EXPLORING LINKAGES

JOEL STEINBERG, XIAXIN LI, AND NIRMIT JALLAWAR

Abstract. In this paper we explore the solution space to the Alt-Burmester synthesis problem for the four bar linkage and the Stephenson 3 six bar linkage using polynomial systems.

Contents

1. Introduction

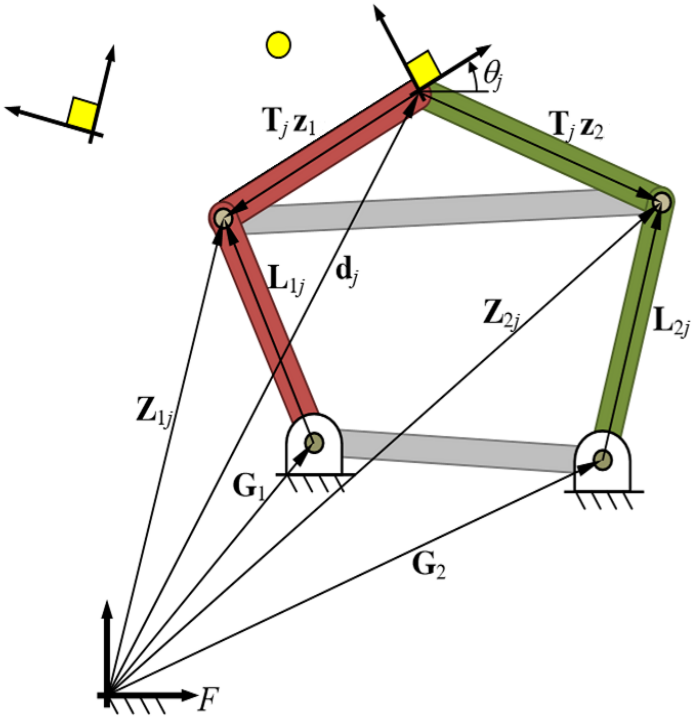
In this paper we will explore linkages of both the 4 bar and 6 bar varieties, setting up polynomial systems, solving them and trying to infer from the results. We will focus on solving the synthesis problems using numerical methods with Bertini through Macaulay2. We will also try to find the fastest method to solve these polynomial systems.

2. 4 bar linkages

In this section we briefly describe the Alt-Burmester synthesis problem for 4 bar linkages and specify the polynomial system used for solving it. We then compare our results for the degree of different specifications to those found by Brake, Hauenstein, Murray, Myszka, and Wampler (BHMMW) [1], and compare how different methods of breaking up the angle constraint into linear subsystems affect the runtime.

Date: August 14, 2020.

FIGURE 1. A four bar linkage [1]



2.1. **Outline of the 4 bar Alt-Burmester Synthesis Problem.**

The objective of the Alt-Burmester 4 bar synthesis problem is to construct a four bar linkage whose top, or coupler point, can pass through an arbitrary set of points given as parameters. An example of a four bar linkage is given in Figure 1. A 4 bar linkage consists of the following:

(1) Fixed points: The four bar linkage consists of two fixed points, denoted G_1 and G_2 in Figure 1, which are invariant as the linkage moves.
(2) Proximal Links: The two proximal links connect the two fixed points G_1 and G_2 to the two circle points, Z_{1j} and Z_{2j} respectively in Figure 1.
(3) Distal Links: There are two distal links which connect the circle points Z_{1j} and Z_{2j} to the coupler point, located at d_j.
(4) Dyads: Each arm of the linkage, which are red and green in Figure 1, are referred to as dyads.
(5) A coupler triangle: The triangle between the two circle points Z_{1j} and Z_{2j} and the coupler point is referred to as the coupler triangle.

As the linkage moves, the two circle points Z_{1j} and Z_{2j} circle around the fixed points G_1 and G_2 respectively (hence the name), while the coupler triangle remains fixed. In other words, the angle between the two distal links remains constant as the linkage moves.

For any given point, it is also possible to specify a rotation angle along with the point. The rotation angle is the angle which you would like the coupler triangle to be rotated

by when it fits a given point. An example is given by θ_j in Figure 1. When a rotation angle is specified along with a point, we refer to both together as a pose. The rotated distal links are given by $T_j z_1$ and $T_j z_2$ in Figure 1, where z_1 and z_2 remain constant as the linkage fits different points, and T_j represents a rotation matrix corresponding to a given rotation angle θ_j, i.e.

$$T_j = \begin{bmatrix} \cos\theta_j & -\sin\theta_j \\ \sin\theta_j & \cos\theta_j \end{bmatrix}.$$

We typically subscript values for a given point or pose with the letter j, for example d_j is the point the 4 bar linkage is fitting for point (or pose) j.

2.2. **Setup of the polynomial system.**

The setup of the polynomial system used in this paper is analogous to that found in BHMMW [1], and variables for this problem are illustrated in Figure 1.

2.2.1. *Isotropic coordinates.*

To allow for complex solutions and to conveniently represent rotations, instead of working in $\mathbb{R}^2$ we use isotropic coordinates in specifying this problem [2]. A vector (x, y) in $\mathbb{C}^2$ is represented in isotropic coordinates by

$$(x + y\sqrt{-1}, x - y\sqrt{-1}).$$

In other words, if $(z, \bar{z})$ is the isotropic representation of a vector (x, y) in $\mathbb{C}^2$, we have

$$\begin{bmatrix} z \\ \bar{z} \end{bmatrix} = \begin{bmatrix} 1 & \sqrt{-1} \\ 1 & -\sqrt{-1} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \text{ and } \begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ -\sqrt{-1} & \sqrt{-1} \end{bmatrix} \begin{bmatrix} z \\ \bar{z} \end{bmatrix}.$$

From this, it follows that for any scalars c_1 and c_2 in $\mathbb{C}$,

$$c_1 x + c_2 y \;=\; \frac{c_1}{2}(z + \bar{z}) - \frac{i c_2}{2}(z - \bar{z}).$$

Additionally, if (x, y) is in $\mathbb{R}^2$, the coordinates of the isotropic representation of (x, y) given by $(z, \bar{z})$ will be complex conjugates, so

$$\|(x, y)\|_2^2 = x^2 + y^2 \;=\; z\bar{z}. \tag{1}$$

Isotropic coordinates are particularly convenient for representing vectors rotated by a rotation matrix. Let (x, y) be a vector in $\mathbb{R}^2$, and let T_j be the rotation matrix corresponding to some angle θ_j. Then,

$$T_j \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\theta_j & -\sin\theta_j \\ \sin\theta_j & \cos\theta_j \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\theta_j x - \sin\theta_j y \\ \sin\theta_j x + \cos\theta_j y \end{bmatrix}.$$

If we transform this into isotropic coordinates and apply Euler's formula we get:

$$\begin{aligned}
\begin{bmatrix} 1 & \sqrt{-1} \\ 1 & -\sqrt{-1} \end{bmatrix} T_j \begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} 1 & \sqrt{-1} \\ 1 & -\sqrt{-1} \end{bmatrix} \begin{bmatrix} \cos\theta_j x - \sin\theta_j y \\ \sin\theta_j x + \cos\theta_j y \end{bmatrix} \\
&= \begin{bmatrix} x(\cos\theta_j + \sqrt{-1}(\sin\theta_j)) + y\sqrt{-1}(\cos\theta_j + \sqrt{-1}(\sin\theta_j)) \\ x(\cos\theta_j - \sqrt{-1}(\sin\theta_j)) - y\sqrt{-1}(\cos\theta_j - \sqrt{-1}(\sin\theta_j)) \end{bmatrix} \\
&= \begin{bmatrix} e^{i\theta_j} & e^{-i\theta_j} \end{bmatrix} \begin{bmatrix} x + y\sqrt{-1} \\ x - y\sqrt{-1} \end{bmatrix} \\
&= \begin{bmatrix} e^{i\theta_j} & e^{-i\theta_j} \end{bmatrix} \begin{bmatrix} 1 & \sqrt{-1} \\ 1 & -\sqrt{-1} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.
\end{aligned}$$

In other words, the rotation of a real vector (x, y) by angle θ_j in isotropic coordinates is given by $\begin{bmatrix} e^{i\theta_j} & e^{-i\theta_j} \end{bmatrix}$ times the vector (x, y) in isotropic coordinates. As a result, for a given point or pose j and dyad r we can represent the rotated distal links in isotropic coordinates by

$$\begin{bmatrix} T_j z_r & \bar{T}_j \bar{z}_r \end{bmatrix}.$$

This corresponds to the vectors $T_j z_1$ and $T_j z_2$ in Figure 1. If j is a pose and we are given a rotation angle θ_j, then T_j and $\bar{T}_j$ are parameters given by $e^{i\theta_j}$ and $e^{-i\theta_j}$ respectively. If j is a point, then T_j and $\bar{T}_j$ are complex variables.

2.2.2. *Constraints.*

For the general 4 bar synthesis problem, we are specified N points and M poses. These consist of points for the linkage to fit in isotropic coordinates denoted by $(d_j, \bar{d}_j)$ for $j = 1, ..., M + N$, and rotation angles θ_j for $j = 1, ..., M$ corresponding to each of the M poses (we assume without loss of generality that the poses are specified first). We follow an identical procedure to that used by BHMMW for specifying the polynomial system.

The first set of constraints imposes the restriction that for every point or pose j, the top of each dyad r must be located at $(d_j, \bar{d}_j)$, i.e. for $r = 1, 2$ and $j = 1, ..., M + N$,

$$d_j = G_r + L_{rj} - T_j z_r \tag{2}$$

$$\bar{d}_j = \bar{G}_r + \bar{L}_{rj} - \bar{T}_j \bar{z}_r. \tag{3}$$

However, these constraints alone are not sufficient as the vectors $(L_{rj}, \bar{L}_{rj})$, corresponding to the orientation of each proximal link for a point j, can have variable magnitudes as the linkage fits different points under these constraints. As a result, we also impose the restriction that the length of each proximal link must remain constant across all points j. Using the result from equation (1) this implies

$$L_{r1}\bar{L}_{r1} = L_{rj}\bar{L}_{rj}, \tag{4}$$

where j varies from $2, ..., M + N$. Finally, when j is a point we are not specified a rotation angle, so T_j and $\bar{T}_j$ are variables. If T_j and $\bar{T}_j$ corresponded to a valid rotation angle θ_j we would have

$$T_j \bar{T}_j = e^{i\theta_j} e^{-i\theta_j} = 1. \tag{5}$$

As a result, we impose this constraint on T_j and $\bar{T}_j$ for j from $M+1$ to $M+N$. It should be noted however that this is a necessary condition for T_j and $\bar{T}_j$ to be valid, not a sufficient condition. This is discussed further in Section 2.5.1.

To reduce the number of variables for this problem, as is done in BHMMW, we rearrange equations 2 and 3 for L_{rj} and $\bar{L}_{rj}$ respectively and then substitute into the length constraints for the proximal links given by equation 4. This yields

$$(d_1 - G_r + T_1 z_r)(\bar{d}_1 - \bar{G}_r + \bar{T}_1 \bar{z}_r) = (d_j - G_r + T_j z_r)(\bar{d}_j - \bar{G}_r + \bar{T}_j \bar{z}_r) \tag{6}$$

for each dyad r and point/pose j.

2.3. **Specification summary.**

In short, our specification for the 4 bar problem is as follows. Let N be the number of points specified, let M be the number of poses specified, and assume that $j = 1, ..., M$, correspond to poses while $j = M+1, ..., M+N$ correspond to points. The parameters for this problem are given by

$$d_j, \bar{d}_j$$

for $j = 1, ..., M+N$, and

$$T_j, \bar{T}_j$$

for $j = 1, ..., M$. The variables for this problem are given by

$$G_r, \bar{G}_r$$
$$z_r, \bar{z}_r$$
$$T_j, \bar{T}_j$$

for $r = 1, 2$ and $j = M+1, ..., M+N$. Finally, the constraints for this problem are given by

$$(d_1 - G_r + T_1 z_r)(\bar{d}_1 - \bar{G}_r + \bar{T}_1 \bar{z}_r) = (d_j - G_r + T_j z_r)(\bar{d}_j - \bar{G}_r + \bar{T}_j \bar{z}_r)$$

for $r = 1, 2$ and $j = 1, ..., M+N$, and

$$T_j \bar{T}_j = 1$$

for $j = M+1, ..., M+N$.

2.4. **Our findings.**

We used Macaulay2 to call Bertini to find the solutions for the Alt-Burmester problem.

We used Bertini out of the box, setting up the system in Macaulay2 and then calling Bertini for solving the system.

2.4.1. *Summary.*

The results of running computations for the different cases are summarized in Table 1. We can see that number of solution varies from 1 for the $(2, 1)$ case to 2388 for the $(2, 5)$ case. The latest code is available on our GitHub page. [1]

[1] https://github.com/jallawar/fourBarBertiniSolve

	Poses					
Points	0	1	2	3	4	5
0	*	1	4	16	16	16
1	*	7	24	64	48	
2	4	43	134	194	60	
3	54	234	552	362		
4	256		1554	402		
5	1382		2388			
6			2224			
7						
8						

TABLE 1. Table summarizing the number of solutions projected onto the center point coordinates

2.4.2. *Cases with only poses specified.*

Our method can successfully find the correct number of solutions for cases where we only have M poses and no precision points ($N = 0$). The cases with just 0 or 1 points are not in the scope of our code.

For the other cases we have listed the degree in Table 1.

2.4.3. *Run times of 4-bar linkage cases.*

In this section we summarize the runtimes for solving the 4 bar synthesis problem using homotopies beginning from different start systems. The start systems differ based on how they alter the angle constraints, $T_j\bar{T}_j = 1$. We use (M, N, S) to denote the case of M poses, N points, with scale S, where scale refers to the maximum magnitude of specified points. Table 2 summarizes the time performance of different homotopy methods of the (4,1,1) case in seconds.

Homotopy Methods	1st	2nd	3rd	4th	5th	6th	Average
NAIVE	27.2161	50.4405	12.9031	28.0468	12.2433	43.2775	29.0212
MONIC	11.0640	39.5249	33.6356	41.8097	46.5984	27.4726	33.3509
SUPERSYM	44.5919	18.1906	11.2339	53.2096	11.7784	26.3301	27.5558
SUPERCONJ	11.8610	37.8893	20.9750	11.0921	46.2271	47.1744	29.2032
UNITPMONE	22.7459	11.5785	46.4966	11.4366	22.5805	37.7281	25.4277
UNITONE	37.6458	43.4926	12.2187	48.8843	29.4407	11.0176	30.4500
UNITii	86.7282	11.5750	53.8651	11.1303	11.2865	22.2495	32.8057
UNITPMii	65.0680	23.9112	13.1919	11.1793	12.9529	29.2141	25.9196
ORIGIN	23.0851	16.2737	11.4649	18.2650	18.9897	10.9133	16.4986

TABLE 2. Performance of homotopy methods for (4,1,1) case

From this chart, we can see that the ORIGIN homotopy method is significantly better than the other homotopy methods in terms of computation time.

Other than this, we can see that the variance of computation time is large among all homotopy methods, indicating a high inconsistency of computation time of all homotopy methods. Methods with relatively low variance in the long term are: UNITPMONE, UNITONE, and ORIGIN.

2.4.4. *Comparing the competency of the Pre-circle and the Post-circle steps.*

In the previous subsection, the times listed in Table 2 are the sum of times of the Pre-circle and the Post-circle steps. In fact, the Post-circle step takes significantly less time compared to the Pre-circle step. However, it always find less solutions than the Pre-circle step, as illustrated by the next table. Particularly, for the (4,1,1) case, there should be 48 solutions [1]. The next table summarizes the number of solutions found and the calculation time (rounded to nearest integer) by Pre-circle and Post-circle steps of three attempts for each homotopy method, where (A,B;C,D) means that Pre-circle finds A solutions in B seconds while Post-circle finds C solutions in D seconds.

Homotopy Methods	1st	2nd	3rd
NAIVE	(46,33;44,0)	(48,12;48,0)	(45,11;45,0)
MONIC	(48,22;48,0)	(48,45;48,0)	(48,47;48,0)
SUPERSYM	(43,12;42,0)	(48,46;48,0)	(47,30;47,0)
SUPERCONJ	(46,27;41,19)	(48,12;47,1)	(48,38;46,38)
UNITPMONE	(46,13;45,0)	(48,26;48,0)	(48,20;48,0)
UNITONE	(44,27;41,5)	(48,33;44,4)	(43,44;41,0)
UNITii	(48,35;48,1)	(46,11;46,1)	(41,32;41,0)
UNITPMii	(43,52;40,0)	(47,11;45,1)	(46,35;46,0)
Origin	(47,28;44,4)	(48,22;46,2)	(48,11;46,5)

TABLE 3. Performance of Pre-circle and Post-circle steps for (4,1,1) case

From this table, we can again see the high inconstancy of the homotopy methods. However, more attempts have shown that the MONIC homotopy method is relatively steady in finding all 48 solutions. The UNITONE ,UNITii and UNITPMii methods often find significantly less solutions, which means they are relatively less reliable than other methods in getting all of the solutions.

Also notable is that the SUPERCONJ methods can have poor performance of the Post-circle step, in term of computation time.

Finally, we list the performance of NAIVE and ORIGIN homotopy method of a bigger case, namely (4,2,1), using the same notation as Table 3. There should be 60 solutions.

From this table and extra attempts, we can conclude that the Post-circle step of the NAIVE method generally takes less than 1 second and will find most of solutions, while the Post-circle step of the ORIGIN method usually takes a long time and also does not find many potential solutions. The Pre-circle step of both homotopy methods will find most solutions in an unpredictable amount of time.

Homotopy Methods	1st	2nd	3rd
NAIVE	(59,127;59,0)	(56,41;53,0)	(60,42;60,0)
ORIGIN	(58,133;49,5)	(60,44;46,97)	(60,132;41,53)

TABLE 4. Performance of Pre-circle and Post-circle steps for (4,2,1) case

2.5. **Limitations.**

The findings can be reproduced using the provided code. However a few limitations must be noted.

2.5.1. *Magnitude of θ_j.*

The magnitude of θ_j is assumed to be 1, but it is not possible to constrain it to be exactly equal to one using just polynomial systems. With that, the magnitude of θ_j may vary a little from 1.

2.5.2. *Constraints for positive dimensional solution sets.*

When solving systems with positive dimensional solution sets, the constraints chosen were of the form

$$\begin{aligned}
&c_1(G_1+\bar{G_1})+c_2(G_2+\bar{G_2})\\
&+c_3\sqrt{-1}(G_1-\bar{G_1})+c_4\sqrt{-1}(G_2-\bar{G_2})\\
&+c_5(z_1+\bar{z_1})+c_6(z_2+\bar{z_2})\\
&+c_7\sqrt{-1}(z_1-\bar{z_1})+c_8\sqrt{-1}(z_2-\bar{z_2})+c_9=0
\end{aligned}$$

where $c_i, i = 1, 2, ...9$ are real constants. This form of linear constraint was chosen because for real solutions this acts as a real constraint on each of the individual components of our non-rotation variables. For example, suppose $x = (x_1, x_2)$ is an element of $\mathbb{R}^2$, and let $y = (y_1, y_2)$ be x in isotropic coordinates, so

$$y = \begin{bmatrix} 1 & \sqrt{-1} \\ 1 & -\sqrt{-1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Then for some constant c in $\mathbb{R}$ we have

$$c(y_1+y_2) = c\begin{bmatrix}1 & 1\end{bmatrix}\begin{bmatrix} 1 & \sqrt{-1} \\ 1 & -\sqrt{-1} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 2cx_1$$

$$c\sqrt{-1}(y_1-y_2) = c\sqrt{-1}\begin{bmatrix}1 & -1\end{bmatrix}\begin{bmatrix} 1 & \sqrt{-1} \\ 1 & -\sqrt{-1} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = -2cx_2$$

In short, this form is analogous to a real constraint for each of the real coordinates of our variables.

When we choose affine constraints where the conjugate variables do not have the same coefficients we do not get the same number of solutions. We hypothesize that this may be due to the symmetric nature of the conjugates, but more insight is needed.

2.5.3. *Variability in the number of solutions.*

When solving systems, Bertini stores intermediate results as floating point numbers which may cause an undue approximation that gets carried over to the next part. This is highly alleviated with Bertini using adaptive precision but this is important enough to be noted since some computations may result in lower numbers of solutions than expected. Restarting the M2 system and re-running the code may help achieve the expected number of solutions in this case.

2.5.4. *Computation times.*

Computing all the solutions for large cases such as (1,5) takes a lot of computing power and time. Lack of consistency in the number of solutions coupled with long computation times has led us to omit a few cases from Table 1.

3. Stephenson-3 Six bar linkage

The Alt-Burmester synthesis problem can be re-framed for linkages other than the 4 bar linkage. In this section we will apply the procedures used for the four bar linkage to solve a similar synthesis problem for the Stephenson 3 six bar linkage. First we will give a brief description of the Stephenson 3 six bar linkage and outline the system of polynomial equations used to solve it. We will then give some basic findings for the degree and dimension of the problem.

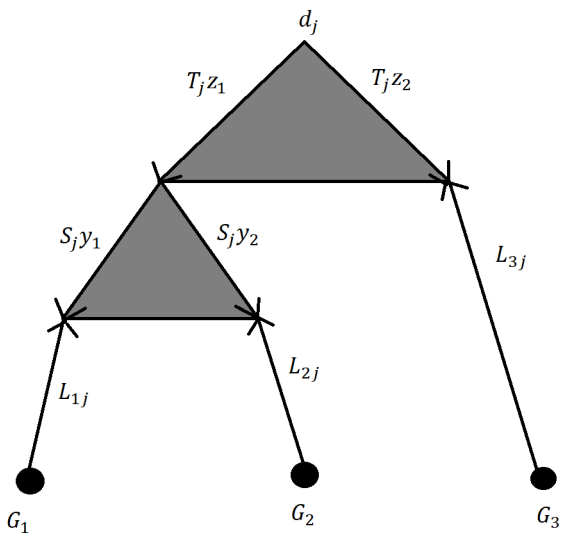


Figure 2. A Stephenson 3 six bar linkage

3.1. **Description and Outline of Polynomial System.**

The Stephenson 3 linkage consists of three fixed points, 3 proximal links, and two coupler triangles stacked on top of one another. An example can be seen in Figure 2. Another way of viewing the Stephenson 3 linkage is as a four bar linkage with an additional coupler triangle appended on top. As in the 4 bar case, the general purpose of the synthesis problem is to construct a six bar linkage whose coupler point, located at d_j in Figure 2, passes through a set of points which are specified as parameters.

To solve the six bar we follow a similar procedure and notation to the initial 4 bar case. First, we denote specified points by $(d_j, \bar{d}_j)$ in isotropic coordinates. The analogue of a pose is not as clear as that for a point because there are two coupler triangles in the Stephenson 3 six bar. To deal with this we assume that a pose specifies rotation angles for both coupler triangles, whereas a point does not specify a rotation angle for either.

Similarly as in the initial 4 bar construction, we denote fixed points by G_r for r from 1 to 3 and the proximal links for a point j by L_{rj}. For each coupler triangle, we define fixed vectors z_1, z_2 and y_1, y_2 which are analogous to z_1 and z_2 from the initial setup, and let $T_j, \bar{T}_j$ and $S_j, \bar{S}_j$ play the roles of the rotation matrices in isotropic coordinates for each triangle.

Our first set of constraints force the end of the paths from each fixed point to the coupler point to be located at the specified point d_j, i.e.

$$d_j = G_1 + L_{1j} - S_j y_1 - T_j z_1 \tag{7}$$

$$d_j = G_2 + L_{2j} - S_j y_2 - T_j z_1 \tag{8}$$

$$d_j = G_3 + L_{3j} - T_j z_2 \tag{9}$$

and

$$\bar{d}_j = \bar{G}_1 + \bar{L}_{1j} - \bar{S}_j \bar{y}_1 - \bar{T}_j \bar{z}_1 \tag{10}$$

$$\bar{d}_j = \bar{G}_2 + \bar{L}_{2j} - \bar{S}_j \bar{y}_2 - \bar{T}_j \bar{z}_1 \tag{11}$$

$$\bar{d}_j = \bar{G}_3 + \bar{L}_{3j} - \bar{T}_j \bar{z}_2 \tag{12}$$

for $j = 1, ..., N + M$. In addition, we impose necessary constraints on T_j, $\bar{T}_j$, S_j and $\bar{S}_j$ to be valid:

$$\begin{aligned} T_j \bar{T}_j &= 1 \\ S_j \bar{S}_j &= 1 \end{aligned}$$

for $j = M + 1, ..., M + N$. Finally, we constrain the proximal links to all have a fixed length, so

$$L_{rj} \bar{L}_{rj} = L_{r1} \bar{L}_{r1} \tag{13}$$

for $r = 1, 2, 3$ and $j = 2, ..., M + N$. As was done in the 4 bar case, to eliminate each L_{rj} and $\bar{L}_{rj}$ we solve equations 7, 8 and 9 for their respective L_{rj} and equations 10, 11 and 12 for their respective $\bar{L}_{rj}$ and substitute into the length constraints 13.

3.2. **Results.**

The results for several different attempts at solving the six bar are given in Table 5. We solved the 6 bar system using multiregeneration through Bertini, and the code

for setting up the system to solve with Bertini and with PHCpack are available on our GitHub.[2]

M	N	Dim	Deg	Dim 2	Deg 2
5	0	2	496		
5	1	1	1707		
4	2	3	18073	4	125

TABLE 5. Results for the Stephenson 3 Six Bar Linkage

The main difficulty in gathering results for the Stephenson 3 six bar was the amount of time needed to solve each system. For the case with 5 poses and 0 points the system solves rather quickly: less than one minute. However, increasing the number of points from N to $N+1$ increases the number of variables by 4: T_{N+1}, $\bar{T}_{N+1}$, S_{N+1}, and $\bar{S}_{N+1}$. This drastically increases the amount of time it takes to solve the system. For the case with 4 poses and 2 points, the computation took slightly less than a week.

Looking at Table 5, the first noticeable observation contrasting the 6 bar to the 4 bar is that the 4 pose 2 point system was multidimensional, whereas there were not multidimensional systems in the 4 bar case. However, even for this case a vast majority of solutions are in a space of dimension 3. BHMMW found that the dimension of the polynomial system for the four bar linkage follows the pattern

$$dim_4(M, N) = 10 - 2M - N.$$

In the case of the 6 bar we see a similar pattern beginning to form as

$$dim_6(M, N) = 17 - 3M - N$$

with the caveat that the (4,2) case is multidimensional.

3.3. **Real solutions of particular cases of the Stephenson-3 six bar linkage.**

To find the real solutions among all the complex solutions, Java is used to take the input text file of certain format of all complex solutions, and output the set of all real solutions. The code is written as a text file. Its name is "JAVAcode.txt" and can be found on GitHub.

The 4-pose-2-point case was chosen to find real solutions. There are more than 18000 complex solutions to the 4-pose-2-point case, but unfortunately, no real solutions were found among them. However, this code can be used in the future to filter big text files with complex solutions.

4. CONCLUSION

In this paper, we explored the solution spaces to Alt-Burmester synthesis problems for four and six bar linkages. For four bar linkages, we tried to replicate the results found in BHMMW [1] and looked at how the computation time changed as we used different methods to break up the angle constraint into linear subsystems. In addition,

[2]These files are generate-input-with-M2.m2 and sixbartake1.m2 respectively.

we extended the four bar approach to the Stephenson 3 six bar linkage, looking into the degree and dimension for the Stephenson 3 six bar system.

References

[1] Daniel A Brake, Jonathan D Hauenstein, Andrew P Murray, David H Myszka, and Charles W Wampler. The complete solution of Alt–Burmester synthesis problems for four-bar linkages. *Journal of Mechanisms and Robotics*, 8(4), 2016. 1, 2, 3, 7, 11

[2] Charles W Wampler. Isotropic coordinates, circularity and bezout numbers: planar kinematics from a new perspective. In *Proceedings of the 1996 ASME Design Engineering Technical Conference, Irvine, California August*, pages 18–22, 1996. 3