2018

Concerning the Construction of Four-bar Linkages and Their Topological Configuration-spaces

Peter K. Servatius
Bard College

CONCERNING THE CONSTRUCTION OF FOUR-BAR LINKAGES AND THEIR TOPOLOGICAL CONFIGURATION-SPACES.

A Senior Project submitted to
The Division of Science, Mathematics, and Computing
of
Bard College

by
Peter Kareem Elias Rizk Servatius

Annandale-on-Hudson, New York
October, 2017

Abstract

For a given linkage with one degree of freedom we can analyze the coupler curve created by any selected tracer point in relation to a driver link. The Watt Engine is a four-bar linkage constructed such that the tracer point draws an approximate straight line along a section of the coupler curve. We will explore the family of linkages that are created using Watt's parameters, along with linkages designed by other inventors; looking at methodologies of creating a linkage and the defining what we mean by approximate straight-line motion. Ultimately we will be creating our own linkage using what we have learned from previous inventors, and our new found classifications and definitions for linkage construction.

Contents

Dedication

I dedicate this senior project to my family: as the youngest of the throng, each member has done their fair share of helping me through my academic career. So this goes out to Herman, Brigitte, Rainer, Irma, Mary, Cecilia, Edward, Labeeby, Chutimon, Brian, Martin, and the rest.

Acknowledgments

I would like to acknowledge the help I received from many great individuals throughout my life and time at Bard. Most notably are: the other extra, my parents, and the best siblings a little brother could ask for. I can't help but mention some top-notch advisers: Matthew Deady and Amir Barghi both providing wonderful guidance and plenty of inspiration. I must thank Professor Kerri-Ann Norton for the impeccable coding expertise and advice that got my garbled code working in a heartbeat. Surviving Bard would not have been possible without the long list of friends who kept me sane, in no particular order: Avery, Hunter, Hazaiah, Simon, Emma, Hannah, Emma, Woanjun, Noah, Caleb, Kyle, Josie, Anna, Kate, Shane, Nora, Noah, Cecilia, Rachael, Drew, and a host of others; I thank you all. Also of note, I would like to thank whoever discovered chocolate as the copious consumption of it has helped with every aspect of this project, and it has helped develop who I am as a person.

1
Introduction

A linkage in n-space consists of rigid bodies (the links) hinged together along $n-1$-dimensional hinges. So in the plane the hinges are points and we may consider the bodies to be line segments. In 3-space the hinges are lines (think of the hinges of a door), and the links are planar polygons.

Kempe's universality theorem [8] states that any finite length segment of an algebraic curve can be drawn using a (planar) linkage. However, since he simply states that some linkage exists for any given curve, Kempe's findings don't particularly clear up the matter of how best to construct the linkage in question, nor how variations of a single linkage are accounted for.

We know that the definition of a Grashof four-bar linkage is at least one bar revolves fully about one of its endpoints, this is achieved as long as the linkage satisfies the requirement that the sum of the lengths of the shortest and longest bars of the linkage is less than or equal to the sum of the lengths of the remaining two bars [11].

We know that the equation for the coupler curve of Watt's engine is:[6], see also [7]

$$r^2 = b^2 - (a\sin(\theta) \pm \sqrt{c^2 - a^2\cos^2(\theta)})$$

which is not an equation of a straight line.

However, we know that a Peaucellier cell creates a true straight line segment as a coupler curve due to the specific geometric design of the linkage. In the book *How Round is Your Circle*, Bryant and Sagwin describe how the Peaucellier cell's geometric construction allows the actuated bar to force the tracer point of the linkage to draw a straight line.[1] They do not give a formula for the length of the straight line segment drawn by the mechanism.

2

History and Basics

2.1 Some interesting linkages

2.1.1 The Watt engine

The construction of a basic Watt engine, as shown in Figure 2.1.1, is defined as follows. The link AD is held fixed, typically not drawn, the two links AB and CD are then forced to described contrary arcs staggered by the length of link BC. The tracer point, the point which draws the graph this linkage was designed for, is in the center of link BC. According to Watt the coupler curve of this center point on link BC would be forced to describe an approximate straight line. [12] The straight line motion of this linkage is only a section of the complete coupler curve, the complete curve resembling a figure-8.

Something that interests me is whether or not we can improve the Watt engine by adding a polygon onto link BC, or simply moving the tracer point elsewhere on BC. If it is indeed possible to improve the straight line motion, can we create a general formula for doing so? Is there a way to analyze the equation for the motion of the Watt engine to figure out whether a finer approximation for straight line motion is possible rather than using the standard construction.

We can prove that the Watt engine used in the definition above is a non-Grashof four-bar linkage. The term Grashof implies that the shortest and longest bars of the linkage, S and L respectively, are less than or equal to the remaining two bars, in this case, $2X$.[11] Since the

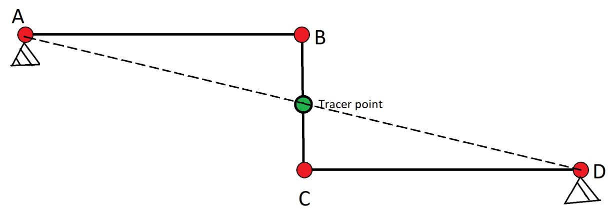

Figure 2.1.1. Watt engine definition

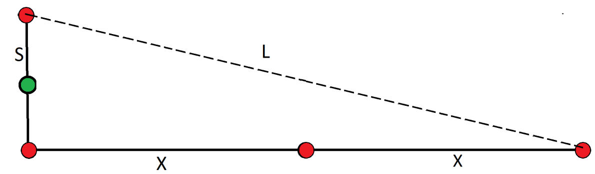

Figure 2.1.2. Watt engine definition

definition of whether a linkage is Grashof does not depend on the order in which the links are connected, we can redraw the Watt engine with an isomorphic map in Figure 2.1.2.

Using the property of triangles, we know that the hypotenuse of a right triangle is greater than either of the other two sides, so we know that $L > 2X$ which implies that $S + L > 2X$ which is a violation of the term Grashof, which would have required that $S + L \leq 2X$.

The only way a Watt engine linkage could be considered Grashof, is when the tracer link S and the fixed link L are drawn to have equal length. At this point it is clear that $L + S = 2X$ so the construction would be Grashof, but since the Watt engine was designed to produce a particular curve given a particular set of links, specific constructions such as this are not relevant for our interests.

2.1.2 The Chebyshev Linkage

Chebyshev had a somewhat different approach when attempting to attain approximate straight line motion. He differs from Watt in the sense that his linkage, the Chebyshev linkage and its cognate, the Lambda Linkage, have precisely proportioned lengths for each of the links in the respective linkages. Chebyshev moves away from Watt's rough sketch of a linkage to a more carefully thought out construction, ultimately creating a linkage whose coupler curve draws a

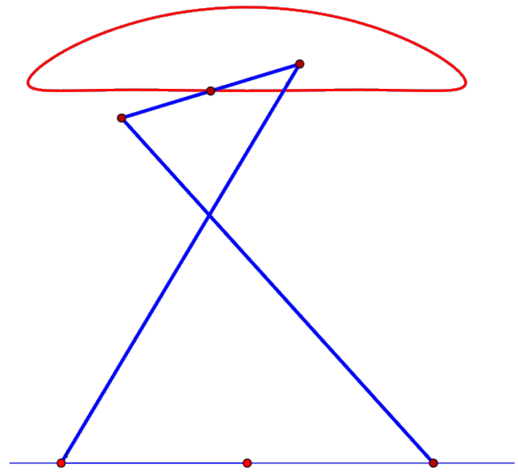

Figure 2.1.3. A Charming Chebyshev Linkage

more accurate straight line approximation than Watt.[9] The Chebyshev linkage is quite simple in construction, a fixed bar four units in length, is neighbored on both end points by links of 5 units in length, these last two joined by a short bar of length 2 units[2].

The lambda linkage, named for its resemblance to the Greek letter λ, is a cognate of the Chebyshev linkage(see section 2.3). The pure four-bar linkage section of the lambda linkage, has the dimensions of half that of the Chebyshev linkage, but due to its unique extended link, the Lambda linkage, draws the same coupler curve as its older brother. The Lambda linkage is famously used in Chebyshev's Plantigrade machine, where one lambda linkage is used for each of the machine's four legs.

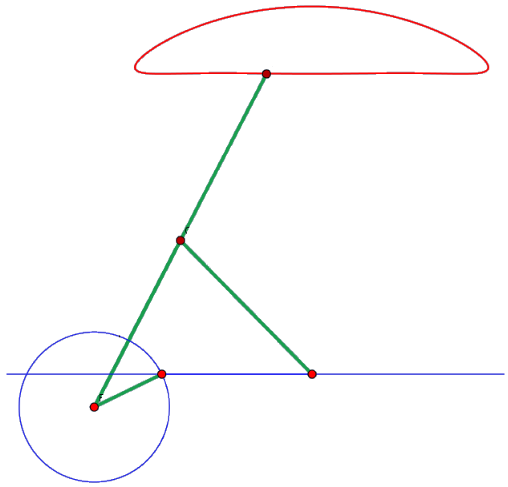

Figure 2.1.4. A Lovely Lambda Linkage

2.2 Parametrization of a four-bar linkage with one fixed bar

We first compute the intersection points of two circles, one with center (a, b) and radius R_1 and one with center $(0, 0)$ and radius R_2.

Equation for circle 1: $(x - a)^2 + (y - b)^2 = R_1^2$

Equation for circle 2: $x^2 + y^2 = R_2^2$.

Subtracting the two equations gives a linear relationship between x and y:

$$-2ax + a^2 - 2yb + b^2 = R_1^2 - R_2^2$$

or,

$$\frac{1}{2}(a^2 + b^2 + R_2^2 - R_1^2) = (a, b) \cdot (x, y)$$

To get a unit vector in the (a, b) direction we multiply by a factor of $\frac{1}{\sqrt{(a^2+b^2)}}$

$$\frac{(a^2 + b^2 + R_2^2 - R_1^2)}{2\sqrt{a^2 + b^2}} = (\frac{a}{\sqrt{a^2 + b^2}}, \frac{b}{\sqrt{a^2 + b^2}}) \cdot (x, y)$$

To make our lives easier, we will now designate variables to take the place of complicated sections of this computation.

$$
\begin{aligned}
S &= \frac{(a^2+b^2+R_2^2-R_1^2)}{2\sqrt{a^2+b^2}} \\
S_x &= \left(\frac{(a^2+b^2+R_2^2-R_1^2)}{2(a^2+b^2)}\right)\cdot a \\
S_y &= \left(\frac{(a^2+b^2+R_2^2-R_1^2)}{2(a^2+b^2)}\right)\cdot b \\
T &= \sqrt{R_2^2-S^2}
\end{aligned}
$$

This makes solving for the definitions for x and y simpler to stomach:

$$x = S_x \pm \left(\frac{b}{\sqrt{a^2+b^2}}\right)T$$

$$y = S_y \pm \left(\frac{a}{\sqrt{a^2+b^2}}\right)T$$

This model gets interesting once we make a and b functions of an angle θ. In order to complete our Watt Engine, we need to add another link to this linage, with some length R_3, and we will position its fixed point at some point (m, n). Now to give this whole computation a purpose we define $a = m - l\cos(\theta)$ and define $b = n + l\sin(\theta)$.

Now we can try this out for particular values. For a start, we can take $R_3 = R_1 = R_2 = 1$ and $m = 2$, $n = 1$, that makes for a nice Watt engine and Mathematica will happily draw the figure 8 after the correct limits for θ are chosen.

If we are looking at the Watt engine in its theoretical sense, we would not limit its movement which means we would be considering the entire coupler curve, hence we let θ range from 0 to 2π. If we want to restrict the motion to just the "straight line part" we need to be a bit more careful. The question then becomes: when does approximate straight line motion cease to be approximate, or in other words, at what point can we consider a curve to be approximately straight?

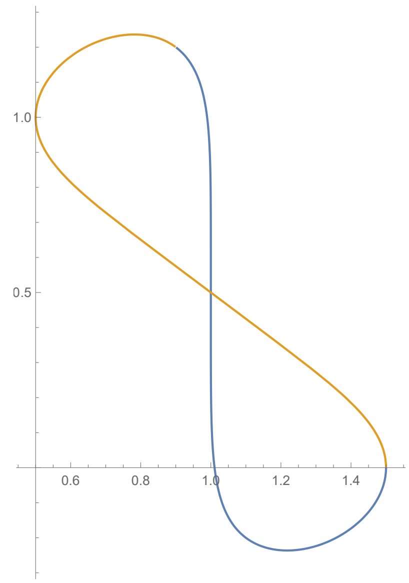

Figure 2.2.1. Coupler Curve computed by Mathematica

2.3 Cognates of linkages

2.3.1 A useful lemma

Lemma 2.3.1. *Given any two neighboring links of different lengths, we can construct a unique point X such that when connected with two additional links to the original two neighboring links, the construction forms a parallelogram.*

Proof. Suppose we have two neighboring links AB and BC, and suppose further that these two links are at different angles with respect to the origin. We can construct a straight line passing through point A which is parallel to link BC. Similarly, there is a straight line passing through point C which is parallel with link AB. Since the links AB and BC are at different angles, these two lines must intersect at a point X. We can now connect point X to the system by constructing the links AX, and XC. We now have that AX is parallel to BC and that XC is parallel to AB. From this we know that AX, AB, BC, XC is a parallelogram by definition.

At this point the proof is not complete, as the above is merely dealing with snapshots. Yes we can of course construct a parallelogram using two neighboring links, but we must still show that the motion is continuous.

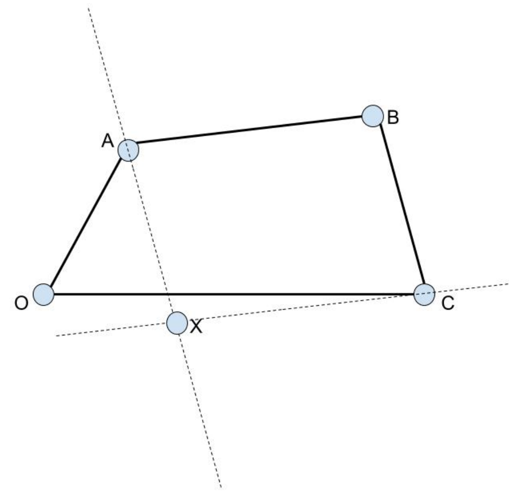

Figure 2.3.1. Linkage $OABC$, with unique point X

Let $\boldsymbol{AB}$ and $\boldsymbol{BC}$ be neighboring links in vector form, and let D be the position vector $D = A + C - B$. Then $D - A = C - B$, hence $\boldsymbol{DA}$ is parallel to$\boldsymbol{CB}$ and $|D - A| = |C - B|$. In the same way we find that $D - C = A - B$, hence $\boldsymbol{DC}$ is parallel to $\boldsymbol{AB}$ and $|D - C| = |B - A|$. Thus we can conclude that $\boldsymbol{AB}$,$\boldsymbol{BC}$,$\boldsymbol{CD}$,$\boldsymbol{DA}$ forms a parallelogram. Moreover, D is continuous for any motion of the linkage. We can use this unique point D to show that the motion of X is continuous. $\square$

2.3.2 Compass construction of a parallelogram

Given two sides of a parallelogram AB and BC, we can constructed a fourth point X by intersecting a circle of radius $|BC|$ about A and a circle or radius AB about point C. The two circles now intersect at two points, and these two points are on different sides on the line connecting the centers of the circles. We now select X to be the intersection point on the opposite side of the line through AC with respect to B. This ensures that $ABCX$ is a parallelogram.

There are three extreme cases, two where A, B, and C lie on the same line. In these two cases, where B is either outside AC, or inside, the circles constructed on A and C intersect at only one point, where X must be placed which lies on the same line as A, B, and C.

The third case is where A and C are at the same location on the plane, but are not connected to the same links, this forces B to be at a position where $AB = BC$. Now if we attempt to find X

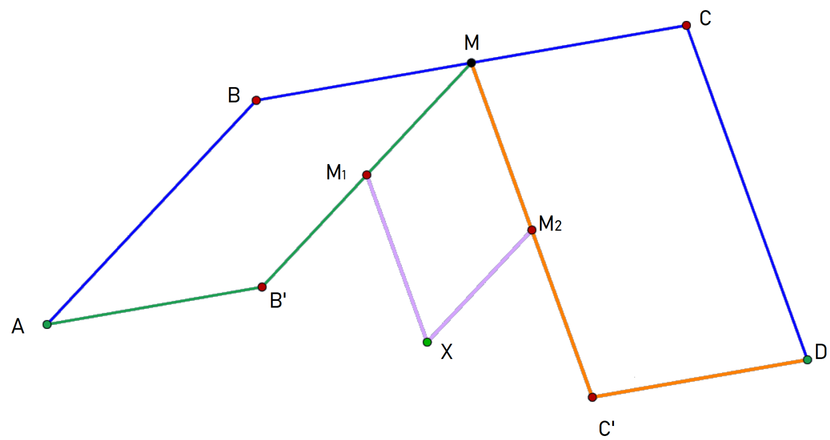

Figure 2.3.2. Cognates on a Linkage ABCD

by intersecting the two circles as normal, we find that they have the same radius, and therefore we have infinitely many solutions. This situation is the result of $ABCX$ being a rhombus. This is a highly chaotic linkage.

2.3.3 Cognates

Let us start with a simple mechanism with fixed points A and D, moving bars AB, BC, and CD. M denotes the midpoint of bar BC. The motion of this mechanism is not altered in any way by adding bars AB' of length BM, MB' of length AB, because adding bars of these lengths creates a rhombus $ABMB'$. Adding a rhombus changes nothing about the motion of the original linkage as it is adding a Grashof linkage which contains no cusp points, since the rhombus can fold in on itself and unfold completely without hindering the original linkage. We can add yet another rhombus by adding the link MC' of length CD and $C'D$ of length MC. Take a look at **Figure 2.4.1** for an visualization of the construction.

Let M_1 be the midpoint of MB' and M_2 be the midpoint of MC' and X be the fourth point of the parallelogram MM_1M_2X. Then to get from A to X we can walk from A to B_1 to M_1 and then to X: $AX = AB_1 + B_1M_1 + M_1X = BM + 1/2AB + 1/2CD = 1/2BC + 1/2AB + 1/2CD = 1/2(AB + BC + CD) = 1/2AD$. So X is the midpoint AB, no matter where points B and C are as the mechanism moves.

The mechanisms $AB'M_1X$ and $XM_2C'D$ are called cognates of $ABCD$. Putting the tracer point M on the extended middle bar of the cognates, we get coupler curves identical to the original curve. The new mechanism is smaller, the moving links are half the size of the original ones but they are connected in a different order with respect to their length.

We can use our parametrization to get equations for the cognates by simply changing $(a+x)/2$ and $(b+y)/2$ to $2x-a$ and $2y-b$.

3
General mechanism construction

3.1 Introduction to configuration spaces and topology

Jordan and Steiner define the configuration space of a mechanical linkage in $\mathbb{R}^2$ as the totality of all its admissible positions in the Euclidean plane. Some links may be pinned down with respect to the given coordinate system [5]. They state that it is well known that the configuration space of a linkage is a compact algebraic variety naturally embedded in $\mathbb{R}^{2n}$, where n is the number of vertices of the graph underlying the linkage and then prove the converse constructively. See also [10].

Sossinsky further studies configuration spaces of planar mechanical linkages with one degree of freedom in [14] and he points out that a purely topological approach is not enough but a finer geometric approach is needed. A recent survey of configuration spaces of planar linkages is found in [15].

We try to explain topological and geometric properties of the configuration space of a four-bar linkage.

3.2 Topology

3.2.1 Configuration space

A four-bar linkage consists of four vertices A, B, C, and D and four length constraints for the bars of the linkage, AB, BC, CD, and DA. For our purposes we will pin vertex A at the origin and pin vertex B some distance from A on the x axis. Now we construct C and D somewhere within the xy plane provided they adhear to the length constrants of a given linkage.

By writing the coordinates of the vertices as a row vector, we can view a four-bar linkage as a vertex in 8-space, $\mathbb{R}^8$. We want to consider all the points in $\mathbb{R}^8$ from a particular given linkage as the configuration space of that particular linkage. Since we have pinned A to the origin and B to the x axis, and we have four length constraints restricting the position of the other two points, the configuration space will be a 1 parameter curve in $\mathbb{R}^8$. This follows from the fact that we had 8 variables to begin with, 3 of which we set to 0 when A and B were pinned which left us with 5 free variables, further that we had 4 quadratic equations constraining the remaining 5 variables, leaving 1 free variable within this 8-space system. Now we want to classify what kind of curves we can get as configuration spaces of four-bar linkages.

3.2.2 The Empty Space

Lemma 3.2.1. *Let a, b, c, and d be the lengths of the bars of a four-bar linkage. Without loss of generality, assume that $a \geq b \geq c \geq d$. If a is larger than $b + c + d$, then the configuration space is empty. (We will call this a Dopey linkage since it cannot be constructed and therefore is just plane silly.)*

Proof. Any four-bar linkage with bar lengths a, b, c, d can be subdivided into two triangles by inserting a diagonal, of some length x, as seen in figure figure 3.2.1.

By the triangular inequality we have that $x + b > a$, and $d + x > x$. Adding these two inequalities gives us $b + c + d + x > a + x$, or by canceling x; $b + c + d > a$. Hence we know for any four-bar linkage, the sum of any three bars, must be larger than the remaining bar. This

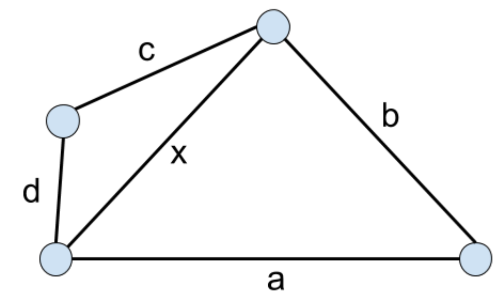

Figure 3.2.1. Triangulation of a four-bar linkage

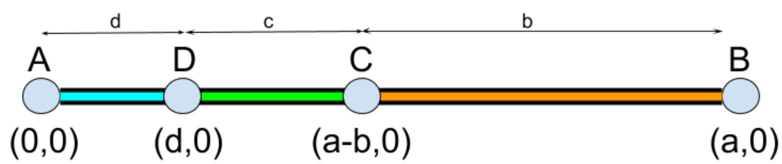

Figure 3.2.2. Linkage with $a = b + c + d$

shows by contradiction, that a linkage where the longest bar is greater in length than the sum of the other three cannot be constructed, thus rendering the configuration space empty. □

3.2.3 Configuration space is a point

Lemma 3.2.2. *If the longest bar is equal to the length of the sum of the other three links, then the configuration space is a point. (We will call this linkage a Grumpy linkage, since it doesn't ever want to move...because it can't.)*

Proof. If $a = b + c + d$ then we place the longest bar AB along the x axis, with A at the origin, then the placements of C and D are uniquely determined by the length constrictions of the linkage. See figure 3.2.2. □

3.2.4 Configuration space is a topological circle

Lemma 3.2.3. *If we assume for a four-bar linkage that the sum of the longest and shortest bars is* ***longer*** *than the sum of the other two, then the configuration space is a topological circle. (We*

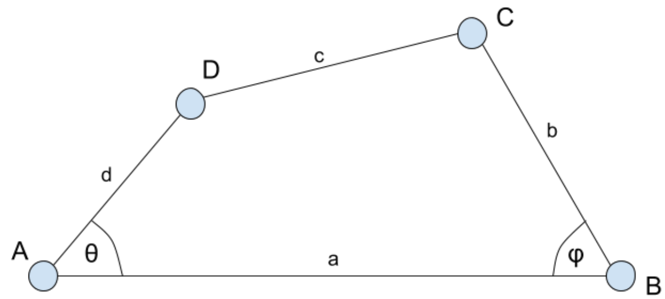

Figure 3.2.3. Linkage with $a + d > b + c$

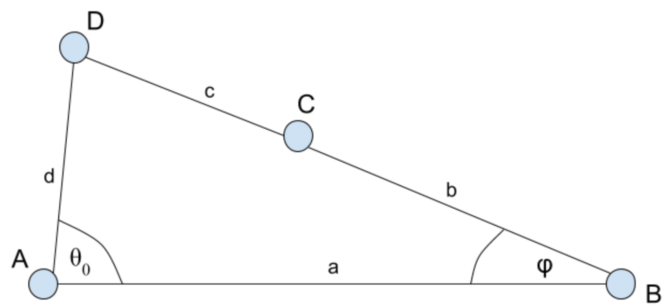

Figure 3.2.4. Extreme angle θ_0 of linkage with $a + d > c + b$

will call this type of linkage a Sleepy linkage, as it doesn't move all that much, and it doesn't do anything unpredictable.)

Proof. Again we position A at the origin, and B at $(a, 0)$. Then C has coordinates $(a - \cos(\phi), b\sin(\phi))$ and D has coordinates $(d\cos(\theta), d\sin(\theta))$, see figure 3.2.3. Moreover the following equation must be satisfied:

$$(d\cos(\theta) - (a - \cos(\phi)))^2 + (d\sin(\theta) - b\sin(\phi)) = c^2$$

Here the angle θ can be moved either positively or negatively to two extreme points, in both cases where DCB lie on a straight line, at which point there exists an extreme angle $\pm\theta_0$ which θ cannot exceed.

Given an angle θ such that $-\theta_0 < \theta < \theta_0$, this quadratic equation listed above gives two solutions for the angle ϕ in terms of θ. If θ approaches $\pm\theta_0$ then the two solutions for ϕ become arbitrarily close to each other. For $\theta = \pm\theta$ the two solutions for ϕ are the same.

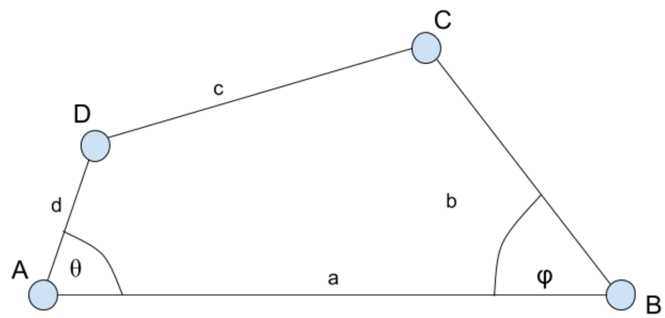

Figure 3.2.5. Linkage with $a + d < c + b$

For any θ beyond either of these extreme points, the linkage cannot be constructed with the given length constraints. Note that no bar has the ability to rotate a full 360 degrees with respect to a neighboring bar. This four-bar linkage is not Grashof.

We argue that the configuration space is a simple closed curve in $\mathbb{R}^8$ by projecting the configuration space onto the xy plane, by graphing just the y coordinates of the points C and D. Doing so yields a simple closed curve. $\square$

3.2.5 Configuration space is two disjoint topological circles

Lemma 3.2.4. *If we assume for a four-bar linkage that the sum of the longest and shortest bars is **shorter** than the sum of the other two, then the configuration space consists of two disjoint topological circles. (We will call this linkage Bashful, since the two topological circles do not touch and therefore seem shy of each other.)*

Proof. As always, we position A at the origin, and B at $(a, 0)$. Then C has coordinates $(a - \cos(\phi), b\sin(\phi))$ and D has coordinates $(d\cos(\theta), d\sin(\theta))$, see figure 3.2.5. Again, just as the previous section showed, the following equation must be satisfied:

$$(d\cos(\theta) - (a - \cos(\phi)))^2 + (d\sin(\theta) - b\sin(\phi)) = c^2$$

Here the angle θ can assume any value between 0 and 2π without restriction, due to the bar AD being able rotate around the point A without restriction. Even so, the above quadratic equation will still output two values for ϕ in relation to a given angle θ.

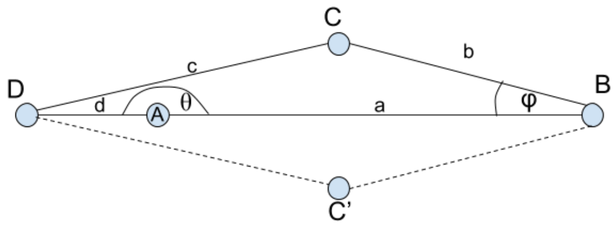

Figure 3.2.6. Extreme angle θ_0 of linkage with $a + d < c + b$

Given any angle θ the quadratic equation listed above gives two solutions for the angle ϕ in terms of θ. These two solutions for ϕ are never close to each other; the closest they can get is for $\theta = \pi$ see figure 3.2.6.

We argue that configuration space consists of two simple closed curves in $\mathbb{R}^8$ by projecting the configuration space onto the xy plane, by graphing just the y coordinates of the points C and D. Doing so yields two simple closed curves. $\square$

3.2.6 Configuration space is two topological circles touching at a point

Lemma 3.2.5. *When the sum of the largest and shortest bars of a four-bar linkage are equal in length to the sum of the two remaining bars, then the configuration space consists of two topological circles joined at a unique intersection point. (We will call this linkage Sneezy, since at the one intersection point of the topological circles, the linkage could possibly sneeze from one circle to the other without warning.)*

Proof. Consider a four-bar linkage for the sum of the lengths of the longest and shortest bars is equal to the sum of the lengths of the other two bars. These length constraints describe a Grashof linkage, so without loss of generality, we can have the shortest bar be neighboring the longest bar, as shown in Figure 3.2.7. When we consider this linkage, there is one unique point where the linkage has a choice to do four different actions. In $\mathbb{R}^8$ the coordinates of this unique point are $(0, 0, a, 0, a - b, 0, -d, 0)$. From this position the bar d has the ability to move either in a positive or negative angle θ, allowing four distinct solutions for ϕ and the location of the point C.

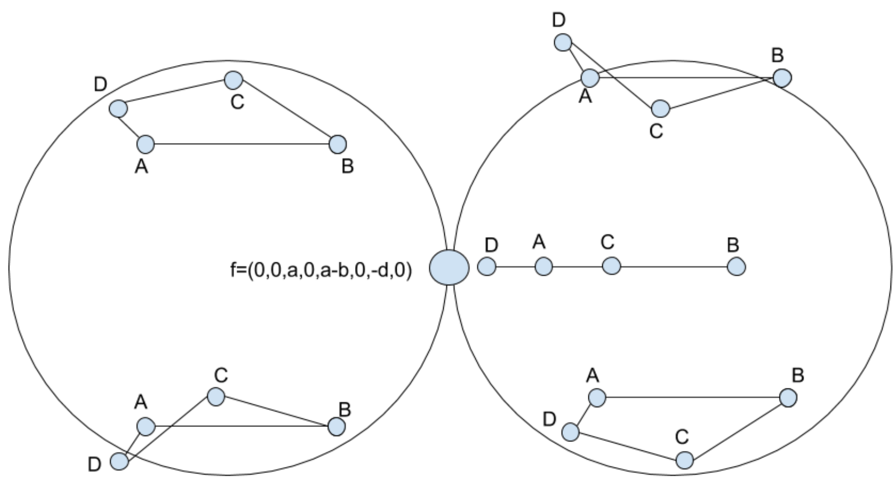

Figure 3.2.7. configuration space of $a + d = c + b$

If the bar d were to rotate upwards, the point C has the choice either to move up in relation to point D, or to move down.

If the bar d were to rotate down, the point C again has the choice to either move upwards with respect to D or down.

This shows us that the configuration space of this type of four-bar linkage is two topological circles intersecting at our special point $(0, 0, a, 0, a - b, 0, -d, 0)$.

□

3.2.7 Configuration space of a four-bar linkage with two sets of equal length bars

Lemma 3.2.6. *If the lengths of the bars a, b, c, and d are governed by the length constraints of $d + a = b + c$ and $d - a = c - b$, then the configuration space consists of two topological circles touching at two points. (We will call this linkage Happy, since it has so much freedom of choice; it can assume many configurations and comfortably perform its motions in any one of them or switch between them.)*

Proof. Adding or subtracting these two constraint equations gives us that $d = c$ and that $a = b$ respectively the four-bar mechanism has two special positions as seen in Figure 3.2.8. As we have always normalized, A is at $(0, 0)$, B is at $(a, 0)$ which forces the C and D in the special

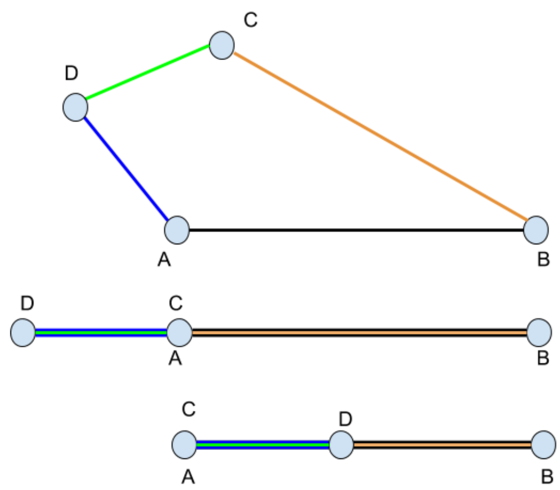

Figure 3.2.8. Special points of four-bar linkage with two sets of equal length bars

positions to have coordinates $C = (0, 0)$ and $D = (\pm d, 0)$ (depending at which special case we are talking about).

Now if we look at this four-bar linkage in 8-space, these two special positions have the coordinates: (0,0,a,0,0,0,-d,0) for the first case, and (0,0,a,0,0,0,d,0) for the second as seen in Figure 3.2.9. Now we can show that there are four unique paths we can take in 8-space in order to get from these two points, now labeled f_1 and f_2 respectively.

□

3.2.8 Configuration space when all four bars are of equal length

Lemma 3.2.7. *If the four-bar linkage is constructed with all four links of equal length, then the configuration space consists of three topological circles, each two of which intersect at a point. (We will call this linkage Doc, as it has the largest number of special points of all the linkage types, making it the true leader of the linkage dwarfs.)*

Proof. Even though this last case is the most complicated and different compared to the others, the proof is of a simpler nature. The equations for the length restrictions of the bars of the four-bar linkage are now $a = b = c = d$.

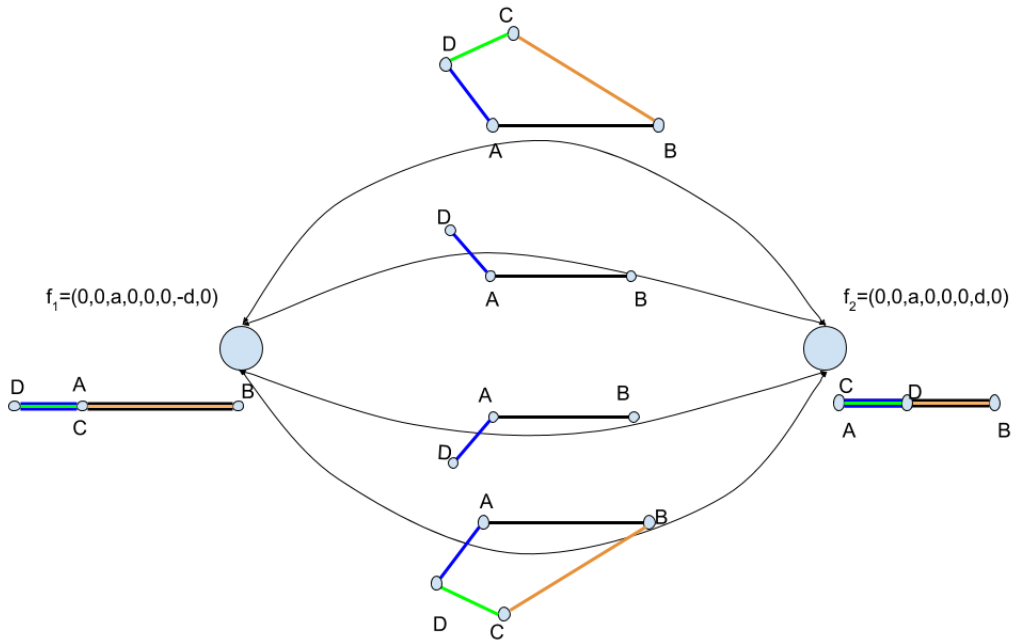

Figure 3.2.9. Configuration space of four-bar linkage with two sets of equal length bars

Without loss of generality we can set $a = 1$. Doing so gives rise to three special cases of the positions of A, B, C, and D, we will define these three points f_1, f_2, and f_3 as the points:

$$f_1 = (0, 0, 1, 0, 2, 0, 1, 0)$$

$$f_2 = (0, 0, 1, 0, 0, 0, -1, 0)$$

$$f_3 = (0, 0, 1, 0, 0, 0, 1, 0)$$

From each of these three points, there are two distinct ways to get to either other point. This can be visualized in figure 3.2.10

If we want to describe the paths from $f_1 \to f_2$, we first observe that $\phi = \theta$ because our four-bar linkage is a parallelogram. This forces the y coordinates of C and D to either both be positive, or both be negative. The top arc in figure 3.2.10 indicates the following curve:

$$(0, 0, 1, 0, 1 + \cos(\theta), \sin(\theta), \cos(\theta), \sin(\theta))$$

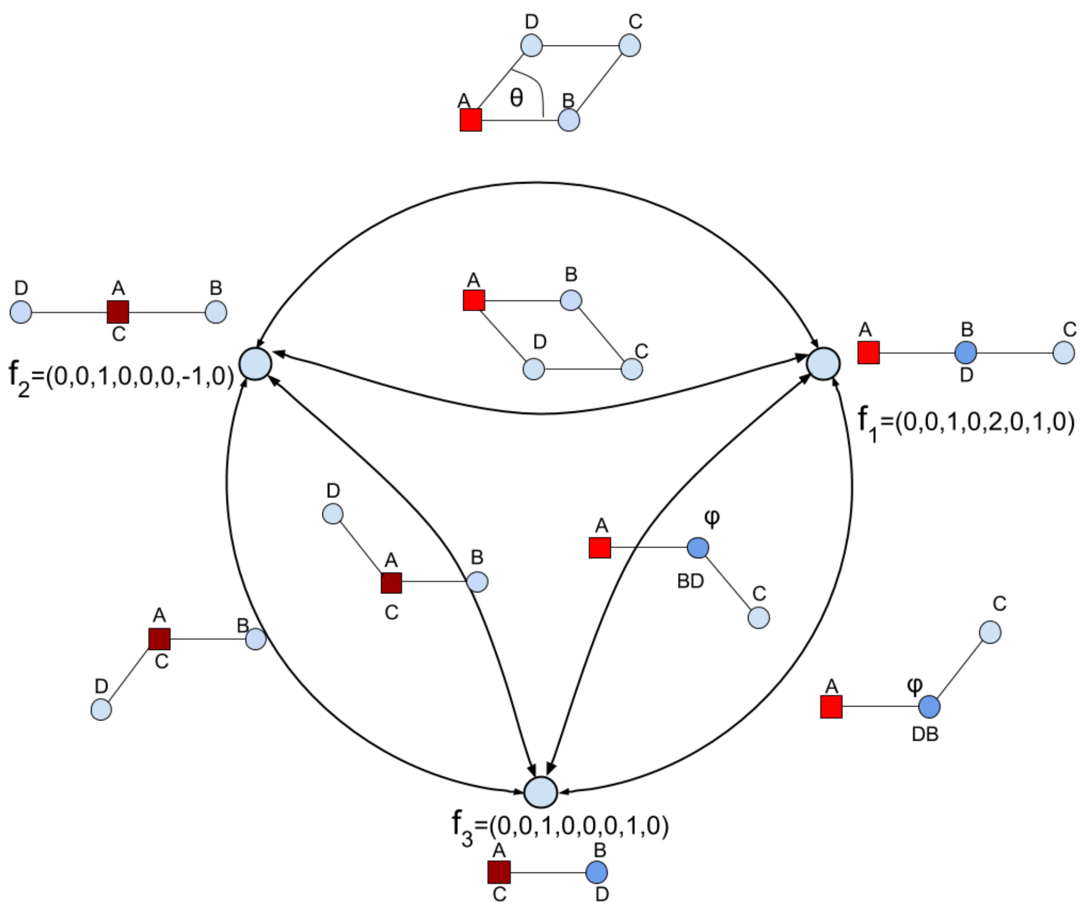

Figure 3.2.10. Configuration space of equal length bars

for θ between 0 and π. The lower arc from $f_1 \to f_2$ indicates the curve:

$$(0, 0, 1, 0, 1 + \cos(\theta), -\sin(\theta), \cos(\theta), -\sin(\theta))$$

which is the same curve as before, but with θ ranging from π to 2π.

If we want to go from $f_2 \to f_3$ we again have two options, with the outside arc in figure 3.2.10 describing the curve:

$$(0, 0, 1, 0, 0, 0, \cos(\theta), \sin(\theta)$$

and the inside arc describing the curve:

$$(0, 0, 1, 0, 0, 0, \cos(\theta), -\sin(\theta))$$

with θ ranging from pi to 2π.

If we want to go from $f_3 \to f_1$ we still have two paths to choose from as seen in figure 3.2.10. The difference here from the rest of the figure, is that θ is constantly 0 throughout the duration of either path. What angle is actually changing is ϕ. The outside are being described by the curve:

$$(0, 0, 1, 0, 1 + \cos(\phi), \sin(phi), 1, 0)$$

and finally the inside curve shown by:

$$(0, 0, 1, 0, 1 + \cos(\phi), -\sin(\phi), 1, 0)$$

with ϕ ranging from 0 to π.

□

3.3 The set of realizable normalized four-bar linkages.

In the book [3] it is shown that the topology the configuration space of a polygon depends only on the relative edge lengths of the bars.

We can illustrate this by the following example. Say we have a linkage with $a = b < c = d$. We can build a four-bar linkage from these edge lengths either as a parallelogram or as a kite(as was drawn in Lemma 3.2.6).

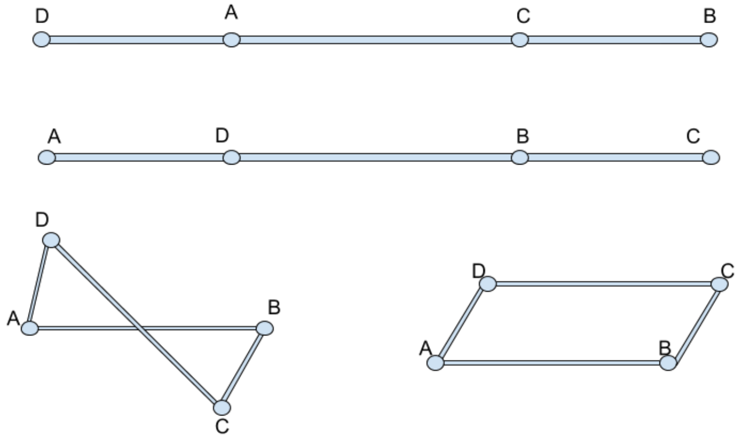

Figure 3.3.1. Special Positions of Parallelogram Linkage

We described the configuration space of the kite as two circles touching at two points 3.2.9. For the parallelogram we also get two flat positions, as seen in figure 3.3.1 but the vertices never coincide. Nevertheless from the flat positions which have coordinates $(0, 0, a, 0, a + c, 0, d, 0)$ and $(0, 0, a, 0, a - c, 0, -d, 0)$, there are also four possible ways to move between these two points. Either we can rotate both C and D up, or both down, or alternatively, we can move C and D contrarily from each other; if C rotates up D must go down and vice versa.

So although the kite and the parallelogram constructions of this linkage look and act differently, their configuration spaces are isomorphic.

Since the order of the edge lengths is not important nor are the absolute lengths, we normalize the edge lengths to sum up to 12, (any other constant will do, but 12 has a lovely number of divisors); $a + b + c + d = 12$. Suppose further that this four-bar linkage is constructed with links $a \geq b \geq c \geq d$.

From this constraint, we can see that the length of the bar a cannot be greater than 6, otherwise the linkage cannot be constructed, rendering the configuration space empty (as seen in Lemma 3.2.1).

If a were to be exactly equal 6, then the sum of the other three bars, $b + c + d = 6$, rendering the configuration space of a specific linkage as a point. Note that $b + c + d = 6$ is the equation of

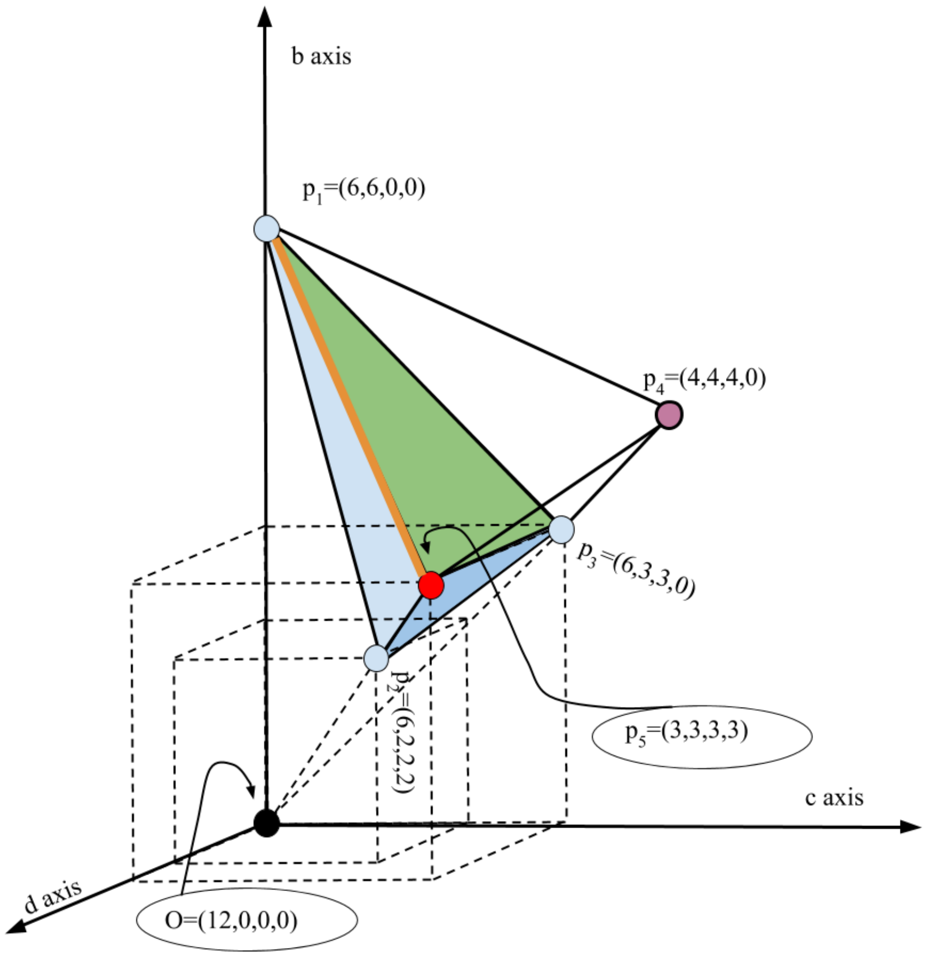

Figure 3.3.2. Configuration Types

a plane in $\mathbb{R}^3$ in the variables b, c, and d. Now we do not want to consider the entire plane, since we want all the variables to be positive and also require that $b \geq c \geq d$. Under these restrictions, all the points that satisfy $b + c + d = 6$ lie on the blue triangle described by points p_1, p_2, p_3 in Figure 3.3.2.

Any points extending from the origin up to, but not including, these three points, we have a shape that describes a tetrahedron which contains all the points where $b + c + d < 6$ forcing $a > 6$, thus rendering a non-constructible four-bar linkage. Any points coinciding with p_1, p_2 or p_3, or on the triangle they describe, form a constructible linkage whose configuration space is a point(as described in Lemma 3.2.2).

To determine the boundaries for constructible four-bar linkages with a circle as its configuration space, refer to Lemma 3.2.3, from which we can gather that $a + d > c + b$. By our normalization this would mean that $a = 12 - b - c - d$ so by substituting in for a we have that $12 - b - c - d + d > b + c$ which implies that $6 > b + c$. Observe that $b + c = 6$ is the equation of a plane in bcdspace, which is perpendicular to the cb plane. This plane is a boundary for the inequality $6 > b + c$ which is described by the blue tetrahedron under the green plane in Figure 3.3.2, this tetrahedron has the corners $p_1, p-2, p_3,$ and p_5. Its interior points correspond to four-bar linkages whose configuration space is a circle (the corners are not included, and are boundaries themselves).

For the case described by the Lemma 3.2.4, the inequality is the opposite of the previous, i.e. $a + d < b + c$. The inequality is satisfied by the points above the green plane in Figure 3.3.2. The points of this inequality lie in a tetrahedron with corners $p_1, p_3, p_5,$ and p_4.

The green plane itself(excluding its boundary line from p_1 to p_5) corresponds to Lemma 3.2.5, where the configuration space is two circles touching at a point.

The line segment from p_1 to p_5 corresponds to Lemma 3.2.6 because that is the one line where $a = b$ and $c = d$.

And finally the singular point p_5 corresponds to Lemma 3.2.7, where all bar lengths are equal, so $a = b = c = d$.

What is now possible with this normalization and careful categories, is we can take a bar of length 12(again it does not have to be 12, 12 is just a nice number) and divide this bar up into four lengths, and the resulting linkage will correspond to one point within this diagram. From which we can determine the characteristics of the linkage.

If we cut a bar of length 12 in to four random bars, we can also compute the probability of whether or not the linkage is constructible, Grashof, stuck, or any of the other types possible.

3.3.1 Probability to get a four-bar linkage of a particular type

Assume we have a bar of fixed length, no need to determine its length at the moment. We cut this rod randomly into four pieces and we ask "What is the probability that assembling these four pieces will create one of the linkage types described above?" To this end we compute the volumes of the tetrahedra found in 3.3.2.

Every possibility corresponds to a point inside the tetrahedron with vertices $(0, 0, 0)$, $(6, 0, 0)$, $(4, 4, 0)$, and $(3, 3, 3)$. Note that in the figure, our first coordinate is ignored(and not drawn) for each point, as it comes from normalizing the length. The volume of this tetrahedron, which gives us the total probability space, can be computed using the formula for the scalar triple product found from our dear friend: the calculus book [16], page 660 in chapter 9.

Total volume (V_t) is:

$$V_t = \frac{1}{6}\begin{vmatrix} 6 & 0 & 0 \\ 4 & 4 & 0 \\ 3 & 3 & 3 \end{vmatrix} = \frac{1}{6}(4 \cdot 6 \cdot 3) = 12.$$

The volume of the tetrahedron below the blue volume V_{dopey} is:

$$V_{dopey} = \frac{1}{6}\begin{vmatrix} 6 & 0 & 0 \\ 2 & 2 & 2 \\ 3 & 3 & 0 \end{vmatrix} = \frac{1}{6}(6 \cdot 2 \cdot 3) = 6.$$

Now for the blue tetrahedron we need to do a little conversion, since none of the points is the origin, we want to move the tetrahedron until one point is, and compute its volume from there. Hence we have $\begin{pmatrix} 2 & 2 & 2 \\ 3 & 3 & 0 \\ 6 & 0 & 0 \\ 3 & 3 & 3 \end{pmatrix} - \begin{pmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & -2 \\ 4 & -2 & -2 \\ 1 & 1 & 1 \end{pmatrix}$ Therefore the blue tetrahedron has volume V_{blue} is:

$$V_{blue} = \frac{1}{6}\begin{vmatrix} 1 & 1 & -2 \\ 4 & -2 & -2 \\ 1 & 1 & 1 \end{vmatrix} = \frac{1}{6}|(-2 - 2 - 8 - 4 - 4 + 2)| = \frac{1}{6}(18) = 3.$$

This leaves us with the tetrahedron above the blue volume/green plane, $V_{bashful}$ is:

$$V_{bashful} = \frac{1}{6}\begin{vmatrix} 3 & -3 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 3 \end{vmatrix} = \frac{1}{6}|(9 + 9)| = 3.$$

Hence it follows that the probability of getting four pieces that cannot be assembled into a quadrilateral is $\frac{6}{12} = \frac{1}{2}$. The probability of getting a constructible linkage is also one half, and the probability is split evenly between getting a sleepy mechanism and a bashful one. The other types of linkages, being lines, points, and planes on these tetrahedra, have probability 0.

4

Straight-Line Program

4.1 Theory of the code

Here we give an explanation of the code to be used to calculate the results of this project:

The main purpose of the processing code is to create a visual and computational aid for calculating the length of the straight line section of any imputed four-bar's coupler curve. Once we are able to determine a length of the straight-line portion of a curve, it allows us to determine which linkages perform their approximate straight-line motions more effectively than others.

4.1.1 Inputs

The program, whether it is for calculating the curve of the Watt engine or the Chebyshev machine, takes six inputs: four for the lengths of the bars and two for the x and y coordinates of the fixed bar endpoint[We only need to input one since it is assumed that the other endpoint is at $(0, 0)$].

4.1.2 Calculations

The calculations inside the Processing code are for the most part taken from the mathematica code, listed in the appendix, with some additional conditions. These calculations for determining the coupler curve are documented in the section on the Parametrization of a four-bar Linkage With One Fixed Bar. The added calculations for the Processing code are as follows: (1) Angles

ϕ, ψ, and θ are added in such a way as to ensure Processing does not attempt to draw points or gather data at positions where the linkage would be broken, i.e. not properly constructed.

(2) Arrays are used to store the values of each individual point.

(3) For a point to be displayed on the canvas, the point must have a slope within a user-implemented slope parameter.

4.1.3 Functions

The two functions that are used in the program are "void setup()" and "void display()." Setup is a built in processing function intended for the initialization of the program, or in other words, a place for the programmer to construct what their program should look like and how it should behave. The function Display was created uniquely for this program. Display has the task for listing through the Array for all the points of the coupler curve, and determining whether or not two neighboring points have a slope which matches the program's slope parameter: if the slope is within the correct bounds, the program draws the point, else it discards it. So in short, the display function weeds out the points which don't lead to or could be considered a straight-line.

4.1.4 IF Statements and For Loops.

The main For Loop of the program does all the calculation of where the mid point of the middle bar should be, and sets it that x and y coordinate into the corresponding data array. The loop of the function then changes the angle by the loop iterator, and does the whole calculation again with the new angle, thus giving a new point and a new input to the array. The loop will continue to loop until the program reaches the length of the loop iterator, and the array is filled.

4.1.5 Arrays

There are four arrays in this program, two for each set of x,y coordinates. The program draws the complete coupler curve by cutting the curve in two, and drawing each half separately, and because of that, there are two sets of x,y coordinates. They describe the same curve, just different sections of it. These four arrays are filled with the resulting calculations of the main For Loop of the program, giving each slot of the array the task of holding the x or y coordinate of a single point. The size of the array is determined by how many dots the user wants use to draw the coupler curve: the higher number of dots leads to a more accurate coupler curve.

4.1.6 Outputs

The Program outputs a visual representation of the straight line section of the coupler curve. It can also output the x and y coordinates of each point along that straight line section and its slope, thus allowing us to calculate the length and approximate straightness of the section.

4.2 Results and Findings

4.2.1 Predictions

From all of our preliminary research, the Chebyshev linkage seemed to be voiced as the more successful linkage, and therefore we assumed that this linkage should give us a longer straight line section given a stricter deviation parameter. However, since Watt never gave strict length constraints for his linkage, he actually created a family of linkages without explicitly meaning to. Any linkage with two oscillating arms describing contrary arcs suspending a tracer bar between them could qualify as being a Watt-style linkage. So exactly which Watt linkage Chebyshev was pitting his Lambda linkage against we are not sure, so going into this experiment there was a possible chance of Crawford's Watt linkage being the superior linkage.

4.2.2 Code Results

The results given by setting the required deviation to 0.05 from a straight line, gave us an expected result that Chebyshev's linkage produced a longer straight line section, as seen in Figures 4.2.1 and 4.2.2. To make the test fair, we made both linkages of relative size, since the total length(all the lengths of the bars added together) of the Chebyshev linkage is a pleasant 16 units, we scaled down the Crawford linkage such that instead of the total length being roughly 556.38 inches, it was also only 16 units.

However, a surprising result we found was that while the Chebyshev linkage created a longer straight line section at the .05 parameter mark, it in fact was not as true to a straight line as

Figure 4.2.1. Crawford linkage @ .05

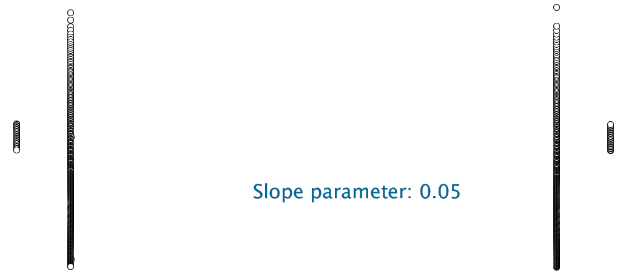

Figure 4.2.2. Chebyshev linkage @ .05

Figure 4.2.3. Crawford linkage @ .012

Figure 4.2.4. Chebyshev linkage @ .012

the Crawford linkage. If the parameter was increased in accuracy to anywhere below .0194, the Chebyshev straight line segment began to be divided into multiple shorter sections, while the Crawford linkage remained whole. In fact, not too far past .012 being set as the slope deviation parameter, the gaps between the split sections of the Chebyshev linkage are so great that its longest straight line segment is shorter than the undivided Crawford linkage segment, as seen in Figures 4.2.3 and 4.2.4.

This is a stark contradiction to the claim made by Chebyshev; in his book [2] he states that the deviation from a straight line is superior to Watt's engine by almost double. We must be careful here, as in our experiment we were calculating slope, and judging what we deemed to be straight by the calculating the slopes of neighboring points, whereas Chebyshev was calculating the raw distance from a straight line. Also of note, we have seemingly no way of telling which Watt linkage Chebyshev was using in his experiment, so while against the Crawford linkage Chebyshev seems to have done quite poorly, there very well may be another Watt variant out there for which the Chebyshev linkage is superior to.

The difficulty of being able to tell exactly what Chebyshev meant by his derivation stems from Chebyshev's calculations and conclusions possibly being lost in translation in the literal sense. The only direct account of them we could find was a 1899 copy of his memoir written in French. While the math can speak for itself for the most part, definitions for units, special symbols, and phrases have changed over time, so the translation of them can be difficult if not outright impossible. But the general idea that his linkage was superior to Watt's, has been retold and rewritten over and over in academia, seemingly without a thorough check. While this experiment could count as a check, again since we do not know which Watt linkage he was comparing his lambda linkage to, our experiment cannot qualify as a fair check.

5
Building a linkage

5.1 A Very Splendid and Worthwhile Linkage

5.1.1 Introduction and methodology

We know that from the construction of the Watt linkage, a whole family of linkages can be constructed. From that it should follow that there exists a set of optimal length constraints for a particular Watt type linkage such that its approximate straight line segment is longer than the rest of the linkages in the Watt family. As nice as it would be to find the ultimate Watt linkage, if such a linkage exists, there isn't enough time in our senior project to search for it.

Instead we shall follow in the footsteps of Peaucellier and Lipkin, to find an exact straight line mechanism. We know that a singular four-bar alone cannot achieve perfect straight line motion on its own, but multiple four-bars working together, manipulating each other, can accomplish this goal. For this reason we will incorporate three four-bar linkages fused together, similar to the Peaucellier cell. The Peaucellier cell can be seen as the combination of three four-bar mechanisms if one takes note of how the bars interact with each other.

In 5.1.1, the fixed bar is AB and the tracer point is F, (in green). The only construction requirements the Peaucellier linkages have are that CD, DF, FE, and EC are all the same length, bars EA and DA are of equal length, and finally it is required that as BC rotates about the point B, that point C passes directly through point A. All of this ensures that the point

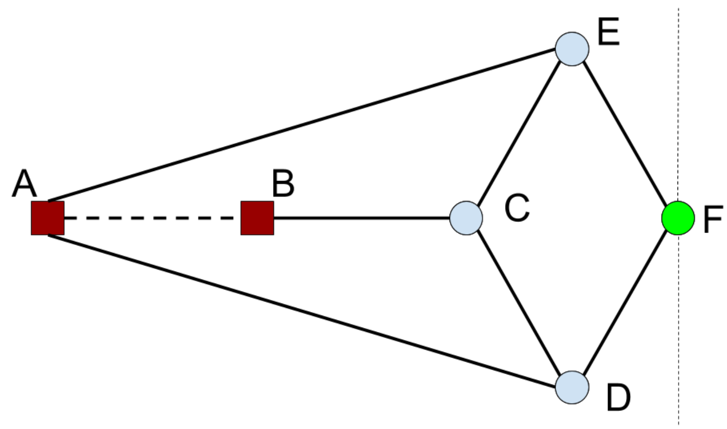

Figure 5.1.1. The Peaucellier Lipkin Linkage

F will draw a piece of an arc of an infinitely large circle, which means that the coupler curve that F describes is a straight line segment. The Peaucellier linkage works off the principle of the inverse, or inversion, of a circle. We can see from Figure 5.1.1 that the Peaucellier cell is a combination of three linkages when we note that bars AB, BC, CD, and DA, form a closed linkage, i.e. it would be a fully functioning linkage on its own, we will call this linkage the first of three. The second closed linkage is AB, BC, CE, and EA, which is the mirror image of the first, simply flipped over the horizontal. For the third we have CD, DF, FE, and EC which is a linkage with no fixed bar. It is the combination of these three linkages that creates the unique motion of the Peaucellier cell. In line with this mode of thinking, we will begin to construct our linkage.

5.1.2 Properties of four-bar linkages

Our linkage will take inspiration from both the Peaucellier and also the Sarrus linkage, all while adding a twist of our own (pun intended). What the Sarrus linkage and the Peaucellier linkage have in common is the compression of a rhombus by squeezing two opposing corners, i.e. from the figure 5.1.1 we can see that the point C is pushed into or pulled away from point F. It is the same sort of concept with the Sarrus linkage, except that in the case of the Sarrus linkage, this squeezing of the rhombus is happening in three dimensions [4]. In our linkage we will incorporate

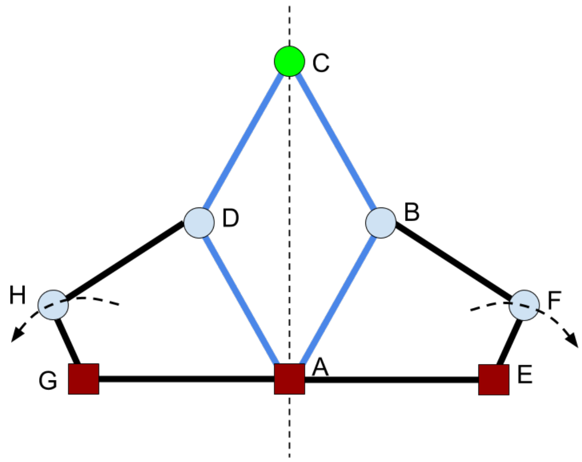

Figure 5.1.2. Peter's very splendid and worthwhile linkage using a Bashful-Doc-Bashful configuration

the compression of a rhombus to create straight line motion, but unlike the Sarrus linkage, it will be created in the plane, and unlike the Peaucellier cell, the rhombus will not be itself rotating.

Instead our rhombus will act like the first rung of a scissor lift. Our rhombus, which we will defined as having links AB, BC, CD, and DA and will have vertex A pinned to the plane. Vertices D and B will be the corners of the rhombus which we will pull or push in order to send point C on a linear path. We then attached two linkages on either side of the rhombus with reflective symmetry, these linkages can be either Bashful or Happy, in order to pull and squeeze vertices D and B, . The twist on our linkage comes from attaching a pulley with a twisted belt to the driving arms, GH and EF, of both of the non-rhombus linkages, see 5.1.2. The purpose of the pulley, as seen from the mechanical source book [13], forces links GH and EF to rotate in opposite directions from each other.

The differences between the two linkages, in Figure 5.1.3 and in Figure 5.1.2 are mainly the size of the straight-line segment which can be drawn, as well as the stability of the motion, but we'll get into stability in a later section. With the Happy-Doc-Happy configuration, the rhombus is flattened and stretched completely vertically both above and below the horizontal, i.e. the point C is allowed to pass right through point A making a straight line segment four times the length of a single bar of the rhombus. Whereas the Bashful-Doc-Bashful configuration

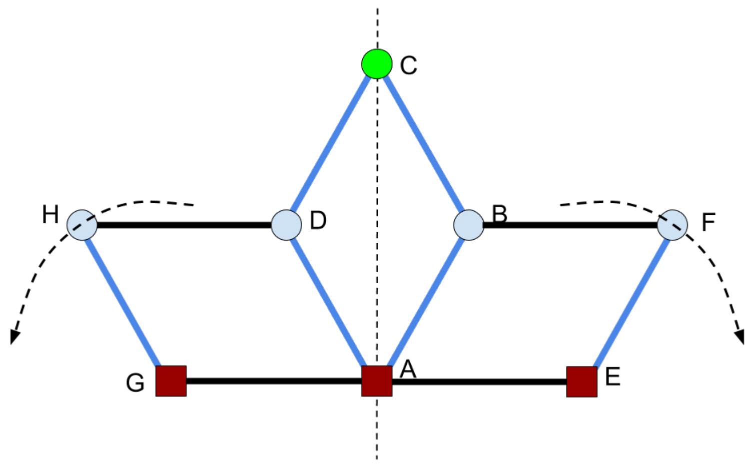

Figure 5.1.3. Peter's very splendid and worthwhile linkage, using a Happy-Doc-Happy configuration

creates a straight line segment above or below the horizontal depending on the initial position of the mechanism. The length of the segment drawn by the tracer is determined by the size and dimensions of the Grashof linkage used. In the Bashful-Doc-Bashful configuration, if the driving bars are short, then the straight line segment is also on the short end, but drawn in a simple manner, i.e. in a predictable fashion.

For the configuration space of this linkage being Doc-Doc-Doc, i.e. the two driving links on either side of the rhombus are also built with bars of all equal length, then we get a linkage which is prone to all sorts of chaos. As seen in the configuration space chapter, a Doc style linkage has the highest configuration space complexity. The linkages on either side of the rhombus could unexpectedly change their behavior into a 2-bar like action, and the linkage's straight line motion would then be interrupted until the driving linkages decided to change configuration space components. If we wanted to avoid having the complications that Doc type linkages are cursed with, but retain as much as possible the length of the straight line segment drawn by it, we can subtract ϵ from the driver bar lengths, i.e. shorten GH and EF by just a bit, so that the rhombus is now actuated by two Bashful linkages, with bars of almost equal length. Now since the linkages would be Bashful, we know that the motion of the driver linkages is not chaotic, and purely depends on which side of the horizontal the linkages are built. The straight line segment remains solely on one side of the horizontal, depending on construction. The only issue now being that the tracer point C draws the straight line segment in somewhat of an ugly way, aesthetically speaking: as the driving bar completes its rotation the tracer point draws the full segment more than once.

More specifically, and this explanation only applies to a Bashful-Doc-Bashful configuration, where the Bashful linkages are constructed very similarly to a Doc linkage, the motion of the point D, or B is very important. Assuming the motion of the driver bar HG starts when vertical, as H goes around in its Grashof manner, the motion of C will appear chaotic but our knowledge of configuration spaces helps us very much. We know since these driver linkages are Bashful, that the point D can never flip over to the other component of its configuration space during normal motion. This means as H is going around, the Bashful linkages goes from looking like an ordinary four-bar, to resembling a 2-bar, however never truly being one. Which means as H is passing through point A, that point D must whip from its seemingly 2-bar location near G to its normal four-bar location near E extremely quickly. This forces the point C, which has already completed a full drawing of the straight line segment to draw the segment again, only in reverse. As H goes back to being vertical, the point C is forced to 2 draw segments of a straight line, of equal length to each other, but smaller than the first two segments. In total, the point C draws four segments, one set of two being the longest possible segment, and a secondary set of 2 shorter straight line segments, all of which are drawn on top of each other and drawn at different velocities.

5.1.3 Coupler curve of our linkage

We know the coupler curve of our point C on our linkage is part of a perfect straight line. This is clear when we observe the construction of the linkage: since the twisted-band pulley is rotating links GH and EF in opposite directions, the links DA and BA are drawn in opposite directions, and since links DC and CB make up the rest of the rhombus $ABCD$, point C has no other option but to draw a straight line. Kempe, in his lecture "How To Draw a Straight Line" [9], mentions a linkage with just such a property, but gave no details on how the opposite motion could be achieved. So we are not the first mathematician to discover the mechanics of the rhombus, nor the first to realize that two bars forced to rotate in opposite directions can create straight line motion in a four-bar mechanism, but what we have achieved is a the documentation of the construction of a new linkage with solid explanation of how to accomplish bars rotating in opposite motion, ultimately to achieve straight line motion.

6
Appendices

6.1 Appendix

6.1.1 Mathematica Code

ClearAll[r,q,l,m,n,a,b,s,t,sx,sy,xp,xm,yp,ym]

r=1;

q=1;

l=1;

m=2;

n=1;

$a[\theta\_] := m - l * Cos[h]$

$b[\theta\_] := n + l * Sin[h]$

$s[\theta\_] := ((a[\theta]^2) + (b[\theta]^2) + (q^2) - (r^2))/(2 * \sqrt{[(a[\theta]^2) + (b[h]^2)]})$

$t[\theta\_] := \sqrt{[(q^2) - (s[\theta]^2)]}$

$sx[\theta\_] := (((a[\theta]^2) + (b[\theta]^2) + (q^2) - (r^2))/(2 * (a[\theta]^2 + b[\theta]^2))) * a[\theta]$

$sy[\theta\_] := (((a[\theta]^2) + (b[\theta]^2) + (q^2) - (r^2))/(2 * (a[\theta]^2 + b[\theta]^2))) * b[\theta]$

$xp[\theta\_] := sx[\theta] + (((b[\theta])\sqrt{[(a[\theta]^2) + (b[\theta]^2)]})t[\theta])$

$yp[\theta\_] := sy[\theta] - (((a[\theta])\sqrt{[(a[\theta]^2) + (b[\theta]^2)]})t[\theta])$

$xm[\theta\_] := sx[\theta] - (((b[\theta])\sqrt{[(a[\theta]^2) + (b[\theta]^2)]})t[\theta])$

$ym[\theta\_] := sy[\theta] + (((a[\theta])\sqrt{[(a[\theta]^2) + (b[\theta]^2)]})t[\theta])$

$\text{ParametricPlot}[((a[\theta] + xp[\theta])/2), ((b[\theta] + yp[\theta])/2), ((a[\theta] + xm[\theta])/2), ((b[\theta] + ym[\theta])/2), \theta, 0, 2\pi]$

6.1.2 Processing Code for Chebyshev linkage

```
// Initialization of mechanism and program.
//
// Changing any number in this list will yield a program which works
//
float middlebar = 200; // The length of the middle bar. Not negative. INPUT
float leftbar = 500; // The length of the left bar. Not negative.
float rightbar = 500; // The length of the right bar. Not negative.
float m = 0.0; // The this is the x coordniate of the endpoint.
float n = 400.0; // The this is the y coordinate of the rightbar endpoint.
int numints = 1000; // The number of intervals in each of the two regions.
// Initialization of dependent Program Parameters
// Below terms should not be changed.
float fixedbar = sqrt(m*m+n*n); // the length of the fixed bar
int loopyloop = 2*numints + 1; // The number of points computed. (N+1 in each interval)
float theta0 = atan(n/m); // The angle of the fixed bar with respect to horizontal
float phi = 0;
// The smallest angle between the fixed bar and right bar in range 0 to pi
float psi = PI;
// The largest angle between the fixed bar and right bar in range 0 to pi
float loopincrement = (psi-phi)/numints; // The angle change for each iteration of the loop
float first_theta_min = theta0 - psi; // Initial angle for the first angle group
float next_theta_min = theta0 + phi; // Initial angle for the second angle group
float theta = first_theta_min; // Initialization of theta
float a, b, p, s, t, x, y, sx, sy, xp, yp, xm, ym, x2, y2;
```

```
float[] Ax1=new float[loopyloop+1];
float[] Ax2=new float[loopyloop+1];
float[] Ay1=new float[loopyloop+1];
float[] Ay2=new float[loopyloop+1];
float slope(float a, float b, float p, float q) {
float lineslope = (a/(p-q))-(b/(p-q));
return lineslope;
}
void setup() {
size(2000, 2000);
background(200);
frameRate(60);
if (leftbar + middlebar ¡ fixedbar + rightbar)
{
psi=acos((rightbar*rightbar+fixedbar*fixedbar-(leftbar+middlebar)*(leftbar+middlebar)) /(2*rightbar*fixedbar));
}
if (abs(leftbar - middlebar) ¿ abs(fixedbar - rightbar))
phi = acos((rightbar*rightbar+fixedbar*fixedbar-(leftbar-middlebar)*(leftbar-middlebar)) /(2*rightbar*fixedbar));
}
for (int i=0; i < loopyloop+1; i++) {
a = m-rightbar*cos(theta);
b = n+rightbar*sin(theta);
p = sqrt((a*a)+(b*b));
s = ((a*a) + (b*b) + (leftbar*leftbar) - (middlebar*middlebar))/(2*sqrt((a*a) +(b*b)));
t = sqrt((leftbar*leftbar)-(s*s));
```

sx = a*(s/p);

sy = b*(s/p);

xp = (sx+((b/p)*t));

yp = (sy-((a/p)*t));

xm = (sx-((b/p)*t));

ym = (sy+((a/p)*t));

x = ((a+xp)/2); //sets the x coordinate of our tracer point of our coupler curve to be in the middle of the middle bar.

y = ((b+yp)/2); //sets the y coordinate of our tracer point of our coupler curve tobe in the middle of the middle bar.

//we shift things over by n, to get the whole curve in the frame.

x2 = ((a+xm)/2);

y2 = ((b+ym)/2);

theta = theta + loopincrement; //changes theta for the next loop pass

if (i == numints) {

theta = next_theta_min;

}

Ax1[i]=x;

Ax2[i]=x2;

Ay1[i]=y;

Ay2[i]=y2;

//if the first interval is just finished, we skip to the second.

}

println(middlebar+rightbar+leftbar+fixedbar); //println is mainly used to debug the program, but also display any data we need.

display(Ax1, Ay1, Ax2, Ay2);

}

```
void display(float[] Ax1, float[] Ay1, float[] Ax2, float[] Ay2) {
for (int i=0; i < loopyloop; i++) {
if (abs(slope(Ax1[i+1], Ax1[i], Ay1[i+1], Ay1[i]))¡.0192) {
ellipse(Ax1[i]+500, Ay1[i]+500, 10, 10);
println(abs(Ay1[i]));
}
}
for (int i=0; i < loopyloop; i++) {
if (abs(slope(Ax2[i+1], Ax2[i], Ay2[i+1], Ay2[i]))¡.0192)
ellipse(Ax2[i]+500, Ay2[i]+500, 10, 10);
}
}
}
```

6.1.3 Processing code for Crawford linkage.

```
float MidBar=27;
float LBar=132;
float RBar=132;
float FixBar=sqrt((RBar+LBar)*(RBar+LBar)+(MidBar*MidBar));
float TotLength=MidBar+LBar+RBar+FixBar;
float totalsize=16;
float correctsizemaker =TotLength/totalsize; //once this is factored into the length of each bar, the total length will then be 16.
float middlebar = (MidBar/correctsizemaker)*100; // The length of the middle bar. Not negative. INPUT, we multiply by 100 to see it visually
float leftbar = (LBar/correctsizemaker)*100; // The length of the left bar. Not negative.
float rightbar = (RBar/correctsizemaker)*100; // The length of the right bar. Not negative.
float m = 2*(LBar/correctsizemaker)*100; // The this is the x coordniate of the endpoint.
```

```
float n = (MidBar/correctsizemaker)*100; // The this is the y coordinate of the rightbar endpoint.
// Initialization of dependent Program Parameters
// Below terms should not be changed.
float fixedbar = sqrt(m*m+n*n); // the length of the fixed bar
int loopyloop = 2*numints + 1; // The number of points computed. (N+1 in each interval)
float theta0 = atan(n/m); // The angle of the fixed bar with respect to horizontal
float phi = 0;
// The smallest angle between the fixed bar and right bar in range 0 to pi
float psi = PI;
// The largest angle between the fixed bar and right bar in range 0 to pi
float loopincrement = (psi-phi)/numints; // The angle change for each iteration of the loop
float first_theta_min = theta0 - psi; // Initial angle for the first angle group
float next_theta_min = theta0 + phi; // Initial angle for the second angle group
float theta = first_theta_min; // Initialization of theta
float a, b, p, s, t, x, y, sx, sy, xp, yp, xm, ym, x2, y2;
float[] Ax1=new float[loopyloop+1];
float[] Ax2=new float[loopyloop+1];
float[] Ay1=new float[loopyloop+1];
float[] Ay2=new float[loopyloop+1];
float slope(float a, float b, float p, float q) {
float lineslope = (a/(p-q))-(b/(p-q));
return lineslope;
}
void setup() {
size(2000, 2000);
background(200);
```

frameRate(60);

if (leftbar + middlebar ¡ fixedbar + rightbar)

{

psi= acos((rightbar*rightbar+fixedbar*fixedbar-(leftbar+middlebar)*(leftbar+middlebar)) /(2*rightbar*fixedbar));

}

if (abs(leftbar - middlebar) ¿ abs(fixedbar - rightbar))

phi = acos((rightbar*rightbar+fixedbar*fixedbar-(leftbar-middlebar)*(leftbar-middlebar)) /(2*rightbar*fixedbar));

}

for (int i=0; i < loopyloop+1; i++) {

a = m-rightbar*cos(theta);

b = n+rightbar*sin(theta);

p = sqrt((a*a)+(b*b));

s = ((a*a) + (b*b) + (leftbar*leftbar) - (middlebar*middlebar))/(2*sqrt((a*a) +(b*b)));

t = sqrt((leftbar*leftbar)-(s*s));

sx = a*(s/p);

sy = b*(s/p);

xp = (sx+((b/p)*t));

yp = (sy-((a/p)*t));

xm = (sx-((b/p)*t));

ym = (sy+((a/p)*t));

x = ((a+xp)/2); //sets the x coordinate of our tracer point of our coupler curve to be in the middle of the middle bar.

y = ((b+yp)/2); //sets the y coordinate of our tracer point of our coupler curve tobe in the middle of the middle bar.

//we shift things over by n, to get the whole curve in the frame.

x2 = ((a+xm)/2);

y2 = ((b+ym)/2);

theta = theta + loopincrement; //changes theta for the next loop pass

if (i == numints) {

theta = next_theta_min;

}

Ax1[i]=x;

Ax2[i]=x2;

Ay1[i]=y;

Ay2[i]=y2;

//if the first interval is just finished, we skip to the second.

}

println(middlebar+rightbar+leftbar+fixedbar); //println is mainly used to debug the program, but also display any data we need.

display(Ax1, Ay1, Ax2, Ay2);

}

void display(float[] Ax1, float[] Ay1, float[] Ax2, float[] Ay2) {

for (int i=0; i < loopyloop; i++) {

if (abs(slope(Ax1[i+1], Ax1[i], Ay1[i+1], Ay1[i]))¡.0192) {

ellipse(Ax1[i]+500, Ay1[i]+500, 10, 10);

println(abs(Ay1[i]));

}

}

for (int i=0; i < loopyloop; i++) {

if (abs(slope(Ax2[i+1], Ax2[i], Ay2[i+1], Ay2[i]))¡.0192)

ellipse(Ax2[i]+500, Ay2[i]+500, 10, 10);

}

```
}
}
```

Bibliography

[1] John Bryant and Chris Sangwin, *How round is your circle?: Where engineering and mathematics meet*, Princeton University Press, 2008.

[2] P.L. Chebyshev and A.A. Markov, *Oeuvres de P.L. Tchebychef*, St. Petersbourg, Commissionaires de l'Academie imperiale des sciences, Unknown Month 1899.

[3] Michael Farber, *Invitation to topological robotics*, Zurich Lectures in Advanced Mathematics, European Mathematical Society (EMS), Zürich, 2008. MR2455573

[4] Michael Goldberg, *Polyhedral linkages*, National Mathematics Magazine **16** (1942), no. 7, 323–332.

[5] D. Jordan and M. Steiner, *Configuration spaces of mechanical linkages*, Discrete Comput. Geom. **22** (1999), no. 2, 297–315. MR1698549

[6] G. Keady, P. Scales, and S. Z. Nmeth, *Watt linkages and quadrilaterals*, The Mathematical Gazette **88** (2004), no. 513, 475–492.

[7] Grant Keady, P J. Scales, and Gary Fitz-Gerald, *Envelopes generated by circular-arc coupler-bars in james watt's four-bar linkages* (201711).

[8] A. B. Kempe, *On a general method of describing plane curves of the nth degree by linkwork*, Proceedings of the London Mathematical Society **s1-7**, no. 1, 213–216, available at `https://londmathsoc.onlinelibrary.wiley.com/doi/pdf/10.1112/plms/s1-7.1.213`.

[9] Alfred Bray Kempe, *How round is your circle?: Where engineering and mathematics meet*, Ithaca, New York: Cornell University Library, 1877.

[10] Olivier Mermoud and Marcel Steiner, *Visualisation of configuration spaces of polygonal linkages*, J. Geom. Graph. **4** (2000), no. 2, 147–157. MR1829539

[11] M. Muller, *A novel classification of planar four-bar linkages and its application to the mechanical analysis of animal systems*, Philosophical Transactions: Biological Sciences **351** (1996), no. 1340, 689–720.

[12] Eric Robinson and A. E. Musson., *James watt and the steam revolution: a documentary history.*, London: Adams and Dart, 1969.

[13] N. Sclater and N. Chironis, *Mechanisms and mechanical devices sourcebook, fourth edition*, Mcgraw-hill, 2007.

[14] A. B. Sossinsky, *Configuration spaces of planar mechanical linkages with one degree of freedom*, Russ. J. Math. Phys. **15** (2008), no. 4, 530–541. MR2470855

[15] Alexey Sossinsky, *Configuration spaces of planar linkages*, Handbook of Teichmüller theory. Vol. VI, 2016, pp. 335–373. MR3618193

[16] J. Stewart, *Calculus: Concepts and contexts*, Available 2010 Titles Enhanced Web Assign Series, Cengage Learning, 2009.