_dim
2685       {
2686         \tl_item:Nn \l__spath_tmpa_tl {1}
2687       }
2688       \dim_set:Nn \l__spath_tmpb_dim
2689       {
2690         \tl_item:Nn \l__spath_tmpa_tl {2}
2691       }
2692
2693       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2694       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl }
2695
2696       \tl_put_right:NV \l__spath_tmpb_tl \c_spath_moveto_tl
2697       \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2698       \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2699

```



```

2700 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
2701 \tl_put_right:Nx \l__spath_tmpb_tl
2702 {
2703   {
2704     \fp_to_dim:n { \l__spath_tmpd_tl + \l__spath_tmpa_dim }
2705   }
2706 }
2707 \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2708
2709 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
2710 \tl_put_right:Nx \l__spath_tmpb_tl
2711 {
2712   {
2713     \fp_to_dim:n { \l__spath_tmpd_tl + \l__spath_tmpa_dim }
2714   }
2715 }
2716 \tl_put_right:Nx \l__spath_tmpb_tl
2717 {
2718   {
2719     \fp_to_dim:n { \l__spath_tmpe_tl + \l__spath_tmpb_dim }
2720   }
2721 }
2722
2723 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_lineto_tl
2724 \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2725 \tl_put_right:Nx \l__spath_tmpb_tl
2726 {
2727   {
2728     \fp_to_dim:n { \l__spath_tmpe_tl + \l__spath_tmpb_dim }
2729   }
2730 }
2731
2732 \tl_put_right:NV \l__spath_tmpb_tl \c_spath_closepath_tl
2733 \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2734 \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2735
2736 }
2737 {
2738   \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2739   \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
2740   \__spath_tl_put_right_braced:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
2741 }
2742 }
2743 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
2744 \group_end:
2745 }
2746 \cs_generate_variant:Nn \__spath_replace_rectangles:n {V}
2747 \cs_new_protected_nopar:Npn \spath_replace_rectangles:Nn #1#2
2748 {
2749   \__spath_replace_rectangles:n {#2}
2750   \tl_set_eq:NN #1 \g__spath_output_tl
2751   \tl_gclear:N \g__spath_output_tl
2752 }
2753 \cs_generate_variant:Nn \spath_replace_rectangles:Nn {NV, cV, cv, Nv}

```

```

2754 \cs_new_protected_nopar:Npn \spath_replace_rectangles:N #1
2755 {
2756   \spath_replace_rectangles:NV #1#1
2757 }
2758 \cs_generate_variant:Nn \spath_replace_rectangles:N {c}
2759 \cs_new_protected_nopar:Npn \spath_greplace_rectangles:Nn #1#2
2760 {
2761   \__spath_replace_rectangles:n {#2}
2762   \tl_gset_eq:NN #1 \g__spath_output_tl
2763   \tl_gclear:N \g__spath_output_tl
2764 }
2765 \cs_generate_variant:Nn \spath_greplace_rectangles:Nn {NV, cV, cv, Nv}
2766 \cs_new_protected_nopar:Npn \spath_greplace_rectangles:N #1
2767 {
2768   \spath_greplace_rectangles:NV #1#1
2769 }
2770 \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.

```

2771 \cs_new_protected_nopar:Npn \__spath_remove_empty_components:n #1
2772 {
2773   \group_begin:
2774   \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
2775   \tl_clear:N \l__spath_tmpa_tl
2776   \seq_map_inline:Nn \l__spath_tmpa_seq
2777   {
2778     \int_compare:nF
2779     {
2780       \tl_count:n {##1} == 3
2781     }
2782     {
2783       \tl_put_right:Nn \l__spath_tmpa_tl {##1}
2784     }
2785   }
2786   \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
2787   \group_end:
2788 }
2789 \cs_new_protected_nopar:Npn \spath_remove_empty_components:Nn #1#2
2790 {
2791   \__spath_remove_empty_components:n {#2}
2792   \tl_set_eq:NN #1 \g__spath_output_tl
2793   \tl_gclear:N \g__spath_output_tl
2794 }
2795 \cs_generate_variant:Nn \spath_remove_empty_components:Nn {NV}
2796 \cs_new_protected_nopar:Npn \spath_remove_empty_components:N #1
2797 {
2798   \spath_remove_empty_components:NV #1#1
2799 }
2800 \cs_generate_variant:Nn \spath_remove_empty_components:N {c}
2801 \cs_new_protected_nopar:Npn \spath_gremove_empty_components:Nn #1#2
2802 {
2803   \__spath_remove_empty_components:n {#2}

```

```

2804 \tl_gset_eq:NN #1 \g__spath_output_tl
2805 \tl_gclear:N \g__spath_output_tl
2806 }
2807 \cs_generate_variant:Nn \spath_gremove_empty_components:Nn {NV}
2808 \cs_new_protected_nopar:Npn \spath_gremove_empty_components:N #1
2809 {
2810   \spath_gremove_empty_components:NV #1#1
2811 }
2812 \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.

```

2813 \prg_new_protected_conditional:Npnn \spath_if_eq:nn #1#2 { T, F, TF }
2814 {
2815   \group_begin:
2816   \tl_set:Nn \l__spath_tmpa_tl {#1}
2817   \tl_set:Nn \l__spath_tmpb_tl {#2}
2818   \bool_gset_true:N \g__spath_tmpa_bool
2819   \int_compare:nNnTF
2820   {\tl_count:N \l__spath_tmpa_tl}
2821   =
2822   {\tl_count:N \l__spath_tmpb_tl}
2823   {
2824     \int_step_inline:nnnn {1} {3} {\tl_count:N \l__spath_tmpa_tl}
2825     {
2826       \tl_set:Nx \l__spath_tmpc_tl {\tl_item:Nn \l__spath_tmpa_tl {##1}}
2827       \tl_set:Nx \l__spath_tmpd_tl {\tl_item:Nn \l__spath_tmpb_tl {##1}}
2828       \tl_if_eq:NNF \l__spath_tmpc_tl \l__spath_tmpd_tl
2829       {
2830         \bool_gset_false:N \g__spath_tmpa_bool
2831       }
2832       \dim_set:Nn \l__spath_tmpa_dim {\tl_item:Nn \l__spath_tmpa_tl {##1+1}}
2833       \dim_set:Nn \l__spath_tmpb_dim {\tl_item:Nn \l__spath_tmpb_tl {##1+1}}
2834       \dim_compare:nF
2835       {
2836         \dim_abs:n
2837         {
2838           \l__spath_tmpa_dim - \l__spath_tmpb_dim
2839         }
2840         < 0.001pt
2841       }
2842       {
2843         \bool_gset_false:N \g__spath_tmpa_bool
2844       }
2845       \dim_set:Nn \l__spath_tmpa_dim {\tl_item:Nn \l__spath_tmpa_tl {##1+2}}
2846       \dim_set:Nn \l__spath_tmpb_dim {\tl_item:Nn \l__spath_tmpb_tl {##1+2}}
2847       \dim_compare:nF
2848       {
2849         \dim_abs:n
2850         {
2851           \l__spath_tmpa_dim - \l__spath_tmpb_dim
2852         }
2853         < 0.001pt

```

```

2854     }
2855     {
2856         \bool_gset_false:N \g__spath_tmpa_bool
2857     }
2858 }
2859 }
2860 {
2861     \bool_gset_false:N \g__spath_tmpa_bool
2862 }
2863 \group_end:
2864 \bool_if:NTF \g__spath_tmpa_bool
2865 {
2866     \prg_return_true:
2867 }
2868 {
2869     \prg_return_false:
2870 }
2871 }
2872 \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
2873 \cs_new_protected_nopar:Npn \__spath_split_curve:nn #1#2
2874 {
2875     \group_begin:
2876     \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_moveto_tl
2877     \tl_put_right:Nx \l__spath_tmpa_tl {
2878         {\tl_item:nn {#1} {2}}
2879         {\tl_item:nn {#1} {3}}
2880     }
2881     \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curvetoa_tl
2882     \tl_put_right:Nx \l__spath_tmpa_tl
2883     {
2884         {\fp_to_dim:n
2885         {
2886             (1 - #2) * \tl_item:nn {#1} {2} + (#2) * \tl_item:nn {#1} {5}
2887         }}
2888         {\fp_to_dim:n
2889         {
2890             (1 - #2) * \tl_item:nn {#1} {3} + (#2) * \tl_item:nn {#1} {6}
2891         }}
2892     }
2893
2894     \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curvetob_tl
2895     \tl_put_right:Nx \l__spath_tmpa_tl
2896     {
2897         {\fp_to_dim:n
2898         {
2899             (1 - #2)^2 * \tl_item:nn {#1} {2}
2900             + 2 * (1 - #2) * (#2) * \tl_item:nn {#1} {5}
2901             + (#2)^2 * \tl_item:nn {#1} {8}

```

```

2902     }}
2903     {\fp_to_dim:n
2904     {
2905       (1 - #2)^2 * \tl_item:nn {#1} {3}
2906       + 2 * (1 - #2) * (#2) * \tl_item:nn {#1} {6}
2907       + (#2)^2 * \tl_item:nn {#1} {9}
2908     }}
2909   }
2910
2911   \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curveto_tl
2912   \tl_put_right:Nx \l__spath_tmpa_tl
2913   {
2914     {\fp_to_dim:n
2915     {
2916       (1 - #2)^3 * \tl_item:nn {#1} {2}
2917       + 3 * (1 - #2)^2 * (#2) * \tl_item:nn {#1} {5}
2918       + 3 * (1 - #2) * (#2)^2 * \tl_item:nn {#1} {8}
2919       + (#2)^3 * \tl_item:nn {#1} {11}
2920     }}
2921     {\fp_to_dim:n
2922     {
2923       (1 - #2)^3 * \tl_item:nn {#1} {3}
2924       + 3 * (1 - #2)^2 * (#2) * \tl_item:nn {#1} {6}
2925       + 3 * (1 - #2) * (#2)^2 * \tl_item:nn {#1} {9}
2926       + (#2)^3 * \tl_item:nn {#1} {12}
2927     }}
2928   }
2929
2930   \tl_gclear:N \g__spath_output_tl
2931   \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpa_tl
2932
2933   \tl_clear:N \l__spath_tmpa_tl
2934   \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_moveto_tl
2935   \tl_put_right:Nx \l__spath_tmpa_tl
2936   {
2937     {\fp_to_dim:n
2938     {
2939       (1 - #2)^3 * \tl_item:nn {#1} {2}
2940       + 3 * (1 - #2)^2 * (#2) * \tl_item:nn {#1} {5}
2941       + 3 * (1 - #2) * (#2)^2 * \tl_item:nn {#1} {8}
2942       + (#2)^3 * \tl_item:nn {#1} {11}
2943     }}
2944     {\fp_to_dim:n
2945     {
2946       (1 - #2)^3 * \tl_item:nn {#1} {3}
2947       + 3 * (1 - #2)^2 * (#2) * \tl_item:nn {#1} {6}
2948       + 3 * (1 - #2) * (#2)^2 * \tl_item:nn {#1} {9}
2949       + (#2)^3 * \tl_item:nn {#1} {12}
2950     }}
2951   }
2952
2953   \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curvetoa_tl
2954   \tl_put_right:Nx \l__spath_tmpa_tl
2955   {

```

```

2956 {\fp_to_dim:n
2957 {
2958   (1 - #2)^2 * \tl_item:nn {#1} {5}
2959   + 2 * (1 - #2) * (#2) * \tl_item:nn {#1} {8}
2960   + (#2)^2 * \tl_item:nn {#1} {11}
2961 }}
2962 {\fp_to_dim:n
2963 {
2964   (1 - #2)^2 * \tl_item:nn {#1} {6}
2965   + 2 * (1 - #2) * (#2) * \tl_item:nn {#1} {9}
2966   + (#2)^2 * \tl_item:nn {#1} {12}
2967 }}
2968 }
2969 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curvetob_tl
2970 \tl_put_right:Nx \l__spath_tmpa_tl
2971 {
2972   {\fp_to_dim:n
2973   {
2974     (1 - #2) * \tl_item:nn {#1} {8} + (#2) * \tl_item:nn {#1} {11}
2975   }}
2976   {\fp_to_dim:n
2977   {
2978     (1 - #2) * \tl_item:nn {#1} {9} + (#2) * \tl_item:nn {#1} {12}
2979   }}
2980 }
2981 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_curveto_tl
2982 \tl_put_right:Nx \l__spath_tmpa_tl {
2983   {\tl_item:nn {#1} {11}}
2984   {\tl_item:nn {#1} {12}}
2985 }
2986
2987 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpa_tl
2988 \group_end:
2989 }
2990 \cs_generate_variant:Nn \__spath_split_curve:nn {nv, nV}
2991 \cs_new_protected_nopar:Npn \spath_split_curve:NNnn #1#2#3#4
2992 {
2993   \__spath_split_curve:nn {#3}{#4}
2994   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
2995   \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
2996   \tl_gclear:N \g__spath_output_tl
2997 }
2998 \cs_generate_variant:Nn \spath_split_curve:NNnn {NNnV, NNVn, NNVV}
2999 \cs_new_protected_nopar:Npn \spath_gsplit_curve:NNnn #1#2#3#4
3000 {
3001   \__spath_split_curve:nn {#3}{#4}
3002   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3003   \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3004   \tl_gclear:N \g__spath_output_tl
3005 }
3006 \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:NNn` Possibly splits a bezier curve to ensure that the pieces don't self-intersect. Figuring out
`\spath_maybe_gsplit_curve:NNn`

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!

```

3007 \cs_new_protected_nopar:Npn \__spath_maybe_split_curve:n #1
3008 {
3009   \group_begin:
3010   \fp_set:Nn \l__spath_tmpa_fp
3011   {
3012     (
3013       \tl_item:nn {#1} {3}
3014       - 3 * \tl_item:nn {#1} {6}
3015       + 3 * \tl_item:nn {#1} {9}
3016       - \tl_item:nn {#1} {12}
3017     )
3018     *
3019     (3 * \tl_item:nn {#1} {8} - 3 * \tl_item:nn {#1} {11})
3020     -
3021     (
3022       \tl_item:nn {#1} {2}
3023       - 3 * \tl_item:nn {#1} {5}
3024       + 3 * \tl_item:nn {#1} {8}
3025       - \tl_item:nn {#1} {11}
3026     )
3027     *
3028     (3 * \tl_item:nn {#1} {9} - 3 * \tl_item:nn {#1} {12})
3029   }
3030   \fp_set:Nn \l__spath_tmpb_fp
3031   {
3032     (
3033       \tl_item:nn {#1} {2}
3034       - 3 * \tl_item:nn {#1} {5}
3035       + 3 * \tl_item:nn {#1} {8}
3036       - \tl_item:nn {#1} {11}
3037     )
3038     *
3039     (
3040       3 * \tl_item:nn {#1} {6}
3041       - 6 * \tl_item:nn {#1} {9}
3042       + 3 * \tl_item:nn {#1} {12}
3043     )
3044     -
3045     (
3046       \tl_item:nn {#1} {3}
3047       - 3 * \tl_item:nn {#1} {6}
3048       + 3 * \tl_item:nn {#1} {9}
3049       - \tl_item:nn {#1} {12}
3050     )
3051     *
3052     (
3053       3 * \tl_item:nn {#1} {5}
3054       - 6 * \tl_item:nn {#1} {8}
3055       + 3 * \tl_item:nn {#1} {11}
3056     )
3057   }

```

```

3058 \fp_compare:nTF
3059 {
3060   \l__spath_tmpb_fp != 0
3061 }
3062 {
3063   \fp_set:Nn \l__spath_tmpa_fp {.5 * \l__spath_tmpa_fp / \l__spath_tmpb_fp}
3064   \bool_if:nTF
3065   {
3066     \fp_compare_p:n {0 < \l__spath_tmpa_fp}
3067     &&
3068     \fp_compare_p:n {\l__spath_tmpa_fp < 1}
3069   }
3070   {
3071     \__spath_split_curve:nV {#1} \l__spath_tmpa_fp
3072   }
3073   {
3074     \tl_gset:Nn \g__spath_output_tl { {#1} {} }
3075   }
3076 }
3077 {
3078   \tl_gset:Nn \g__spath_output_tl { {#1} {} }
3079 }
3080 \group_end:
3081 }
3082 \cs_new_protected_nopar:Npn \spath_maybe_split_curve:NNn #1#2#3
3083 {
3084   \__spath_maybe_split_curve:n {#3}
3085   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3086   \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3087   \tl_gclear:N \g__spath_output_tl
3088 }
3089 \cs_generate_variant:Nn \spath_maybe_split_curve:NNn { NNV }
3090 \cs_new_protected_nopar:Npn \spath_maybe_gsplit_curve:NNn #1#2#3
3091 {
3092   \__spath_maybe_split_curve:n {#3}
3093   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3094   \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3095   \tl_gclear:N \g__spath_output_tl
3096 }
3097 \cs_generate_variant:Nn \spath_maybe_gsplit_curve:NNn { NNV }

```

(End of definition for \spath_maybe_split_curve:NNn and \spath_maybe_gsplit_curve:NNn.)

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

```

3098 \cs_new_protected_nopar:Npn \__spath_split_curves:n #1
3099 {
3100   \group_begin:
3101   \tl_set:Nn \l__spath_tmpa_tl {#1}
3102   \tl_clear:N \l__spath_tmpb_tl
3103   \tl_clear:N \l__spath_tmpc_tl
3104   \bool_do_until:nn
3105   {
3106     \tl_if_empty_p:N \l__spath_tmpa_tl
3107   }

```



```

3108 {
3109   \tl_set:Nx \l__spath_tmpc_tl {\tl_head:N \l__spath_tmpa_tl}
3110   \token_case_meaning:NnF \l__spath_tmpc_tl
3111   {
3112     \c_spath_curvetoa_tl
3113     {
3114       \tl_clear:N \l__spath_tmpd_tl
3115       \tl_set_eq:NN \l__spath_tmpd_tl \c_spath_moveto_tl
3116       \tl_put_right:Nx \l__spath_tmpd_tl
3117       {
3118         { \dim_use:N \l__spath_tmpa_dim }
3119         { \dim_use:N \l__spath_tmpb_dim }
3120       }
3121       \dim_set:Nn \l__spath_tmpa_dim
3122       {
3123         \tl_item:Nn \l__spath_tmpa_tl {8}
3124       }
3125       \dim_set:Nn \l__spath_tmpb_dim
3126       {
3127         \tl_item:Nn \l__spath_tmpa_tl {9}
3128       }
3129       \prg_replicate:nn {3}
3130       {
3131         \tl_put_right:Nx \l__spath_tmpd_tl
3132         {
3133           \tl_item:Nn \l__spath_tmpa_tl {1}
3134           {\tl_item:Nn \l__spath_tmpa_tl {2}}
3135           {\tl_item:Nn \l__spath_tmpa_tl {3}}
3136         }
3137         \prg_replicate:nn {3}
3138         {
3139           \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3140         }
3141       }
3142
3143       \spath_maybe_split_curve:NNV
3144       \l__spath_tmpd_tl
3145       \l__spath_tmpe_tl
3146       \l__spath_tmpd_tl
3147       \prg_replicate:nn {3}
3148       {
3149         \tl_set:Nx \l__spath_tmpd_tl {\tl_tail:N \l__spath_tmpd_tl}
3150         \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3151       }
3152       \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpd_tl
3153       \tl_put_right:NV \l__spath_tmpb_tl \l__spath_tmpe_tl
3154     }
3155   }
3156   {
3157     \dim_set:Nn \l__spath_tmpa_dim
3158     {
3159       \tl_item:Nn \l__spath_tmpa_tl {2}
3160     }
3161     \dim_set:Nn \l__spath_tmpb_dim

```

```

3162     {
3163       \tl_item:Nn \l__spath_tmpa_tl {3}
3164     }
3165     \tl_put_right:Nx \l__spath_tmpb_tl
3166     {
3167       \tl_item:Nn \l__spath_tmpa_tl {1}
3168       {\tl_item:Nn \l__spath_tmpa_tl {2}}
3169       {\tl_item:Nn \l__spath_tmpa_tl {3}}
3170     }
3171     \prg_replicate:nn {3}
3172     {
3173       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3174     }
3175   }
3176 }
3177 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpb_tl
3178 \group_end:
3179 }
3180 \cs_new_protected_nopar:Npn \spath_split_curves:Nn #1#2
3181 {
3182   \__spath_split_curves:n {#2}
3183   \tl_set_eq:NN #1 \g__spath_output_tl
3184   \tl_gclear:N \g__spath_output_tl
3185 }
3186 \cs_generate_variant:Nn \spath_split_curves:Nn {NV, cV, cn, cv }
3187 \cs_new_protected_nopar:Npn \spath_split_curves:N #1
3188 {
3189   \spath_split_curves:NV #1#1
3190 }
3191 \cs_generate_variant:Nn \spath_split_curves:N {c}
3192 \cs_new_protected_nopar:Npn \spath_gsplit_curves:Nn #1#2
3193 {
3194   \__spath_split_curves:n {#2}
3195   \tl_gset_eq:NN #1 \g__spath_output_tl
3196   \tl_gclear:N \g__spath_output_tl
3197 }
3198 \cs_generate_variant:Nn \spath_gsplit_curves:Nn {NV, cV, cn, cv }
3199 \cs_new_protected_nopar:Npn \spath_gsplit_curves:N #1
3200 {
3201   \spath_gsplit_curves:NV #1#1
3202 }
3203 \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
3204 \cs_new_protected_nopar:Npn \__spath_split_line:nn #1#2
3205 {
3206   \group_begin:
3207   \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_moveto_tl
3208   \tl_put_right:Nx \l__spath_tmpa_tl {
3209     {\tl_item:nn {#1} {2}}
3210     {\tl_item:nn {#1} {3}}
3211   }

```

```

3212 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_lineto_tl
3213 \tl_put_right:Nx \l__spath_tmpa_tl
3214 {
3215   {\fp_to_dim:n
3216     {
3217       (1 - #2) * \tl_item:nn {#1} {2} + (#2) * \tl_item:nn {#1} {5}
3218     }}
3219   {\fp_to_dim:n
3220     {
3221       (1 - #2) * \tl_item:nn {#1} {3} + (#2) * \tl_item:nn {#1} {6}
3222     }}
3223 }
3224 \tl_gclear:N \g__spath_output_tl
3225 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpa_tl
3226
3227 \tl_clear:N \l__spath_tmpa_tl
3228 \tl_set_eq:NN \l__spath_tmpa_tl \c_spath_moveto_tl
3229 \tl_put_right:Nx \l__spath_tmpa_tl
3230 {
3231   {\fp_to_dim:n
3232     {
3233       (1 - #2) * \tl_item:nn {#1} {2} + (#2) * \tl_item:nn {#1} {5}
3234     }}
3235   {\fp_to_dim:n
3236     {
3237       (1 - #2) * \tl_item:nn {#1} {3} + (#2) * \tl_item:nn {#1} {6}
3238     }}
3239 }
3240 \tl_put_right:NV \l__spath_tmpa_tl \c_spath_lineto_tl
3241 \tl_put_right:Nx \l__spath_tmpa_tl {
3242   {\tl_item:nn {#1} {5}}
3243   {\tl_item:nn {#1} {6}}
3244 }
3245
3246 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpa_tl
3247 \group_end:
3248 }
3249 \cs_new_protected_nopar:Npn \spath_split_line:NNnn #1#2#3#4
3250 {
3251   \__spath_split_line:nn {#3}{#4}
3252   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3253   \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3254   \tl_gclear:N \g__spath_output_tl
3255 }
3256 \cs_generate_variant:Nn \spath_split_line:NNnn {NNnV, NNVn, NNVV}
3257 \cs_new_protected_nopar:Npn \spath_gsplit_line:NNnn #1#2#3#4
3258 {
3259   \__spath_split_line:nn {#3}{#4}
3260   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3261   \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3262   \tl_gclear:N \g__spath_output_tl
3263 }
3264 \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

3265 \cs_new_protected_nopar:Npn \__spath_split_rectangle:nn #1#2
3266 {
3267   \group_begin:
3268   \spath_open:Nn \l__spath_tmpa_tl {#1}
3269   \fp_set:Nn \l__spath_tmpa_fp {4*{#2}}
3270   \spath_split_at:NNVV
3271   \l__spath_tmpa_tl \l__spath_tmpb_tl \l__spath_tmpa_tl \l__spath_tmpa_fp
3272   \__spath_append_no_move:VV \l__spath_tmpb_tl \l__spath_tmpa_tl
3273   \group_end:
3274 }
3275 \cs_new_protected_nopar:Npn \spath_split_rectangle:Nnn #1#2#3
3276 {
3277   \__spath_split_rectangle:nn {#2}{#3}
3278   \tl_set_eq:NN #1 \g__spath_output_tl
3279   \tl_gclear:N \g__spath_output_tl
3280 }
3281 \cs_generate_variant:Nn \spath_split_rectangle:Nnn {NnV, NVn, NVV}
3282 \cs_new_protected_nopar:Npn \spath_gsplit_rectangle:Nnn #1#2#3
3283 {
3284   \__spath_split_rectangle:nn {#2}{#3}
3285   \tl_gset_eq:NN #1 \g__spath_output_tl
3286   \tl_gclear:N \g__spath_output_tl
3287 }
3288 \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
\spath_split_at_normalised_keep_middle:Nnn
\spath_split_at_normalised_keep_middle:Nn
\spath_gsplit_at_normalised_keep_middle:Nnn
\spath_gsplit_at_normalised_keep_middle: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.

```

3289 \cs_new_protected_nopar:Npn \__spath_split_at:nn #1#2
3290 {
3291   \group_begin:
3292   \int_set:Nn \l__spath_tmpa_int {\fp_to_int:n {floor{#2} + 1}}
3293   \fp_set:Nn \l__spath_tmpa_fp {#2 - floor{#2}}
3294
3295   % Is split point near one end or other of a component?
3296   \fp_compare:nT
3297   {
3298     \l__spath_tmpa_fp < 0.01
3299   }
3300   {
3301     % Near the start, so we'll place it at the start
3302     \fp_set:Nn \l__spath_tmpa_fp {0}
3303   }
3304   \fp_compare:nT
3305   {
3306     \l__spath_tmpa_fp > 0.99
3307   }
3308   {
3309     % Near the end, so we'll place it at the end
3310     \fp_set:Nn \l__spath_tmpa_fp {0}

```

```

3311     \int_incr:N \l__spath_tmpe_int
3312 }
3313
3314 \int_zero:N \l__spath_tmpe_int
3315 \bool_set_true:N \l__spath_tmpe_bool
3316
3317 \tl_set:Nn \l__spath_tmpe_tl {#1}
3318
3319 \dim_zero:N \l__spath_tmpe_dim
3320 \dim_zero:N \l__spath_tmpe_dim
3321
3322 % Remember if the component is closed
3323 \spath_finalaction:N \l__spath_tmpe_tl \l__spath_tmpe_tl
3324
3325 \bool_set:Nn \l__spath_closed_bool
3326 {
3327     \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_closepath_tl
3328     ||
3329     \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_rectcorner_tl
3330 }
3331
3332 % Open it
3333 \spath_open:N \l__spath_tmpe_tl
3334
3335 \tl_clear:N \l__spath_tmpe_tl
3336 \tl_clear:N \l__spath_tmpe_tl
3337 \tl_clear:N \l__spath_tmpe_tl
3338 \tl_clear:N \l__spath_tmpe_tl
3339
3340 \bool_until_do:nn {
3341     \tl_if_empty_p:N \l__spath_tmpe_tl
3342     ||
3343     \int_compare_p:n { \l__spath_tmpe_int == \l__spath_tmpe_int }
3344 }
3345 {
3346     \tl_set:Nx \l__spath_tmpe_tl {\tl_head:N \l__spath_tmpe_tl}
3347     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3348     \token_case_meaning:Nn \l__spath_tmpe_tl
3349     {
3350         \c_spath_lineto_tl
3351         {
3352             \int_incr:N \l__spath_tmpe_int
3353         }
3354         \c_spath_curvetoa_tl
3355         {
3356             \int_incr:N \l__spath_tmpe_int
3357         }
3358         \c_spath_rectcorner_tl
3359         {
3360             \int_incr:N \l__spath_tmpe_int
3361         }
3362     }
3363     \int_compare:nT { \l__spath_tmpe_int < \l__spath_tmpe_int }
3364     {

```

```

3365 \tl_put_right:NV \l__spath_tmpc_tl \l__spath_tmpf_tl
3366
3367 \tl_put_right:Nx \l__spath_tmpc_tl
3368 {{ \tl_head:N \l__spath_tmpe_tl }}
3369 \dim_set:Nn \l__spath_tmpe_dim {\tl_head:N \l__spath_tmpe_tl}
3370 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3371
3372 \tl_put_right:Nx \l__spath_tmpc_tl
3373 {{ \tl_head:N \l__spath_tmpe_tl }}
3374 \dim_set:Nn \l__spath_tmpe_dim {\tl_head:N \l__spath_tmpe_tl}
3375 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3376
3377 }
3378 }
3379
3380 \tl_clear:N \l__spath_tmpd_tl
3381 \tl_put_right:NV \l__spath_tmpd_tl \c_spath_moveto_tl
3382 \tl_put_right:Nx \l__spath_tmpd_tl
3383 {
3384   {\dim_use:N \l__spath_tmpe_dim}
3385   {\dim_use:N \l__spath_tmpe_dim}
3386 }
3387
3388 \fp_compare:nTF
3389 {
3390   \l__spath_tmpe_fp == 0
3391 }
3392 {
3393   \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpd_tl
3394   \tl_if_empty:NF \l__spath_tmpe_tl
3395   {
3396     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpf_tl
3397     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3398   }
3399 }
3400 {
3401
3402   \token_case_meaning:Nn \l__spath_tmpf_tl
3403   {
3404     \c_spath_lineto_tl
3405     {
3406       \tl_put_right:NV \l__spath_tmpd_tl \l__spath_tmpf_tl
3407       \tl_put_right:Nx \l__spath_tmpd_tl
3408       {{ \tl_head:N \l__spath_tmpe_tl }}
3409       \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3410
3411       \tl_put_right:Nx \l__spath_tmpd_tl
3412       {{ \tl_head:N \l__spath_tmpe_tl }}
3413       \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3414
3415       \spath_split_line:NNVV
3416       \l__spath_tmpe_tl
3417       \l__spath_tmpe_tl
3418       \l__spath_tmpe_tl

```

```

3419     \l__spath_tmpe_tl
3420
3421     \prg_replicate:nn {3} {
3422         \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3423     }
3424
3425     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3426     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3427 }
3428 \c_spath_curvetoa_tl
3429 {
3430     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3431     \tl_put_right:Nx \l__spath_tmpe_tl
3432     {{ \tl_head:N \l__spath_tmpe_tl }}
3433     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3434
3435     \tl_put_right:Nx \l__spath_tmpe_tl
3436     {{ \tl_head:N \l__spath_tmpe_tl }}
3437     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3438
3439     \prg_replicate:nn {2} {
3440
3441         \tl_put_right:Nx \l__spath_tmpe_tl
3442         { \tl_head:N \l__spath_tmpe_tl }
3443         \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3444
3445         \tl_put_right:Nx \l__spath_tmpe_tl
3446         {{ \tl_head:N \l__spath_tmpe_tl }}
3447         \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3448
3449         \tl_put_right:Nx \l__spath_tmpe_tl
3450         {{ \tl_head:N \l__spath_tmpe_tl }}
3451         \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3452     }
3453
3454     \spath_split_curve:NNVV
3455     \l__spath_tmpe_tl
3456     \l__spath_tmpe_tl
3457     \l__spath_tmpe_tl \l__spath_tmpe_tl
3458
3459     \prg_replicate:nn {3} {
3460         \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3461     }
3462
3463     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3464     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3465 }
3466
3467 \c_spath_rectcorner_tl
3468 {
3469     \tl_clear:N \l__spath_tmpe_tl
3470     \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3471
3472     \tl_put_right:Nx \l__spath_tmpe_tl {{\tl_head:N \l__spath_tmpe_tl}}

```

```

3473 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
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
3477 \tl_put_right:Nx \l__spath_tmpd_tl {\tl_head:N \l__spath_tmpe_tl}
3478 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3479
3480 \tl_put_right:Nx \l__spath_tmpd_tl {\tl_head:N \l__spath_tmpe_tl}}
3481 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
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
3485 \spath_split_rectangle:NVV
3486 \l__spath_tmpe_tl
3487 \l__spath_tmpd_tl
3488 \l__spath_tmpe_tl
3489
3490 \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3491 \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3492 }
3493
3494 }
3495 }
3496
3497 \bool_if:NT \l__spath_closed_bool
3498 {
3499 \prg_replicate:nn {3}
3500 {
3501 \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
3502 }
3503 \tl_put_right:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
3504 \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpe_tl
3505 \tl_clear:N \l__spath_tmpe_tl
3506 }
3507
3508 \tl_gclear:N \g__spath_output_tl
3509 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpe_tl
3510 \__spath_tl_gput_right_braced:NV \g__spath_output_tl \l__spath_tmpe_tl
3511 \group_end:
3512 }
3513 \cs_generate_variant:Nn \__spath_split_at:nn {nV, VV}
3514 \cs_new_protected_nopar:Npn \__spath_split_at_normalised:nn #1#2
3515 {
3516 \group_begin:
3517 \spath_reallength:Nn \l__spath_tmpe_int {#1}
3518
3519 \tl_set:Nx \l__spath_tmpe_tl
3520 {\fp_to_decimal:n {(#2) * (\l__spath_tmpe_int)}}
3521 \__spath_split_at:nV {#1} \l__spath_tmpe_tl
3522 \group_end:
3523 }
3524 \cs_generate_variant:Nn \__spath_split_at_normalised:nn {nV}
3525 \cs_new_protected_nopar:Npn \spath_split_at:NNnn #1#2#3#4
3526 {

```



```

3527 \__spath_split_at:nn {#3}{#4}
3528 \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3529 \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3530 \tl_gclear:N \g__spath_output_tl
3531 }
3532 \cs_generate_variant:Nn \spath_split_at:NNnn {NNVn, NNVV, NNnV}
3533 \cs_new_protected_nopar:Npn \spath_gsplit_at:NNnn #1#2#3#4
3534 {
3535   \__spath_split_at:nn {#3}{#4}
3536   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3537   \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3538   \tl_gclear:N \g__spath_output_tl
3539 }
3540 \cs_generate_variant:Nn \spath_gsplit_at:NNnn {NNVn, NNVV, NNnV}
3541 \cs_new_protected_nopar:Npn \spath_split_at_keep_start:Nnn #1#2#3
3542 {
3543   \__spath_split_at:nn {#2}{#3}
3544   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3545   \tl_gclear:N \g__spath_output_tl
3546 }
3547 \cs_generate_variant:Nn \spath_split_at_keep_start:Nnn {NVn}
3548 \cs_new_protected_nopar:Npn \spath_split_at_keep_start:Nn #1#2
3549 {
3550   \spath_split_at_keep_start:NVn #1#1{#2}
3551 }
3552 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_start:Nnn #1#2#3
3553 {
3554   \__spath_split_at:nn {#2}{#3}
3555   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3556   \tl_gclear:N \g__spath_output_tl
3557 }
3558 \cs_generate_variant:Nn \spath_gsplit_at_keep_start:Nnn {NVn}
3559 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_start:Nn #1#2
3560 {
3561   \spath_gsplit_at_keep_start:NVn #1#1{#2}
3562 }
3563 \cs_new_protected_nopar:Npn \spath_split_at_keep_end:Nnn #1#2#3
3564 {
3565   \__spath_split_at:nn {#2}{#3}
3566   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3567   \tl_gclear:N \g__spath_output_tl
3568 }
3569 \cs_generate_variant:Nn \spath_split_at_keep_end:Nnn {NVn}
3570 \cs_new_protected_nopar:Npn \spath_split_at_keep_end:Nn #1#2
3571 {
3572   \spath_split_at_keep_end:NVn #1#1{#2}
3573 }
3574 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_end:Nnn #1#2#3
3575 {
3576   \__spath_split_at:nn {#2}{#3}
3577   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3578   \tl_gclear:N \g__spath_output_tl
3579 }
3580 \cs_generate_variant:Nn \spath_gsplit_at_keep_end:Nnn {NVn}

```

```

3581 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_end:Nn #1#2
3582 {
3583   \spath_gsplit_at_keep_end:NVn #1#1{#2}
3584 }
3585 \cs_new_protected_nopar:Npn \__spath_split_at_twice:nnn #1#2#3
3586 {
3587   \group_begin:
3588   \__spath_split_at:nn {#1}{#3}
3589
3590   \int_set:Nn \l__spath_tmpa_int {\fp_to_int:n {floor(#2)}}
3591   \fp_set:Nn \l__spath_tmpa_fp {#2 - floor(#2)}
3592   \int_set:Nn \l__spath_tmpb_int {\fp_to_int:n {floor(#3)}}
3593   \fp_set:Nn \l__spath_tmpb_fp {#3 - floor(#3)}
3594   \int_compare:nNnTF {\l__spath_tmpa_int} = {\l__spath_tmpb_int}
3595   {
3596     \fp_set:Nn \l__spath_tmpb_fp
3597     {
3598       \l__spath_tmpa_int + \l__spath_tmpa_fp / \l__spath_tmpb_fp
3599     }
3600   }
3601   {
3602     \fp_set:Nn \l__spath_tmpb_fp {#2}
3603   }
3604
3605   \tl_set:Nx \l__spath_tmpa_tl {\tl_item:Nn \g__spath_output_tl {1}}
3606   \tl_set:Nx \l__spath_tmpc_tl {\tl_item:Nn \g__spath_output_tl {2}}
3607   \tl_set:Nx \l__spath_tmpb_tl {\fp_to_decimal:N \l__spath_tmpb_fp }
3608   \__spath_split_at:VV \l__spath_tmpa_tl \l__spath_tmpb_tl
3609   \tl_gput_right:NV \g__spath_output_tl \l__spath_tmpc_tl
3610   \group_end:
3611 }
3612 \cs_generate_variant:Nn \__spath_split_at_twice:nnn {nVV}
3613 \cs_new_protected_nopar:Npn \spath_split_at_keep_middle:Nnnn #1#2#3#4
3614 {
3615   \__spath_split_at_twice:nnn {#2}{#3}{#4}
3616   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3617   \tl_gclear:N \g__spath_output_tl
3618 }
3619 \cs_generate_variant:Nn \spath_split_at_keep_middle:Nnnn {NVnn,NnVV}
3620 \cs_new_protected_nopar:Npn \spath_split_at_keep_middle:Nnn #1#2#3
3621 {
3622   \spath_split_at_keep_middle:NVnn #1#1{#2}{#3}
3623 }
3624 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_middle:Nnnn #1#2#3#4
3625 {
3626   \spath_split_at_keep_middle:Nnnn #1{#2}{#3}{#4}
3627   \tl_gset_eq:NN #1#1
3628 }
3629 \cs_generate_variant:Nn \spath_gsplit_at_keep_middle:Nnnn {NVnn}
3630 \cs_new_protected_nopar:Npn \spath_gsplit_at_keep_middle:Nnn #1#2#3
3631 {
3632   \spath_gsplit_at_keep_middle:NVnn #1#1{#2}{#3}
3633 }
3634 \cs_new_protected_nopar:Npn \spath_split_at_normalised:NNnn #1#2#3#4

```

```

3635 {
3636   \__spath_split_at_normalised:nn {#3}{#4}
3637   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3638   \tl_set:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3639   \tl_gclear:N \g__spath_output_tl
3640 }
3641 \cs_generate_variant:Nn \spath_split_at_normalised:NNnn {NNVn, NNVV, NNnV, ccvn}
3642 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised:NNnn #1#2#3#4
3643 {
3644   \__spath_split_at_normalised:nn {#3}{#4}
3645   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3646   \tl_gset:Nx #2 {\tl_item:Nn \g__spath_output_tl {2}}
3647   \tl_gclear:N \g__spath_output_tl
3648 }
3649 \cs_generate_variant:Nn \spath_gsplit_at_normalised:NNnn {NNVn, NNVV, NNnV, ccvn}
3650 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_start:Nnn #1#2#3
3651 {
3652   \__spath_split_at_normalised:nn {#2}{#3}
3653   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3654   \tl_gclear:N \g__spath_output_tl
3655 }
3656 \cs_generate_variant:Nn \spath_split_at_normalised_keep_start:Nnn {NVn}
3657 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_start:Nn #1#2
3658 {
3659   \spath_split_at_normalised_keep_start:NVn #1#1{#2}
3660 }
3661 \cs_generate_variant:Nn \spath_split_at_normalised_keep_start:Nn {cn}
3662 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_start:Nnn #1#2#3
3663 {
3664   \__spath_split_at_normalised:nn {#2}{#3}
3665   \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {1}}
3666   \tl_gclear:N \g__spath_output_tl
3667 }
3668 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_start:Nnn {NVn}
3669 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_start:Nn #1#2
3670 {
3671   \spath_gsplit_at_normalised_keep_start:NVn #1#1{#2}
3672 }
3673 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_start:Nn {cn}
3674 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_end:Nnn #1#2#3
3675 {
3676   \__spath_split_at_normalised:nn {#2}{#3}
3677   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3678   \tl_gclear:N \g__spath_output_tl
3679 }
3680 \cs_generate_variant:Nn \spath_split_at_normalised_keep_end:Nnn {NVn}
3681 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_end:Nn #1#2
3682 {
3683   \spath_split_at_normalised_keep_end:NVn #1#1{#2}
3684 }
3685 \cs_generate_variant:Nn \spath_split_at_normalised_keep_end:Nn {cn}
3686 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_end:Nnn #1#2#3
3687 {
3688   \__spath_split_at_normalised:nn {#2}{#3}

```

```

3689 \tl_gset:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3690 \tl_gclear:N \g__spath_output_tl
3691 }
3692 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_end:Nnn {NVn}
3693 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_end:Nn #1#2
3694 {
3695   \spath_gsplit_at_normalised_keep_end:NVn #1#1{#2}
3696 }
3697 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_end:Nn {cn}
3698 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_middle:Nnnn #1#2#3#4
3699 {
3700   \group_begin:
3701   \spath_reallength:Nn \l__spath_tmpa_int {#2}
3702
3703   \tl_set:Nx \l__spath_tmpa_tl
3704   {\fp_to_decimal:n {(#3) * (\l__spath_tmpa_int)}}
3705   \tl_set:Nx \l__spath_tmpb_tl
3706   {\fp_to_decimal:n {(#4) * (\l__spath_tmpa_int)}}
3707   \__spath_split_at_twice:nVV {#2} \l__spath_tmpa_tl \l__spath_tmpb_tl
3708   \group_end:
3709
3710   \tl_set:Nx #1 {\tl_item:Nn \g__spath_output_tl {2}}
3711   \tl_gclear:N \g__spath_output_tl
3712 }
3713 \cs_generate_variant:Nn \spath_split_at_normalised_keep_middle:Nnnn {NVnn,NnVV}
3714 \cs_new_protected_nopar:Npn \spath_split_at_normalised_keep_middle:Nnn #1#2#3
3715 {
3716   \spath_split_at_normalised_keep_middle:NVnn #1#1{#2}{#3}
3717 }
3718 \cs_generate_variant:Nn \spath_split_at_normalised_keep_middle:Nnn {cnn}
3719 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_middle:Nnnn #1#2#3#4
3720 {
3721   \spath_split_at_normalised_keep_middle:Nnnn #1{#2}{#3}{#4}
3722   \tl_gset_eq:NN #1#1
3723 }
3724 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_middle:Nnnn {NVnn}
3725 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised_keep_middle:Nnn #1#2#3
3726 {
3727   \spath_gsplit_at_normalised_keep_middle:NVnn #1#1{#2}{#3}
3728 }
3729 \cs_generate_variant:Nn \spath_gsplit_at_normalised_keep_middle:Nnn {cnn}
3730 \cs_new_protected_nopar:Npn \spath_split_at:Nnn #1#2#3
3731 {
3732   \__spath_split_at:nn {#2}{#3}
3733   \tl_set:Nx #1
3734   {
3735     \tl_item:Nn \g__spath_output_tl {1}
3736     \tl_item:Nn \g__spath_output_tl {2}
3737   }
3738   \tl_gclear:N \g__spath_output_tl
3739 }
3740 \cs_generate_variant:Nn \spath_split_at:Nnn {NVn, NVV}
3741 \cs_new_protected_nopar:Npn \spath_split_at:Nn #1#2
3742 {

```

```

3743 \spath_split_at:NVn #1#1{#2}
3744 }
3745 \cs_new_protected_nopar:Npn \spath_gsplit_at:Nnn #1#2#3
3746 {
3747   \__spath_split_at:nn {#2}{#3}
3748   \tl_gset:Nx #1
3749   {
3750     \tl_item:Nn \g__spath_output_tl {1}
3751     \tl_item:Nn \g__spath_output_tl {2}
3752   }
3753   \tl_gclear:N \g__spath_output_tl
3754 }
3755 \cs_generate_variant:Nn \spath_gsplit_at:Nnn {NVn, NVV}
3756 \cs_new_protected_nopar:Npn \spath_gsplit_at:Nn #1#2
3757 {
3758   \spath_gsplit_at:NVn #1#1{#2}
3759 }
3760 \cs_new_protected_nopar:Npn \spath_split_at_normalised:Nnn #1#2#3
3761 {
3762   \__spath_split_at_normalised:nn {#2}{#3}
3763   \tl_set:Nx #1
3764   {
3765     \tl_item:Nn \g__spath_output_tl {1}
3766     \tl_item:Nn \g__spath_output_tl {2}
3767   }
3768   \tl_gclear:N \g__spath_output_tl
3769 }
3770 \cs_generate_variant:Nn \spath_split_at_normalised:Nnn {NVn, NVV}
3771 \cs_new_protected_nopar:Npn \spath_split_at_normalised:Nn #1#2
3772 {
3773   \spath_split_at_normalised:NVn #1#1{#2}
3774 }
3775 \cs_generate_variant:Nn \spath_split_at_normalised:Nn {cn}
3776 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised:Nnn #1#2#3
3777 {
3778   \__spath_split_at_normalised:nn {#2}{#3}
3779   \tl_gset:Nx #1
3780   {
3781     \tl_item:Nn \g__spath_output_tl {1}
3782     \tl_item:Nn \g__spath_output_tl {2}
3783   }
3784   \tl_gclear:N \g__spath_output_tl
3785 }
3786 \cs_generate_variant:Nn \spath_gsplit_at_normalised:Nnn {NVn, NVV}
3787 \cs_new_protected_nopar:Npn \spath_gsplit_at_normalised:Nn #1#2
3788 {
3789   \spath_gsplit_at_normalised:NVn #1#1{#2}
3790 }
3791 \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.
3792 \cs_new_protected_nopar:Npn \__spath_shorten_at_end:nn #1#2
3793 {
3794   \int_compare:nTF
3795   {
3796     \tl_count:n {#1} > 3
3797   }
3798   {
3799     \group_begin:
3800     \tl_set:Nn \l__spath_tmpa_tl {#1}
3801     \tl_reverse:N \l__spath_tmpa_tl
3802
3803     \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {3}}
3804
3805     \tl_clear:N \l__spath_tmpe_tl
3806     \tl_if_eq:NNTF \l__spath_tmpb_tl \c_spath_curveto_tl
3807     {
3808       \int_set:Nn \l__spath_tmpa_int {3}
3809     }
3810     {
3811       \int_set:Nn \l__spath_tmpa_int {1}
3812     }
3813
3814     \prg_replicate:nn { \l__spath_tmpa_int }
3815     {
3816       \tl_put_right:Nx \l__spath_tmpe_tl
3817       {
3818         {\tl_head:N \l__spath_tmpa_tl}
3819       }
3820       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3821       \tl_put_right:Nx \l__spath_tmpe_tl
3822       {
3823         {\tl_head:N \l__spath_tmpa_tl}
3824       }
3825       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3826       \tl_put_right:Nx \l__spath_tmpe_tl
3827       {
3828         \tl_head:N \l__spath_tmpa_tl
3829       }
3830       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
3831     }
3832
3833     \tl_put_right:Nx \l__spath_tmpe_tl
3834     {
3835       {\tl_item:Nn \l__spath_tmpa_tl {1}}
3836       {\tl_item:Nn \l__spath_tmpa_tl {2}}
```

```

3837 }
3838 \tl_put_right:NV \l__spath_tmpe_tl \c_spath_moveto_tl
3839
3840 \tl_reverse:N \l__spath_tmpe_tl
3841
3842 \fp_set:Nn \l__spath_tmpe_fp
3843 {
3844   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {4}}
3845   -
3846   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {1}}
3847 }
3848
3849 \fp_set:Nn \l__spath_tmpe_fp
3850 {
3851   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {5}}
3852   -
3853   \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {2}}
3854 }
3855
3856 \fp_set:Nn \l__spath_tmpe_fp
3857 {
3858   sqrt(
3859     \l__spath_tmpe_fp * \l__spath_tmpe_fp
3860     +
3861     \l__spath_tmpe_fp * \l__spath_tmpe_fp
3862   ) * \l__spath_tmpe_int
3863 }
3864
3865 \fp_compare:nTF
3866 {
3867   \l__spath_tmpe_fp > #2
3868 }
3869 {
3870
3871   \fp_set:Nn \l__spath_tmpe_fp
3872   {
3873     (\l__spath_tmpe_fp - #2) / \l__spath_tmpe_fp
3874   }
3875
3876   \tl_reverse:N \l__spath_tmpe_tl
3877
3878   \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_curveto_tl
3879   {
3880     \spath_split_curve:NNVV
3881     \l__spath_tmpe_tl
3882     \l__spath_tmpe_tl
3883     \l__spath_tmpe_tl
3884     \l__spath_tmpe_fp
3885   }
3886   {
3887     \spath_split_line:NNVV
3888     \l__spath_tmpe_tl
3889     \l__spath_tmpe_tl
3890     \l__spath_tmpe_tl

```

```

3891     \l__spath_tmpc_fp
3892 }
3893
3894 \prg_replicate:nn {3}
3895 {
3896     \tl_set:Nx \l__spath_tmpc_tl {\tl_tail:N \l__spath_tmpc_tl}
3897 }
3898
3899 \tl_put_right:NV \l__spath_tmpa_tl \l__spath_tmpc_tl
3900
3901 }
3902 {
3903
3904     \int_compare:nT
3905     {
3906         \tl_count:N \l__spath_tmpa_tl > 3
3907     }
3908     {
3909         \dim_set:Nn \l__spath_tmpa_dim {\fp_to_dim:n {#2 - \l__spath_tmpc_fp } }
3910         \spath_shorten_at_end:NV \l__spath_tmpa_tl \l__spath_tmpa_dim
3911     }
3912 }
3913
3914 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
3915 \group_end:
3916 }
3917 {
3918     \tl_gset:Nn \g__spath_output_tl {#1}
3919 }
3920 }
3921 \cs_new_protected_nopar:Npn \spath_shorten_at_end:Nnn #1#2#3
3922 {
3923     \__spath_shorten_at_end:nn {#2}{#3}
3924     \tl_set_eq:NN #1 \g__spath_output_tl
3925     \tl_gclear:N \g__spath_output_tl
3926 }
3927 \cs_generate_variant:Nn \spath_shorten_at_end:Nnn {NVV, cnn, cVV, NVn}
3928 \cs_new_protected_nopar:Npn \spath_shorten_at_end:Nn #1#2
3929 {
3930     \spath_shorten_at_end:NVn #1#1{#2}
3931 }
3932 \cs_generate_variant:Nn \spath_shorten_at_end:Nn {cn, cV, NV}
3933 \cs_new_protected_nopar:Npn \spath_gshorten_at_end:Nnn #1#2#3
3934 {
3935     \__spath_shorten_at_end:nn {#2}{#3}
3936     \tl_gset_eq:NN #1 \g__spath_output_tl
3937     \tl_gclear:N \g__spath_output_tl
3938 }
3939 \cs_generate_variant:Nn \spath_gshorten_at_end:Nnn {NVV, cnn, cVV, NVn}
3940 \cs_new_protected_nopar:Npn \spath_gshorten_at_end:Nn #1#2
3941 {
3942     \spath_gshorten_at_end:NVn #1#1{#2}
3943 }
3944 \cs_generate_variant:Nn \spath_gshorten_at_end:Nn {cn, cV, NV}

```


(End of definition for \spath_shorten_at_end:Nnn.)

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

```

3945 \cs_new_protected_nopar:Npn \__spath_shorten_at_start:nn #1#2
3946 {
3947   \int_compare:nTF
3948   {
3949     \tl_count:n {#1} > 3
3950   }
3951   {
3952     \group_begin:
3953     \tl_set:Nn \l__spath_tmpa_tl {#1}
3954
3955     \tl_set:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpa_tl {4}}
3956
3957     \tl_clear:N \l__spath_tmpe_tl
3958
3959     \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_curvetoa_tl
3960     {
3961       \int_set:Nn \l__spath_tmpe_int {3}
3962     }
3963     {
3964       \int_set:Nn \l__spath_tmpe_int {1}
3965     }
3966
3967     \tl_set_eq:NN \l__spath_tmpe_tl \c_spath_moveto_tl
3968     \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3969
3970     \prg_replicate:nn { \l__spath_tmpe_int }
3971     {
3972       \__spath_tl_put_right_braced:Nx
3973       \l__spath_tmpe_tl
3974       {\tl_item:Nn \l__spath_tmpe_tl {1}}
3975       \__spath_tl_put_right_braced:Nx
3976       \l__spath_tmpe_tl
3977       {\tl_item:Nn \l__spath_tmpe_tl {2}}
3978       \tl_put_right:Nx \l__spath_tmpe_tl {\tl_item:Nn \l__spath_tmpe_tl {3}}
3979
3980       \prg_replicate:nn {3}
3981       {
3982         \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl }
3983       }
3984     }
3985     \__spath_tl_put_right_braced:Nx
3986     \l__spath_tmpe_tl
3987     {\tl_item:Nn \l__spath_tmpe_tl {1}}
3988     \__spath_tl_put_right_braced:Nx
3989     \l__spath_tmpe_tl
3990     {\tl_item:Nn \l__spath_tmpe_tl {2}}
3991
3992     \fp_set:Nn \l__spath_tmpe_fp
3993     {
3994       \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {5}}
3995       -

```

```

3996     \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {2}}
3997 }
3998
3999 \fp_set:Nn \l__spath_tmpb_fp
4000 {
4001     \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {6}}
4002     -
4003     \dim_to_fp:n {\tl_item:Nn \l__spath_tmpe_tl {3}}
4004 }
4005
4006 \fp_set:Nn \l__spath_tmpe_fp
4007 {
4008     sqrt(
4009         \l__spath_tmpe_fp * \l__spath_tmpe_fp
4010         +
4011         \l__spath_tmpe_fp * \l__spath_tmpe_fp
4012     )
4013     *
4014     \l__spath_tmpe_int
4015 }
4016
4017 \fp_compare:nTF
4018 {
4019     \l__spath_tmpe_fp > #2
4020 }
4021 {
4022
4023     \fp_set:Nn \l__spath_tmpe_fp
4024     {
4025         #2/ \l__spath_tmpe_fp
4026     }
4027
4028     \tl_if_eq:NNTF \l__spath_tmpe_tl \c_spath_curvetoa_tl
4029     {
4030         \spath_split_curve:NNVV
4031         \l__spath_tmpe_tl
4032         \l__spath_tmpe_tl
4033         \l__spath_tmpe_tl
4034         \l__spath_tmpe_fp
4035     }
4036     {
4037         \spath_split_line:NNVV
4038         \l__spath_tmpe_tl
4039         \l__spath_tmpe_tl
4040         \l__spath_tmpe_tl
4041         \l__spath_tmpe_fp
4042     }
4043
4044     \prg_replicate:nn {2}
4045     {
4046         \tl_set:Nx \l__spath_tmpe_tl {\tl_tail:N \l__spath_tmpe_tl}
4047     }
4048
4049     \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpe_tl

```

```

4050 }
4051 {
4052 {
4053
4054 \tl_put_left:NV \l__spath_tmpa_tl \c_spath_moveto_tl
4055
4056 \int_compare:nT
4057 {
4058 \tl_count:N \l__spath_tmpa_tl > 3
4059 }
4060 {
4061 \dim_set:Nn \l__spath_tmpa_dim {\fp_to_dim:n {#2 - \l__spath_tmpc_fp } }
4062 \spath_shorten_at_start:NV \l__spath_tmpa_tl \l__spath_tmpa_dim
4063 }
4064 }
4065
4066 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
4067 \group_end:
4068 }
4069 {
4070 \tl_gset:Nn \g__spath_output_tl {#1}
4071 }
4072 }
4073 \cs_new_protected_nopar:Npn \spath_shorten_at_start:Nnn #1#2#3
4074 {
4075 \__spath_shorten_at_start:nn {#2}{#3}
4076 \tl_set_eq:NN #1 \g__spath_output_tl
4077 \tl_gclear:N \g__spath_output_tl
4078 }
4079 \cs_generate_variant:Nn \spath_shorten_at_start:Nnn {NVV, cnn, cVV, NVn}
4080 \cs_new_protected_nopar:Npn \spath_shorten_at_start:Nn #1#2
4081 {
4082 \spath_shorten_at_start:NVn #1#1{#2}
4083 }
4084 \cs_generate_variant:Nn \spath_shorten_at_start:Nn {cn, cV, NV}
4085 \cs_new_protected_nopar:Npn \spath_gshorten_at_start:Nnn #1#2#3
4086 {
4087 \__spath_shorten_at_start:nn {#2}{#3}
4088 \tl_gset_eq:NN #1 \g__spath_output_tl
4089 \tl_gclear:N \g__spath_output_tl
4090 }
4091 \cs_generate_variant:Nn \spath_gshorten_at_start:Nnn {NVV, cnn, cVV, NVn}
4092 \cs_new_protected_nopar:Npn \spath_gshorten_at_start:Nn #1#2
4093 {
4094 \spath_gshorten_at_start:NVn #1#1{#2}
4095 }
4096 \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.

```

4097 \cs_new_protected_nopar:Npn \spath_shorten_at_both_ends:Nnn #1#2#3
4098 {
4099 \spath_shorten_at_start:Nnn #1{#2}{#3}

```

```

4100 \spath_shorten_at_end:Nnn #1{#2}{#3}
4101 }
4102 \cs_new_protected_nopar:Npn \spath_shorten_at_both_ends:Nn #1#2
4103 {
4104   \spath_shorten_at_start:Nn #1{#2}
4105   \spath_shorten_at_end:Nn #1{#2}
4106 }
4107 \cs_generate_variant:Nn \spath_shorten_at_both_ends:Nn {cn, cV, NV}
4108 \cs_new_protected_nopar:Npn \spath_gshorten_at_both_ends:Nnn #1#2#3
4109 {
4110   \spath_gshorten_at_start:Nnn #1{#2}{#3}
4111   \spath_gshorten_at_end:Nnn #1{#2}{#3}
4112 }
4113 \cs_new_protected_nopar:Npn \spath_gshorten_at_both_ends:Nn #1#2
4114 {
4115   \spath_gshorten_at_start:Nn #1{#2}
4116   \spath_gshorten_at_end:Nn #1{#2}
4117 }
4118 \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` Get the location of a point on a path, using the same location specification as the intersection library.

```

\spath_gpoint_at:Nnn
4119 \cs_new_protected_nopar:Npn \__spath_point_at:nn #1#2
4120 {
4121   \group_begin:
4122   \int_set:Nn \l__spath_tmpa_int {\fp_to_int:n {floor(#2) + 1}}
4123   \fp_set:Nn \l__spath_tmpa_fp {#2 - floor(#2)}
4124
4125   \spath_segments_to_seq:Nn \l__spath_tmpa_seq {#1}
4126
4127   \int_compare:nTF
4128   {
4129     \l__spath_tmpa_int < 1
4130   }
4131   {
4132     \spath_initialpoint:Nn \l__spath_tmpc_tl {#1}
4133   }
4134   {
4135     \int_compare:nTF
4136     {
4137       \l__spath_tmpa_int > \seq_count:N \l__spath_tmpa_seq
4138     }
4139     {
4140       \spath_finalpoint:Nn \l__spath_tmpc_tl {#1}
4141     }
4142     {
4143
4144       \tl_set:Nx
4145       \l__spath_tmpa_tl
4146       {\seq_item:Nn \l__spath_tmpa_seq { \l__spath_tmpa_int } }

```

```

4147
4148 \int_compare:nTF
4149 {
4150   \tl_count:N \l__spath_tmpa_tl > 3
4151 }
4152 {
4153   \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {4}}
4154 }
4155 {
4156   \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {1}}
4157 }
4158
4159 \tl_clear:N \l__spath_tmpc_tl
4160
4161 \token_case_meaning:Nn \l__spath_tmpb_tl
4162 {
4163   \c_spath_moveto_tl
4164   {
4165     \tl_set:Nx \l__spath_tmpc_tl
4166     {
4167       {
4168         \tl_item:Nn \l__spath_tmpa_tl {2}
4169       }
4170       {
4171         \tl_item:Nn \l__spath_tmpa_tl {3}
4172       }
4173     }
4174   }
4175
4176   \c_spath_lineto_tl
4177   {
4178     \tl_set:Nx \l__spath_tmpc_tl
4179     {
4180       {\fp_to_dim:n
4181         {
4182           (1 - \l__spath_tmpa_fp) * ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4183         +
4184           \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4185         }
4186       }
4187       {\fp_to_dim:n
4188         {
4189           (1 - \l__spath_tmpa_fp) * ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4190         +
4191           \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4192         }
4193       }
4194     }
4195   }
4196
4197   \c_spath_rectsize_tl
4198   {
4199     \fp_compare:nTF
4200     {

```

```

4201     \l__spath_tmpa_fp <= .25
4202 }
4203 {
4204     \tl_set:Nx \l__spath_tmpc_tl
4205     {
4206         {\fp_to_dim:n
4207             {
4208                 ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4209                 +
4210                 4 * \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4211             }
4212         }
4213         {\fp_to_dim:n {\tl_item:Nn \l__spath_tmpa_tl {3} } }
4214     }
4215 }
4216 {
4217     \fp_compare:nTF
4218     {
4219         \l__spath_tmpa_fp <= .5
4220     }
4221     {
4222         \tl_set:Nx \l__spath_tmpc_tl
4223         {
4224             {\fp_to_dim:n
4225                 {
4226                     ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4227                     +
4228                     ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4229                 }
4230             }
4231             {\fp_to_dim:n
4232                 {
4233                     ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4234                     +
4235                     (4 * (\l__spath_tmpa_fp) - 1) * ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4236                 }
4237             }
4238         }
4239     }
4240 }
4241 {
4242     \fp_compare:nTF
4243     {
4244         \l__spath_tmpa_fp <= .75
4245     }
4246     {
4247         \tl_set:Nx \l__spath_tmpc_tl
4248         {
4249             {\fp_to_dim:n
4250                 {
4251                     ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4252                     +
4253                     (3 - 4 * (\l__spath_tmpa_fp)) * ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4254                 }
4255             }
4256         }
4257     }
4258 }

```

```

4255         {\fp_to_dim:n
4256         {
4257             ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4258             +
4259             ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4260         }
4261     }
4262 }
4263
4264 }
4265 {
4266     \tl_set:Nx \l__spath_tmpc_tl
4267     {
4268         {\fp_to_dim:n
4269         {
4270             ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4271         }
4272         }
4273         {\fp_to_dim:n
4274         {
4275             ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4276             +
4277             (4 - 4 *(\l__spath_tmpa_fp)) * ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4278         }
4279         }
4280     }
4281 }
4282 }
4283 }
4284 }
4285
4286 \c_spath_closepath_tl
4287 {
4288     \tl_set:Nx \l__spath_tmpc_tl
4289     {
4290         {\fp_to_dim:n
4291         {
4292             (1 - \l__spath_tmpa_fp) * ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4293             +
4294             \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4295         }
4296         }
4297         {\fp_to_dim:n
4298         {
4299             (1 - \l__spath_tmpa_fp) * ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4300             +
4301             \l__spath_tmpa_fp * ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4302         }
4303         }
4304     }
4305 }
4306
4307 \c_spath_curvetoa_tl
4308 {

```

```

4309 \tl_set:Nx \l__spath_tmpc_tl
4310 {
4311   {\fp_to_dim:n
4312     {
4313       (1 - \l__spath_tmpa_fp)^3 * \tl_item:Nn \l__spath_tmpa_tl {2}
4314       + 3 * (1 - \l__spath_tmpa_fp)^2 * (\l__spath_tmpa_fp)
4315       * \tl_item:Nn \l__spath_tmpa_tl {5}
4316       + 3 * (1 - \l__spath_tmpa_fp) * (\l__spath_tmpa_fp)^2
4317       * \tl_item:Nn \l__spath_tmpa_tl {8}
4318       + (\l__spath_tmpa_fp)^3 * \tl_item:Nn \l__spath_tmpa_tl {11}
4319     }}
4320   {\fp_to_dim:n
4321     {
4322       (1 - \l__spath_tmpa_fp)^3 * \tl_item:Nn \l__spath_tmpa_tl {3}
4323       + 3 * (1 - \l__spath_tmpa_fp)^2 * (\l__spath_tmpa_fp)
4324       * \tl_item:Nn \l__spath_tmpa_tl {6}
4325       + 3 * (1 - \l__spath_tmpa_fp) * (\l__spath_tmpa_fp)^2
4326       * \tl_item:Nn \l__spath_tmpa_tl {9}
4327       + (\l__spath_tmpa_fp)^3 * \tl_item:Nn \l__spath_tmpa_tl {12}
4328     }}
4329   }
4330 }
4331 }
4332 }
4333 }
4334
4335 \tl_gclear:N \g__spath_output_tl
4336 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpc_tl
4337 \group_end:
4338 }
4339 \cs_new_protected_nopar:Npn \spath_point_at:Nnn #1#2#3
4340 {
4341   \__spath_point_at:nn {#2}{#3}
4342   \tl_set_eq:NN #1 \g__spath_output_tl
4343   \tl_gclear:N \g__spath_output_tl
4344 }
4345 \cs_generate_variant:Nn \spath_point_at:Nnn {NVn, NVV, NnV}
4346 \cs_new_protected_nopar:Npn \spath_gpoint_at:Nnn #1#2#3
4347 {
4348   \__spath_point_at:nn {#2}{#3}
4349   \tl_gset_eq:NN #1 \g__spath_output_tl
4350   \tl_gclear:N \g__spath_output_tl
4351 }
4352 \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`

```

4353 \cs_new_protected_nopar:Npn \__spath_tangent_at:nn #1#2
4354 {
4355   \group_begin:
4356   \int_set:Nn \l__spath_tmpa_int {\fp_to_int:n {floor(#2) + 1}}
4357   \fp_set:Nn \l__spath_tmpa_fp {#2 - floor(#2)}

```



```

4358
4359 \spath_segments_to_seq:Nn \l__spath_tmpa_seq {#1}
4360
4361 \int_compare:nT
4362 {
4363   \l__spath_tmpa_int < 1
4364 }
4365 {
4366   \int_set:Nn \l__spath_tmpa_int {1}
4367   \fp_set:Nn \l__spath_tmpa_fp {0}
4368 }
4369
4370 \int_compare:nT
4371 {
4372   \l__spath_tmpa_int > \seq_count:N \l__spath_tmpa_seq
4373 }
4374 {
4375   \int_set:Nn \l__spath_tmpa_int { \seq_count:N \l__spath_tmpa_seq }
4376   \fp_set:Nn \l__spath_tmpa_fp {1}
4377 }
4378
4379 \tl_set:Nx
4380 \l__spath_tmpa_tl
4381 {\seq_item:Nn \l__spath_tmpa_seq { \l__spath_tmpa_int } }
4382
4383 \int_compare:nTF
4384 {
4385   \tl_count:N \l__spath_tmpa_tl > 3
4386 }
4387 {
4388   \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {4}}
4389 }
4390 {
4391   \tl_set:Nx \l__spath_tmpb_tl {\tl_item:Nn \l__spath_tmpa_tl {1}}
4392 }
4393
4394 \tl_clear:N \l__spath_tmpc_tl
4395
4396 \token_case_meaning:Nn \l__spath_tmpb_tl
4397 {
4398   \c_spath_moveto_tl
4399   {
4400     \tl_set:Nx \l__spath_tmpc_tl
4401     {
4402       {
4403         \tl_item:Nn \l__spath_tmpa_tl {2}
4404       }
4405       {
4406         \tl_item:Nn \l__spath_tmpa_tl {3}
4407       }
4408     }
4409   }
4410
4411   \c_spath_lineto_tl

```

```

4412 {
4413   \tl_set:Nx \l__spath_tmpc_tl
4414   {
4415     {\fp_to_dim:n
4416       {
4417         ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4418         -
4419         ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4420       }
4421     }
4422     {\fp_to_dim:n
4423       {
4424         ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4425         -
4426         ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4427       }
4428     }
4429   }
4430 }
4431
4432 \c_spath_rectsize_tl
4433 {
4434   \fp_compare:nTF
4435   {
4436     \l__spath_tmpa_fp <= .25
4437   }
4438   {
4439     \tl_set:Nx \l__spath_tmpc_tl
4440     {
4441       {\fp_to_dim:n
4442         {
4443           \tl_item:Nn \l__spath_tmpa_tl {5}
4444         }
4445       }
4446       {0pt}
4447     }
4448   }
4449   {
4450     \fp_compare:nTF
4451     {
4452       \l__spath_tmpa_fp <= .5
4453     }
4454     {
4455       \tl_set:Nx \l__spath_tmpc_tl
4456       {
4457         {0pt}
4458         {\fp_to_dim:n
4459           {
4460             ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4461           }
4462         }
4463       }
4464     }
4465   }

```

```

4466         \fp_compare:nTF
4467         {
4468             \l__spath_tmpa_fp <= .75
4469         }
4470         {
4471             \tl_set:Nx \l__spath_tmpc_tl
4472             {
4473                 {\fp_to_dim:n
4474                 {
4475                     -( \tl_item:Nn \l__spath_tmpa_tl {5} )
4476                 }
4477                 }
4478                 {0pt}
4479             }
4480
4481         }
4482         {
4483             \tl_set:Nx \l__spath_tmpc_tl
4484             {
4485                 {0pt}
4486                 {\fp_to_dim:n
4487                 {
4488                     - ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4489                 }
4490                 }
4491             }
4492         }
4493     }
4494 }
4495 }
4496
4497 \c_spath_closepath_tl
4498 {
4499     \tl_set:Nx \l__spath_tmpc_tl
4500     {
4501         {\fp_to_dim:n
4502         {
4503             ( \tl_item:Nn \l__spath_tmpa_tl {5} )
4504             -
4505             ( \tl_item:Nn \l__spath_tmpa_tl {2} )
4506         }
4507         }
4508         {\fp_to_dim:n
4509         {
4510             ( \tl_item:Nn \l__spath_tmpa_tl {6} )
4511             -
4512             ( \tl_item:Nn \l__spath_tmpa_tl {3} )
4513         }
4514         }
4515     }
4516 }
4517
4518 \c_spath_curvetoa_tl
4519 {

```

```

4520 \tl_set:Nx \l__spath_tmpc_tl
4521 {
4522   {\fp_to_dim:n
4523     {
4524       3*(1 - \l__spath_tmpa_fp)^2 * (\tl_item:Nn \l__spath_tmpa_tl {5}
4525       - \tl_item:Nn \l__spath_tmpa_tl {2})
4526       + 6 * (1 - \l__spath_tmpa_fp) * (\l__spath_tmpa_fp) *
4527       (\tl_item:Nn \l__spath_tmpa_tl {8}
4528       - \tl_item:Nn \l__spath_tmpa_tl {5})
4529       + 3*(\l__spath_tmpa_fp)^2 * (\tl_item:Nn \l__spath_tmpa_tl {11}
4530       - \tl_item:Nn \l__spath_tmpa_tl {8})
4531     }
4532   }
4533   {\fp_to_dim:n
4534     {
4535       3*(1 - \l__spath_tmpa_fp)^2 * (\tl_item:Nn \l__spath_tmpa_tl {6}
4536       - \tl_item:Nn \l__spath_tmpa_tl {3})
4537       + 6 * (1 - \l__spath_tmpa_fp) * (\l__spath_tmpa_fp) *
4538       (\tl_item:Nn \l__spath_tmpa_tl {9}
4539       - \tl_item:Nn \l__spath_tmpa_tl {6})
4540       + 3*(\l__spath_tmpa_fp)^2 * (\tl_item:Nn \l__spath_tmpa_tl {12}
4541       - \tl_item:Nn \l__spath_tmpa_tl {9})
4542     }
4543   }
4544 }
4545 }
4546
4547 \tl_gclear:N \g__spath_output_tl
4548 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpc_tl
4549 \group_end:
4550 }
4551 \cs_new_protected_nopar:Npn \spath_tangent_at:Nnn #1#2#3
4552 {
4553   \__spath_tangent_at:nn {#2}{#3}
4554   \tl_set_eq:NN #1 \g__spath_output_tl
4555   \tl_gclear:N \g__spath_output_tl
4556 }
4557 \cs_generate_variant:Nn \spath_tangent_at:Nnn {NVn, NVV, NnV}
4558 \cs_new_protected_nopar:Npn \spath_gtangent_at:Nnn #1#2#3
4559 {
4560   \__spath_tangent_at:nn {#2}{#3}
4561   \tl_gset_eq:NN #1 \g__spath_output_tl
4562   \tl_gclear:N \g__spath_output_tl
4563 }
4564 \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.

```

4565 \cs_new_protected_nopar:Npn \__spath_transformation_at:nn #1#2
4566 {
4567   \group_begin:
4568   \tl_clear:N \l__spath_tmpa_tl

```

```

4569 \__spath_tangent_at:nn {#1}{#2}
4570 \tl_set_eq:NN \l__spath_tmpb_tl \g__spath_output_tl
4571 \fp_set:Nn \l__spath_tmpa_fp
4572 {
4573   sqrt(
4574     (\tl_item:Nn \l__spath_tmpb_tl {1})^2
4575     +
4576     (\tl_item:Nn \l__spath_tmpb_tl {2})^2
4577   )
4578 }
4579 \fp_compare:nTF {\l__spath_tmpa_fp = 0}
4580 {
4581   \fp_set:Nn \l__spath_tmpa_fp {1}
4582   \fp_set:Nn \l__spath_tmpb_fp {0}
4583 }
4584 {
4585   \fp_set:Nn \l__spath_tmpb_fp
4586   { (\tl_item:Nn \l__spath_tmpb_tl {2}) / \l__spath_tmpa_fp }
4587   \fp_set:Nn \l__spath_tmpa_fp
4588   { (\tl_item:Nn \l__spath_tmpb_tl {1}) / \l__spath_tmpa_fp }
4589 }
4590 \tl_set:Nx \l__spath_tmpa_tl
4591 {
4592   { \fp_to_decimal:n { \l__spath_tmpa_fp } }
4593   { \fp_to_decimal:n { \l__spath_tmpb_fp } }
4594   { \fp_to_decimal:n {- \l__spath_tmpb_fp } }
4595   { \fp_to_decimal:n { \l__spath_tmpa_fp } }
4596 }
4597 \__spath_point_at:nn {#1}{#2}
4598 \tl_put_right:NV \l__spath_tmpa_tl \g__spath_output_tl
4599 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
4600 \group_end:
4601 }
4602 \cs_new_protected_nopar:Npn \spath_transformation_at:Nnn #1#2#3
4603 {
4604   \__spath_transformation_at:nn {#2}{#3}
4605   \tl_set_eq:NN #1 \g__spath_output_tl
4606   \tl_gclear:N \g__spath_output_tl
4607 }
4608 \cs_generate_variant:Nn \spath_transformation_at:Nnn {NVn, NVV, NnV, NvV}
4609 \cs_new_protected_nopar:Npn \spath_gtransformation_at:Nnn #1#2#3
4610 {
4611   \__spath_transformation_at:nn {#2}{#3}
4612   \tl_gset_eq:NN #1 \g__spath_output_tl
4613   \tl_gclear:N \g__spath_output_tl
4614 }
4615 \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 4616 \cs_new_protected_nopar:Npn \spath_intersect:NN #1#2
4617 {
4618   \pgfintersectionofpaths%
4619   {%
4620     \pgfsetpath #1
4621   }{%
4622     \pgfsetpath #2
4623   }
4624 }
4625 \cs_new_protected_nopar:Npn \spath_intersect:nn #1#2
4626 {
4627   \tl_set:Nn \l__spath_intersecta_tl {#1}
4628   \tl_set:Nn \l__spath_intersectb_tl {#2}
4629   \spath_intersect:NN \l__spath_intersecta_tl \l__spath_intersectb_tl
4630 }

```

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

`\spath_split_component_at_intersections:Nnn`

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

```

4631 \cs_new_protected_nopar:Npn \__spath_split_component_at_intersections:nn #1#2
4632 {
4633   \group_begin:
4634   \tl_clear:N \l__spath_tmpe_tl
4635   \seq_clear:N \l__spath_tmpb_seq
4636
4637   % Find the intersections of these segments
4638   \tl_set:Nn \l__spath_tmpb_tl {#1}
4639   \tl_set:Nn \l__spath_tmpe_tl {#2}
4640
4641   % Remember if the component is closed
4642   \spath_finalaction:NV \l__spath_tmpe_tl \l__spath_tmpb_tl
4643
4644   \bool_set:Nn \l__spath_closed_bool
4645   {
4646     \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_closepath_tl
4647     ||
4648     \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_rectcorner_tl
4649   }
4650
4651   % Open it
4652   \spath_open:N \l__spath_tmpb_tl
4653
4654   \spath_reallength:NV \l__spath_tmpe_int \l__spath_tmpb_tl
4655
4656   % Sort intersections along the component
4657   \pgfintersectionsorthbyfirstpath
4658   \spath_intersect:NN \l__spath_tmpe_tl \l__spath_tmpe_tl
4659
4660   % If we get intersections

```

```

4661 \int_compare:nT {\pgfintersectionsolutions > 0}
4662 {
4663   % Find the times of the intersections on the component
4664   \int_step_inline:nnnn {1} {1} {\pgfintersectionsolutions}
4665   {
4666     \pgfintersectiongetsolutiontimes{##1}{\l__spath_tmph_tl}{\l__spath_tmpi_tl}
4667     \seq_put_left:NV \l__spath_tmpb_seq \l__spath_tmph_tl
4668   }
4669
4670   \seq_get_left:NN \l__spath_tmpb_seq \l__spath_tmpa_tl
4671   \fp_compare:nT
4672   {
4673     \l__spath_tmpa_tl > \l__spath_tmpa_int - .01
4674   }
4675   {
4676     \bool_set_false:N \l__spath_closed_bool
4677   }
4678
4679   \seq_get_right:NN \l__spath_tmpb_seq \l__spath_tmpa_tl
4680   \fp_compare:nT
4681   {
4682     \l__spath_tmpa_tl < .01
4683   }
4684   {
4685     \bool_set_false:N \l__spath_closed_bool
4686   }
4687
4688   \tl_set:Nn \l__spath_tmpg_tl {-1}
4689
4690   \seq_map_inline:Nn \l__spath_tmpb_seq
4691   {
4692     \tl_set:Nn \l__spath_tmph_tl {##1}
4693
4694     \tl_set_eq:NN \l__spath_tmpa_tl \l__spath_tmph_tl
4695     \int_compare:nT
4696     {
4697       \fp_to_int:n {floor( \l__spath_tmph_tl ) }
4698       =
4699       \fp_to_int:n {floor( \l__spath_tmpg_tl ) }
4700     }
4701     {
4702       \tl_set:Nx \l__spath_tmph_tl
4703       {
4704         \fp_eval:n {
4705           floor( \l__spath_tmph_tl )
4706           +
4707           ( \l__spath_tmph_tl - floor( \l__spath_tmph_tl ) )
4708           /
4709           ( \l__spath_tmpg_tl - floor( \l__spath_tmpg_tl ) )
4710         }
4711       }
4712     }
4713     \tl_set_eq:NN \l__spath_tmpg_tl \l__spath_tmpa_tl
4714

```

```

4715     \spath_split_at:NNVV
4716     \l__spath_tmpd_tl
4717     \l__spath_tmpf_tl
4718     \l__spath_tmpe_tl
4719     \l__spath_tmph_tl
4720
4721     \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmpf_tl
4722     \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmph_tl
4723
4724   }
4725
4726
4727   \tl_put_left:NV \l__spath_tmpe_tl \l__spath_tmph_tl
4728
4729   \spath_remove_empty_components:N \l__spath_tmpe_tl
4730
4731   \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmph_tl
4732 }
4733
4734
4735 \bool_if:NT \l__spath_closed_bool
4736 {
4737   \spath_join_component:Nn \l__spath_tmpe_tl {1}
4738 }
4739
4740 \tl_gclear:N \g__spath_output_tl
4741 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpe_tl
4742
4743 \group_end:
4744 }
4745 \cs_new_protected_nopar:Npn \spath_split_component_at_intersections:Nnn #1#2#3
4746 {
4747   \__spath_split_component_at_intersections:nn {#2}{#3}
4748   \tl_set_eq:NN #1 \g__spath_output_tl
4749   \tl_gclear:N \g__spath_output_tl
4750 }
4751 \cs_generate_variant:Nn \spath_split_component_at_intersections:Nnn {NVn, NVV}
4752 \cs_new_protected_nopar:Npn \spath_split_component_at_intersections:Nn #1#2
4753 {
4754   \spath_split_component_at_intersections:NVn #1#1{#2}
4755 }
4756 \cs_generate_variant:Nn \spath_split_component_at_intersections:Nn {cn, cv}
4757 \cs_new_protected_nopar:Npn \spath_gsplit_component_at_intersections:Nnn #1#2#3
4758 {
4759   \__spath_split_component_at_intersections:nn {#2}{#3}
4760   \tl_gset_eq:NN #1 \g__spath_output_tl
4761   \tl_gclear:N \g__spath_output_tl
4762 }
4763 \cs_generate_variant:Nn \spath_gsplit_component_at_intersections:Nnn {NVn, NVV}
4764 \cs_new_protected_nopar:Npn \spath_gsplit_component_at_intersections:Nn #1#2
4765 {
4766   \spath_gsplit_component_at_intersections:NVn #1#1{#2}
4767 }
4768 \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:Nnn
\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
4769 \cs_new_protected_nopar:Npn \__spath_split_path_at_intersections:nn #1#2
4770 {
4771   \group_begin:
4772
4773   \seq_clear:N \l__spath_tmpa_seq
4774   \seq_clear:N \l__spath_tmpb_seq
4775
4776   \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
4777   \seq_map_inline:Nn \l__spath_tmpa_seq
4778   {
4779     \spath_split_component_at_intersections:Nnn \l__spath_tmpa_tl {##1} {#2}
4780     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
4781   }
4782
4783   \tl_gclear:N \g__spath_output_tl
4784   \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpb_seq {} }
4785   \group_end:
4786 }
4787 \cs_new_protected_nopar:Npn \spath_split_path_at_intersections:Nnn #1#2#3
4788 {
4789   \__spath_split_path_at_intersections:nn {#2}{#3}
4790   \tl_set_eq:NN #1 \g__spath_output_tl
4791   \tl_gclear:N \g__spath_output_tl
4792 }
4793 \cs_generate_variant:Nn \spath_split_path_at_intersections:Nnn
4794 {NVn, NVV, cVn, cVV, cvn, cvv}
4795 \cs_new_protected_nopar:Npn \spath_split_path_at_intersections:Nn #1#2
4796 {
4797   \spath_split_path_at_intersections:NVn #1#1{#2}
4798 }
4799 \cs_generate_variant:Nn \spath_split_path_at_intersections:Nn {cv, NV}
4800 \cs_new_protected_nopar:Npn \spath_gsplit_path_at_intersections:Nnn #1#2#3
4801 {
4802   \__spath_split_path_at_intersections:nn {#2}{#3}
4803   \tl_gset_eq:NN #1 \g__spath_output_tl
4804   \tl_gclear:N \g__spath_output_tl
4805 }
4806 \cs_generate_variant:Nn \spath_gsplit_path_at_intersections:Nnn
4807 {NVn, NVV, cVn, cVV, cvn, cvv}
4808 \cs_new_protected_nopar:Npn \spath_gsplit_path_at_intersections:Nn #1#2
4809 {
4810   \spath_gsplit_path_at_intersections:NVn #1#1{#2}
4811 }
4812 \cs_generate_variant:Nn \spath_gsplit_path_at_intersections:Nn {cv, NV}
4813 \cs_new_protected_nopar:Npn \spath_split_at_intersections:NNnn #1#2#3#4
4814 {
4815   \__spath_split_path_at_intersections:nn {#3}{#4}
4816   \tl_set_eq:NN #1 \g__spath_output_tl
4817   \__spath_split_path_at_intersections:nn {#4}{#3}
4818   \tl_set_eq:NN #2 \g__spath_output_tl

```

```

4819 \tl_gclear:N \g__spath_output_tl
4820 }
4821 \cs_generate_variant:Nn \spath_split_at_intersections:NNnn
4822 {NNVn, NNVV, ccVn, ccVV, ccvn, ccvv}
4823 \cs_new_protected_nopar:Npn \spath_split_at_intersections:NN #1#2
4824 {
4825 \spath_split_at_intersections:NNVV #1#2#1#2
4826 }
4827 \cs_generate_variant:Nn \spath_split_at_intersections:NN {cc}
4828 \cs_new_protected_nopar:Npn \spath_gsplit_at_intersections:NNnn #1#2#3#4
4829 {
4830 \__spath_split_path_at_intersections:nn {#3}{#4}
4831 \tl_gset_eq:NN #1 \g__spath_output_tl
4832 \__spath_split_path_at_intersections:nn {#4}{#3}
4833 \tl_gset_eq:NN #2 \g__spath_output_tl
4834 \tl_gclear:N \g__spath_output_tl
4835 }
4836 \cs_generate_variant:Nn \spath_gsplit_at_intersections:NNnn
4837 {NNVn, NNVV, ccVn, ccVV, ccvn, ccvv}
4838 \cs_new_protected_nopar:Npn \spath_gsplit_at_intersections:NN #1#2
4839 {
4840 \spath_gsplit_at_intersections:NNVV #1#2#1#2
4841 }
4842 \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.

```

4843 \cs_new_protected_nopar:Npn \__spath_split_component_at_self_intersections:n #1
4844 {
4845 \group_begin:
4846 \tl_set:Nn \l__spath_tmpe_tl {#1}

```

Remember if the component is closed

```

4847 \spath_finalaction:NV \l__spath_tmpe_tl \l__spath_tmpe_tl
4848
4849 \bool_set:Nn \l__spath_closed_bool
4850 {
4851 \tl_if_eq_p:NN \l__spath_tmpe_tl \c_spath_closepath_tl
4852 }
4853

```

Copy the path

```

4854 \tl_set:Nn \l__spath_tmpe_tl {#1}
4855

```

Open the path

```

4856 \spath_open:N \l__spath_tmpe_tl

```

Ensure beziers don't self-intersect

```

4857 \spath_split_curves:N \l__spath_tmpe_tl
4858

```

Make a copy for later

```

4859 \tl_set_eq:NN \l__spath_tmpg_tl \l__spath_tmpe_tl
4860

```

Clear some token lists and sequences

```

4861 \tl_clear:N \l__spath_tmpd_tl
4862 \seq_clear:N \l__spath_tmpb_seq
4863 \int_zero:N \l__spath_tmpa_int
4864
4865 \pgfintersectionsorthbyfirstpath
4866

```

Split the path into a sequence of segments

```

4867 \spath_segments_to_seq:NV \l__spath_tmpa_seq \l__spath_tmpe_tl
4868
4869 \seq_map_indexed_inline:Nn \l__spath_tmpa_seq
4870 {
4871   \seq_map_indexed_inline:Nn \l__spath_tmpa_seq
4872   {

```

Don't intersect a segment with itself

```

4873   \int_compare:nF
4874   {
4875     ##1 == ####1
4876   }
4877   {
4878     \spath_intersect:nn {##2} {####2}
4879
4880     \int_compare:nT {\pgfintersectionsolutions > 0}
4881     {

```

Find the times of the intersections on each path

```

4882     \int_step_inline:nnnn {1} {1} {\pgfintersectionsolutions}
4883     {
4884       \pgfintersectiongetsolutiontimes
4885       {#####1}{\l__spath_tmpb_tl}{\l__spath_tmpe_tl}
4886
4887       \bool_if:nT
4888       {
4889         !(
4890           \fp_compare_p:n { \l__spath_tmpb_tl > .99 }
4891           &&
4892           \int_compare_p:n {##1 + 1 == ####1}
4893         )
4894         &&
4895         !(
4896           \fp_compare_p:n { \l__spath_tmpb_tl < .01 }
4897           &&
4898           \int_compare_p:n {##1 - 1 == ####1}
4899         )
4900         &&
4901         !(
4902           \l__spath_closed_bool
4903           &&
4904           \fp_compare_p:n { \l__spath_tmpb_tl < .01 }
4905           &&
4906           \int_compare_p:n {##1 == 1}
4907           &&
4908           \int_compare_p:n {\seq_count:N \l__spath_tmpa_seq == ####1}

```

```

4909         )
4910         &&
4911         !(
4912         \l__spath_closed_bool
4913         &&
4914         \fp_compare_p:n { \l__spath_tmpb_tl > .99 }
4915         &&
4916         \int_compare_p:n {####1 == 1}
4917         &&
4918         \int_compare_p:n {\seq_count:N \l__spath_tmpa_seq == ##1}
4919         )
4920     }
4921     {
4922         \tl_set:Nx \l__spath_tmpa_tl
4923         {\fp_to_decimal:n {\l__spath_tmpb_tl + ##1 - 1}}
4924         \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
4925     }
4926 }
4927 }
4928 }
4929 }
4930 }
4931

```

Sort the sequence by reverse order along the path

```

4932 \seq_sort:Nn \l__spath_tmpb_seq
4933 {
4934     \fp_compare:nNnTF { ##1 } < { ##2 }
4935     { \sort_return_swapped: }
4936     { \sort_return_same: }
4937 }
4938
4939 \seq_get_left:NN \l__spath_tmpb_seq \l__spath_tmpa_tl
4940 \fp_compare:nT
4941 {
4942     \l__spath_tmpa_tl > \seq_count:N \l__spath_tmpa_seq - .01
4943 }
4944 {
4945     \bool_set_false:N \l__spath_closed_bool
4946 }
4947 \seq_get_right:NN \l__spath_tmpb_seq \l__spath_tmpa_tl
4948 \fp_compare:nT
4949 {
4950     \l__spath_tmpa_tl < .01
4951 }
4952 {
4953     \bool_set_false:N \l__spath_closed_bool
4954 }
4955

```

Restore the original copy of the path

```

4956 \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpg_tl
4957

```

Clear the token lists

```

4958 \tl_clear:N \l__spath_tmpf_tl
4959 \tl_clear:N \l__spath_tmph_tl
4960 \tl_clear:N \l__spath_tmpg_tl
4961
4962 \tl_set:Nn \l__spath_tmphi_tl {-1}
4963
4964 \seq_map_inline:Nn \l__spath_tmpb_seq
4965 {
4966   \tl_set:Nn \l__spath_tmpb_tl {##1}
4967   \tl_set_eq:NN \l__spath_tmpha_tl \l__spath_tmpb_tl
4968   \int_compare:nT
4969   {
4970     \fp_to_int:n {floor( \l__spath_tmpb_tl ) }
4971     =
4972     \fp_to_int:n {floor( \l__spath_tmphi_tl ) }
4973   }
4974   {
4975     \tl_set:Nx \l__spath_tmpb_tl
4976     {
4977       \fp_eval:n {
4978         floor( \l__spath_tmpb_tl )
4979         +
4980         ( \l__spath_tmpb_tl - floor( \l__spath_tmpb_tl ) )
4981         /
4982         ( \l__spath_tmphi_tl - floor( \l__spath_tmphi_tl ) )
4983       }
4984     }
4985   }
4986   \tl_set_eq:NN \l__spath_tmphi_tl \l__spath_tmpha_tl
4987
4988   \spath_split_at:NNVV
4989   \l__spath_tmpf_tl
4990   \l__spath_tmph_tl
4991   \l__spath_tmpe_tl
4992   \l__spath_tmpb_tl
4993
4994   \tl_put_left:NV \l__spath_tmpg_tl \l__spath_tmph_tl
4995   \tl_set_eq:NN \l__spath_tmpe_tl \l__spath_tmpf_tl
4996
4997 }
4998
4999 \tl_put_left:NV \l__spath_tmpg_tl \l__spath_tmpe_tl
5000
5001 \tl_if_empty:NT \l__spath_tmpg_tl
5002 {
5003   \tl_set_eq:NN \l__spath_tmpg_tl \l__spath_tmpe_tl
5004 }
5005
5006 \spath_remove_empty_components:N \l__spath_tmpg_tl
5007
Do something with closed
5008 \bool_if:NT \l__spath_closed_bool
5009 {
5010   \spath_join_component:Nn \l__spath_tmpg_tl {1}

```

```

5011 }
5012
5013 \tl_gclear:N \g__spath_output_tl
5014 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpg_tl
5015 \group_end:
5016 }
5017 \cs_new_protected_nopar:Npn \spath_split_component_at_self_intersections:Nn #1#2
5018 {
5019   \__spath_split_component_at_self_intersections:n {#2}
5020   \tl_set_eq:NN #1 \g__spath_output_tl
5021   \tl_gclear:N \g__spath_output_tl
5022 }
5023 \cs_generate_variant:Nn \spath_split_component_at_self_intersections:Nn {NV}
5024 \cs_new_protected_nopar:Npn \spath_split_component_at_self_intersections:N #1
5025 {
5026   \spath_split_component_at_self_intersections:NV #1#1
5027 }
5028 \cs_generate_variant:Nn \spath_split_component_at_self_intersections:N {c}
5029 \cs_new_protected_nopar:Npn \spath_gsplit_component_at_self_intersections:Nn #1#2
5030 {
5031   \__spath_split_component_at_self_intersections:n {#2}
5032   \tl_gset_eq:NN #1 \g__spath_output_tl
5033   \tl_gclear:N \g__spath_output_tl
5034 }
5035 \cs_generate_variant:Nn \spath_gsplit_component_at_self_intersections:Nn {NV}
5036 \cs_new_protected_nopar:Npn \spath_gsplit_component_at_self_intersections:N #1
5037 {
5038   \spath_gsplit_component_at_self_intersections:NV #1#1
5039 }
5040 \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 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.

```

5041 \cs_new_protected_nopar:Npn \__spath_split_at_self_intersections:n #1
5042 {
5043   \group_begin:
5044   \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
5045   \seq_clear:N \l__spath_tmpb_seq
5046   \seq_clear:N \l__spath_tmppc_seq

```

Iterate over the components of the original path.

```

5047   \bool_do_until:nn
5048   {
5049     \seq_if_empty_p:N \l__spath_tmpa_seq
5050   }
5051   {

```

Get the next component

```

5052     \seq_pop_left:NN \l__spath_tmpa_seq \l__spath_tmppa_tl

```

Copy for later

```

5053     \tl_set_eq:NN \l__spath_tmppc_tl \l__spath_tmppa_tl

```

```

5054     \int_compare:nT
5055     {
5056         \tl_count:N \l__spath_tmpa_tl > 3
5057     }
5058     {
Split against itself
5059         \spath_split_component_at_self_intersections:N \l__spath_tmpa_tl
Grab the rest of the path
5060         \tl_set:Nx \l__spath_tmpb_tl
5061         {
5062             \seq_use:Nn \l__spath_tmpb_seq {}
5063             \seq_use:Nn \l__spath_tmpa_seq {}
5064         }
Split against the rest of the path
5065         \spath_split_path_at_intersections:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
5066     }
Save the split path
5067     \seq_put_right:NV \l__spath_tmpc_seq \l__spath_tmpa_tl
Add the original copy to the sequence of processed components
5068     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpc_tl
5069 }
5070
5071 \tl_gclear:N \g__spath_output_tl
5072 \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpc_seq {} }
5073 \group_end:
5074 }
5075 \cs_generate_variant:Nn \__spath_split_at_self_intersections:n {V, v}
5076 \cs_new_protected_nopar:Npn \spath_split_at_self_intersections:Nn #1#2
5077 {
5078     \__spath_split_at_self_intersections:n {#2}
5079     \tl_set_eq:NN #1 \g__spath_output_tl
5080     \tl_gclear:N \g__spath_output_tl
5081 }
5082 \cs_generate_variant:Nn \spath_split_at_self_intersections:Nn {NV, cn, cV, cv}
5083 \cs_new_protected_nopar:Npn \spath_split_at_self_intersections:N #1
5084 {
5085     \spath_split_at_self_intersections:NV #1#1
5086 }
5087 \cs_generate_variant:Nn \spath_split_at_self_intersections:N {c}
5088 \cs_new_protected_nopar:Npn \spath_gsplit_at_self_intersections:Nn #1#2
5089 {
5090     \__spath_split_at_self_intersections:n {#2}
5091     \tl_gset_eq:NN #1 \g__spath_output_tl
5092     \tl_gclear:N \g__spath_output_tl
5093 }
5094 \cs_generate_variant:Nn \spath_gsplit_at_self_intersections:Nn {NV, cn, cV, cv}
5095 \cs_new_protected_nopar:Npn \spath_gsplit_at_self_intersections:N #1
5096 {
5097     \spath_gsplit_at_self_intersections:NV #1#1
5098 }
5099 \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 Join the specified component of the spath to its predecessor.
\spath_join_component:Nn
\spath_gjoin_component:Nnn
\spath_gjoin_component:Nn
5100 \cs_new_protected_nopar:Npn \__spath_join_component:nn #1#2
5101 {
5102   \group_begin:
5103   \spath_numberofcomponents:Nn \l__spath_tmpa_int {#1}
5104
5105   \bool_if:nTF
5106   {
5107     \int_compare_p:n { #2 >= 1 }
5108     &&
5109     \int_compare_p:n { #2 <= \l__spath_tmpa_int }
5110   }
5111   {
5112     \int_compare:nTF
5113     {
5114       #2 == 1
5115     }
5116     {
5117       \int_compare:nTF
5118       {
5119         \l__spath_tmpa_int == 1
5120       }
5121       {
5122         \tl_set:Nn \l__spath_tmpa_tl {#1}
5123         \spath_initialpoint:Nn \l__spath_tmpb_tl {#1}
5124         \tl_put_right:NV \l__spath_tmpa_tl \c_spath_closepath_tl
5125         \tl_put_right:NV \l__spath_tmpa_tl \l__spath_tmpb_tl
5126         \tl_gclear:N \g__spath_output_tl
5127         \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
5128       }
5129       {
5130         \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
5131         \seq_pop_left:NN \l__spath_tmpa_seq \l__spath_tmpa_tl
5132
5133         \prg_replicate:nn {3}
5134         {
5135           \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
5136         }
5137
5138         \seq_put_right:NV \l__spath_tmpa_seq \l__spath_tmpa_tl
5139
5140         \tl_gclear:N \g__spath_output_tl
5141         \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpa_seq {}}
5142       }
5143     }
5144     {
5145       \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
5146
5147       \seq_clear:N \l__spath_tmpb_seq
5148       \seq_map_indexed_inline:Nn \l__spath_tmpa_seq
5149       {
5150         \tl_set:Nn \l__spath_tmpa_tl {##2}

```



```

5151     \int_compare:nT {##1 = #2}
5152     {
5153         \prg_replicate:nn {3}
5154         {
5155             \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
5156         }
5157     }
5158     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
5159 }
5160
5161     \tl_gclear:N \g__spath_output_tl
5162     \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpb_seq {}}
5163 }
5164 }
5165 {
5166     \tl_gclear:N \g__spath_output_tl
5167     \tl_gset:Nn \g__spath_output_tl {#1}
5168 }
5169
5170 \group_end:
5171 }
5172 \cs_new_protected_nopar:Npn \spath_join_component:Nnn #1#2#3
5173 {
5174     \__spath_join_component:nn {#2}{#3}
5175     \tl_set_eq:NN #1 \g__spath_output_tl
5176     \tl_gclear:N \g__spath_output_tl
5177 }
5178 \cs_generate_variant:Nn \spath_join_component:Nnn {NVn, NVV}
5179 \cs_new_protected_nopar:Npn \spath_join_component:Nn #1#2
5180 {
5181     \spath_join_component:NVn #1#1{#2}
5182 }
5183 \cs_generate_variant:Nn \spath_join_component:Nn {cn, NV, cV}
5184 \cs_new_protected_nopar:Npn \spath_gjoin_component:Nnn #1#2#3
5185 {
5186     \__spath_join_component:nn {#2}{#3}
5187     \tl_gset_eq:NN #1 \g__spath_output_tl
5188     \tl_gclear:N \g__spath_output_tl
5189 }
5190 \cs_generate_variant:Nn \spath_gjoin_component:Nnn {NVn, NVV}
5191 \cs_new_protected_nopar:Npn \spath_gjoin_component:Nn #1#2
5192 {
5193     \spath_gjoin_component:NVn #1#1{#2}
5194 }
5195 \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
\spath_spot_weld_components:N
\spath_spot_gweld_components:Nn
\spath_spot_gweld_components:N

```

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

```

5196 \cs_new_protected_nopar:Npn \__spath_spot_weld_components:n #1
5197 {
5198     \group_begin:
5199     \dim_zero:N \l__spath_move_x_dim

```

```

5200 \dim_zero:N \l__spath_move_y_dim
5201
5202 \spath_components_to_seq:Nn \l__spath_tmpa_seq {#1}
5203 \seq_clear:N \l__spath_tmpb_seq
5204 \dim_set:Nn \l__spath_move_x_dim {\tl_item:nn {#1} {2} + 10 pt}
5205 \dim_set:Nn \l__spath_move_y_dim {\tl_item:nn {#1} {3} + 10 pt}
5206
5207 \int_set:Nn \l__spath_tmpa_int {\seq_count:N \l__spath_tmpa_seq}
5208
5209 \seq_map_inline:Nn \l__spath_tmpa_seq
5210 {
5211   \tl_set:Nn \l__spath_tmpa_tl {##1}
5212   \bool_if:nT
5213   {
5214     \dim_compare_p:n
5215     {
5216       \dim_abs:n
5217       {\l__spath_move_x_dim - \tl_item:Nn \l__spath_tmpa_tl {2} }
5218       < 0.01pt
5219     }
5220     &&
5221     \dim_compare_p:n
5222     {
5223       \dim_abs:n
5224       {\l__spath_move_y_dim - \tl_item:Nn \l__spath_tmpa_tl {3} }
5225       < 0.01pt
5226     }
5227   }
5228   {
5229     \prg_replicate:nn {3}
5230     {
5231       \tl_set:Nx \l__spath_tmpa_tl {\tl_tail:N \l__spath_tmpa_tl}
5232     }
5233     \int_decr:N \l__spath_tmpa_int
5234   }
5235   \tl_if_empty:NF \l__spath_tmpa_tl
5236   {
5237     \tl_reverse:N \l__spath_tmpa_tl
5238     \dim_set:Nn \l__spath_move_x_dim {\tl_item:Nn \l__spath_tmpa_tl {2}}
5239     \dim_set:Nn \l__spath_move_y_dim {\tl_item:Nn \l__spath_tmpa_tl {1}}
5240     \tl_reverse:N \l__spath_tmpa_tl
5241   }
5242   \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpa_tl
5243 }
5244
5245 \tl_set:Nx \l__spath_tmpa_tl {\seq_use:Nn \l__spath_tmpb_seq {} }
5246 \spath_components_to_seq:N \l__spath_tmpb_seq \l__spath_tmpa_tl
5247
5248
5249 \spath_initialpoint:Nn \l__spath_tmpa_tl {#1}
5250 \spath_finalpoint:Nn \l__spath_tmpb_tl {#1}
5251
5252 \bool_if:nT
5253 {

```

```

5254 \dim_compare_p:n
5255 {
5256   \dim_abs:n
5257   {
5258     \tl_item:Nn \l__spath_tmpa_tl {1} - \tl_item:Nn \l__spath_tmpb_tl {1}
5259   }
5260   <
5261   0.01pt
5262 }
5263 &&
5264 \dim_compare_p:n
5265 {
5266   \dim_abs:n
5267   {
5268     \tl_item:Nn \l__spath_tmpa_tl {2} - \tl_item:Nn \l__spath_tmpb_tl {2}
5269   }
5270   <
5271   0.01pt
5272 }
5273 }
5274 {
5275   \int_compare:nTF
5276   {
5277     \seq_count:N \l__spath_tmpb_seq > 1
5278   }
5279   {
5280     \seq_pop_left:NN \l__spath_tmpb_seq \l__spath_tmpb_tl
5281
5282     \prg_replicate:nn {3}
5283     {
5284       \tl_set:Nx \l__spath_tmpb_tl {\tl_tail:N \l__spath_tmpb_tl}
5285     }
5286     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpb_tl
5287   }
5288   {
5289     \tl_set:NV \l__spath_tmpb_tl \c_spath_closepath_tl
5290     \tl_put_right:Nx \l__spath_tmpb_tl
5291     {
5292       { \tl_item:Nn \l__spath_tmpa_tl {1} }
5293       { \tl_item:Nn \l__spath_tmpa_tl {2} }
5294     }
5295     \seq_put_right:NV \l__spath_tmpb_seq \l__spath_tmpb_tl
5296   }
5297 }
5298
5299 \tl_gset:Nx \g__spath_output_tl {\seq_use:Nn \l__spath_tmpb_seq {}}
5300 \group_end:
5301 }
5302 \cs_new_protected_nopar:Npn \spath_spot_weld_components:Nn #1#2
5303 {
5304   \__spath_spot_weld_components:n {#2}
5305   \tl_set_eq:NN #1 \g__spath_output_tl
5306   \tl_gclear:N \g__spath_output_tl
5307 }

```

```

5308 \cs_generate_variant:Nn \spath_spot_weld_components:Nn {NV, cV, cn}
5309 \cs_new_protected_nopar:Npn \spath_spot_weld_components:N #1
5310 {
5311   \spath_spot_weld_components:NV #1#1
5312 }
5313 \cs_generate_variant:Nn \spath_spot_weld_components:N {c}
5314 \cs_new_protected_nopar:Npn \spath_spot_gweld_components:Nn #1#2
5315 {
5316   \__spath_spot_weld_components:n {#2}
5317   \tl_gset_eq:NN #1 \g__spath_output_tl
5318   \tl_gclear:N \g__spath_output_tl
5319 }
5320 \cs_generate_variant:Nn \spath_spot_gweld_components:Nn {NV, cV, cn}
5321 \cs_new_protected_nopar:Npn \spath_spot_gweld_components:N #1
5322 {
5323   \spath_spot_gweld_components:NV #1#1
5324 }
5325 \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.

```

5326 \cs_new_protected_nopar:Npn \__spath_convert_to_svg:n #1
5327 {
5328   \group_begin:
5329   \tl_clear:N \l__spath_tmpa_tl
5330   \tl_put_right:Nn \l__spath_tmpa_tl
5331   {
5332     <?xml~ version="1.0"~ standalone="no"?>
5333     \iow_newline:
5334     <!DOCTYPE~ svg~ PUBLIC~ "-//W3C//DTD SVG 1.1//EN"~
5335     "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
5336     \iow_newline:
5337     <svg~ xmlns="http://www.w3.org/2000/svg"~ version="1.1"~viewBox="
5338   }
5339
5340   \spath_minbb:Nn \l__spath_tmpb_tl {#1}
5341   \spath_maxbb:Nn \l__spath_tmpe_tl {#1}
5342   \tl_put_right:Nx \l__spath_tmpa_tl
5343   {
5344     \dim_to_decimal:n
5345     {
5346       \tl_item:Nn \l__spath_tmpb_tl {1} - 10pt
5347     }
5348     \exp_not:n {~}
5349     \dim_to_decimal:n
5350     {
5351       \tl_item:Nn \l__spath_tmpb_tl {2} - 10pt
5352     }
5353     \exp_not:n {~}
5354     \dim_to_decimal:n
5355     {

```

```

5356     \tl_item:Nn \l__spath_tmpc_tl {1}
5357     -
5358     \tl_item:Nn \l__spath_tmpb_tl {1}
5359     + 20pt
5360   }
5361   \exp_not:n {~}
5362   \dim_to_decimal:n
5363   {
5364     \tl_item:Nn \l__spath_tmpc_tl {2}
5365     -
5366     \tl_item:Nn \l__spath_tmpb_tl {2}
5367     + 20pt
5368   }
5369 }
5370
5371 \tl_put_right:Nn \l__spath_tmpa_tl
5372 {
5373   ">
5374   \iow_newline:
5375   <path~ d="
5376 }
5377 \tl_set:Nn \l__spath_tmpc_tl {use:n}
5378 \tl_map_inline:nn {#1}
5379 {
5380   \tl_set:Nn \l__spath_tmpb_tl {##1}
5381   \token_case_meaning:NnF \l__spath_tmpb_tl
5382   {
5383     \c_spath_moveto_tl
5384     {
5385       \tl_put_right:Nn \l__spath_tmpa_tl {M~}
5386       \tl_set:Nn \l__spath_tmpc_tl {use:n}
5387     }
5388     \c_spath_lineto_tl
5389     {
5390       \tl_put_right:Nn \l__spath_tmpa_tl {L~}
5391       \tl_set:Nn \l__spath_tmpc_tl {use:n}
5392     }
5393     \c_spath_closepath_tl
5394     {
5395       \tl_put_right:Nn \l__spath_tmpa_tl {Z~}
5396       \tl_set:Nn \l__spath_tmpc_tl {use_none:n}
5397     }
5398     \c_spath_curvetoa_tl
5399     {
5400       \tl_put_right:Nn \l__spath_tmpa_tl {C~}
5401       \tl_set:Nn \l__spath_tmpc_tl {use:n}
5402     }
5403     \c_spath_curvetob_tl {
5404       \tl_set:Nn \l__spath_tmpc_tl {use:n}
5405     }
5406     \c_spath_curveto_tl {
5407       \tl_set:Nn \l__spath_tmpc_tl {use:n}
5408     }
5409   }

```

```

5410 {
5411   \tl_put_right:Nx
5412   \l__spath_tmpa_tl
5413   {\use:c { \l__spath_tmpe_tl } {\dim_to_decimal:n {##1}} ~}
5414 }
5415 }
5416 \tl_put_right:Nn \l__spath_tmpa_tl
5417 {
5418   "~ fill="none"~ stroke="black"~ />
5419   \iow_newline:
5420   </svg>
5421   \iow_newline:
5422 }
5423 \tl_gset_eq:NN \g__spath_output_tl \l__spath_tmpa_tl
5424 \group_end:
5425 }
5426 \cs_new_protected_nopar:Npn \spath_convert_to_svg:Nn #1#2
5427 {
5428   \__spath_convert_to_svg:n {#2}
5429   \tl_set_eq:NN #1 \g__spath_output_tl
5430   \tl_gclear:N \g__spath_output_tl
5431 }
5432 \cs_new_protected_nopar:Npn \spath_gconvert_to_svg:Nn #1#2
5433 {
5434   \__spath_convert_to_svg:n {#2}
5435   \tl_gset_eq:NN #1 \g__spath_output_tl
5436   \tl_gclear:N \g__spath_output_tl
5437 }

```

(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.

```

5438 \iow_new:N \g__spath_stream
5439 \cs_new_protected_nopar:Npn \spath_export_to_svg:nn #1#2
5440 {
5441   \group_begin:
5442   \spath_convert_to_svg:Nn \l__spath_tmpa_tl {#2}
5443   \iow_open:Nn \g__spath_stream {#1 .svg}
5444   \iow_now:Nx \g__spath_stream
5445   {
5446     \tl_use:N \l__spath_tmpa_tl
5447   }
5448   \iow_close:N \g__spath_stream
5449   \group_end:
5450 }
5451 \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.

```

5452 \cs_new_protected_nopar:Npn \spath_show:n #1
5453 {
5454   \int_step_inline:nnnn {1} {3} {\tl_count:n {#1}}
5455   {
5456     \iow_term:x {

```

```

5457         \tl_item:nn {#1} {##1}
5458         {\tl_item:nn {#1} {##1+1}}
5459         {\tl_item:nn {#1} {##1+2}}
5460     }
5461 }
5462 }
5463 \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
5464 \cs_new_protected_nopar:Npn \spath_get_current_path:N #1
5465 {
5466     \pgfsyssoftpath@getcurrentpath #1
5467 }
5468 \cs_generate_variant:Nn \spath_get_current_path:N {c}
5469 \cs_new_protected_nopar:Npn \spath_gget_current_path:N #1
5470 {
5471     \pgfsyssoftpath@getcurrentpath #1
5472     \tl_gset_eq:NN #1 #1
5473 }
5474 \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.

```

5475 \cs_new_protected_nopar:Npn \spath_protocol_path:n #1
5476 {
5477     \group_begin:
5478     \spath_minbb:Nn \l__spath_tmpa_tl {#1}
5479     \dim_set:Nn \l__spath_tmpa_dim {\tl_item:Nn \l__spath_tmpa_tl {1}}
5480     \dim_set:Nn \l__spath_tmpb_dim {\tl_item:Nn \l__spath_tmpa_tl {2}}
5481     \pgf@protocolsizes\l__spath_tmpa_dim\l__spath_tmpb_dim
5482
5483     \spath_maxbb:Nn \l__spath_tmpa_tl {#1}
5484     \dim_set:Nn \l__spath_tmpa_dim {\tl_item:Nn \l__spath_tmpa_tl {1}}
5485     \dim_set:Nn \l__spath_tmpb_dim {\tl_item:Nn \l__spath_tmpa_tl {2}}
5486     \pgf@protocolsizes\l__spath_tmpa_dim\l__spath_tmpb_dim
5487     \group_end:
5488 }
5489 \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
5490 \cs_new_protected_nopar:Npn \spath_set_current_path:N #1
5491 {
5492     \pgf@resetpathsizes

```

```

5493 \spath_protocol_path:V #1
5494 \pgfsyssoftpath@setcurrentpath #1
5495 }
5496 \cs_new_protected_nopar:Npn \spath_set_current_path:n #1
5497 {
5498   \tl_set:Nn \l__spath_tmpa_tl {#1}
5499   \spath_set_current_path:N \l__spath_tmpa_tl
5500 }
5501 \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.

```

5502 \cs_new_protected_nopar:Npn \spath_use_path:nn #1#2
5503 {
5504   \spath_set_current_path:n {#1}
5505   \pgfusepath{#2}
5506 }

```

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

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

```

5507 \cs_new_protected_nopar:Npn \spath_tikz_path:nn #1#2
5508 {
5509   \tl_if_empty:nF {#2}
5510   {
5511     \path[#1] \pgfextra{
5512       \spath_set_current_path:n {#2}
5513       \tl_put_left:Nn \tikz@preactions {\def\tikz@actions@path{#2}}
5514     };
5515   }
5516 }
5517 \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.

```

5518 \cs_new_protected_nopar:Npn \spath_set_tikz_data:n #1
5519 {
5520   \spath_finalpoint:Nn \l__spath_tmpa_tl {#1}
5521   \tl_set:Nx \l__spath_tmpa_tl
5522   {
5523     \exp_not:c {pgf@x}=\tl_item:Nn \l__spath_tmpa_tl {1} \relax
5524     \exp_not:c {pgf@y}=\tl_item:Nn \l__spath_tmpa_tl {2} \relax
5525   }
5526   \use:c {pgf@process}{%
5527     \tl_use:N \l__spath_tmpa_tl
5528     \pgftransforminvert
5529     \use:c {pgf@pos@transform@glob}
5530   }
5531   \tl_set:Nx \l__spath_tmpa_tl
5532   {
5533     \exp_not:c {tikz@lastx}=\exp_not:c {pgf@x} \relax
5534     \exp_not:c {tikz@lasty}=\exp_not:c {pgf@y} \relax
5535     \exp_not:c {tikz@lastxsaved}=\exp_not:c {pgf@x} \relax

```



```

5536 \exp_not:c {tikz@lastysaved}=\exp_not:c {pgf@y} \relax
5537 }
5538 \tl_use:N \l__spath_tmpa_tl
5539 \spath_finalmovepoint:Nn \l__spath_tmpa_tl {#1}
5540 \bool_if:NT \l_spath_movetorelevant_bool
5541 {
5542   \ifpgfsyssoftpathmovetorelevant%
5543   \tl_gset_eq:cN {pgfsyssoftpath@lastmoveto} \l__spath_tmpa_tl
5544   \fi
5545 }
5546 \tl_set:Nx \l__spath_tmpa_tl
5547 {
5548   \exp_not:c {pgf@x}=\tl_item:Nn \l__spath_tmpa_tl {1} \relax
5549   \exp_not:c {pgf@y}=\tl_item:Nn \l__spath_tmpa_tl {2} \relax
5550 }
5551 \use:c {pgf@process}{%
5552   \tl_use:N \l__spath_tmpa_tl
5553   \pgftransforminvert
5554   \use:c {pgf@pos@transform@glob}
5555 }
5556 \bool_if:NT \l_spath_movetorelevant_bool
5557 {
5558   \dim_if_exist:cT {tikz@lastmovetox}
5559   {
5560     \tl_set:Nx \l__spath_tmpa_tl
5561     {
5562       \exp_not:c {tikz@lastmovetox}=\exp_not:c {pgf@x} \relax
5563       \exp_not:c {tikz@lastmovetoy}=\exp_not:c {pgf@y} \relax
5564     }
5565     \tl_use:N \l__spath_tmpa_tl
5566   }
5567 }
5568 \tl_clear_new:c {tikz@timer}
5569 \tl_set:cn {tikz@timer}
5570 {
5571   \pgftransformreset
5572   \spath_reallength:Nn \l__spath_tmpa_int {#1}
5573   \tl_set_eq:Nc \l__spath_tmpb_tl {tikz@time}
5574   \tl_set:Nx \l__spath_tmpb_tl
5575   {\fp_to_decimal:n {(\l__spath_tmpb_tl) * (\l__spath_tmpa_int)}}
5576   \spath_transformation_at:NnV \l__spath_tmpc_tl {#1} \l__spath_tmpb_tl
5577
5578   \tl_set:Nx \l__spath_tmpa_tl
5579   {
5580     \exp_not:N \pgfpoint
5581     { \tl_item:Nn \l__spath_tmpc_tl {5} }
5582     { \tl_item:Nn \l__spath_tmpc_tl {6} }
5583   }
5584   \exp_args:NV \pgftransformshift \l__spath_tmpa_tl
5585
5586   \ifpgfresetnontranslationatime
5587   \pgftransformresetnontranslations
5588   \fi
5589

```

```

5590 \ifpgfslopedatetime
5591
5592 \tl_set:Nx \l__spath_tmpa_tl
5593 {
5594   { \tl_item:Nn \l__spath_tmpc_tl {1} }
5595   { \tl_item:Nn \l__spath_tmpc_tl {2} }
5596   { \tl_item:Nn \l__spath_tmpc_tl {3} }
5597   { \tl_item:Nn \l__spath_tmpc_tl {4} }
5598 }
5599 \ifpgfallowupsidedownatime
5600 \else
5601 \fp_compare:nT { \tl_item:Nn \l__spath_tmpc_tl {4} < 0 }
5602 {
5603   \tl_set:Nx \l__spath_tmpa_tl
5604   {
5605     { \fp_eval:n { - (\tl_item:Nn \l__spath_tmpc_tl {1}) } }
5606     { \fp_eval:n { - (\tl_item:Nn \l__spath_tmpc_tl {2}) } }
5607     { \fp_eval:n { - (\tl_item:Nn \l__spath_tmpc_tl {3}) } }
5608     { \fp_eval:n { - (\tl_item:Nn \l__spath_tmpc_tl {4}) } }
5609   }
5610 }
5611 \fi
5612 \tl_put_right:Nn \l__spath_tmpa_tl {{\pgfpointorigin}}
5613 \exp_last_unbraced:NV \pgftransformcm \l__spath_tmpa_tl
5614 \fi
5615 }
5616 }
5617 \cs_generate_variant:Nn \spath_set_tikz_data:n {V, v}

```

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

4 The TikZ interface

```

5618 <@@=tikzspath>

```

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

```

5619 \RequirePackage{spath3}
5620 \ExplSyntaxOn
5621
5622 \tl_new:N \l__tikzspath_tmpa_tl
5623 \tl_new:N \l__tikzspath_tmpb_tl
5624 \tl_new:N \l__tikzspath_tmpc_tl
5625 \tl_new:N \l__tikzspath_tmpd_tl
5626 \tl_new:N \l__tikzspath_tmpe_tl
5627 \tl_new:N \l__tikzspath_tmpf_tl
5628
5629 \int_new:N \l__tikzspath_tmpa_int
5630 \seq_new:N \l__tikzspath_tmpa_seq
5631 \seq_new:N \l__tikzspath_tmpb_seq
5632 \seq_new:N \l__tikzspath_tmpc_seq
5633 \seq_new:N \l__tikzspath_tmpd_seq
5634
5635 \tl_new:N \l__tikzspath_current_tl

```

```

5636 \tl_new:N \l__tikzspath_reverse_tl
5637 \tl_new:N \l__tikzspath_prefix_tl
5638 \tl_new:N \l__tikzspath_suffix_tl
5639 \tl_new:N \g__tikzspath_smuggle_tl
5640 \tl_new:N \g__tikzspath_output_tl
5641 \tl_new:N \l__tikzspath_check_tl
5642 \clist_new:N \g__tikzspath_output_clist
5643 \seq_new:N \g__tikzspath_tmpa_seq
5644 \seq_new:N \g__tikzspath_tmpb_seq
5645 \seq_new:N \g__tikzspath_output_seq
5646 \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.

```

5647 \msg_new:nnn { spath3 } { missing soft path }
5648 { Soft~ path~ #1~ doesn't~ exist~ \msg_line_context:}
5649 \msg_new:nnnn { spath3 } { empty soft path }
5650 { Soft~ path~ #1~ is~ empty~ \msg_line_context:}
5651 {If~ it~ was~ defined~ inside~ a~ group,~ try~ using~ "save~ global". }
5652 \msg_new:nnn { spath3 } { load intersections }
5653 { You~ need~ to~ load~ the~ "intersections"~ library~
5654   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.

```

5655 \tl_set:Nn \l__tikzspath_prefix_tl {tikz@intersect@path@name@}
5656 \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.

```

5657 \tl_new:N \g__tikzspath_tikzfinish_tl
5658 \tl_new:N \l__tikzspath_tikzfinish_outside_tl
5659 \cs_new_protected_nopar:Npn \spath_at_end_of_path:
5660 {
5661   \tl_use:N \g__tikzspath_tikzfinish_tl
5662   \tl_gclear:N \g__tikzspath_tikzfinish_tl
5663 }
5664 \tl_put_right:Nn \tikz@finish {\spath_at_end_of_path:}
5665
5666 \tikzset{
5667   every~ scope/.append~ style={
5668     execute~ at~ begin~ scope={
5669       \tl_set_eq:NN \l__tikzspath_tikzfinish_outside_tl \g__tikzspath_tikzfinish_tl
5670     },
5671     execute~ at~ end~ scope={
5672       \tl_use:N \g__tikzspath_tikzfinish_tl

```

```

5673         \tl_gclear:N \g__tikzspath_tikzfinish_tl
5674         \tl_gset_eq:NN \g__tikzspath_tikzfinish_tl \l__tikzspath_tikzfinish_outside_tl
5675     },
5676 },
5677 }

```

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.

```

5678 \tl_new:N \l__tikzspath_tikzpath_finish_tl
5679
5680 \cs_new_protected_nopar:Npn \__tikzspath_at_end_of_path_construction:
5681 {
5682     \tl_use:N \l__tikzspath_tikzpath_finish_tl
5683     \tl_clear:N \l__tikzspath_tikzpath_finish_tl
5684 }
5685
5686 \tl_put_left:Nn \tikz@finish {\__tikzspath_at_end_of_path_construction:}

```

Code for saving a path

```

5687 \cs_new_protected_nopar:Npn \spath_save_path:Nn #1#2
5688 {
5689     \tl_if_empty:NF \g__tikzspath_tikzfinish_tl
5690     {
5691         \tl_use:N \g__tikzspath_tikzfinish_tl
5692     }
5693     \tl_gput_right:Nn \g__tikzspath_tikzfinish_tl
5694     {
5695         \tl_clear_new:N #1
5696         \tl_set:Nn #1 {#2}
5697     }
5698 }
5699 \cs_generate_variant:Nn \spath_save_path:Nn {cn, NV, cV}
5700
5701 \cs_new_protected_nopar:Npn \spath_gsave_path:Nn #1#2
5702 {
5703     \tl_gput_right:Nn \g__tikzspath_tikzfinish_tl
5704     {
5705         \tl_gclear_new:N #1
5706         \tl_gset:Nn #1 {#2}
5707     }
5708 }
5709 \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.

```

5710 \cs_new_protected_nopar:Npn \__tikzspath_process_tikz_point:Nn #1#2
5711 {
5712     \group_begin:
5713     \use:c {tikz@scan@one@point} \use:n #2 \scan_stop:
5714     \tl_gset:Nx \g__tikzspath_output_tl
5715     {
5716         { \dim_use:c {pgf@x} }
5717         { \dim_use:c {pgf@y} }
5718     }
5719     \group_end:

```

```

5720 \tl_set_eq:NN #1 \g__tikzspath_output_tl
5721 \tl_gclear:N \g__tikzspath_output_tl
5722 }

```

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

`__tikzspath_tikzset:n` Wrapper around `\tikzset` for expansion.

```

5723 \cs_set_eq:NN \__tikzspath_tikzset:n \tikzset
5724 \cs_generate_variant:Nn \__tikzspath_tikzset:n {V, v}

```

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

`s:nnnn__tikzspath_check_three_paths:nnnn`

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

```

5725 \cs_new_protected_nopar:Npn \__tikzspath_check_path:nnn #1#2#3
5726 {
5727   \tl_set:Nn \l__tikzspath_check_tl {#3}
5728   \tl_if_exist:cTF {\__tikzspath_path_name:n {#2}}
5729   {
5730     \tl_if_empty:cTF {\__tikzspath_path_name:n {#2}}
5731     {
5732       \msg_warning:nnn { spath3 } { empty soft path } { #2 }
5733     }
5734     {
5735       \tl_set:Nn \l__tikzspath_check_tl {
5736         #1 {\__tikzspath_path_name:n {#2}}
5737       }
5738     }
5739   }
5740   {
5741     \msg_warning:nnx { spath3 } { missing soft path } { #2 }
5742   }
5743   \tl_use:N \l__tikzspath_check_tl
5744 }
5745 \cs_new_protected_nopar:Npn \__tikzspath_check_two_paths:nnnn #1#2#3#4
5746 {
5747   \__tikzspath_check_path:nnn {
5748     \__tikzspath_check_path:nnn {#1}{#2}{#4}
5749   }{#3}{#4}
5750 }
5751 \cs_new_protected_nopar:Npn \__tikzspath_check_three_paths:nnnnn #1#2#3#4#5
5752 {
5753   \__tikzspath_check_path:nnn {
5754     \__tikzspath_check_path:nnn {
5755       \__tikzspath_check_path:nnn {#1}{#2}{#5}
5756     }{#3}{#5}
5757   }{#4}{#5}
5758 }
5759 \cs_generate_variant:Nn \__tikzspath_check_path:nnn {nVn}
5760 \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:nnnnn`.)

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.

```
5761 \cs_new_protected_nopar:Npn \__tikzspath_maybe_current_path:nn #1#2
5762 {
5763   \tl_if_eq:nnT {#2} {current}
5764   {
5765     \spath_get_current_path:c {\__tikzspath_path_name:n {#2}}
5766   }
5767   #1 {#2}
5768 }
5769 \cs_new_protected_nopar:Npn \__tikzspath_maybe_current_path_reuse:nnn #1#2#3
5770 {
5771   \bool_set_true:N \l_spath_movetorelevant_bool
5772   \tl_if_eq:nnT {#2} {current}
5773   {
5774     \spath_get_current_path:c {\__tikzspath_path_name:n {#2}}
5775   }
5776   #1 {#2} #3
5777   \tl_if_eq:nnT {#2} {current}
5778   {
5779     \tl_if_empty:cF {\__tikzspath_path_name:n {#2}}
5780     {
5781       \spath_set_current_path:c {\__tikzspath_path_name:n {#2}}
5782       \spath_set_tikz_data:v {\__tikzspath_path_name:n {#2}}
5783     }
5784   }
5785 }
5786 \cs_new_protected_nopar:Npn \__tikzspath_maybe_current_two_paths_reuse_both:nnnn #1#2#3#4
5787 {
5788   \bool_set_true:N \l_spath_movetorelevant_bool
5789   \tl_if_eq:nnT {#2} {current}
5790   {
5791     \spath_get_current_path:c {\__tikzspath_path_name:n {#2}}
5792   }
5793   \tl_if_eq:nnT {#3} {current}
5794   {
5795     \spath_get_current_path:c {\__tikzspath_path_name:n {#3}}
5796   }
5797   #1 {#2} {#3} #4
5798   \tl_if_eq:nnT {#2} {current}
5799   {
5800     \tl_if_empty:cF {\__tikzspath_path_name:n {#2}}
5801     {
5802       \spath_set_current_path:c {\__tikzspath_path_name:n {#2}}
5803       \spath_set_tikz_data:v {\__tikzspath_path_name:n {#2}}
5804     }
5805   }
5806   \tl_if_eq:nnT {#3} {current}
5807   {
5808     \tl_if_empty:cF {\__tikzspath_path_name:n {#3}}
5809     {
5810       \spath_set_current_path:c {\__tikzspath_path_name:n {#3}}
5811       \spath_set_tikz_data:v {\__tikzspath_path_name:n {#3}}
5812     }
5813   }
5814 }
```

```

5813 }
5814 }
5815 \cs_new_protected_nopar:Npn \__tikzspath_maybe_current_two_paths_reuse_first:nnnn #1#2#3#4
5816 {
5817   \bool_set_true:N \l_spath_movetorelevant_bool
5818   \tl_if_eq:nnT {#2} {current}
5819   {
5820     \spath_get_current_path:c {\__tikzspath_path_name:n {#2}}
5821   }
5822   \tl_if_eq:nnT {#3} {current}
5823   {
5824     \spath_get_current_path:c {\__tikzspath_path_name:n {#3}}
5825   }
5826   #1 {#2} {#3} #4
5827   \tl_if_eq:nnT {#2} {current}
5828   {
5829     \tl_if_empty:cF {\__tikzspath_path_name:n {#2}}
5830     {
5831       \spath_set_current_path:c {\__tikzspath_path_name:n {#2}}
5832       \spath_set_tikz_data:v {\__tikzspath_path_name:n {#2}}
5833     }
5834   }
5835 }
5836 \cs_new_protected_nopar:Npn \__tikzspath_maybe_current_two_paths_reuse_second:nnnn #1#2#3#4
5837 {
5838   \bool_set_true:N \l_spath_movetorelevant_bool
5839   \tl_if_eq:nnT {#2} {current}
5840   {
5841     \spath_get_current_path:c {\__tikzspath_path_name:n {#2}}
5842   }
5843   \tl_if_eq:nnT {#3} {current}
5844   {
5845     \spath_get_current_path:c {\__tikzspath_path_name:n {#3}}
5846   }
5847   #1 {#2} {#3} #4
5848   \tl_if_eq:nnT {#3} {current}
5849   {
5850     \tl_if_empty:cF {\__tikzspath_path_name:n {#3}}
5851     {
5852       \spath_set_current_path:c {\__tikzspath_path_name:n {#3}}
5853       \spath_set_tikz_data:v {\__tikzspath_path_name:n {#3}}
5854     }
5855   }
5856 }
5857 \cs_generate_variant:Nn \__tikzspath_maybe_current_path:nn {nV}
5858 \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.

```

5859 \cs_new_protected_nopar:Npn \__tikzspath_seq_from_foreach:Nnn #1#2#3
5860 {

```

```

5861 \group_begin:
5862 \seq_gclear:N \g__tikzspath_output_seq
5863
5864 \tl_if_empty:nTF {#3}
5865 {
5866   \int_step_inline:nnnn {1}{1} {#2}
5867   {
5868     \seq_gput_right:Nn \g__tikzspath_output_seq {##1}
5869   }
5870 }
5871 {
5872   \foreach \l__tikzspath_tmpa_tl in {#3}
5873   {
5874     \int_compare:nTF { \l__tikzspath_tmpa_tl > 0 }
5875     {
5876       \seq_gput_right:NV \g__tikzspath_output_seq \l__tikzspath_tmpa_tl
5877     }
5878     {
5879       \seq_gput_right:Nx \g__tikzspath_output_seq
5880       {\int_eval:n {#2 - \l__tikzspath_tmpa_tl}}
5881     }
5882   }
5883   \seq_gsort:Nn \g__tikzspath_output_seq
5884   {
5885     \int_compare:nNnTF {##1} < {##2}
5886     { \sort_return_same: }
5887     { \sort_return_swapped: }
5888   }
5889 }
5890 \group_end:
5891 \seq_set_eq:NN #1 \g__tikzspath_output_seq
5892 \seq_gclear:N \g__tikzspath_output_seq
5893 }
5894 \cs_generate_variant:Nn \__tikzspath_seq_from_foreach:Nnn {NVV, NVn}
5895 %

```

(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.

```

5896 \cs_new:Npn \__tikzspath_path_name:n #1
5897 {
5898   \tl_use:N \l__tikzspath_prefix_tl
5899   #1
5900   \tl_use:N \l__tikzspath_suffix_tl
5901 }
5902 \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.

```

5903 \bool_new:N \l__tikzspath_reverse_bool
5904 \bool_new:N \l__tikzspath_weld_bool
5905 \bool_new:N \l__tikzspath_move_bool
5906 \bool_new:N \l__tikzspath_global_bool

```



```

5907 \bool_new:N \l__tikzspath_current_transformation_bool
5908 \tl_new:N \l__tikzspath_joinpath_tl
5909 \tl_new:N \l__tikzspath_transformation_tl
5910
5911 \cs_new_protected_nopar:Npn \__tikzspath_set_bool:Nn #1#2
5912 {
5913   \tl_if_eq:nnTF {#2}{false}
5914   {
5915     \bool_set_false:N #1
5916   }
5917   {
5918     \bool_set_true:N #1
5919   }
5920 }
5921 \tikzset {
5922   spath/join/.is~ family,
5923   spath/join/.cd,
5924   reverse/.code = {
5925     \__tikzspath_set_bool:Nn \l__tikzspath_reverse_bool {#1}
5926   },
5927   reverse/.default = true,
5928   weld/.code = {
5929     \__tikzspath_set_bool:Nn \l__tikzspath_weld_bool {#1}
5930   },
5931   weld/.default = true,
5932   no~ weld/.code = {
5933     \__tikzspath_set_bool:Nn \l__tikzspath_weld_bool {#1}
5934     \bool_set:Nn \l__tikzspath_weld_bool {! \l__tikzspath_weld_bool}
5935   },
5936   no~ weld/.default = true,
5937   move/.code = {
5938     \__tikzspath_set_bool:Nn \l__tikzspath_move_bool {#1}
5939   },
5940   move/.default = true,
5941   no~ move/.code = {
5942     \__tikzspath_set_bool:Nn \l__tikzspath_move_bool {#1}
5943     \bool_set:Nn \l__tikzspath_move_bool {! \l__tikzspath_move_bool}
5944   },
5945   no~ move/.default = true,
5946   global/.code = {
5947     \__tikzspath_set_bool:Nn \l__tikzspath_global_bool {#1}
5948   },
5949   global/.default = true,
5950   use~ current~ transformation/.code={
5951     \__tikzspath_set_bool:Nn \l__tikzspath_current_transformation_bool {#1}
5952   },
5953   use~ current~ transformation/.default = true,
5954   transform/.store-in=\l__tikzspath_transformation_tl,
5955   .unknown/.code = {
5956     \tl_set_eq:NN \l__tikzspath_joinpath_tl \pgfkeyscurrentname
5957   }
5958 }

```

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).

```
5959 \cs_set_eq:NN \GetComponentOf \clist_item:Nn
```

4.1 Helper Functions

```
\__tikzspath_use_path:n Use a path, possibly manipulating it first.
5960 \cs_new_protected_nopar:Npn \__tikzspath_use_path:n #1
5961 {
5962   \tl_set:Nn \l__tikzspath_joinpath_tl {#1}
5963   \spath_get_current_path:N \l__tikzspath_current_tl
5964
5965   \bool_if:NT \l__tikzspath_reverse_bool
5966   {
5967     \spath_reverse:N \l__tikzspath_joinpath_tl
5968   }
5969
5970   \bool_if:NT \l__tikzspath_current_transformation_bool
5971   {
5972     \pgfgettransform \l__tikzspath_tmpa_tl
5973     \spath_transform:NV
5974     \l__tikzspath_joinpath_tl
5975     \l__tikzspath_tmpa_tl
5976   }
5977
5978   \tl_if_empty:NF \l__tikzspath_transformation_tl
5979   {
5980     \group_begin:
5981     \pgftransformreset
5982     \__tikzspath_tikzset:V \l__tikzspath_transformation_tl
5983     \pgfgettransform \l__tikzspath_tmpa_tl
5984     \tl_gset:Nn \g__tikzspath_smuggle_tl
5985     {
5986       \spath_transform:Nnnnnnn
5987       \l__tikzspath_joinpath_tl
5988     }
5989     \tl_gput_right:NV \g__tikzspath_smuggle_tl \l__tikzspath_tmpa_tl
5990     \group_end:
5991     \tl_use:N \g__tikzspath_smuggle_tl
5992   }
5993
5994   \bool_if:NT \l__tikzspath_move_bool
5995   {
5996     \tl_if_empty:NTF \l__tikzspath_current_tl
5997     {
5998       \tl_set:Nn \l__tikzspath_tmpc_tl { {0pt} {0pt} }
5999     }
6000     {
6001       \spath_finalpoint:NV
6002       \l__tikzspath_tmpc_tl
6003       \l__tikzspath_current_tl
6004     }
6005     \spath_translate_to:NV \l__tikzspath_joinpath_tl \l__tikzspath_tmpc_tl
```

```

6006 }
6007
6008 \tl_if_empty:NTF \l__tikzspath_current_tl
6009 {
6010   \tl_if_empty:NTF \l__tikzspath_joinpath_tl
6011   {
6012     \tl_set_eq:NN \l__tikzspath_current_tl \c_spath_moveto_tl
6013     \tl_put_right:Nn \l__tikzspath_current_tl {{0pt}}{0pt}}
6014   }
6015   {
6016     \tl_set_eq:NN \l__tikzspath_current_tl \l__tikzspath_joinpath_tl
6017   }
6018 }
6019 {
6020
6021   \tl_clear:N \l__tikzspath_tmpa_tl
6022   \tl_set:Nn \l__tikzspath_tmpa_tl {spath_}
6023
6024   \tl_put_right:Nn \l__tikzspath_tmpa_tl {append}
6025
6026   \bool_if:NT \l__tikzspath_weld_bool
6027   {
6028     \tl_put_right:Nn \l__tikzspath_tmpa_tl {_no_move}
6029     \spath_numberofcomponents:NV \l__tikzspath_tmpa_int \l__tikzspath_joinpath_tl
6030     \int_compare:nT {\l__tikzspath_tmpa_int == 1}
6031     {
6032       \bool_set_false:N \l_spath_movetorelevant_bool
6033     }
6034   }
6035   \tl_put_right:Nn \l__tikzspath_tmpa_tl {:NV}
6036
6037   \use:c {\tl_use:N \l__tikzspath_tmpa_tl }
6038   \l__tikzspath_current_tl
6039   \l__tikzspath_joinpath_tl
6040 }
6041
6042 \spath_set_current_path:N \l__tikzspath_current_tl
6043 \spath_set_tikz_data:V \l__tikzspath_joinpath_tl
6044 }
6045 \cs_generate_variant:Nn \__tikzspath_use_path:n {V, v}

```

(End of definition for __tikzspath_use_path:n.)

__tikzspath_join_with:nn

```

6046 \cs_new_protected_nopar:Npn \__tikzspath_join_with:Nn #1#2
6047 {
6048   \tl_set:Nn \l__tikzspath_joinpath_tl {#2}
6049
6050   \bool_if:NT \l__tikzspath_reverse_bool
6051   {
6052     \spath_reverse:N \l__tikzspath_joinpath_tl
6053   }
6054
6055   \tl_if_empty:NF \l__tikzspath_transformation_tl

```

```

6056 {
6057   \group_begin:
6058   \pgftransformreset
6059   \tikzspath_tikzset:V \l__tikzspath_transformation_tl
6060   \pgfgettransform \l__tikzspath_tmpa_tl
6061   \tl_gset:Nn \g__tikzspath_smuggle_tl
6062   {
6063     \spath_transform:Nnnnnnn
6064     \l__tikzspath_joinpath_tl
6065   }
6066   \tl_gput_right:NV \g__tikzspath_smuggle_tl \l__tikzspath_tmpa_tl
6067   \group_end:
6068   \tl_use:N \g__tikzspath_smuggle_tl
6069 }
6070
6071 \bool_if:NT \l__tikzspath_move_bool
6072 {
6073   \spath_finalpoint:NV
6074   \l__tikzspath_tmpc_tl
6075   #1
6076   \spath_translate_to:NV \l__tikzspath_joinpath_tl \l__tikzspath_tmpc_tl
6077 }
6078
6079 \tl_clear:N \l__tikzspath_tmpa_tl
6080 \tl_set:Nn \l__tikzspath_tmpa_tl {spath_}
6081
6082 \bool_if:NT \l__tikzspath_global_bool
6083 {
6084   \tl_put_right:Nn \l__tikzspath_tmpa_tl {g}
6085 }
6086
6087 \tl_put_right:Nn \l__tikzspath_tmpa_tl {append}
6088
6089 \bool_if:NT \l__tikzspath_weld_bool
6090 {
6091   \tl_put_right:Nn \l__tikzspath_tmpa_tl {no_move}
6092 }
6093 \tl_put_right:Nn \l__tikzspath_tmpa_tl {:NV}
6094
6095 \cs_if_exist:cF {\tl_use:N \l__tikzspath_tmpa_tl}
6096 {
6097   \tl_show:N \l__tikzspath_tmpa_tl
6098 }
6099
6100 \use:c {\tl_use:N \l__tikzspath_tmpa_tl } #1
6101 \l__tikzspath_joinpath_tl
6102 }
6103 \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.

```

6104 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with_aux:nnn #1#2#3
6105 {

```

```

6106 \group_begin:
6107 \tl_set:Nn \l__tikzspath_tmpc_tl {#1}
6108 \tl_if_empty:nT {#3}
6109 {
6110   \spath_spot_weld_components:N \l__tikzspath_tmpc_tl
6111 }
6112
6113 \spath_numberofcomponents:NV \l__tikzspath_tmpa_int \l__tikzspath_tmpc_tl
6114 \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpb_seq \l__tikzspath_tmpa_int {#3}
6115
6116 \spath_components_to_seq:NV \l__tikzspath_tmpa_seq \l__tikzspath_tmpc_tl
6117
6118 \seq_pop_left:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6119 \seq_pop_left:NN \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6120
6121 \seq_map_indexed_inline:Nn \l__tikzspath_tmpa_seq
6122 {
6123   \int_compare:nTF
6124   {
6125     ##1 == \l__tikzspath_tmpb_tl
6126   }
6127   {
6128     \seq_pop_left:NNF \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6129     {
6130       \tl_set:Nn \l__tikzspath_tmpb_tl {-1}
6131     }
6132     \spath_splice_between:Nnn \l__tikzspath_tmpa_tl {#2} {##2}
6133   }
6134   {
6135     \tl_put_right:Nn \l__tikzspath_tmpa_tl {##2}
6136   }
6137 }
6138 \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpa_tl
6139 \group_end:
6140 }
6141 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with:Nnnn #1#2#3#4
6142 {
6143   \__tikzspath_join_components_with_aux:nnn {#2}{#3}{#4}
6144   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6145   \tl_gclear:N \g__tikzspath_output_tl
6146 }
6147 \cs_generate_variant:Nn \__tikzspath_join_components_with:Nnnn {NVnn}
6148 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with:Nnn #1#2#3
6149 {
6150   \__tikzspath_join_components_with:NVnn #1#1{#2}{#3}
6151 }
6152 \cs_generate_variant:Nn \__tikzspath_join_components_with:Nnn {cvV}
6153 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_with:Nnnn #1#2#3#4
6154 {
6155   \__tikzspath_join_components_with_aux:nnn {#2}{#3}{#4}
6156   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6157   \tl_gclear:N \g__tikzspath_output_tl
6158 }
6159 \cs_generate_variant:Nn \__tikzspath_gjoin_components_with:Nnnn {NVnn}

```

```

6160 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_with:Nnn #1#2#3
6161 {
6162   \__tikzspath_gjoin_components_with:NVnn #1#1{#2}{#3}
6163 }
6164 \cs_generate_variant:Nn \__tikzspath_gjoin_components_with:Nnn {cvV}
6165 \cs_new_protected_nopar:Npn \__tikzspath_join_components_upright_with_aux:nnn #1#2#3
6166 {
6167   \group_begin:
6168   \tl_set:Nn \l__tikzspath_tmpc_tl {#1}
6169   \tl_if_empty:nT {#3}
6170   {
6171     \spath_spot_weld_components:N \l__tikzspath_tmpc_tl
6172   }
6173   \spath_numberofcomponents:NV \l__tikzspath_tmpa_int \l__tikzspath_tmpc_tl
6174   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpb_seq \l__tikzspath_tmpa_int {#3}
6175   \spath_components_to_seq:NV \l__tikzspath_tmpa_seq \l__tikzspath_tmpc_tl
6176   \seq_pop_left:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6177   \seq_pop_left:NN \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6178   \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6179   \spath_transform:NVnnnnnn \l__tikzspath_tmpd_tl \l__tikzspath_tmpc_tl {1}{0}{0}{-
6180   1}{Opt}{Opt}
6181
6182   \seq_map_indexed_inline:Nn \l__tikzspath_tmpa_seq
6183   {
6184     \int_compare:nTF
6185     {
6186       ##1 == \l__tikzspath_tmpb_tl
6187     }
6188     {
6189       \seq_pop_left:NNF \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6190       {
6191         \tl_set:Nn \l__tikzspath_tmpb_tl {-1}
6192       }
6193     }
6194     \spath_finalpoint:NV \l__tikzspath_tmpe_tl \l__tikzspath_tmpa_tl
6195     \spath_initialpoint:Nn \l__tikzspath_tmpf_tl {##2}
6196
6197     \dim_compare:nTF
6198     {
6199       \tl_item:Nn \l__tikzspath_tmpe_tl {1}
6200       >
6201       \tl_item:Nn \l__tikzspath_tmpf_tl {1}
6202     }
6203     {
6204       \spath_splice_between:NVn
6205       \l__tikzspath_tmpa_tl
6206       \l__tikzspath_tmpd_tl
6207       {##2}
6208     }
6209   }
6210 }
6211
6212 {

```

```

6213         \spath_splice_between:NVn
6214         \l__tikzspath_tmpa_tl
6215         \l__tikzspath_tmpc_tl
6216         {##2}
6217     }
6218 }
6219 {
6220     \tl_put_right:Nn \l__tikzspath_tmpa_tl {##2}
6221 }
6222 }
6223 \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpa_tl
6224 \group_end:
6225 }
6226 \cs_new_protected_nopar:Npn \__tikzspath_join_components_upright_with:Nnnn #1#2#3#4
6227 {
6228     \__tikzspath_join_components_upright_with_aux:nnn {#2}{#3}{#4}
6229     \tl_set_eq:NN #1 \g__tikzspath_output_tl
6230     \tl_gclear:N \g__tikzspath_output_tl
6231 }
6232 \cs_generate_variant:Nn \__tikzspath_join_components_upright_with:Nnnn {NVnn}
6233 \cs_new_protected_nopar:Npn \__tikzspath_join_components_upright_with:Nnn #1#2#3
6234 {
6235     \__tikzspath_join_components_upright_with:NVnn #1#1{#2}{#3}
6236 }
6237 \cs_generate_variant:Nn \__tikzspath_join_components_upright_with:Nnn {cvV}
6238 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_upright_with:Nnnn #1#2#3#4
6239 {
6240     \__tikzspath_join_components_upright_with_aux:nnn {#2}{#3}{#4}
6241     \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6242     \tl_gclear:N \g__tikzspath_output_tl
6243 }
6244 \cs_generate_variant:Nn \__tikzspath_gjoin_components_upright_with:Nnnn {NVnn}
6245 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_upright_with:Nnn #1#2#3
6246 {
6247     \__tikzspath_gjoin_components_upright_with:NVnn #1#1{#2}{#3}
6248 }
6249 \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.

```

6250 \cs_new_protected_nopar:Npn \__tikzspath_get_components_aux:n #1
6251 {
6252     \clist_gclear_new:N \g__tikzspath_output_clist
6253     \spath_components_to_seq:Nn \l__tikzspath_tmpa_seq {#1}
6254
6255     \seq_map_inline:Nn \l__tikzspath_tmpa_seq
6256     {
6257         \spath_anonymous:N \l__tikzspath_tmpa_tl
6258         \tl_new:c {\__tikzspath_path_name:V \l__tikzspath_tmpa_tl}
6259         \tl_set:cn {\__tikzspath_path_name:V \l__tikzspath_tmpa_tl} {##1}
6260         \clist_gput_right:NV \g__tikzspath_output_clist \l__tikzspath_tmpa_tl
6261     }

```

```

6262 }
6263 \cs_new_protected_nopar:Npn \__tikzspath_get_components:Nn #1#2
6264 {
6265   \clist_clear_new:N #1
6266   \__tikzspath_get_components_aux:n {#2}
6267   \clist_set_eq:NN #1 \g__tikzspath_output_clist
6268   \clist_gclear:N \g__tikzspath_output_clist
6269 }
6270 \cs_generate_variant:Nn \__tikzspath_get_components:Nn {NV, Nv}
6271 \cs_new_protected_nopar:Npn \__tikzspath_gget_components:Nn #1#2
6272 {
6273   \clist_gclear_new:N #1
6274   \__tikzspath_get_components_aux:n {#2}
6275   \clist_gset_eq:NN #1 \g__tikzspath_output_clist
6276   \clist_gclear:N \g__tikzspath_output_clist
6277 }
6278 \cs_generate_variant:Nn \__tikzspath_gget_components:Nn {NV, Nv}

```

(End of definition for __tikzspath_get_components:Nn.)

__tikzspath_render_components:n

```

6279 \cs_new_protected_nopar:Npn \__tikzspath_render_components:nn #1#2
6280 {
6281   \group_begin:
6282   \spath_components_to_seq:Nn \l__tikzspath_tmpa_seq {#2}
6283   \seq_map_indexed_inline:Nn \l__tikzspath_tmpa_seq
6284   {
6285     \spath_tikz_path:nn
6286     {
6287       every~ spath~ component/.try,
6288       spath ~component~ ##1/.try,
6289       spath ~component/.try={##1},
6290       every~ #1~ component/.try,
6291       #1 ~component~ ##1/.try,
6292       #1 ~component/.try={##1},
6293     }
6294     {
6295       ##2
6296     }
6297   }
6298   \group_end:
6299 }
6300 \cs_generate_variant:Nn \__tikzspath_render_components:nn {nv}

```

(End of definition for __tikzspath_render_components:n.)

__tikzspath_insert_gaps_after_components:nn

```

6301 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_components_aux:nnn #1#2#3
6302 {
6303   \group_begin:
6304   \spath_numberofcomponents:Nn \l__tikzspath_tmpa_int {#1}
6305   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_int {#3}
6306
6307   \tl_if_empty:nT {#3}
6308   {

```



```

6309 \seq_pop_right:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6310 }
6311
6312 \seq_clear:N \l__tikzspath_tmpb_seq
6313 \seq_map_inline:Nn \l__tikzspath_tmpa_seq {
6314 \seq_put_right:Nx
6315 \l__tikzspath_tmpb_seq
6316 {\int_eval:n
6317 {
6318 \int_mod:nn { ##1 }{ \l__tikzspath_tmpa_int } + 1
6319 }
6320 }
6321 }
6322
6323 \spath_components_to_seq:Nn \l__tikzspath_tmpc_seq {##1}
6324
6325 \seq_clear:N \l__tikzspath_tmpd_seq
6326 \seq_map_indexed_inline:Nn \l__tikzspath_tmpc_seq
6327 {
6328 \tl_set:Nn \l__tikzspath_tmpa_tl {##2}
6329 \seq_if_in:NnT \l__tikzspath_tmpa_seq {##1}
6330 {
6331 \spath_shorten_at_end:Nn \l__tikzspath_tmpa_tl {(##2)/2}
6332 }
6333 \seq_if_in:NnT \l__tikzspath_tmpb_seq {##1}
6334 {
6335 \spath_shorten_at_start:Nn \l__tikzspath_tmpa_tl {(##2)/2}
6336 }
6337 \seq_put_right:NV \l__tikzspath_tmpd_seq \l__tikzspath_tmpa_tl
6338 }
6339 \tl_gset:Nx \g__tikzspath_output_tl {\seq_use:Nn \l__tikzspath_tmpd_seq {} }
6340 \group_end:
6341 }
6342 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_components:Nnnn #1#2#3#4
6343 {
6344 \__tikzspath_insert_gaps_after_components_aux:nnn {##2}{##3}{##4}
6345 \tl_set_eq:NN #1 \g__tikzspath_output_tl
6346 \tl_gclear:N \g__tikzspath_output_tl
6347 }
6348 \cs_generate_variant:Nn \__tikzspath_insert_gaps_after_components:Nnnn {NVnn}
6349 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_components:Nnn #1#2#3
6350 {
6351 \__tikzspath_insert_gaps_after_components:NVnn #1#1{##2}{##3}
6352 }
6353 \cs_generate_variant:Nn \__tikzspath_insert_gaps_after_components:Nnn {cnn, cVV}
6354 \cs_new_protected_nopar:Npn \__tikzspath_ginsert_gaps_after_components:Nnnn #1#2#3#4
6355 {
6356 \__tikzspath_insert_gaps_after_components_aux:nnn {##2}{##3}{##4}
6357 \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6358 \tl_gclear:N \g__tikzspath_output_tl
6359 }
6360 \cs_generate_variant:Nn \__tikzspath_ginsert_gaps_after_components:Nnnn {NVnn}
6361 \cs_new_protected_nopar:Npn \__tikzspath_ginsert_gaps_after_components:Nnn #1#2#3
6362 {

```

```

6363 \tikzspath_ginsert_gaps_after_components:NVnn #1#1{#2}{#3}
6364 }
6365 \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`

```

6366 \cs_new_protected_nopar:Npn \tikzspath_insert_gaps_after_segments_aux:nnn #1#2#3
6367 {
6368   \group_begin:
6369   \spath_reallength:Nn \l__tikzspath_tmpa_int {#1}
6370   \tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_int {#3}
6371
6372   \tl_if_empty:nT {#3}
6373   {
6374     \seq_pop_right:NN \l__tikzspath_tmpb_seq \l__tikzspath_tmpa_tl
6375   }
6376
6377   \seq_clear:N \l__tikzspath_tmpb_seq
6378   \seq_map_inline:Nn \l__tikzspath_tmpa_seq {
6379     \seq_put_right:Nx
6380     \l__tikzspath_tmpb_seq
6381     {\int_eval:n
6382      {
6383        \int_mod:nn { ##1 }{\l__tikzspath_tmpa_int } + 1
6384      }
6385     }
6386   }
6387
6388   \spath_segments_to_seq:Nn \l__tikzspath_tmpc_seq {#1}
6389
6390   \seq_clear:N \l__tikzspath_tmpd_seq
6391   \seq_map_indexed_inline:Nn \l__tikzspath_tmpc_seq
6392   {
6393     \tl_set:Nn \l__tikzspath_tmpa_tl {##2}
6394     \seq_if_in:NnT \l__tikzspath_tmpa_seq {##1}
6395     {
6396       \spath_shorten_at_end:Nn \l__tikzspath_tmpa_tl {(#2)/2}
6397     }
6398     \seq_if_in:NnT \l__tikzspath_tmpb_seq {##1}
6399     {
6400       \spath_shorten_at_start:Nn \l__tikzspath_tmpa_tl {(#2)/2}
6401     }
6402     \seq_put_right:NV \l__tikzspath_tmpd_seq \l__tikzspath_tmpa_tl
6403   }
6404   \tl_gset:Nx \g__tikzspath_output_tl {\seq_use:Nn \l__tikzspath_tmpd_seq {} }
6405   \group_end:
6406 }
6407 \cs_new_protected_nopar:Npn \tikzspath_insert_gaps_after_segments:Nnnn #1#2#3#4
6408 {
6409   \tikzspath_insert_gaps_after_segments_aux:nnn {#2}{#3}{#4}
6410   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6411   \tl_gclear:N \g__tikzspath_output_tl
6412 }

```

```

6413 \cs_generate_variant:Nn \__tikzspath_insert_gaps_after_segments:Nnnn {NVnn}
6414 \cs_new_protected_nopar:Npn \__tikzspath_insert_gaps_after_segments:Nnn #1#2#3
6415 {
6416   \__tikzspath_insert_gaps_after_segments:NVnn #1#1{#2}{#3}
6417 }
6418 \cs_generate_variant:Nn \__tikzspath_insert_gaps_after_segments:Nnn {cnn, cVV}
6419 \cs_new_protected_nopar:Npn \__tikzspath_ginsert_gaps_after_segments:Nnnn #1#2#3#4
6420 {
6421   \__tikzspath_insert_gaps_after_segments_aux:nnn {#2}{#3}{#4}
6422   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6423   \tl_gclear:N \g__tikzspath_output_tl
6424 }
6425 \cs_generate_variant:Nn \__tikzspath_ginsert_gaps_after_segments:Nnnn {NVnn}
6426 \cs_new_protected_nopar:Npn \__tikzspath_ginsert_gaps_after_segments:Nnn #1#2#3
6427 {
6428   \__tikzspath_ginsert_gaps_after_segments:NVnn #1#1{#2}{#3}
6429 }
6430 \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`

```

6431 \cs_new_protected_nopar:Npn \__tikzspath_join_components_aux:nn #1#2
6432 {
6433   \group_begin:
6434
6435   \tl_set:Nn \l__tikzspath_tmpa_tl {#1}
6436   \spath_numberofcomponents:NV \l__tikzspath_tmpa_int \l__tikzspath_tmpa_tl
6437   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_int {#2}
6438
6439   \seq_reverse:N \l__tikzspath_tmpa_seq
6440
6441   \seq_map_inline:Nn \l__tikzspath_tmpa_seq
6442   {
6443     \spath_join_component:Nn \l__tikzspath_tmpa_tl {##1}
6444   }
6445   \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpa_tl
6446   \group_end:
6447 }
6448 \cs_new_protected_nopar:Npn \__tikzspath_join_components:Nnn #1#2#3
6449 {
6450   \__tikzspath_join_components_aux:nn {#2}{#3}
6451   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6452   \tl_gclear:N \g__tikzspath_output_tl
6453 }
6454 \cs_generate_variant:Nn \__tikzspath_join_components:Nnn {NVn}
6455 \cs_new_protected_nopar:Npn \__tikzspath_join_components:Nn #1#2
6456 {
6457   \__tikzspath_join_components:NVn #1#1{#2}
6458 }
6459 \cs_generate_variant:Nn \__tikzspath_join_components:Nn {cn}
6460 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components:Nnn #1#2#3
6461 {
6462   \__tikzspath_join_components_aux:nn {#2}{#3}

```

```

6463 \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6464 \tl_gclear:N \g__tikzspath_output_tl
6465 }
6466 \cs_generate_variant:Nn \__tikzspath_gjoin_components:Nnn {NVn}
6467 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components:Nn #1#2
6468 {
6469   \__tikzspath_gjoin_components:NVn #1#1{#2}
6470 }
6471 \cs_generate_variant:Nn \__tikzspath_gjoin_components:Nn {cn}

```

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

`__tikzspath_join_components_with_bezier:Nn`

```

6472 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with_bezier_aux:nn #1#2
6473 {
6474   \group_begin:
6475   \tl_set:Nn \l__tikzspath_tmpc_tl {#1}
6476   \tl_if_empty:nT {#2}
6477   {
6478     \spath_spot_weld_components:N \l__tikzspath_tmpc_tl
6479   }
6480
6481   \spath_numberofcomponents:N \l__tikzspath_tmpa_int \l__tikzspath_tmpc_tl
6482   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpb_seq \l__tikzspath_tmpa_int {#2}
6483
6484   \spath_components_to_seq:N \l__tikzspath_tmpa_seq \l__tikzspath_tmpc_tl
6485
6486   \seq_pop_left:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6487   \seq_pop_left:NN \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6488
6489   \seq_map_indexed_inline:Nn \l__tikzspath_tmpa_seq
6490   {
6491     \int_compare:nTF
6492     {
6493       ##1 == \l__tikzspath_tmpb_tl
6494     }
6495     {
6496       \seq_pop_left:NNF \l__tikzspath_tmpb_seq \l__tikzspath_tmpb_tl
6497       {
6498         \tl_set:Nn \l__tikzspath_tmpb_tl {-1}
6499       }
6500       \spath_curve_between:Nn \l__tikzspath_tmpa_tl {##2}
6501     }
6502     {
6503       \tl_put_right:Nn \l__tikzspath_tmpa_tl {##2}
6504     }
6505   }
6506   \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpa_tl
6507   \group_end:
6508 }
6509 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with_bezier:Nnn #1#2#3
6510 {
6511   \__tikzspath_join_components_with_bezier_aux:nn {#2}{#3}
6512   \tl_set_eq:NN #1 \g__tikzspath_output_tl

```

```

6513 \tl_gclear:N \g__tikzspath_output_tl
6514 }
6515 \cs_generate_variant:Nn \__tikzspath_join_components_with_bezier:Nnn {NVn}
6516 \cs_new_protected_nopar:Npn \__tikzspath_join_components_with_bezier:Nn #1#2
6517 {
6518   \__tikzspath_join_components_with_bezier:NVn #1#1{#2}
6519 }
6520 \cs_generate_variant:Nn \__tikzspath_join_components_with_bezier:Nn {cV}
6521 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_with_bezier:Nnn #1#2#3
6522 {
6523   \__tikzspath_join_components_with_bezier_aux:nn {#2}{#3}
6524   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6525   \tl_gclear:N \g__tikzspath_output_tl
6526 }
6527 \cs_generate_variant:Nn \__tikzspath_gjoin_components_with_bezier:Nnn {NVn}
6528 \cs_new_protected_nopar:Npn \__tikzspath_gjoin_components_with_bezier:Nn #1#2
6529 {
6530   \__tikzspath_gjoin_components_with_bezier:NVn #1#1{#2}
6531 }
6532 \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

```

6533 \cs_new_protected_nopar:Npn \__tikzspath_remove_components_aux:nn #1#2
6534 {
6535   \group_begin:
6536
6537   \spath_numberofcomponents:Nn \l__tikzspath_tmpa_int {#1}
6538   \__tikzspath_seq_from_foreach:NVn \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_int {#2}
6539
6540   \spath_components_to_seq:Nn \l__tikzspath_tmpb_seq {#1}
6541
6542   \seq_pop_left:NNF \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6543   {
6544     \tl_clear:N \l__tikzspath_tmpa_tl
6545   }
6546
6547   \seq_clear:N \l__tikzspath_tmpc_seq
6548   \seq_map_indexed_inline:Nn \l__tikzspath_tmpb_seq
6549   {
6550     \tl_set:Nn \l__tikzspath_tmpb_tl {##1}
6551     \tl_if_eq:NNTF \l__tikzspath_tmpb_tl \l__tikzspath_tmpa_tl
6552     {
6553       \seq_pop_left:NNF \l__tikzspath_tmpa_seq \l__tikzspath_tmpa_tl
6554       {
6555         \tl_clear:N \l__tikzspath_tmpa_tl
6556       }
6557     }
6558     {
6559       \seq_put_right:Nn \l__tikzspath_tmpc_seq {##2}
6560     }
6561   }
6562 }

```

```

6563 \tl_gset:Nx \g__tikzspath_output_tl {\seq_use:Nn \l__tikzspath_tmpc_seq {} }
6564 \group_end:
6565 }
6566 \cs_new_protected_nopar:Npn \__tikzspath_remove_components:Nnn #1#2#3
6567 {
6568   \__tikzspath_remove_components_aux:nn {#2}{#3}
6569   \tl_set_eq:NN #1 \g__tikzspath_output_tl
6570   \tl_gclear:N \g__tikzspath_output_tl
6571 }
6572 \cs_generate_variant:Nn \__tikzspath_remove_components:Nnn {NVn}
6573 \cs_new_protected_nopar:Npn \__tikzspath_remove_components:Nn #1#2
6574 {
6575   \__tikzspath_remove_components:NVn #1#1{#2}
6576 }
6577 \cs_generate_variant:Nn \__tikzspath_remove_components:Nn {cn}
6578 \cs_new_protected_nopar:Npn \__tikzspath_gremove_components:Nnn #1#2#3
6579 {
6580   \__tikzspath_remove_components_aux:nn {#2}{#3}
6581   \tl_gset_eq:NN #1 \g__tikzspath_output_tl
6582   \tl_gclear:N \g__tikzspath_output_tl
6583 }
6584 \cs_generate_variant:Nn \__tikzspath_gremove_components:Nnn {NVn}
6585 \cs_new_protected_nopar:Npn \__tikzspath_gremove_components:Nn #1#2
6586 {
6587   \__tikzspath_gremove_components:NVn #1#1{#2}
6588 }
6589 \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
6590 \cs_new_protected_nopar:Npn \__tikzspath_transform_to_aux:nn #1#2
6591 {
6592   \group_begin:
6593   \spath_reallength:Nn \l__tikzspath_tmpa_int {#2}
6594
6595   \tl_set:Nx \l__tikzspath_tmpb_tl
6596   {\fp_to_decimal:n {(#1) * (\l__tikzspath_tmpa_int)}}
6597   \spath_transformation_at:NnV \l__tikzspath_tmpc_tl {#2} \l__tikzspath_tmpb_tl
6598   \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpc_tl
6599   \group_end:
6600 }
6601 \cs_new_protected_nopar:Npn \__tikzspath_transform_to:nn #1#2
6602 {
6603   \__tikzspath_transform_to_aux:nn {#1}{#2}
6604   \exp_last_unbraced:NV \pgfsettransformentries \g__tikzspath_output_tl
6605   \tl_gclear:N \g__tikzspath_output_tl
6606 }
6607 \cs_generate_variant:Nn \__tikzspath_transform_to:nn {nv}
6608 \cs_new_protected_nopar:Npn \__tikzspath_transform_upright_to:nn #1#2
6609 {
6610   \__tikzspath_transform_to_aux:nn {#1}{#2}
6611   \fp_compare:nT { \tl_item:Nn \g__tikzspath_output_tl {4} < 0 }
6612   {

```

```

6613 \tl_gset:Nx \g__tikzspath_output_tl
6614 {
6615   { \fp_eval:n { - (\tl_item:Nn \g__tikzspath_output_tl {1}) } }
6616   { \fp_eval:n { - (\tl_item:Nn \g__tikzspath_output_tl {2}) } }
6617   { \fp_eval:n { - (\tl_item:Nn \g__tikzspath_output_tl {3}) } }
6618   { \fp_eval:n { - (\tl_item:Nn \g__tikzspath_output_tl {4}) } }
6619   { \tl_item:Nn \g__tikzspath_output_tl {5} }
6620   { \tl_item:Nn \g__tikzspath_output_tl {6} }
6621 }
6622 }
6623 \exp_last_unbraced:N \pgfsettransformentries \g__tikzspath_output_tl
6624 \tl_gclear:N \g__tikzspath_output_tl
6625 }
6626 \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.

```

6627 \tikzset{

```

We're in the `spath` key family.

```

6628   spath/.is~family,
6629   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.

```

6630   set~ prefix/.store~ in=\l__tikzspath_prefix_tl,
6631   prefix/.is~choice,
6632   prefix/default/.style={
6633     /tikz/spath/set~ prefix=tikz@intersect@path@name@
6634   },
6635   set~ suffix/.store~ in=\l__tikzspath_suffix_tl,
6636   suffix/.is~choice,
6637   suffix/default/.style={
6638     /tikz/spath/set~ suffix={}
6639   },
6640   set~ name/.style={
6641     /tikz/spath/prefix=#1,
6642     /tikz/spath/suffix=#1
6643   },

```

Hook in to the end of the path construction

```

6644   at~ end~ path~ construction/.code={
6645     \tl_put_right:Nn \l__tikzspath_tikzpath_finish_tl {#1}
6646   },

```

Keys for saving and cloning a soft path.

```

6647   save/.code={
6648     \tikz@addmode{
6649       \spath_get_current_path:N \l__tikzspath_tmpa_tl
6650       \spath_bake_round:N \l__tikzspath_tmpa_tl \l__tikzspath_tmpa_tl
6651       \spath_bake_shorten:N \l__tikzspath_tmpa_tl \l__tikzspath_tmpa_tl

```

```

6652     \spath_save_path:cV {\__tikzspath_path_name:n {#1}}
6653     \l__tikzspath_tmpa_tl
6654   }
6655 },
6656 save~ global/.code={
6657   \tikz@addmode{
6658     \spath_get_current_path:N \l__tikzspath_tmpa_tl
6659     \spath_bake_round:NV \l__tikzspath_tmpa_tl \l__tikzspath_tmpa_tl
6660     \spath_bake_shorten:NV \l__tikzspath_tmpa_tl \l__tikzspath_tmpa_tl
6661     \spath_gsave_path:cV {\__tikzspath_path_name:n {#1}}
6662     \l__tikzspath_tmpa_tl
6663   }
6664 },
6665 clone/.code~ 2~ args={
6666   \__tikzspath_maybe_current_path:nn
6667   {
6668     \__tikzspath_check_path:nnn {
6669       \tl_clear_new:c {\__tikzspath_path_name:n {#1}}
6670       \tl_set_eq:cc {\__tikzspath_path_name:n {#1}}
6671     }
6672   }
6673   {#2}{}
6674 },
6675 clone~ global/.code~ 2~ args={
6676   \__tikzspath_maybe_current_path:nn
6677   {
6678     \__tikzspath_check_path:nnn {
6679       \tl_gclear_new:c {\__tikzspath_path_name:n {#1}}
6680       \tl_gset_eq:cc {\__tikzspath_path_name:n {#1}}
6681     }
6682   }
6683   {#2}{}
6684 },

```

Saves a soft path to the aux file.

```

6685 save~ to~ aux/.code={
6686   \__tikzspath_maybe_current_path:nn
6687   {
6688     \__tikzspath_check_path:nnn {
6689       \spath_save_to_aux:c
6690     }
6691   }
6692   {#1}
6693   {}
6694 },

```

Exports the path as an SVG file.

```

6695 export~ to~ svg/.code={
6696   \__tikzspath_maybe_current_path:nn
6697   {
6698     \__tikzspath_check_path:nnn {
6699       \spath_export_to_svg:nv {#1}
6700     }
6701   }

```



```

6702     {#1}
6703     {}
6704 },

```

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).

```

6705 use/.code={
6706     \bool_set_false:N \l__tikzspath_reverse_bool
6707     \bool_set_false:N \l__tikzspath_weld_bool
6708     \bool_set_false:N \l__tikzspath_move_bool
6709     \bool_set_false:N \l__tikzspath_current_transformation_bool
6710     \bool_set_true:N \l_spath_movetorelevant_bool
6711     \tl_clear:N \l__tikzspath_joinpath_tl
6712     \tl_clear:N \l__tikzspath_transformation_tl
6713     \tikzset{
6714         spath/join/.cd,
6715         #1
6716     }
6717
6718     \__tikzspath_check_path:nVn
6719     {
6720         \__tikzspath_use_path:v
6721     } \l__tikzspath_joinpath_tl {}
6722
6723 },

```

Some aliases for the above.

```

6724 restore/.style={/tikz/spath/use={#1}},
6725 restore~ reverse/.style={/tikz/spath/use={reverse, #1}},
6726 append/.style={/tikz/spath/use={move, weld, #1}},
6727 append~ no~ move/.style={/tikz/spath/use={weld, #1}},
6728 append~ reverse/.style={/tikz/spath/use={move, weld, reverse, #1}},
6729 append~ reverse~ no~ move/.style={/tikz/spath/use={weld, reverse, #1}},
6730 insert/.style={/tikz/spath/use={#1}},
6731 insert~ reverse/.style={/tikz/spath/use={reverse, #1}},

```

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

```

6732 show~current~path/.code={
6733     \tikz@addmode{
6734         \pgfsyssoftpath@getcurrentpath\l__tikzspath_tmpa_tl
6735         \iow_term:n {---- current~ soft~ path~ ---}
6736         \spath_show:V \l__tikzspath_tmpa_tl
6737     }
6738 },

```

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

```

6739 show/.code={
6740     \__tikzspath_check_path:nnn {
6741         \iow_term:n {---- soft~ path~ #1~ ---}
6742         \spath_show:v
6743     } {#1} {}
6744 },

```

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.

```

6745 join~ with/.code~ 2~ args={
6746   \bool_set_false:N \l__tikzspath_reverse_bool
6747   \bool_set_false:N \l__tikzspath_weld_bool
6748   \bool_set_false:N \l__tikzspath_move_bool
6749   \bool_set_false:N \l__tikzspath_global_bool
6750   \bool_set_false:N \l__tikzspath_current_transformation_bool
6751   \tl_clear:N \l__tikzspath_joinpath_tl
6752   \tl_clear:N \l__tikzspath_transformation_tl
6753   \tikzset{
6754     spath/join/.cd,
6755     #2
6756   }
6757
6758   \__tikzspath_maybe_current_path_reuse:nnn
6759   {
6760     \__tikzspath_check_two_paths:nnVn
6761     {
6762       \__tikzspath_join_with:cv
6763     }
6764   } {#1} { \l__tikzspath_joinpath_tl {} }
6765 },

```

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

```

6766 spot~ weld/.code={
6767   \__tikzspath_maybe_current_path_reuse:nnn
6768   {
6769     \__tikzspath_check_path:nnn
6770     {
6771       \spath_spot_weld_components:c
6772     }
6773   } {#1} { {} }
6774 },
6775 spot~ weld~ globally/.code={
6776   \__tikzspath_maybe_current_path_reuse:nnn
6777   {
6778     \__tikzspath_check_path:nnn
6779     {
6780       \spath_spot_gweld_components:c
6781     }
6782   } {#1} { {} }
6783 },

```

Reverses the named path.

```

6784 reverse/.code={
6785   \__tikzspath_maybe_current_path_reuse:nnn
6786   {
6787     \__tikzspath_check_path:nnn
6788     {
6789       \spath_reverse:c
6790     }

```

```

6791     } {#1} { {} }
6792 },
6793 reverse~ globally/.code={
6794     \__tikzspath_maybe_current_path_reuse:nnn
6795     {
6796         \__tikzspath_check_path:nnn
6797         {
6798             \spath_greverse:c
6799         }
6800     } {#1} { {} }
6801 },

```

Adjust a path to span between two points.

```

6802 span/.code ~n~ args={3}{
6803     \__tikzspath_maybe_current_path_reuse:nnn
6804     {
6805         \__tikzspath_check_path:nnn
6806         {
6807             \__tikzspath_process_tikz_point:Nn \l__tikzspath_tmpa_tl {#2}
6808             \__tikzspath_process_tikz_point:Nn \l__tikzspath_tmpb_tl {#3}
6809             \spath_span:cVV
6810         }
6811     } {#1} { \use_none:nn \l__tikzspath_tmpa_tl \l__tikzspath_tmpb_tl }
6812 },
6813 span~ global/.code ~n~ args={3}{
6814     \__tikzspath_maybe_current_path_reuse:nnn
6815     {
6816         \__tikzspath_check_path:nnn
6817         {
6818             \__tikzspath_process_tikz_point:Nn \l__tikzspath_tmpa_tl {#2}
6819             \__tikzspath_process_tikz_point:Nn \l__tikzspath_tmpb_tl {#3}
6820             \spath_span:cVV
6821         }
6822     } {#1} { \use_none:nn \l__tikzspath_tmpa_tl \l__tikzspath_tmpb_tl }
6823 },

```

Defines a to path

```

6824 to/.style={
6825     to~path={
6826         [
6827             spath/span={#1}{(\tikztostart)}{(\tikztotarget)},
6828             spath/use={#1,weld},
6829         ]
6830         \tikztonodes
6831     }
6832 },

```

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

```

6833 splice/.code ~n~ args={3}{
6834     \__tikzspath_maybe_current_path_reuse:nnn
6835     {
6836         \__tikzspath_check_three_paths:nnnnn
6837     }

```

```

6838     \spath_splice_between:cvv
6839   }
6840   } {#1} { {#2} {#3} {} }
6841 },
6842 splice~ global/.code ~n~ args={3}{
6843   \__tikzspath_maybe_current_path_reuse:nnn
6844   {
6845     \__tikzspath_check_three_paths:nnnnn
6846     {
6847       \spath_gsplice_between:cvv
6848     }
6849   } {#1} { {#2} {#3} {} }
6850 },

```

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”.

```

6851 join~ components~ with/.code~2~args={
6852   \tl_if_head_is_group:nTF {#2}
6853   {
6854     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
6855     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
6856   }
6857   {
6858     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6859     \tl_clear:N \l__tikzspath_tmpd_tl
6860   }
6861
6862   \__tikzspath_maybe_current_path_reuse:nnn
6863   {
6864     \__tikzspath_check_two_paths:nnVn
6865     {
6866       \__tikzspath_join_components_with:cvV
6867     }
6868   } {#1} { \l__tikzspath_tmpc_tl \use_none:n \l__tikzspath_tmpd_tl }
6869 },
6870 join~ components~ globally~ with/.code~2~args={
6871   \tl_if_head_is_group:nTF {#2}
6872   {
6873     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
6874     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
6875   }
6876   {
6877     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6878     \tl_clear:N \l__tikzspath_tmpd_tl
6879   }
6880
6881   \__tikzspath_maybe_current_path_reuse:nnn
6882   {
6883     \__tikzspath_check_two_paths:nnVn
6884     {

```

```

6885     \tikzspath_gjoin_components_with:cvV
6886   }
6887   } {#1} { \l__tikzspath_tmpc_tl \use_none:n \l__tikzspath_tmpd_tl }
6888 },
6889 join~ components~ upright~ with/.code~2~args={
6890   \tl_if_head_is_group:nTF {#2}
6891   {
6892     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
6893     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
6894   }
6895   {
6896     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6897     \tl_clear:N \l__tikzspath_tmpd_tl
6898   }
6899
6900   \__tikzspath_maybe_current_path_reuse:nnn
6901   {
6902     \__tikzspath_check_two_paths:nnVn
6903     {
6904       \__tikzspath_join_components_upright_with:cvV
6905     }
6906   } {#1} { \l__tikzspath_tmpc_tl \use_none:n \l__tikzspath_tmpd_tl }
6907 },
6908 join~ components~ globally~ upright~ with/.code~2~args={
6909   \tl_if_head_is_group:nTF {#2}
6910   {
6911     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
6912     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
6913   }
6914   {
6915     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
6916     \tl_clear:N \l__tikzspath_tmpd_tl
6917   }
6918
6919   \__tikzspath_maybe_current_path_reuse:nnn
6920   {
6921     \__tikzspath_check_two_paths:nnVn
6922     {
6923       \__tikzspath_gjoin_components_upright_with:cvV
6924     }
6925   } {#1} { \l__tikzspath_tmpc_tl \use_none:n \l__tikzspath_tmpd_tl }
6926 },
6927 join~ components~ with~ bezier/.code={
6928   \tl_if_head_is_group:nTF {#1}
6929   {
6930     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#1} {1} }
6931     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#1} {2} }
6932   }
6933   {
6934     \tl_set:Nn \l__tikzspath_tmpc_tl {#1}
6935     \tl_clear:N \l__tikzspath_tmpd_tl
6936   }
6937
6938   \__tikzspath_maybe_current_path_reuse:nVn

```

```

6939 {
6940   \tikzspath_check_path:nnn
6941   {
6942     \tikzspath_join_components_with_bezier:cV
6943   }
6944   } \l__tikzspath_tmpc_tl { \use_none:n \l__tikzspath_tmpd_tl }
6945 },
6946 join~ components~ globally~ with~ bezier/.code~2~args={
6947   \tikzspath_maybe_current_path_reuse:nnn
6948   {
6949     \tikzspath_check_path:nnn
6950     {
6951       \tikzspath_gjoin_components_with_bezier:cn
6952     }
6953   } {#1} { \use_none:n {#2} }
6954 },

```

Close a path.

```

6955 close/.code={
6956   \tikzspath_maybe_current_path_reuse:nnn
6957   {
6958     \tikzspath_check_path:nnn
6959     {
6960       \spath_close:c
6961     }
6962   } {#1} { {} }
6963 },
6964 close~ globally/.code={
6965   \tikzspath_maybe_current_path_reuse:nnn
6966   {
6967     \tikzspath_check_path:nnn
6968     {
6969       \spath_gclose:c
6970     }
6971   } {#1} { {} }
6972 },

```

Open a path.

```

6973 open/.code={
6974   \tikzspath_maybe_current_path_reuse:nnn
6975   {
6976     \tikzspath_check_path:nnn
6977     {
6978       \spath_open:c
6979     }
6980   } {#1} { {} }
6981 },
6982 open~ globally/.code={
6983   \tikzspath_maybe_current_path_reuse:nnn
6984   {
6985     \tikzspath_check_path:nnn
6986     {
6987       \spath_gopen:c
6988     }

```

```

6989     } {#1} { {} }
6990 },

```

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

```

6991 adjust~ and~ close/.code={
6992   \__tikzspath_maybe_current_path_reuse:nnn
6993   {
6994     \__tikzspath_check_path:nnn
6995     {
6996       \spath_adjust_close:c
6997     }
6998   } {#1} { {} }
6999 },
7000 adjust~ and~ close~ globally/.code={
7001   \__tikzspath_maybe_current_path_reuse:nnn
7002   {
7003     \__tikzspath_check_path:nnn
7004     {
7005       \spath_adjust_gclose:c
7006     }
7007   } {#1} { {} }
7008 },

```

Close a path with another path.

```

7009 close~ with/.code~ 2~ args={
7010   \__tikzspath_maybe_current_path_reuse:nnn
7011   {
7012     \__tikzspath_check_two_paths:nnnn
7013     {
7014       \spath_close_with:cv
7015     }
7016   } {#1} { {#2} {} }
7017 },
7018 close~ globally~ with/.code~ 2~ args={
7019   \__tikzspath_maybe_current_path_reuse:nnn
7020   {
7021     \__tikzspath_check_two_paths:nnnn
7022     {
7023       \spath_gclose_with:cv
7024     }
7025   } {#1} { {#2} {} }
7026 },

```

Close a path with a curve.

```

7027 close~ with~ curve/.code={
7028   \__tikzspath_maybe_current_path_reuse:nnn
7029   {
7030     \__tikzspath_check_path:nnn
7031     {
7032       \spath_close_with_curve:c
7033     }
7034   } {#1} { {} }
7035 },
7036 close~ globally~ with~ curve/.code={

```

```

7037 \tikzspath_maybe_current_path_reuse:nnn
7038 {
7039   \tikzspath_check_path:nnn
7040   {
7041     \spath_gclose_with_curve:c
7042   }
7043 } {#1} { {} }
7044 },

```

These keys shorten the path by a dimension.

```

7045 shorten~ at~ end/.code~ 2~ args={
7046   \tikzspath_maybe_current_path_reuse:nnn
7047   {
7048     \tikzspath_check_path:nnn
7049     {
7050       \spath_shorten_at_end:cn
7051     }
7052   } {#1} { \use_none:n {#2} }
7053 },
7054 shorten~ at~ start/.code~ 2~ args ={
7055   \tikzspath_maybe_current_path_reuse:nnn
7056   {
7057     \tikzspath_check_path:nnn
7058     {
7059       \spath_shorten_at_start:cn
7060     }
7061   } {#1} { \use_none:n {#2} }
7062 },
7063 shorten~ at~ both~ ends/.code~ 2~ args={
7064   \tikzspath_maybe_current_path_reuse:nnn
7065   {
7066     \tikzspath_check_path:nnn
7067     {
7068       \spath_shorten_at_both_ends:cn
7069     }
7070   } {#1} { \use_none:n {#2} }
7071 },
7072 shorten~ globally~ at~ end/.code~ 2~ args={
7073   \tikzspath_maybe_current_path_reuse:nnn
7074   {
7075     \tikzspath_check_path:nnn
7076     {
7077       \spath_gshorten_at_end:cn
7078     }
7079   } {#1} { \use_none:n {#2} }
7080 },
7081 shorten~ globally~ at~ start/.code~ 2~ args ={
7082   \tikzspath_maybe_current_path_reuse:nnn
7083   {
7084     \tikzspath_check_path:nnn
7085     {
7086       \spath_gshorten_at_start:cn
7087     }
7088   } {#1} { \use_none:n {#2} }

```



```

7089 },
7090 shorten~ globally~ at~ both~ ends/.code~ 2~ args={
7091   \__tikzspath_maybe_current_path_reuse:nnn
7092   {
7093     \__tikzspath_check_path:nnn
7094     {
7095       \spath_gshorten_at_both_ends:cn
7096     }
7097   } {#1} { \use_none:n {#2} }
7098 },

```

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

```

7099 split~ at/.code~ 2~ args={
7100   \__tikzspath_maybe_current_path_reuse:nnn
7101   {
7102     \__tikzspath_check_path:nnn
7103     {
7104       \spath_split_at_normalised:cn
7105     }
7106   } {#1} { {} {#2} }
7107 },
7108 split~ globally~ at/.code~ 2~ args={
7109   \__tikzspath_maybe_current_path_reuse:nnn
7110   {
7111     \__tikzspath_check_path:nnn
7112     {
7113       \spath_gsplit_at_normalised:cn
7114     }
7115   } {#1} { {} {#2} }
7116 },
7117 split~ at~ into/.code~ n~ args={4}{
7118   \__tikzspath_maybe_current_path_reuse:nnn
7119   {
7120     \__tikzspath_check_path:nnn
7121     {
7122       \spath_split_at_normalised:ccvn {\__tikzspath_path_name:n {#1}}
7123       {\__tikzspath_path_name:n {#2}}
7124     }
7125   } {#3} { \use_none:n {#4} }
7126 },
7127 split~ globally~ at~ into/.code~ n~ args={4}{
7128   \__tikzspath_maybe_current_path_reuse:nnn
7129   {
7130     \__tikzspath_check_path:nnn
7131     {
7132       \spath_gsplit_at_normalised:ccvn {\__tikzspath_path_name:n {#1}}
7133       {\__tikzspath_path_name:n {#2}}
7134     }
7135   } {#3} { \use_none:n {#4} }
7136 },
7137 split~ at~ keep~ start/.code~ 2~ args={
7138   \__tikzspath_maybe_current_path_reuse:nnn
7139   {

```

```

7140     \tikzspath_check_path:nnn
7141     {
7142     \spath_split_at_normalised_keep_start:cn
7143     }
7144     } {#1} { \use_none:n {#2} }
7145 },
7146 split~ globally~ at~ keep~ start/.code~ 2~ args={
7147     \tikzspath_maybe_current_path_reuse:nnn
7148     {
7149     \tikzspath_check_path:nnn
7150     {
7151     \spath_gsplit_at_normalised_keep_start:cn
7152     }
7153     } {#1} { \use_none:n {#2} }
7154 },
7155 split~ at~ keep~ end/.code~ 2~ args={
7156     \tikzspath_maybe_current_path_reuse:nnn
7157     {
7158     \tikzspath_check_path:nnn
7159     {
7160     \spath_split_at_normalised_keep_end:cn
7161     }
7162     } {#1} { \use_none:n {#2} }
7163 },
7164 split~ globally~ at~ keep~ end/.code~ 2~ args={
7165     \tikzspath_maybe_current_path_reuse:nnn
7166     {
7167     \tikzspath_check_path:nnn
7168     {
7169     \spath_gsplit_at_normalised_keep_end:cn
7170     }
7171     } {#1} { \use_none:n {#2} }
7172 },
7173 split~ at~ keep~ middle/.code~ n~ args={3}{
7174     \tikzspath_maybe_current_path_reuse:nnn
7175     {
7176     \tikzspath_check_path:nnn
7177     {
7178     \spath_split_at_normalised_keep_middle:cnn
7179     }
7180     } {#1} { \use_none:nn {#2} {#3} }
7181 },
7182 split~ globally~ at~ keep~ middle/.code~ n~ args={3}{
7183     \tikzspath_maybe_current_path_reuse:nnn
7184     {
7185     \tikzspath_check_path:nnn
7186     {
7187     \spath_gsplit_at_normalised_keep_middle:cnn
7188     }
7189     } {#1} { \use_none:nn {#2} {#3} }
7190 },

```

This translates the named path.

```

7191 translate/.code~ n~ args={3}{

```

```

7192 \tikzspath_maybe_current_path_reuse:nnn
7193 {
7194   \tikzspath_check_path:nnn
7195   {
7196     \spath_translate:cnn
7197   }
7198   } {#1} { \use_none:nn {#2}{#3} }
7199 },
7200 translate~ globally/.code~ n~ args={3}{
7201   \tikzspath_maybe_current_path_reuse:nnn
7202   {
7203     \tikzspath_check_path:nnn
7204     {
7205       \spath_gtranslate:cnn
7206     }
7207     } {#1} { \use_none:nn {#2}{#3} }
7208 },

```

This normalises the named path.

```

7209 normalise/.code={
7210   \tikzspath_maybe_current_path_reuse:nnn
7211   {
7212     \tikzspath_check_path:nnn
7213     {
7214       \spath_normalise:c
7215     }
7216     } {#1} { {} }
7217 },
7218 normalise~ globally/.code={
7219   \tikzspath_maybe_current_path_reuse:nnn
7220   {
7221     \tikzspath_check_path:nnn
7222     {
7223       \spath_gnormalise:c
7224     }
7225     } {#1} { {} }
7226 },

```

Transforms the named path using TikZ transformation specifications.

```

7227 transform/.code~ 2~ args={
7228   \group_begin:
7229   \pgftransformreset
7230   \tikzset{#2}
7231   \pgfgettransform \l__tikzspath_tmpa_tl
7232   \tl_gset_eq:NN \g__tikzspath_smuggle_tl \l__tikzspath_tmpa_tl
7233   \group_end:
7234
7235   \tikzspath_maybe_current_path_reuse:nnn
7236   {
7237     \tikzspath_check_path:nnn
7238     {
7239       \spath_transform:cV
7240     }
7241     } {#1} { \use_none:n \g__tikzspath_smuggle_tl }

```

```

7242 },
7243 transform~globally/.code~ 2~ args={
7244   \group_begin:
7245   \pgftransformreset
7246   \tikzset{#2}
7247   \pgfgettransform \l__tikzspath_tmpa_tl
7248   \tl_gset_eq:NN \g__tikzspath_smuggle_tl \l__tikzspath_tmpa_tl
7249   \group_end:
7250
7251   \__tikzspath_maybe_current_path_reuse:nnn
7252   {
7253     \__tikzspath_check_path:nnn
7254     {
7255       \spath_gtransform:cV
7256     }
7257   } {#1} { \use_none:n \g__tikzspath_smuggle_tl }
7258 },

```

Splits first path where it intersects with the second.

```

7259 split~ at~ intersections~ with/.code~ 2~ args={
7260   \tl_if_exist:cTF
7261   {
7262     tikz@library@intersections@loaded
7263   }
7264   {
7265     \__tikzspath_maybe_current_two_paths_reuse_first:nnnn
7266     {
7267       \__tikzspath_check_two_paths:nnnn
7268       {
7269         \spath_split_path_at_intersections:cv
7270       }
7271     } {#1} {#2} { {} }
7272   }
7273   {
7274     \msg_warning:nn { spath3 } { load intersections }
7275   }
7276 },
7277 split~ globally~ at~ intersections~ with/.code~ n~ args={2}{
7278   \tl_if_exist:cTF
7279   {
7280     tikz@library@intersections@loaded
7281   }
7282   {
7283     \__tikzspath_maybe_current_two_paths_reuse_first:nnnn
7284     {
7285       \__tikzspath_check_two_paths:nnnn
7286       {
7287         \spath_gsplit_path_at_intersections:cv
7288       }
7289     } {#1} {#2} { {} }
7290   }
7291   {
7292     \msg_warning:nn { spath3 } { load intersections }
7293   }

```

7294 },

Splits two paths at their mutual intersections.

```
7295 split~ at~ intersections/.code~ n~ args={2}{
7296   \tl_if_exist:cTF
7297   {
7298     tikz@library@intersections@loaded
7299   }
7300   {
7301     \__tikzspath_maybe_current_two_paths_reuse_both:nnnn
7302     {
7303       \__tikzspath_check_two_paths:nnnn
7304       {
7305         \spath_split_at_intersections:cc
7306       }
7307     } {#1} {#2} { {} }
7308   }
7309   {
7310     \msg_warning:nn { spath3 } { load intersections }
7311   }
7312 },
7313 split~ globally~ at~ intersections/.code~ n~ args={2}{
7314   \tl_if_exist:cTF
7315   {
7316     tikz@library@intersections@loaded
7317   }
7318   {
7319     \__tikzspath_maybe_current_two_paths_reuse_both:nnnn
7320     {
7321       \__tikzspath_check_two_paths:nnnn
7322       {
7323         \spath_gsplit_at_intersections:cc
7324       }
7325     } {#1} {#2} { {} }
7326   }
7327   {
7328     \msg_warning:nn { spath3 } { load intersections }
7329   }
7330 },
```

Splits a path at its self-intersections.

```
7331 split~ at~ self~ intersections/.code={
7332   \tl_if_exist:cTF
7333   {
7334     tikz@library@intersections@loaded
7335   }
7336   {
7337     \__tikzspath_maybe_current_path_reuse:nnn
7338     {
7339       \__tikzspath_check_path:nnn
7340       {
7341         \spath_split_at_self_intersections:c
7342       }
7343     } {#1} { {} }
```

```

7344     }
7345     {
7346       \msg_warning:nn { spath3 } { load intersections }
7347     }
7348   },
7349   split~ globally~ at~ self~ intersections/.code={
7350     \tl_if_exist:cTF
7351     {
7352       tikz@library@intersections@loaded
7353     }
7354     {
7355       \__tikzspath_maybe_current_path_reuse:nnn
7356       {
7357         \__tikzspath_check_path:nnn
7358         {
7359           \spath_gsplit_at_self_intersections:c
7360         }
7361       } {#1} { {} }
7362     }
7363     {
7364       \msg_warning:nn { spath3 } { load intersections }
7365     }
7366   },

```

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

```

7367   get~ components~ of/.code~ 2~ args={
7368     \__tikzspath_maybe_current_path:nn
7369     {
7370       \__tikzspath_check_path:nnn {
7371         \__tikzspath_get_components:Nv #2
7372       }
7373     }
7374     {#1}
7375     {}
7376   },
7377   get~ components~ of~ globally/.code~ 2~ args={
7378     \__tikzspath_maybe_current_path:nn
7379     {
7380       \__tikzspath_check_path:nnn {
7381         \__tikzspath_gget_components:Nv #2
7382       }
7383     }
7384     {#1}
7385     {}
7386   },

```

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

```

7387   render~ components/.code={
7388     \__tikzspath_maybe_current_path:nn
7389     {
7390       \__tikzspath_check_path:nnn {
7391         \__tikzspath_render_components:nv {#1}

```

```

7392     }
7393   }
7394   {#1}
7395   {}
7396 },

```

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.

```

7397 insert~ gaps~ after~ components/.code~ 2~ args={
7398   \tl_if_head_is_group:nTF {#2}
7399   {
7400     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
7401     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
7402   }
7403   {
7404     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
7405     \tl_clear:N \l__tikzspath_tmpd_tl
7406   }
7407
7408   \__tikzspath_maybe_current_path_reuse:nnn
7409   {
7410     \__tikzspath_check_path:nnn
7411     {
7412       \__tikzspath_insert_gaps_after_components:cVV
7413     }
7414   } {#1} { \use_none:nn \l__tikzspath_tmpc_tl \l__tikzspath_tmpd_tl }
7415 },
7416 insert~ gaps~ globally~ after~ components/.code~ 2~ args={
7417   \tl_if_head_is_group:nTF {#2}
7418   {
7419     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
7420     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
7421   }
7422   {
7423     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
7424     \tl_clear:N \l__tikzspath_tmpd_tl
7425   }
7426
7427   \__tikzspath_maybe_current_path_reuse:nnn
7428   {
7429     \__tikzspath_check_path:nnn
7430     {
7431       \__tikzspath_ginsert_gaps_after_components:cVV
7432     }
7433   } {#1} { \use_none:nn \l__tikzspath_tmpc_tl \l__tikzspath_tmpd_tl }
7434 },

```

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.

```

7435 insert~ gaps~ after~ segments/.code~ 2~ args={
7436   \tl_if_head_is_group:nTF {#2}
7437   {
7438     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
7439     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }

```

```

7440 }
7441 {
7442   \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
7443   \tl_clear:N \l__tikzspath_tmpd_tl
7444 }
7445
7446 \__tikzspath_maybe_current_path_reuse:nnn
7447 {
7448   \__tikzspath_check_path:nnn
7449   {
7450     \__tikzspath_insert_gaps_after_segments:cVV
7451   }
7452   } {#1} { \use_none:nn \l__tikzspath_tmpc_tl \l__tikzspath_tmpd_tl }
7453 },
7454 insert~ gaps~ globally~ after~ segments/.code~ 2~ args={
7455   \tl_if_head_is_group:nTF {#2}
7456   {
7457     \tl_set:Nx \l__tikzspath_tmpc_tl { \tl_item:nn {#2} {1} }
7458     \tl_set:Nx \l__tikzspath_tmpd_tl { \tl_item:nn {#2} {2} }
7459   }
7460   {
7461     \tl_set:Nn \l__tikzspath_tmpc_tl {#2}
7462     \tl_clear:N \l__tikzspath_tmpd_tl
7463   }
7464
7465   \__tikzspath_maybe_current_path_reuse:nnn
7466   {
7467     \__tikzspath_check_path:nnn
7468     {
7469       \__tikzspath_ginsert_gaps_after_segments:cVV
7470     }
7471     } {#1} { \use_none:nn \l__tikzspath_tmpc_tl \l__tikzspath_tmpd_tl }
7472 },
7473 join~ components/.code~ 2~ args={
7474   \__tikzspath_maybe_current_path_reuse:nnn
7475   {
7476     \__tikzspath_check_path:nnn
7477     {
7478       \__tikzspath_join_components:cn
7479     }
7480     } {#1} { \use_none:n {#2} }
7481 },
7482 join~ components~ globally/.code~ 2~ args={
7483   \__tikzspath_maybe_current_path_reuse:nnn
7484   {
7485     \__tikzspath_check_path:nnn
7486     {
7487       \__tikzspath_gjoin_components:cn
7488     }
7489     } {#1} { \use_none:n {#2} }
7490 },

```

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


```

7491 remove~ empty~ components/.code={
7492   \__tikzspath_maybe_current_path_reuse:nnn
7493   {
7494     \__tikzspath_check_path:nnn
7495     {
7496       \spath_remove_empty_components:c
7497     }
7498   } {#1} { {} }
7499 },
7500 remove~ empty~ components~ globally/.code={
7501   \__tikzspath_maybe_current_path_reuse:nnn
7502   {
7503     \__tikzspath_check_path:nnn
7504     {
7505       \spath_gremove_empty_components:c
7506     }
7507   } {#1} { {} }
7508 },

```

Replace all line segments by Bézier curves.

```

7509 replace~ lines/.code={
7510   \__tikzspath_maybe_current_path_reuse:nnn
7511   {
7512     \__tikzspath_check_path:nnn
7513     {
7514       \spath_replace_lines:c
7515     }
7516   } {#1} { {} }
7517 },
7518 replace~ lines~ globally/.code={
7519   \__tikzspath_maybe_current_path_reuse:nnn
7520   {
7521     \__tikzspath_check_path:nnn
7522     {
7523       \spath_greplace_lines:c
7524     }
7525   } {#1} { {} }
7526 },

```

Remove the specified components.

```

7527 remove~ components/.code~ 2~ args={
7528   \__tikzspath_maybe_current_path_reuse:nnn
7529   {
7530     \__tikzspath_check_path:nnn
7531     {
7532       \__tikzspath_remove_components:cn
7533     }
7534   } {#1} { \use_none:n {#2} }
7535 },
7536 remove~ components~ globally/.code~ 2~ args={
7537   \__tikzspath_maybe_current_path_reuse:nnn
7538   {
7539     \__tikzspath_check_path:nnn
7540     {

```

```

7541     \tikzspath_gremove_components:cn
7542   }
7543   } {#1} { \use_none:n {#2} }
7544 },

```

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.

```

7545   draft~ mode/.code={
7546     \tikzspath_set_bool:Nn \l_tikzspath_draft_bool {#1}
7547   },
7548   maybe~ spot~ weld/.code={
7549     \bool_if:NF \l_tikzspath_draft_bool
7550     {
7551       \tikzspath_maybe_current_path_reuse:nnn
7552       {
7553         \tikzspath_check_path:nnn
7554         {
7555           \spath_spot_weld_components:c
7556         }
7557       } {#1} { {} }
7558     }
7559   },
7560   maybe~ spot~ weld~ globally/.code={
7561     \bool_if:NF \l_tikzspath_draft_bool
7562     {
7563       \tikzspath_maybe_current_path_reuse:nnn
7564       {
7565         \tikzspath_check_path:nnn
7566         {
7567           \spath_spot_gweld_components:c
7568         }
7569       } {#1} { {} }
7570     }
7571   },

```

Set the transformation to lie along a path.

```

7572   transform~ to/.code~ 2~ args={
7573     \tikzspath_maybe_current_path:nn
7574     {
7575       \tikzspath_check_path:nnn {
7576         \tikzspath_transform_to:nv {#2}
7577       }
7578     }
7579     {#1}
7580     {
7581       \pgfsettransformentries {1}{0}{0}{1}{0pt}{0pt}
7582     }
7583   },

```

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

```

7584   upright~ transform~ to/.code~ 2~ args={
7585     \tikzspath_maybe_current_path:nn

```

```

7586 {
7587   \tikzspath_check_path:nnn {
7588     \tikzspath_transform_upright_to:nv {#2}
7589   }
7590 }
7591 {#1}
7592 {
7593   \pgfsettransformentries {1}{0}{1}{0pt}{0pt}
7594 }
7595 },

```

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

```

7596 knot/.style~ n~ args={3}{
7597   /tikz/spath/split~ at~ self~ intersections=#1,
7598   /tikz/spath/remove~ empty~ components=#1,
7599   /tikz/spath/insert~ gaps~ after~ components={#1}{#2}{#3},
7600   /tikz/spath/maybe~ spot~ weld=#1,
7601   /tikz/spath/render~ components=#1
7602 },
7603 global~ knot/.style~ n~ args={3}{
7604   /tikz/spath/split~ globally~ at~ self~ intersections=#1,
7605   /tikz/spath/remove~ empty~ components~ globally=#1,
7606   /tikz/spath/insert~ gaps~ globally~ after~ components={#1}{#2}{#3},
7607   /tikz/spath/maybe~ spot~ weld~ globally=#1,
7608   /tikz/spath/render~ components=#1
7609 },
7610 arrow~ shortening/.code={
7611   \tikzspath_set_bool:Nn \l_spath_arrow_shortening_bool {#1}
7612 },

```

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

```

7613 show/.default=current,
7614 spot~ weld/.default=current,
7615 spot~ weld~ globally/.default=current,
7616 reverse/.default=current,
7617 reverse~ globally/.default=current,
7618 close/.default=current,
7619 close~ globally/.default=current,
7620 open/.default=current,
7621 open~ globally/.default=current,
7622 adjust~ and~ close/.default=current,
7623 adjust~ and~ close~ globally/.default=current,
7624 close~ with~ curve/.default=current,
7625 close~ globally~ with~ curve/.default=current,
7626 remove~ empty~ components/.default=current,
7627 remove~ empty~ components~ globally/.default=current,
7628 replace~ lines/.default=current,
7629 replace~ lines~ globally/.default=current,
7630 maybe~ spot~ weld/.default=current,
7631 maybe~ spot~ weld~ globally/.default=current,
7632 }

```

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

```

7633 \cs_new_protected_nopar:Npn \__tikzspath_get_point_at:nn #1#2
7634 {
7635   \group_begin:
7636   \spath_reallength:Nn \l__tikzspath_tmpa_int {#2}
7637   \tl_set:Nx \l__tikzspath_tmpb_tl
7638   {\fp_to_decimal:n {(#1) * (\l__tikzspath_tmpa_int)}}
7639   \spath_point_at:NnV \l__tikzspath_tmpc_tl {#2} \l__tikzspath_tmpb_tl
7640
7641   \tl_clear:N \l__tikzspath_tmpd_tl
7642   \tl_put_right:Nn \l__tikzspath_tmpd_tl {\pgf@x=}
7643   \tl_put_right:Nx \l__tikzspath_tmpd_tl {\tl_item:Nn \l__tikzspath_tmpc_tl {1}}
7644   \tl_put_right:Nn \l__tikzspath_tmpd_tl {\relax}
7645   \tl_put_right:Nn \l__tikzspath_tmpd_tl {\pgf@y=}
7646   \tl_put_right:Nx \l__tikzspath_tmpd_tl {\tl_item:Nn \l__tikzspath_tmpc_tl {2}}
7647   \tl_put_right:Nn \l__tikzspath_tmpd_tl {\relax}
7648
7649   \tl_gset_eq:NN \g__tikzspath_output_tl \l__tikzspath_tmpd_tl
7650   \group_end:
7651 }
7652 \cs_generate_variant:Nn \__tikzspath_get_point_at:nn {VV, Vn, Vv}
7653
7654 \tikzdeclarecoordinatesystem{spath}{%
7655   \group_begin:
7656   \tl_set:Nn \l__tikzspath_tmpa_tl {#1}
7657   \tl_trim_spaces:N \l__tikzspath_tmpa_tl
7658
7659   \seq_set_split:NnV \l__tikzspath_tmpa_seq {~} \l__tikzspath_tmpa_tl
7660   \seq_pop_right:NN \l__tikzspath_tmpa_seq \l__tikzspath_tmpb_tl
7661
7662   \tl_set:Nx \l__tikzspath_tmpa_tl { \seq_use:Nn \l__tikzspath_tmpa_seq {~} }
7663
7664   \__tikzspath_maybe_current_path:nV
7665   {
7666     \__tikzspath_check_path:nnn {
7667       \__tikzspath_get_point_at:Vv \l__tikzspath_tmpb_tl
7668     }
7669   }
7670   \l__tikzspath_tmpa_tl
7671   {
7672     \tl_gset_eq:NN \g__tikzspath_output_tl \pgfpntorigin
7673   }
7674
7675   \group_end:
7676   \use:c {pgf@process}{%
7677     \tl_use:N \g__tikzspath_output_tl
7678     \pgftransforminvert
7679     \use:c {pgf@pos@transform@glob}
7680   }
7681   \tl_gclear:N \g__tikzspath_output_tl
7682 }
7683
7684 \ExplSyntaxOff

```

5 The Calligraphy Package

7685 <@@=cal>

5.1 Initialisation

```
7686 \RequirePackage{spath3}
7687 \ExplSyntaxOn
7688
7689 \tl_new:N \l__cal_tmpa_tl
7690 \tl_new:N \l__cal_tmpb_tl
7691 \tl_new:N \l__cal_tmp_path_tl
7692 \tl_new:N \l__cal_tmp_rpath_tl
7693 \tl_new:N \l__cal_tmp_rpathb_tl
7694 \tl_new:N \l__cal_tmp_patha_tl
7695
7696 \seq_new:N \l__cal_tmpa_seq
7697
7698 \int_new:N \l__cal_tmpa_int
7699 \int_new:N \l__cal_tmpb_int
7700 \int_new:N \g__cal_path_component_int
7701 \int_new:N \g__cal_label_int
7702
7703 \fp_new:N \l__cal_tmpa_fp
7704 \fp_new:N \l__cal_tmpb_fp
7705 \fp_new:N \l__cal_tmpe_fp
7706 \fp_new:N \l__cal_tmpe_fp
7707 \fp_new:N \l__cal_tmpe_fp
7708
7709 \dim_new:N \l__cal_tmpa_dim
7710 \dim_new:N \l__cal_tmpb_dim
7711 \dim_new:N \l__cal_tmpe_dim
7712 \dim_new:N \l__cal_tmpe_dim
7713 \dim_new:N \l__cal_tmpe_dim
7714 \dim_new:N \l__cal_tmpe_dim
7715 \dim_new:N \l__cal_tmpe_dim
7716 \dim_new:N \l__cal_tmpe_dim
7717
7718 \bool_new:N \l__cal_annotate_bool
7719 \bool_new:N \l__cal_taper_start_bool
7720 \bool_new:N \l__cal_taper_end_bool
7721 \bool_new:N \l__cal_taperable_bool
7722
7723 \dim_new:N \l__cal_taper_width_dim
7724 \dim_new:N \l__cal_line_width_dim
7725
7726 \bool_set_true:N \l__cal_taper_start_bool
7727 \bool_set_true:N \l__cal_taper_end_bool
7728
7729 \cs_generate_variant:Nn \tl_put_right:Nn {Nv}
7730
7731 \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.

```

7732 \tikzset{
7733   define-pen/.code={
7734     \tikzset{pen-name=#1}
7735     \pgf@relevantforpicturesizefalse
7736     \tikz@addmode{
7737       \pgfsyssoftpath@getcurrentpath\l__cal_tmpa_tl
7738       \spath_components_to_seq:NV \l__cal_tmpa_seq \l__cal_tmpa_tl
7739       \seq_gclear_new:c {g__cal_pen_\pgfkeysvalueof{/tikz/pen-name}_seq}
7740       \seq_gset_eq:cN
7741       {g__cal_pen_\pgfkeysvalueof{/tikz/pen-name}_seq} \l__cal_tmpa_seq
7742       \pgfusepath{discard}%
7743     }
7744   },
7745   define-pen/.default={default},
7746   use-pen/.code={
7747     \tikzset{pen-name=#1}
7748     \int_gzero:N \g__cal_path_component_int
7749     \cs_set_eq:NN \pgfpathmoveto \cal_moveto:n
7750     \tikz@addmode{
7751       \pgfsyssoftpath@getcurrentpath\l__cal_tmpa_tl
7752       \spath_components_to_seq:NV \l__cal_tmpa_seq \l__cal_tmpa_tl
7753       \tl_if_exist:cTF {g__cal_pen_\pgfkeysvalueof{/tikz/pen-name}_seq}
7754       {
7755         \cal_path_create:Nc \l__cal_tmpa_seq
7756         {g__cal_pen_\pgfkeysvalueof{/tikz/pen-name}_seq}
7757       }
7758       {
7759         \msg_warning:nx { calligraphy } { undefined pen }
7760         { \pgfkeysvalueof{/tikz/pen-name} }
7761       }
7762     }
7763   },
7764   use-pen/.default={default},
7765   pen-name/.initial={default},
7766   copperplate/.style={pen-name=copperplate},
7767   pen-colour/.initial={black},
7768   weight/.is-choice,
7769   weight/heavy/.style={
7770     line-width=\pgfkeysvalueof{/tikz/heavy-line-width},
7771     taper-width=\pgfkeysvalueof{/tikz/light-line-width},
7772   },
7773   weight/light/.style={
7774     line-width=\pgfkeysvalueof{/tikz/light-line-width},
7775     taper-width=0pt,
7776   },
7777   heavy/.style={
7778     weight=heavy
7779   },
7780   light/.style={
7781     weight=light
7782   },

```

```

7783 heavy~line~width/.initial=2pt,
7784 light~line~width/.initial=1pt,
7785 taper/.is~choice,
7786 taper/.default=both,
7787 taper/none/.style={
7788     taper~start=false,
7789     taper~end=false,
7790 },
7791 taper/both/.style={
7792     taper~start=true,
7793     taper~end=true,
7794 },
7795 taper/start/.style={
7796     taper~start=true,
7797     taper~end=false,
7798 },
7799 taper/end/.style={
7800     taper~start=false,
7801     taper~end=true,
7802 },
7803 taper~start/.code={
7804     \tl_if_eq:nnTF {#1} {true}
7805     {
7806         \bool_set_true:N \l__cal_taper_start_bool
7807     }
7808     {
7809         \bool_set_false:N \l__cal_taper_start_bool
7810     }
7811 },
7812 taper~start/.default={true},
7813 taper~end/.code={
7814     \tl_if_eq:nnTF {#1} {true}
7815     {
7816         \bool_set_true:N \l__cal_taper_end_bool
7817     }
7818     {
7819         \bool_set_false:N \l__cal_taper_end_bool
7820     }
7821 },
7822 taper~end/.default={true},
7823 taper~width/.code={\dim_set:Nn \l__cal_taper_width_dim {#1}},
7824 nib~style/.code~2~args={
7825     \tl_clear_new:c {l__cal_nib_style_#1}
7826     \tl_set:cn {l__cal_nib_style_#1} {#2}
7827 },
7828 stroke~style/.code~2~args={
7829     \tl_clear_new:c {l__cal_stroke_style_#1}
7830     \tl_set:cn {l__cal_stroke_style_#1} {#2}
7831 },
7832 this~stroke~style/.code={
7833     \tl_clear_new:c
7834     {l__cal_stroke_inline_style_ \int_use:N \g__cal_path_component_int}
7835     \tl_set:cn
7836     {l__cal_stroke_inline_style_ \int_use:N \g__cal_path_component_int} {#1}

```

```

7837 },
7838 annotate/.style={
7839   annotate~if,
7840   annotate~reset,
7841   annotation~style/.update~value={#1},
7842 },
7843 annotate~if/.default={true},
7844 annotate~if/.code={
7845   \tl_if_eq:nnTF {#1} {true}
7846   {
7847     \bool_set_true:N \l__cal_annotate_bool
7848   }
7849   {
7850     \bool_set_false:N \l__cal_annotate_bool
7851   }
7852 },
7853 annotate~reset/.code={
7854   \int_gzero:N \g__cal_label_int
7855 },
7856 annotation~style/.initial={draw,->},
7857 annotation~shift/.initial={(0,lex)},
7858 every~annotation~node/.initial={anchor=south~west},
7859 annotation~node~style/.code~2~args={
7860   \tl_clear_new:c {l__cal_annotation_style_ #1 _tl}
7861   \tl_set:cn {l__cal_annotation_style_ #1 _tl}{#2}
7862 },
7863 tl~use:N/.code={
7864   \exp_args:NV \pgfkeysalso #1
7865 },
7866 tl~use:c/.code={
7867   \tl_if_exist:cT {#1}
7868   {
7869     \exp_args:Nv \pgfkeysalso {#1}
7870   }
7871 },
7872 /handlers/.update~style/.code={
7873   \tl_if_eq:nnF {#1} {\pgfkeysnovalue}
7874   {
7875     \pgfkeys{\pgfkeyscurrentpath/.code=\pgfkeysalso{#1}}
7876   }
7877 },
7878 /handlers/.update~value/.code={
7879   \tl_if_eq:nnF {#1} {\pgfkeysnovalue}
7880   {
7881     \pgfkeyssetvalue{\pgfkeyscurrentpath}{#1}
7882   }
7883 },
7884 }

```

Some wrappers around the TikZ keys.

```

7885 \NewDocumentCommand \pen { 0{} }
7886 {
7887   \path[define~ pen,every~ calligraphy~ pen/.try,#1]
7888 }
7889

```



```

7890 \NewDocumentCommand \definepen { 0{} }
7891 {
7892   \tikz \path[define~ pen,every~ calligraphy~ pen/.try,#1]
7893 }
7894
7895 \NewDocumentCommand \calligraphy { 0{} }
7896 {
7897   \path[use~ pen,every~ calligraphy/.try,#1]
7898 }

```

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

```

7899 \cs_new_protected_nopar:Npn \cal_path_create:NN #1#2
7900 {
7901   \int_zero:N \l__cal_tmpa_int
7902   \seq_map_inline:Nn #1
7903   {
7904     \int_compare:nT {\tl_count:n {##1} > 3}
7905     {
7906
7907       \int_incr:N \l__cal_tmpa_int
7908       \int_zero:N \l__cal_tmpb_int
7909
7910       \tl_set:Nn \l__cal_tmp_path_tl {##1}
7911       \spath_open:N \l__cal_tmp_path_tl
7912       \spath_reverse:NV \l__cal_tmp_rpath_tl \l__cal_tmp_path_tl
7913
7914       \seq_map_inline:Nn #2
7915       {
7916         \int_incr:N \l__cal_tmpb_int
7917         \group_begin:
7918         \pgfsys@beginscope
7919         \cal_apply_style:c {l__cal_stroke_style_ \int_use:N \l__cal_tmpa_int}
7920         \cal_apply_style:c {l__cal_stroke_inline_style_ \int_use:N \l__cal_tmpa_int}
7921         \cal_apply_style:c {l__cal_nib_style_ \int_use:N \l__cal_tmpb_int}
7922
7923         \spath_initialpoint:Nn \l__cal_tmpa_tl {####1}
7924         \tl_set_eq:NN \l__cal_tmp_patha_tl \l__cal_tmp_path_tl
7925         \spath_translate:NV \l__cal_tmp_patha_tl \l__cal_tmpa_tl
7926
7927         \int_compare:nTF {\tl_count:n {####1} == 3}
7928         {
7929           \cal_at_least_three:N \l__cal_tmp_patha_tl
7930           \spath_protocol_path:V \l__cal_tmp_patha_tl
7931
7932           \tikz@options
7933           \dim_set:Nn \l__cal_line_width_dim {\pgflinewidth}
7934           \cal_maybe_taper:N \l__cal_tmp_patha_tl
7935         }
7936         {
7937           \spath_weld:Nn \l__cal_tmp_patha_tl {####1}

```

```

7938 \spath_weld:N \l__cal_tmp_patha_tl \l__cal_tmp_rpath_tl
7939 \spath_reverse:Nn \l__cal_tmp_rpathb_tl {###1}
7940 \spath_weld:N \l__cal_tmp_patha_tl \l__cal_tmp_rpathb_tl
7941
7942 \tl_clear:N \l__cal_tmpa_tl
7943 \tl_set:Nn \l__cal_tmpa_tl
7944 {
7945   fill=\pgfkeysvalueof{/tikz/pen-colour},
7946   draw=none
7947 }
7948 \tl_if_exist:cT {l__cal_stroke_style_ \int_use:N \l__cal_tmpa_int}
7949 {
7950   \tl_put_right:Nv \l__cal_tmpa_tl
7951   {l__cal_stroke_style_ \int_use:N \l__cal_tmpa_int}
7952 }
7953 \tl_if_exist:cT {l__cal_stroke_inline_style_ \int_use:N \l__cal_tmpa_int}
7954 {
7955   \tl_put_right:Nn \l__cal_tmpa_tl {,}
7956   \tl_put_right:Nv \l__cal_tmpa_tl
7957   {l__cal_stroke_inline_style_ \int_use:N \l__cal_tmpa_int}
7958 }
7959 \tl_if_exist:cT {l__cal_nib_style_ \int_use:N \l__cal_tmpb_int}
7960 {
7961   \tl_put_right:Nn \l__cal_tmpa_tl {,}
7962   \tl_put_right:Nv \l__cal_tmpa_tl
7963   {l__cal_nib_style_ \int_use:N \l__cal_tmpb_int}
7964 }
7965 \spath_tikz_path:VV \l__cal_tmpa_tl \l__cal_tmp_patha_tl
7966 }
7967 \pgfsys@endscope
7968 \group_end:
7969 }
7970
7971 \bool_if:NT \l__cal_annotate_bool
7972 {
7973   \seq_get_right:NN #2 \l__cal_tmpa_tl
7974   \spath_finalpoint:N \l__cal_tmpa_tl \l__cal_tmpa_tl
7975   \spath_translate:N \l__cal_tmp_path_tl \l__cal_tmpa_tl
7976   \tikz@scan@one@point
7977   \pgfutil@firstofone
7978   \pgfkeysvalueof{/tikz/annotation-shift}
7979
7980   \spath_translate:Nnn \l__cal_tmp_path_tl {\pgf@x} {\pgf@y}
7981
7982   \pgfkeysgetvalue{/tikz/annotation-style}{\l__cal_tmpa_tl}
7983   \spath_tikz_path:VV \l__cal_tmpa_tl \l__cal_tmp_path_tl
7984
7985   \spath_finalpoint:N \l__cal_tmpa_tl \l__cal_tmp_path_tl
7986
7987   \exp_last_unbraced:N \pgfqpoint \l__cal_tmpa_tl
7988   \begin{scope}[reset~ cm]
7989   \node[
7990     every-annotation-node/.try,
7991     tl-use:c = {l__cal_annotation_style_ \int_use:N \l__cal_tmpa_int _tl}

```

```

7992         ] at (\pgf@x,\pgf@y) {\int_use:N \l__cal_tmpa_int};
7993     \end{scope}
7994 }
7995 }
7996 }
7997 }
7998 \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.

```

7999 \cs_new_eq:NN \cal_orig_moveto:n \pgfpathmoveto
8000 \cs_new_nopar:Npn \cal_moveto:n #1
8001 {
8002     \int_gincr:N \g__cal_path_component_int
8003     \cal_orig_moveto:n {#1}
8004 }

```

(End of definition for \cal_moveto:n.)

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

```

8005 \cs_new_nopar:Npn \cal_apply_style:N #1
8006 {
8007     \tl_if_exist:NT #1 {
8008         \exp_args:NV \tikzset #1
8009     }
8010 }
8011 \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.

```

8012 \cs_new_protected_nopar:Npn \cal_at_least_three:Nn #1#2
8013 {
8014     \spath_reallength:Nn \l__cal_tmpa_int {#2}
8015     \tl_clear:N \l__cal_tmpb_tl
8016     \tl_set:Nn \l__cal_tmpb_tl {#2}
8017     \int_compare:nTF {\l__cal_tmpa_int = 1}
8018     {
8019         \spath_split_at:Nn \l__cal_tmpb_tl {2/3}
8020         \spath_split_at:Nn \l__cal_tmpb_tl {1/2}
8021     }
8022     {
8023         \int_compare:nT {\l__cal_tmpa_int = 2}
8024         {
8025             \spath_split_at:Nn \l__cal_tmpb_tl {1.5}
8026             \spath_split_at:Nn \l__cal_tmpb_tl {.5}
8027         }
8028     }
8029     \tl_set_eq:NN #1 \l__cal_tmpb_tl
8030 }
8031 \cs_generate_variant:Nn \cal_at_least_three:Nn {NV}
8032 \cs_new_protected_nopar:Npn \cal_at_least_three:N #1

```

```

8033 {
8034   \cal_at_least_three:Nv #1#1
8035 }
8036 \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.

```

8037 \cs_new_protected_nopar:Npn \cal_maybe_taper:N #1
8038 {
8039   \tl_set_eq:NN \l__cal_tmpa_tl #1
8040
8041   \bool_if:NT \l__cal_taper_start_bool
8042   {
8043
8044     \dim_set:Nn \l__cal_tmpa_dim {\tl_item:Nn \l__cal_tmpa_tl {2}}
8045     \dim_set:Nn \l__cal_tmpb_dim {\tl_item:Nn \l__cal_tmpa_tl {3}}
8046     \tl_set:Nx \l__cal_tmpb_tl {\tl_item:Nn \l__cal_tmpa_tl {4}}
8047
8048     \token_case_meaning:NnF \l__cal_tmpb_tl
8049     {
8050       \c_spath_lineto_tl
8051       {
8052
8053         \bool_set_true:N \l__cal_taperable_bool
8054         \dim_set:Nn \l__cal_tmpg_dim {\tl_item:Nn \l__cal_tmpa_tl {5}}
8055         \dim_set:Nn \l__cal_tmph_dim {\tl_item:Nn \l__cal_tmpa_tl {6}}
8056         \dim_set:Nn \l__cal_tmpe_dim {(2\l__cal_tmpa_dim + \l__cal_tmpg_dim)/3}
8057         \dim_set:Nn \l__cal_tmpe_dim {(2\l__cal_tmph_dim + \l__cal_tmpe_dim)/3}
8058         \dim_set:Nn \l__cal_tmpe_dim {(\l__cal_tmpe_dim + 2\l__cal_tmph_dim)/3}
8059         \dim_set:Nn \l__cal_tmpe_dim {(\l__cal_tmpe_dim + 2\l__cal_tmph_dim)/3}
8060         \prg_replicate:nn {4}
8061         {
8062           \tl_set:Nx \l__cal_tmpa_tl {\tl_tail:N \l__cal_tmpa_tl}
8063         }
8064         \tl_put_left:Nv \l__cal_tmpa_tl \c_spath_moveto_tl
8065       }
8066       \c_spath_curvetoa_tl
8067       {
8068         \bool_set_true:N \l__cal_taperable_bool
8069         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {5}}
8070         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {6}}
8071         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {8}}
8072         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {9}}
8073         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {11}}
8074         \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {12}}
8075         \prg_replicate:nn {10}
8076         {
8077           \tl_set:Nx \l__cal_tmpa_tl {\tl_tail:N \l__cal_tmpa_tl}
8078         }
8079         \tl_put_left:Nv \l__cal_tmpa_tl \c_spath_moveto_tl
8080       }
8081     }
8082   {

```

```

8083     \bool_set_false:N \l__cal_taperable_bool
8084 }
8085
8086 \bool_if:NT \l__cal_taperable_bool
8087 {
8088     \__cal_taper_aux:
8089 }
8090
8091 }
8092
8093 \bool_if:NT \l__cal_taper_end_bool
8094 {
8095
8096     \dim_set:Nn \l__cal_tmpa_dim {\tl_item:Nn \l__cal_tmpa_tl {-2}}
8097     \dim_set:Nn \l__cal_tmpb_dim {\tl_item:Nn \l__cal_tmpa_tl {-1}}
8098     \tl_set:Nx \l__cal_tmpb_tl {\tl_item:Nn \l__cal_tmpa_tl {-3}}
8099
8100     \token_case_meaning:NnF \l__cal_tmpb_tl
8101     {
8102         \c_spath_lineto_tl
8103         {
8104
8105             \bool_set_true:N \l__cal_taperable_bool
8106             \dim_set:Nn \l__cal_tmpg_dim {\tl_item:Nn \l__cal_tmpa_tl {-5}}
8107             \dim_set:Nn \l__cal_tmph_dim {\tl_item:Nn \l__cal_tmpa_tl {-4}}
8108             \dim_set:Nn \l__cal_tmpe_dim {(2\l__cal_tmpa_dim + \l__cal_tmph_dim)/3}
8109             \dim_set:Nn \l__cal_tmpe_dim {(2\l__cal_tmph_dim + \l__cal_tmpe_dim)/3}
8110             \dim_set:Nn \l__cal_tmpe_dim {(\l__cal_tmpa_dim + 2\l__cal_tmpe_dim)/3}
8111             \dim_set:Nn \l__cal_tmpe_dim {(\l__cal_tmpe_dim + 2\l__cal_tmph_dim)/3}
8112             \tl_reverse:N \l__cal_tmpa_tl
8113             \prg_replicate:nn {3}
8114             {
8115                 \tl_set:Nx \l__cal_tmpa_tl {\tl_tail:N \l__cal_tmpa_tl}
8116             }
8117             \tl_reverse:N \l__cal_tmpa_tl
8118         }
8119         \c_spath_curveto_tl
8120         {
8121             \bool_set_true:N \l__cal_taperable_bool
8122             \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-5}}
8123             \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-4}}
8124             \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-8}}
8125             \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-7}}
8126             \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-11}}
8127             \dim_set:Nn \l__cal_tmpe_dim {\tl_item:Nn \l__cal_tmpa_tl {-10}}
8128             \tl_reverse:N \l__cal_tmpe_dim
8129             \prg_replicate:nn {9}
8130             {
8131                 \tl_set:Nx \l__cal_tmpa_tl {\tl_tail:N \l__cal_tmpe_dim}
8132             }
8133             \tl_reverse:N \l__cal_tmpe_dim
8134         }
8135     }
8136     {

```

```

8137     \bool_set_false:N \l__cal_taperable_bool
8138   }
8139
8140   \bool_if:NT \l__cal_taperable_bool
8141   {
8142     \__cal_taper_aux:
8143   }
8144
8145 }
8146
8147 \pgfsyssoftpath@setcurrentpath\l__cal_tmpa_tl
8148 \pgfsetstrokecolor{\pgfkeysvalueof{/tikz/pen-colour}}
8149 \pgfusepath{stroke}
8150
8151 }

```

(End of definition for \cal_maybe_taper:N.)

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

```

8152 \cs_new_protected_nopar:Npn \__cal_taper_aux:
8153 {
8154   \tl_clear:N \l__cal_tmpb_tl
8155   \tl_put_right:NV \l__cal_tmpb_tl \c_spath_moveto_tl
8156
8157   \fp_set:Nn \l__cal_tmpa_fp
8158   {
8159     \l__cal_tmpd_dim - \l__cal_tmpb_dim
8160   }
8161   \fp_set:Nn \l__cal_tmpb_fp
8162   {
8163     \l__cal_tmpa_dim - \l__cal_tmpe_dim
8164   }
8165   \fp_set:Nn \l__cal_tmpe_fp
8166   {
8167     ( $\l__cal_tmpa\_fp^2 + \l__cal_tmpb\_fp^2$ ).5
8168   }
8169
8170   \fp_set:Nn \l__cal_tmpa_fp
8171   {
8172     .5*\l__cal_taper_width_dim
8173     *
8174     \l__cal_tmpa_fp / \l__cal_tmpe_fp
8175   }
8176   \fp_set:Nn \l__cal_tmpb_fp
8177   {
8178     .5*\l__cal_taper_width_dim
8179     *
8180     \l__cal_tmpb_fp / \l__cal_tmpe_fp
8181   }
8182
8183   \fp_set:Nn \l__cal_tmpe_fp
8184   {
8185     \l__cal_tmph_dim - \l__cal_tmpe_dim
8186   }

```

```

8187 \fp_set:Nn \l__cal_tmpd_fp
8188 {
8189   \l__cal_tmpe_dim - \l__cal_tmpg_dim
8190 }
8191 \fp_set:Nn \l__cal_tmpe_fp
8192 {
8193   (\l__cal_tmpe_fp^2 + \l__cal_tmpd_fp^2)^.5
8194 }
8195
8196 \fp_set:Nn \l__cal_tmpe_fp
8197 {
8198   .5*\l__cal_line_width_dim
8199   *
8200   \l__cal_tmpe_fp / \l__cal_tmpe_fp
8201 }
8202 \fp_set:Nn \l__cal_tmpd_fp
8203 {
8204   .5*\l__cal_line_width_dim
8205   *
8206   \l__cal_tmpd_fp / \l__cal_tmpe_fp
8207 }
8208
8209 \tl_put_right:Nx \l__cal_tmpe_tl
8210 {
8211   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim}}
8212   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim}}
8213 }
8214
8215 \tl_put_right:NV \l__cal_tmpe_tl \c_spath_curvetoa_tl
8216
8217 \tl_put_right:Nx \l__cal_tmpe_tl
8218 {
8219   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim}}
8220   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim}}
8221 }
8222
8223 \tl_put_right:NV \l__cal_tmpe_tl \c_spath_curvetob_tl
8224
8225 \tl_put_right:Nx \l__cal_tmpe_tl
8226 {
8227   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim}}
8228   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim}}
8229 }
8230
8231 \tl_put_right:NV \l__cal_tmpe_tl \c_spath_curvetoa_tl
8232
8233 \tl_put_right:Nx \l__cal_tmpe_tl
8234 {
8235   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim}}
8236   {\dim_eval:n { \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim}}
8237 }
8238
8239 \tl_put_right:NV \l__cal_tmpe_tl \c_spath_curvetoa_tl
8240

```

```

8241 \tl_put_right:Nx \l__cal_tmpb_tl
8242 {
8243   {
8244     \dim_eval:n
8245     {
8246       \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim
8247       - \fp_to_dim:n{ 1.32 * \l__cal_tmpe_fp
8248     }
8249   }
8250 }
8251 {
8252   \dim_eval:n
8253   {
8254     \fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim
8255     + \fp_to_dim:n {1.32* \l__cal_tmpe_fp
8256   }
8257 }
8258 }
8259 }
8260
8261 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curvetob_tl
8262
8263 \tl_put_right:Nx \l__cal_tmpb_tl
8264 {
8265   {
8266     \dim_eval:n
8267     {
8268       -\fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim
8269       - \fp_to_dim:n {1.32 * \l__cal_tmpe_fp
8270     }
8271   }
8272 }
8273 {
8274   \dim_eval:n
8275   {
8276     -\fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim
8277     + \fp_to_dim:n {1.32 * \l__cal_tmpe_fp
8278   }
8279 }
8280 }
8281 }
8282
8283 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curveto_tl
8284
8285 \tl_put_right:Nx \l__cal_tmpb_tl
8286 {
8287   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim}}
8288   {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_fp + \l__cal_tmpe_dim}}
8289 }
8290
8291 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curvetoa_tl
8292
8293 \tl_put_right:Nx \l__cal_tmpb_tl
8294 {

```



```

8295     {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim}}
8296     {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim}}
8297 }
8298
8299 \tl_put_right:NV \l__cal_tmpe_dim \c_spath_curvetob_tl
8300
8301 \tl_put_right:Nx \l__cal_tmpe_dim
8302 {
8303     {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim}}
8304     {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim}}
8305 }
8306
8307 \tl_put_right:NV \l__cal_tmpe_dim \c_spath_curveto_tl
8308
8309 \tl_put_right:Nx \l__cal_tmpe_dim
8310 {
8311     {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim}}
8312     {\dim_eval:n { -\fp_to_dim:N \l__cal_tmpe_dim}}
8313 }
8314
8315 \tl_put_right:NV \l__cal_tmpe_dim \c_spath_curvetoa_tl
8316
8317 \tl_put_right:Nx \l__cal_tmpe_dim
8318 {
8319     {
8320         \dim_eval:n
8321         {
8322             -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim
8323             + \fp_to_dim:n{ 1.32 * \l__cal_tmpe_dim}
8324         }
8325     }
8326     {
8327         \dim_eval:n
8328         {
8329             -\fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim
8330             - \fp_to_dim:n {1.32* \l__cal_tmpe_dim}
8331         }
8332     }
8333 }
8334
8335 \tl_put_right:NV \l__cal_tmpe_dim \c_spath_curvetob_tl
8336
8337 \tl_put_right:Nx \l__cal_tmpe_dim
8338 {
8339     {
8340         \dim_eval:n
8341         {
8342             \fp_to_dim:N \l__cal_tmpe_dim + \l__cal_tmpe_dim
8343             + \fp_to_dim:n {1.32 * \l__cal_tmpe_dim}
8344         }
8345     }
8346     {
8347         \dim_eval:n
8348         {

```

```

8349         \fp_to_dim:N \l__cal_tmpb_fp + \l__cal_tmpb_dim
8350         - \fp_to_dim:n {1.32 * \l__cal_tmpa_fp}
8351     }
8352 }
8353 }
8354
8355 \tl_put_right:NV \l__cal_tmpb_tl \c_spath_curveto_tl
8356
8357 \tl_put_right:Nx \l__cal_tmpb_tl
8358 {
8359     {\dim_eval:n { \fp_to_dim:N \l__cal_tmpa_fp + \l__cal_tmpa_dim}}
8360     {\dim_eval:n { \fp_to_dim:N \l__cal_tmpb_fp + \l__cal_tmpb_dim}}
8361 }
8362
8363 \pgfsyssoftpath@setcurrentpath\l__cal_tmpb_tl
8364 \pgfsetfillcolor{\pgfkeysvalueof{/tikz/pen~colour}}
8365 \pgfusepath{fill}
8366 }

```

(End of definition for `__cal_taper_aux:`)

Defines a copperplate pen.

```

8367 \tl_set:Nn \l__cal_tmpa_tl {\pgfsyssoftpath@movetotoken{Opt}{Opt}}
8368 \spath_components_to_seq:NV \l__cal_tmpa_seq \l__cal_tmpa_tl
8369 \seq_gclear_new:N \g__cal_pen_copperplate_seq
8370 \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.

```

8371 \DeclareDocumentCommand \CopperplatePath { m }
8372 {
8373     \spath_components_to_seq:NV \l__cal_tmpa_seq #1
8374     \cal_path_create:NN \l__cal_tmpa_seq \g__cal_pen_copperplate_seq
8375 }

```

(End of definition for `\CopperplatePath`.)

```

8376 \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.

```

8377 \expandafter\ifx\curname pgfdeclarededecoration\endcsname\relax
8378 \else
8379 \pgfdeclarededecoration{calligraphic brace}{brace}%
8380 {%
8381     \state{brace}[width=+\pgfdecoratedremainingdistance,next state=final]%
8382     {%
8383         \pgfsyssoftpath@setcurrentpath{\pgfutil@empty}%
8384         \pgfpathmoveto{\pgfpointorigin}%
8385         \pgfpathcurveto%
8386         {%
8387             \pgfpoint%
8388             {.15\pgfdecorationsegmentamplitude}%

```

```

8389     {.3\pgfdecorationsegmentamplitude}%
8390 }%
8391 {%
8392     \pgfqpoint%
8393     {.5\pgfdecorationsegmentamplitude}%
8394     {.5\pgfdecorationsegmentamplitude}%
8395 }%
8396 {%
8397     \pgfqpoint%
8398     {\pgfdecorationsegmentamplitude}%
8399     {.5\pgfdecorationsegmentamplitude}%
8400 }%
8401 {%
8402     \pgftransformxshift%
8403     {+\pgfdecorationsegmentaspect\pgfdecoratedremainingdistance}%
8404     \pgfpathlineto%
8405     {%
8406         \pgfqpoint%
8407         {-\pgfdecorationsegmentamplitude}%
8408         {.5\pgfdecorationsegmentamplitude}%
8409     }%
8410     \pgfpathcurveto
8411     {%
8412         \pgfqpoint%
8413         {- .5\pgfdecorationsegmentamplitude}%
8414         {.5\pgfdecorationsegmentamplitude}%
8415     }%
8416     {%
8417         \pgfqpoint%
8418         {- .15\pgfdecorationsegmentamplitude}%
8419         {.7\pgfdecorationsegmentamplitude}%
8420     }%
8421     {%
8422         \pgfqpoint%
8423         {0\pgfdecorationsegmentamplitude}%
8424         {1\pgfdecorationsegmentamplitude}%
8425     }%
8426     \pgfpathmoveto%
8427     {%
8428         \pgfqpoint%
8429         {0\pgfdecorationsegmentamplitude}%
8430         {1\pgfdecorationsegmentamplitude}%
8431     }%
8432     \pgfpathcurveto%
8433     {%
8434         \pgfqpoint%
8435         {.15\pgfdecorationsegmentamplitude}%
8436         {.7\pgfdecorationsegmentamplitude}%
8437     }%
8438     {%
8439         \pgfqpoint%
8440         {.5\pgfdecorationsegmentamplitude}%
8441         {.5\pgfdecorationsegmentamplitude}%
8442     }%

```

```

8443     {%
8444         \pgfqpoint%
8445         {\pgfdecorationsegmentamplitude}%
8446         {.5\pgfdecorationsegmentamplitude}%
8447     }%
8448 }%
8449 {%
8450     \pgftransformxshift{+\pgfdecoratedremainingdistance}%
8451     \pgfpathlineto%
8452     {%
8453         \pgfqpoint%
8454         {-\pgfdecorationsegmentamplitude}%
8455         {.5\pgfdecorationsegmentamplitude}%
8456     }%
8457     \pgfpathcurveto%
8458     {%
8459         \pgfqpoint%
8460         {- .5\pgfdecorationsegmentamplitude}%
8461         {.5\pgfdecorationsegmentamplitude}%
8462     }%
8463     {%
8464         \pgfqpoint%
8465         {- .15\pgfdecorationsegmentamplitude}%
8466         {.3\pgfdecorationsegmentamplitude}%
8467     }%
8468     {\pgfqpoint{Opt}{Opt}}%
8469 }%
8470 \tikzset{%
8471     taper width=.5\pgflinewidth,%
8472     taper%
8473 }%%
8474 \pgfsyssoftpath@getcurrentpath\cal@tmp@path%
8475 \CopperplatePath{\cal@tmp@path}%
8476 }%
8477 \state{final}{}%
8478 }%

```

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

```

8479 \pgfdeclaredecoration{calligraphic straight parenthesis}{brace}
8480 {
8481     \state{brace}[width=+\pgfdecoratedremainingdistance,next state=final]%
8482     {%
8483         \pgfsyssoftpath@setcurrentpath{\pgfutil@empty}%
8484         \pgfpathmoveto{\pgfpointorigin}%
8485         \pgfpathcurveto%
8486         {%
8487             \pgfqpoint%
8488             {.76604\pgfdecorationsegmentamplitude}%
8489             {.64279\pgfdecorationsegmentamplitude}%
8490         }%
8491         {%
8492             \pgfqpoint%
8493             {2.3333\pgfdecorationsegmentamplitude}%

```

```

8494     {\pgfdecorationsegmentamplitude}%
8495 }%
8496 {%
8497   \pgfqpoint%
8498   {3.3333\pgfdecorationsegmentamplitude}%
8499   {\pgfdecorationsegmentamplitude}%
8500 }%
8501 {%
8502   \pgftransformxshift{+\pgfdecoratedremainingdistance}%
8503   \pgfpathlineto%
8504   {%
8505     \pgfqpoint%
8506     {-3.3333\pgfdecorationsegmentamplitude}%
8507     {\pgfdecorationsegmentamplitude}%
8508   }%
8509   \pgfpathcurveto%
8510   {%
8511     \pgfqpoint%
8512     {-2.3333\pgfdecorationsegmentamplitude}%
8513     {\pgfdecorationsegmentamplitude}%
8514   }%
8515   {%
8516     \pgfqpoint%
8517     {-.76604\pgfdecorationsegmentamplitude}%
8518     {.64279\pgfdecorationsegmentamplitude}%
8519   }%
8520   {\pgfqpoint{0pt}{0pt}}%
8521 }%
8522 \tikzset{%
8523   taper width=.5\pgflinewidth,%
8524   taper%
8525 }%
8526 \pgfsyssoftpath@getcurrentpath\cal@tmp@path%
8527 \CopperplatePath{\cal@tmp@path}%
8528 }%
8529 \state{final}{}%
8530 }

```

The third is a curved parenthesis.

```

8531 \pgfdeclaredecoration{calligraphic curved parenthesis}{brace}
8532 {
8533   \state{brace}[width=+\pgfdecoratedremainingdistance,next state=final]%
8534   {%
8535     \pgfsyssoftpath@setcurrentpath{\pgfutil@empty}%
8536     \pgfpathmoveto{\pgfpointorigin}%
8537     \pgf@xa=\pgfdecoratedremainingdistance\relax%
8538     \advance\pgf@xa by -1.5890\pgfdecorationsegmentamplitude\relax%
8539     \edef\cgrphy@xa{\the\pgf@xa}%
8540     \pgfpathcurveto%
8541     {%
8542       \pgfqpoint%
8543       {1.5890\pgfdecorationsegmentamplitude}%
8544       {1.3333\pgfdecorationsegmentamplitude}%
8545     }%
8546     {\pgfqpoint{\cgrphy@xa}{1.3333\pgfdecorationsegmentamplitude}}%

```

```

8547     {\pgfqpoint{\pgfdecoratedremainingdistance}{0pt}}}%
8548     \tikzset{%
8549         taper width=.5\pgflinewidth,%
8550         taper%
8551     }%
8552     \pgfsyssoftpath@getcurrentpath\cal@tmp@path%
8553     \CopperplatePath{\cal@tmp@path}%
8554 }%
8555 \state{final}{}%
8556 }

```

End the conditional for if pgfdecoration module is loaded

```

8557 \fi

```

6 Drawing Knots

```

8558 <@=knot>

```

6.1 Initialisation

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

```

8559 \RequirePackage{spath3}
8560 \usetikzlibrary{intersections,spath3}
8561
8562 \ExplSyntaxOn
8563
8564 \tl_new:N \l__knot_tmpa_tl
8565 \tl_new:N \l__knot_tmpb_tl
8566 \tl_new:N \l__knot_tmpc_tl
8567 \tl_new:N \l__knot_tmpd_tl
8568 \tl_new:N \l__knot_tmpg_tl
8569 \tl_new:N \l__knot_redraws_tl
8570 \tl_new:N \l__knot_clip_width_tl
8571 \tl_new:N \l__knot_name_tl
8572 \tl_new:N \l__knot_node_tl
8573 \tl_new:N \l__knot_aux_tl
8574 \tl_new:N \l__knot_auxa_tl
8575 \tl_new:N \l__knot_prefix_tl
8576
8577 \seq_new:N \l__knot_segments_seq
8578
8579 \int_new:N \l__knot_tmpa_int
8580 \int_new:N \l__knot_strands_int
8581 \int_new:N \g__knot_intersections_int
8582 \int_new:N \g__knot_filaments_int
8583 \int_new:N \l__knot_component_start_int
8584
8585 \fp_new:N \l__knot_tmpa_fp
8586 \fp_new:N \l__knot_tmpb_fp
8587
8588 \dim_new:N \l__knot_tmpa_dim
8589 \dim_new:N \l__knot_tmpb_dim
8590 \dim_new:N \l__knot_tolerance_dim
8591 \dim_new:N \l__knot_redraw_tolerance_dim

```

```

8592 \dim_new:N \l__knot_clip_bg_radius_dim
8593 \dim_new:N \l__knot_clip_draw_radius_dim
8594
8595 \bool_new:N \l__knot_draft_bool
8596 \bool_new:N \l__knot_ignore_ends_bool
8597 \bool_new:N \l__knot_self_intersections_bool
8598 \bool_new:N \l__knot_splits_bool
8599 \bool_new:N \l__knot_super_draft_bool
8600
8601 \bool_new:N \l__knot_prepend_prev_bool
8602 \bool_new:N \l__knot_append_next_bool
8603 \bool_new:N \l__knot_skip_bool
8604 \bool_new:N \l__knot_save_bool
8605 \bool_new:N \l__knot_debugging_bool
8606
8607 \seq_new:N \g__knot_nodes_seq
8608
8609 \bool_set_true:N \l__knot_ignore_ends_bool
      Configuration is via TikZ keys and styles.
8610 \tikzset{
8611   spath/prefix/knot/.style={
8612     spath/set~ prefix=knot strand,
8613   },
8614   spath/suffix/knot/.style={
8615     spath/set~ suffix={},
8616   },
8617   knot/.code={
8618     \tl_if_eq:nnTF {#1} {none}
8619     {
8620       \tikz@addmode{\tikz@mode@doublefalse}
8621     }
8622     {
8623       \tikz@addmode{\tikz@mode@doubletrue}
8624       \tl_if_eq:nnTF {\pgfkeysnovalue} {#1}
8625       {
8626         \tikz@addoption{\pgfsetinnerstrokecolor{.}}
8627       }
8628       {
8629         \pgfsetinnerstrokecolor{#1}
8630       }
8631       \tikz@addoption{
8632         \pgfsetstrokecolor{knotbg}
8633       }
8634       \tl_set:Nn \tikz@double@setup{
8635         \pgfsetinnerlinewidth{\pgflinewidth}
8636         \pgfsetlinewidth{\dim_eval:n {\tl_use:N \l__knot_gap_tl \pgflinewidth}}
8637       }
8638     }
8639   },
8640   knot~ gap/.store~ in=\l__knot_gap_tl,
8641   knot~ gap=3,
8642   knot~ diagram/.is~family,
8643   knot~ diagram/.unknown/.code={
8644     \tl_set_eq:NN \l__knot_tmpa_tl \pgfkeyscurrentname

```

```

8645     \pgfkeysalso{
8646       /tikz/\l__knot_tmpa_tl=#1
8647     }
8648   },
8649   background~ colour/.code={%
8650     \colorlet{knotbg}{#1}%
8651   },
8652   background~ color/.code={%
8653     \colorlet{knotbg}{#1}%
8654   },
8655   background~ colour=white,
8656   knot~ diagram,
8657   name/.store~ in=\l__knot_name_tl,
8658   name={knot},
8659   save~ intersections/.is~ choice,
8660   save~ intersections/.default=true,
8661   save~ intersections/true/.code={
8662     \bool_set_true:N \l__knot_save_bool
8663   },
8664   save~ intersections/false/.code={
8665     \bool_set_false:N \l__knot_save_bool
8666   },
8667   every~ strand/.style={draw},
8668   ignore~ endpoint~ intersections/.code={
8669     \tl_if_eq:nnTF {#1} {true}
8670     {
8671       \bool_set_true:N \l__knot_ignore_ends_bool
8672     }
8673     {
8674       \bool_set_false:N \l__knot_ignore_ends_bool
8675     }
8676   },
8677   ignore~ endpoint~ intersections/.default=true,
8678   consider~ self~ intersections/.is~choice,
8679   consider~ self~ intersections/true/.code={
8680     \bool_set_true:N \l__knot_self_intersections_bool
8681     \bool_set_true:N \l__knot_splits_bool
8682   },
8683   consider~ self~ intersections/false/.code={
8684     \bool_set_false:N \l__knot_self_intersections_bool
8685     \bool_set_false:N \l__knot_splits_bool
8686   },
8687   consider~ self~ intersections/no~ splits/.code={
8688     \bool_set_true:N \l__knot_self_intersections_bool
8689     \bool_set_false:N \l__knot_splits_bool
8690   },
8691   consider~ self~ intersections/.default={true},
8692   clip~ radius/.code={
8693     \dim_set:Nn \l__knot_clip_bg_radius_dim {#1}
8694     \dim_set:Nn \l__knot_clip_draw_radius_dim {#1+2pt}
8695   },
8696   clip~ draw~ radius/.code={
8697     \dim_set:Nn \l__knot_clip_draw_radius_dim {#1}
8698   },

```



```

8699 clip~ background~ radius/.code={
8700   \dim_set:Nn \l__knot_clip_bg_radius_dim {#1}
8701 },
8702 clip~ radius=10pt,
8703 end~ tolerance/.code={
8704   \dim_set:Nn \l__knot_tolerance_dim {#1}
8705 },
8706 end~ tolerance=14pt,
8707 clip/.style={
8708   clip
8709 },
8710 background~ clip/.style={
8711   clip
8712 },
8713 clip~ width/.code={
8714   \tl_set:Nn \l__knot_clip_width_tl {#1}
8715 },
8716 clip~ width=3,
8717 flip~ crossing/.code={%
8718   \tl_clear_new:c {l__knot_crossing_#1}
8719   \tl_set:cn {l__knot_crossing_#1} {x}
8720 },
8721 ignore~ crossing/.code={%
8722   \tl_clear_new:c {l__knot_ignore_crossing_#1}
8723   \tl_set:cn {l__knot_ignore_crossing_#1} {x}
8724 },
8725 draft~ mode/.is~ choice,
8726 draft~ mode/off/.code={%
8727   \bool_set_false:N \l__knot_draft_bool
8728   \bool_set_false:N \l__knot_super_draft_bool
8729 },
8730 draft~ mode/crossings/.code={%
8731   \bool_set_true:N \l__knot_draft_bool
8732   \bool_set_false:N \l__knot_super_draft_bool
8733 },
8734 draft~ mode/strands/.code={%
8735   \bool_set_true:N \l__knot_draft_bool
8736   \bool_set_true:N \l__knot_super_draft_bool
8737 },
8738 debug/.is~ choice,
8739 debug/true/.code={
8740   \bool_set_true:N \l__knot_debugging_bool
8741 },
8742 debug/false/.code={
8743   \bool_set_false:N \l__knot_debugging_bool
8744 },
8745 debug/.default=true,
8746 draft/.is~ family,
8747 draft,
8748 crossing~ label/.style={
8749   overlay,
8750   fill=white,
8751   fill~ opacity=.5,
8752   text~ opacity=1,

```

```

8753     text=blue,
8754     pin~ edge={blue,<-}
8755 },
8756 strand~ label/.style={
8757     overlay,
8758     circle,
8759     draw=purple,
8760     fill=white,
8761     fill~ opacity=.5,
8762     text~ opacity=1,
8763     text=purple,
8764     inner~ sep=0pt
8765 },
8766 }

```

`\knot_debug:n` Debugging

```

8767 \cs_new_nopar:Npn \knot_debug:n #1
8768 {
8769     \bool_if:NT \l__knot_debugging_bool
8770     {
8771         \iow_term:n {===Knot~ debug: #1===}
8772     }
8773 }
8774
8775 \cs_new_nopar:Npn \knot_show_strand:n #1
8776 {
8777     \bool_if:NT \l__knot_debugging_bool
8778     {
8779         \iow_term:n {===Knot~ debug: #1===}
8780         \spath_show:v {knot #1}
8781     }
8782 }
8783
8784 \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.

```

8785 \cs_new_nopar:Npn \knot_apply_style:N #1
8786 {
8787     \knot_debug:n {knot~ apply~ style}
8788     \tl_if_exist:NT #1 {
8789         \exp_args:NV \tikzset #1
8790     }
8791 }
8792 \cs_generate_variant:Nn \knot_apply_style:N {c}

```

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

```

8793 \NewDocumentCommand \flipcrossings {m}
8794 {
8795     \tikzset{knot~ diagram/flip~ crossing/.list={#1}}%
8796 }

```

(End of definition for \flipcrossings.)

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

```

8797 \NewDocumentCommand \strand { 0{} }
8798 {
8799   \int_incr:N \l__knot_strands_int
8800   \tl_clear_new:c {l__knot_options_strand \int_use:N \l__knot_strands_int}
8801   \tl_set:cn {l__knot_options_strand \int_use:N \l__knot_strands_int} {#1}
8802   \path[#1,spath/set~ name=knot,spath/save=\int_use:N \l__knot_strands_int]
8803 }

```

(End of definition for \strand.)

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

```

8804 \NewDocumentEnvironment{knot} { 0{} }
8805 {
8806   \knot_initialise:n {#1}
8807 }
8808 {
8809   \knot_render:
8810 }

```

(End of definition for knot.)

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

```

8811 \cs_new_protected_nopar:Npn \knot_initialise:n #1
8812 {
8813   \knot_debug:n {knot~ initialise}
8814   \tikzset{knot~ diagram/.cd,every~ knot~ diagram/.try,#1}
8815   \int_zero:N \l__knot_strands_int
8816   \tl_clear:N \l__knot_redraws_tl
8817   \seq_gclear:N \g__knot_nodes_seq
8818 }

```

(End of definition for \knot_initialise:n.)

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

```

8819 \cs_new_protected_nopar:Npn \knot_render:
8820 {
8821   \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).

```

8822   \pgfscope
8823   \pgftransformreset

```

Set the dimension for deciding when to include neighbouring strands

```

8824   \dim_set:Nn \l__knot_redraw_tolerance_dim {\fp_to_dim:n
8825     {
8826       sqrt(2) * max(\l__knot_clip_bg_radius_dim, \l__knot_clip_draw_radius_dim)
8827     }
8828   }

```

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

```

8829   \int_step_function:nnnN {1} {1} {\l__knot_strands_int} \knot_draw_strand:n

```

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

```
8830 \bool_if:NF \l__knot_super_draft_bool
8831 {
```

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.

```
8832 \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.

```
8833 \bool_if:NTF \l__knot_self_intersections_bool
8834 {
8835   \knot_split_strands:
8836   \int_set_eq:NN \l__knot_tmpa_int \g__knot_filaments_int
8837   \tl_set:Nn \l__knot_prefix_tl {filament}
8838 }
8839 {
8840   \int_set_eq:NN \l__knot_tmpa_int \l__knot_strands_int
8841   \tl_set:Nn \l__knot_prefix_tl {strand}
8842 }
```

Initialise the intersection count.

```
8843 \int_gzero:N \g__knot_intersections_int
```

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

```
8844 \tl_clear:N \l__knot_node_tl
8845 \bool_if:NT \l__knot_draft_bool
8846 {
8847   \tl_set:Nn \l__knot_node_tl {
8848     \exp_not:N \node[coordinate,
8849       pin={
8850         node~ contents={\int_use:N \g__knot_intersections_int},
8851         knot~ diagram/draft/crossing~ label,
8852         knot~ diagram/draft/crossing~
8853         \int_use:N \g__knot_intersections_int \c_space_tl label/.try
8854       }
8855     }]
8856   }
8857 }
```

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

```
8858 \int_step_variable:nnnNn {1} {1} {\l__knot_tmpa_int - 1} \l__knot_tmpa_tl
8859 {
8860   \int_step_variable:nnnNn
8861   {\tl_use:N \l__knot_tmpa_tl + 1}
8862   {1}
8863   {\l__knot_tmpa_int} \l__knot_tmpb_tl
8864   {
8865     \knot_intersections:VV \l__knot_tmpa_tl \l__knot_tmpb_tl
8866   }
8867 }
```

If any redraws were requested, do them here.

```
8868 \tl_use:N \l__knot_redraws_tl
```

Draw the crossing nodes

```
8869 \seq_use:Nn \g__knot_nodes_seq {}
8870 }
```

Close the scope

```
8871 \endpgfscope
8872 \knot_debug:x {knot~rendered,
8873 ~found~\int_use:N \g__knot_intersections_int \c_space_tl~intersections}
8874 }
```

(End of definition for \knot_render:.)

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

```
8875 \cs_new_protected_nopar:Npn \knot_draw_strand:n #1
8876 {
8877 \knot_debug:n {knot~ draw~ strand~ #1}
8878 \pgfscope
8879 \group_begin:
8880 \spath_bake_round:c {knot strand #1}
8881 \tl_set:Nn \l__knot_tmpa_tl {knot~ diagram/every~ strand/.try,}
8882 \tl_put_right:Nv \l__knot_tmpa_tl {l__knot_options_strand #1}
8883 \tl_put_right:Nn \l__knot_tmpa_tl
8884 {
8885 ,
8886 knot~ diagram/only~ when~ rendering/.try,
8887 only~ when~ rendering/.try,
8888 }
8889 \knot_show_strand:n {strand #1}
8890 \spath_tikz_path:Vv \l__knot_tmpa_tl {knot strand #1}
8891 \group_end:
8892 \endpgfscope
8893 }
8894 \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.

```
8895 \cs_new_protected_nopar:Npn \knot_draw_labels:n #1
8896 {
8897 \knot_debug:n {knot~ draw~ labels}
8898 \bool_if:NT \l__knot_draft_bool
8899 {
8900 \spath_finalpoint:Nv \l__knot_tmpb_tl {knot strand #1}
8901 \dim_set:Nn \l__knot_tmpa_dim {\tl_item:Nn \l__knot_tmpb_tl {1}}
8902 \dim_set:Nn \l__knot_tmpb_dim {\tl_item:Nn \l__knot_tmpb_tl {2}}
8903 \node[
8904 knot~ diagram/draft/strand-label
8905 ] at (\l__knot_tmpa_dim,\l__knot_tmpb_dim) {#1};
8906 \spath_initialpoint:Nv \l__knot_tmpb_tl {knot strand #1}
8907 \dim_set:Nn \l__knot_tmpa_dim {\tl_item:Nn \l__knot_tmpb_tl {1}}
8908 \dim_set:Nn \l__knot_tmpb_dim {\tl_item:Nn \l__knot_tmpb_tl {2}}
8909 \node[
8910 knot~ diagram/draft/strand-label
```

```

8911 ] at (\l__knot_tmpa_dim,\l__knot_tmpb_dim) {#1};
8912 }
8913 \bool_if:nT {
8914   \l__knot_self_intersections_bool
8915   &&
8916   \l__knot_splits_bool
8917 }
8918 {
8919   \tl_clear:N \l__knot_tmpa_tl
8920   \spath_remove_empty_components:c {knot strand #1}
8921   \spath_initialpoint:Nv \l__knot_tmpa_tl {knot strand #1}
8922   \tl_put_left:Nv \l__knot_tmpa_tl \c_spath_moveto_tl
8923   \spath_segments_to_seq:Nv \l__knot_segments_seq {knot strand #1}
8924   \seq_map_function:NN \l__knot_segments_seq \knot_split_self_intersects:N
8925   \tl_set_eq:cN {knot strand #1} \l__knot_tmpa_tl
8926 }
8927 }

```

(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!

```

8928 \cs_new_protected_nopar:Npn \knot_split_self_intersects:N #1
8929 {
8930   \knot_debug:n {knot~ split~ self~ intersects}
8931   \tl_set:Nx \l__knot_tmpc_tl {\tl_item:nn {#1} {4}}
8932   \token_case_meaning:NnF \l__knot_tmpc_tl
8933   {
8934     \c_spath_curvetoa_tl
8935     {
8936       \fp_set:Nn \l__knot_tmpa_fp
8937       {
8938         (\tl_item:nn {#1} {3} - 3 * \tl_item:nn {#1} {6}
8939         + 3 * \tl_item:nn {#1} {9} - \tl_item:nn {#1} {12})
8940         *
8941         (3 * \tl_item:nn {#1} {8} - 3 * \tl_item:nn {#1} {11})
8942         -
8943         (\tl_item:nn {#1} {2} - 3 * \tl_item:nn {#1} {5}
8944         + 3 * \tl_item:nn {#1} {8} - \tl_item:nn {#1} {11})
8945         *
8946         (3 * \tl_item:nn {#1} {9} - 3 * \tl_item:nn {#1} {12})
8947       }
8948       \fp_set:Nn \l__knot_tmpb_fp
8949       {
8950         (\tl_item:nn {#1} {2} - 3 * \tl_item:nn {#1} {5}
8951         + 3 * \tl_item:nn {#1} {8} - \tl_item:nn {#1} {11})
8952         *
8953         (3 * \tl_item:nn {#1} {6} - 6 * \tl_item:nn {#1} {9}
8954         + 3 * \tl_item:nn {#1} {12})
8955         -
8956         (\tl_item:nn {#1} {3} - 3 * \tl_item:nn {#1} {6}
8957         + 3 * \tl_item:nn {#1} {9} - \tl_item:nn {#1} {12})

```

```

8958      *
8959      (3 * \tl_item:nn {#1} {5} - 6 * \tl_item:nn {#1} {8}
8960      + 3 * \tl_item:nn {#1} {11})
8961    }
8962    \fp_compare:nTF
8963    {
8964      \l__knot_tmpb_fp != 0
8965    }
8966    {
8967      \fp_set:Nn \l__knot_tmpa_fp {.5 * \l__knot_tmpa_fp / \l__knot_tmpb_fp}
8968      \bool_if:nTF
8969      {
8970        \fp_compare_p:n {0 < \l__knot_tmpa_fp}
8971        &&
8972        \fp_compare_p:n {\l__knot_tmpa_fp < 1}
8973      }
8974      {
8975        \spath_split_curve:NNnV
8976        \l__knot_tmpc_tl
8977        \l__knot_tmpd_tl
8978        {#1}
8979        \l__knot_tmpa_fp
8980        \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8981        \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8982        \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8983        \tl_set:Nx \l__knot_tmpd_tl {\tl_tail:N \l__knot_tmpd_tl}
8984        \tl_set:Nx \l__knot_tmpd_tl {\tl_tail:N \l__knot_tmpd_tl}
8985        \tl_set:Nx \l__knot_tmpd_tl {\tl_tail:N \l__knot_tmpd_tl}
8986        \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpc_tl
8987        \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpd_tl
8988      }
8989      {
8990        \tl_set:Nn \l__knot_tmpc_tl {#1}
8991        \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8992        \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8993        \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
8994        \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpc_tl
8995      }
8996    }
8997    {
8998      \tl_set:Nn \l__knot_tmpc_tl {#1}
8999      \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
9000      \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
9001      \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
9002      \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpc_tl
9003    }
9004  }
9005  \c_spath_lineto_tl
9006  {
9007    \tl_set:Nn \l__knot_tmpc_tl {#1}
9008    \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
9009    \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
9010    \tl_set:Nx \l__knot_tmpc_tl {\tl_tail:N \l__knot_tmpc_tl}
9011    \tl_put_right:NV \l__knot_tmpa_tl \l__knot_tmpc_tl

```

```

9012     }
9013   }
9014   {
9015     \tl_put_right:Nn \l__knot_tmpa_tl {#1}
9016   }
9017 }

```

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

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

```

9018 \cs_new_protected_nopar:Npn \knot_intersections:nn #1#2
9019 {
9020   \knot_debug:x {knot~ intersections~ between~
9021     \l__knot_prefix_tl \c_space_tl #1~ and~ #2}
9022   \group_begin:
9023   \tl_set_eq:NN \l__knot_tmpa_tl \l__knot_prefix_tl
9024   \tl_put_right:Nn \l__knot_tmpa_tl {#1}
9025   \tl_set_eq:NN \l__knot_tmpb_tl \l__knot_prefix_tl
9026   \tl_put_right:Nn \l__knot_tmpb_tl {#2}
9027   \tl_set_eq:Nc \l__knot_tmpc_tl {knot \tl_use:N \l__knot_tmpa_tl}
9028   \tl_set_eq:Nc \l__knot_tmpe_tl {knot \tl_use:N \l__knot_tmpb_tl}
9029
9030   \bool_if:nTF {
9031     \l__knot_save_bool
9032     &&
9033     \tl_if_exist_p:c {
9034       knot~ intersections~
9035       \tl_use:N \l__knot_name_tl -
9036       \tl_use:N \l__knot_tmpa_tl -
9037       \tl_use:N \l__knot_tmpb_tl
9038     }
9039   }
9040   {
9041     \tl_use:c
9042     {
9043       knot~ intersections~ \tl_use:N \l__knot_name_tl -
9044       \tl_use:N \l__knot_tmpa_tl -
9045       \tl_use:N \l__knot_tmpb_tl
9046     }
9047   }
9048   {
9049     \pgfintersectionofpaths{\pgfsetpath\l__knot_tmpe_tl}{\pgfsetpath\l__knot_tmpe_tl}
9050
9051   }
9052
9053   \knot_debug:x {found~\pgfintersectionsolutions\c_space_tl~ intersections}
9054   \int_compare:nT {\pgfintersectionsolutions > 0}
9055   {
9056     \int_step_function:nnnN
9057     {1}
9058     {1}
9059     {\pgfintersectionsolutions}
9060     \knot_do_intersection:n
9061   }

```



```

9062
9063 \knot_save_intersections:VV \l__knot_tmpa_tl \l__knot_tmpb_tl
9064 \group_end:
9065 }

```

(End of definition for \knot_intersections:nn.)

\knot_save_intersections:nn

```

9066 \cs_new_protected_nopar:Npn \knot_save_intersections:nn #1#2
9067 {
9068   \knot_debug:n {knot~ save~ intersections}
9069   \bool_if:NT \l__knot_save_bool
9070   {
9071     \tl_clear:N \l__knot_aux_tl
9072     \tl_put_right:Nn \l__knot_aux_tl
9073     {
9074       \def\pgfintersectionsolutions
9075     }
9076     \tl_put_right:Nx \l__knot_aux_tl
9077     {
9078       {\int_eval:n {\pgfintersectionsolutions}}
9079     }
9080     \int_compare:nT {\pgfintersectionsolutions > 0}
9081     {
9082       \int_step_inline:nnnn {1} {1} {\pgfintersectionsolutions}
9083       {
9084         \pgfpointintersectionsolution{##1}
9085         \dim_set:Nn \l__knot_tmpa_dim {\pgf@x}
9086         \dim_set:Nn \l__knot_tmpb_dim {\pgf@y}
9087         \tl_put_right:Nn \l__knot_aux_tl
9088         {
9089           \expandafter\def\csname pgfpoint@intersect@solution@##1\endcsname
9090         }
9091         \tl_put_right:Nx \l__knot_aux_tl
9092         {
9093           {
9094             \exp_not:N \pgf@x
9095             =
9096             \dim_use:N \l__knot_tmpa_dim
9097             \exp_not:N \relax
9098             \exp_not:N \pgf@y
9099             =
9100             \dim_use:N \l__knot_tmpb_dim
9101             \exp_not:N \relax
9102           }
9103         }
9104       }
9105       \tl_set:Nn \l__knot_auxa_tl {\expandafter \gdef \csname knot~ intersections~}
9106       \tl_put_right:Nx \l__knot_auxa_tl {\tl_use:N \l__knot_name_tl - #1 - #2}
9107       \tl_put_right:Nn \l__knot_auxa_tl {\endcsname}
9108       \tl_put_right:Nx \l__knot_auxa_tl {\tl_to_str:N \l__knot_aux_tl}
9109       \protected@write\@auxout{}\tl_to_str:N \l__knot_auxa_tl}
9110     }
9111   }

```

```

9112 }
9113 \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.

```

9114 \cs_new_protected_nopar:Npn \knot_do_intersection:n #1
9115 {
9116   \knot_debug:n {knot~ do~ intersection~ #1}

```

Get the intersection coordinates.

```

9117   \pgfpointintersectionsolution{#1}
9118   \dim_set:Nn \l__knot_tmpa_dim {\pgf@x}
9119   \dim_set:Nn \l__knot_tmpb_dim {\pgf@y}
9120   \knot_debug:x {intersection~at~
9121     (\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.

```

9122   \bool_set_false:N \l__knot_skip_bool
9123   \bool_if:NT \l__knot_self_intersections_bool
9124   {

```

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

```

9125     \tl_set:Nn \l__knot_tmpc_tl {knot previous}
9126     \tl_put_right:NV \l__knot_tmpc_tl \l__knot_tmpa_tl
9127     \tl_set:Nv \l__knot_tmpc_tl \l__knot_tmpc_tl
9128     \tl_if_eq:NNT \l__knot_tmpc_tl \l__knot_tmpb_tl
9129     {
9130       \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpb_tl {final point}
9131       {
9132         \bool_set_true:N \l__knot_skip_bool
9133       }
9134     }
9135
9136     \tl_set:Nn \l__knot_tmpc_tl {knot previous}
9137     \tl_put_right:NV \l__knot_tmpc_tl \l__knot_tmpb_tl
9138     \tl_set:Nv \l__knot_tmpc_tl \l__knot_tmpc_tl
9139     \tl_if_eq:NNT \l__knot_tmpc_tl \l__knot_tmpa_tl
9140     {
9141       \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpa_tl {final point}
9142       {
9143         \bool_set_true:N \l__knot_skip_bool
9144       }
9145     }
9146   }

```

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

```

9147   \bool_if:NT \l__knot_ignore_ends_bool
9148   {
9149     \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpa_tl {initial point}
9150     {
9151       \bool_set_true:N \l__knot_skip_bool
9152     }
9153     \knot_test_endpoint:NVnT \l__knot_tolerance_dim \l__knot_tmpa_tl {final point}

```

```

9154     {
9155         \bool_set_true:N \l__knot_skip_bool
9156     }
9157     \knot_test_endpoint:N \l__knot_tolerance_dim \l__knot_tmpb_tl {initial point}
9158     {
9159         \bool_set_true:N \l__knot_skip_bool
9160     }
9161     \knot_test_endpoint:N \l__knot_tolerance_dim \l__knot_tmpb_tl {final point}
9162     {
9163         \bool_set_true:N \l__knot_skip_bool
9164     }
9165 }

```

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

```

9166 \bool_if:NF \l__knot_skip_bool
9167 {
9168
9169     \int_gincr:N \g__knot_intersections_int
9170     \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.

```

9171     \bool_if:nF
9172     {
9173         \tl_if_exist_p:c {l__knot_ignore_crossing_ \int_use:N
9174             \g__knot_intersections_int}
9175         &&
9176         ! \tl_if_empty_p:c {l__knot_ignore_crossing_ \int_use:N
9177             \g__knot_intersections_int}
9178     }
9179     {

```

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

```

9180     \bool_if:nTF
9181     {
9182         \tl_if_exist_p:c {l__knot_crossing_ \int_use:N
9183             \g__knot_intersections_int}
9184         &&
9185         ! \tl_if_empty_p:c {l__knot_crossing_ \int_use:N
9186             \g__knot_intersections_int}
9187     }
9188     {
9189         \tl_set_eq:NN \l__knot_tmpg_tl \l__knot_tmpb_tl
9190     }
9191     {
9192         \tl_set_eq:NN \l__knot_tmpg_tl \l__knot_tpa_tl
9193     }

```

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.

```

9194     \bool_if:NT \l__knot_self_intersections_bool
9195     {
9196         \knot_test_endpoint:N \l__knot_tmpg_tl

```

```

9197     \l__knot_redraw_tolerance_dim \l__knot_tmpg_tl {initial point}
9198     {
9199         \bool_set_true:N \l__knot_prepend_prev_bool
9200     }
9201     {
9202         \bool_set_false:N \l__knot_prepend_prev_bool
9203     }
9204     \knot_test_endpoint:NvNT
9205     \l__knot_redraw_tolerance_dim \l__knot_tmpg_tl {final point}
9206     {
9207         \bool_set_true:N \l__knot_append_next_bool
9208     }
9209     {
9210         \bool_set_false:N \l__knot_append_next_bool
9211     }

```

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

```

9212     \bool_if:nT
9213     {
9214         \l__knot_prepend_prev_bool || \l__knot_append_next_bool
9215     }
9216     {
9217         \tl_clear_new:c {knot \tl_use:N \l__knot_prefix_tl -1}
9218         \tl_set_eq:cc
9219         {knot \tl_use:N \l__knot_prefix_tl -1}
9220         {knot \tl_use:N \l__knot_tmpg_tl}
9221
9222         \tl_clear_new:c {l__knot_options_ \tl_use:N \l__knot_prefix_tl -1}
9223         \tl_set_eq:cc
9224         {l__knot_options_ \tl_use:N \l__knot_prefix_tl -1}
9225         {l__knot_options_ \tl_use:N \l__knot_tmpg_tl}
9226
9227         \bool_if:nT
9228         {
9229             \l__knot_prepend_prev_bool
9230             &&
9231             \tl_if_exist_p:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9232             &&
9233             !\tl_if_empty_p:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9234         }
9235         {
9236             \knot_debug:x {Prepending~
9237                 \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}}
9238             \spath_prepend_no_move:cv
9239             {knot \tl_use:N \l__knot_prefix_tl -1}
9240             {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.

```

9241     \bool_if:nT
9242     {
9243         \l__knot_splits_bool
9244         &&
9245         \tl_if_exist_p:c {knot previous
9246             \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}

```

```

9247     }
9248     &&
9249     !\tl_if_empty_p:c {knot previous
9250       \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9251   }
9252 }
9253 {
9254   \knot_test_endpoint:NvnT
9255   \l__knot_redraw_tolerance_dim
9256   {knot previous \tl_use:N \l__knot_tmpg_tl}
9257   {initial point}
9258   {
9259     \knot_debug:x {Prepending~
9260       \tl_use:c {knot previous
9261         \tl_use:c {knot previous \tl_use:N \l__knot_tmpg_tl}
9262       }
9263     }
9264     \spath_prepend_no_move:cv
9265     {knot \tl_use:N \l__knot_prefix_tl -1}
9266     {knot \tl_use:c
9267       {knot previous \tl_use:c
9268         {knot previous \tl_use:N \l__knot_tmpg_tl}
9269       }
9270     }
9271     \tl_set_eq:Nc \l__knot_tmpa_tl
9272     {knot \tl_use:N \l__knot_prefix_tl -1}
9273   }
9274 }
9275 }
9276

```

Now the same for appending.

```

9277   \bool_if:nT
9278   {
9279     \l__knot_append_next_bool
9280     &&
9281     \tl_if_exist_p:c {knot next \tl_use:N \l__knot_tmpg_tl}
9282     &&
9283     !\tl_if_empty_p:c {knot next \tl_use:N \l__knot_tmpg_tl}
9284   }
9285   {
9286     \knot_debug:x {Appending~
9287       \tl_use:c {knot next \tl_use:N \l__knot_tmpg_tl}}
9288     \spath_append_no_move:cv
9289     {knot \tl_use:N \l__knot_prefix_tl -1}
9290     {knot \tl_use:c {knot next \tl_use:N \l__knot_tmpg_tl}}
9291     \bool_if:nT
9292     {
9293       \l__knot_splits_bool
9294       &&
9295       \tl_if_exist_p:c {knot next \tl_use:c { knot next \tl_use:N
9296         \l__knot_tmpg_tl}}
9297       &&
9298       !\tl_if_empty_p:c {knot next
9299         \tl_use:c { knot next \tl_use:N \l__knot_tmpg_tl}

```

```

9300     }
9301   }
9302   {
9303     \knot_debug:x {Testing~ whether~ to~ append~
9304       {knot next \tl_use:c { knot next \tl_use:N \l__knot_tmpg_tl}}
9305     }
9306     \knot_test_endpoint:NvnT
9307     \l__knot_redraw_tolerance_dim
9308     {knot next \tl_use:N \l__knot_tmpg_tl}
9309     {final point}
9310     {
9311       \knot_debug:x {Appending~
9312         {knot next \tl_use:c { knot next \tl_use:N \l__knot_tmpg_tl}}
9313       }
9314       \spath_append_no_move:cv
9315       {knot \tl_use:N \l__knot_prefix_tl -1}
9316       {knot \tl_use:c
9317         {knot next \tl_use:c
9318           {knot next \tl_use:N \l__knot_tmpg_tl}
9319         }
9320       }
9321     }
9322   }
9323 }
9324 \tl_set:Nn \l__knot_tmpg_tl {\tl_use:N \l__knot_prefix_tl -1}
9325 }
9326 }

```

Now we render the crossing.

```

9327   \pgfscope
9328   \group_begin:
9329   \tikzset{
9330     knot~ diagram/every~ intersection/.try,
9331     every~ intersection/.try,
9332     knot~ diagram/intersection~ \int_use:N \g__knot_intersections_int/.try
9333   }
9334   \knot_draw_crossing:VVV \l__knot_tmpg_tl \l__knot_tmpa_dim \l__knot_tmpb_dim
9335   \coordinate
9336   (\l__knot_name_tl \c_space_tl \int_use:N \g__knot_intersections_int)
9337   at (\dim_use:N \l__knot_tmpa_dim, \dim_use:N \l__knot_tmpb_dim);
9338   \group_end:
9339   \endpgfscope

```

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

```

9340   }

```

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

```

9341   \tl_if_empty:NF \l__knot_node_tl
9342   {
9343     \seq_gpush:Nx
9344     \g__knot_nodes_seq
9345     {
9346       \l__knot_node_tl
9347       at
9348       (\dim_use:N \l__knot_tmpa_dim, \dim_use:N \l__knot_tmpb_dim) {};

```

```

9349     }
9350   }
9351 }
9352 }
9353
9354 \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.

```

9355 \prg_new_conditional:Npnn \knot_test_endpoint:NN #1#2 {p,T,F,TF}
9356 {
9357   \dim_compare:nTF
9358   {
9359     \dim_abs:n { \l__knot_tmpa_dim - \tl_item:Nn #2 {1}}
9360     +
9361     \dim_abs:n { \l__knot_tmpb_dim - \tl_item:Nn #2 {2}}
9362     <
9363     #1
9364   }
9365   {
9366     \prg_return_true:
9367   }
9368   {
9369     \prg_return_false:
9370   }
9371 }

```

(End of definition for \knot_test_endpoint:N.)

`\knot_test_endpoint:nn` Wrapper around the above.

```

9372 \prg_new_protected_conditional:Npnn \knot_test_endpoint:Nnn #1#2#3 {T,F,TF}
9373 {
9374   \use:c {spath_#3:Nv} \l__knot_tmpd_tl {knot #2}
9375   \knot_test_endpoint:NNTF #1 \l__knot_tmpd_tl
9376   {
9377     \prg_return_true:
9378   }
9379   {
9380     \prg_return_false:
9381   }
9382 }
9383
9384 \cs_generate_variant:Nn \knot_test_endpoint:NnnT {NVnT,NvnT}
9385 \cs_generate_variant:Nn \knot_test_endpoint:NnnF {NVnF,NvnF}
9386 \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.

```

9387 \cs_new_protected_nopar:Npn \knot_draw_crossing:nnn #1#2#3
9388 {
9389   \knot_debug:n {knot~ draw~ crossing}
9390   \group_begin:
9391   \pgfscope

```

```

9392 \path[knot~ diagram/background~ clip] (#2, #3)
9393 circle[radius=\l__knot_clip_bg_radius_dim];
9394
9395 \tl_set:Nn \l__knot_tmpa_tl {knot~ diagram/every~ strand/.try,}
9396 \tl_if_exist:cT {l__knot_options_ #1}
9397 {
9398 \tl_put_right:Nv \l__knot_tmpa_tl {l__knot_options_ #1}
9399 }
9400 \tl_put_right:Nn \l__knot_tmpa_tl
9401 {
9402 ,knotbg
9403 ,line~ width= \tl_use:N \l__knot_clip_width_tl * \pgflinewidth
9404 }
9405 \spath_tikz_path:Vv \l__knot_tmpa_tl {knot #1}
9406
9407 \endpgfscope
9408
9409 \pgfscope
9410 \path[knot~ diagram/clip] (#2, #3)
9411 circle[radius=\l__knot_clip_draw_radius_dim];
9412
9413 \tl_set:Nn \l__knot_tmpa_tl {knot~ diagram/every~ strand/.try,}
9414 \tl_if_exist:cT {l__knot_options_ #1}
9415 {
9416 \tl_put_right:Nv \l__knot_tmpa_tl {l__knot_options_ #1}
9417 }
9418 \tl_put_right:Nn \l__knot_tmpa_tl
9419 {
9420 ,knot~ diagram/only~ when~ rendering/.try
9421 ,only~ when~ rendering/.try
9422 }
9423 \spath_tikz_path:Vv \l__knot_tmpa_tl {knot #1}
9424
9425 \endpgfscope
9426 \group_end:
9427 }
9428
9429 \cs_generate_variant:Nn \knot_draw_crossing:nnn {nVV, VVV}
9430
9431 \cs_new_protected_nopar:Npn \knot_draw_crossing:nn #1#2
9432 {
9433 \tikz@scan@one@point\pgfutil@firstofone #2 \relax
9434 \knot_draw_crossing:nVV {#1} \pgf@x \pgf@y
9435 }

```

(End of definition for `\knot_draw_crossing:nnn`.)

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

```

9436 \cs_new_protected_nopar:Npn \knot_split_strands:
9437 {
9438 \knot_debug:n {knot~ split~ strands}
9439 \int_gzero:N \g__knot_filaments_int
9440 \int_step_function:nnnN {1} {1} {\l__knot_strands_int} \knot_split_strand:n
9441 \int_step_function:nnnN {1} {1} {\g__knot_filaments_int} \knot_compute_nexts:n
9442 }

```


(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.

```

9443 \cs_new_protected_nopar:Npn \knot_compute_nexts:n #1
9444 {
9445   \knot_debug:n {knot~ compute~ nexts}
9446   \tl_clear_new:c {knot next \tl_use:c {knot previous filament #1}}
9447   \tl_set:cn {knot next \tl_use:c {knot previous filament #1}} {filament #1}
9448 }

```

(End of definition for \knot_compute_nexts:n.)

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

```

9449 \cs_new_protected_nopar:Npn \knot_split_strand:n #1
9450 {
9451   \knot_debug:n {knot~ split~ strand}
9452   \int_set_eq:NN \l__knot_component_start_int \g__knot_filaments_int
9453   \int_incr:N \l__knot_component_start_int
9454   \tl_set_eq:Nc \l__knot_tmpa_tl {l__knot_options_strand #1}
9455   \spath_remove_empty_components:c {knot strand #1}
9456   \spath_segments_to_seq:Nv \l__knot_segments_seq {knot strand #1}
9457   \seq_map_function:NN \l__knot_segments_seq \knot_save_filament:N
9458 }

```

(End of definition for \knot_split_strand:n.)

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

```

9459 \cs_new_protected_nopar:Npn \knot_save_filament:N #1
9460 {
9461   \knot_debug:n {knot~ save~ filament}
9462   \tl_set:Nx \l__knot_tmpb_tl {\tl_item:nn {#1} {4}}
9463   \token_case_meaning:NnF \l__knot_tmpb_tl
9464   {
9465     \c_spath_moveto_tl
9466     {
9467       \int_compare:nT {\l__knot_component_start_int < \g__knot_filaments_int}
9468       {
9469         \int_set_eq:NN \l__knot_component_start_int \g__knot_filaments_int
9470       }
9471     }
9472     \c_spath_lineto_tl
9473     {
9474       \int_gincr:N \g__knot_filaments_int
9475       \tl_clear_new:c {knot filament \int_use:N \g__knot_filaments_int}
9476       \tl_set:cn {knot filament \int_use:N \g__knot_filaments_int} {#1}
9477
9478       \tl_clear_new:c {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9479       \tl_set_eq:cN {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9480       \l__knot_tmpa_tl
9481
9482       \tl_clear_new:c {knot previous filament \int_use:N \g__knot_filaments_int}
9483       \int_compare:nF {\l__knot_component_start_int == \g__knot_filaments_int}
9484       {

```

```

9485     \tl_set:cx {knot previous filament \int_use:N \g__knot_filaments_int}
9486     {filament \int_eval:n {\g__knot_filaments_int - 1}}
9487   }
9488 }
9489 \c_spath_curvetoa_tl
9490 {
9491   \int_gincr:N \g__knot_filaments_int
9492   \tl_clear_new:c {knot filament \int_use:N \g__knot_filaments_int}
9493   \tl_set:cn {knot filament \int_use:N \g__knot_filaments_int} {#1}
9494   \tl_clear_new:c {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9495   \tl_set_eq:cN {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9496   \l__knot_tmpa_tl
9497
9498   \tl_clear_new:c {knot previous filament \int_use:N \g__knot_filaments_int}
9499   \int_compare:nF {\l__knot_component_start_int == \g__knot_filaments_int}
9500   {
9501     \tl_set:cx
9502     {knot previous filament \int_use:N \g__knot_filaments_int}
9503     {filament \int_eval:n {\g__knot_filaments_int - 1}}
9504   }
9505 }
9506 \c_spath_closepath_tl
9507 {
9508   \int_gincr:N \g__knot_filaments_int
9509   \tl_clear_new:c {knot filament \int_use:N \g__knot_filaments_int}
9510   \tl_clear:N \l__knot_tmpa_tl
9511   \tl_put_right:Nx
9512   {
9513     \tl_item:nn {#1} {1}\tl_item:nn {#1} {2}\tl_item:nn {#1} {3}
9514   }
9515   \tl_put_right:NV \l__knot_tmpa_tl \c_spath_lineto_tl
9516   \tl_put_right:Nx {\tl_item:nn {#1} {5}\tl_item:nn {#1} {6}}
9517
9518   \tl_set:cV {knot filament \int_use:N \g__knot_filaments_int} \l__knot_tmpa_tl
9519   \tl_set_eq:cN {l__knot_options_filament \int_use:N \g__knot_filaments_int}
9520   \l__knot_tmpa_tl
9521   \tl_clear_new:c {knot previous filament \int_use:N \g__knot_filaments_int}
9522   \int_compare:nF {\l__knot_component_start_int == \g__knot_filaments_int}
9523   {
9524     \tl_set:cx
9525     {knot previous filament \int_use:N \g__knot_filaments_int}
9526     {filament \int_eval:n {\g__knot_filaments_int - 1}}
9527   }
9528   \tl_set:cx
9529   {knot previous filament \int_use:N \l__knot_component_start_int}
9530   {filament \int_use:N \g__knot_filaments_int}
9531 }
9532 }
9533 {
9534 }
9535 }

```

(End of definition for \knot_save_filament:N.)

```

\redraw The user can redraw segments of the strands at specific locations.
9536 \NewDocumentCommand \redraw { m m }
9537 {
9538 % \tikz@scan@one@point\pgfutil@firstofone #2 \relax
9539 \tl_put_right:Nn \l__knot_redraws_tl {\knot_draw_crossing:nn}
9540 \tl_put_right:Nx \l__knot_redraws_tl {
9541 {strand #1} {#2}% {\dim_use:N \pgf@x} {\dim_use:N \pgf@y}
9542 }
9543 }

```

(End of definition for \redraw.)

```
9544 \ExplSyntaxOff
```

<@@=>

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

```

9545 \def\pgf@sh__knotknotanchor#1#2{%
9546 \anchor{#2 north west}{%
9547 \csname pgf@anchor@knot #1@north west\endcsname%
9548 \pgf@x=#2\pgf@x%
9549 \pgf@y=#2\pgf@y%
9550 }%
9551 \anchor{#2 north east}{%
9552 \csname pgf@anchor@knot #1@north east\endcsname%
9553 \pgf@x=#2\pgf@x%
9554 \pgf@y=#2\pgf@y%
9555 }%
9556 \anchor{#2 south west}{%
9557 \csname pgf@anchor@knot #1@south west\endcsname%
9558 \pgf@x=#2\pgf@x%
9559 \pgf@y=#2\pgf@y%
9560 }%
9561 \anchor{#2 south east}{%
9562 \csname pgf@anchor@knot #1@south east\endcsname%
9563 \pgf@x=#2\pgf@x%
9564 \pgf@y=#2\pgf@y%
9565 }%
9566 \anchor{#2 north}{%
9567 \csname pgf@anchor@knot #1@north\endcsname%
9568 \pgf@x=#2\pgf@x%
9569 \pgf@y=#2\pgf@y%
9570 }%
9571 \anchor{#2 east}{%
9572 \csname pgf@anchor@knot #1@east\endcsname%
9573 \pgf@x=#2\pgf@x%
9574 \pgf@y=#2\pgf@y%
9575 }%
9576 \anchor{#2 west}{%
9577 \csname pgf@anchor@knot #1@west\endcsname%
9578 \pgf@x=#2\pgf@x%
9579 \pgf@y=#2\pgf@y%
9580 }%
9581 \anchor{#2 south}{%
9582 \csname pgf@anchor@knot #1@south\endcsname%

```

```

9583     \pgf@x=#2\pgf@x%
9584     \pgf@y=#2\pgf@y%
9585   }%
9586 }

```

(End of definition for \pgf@sh__knotknotanchor.)

knot_crossing

```

9587 \pgfdeclareshape{knot crossing}
9588 {
9589   \inheritsavedanchors[from=circle] % this is nearly a circle
9590   \inheritanchorborder[from=circle]
9591   \inheritanchor[from=circle]{north}
9592   \inheritanchor[from=circle]{north west}
9593   \inheritanchor[from=circle]{north east}
9594   \inheritanchor[from=circle]{center}
9595   \inheritanchor[from=circle]{west}
9596   \inheritanchor[from=circle]{east}
9597   \inheritanchor[from=circle]{mid}
9598   \inheritanchor[from=circle]{mid west}
9599   \inheritanchor[from=circle]{mid east}
9600   \inheritanchor[from=circle]{base}
9601   \inheritanchor[from=circle]{base west}
9602   \inheritanchor[from=circle]{base east}
9603   \inheritanchor[from=circle]{south}
9604   \inheritanchor[from=circle]{south west}
9605   \inheritanchor[from=circle]{south east}
9606   \inheritanchorborder[from=circle]
9607   \pgf@sh__knotknotanchor{crossing}{2}
9608   \pgf@sh__knotknotanchor{crossing}{3}
9609   \pgf@sh__knotknotanchor{crossing}{4}
9610   \pgf@sh__knotknotanchor{crossing}{8}
9611   \pgf@sh__knotknotanchor{crossing}{16}
9612   \pgf@sh__knotknotanchor{crossing}{32}
9613   \backgroundpath{
9614     \pgfutil@tempdima=\radius%
9615     \pgfmathsetlength{\pgf@xb}{\pgfkeysvalueof{/pgf/outer xsep}}%
9616     \pgfmathsetlength{\pgf@yb}{\pgfkeysvalueof{/pgf/outer ysep}}%
9617     \ifdim\pgf@xb<\pgf@yb%
9618       \advance\pgfutil@tempdima by-\pgf@yb%
9619     \else%
9620       \advance\pgfutil@tempdima by-\pgf@xb%
9621     \fi%
9622   }
9623 }

```

(End of definition for knot crossing.)

knot_over_cross

```

9624 \pgfdeclareshape{knot over cross}
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     \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9646     \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9647     \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@ya}}
9648     \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9649 }
9650 \foregroundpath{
9651 % store lower right in xa/ya and upper right in xb/yb
9652     \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9653     \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9654     \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9655     \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@ya}}
9656 }
9657 }

```

(End of definition for *knot over cross*.)

`knot_under_cross`

```

9658 \pgfdeclareshape{knot under cross}
9659 {
9660     \inheritssavedanchors[from=rectangle] % this is nearly a circle
9661     \inheritanchorborder[from=rectangle]
9662     \inheritanchor[from=rectangle]{north}
9663     \inheritanchor[from=rectangle]{north west}
9664     \inheritanchor[from=rectangle]{north east}
9665     \inheritanchor[from=rectangle]{center}
9666     \inheritanchor[from=rectangle]{west}
9667     \inheritanchor[from=rectangle]{east}
9668     \inheritanchor[from=rectangle]{mid}
9669     \inheritanchor[from=rectangle]{mid west}
9670     \inheritanchor[from=rectangle]{mid east}
9671     \inheritanchor[from=rectangle]{base}
9672     \inheritanchor[from=rectangle]{base west}
9673     \inheritanchor[from=rectangle]{base east}
9674     \inheritanchor[from=rectangle]{south}
9675     \inheritanchor[from=rectangle]{south west}
9676     \inheritanchor[from=rectangle]{south east}
9677     \inheritanchorborder[from=rectangle]
9678     \backgroundpath{
9679         \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y

```

```

9680 \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9681 \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9682 \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9683 }
9684 \foregroundpath{
9685 % store lower right in xa/ya and upper right in xb/yb
9686 \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9687 \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9688 \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9689 \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9690 }
9691 }

```

(End of definition for *knot under cross*.)

knot_vert

```

9692 \pgfdeclareshape{knot vert}
9693 {
9694 \inheritsavedanchors[from=rectangle] % this is nearly a circle
9695 \inheritanchorborder[from=rectangle]
9696 \inheritanchor[from=rectangle]{north}
9697 \inheritanchor[from=rectangle]{north west}
9698 \inheritanchor[from=rectangle]{north east}
9699 \inheritanchor[from=rectangle]{center}
9700 \inheritanchor[from=rectangle]{west}
9701 \inheritanchor[from=rectangle]{east}
9702 \inheritanchor[from=rectangle]{mid}
9703 \inheritanchor[from=rectangle]{mid west}
9704 \inheritanchor[from=rectangle]{mid east}
9705 \inheritanchor[from=rectangle]{base}
9706 \inheritanchor[from=rectangle]{base west}
9707 \inheritanchor[from=rectangle]{base east}
9708 \inheritanchor[from=rectangle]{south}
9709 \inheritanchor[from=rectangle]{south west}
9710 \inheritanchor[from=rectangle]{south east}
9711 \inheritanchorborder[from=rectangle]
9712 \backgroundpath{
9713 % store lower right in xa/ya and upper right in xb/yb
9714 \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9715 \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9716 \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9717 \pgfpathlineto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9718 \pgfpathmoveto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9719 \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9720 }
9721 }

```

(End of definition for *knot vert*.)

knot_horiz

```

9722 \pgfdeclareshape{knot horiz}
9723 {
9724 \inheritsavedanchors[from=rectangle] % this is nearly a circle
9725 \inheritanchorborder[from=rectangle]
9726 \inheritanchor[from=rectangle]{north}

```

```

9727 \inheritanchor[from=rectangle]{north west}
9728 \inheritanchor[from=rectangle]{north east}
9729 \inheritanchor[from=rectangle]{center}
9730 \inheritanchor[from=rectangle]{west}
9731 \inheritanchor[from=rectangle]{east}
9732 \inheritanchor[from=rectangle]{mid}
9733 \inheritanchor[from=rectangle]{mid west}
9734 \inheritanchor[from=rectangle]{mid east}
9735 \inheritanchor[from=rectangle]{base}
9736 \inheritanchor[from=rectangle]{base west}
9737 \inheritanchor[from=rectangle]{base east}
9738 \inheritanchor[from=rectangle]{south}
9739 \inheritanchor[from=rectangle]{south west}
9740 \inheritanchor[from=rectangle]{south east}
9741 \inheritanchorborder[from=rectangle]
9742 \foregroundpath{
9743 % store lower right in xa/ya and upper right in xb/yb
9744   \southwest \pgf@xa=\pgf@x \pgf@ya=\pgf@y
9745   \northeast \pgf@xb=\pgf@x \pgf@yb=\pgf@y
9746   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@ya}}
9747   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@ya}}
9748   \pgfpathmoveto{\pgfqpoint{\pgf@xa}{\pgf@yb}}
9749   \pgfpathlineto{\pgfqpoint{\pgf@xb}{\pgf@yb}}
9750 }
9751 }

```

(End of definition for *knot horiz.*)