5-2018

# Gait Modeling Using Genetic Algorithm Optimized Four-Bar Linkages

Hazen James Hamather

# Gait Modeling Using Genetic Algorithm Optimized Four-Bar Linkages

A Thesis

Submitted to the Faculty

of

Rose-Hulman Institute of Technology

by

Hazen James Hamather

In Partial Fulfillment of the Requirements of the Degree

of

Master of Science in Mechanical Engineering

May 2018

**ROSE-HULMAN INSTITUTE OF TECHNOLOGY**

**Final Examination Report**

**Hazen Hamather** — Name

**Mechanical Engineering** — Graduate Major

Thesis Title **Gait Modeling Using Genetic Algorithm Optimized Four-Bar Linkages**

**DATE OF EXAM:** **April 25, 2018**

**EXAMINATION COMMITTEE:**

| | Thesis Advisory Committee | Department |
|---|---|---|
| Thesis Advisor: | **Bradley Burchett** | **ME** |
| | **David Purdy** | **ME** |
| | **Rebecca Bercich** | **ME** |
| | **Richard Stamper** | **EMGT** |
| | | |

**PASSED** X **FAILED** ____

## Abstract

Hamather, Hazen J.

M.S.M.E.

Rose-Hulman Institute of Technology

May 2018

Gait Modeling Using Genetic Algorithm Optimized Four-Bar Linkages

Thesis Advisor: Dr. Bradley T. Burchett

Important in diagnosing gait abnormalities and pathologies is knowing the position of the leg at various points throughout the gait cycle. This is currently done with motion capture technology but the demand for Inertial Measurement Unit (IMU) based navigation and position tracking has been on the rise. A required component of this alternative is a gait model that can accurately predict the position of points of interest.

In this thesis, a Kalman Filter is constructed using a contrived model to test if, given an accurate gait model, the filter can converge to an accurate and true position solution. Also presented is a Genetic Algorithm approach to dynamic system modeling. The dynamic system is made up of a four-bar linkage and has the ability to adapt to different gaits, both healthy and pathological.

Results for the Kalman Filter are illustrated through convergence plots, and final position solutions and results for the Genetic Algorithm are given by position solutions of the four-bar linkage. These results show that a genetic approach is robust and has application in gait analysis.

**TO**

To my parents, who showed me at a young age what it meant to work hard and be grateful for all that I have. Your constant sacrifice has led me to where I am.

To my wife, Elizabeth, who has walked this winding road with me. Through the highs and lows, you have been nothing but loving and encouraging from the start.

To my son, Hazen, who has brought so much joy into our lives over the last 14 months. May you look back on this someday and be reminded how much you are loved.

**Acknowledgments**

I would like to acknowledge the following people for the role they played in getting this idea off the ground and turning it into something tangible.

Dr. Burchett

Dr. Hobey Tam

Tanya Colonna

Dr. Bercich

Dr. Purdy

Dr. Stamper

Dr. Susan Smith

Ray Bland

Dr. Alfred Finch

## Table of Contents

## List of Figures

## List of Tables

# 1. INTRODUCTION

Humans have perfected the art of bipedal locomotion in its many forms. Competitions exist to test who can run the fastest, run the furthest, jump the highest, and jump the longest. Each day, the majority of humans are able to walk upright and unassisted on their two lower limbs. In order for this phenomenon to be possible, a sophisticated and finely tuned controller, as well as a complex and versatile skeletomuscular system, must exist. The finely tuned controller is obviously the brain, and the skeletomuscular system includes each bone and muscle of the leg distal to the hip. Other components aid in bipedalism, such as the spinal cord, nerves, tendons, and ligaments, but these elements are not included in this research.

## 1.1 Bipedal locomotive definitions, structure, and components

The system responsible for the ability to walk is complex for a good reason. For a single body to be capable of not only walking, but jogging, sprinting, jumping, and more, joints and other structures must be suited for multiple use cases. To add another level of sophistication, the human body has the capacity to alter its structure and composition when influenced by outside stimuli. An example of this occurs when athletes are injured. An athlete who experiences a broken ankle will be unable to play for at least 6 weeks while the injury heals. During the healing period, mobility is limited, and other structures such as the Medial Collateral Ligament (MCL) may become more fragile and susceptible to injury, unless properly reconditioned. Structure alteration also occurs in the other direction. Lifting weights and exercising encourages muscle growth due to self-induced stress.

Fundamental to understanding the gait cycle and joint dynamics, the anatomical axes describe movements of structures in a defined way. Figure 1 [1] displays these axes.

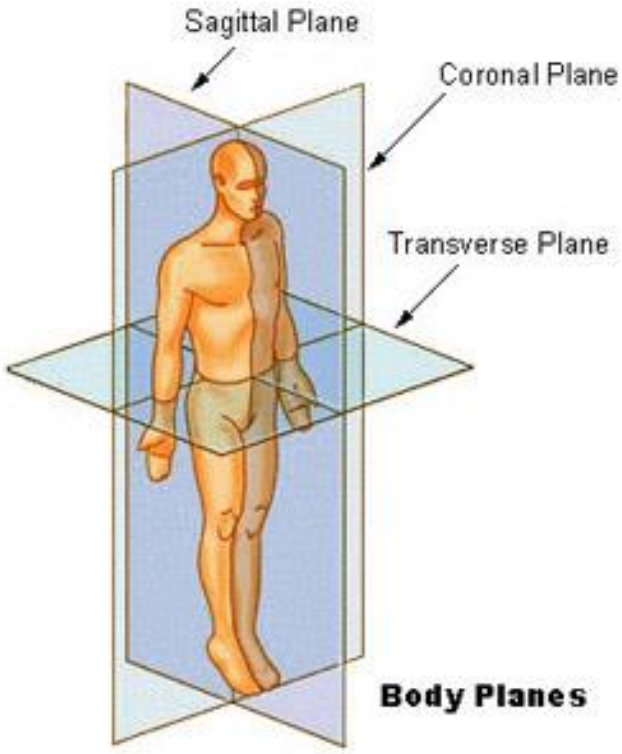


**Figure 1: Anatomical axes**

The plane cutting the body in half in the vertical direction is the transverse plane. Cranial (superior) to the transverse plane is the head, and caudal (inferior) to the same plane are the feet. The plane running through the center of the chest is the sagittal plane which cuts the body into left and right, referred to as lateral left and lateral right. The final plane is the frontal (coronal) plane, which cuts the body into front and back. Ventral (anterior) refers to the front, and dorsal (posterior) refers to the back. Figure 2 [2] better describes movements of structures and joints [3].

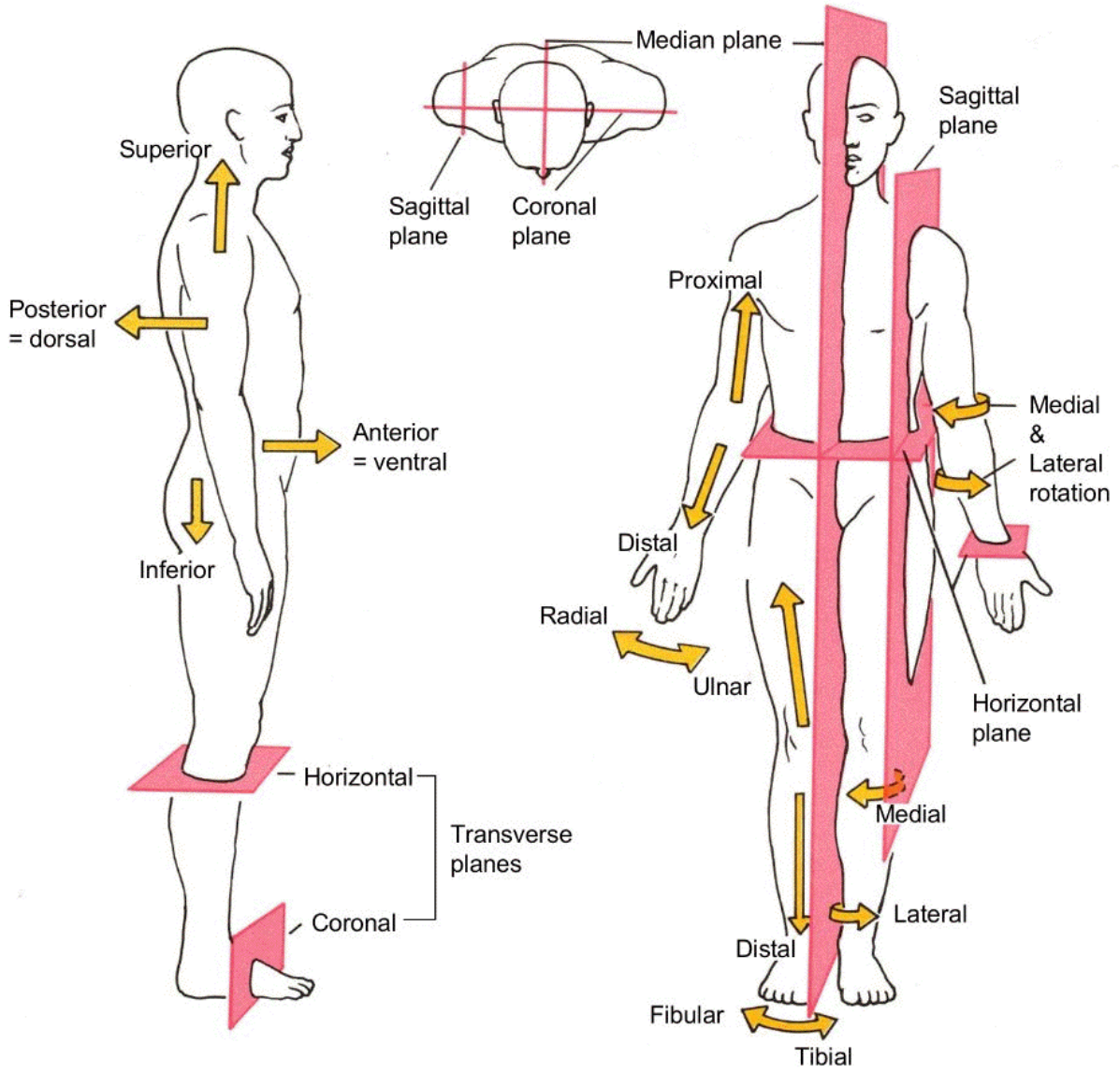


**Figure 2: Anatomical directional terms**

A healthy gait begins with the coxofemoral joint, otherwise known as the hip. This joint includes a socket called the acetabulum allowing the femur and its femoral head to join the torso [3]. A diagram of this is located in Figure 3 [4].

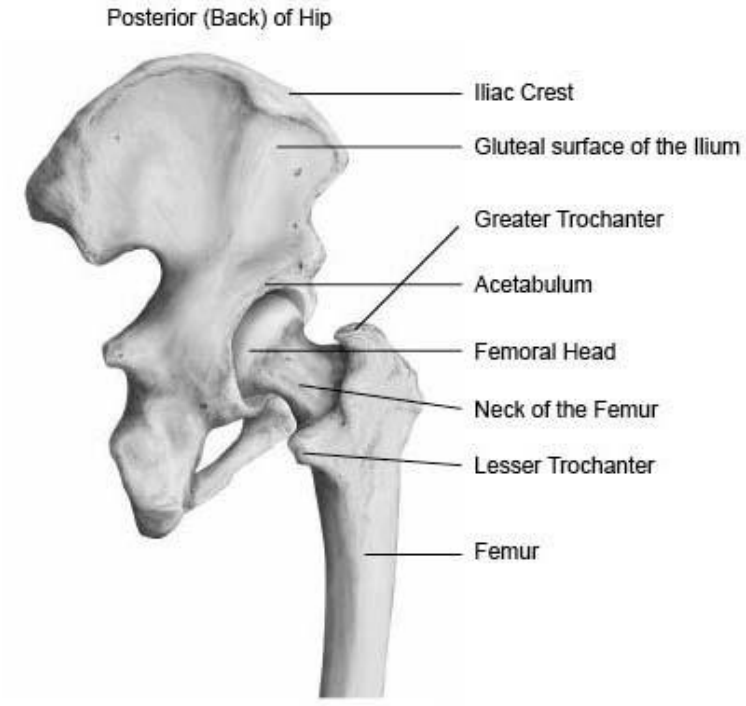


**Figure 3: Diagram of the coxofemoral joint including pelvis and femur**

Throughout the gait cycle, the coxofemoral joint will experience a range of motion from approximately 25° flexion to 20° extension [5]. These values may vary slightly but should remain consistent for healthy gait patterns.

Next is the knee. Seen in Figure 4 [6], this joint is actually comprised up of two symbiotic joints, the knee has a constantly changing center of rotation, resulting in an architecture that has proven itself difficult to model accurately [3]. The two joints making up the knee are the tibiofemoral joint and the patellofemoral joint. The tibiofemoral joint functions as the joint between the distal femur and proximal tibia while the patellofemoral joint serves to connect the posterior patella and the femur [3].


**Figure 4: Diagram of the knee joint showing entering femur and exiting tibia and fibula**

The knee serves to provide mobility yet stability between the femur, tibia, and fibula. Aiding the two joints are the four ligaments of the knee commonly known as the LCL, ACL, PCL, and MCL. Good range of motion for this joint is approximately 5° flexion through 60° flexion [5]. Note that healthy knees should not exceed 5° flexion due to the risk of hyperextension, resulting in injury. The aforementioned ligaments aid in restraining the knee to safe ranges of motion.

The connector between knee and ankle is the tibia and fibula. The models used in this research assume there is a single link, the tibia, joining the knee and ankle together. Both bones have fixed ends, allowing for the simplification. If the position of one is known, the position of the other can be calculated using basic anatomy.

The ankle and foot consists of 28 bones combining to create 25 joints which can be seen in Figure 5 [7]. With this complexity, the ankle is capable of dorsiflexion and plantarflexion in the sagittal plane, inversion and eversion in the frontal plane, and abduction and adduction in the transverse plane.

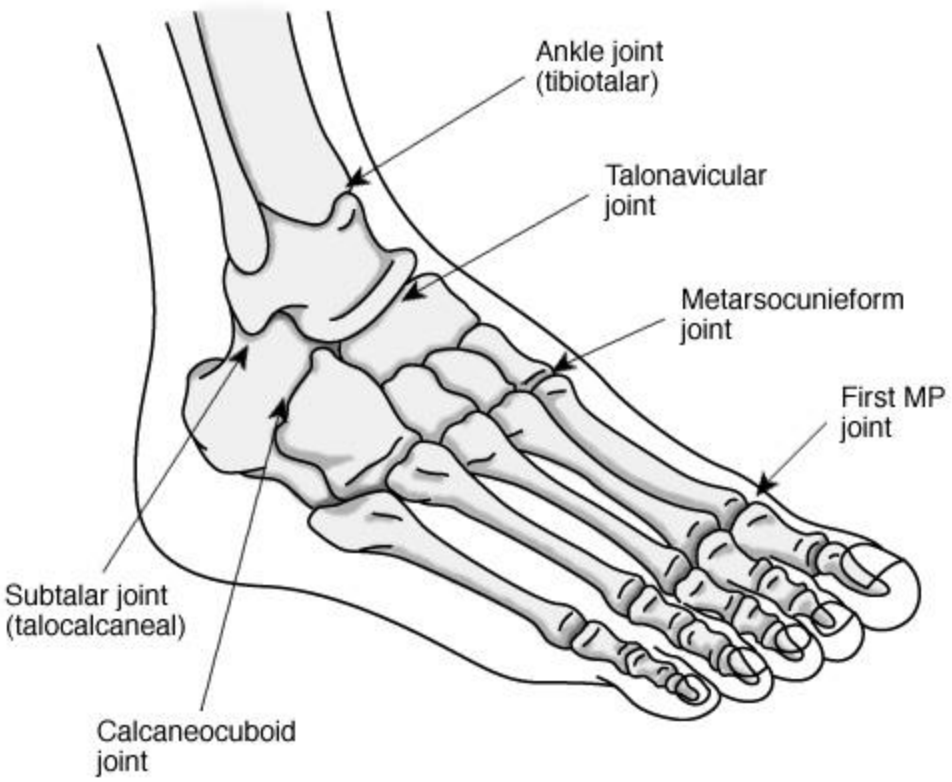


**Figure 5: Simple ankle and foot complex**

The major joints in the ankle are the ankle joint itself, the tibiotalar, the subtalar joint, and the talonavicular joint. These joints provide stability and agility for the gait cycle but add a modeling complexity that is not necessary for a two-dimensional model. Therefore, the ankle is modeled as a pin joint.

## 1.2 Gait Analysis and Dynamics

Similar to each person having a different fingerprint, individual gait is unique, but, in most cases, the same characteristics identify healthiness of gait. Those characteristics are range of motion, torque demand, and muscle action. For the purposes of this research, all of the focus has been on range of motion, but it is understood that torque demand and muscle action give large amounts of insight that is being overlooked.

Before delving into analysis and dynamics, the following explanation of the gait cycle assumes the right leg is the subject of analysis. For reference, refer to Figure 6 [8] for the following paragraphs.

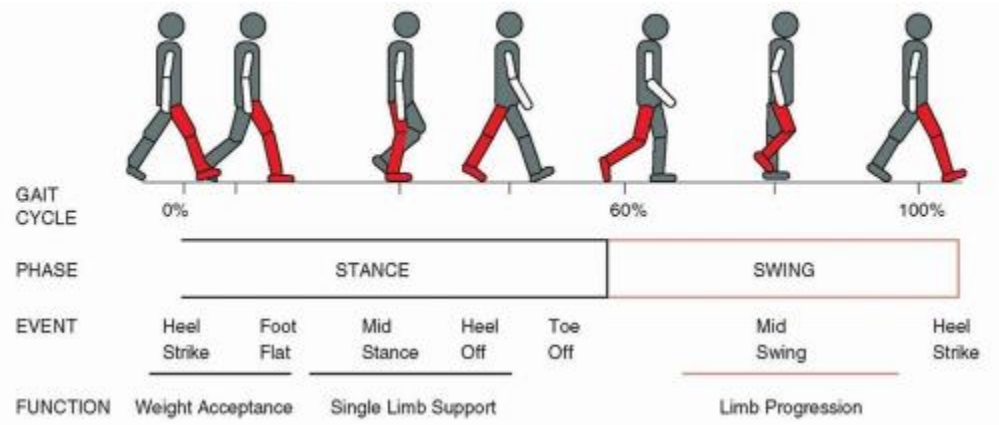


**Figure 6: Diagram of the gait cycle with specific events labeled**

The cycle begins with the initial contact. Initial contact occurs when the right heel makes its first contact with the ground. The next event to take place is the load response followed by heel off. At this instant, the right heel begins to rise off the ground, and the opposite (left) leg becomes the leading leg. After the leading leg makes contact and accepts the weight, the right leg experiences toe off, beginning its terminal swing to end at the original starting point, initial contact of the right heel.

At the opening of the cycle, the subject is in the stance phase, labeled as such to signify the right leg is bearing the weight of the body. The swing phase exists from right foot toe off to initial contact for the next gait cycle. Appropriately named, the swing phase earned its name from the observation that the right leg is no longer weight bearing but rather swinging through the hips' range of motion to return to a stance phase position [5]. To analyze the left leg, simply 1) offset the beginning of the gait cycle such that the opposite initial contact becomes initial contact or 2) say initial contact is the left leg instead of the right leg. Much like setting a coordinate system, defining which foot signals initial contact is semantics.

## 1.3 Introduction to this work

Researchers have heavily studied the healthy gait cycle, but recently scientists, doctors, and engineers have been devoting more time and energy to gait pathologies. Defined to be

diseases and abnormalities in gait, these pathologies take many different forms. A few simple gait pathologies include internal rotation of the hip, extension thrust of the knee, foot slap of the ankle, and clawed toes [5]. With so many pathologies to study, this research focuses on two common gait abnormalities: foot drag and drop foot.

Patients who have suffered a stroke will often exhibit symptoms of foot drag. Strokes have a tendency to paralyze (or at least severely inhibit mobility in) half of the patient's body along the sagittal plane, and the paralyzed side will typically show signs of a dragging foot. Referring to Figure 6, a dragging foot involves sustained contact of the right foot during the swing phase [5]. Dragging is not just hard on shoes and socks, it presents a fall hazard to the geriatric community already plagued with decreasing bone density.

Patients diagnosed with Parkinson's disease commonly show signs of another gait pathology known as drop foot or foot-flat contact. Occurring during the loading response period, foot-flat contact, in essence, removes the first 10% of the gait cycle by substituting initial contact of the heel for initial contact of the entire foot. Weight acceptance occurs in an instant resulting in momentary instability of the subject [5].

BlackTop Labs LLC seeks to be the future standard of care in gait diagnostics and rehabilitation. With this in mind, their focus is on wearable technologies that can give specific insights on patients who suffer from either foot drag or drop foot. BlackTop Labs LLC has collaborated with Hamather to make this happen. The wearable technologies are inertial measurement units (IMU) capable of measure angular rates in the three different planes and these sensors have Gaussian noise, allowing for the use of a Kalman Filter when a correct gait model is present. The results of this research will demonstrate that it is possible to create two-dimensional, patient-specific gait models by use of a Genetic Algorithm to be implemented within a Kalman Filter. While not explicitly focusing on Kalman filtering, this research will cover basic filtering techniques and illustrate the power and versatility of the Genetic Algorithm.

## 1.4 Current technologies

In its current state, gait analysis technologies leverage either a doctor with specialized training or a motion capture technology. Not to diminish the ability to spot gait pathologies, but doctors and physical therapists are great at telling a patient what is wrong, not exactly the severity of the problem. The benefit of this procedure, since it involves just a doctor, is

affordability. Most insurance companies will pay for this treatment, but in exchange for affordability, the patient loses a high degree of accuracy.

A process not directly involving doctors is motion capture technology as seen in Figure 7. Cameras surround the patient who wears reflective spheres. These cameras both emit and detect infrared light not seen by the human eye. The software is then able to detect the reflective spheres and piece together a three-dimensional picture of the gait cycle for analysis. These software and camera packages are accurate to within a few millimeters, but they are expensive and not covered by most insurance companies, resulting in a new market opportunity for BlackTop Labs. This is where the company seeks to make a beachhead in the medical device market. If motion capture can be replaced with a cheaper option that offers the same accuracy, then BlackTop labs will become the standard of diagnosis and standard of care in gait rehabilitation and ultimately improve patient outcomes.

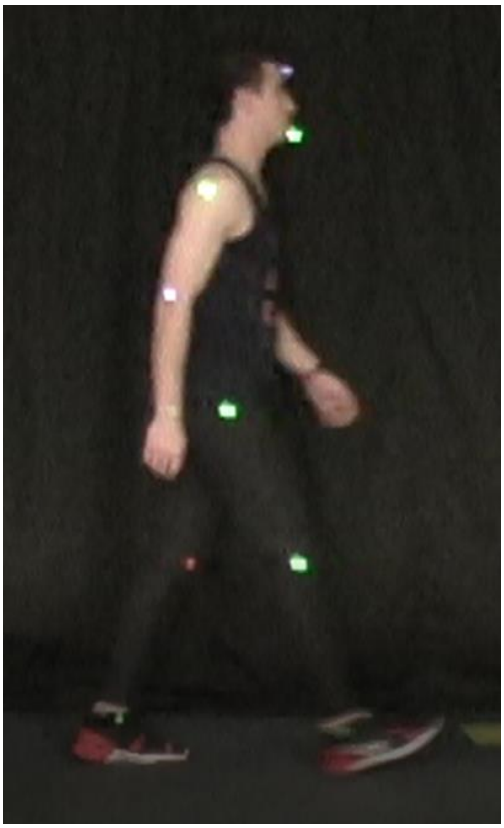

**Figure 7: Motion capture with reflective spheres at initial contact**

## 2. ACQUIRING GAIT DATA

This section deals with gathering reasonable data for an Extended Kalman Filter (EKF) and a Genetic Algorithm (GA). A model was created in order to generate known data that could then be corrupted to test the capabilities and sophistication of both an EKF and a GA.

### 2.1 Simulated gait data via pull toy model

The first model stemmed from an old pull behind toy. Originally built by the Amish and used by children, the legs move up, down, and sideways, giving the look of a trotting horse. The idea for this model comes from Figure 8.

**Figure 8: Pull toy simulating a double jointed model**

Figure 9 gives the two-link equivalent system used in the kinematics derivation.

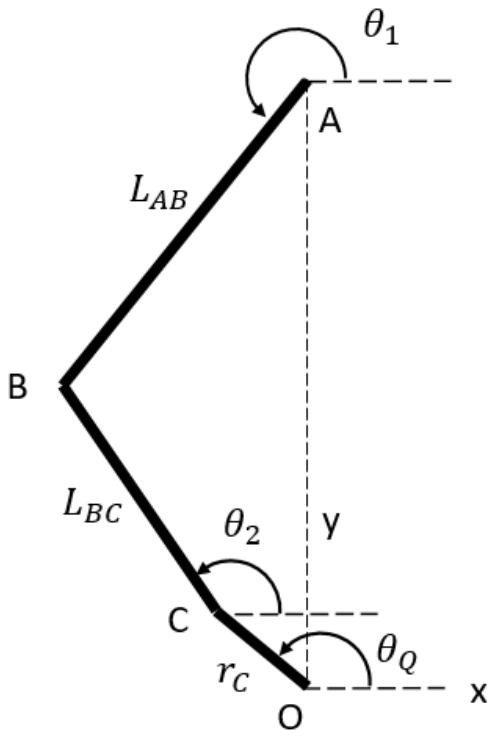


**Figure 9: Two link dynamic model of femur and tibia**

The model begins with a few assumptions. The first assumption is constant velocity. The horse will move forward at a constant velocity of

$$\mathrm{v} = 1\frac{\mathrm{m}}{\mathrm{s}} \tag{2.1}$$

to make the calculations minutely simpler. The length of the lower link connecting to the input crank to link $\mathrm{L_{BC}}$ is

$$\mathrm{r_C} = 0.17\ \mathrm{m}. \tag{2.2}$$

Calculating the angular velocity of the wheel is simple,

$$\omega_{\mathrm{Q}} = \frac{\mathrm{v}}{\mathrm{r_C}} \tag{2.3}$$

and the length of the first link is chosen to be similar to that of the average human femur

$$\mathrm{L_{AB}} = 0.48\ \mathrm{m}. \tag{2.4}$$

The second link in the model has a length similar to that of the average tibia

$$L_{BC} = 0.36 \text{ m}. \tag{2.5}$$

With a designed system, it is now possible to perform the kinematic analysis required to gather angular rate data. With constant $\omega_Q$, it is possible to calculate the angle at which the wheel has rotated at each instant in time

$$\theta_Q(t) = \omega t. \tag{2.6}$$

Using $\theta_Q(t)$, the position of point $C$ can be calculated for all time t. From here on, $\theta_Q$ is used instead of $\theta_Q(t)$ for simplicity.

$$r_{C_x} = r_C \cos(\theta_Q) \tag{2.7}$$

$$r_{C_y} = r_C \sin(\theta_Q) \tag{2.8}$$

as well as the velocity

$$v_{C_x} = -\omega_Q r_C \sin(\theta_Q) \tag{2.9}$$

$$v_{C_y} = \omega_Q r_C \cos(\theta_Q). \tag{2.10}$$

Initial conditions are

$$\theta_{1_0} = 200° \tag{2.11}$$

$$\theta_{2_0} = 110° \tag{2.12}$$

and the pivot acting as the hip joint for the toy is assumed to have no vertical displacement placing

$$r_{A_x} = r_{C_x} + L_{BC} \cos(\theta_{2_0}) + L_{AB} \cos(\theta_{1_0} - \pi) \tag{2.13}$$

$$r_{A_y} = r_{C_y} + L_{BC} \sin(\theta_{2_0}) + L_{AB} \sin(\theta_{1_0} - \pi)) \tag{2.14}$$

for all time t.

With no vertical displacement, the velocity of A is

$$v_{A_X} = \omega_Q r_Q \tag{2.15}$$

$$v_{A_y} = 0. \tag{2.16}$$

The position and velocity of link joint B is computed from both directions A and C. This places a constraint on the solution of B. Beginning with the definition of position

$$\overrightarrow{r_B} = \vec{r}_A + \vec{r}_{B_A} \tag{2.17}$$

the position of B is represented as

$$\vec{r}_B = \vec{r}_A + L_{AB}(\sin(\theta_1)i + \cos(\theta_1)\,j). \tag{2.18}$$

Breaking this vector into its components gives

$$r_{B_x} = r_{A_x} + L_{AB}cos(\theta_1) \tag{2.19}$$

$$r_{B_y} = r_{A_y} + L_{AB}\sin(\theta_1). \tag{2.20}$$

From the other direction, the position of B is

$$\vec{r}_B = \vec{r}_C + \vec{r}_{B_C} \tag{2.21}$$

yielding

$$\vec{r}_B = \vec{r}_C + L_{BC}(cos(\theta_2)\,\hat{\imath} + sin(\theta_2)\,\hat{\jmath}). \tag{2.22}$$

Simplifying this system into its components gives

$$r_{B_x} = r_{C_x} + L_{AB}\cos(\theta_2) \tag{2.23}$$

$$r_{B_y} = r_{C_y} + L_{AB}\sin(\theta_2). \tag{2.24}$$

The velocity components of B starting with point C gives

$$\vec{v}_B = \vec{v}_C + \vec{\omega}_{BC} \times \vec{r}_{B_C}. \tag{2.25}$$

Expanding (2.26) shows

$$\vec{v}_B = \vec{v}_C + \omega_{BC}\hat{k} \times L_{BC}(\cos(\theta_2)\,\hat{\imath} + \sin(\theta_2)\,\hat{\jmath}) \tag{2.26}$$

and breaking this into its components gives

$$v_{B_x} = -\omega_Q r_C \sin(\theta_Q) - \omega_{BC}L_{BC}\sin(\theta_2) \tag{2.27}$$

$$v_{B_y} = \omega_Q r_C \cos(\theta_Q) + \omega_{BC}L_{BC}\cos(\theta_2). \tag{2.28}$$

Representing the velocity of B from point A yields

$$\vec{v}_B = \vec{v}_A + \vec{\omega}_{AB} \times \vec{r}_{B_A}. \tag{2.29}$$

Expanding the cross product gives

$$\vec{v}_B = \vec{v}_A + \omega_{AB}\hat{k} \times L_{AB}(\cos(\theta_1)\,\hat{\imath} + \sin(\theta_1)\,\hat{\jmath}). \tag{2.30}$$

Finally, simplifying (2.31) into the principle components shows that

$$v_{B_x} = -\omega_{AB} L_{AB} \sin(\theta_1) \tag{2.31}$$

$$v_{B_y} = \omega_{AB} L_{AB} \cos(\theta_1). \tag{2.32}$$

The state vector of the system [9] turns out to be

$$x = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \omega_{AB} \\ \omega_{BC} \end{bmatrix} \tag{2.33}$$

and here arises a problem. After taking the derivative of the state vector, angular acceleration terms appear. The current kinematics equations provide no way of estimating these parameters, as angular accelerations never appear in any of the equations.

$$\dot{x} = \begin{bmatrix} \omega_{AB} \\ \omega_{BC} \\ \alpha_{AB} \\ \alpha_{BC} \end{bmatrix} \tag{2.34}$$

If angular accelerations are assumed to be zero, an observability issue arises. This issue was discovered in the first system design. In the linear case, the observability matrix is created with combinations of the state and measurement vectors.

$$S = \begin{bmatrix} C \\ CA \\ CA^2 \\ . \\ . \\ . \\ CA^{n-1} \end{bmatrix} \tag{2.35}$$

The rank of S must be equal to the number of states in the system, where C and A are output and state matrices, respectively. When angular accelerations are assumed to be zero, the

rank becomes two resulting in an unobservable system which derails the Kalman filter. This was a sticking point early in the research that proved to be a valuable learning experience.

The same process used for velocity kinematics governs the angular acceleration kinematics, and the finer details are omitted here. Angular acceleration kinematics say

$$\overrightarrow{a_B} = \vec{a}_A + \vec{\alpha}_{AB} \times \vec{r}_{B_A} + \vec{\omega}_{AB} \times (\vec{\omega}_{AB} \times \vec{r}_{B_A}) \tag{2.36}$$

and

$$\overrightarrow{a_B} = \vec{a}_C + \vec{\alpha}_{BC} \times \vec{r}_{B_C} + \vec{\omega}_{BC} \times (\vec{\omega}_{BC} \times \vec{r}_{B_C}). \tag{2.37}$$

After expanding each term and taking the correct cross products, breaking everything down into x and y components yields

$$-\alpha_{AB} L_{AB} \sin(\theta_1) - \omega_{AB}^2 L_{AB} \cos(\theta_1) = -\omega_Q^2 r_C \cos(\theta_Q) - \alpha_{BC} L_{BC} \sin(\theta_2) - \omega_{BC}^2 L_{BC} \cos(\theta_2) \tag{2.38}$$

in the x-direction and

$$\alpha_{AB} L_{AB} \cos(\theta_1) - \omega_{AB}^2 L_{AB} \sin(\theta_1) = -\omega_Q^2 r_C \sin(\theta_Q) + \alpha_{BC} L_{BC} \cos(\theta_2) - \omega_{BC}^2 L_{BC} \sin(\theta_2) \tag{2.39}$$

in the y-direction. The system is now observable for this setup and ready for Kalman Filter implementation.

## 2.2 Four-bar linkage model

Instead of using a two-link system with a constant angular rate to drive the model forward, a more realistic model is desired. Simple pedestrian observation shows that ankle path does not look like the circle from the two-link model seen in Figure 10. But, this does not negate the previous model. Using contrived data for the EKF will tell whether a Kalman Filter is suitable for this type of problem.

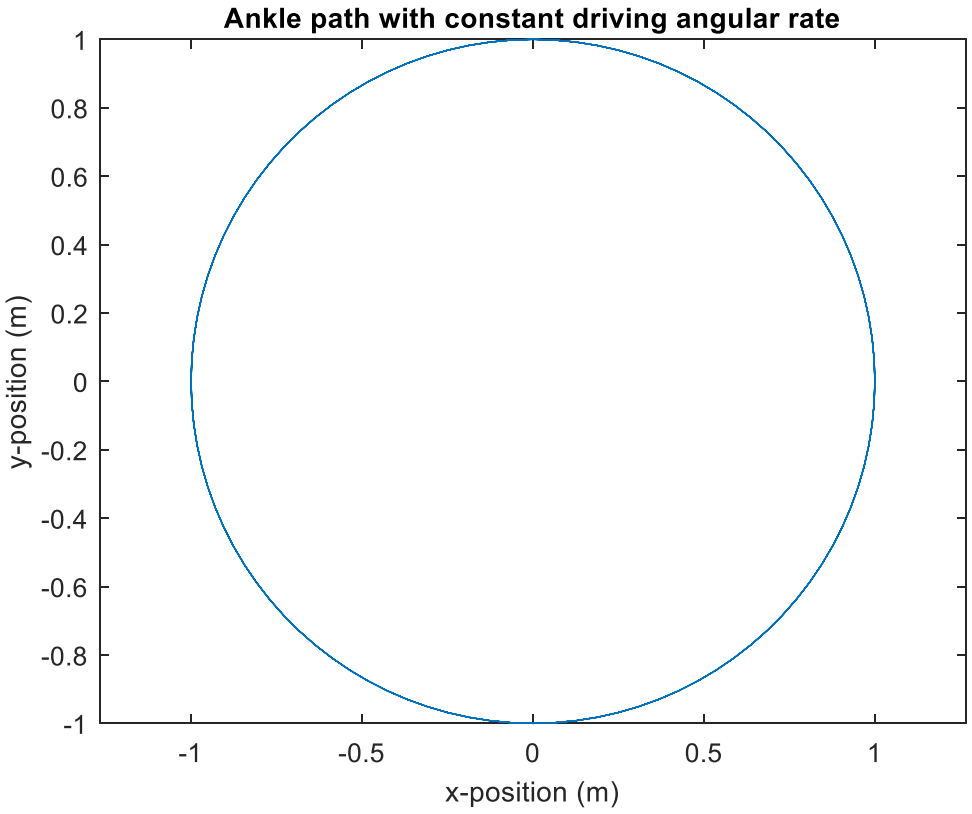


**Figure 10: Ankle path from two-link kinematic model with constant driving angular rate**

Using a four-bar linkage is the way to accomplish a more realistic representation of driving healthy gait. A four-bar linkage is precisely what it sounds like, but this gait model will use a special type of four-bar system, the Hoeken linkage, named after Karl Hoeken.

A traditional four-bar linkage system uses four links with one of the links acting as the crank. Figure 11 shows a basic four-bar system with link $a$ acting as the crank where all links are connected with a pin joint. Varying the length of the other links produces different system behaviors. Small changes in length lengths can have drastic effects and Karl Hoeken discovered a special case that has interesting characteristics.

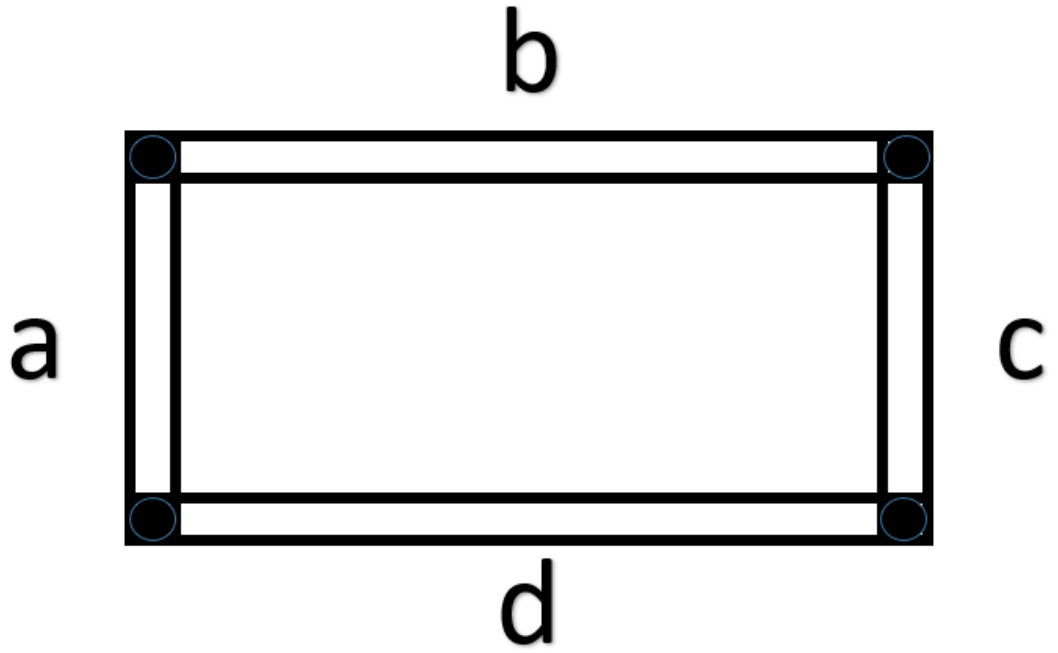


**Figure 11: Four-bar linkage**

The Hoeken system also has labeled joint connections such as those found in Figure 12.

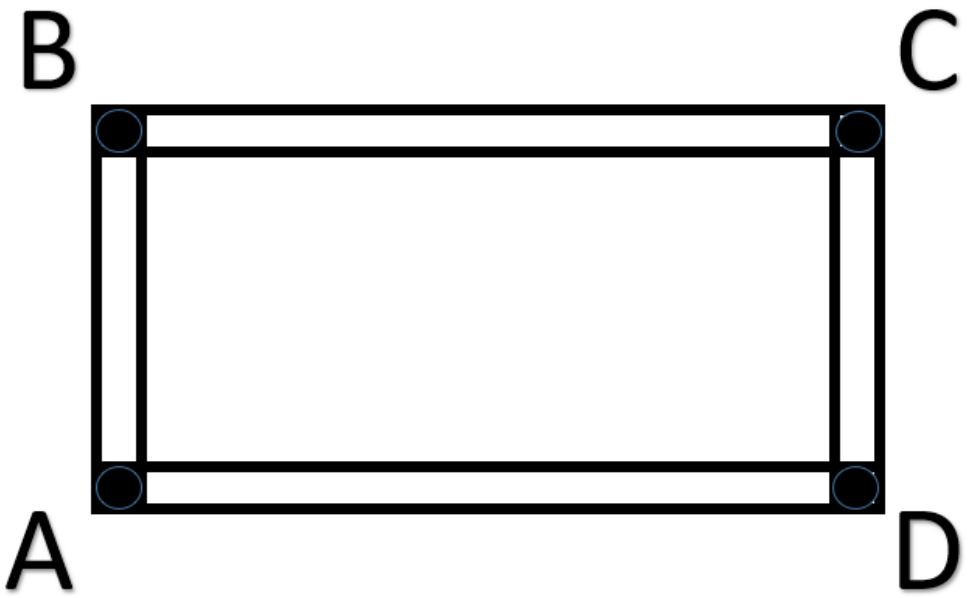


**Figure 12: Four-bar linkage revolute joints**

The Hoeken system uses link $a$ as the driving link, and it must be the shortest link. For a given value of a, other link lengths are as follows:

$$b = 2.5a \tag{2.41}$$

$$c = 2.5a \tag{2.42}$$

$$d = 2a. \tag{2.43}$$

These dimensions yield the path seen in Figure 13 when $a = 2$.

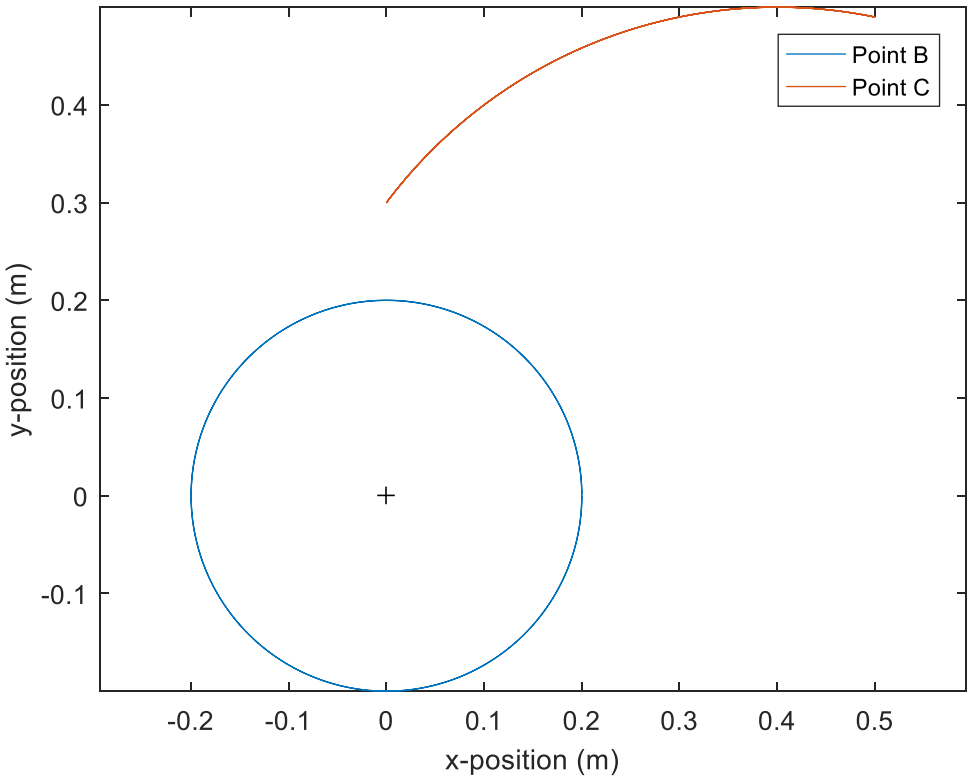


**Figure 13: Hoeken link revolute joint paths**

This may not resemble any form of recognizable gait, but something happens when the length of b is doubled and referred to as e. Doing this results in a system that looks like Figure 14.

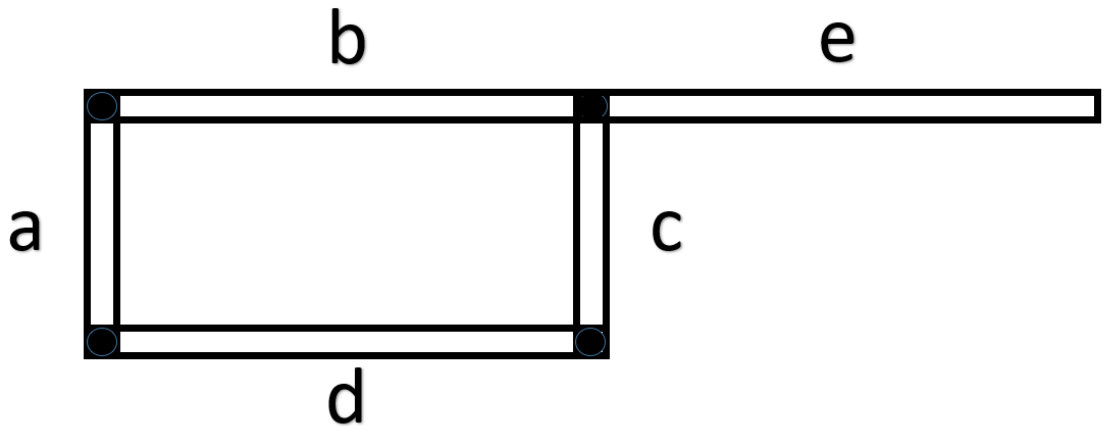


**Figure 14: Complete Hoeken system**

The connection between link b and c is still allowed to pivot, but the path made by the end of link e is very interesting. Seeing a snapshot of the entire system with the path of the end effector gives Figure 15.

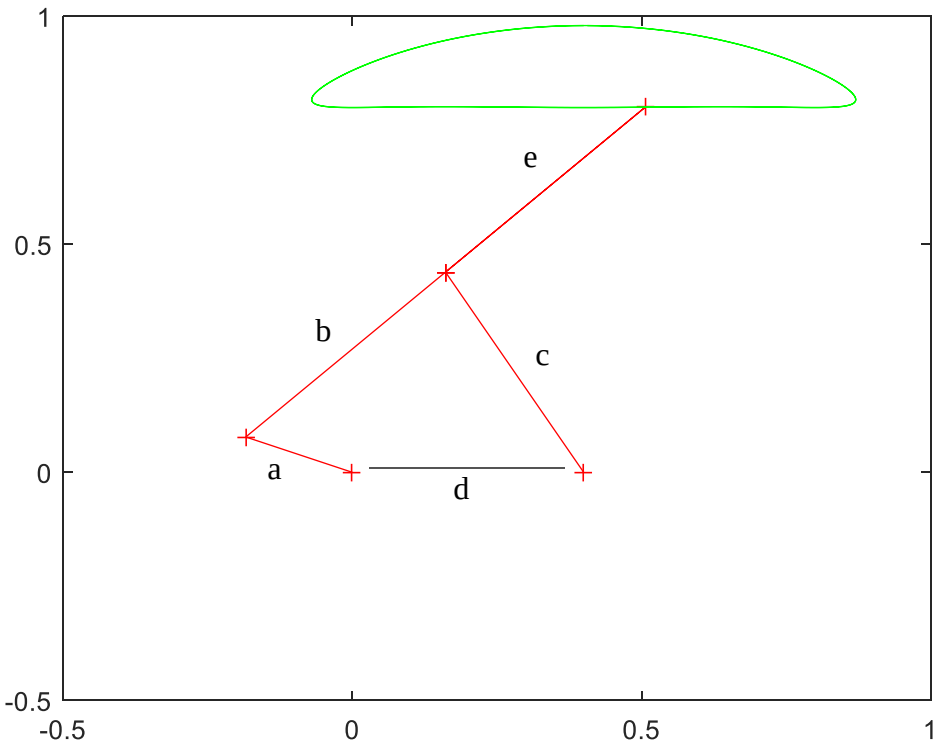


**Figure 15: Path of end effector e with Hoeken link system**

Although this system may not exactly model healthy gait ankle path, this system properly models heel-strike through toe-off, which is more accurate than the previous two-link model.

Because each person's gait is unique, a Genetic Algorithm (GA) is explored here for parameter estimation. Allowing a GA to optimize four-bar parameters to model human gait is a novel solution. Unique gait models tailored to patients will give the Kalman filter a good nonlinear model and improve its accuracy.

It may seem as if the two-link model is being thrown away, but the two-link model dynamics are still relevant. The femur and tibia are still modeled as single links connected by a pin, where the only difference is the way the model is driven. The end effector of link *e* acts as the ankle joint with a known path.

## 2.3 Motion capture with Qualisys and strap down IMU

Real data was captured using Qualisys software and accompanying cameras. Three stationary cameras point at a treadmill where the test subject would be performing different gait

cycles. These cycles were healthy gait, drop foot, and drag foot, and the system setup can be seen in Figure 16.


**Figure 16: Motion capture equipment setup for real data collection**

The data collection process yielded a frame-by-frame animation of each gait cycle where a single frame looks like Figure 17.

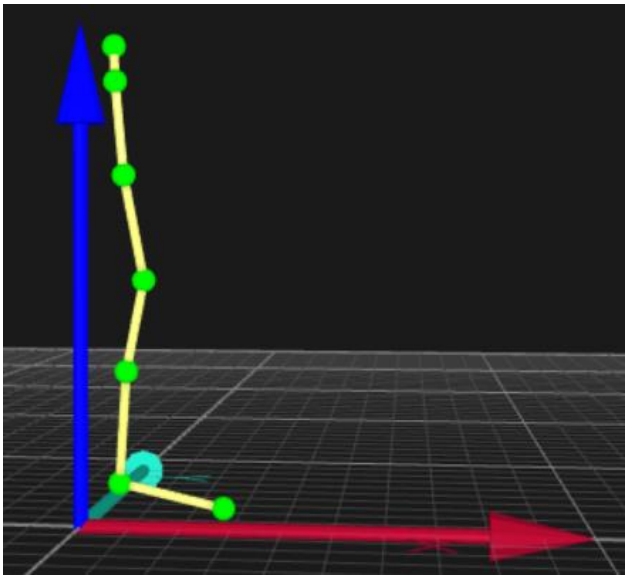

**Figure 17: Single frame of Qualisys data**

Although data were collected in three dimensions, it was projected into two dimensions (XZ plane) for simplicity, as the majority of gait is two dimensional. After some data smoothing, the angular rates of the femur and tibia can be found in Figure 18, Figure 19, and Figure 20 below. Note, the individual demonstrating the gait pathologies in this research does not suffer from any kind of gait pathology. An individual with a healthy gait simulated drop foot and drag foot.

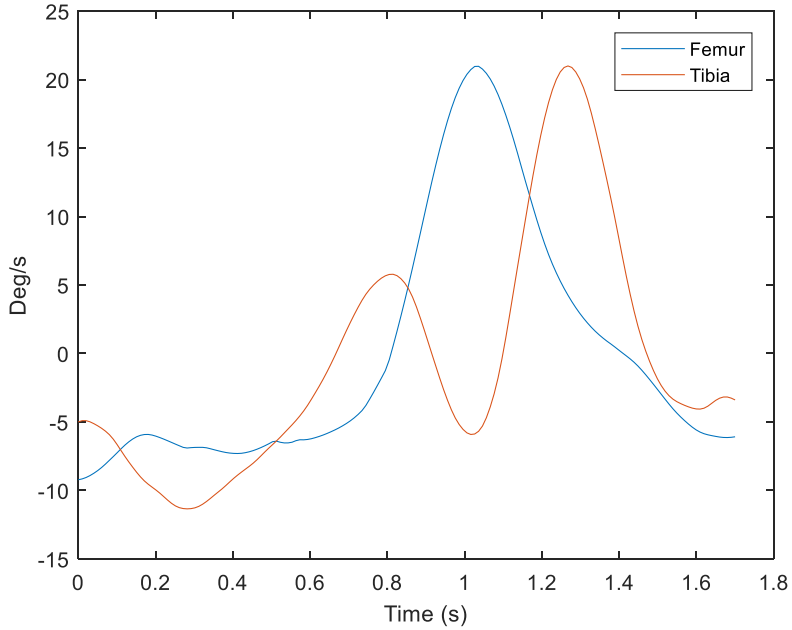


**Figure 18: Healthy gait femur and tibia angular rates from motion capture**

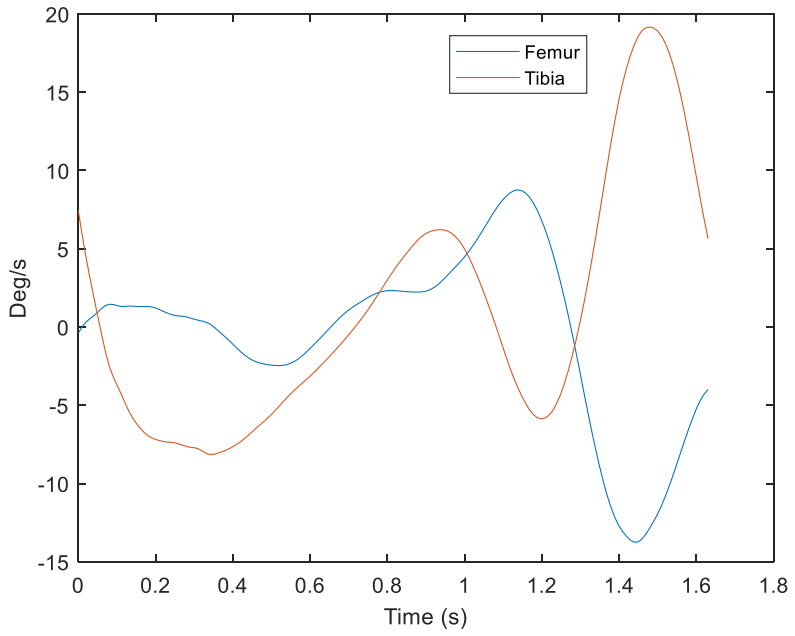


**Figure 19: Drop foot femur and tibia angular rates from motion capture**

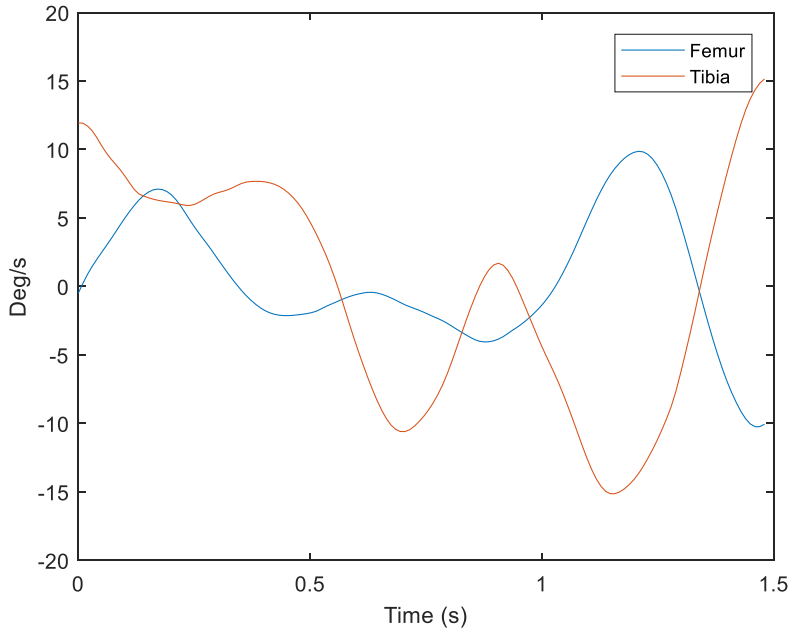


**Figure 20: Drag foot femur and tibia angular rates from motion capture**

Each of these gait cycles were performed at 1.5mph on a treadmill to achieve consistency. The position data for multiple points of interest was obtained, and this position data will be important later. The ankle path for each type of gait can be found in Figure 21. Although

each set of data does not line up and since all optimizations on this data were performed using position, it is not necessary to be aligned.

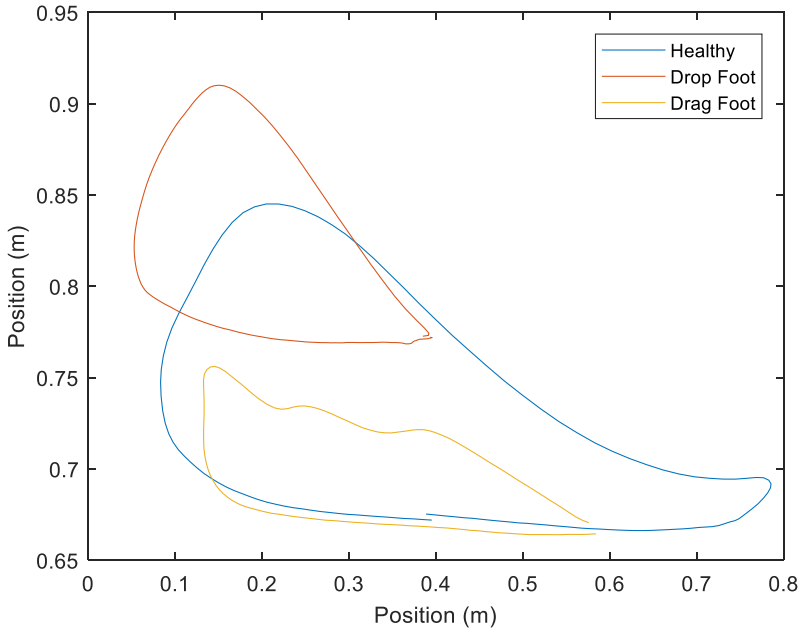


**Figure 21: Ankle path for one gait cycle for all gaits**

As previously mentioned, these ankle paths do not closely follow the assumption of a Hoeken path. The ankle path for all shapes have a tear-drop shape which is not indicative of a Hoeken system. But after reviewing motion capture data, it was discovered that the mid-tibia path promises a more traditional Hoeken path. For this reason, ankle path and mid-tibia path were used in optimizations. Figure 22 shows the mid-tibia path for the three different gait cycles. Note, both ankle and mid-tibia paths were considered in this research although the mid-tibia path showed more promise at the initial observation.

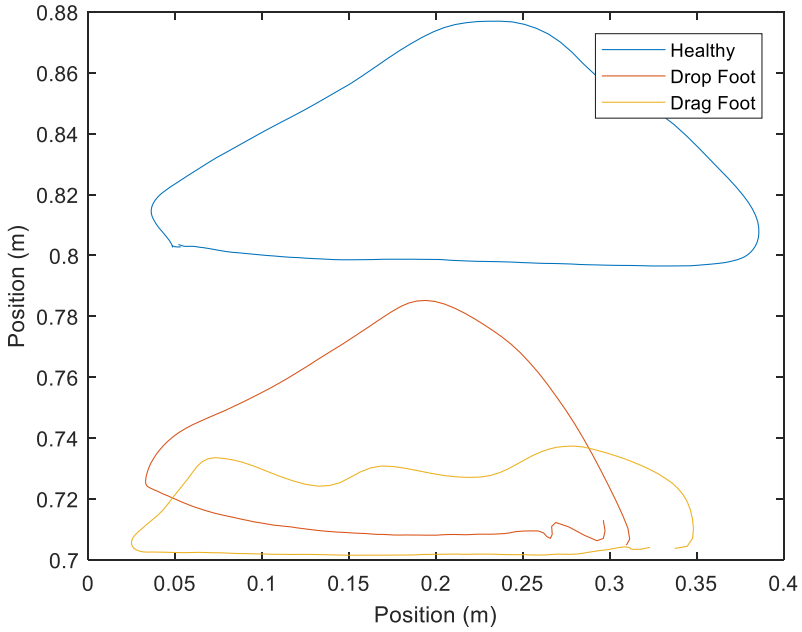


**Figure 22: Mid-tibia path for one gait cycle for all gaits**

# 3. GAIT KINEMATICS

Multiple iterations of kinematic approaches were necessary for the formulation of the working model. Gait kinematics are unique to individuals, creating a difficult problem to solve with a general solution. This system will have known kinematics, making it easy to model, but will be nonlinear, allowing for the use of an Extended Kalman Filter.

## 3.1 Model to state space

The Kalman Filter both relies on and requires a system dynamics model to perform correctly. This system model is represented in the form of state space equations and the following section deals with their derivation.

As mentioned in section 2.1, four components comprise the state vector for the two-link model. Listed again for convenience is the vector from (2.34).

$$x = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \omega_{AB} \\ \omega_{BC} \end{bmatrix} \tag{3.1}$$

To use state space convection properly, $\dot{x}$ is required.

$$\dot{x} = \begin{bmatrix} \omega_{AB} \\ \omega_{BC} \\ \alpha_{AB} \\ \alpha_{BC} \end{bmatrix} \tag{3.2}$$

Knowing that each of the equations that make up the derivative of the state vector are nonlinear, a nonlinear filter is required.

Angular rates can be found with a simple matrix inversion of the form

$$\begin{Bmatrix} \omega_{AB} \\ \omega_{BC} \end{Bmatrix} = \begin{bmatrix} -L_{AB}\sin(\theta_1) & L_{BC}\sin(\theta_2) \\ L_{AB}\cos(\theta_1) & -L_{BC}\cos(\theta_2) \end{bmatrix}^{-1} \begin{Bmatrix} -\omega_Q r_C \sin(\theta_Q) \\ \omega_Q r_C \cos(\theta_Q) \end{Bmatrix} \tag{3.3}$$

where, for readability, the matrix to be inverted is herein referred to as $\Xi$, and the right side vector is G.

$$\Xi = \begin{bmatrix} -L_{AB}\sin(\theta_1) & L_{BC}\sin(\theta_2) \\ L_{AB}\cos(\theta_1) & -L_{BC}\cos(\theta_2) \end{bmatrix} \tag{3.4}$$

In the same way, the angular accelerations can be written in the form

$$\begin{Bmatrix}\alpha_{AB}\\ \alpha_{BC}\end{Bmatrix} = \begin{bmatrix}-L_{AB}\sin(\theta_1) & L_{BC}\sin(\theta_2)\\ L_{AB}\cos(\theta_1) & -L_{BC}\cos(\theta_2)\end{bmatrix}^{-1} \begin{Bmatrix}\omega_{AB}^2 L_{AB}\cos(\theta_1) - \omega_{BC}^2 L_{BC}\cos(\theta_2) - \omega_Q^2 r_C\cos(\theta_Q)\\ \omega_{AB}^2 L_{AB}\sin(\theta_1) - \omega_{BC}^2 L_{BC}\sin(\theta_2) - \omega_Q^2 r_C\sin(\theta_Q)\end{Bmatrix}. \quad (3.5)$$

Another note, $\Xi$ appears in both systems in the same place. In this set of equations, the right side vector is herein referred to as F. These equations leverage the simplicity of the model. The input link drives link $L_{BC}$, and the input link has a known angular rate, $\omega_Q$.

# 4. EXTENDED KALMAN FILTER

## 4.1 Brief history of the Kalman Filter

In 1960, Rudolf Kalman published a paper titled *A New Approach to Linear Filtering and Prediction Problems* and, unbeknownst to him at the time; this paper forever changed the way engineers do filtering, prediction, and estimation. The Kalman filter excels at making sense of noisy measurements when combined with a dynamic model. It is a set of recursive equations to estimate the state of a process in such a way as to minimize the mean of the squared error [10]. With advancing technology, anyone with a modern computer can utilize the power of the Kalman filter. The following sections utilize the notation found in [11].

## 4.2 Introduction to the Discrete-Time Kalman Filter

For the Discrete-Time Kalman filter (DKF) to work properly, the system and measurement model should be of the form

$$x_{k+1} = A_k x_k + B_k u_k + G_k w_k \tag{4.1}$$

$$z_k = H_k x_k + v_k \tag{4.2}$$

where $x_k$ is the estimated value of the state from the previous time step, and $z_k$ is the measurement from the current time step. For this research, there is no control vector which results in (4.1) becoming

$$x_{k+1} = A_k x_k + G_k w_k. \tag{4.3}$$

One of the key assumptions of the DKF is that white noise processes operate on both the system and measurement model, but they are uncorrelated. These white noise processes are zero mean and covariance $Q_k$ and $R_k$ represented by

$$w_k \sim (0, Q_k) \tag{4.4}$$

$$v_k \sim (0, R_k). \tag{4.5}$$

Note, each term is k subscripted, denoting that it is possible and certainly valid that each term in the filter is time-varying. Due to the nature of this work, (4.3) can be simplified to be

$$x_{k+1} = A_k x_k + Gw \tag{4.6}$$

since G and w will not be changing with time. In the same way, (4.2) can also be simplified to be

$$z_k = H_k x_k + v \tag{4.7}$$

since v is not time-varying.

There are two components of the Discrete-Time Kalman filter, and they are the time update and measurement update equations. The time update equations are the propagation of the system model given by

$$\hat{x}_k^- = A_k x_{k-1} + Gw \tag{4.8}$$

$$P_k^- = A_k P_{k-1} A_k^T + Q \tag{4.9}$$

where the superscript minus denotes an *a priori* estimate meaning it is an uninformed guess at where the system is currently at given the previous state and the system dynamics model. P is the error covariance matrix detailing how each of the variables influences each other.

The measurement update equation informs the time update equation of the current measurement and seeks to accurately combine the two, using a Kalman gain found with the *a priori* error covariance matrix and noise R.

$$K = P_k^- H^T (H P_k^- H^T + R)^{-1} \tag{4.10}$$

This gain weights the measurements and predictions in the form of

$$\hat{x}_k = \hat{x}_k^- + K(z_k - H\hat{x}_k^-) \tag{4.11}$$

and now it is necessary to propagate the error covariance using the Kalman gain

$$P_k = (I - KH)P_k^-. \tag{4.12}$$

One of the powerful properties of the Kalman filter is the information contained in a single Kalman gain. Rather than needing to iterate over previous gains, each propagation embeds the state error history into a single Kalman gain requiring less computational load.

The DKF is an excellent state estimator but has a fatal flaw in its current form. Systems that cannot be written in the form found in (4.1) will not fit this model, and this filter will not suffice. For a more advanced model that includes nonlinearities, a more sophisticated filter is required.

## 4.3 Introduction to the Extended Kalman Filter

The Extended Kalman filter (EKF) is simply an extension of the DKF but rather than having a linear combination of terms; the EKF is a nonlinear combination that takes the form

$$\dot{x} = a(x, u, t) + G(t)w. \tag{4.13}$$

Keeping in mind this research does not deal with control input, (4.13) becomes

$$\dot{x} = a(x, t) + G(t)w \tag{4.14}$$

and the measurement model is given by

$$z = h(x, t) + v \tag{4.15}$$

where w and v are still white noise processes.

The system model equations are written in continuous time and for simplicity of implementation, the EKF is implemented in continuous time, resulting in a Continuous – Continuous EKF. Analogous to the time and measurement update, this filter has estimate update and an error covariance update.

The estimate update is

$$\dot{\hat{x}} = a(\hat{x}, t) + K[z - h(\hat{x})] \tag{4.16}$$

where $a(\hat{x}, t)$ are the nonlinear model equations and $h(\hat{x})$ are the measurement estimates.

Propagating the error covariance is given by

$$\dot{P} = AP + PA^T + GQG^T - PH^T R^{-1} HP \tag{4.17}$$

and the Kalman gain is calculated by

$$K = PH^T(\hat{x}, t)R^{-1}. \tag{4.18}$$

What makes the EKF unique and accurate for nonlinear systems is the presence of Jacobians in the error covariance update.

$$A(x, t) = \frac{\partial a(x, t)}{\partial x} \tag{4.19}$$

$$H(x, t) = \frac{\partial h(x, t)}{\partial x} \tag{4.20}$$

The easiest way to move the filter forward is using the built-in MATLAB differential equation solver ode45. The Kalman filter is placed into a state space function and, when necessary, called by ode45 during the iterative process.

An interesting thing to note here is that an RK4 method was in the original design. After implementation, the Kalman filter consistently diverged, and this is due to the invariant time step used by the solver.

## 4.4 EKF Implementation with two-link model

MATLAB's built-in ode45() function was the differential equation solver for this case. Its varying time step proved to be the best way to solve this system. All components are simple to implement except for the Jacobians in the error covariance update that account for the nonlinearities.

The nonlinear equations governing this system are buried inside $a(\hat{x}, t)$ of (4.16).

$$a(\hat{x}, t) = \begin{Bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\omega}_1 \\ \dot{\omega}_2 \end{Bmatrix} = \begin{Bmatrix} \omega_1 \\ \omega_2 \\ \alpha_1 \\ \alpha_2 \end{Bmatrix} \tag{4.21}$$

A complete Jacobian $A(x, t)$ is created by

$$A(x, t) = \begin{bmatrix} \frac{\partial \begin{Bmatrix} \omega_1 \\ \omega_2 \end{Bmatrix}}{\partial \theta_1} & \frac{\partial \begin{Bmatrix} \omega_1 \\ \omega_2 \end{Bmatrix}}{\partial \theta_2} & \frac{\partial \begin{Bmatrix} \omega_1 \\ \omega_2 \end{Bmatrix}}{\partial \omega_1} & \frac{\partial \begin{Bmatrix} \omega_1 \\ \omega_2 \end{Bmatrix}}{\partial \omega_2} \\ \frac{\partial \begin{Bmatrix} \alpha_1 \\ \alpha_2 \end{Bmatrix}}{\partial \theta_1} & \frac{\partial \begin{Bmatrix} \alpha_1 \\ \alpha_2 \end{Bmatrix}}{\partial \theta_2} & \frac{\partial \begin{Bmatrix} \alpha_1 \\ \alpha_2 \end{Bmatrix}}{\partial \omega_1} & \frac{\partial \begin{Bmatrix} \alpha_1 \\ \alpha_2 \end{Bmatrix}}{\partial \omega_2} \end{bmatrix} \tag{4.22}$$

which is quickly simplified to yield

$$A(x, t) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{\partial \begin{Bmatrix} \alpha_1 \\ \alpha_2 \end{Bmatrix}}{\partial \theta_1} & \frac{\partial \begin{Bmatrix} \alpha_1 \\ \alpha_2 \end{Bmatrix}}{\partial \theta_2} & \frac{\partial \begin{Bmatrix} \alpha_1 \\ \alpha_2 \end{Bmatrix}}{\partial \omega_1} & \frac{\partial \begin{Bmatrix} \alpha_1 \\ \alpha_2 \end{Bmatrix}}{\partial \omega_2} \end{bmatrix}. \tag{4.23}$$

A complete Jacobian $H(x, t)$ is given by

$$H(x,t) = \begin{bmatrix} 0 & 0 & \frac{\partial \begin{Bmatrix} \omega_1 \\ \omega_2 \end{Bmatrix}}{\partial \theta_1} & \frac{\partial \begin{Bmatrix} \omega_1 \\ \omega_2 \end{Bmatrix}}{\partial \theta_2} \end{bmatrix}. \tag{4.24}$$

where the first two columns are zeros. Recall $\Xi$, from Section 3.1.1, as

$$\Xi = \begin{bmatrix} -L_{AB}\sin(\theta_1) & L_{BC}\sin(\theta_2) \\ L_{AB}\cos(\theta_1) & -L_{BC}\cos(\theta_2) \end{bmatrix}^{-1}. \tag{4.25}$$

Finding the partials from (4.23) and (4.24) will require the derivative of both angular rate and angular acceleration. Redefining angular rate gives

$$\begin{Bmatrix} \omega_{AB} \\ \omega_{BC} \end{Bmatrix} = \Xi^{-1} G \tag{4.26}$$

where

$$G = \begin{Bmatrix} -\omega_Q r_C \sin(\theta_Q) \\ \omega_Q r_C \cos(\theta_Q) \end{Bmatrix} \tag{4.27}$$

from Section 3.1.1.

Taking a derivate of a matrix inverse numerically is rather challenging, but it is possible to exploit an algebra rule stating

$$\frac{\partial \mathbf{Z}^{-1}}{\partial x} = -\mathbf{Z}^{-1}\frac{\partial \mathbf{Z}}{\partial x}\mathbf{Z}^{-1}. \tag{4.28}$$

The product rule is still applicable meaning

$$\frac{\partial \begin{Bmatrix} \omega_{AB} \\ \omega_{BC} \end{Bmatrix}}{\partial x} = -\Xi^{-1}\frac{\partial \Xi}{\partial \theta_i}\Xi^{-1}G + \Xi^{-1}\frac{\partial G}{\partial \theta_i} \tag{4.29}$$

where $G$ is a constant value resulting in a derivative of zero for all states allowing (4.29) to become

$$\frac{\partial \begin{Bmatrix} \omega_{AB} \\ \omega_{BC} \end{Bmatrix}}{\partial x} = -\Xi^{-1}\frac{\partial \Xi}{\partial \theta_i}\Xi^{-1}G. \tag{4.30}$$

Recall that $\Xi$ appears in the angular acceleration equations as well. Although not as simple as angular rate, the partials of the angular acceleration are found in the same way.

$$\begin{Bmatrix}\alpha_{AB}\\ \alpha_{BC}\end{Bmatrix} = \Xi^{-1}\mathrm{F} \tag{4.31}$$

where *F* is the right side of the angular rate equations.

$$\mathrm{F} = \begin{Bmatrix}\omega_{AB}^2 L_{AB}\cos(\theta_1) - \omega_{BC}^2 L_{BC}\cos(\theta_2) - \omega_Q^2 r_C\cos(\theta_Q)\\ \omega_{AB}^2 L_{AB}\sin(\theta_1) - \omega_{BC}^2 L_{BC}\sin(\theta_2) - \omega_Q^2 r_C\sin(\theta_Q)\end{Bmatrix}. \tag{4.32}$$

Employing the same method from (4.28), the partials of angular accelerations become

$$\frac{\partial\begin{Bmatrix}\alpha_{AB}\\ \alpha_{BC}\end{Bmatrix}}{\partial \mathrm{x}} = -\Xi^{-1}\frac{\partial\Xi}{\partial \mathrm{x}}\Xi^{-1}\mathrm{F} + \Xi^{-1}\frac{\partial \mathrm{F}}{\partial \mathrm{x}} \tag{4.33}$$

and unlike (4.29), this equation will not simplify due to F being a function of all four states.

# 5. GENETIC ALGORITHM

The following sections deal with gait parameter analysis in the form of a genetic algorithm (GA). The premise is using a four-bar Hoeken linkage system to generate a more realistic gait path but with variable parameters identifiable with a GA. This allows a personalized gait model to be implemented inside a Kalman Filter. Similar things have been done by Dutch artist, Theo Jansen, who used a GA to identify gait parameters for his Strandbeest [12]. Waldron also worked on making machines walk and was interested in walking kinematics similar to this research [13].

## 5.1 Introduction to Genetic Algorithms

Known as a type of evolutionary algorithm, GAs employ the idea of survival of the fittest. Robert King says it well with:

> *The techniques emulate natural evolution of individual structures through processes inspired by natural selection and reproduction. These processes depend on the fitness of the individuals to survive and reproduce in a hostile environment. Evolution can be viewed as an optimization process that can be emulated by a computer. Evolutionary Computation is essentially a stochastic search technique with remarkable abilities for searching for global solutions. [14]*

Even Robert King had to rely on another metaphor to get his point across. He uses the words of Zbigniew Michalewicz from his 1992 book *Genetic Algorithms + Data Structures = Evolution Programs.* Michalewicz says:

> *Do what nature does. Let us take rabbits as an example: at any given time there is a population of rabbits. These faster, smarter rabbits are less likely to be eaten by foxes, and therefore more of them survive to do what rabbits do best: make more rabbits. Of course, some of the slower, dumber rabbits will survive just because they are lucky. This surviving population of rabbits starts breeding. The breeding results in a good mixture of rabbit genetic material: some slow rabbits breed with fast rabbits, some fast with fast, some smart*

*rabbits with dumb rabbits, and so on. And on the top of that, nature throws in a 'wild hare' every once in a while by mutating some of the rabbit genetic material. The resulting baby rabbits will (on average) be faster and smarter than those in the original population because more faster, smarter parents survived the foxes… [15]*

## 5.2 GA Considerations

Five big issues should be considered with a GA:

1. What is the range of candidate solutions?
2. What is the cost function?
3. How large is the initial population?
4. How will the population be encoded and decoded at each iteration?
5. How many generations are needed to satisfy the problem?

When creating a GA, the range of candidate solutions is important. Candidate solutions are the range of values the algorithm can explore. If the problem to be optimized deals with human height, perhaps a range of 0-5m is not exactly giving the GA the best chance at the optimal solution. Using a range of 1-2.35m would allow exploration of more possible solutions.

The cost function is perhaps the most important piece of the entire genetic algorithm. GAs are not intelligent, as they will attempt to exploit any loophole in the cost function. If the cost function is not well defined, the answer will most likely not be optimal. A cost function provides a way to assign a fitness to each individual in the population. The cost function in this GA is Euclidean distance, and fitness is the inverse of the cost so that low cost is equated to high fitness and high cost translates to a low fitness.

$$\text{cost} = (x - x_i)^2 + (y - y_i)^2 \tag{5.1}$$

The size of the initial population will depend on the available computing power. These algorithms tend to be on the heavier side of the computational scale. Typically, a population size of 30-50 will suffice, but if plenty of power is available, population sizes upwards of 100 are feasible. Note, always use an even number, so each member will have a mate.

Population encoding and decoding are very important. Populations are initialized as binary vectors or strings. MATLAB has nice built-in functions to handle both binary vectors

strings, so either is fine. An example of how everything fits together will be at the end of this section. The inner workings of GAs are not completely obvious.

The final issue to consider is the number of generations needed to solve the problem. This is another case of how much computational power is available. Pushing generations into the thousands tends to get time-consuming, and solutions begin to become asymptotic. Keeping a record of the fittest individual ever encountered through all populations is one method of checking if higher generations yield better results. The device used in this research had an Intel Core i7 and took approximately 5 minutes to run one simulation with a population of 100 for 800 generations. The problem of long run-times can be avoided if there is adequate stopping criteria. If the acceptable error in the result meets the criteria, the GA can be terminated.

## 5.3 GA Example

Here is a simple example. Suppose, instead of doing a minimization optimization, the problem is to maximize a specific function. This is no problem. The GA can handle maximization and minimization. The cost function is

$$\text{cost} = f = x^2. \tag{5.2}$$

The population size is deemed to be 4 (could be 6, 8, 100, etc.). Random binary sequences are generated to give the initial starting population. A length of 5 bits is chosen.

Each member of the population has their binary value decoded to be a decimal value. This is where scaling happens to account for the range of candidate solutions. If the search space was between 0 and 2, simply normalize all decimal values to be within that range. At this point, the members are ready to go through the cost function. Each member gets their chance at the function, and their values are recorded.

Next, each member needs to be assigned a fitness as a percentage of the population's total fitness. This is found with

$$\text{fitness} = \frac{\text{cost}_i}{\sum \text{cost}} \tag{5.3}$$

where each member's cost is divided by the total population cost. Fitness can now be recorded.

It is important to notice that if the sum of fitness is taken, it will be equal to 1. If the population is sorted by fitness and the cumulative sum is taken, the sum look like Figure 23.

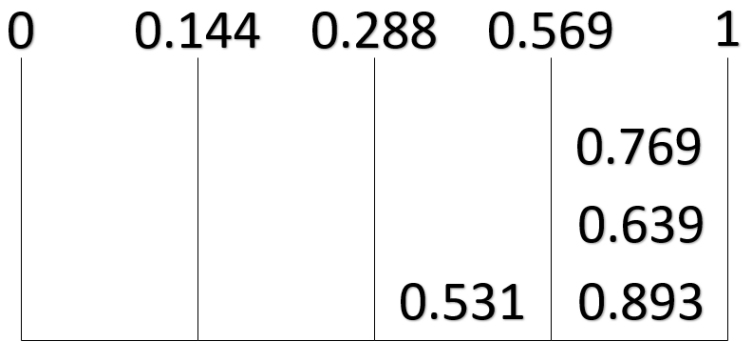


**Figure 23: GA example – mating votes**

The values at the top of the figure are the cumulative sum of the fitness after being sorted in ascending order. A random function generates a list of numbers between 0 and 1, the length of the number of individuals in the population, and those numbers get sorted into bins. The number of values in each is the number of mating votes for a specific member. This process does not explicitly remove unfit members from the population but it does make it much harder to enter the mating pool. The less fit individuals have a smaller bin size while the fittest individuals have larger bins. The chances of a randomly generated number falling between 0.569 and 1 is greater than the chances of a randomly generated number falling between 0 and 0.144. In this way, the GA gives the best chances to the fittest individuals. A completed GA is shown in Table 1.

**Table 1: GA example – mating votes**

| Member ID | Binary Value | Dec. Value | Cost | Fitness | Mating Votes |
|---|---|---|---|---|---|
| 1 | 01111 | 15 | 225 | 0.144 | 0 |
| 2 | 11010 | 26 | 676 | 0.431 | 3 |
| 3 | 10101 | 21 | 441 | 0.281 | 1 |
| 4 | 01111 | 15 | 225 | 0.144 | 0 |

The Genetic algorithm has done its job this iteration. It has randomly selected the fittest members of the population to reproduce.

Reproduction is actually the simplest operation of the entire algorithm. The mating order is randomly selected. In this case, the order is [4 3 2 1]. The 4th member of the mating pool will mate with the 3rd member of the mating pool. Likewise, the 2nd member of the mating pool will mate with the 1st member of the mating pool. To create the mating pool, each member of the population gets into the pool n times, where n is the number of mating votes received. For this iteration, the mating pool is

10101

11010

11010

11010.

The order of the initial pool does not matter since all mates are randomly selected after establishing the pool. The mating pool sorted in mating order becomes

11010

11010

11010

10101.

To reproduce, a random crossover point is selected (the value is between zero and the number of bits), and the genes are swapped at that point. For the first pair, the crossover point is three. Both members are split after the third gene and are swapped with one another. The first pair become

11010

11010

and the second pair become

11001

10110.

Similar to biologic mutations, Genetic Algorithms should also experience mutations. Prior to beginning a GA optimization, a mutation probability is chosen, a value typically between

0.002 and 0.02. This probability applies to each bit in the population, meaning during the mutation phase, each bit has between a 0.2% and 2% chance of mutation. The idea is mutations can significantly increase or decrease member fitness which can be both helpful and detrimental, but it is still recommended. Bad mutations resulting in unhealthy member fitness will usually prevent that member from mating, but good mutations can discover more fit individuals. Implementing a mutation means a bit is switched to be the opposite value (0 becomes 1 or 1 becomes 0).

After crossover and mutation, the new members enter the next iteration of the GA as the new population. Table 2 shows the new population.

**Table 2: GA example – new population**

| Member ID | Binary Value | Dec. Value | Cost | Fitness | Mating Votes |
|---|---|---|---|---|---|
| 1 | 11010 | 26 | | | |
| 2 | 11010 | 26 | | | |
| 3 | 11001 | 25 | | | |
| 4 | 10110 | 22 | | | |

The algorithm is now ready to use the new members to maximize the cost function. The members with lower fitness have a lower likelihood of reproducing, giving a higher population fitness in subsequent generations. This continues until the stopping criteria are satisfied, resulting in an optimized solution.

# 6. RESULTS & DISCUSSION

## 6.1 Two-Link model EKF

This iteration of the Kalman Filter used the dynamics from the Kinematics model, which made validating the filter simple. The initial conditions of the system placed

$$\theta_1 = 200° \tag{6.1}$$

and

$$\theta_2 = 110°. \tag{6.2}$$

The kinematic model provides true values to act as a baseline for judging the Kalman filter. True angles and true angular rates using the above initial conditions are found in Figure 24 and Figure 25, respectively.

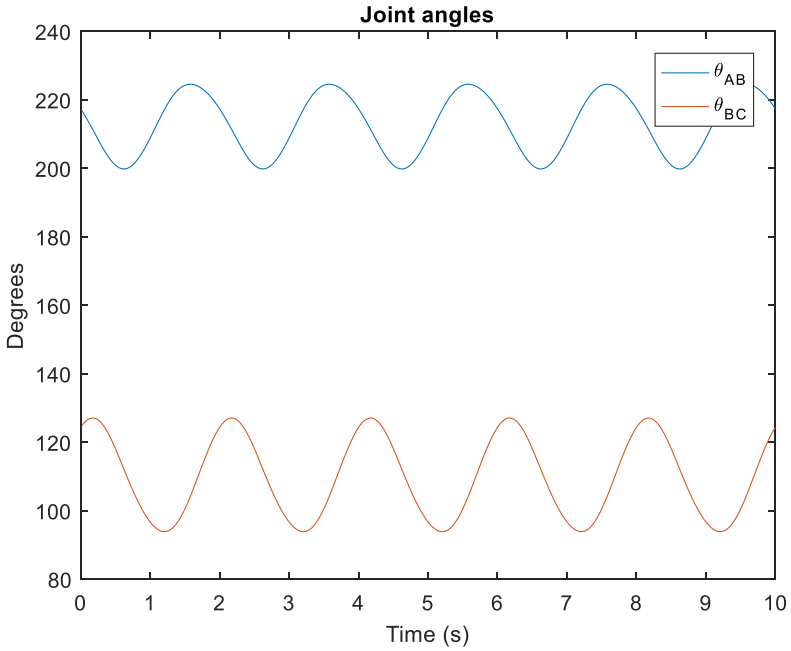


**Figure 24: True values of joint angles for the two-link model**

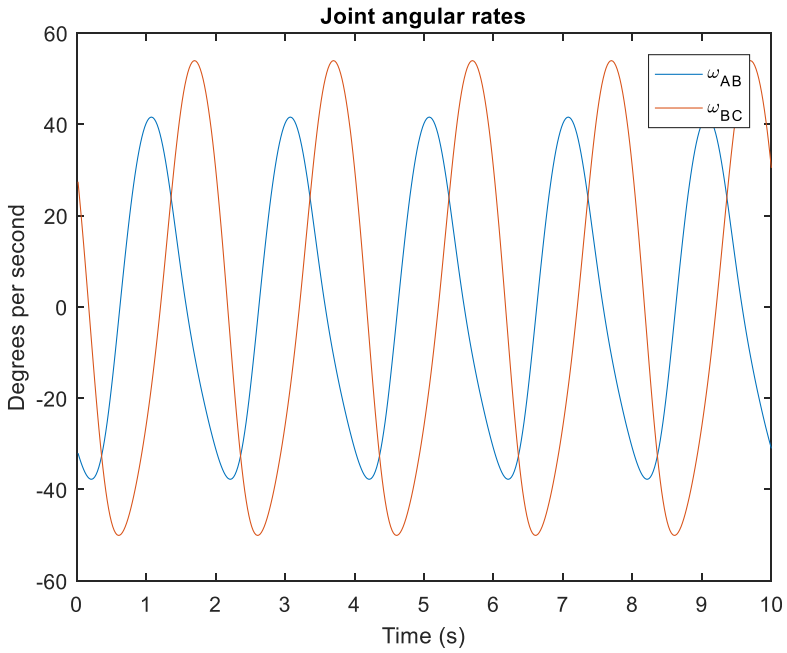


**Figure 25: True values of joint angular rates for the two-link model**

The data from Figure 25 was then corrupted with a Gaussian noise with a standard deviation of 0.08 radians per second and is seen in Figure 26. This equates to adding an uncertainty of roughly $4.5\,\frac{deg}{s}$ to the angular rates. The addition of uncertainty simulates noisy IMU measurements, adding to the difficulty of the problem.

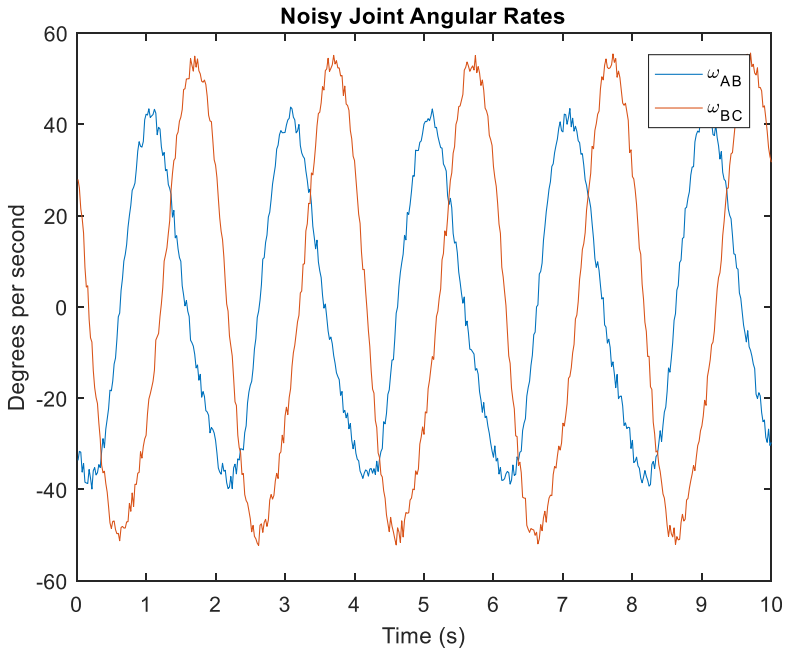


**Figure 26: Noisy joint angular rates with noise with a SD of 0.08 rad/s**

Due to the Kalman filter using the same model as found in the kinematic model, the process noise covariance, Q, will be known. In practice, this will most likely not be the case since the model cannot predict everything but in this case,

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6.3}$$

The error covariance is initialized to be

$$P = 1000 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.4}$$

and the measurement noise covariance was tuned to be

$$R = 0.5. \tag{6.5}$$

To be sure the Kalman filter is capable of converging, the initial joint angles were set incorrectly.

$$\theta_1 = 171° \tag{6.6}$$

$$\theta_2 = 81° \tag{6.7}$$

The percent difference for $\theta_1$ is over 15%, and $\theta_2$ is just over 30%. These values were chosen because it is unlikely a doctor or clinician would ever miss an initial angle by this much, given the number of tools available to them. Estimating initial angle this poorly would most likely never happen, but this shows the power of the Kalman filter.

Running the Kalman filter over the noisy data yields the convergence plot found in Figure 27. Convergence is the difference between the known states from the test data and the Kalman states produced by the Kalman Filter.

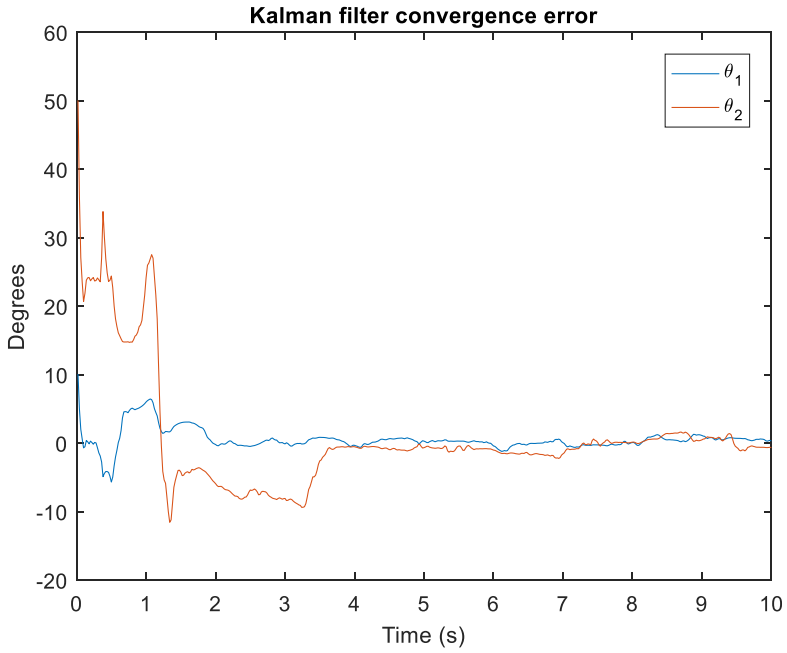


**Figure 27: Kalman filter convergence error with angular rate noise of SD 0.08 rad/s and initial angle guess incorrect by 0.5 rad**

The filter eventually converges, but it does take about 3.5 seconds, which is certainly not ideal, especially if working with limited space. The filtered angular rates also converge but in a shorter amount of time, about 2 seconds. This is seen in Figure 28.

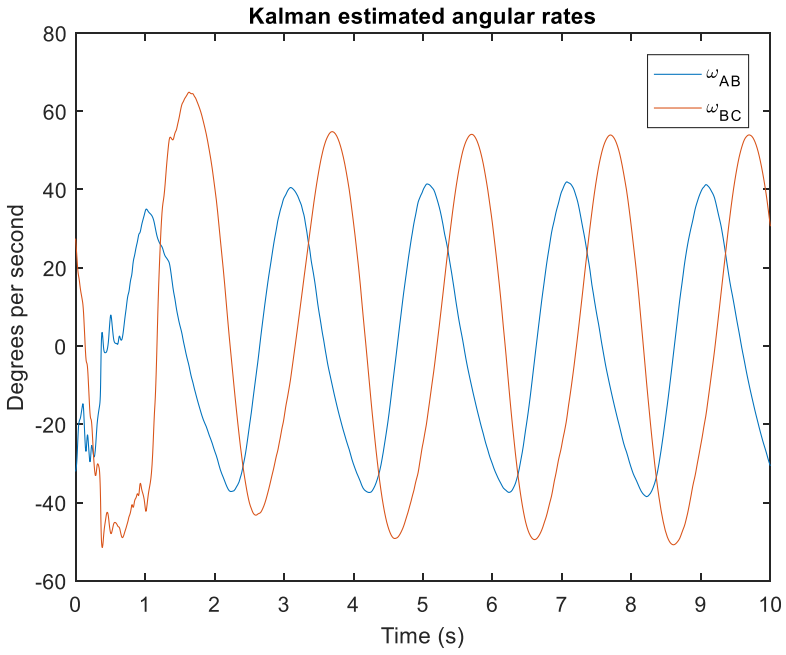


**Figure 28: Kalman filter angular rates with input noise of SD 0.08 rad/s**

The best way to show convergence is by looking at the estimated joint angles found in Figure 29. A convergence time of approximately 3.5 seconds is seen when the Kalman filter's $\theta_2$ converges to the true value.

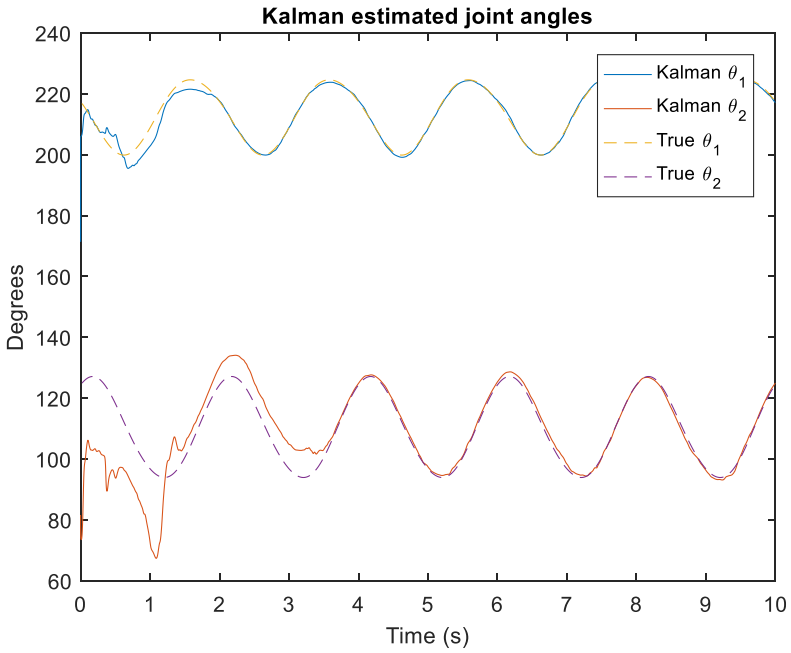


**Figure 29: Kalman filter joint angle with angular rate noise of SD 0.08 rad/s and initial angle guess incorrect by 0.5 rad**

Interesting to note here is that although the filter took a long time to converge for $\theta_2$, convergence for $\theta_1$ was achieved rather quickly, around the 1-second mark. Simply improving the initial guess of the joint angles at the beginning proves to be a powerful solution to that problem. Figure 30 shows the result.

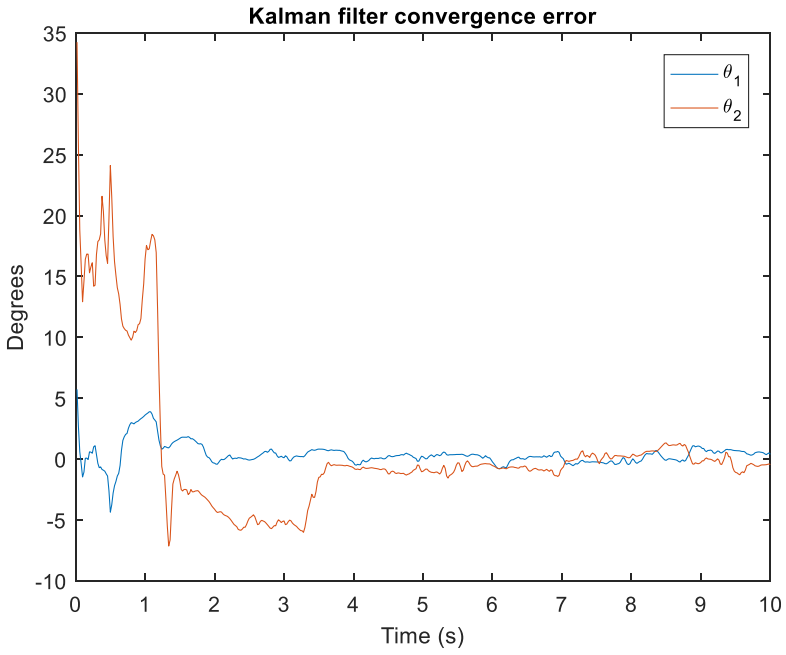


**Figure 30: Kalman filter convergence error with noise of SD 0.08 rad/s and initial guess incorrect by 0.25 rad**

Reducing the initial joint estimate error by a factor of 2 significantly improves the convergence time. Rather than 3.5 seconds, both joint angles converge around 1 second, which is three times as fast. This was achieved by being within 15 degrees of the actual starting angle.

Kalman angular rates also look much better in Figure 31 than in Figure 28. Convergence here is seen around the 1.5-second mark.

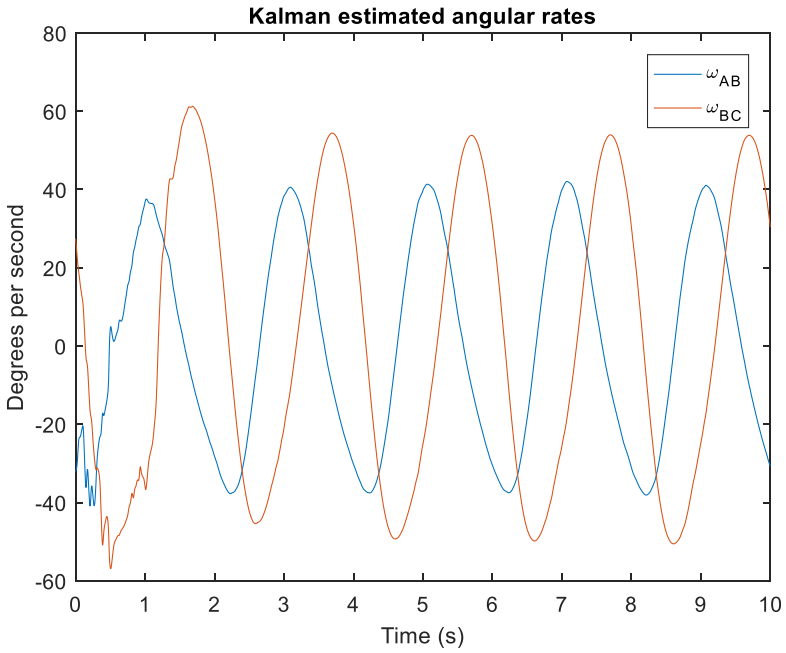


**Figure 31: Kalman filter angular rates with noise of 0.08 rad/s and initial guess incorrect by 0.25 rad**

The Kalman estimate joint angles convergence is shown in Figure 32. Similar to the angular rates, the joint angle estimations are much tighter than previously. Convergence is achieved in approximately 1 second.

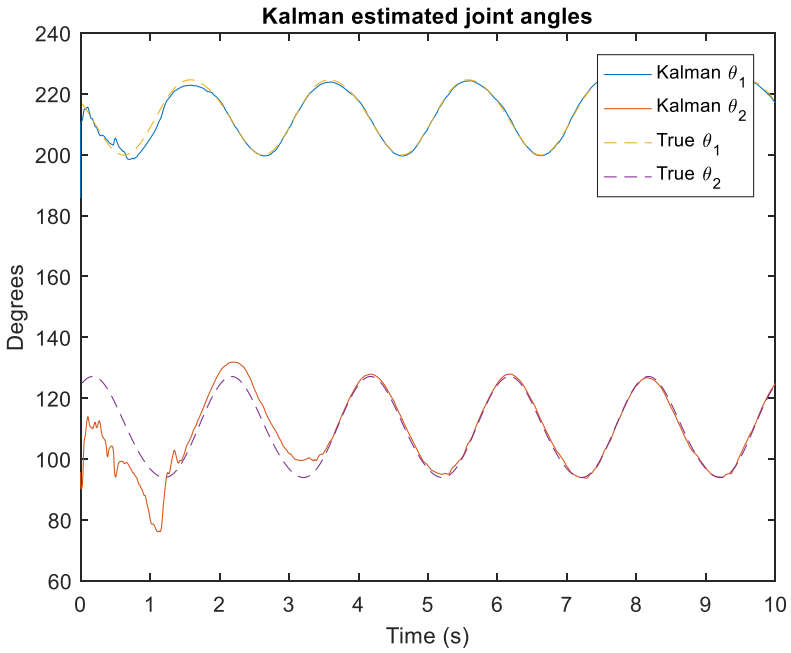


**Figure 32: Kalman filter joint angle with noise of SD 0.08 rad/s and initial guess incorrect by 0.25 rad**

Great convergence is achieved here, but the filter needs to be pushed a bit more. Simulating poor IMU performance, the noise is increased from 0.08 $\frac{rad}{s}$ to 0.3 $\frac{rad}{s}$. This increase simulates IMUs that have a measurement standard deviation of 17 degrees per second which is a good representation of an inexpensive, off-the-shelf IMU. The convergence error is shown in Figure 33.

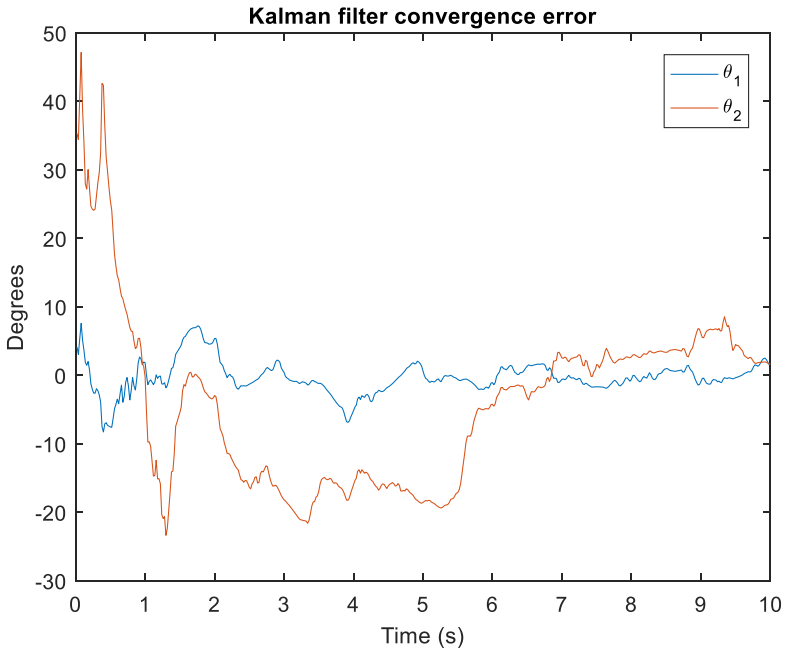


**Figure 33: Kalman filter convergence error with noise of SD 0.3 rad/s and initial guess incorrect by 0.25 rad**

Although not as past iterations, this shows that convergence is still possible even with inaccurate gyroscopes. Convergence here is achieved in roughly 6 seconds, which could suffice in certain scenarios. Estimated angular rates are found in Figure 34 while estimated joint angles are seen in Figure 35.

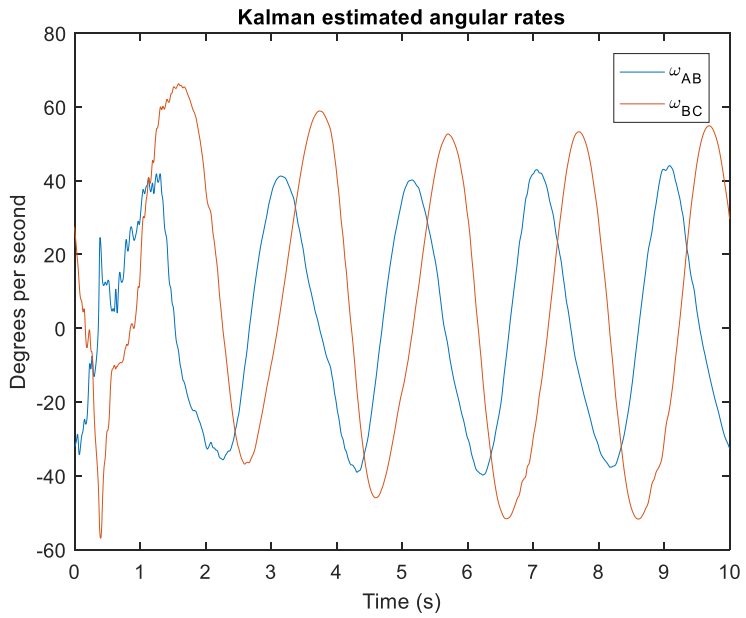


**Figure 34: Kalman filter angular rates with noise of 0.3 rad/s and initial guess incorrect by 0.25 rad**

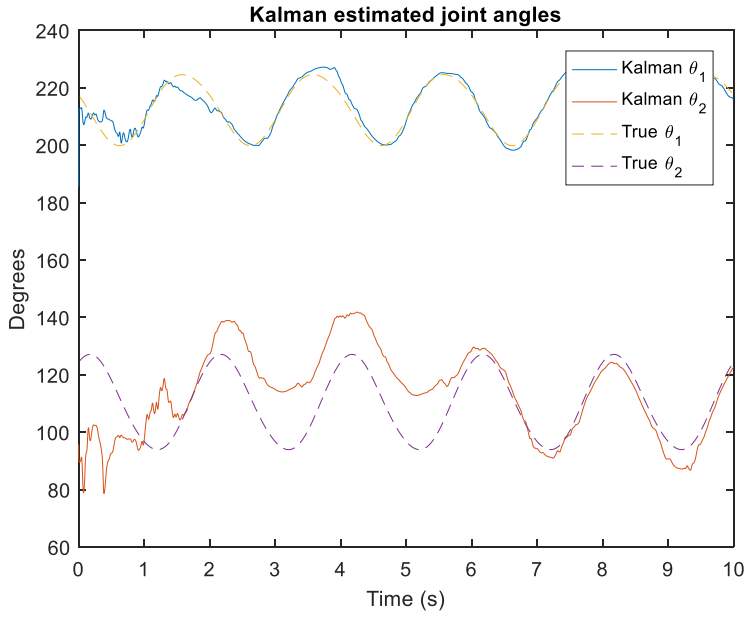


**Figure 35: Kalman filter joint angle with noise of SD 0.3 rad/s and initial guess incorrect by 0.25 rad**

The EKF confirms that, when combined with a good model, moderately inaccurate wearable gyroscopes can compete with traditional motion captures. The drawback with these

inaccurate wearables is the convergence time. Having a convergence time of 6-7 seconds could prove to be troublesome for patients with severe gait troubles, but that issue can be curbed with the use of more accurate, and expensive, gyroscopes.

## 6.2 Genetic Algorithm using Hoeken ankle path

This section deals with generating a useable gait model for a Kalman Filter based on the known positions given the pull toy model. The genetic algorithm approach to estimating the four-bar Hoeken system for generated data has proved to be fruitful. The method for testing the GA has consisted of gradually increasing the number of parameters to estimate. The data was generated with

$$a = 0.2 \text{ m} \tag{6.8}$$

resulting in

$$b = 0.5 \text{ m} \tag{6.9}$$

$$c = 0.5 \text{ m} \tag{6.10}$$

$$d = 0.4 \text{ m} \tag{6.11}$$

$$e = 1 \text{ m}. \tag{6.12}$$

The first test for the GA is attempting to estimate the length of link e, given the length of all other links. The performance is measured by the average cost of the population. The solution quickly becomes asymptotic around generation 20 as seen in Figure 36. In these position based solutions, the cost function is Euclidean distance between motion capture position data and GA position data.

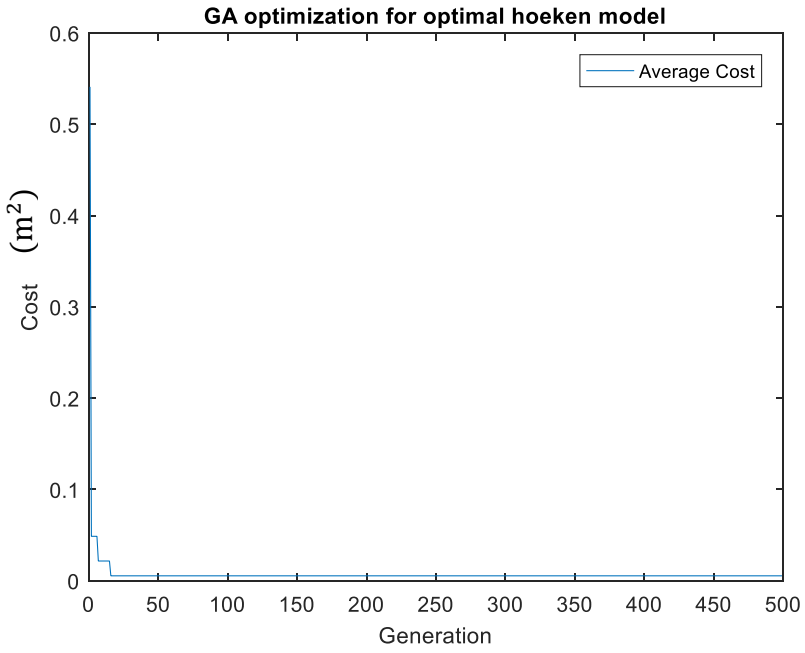


**Figure 36: GA optimization searching for link e**

In this search, the fittest member of the population occurred in generation 385 and had a value of

$$e = 1.00293 \text{ m}. \tag{6.13}$$

The solution fully converged at generation 385, and subsequent generations were never as good. The resultant path from this link length is given in Figure 37.

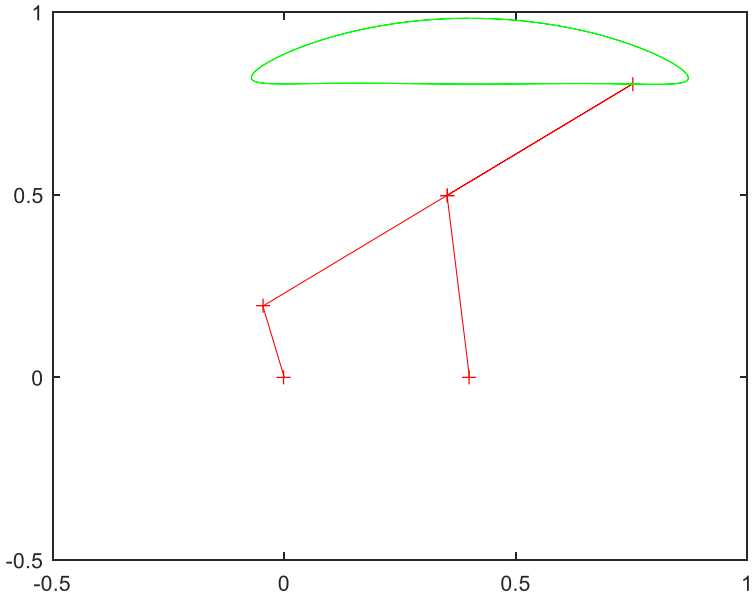


**Figure 37: GA optimized path for Hoeken system searching for link e**

This path is not noticeably different than the path seen in Figure 15 from Section 2. Progressively expanding the number of variables the GA searched for resulted in a complete Hoeken system estimation. The progressive expansion was to explore how the addition of more optimization variables affected the accuracy. As expected, more optimization variables resulted in reduced accuracy when population sizes and generations were held relatively constant due to computation power and time. This result of an entire system optimization is found in Figure 38. Intermediate optimizations can be found in 9.1.

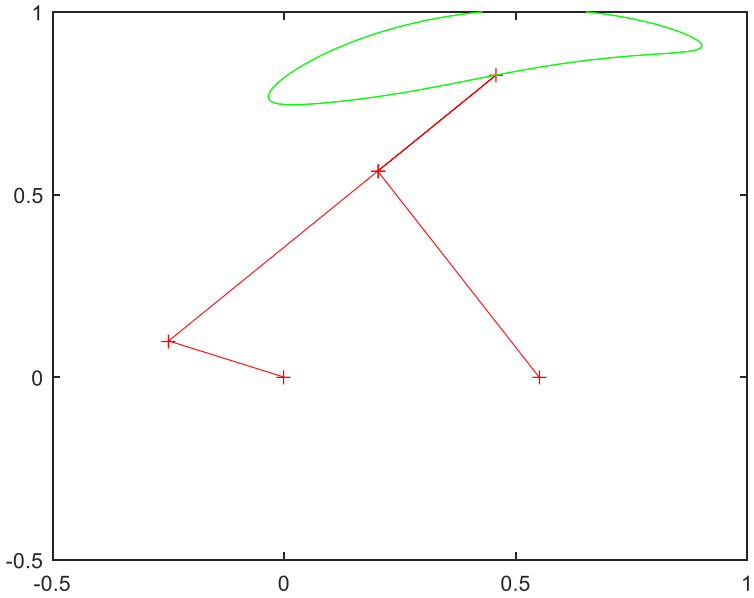


**Figure 38: GA Optimization for Hoeken searching for all links**

Considering the algorithm started with random values and converged to something that closely resembles the real path was a great start to using the GA for gait parameter estimation. This shows that given position data, the GA is a powerful tool to create patient-specific gait models that can be used in the time update equations of the Kalman Filter.

## 6.3 Unconstrained GA using motion capture ankle paths

Herein, the terms constrained and unconstrained will frequently be used. These ideas go back to the original Hoeken idea. An unconstrained system is a system whose geometry is not constrained to be that of a Hoeken system. This gives the GA an opportunity to seek out different variations of four bar systems. Opposite the unconstrained system is the constrained system, which forces the algorithm to search for Hoeken systems explicitly. This reduces the freedom of the optimization but gives a more focused search.

Figure 39 and Figure 40 show the unconstrained results of the GA using healthy gait ankle paths. The fit is clearly not great, and the cost of the system is relatively high compared to the next section. 9.2 shows the optimized paths and associated costs for the two gait pathologies drop foot and drag foot. For legend brevity, motion capture has been shortened to “Mo-cap.”

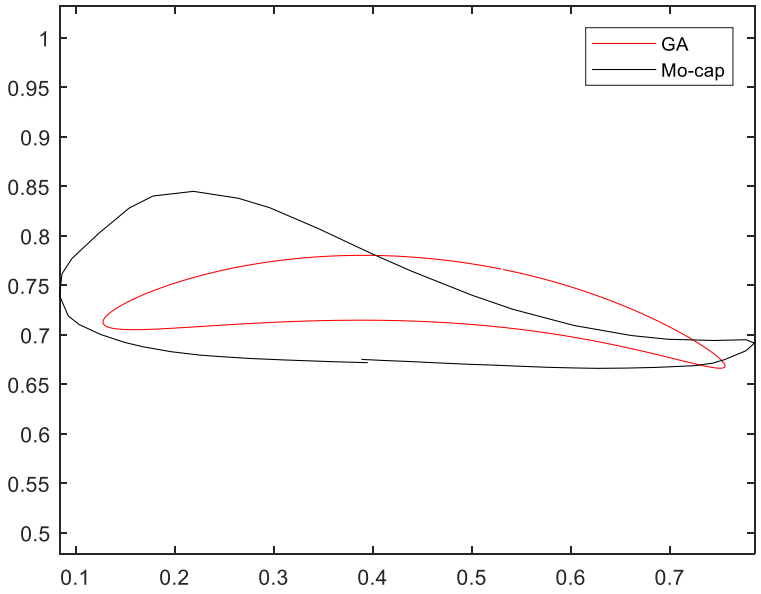


**Figure 39: Unconstrained GA optimization using healthy gait ankle path**

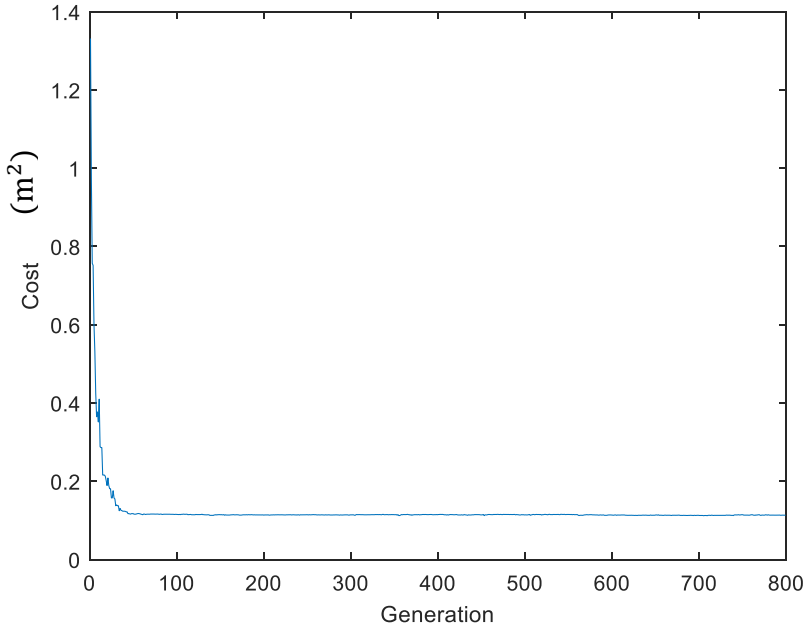


**Figure 40: Unconstrained GA optimization cost using healthy gait ankle path**

## 6.4 Constrained Hoeken GA using motion capture ankle paths

Next, the true ankle path from motion capture was used to train the GA. Like before, the gait pathologies drop foot and drag foot were simulated in addition to a healthy gait cycle. The models generated in this section were forced to be Hoeken, conforming to the required geometries. Figure 41 and Figure 42 show the optimized Hoeken system path for a healthy gait cycle and the associated cost, respectively.

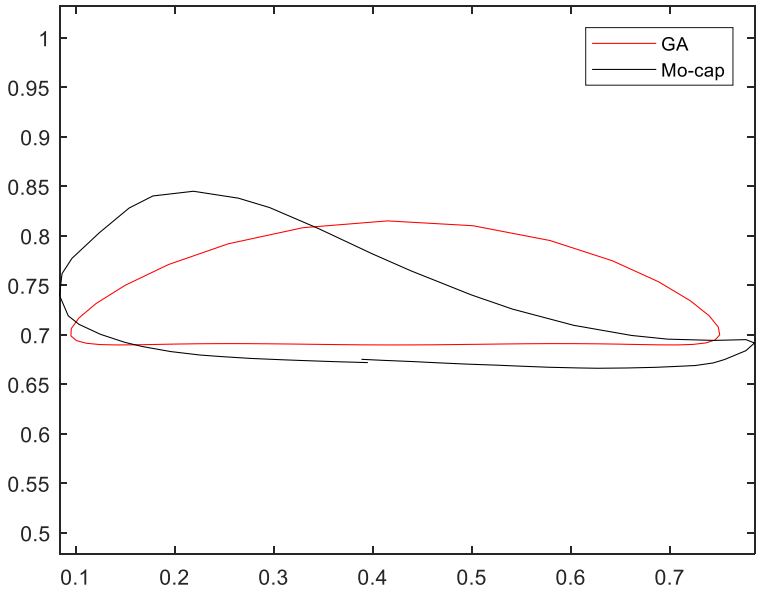


**Figure 41: Constrained Hoeken GA optimization using healthy gait ankle path**

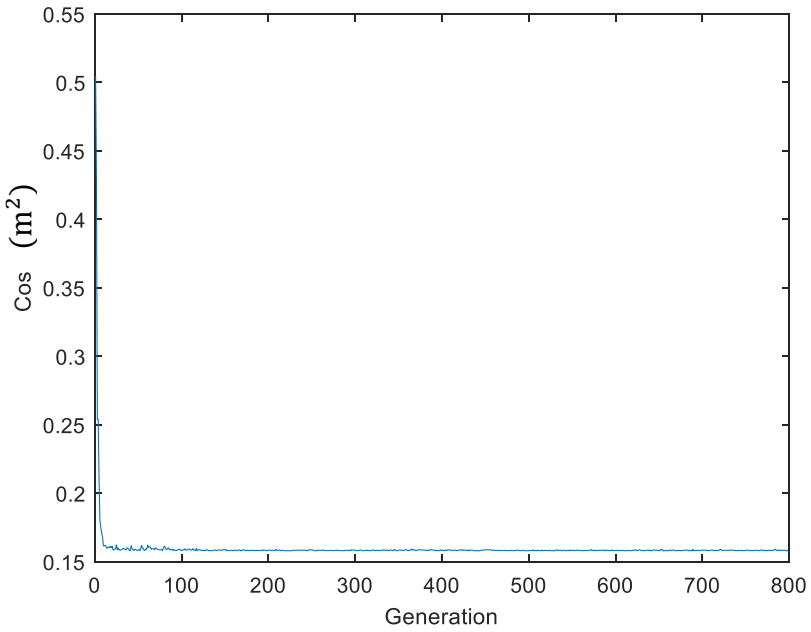


**Figure 42: Constrained Hoeken GA optimization cost using healthy gait ankle path**

From looking at both the fit and the cost of the two approaches, the constrained system gave a model with half the cost of the unconstrained system. 9.3 shows the optimized paths and associated costs for the two gait pathologies drop foot and drag foot. The constrained GA offered better results for the pathologies when compared to unconstrained.

## 6.5 Unconstrained GA using motion capture mid-tibia paths

During motion capture, a marker was placed around the midpoint of the tibia to help capture angular rate, but this marker position showed an interesting phenomenon. All three gait cycles more closely resemble a Hoeken system than data collected at the ankle. Similar to 6.3, Figure 43 and Figure 44 show the optimized system path for a healthy gait cycle as well as the associated cost. 9.4 contains the optimized paths and associated costs for the two gait pathologies drop foot and drag foot.

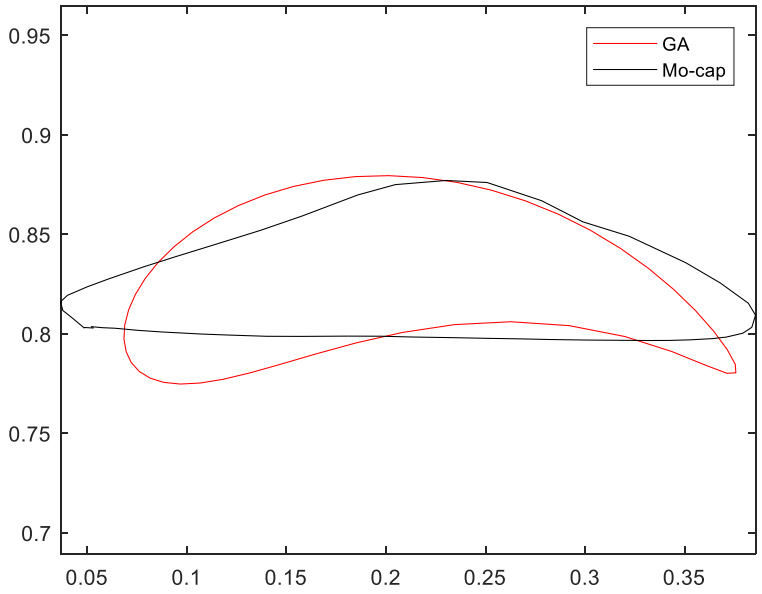


**Figure 43: Unconstrained GA optimization using healthy gait mid-tibia path**

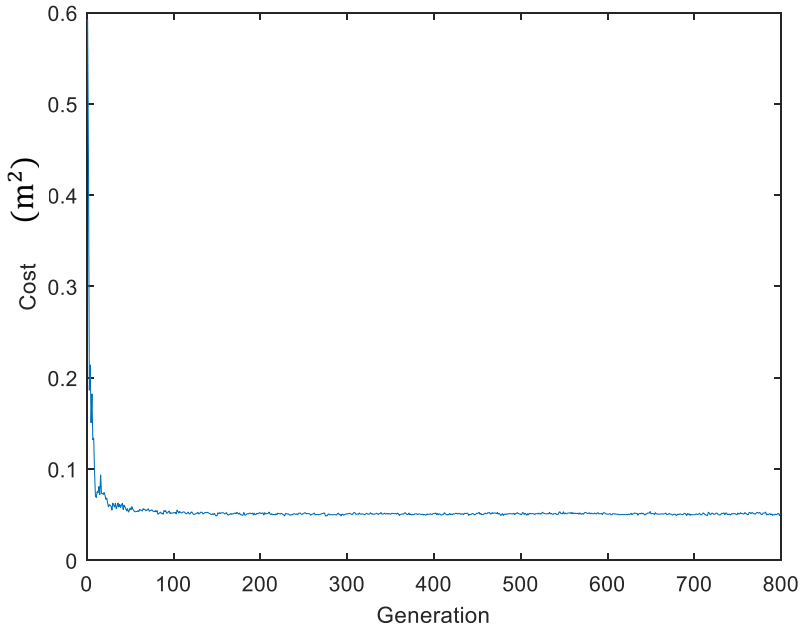


**Figure 44: Unconstrained GA optimization cost using healthy gait mid-tibia path**

## 6.6 Constrained Hoeken GA using motion capture mid-tibia paths

Similar to 6.4, the algorithm was constrained to Hoeken geometries only, and the results are found in Figure 45 and Figure 46.

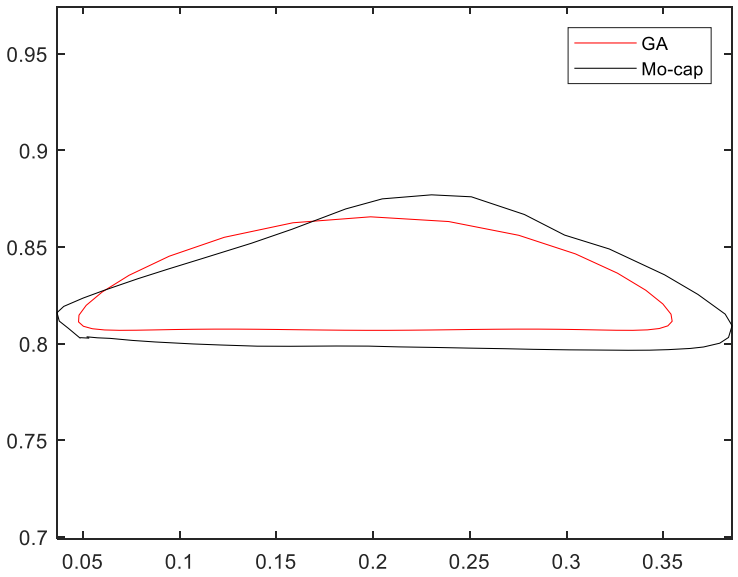


**Figure 45: Constrained GA optimization using healthy gait mid-tibia path**

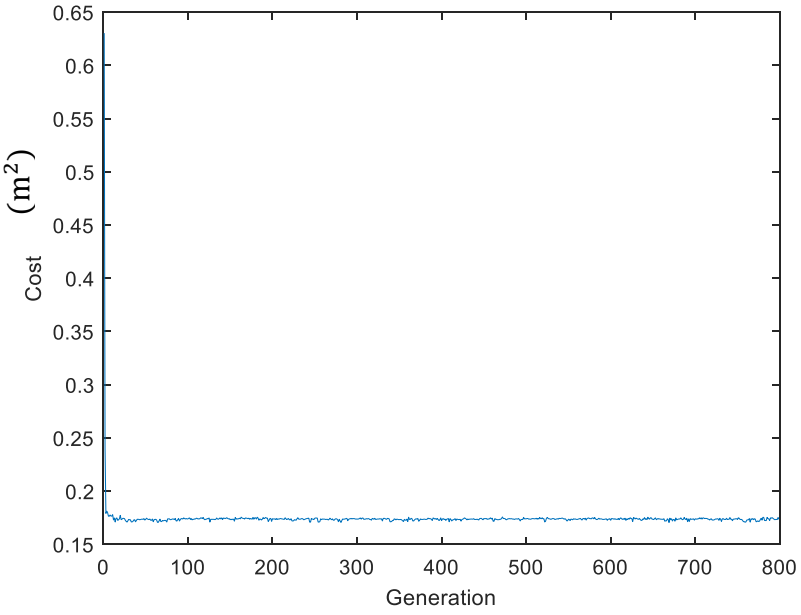


**Figure 46: Constrained GA optimization cost using healthy gait mid-tibia path**

Visually, the fit of the constrained model is better, but the cost functions tell a different story. There exists approximately a 70% decrease in performance with the constrained model.

This is most likely attributed to unconstrained system's flexibility. 9.5 contains the optimized paths and associated costs for the two gait pathologies drop foot and drag foot.

## 6.7 Unconstrained GA using contrived four-bar angular rates

The following sections deal with a genetic algorithm optimized using contrived data. The data came from a known four-bar system with a crank operating at a known, constant angular rate. This information was then used to compute contrived femur and tibia angular rates to use in the cost function of the genetic algorithm. Figure 47 and Figure 49 show the fit of the optimizations while Figure 48 and Figure 50 show the associated costs.

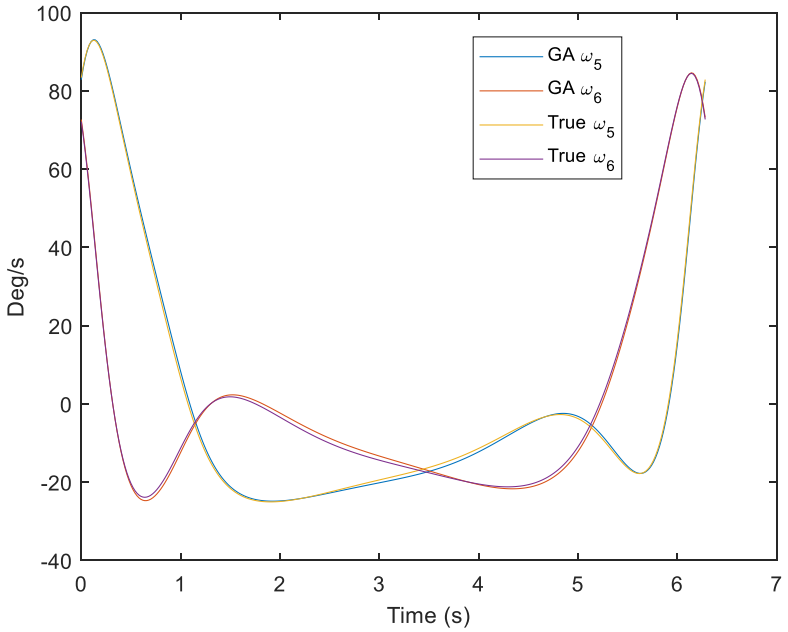


**Figure 47: Contrived unconstrained GA Optimization using four-bar system**

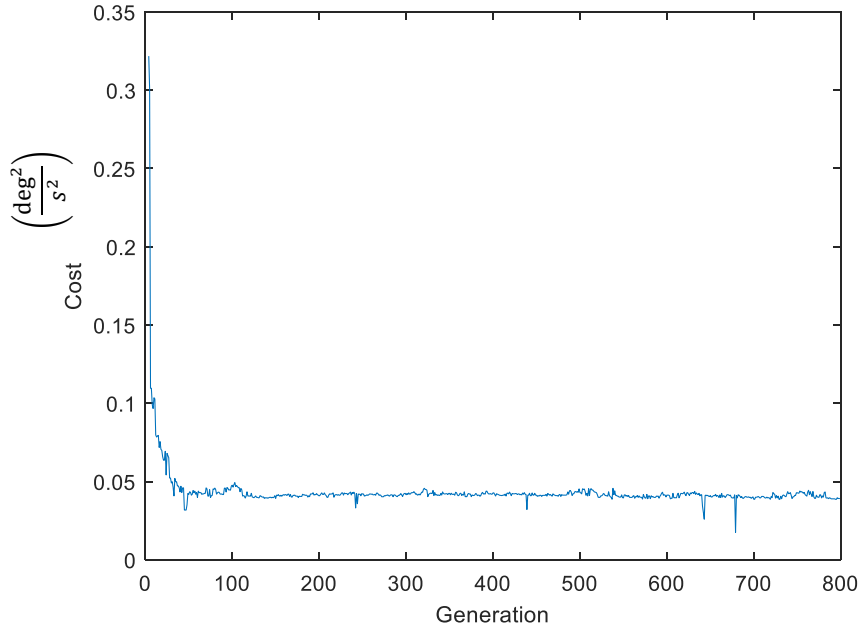


**Figure 48: Contrived unconstrained GA optimization cost using four-bar system**

## 6.8 Constrained Hoeken GA using contrived four-bar angular rates

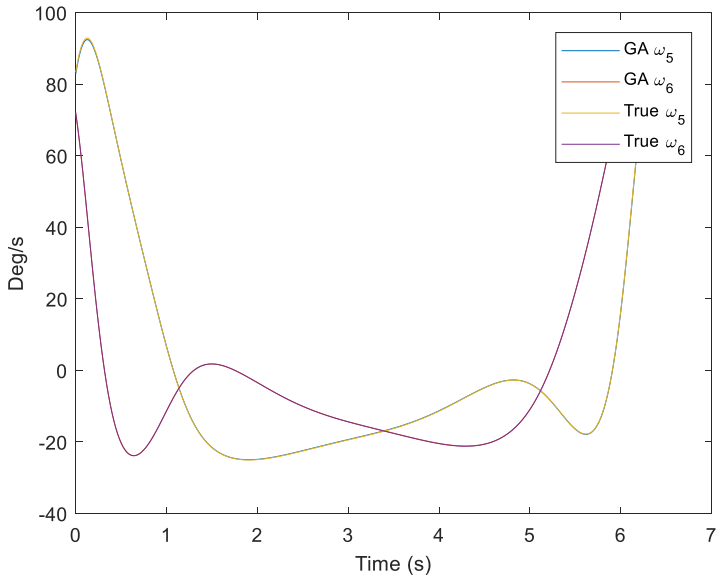


**Figure 49: Contrived Hoeken constrained GA optimization using four-bar system**

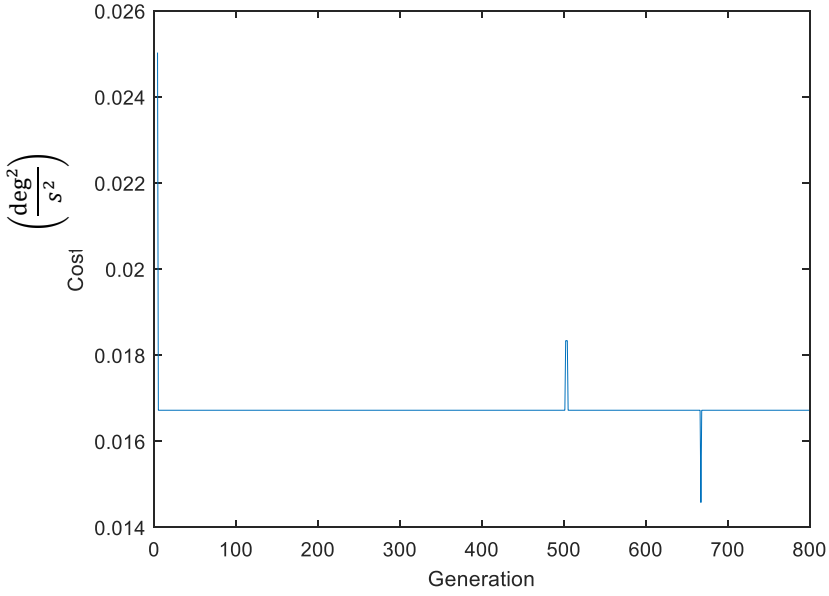


**Figure 50: Contrived Hoeken constrained GA optimization cost using four-bar system**

## 6.9 Unconstrained GA using motion capture angular rates

Following the same process as in previous sections, here the unconstrained GA was actually trained on the motion capture angular rates of the femur and tibia. Running for 800 generations and a population size of 1000, the results of the optimization are found in Figure 51

and Figure 52. 9.6 contains the optimized paths and associated costs for the two gait pathologies drop foot and drag foot.

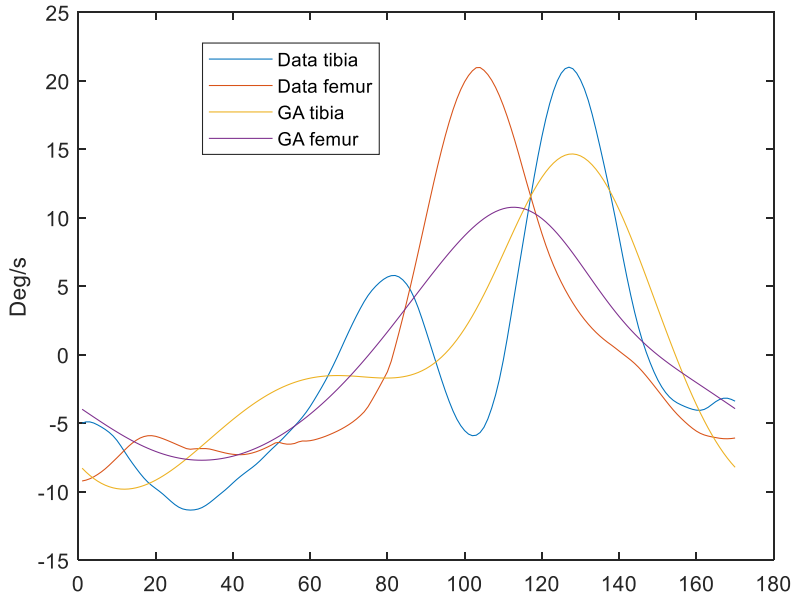


**Figure 51: Unconstrained GA optimization using healthy gait mid-tibia angular rates**

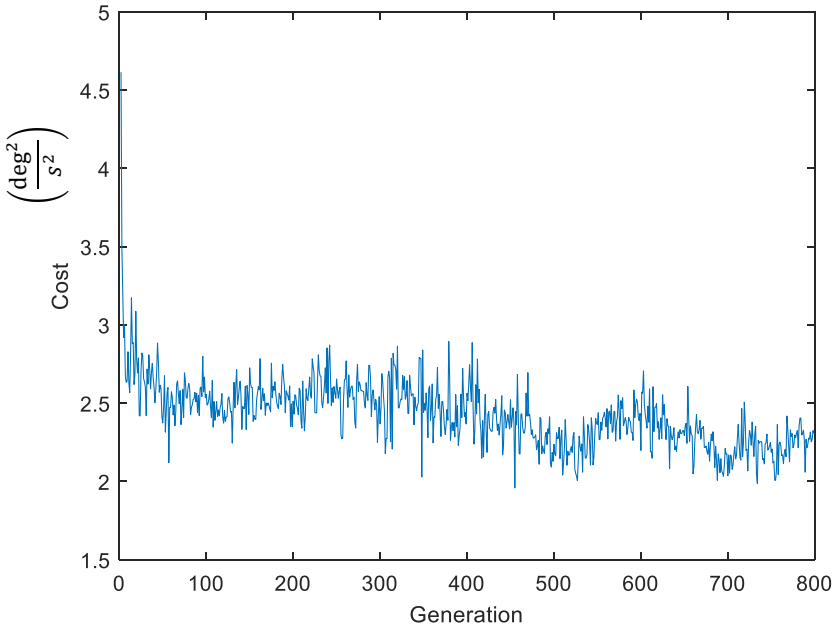


**Figure 52: Unconstrained GA optimization cost using healthy gait mid-tibia angular rates**

The optimizations of the angular rates do not fit the real data with a low cost. Each member of the population in the unconstrained optimization had 11 variables to be optimized. This causes the "noisy" cost function. If this work is to be improved, using a larger population size with more generations and more computing power would be necessary.

## 6.10 Constrained Hoeken GA using motion capture angular rates

The last GA optimization was performed on the same set of femur and tibia angular rates but constrained to satisfy Hoeken geometries. The results of the optimization are found in Figure 53 and Figure 54. Appendix 9.7 depicts the optimized paths and associated costs for the two gait pathologies drop foot and drag foot. Similar to the unconstrained case, the constrained case had 7 variables to be optimized, many of which depend on each other and cause the shaky cost plot.

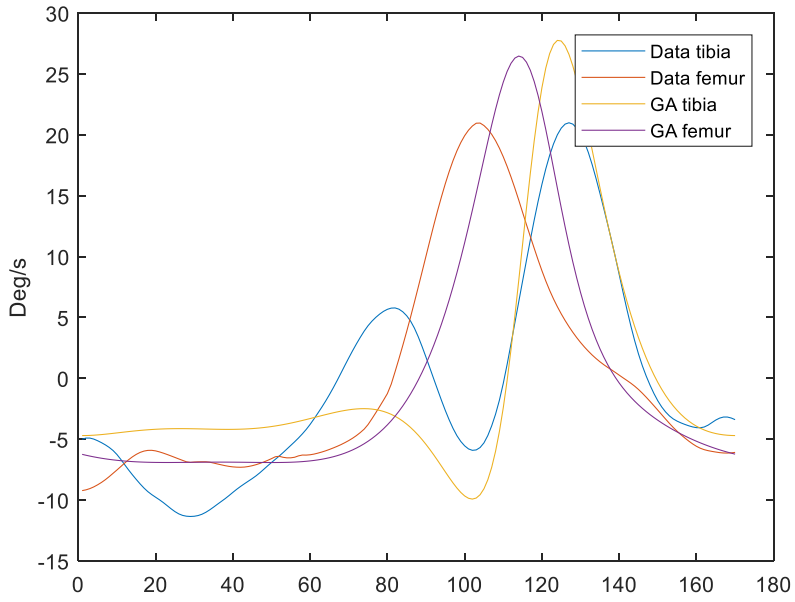


**Figure 53: Constrained Hoeken GA optimization using healthy gait angular rates**

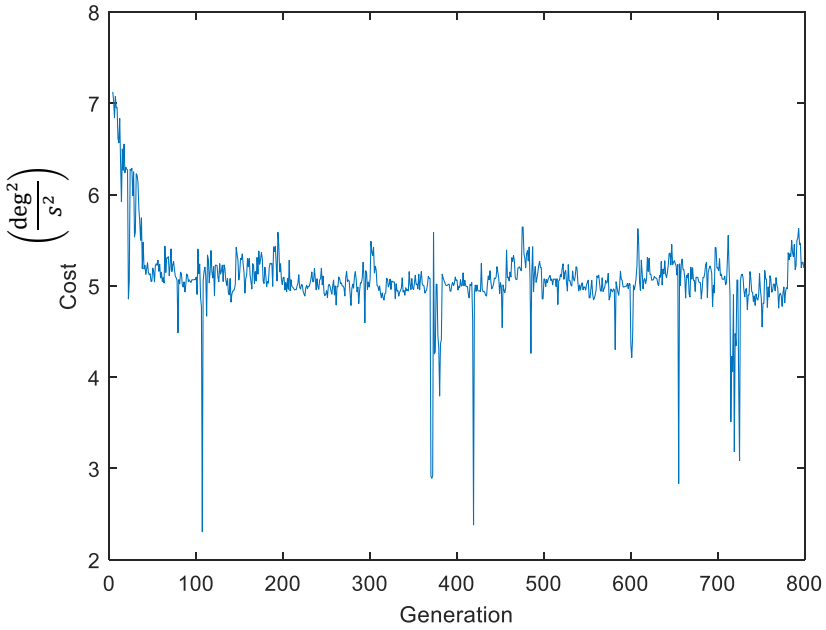


**Figure 54: Constrained Hoeken GA optimization cost using healthy gait angular rates**

The GA trained on the angular rates of the femur and tibia did not offer as low of a cost function as desired. Both unconstrained and constrained optimizations offered about the same cost function when looking at the fittest individual in the population. Note that for all these trials, a different type of four-bar system was attempted. Rather than having the end effector of the Hoeken system, point e, be on the same line as point b, an angular offset was simulated in hopes of achieving a better cost by giving the algorithm more freedom. This method proved to be unsuccessful.

## 6.11 GA four-bar system link lengths

Optimized four-bar system link lengths were also documented. In the case of the motion capture ankle paths, Table 3 and Table 4 show how both the unconstrained and constrained systems compare.

**Table 3: Unconstrained GA optimized four-bar lengths using ankle path**

| Link (m) | Healthy | Drop Foot | Drag Foot |
|---|---|---|---|
| **a** | 0.118 | 0.0593 | 0.1129 |
| **b** | 0.5287 | 0.4855 | 0.5260 |

| | | | |
|---|---|---|---|
| **c** | 0.4599 | 0.3648 | 0.4848 |
| **d** | 0.2043 | 0.1827 | 0.3553 |
| **e** | 0.354 | 0.2413 | 0.6137 |

**Table 4: Constrained GA optimized four-bar lengths using ankle path**

| Link (m) | Healthy | Drop Foot | Drag Foot |
|---|---|---|---|
| **a** | 0.1489 | 0.0690 | 0.0916 |
| **b** | 0.3723 | 0.1724 | 0.2291 |
| **c** | 0.3723 | 0.1724 | 0.2291 |
| **d** | 0.2978 | 0.1379 | 0.1833 |
| **e** | 0.7445 | 0.3448 | 0.4582 |

If healthy gait is assumed to be the correct mode of walking, the percent difference in GA optimized link lengths are found in Table 5 and Table 6.

**Table 5: Percent difference in unconstrained GA optimized length lengths using ankle path**

| Link (m) | Healthy | Drop Foot | Drag Foot |
|---|---|---|---|
| **a** | 0.118 | -66% | -4.4% |
| **b** | 0.5287 | -8.5% | < -1 % |
| **c** | 0.4599 | -23% | 5.3% |

| | | | |
|---|---|---|---|
| **d** | 0.2043 | -11% | 54% |
| **e** | 0.354 | -38% | 54% |

**Table 6: Percent difference in constrained GA optimized length lengths using ankle path**

| Link (m) | Healthy | Drop Foot | Drag Foot |
|---|---|---|---|
| **a** | 0.1489 | -73% | -48% |
| **b** | 0.3723 | | |
| **c** | 0.3723 | | |
| **d** | 0.2978 | | |
| **e** | 0.7445 | | |

Illustrated in Table 7 is the ratio of all components relative to link $a$ for the unconstrained optimization.

**Table 7: Relative lengths of system compared to link a using ankle path**

| Link (m) | Healthy | Drop Foot | Drag Foot |
|---|---|---|---|
| **a** | 1.00 | 1.00 | 1.00 |
| **b** | 4.48 | 8.19 | 4.66 |
| **c** | 3,90 | 6.15 | 4.29 |
| **d** | 1.73 | 3.08 | 3.15 |
| **e** | 3.00 | 4.07 | 5.44 |

When using the mid-tibia path to train the GA, Table 8 and Table 9 show the results when the system is unconstrained and constrained, respectively.

**Table 8: Unconstrained GA optimized four-bar lengths using mid-tibia path**

| Link (m) | Healthy | Drop Foot | Drag Foot |
|---|---|---|---|
| **a** | 0.1186 | 0.0656 | 0.0477 |
| **b** | 0.6433 | 0.5246 | 0.6521 |
| **c** | 0.4295 | 0.4343 | 0.6346 |
| **d** | 0.3552 | 0.1827 | 0.0963 |
| **e** | 0.1793 | 0.1820 | 0.2299 |

**Table 9: Constrained GA optimized four-bar lengths using mid-tibia path**

| Link (m) | Healthy | Drop Foot | Drag Foot |
|---|---|---|---|
| **a** | 0.0795 | 0.0548 | 0.0460 |
| **b** | 0.1986 | 0.1370 | 0.1150 |
| **c** | 0.1986 | 0.1370 | 0.1150 |
| **d** | 0.1589 | 0.1096 | 0.0920 |
| **e** | 0.3973 | 0.2739 | 0.2300 |

If, again, healthy gait is assumed to be the correct method of walking, the percent difference in GA optimized length lengths is found in Table 10 and Table 11.

**Table 10: Percent difference in unconstrained GA optimized length using mid-tibia path**

| Link (m) | Healthy | Drop Foot | Drag Foot |
|---|---|---|---|
| **a** | 0.1186 | -58% | -85% |
| **b** | 0.6433 | -20% | 1.4% |
| **c** | 0.4295 | 1.1% | 39% |
| **d** | 0.3552 | -64% | -115% |
| **e** | 0.1793 | 1.5% | 25% |

**Table 11: Percent difference in constrained GA optimized length using mid-tibia path**

| Link (m) | Healthy | Drop Foot | Drag Foot |
|---|---|---|---|
| **a** | 0.0795 | -37% | -54% |
| **b** | 0.1986 | | |
| **c** | 0.1986 | | |
| **d** | 0.1589 | | |
| **e** | 0.3973 | | |

Similar to the ankle path method, Table 12 shows the ratio of all components relative to link $a$ for the unconstrained optimization case.

**Table 12: Relative lengths of system compared to a using mid-tibia path**

| Link (m) | Healthy | Drop Foot | Drag Foot |
|---|---|---|---|
| **a** | 1.00 | 1.00 | 1.00 |
| **b** | 5.42 | 8.00 | 13.7 |
| **c** | 3.62 | 6.62 | 13.3 |
| **d** | 2.99 | 2.79 | 2.02 |
| **e** | 1.51 | 2.77 | 4.82 |

# 7. CONCLUSION

The next three sections deal with the future of this work and the final section concludes the research.

## 7.1 Implement GA models alongside Kalman Filter

The ideal use case for this system is implemented with a Kalman Filter. When training the GA on strictly path data, the model knows nothing about the angular rate at which the input crank (link a) rotates. One of the assumptions made earlier may not hold true: the input crank spins at a constant angular rate. Given the complexity of the human body, the likelihood of anything rotating at a constant rate is small which may result in an even more complex model that is presented here. To place this model inside a Kalman Filter, a good estimate of the crank input would need to be known, but this can be explored with more data collection.

## 7.2 Build IMU system to collect real angular rate data

The initial goal of this research was to use angular rate and acceleration data from gyroscopes and accelerometers to create a wearable device that could replace current motion capture technology. After it was discovered the Kalman filter needed a better gait model, the scope shifted to prove whether or not a Genetic Algorithm could recreate a subject's gait given only position information of landmark features. Although the motion capture angular rate data was good, moving away from motion capture completely is the best direction to proceed so as to train the GA on IMU data rather than motion capture in hopes of different results. An Arduino with off-the-shelf IMUs would allow a GA to be trained on true IMU data and give another depth to the research.

## 7.3 Collect more real gait data to test GA on a larger sample size

More data is always better and using the data of one individual gives a great starting point for future research. More data for the GA to be trained and tested on will determine the overall reliability of the system as a dynamic and flexible model generator.

## 7.4 Train a neural network to automatically classify gait

Building upon Section 7.3 is collecting enough data to train a neural network (NN) on the GA output. The GA can output as many parameters as need be estimated and those parameters can then act as inputs to a NN. With enough data, NNs have the power to "see" and understand trends in data in a way humans have yet to perfect. Constructing a NN would be adding depth to what is found with both motion capture and GAs. Knowledge of the ratios of lengths and percent differences in the links described in Table 11 would be a great start to training a model, especially if more data was collected.

## 7.5 Conclusion

Contrived data and kinematics that resemble human gait have been implemented within an Extended Kalman Filter. The filter, even with noisy data, converged to the true value of the states and showed that, given a reasonably accurate gait model, convergence is possible with off the shelf Inertial Measurement Units. The problem was realized to be the lack of a real gait model suitable for Kalman Filter implementation. The goal of this research was to create a real model capable of accurately predicting the states in real time using real data.

Gait data, both healthy and pathological, has been collected and used to train an intelligent model by a Genetic Algorithm. The use of a constrained Hoeken system for the ankle path was a better path assumption than an unconstrained model. For the mid-tibia, the opposite was true. If using mid-tibia position data, the unconstrained system had a lower final cost. Finally, when using angular rate data in the optimization cost function, the unconstrained model proved to be better here too. Although mediocre, the cost does show the algorithm was approaching a lower cost.

For the first time, a human gait model has been driven by an underlying four-bar mechanism optimized by a genetic algorithm. The potential impact of these findings are limitless and the idea that patients can have a tailor fit model is within reach. BlackTop Labs can use these findings to move forward with assurance that a Kalman Filter can converge for cyclic cycles resembling gait and a Genetic Algorithm can be used as a starting point for generating a nonlinear kinematic model for the filter.

# 8. REFERENCES


[1] "National Cancer Institute SEER Training Modules," [Online]. Available: https://training.seer.cancer.gov/anatomy/body/terminology.html#planes. [Accessed Mar 2018].

[2] "Basic Human Anatomy," Dartmouth Medical School, 2009. [Online]. Available: http://www.dartmouth.edu/~humananatomy/figures/chapter_1/1_1.HTM. [Accessed Mar 2018].

[3] P. Levangie and C. Norkin, Joint structure and function, F. A. Davis Company, 2011.

[4] "Hip and Pelvis Mcvay Physical Therapy," [Online]. Available: http://www.thepinsta.com/hip-joint-bones_keuU*qwaeTb5m3hyVqsmgrCMyrE0KyGGh%7COr2d9bbH4/. [Accessed Mar 2018].

[5] Los Amigos Research and Education Institute, Inc., Observational gait analysis, Downey: Rancho Los Amigos National Rehabilitation Center, 2001.

[6] "A Star Maths and Physics," [Online]. Available: https://astarmathsandphysics.com/a-level-physics-notes/medical-physics/2845-artificial-knees.html. [Accessed Mar 2018].

[7] "Openi," [Online]. Available: https://openi.nlm.nih.gov/detailedresult.php?img=PMC2323000_1546-0096-6-6-1&req=4. [Accessed Mar 2018].

[8] "Neupsy Key," [Online]. Available: https://neupsykey.com/gait-disorders-3/. [Accessed Mar 2018].

[9] T. Bennett, R. Jafari and N. Gans, "An Extended Kalman Filter to Estimate Human Gait Parameters and Walking Distance," in *American Control Conference*, Washington, DC, 2013.

[10] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," University of North Carolina at Chapel Hill, Chapel Hill, 2006.

[11] F. L. Lewis, L. Xie and D. Popa, Optimal and Robust Estimation With an Introduction to Stochastic Control Theory, Boca Raton: Taylor & Francis Group, 2008.

[12] T. Jansen, "Theo Jansen's STRANDBEEST," [Online]. Available: http://www.strandbeest.com/index.php.

[13] K. Waldron and G. Kinzel, Kinematics, Dynamics, and Design of Machinery, Wiley, 2003.

[14] R. E. King, Computational Intelligence in Control Engineering, Marcel Dekker, 1999.

[15] Z. Michalewicz, Genetic Algorithms Data Structures = Evolution Programs, Springer, 1992.

# 9. APPENDIX

## 9.1 Pull toy GA

Average cost stabilized around generation 20 but stumbled upon a better solution around generation 250. Since the solution was stable from generation 20 until generation 250, a mutation most likely occurred around the 250 mark that gave a better solution to the problem. This is a case where mutation helped the algorithm.

Although the average cost of the population remained relatively constant after generation 20, the fittest member of the species actually lived in generation 440. This ankle path for this individual is found in Figure 55.

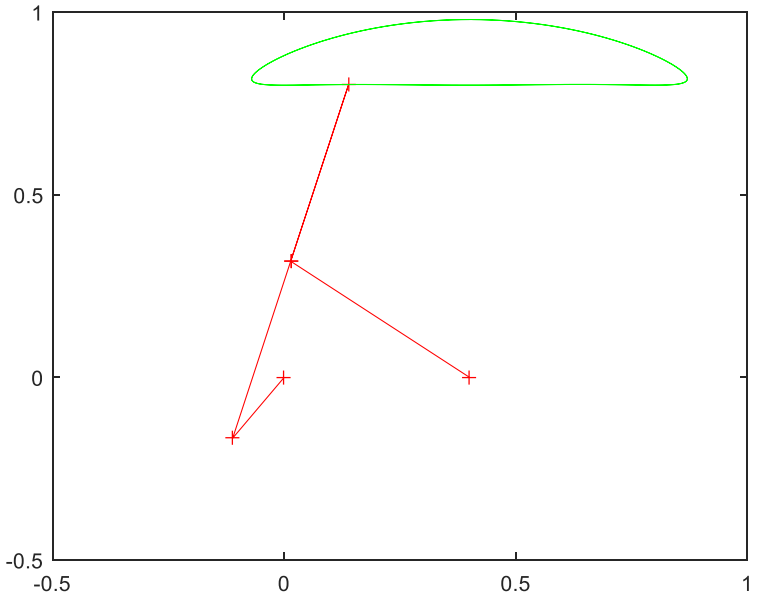


**Figure 55: GA optimization path for Hoeken system searching for links a and e**

This again shows the GA can handle two parameters. Loosening yet another parameter, link $c$ is also unknown. The convergence achieved for this search is shown in Figure 56.

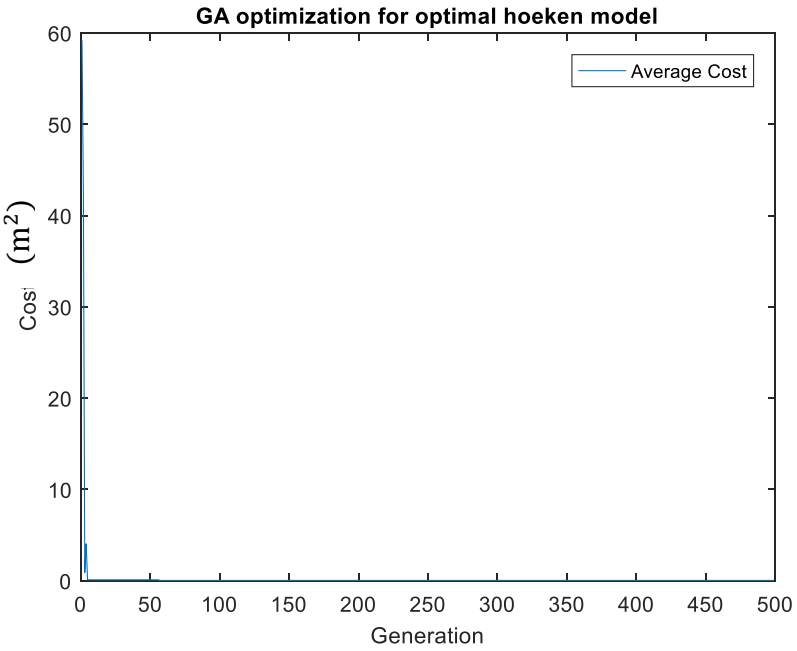


**Figure 56: GA optimization searching for links a, c, and e**

Average cost stabilized within the first 10 generations. Members are still getting increasingly fit until generation 375 which held the fittest member of the population. The fittest member gave the ankle trajectory shown in Figure 57.

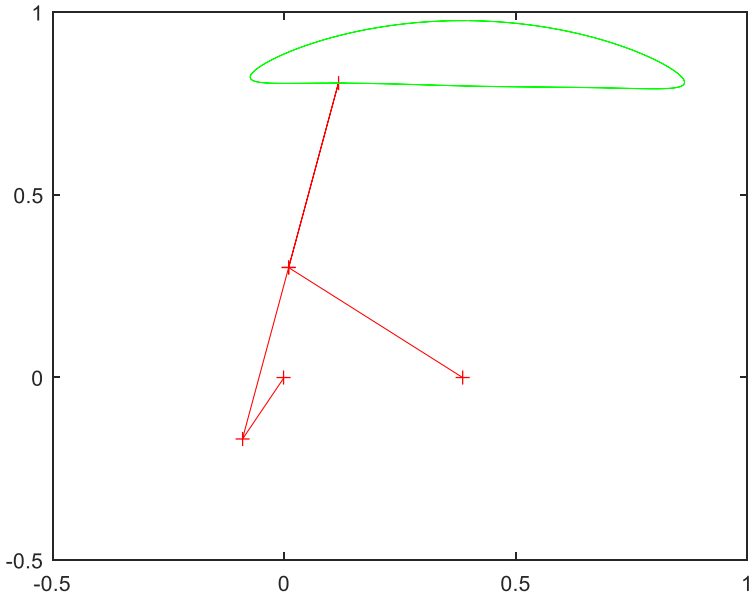


**Figure 57: GA optimization path for Hoeken searching for links a, c, and e**

This ankle trajectory looks to be slightly different than the ideal case, but three of the five parameters have been successfully estimated. The next search added more generations and increased the population size as well as added another parameter to search for. As more parameters are optimized, the problem gets more difficult at a nonlinear rate. The addition of more generations and a large population size takes care of the increasing difficulty.

The convergence achieved with estimating four parameters is shown in Figure 58. The convergence was not spectacular, but the solution did become asymptotic around 10 generations.

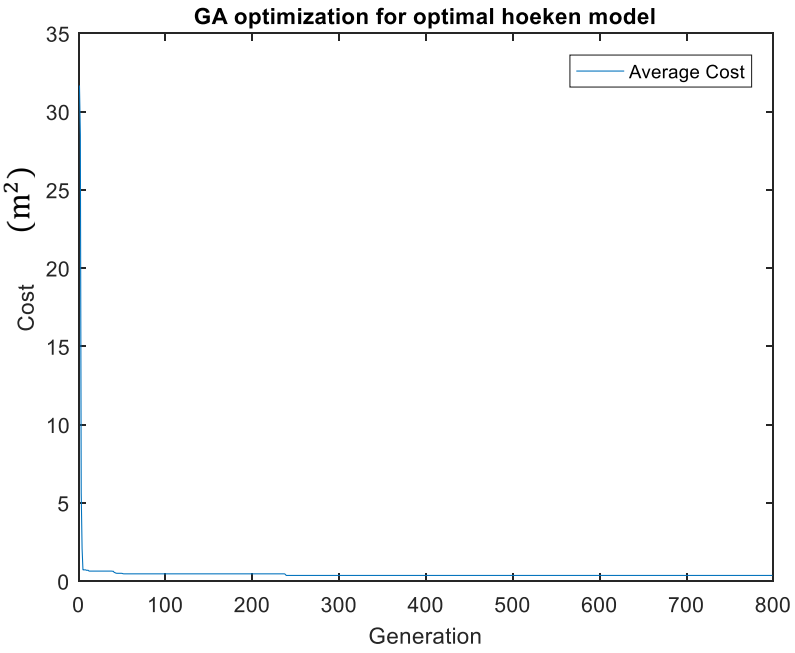


**Figure 58: GA optimization searching for links a, c, d, and e**

The fittest member of the population lived in generation 435, so increasing the maximum number of generations proved to be unnecessary in this case. The ankle trajectory created by this search is given by Figure 59.

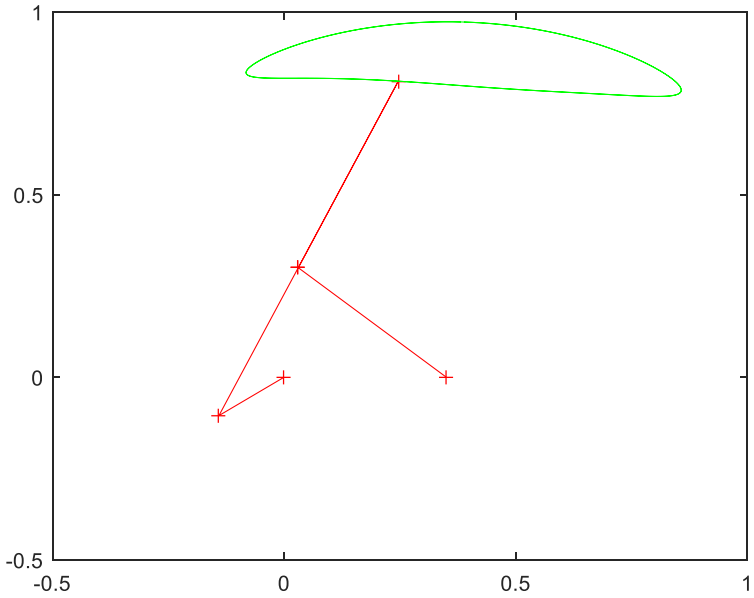


**Figure 59: GA optimization of Hoeken while searching for links a, c, d, and e**

This trajectory is clearly not perfect, but it still does resemble the ideal case. The idea is that a Kalman filter will be able to use this as a model and correct itself.

## 9.2 Unconstrained GA using motion capture ankle paths

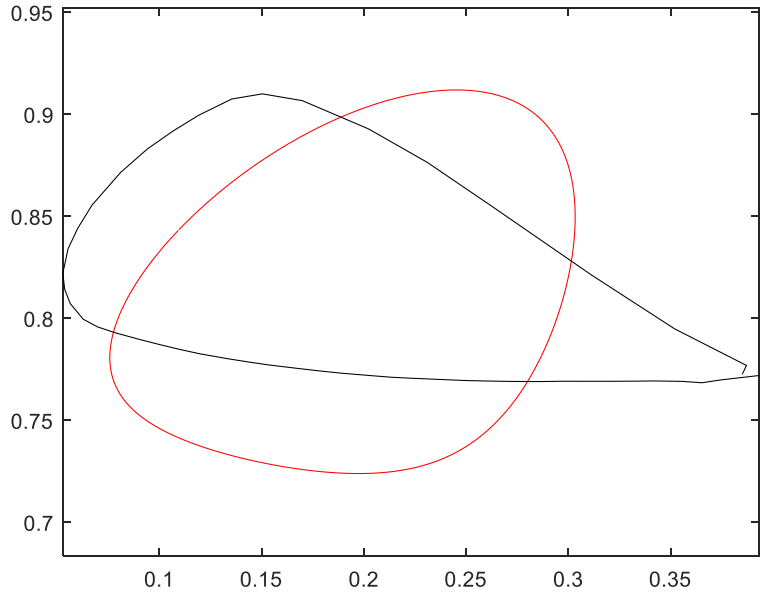


**Figure 60: Unconstrained GA optimization using drop foot ankle path**

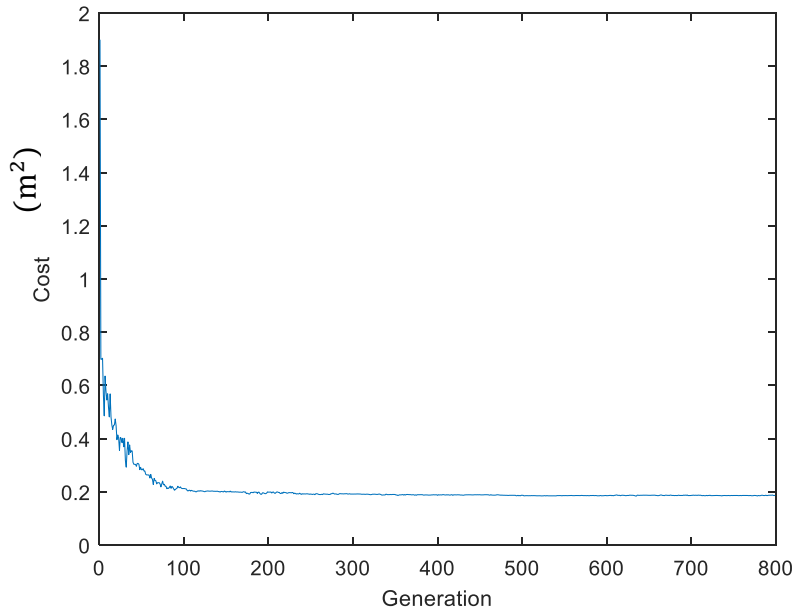


**Figure 61: Unconstrained GA optimization cost using drop foot ankle path**

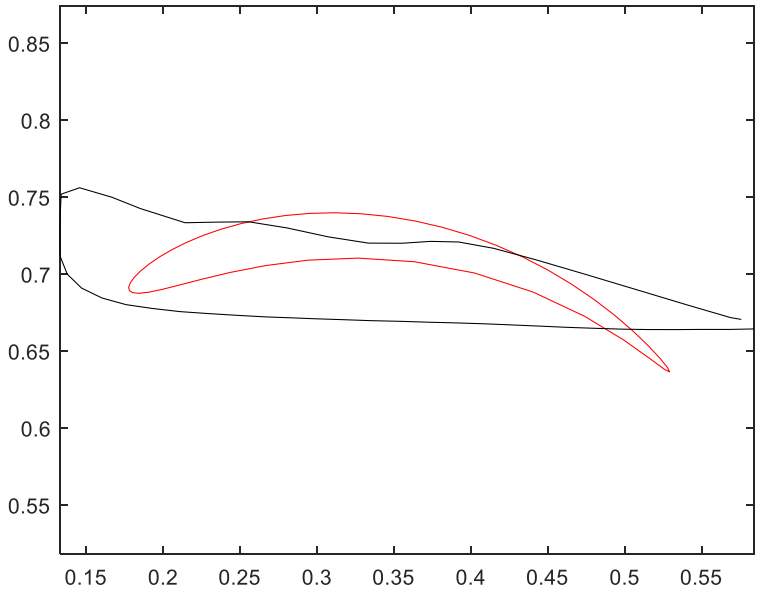


**Figure 62: Unconstrained GA optimization using drag foot ankle path**

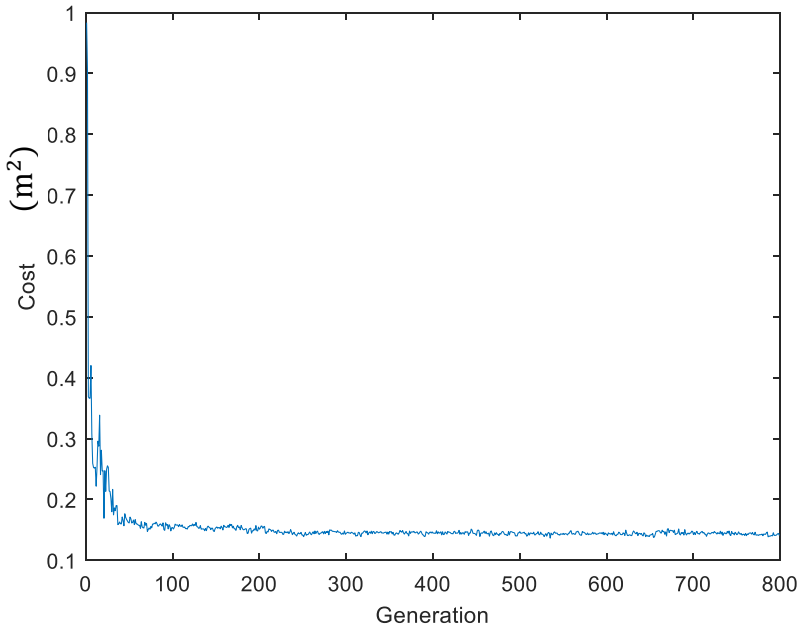


**Figure 63: Unconstrained GA optimization cost using drag foot ankle path**

## 9.3 Constrained Hoeken GA using motion capture ankle paths

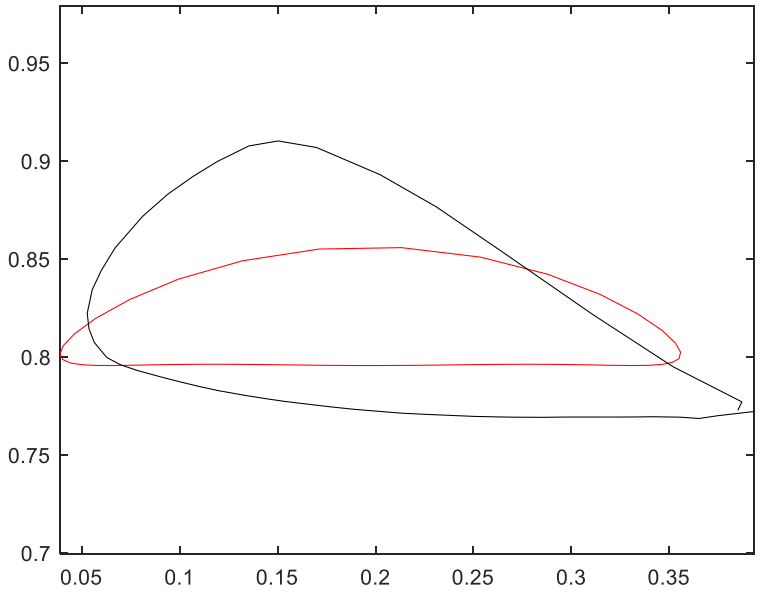


**Figure 64: Constrained Hoeken GA optimization using drop foot ankle path**

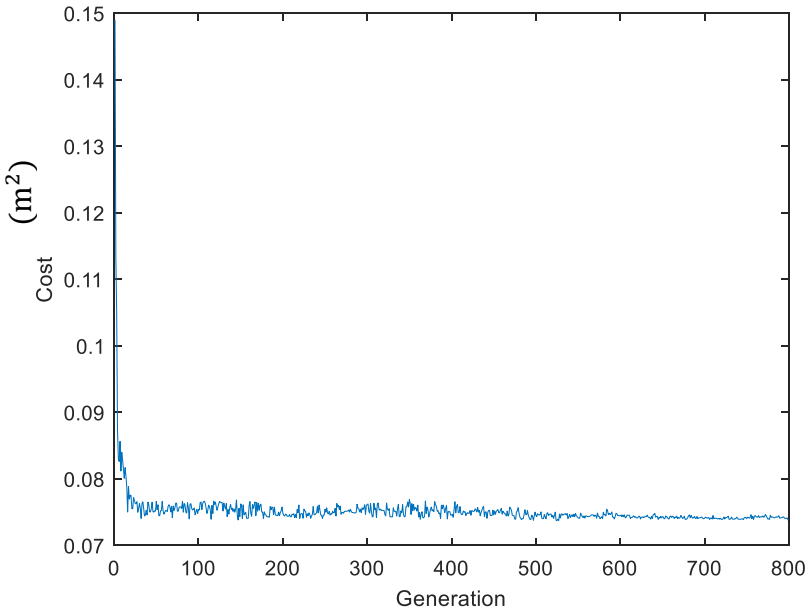


**Figure 65: Constrained Hoeken GA optimization cost using drop foot ankle path**

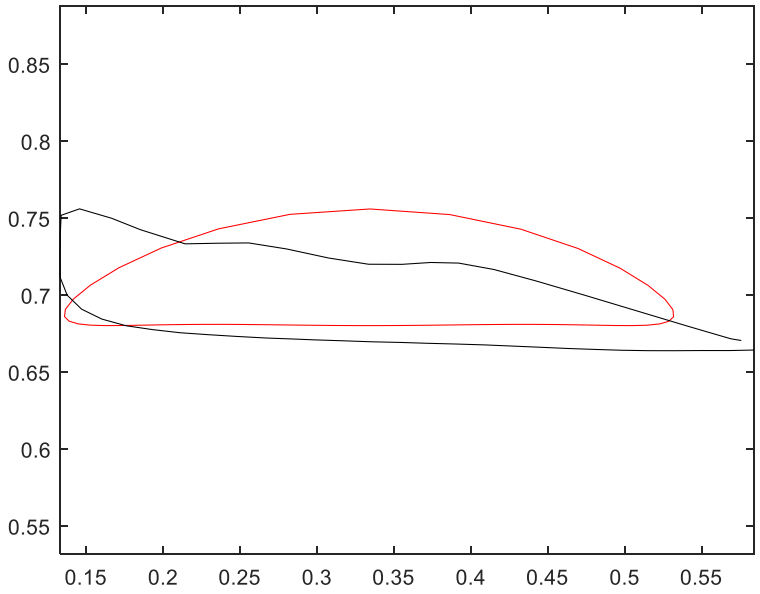


**Figure 66: Constrained Hoeken GA optimization using drag foot ankle path**

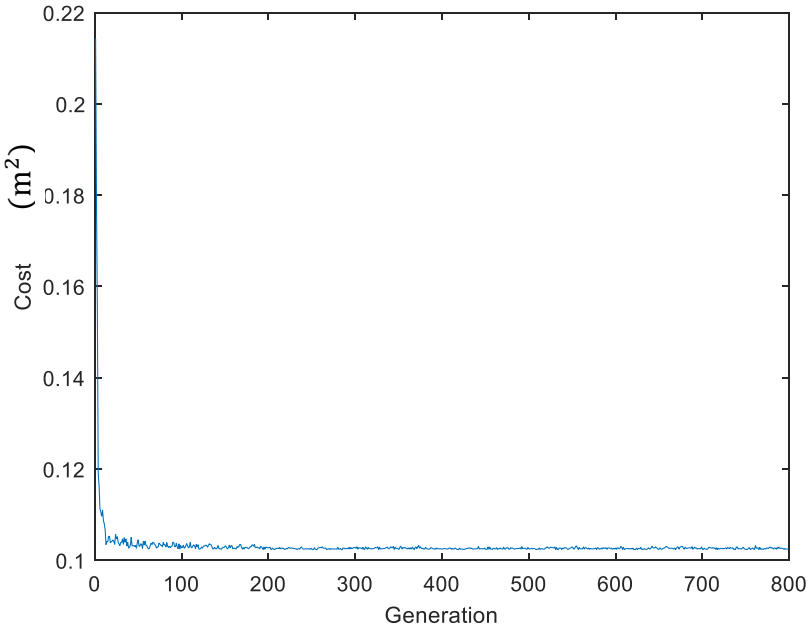


**Figure 67: Constrained Hoeken GA optimization cost using drag foot ankle path**

## 9.4 Unconstrained GA using motion capture mid-tibia paths

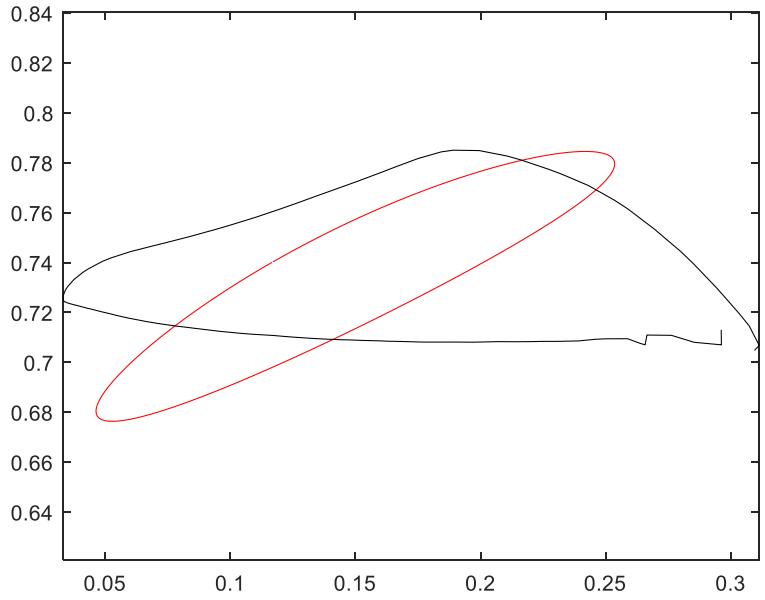


**Figure 68: Unconstrained GA optimization using drop foot mid-tibia path**

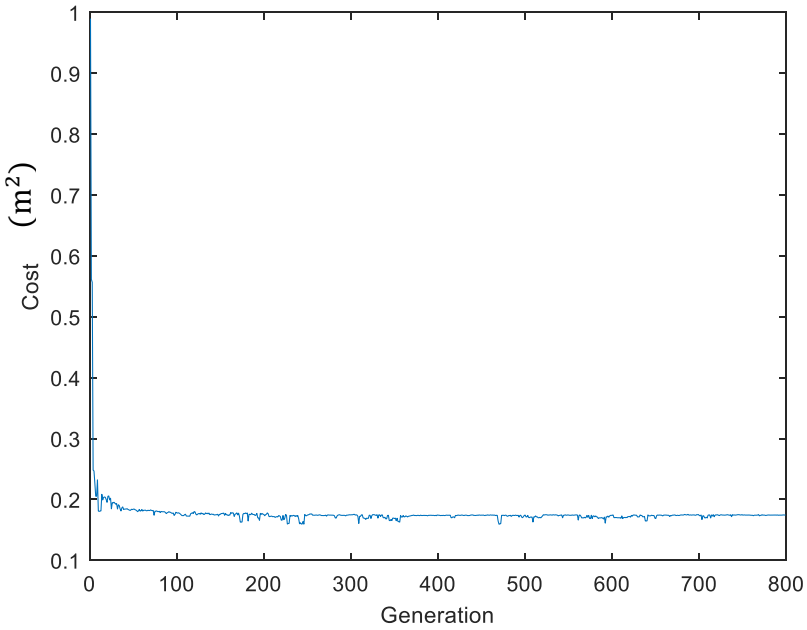


**Figure 69: Unconstrained GA optimization cost using drop foot mid-tibia path**

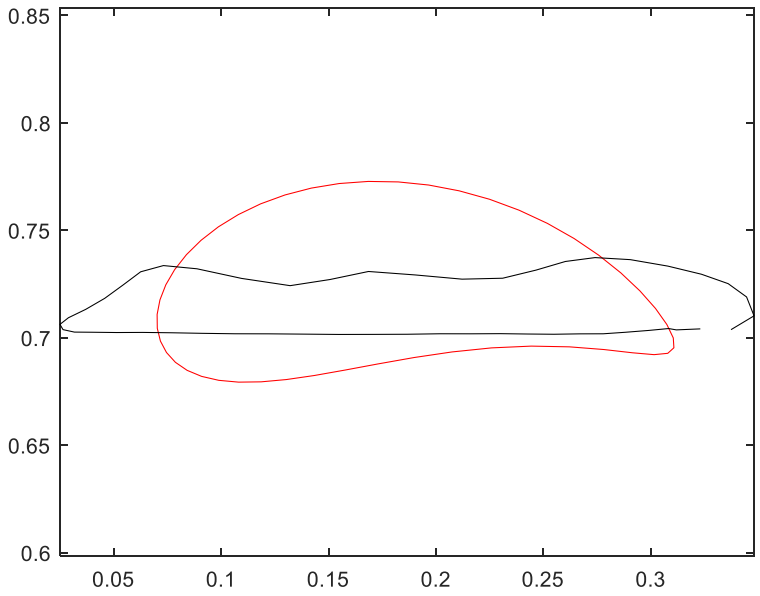


**Figure 70: Unconstrained GA optimization using drag foot mid-tibia path**

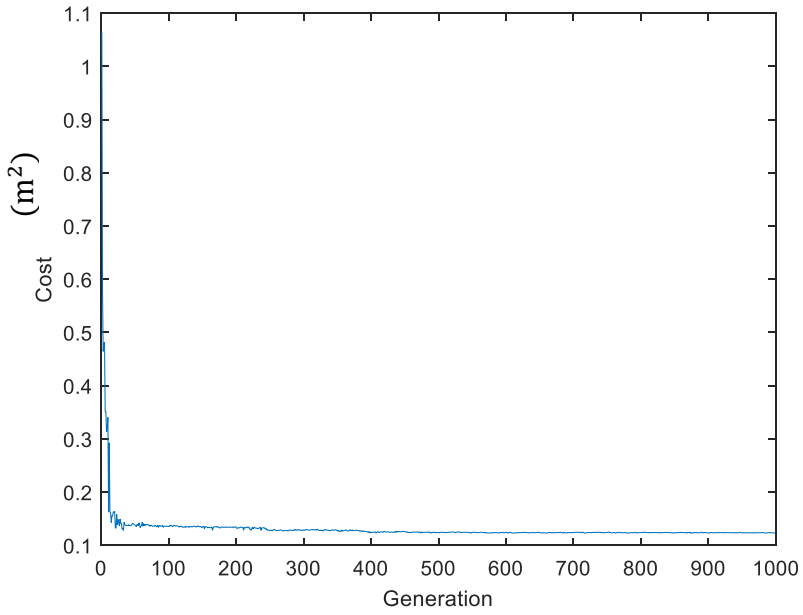


**Figure 71: Unconstrained GA optimization cost using drag foot mid-tibia path**

## 9.5 Constrained Hoeken GA using motion capture mid-tibia paths

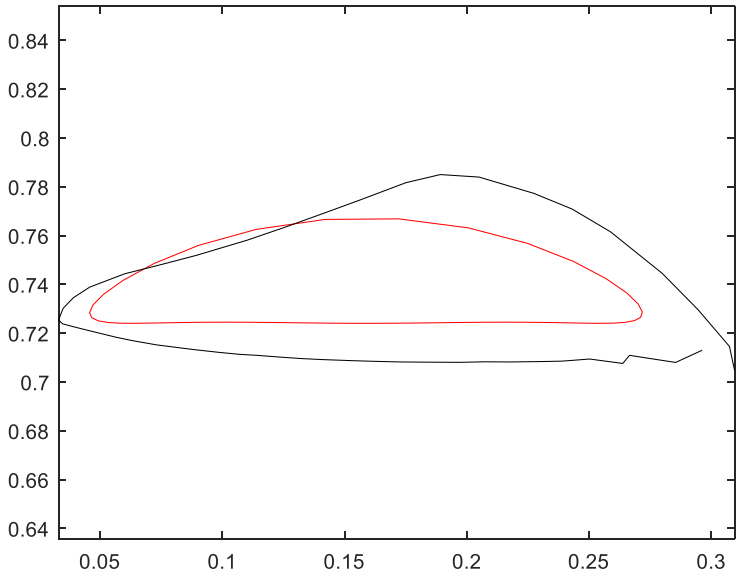


**Figure 72: Constrained GA optimization using drop foot mid-tibia path**

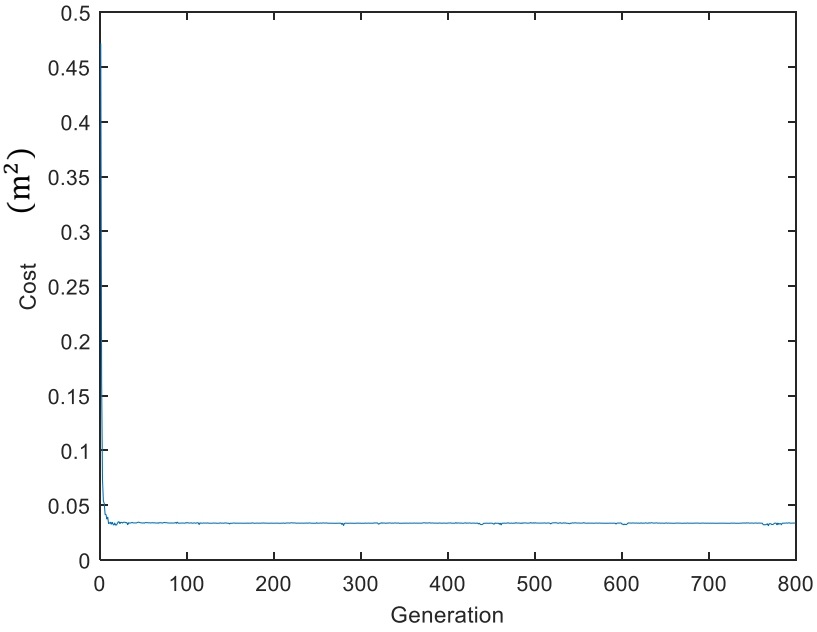


**Figure 73: Constrained GA optimization cost using drop foot mid-tibia path**

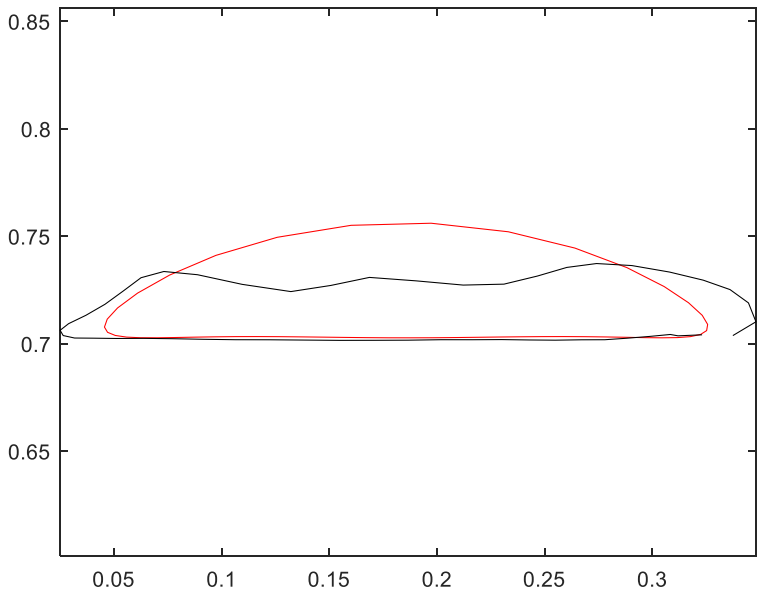


**Figure 74: Constrained GA optimization using drag foot mid-tibia path**

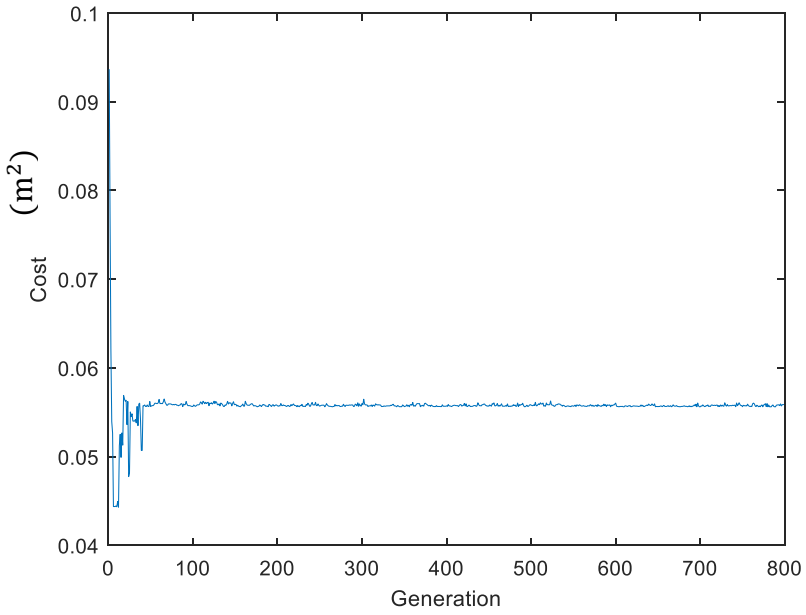


**Figure 75: Constrained GA optimization cost using drag foot mid-tibia path**

## 9.6 Unconstrained GA using motion capture angular rates

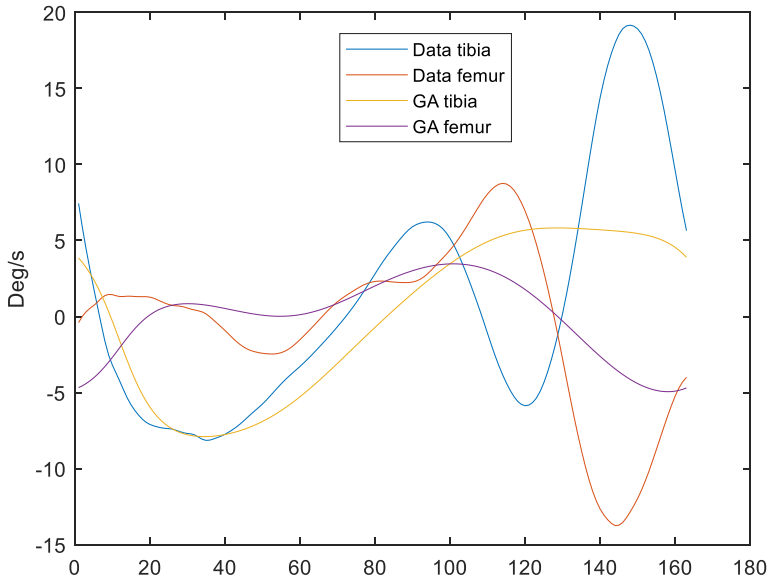


**Figure 76: Unconstrained GA optimization using drop foot motion capture angular rates**

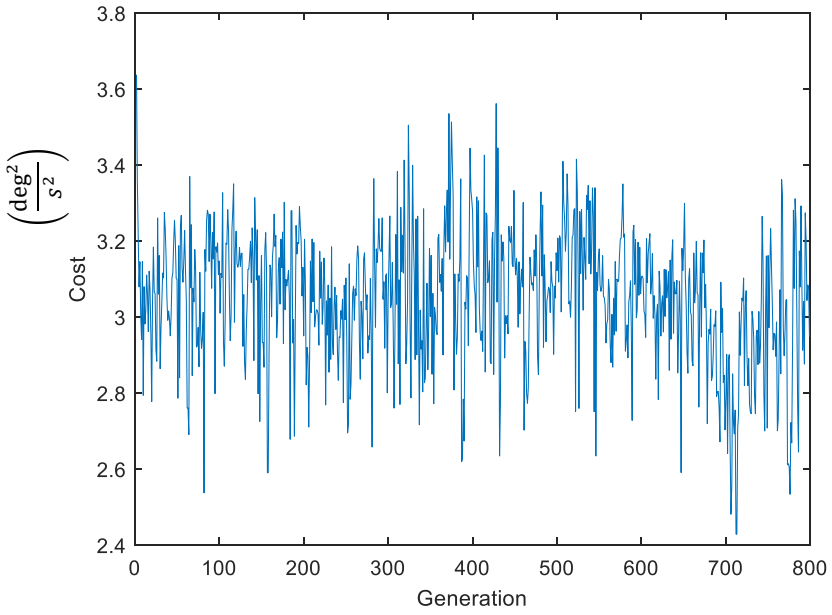


**Figure 77: Unconstrained GA optimization cost using drop foot motion capture angular rates**

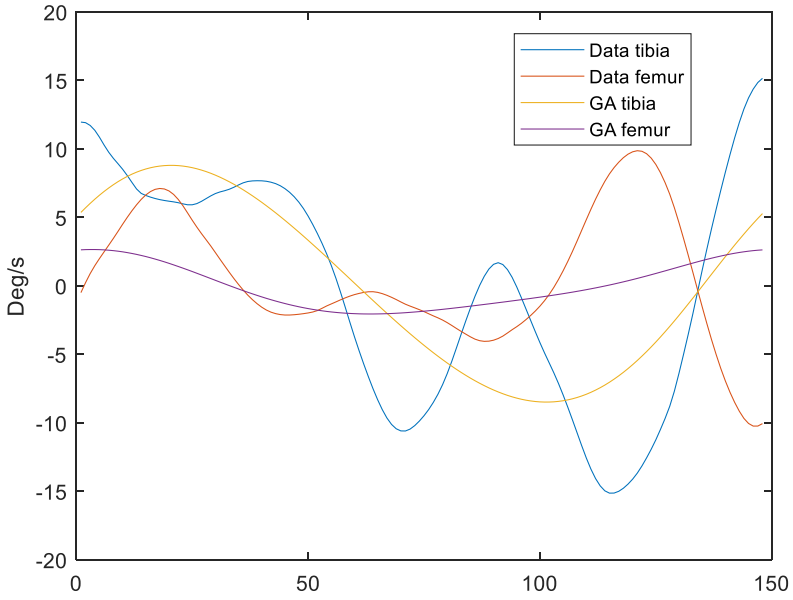


**Figure 78: Unconstrained GA optimization using drag foot motion capture angular rates**

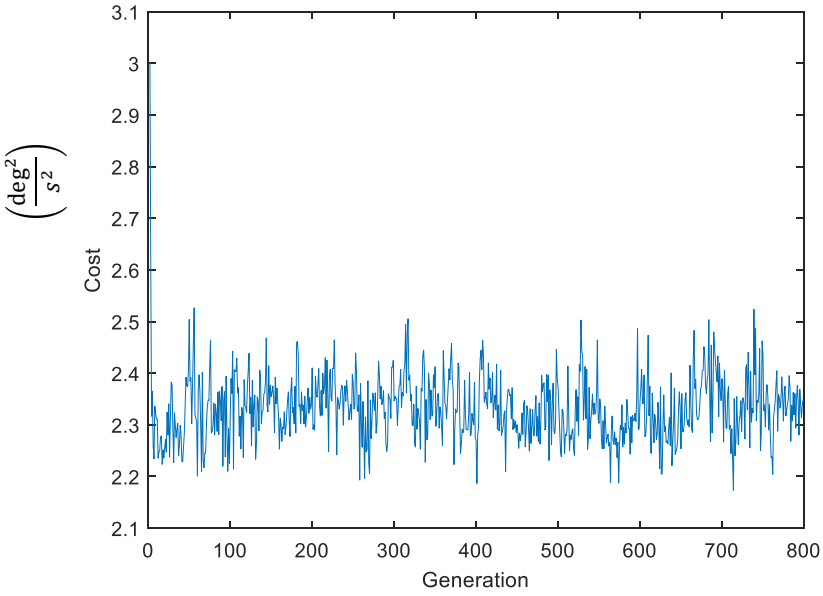


**Figure 79: Unconstrained GA optimization cost using drag foot motion capture angular rates**

## 9.7 Constrained Hoeken GA using motion capture angular rates

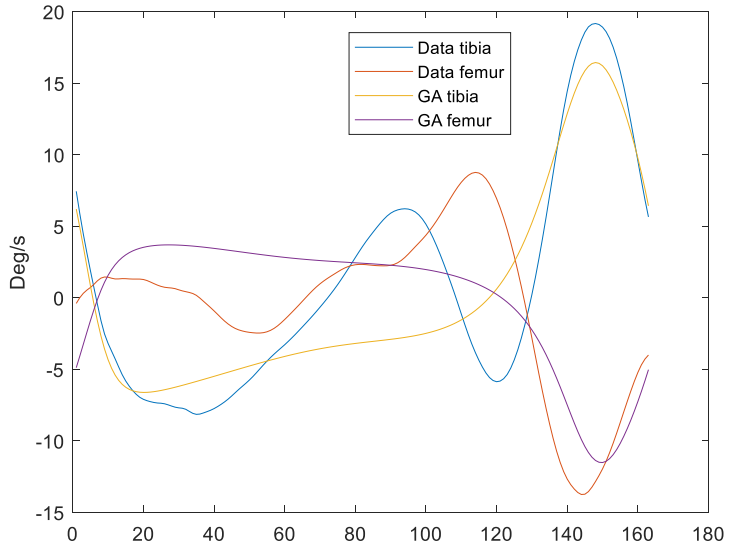


**Figure 80: Constrained Hoeken GA optimization using drop foot angular rates**

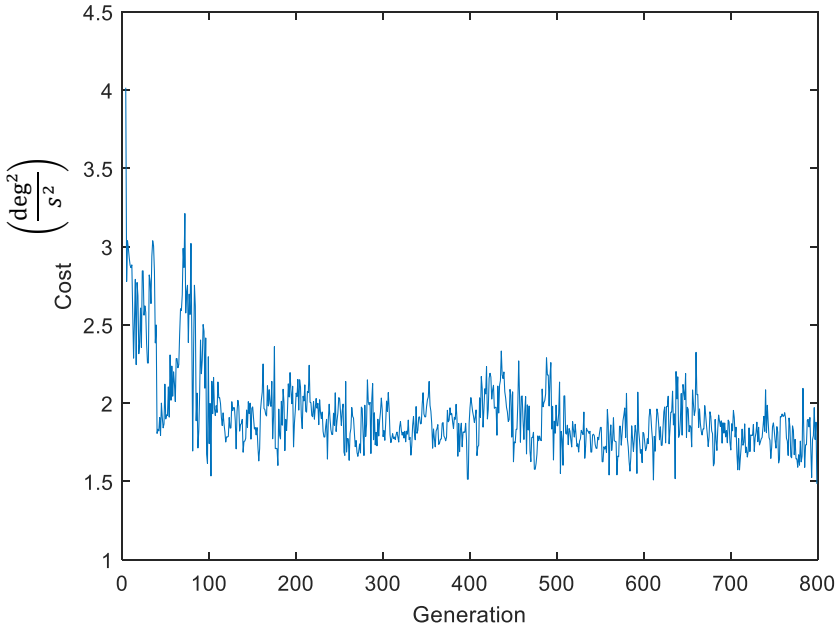


**Figure 81: Constrained Hoeken GA optimization cost using drop foot angular rates**

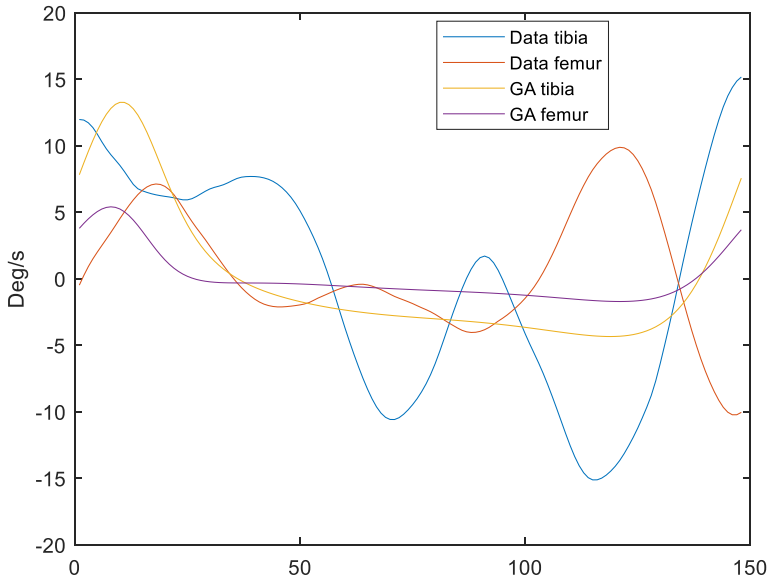


**Figure 82: Constrained Hoeken GA optimization using drag foot angular rates**

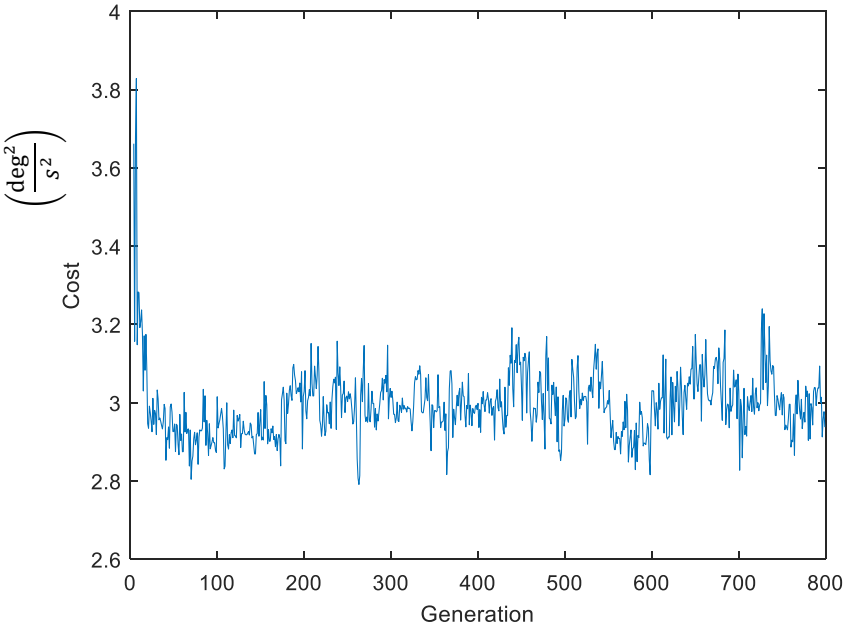


**Figure 83: Constrained Hoeken GA optimization cost using drag foot angular rates**