Appendix B

HIPAIR Software

In this appendix, we describe the HIPAIR mechanism design program built around configuration space computation, visualization, and manipulation. We describe installation, the GUI, and the input format. The program is written in Ansi C++ using openGL and glut. It is free for education and research. Send comments, bug reports, and request for commercial licenses to Elisha Sacks (eps@cs.purdue.edu).

B.1 Installation

The website is `http://www.cs.purdue.edu/archives/2008/eps/`. The program runs under Linux and Windows. The archive `hipair.tar` contains the source files, which are the same for both operating systems, and the `ex` subdirectory, which contains input files for most of the mechanisms in Appendix A. The file `hipair.exe` is a windows executable.

Here is how to install and run the Linux version.

```
% tar -xvf hipair.tar
% make
% hipair ex/3finger
```

If the argument is omitted, HIPAIR prompts for it and reads it from standard input. In Windows, create a project with the source file suffix changed to .cpp and with glut installed.

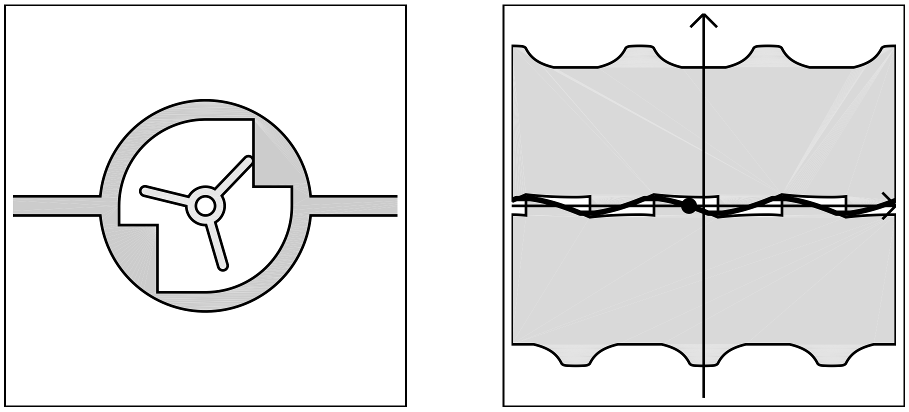

Figure B.1: Parts window and cspace window for ex/3finger.

B.2 Graphical User Interface (GUI)

The program opens a parts windows and one cspace (configuration space) window per pair (Figure B.1). The windows can be resized. The parts window shows the parts in the current configuration. Each cspace window shows its configuration space with the animation path in red, the current configuration as a green dot, and the contact zone in green when requested. The program normally display the current animation step. The user can increment or decrement the step, or can animate the mechanism. The animation starts at the current step and ends at the last step or when interrupted. The animation speed is specified as a multiple of $1/30$ of a second. The user can specify an arbitrary configuration with the mouse. The next animation request resets the configuration to the current animation step. The user can set the contact zone width and can toggle contact zone display.

The program is controlled by the mouse. Press, drag, and release the left button to zoom to a rectangle. Click the middle button in a cspace window to set its two configuration parameters to the mouse position. Click the right button to bring up this menu. The menu entries can also be invoked by typing the letters in parentheses.

- Start/stop animation (a).
- Next animation step (n).
- Previous animation step (p).
- Set animation speed (s).

- Draw parts in wireframe/filled (f).
- Restore initial viewing box (r).
- Show/hide contact zones (z).
- Set contact zone width (w).
- Save window in postscript (o).
- Exit (q).

B.3 Mechanisms

A mechanism consists of planar parts that form higher kinematic pairs. HIPAIR computes configuration spaces for the pairs, computes the part motions for given driving motions, and computes contact zones that bound the kinematic variation for the given part tolerances. This section explains these concepts.

B.3.1 Parts

A part is specified in a part coordinate system. The part consists of regions in the xy plane that are extruded over intervals on the z axis. A region has an outer boundary and may also have inner boundaries. A boundary is a simple loop of line and circle segments.

For example, a cylinder of height 10mm is specified as a circle in the xy plane that is extruded over $0 \leq z \leq 10$. The region boundaries of a slice must be disjoint, but slices may overlap. A part has an initial position and orientation in world xy coordinates. It has one degree of freedom: rotation around its origin, which is fixed at its initial position, or translation along a line that passes through its initial position.

Figure B.2a shows a simple mechanism comprised of two parts. Each part consists of a single slice with z range $[0, 1]$. The first part is a unit square. Its origin is the bottom, left corner. Its boundary consists of four line segments. The part rotates with initial position $(1, 1)$ and orientation 0. The second part is a circle of radius 1 whose origin is the circle center. The part translates horizontally with initial position $(3, 1)$ and orientation 0.

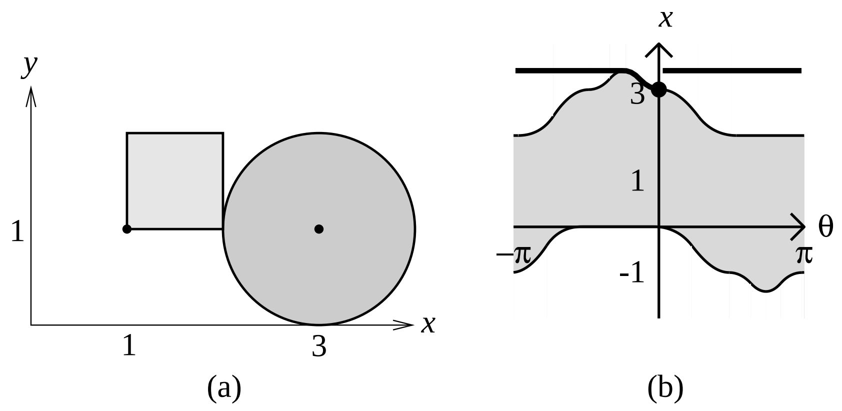

Figure B.2: Square/circle pair (a) and its configuration space partition (b).

B.3.2 Configuration spaces of fixed-axis pairs

Interactions between pairs of parts are modeled as configuration spaces. The configuration space is a two-dimensional manifold whose coordinates are the part degrees of freedom. Figure B.2b shows the configuration space of the square/circle pair: it is a cylinder whose coordinates are the square orientation θ and the rectangle offset x. The configuration space is a torus when both parts rotate and is a rectangle when both translate. Configuration space partitions into free space where the parts do not touch (the white region in the figure), blocked space where they overlap (the grey region), and contact space where they touch (the black curves). The green dot marks the displayed configuration of $(0, 3)$.

B.3.3 Kinematic simulation

Kinematic simulation takes a driving part motion as input and computes motions for the other parts that prevent part overlap. In our example, the driving motion is clockwise rotation of the square by one full turn. The circle translates clear of the square then stands still. The motion path is the thick black curve in Figure B.2b.

B.3.4 Contact zones of fixed-axis pairs

Variation in the part shapes is modeled as a zone around the nominal part boundaries where the actual boundaries are allowed to lie. HIPAIR supports constant-width offsets. As the boundaries of the parts in a kinematic pair range over their

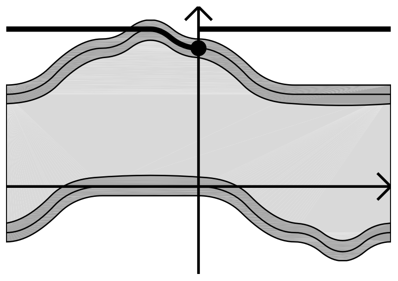

Figure B.3: Square/circle contact zone.

zones, their contact spaces vary in a zone around the nominal space, called the contact zone of the pair. The contact zone topology and geometry model the kinematic variation due to the part variation. Figure B.3 shows the square/circle contact zone for offsets of 0.1 on both parts. It forms a narrow band around the nominal contact space, which shows that the part variation merely perturbs the nominal kinematic function.

B.4 Input format

The input is a mechanism, simulation data, and display data.

B.4.1 Mechanisms

The mechanism format is specified by the following BNF formulas. The first line specifies the number of parts and pairs, then come the parts, then the pairs. The parts are assigned indices starting from zero. A pair is specified by two part indices and a configuration space computation accuracy.

MECHANISM	=	1 nparts npairs 0 0 PART* PAIR*
PART	=	name nslices color parameters SLICE$^+$
SLICE	=	nboundaries zlower zupper BOUNDARY$^+$
BOUNDARY	=	ngroups SGROUP$^+$
SGROUP	=	nsegments SEGMENT$^+$ noffsets offset*
SEGMENT	=	LINE_SEGMENT or ARC_SEGMENT
LINE_SEGMENT	=	index 0 tx ty hx hy flag
ARC_SEGMENT	=	index 1 tx ty hx hy flag cx cy r s e
PAIR	=	index1 index2 accuracy

B.4.2 Parts

A part is specified by a name, number of slices, color, parameters, and slices. The name is a character string. The color is three integers in the range $[0, 255]$ followed by a dummy integer. The parameters are seven doubles. Parameter 1 is the motion type: 1 for translation along the x axis, 2 for translation along the y axis, 3 for rotation, and 4 for translation along a slanted axis. Parameters 2–4 are the initial position and orientation of the part. Parameters 5–6 are the lower and upper bounds of the part motion parameter. For motion type 4, parameter 7 is the translation axis slope. The motion parameter and its bounds are measured along this axis.

A slice is specified by the number of boundaries, the lower and upper z values, the outer boundary, and the inner boundaries. A boundary is specified by the number of segment groups followed by the groups. A group is a sequence of incident segments and a sequence of offsets. A segment is specified by a dummy index, 0 or 1 for a line or circle, a tail (t_x, t_y), a head (h_x, h_y), and a flag that is 1 when the part interior lies to the left when the segment is traversed from tail to head. A circle segment also has a center (c_x, c_y), radius r, start angle s, and end angle e.

The segment in the group are repeated *noffset* times and the ith copy is offset by the ith element of the offset sequence. The offset is along the part motion axis: the segments of a rotating part are rotated and the segments of a translating part are translated. Offsets are optional, since any boundary can be specified as one group with noffset 1 and offset 0.

Figure B.4 illustrates the file format on the square/circle mechanism from Figure B.2. The first line indicates two parts and one pair. Lines 3–11 describe the first part. Line 3: name "square", one slice, and RGB color $(255, 0, 0)$. Line 4: motion type 3 (rotation), initial position $(1.0, 1.0)$, initial orientation 0.0 radians, and rotation range $[-\pi, \pi]$ radians. Line 5: the slice has one boundary and has z extent $[0.0, 1.0]$. Line 6: the boundary consists of one sgroup with four segments. Lines 7-10: four line segments. Line 11: trivial shift sequence, explained next. Lines 13–20 specify the circle: name "circle", one slice, RGB color $(0, 255, 0)$, motion type 1 (x translation), initial position $(1.0, 4.0)$, initial orientation 0.0, translation range $[-5, 5]$, the slice has one boundary with two segments, two circle segments, and the shift sequence. The last line specifies the one pair: part index 0 (the square), part index 1 (the circle), and configuration space accuracy 0.001.

Figure B.5 shows a Geneva mechanism. The configuration space is a torus be-

```
1 1 2 1 0 0

square 1 255 0 0 0
3.0 1.0 1.0 0.0 -3.14159 3.14159 0.0
1 0.0 1.0
1 4
0 0 0.0 0.0 1.0 0.0 1
0 0 1.0 0.0 1.0 1.0 1
0 0 1.0 1.0 0.0 1.0 1
0 0 0.0 1.0 0.0 0.0 1
1 0.0

circle 1 0 255 0 0
1.0 4.0 1.0 0.0 -5.0 5.0 0.0
1 0.0 1.0
1 2
0 1 -1.0 0.0 1.0 0.0 1 0.0 0.0 1.0 -3.14159 0.0
0 1 1.0 0.0 -1.0 0.0 1 0.0 0.0 1.0  0.0 3.14159
1 0.0

0 1 0.001
```

Figure B.4: Square/circle file format.

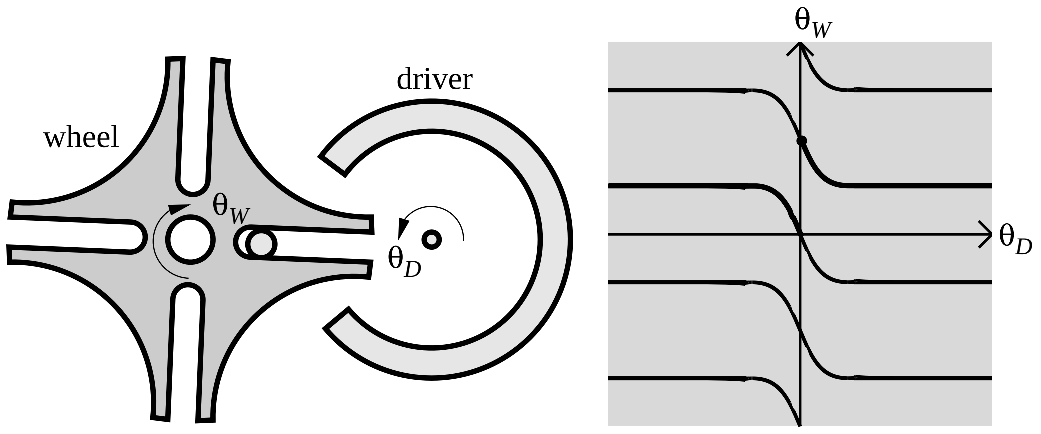

Figure B.5: Geneva pair and its configuration space.

cause both parts rotate. The driver has three slices with z interval $[0, 1]$: a locking arc, a driving pin, and a central pin. The three slices are connected by a plate (not shown) with z interval $[-1, 0]$. The wheel has one slice. The outer boundary consists of a six-segment sgroup with four offsets. The inner boundary is one circle. Figure B.6 shows the follower specification. The offsets, $0, \pi/2, \pi, 3\pi/2$ radians, generate four rotated copies of the six segments.

B.4.3 Simulation and display data

The simulation data is specified by the following BNF formulas.

MOTIONS	=	nm MOT_DESCR* CONFIG
MOT_DESCR	=	mtime nm_parts PART_MOT_DESCR*
PART_MOT_DESCR	=	pindex velocity
CONFIG	=	one double per part

The number of motions is followed by one descriptor per motion. A motion descriptor is the motion time, the number of part motions, then the part motion descriptors. A part motion descriptor is a part index followed by a velocity. The motion begins with the part motion parameters specified in CONFIGURATION. In the square/circle example, the motion sequence of rotating the square with angular velocity -1.0 for 7.0 seconds then translating the circle with x velocity 3.0 for 5.0 seconds has the following file descriptor. The starting configuration is $\theta = 0$ and $x = 3$.

```
follower 1 0 255 0 0
3.0 0.0 0.0 0.0 -3.14159 3.14159 0.0
2 0.0 1.0
1 6
0 0 6.0 -0.5 2.0 -0.5 1
0 1 2.0 0.5 2.0 -0.5 0 2.0 0.0 0.5 1.5708 -1.5708
0 0 2.0 0.5 6.0 0.5 1
0 0 6.0 0.5 5.9585 0.9983 1
0 1 0.9983 5.9585 5.9585 0.9983 0 5.6568
  5.6568 4.6683 3.07693 -1.5061
0 0 0.9982 5.9585 0.5 6.0 1
4 0.0 1.5708 3.14153 4.712389
1 1
0 1 -0.75 0.0 -0.75 0.0 0 0.0 0.0 0.75
  -3.14159 3.14159
1 0.0
```

Figure B.6: Follower file format.

```
2
7.0 1 0 -1.0
5.0 1 1 3.0
0 3
```

The display data is two dummy values followed by a part bounding box. It is

```
0 0 -5 5 -5 5
```

in our example.