Master of Science in Engineering
MSc(Eng)

Kinematic Modeling and Dynamic Aspects of an Accelerating Quadruped

Casey van der Leek

Supervisor: Prof. Fred Nicolls
A dissertation submitted to the Department of Electrical Engineering,
University of Cape Town, in fulfilment of the requirements for the degree of
Master of Science in Engineering

Cape Town, March 2022

Declaration

I declare that this dissertation is my own, original and unaided work. It is being submitted for the degree of Master of Science in Engineering in Mechatronics at the University of Cape Town.

It has not been submitted previously for any degree or examination at any other university or institution.

I know the meaning of plagiarism and declare that all the work in the document, save for that which is properly acknowledged, is my own. This dissertation has been submitted to the Turnitin module (or equivalent similarity and originality checking software) and I confirm that my supervisor has seen my report and any concerns revealed by such have been resolved with my supervisor.

Signature of the Author

Cape Town, March 2022

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Prof. Fred Nicolls, for the informative discussions, constructive suggestions and advice throughout this extended research project. Also a special word of thanks to Stacey Shield and Dr. Callen Fisher for sharing some of their knowledge and the interesting conversations we had that related to my work.

Much appreciation goes to Alastair During and the technical staff for being so helpful in providing me with data and notes on the biomechanics experimental work that was undertaken at the Sports Science Institute of the University of Cape Town.

Thanks also goes to the Ann Van Dyk Cheetah Centre, Hartbeespoort, Pretoria for availing themselves to UCT and allowing the acquisition of the cheetah video data.

Appreciation also goes to Emmanuel Ali, Anicet Hounkanrin, Robyn Verrinder and Dr. Amir Patel.

Finally, thanks to Dr. Jonathan Glenday of the Department of Biomedical Engineering, UCT for sharing the doctoral work he did on the human shoulder joint.

Abstract

Bio-inspired robotics engineers look to the natural world for clues to aspects of motion dynamics and morphologies that may be incorporated in the design of these robots. The mimicking and transfer of these aspects of a live subject to a modern day robot is limited by the technologies available such as computational resources, materials engineering, mathematical modeling constraints and efficient systems engineering. With this in mind, a reasonable strategy is to reproduce the functionality of a subject with current technology.

A monocular camera and deep learning algorithm allow non-invasive image pose extraction of an accelerating cheetah subject, which is represented as a mechanism of rigid links interconnected by joints, and this information forms the data basis of subsequent operations. In addition, a non-linear least squares optimiser is formulated and coded specifically for the quadruped robot that produces estimates of the relative link angles, a base link length and trajectory of a reference point so that a three dimensional configuration evolution of the system is rendered.

A secondary consideration is the deployment of inverse kinematics to determine the end effector trajectory of the front leg, both in the real spatial frames and phase space domains, as well as the angular rates required for these target manifolds. The parameterised inverse kinematics models were also able to generate smooth task space trajectories to within acceptable tolerances of the target position and for a single, full gait the corresponding joint space trajectories were deemed to be sufficiently closed.

Contents

List of Figures

List of Tables

Acronyms

1 Introduction **1**

1.1 Preliminaries 1

1.2 Research Objectives 2

1.3 Outline of Work 3

1.3.1 Problems and Scope 3

1.3.2 Proposed Solution Plan 4

1.3.3 Project Content 6

2 Theoretical Fundamentals **7**

2.1 Convolutional Neural Networks 7

2.2 Vision Systems and Image Processing 8

2.2.1 Virtual Camera Model 8

2.2.2 Vision System 11

2.3 Polar Transforms 11

2.4 Vector Fields 13

2.5 Non-linear Least Squares (LSQ) Optimisation 15

2.5.1 Optimisation Mathematics 15

2.5.2 First-Order Optimality Measure 18

2.5.3 Function Evaluations 19
2.5.4 Damped Least Squares Parameter λ 19
2.6 Inverse Kinematics (IK) Methods 20
2.6.1 Least Norm Solution 21
2.6.2 Gradient Projection 21
2.6.3 Weighted Least Norm 22
2.6.4 Extended Jacobian 22
2.6.5 Damped Least Squares 24
2.7 Inverse Kinematics for Trajectory Generation 25
2.8 Overview 26

3 Systems and Metric Definitions 27

3.1 Configuration Definition 27
3.2 Coordinate Systems 30
3.3 Transforms 31
3.4 Phase States 32
3.5 Degree of Closure (DOC) for a Trajectory 32

4 Methodology 35

4.1 Data Acquisition 36
4.2 Objective Vector Functions for the Optimiser 37
4.3 Model Parameters 42
4.4 Motion Constraints 43
4.5 Project Elements and Work Flow 43
4.6 Related Work 44
4.6.1 Deep Learning and Pose Estimation 44
4.6.2 Modeling of Legged Robots 45
4.7 Overview 46

5 Experimental Setup, Data Acquisition and Implementation 47

5.1 Markerless Pose Extraction using a Deep Neural Network 48
5.1.1 Network Training 49
5.1.2 Joint Tracking 50
5.1.3 Coding 50
5.2 Angle Variables 54
5.2.1 Configurations 54
5.2.2 Camera 54
5.3 Link Lengths 54
5.4 Angle Transform Equations 55
5.5 Optimisation Strategies 56
5.6 Coding the Optimisation 57
5.7 Inverse Kinematics Modeling 60
5.8 Overview 61

6 Results, Findings and Model Evaluation 62

6.1 Image Pose Extraction 62
6.2 Least Squares Optimisation 65
6.2.1 Output Indicators 66
6.2.2 Matrix Data in Graphic Form 69
6.2.3 Configurations 72
6.2.4 Trajectories and Dynamics of Reference Joint 75
6.3 Vector Fields 83
6.4 Inverse Kinematics 85
6.4.1 Model Parameters 86
6.4.2 Trajectory Generation for End Effector Tracking 87
6.4.3 Trajectory Closure 95

7 Conclusions 101

7.1 Image Extraction by a Trained Network 101

7.2 Configuration Optimisation 101
7.2.1 Basic Data and Code Indicators 101
7.2.2 Vector Fields 102
7.2.3 Configuration Plots 102
7.3 Inverse Kinematics 103
7.3.1 Parameters 103
7.3.2 Trajectories 103

8 Recommendations on Future Work 105

8.1 Least Squares Optimisation 105
8.2 Inverse Kinematics 106
8.3 Future Research 106
8.3.1 Humanoids 107
8.3.2 Quadrupeds 107
8.3.3 Surgical and Service Robots 108
8.3.4 Autonomous Quadrupeds in Space Exploration 109

Bibliography 110

A Joint Space Trajectories 116

B Optimisation Functions 118

C Non Convergent Trajectories 119

D Vector Fields 121

E Quadruped Configurations 123

List of Figures

1 The Pneupard is an experimental light weight, biomimetic pneumatic powered quadruped similar to the proposed model presented in this publication (Image credit: Osaka University). 2

2 Camera model showing the virtual image generated from a real world 3D object. The desired pixel image is processed from the film image according to the physical properties of the actual camera being used. The respective origins of the frames are indicated by O, O_f and O_w. . 8

3 The simple linear geometry that is fundamental in obtaining the image coordinates of a point in space. 9

4 The variables associated with the polar transformation equations. The vector with magnitude r and direction θ describes a point P that traces out the curve C. 12

5 An arbitrary curve trajectory C, showing the radial and tangential components of the tangential velocity of a point P on C. 12

6 The vector field generation for the incremental movement of the robot link in (i) from time t_i to t_{i+1}. Schematics (ii), (iii) and (iv) show how the field is created for a generic $link_{j+1}$ as it undergoes five incremental movements which are relative to its prior $link_j$. The field is represented as a time line with the associated vectors at each interval. 13

7 Solution integral curve $x(t)$ with the associated vector field $F(x(t))$ at the specific solution points. 14

8 The vector $\mathbf{V_{uv}}$ is the projection (using polar transforms) of $\mathbf{r}$ onto the local UV axis of the parent link. In this way relative motions can be displayed for all links and the sequence of vectors obtained thus generating a field. 15

9 A hypothetical non-linear vector function with estimated solution vector and actual data points for a model with two parameters x_1 and x_2. The green dot shows the actual solution point x and the red point is the solvers' estimate x_{est}. Blue dots represent a few of the known data points a_i that are used by the optimisation to achieve the estimate. 17

10 Another illustrative two parameter model example and how the initial guesses of the solution vector at the start of the algorithm search may affect the outcome. The initial search position for the red path results in a sub-optimal solution and this is confirmed by the magnitude of the squares of the residuals at this solution. The other two searches terminate at near optimal solutions (blue and brown paths). The actual solution is shown by the green square. 18

11 Universal (U) and spherical (S) joints. All joints for the quadruped serial link mechanism are based on universal joints. If spherical joints were to be used an additional rotational axis would be introduced allowing the links to twist relative to their parent links. 28

12 The quadruped 8-link configuration that represents the animal subject's locomotion generating limbs and appendages. Translational motion is directed from left to right with the reference joint J#0 at the junction of link r_5 (spine) and r_6 (back leg) of the mechanism. All angles are measured relative to their parent link with θ_9 being measured relative to a horizontal reference line, the U axis of the local robot system the origin of which is the reference joint. 29

13 All the descriptor planes that relate the local link motion of the quadruped with respect to the reference joint ($J\#0$). The three planes are the sagittal (UV) side, transverse (UW) top elevation and frontal (VW) planes. The permitted angle rotations θ and ϕ are shown in their operational planes. Link twist angles γ are all assumed zero for simplicity. 29

14 The canonical local robot (UVW) coordinate system showing the angle definitions in their respective planes for link rL_i. The origin of the UVW frame is the reference joint. Link twist angles are not modeled so γ is zero for all links. 30

15 The camera and local robot coordinate systems. The camera rotations about each of the XYZ axes are ξ, ψ and ζ respectively. The first spine link coincides with the U axis of the local robot system frame and its origin equates to the reference joint. 31

16 Illustrative phase state diagrams for the double pendulum model with the associated vector fields (shown as the cyan coloured arrows). The generalised coordinates of the position of the top and lower point masses are θ_1 and θ_2 respectively. Investigations were done using this model to provide simulated data for a two link serial planar mechanism, however this work did not fall within the scope of this project. . 33

17 The diagram describing the parameter D_{sf} metric for state trajectory closure. $Y1$ relates angular rates for the given angle domain $X1$ for the state space of a particular variable. The distance between the start (circle) and end (square) points indicates the degree of closure of an orbital path. 33

18 Flowchart for determining the parameter values for an inverse kinematics method. 42

19 Flowchart with the design elements, processes, methods and key outputs of this project. The graphics (i) to (iv) show how the video image of the subject is used to obtain the input image for the least squares optimiser. 44

20 Animal subjects with markers that are physically fixed on strategic joint positions to facilitate manual image labeling for DeepLabCut CNN training data. 49

21 Animal subject with CNN-Resnet50 predicted joint positions by DeepLabCut and a composite graphic with the 8-link configuration superimposed. 50

22 Learning rate data for a typical network training session showing the reduction in errors (loss) and the learning rate variation as the number of iterations progresses. 51

23 The subject at various orientations and magnification showing the labels as predicted by the pose estimation algorithm, DeepLabCut, that is based on deep neural network (specifically CNN) machine learning. The 190×50 pixel blocks are substantially smaller than the average used in for all three of the quadruped gaits investigated here. Blocks of this size were avoided, as far as possible, by appropriate video cropping. 52

24 A sample mechanism showing link lengths, joints and angle definitions within the UVW local system. The variables here are used to define the 3D joint positions according to the transform equations. 56

25 The image data points of the unseen video sequence showing the Convolutional Neural Network (CNN) estimated pixel values of the labels. All labels for the legs, spine and tail are shown. The subject moves from left to right and a composite plot of these points on the original images correlates well with the fixed markers. 63

26 The predicted joint trajectory points in the image plane using machine learning (DeepLabCut) together with the associated probabilities of correct occurrence are shown here. The pixel values as a function of the configuration number (or equivalently time) is shown in (i) and the likelihood or confidence in the labels' correctness as given by DeepLabCut is shown in (ii). 64

27 Original sample images and the predicted image configurations by the network. During the course of the network training the estimates on unseen images were not perfect but this was improved upon by retraining, refinements and parameter adjustments in the configuration files. 64

28 Typical illustrative output data by Matlab for a single solution point. The 21 variables comprising the solution vector includes the generalised position coordinates for all the links, the position coordinates of the reference joint, camera rotation angles and the base link length. 65

29 Typical Matlab output of selected performance indicators. The solution vector of 21 variables and the current function values are shown for a specific iteration. With reference to Figure 28, note the small values for the ϕ angle variables. 66

30 Typical Matlab output graphics for the total number of function evaluations and the final step size that the solver takes at solution. For this solution point the solver required 1358 function computations in total and only 30 iterations (coincidentally equal to the number of configurations here). Notice the steep convergence towards the desired zero line for the final step size taken by the optimiser for the last iteration. The data shown here is for the solving of the second configuration and the others were of a similar order of magnitude. . . 67

31 The number of iterations and first-order optimalities for all the configurations. Low first-order optimality values close to zero are a good indication that the solutions are near to being optimal. 67

32 The mean objective function residuals and final step sizes. The mean residuals or errors are the average displacements in hyperspace between the estimated vector and solution points. The final step sizes are sufficiently close to zero and contribute to the validation statements of the solver. Notice the improvement after the first iteration. From the second iteration angular rates can be computed and the contributions of the other vector functions are now included in the solution. 68

33 The high number of function evaluations due, in part, to the use of centrally based gradient computations by finite differences. 69

34 The mean values of all the Jacobian arrays as evaluated by the solver. 70

35 The average of the Jacobian matrix element values for only the angle variables of the quadruped mechanism. Note the smaller values for all the ϕ variables (variable numbers 8 to 14). 70

36 The gradients matrix values are linked to the optimality of the optimisation estimates. Low values indicate that the search has effectively ended with all search directions static at the solution point. 71

37 The mean error values for the joint image estimates are shown for all configurations. Notice the peak for joint #9 (mid-spine joint). The data shown here is repeated in Figure 38 as a two dimensional image error data matrix with the joint index numbers corresponding to the joint numbers of the mechanism. 72

38 The mean error image array data above show the differences between the actual and estimated pixel values for all the robot joints and averaged over the entire sequence. The schematics define the interpretation of the matrix data according to the configuration's joints. Note the peak error for joint #9 (the mid-spine joint) at joint index #5. The schematic at lower right is a reference for the graphic directly above it. The errors are specified for a xy pixel pair corresponding to a joint index number which in turn represents a specific joint number of the mechanism. 73

39 Least squares solver and truth images of a single sample configuration. All the other configurations also showed very close correlation. See Appendix E. 74

40 The photos from the original accelerating cheetah video and the reprojected least squares optimisation configuration images are shown for the sequence at every sixth instance. 75

41 The optimised camera rotation angles and the base link lengths for each of the 30 configurations. 76

42 The progression of the estimated reference joint trajectory (blue) in the image plane. Note its approximately sinusoidal path of about $3Hz$, confirming the assumed three gait motion of the subject. 76

43 A composite graphic of the estimated reprojected images in pixel coordinates. The estimated 3D poses by the solver are input into the camera model to give each of the pixelated configurations. 77

44 The composite side views of the quadruped for the first gait (left) and the first two gaits (right) in its local UVW coordinate system. The reference joints observer coordinates have been included so the origin of the UVW shows the corresponding shift. 77

45 The magnitude of the velocity and acceleration of the reference joint. All three linear motion components were calculated from its position data and interpolated and using the higher sample rate the linear velocity and accelerations were obtained. 78

46 Illustrative graphics showing the subject with the estimated image configuration, together with the side view (UV) and top elevation (UW) view plots. 79

47 The first gait sequence of nine configurations is represented here. The photograph and reprojected image composites together with the corresponding configuration plane views are shown. 80

48 A composite of the full quadruped configurations in the UV plane (side view) with the reference joint located at $(0, 0, 0)$ in the local system. 81

49 The top and front view spine patterns developed over the motion sequence. 81

50 The front views (looking head-on to the quadruped) show the distributions of the legs. The back leg, particularly, shows a definite pattern; at high speeds (shown in green) there is a greater deviation from the sagittal plane, as is to be expected. 82

51 Estimated reference joint trajectories. At left are the three dimensional components of the reference joint trajectory, with all camera extrinsics fixed to zero, in the observers XYZ-coordinate frame. The trajectories at right are the result of the solver having the camera angles as variables. 82

52 The full configuration sequence of the quadruped mechanism in the camera frame system with origin $(0, 0, 0)$ represented by the yellow dot. The start position is at left (red) and it traverses through space until its final position at right (green) after an estimated distance of $12.8m$ after 0.9667 seconds. It achieves an estimated maximum speed of $21m.s^{-1}$. 84

53 The target trajectory for the inverse kinematics models is generated by the front leg of the quadruped. The early phases of the trajectory are shown as red and terminate in green. 85

54 Inverse kinematics parameter λ as determined by a non-linear least squares optimisation for the damped least squares method. 87

55 Inverse kinematics constants as determined by a non-linear least squares optimisation for the Weighted Least Norm method. The variables that comprise the diagonals of the weighting matrix W are $w_1, ..., w_{10}$. 88

56 Inverse kinematics parameters $g_1, g_2, ..., g_5$ and k, for the gradient projection method as determined by the non-linear least squares optimisation solver. 89

57 The estimated parameter α values for the extended Jacobian method together with its average of 0.643. 89

58 The variation of the scaling factor α as computed online for the GP and WLN methods. The mean estimated value as determined by the EJ method is indicated by the dashed line. 90

59 The DLS method produced this trajectory (magenta) given a target point (red). A similar result was obtained with the WLN method. The cyan coloured trajectory is the actual path taken by the end effector. Notice how the IK determined path of the end effector is smoother and less oscillatory, particularly in the UV plane, than the original actual trajectory. 90

60 The iteration times for all of the IK models. 91

61 The rapid convergence to the target is shown by the distance in task space between the current model estimate and the target. The data shown here was for the DLS with the WLN method showing a similar trend. 91

62 All the configuration sequences and proposed solution trajectories are similar for the DLS and WLN algorithms. 92

63 The optimal augmenting function that is a harmonic kinematics mapping for the EJ algorithm. The variables q_1 and q_2 are relative angles corresponding to two links. 93

64 By using a suitable harmonic augmenting function with the extended Jacobian method, quick convergence to the target is achieved with no singularities. The computation of α online (ii) results in a faster convergence to the target than using $\alpha = 0.643$ constant (i). 93

65 The optimal trajectory produced by the extended Jacobian method. The optimisation function took the form of a harmonic augmenting function with the variables being selected relative motion angles. Every hundredth configuration is shown. 94

66 The gradient projection algorithm produced this trajectory in only 264 iterations with each iteration taking on average 0.36 microseconds. Every hundredth configuration is shown too. 94

67 The front leg end effector trajectory as estimated by least squares optimisation. 96

68 The phase states of the joint angle of the end effector link of the front leg as determined by the solver. Three gait cycles have been isolated here and the full cycle (lower right) shows convergence to a limit cycle. 96

69 Demarcated boundaries for all three gaits. The second gait is defined as occurring from the configuration index numbers 113 to 201 based on the position and velocity data of the end effector. 97

70 The nearly closed trajectories of the end effector, in Cartesian coordinates, for the second gait. 97

71 The phase states for the primary motion plane of the end effector as determined by the least squares solver. 98

72 The primary phase states for all the joints as determined by the damped least squares (left set) and the extended Jacobian (right set) methods. 98

73 The effects of computing the scaling factor online and applying it to the angular rates vector shows similar to a very slight degradation in performance (ii) with slightly less smooth trajectories and nominal to increased Degree of Closure (DOC)s when using the weighted least norm (WLN) method. 99

74 The Kengoro humanoid robot with intricate musculo-skeletal links (Image credit: JSK Laboratory, University of Tokyo). 107

75 The Versius Surgical System (Image credit: CMR Surgical). 108

76 Joint trajectory estimates in θ phase space by the gradient projection method. 116

77 Joint trajectory estimates in ϕ phase space by the damped least squares method. 117

78 Non convergent trajectory resulting from an unsuitable cost function: $g = (\theta_2 - \theta_1)^2(\theta_1 - \theta_3)^2 + (\theta_3 - \theta_4)^2(\theta_4 - \theta_1)^2$. The graphic below it shows the distance between the current and target points confirming failure to converge. 119

79 Chaotic, non convergent trajectory resulting from an unsuitable cost function: $g = (\theta_2 - \theta_1)(\theta_2 - \theta_3) + (\theta_3 - \theta_4)(\theta_4 - \theta_1)$. The graphic below it shows the distance between the current and target points confirming failure to converge. 120

80 Discontinuous, non convergent trajectory resulting from an unsuitable cost function: $g = (\theta_3 - \theta_2)(\theta_2 - \theta_3) + (\theta_3 - \theta_4)(\theta_4 - \theta_1)$. The graphic below it shows the distance between the current and target points confirming failure to converge. 120

81 Vector Field I. 121

82 Vector Field II. 122

83 Polar velocity magnitudes for (i) front leg, (ii) spine and (iii) back leg links for all of the configurations. 122

84 The first set of 9 configurations (out of 30). At left are the composite image configurations showing the truth (black dashed line) and reprojected (black solid line) images and at right the computed quadruped configurations (red) in the sagittal plane. 123

85 The second set of 9 configurations (out of 30). At left are the composite image configurations showing the truth (black dashed line) and reprojected (black solid line) images and at right the computed quadruped configurations (red) in the sagittal plane. 124

86 The last set of 12 configurations (out of 30). At left are the composite image configurations showing the truth (black dashed line) and reprojected (black solid line) images and at right the computed quadruped configurations (red) in the sagittal plane. 124

List of Tables

1 Mean polar vector magnitudes for the back leg (BL), spine (S) and front leg (FL) links. The highly mobile back leg shows high mean values in comparison to the front leg. 84

2 Trajectory generation inverse kinematics data from current to target point. 92

3 Trajectory closure percentages in θ joint space for all the inverse kinematic models in the UV plane. 99

4 Trajectory closure percentages in ϕ joint space for all the inverse kinematic models in the UW plane. 99

5 Performance indicators of all the inverse kinematic models for trajectory closure in the UV plane. 99

Acronyms

AI artificial intelligence.

CNN Convolutional Neural Network.

COM centre of mass.

CPU Central Processing Unit.

DH Denavit-Hartenberg.

DLC DeepLabCut.

DLS damped least squares.

DOC Degree of Closure.

DOF Degrees of Freedom.

EE end effector.

EJ extended Jacobian.

GP gradient projection.

GPS Global Positioning System.

GPU Graphical Processing Unit.

GUI	Graphical User Interface.
IK	Inverse Kinematics.
IMU	Inertial Measurement Unit.
LIP	linear inverted pendulum.
LN	least norm.
LSQ	Least Squares.
LVM	Levenberg-Marquardt.
ODE	ordinary differential equation.
PD	proportional derivative.
QR	Quadratic Root.
RC	Robot Configuration.
SDLS	selectively damped least squares.
SFL	spine-front leg.
SLIP	spring-loaded inverted pendulum.
WLN	weighted least norm.

Chapter 1

Introduction

The natural world has provided humans with vision, insight and inspiration to the development of flying machines through to today's bio-inspired robots.
This project sets out to use the imaged postures of a cheetah to realise the corresponding three dimensional poses and to generate motion trajectories for an equivalent quadruped mechanism.

1.1 Preliminaries

The locomotion of modern day bio-inspired robots require a study of the subjects' motion and the information derived from this can be used for a whole host of applications from the development of steady gait sequences, trajectory tracking, required control outputs and so on. All the spatial motion data can also be used in hybrid control systems that make use of machine learning. The design of robotic systems requires that real time configuration tracking be accurate and that its motion is generated according to some specified trajectory. These trajectories can be computed offline and serve as a database for later use for steady robot motion with repeating gaits. There is no general formalised non-linear systems theory [35],[36] and each design problem is unique, since there are no systematic procedures nor are there any single methods valid for analysis.

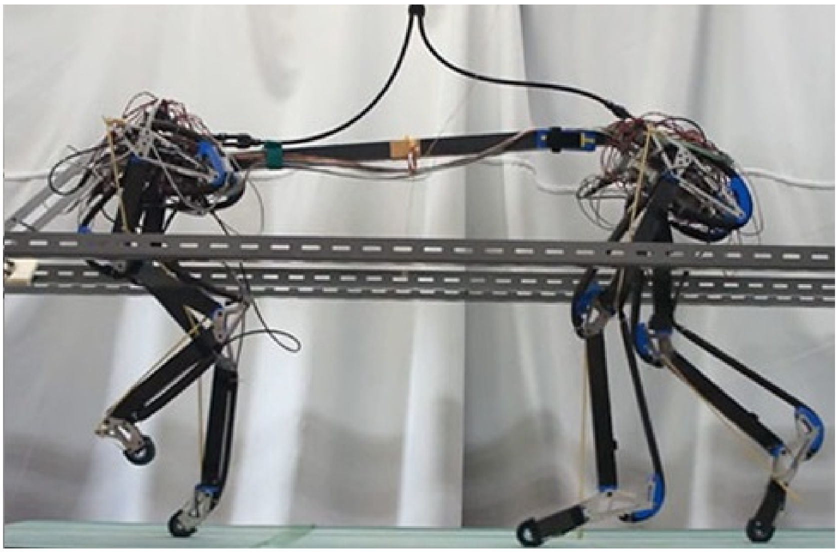


Figure 1: The Pneupard is an experimental light weight, biomimetic pneumatic powered quadruped similar to the proposed model presented in this publication (Image credit: Osaka University).

The analytical solution or single solution trajectory of a system variable that varies with time is seldom obtainable for highly non-linear systems and it is then best to do a qualitative analysis using methods that are based on conserved energies, sudden changes in behaviour with a change in physical parameter and the generation of sets of simulated state trajectories to name a few.

1.2 Research Objectives

This research project makes use of an accelerating cheetah animal subject and optimisation theory to estimate the 3D motion sequence and associated poses using a monocular video camera system and machine learning. The reduced and simplified configurations of the modeled quadruped are reconstructed and the trajectory trace in space is rendered as an animation. The main part of this work therefore relates to the theoretical derivation, formulation and validation of the output of the solver using various indicators.

Of secondary interest are the relative link motions and the path followed by the extremity of the front leg in 3D space, and the required actuations for this are to be resolved using inverse kinematics theory [50],[51],[52],[63],[65]. Four inverse kinematics models are parameterised and a comparative evaluation is made to assess their ability to (i) produce a smooth, economic end effector (EE) trajectory to a target point and (ii) their effectiveness in generating closed joint space trajectories for a given closed loop end effector trajectory in the spatial workspace. The final objective of this project therefore achieves the angular rates required for the spine and front leg links of a quadruped that could be used in a control application.

1.3 Outline of Work

The work flow is described from the model development, acquisition of the raw data and its processing and then the implementation of the theoretical models to achieve the three goals described above.

1.3.1 Problems and Scope

The solution process to the problems faced in this project entail a logical and sequenced approach to data acquisition, systems definitions, mathematical modeling and parameterisation thereof. The estimated image configurations from the video need to be obtained using the open source toolbox DeepLabCut [38] that allows the user to train a deep neural network for the purpose of pose estimation. Once a representative mechanism that approximates the quadruped is established, the mathematical models that characterise a non-linear least squares [67] optimisation need to be formulated. Careful thought must also be given to how the dimensions of the configuration are modeled. The visco-elastic nature of the subject complicates all of this. Once the estimated 3D poses in space has been determined these will be used as inputs to the various parameterised kinematics models which in turn will give the actuations required for the tracking problem.

In formulating a description of a system one seeks a definition that includes independent variables that uniquely and simply define the system. The motion of a simple pendulum, for instance, can be uniquely described by a generalised coordinate such as the angle θ that it makes with respect to a reference line with the state of the system being described by its position in the phase state space, in this case the coordinates θ and $\dot{\theta}$. The trajectory trace is called the phase portrait [30].

1.3.2 Proposed Solution Plan

In order to solve the pose estimates, insight into the theory of virtual camera modeling, spatial geometry, vector fields [13], coordinate transforms, non-linear least squares optimisation [67] and the mechanics of the inverse kinematics models is required. An adequate static model of the quadruped is proposed, its geometrical and assumed dynamic limitations are set within an operating manifold and the component functions for the optimisation are developed within this framework. For the inverse kinematics part, optimisation theory is again used to determine the model parameters. With advances in machine learning and artificial intelligence (AI) the need for a closed-form parameterised deterministic model of the full robot system can be largely circumvented. With the development of more complex legged robots involving higher order linkage systems this is especially true; in such instances it is near impossible to find a deterministic mapping from the command space to the configuration space [16].

The double pendulum and n-link models are all standard dynamic and control models, together with their derivatives the pendubot and acrobot. Other approaches for generating locomotion trajectories and that focus on the dynamics covered by single point masses are the linear inverted pendulum (LIP) and spring-loaded inverted pendulum (SLIP) models. The motions of these reduced models can be generated using estimation/optimisation techniques and these methods have been applied to intrinsically compliant biped robots [1] with the LIP a common model for tracking the centre of mass (COM) [40].

Vector fields are a form of geometric algebra in that they are closely associated with ordinary differential equations that describe the dynamics of a system. The most obvious is the visual description of the velocity flow (others include pressure and thermal gradients, force fields and height gradients of contour maps) and they give the instantaneous path direction of the solution trajectory at a point in state space, for instance. The vector fields therefore give information to the solutions of an ordinary differential equation (ODE) without having to solve it. Non-linear systems are inherently complex to solve so instead of resolving analytically, clues to behaviour are obtained using linearisation and the information of their behaviour at stationary (equilibrium) and other points of interest. Together with other geometrical realisations depicting variables and parameter changes, the qualitative behaviour can be deduced.

For floating based platforms, such as the quadruped in this project, there are additional non-linearities hidden in the dynamics resulting from invariant contacts and displayed in the end effector ground contact manifolds. The use of single imaging sensors is a powerful tool for acquiring sensor data since it is remote, simple, unobtrusive and a system configuration can be acquired. Together with machine learning and an optimisation algorithm, where the parameters can be optimised and/or learnt, this provides for efficient tracking without the need for real-time state estimation, especially in situations with highly non-linear motion that may show highly discontinuous end effector trajectories. Used in parallel with an optimisation algorithm a monocular vision system is all that is needed for a 3D trajectory realisation of a multi-link robotic system.

This project is primarily concerned with the 3D motion of an accelerating quadruped on uneven terrain using vision systems and a non-linear Least Squares (LSQ) optimisation algorithm [67] that is custom developed for this specific application. It makes use of the video image sequences of an animal subject as truth inputs for the configuration optimisations. In order to solve the problems of trajectory generation, the first step is to obtain the parameters for the inverse kinematics methods as well as the development of the implicit optimisation functions where applicable. Termination criteria are coded and the current estimates are tracked until suitable closures are attained. The qualitative joint trajectories and configurations are assessed visually and any correspondences with expected behaviours are noted. Comparisons with the quadruped are not easily made under accelerating conditions since ground contact forces are highly variable in addition to the obvious differences in physiology. The basic theory that is utilised here is presented in Chapter 2 that includes introductory notes on machine learning and the technical descriptions that relate to modeling the vision systems, non-linear optimisation and the inverse kinematics models.

1.3.3 Project Content

The definitions of the systems and tools for implementing the problem solving strategies are made in Chapter 3 and an outline of the methods and approaches used in order to solve the problems are provided in Chapter 4. The details of the practical data acquisition, processing and implementation are described in Chapter 5 with the full results, analysis and discussions shown in Chapter 6. The conclusions drawn from the results are discussed in Chapter 7, with recommendations provided for any future work on this subject given in Chapter 8. Finally, possible future research directions, at large, for humanoids and quadrupeds is also given, together with some of the latest and advanced robotic systems.

Chapter 2

Theoretical Fundamentals

The basic theoretical concepts and definitions that are used in this research are discussed and formalised in this chapter. Introductory notes to machine learning, the basic derivation of the virtual camera model and the geometric realisation of vectors are presented.

The theory of non-linear least squares optimisation is described in detail together with key performance indicators that are used to evaluate the credibility and accuracy of the data output. The theory of inverse kinematics models is outlined in detail, one of which forms the basis for the optimisation algorithm used.

2.1 Convolutional Neural Networks

The deep learning theory of ConvNets or CNNs involves multi-layered and weighted learning networks that have their internal parameters changed iteratively until they are "trained" and are then able to predict correctly (within reasonable limits) when new unseen input data is presented. CNNs are mainly involved in intelligent image data processing from object classification, clustering based on similarity or object detection such as the application here. They process hyper dimensional image arrays or tensors (nested arrays) and filtering functions that convolve with input images to detect the specified feature(s). Transfer learning based algorithms are employed here, so the trained network can be applied to similar anatomically based life forms

[38], such as the leopard. The essence of the software operation is to train the feature detectors of the anatomical points as defined by the user via an interactive Graphical User Interface (GUI) with all commands of the process being from a simple Python terminal interface. DeepLabCut is the transfer learning algorithm [38] that is used to generate a trained network for the purposes of creating a video sequence of poses with the estimated positions of predefined markers on an unseen sequence of images of a live subject.

2.2 Vision Systems and Image Processing

The derivation of a camera model is outlined together with notes on the vision system used.

2.2.1 Virtual Camera Model

The model equation that gives the pixel coordinates for a given point in space is based on the standard pinhole camera whose aperture is a point of infinitely small dimensions that ensures all projected points are not subject to blurring. Projective geometry is used to construct the theoretical transforms together with transform

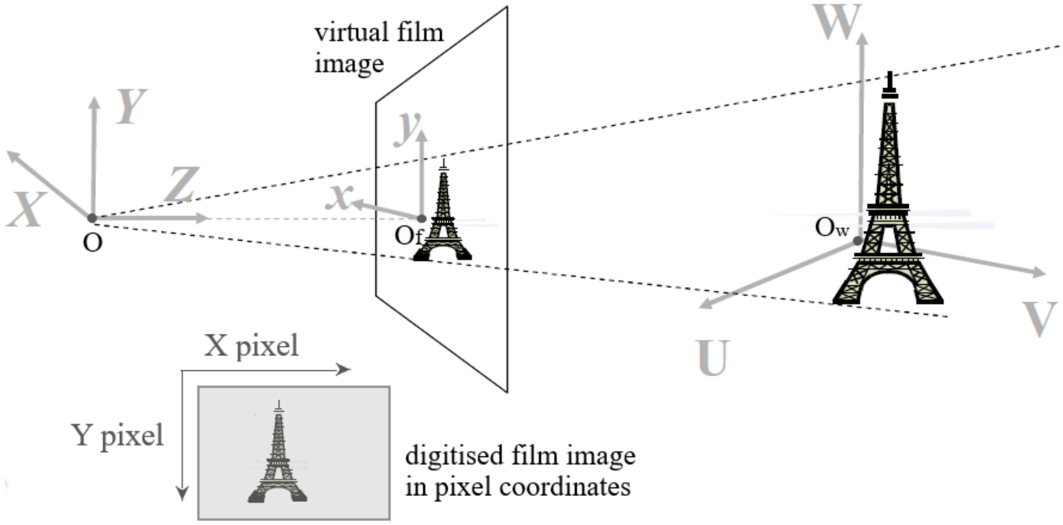


Figure 2: Camera model showing the virtual image generated from a real world 3D object. The desired pixel image is processed from the film image according to the physical properties of the actual camera being used. The respective origins of the frames are indicated by O, O_f and O_w.

matrices that will hold information on the camera's intrinsic physical properties and any applied camera rotations and translations are described by the extrinsic variables. The virtual image is the inversion of the image that the camera sees with both being located at the same distance, f, along the optical axis away from the optical centre O. By making use of the fact that a Cartesian point is a line in homogeneous coordinates and the invariance of these coordinates to scaling the spatial world system coordinates can be projected to the image frame coordinates. In this way there is a transform from world to camera to film to pixel coordinates and P_t is the matrix that performs this transformation. The key here is the perspective projection of the world system to the film plane as shown in Figure 3. The variables

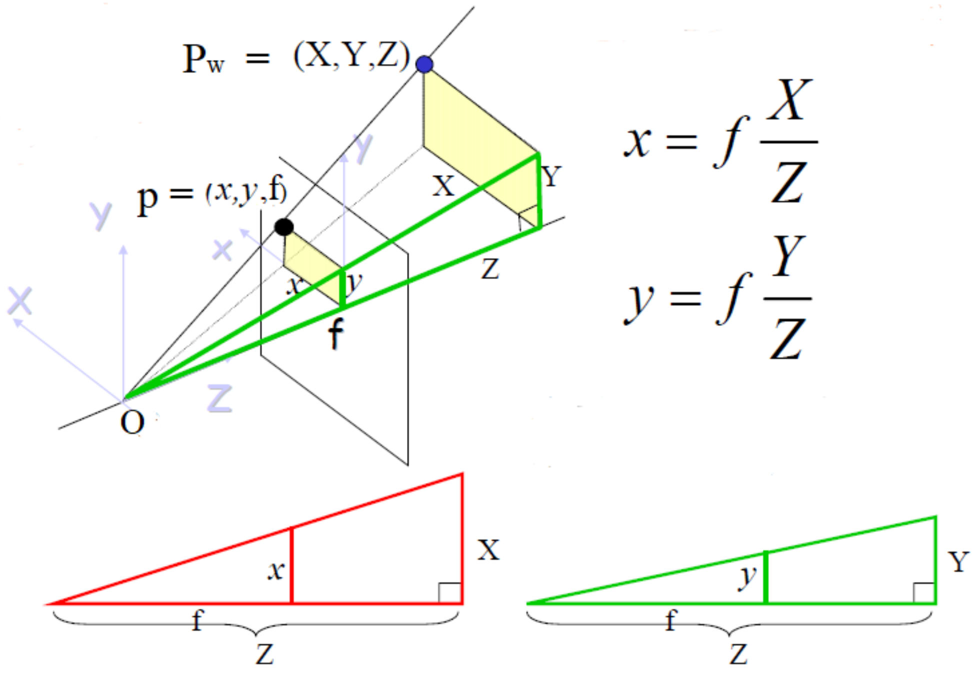


Figure 3: The simple linear geometry that is fundamental in obtaining the image coordinates of a point in space.

o_x and o_y are the pixel coordinates of the optical centre in the film plane, or alternatively the offsets of the pixel and film plane origins, and where the image coordinate axes are not perfectly orthogonal then the skew is given by $s = f_y \tan\delta$ where δ is this axes (rotation) deviation from the norm. For perfectly square pixels $\delta = 0$ so the skew will be zero too. The intrinsic variables f_x and f_y are focal lengths in pixels depending on the pixel resolutions μ_x and μ_y (pixels/unit (mm) along their respective axes), so $f_x = \mu_x f$ and $f_y = \mu_y f$, with f being the focal length in mm of the camera used and as shown in Figure 3.

If one were to align and orientate the camera and system coordinate frames exactly, a translation followed by a rotation would be required. The respective vector t and R matrix with its component definitions for these are,

$$t = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}, \tag{2.1}$$

$$R = [R_x]\ [R_y]\ [R_z] \tag{2.2}$$

where

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\xi & -\sin\xi \\ 0 & \sin\xi & \cos\xi \end{bmatrix} \quad R_y = \begin{bmatrix} \cos\psi & 0 & \sin\psi \\ 0 & 1 & 0 \\ -\sin\psi & 0 & \cos\psi \end{bmatrix} \quad R_z = \begin{bmatrix} \cos\zeta & -\sin\zeta & 0 \\ \sin\zeta & \cos\zeta & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The camera rotation angles ξ, ψ and ζ are defined as in Figure 15 (Section 3.2) and by the PATB convention.

If K is the intrinsic camera matrix the total camera matrix is given by $P_t = K\,[R|t]$ with t the translational column vector that is amended to the rotation matrix R and with

$$K = \begin{bmatrix} f_x & s & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.3}$$

A standard camera model with projection and rotation-translation matrices relate 3D coordinates to the final 2D pixel image. If P_t is the total camera matrix, which includes camera intrinsics and extrinsics (rotation and translation variables), and $\mathbf{X}$ is a homogeneous vector with the 3D global world system points and σ a focal (depth) length scaling factor, then the homogeneous image vector $\mathbf{x}$ is given by

$$\sigma\mathbf{x} = P_t\mathbf{X}. \tag{2.4}$$

One has the choice of dealing with the formulation as it is and evaluating σ for each new configuration, or operating with the implicitly scaled pixel equations directly. Each system has its own global coordinate axis for convenience with the camera extrinsic matrix defining the 3D translation and rotation of the camera's local coordinate system with respect to this. Each rotation matrix defines rotations in all three of the respective directions, and similarly the translation matrix, with the total camera matrix amended to make allowance for the homogeneous nature of the projection. The location of a system in the global coordinate system was defined as the homogeneous vector $\mathbf{X}$ in terms of the following variables: the position of the reference joint in 3D with all joint positions being defined by the link length and the two angles in the sagittal and transverse planes.

2.2.2 Vision System

The single camera system used here makes no use of a multi-camera stereo system where homographies could have been used. Image data from multiple cameras were, however, available for training of the neural network but a single monocular vision system using the camera model described here is all that is needed. A vision system needs to be developed in conjunction with transition equations that solve for the spatial coordinates using angle data and link lengths only.

2.3 Polar Transforms

The polar transforms for any point defined in polar coordinates can be used to obtain the equivalent position in a Cartesian space. The magnitude of the radial velocity and angular velocity components can be computed using only the variables defined in the basic geometrical definition as follows:

$$x = r\cos\theta \quad \text{and} \quad y = r\sin\theta \tag{2.5}$$

$$\dot{r} = \frac{x\dot{x} + y\dot{y}}{r} \quad \text{and} \quad \dot{\theta} = \frac{x\dot{y} - y\dot{x}}{r^2}. \tag{2.6}$$

Consider the arbitrary spiral trajectory shown in Figure 4. A point P has a tan-

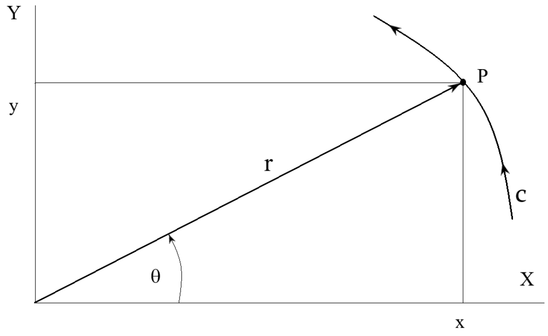


Figure 4: The variables associated with the polar transformation equations. The vector with magnitude r and direction θ describes a point P that traces out the curve C.

gential velocity of magnitude v with its corresponding components as shown. The magnitude of the velocity vector $v = v_x + v_y$ where

$$v_x = \dot{x} = v_r \cos\theta - v_\theta \sin\theta \quad \text{and} \quad v_y = \dot{y} = v_r \sin\theta + v_\theta \cos\theta. \tag{2.7}$$

In any problem v_r and v_θ can be specified as functions of θ or time for instance and for circular trajectories this simplifies to $v_r = 0$ and $v_\theta = r\dot{\theta}$.

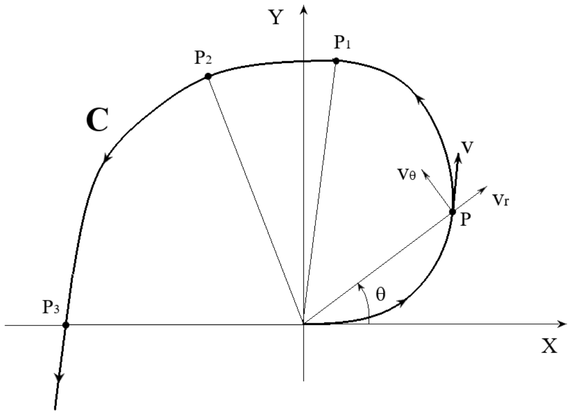


Figure 5: An arbitrary curve trajectory C, showing the radial and tangential components of the tangential velocity of a point P on C.

2.4 Vector Fields

Vector fields are a form of geometric algebra in that they are closely associated with ordinary differential equations that describe the dynamics of a system. The most obvious is the visual description of the velocity flow (others include pressure and thermal gradients, force fields and height gradients of contour maps) and they give the instantaneous path direction of the solution trajectory at a point in state space, for instance. These paths are variously called integral paths, trajectories, orbits or flows and, critically, the series of paths so formed is the phase portrait (discussed in Section 3.4) of the system of differential equations for a specific generalised state variable. The vector fields therefore give information on the solutions of an ordinary

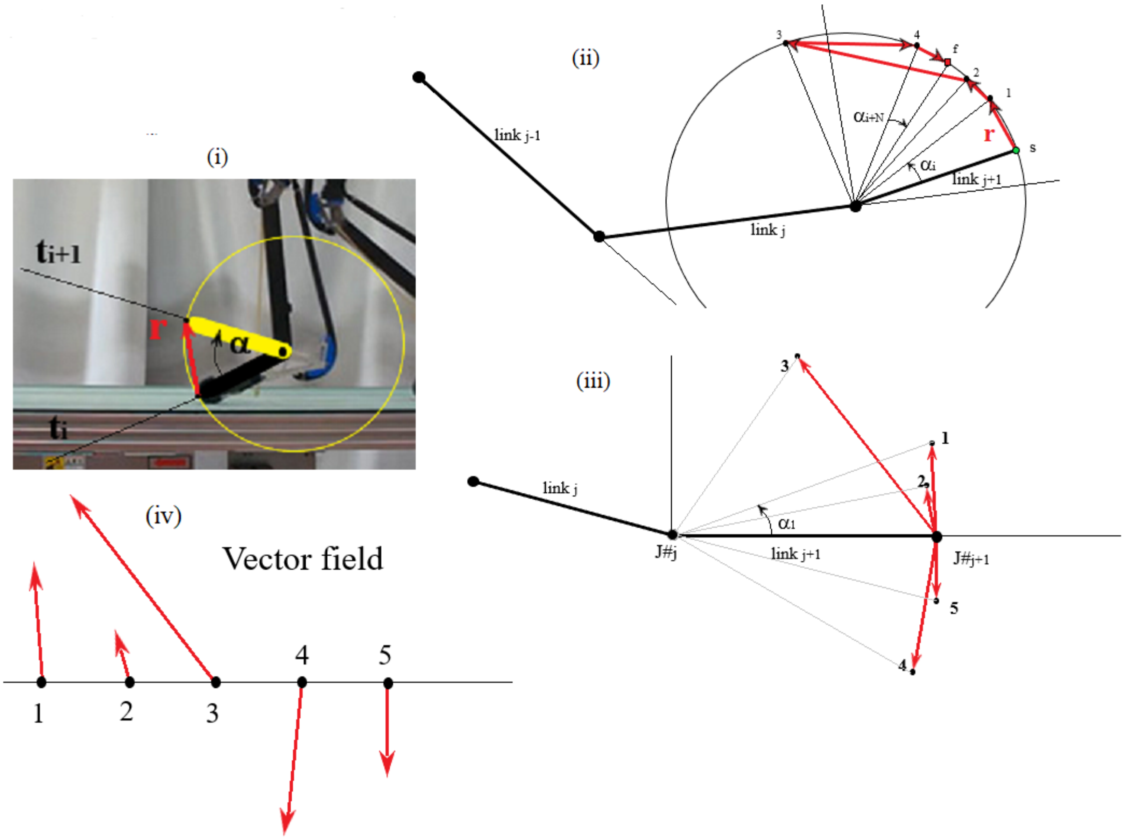


Figure 6: The vector field generation for the incremental movement of the robot link in (i) from time t_i to t_{i+1}. Schematics (ii), (iii) and (iv) show how the field is created for a generic $link_{j+1}$ as it undergoes five incremental movements which are relative to its prior $link_j$. The field is represented as a time line with the associated vectors at each interval.

differential equation without having to solve it. The topological organisation of the phase portrait is determined by the critical (stationary or equilibrium) points.

The vector functions are formulated for the non-linear least squares algorithm that reflect the basic geometry, physiology and expected dynamics of the quadruped. Direction or vector fields are mathematically the same thing as a set of ordinary differential equations, both of which give rise to a series of transformations of state space called *flows*.

Consider a second order autonomous system described by $\dot{\mathbf{x}} = F(\mathbf{x})$, as shown in

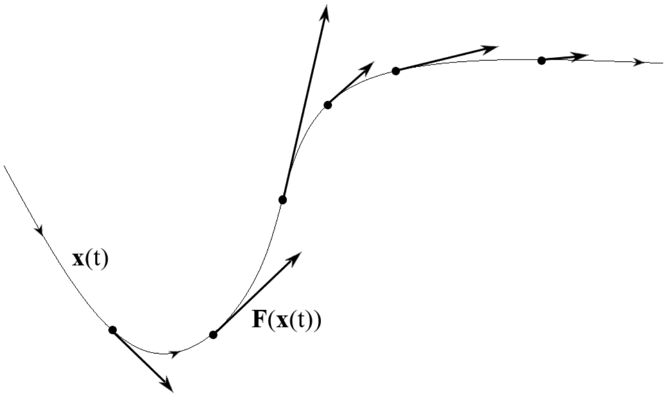


Figure 7: Solution integral curve $x(t)$ with the associated vector field $F(x(t))$ at the specific solution points.

Figure 7. Let the system variables be x_1 and x_2 so that at each $\mathbf{x} = (x_1, x_2) \in \Re^2$ the function F is attached. Then F is a continuous time-independent vector field with $x_1(t)$ and $x_2(t)$ the solution paths or trajectories, both of which can be considered to be the parametric representations of a directed path in their respective phase spaces. The solutions are thus the integral paths for the vector field. For non-autonomous systems the time dependent vector fields are applicable and the mapping is from a domain that now also specifies the time interval for which it is valid.

The polar vectors (see Section 2.3) also generate a field over time but these simply describe the magnitude and direction of the tangential velocity at a point. As an illustrative example, consider the angular velocity of the end effector given over 300 sequential configurations and the field of velocity vectors it generates. The incremental direction change of a vector is determined by the magnitude of the angle changes α that the end effector link undergoes and the magnitude of the vectors is simply the angular velocity of the link. The geometrical representation of this is shown in Figure 8. The vector $\mathbf{r}$ defines the incremental change in position of an end effector link r_i over a time interval T and relative angle change α. The projected vectors using polar transforms are also shown on the respective axes. The parent link of the end effector is r_{i+1} and it defines the reference line for the initial angle position of r_i at the start of the analysis period.

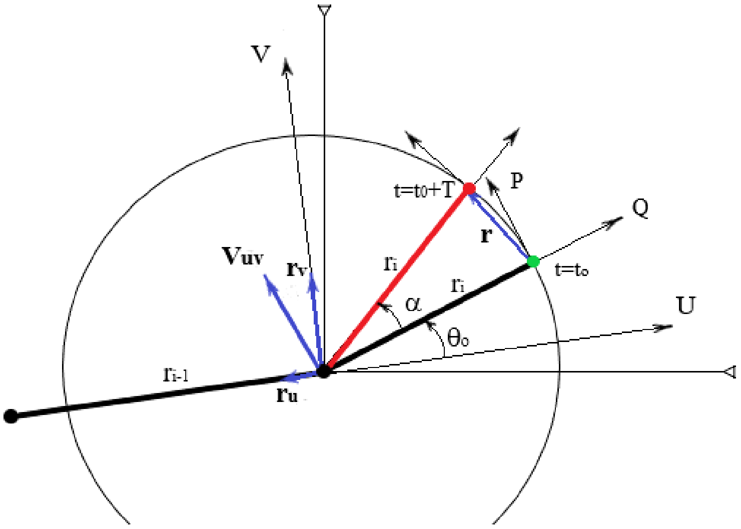


Figure 8: The vector $\mathbf{V_{uv}}$ is the projection (using polar transforms) of $\mathbf{r}$ onto the local UV axis of the parent link. In this way relative motions can be displayed for all links and the sequence of vectors obtained thus generating a field.

2.5 Non-linear Least Squares (LSQ) Optimisation

Least squares optimisation is a powerful tool for determining the parameters or variables of a proposed model given a set of output data points or desired trajectory points.

2.5.1 Optimisation Mathematics

Non-linear systems are inherently complex to solve so instead of resolving analytically, clues to behaviour are obtained using linearisation and the information of their behaviour at stationary (equilibrium) and other points of interest. Together with other geometrical realisations depicting variables and parameter changes, the qualitative behaviour can be deduced. For floating based-platforms, such as the quadruped in this project, there are additional non-linearities hidden in the dynamics resulting from invariant contacts and displayed in the end effector ground contact manifolds. Estimated link angle data obtained from the actual video sequences of the animal subject are crucial for accurate optimisation. The fundamental theory of non-linear least squares optimisation is based on optimising a cost vector function, the objective function, by changing and updating model parameters that compare the current iterative solution with that of the actual data. It operates similarly to minimising the error between the actual and desired outputs in the feedback loop of a control system. Graphically it is described as minimising the square of the distance between estimated and known points and encapsulated in

$$\min_{x} \sum_{i=1}^{m} f_i(x)^2 = \min_{x} \|f(x)\|_2^2 = \min_{x} f(x), \tag{2.8}$$

where

$$f(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_m(x) \end{bmatrix}, \tag{2.9}$$

with $f_i(x)$ differentiable functions of a vector or matrix variable x, the function $f(x) : \Re^n \rightarrow \Re^m$ and $\|f(x)\|_2^2$ being the squared norm (2-norm) of the residual (i.e. error) at x. This is illustrated by way of a hypothetical non-linear curved surface manifold comprising all the squares of $f_i(x) = \|x - a_i\|$ and the resulting contour lines (see Figure 9) with the actual data and estimated points. In most practical cases the function $f(x)$ will also depend on model parameters that require solving, such as the coefficients in a cubic polynomial, where a further refinement in accuracy can be done by applying an orthogonal distance regression fit as opposed to a standard least squares fit. As the number of variables n in a problem increases without a commensurate increase in the number of vector functions f_i (analogous to the number of simultaneous equations in the linear context: $f(x) = Ax - b$, with

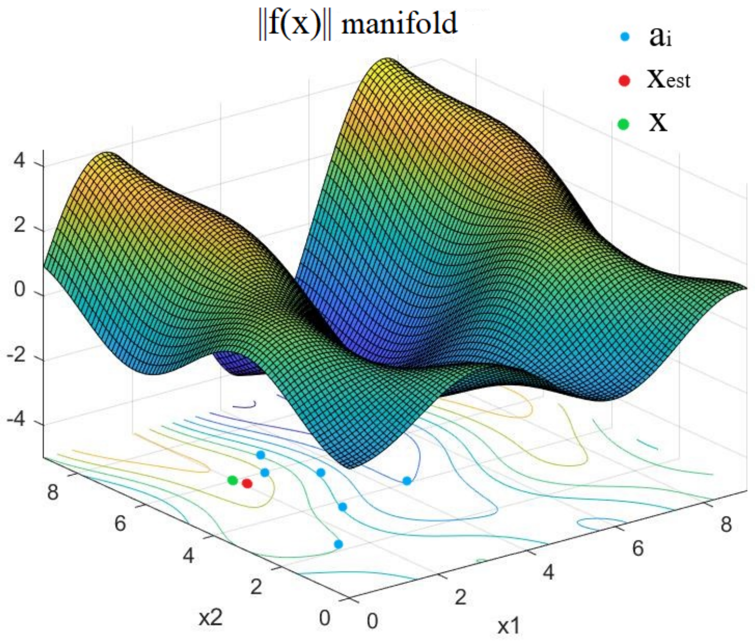


Figure 9: A hypothetical non-linear vector function with estimated solution vector and actual data points for a model with two parameters x_1 and x_2. The green dot shows the actual solution point x and the red point is the solvers' estimate x_{est}. Blue dots represent a few of the known data points a_i that are used by the optimisation to achieve the estimate.

A a matrix operator on the vector x), the system tends to become indeterminate. This format is based on the Gauss-Newton method which broadly speaking starts at any point, and is followed by the affine approximation (i.e. linearisation) of $f(x)$ around this point, which involves computing the Jacobian of $f(x)$ at this point and then an update step.

However, when the system is underdetermined ($n > m$) the update step becomes problematic and also when there is not a reduction in $\|f(x)\|^2$. By introducing an implicit regularisation parameter that is coupled with a distance function forces the estimates into close proximity of the solution.

A variation on this is the so-called Levenberg-Marquardt (LVM) algorithm, which includes Quadratic Root (QR) factorisation with iterations starting from multiple points that may converge to an optimal solution. Even if the solution is sub-optimal it will be guaranteed to be in relative close proximity to the actual data point(s). The LVM is robust in the sense of being able to handle underdetermined systems without constraints. The sparsity of the Jacobian matrix will affect whether the solution is optimal or not. Another important measure of the optimisation process is whether the smooth hyperspace objective function f is at a minimum where the gradient is zero. This measure is termed the first-order optimality measure or the infinity norm (i.e. maximum absolute value) of the gradient of the objective function and it need only be small (close to zero) to within a certain tolerance for f to be minimal. In some instances relative measures will be used to account for the scale of a problem.

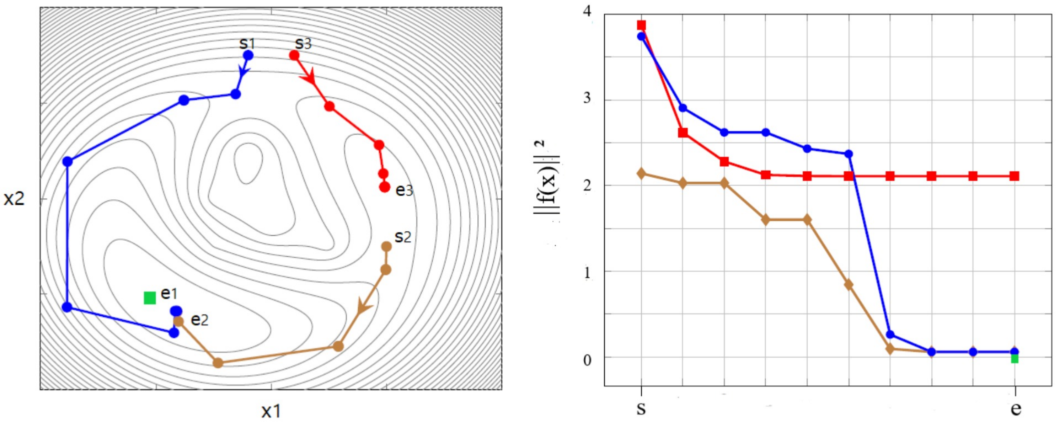


Figure 10: Another illustrative two parameter model example and how the initial guesses of the solution vector at the start of the algorithm search may affect the outcome. The initial search position for the red path results in a sub-optimal solution and this is confirmed by the magnitude of the squares of the residuals at this solution. The other two searches terminate at near optimal solutions (blue and brown paths). The actual solution is shown by the green square.

The trust-region-reflective algorithm is another way of solving non-linear minimisation problems but is only valid if the system is overdetermined or at least if the row dimension of f is equal to the number of variables.

2.5.2 First-Order Optimality Measure

An indicator used to describe how close a point x is to being optimal is the so called first-order optimality measure for the algorithm used here. This indicator makes use of the gradient value of the objective function $f(x)$. If $\min_x f(x)$ is a smooth unconstrained problem, then the maximum absolute value (infinity norm) of the gradient of $f(x)$ is the first-order optimality, F_O, which can be expressed mathematically as

$$F_O = \min_x |(\nabla f(x))_i| = \|\nabla f(x)\|_\infty. \tag{2.10}$$

Note that a point which has $F_O = 0$ does not mean that it *is* necessarily a minimum but *if* the point is a minimum then the condition $F_O = 0$ must hold.

2.5.3 Function Evaluations

During the optimisation the solver computes intermediate steps that are performed near the current iteration in order to find the next point, such as evaluation of the objective function or estimating a gradient by finite differences. At any step, intermediate calculations may involve evaluating the objective function and constraints, if any, at points near the current iterate x_i. For example, the solver may estimate a gradient by finite differences. At each of these nearby points, the function count is increased by one. Occasionally, the optimisation will attempt and reject a step and the algorithm used in this project does not tally these additional steps to the number of iterations but instead to the number of function evaluations. Furthermore, if the gradient estimates are computed as centered using finite differences then it will take twice as many evaluations and should be more accurate.

2.5.4 Damped Least Squares Parameter λ

The search direction of the Levenberg-Marquardt algorithm, which makes use of the damped least squares model, is guided by the directions of the computed gradients as evaluated by both the Gauss-Newton and steepest descent method, given by

$$\frac{d}{dt}x(t) = -\nabla f(x(t)). \tag{2.11}$$

If, at solution, the error of the objective function is small enough then the solver will favour the Gauss-Newton method ($\lambda \approx 0$), otherwise the steepest descent direction is taken with increasing λ. If the error decreases the step will be rated as a success (as opposed to a failure) and λ is reduced by a factor of 10; else it will be increased by a factor of 10 by the Matlab solver.

2.6 Inverse Kinematics (IK) Methods

Forward or direct kinematics relates the Cartesian coordinates of the EE as a function of the angles of its constituent links, while the inverse kinematics resolves the required angles for a desired end effector position for a discrete time interval.
There are many ways to solve Inverse Kinematics (IK) problems, originating from applications in robotics, such as pseudoinverse methods, Jacobian transpose methods, cyclic coordinate descent methods, quasi-Newton and conjugate gradient methods and also, recently, the use of neural networks and AI methods [63]. The pseudoinverse method is widely used and discussed in research papers, however, it often performs poorly due to instabilities near singularities. The (selectively) damped least squares methods have much better performance [65].
Direct kinematics was used as the transform mechanism in the LSQ optimisation that related the position coordinates of all the joints for given angle vectors, $\mathbf{x} = f(\boldsymbol{\theta})$, while inverse kinematics finds the angle joint vector, $\boldsymbol{\theta} = f^{-1}(\mathbf{x})$ that satisfies the unique joint space configuration $\mathbf{x}$. Direct kinematics describes the relationship between the end effector velocity and joint velocities by $\dot{\mathbf{X}}_{\mathbf{EE}} = J\dot{\boldsymbol{\theta}}$.
The IK models that follow involve the computation of the joint angle vectors for a given end effector velocity $\dot{\mathbf{X}}_{\mathbf{EE}}$. It may be desirable to reduce the kinetic energy of the EE and it will be shown that a cost function can be designed to optimise a specified criterion for a specific joint. Additionally, some of the models below can be adapted to minimise torque or energy. The results obtained previously (but not included here) show that the kinetic energies dominate the spine-front leg (SFL) system and show the greatest variation (compared to the potential energy) and because of this it forms the focus for many energy considerations.

2.6.1 Least Norm Solution

The least norm (LN) method forms the basis of inverse kinematic theory but it is not practically feasible due to singularities almost certainly being present in computing the inverse of the Jacobian, J (and the fact that it is usually non-square) of the end effector position vector. The direct method or least norm solution relates the EE velocity rates in Cartesian coordinates and the angular joint velocity rates as follows:

$$\dot{\boldsymbol{\theta}} = \mathrm{J}^{-1}\dot{\mathbf{X}}_{\mathbf{EE}}. \tag{2.12}$$

Critically, this is valid provided the numerical sampling intervals are small or the EE velocities are low. Singular configurations or relatively local solutions that may not be globally acceptable can occur. This is the basic format for the minimum or least norm solution for the kinematics equations that follow. The Jacobian is a $m \times n$ matrix, and for a redundant manipulator $m < n$. Thus there are an infinite number of joint velocity solutions for a given end effector velocity, with n the joint space dimension and m the size of the Cartesian space containing the EEs position coordinates (number of normal constraints).

2.6.2 Gradient Projection

Evaluating the value of the inverse of the Jacobian and its usually singular condition and low rank require alternative formulations. The gradient projection (GP) method utilizes the formulation for the pseudoinverse of a matrix involving only the matrix and its transpose, J^T, with no inverse matrix computations. It is therefore composed of a minimum norm solution and a homogeneous linear equation solution as follows:

$$\dot{\boldsymbol{\theta}} = \mathrm{J}^{+}\dot{\mathbf{X}}_{\mathbf{EE}} + k\left[I - J^{+}J\right]g. \tag{2.13}$$

If the cost function g is optimised according to selected joint speeds it may have the form $g = \delta(\boldsymbol{\theta})^2$, and [14] specifies it as a fixed constant vector representing any self-motion speed (i.e. angular velocity) and this is also done in [51] where it is a vector of link angle changes. The constant k is another constant scalar model parameter.

2.6.3 Weighted Least Norm

A set of transform relations are defined that give weighted values for each of the elements in the LN solution. By using these together with the definition of the pseudoinverse formulation, the WLN solution is given as:

$$\dot{\boldsymbol{\theta}} = \mathrm{W}^{-1} J^{T} \left[J W^{-1} J^{T} \right] \dot{\mathbf{X}}_{\mathbf{EE}}. \tag{2.14}$$

The matrix W is symmetric and positive definite with weighting variables that need to be determined, but for simplicity it is usually a diagonal matrix [14]. If the dimension of the joint configuration space is n then $W \in \Re^{n \times n}$ and the diagonal elements will be $w_1, ..., w_n$.

2.6.4 Extended Jacobian

The extended Jacobian (EJ) method makes use of a sophisticated nullspace method where a local minimum of an explicitly defined cost function is tracked as a secondary task. The function $g = g(\boldsymbol{\theta})$ is any position variable based cost function that needs to be optimized according to a specific design criterion, and it can be the relative motion speeds (it was used to minimise the knee and angle joint velocities for the quadruped in [14]) of specific joints, or an energy or torque function or even constraints relating to obstacle avoidance. It needs to be minimised with respect to $\boldsymbol{\theta}$ in the nullspace of the Jacobian of the position vector. By extracting the basis vectors for this projection, a matrix

$$G(\boldsymbol{\theta}) = \frac{\partial g}{\partial \boldsymbol{\theta}} N_J \tag{2.15}$$

is generated with $(n - m)$ number of rows. By appending $\frac{\partial G}{\partial \boldsymbol{\theta}}$ under $\frac{\partial F}{\partial \boldsymbol{\theta}}$ with J $=$ $F(\boldsymbol{\theta})$ the extended Jacobian matrix J_e is formed. The angle velocity vector is then given by

$$\dot{\boldsymbol{\theta}} = J_e \left[\frac{\dot{\mathbf{X}}_{\mathbf{EE}}}{\mathbf{O}} \right]. \tag{2.16}$$

It is imperative that J_e maintains full rank (i.e. rank equal to m) during the iteration since its nullity (number of basis vectors) is fixed according to the rank-nullity theorem: $n = rank(J) + nullity(J)$. The nullity of J is simply the dimension of the kernel of J which is given by $\{\mathbf{x} \in K^n | L\mathbf{x} = \mathbf{0}\}$ with the vector $\mathbf{x}$ lying in the row space of J, and L a linear mapping. The EJ algorithm is a pseudoinverse method with an explicit optimisation criterion; optimisation is in the nullspace of the Jacobian using a kinematic cost function $g(\boldsymbol{\theta}) = \sum(\boldsymbol{\theta_i} - \boldsymbol{\theta_{io}})^2$ for all links for example. If the quadruped's end effector is positioned at $\mathbf{X_{EE}}$ then g is fully optimised and $G(\boldsymbol{\theta})$ is mapped to $\mathbf{O}$. Certain formulations [52] include a scalar α coefficient prefixing the generic $J^{-1}\dot{\mathbf{X}}_{\mathbf{EE}}$ least norm term in all the models and this was included in some of the optimisations. This method is just a pseudoinverse operation with an explicit optimisation criterion $g(\boldsymbol{\theta})$ that is performed in the nullspace of J.

The optimisation function is an augmenting kinematics map that, in combination with the kinematics, becomes a local diffeomorphism (isomorphism of smooth manifolds) of the augmented task space and the specific choice of the augmentation depends on the optimal approximation of the pseudoinverse. Work done by [7] proposes a novel formulation of the approximation problem, where a singularity-free region of jointspace is created and the augmenting mapping $g = g(\mathbf{q})$ is harmonic, where $\mathbf{q}$ is a vector of generalised coordinates. The aim now is to find g from a family of harmonic functions that minimises an error (details in [7]). These have the general solution form $g_{a,b}(\mathbf{q}) = q_2(aq_1 + b)\sqrt{1 + q_1^2}$, for any adjacent links with relative angles for some constants a and b.

Scale factor $\boldsymbol{\alpha}$: All the models need parameterisation and [50],[51] mention the use of a scaling factor applied to all angular vector computations. This parameter α may be determined online. In describing the Jacobian transpose method in [15], the transpose of the Jacobian,J^T, is used instead of its inverse together with a scalar scaling factor α to determine the angular rates vector equal to

$$\dot{\boldsymbol{\theta}} = \alpha J^T \dot{\mathbf{X}}_{\mathbf{EE}}. \tag{2.17}$$

This is also applied to all of the inverse kinematics models in [63] and this was found to be a requirement in the application of the models for better performance.

The transpose and inverse operators are not the same but the use of the transpose is justified in terms of virtual forces if α is defined appropriately. A guide to the choice of α is proposed by choosing it such that the change in the position of the EE is exactly $\alpha J J^T \dot{\mathbf{X}}_{\mathbf{EE}}$. This involves computing the normalised Euclidean distances between the computed velocity vector and the quantity $\alpha J J^T \dot{\mathbf{X}}_{\mathbf{EE}}$ which gives

$$\alpha = \frac{\|\dot{\mathbf{X}}_{\mathbf{EE}}\| \|J J^T \dot{\mathbf{X}}_{\mathbf{EE}}\|}{\|J J^T \dot{\mathbf{X}}_{\mathbf{EE}}\|^2}. \tag{2.18}$$

This was used to gauge the value of the α value required for a specific application using a specific inverse kinematics method.

2.6.5 Damped Least Squares

The damped least squares (DLS) method (also known as the Levenberg-Marquardt method) bypasses computation of the inverse or pseudoinverse in solving for the minimum $\dot{\boldsymbol{\theta}}$ that satisfies the steepest descent and instead it determines the solution vector for $\dot{\boldsymbol{\theta}}$ that minimises the quantity $\|J\dot{\boldsymbol{\theta}} - \dot{\mathbf{X}}_{\mathbf{EE}}\|^2 + \lambda^2 \|\dot{\boldsymbol{\theta}}\|^2$. The corresponding normal equation can be rewritten that includes the kinematics variables together with the matrix $\left[J^T J + \lambda^2 I\right]$ which is guaranteed to be non-singular and thus ensuring the algorithm's numerical stability. Since $J^T J$ is a square $n \times n$ matrix, it can be shown that

$$\dot{\boldsymbol{\theta}} = J^T \left[J J^T + {}^2 I\right]^{-1} \dot{\mathbf{X}}_{\mathbf{EE}}. \tag{2.19}$$

It should be noted that the damping constant does affect the stability of the computation; the parameter λ must be large enough so that the solutions near singularities are stable but as the factor increases the convergence rate may be too slow. It will be shown later that optimal parameter values vary (within a relatively narrow band here) and a more accurate and robust way may be to use a dynamically varying parameter as proposed by [63].

A variation on the DLS method is the selectively damped least squares (SDLS) method where numeric filtering is selectively applied to all singular vectors of the

Jacobian and where the damping (accordingly applied to the inverse proportionate amount to the range of motion that is appropriate) is defined in terms of a difficulty rating in reaching the target [65]. The damping factor here is adjusted manually and it is increased until unwanted oscillations were eliminated but at the expense of accuracy in tracking the target positions.

2.7 Inverse Kinematics for Trajectory Generation

Problems with inverse kinematics methods are that they may have multiple or infinitely many solutions, no solutions or no closed form (i.e. analytical or algebraic) solutions, the latter being applicable if the number of constraints is equal to the number of degrees of freedom. The inverse kinematics methods discussed here determine the joint positions of a mechanism for a given absolute position of the end effector by solving $\boldsymbol{\theta} = f^{-1}(\mathbf{x})$, as opposed to an analytical solution (applicable when the number of degrees of freedom is equal to the number of unknowns), where an approximated change of the position of the end effector that corresponds to a joint angle change is given by a column of the Jacobian matrix.

$$J(\boldsymbol{\theta}) = \frac{\partial f}{\partial \boldsymbol{\theta}}. \tag{2.20}$$

Since the Jacobian is a linear approximation of a function in the vicinity of a point (see equation 2.20), provided that the function is differentiable the approximate changes in angle variables required for the end effector position $\mathbf{X_{EE}}$ of a mechanism for very small time intervals are

$$\Delta\boldsymbol{\theta} = J(\boldsymbol{\theta})^{-1}\Delta\mathrm{X}_{\mathbf{EE}}. \tag{2.21}$$

If $\mathbf{X_t}$ is the target position of the end effector and $\mathbf{X}$ is its current position then $\Delta\mathrm{X}_{\mathbf{EE}} = \mathbf{X_t} - \mathbf{X}$. If the Jacobian is about to lose full row rank or if the target position is out of range then movements will be irregular and jerky.

2.8 Overview

Introductory notes on deep learning and the use of DeepLabCut in returning the estimated user defined markers are given firstly. Next, the derivation of the virtual camera model is derived from basic geometry and includes the parameters that model the physical characteristics and operation of the camera. The theory on polar transforms allows an understanding of how the vector fields are projected onto different axes. This concludes the background theory for the tools that are used to obtain and interpret the relevant data.

The basic mathematical operation of the least squares optimiser is presented next. A detailed account of how the solution vector is determined in hyperspace by way of a simple two parameter model illustrates the mechanics of the optimisation process. Lastly, the theory of inverse kinematics introduces the various methods (and their parameters) that are used in determining the angular rates vectors for the quadruped links and the way in which they are able to generate end effector trajectories.

Chapter 3

Systems and Metric Definitions

This chapter defines the link arrangement and joints of the quadruped mechanism as well as the local and global quadruped coordinate systems. The local camera coordinate system and its relation to the local system of the robot is also shown. Pertinent aspects of phase states and their trajectories are also discussed. The degree of closure metric (formulated by the author), useful as a quantitative measure for expected closed loop trajectories, is also defined.

3.1 Configuration Definition

The number of degrees of freedom of the entire mechanism is given by the Chebychev-Gr*ü*bler-Kutzbach criterion (also known as Gr*ü*bler's formula) depending on the type and number of joints, whether the links move in a planar or spatial field and the number of links that constitute the mechanism. If N is the number of links and J the number of joints then Gr*ü*bler's formula is given by

$$N_{DoF} = m(N - 1 - J) + \sum_{i=1}^{J} f_i, \tag{3.1}$$

where f_i is the number of freedoms provided by joint i (or the number of axes associated with joint i). For spatial links $m = 6$ and $m = 3$ for planar links. If c_i

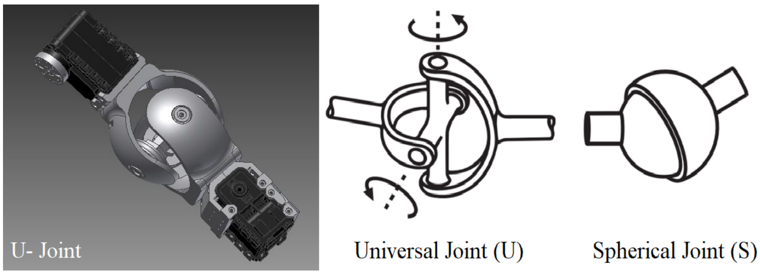


Figure 11: Universal (U) and spherical (S) joints. All joints for the quadruped serial link mechanism are based on universal joints. If spherical joints were to be used an additional rotational axis would be introduced allowing the links to twist relative to their parent links.

is the number of constraints provided by joint i then $m = f_i + c_i$ can be used as a check, although for planar mechanisms c_i can be difficult to ascertain. For a rotary or revolute (R) joint $c_i = 5$ and $f_i = 1$, for Hooke's universal (U) joint $c_i = 4$ and $f_i = 2$ and for a spherical (or ball socket) (S) joint $c_i = 3$ and $f_i = 3$, assuming all link bodies are spatial. It should also be pointed out that if the mechanism is permanently fixed to a ground contact then this will affect the value of N. For each serial system that has one or both ends in ground contact, N increases by one.

According to the Chebychev-Gr$\ddot{u}$bler-Kutzbach criterion, the fixed mechanism shown in Figure 12 will have 20 Degrees of Freedom (DOF) since all links are spatial as opposed to planar. The spine-front-leg system, that creates the EE path within the local robot system, has $DOF = 10$ according to Gr$\ddot{u}$bler's formula, with $m = 6$, $N = 6$, $J = 5$, $f_i = 2$ and with the fixed reference joint effectively being pinned to ground, hence the additional value for N. Once again this manipulator operating in a three-dimensional workspace is redundant by 4 DOF.

DiegoSan is a pneumatic humanoid robot with the number of DOF equal to 44 and it makes use of air pressure to simulate muscle activations with a relatively long time constant of about $80ms$. By using a well parameterised model and reference trajectories, its proportional derivative (PD) controller ensures that the EE paths, speeds, stability and so on are acceptable thanks to the damped dynamics generated in this way. The primary motion plane is the sagittal plane and the optimisation is largely guided here with the transverse plane containing the depth coordinates. Also of interest is the frontal plane where limb deviations out of the sagittal plane

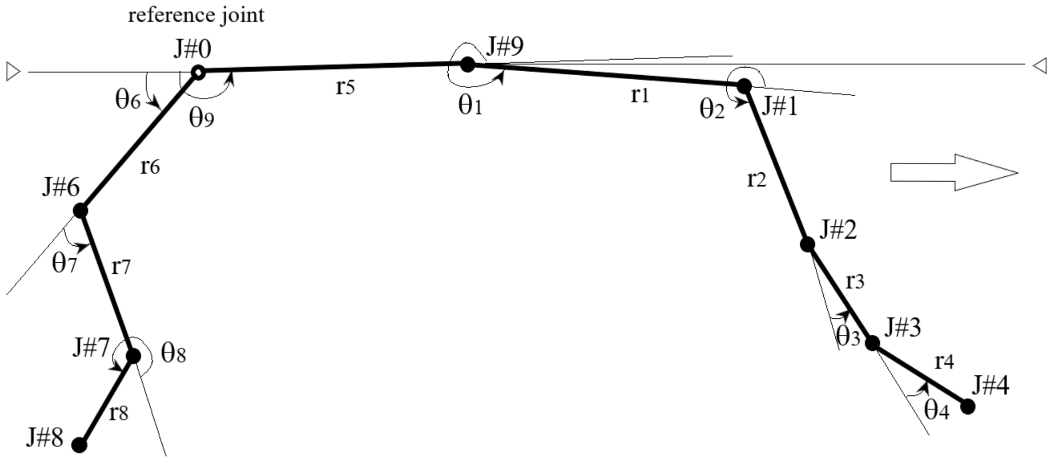


Figure 12: The quadruped 8-link configuration that represents the animal subject's locomotion generating limbs and appendages. Translational motion is directed from left to right with the reference joint J#0 at the junction of link r_5 (spine) and r_6 (back leg) of the mechanism. All angles are measured relative to their parent link with θ_9 being measured relative to a horizontal reference line, the U axis of the local robot system the origin of which is the reference joint.

can be seen directly, and this is helpful in visualising the motion particularly during the deceleration phase. The twist angle γ of each link is excluded in the modeling of

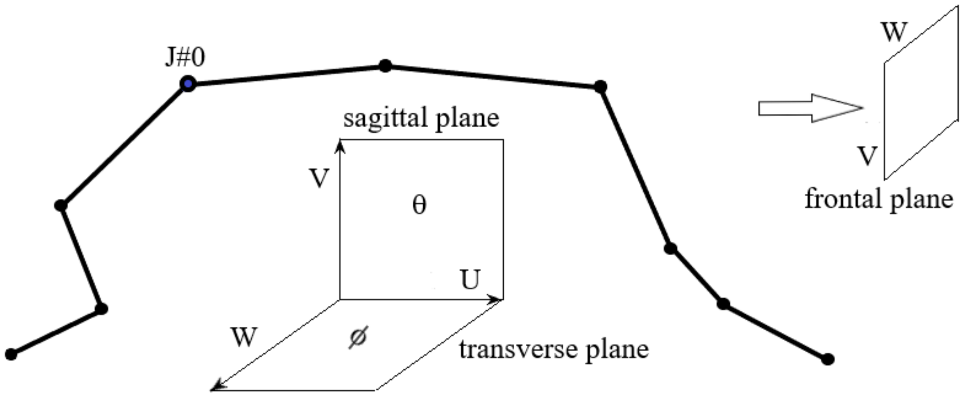


Figure 13: All the descriptor planes that relate the local link motion of the quadruped with respect to the reference joint ($J\#0$). The three planes are the sagittal (UV) side, transverse (UW) top elevation and frontal (VW) planes. The permitted angle rotations θ and ϕ are shown in their operational planes. Link twist angles γ are all assumed zero for simplicity.

the quadruped. Each link angle is defined according to a relative joint angle system

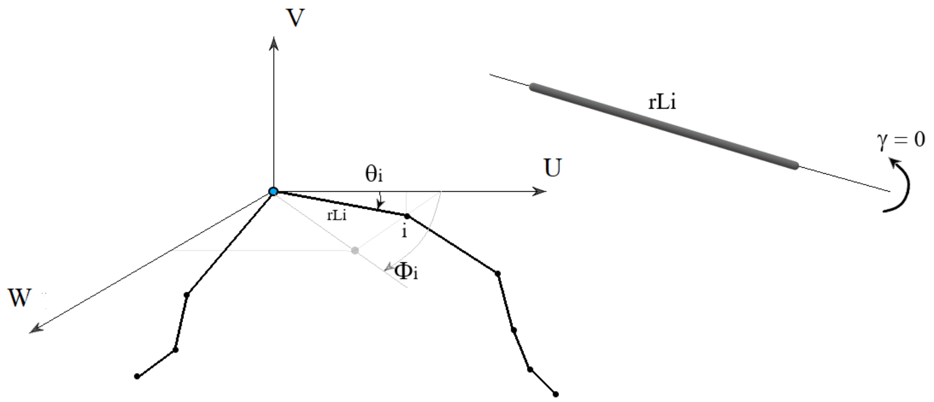


Figure 14: The canonical local robot (UVW) coordinate system showing the angle definitions in their respective planes for link rL_i. The origin of the UVW frame is the reference joint. Link twist angles are not modeled so γ is zero for all links.

based on a reference provided by its parent link and in this way it complements the theoretical aspects that operate. Firstly, this assists in iterative applications where angular rates are consistently defined and comparable and secondly it complements the inverse kinematics formulations since the solutions are obtained numerically such as with [14]. An infinite number of angular rate joint vectors exist for a specified end effector velocity in Cartesian coordinates for a redundant manipulator; in other words, when the number of angle variables is less than the dimension of the task space of the mechanism.

3.2 Coordinate Systems

The camera or observer's coordinate axis system is related in its relative position and orientation to that of the local robot system by the rotation and translation camera extrinsic variables R and t. In order for the optimisation process to work, it must perform operations using variables that uniquely define the reality of the relative positions and orientations of the two systems. Each configuration thus has the spine link fixed to the reference joint coinciding with the U axis; θ_9 and ϕ_9 are

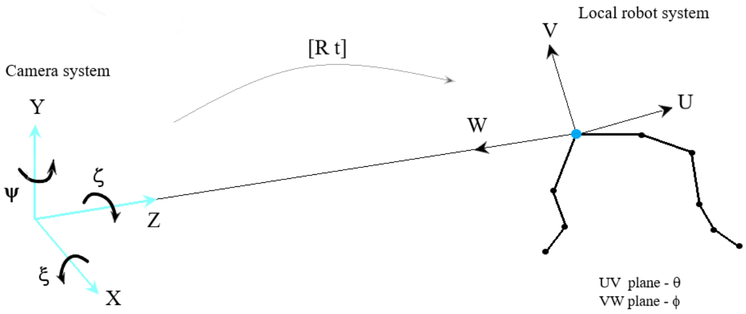


Figure 15: The camera and local robot coordinate systems. The camera rotations about each of the XYZ axes are ξ, ψ and ζ respectively. The first spine link coincides with the U axis of the local robot system frame and its origin equates to the reference joint.

therefore fixed at zero. The 3D coordinates of the reference joint are variables that need to be solved for, instead of t_x, t_y and t_z, and over the full sequence will provide a trace in space of its position. The camera rotation variables are to be solved for and essentially provide the relative orientation. The R matrix components are to be solved for and the t translation vector is fixed at zero. It is assumed that the robot comprises Hooke universal (U) joints so it is modeled as an $8U$-serial open chain mechanism. Only the three dimensional position coordinates are of interest here with the simplification that the links themselves are not able to twist (γ=0) which would require a spherical (S) joint system. A high speed quadruped is being considered here as opposed to a highly dextrous surgical manipulator and hence this simplification is deemed reasonable.

3.3 Transforms

Once the coordinate systems have been translated the rotations are applied. For a given angle rotation about an axis a rotation transform matrix can be generated that effects this orientation in the context of the virtual camera model. The structure of this matrix can be derived using polar transform theory. Since the Jacobian matrix describes the amount of local warping, it can also be employed to derive the rotation transform provided the component functions are described appropriately. Another way is by considering the eigenvalue representation for a system undergoing a purely rotational transform. By taking note of which transform direction (polar to Cartesian or vice-versa) is applicable for each respective rotation the matrices are derived.

3.4 Phase States

The whole question of the state of a system, whether it be a navigation problem or the values of the variables that define the position and dynamics of a robot for instance, all contain random variables that are uncertain and non exact. In formulating a description of a system one seeks a definition that includes independent variables that uniquely and simply define the system. The motion of a simple pendulum for instance can be uniquely described by a generalised coordinate such as the angle θ that it makes with respect to a reference line with the state of the system being described by its position in the phase state space, in this case the coordinates θ and $\dot{\theta}$. The trajectory trace is called the phase portrait or phase state diagram and they form a critical analysis tool in non-linear systems theory where additional information such as stability and theoretical orbital paths may be studied or simulated.

In the present context the configuration state of the quadruped is defined in terms of its generalised coordinates θ and ϕ, so a full graphical representation of its state will be the state space trajectories of all its joint angles in each of the two phase spaces.

3.5 Degree of Closure (DOC) for a Trajectory

A useful metric to quantify the degree to which a phase space trajectory in a particular space is closed is the final separation distance between the start and final points of the trajectory given by D_{sf} as shown below for an arbitrary path. This is a good

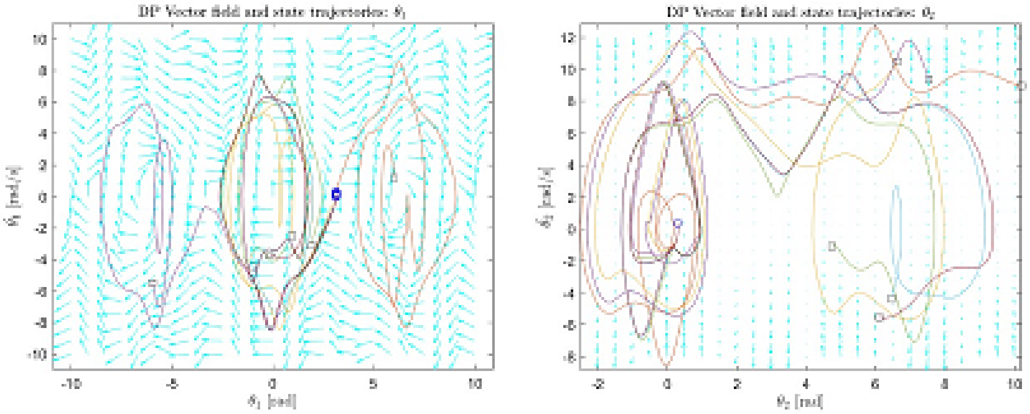


Figure 16: Illustrative phase state diagrams for the double pendulum model with the associated vector fields (shown as the cyan coloured arrows). The generalised coordinates of the position of the top and lower point masses are θ_1 and θ_2 respectively. Investigations were done using this model to provide simulated data for a two link serial planar mechanism, however this work did not fall within the scope of this project.

indicator of performance if the relative trajectories being considered have varying ranges and/or profiles. For comparisons between models, the percentage difference

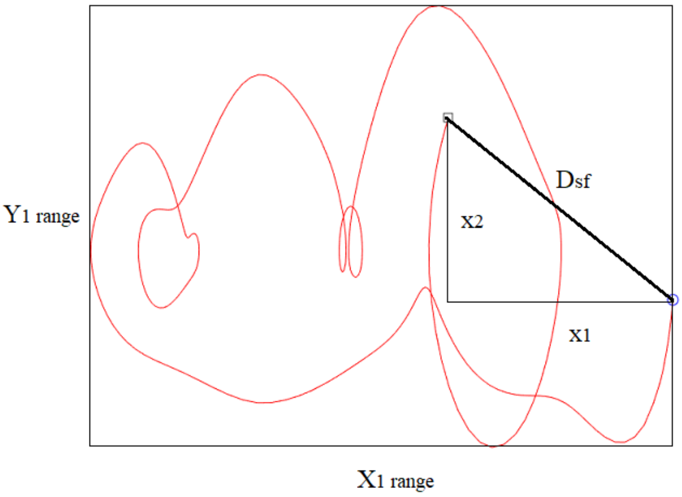


Figure 17: The diagram describing the parameter D_{sf} metric for state trajectory closure. $Y1$ relates angular rates for the given angle domain $X1$ for the state space of a particular variable. The distance between the start (circle) and end (square) points indicates the degree of closure of an orbital path.

of the error in respect to the variable's maximum ranges, $max\{D_{sf}\}$, given by

$$D_{sfERR} = \frac{D_{sf}}{max\{D_{sf}\}}, \tag{3.2}$$

can also be considered. The comparative metric used to assess the performance of the inverse kinematic models is the degree of closure as a percentage:

$$D_{OC} = 100\left(1 - \frac{D_{sf}}{max\{D_{sf}\}}\right). \tag{3.3}$$

If the start and end points of a trajectory are identical then $D_{OC} = 100\%$ and there is full closure while $D_{0C} = 0$ will mean that these points are maximally separated.

Chapter 4

Methodology

In order to solve the problems at hand, a logical flow from the raw video data to the estimated poses and outputs for tracking are implemented. A flow chart that chronicles the steps involved shows the proposed integrated solution system in addition to the theoretical concepts required. The generation of a robustly trained convolutional neural network that is able to provide pose estimates in the form of image data of the animal subject is the first step. The theoretical functionality of the optimisation process is built around a structured set of assumptions regarding the morphology and dynamic constraints of the quadruped. The parameterised kinematics models then use the end effector trajectory coordinates to deliver an angular rate vector for each configuration instance and for the entire time sequence. Kinematics is concerned with the linear and rotational variables that quantify the geometry of motion and its derivatives of a system. Specifically, inverse kinematics computes the joint angles to produce desired end effector trajectories, and so giving the motion plan of a robot.

Relative link angles were used and all motion was relative to a single reference joint, since the aim here is to investigate the kinematics and generate joint trajectories within a local coordinate system. Any global tracking would include a moving reference point since the idea of a static viewpoint and a dynamic system target would necessitate this. On the other hand, the kinetic variables, such as the mass of the links, momenta, reaction forces and applied torques, produce those trajectories.

4.1 Data Acquisition

The use of single imaging sensors is a powerful tool for acquiring sensor data since it is remote, unobtrusive and a system configuration can be acquired. A trained CNN outputs its marker estimates of the quadruped in the pixelated 2D image plane and this data is then offered as the primary input to the least squares optimisation solver. Additional input data for the solver are the expected link proportions, 3D path traces of the reference joint and the nominal selected camera angles as well as the expected angular rate maxima and approximate angle ranges for specific links. The end effector path as estimated by the optimiser then becomes the target trajectory for the inverse kinematics models.

An optimisation algorithm was executed in order to resolve all of the angles in 3D and thus acquire an accurate kinematic solution for a sequence of quadruped configurations. For motion analysis, the subject is modeled as a reduced 8-link universal joint mechanism (see Figure 12) representing the key appendages of the subject.

Machine learning: Machine learning in the form of a CNN and using DeepLabCut (DLC) [38] ResNet50 and a Graphical Processing Unit (GPU) (Nvidia GTX-1080i) processor on a Linux based machine was used for markerless tracking of strategic points. The network was trained and unseen video image sequences were used for data generation of the subject under various gait modes, including nominal trot, high acceleration and deceleration modes. The result is a network that will be used to estimate the image points of all the markers on the accelerating subject which is then passed onto the optimising algorithm for the extraction of all spatial configuration poses. The structure of the trained CNN is identical to that described in Section 2.2 by Chen [48].

4.2 Objective Vector Functions for the Optimiser

The non-linear LSQ approximation was used for the optimisation. All vector functions are non-linear and they collectively reflect the statics and dynamics of the system. In addition to static configuration resolution (image and angle ranges), the vector functions also had to reflect the limits of dynamic motion. The primary motion plane is in the UV plane, in which θ is defined (see Figure 13), and together with the transverse plane angle variable ϕ and all the link length variables, completes the descriptor variables for the spatial locomotion system. As is to be expected, θ will show the maximum values and ranges and constraints relating to this parameter are critical to the LSQ computation. Of secondary importance is motion that deviates from the sagittal plane of the subject and ϕ is assumed to lie in a fairly narrow band for non-accelerating scenarios with marginal increases expected for the accelerating case. The links were assumed rigid and were all scaled according to a single variable base link with length r according to anatomical data and relative lengths from the actual video images. Vector fields generated by all the joints provide insight into the relative motion of the joints and links, so a vector with origin at a specific joint and at the start of a time interval determines the joint position after a time interval. Since the links are all of different lengths, angular and tangential velocity components need to fall within limits, which are determined by the maximum changes of the (relative) angles and their corresponding link lengths. Polar transforms were used to project variables onto a reference frame determined by the parent link where all trajectories were locally circular, since link lengths are assumed rigid. Visco-elastic links would model the system better, but increase the complexity. Approximate actual angles were extracted visually and roughly measured so that realistic expected values for the Levenberg-Marquardt algorithm could be used, for which it is highly sensitive. The Levenberg-Marquardt algorithm was used since the system is underdetermined with the number of unknowns greater than the number of (vector) equations. By ensuring that the initial guesses of all the variables at the start of the search have been carefully selected and within reasonably close proximity to the optimal solution vector, the algorithm is more likely to converge to the actual solution. The magnitude of the squares of the residuals at solution is a key indicator of whether the solutions are near optimal. The other indicators discussed earlier should also be analysed to provide a clearer picture of the accuracy of the optimisation process.

The main vector function $F(x)$ had the generic form

$$F(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \\ f_3(x) \\ f_4(x) \\ f_5(x) \end{bmatrix}, \tag{4.1}$$

with components representing each optimisation strategy. The multivariate variable x is the solution vector that needs to be solved for and it comprises all the angles and base link length of the quadruped and the coordinates of the reference joint and camera rotation angles. The LSQ algorithm then searched for parameter values θ, ϕ and r, to which all other links were referenced, that minimised the square of difference functions as follows,

$$\min_x \|F(x)\|_2^2 = \min_x \sum_{n=1}^{5} f_i(x)^2. \tag{4.2}$$

The optimisation used as actual inputs the virtual image pixel values Λ_i for each of the nine joints and for all thirty configurations. This was the main optimisation strategy which involved the use of a mathematical virtual camera projection model with intrinsics derived before. The 3D world system points as functions of θ, ϕ and r were inputs to the camera model and the resulting LSQ image, Γ_i was optimised for all the number of configurations given by C, according to:

$$f_1(x) = \sum_{i=1}^{C} (\Gamma_i - \Lambda_i)^2. \tag{4.3}$$

As a further refinement, the optimisation was done over expected angle ranges and all operating within the [-π,π] range. Each specific joint angle had an expected range and an expected maximum deviation. The optimisation was done with a vector function of the form

$$f_2(x) = \sum_{i=1}^{C-1} (\delta_i |\Theta_{ave} - \Theta_i| - \epsilon_i)^2. \tag{4.4}$$

Of course, the depth dimension represented by ϕ also contributes to the accuracy of the estimates; not taking this into account, would allow the LSQ algorithm to press into this domain unconstrained and thus follow an incorrect search path:

$$f_3(x) = \sum_{i=1}^{C-1} (\beta_i |\Phi_{ave} - \Phi_i| - \rho_i)^2. \tag{4.5}$$

Tuning parameters δ and β penalised the search for infringements on the assigned range for the angles and their allowable tolerances, specified by the vectors ϵ and ρ, respectively, and subscript *ave* denotes the average (mean) value.

Derivation of the Polar Vector Velocity: Up to this point, only absolute changes of a single variable have been resolved. By introducing dynamic modeling equations where relative changes are involved, further search constraints are introduced that keep rates of changes of key parameters in check, and account for relative increases and decreases in changes of angles. The search is thus steered in the correct direction within the bounds of a sound theoretical basis. If this key structural model is not included the search domain is extended beyond reasonable operating ranges of $\dot{\theta}$. With reference to Figure 8, the magnitude r of the vector $\mathbf{r}$ is derived by using the cosine rule and applied to the isosceles triangle with sides r_i, r_i and $\mathbf{r}$ and subtended angle variable $\alpha = \alpha(t)$ which gives,

$$r = \sqrt{2r_i^2(1 - \cos\alpha)}, \tag{4.6}$$

with only α being a time varying variable. Furthermore, refinement was therefore done using vector calculus and the generation of vector fields so that a vector $\mathbf{r}$ for a specific angle and joint was generated using polar transformations, and it points from the joint position at the start of a time interval of length T to its position at the end of the interval. In other words, it subtends the relative angle change and the linear sector length traversed by the joint so that, by taking the time derivative of $r = r(t)$, the polar vector velocity magnitude relation is

$$\|\dot{\mathbf{r}}\| = \dot{r} = \frac{r_i \sin\alpha}{\sqrt{2}\sqrt{(1-\cos\alpha)}}, \tag{4.7}$$

with α being the relative change between link angles relative to their parent link during the sample time interval. At the start of a cycle and for a specific joint angle, a vector field for all of the angles defining a single configuration is generated with θ and ϕ implicitly defined. Thus the underlying statics and dynamics are accounted for by incorporating the link lengths, angle and angle rate variables between each configuration change:

$$f_4(x) = \sum_{i=1}^{C-1} (\dot{\mathbf{r}}(\Theta) - r_i \dot{\Theta}^2_{max})^2. \tag{4.8}$$

The equivalent formulation in the UW plane using ϕ is then

$$f_5(x) = \sum_{i=1}^{C-1} (\dot{\mathbf{r}}(\Phi) - r_i \dot{\Phi}^2_{max})^2. \tag{4.9}$$

Subscript max denotes maximum expected value here. The resulting vector had dynamic properties with its associated limit (incorporating maximum angular velocity). With each new configuration, new dynamics that have essentially been driven by allowable tolerances are computed. The set values of maximum allowable angular rates are subjective, however, detailed data analysis allows for reasonable limits for each link. Each joint trajectory had relative motion that was nearly circular with respect to its parent joint and approximated thus. Obviously, the joint-to-joint polar vectors would follow along discrete sector lengths that could, for all practical purposes, be approximated as linear since the sample times were small. A rotation matrix can be used to map these vectors, expressed in the body-fixed coordinate frame to a representation in the inertial coordinate frame. Other attitude descriptors include exponential coordinates, quaternions and Euler angles. The main vector functions that form the objective optimisation function are derived from basic theory and whose collective minimisation are theorised to ensure solutions that abide by the proposed statics and main dynamic constraint conditions of the quadruped. Comparisons with the quadruped are not easily made under accelerating conditions since ground contact forces are highly variable in addition to the obvious differences in physiology. The vector functions are formulated for the LSQ algorithm that reflect the basic geometry, physiology and expected dynamics of the quadruped.

Additional parameters that model the deviation from the sagittal plane also serve to accurately make allowances for this. Estimated link angle data obtained from the actual video sequences of the animal subject are crucial for accurate optimisation. Further refinements are required to limit searches within acceptable tolerances that relate changes in configurations within incremental limits that mimic that of a real life system in addition to physiological considerations. The control of robots requires knowledge of all angular velocities and inverse kinematics provides a way of computing these indirectly. The translational motion speed is governed by step lengths which in turn requires sequential placement of the end effectors. The tracking of the EE and the required discrete angle values (and thus instantaneous angular rates) are thus vital for enabling their motion gaits. Usually, for a complex system such as the quadruped, an Inertial Measurement Unit (IMU) is practically used to monitor and track the position and velocity of the COM [14], with the inverse kinematics and the computed applied torques forming an integral part of the control feedback scheme. For a sample time interval the next position placement of the supporting legs is required; the high degrees of freedom with their associated high, if not infinite configuration possibilities, means that solving the inverse kinematics problem is extremely challenging. The level of complexity is increased if other control inputs such as differential ground contact forces and non-planar 3D leg configurations are admitted to the solution vector. In addition, for any given cycle between EE ground contacts, closed joint angle space trajectories corresponding to closed end effector trajectories need to be generated, which ensure no discontinuities and minimal oscillations.

4.3 Model Parameters

Optimisation of the kinematics models' estimated angular rate vectors and those obtained in the least squares solutions of the configuration angles obtained previously is hypothesised to return acceptable model parameters.

The data obtained during the optimisation is used as the truth data to parameterise the inverse kinematics models. Firstly, the optimised angle data is interpolated at high sampling rates (using a spline cubic function) and together with the estimated link lengths are used to determine the trajectory of the end effector in the workspace. The angular rates vector and the EE's linear velocity components are then computed and are entered as input data into the least squares optimisation procedure. In this way the parameters are established for the models as detailed in the schematic in Figure 18. The flowchart for determining a parameter of an inverse kinematics model describes graphically how the data vectors are obtained, manipulated and processed to give the parameter values for the entire sequence. The value of the scaling parameter α may be determined as discussed in Section 2.6.4.

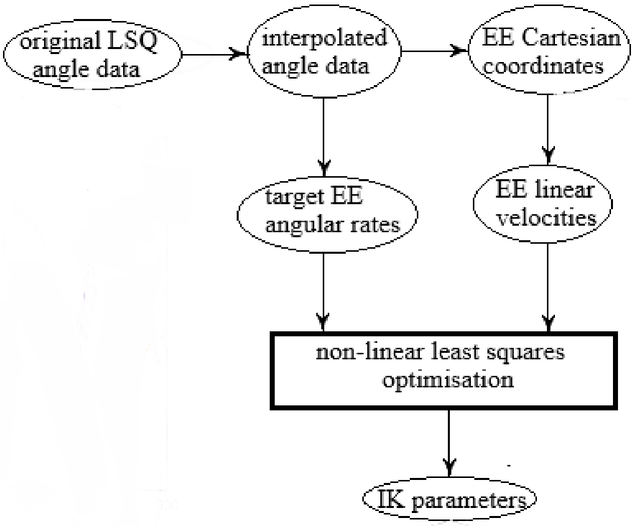


Figure 18: Flowchart for determining the parameter values for an inverse kinematics method.

4.4 Motion Constraints

Robotic control systems ensure that motion is guided within a set of constraints as determined by the application involved or environment. A robot is only able to move within a feasible region defined by the link lengths and the joint limits. If the design is not robust enough, unacceptable behaviour may also occur due to singularities in the model such as when a manipulator is fully outstretched. For underpowered systems the actuations required may not be reachable and smooth manifolds are preferred since target position errors are more likely to be decreasing continuously in contrast with an irregular path where the end effector may be moving away from the target position at times and cause software issues. The latter can be corrected by introducing equality constraints [51] but this was not a problem in this project so they were not included in any of the strategies.

4.5 Project Elements and Work Flow

In order to obtain the final results the basic procedure was to train a network that delivers suitably accurate image data that could be used as input to the least squares optimiser that was developed. The first step is to use a video sequence to manually label the requisite joints by way of a GUI, and using the physical markers fixed to the subject as a guide as to their positions. These labeled images are then used as the training data for the neural network. The labels are then refined and re-labeled as required and the network is re-trained. The workflow therefore progresses from raw video image data that is then processed and used to estimate markers by way of transfer learning. The optimisation solver then gives predictions of the 3D configurations and this data is in turn used to obtain parameterised inverse kinematics models that output the sequential angular velocity vectors for the desired target end effector trajectory. The end effector's path is then the target trajectory for the various inverse kinematics models that in turn produce the actuations required and the results are then compared with similar outcomes expected. The basic flow of the project work and the related components are clarified in Figure 19.

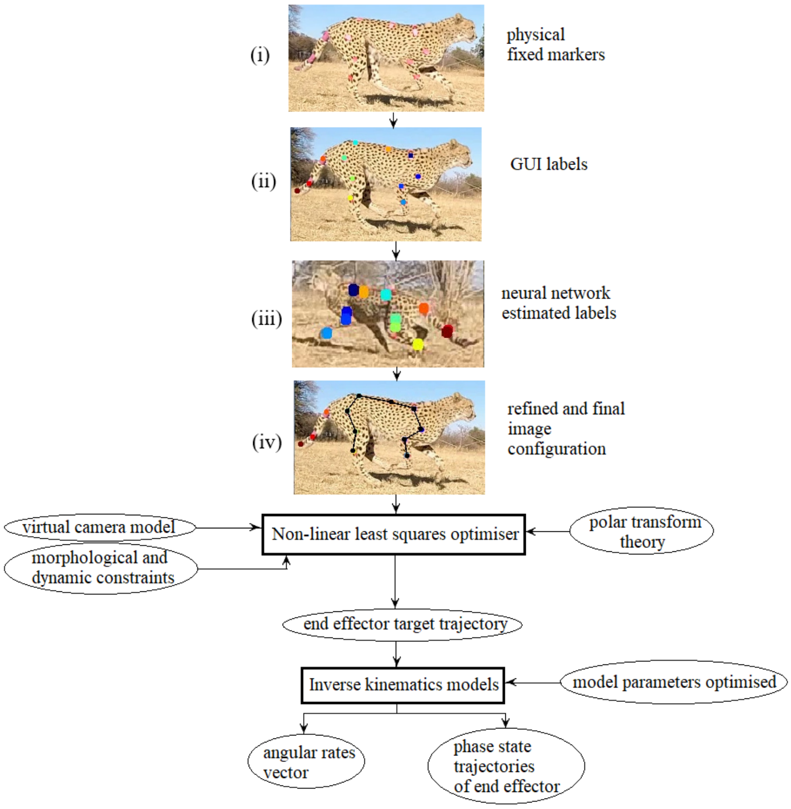

(i)
physical fixed markers
(ii)
GUI labels
(iii)
neural network estimated labels
(iv)
refined and final image configuration
virtual camera model
morphological and dynamic constraints
Non-linear least squares optimiser
polar transform theory
end effector target trajectory
Inverse kinematics models
model parameters optimised
angular rates vector
phase state trajectories of end effector

Figure 19: Flowchart with the design elements, processes, methods and key outputs of this project. The graphics (i) to (iv) show how the video image of the subject is used to obtain the input image for the least squares optimiser.

4.6 Related Work

Previous work done by researchers that overlap and parallel the scope of the contents of this project are mentioned and discussed in this section. It is helpful to consider similar methodologies and study their results and conclusions which will contribute to the general insight and understanding of the problems encountered here.

4.6.1 Deep Learning and Pose Estimation

The definitive publication [38] followed for implementing the transfer learning algorithm, DeepLabCut, in extracting the pose estimates of the cheetah proved invaluable. The process by which the feature detectors are re-trained so that a network that is iteratively improved upon by way of an active learning circle and that is ultimately able to robustly and accurately estimate video sequences where there is large variability in subject poses and lighting was followed. In addition, the video sequence being analysed was captured by a single camera with minimal occlusions as specified by the application, with the work done on the Cheetah Project [38] by Chen and Patel being of particular interest.

The 3D pose estimator developed by Chen [48] also made use of DeeplabCut to estimate the 2D image poses but differed in that it also made use of a multi-camera system and the accompanying direct linear transformation coefficients to synthesise the 3D motion configurations.

Cushway [49] also used a multi-camera system but now included Global Positioning System (GPS) receivers and IMU sensors afixed to the animal subject, relaying all of the signal data to the researcher for capture and processing. An Extended Kalman Filter was developed that included state vectors for each link and one for estimating a reference marker, thereby fully describing the positions and motion dynamics of the quadruped in it's entirety. Similarly, in tracking the upper body poses of a human, research done by [46] involved using a hybrid model; monocular camera images are used as prior distributions that cover the probability of a pose occurring and when combined with a random walk transition model, ensures that the states behave as random walks converging towards a set of commonly observed poses. Since this model demotes divergent poses to less importance it could have been useful in this project.

4.6.2 Modeling of Legged Robots

The planning and efficient trajectory execution of the end effector position of a legged robot within a viable target manifold is it's primary locomotion task. Quadrupeds, bipedal humanoids, hexapods, industrial manipulator arms and more recently robotic arms with flexible links require precise optimal trajectory planning and positioning of their end effectors.

Related publications [14] in this regard involve basic systems modeling (such as the virtual leg method [14]) and the kinematics modeling of the Baxter robot using Denavit-Hartenberg (DH) notations [66]. High level locomotion sequencing, lower level joint actuation and control, contact positioning and timing with the environment and the theory pertaining to inverse kinematics, Jacobians and related matrix manipulations such as QR decomposition and inversions are also dealt with in [1],[18],[19],[20],[25],[35],[40],[42],[43]. The inverse kinematics of the quadruped control method outlined in [14] was particularly relevant in the work that was done on trajectory planning of the end effector in this project.

4.7 Overview

The essential work flow and methods used are detailed here. The acquisition of the input image data using DeepLabCut is described followed by a detailed description and functions of the components of the objective function as well as the derivation of supplemental equations by the author. The working methods for parameterising the various kinematics models' are then given, followed by notes on motion constraints in general. The following section then attempts to clarify how all of the elements involved in the project appear within the work flow. Finally, related publications that were most useful and that supplemented and paralleled part of the work here are cited.

Chapter 5

Experimental Setup, Data Acquisition and Implementation

The practical aspects of acquiring the data of the animal subject, training of a convolutional neural network for pose extraction and the implementation of the optimisation and inverse kinematics are presented in this chapter.
The main points of the practical work done are as follows:

- **Practical Setup:** Trajectory paths were obtained using a four legged animal subject, the cheetah (*Acinonyx Jubatus*), Africa's most endangered big cat; this is the world's fastest land mammal and it is able to accelerate to over $110 km.h^{-1}$ (70 miles per hour) in just over three seconds and with a stride length of seven metres at top speed. Fixed cameras were used to obtain different motion sequences of the subject from different angles and for various gaits, including medium trots which included two repeating gaits in a period of about a second and over a distance of about 9 metres, as well as an accelerating motion from standstill over about 13 metres that entailed three gait cycles in just under a second. Markerless tracking was then done using machine learning by way of a convolutional neural network from which a sequence of actual image point configurations were obtained. These were eleven link configurations that were a simplified geometric representation of the subject.

- **Vision System:** A vision system needs to be developed in conjunction with transition equations that solve for 3D space coordinates using angle data and link lengths only. An optimisation algorithm, which will assist in an understanding of the dynamics of the system, is required which resolves the robot configuration generated by the vision system for all of the sequences with that of the actual configuration data.
  All coding is done in Matlab with some of the optimisations duplicated in Python for verification and confirmation of the basic implementation.

- **Image Data:** The animal subject had numerous physical joint markers fixed to its skin, so that reasonably accurate labeling for the CNN network could be done. The subject would make numerous runs at various gaits and path lines, including standard uniform trots to highly variable accelerations, decelerations and orientations, although they were largely across the general local imaging origin.

- **End Effector Data:** The inverse kinematics models require the instantaneous linear velocities, at high sampling rates, of the end effector. This data is obtained by using the optimisation solver's position data and sampling rate, $T = 0.0333s$, to find the linear velocity components which are then interpolated using a cubic spline and at ten times the original rate.

5.1 Markerless Pose Extraction using a Deep Neural Network

A neural network is trained using video data of the subject so that it can be used to predict the user defined labels on any unseen video. The images are manually labeled using GUIs, and the network is trained, evaluated, refined and re-trained if required. The unseen video of the accelerating subject is then submitted to the network for the image estimates of the labels.

5.1.1 Network Training

Multiple video sequences of the animal subject were used as training and test video data for the DLC algorithm. The subject had markers affixed to it to assist in the manual labeling of the strategic joints identified that would best describe the motion model. Unlike the human subject, the apparent link lengths were variable in appearance. The spine was configured as a 2-link system, however, the links behaved semi-rigidly and more like that of a series of visco-elastic appendages than rigid metal links. The deformation was more apparent in high acceleration conditions.

The learning network was trained under various gait cycles, lighting conditions and levels of visibility due to spurious dust clouds, and occlusion due to extreme sagittal plane deviations and limbs that blocked the line of sight on specific joints [48],[49]. The quadruped was modeled as an ideal 8-link rigid mechanism with variable link lengths based on a single reference link. The links were therefore in proportion to each other based on actual anatomical data for the species. A reference joint was identified and angle measurements were relative to each other or to a reference level passing through it depending on the angle concerned (see Figure 12). Tracking of the reference joint would give the motion trajectory in a translational 3D sense and all link motions would be with respect to this reference joint; thus angle data is all relative here. If one was to model air resistance for example, account would have to be taken of the fact that the subject is capable of speeds of almost $120 km.h^{-1}$ so the actual links experience substantial opposing forces, since the local motion trajectories still need to be taken into account.

User defined labels of the subject are defined according to existing physical markers strategically placed on it.

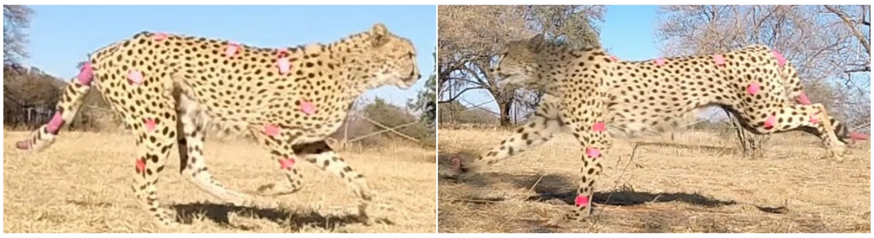


Figure 20: Animal subjects with markers that are physically fixed on strategic joint positions to facilitate manual image labeling for DeepLabCut CNN training data.

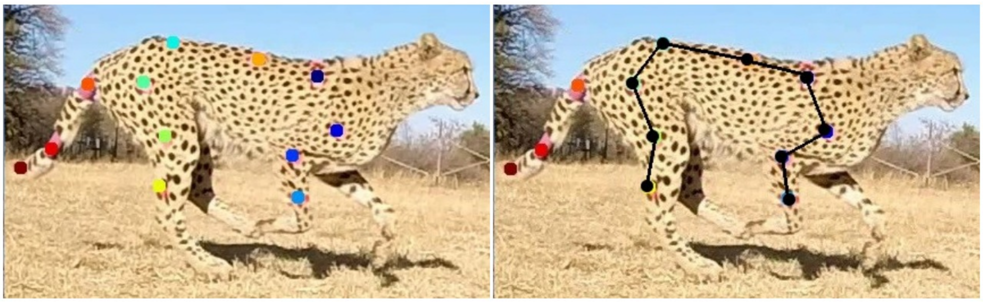


Figure 21: Animal subject with CNN-Resnet50 predicted joint positions by DeepLabCut and a composite graphic with the 8-link configuration superimposed.

5.1.2 Joint Tracking

Markerless tracking of an animal subject was done based on a transfer learning algorithm. This entailed using a convolutional neural network that is trained, using deep learning methods, for a specific task and then applied to a different but related task using a small supervised data set. A state-of-the-art human pose estimation algorithm (DeeperCut) was amended for tracking of user defined features using a relatively small user defined data set of labeled images resulting in DeepLabCut [38]. This deep learning algorithm is robust for estimating misaligned poses, i.e. no fixed position within an image frame, and the powerful feature detectors are able to track points in a variety of lighting and dynamic conditions. It is geared towards behaviours that are consistently captured by one or multiple cameras but with minimal occlusion [38]. It is able to robustly extract body points from a dynamic and cluttered background. In addition, if a feature is lost while tracking due to occlusion or motion blur for instance, it will be detected when visible again unlike in some other imaging methods which require consistent tracking across the video sequence.

5.1.3 Coding

All software is coded in Python Version 3 while the feature detectors are in TensorFlow. The feature detectors are a powerful tool that enable user-defined body points to be learned in specific scenarios.

It has several limitations though; it, ideally, requires a GPU since thousands of iterations are required for adequate training (using a Central Processing Unit (CPU) would take days), the CNN scales with the image pixel size, and occluded points are not tracked which is problematic in multi-legged tracking with the view point orthogonal to the largely translational motion. Since the DLC toolbox code is written in Python 3 commands are entered through a user defined virtual environment and on a Python command window. All the relevant libraries need to be installed, including specific Python packages for graphical user interfaces. Training data is created, the network is trained and evaluated with optional refinement steps and re-training should accuracies not be acceptable. The network can be re-trained in future and used for cross-species applications. A typical training session would make use of about 150 training images (training videos included a GoPro5 camera directed into the sun) and terminate after 33000 iterations as the DLC error loss plateaued at 0.002 with an initial learning rate (determined implicitly by the code) of 0.005 which increased to 0.02 (see Figure 22). Any inaccuracies in the labeled image joint positions will filter down through the LSQ optimisation code and affect the validity of the resulting estimated 3D coordinates. The fact that the external fixed labels on the animals subject's skin do not exactly represent the musculo-skeletal system will have consequences as to the reliability of the data produced, particularly in configuration sequences with irregular and high deformation rates; this was the case in the deceleration analytics where the optimisation performed sub-optimally.

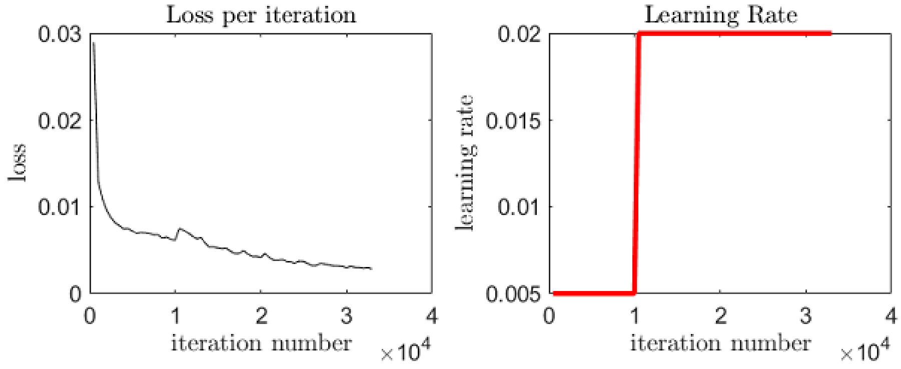


Figure 22: Learning rate data for a typical network training session showing the reduction in errors (loss) and the learning rate variation as the number of iterations progresses.

Output in the form of trajectory plots, the probabilities that the estimated points are correct and the network predicted labeled image sequences of the unseen subject is presented.

Video sequences that showed the subjects at close range were favoured over subjects further away, despite acute viewing angles (either at the start or end sections of a video). It was also presumed that, besides the difficulties in labeling, the increased distance would also introduce higher errors in prediction by DLC due to the lower margins.

The full image size is 1920×1080 pixels but for the purposes of network training they were cropped to more workable sizes in accordance with [38]. The camera (or observer's) coordinate axis system is related in its relative position and orientation to that of the local robot system by the rotation and translation camera extrinsic variables. In order for the optimisation process to work it must perform operations using variables that uniquely define the reality of the relative positions and orien-

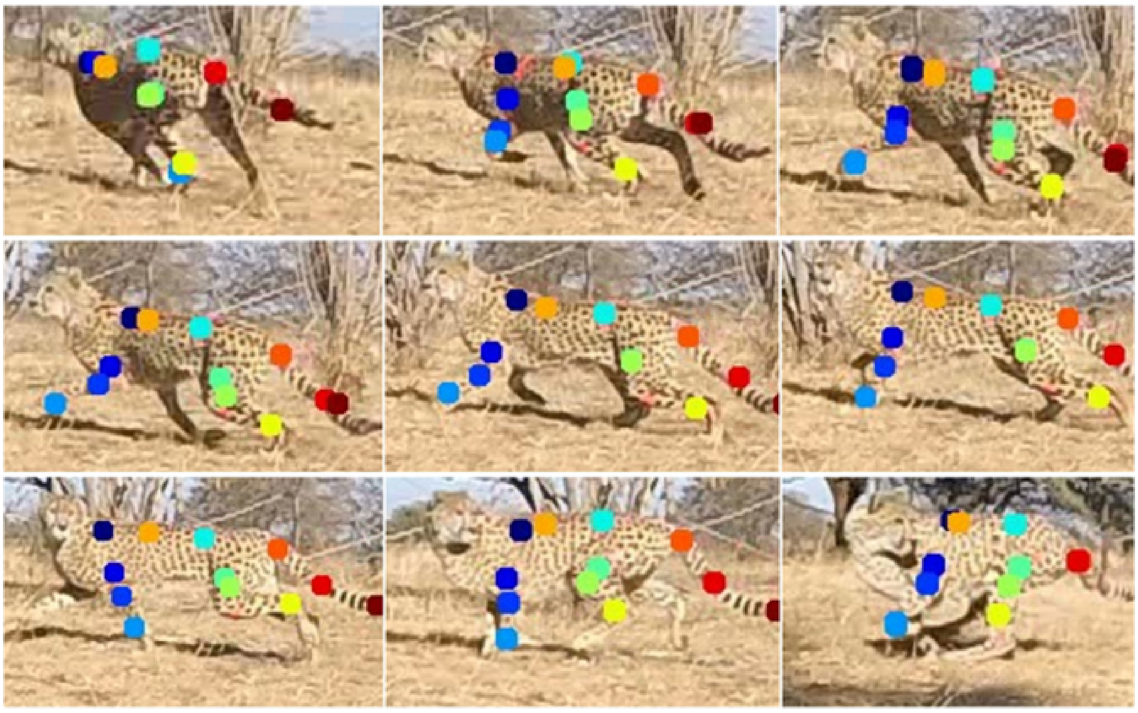


Figure 23: The subject at various orientations and magnification showing the labels as predicted by the pose estimation algorithm, DeepLabCut, that is based on deep neural network (specifically CNN) machine learning. The 190×50 pixel blocks are substantially smaller than the average used in for all three of the quadruped gaits investigated here. Blocks of this size were avoided, as far as possible, by appropriate video cropping.

tations of the two systems. Each configuration thus has the spine link affixed to the reference joint coinciding with the U-axis; θ_9 and ϕ_9 are therefore fixed at zero. The coordinates of the reference joint are variables that need to be solved for, instead of t_x,t_y and t_z, and over the full sequence will provide a trace in space of its position. The camera rotation variables are to be solved for and essentially provide the relative orientation. The R matrix components are to be solved for and the t translation vector is fixed at zero.

It is assumed that the quadruped mechanism comprises Hooke universal joints so it is modeled as an $8U$-serial open chain mechanism. According to the Chebychev-Gr*ü*bler-Kutzbach criterion it will have 20 DOF since all links are spatial as opposed to planar and the positions of all the links are all relative to the reference joint. Since the workspace is 6 DOF this makes this quadruped model a highly redundant mechanism. Only the three dimensional position coordinates are of interest here with the simplification that the links themselves are not able to twist ($\gamma = 0$) which would require a spherical joint system. A high speed quadruped is being considered here as opposed to a highly dextrous surgical manipulator and hence this simplification is deemed reasonable.

Once the coordinate systems have been translated the rotations are applied. For a given angle rotation about an axis a rotation transform matrix can be generated that effects this orientation in the context of the virtual camera model. The structure of this matrix can be derived using polar transform theory. Since the Jacobian describes the amount of local warping it can also be employed to derive the rotation transform provided the component functions are described appropriately. Another way is by considering the eigenvalue representation for a system undergoing a purely rotative transform. By taking note of which transform direction (polar to Cartesian or vice-versa) is applicable for each respective rotation the matrices are derived. The full rotation matrix R for the entire three dimensional orientation transform is then applied after the translation of the origins of the two coordinate systems has been done.

5.2 Angle Variables

The angle variables describe the orientations of the links and rotations performed by the camera in tracking the subject as it moves across its field of view. The position of the quadruped in relation to the camera was discussed in Section 3.2 and the full configuration was defined in Section 3.1.

5.2.1 Configurations

Each configuration will, firstly, be defined by its relative position and orientation as discussed above. The actual configuration within its local system is fully described by θ, ϕ and the link lengths. The first vector function evaluates these variables by comparing the actual image of the subject and the estimated image. It is apparent that the second gait (configurations 10 to 19) provides image data where the subject is at less acute angles than the first and third gait sequences. See Appendix E for the relations between configuration numbers and gait number.

5.2.2 Camera

The expected behaviour of virtual camera model angle rotation variables were considered in order to provide estimates for the optimisation solver. The camera rotation angle ξ about the X-axis was expected to be sinusoidal in keeping with the oscillating motion of the subject and with zero mean, while the remaining Y and Z rotations, ψ and ζ were expected to have largely non-zero values.

5.3 Link Lengths

The end effector link of the front leg, r_4, is the base length for all other links. This value was originally obtained from the maximum physiological data available on the subject and set at $0.28m$. With numerous code runs, trial and error tests and lower solver estimates produced the final proportionality and base link length values. The links are then calculated using the proportionality relations as follows:

$r_1 = (1.545 * rlinkL) = 0.313[m]$

$r_2 = (1.273 * rlinkL) = 0.258[m]$

$r_3 = (1.364 * rlinkL) = 0.277[m]$

$r_4 = rlinkL = 0.203[m]$

$r_5 = (2.045 * rlinkL) = 0.415[m]$

$r_6 = (1.364 * rlinkL) = 0.277[m]$

$r_7 = (1.545 * rlinkL) = 0.313[m]$

$r_8 = (1.133 * rlinkL) = 0.230[m]$

These relations ensure that there is a certain amount of flexibility for the dimension estimates by the solver, but still ensuring an underlying implicit model between the links. The values given here for each link length are, of course, estimates and the solver should give values close to these.

5.4 Angle Transform Equations

The transform equations that are core to the operation of the optimisation are derived from simple geometry applied in three dimensions. As an example, refer to Figure 24 and consider the relative location (u_1,v_1,w_1) of joint #1 to J#0 with assumed location coordinates at $(0, 0, 0)$ relative to the observer or camera position. Here the UVW-axis system is shown with all θ angles defined according to the reference line provided by its prior link and all ϕ (not shown) link angles defined similarly in the UW plane (see Figure 14 for the definition of ϕ):

$$u_1 = r_1 \sin\theta_1 \cos\phi_1 \tag{5.1}$$

$$v_1 = r_1 \sin\theta_1 \tag{5.2}$$

$$w_1 = r_1 \cos\theta_1 \sin\phi_1. \tag{5.3}$$

Similarly for joint #2 located at (u_2,v_2,w_2) in the UVW system:

$$u_2 = u_1 + r_2 \sin(\theta_1 + \theta_2) \cos(\phi_1 + \phi_2). \tag{5.4}$$

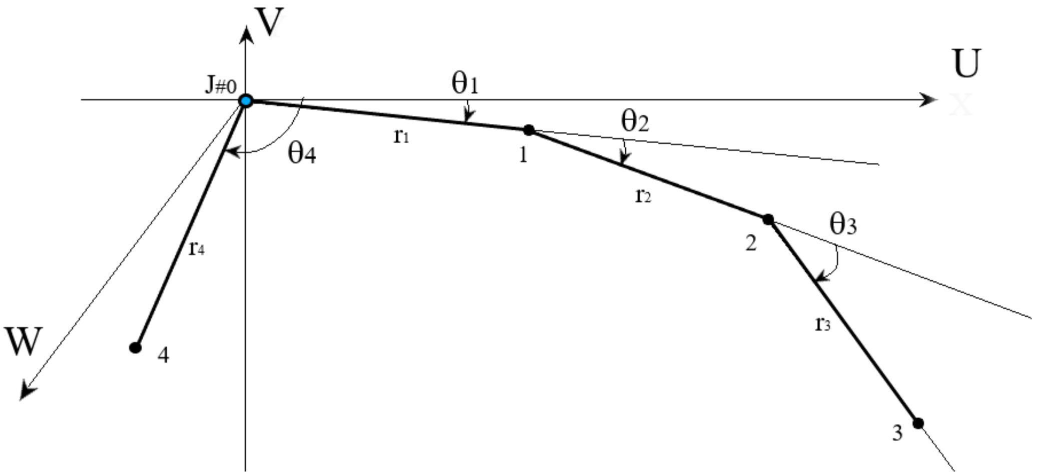


Figure 24: A sample mechanism showing link lengths, joints and angle definitions within the *UVW* local system. The variables here are used to define the 3D joint positions according to the transform equations.

$$v_2 = v_1 + r_2 \sin(\theta_1 + \theta_2) \tag{5.5}$$

$$w_2 = w_1 + r_2 \cos(\theta_1 + \theta_1) \sin(\phi_1 + \phi_2). \tag{5.6}$$

This procedure is extended to all the links of the quadruped and the full transformation equations are thus derived. These are then used whenever the multi-dimensional points of the robot are required such as inputs to the camera model or inverse kinematics model. If this is applied to the quadruped and considering that the camera rotation variables are free, all configurations must be tied to the *UVW* system. Applied here this means that θ_1, ϕ_1 and γ_i (twist angle for link i) are all set to zero and all other angle measurements are relative to r_1.

5.5 Optimisation Strategies

In determining which model strategy to use the idea that each configuration solution is a separate problem and not directly associated with a previous configuration solution ensures a measure of robustness since it does not rely on heuristics with their inherent errors. If one were to use say DH parameterisation where the computed local axis frame system for a link will affect the outcome of frames further down the kinematic chain, then any errors will be propagated down to subsequent links.

The only dependencies are used in peripheral vector functions where limitations on angle and polar vector rates are imposed. Each configuration will, firstly, be defined by its relative position and orientation as discussed above. The actual configuration within its local system is fully described by all the relative link angles in both planes, θ and ϕ, and the link lengths. The first vector function evaluates these variables by comparing the actual image of the subject and the estimated image. It is apparent that the second gait (configurations 10 to 19) provides image data where the subject is at less acute angles than the first and third gait sequences.

5.6 Coding the Optimisation

The raw data requires processing and formatting into matrix format so that the non-linear LSQ optimisation can be implemented. The code is written in Matlab and Python with several supplemental programs and inclusive functions to facilitate the main code.

Angle data that is processed ensures that all angles are in the range $[-\pi, \pi]$ to avoid discontinuities and the use of the four-quadrant-inverse tangent operator ensures avoidance of discontinuities for $\theta \in [-\pi, \pi]$. Once the image data has been obtained it is checked for missing data, possibly due to occlusions, that was generated by DLC. The image data matrices are then supplemented by a program written so that these omitted points can be manually labeled. The link configurations are then generated in the virtual image plane (thus inverted) so that the approximate configuration angle data can be extracted visually or measured on-screen. In the case of the animal subject, the first and last sets of sequences are generally observed at acute angles and the mid-set is largely orthogonal and therefore more accurate. Links that deviate from the main motion plane, such as the arm of a biped/humanoid or a leg of a quadruped at high acceleration, are problematic; their initial positions will not be accurately estimated due to the high depth component that is not extractable through any reverse projection from a monocular camera image. However, since the quadruped has many links only some, if any, will show noticeable deviations while the rest will be in-plane and the optimisation will at least have a higher proportion of correct initial estimates to guide it towards the correct solution vector based on position constraints only.

The accuracy of the optimisation is also only as good as the accuracy of the input joint positions in each actual labeled 'truth' image; any errors here will affect the LSQ optimisation process and filter through to each coupled variable and function.

The question of setting link lengths fixed or variables is also an important factor. If the links are fixed then the optimisation is severely constrained and this approach is, paradoxically, not realistic even if they are in proportion anatomically. Since the surface reference labels are attached to the skin they do not always reflect the true positions of the primary locomotion junctions (joints). A compromise arrangement was done whereby the shortest link and least likely prone to measurement error was used as a reference link to which all others were assigned in proportion. The lower link of the front leg was chosen as the base reference link since it is short, rigid and therefore mostly linear and does not show visco-elastic properties like some of the longer links like the spinal links.

There is quite a range of link length variations that is simply discernible on a visual basis for different gaits and indeed within the same video sequence, specifically unsteady (accelerating) motion. The anatomically correct link lengths are only used for the initial estimates at the start of the algorithm.

The vector of variables that is to be solved for includes all eight θ (planar) and ϕ (depth) angles, the 3D coordinates of the reference joint, the base link length and the six variables relating the camera extrinsic parameters (rotation and translation) which are all fixed for a single configuration, i.e. for each image. The camera intrinsic constants (focal lengths, offsets, skew factor) are determined separately using a different Matlab calibration program.

The twenty-one variable vector $x = [\theta_6, \theta_7, \theta_8, \theta_1, \theta_2, \theta_3, \theta_4, \phi_6, \phi_7, \phi_8, \phi_1, \phi_2, \phi_3, \phi_4$, x_0, y_0, z_0, ξ, ψ, ζ, rLinkL] is to be solved for each configuration. The angles in both the UV (for θ) and UW (for ϕ) planes, the 3D reference joint, the rotation and translation camera extrinsics and the base link length need to be resolved on each sample interval.

Notice that θ_9, ϕ_9, t_x, t_y and t_z are omitted from the optimisation and are set to zero. Although the camera model is designed to focus on a single specific point it is practically feasible to use the same camera extrinsics for each image even if not all points are in the same plane. The optimisation effectively resolves depth by way of a combination of the orthogonal camera translation and the transverse planar angles. The accuracy of the initial estimates is critical to the performance of the optimisation. The theoretical basis rests on the first guess solution vector to be in the vicinity of the actual solution and this is even more so when dealing with highly non-linear and hyper-dimensional vector spaces. Angle estimates were selected from the [-π,π] range and were crudely based on those observed from the image data, which gave some indication of the nominal angle ranges in the sagittal plane. First trial guesses obtained in this way, together with the expected nominal estimates of the other 14 variables, were presented to the solver. The implicit penalising parameters built into the objective function ensured that motions were indirectly constrained on each interval by penalising the solution search if it extended beyond the assigned angle ranges. An iterative process of refinement then took place using the Matlab output data and indicators, reprojected images (quantified by the mean error values for the joint image estimates), the gradients matrix values and mean values of the Jacobian arrays as performance indicators. (The process is a bit like tweaking the diagonals in the covariance matrices in state estimation for near optimality).

If the first values optimised are close to the actual solution then subsequent configurations, being related to the previous one will, statistically, be better approximated. Reasonable limits are also set for deviations from the main motion plane thus setting practical limits to the depth component of a link.

Since angular velocity data is only available on the second iteration, the code makes allowances for all subsequent optimisations with the estimated angles based on those of the second configuration's apparent layout. This pseudo-actual angle data serves as a guide to the qualitative motion and serves as a kind of 'simulation' data which can be compared with the approximated LSQ data. Also, the additional vector function relating the limits of angular rates to those that are estimated is only applicable from the second configuration. The dynamics are now also included as a constraint in the algorithm. The number of times that the limits are violated are recorded together with the actual LSQ performance variables such as the values of the linearised matrices (Jacobians), residual (error) norms, the quality of the solution and so on. Transform equations convert the estimated reference joint coordinates and the angle data into the relative spatial link configurations in 3D.

The graphical outputs include all of the estimated angle data together with the corresponding angular velocities and accelerations, the actual and estimated pixel images of the configurations and the corresponding configurations in the UV plane.

5.7 Inverse Kinematics Modeling

Due to inherent redundancies of a complex link system such as the SFL system the number of solutions may be infinite which means that an infinite number of joint velocity vectors exist for an end effector velocity. An analytical (algebraic) solution is not an option since the number of constraints is not equal to the number of DOF, although an iterative approach and/or the creation of additional artificial constraints is an option. The latter procedure was followed in the previous LSQ optimisations where geometric limits and angular rates were confined to practical limits of the quadruped. Batch iterative solutions (as opposed to instantaneous solutions) for the inverse kinematics using the resolved motion rate control method, $\dot{\boldsymbol{\theta}} = J^{-1}\dot{\mathbf{X}}_{\mathbf{EE}}$, were applied using the widely used GP, WLN and EJ algorithms and ensuring that the sample rates were very high $(0.0033s, 300Hz)$, since the Cartesian velocities were high (of the order of $20m.s^{-1}$), and thus ensuring the validity of this approach.

The basic approach used to determine the parameters of the IK models is to use the previously obtained angular velocities for the end effector as the truth or target data vector in a non-linear least squares process where the model data vector is generated by the respective inverse kinematics model and the parameters are solved for each of the configurations. Assuming that these returned values all lie within a relatively narrow band, the averages are taken and a single parameter value is then used for that specific IK model.

5.8 Overview

The practical aspects such as marker annotations on the cheetah, training, evaluation and refinement steps of the deep learning network are outlined. The proportional link lengths that were used and the angle transform equations used in the coding are given. Practical coding issues that relate to the least squares optimisation and inverse kinematics conclude this section.

Chapter 6

Results, Findings and Model Evaluation

All of the data outputs are presented and an analysis thereof is made. The validity of the data is investigated using the performance indicators and the quadruped's configurations are shown in various planes. The final section presents and discusses all the results for the inverse kinematics models, including the parameters and the synthesis of trajectories in the joint and task spaces.

6.1 Image Pose Extraction

The network was retrained and evaluated, with labels being refined. Since the task here was the analysis of the subject undergoing smooth cyclical movements there was no question of any erratic and thus sparse behaviour, which would have entailed the k-means clustering option during the frame extraction process by DeepLabCut. A uniform extraction algorithm was thus used. The trained network outputs the estimated pixel values of each estimated marker on the subject for the entire video sequence being considered. The pixel values as estimated by the network (see Figures 25 and 26(i)) is used as the input data matrix to the non-linear least squares optimiser as the truth inputs. Associated with the image estimates is the confidence in the labels' correctness as assessed by DeepLabCut (see Figure 26(ii). Each image

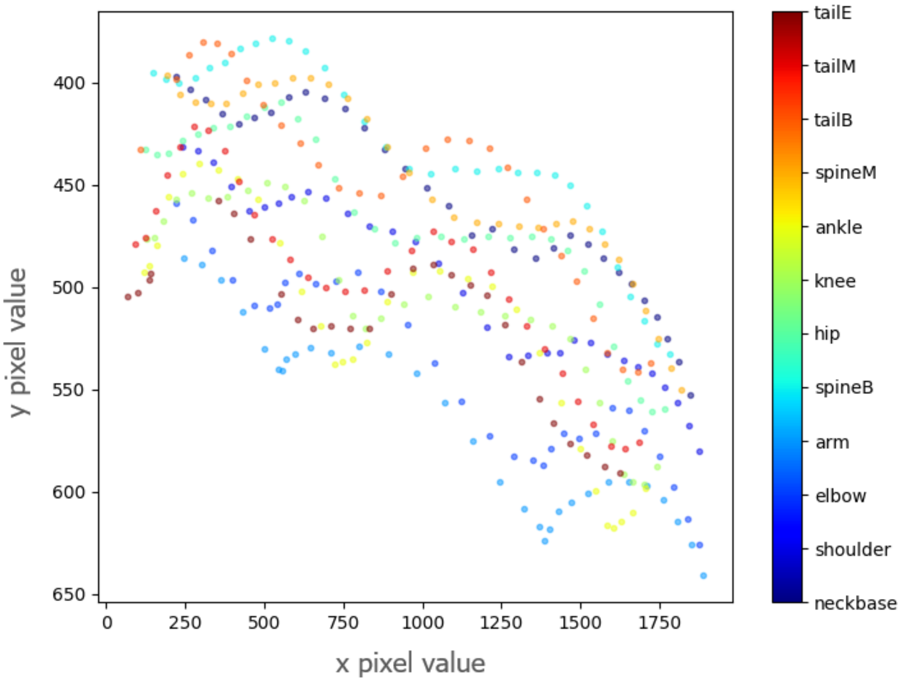


Figure 25: The image data points of the unseen video sequence showing the CNN estimated pixel values of the labels. All labels for the legs, spine and tail are shown. The subject moves from left to right and a composite plot of these points on the original images correlates well with the fixed markers.

configuration (particularly the second gait sequence) was studied carefully so that rough estimates of the primary link angles (θ) could be deduced. This process ensured that angle estimates for the optimiser were sufficiently close to the expected solutions. It should be pointed out that the LVM is particularly sensitive to these initial estimates being too far off from the actual solution values. The estimated labels were not perfect initially, but after refinements and re-training this was improved. Parameter settings such as those specifying maximum pixel changes for a label between images, the initial weights during training and others in the configuration and pose program files were tweaked. The subject was video imaged on a moving platform with the resulting video sequences being used for markerless tracking by a CNN and the resulting data being used for a LSQ approximation.

The animal subject had numerous joint markers affixed to it, so that reasonably accurate labeling for the CNN network could be done. The subject would make numerous runs at various gaits and path lines, including standard uniform trots to highly variable accelerations, decelerations and orientations, although they were

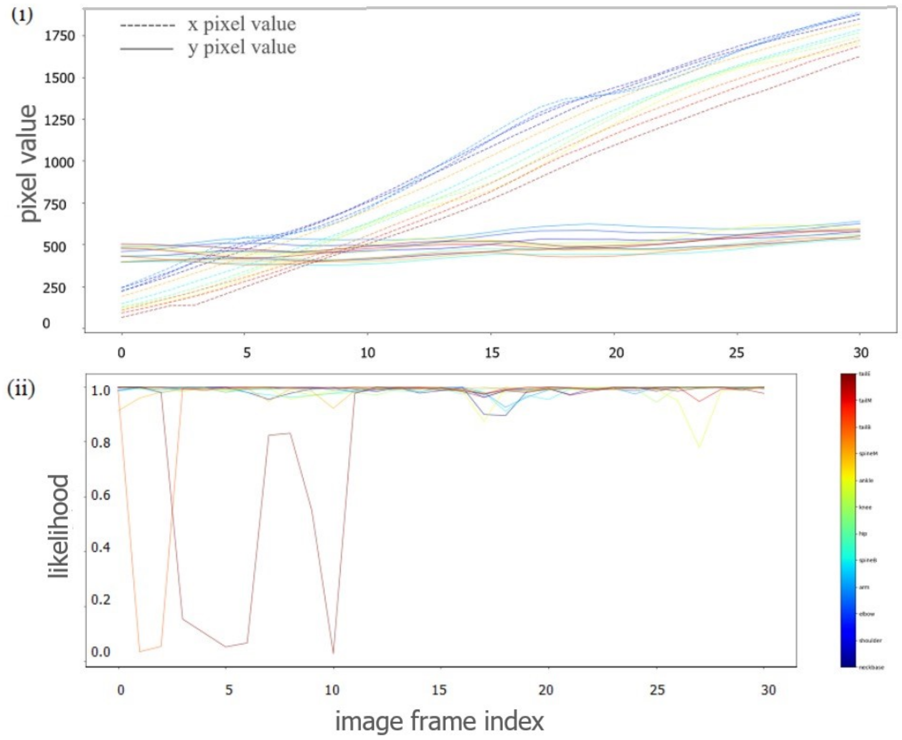


Figure 26: The predicted joint trajectory points in the image plane using machine learning (DeepLabCut) together with the associated probabilities of correct occurrence are shown here. The pixel values as a function of the configuration number (or equivalently time) is shown in (i) and the likelihood or confidence in the labels' correctness as given by DeepLabCut is shown in (ii).

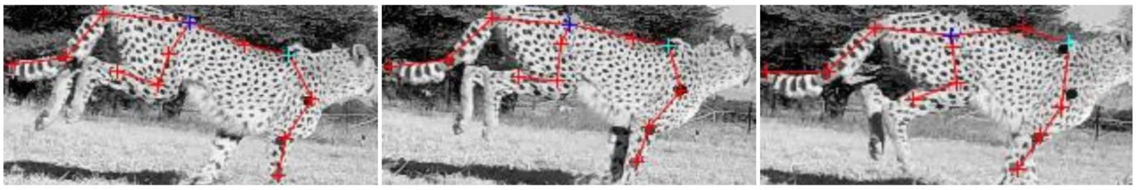


Figure 27: Original sample images and the predicted image configurations by the network. During the course of the network training the estimates on unseen images were not perfect but this was improved upon by re-training, refinements and parameter adjustments in the configuration files.

largely roughly within the general local imaging plane symmetrical about the origin. The images were used to train a DLC network and an unseen video sequence was then labeled, and the resulting video image sequences were used as the actual truth inputs to the LSQ optimisation from which its global 3D coordinates were estimated from the predicted angle data. Sample schematics of the estimated markers and the resulting quadruped mechanism are reprojected onto the respective unseen original video images as shown in Figure 27. See Appendix E for the complete reprojected image sequences.

6.2 Least Squares Optimisation

For each configuration optimisation a Matlab output of the final solution variables is shown for the current vector point, with the values being in units of the respective variables. All angle variables are in radians and the UVW coordinates of the reference joint and the base link length are in metres (see Figures 13, 14 and 15). The low values (less than 0.5 radians or approximately 28°) of all the ϕ variables, the coordinates of the reference joint (particularly the U-axis components), the low and qualitative nature of the camera angle values and the base link lengths (ranging from $0.15m$ to $0.21m$) indicate that the solver is at least delivering realistic outcomes; the accuracies of these require a study of the solver's outputs and flags while angular rate and instantaneous tangential velocity calculations of the data will show its validity. Typical illustrative output data by Matlab shows the main

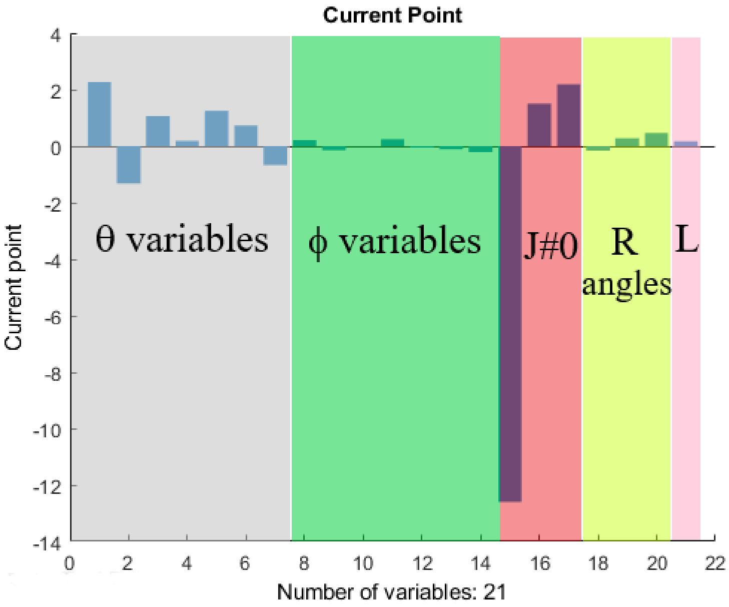


Figure 28: Typical illustrative output data by Matlab for a single solution point. The 21 variables comprising the solution vector includes the generalised position coordinates for all the links, the position coordinates of the reference joint, camera rotation angles and the base link length.

(θ) angle variables within a $[-\pi, \pi]$ band and the secondary (ϕ) angles are mostly small ($[-0.5, 0.5]$). The first variable, which defines θ_6, has a value of just over 2 radians (114°) for this example, while the x component (variable number 15) of the reference joint trajectory is $-12m$. The base link length (last variable) is at about $0.2m$.

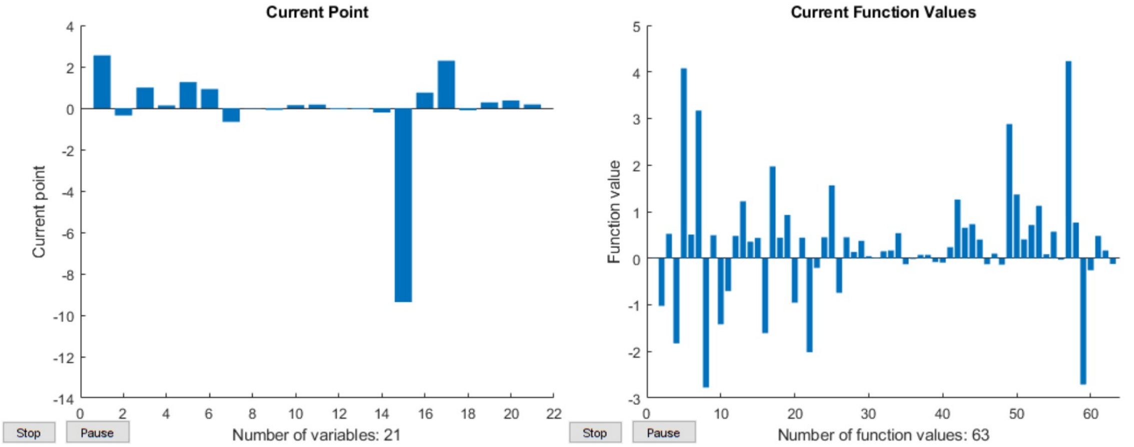


Figure 29: Typical Matlab output of selected performance indicators. The solution vector of 21 variables and the current function values are shown for a specific iteration. With reference to Figure 28, note the small values for the ϕ angle variables.

6.2.1 Output Indicators

The code output graphics and indicators provide information of the optimisation process which assist in assessing both the accuracy and validity of the output data of the solver. Typical output graphics of the solver are displayed in Figures 29 and 30.

Number of Iterations: Considering the scale of this problem (in terms of the number of variables) the number of iterations required to complete the optimisations for each of the thirty configurations was relatively low. Fitting a simple two variable exponential function, for instance, only requires 19 iterations and a function count of 72 using the Levenberg-Marquardt algorithm, while this optimisation needed between 17 and 160 iterations to solve for all of its 21 variables. See Figure31.

First-Order Optimality: An optimal solution vector will mean that the first-order

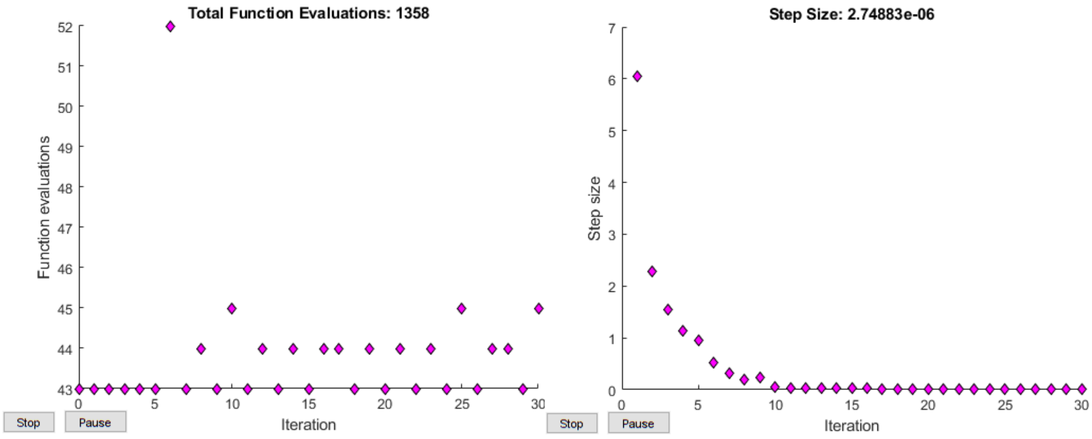


Figure 30: Typical Matlab output graphics for the total number of function evaluations and the final step size that the solver takes at solution. For this solution point the solver required 1358 function computations in total and only 30 iterations (coincidentally equal to the number of configurations here). Notice the steep convergence towards the desired zero line for the final step size taken by the optimiser for the last iteration. The data shown here is for the solving of the second configuration and the others were of a similar order of magnitude.

optimality measure will be zero but this is seldom the case for high dimensional problems. In most cases the optimisation is terminated once the solution is within

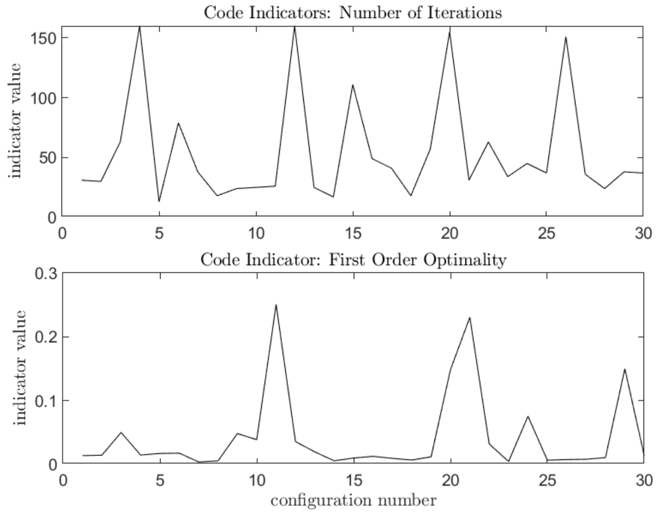


Figure 31: The number of iterations and first-order optimalities for all the configurations. Low first-order optimality values close to zero are a good indication that the solutions are near to being optimal.

a certain tolerance value. The solver achieved suitably low values for most of the configurations. They were mostly around 0.001 with occasional spikes to a maximum of 0.25. Low optimality values are also indicative of low gradient values of the objective function. See Figure31.

Damped Least Squares Parameter λ: The initial value of λ may result in slow progress of the algorithm initially, so it may need to be changed as was found to be the case here. An initial value of $\lambda = 1000$ (up from its default value of 0.01) was found to improve convergence.

Residuals: The value of a single vector function is simply the residual or error between its estimate and the input data, or for the general case the residual is the value of the objective function returned as a vector. For the multi-variate objective function here, a mean is obtained as a measure of the errors for each iteration and they ranged from roughly 0.6 to 2.2. See Figure32.

Squared Norm of Residual: The squared norm or 2-norm of the residual at x is just the sum of the residuals squared i.e. $\sum f(x)^2$.

Norm of Step: The dimensional distance taken by the solver to its final solution is the norm of the step (or step size) equivalent to the magnitude of the last search direction. These are consistently of the order of 10^{-3}. See Figure32.

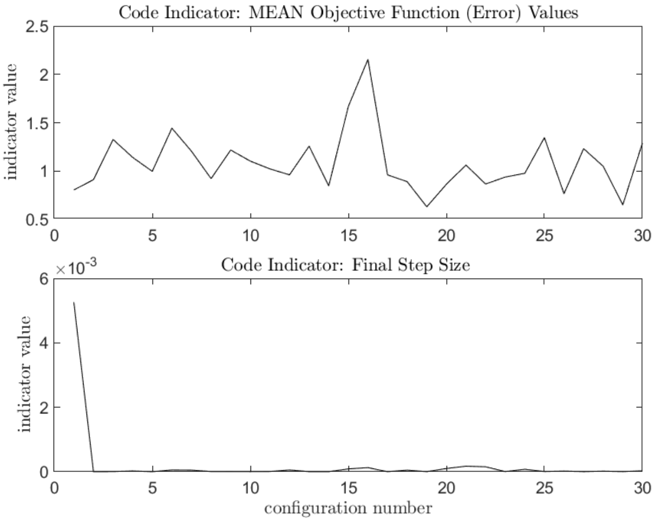


Figure 32: The mean objective function residuals and final step sizes. The mean residuals or errors are the average displacements in hyperspace between the estimated vector and solution points. The final step sizes are sufficiently close to zero and contribute to the validation statements of the solver. Notice the improvement after the first iteration. From the second iteration angular rates can be computed and the contributions of the other vector functions are now included in the solution.

Function Evaluations: The total number of function evaluations (function count)

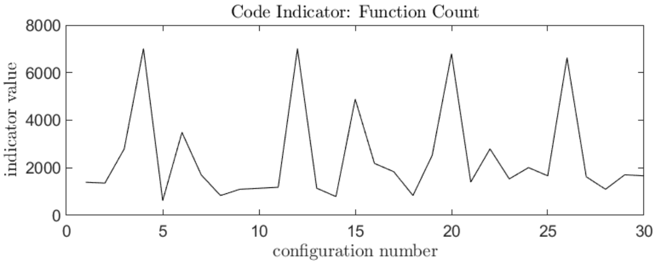


Figure 33: The high number of function evaluations due, in part, to the use of centrally based gradient computations by finite differences.

is high here since the gradient estimates, using finite differences, are centered and it will therefore take twice as many function evaluations. It is qualitatively identical to the number of iterations data plot and it is approximately 46 times higher than the corresponding iteration value. See Figure33.

Exitflags: For all of the instances the magnitude of the search direction was smaller than the specified tolerance or the change in the residual was less than the tolerances and a local minimum was possible. If the default value for the maximum number of iterations is used then 93% (28 out of 30) of the solutions achieve instances where local minima are possible. By increasing the default value for the maximum number of function evaluations a 100% success rate was achieved.

6.2.2 Matrix Data in Graphic Form

Jacobian Matrix: The mean values of the Jacobian of the objective function were taken over the entire sequence with a sparsity of 25% and being non-singular implied that it was continually differentiable at all the vector points. Since the Jacobian is the differential of f at x or the linear approximation (point-wise speaking), it ideally should have all zero elements (sparsity = 100%). The higher the sparsity, the less computationally expensive the optimisation. Critically, the elements of the Jacobian coinciding with the angle variables θ and ϕ, are around 50 with only the camera rotation variables reaching a maximum of about 500. With $m = 21 =$ number of variables and $n = 10 =$ number of vector functions the Matlab command: J = sparse(ones(n,m)), checks for an "Out of memory" error and whether there is sufficient virtual memory. The scale of the optimisation problem is within the

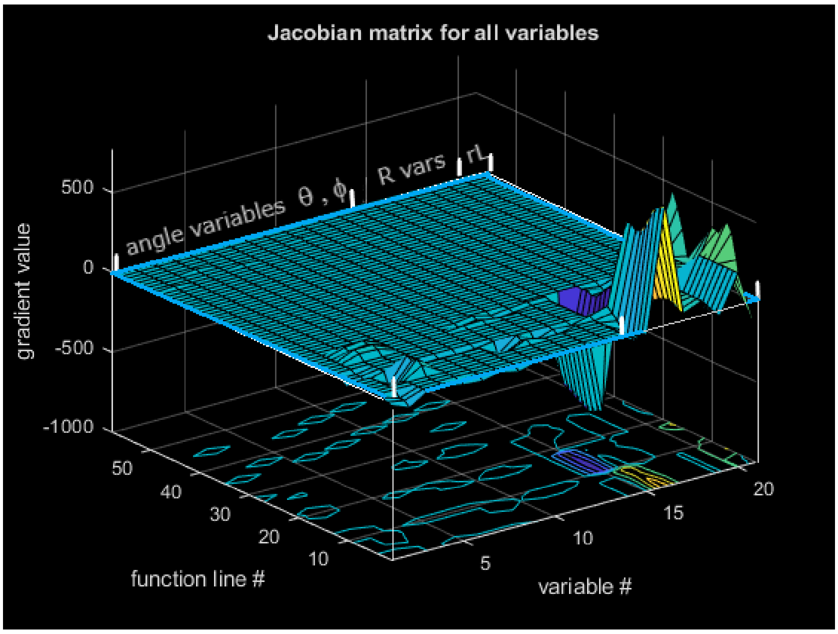


Figure 34: The mean values of all the Jacobian arrays as evaluated by the solver.

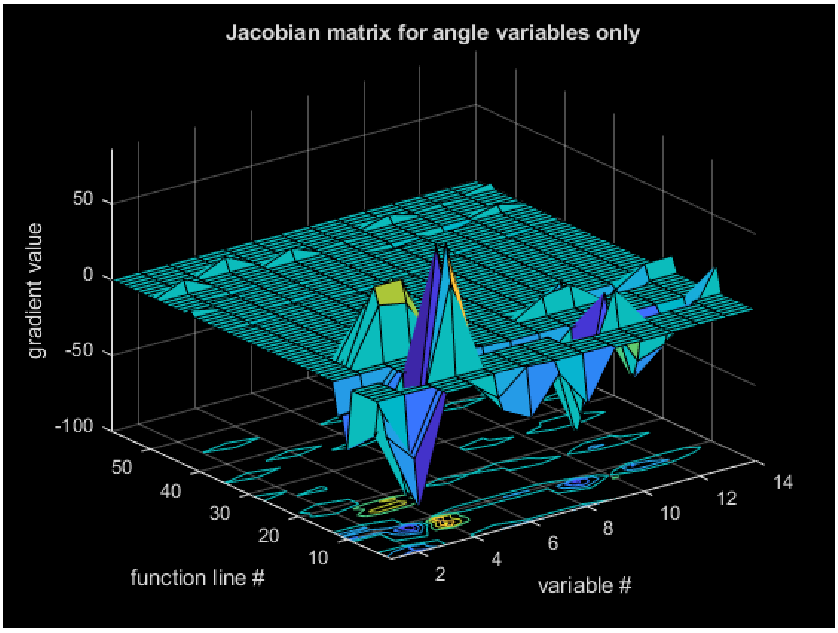


Figure 35: The average of the Jacobian matrix element values for only the angle variables of the quadruped mechanism. Note the smaller values for all the ϕ variables (variable numbers 8 to 14).

capabilities of the machine being used. All the results are shown in Figures 34 and 35.

Gradients Matrix: The low mean gradient values were mostly small reflecting the low first-order optimality values as expected. The graphic in Figure 36 reflects this and also highlights the errors the solver returned in estimating the three components of the reference joint.

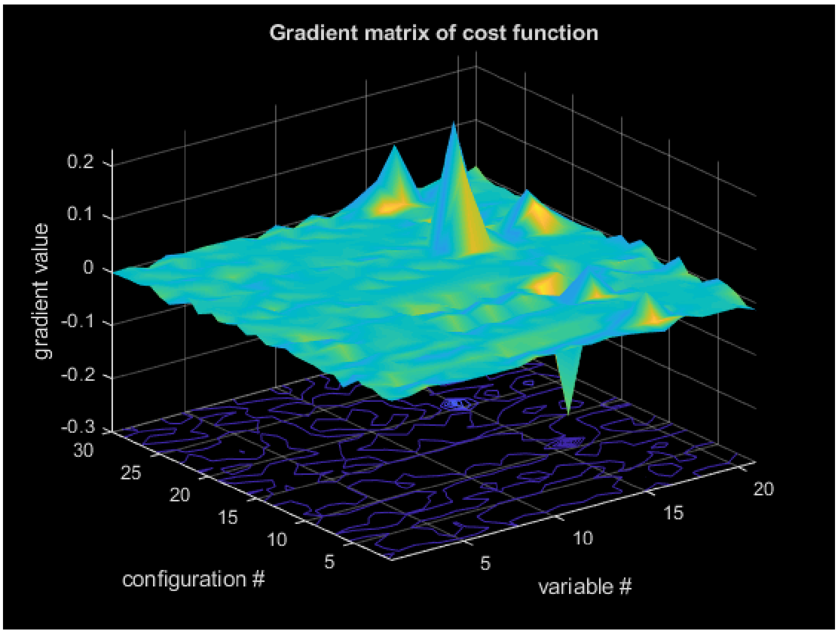


Figure 36: The gradients matrix values are linked to the optimality of the optimisation estimates. Low values indicate that the search has effectively ended with all search directions static at the solution point.

Hessian Matrix: The gradient function is essentially the first derivative of a scalar function $f(x)$ of several variables and the Jacobian of the gradient function will effectively give the second derivatives, known as the Hessian matrix, of the original scalar function. If one was interested in using this data in an application, such as determining the occurrence of inflexion points, then the Hessian $H(x)$ is computed by

$$H(x) = J\left[\nabla f(x)\right]^T . \tag{6.1}$$

The Hessian matrix is not applicable in the current context though.

Image Error Array: A plot of the image error between the actual and estimated values for the entire video time shows that joint #9 displays a definite exception to the norm (see Figure 37). The difference between the actual and estimated pixel images for all of the configurations displays definite error patterns. The x and y pixel sets for each joint and configuration are grouped to deliver the corresponding errors visually. The differences between the truth and reprojected image data expressed as

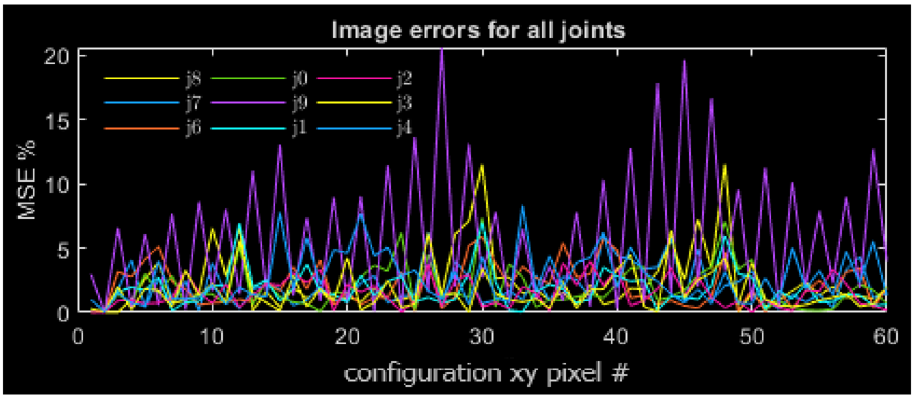


Figure 37: The mean error values for the joint image estimates are shown for all configurations. Notice the peak for joint #9 (mid-spine joint). The data shown here is repeated in Figure 38 as a two dimensional image error data matrix with the joint index numbers corresponding to the joint numbers of the mechanism.

percentages are shown as image errors in Figure 37. If one considers index number 5 (for joint #9), the consistent error is apparent for all configurations (see Figure 38). The correspondences of these matrix graphics and the configuration is made by way of the lower schematics. The maximum mean error was still only about 15 pixels while the majority hovered around the 5 pixel mark.

6.2.3 Configurations

All of the reprojected image estimates were very close to the actual images with errors that ranged as discussed in Section 6.2.2. For illustrative purposes consider the composite images of the solver and truth image for a single configuration as shown in Figure 39. This fact contributes to the notion that the mechanics of the solver have been correctly theorised and that the results are valid and within reasonable accuracies. It is apparent that the second gait (configurations 10 to 19) provides image data where the subject is at less acute angles than the first and third gait sequences. See Appendix E for the relations between configuration numbers and gait number.

The estimates for the base link length (in metres) varied between about 0.125 and 0.23 with a mean of 0.175. However, from the video it is presumed that the second gait sequence would provide the most accurate values, due to the less oblique ob-

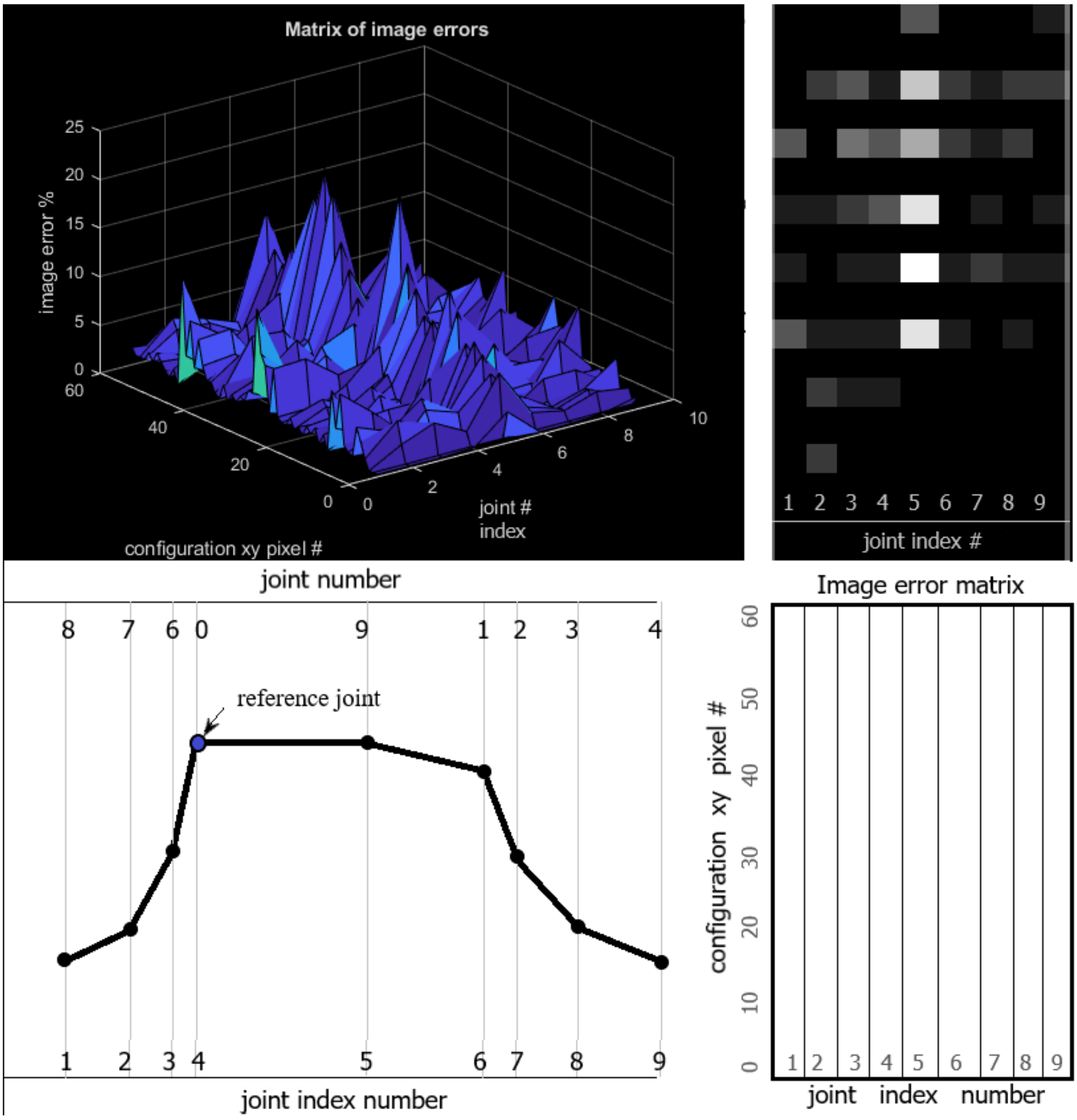


Figure 38: The mean error image array data above show the differences between the actual and estimated pixel values for all the robot joints and averaged over the entire sequence. The schematics define the interpretation of the matrix data according to the configuration's joints. Note the peak error for joint #9 (the mid-spine joint) at joint index #5. The schematic at lower right is a reference for the graphic directly above it. The errors are specified for a xy pixel pair corresponding to a joint index number which in turn represents a specific joint number of the mechanism.

servational angles and largely orthogonal plane arrangement between the primary motion plane of the subject and the line of observation. The mean link length estimated for the second gait was $0.203m$ and this value was used to compute all of the actual full physical configurations in Cartesian coordinates.

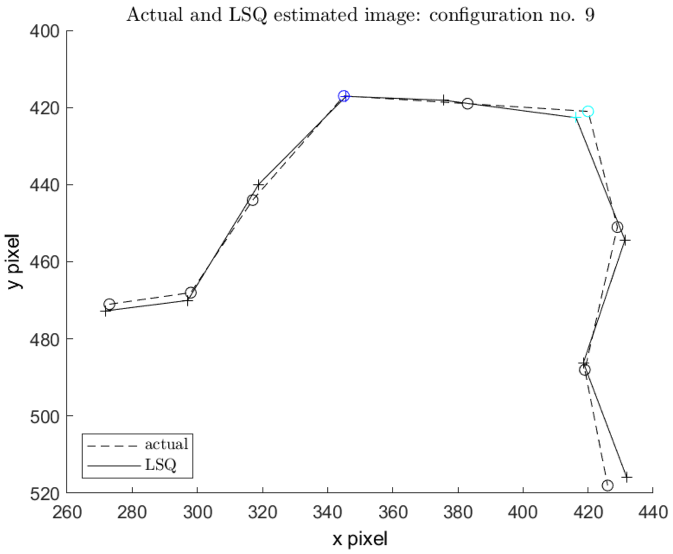


Figure 39: Least squares solver and truth images of a single sample configuration. All the other configurations also showed very close correlation. See Appendix E.

The camera angle variables about each of the XYZ-axes viz., ξ, ψ and ζ, also showed the basic behaviour that was expected. The rotation angle variable ξ showed sinusoidal trends as it tracked the up and down motion of the reference joint about an approximately zero mean. The other two variables tracked the subject as it moved across the field of view and displayed largely non-zero values with an underlying V-shape. The reprojected images from the estimated 3D poses are an early indicator that confirms that the optimiser is giving sensible results and this is illustrated in Figure 40 where every sixth estimate is superimposed on the respective video images. A composite of all these reprojected images onto a pixel plane verifies the correspondences with the actual image data and this is highlighted in Figures 43 and 44. The estimated image configurations are superimposed on the photographs and by comparing their corresponding local configuration realisations in the UV (side view) and UW (top view) planes using the optimised angle and base link length data, the results certainly confirm the correspondences with the original video, qualitatively anyway. The first nine sequences illustrate this in Figure 47. The remaining variables that were optimised viz., the camera rotation angles and the base link length, are shown in Figure 41.

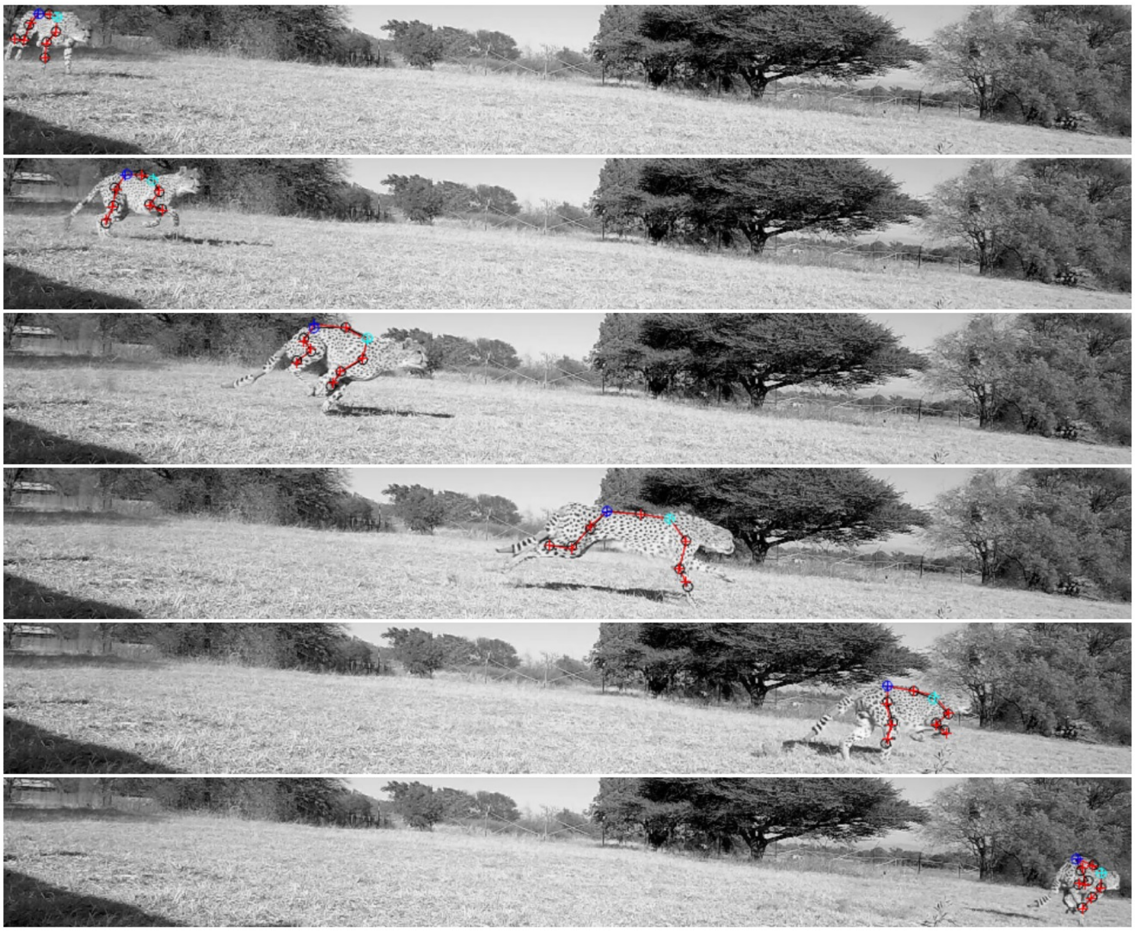


Figure 40: The photos from the original accelerating cheetah video and the reprojected least squares optimisation configuration images are shown for the sequence at every sixth instance.

6.2.4 Trajectories and Dynamics of Reference Joint

If the rotation variables are set to a constant value of zero the reference joint trajectory is smooth. Once included as variables in the optimisation the trajectory is more irregular; the solver must now estimate the camera orientations in addition to the angles and link lengths and the trace of the reference joint.

The shape of the rotation variable about the X-axis shows sinusoidal characteristics as expected [51]. This variable illustrates the tracking angle of the camera as it follows the reference joint projected onto the XY plane.

The angle x component variable of R, ξ, shows a definitive (up and down) tracking curve of the reference joint as while the other two camera variables essentially track

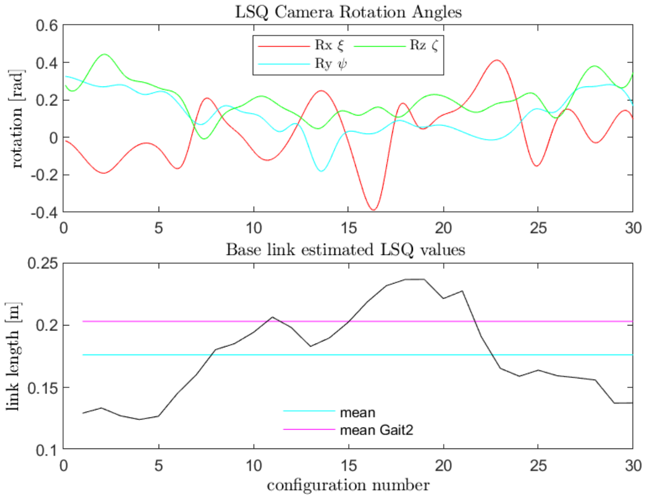


Figure 41: The optimised camera rotation angles and the base link lengths for each of the 30 configurations.

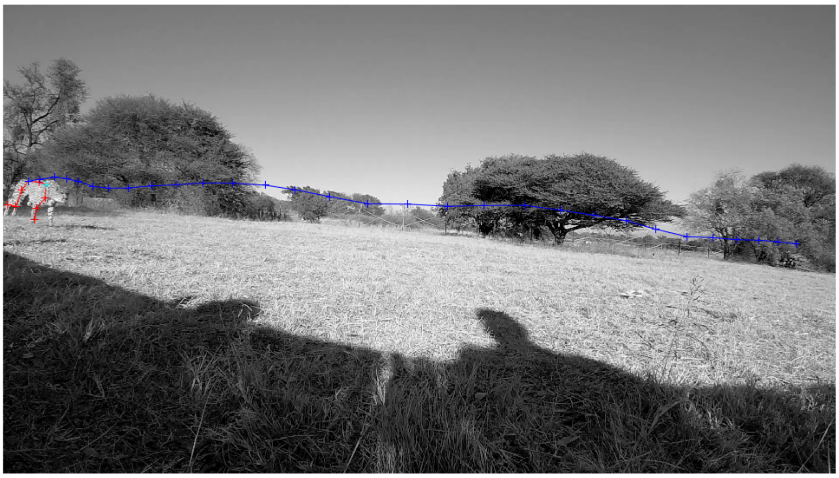


Figure 42: The progression of the estimated reference joint trajectory (blue) in the image plane. Note its approximately sinusoidal path of about $3Hz$, confirming the assumed three gait motion of the subject.

the translational motion in the transverse and frontal planes and they show lower amplitude variations.

If x, y and z are the position coordinates of the reference joint within the global camera system (refer to Figure 15) and using the reference joint position data, then the magnitude of the resultant instantaneous velocity and acceleration of the

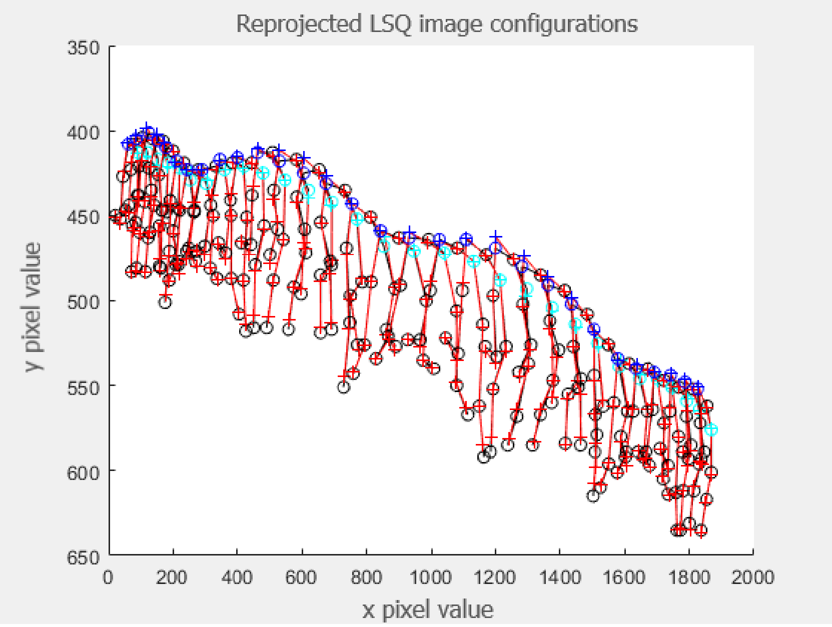


Figure 43: A composite graphic of the estimated reprojected images in pixel coordinates. The estimated 3D poses by the solver are input into the camera model to give each of the pixelated configurations.

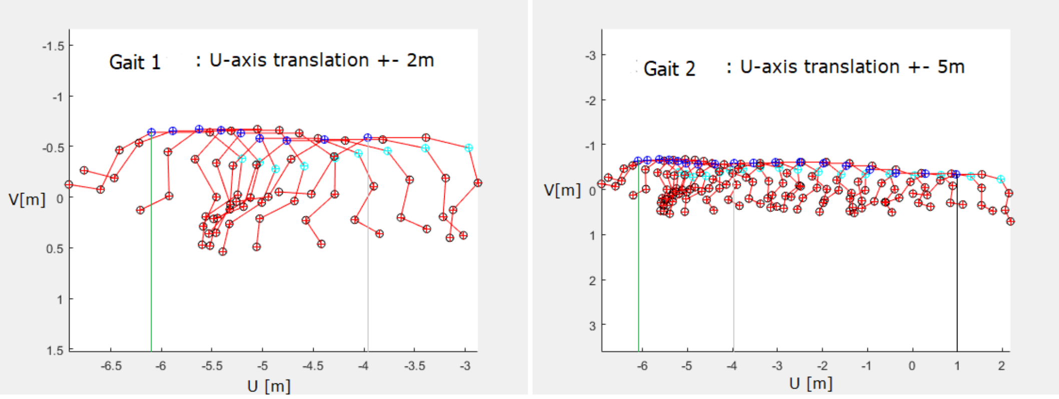


Figure 44: The composite side views of the quadruped for the first gait (left) and the first two gaits (right) in its local UVW coordinate system. The reference joints observer coordinates have been included so the origin of the UVW shows the corresponding shift.

reference joint can be determined as follows:

$$v = \sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2}. \tag{6.2}$$

$$a = \sqrt{\ddot{x}^2 + \ddot{y}^2 + \ddot{z}^2}. \tag{6.3}$$

The velocity data confirms that the subject is accelerating for at least the main part of the sequence. The mean velocity of $12.8m/s$ equates to a distance covered of about 13 metres for the 0.9667 seconds of video time. The acceleration curve shows a mean acceleration/deceleration of $6.3m.s^{-2}$. At the point where the velocity reaches a maximum the acceleration begins to show a decreasing trend as it (presumably) decelerates. The computed velocity and acceleration curves of the reference joint appear in Figure 45. The subject attains a peak speed of $21m.s^{-1}$ from zero in about 0.62 seconds. It is apparent that the subject undergoes a net acceleration which peaks at about two thirds of the way into the sequence. The acceleration value is always non-zero implying that its motion speed is largely increasing (accelerating) or decreasing (decelerating). It is clear from all the image data and the subsequent configuration plots that there is a definite correspondence between the images and resulting 3D poses. This is most noticeable when the photos and the absolute UV plots are compared individually. If the camera rotations were applied back to the fixed reference spine link (r_5) then the actual orientation as seen by the observer can be resolved.

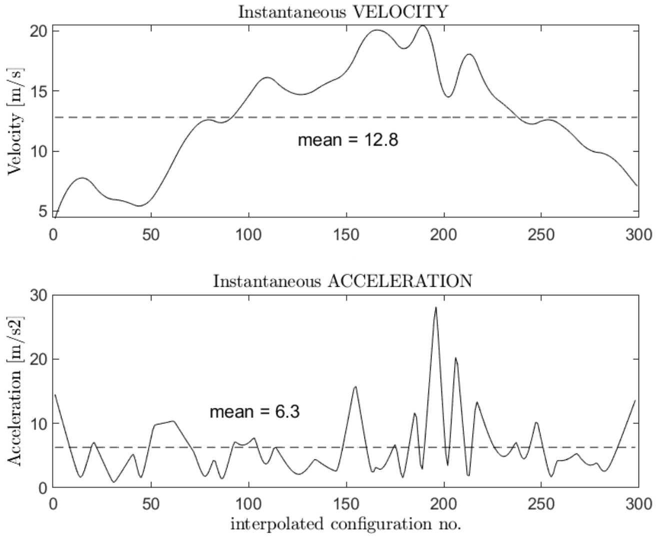


Figure 45: The magnitude of the velocity and acceleration of the reference joint. All three linear motion components were calculated from its position data and interpolated and using the higher sample rate the linear velocity and accelerations were obtained.

For clarity, the side and top views of a sample configuration is illustrated in Figure 46 so that subsequent graphics are better understood. A composite time lapsed side

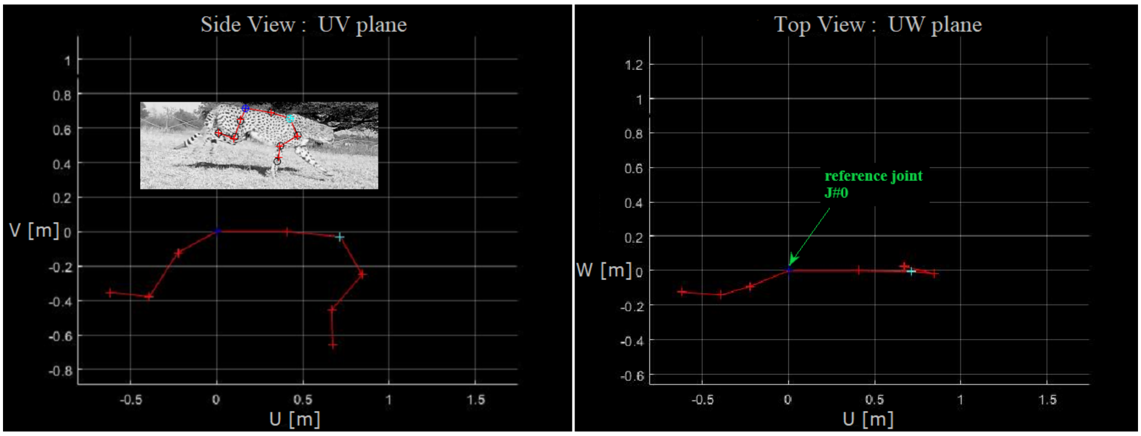


Figure 46: Illustrative graphics showing the subject with the estimated image configuration, together with the side view (UV) and top elevation (UW) view plots.

view of all the estimated configuration positions which are colour coded with those at the start (red) and transitioning to the end (green) as shown in Figure 48. There appears to be a trend of less restricted motions at high speed as the links extend at ever increasing angular velocities to maximise the step size. One can also see how the ground contact points of the front leg migrate back (to the left) and the spine flexes more to generate the increased power requirements for the acceleration. The stance phases show clustering of the extremities of both legs that reflect a locomotion regime similar to a *curvet* gait where the front and back legs of a quadruped are coupled. In the work done by [64] where optimal trajectories were created that minimised the energy costs, a *curvet* gait is created and comparisons may be drawn with the results obtained here.

Views of the spine confirm this increased flexibility that typifies the arched appearance of the cheetah's spine during ground contact (stance phase) and the far reaching extension during the swing phase. The configurations of the back leg (front view) also confirm that the power generated is increasing, particularly by looking front on. At slow speeds the back leg is mostly aligned with the sagittal plane and as the speed is increased the deviation from this plane becomes more pronounced as the linear and rotational impulses experienced by the leg increase due to the higher momenta resulting from the increased speeds. The front leg composites as seen from the front (Figure 50) show a similar pattern with the spine-front leg juncture showing definite

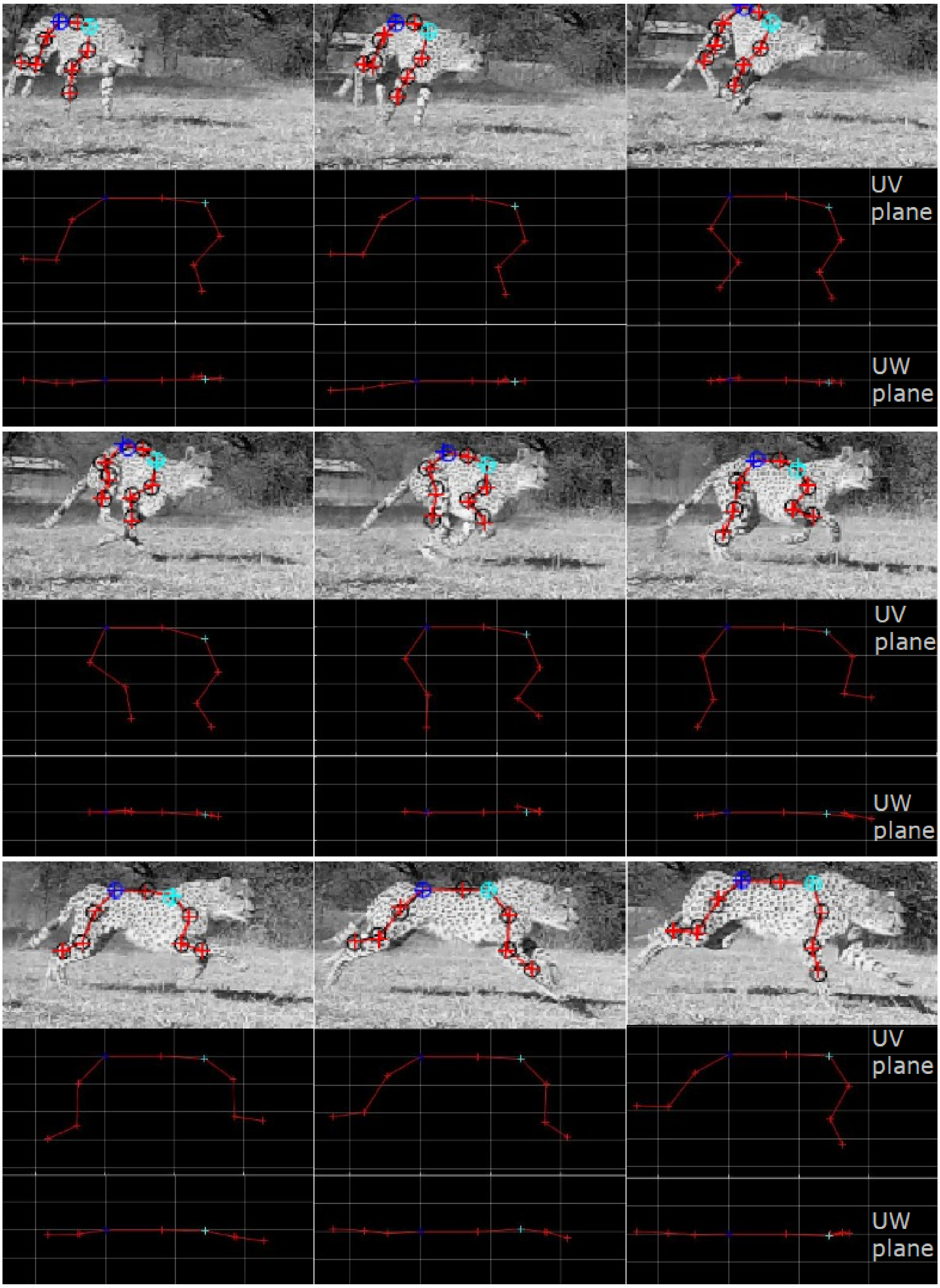


Figure 47: The first gait sequence of nine configurations is represented here. The photograph and reprojected image composites together with the corresponding configuration plane views are shown.

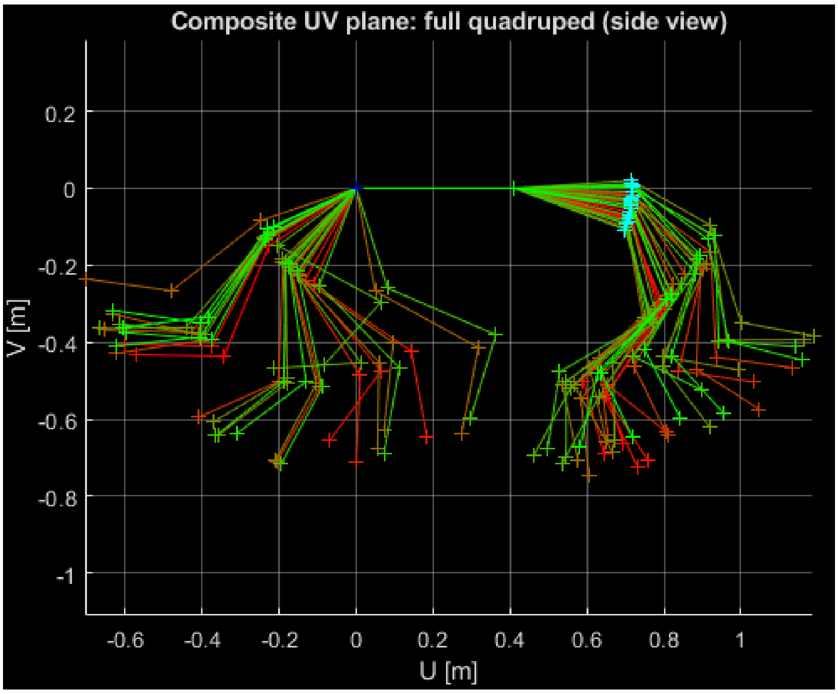


Figure 48: A composite of the full quadruped configurations in the UV plane (side view) with the reference joint located at $(0, 0, 0)$ in the local system.

vertical displacements (about $10cm$) in relation to the reference joint, due to the sinusoidal trajectory of it within the XY camera plane. There also appears to be some *crossover in-swing* of the front leg associated with each fore position of the spine (Figure 49). A general observation is that both have deviations *away* from the sagittal plane, which is a logical consequence of the physiology of the subject, and which confirms that the least squares optimisation process predicted the correct

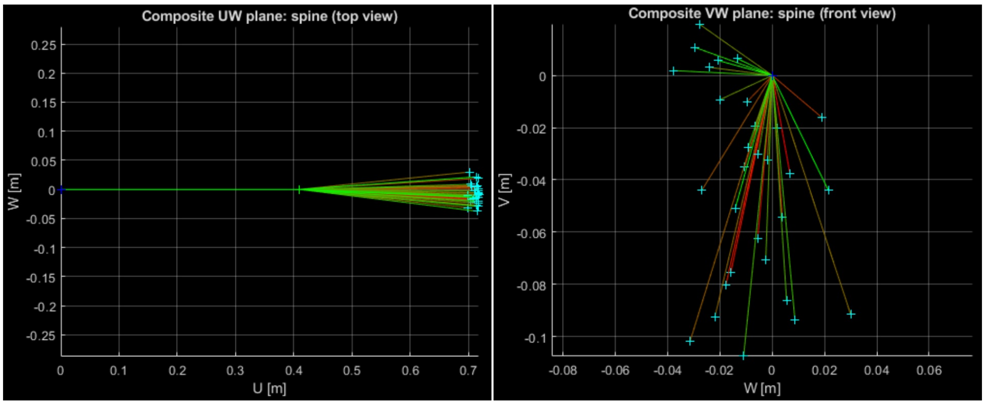


Figure 49: The top and front view spine patterns developed over the motion sequence.

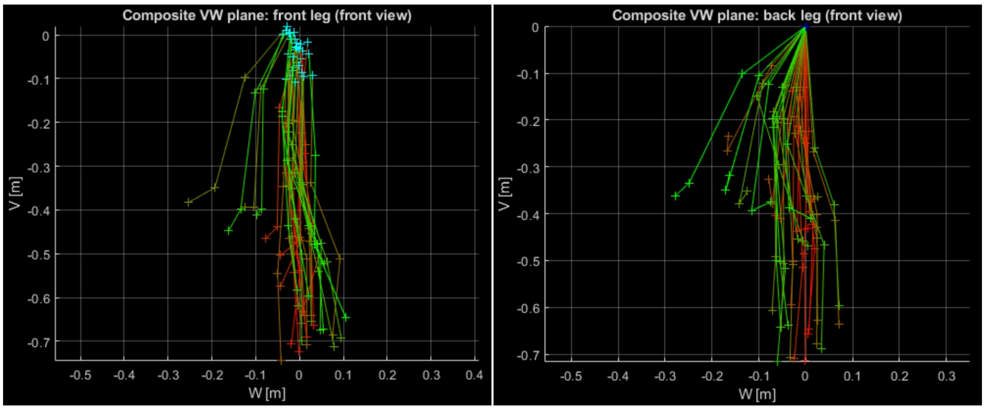


Figure 50: The front views (looking head-on to the quadruped) show the distributions of the legs. The back leg, particularly, shows a definite pattern; at high speeds (shown in green) there is a greater deviation from the sagittal plane, as is to be expected.

generic directions for these dimensional variables. In the same way that the legs of a human undergoing acceleration show an increase in their deviation from the sagittal plane, the upper leg links of the mechanism are directed out and away from the spine links that lie largely in the UV plane.

The solver produced smooth trajectories when the camera angle variables were all fixed and not allowed to rotate while more erratic paths were produced using the current model where the camera extrinsics were free variables and were allowed to change. By altering the structure of the solution vector it is possible to bias the

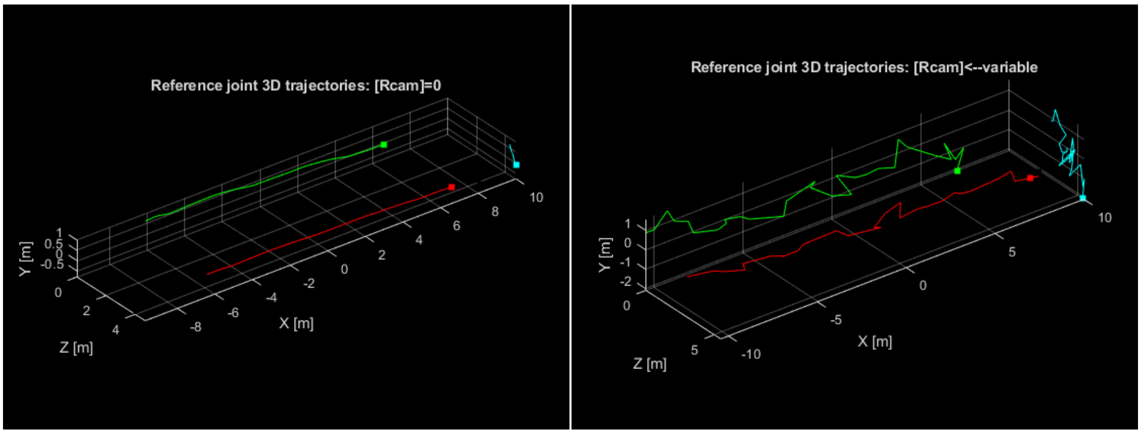


Figure 51: Estimated reference joint trajectories. At left are the three dimensional components of the reference joint trajectory, with all camera extrinsics fixed to zero, in the observers XYZ-coordinate frame. The trajectories at right are the result of the solver having the camera angles as variables.

solution's accuracy towards the configurations themselves or only the translation of the quadruped. The graphic shown in Figure 52 illustrates the passage of the subject through space as determined by the non-linear least squares optimiser using monocular image data and the estimated trajectory of the reference joint in 3D is displayed in Figure 51. The full sequence of configurations as given by the solver, shows the estimated poses of the quadruped mechanism as it traverses through the observer space. There is a smooth transition every 0.033 seconds between each configuration and the underlying sinusoidal motion of the mechanism as a whole is clearly discernible. In addition, the fact that the Y component of the reference joint gradually decreases is in agreement with the knowledge that the terrain is gently sloping down from left to right. The observer position is the origin of the camera XYZ-axis system. Figure 52 also shows that the estimated X component of the subject is kept reasonably constant at about $4m$ and this corresponds with the approximate field estimate. Finally, the first and last configurations are in agreement with the observations. The front leg is in contact with the ground and it is about to be accelerated forward to make ground contact again and therefore synthesise another gait cycle.

6.3 Vector Fields

As noted earlier the vector field of a link may provide important information of its dynamics simply and immediately. The projected components of the $\dot{\mathbf{r}}$ vector onto the auxiliary axis system of its preceding link, offer insightful graphic comparatives. Motions of the links can be quickly assessed in terms of their magnitude, direction and sequential behaviour. The diagrams in Appendix D show the front leg $\mathbf{V_{uv}}$ vectors (see Figure 8 for its definition) for the accelerating quadruped. The leg links showed greater variation in their directional fields, with only the mid-spine link showing relative uniformity. The vector fields for each of the legs and spine show approximately three groupings on a qualitative basis. The spine field, for instance, shows the oscillatory nature of this link with three cycles present. The power generating (*active*) back leg appears to have a more uniform field when compared to the

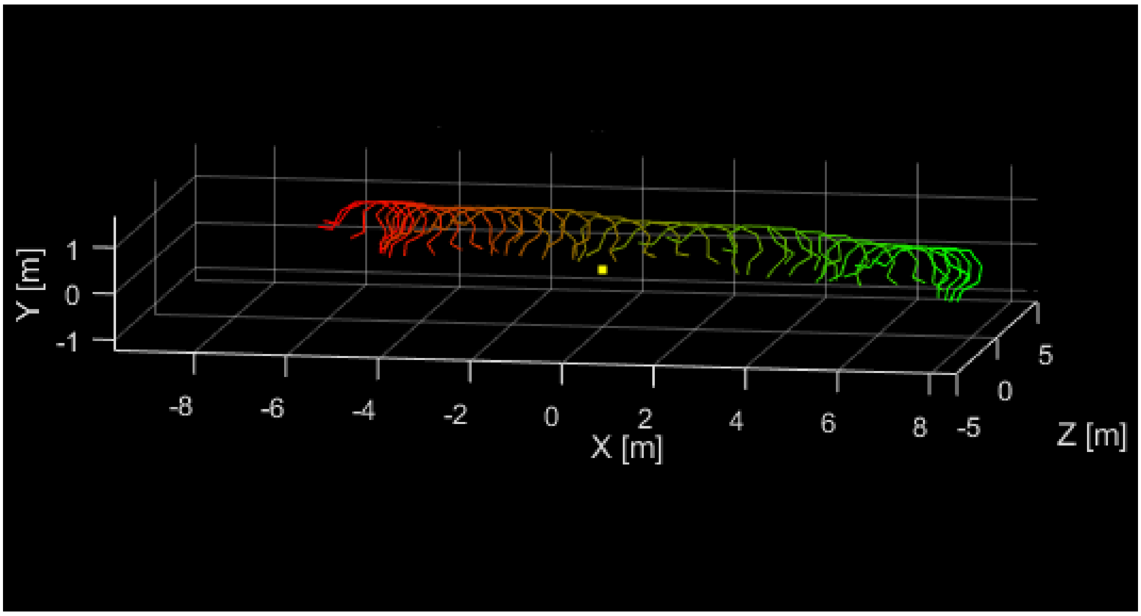


Figure 52: The full configuration sequence of the quadruped mechanism in the camera frame system with origin $(0, 0, 0)$ represented by the yellow dot. The start position is at left (red) and it traverses through space until its final position at right (green) after an estimated distance of $12.8m$ after 0.9667 seconds. It achieves an estimated maximum speed of $21m.s^{-1}$.

(*reactive*) front leg.

The average magnitudes of the polar vectors $\dot{\mathbf{r}}$ indicate higher mean values for the legs than for the spine and the back leg shows considerably higher means than the other links by some margin. These values are shown in Table 1 and the full results appear in Appendix D.

Table 1: Mean polar vector magnitudes for the back leg (BL), spine (S) and front leg (FL) links. The highly mobile back leg shows high mean values in comparison to the front leg.

Mean magnitudes of the polar vector velocities for all links								
sub-system	BL	BL	BL	S	S	FL	FL	FL
link number	#8	#7	#6	#9	#1	#2	#3	#4
$\lvert\dot{\mathbf{r}}\rvert [m.s^{-1}]$	1.39	1.76	1.82	0	1.12	0.60	1.11	1.26

6.4 Inverse Kinematics

There are four inverse kinematics models that are considered here and they require appropriate model parameters or explicit kinematic optimisation criteria (cost functions) for their operation to be acceptable. The 3D Cartesian coordinates of the end effector of the front leg obtained in the least squares optimisation is, in part, the target trajectory for the inverse kinematics models that follow. For a given end effector location these models provide the joint angles for this. Furthermore, for a given closed EE trajectory in task space, corresponding closed joint space trajectories should be generated by the algorithms, the so called repeatability property. The solutions therefore converge to a limit cycle for cyclical orbits. The IK models can also be employed to generate a smooth EE trajectory from a given start position to a final target position. A simple example would be where a serial link robot arm is required to move from its current position to a target position and perform a task such as an industrial robot in a car assembly plant or surgical robot. A controller would require the angular rates or equivalently the changes in joint angles for each processing interval. Depending on the application, a model may be

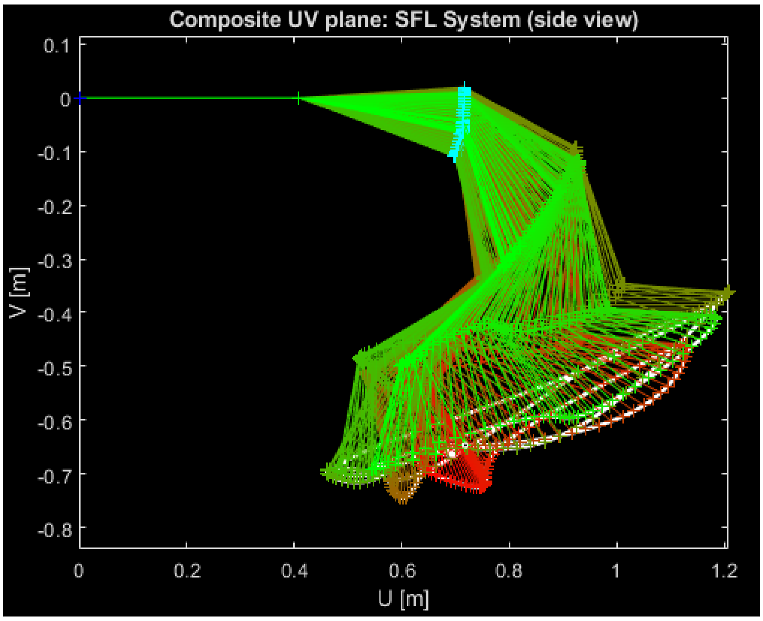


Figure 53: The target trajectory for the inverse kinematics models is generated by the front leg of the quadruped. The early phases of the trajectory are shown as red and terminate in green.

selected that offers operation within computationally allowable speeds. A specific joint is required to operate within a work envelope relative to another joint or at an optimum torque, for instance. In the EJ method the optimisation function can be tailored to minimise energy expenditure, joint torque or joint speed [14]. An optimal cyclic reference trajectory is often required for robots that move at steady motions. This repeatability can be useful and a target trajectory can be used such as in [64]. For accelerating robots the target trajectory is usually not cyclical and each gait will require a different computed online path in real time.

Practical note on coding: The models were coded to work in both 2D and 3D, however due to technical difficulties involved and long computation times for the extended Jacobian and gradient projection methods, they were applied to planar situations only. The fact that spatial trajectories involved θ and ϕ meant that the variable count was doubled so the formulation of a suitable cost function was particularly difficult. Computation times were of the order of hours for the EJ method and a Matlab exception handling error message resulted due to the infeasability of determing the pseudoinverse matrix in the GP method. The DLS and WLN methods easily processed 3D data since they were relatively basic algorithms that also did not involve computation of an inverse or pseudo-inverse Jacobian solution.

6.4.1 Model Parameters

The IK model parameters were obtained as discussed in Section 5.7. These mean values were simply inserted as constants into the models and the input end effector velocity (Cartesian system) data was processed to deliver an output depending on the application. The evaluation of the single parameter λ constant for the damped least squares method showed irregular values but in a narrow band as shown in Figure 54. The extended Jacobian method was not so straight forward. The derivation of a suitable optimisation function was problematic and only by a trial and error approach were two functions isolated. The EJ method also had a scaling factor α that could also be computed online (using the formulation discussed earlier) but its constant (mean) value as discussed in Section 2.6.4 was used for trajectory generations and tracking problems. Apart from a single spike in the α parameter values,

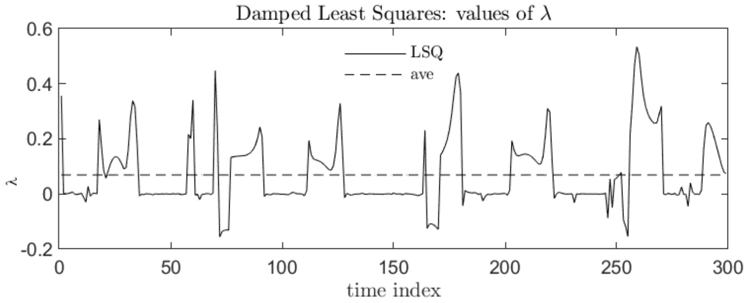


Figure 54: Inverse kinematics parameter λ as determined by a non-linear least squares optimisation for the damped least squares method.

they stay within a narrow range with the data having a mean of 0.643 and standard deviation of 0.9294. The online computation of the scaling factor α and its effects were also investigated.

The EJ and GP methods, which entailed determining the pseudoinverse Jacobian, showed values very near to the parameterised value determined earlier. The parameter values for the GP method as determined by the optimiser are shown in Figure 56. In the case of the DLS and WLN methods α lay within a narrow band, with a higher variation than obtained with the aforementioned methods and with an accompanying degradation in joint space performance (see Figure 57). The results of the parameterisation of these two methods are respectively shown in Figures 54 and 55. The online computation of α (see Figure 58) only showed a marginal improvement when included in the algorithm whereas its inclusion in the other algorithms generally produced no or slightly better performance in terms of processing times and convergence rates. The extended Jacobian method stood out as having both quicker processing times and the fewest number of iterations to reach the target. The gradient projection method also had very high times to process the variables and solution vector on each iteration compared to the others, with the DLS taking many more iterations to reach the target using an online α determined value.

6.4.2 Trajectory Generation for End Effector Tracking

In order to track the end effector a vector of angular rates is determined by the IK model that will affect this for the entire sequence of configurations. A target trajectory was set as the outgoing (or first half) sequence of the second gait of the

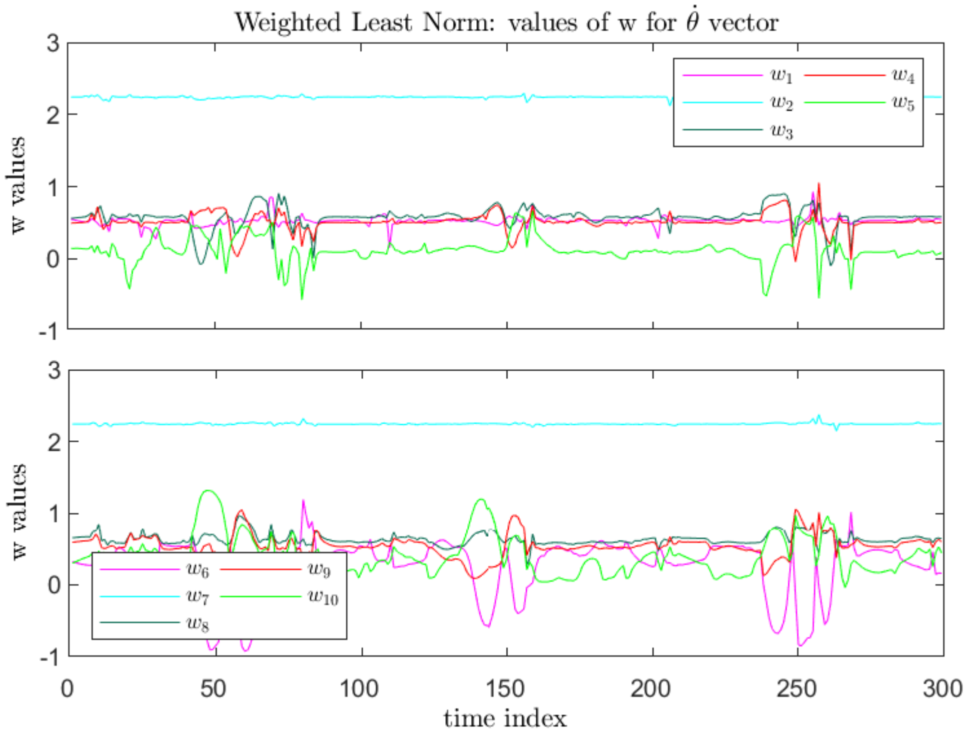


Figure 55: Inverse kinematics constants as determined by a non-linear least squares optimisation for the Weighted Least Norm method. The variables that comprise the diagonals of the weighting matrix W are $w_1, ..., w_{10}$.

quadruped; there is therefore only an outgoing and non-returning trajectory involved here. The motion may be likened to that of a serial link manipulator arm that is required to move optimally from a start position to a final target position within its workspace. The actual LSQ trajectory and that determined by the DLS method in 3D space is shown below where it achieved the $3mm$ approach distance within a few hundred iterations. The exponential decrease between the current and target points was also encountered in [52] where this rapid convergence is ideal. The iteration times for all of the models were based on the time taken for the angular rates vector to be computed which were treated as function handles in the Matlab execution code using a $1.6GHz$ Intel Celeron CPU. Average iteration times varied between $0.0112ms$ and $1.191ms$ with associated standard deviations ranging between 0.0052 and 0.1149. The results for all four methods are shown in Figure 60. In work done by [65] using a Y-link mechanism comprising 7-links and two end effectors, the iteration times varied between $1.1\mu s$ to $2.2\mu s$ where the DLS and Jacobian transpose methods were used on a Pentium $2.8GHz$ machine in C++ code. Only the current

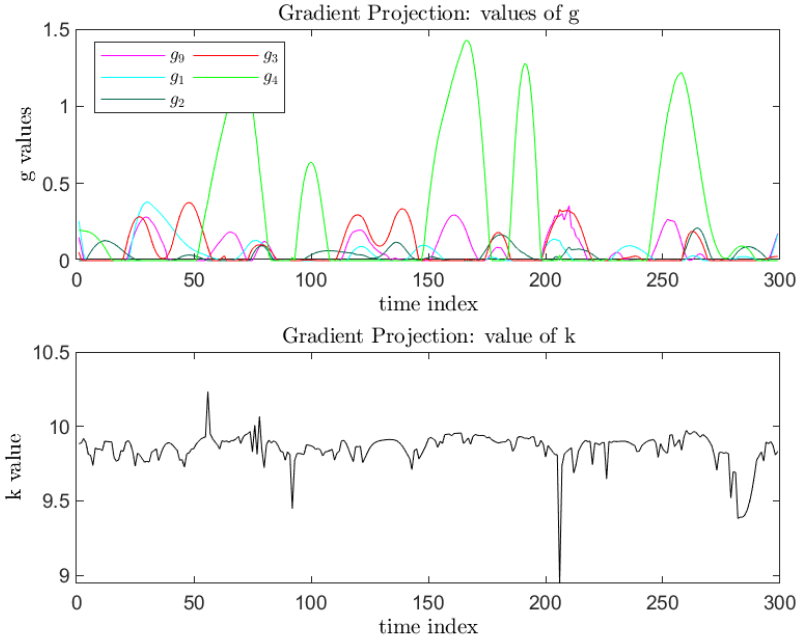


Figure 56: Inverse kinematics parameters $g_1, g_2, ..., g_5$ and k, for the gradient projection method as determined by the non-linear least squares optimisation solver.

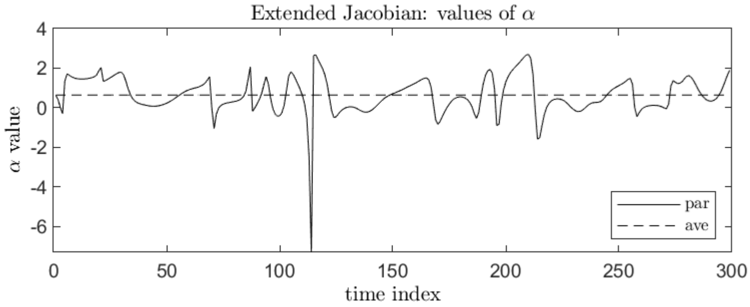


Figure 57: The estimated parameter α values for the extended Jacobian method together with its average of 0.643.

start and end target points are specified together with the orientation of the configuration at the start and the algorithm is expected to output an optimal solution trajectory. The evolution of the configuration sequences thus progress iteratively from the start configuration according to the inverse kinematics model. Practically, the start point is at or close to ground level and the final target point is specified as the maximum point of extension for a specific gait.

The models were evaluated on their ability to achieve the target point within a practical finite number of iterations (i.e. feasible time period), final closure distance, smoothness and length of the trajectory. The minimisation of a cost function

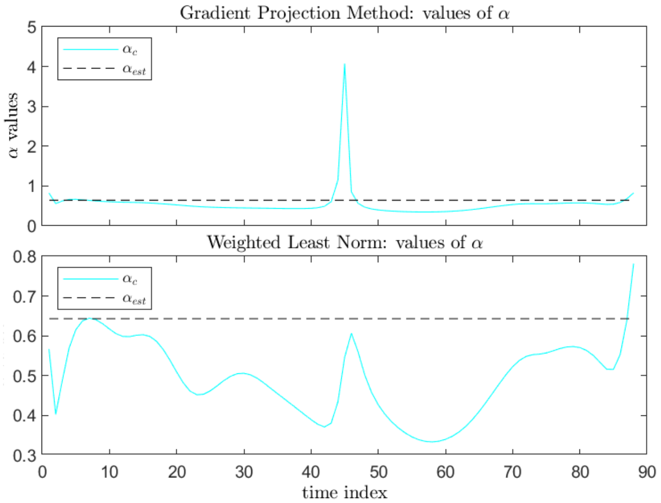


Figure 58: The variation of the scaling factor α as computed online for the GP and WLN methods. The mean estimated value as determined by the EJ method is indicated by the dashed line.

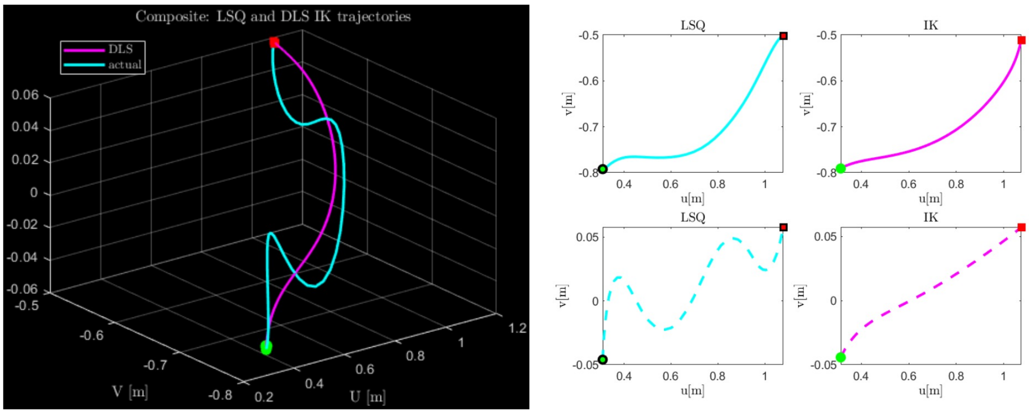


Figure 59: The DLS method produced this trajectory (magenta) given a target point (red). A similar result was obtained with the WLN method. The cyan coloured trajectory is the actual path taken by the end effector. Notice how the IK determined path of the end effector is smoother and less oscillatory, particularly in the UV plane, than the original actual trajectory.

that relates the spatial distance between the current end effector position and its final target position was employed throughout the iteration process. The result is that on each iteration a set of joint angle changes are computed that finally produce a sequence of configurations realising a trajectory for the EE within a set of constraints. Once again the DLS and WLN methods gave similar configuration sequences and thus end effector trajectories. Termination of the code was programmed to occur once the generated trajectory was within a $3mm$ band of the final target point or a

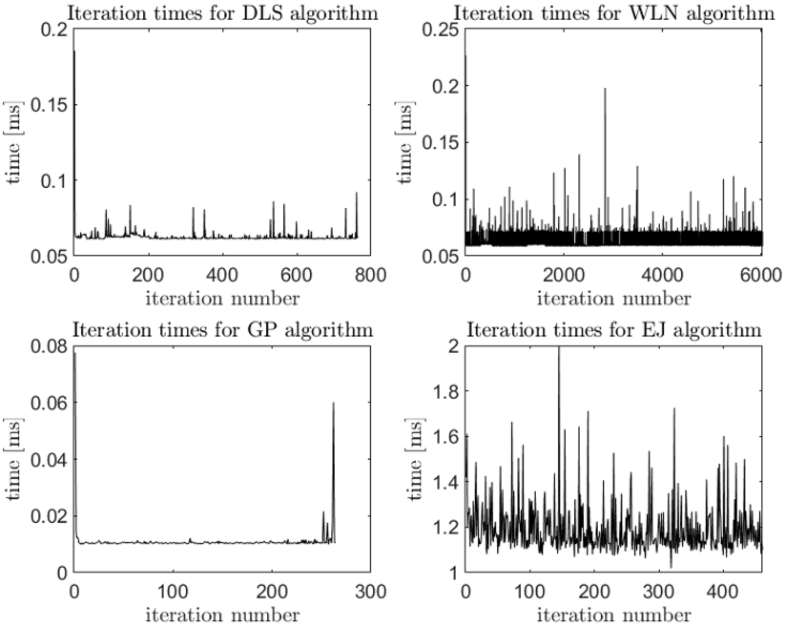


Figure 60: The iteration times for all of the IK models.

code break would take place at a threshold time or distance value determined on a case by case basis where there was slow convergence. The rapid convergence by the end effector to the target point, as computed by the DLS and WLN methods, can be seen in Figure 61 and the associated configuration sequences for this are illustrated in Figure 62.

The formulation of a suitable optimisation function for the extended Jacobian method proved problematic. Various permutations of relative angle rates or kinetic energies were considered. Only a few enabled solutions that honed in on the target, with the others resulting in either Jacobian matrices going singular or estimated trajectories being chaotic, discontinuous and with no convergence to the target point. See Ap-

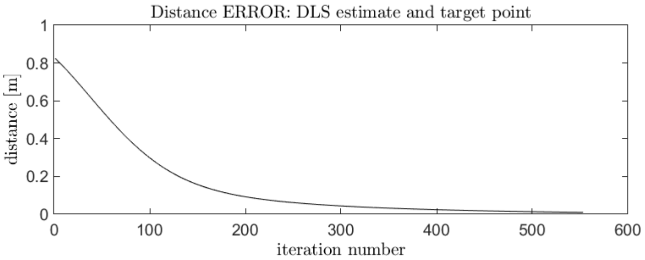


Figure 61: The rapid convergence to the target is shown by the distance in task space between the current model estimate and the target. The data shown here was for the DLS with the WLN method showing a similar trend.

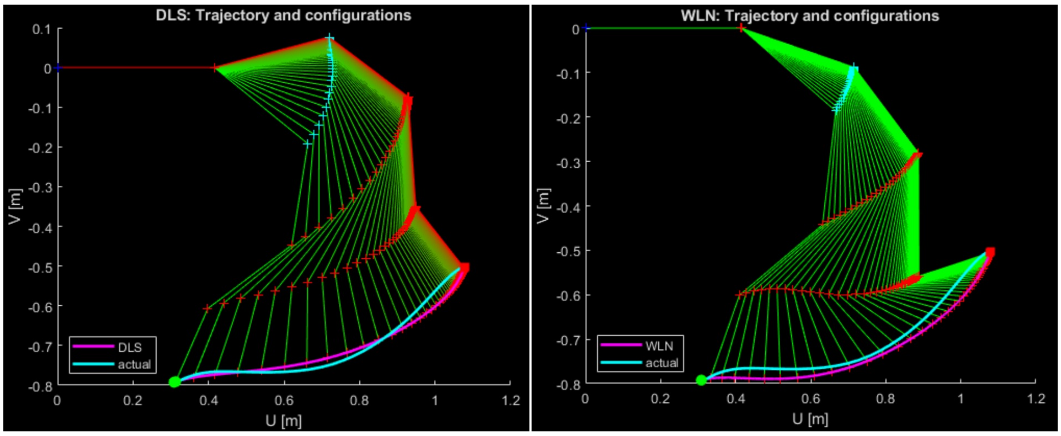


Figure 62: All the configuration sequences and proposed solution trajectories are similar for the DLS and WLN algorithms.

Table 2: Trajectory generation inverse kinematics data from current to target point.

IK method	mean iteration time [ms]	total CPU time [s]	no. of iterations	ERR [mm]
DLS	0.063	213.98	765	3.0
WLN	0.063	1733.50	6100	3.0
EJ	1.191	253.56	587	32.7
EJ*	2.90	203.13	461	32.7
GP	0.0112	1527.20	264	11.0
* α computed online.				

pendix B for these.

The optimisation function that was most suitable had the form of the augmenting harmonic function discussed in Section 2.6.4. An attempt was made to select the values of the constants a and b around the arbitrary nominal values used in [7]. In addition to achieving speedy convergence to the target, the other criteria was that smooth, closed paths were achieved in the configuration space similar or better to those obtained in the other methods. By choosing appropriate relative angle variables and constants a and b, convergence to the target was finally achieved within the specified $3mm$ band. The values used were $a = 1$, $b = 0$ and $\mathbf{q} = (q_1, q_2)$ with $q_1 = (\theta_2 - \theta_3)$ and $q_2 = (\theta_4 - \theta_3)$. The resulting optimal augmenting function is a harmonic mapping with structure as shown in Figure 63. Far fewer iterations are required to reach the target than was done with the WLN method, and in addition the EJ now guarantees that the pseudoinverse Jacobian will maintain full row rank and therefore no singular configurations will occur in the solution. The usefulness

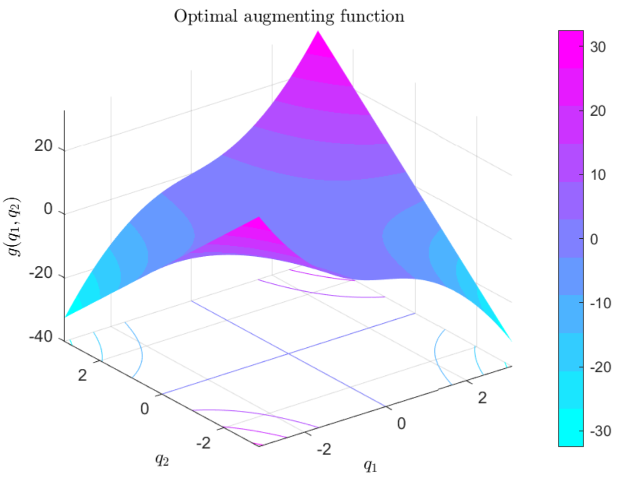


Figure 63: The optimal augmenting function that is a harmonic kinematics mapping for the EJ algorithm. The variables q_1 and q_2 are relative angles corresponding to two links.

of the augmenting function in achieving quicker convergence is highlighted in Figure 64. Total CPU times for the DLS and EJ were quicker by a factor of about 8 compared to the other two methods. See Appendix B for these permutations. The configurations required for the smooth end effector motion to the target by the EJ method can be seen in Figure 65. Finally, the gradient projection method gave a

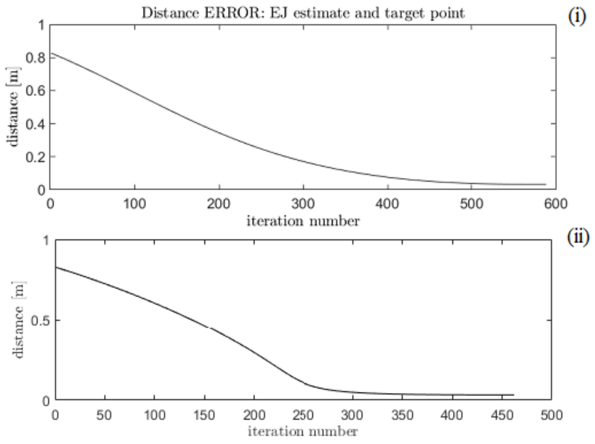


Figure 64: By using a suitable harmonic augmenting function with the extended Jacobian method, quick convergence to the target is achieved with no singularities. The computation of α online (ii) results in a faster convergence to the target than using $\alpha = 0.643$ constant (i).

trajectory solution with very few iterations (only 264) and a final distance of $11mm$ from the target point. The actuation rates were below the average rates for all the

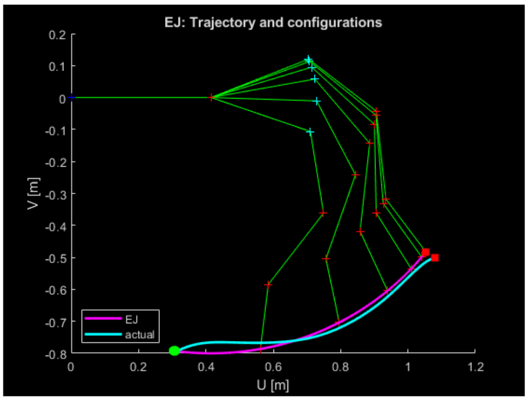


Figure 65: The optimal trajectory produced by the extended Jacobian method. The optimisation function took the form of a harmonic augmenting function with the variables being selected relative motion angles. Every hundredth configuration is shown.

models. The evolution of the links to execute the smooth trace in reaching the target is shown in Figure 66.

The algorithm that enables the target to be reached smoothly and without discontinuities and with the least energy expenditure in a specified time is the one of choice. It is clear that for this application the extended Jacobian method appears to give the better solution, since it reached the target within the fewest number of iterations and comparably low iteration times.

It is also clear from Figure 59 that the DLS and WLN methods were able to generate

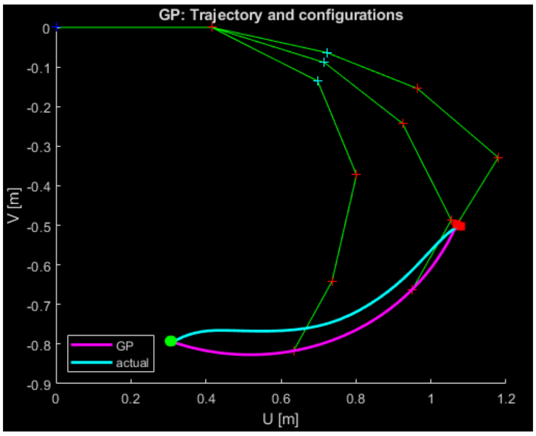


Figure 66: The gradient projection algorithm produced this trajectory in only 264 iterations with each iteration taking on average 0.36 microseconds. Every hundredth configuration is shown too.

smooth, efficient trajectories in the UW transverse plane, with integral path lengths less than the actual path taken. All of the data statistics for the inverse kinematics methods are summarised in Table 2. Performance indicators include the mean iteration and total CPU times taken for the computation, the number of iterations taken to reach the target and the mean error.

6.4.3 Trajectory Closure

Inverse kinematics methods are only practically useful if they are able to lift closed end effector paths to closed joint angle paths [14] and with minimal oscillations or induced self-motion (without a change in EE position) in the joint space. A useful metric to quantify the degree to which a phase space trajectory in a particular space is closed is the final separation distance between the start and final points of the trajectory given by D_{sf} (see Figure 17) as shown below for an arbitrary path. This is a good indicator of performance if the relative trajectories being considered have varying ranges and/or profiles. For comparisons between models, the percentage difference of the error in respect to the variable's maximum ranges can also be considered. The degree of closure of an orbit is therefore an informative metric irrespective of the spatial dimension, so a value of 100 represents a fully closed orbit where the start and termination points coincide exactly. This is all detailed in Section 3.5.

The ability of a model to generate closed joint space trajectories for a closed EE trajectory is key to its correctness and therefore usefulness, so to test the efficacy of each IK model the individual joint space trajectories in phase space were computed for a single gait cycle. The full three gait cycles of the EE, as estimated by least squares optimisation, shows cyclical tendencies in the orbits with the start and end positions close together. The least squares results show that the phase states of the end effector are orbital and converge to a limit cycle with clear evidence of nearly repeating gaits (see Figure 68). The start of each gait is in the lower right quadrant with termination points in close proximity. In order to isolate a single repeating gait it is convenient to select the start/end points which coincide with a ground contact instance. A study of the previous sequences reveals that this occurs at (i) local x position minima, (ii) local absolute y position maxima and (iii) primarily the x velocity

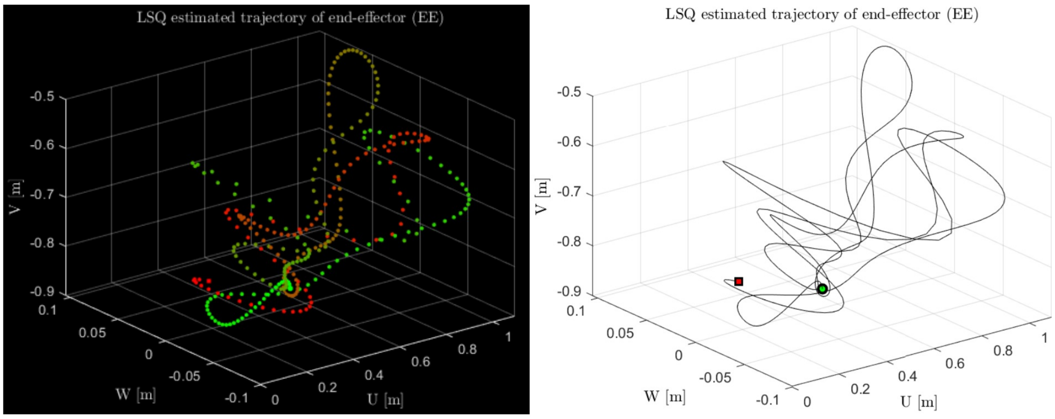


Figure 67: The front leg end effector trajectory as estimated by least squares optimisation.

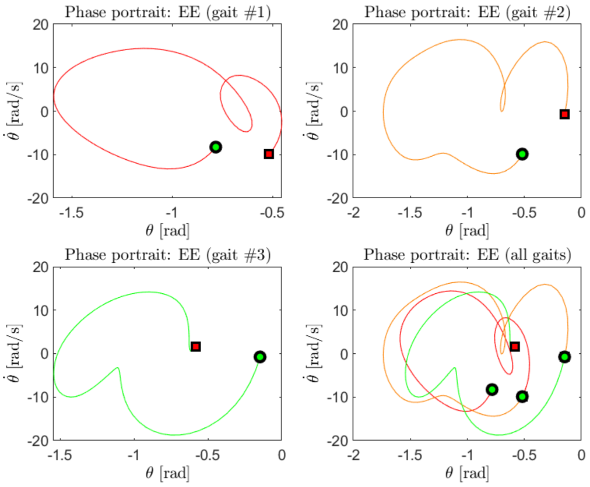


Figure 68: The phase states of the joint angle of the end effector link of the front leg as determined by the solver. Three gait cycles have been isolated here and the full cycle (lower right) shows convergence to a limit cycle.

component being zero. For each ground contact point all the velocity components will be zero in theory but for a given interval point this is rarely true due to implicit model and heuristic data errors. By applying the above criteria several local range values for the onset and termination of all the gaits were isolated and a set of mean configuration index numbers were obtained. Three gaits were thus identified as progressing on the following configuration number intervals $[24, 113]$,$[113, 201]$,$[201, 293]$ with the second gait (configuration #113 to #201) judged as being the most suitable as shown in Figure 69. The complete trajectory of the end effector in 3D space (see Figure 67) with the initial (red), intermediate (amber) and final (red) motion

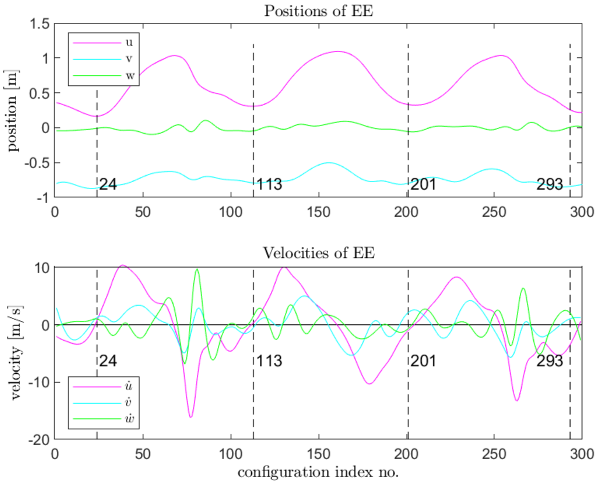


Figure 69: Demarcated boundaries for all three gaits. The second gait is defined as occurring from the configuration index numbers 113 to 201 based on the position and velocity data of the end effector.

sequences displayed for convenience. It is clear from Figure 70 and the fact that the degrees of closure in both dimensions are virtually coincidental that a closed end effector trajectory has been isolated here. All the joint space trajectories gener-

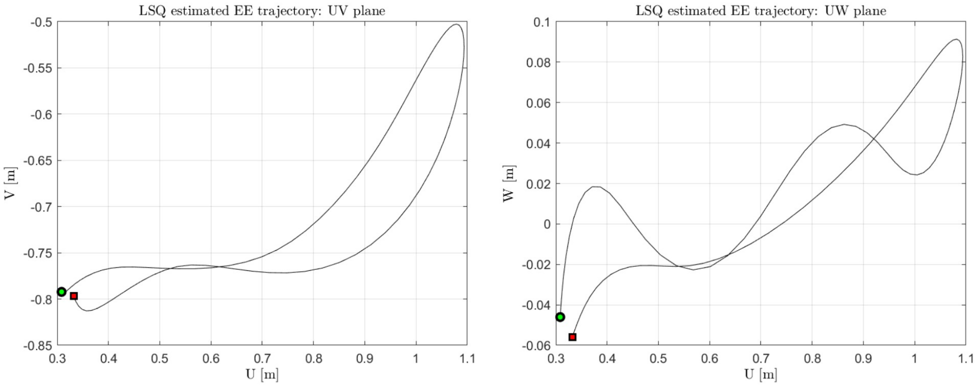


Figure 70: The nearly closed trajectories of the end effector, in Cartesian coordinates, for the second gait.

ated by all of the IK models show a remarkable similarity on a qualitative level, in both phase spaces, with the exception of the EJ method, since it is minimising the shoulder and spine joint velocities. Each algorithm determines the specific values in phase space uniquely though. There is not full closure in phase space (see Figure 71) of the EE as determined by the LSQ solver but this is partly also due to the isolated gait orbit not having 100% closure. From all the phase portraits it is clear

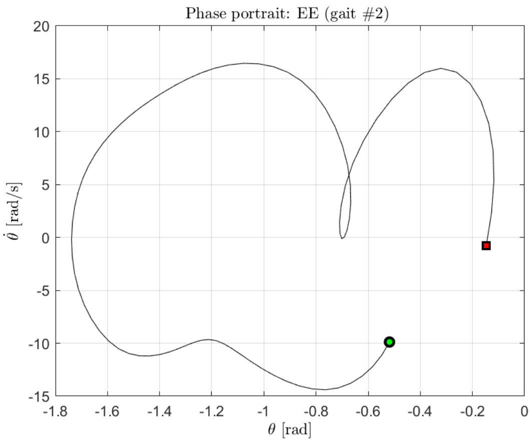


Figure 71: The phase states for the primary motion plane of the end effector as determined by the least squares solver.

that the shoulder joint θ_2 achieves a smooth uniform very nearly closed trajectory except for the EJ method once again where it can be seen to be minimising this motion; the minimisation of the spine joint is also apparent with almost half of the trajectory having small angular velocities.

The angle joints θ_1 and θ_2 show very similar shapes, but with a very different orbit for θ_1 this results in the EJ method delivering a very different end effector trace compared to the other three methods. It can be broadly stated then that all other

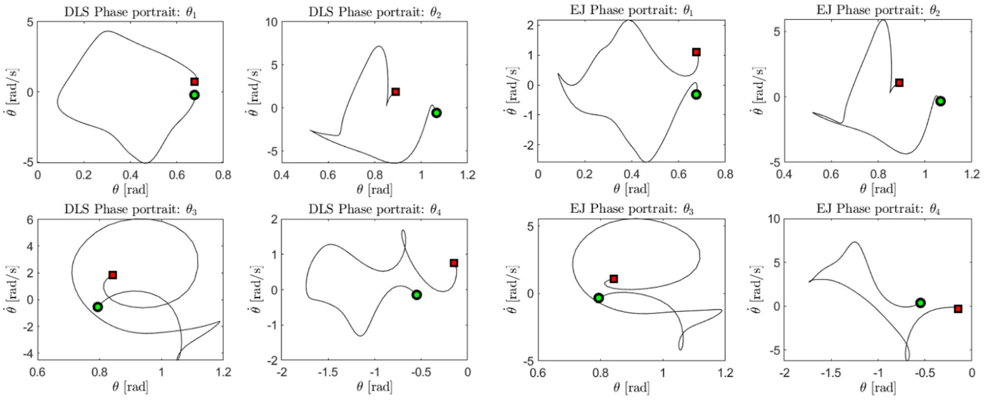


Figure 72: The primary phase states for all the joints as determined by the damped least squares (left set) and the extended Jacobian (right set) methods.

joint spaces achieve near closure and this also applies to the ϕ joint phase spaces. See Figures 72 and 73 as well as Appendix A for all of the phase state plots. Tables 3 and 4 summarises the trajectory closure results. The Matlab coded performance indicators, such as the average iteration computation time and total CPU time, are

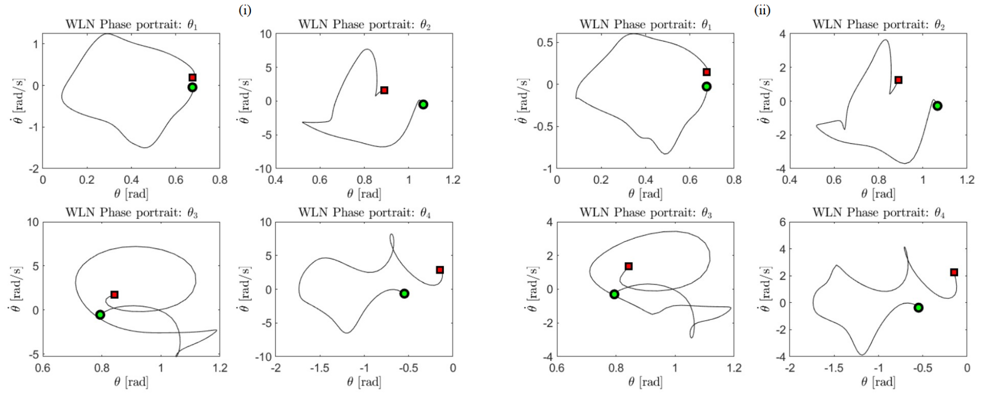


Figure 73: The effects of computing the scaling factor online and applying it to the angular rates vector shows similar to a very slight degradation in performance (ii) with slightly less smooth trajectories and nominal to increased DOCs when using the WLN method.

Table 3: Trajectory closure percentages in θ joint space for all the inverse kinematic models in the UV plane.

Trajectory closure percentages: θ phase space				
IK method	joint: θ_1	joint: θ_2	joint: θ_3	joint: θ_4
DLS	90.04	81.97	77.53	71.11
WLN	91.87	85.37	81.90	75.79
EJ	70.67	85.51	86.13	94.12
GP	89.69	81.96	78.91	72.22

Table 4: Trajectory closure percentages in ϕ joint space for all the inverse kinematic models in the UW plane.

Trajectory closure percentages: ϕ phase space				
IK method	joint: ϕ_1	joint: ϕ_2	joint: ϕ_3	joint: ϕ_4
DLS	65.64	56.35	57.97	51.34
WLN	65.25	51.99	52.69	57.11

shown in Table 5. Average closure percentages ranged from about 77% to 83% for

Table 5: Performance indicators of all the inverse kinematic models for trajectory closure in the UV plane.

Trajectory Closure Performance Indicators			
IK method	mean iteration time [μs]	total CPU time [s]	no. of iterations
DLS	0.71	32.4	88
WLN	0.71	44.6	88
EJ	2.1	122.6	88
GP	2.1	809.6	88

all joints and models, with the EJ method showing relatively low closures of 61% and 62% for the cost function variables θ_1 and θ_2 but recording the highest degree of closure value of 94% (for θ_4) out of all the models. Single iteration times varied from 0.71 to 2.1 microseconds with total program run times for all 88 iterations from 0.5 to 14 minutes. In cases where the inverse Jacobian is evaluated, such as the GP and EJ, computation times are substantially slower with the evaluation of the pseudoinverse for 3D points drawing a Matlab exception error. Issues with computer memory were encountered where lengthy equations involving symbolic variables were encountered, however clearing these variables at specific lines within the code solved this problem.

Chapter 7

Conclusions

7.1 Image Extraction by a Trained Network

A sufficiently well trained network was developed to produce the predicted image data for the defined markers on the animal subject. The reprojected image data on the original images confirmed that the optimiser was delivering sensible data and was one key output for validation purposes.

7.2 Configuration Optimisation

The accuracy of the optimisation results can be validated by studying the software code indicators, Jacobian arrays, gradients matrix, the mean error values for the joint image estimates and the output data (reprojected images, computed configuration poses, etc).

7.2.1 Basic Data and Code Indicators

The actual and predicted image data illustrate that the reprojected data is credible and that the basic mechanics of the functions and sub-systems of the model are working correctly. The generally very low ϕ angles, reference joint predictions and realistic link length estimate values were in broad agreement with the expected values. The collective output of the Matlab indicators viz., number of iterations, first-order optimality values, residuals, norm of the step, function count and exit-flags, that was returned by the optimisation process, showed that the solution vector was close to being optimal. The low gradient matrix values and low mean residuals provide further evidence that the results are acceptably accurate and valid. The final step sizes, at solution, show that the search for a solution has reached termination and it also clearly shows that the operational improvements of the solver increase after the second iteration when the full functionality of the objective function is utilised with the effects of angular rates included. The high function values are a result of the many dimensions of the objective function and the fact that gradients were computed centrally by finite differences during the solution process. Due to the complexity of the quadruped system and the associated objective function, it is difficult to conclude with absolute certainty that the results are totally accurate.

7.2.2 Vector Fields

The vector field profiles illustrated the qualitative behaviour of the links that were deemed to be reasonably representative and in line with expectations.

7.2.3 Configuration Plots

The three dimensional pose configurations that were produced showed that all motions appeared to correspond to the expected local and global motion regimes. Correspondences with the video images themselves and the analysis of the legs and spine in the different motion planes further validated the output of the solver. The velocity and acceleration data that was extracted for the motion of the reference joint reflected trends with direct observations of the subject undergoing acceleration.

7.3 Inverse Kinematics

7.3.1 Parameters

In deriving the parameter constants for the inverse kinematics models, it was found that the values were fairly constant for the sequence considered. This was not the case for the λ parameter in the damped least squares equation and most notably the g constants assigned to the gradient projection model. An adaptive model could be used, but the short time periods involved here ensured that the mean values were taken. Only one of the five constants had large variations (1 to 5.5) and the others varied below 2. Of course, this is a relative assertion but the final parameters proved adequate. As predicted the least norm solutions were erroneous and meaningless. The gradient projection method showed to be useful but the very long computation times meant that it was not considered for further applications. The weighted least norm and damped least squares methods showed good performance. Methods that do not require a structured representation of the dynamics and that instead rely mainly on the robot kinematics are preferred, since it is more robust to parameter errors [20].

7.3.2 Trajectories

The damped least squares and weighted least norm methods were able to generate optimal trajectories of the end effector in three dimensions. The gradient projection method was also able to realise optimal trajectories but at high computational expense. The extended Jacobian method delivered trajectories that were sub-optimal, but with the added criterion that the angular rate of the shoulder joint speed is to be minimised. Difficulties in determining an appropriate cost function for the EJ method were experienced. For the closed loop trajectory problem the models were able to achieve near repeatability which entailed generating closed joint space trajectories for the (almost) desired closed end effector trajectory. The extended Jacobian method delivered results that showed that the shoulder angular rate was minimised in comparison with the phase state trajectories of the other three models.

Although of secondary importance, the trajectory closures in the second generalised variable (ϕ) had fair to reasonably high closures for the DLS and WLN methods, that provided three dimensional trajectories.

Chapter 8

Recommendations on Future Work

Based on the results obtained and the conclusions drawn, the following recommendations are provided.

8.1 Least Squares Optimisation

Investigations into obtaining the initial estimates for the LSQ solver more efficiently and accurately during the entire iteration process should improve the accuracy of the estimates. The development of additional vector functions that will add to the overall model description and aid in solving the unknown variables, particularly the camera rotation angles, is suggested.

The effect of a twist γ_i in any link i may be modeled by determining by how much it affects the θ_{i+1} and ϕ_{i+1} values of $link_{i+1}$; the change in the two angle variables change the coordinates and these will result in small changes projected onto the UVW-system which are then incorporated in the model. If this twist is not transferable then this means that the quadruped now has spherical joints and the resulting $8S$-serial open chain mechanism now has 27 degrees of freedom (up from 20 DOFs for the $8U$-serial mechanism) and this complicates the computations.

A more accurate mechanism could be introduced that includes the visco-elastic properties of a lightweight quadruped in future work, which would entail a far more complex model for the link proportion equations and the effective angles between the links themselves.

An implicit evaluation function for determining the degree of optimality could be included in the optimiser in future.

8.2 Inverse Kinematics

In applications where computation times are not critical and higher accuracies are desired, it may be best to determine the inverse kinematics parameters online and either iteratively or using batch iterations. A constraint function should also be included that eliminates spiking values that are beyond a threshold standard deviation of the iteration current mean or running batch average.

The comparative effects of applying the online α scaling function to the IK model being used and the improvements in target convergence, computation time and joint trajectory closures, should also be investigated. In addition, the work on numerical filtering, similar to that done in the selectively damped least squares, may be applicable to the IK algorithms where applicable.

8.3 Future Research

The development of legged robots of the future will, broadly speaking, involve more use of machine learning and AI to tackle problems such as uneven terrain and external environmental effects that cannot be programmed into any control strategy, where there may be highly discontinuous and/or nonlinear setpoints that are not easily modeled or that occur beyond real-time processing times.

Hybrid algorithms may be developed using non-linear optimisations in conjunction with machine learning provided they are computationally feasible in real time applied to high acceleration modes.

If the latest advances are anything to go by, there may be complex multiple subsystems that operate on mimicking bio-muscular skeletal systems.

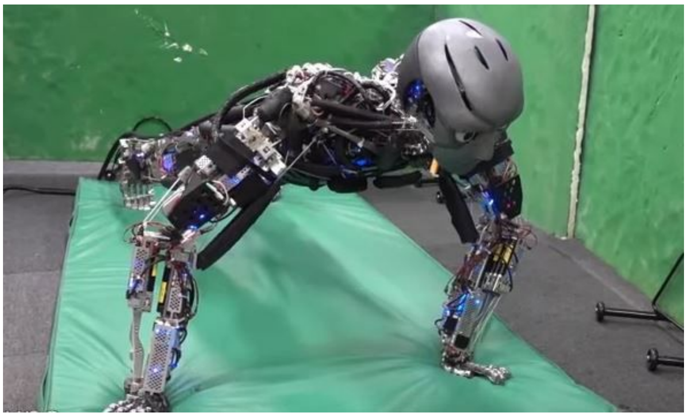


Figure 74: The Kengoro humanoid robot with intricate musculo-skeletal links (Image credit: JSK Laboratory, University of Tokyo).

8.3.1 Humanoids

Highly dextrous humanoid robots that are able to move and interact seemlessly with humans are a realisable future scenario. The motion kinematics will depend on the joint systems employed (rotary, prismatic or hybrid versions) and the actuator and physical limits of the technology. Some advanced humanoids like Kengoro have 174 maneuverable joints and 116 actuators that ensure high motion ranges and are cooled by a network of heat exchangers.

8.3.2 Quadrupeds

In the context and scope of this project, future research could include developing further optimisation functions that minimise energy or torque along trajectory paths for the end effectors. If a 4-link system should be proposed for each of the legs, this will complicate any kinematic optimisation for all the joint angles and further vector functions that parameterise constraints and physical limitations would have to be introduced. Robots operating at high speed and/or in rough terrain will be subject to chaotic dynamics [16] and quick online adaptation to these changes is vital. Current quadrupeds have applications in the military where payloads can be carried over uneven and steep terrain. They are heavy duty machines and are robust to invariant contact forces and they have self stabilising control algorithms that counteract externally applied random events.

Other applications are in security monitoring and the delivery of lightweight items to humans in urban areas.

8.3.3 Surgical and Service Robots

The DaVinci Surgical Robot is currently one of the most advanced surgical robots together with that of Versius. Assistive surgical robots are now applied that require guidance and there are many systems that perform non-invasive surgery. Future work that is proposed here is an assistive surgical arm that uses deep learning and AI to accept verbal commands and possibly physical cues from a human surgeon. The application will be task specific and will require comprehensive medical image training data, repeatable trajectory priors and the ability to mimic the upper torso motions and match or exceed the dexterity of a human hand. Work done in this regard include [25], [44] and particularly [46]. Furthermore, future research into advanced haptic sensory technology will ultimately allow the robot to quickly distinguish between muscle, tissue, cartilage, bone etc. and thus have the ability to navigate its way to target specific locations within the human body that are not easily accessible or identifiable.

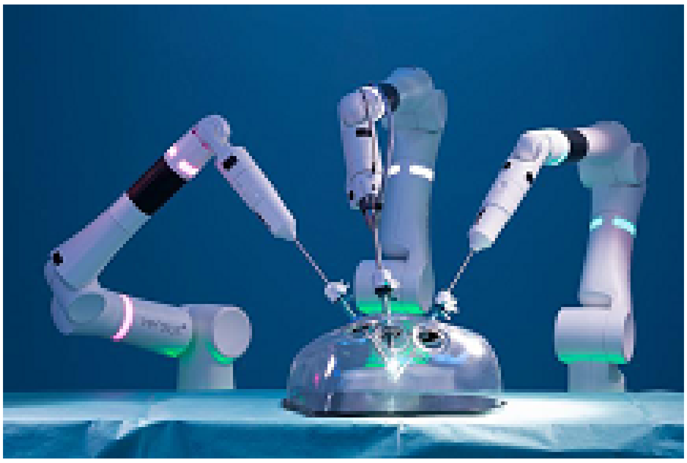


Figure 75: The Versius Surgical System (Image credit: CMR Surgical).

8.3.4 Autonomous Quadrupeds in Space Exploration

High speed quadrupeds that are able to negotiate obstacles with full autonomy on remote locations in space seem like an option for extraplanetary surveying and exploration. Environments with lower gravity will make it easier to move quickly and carry high payloads.

Sealed pneumatic powered systems may be preferred over electrically motorised ones where the risk of contamination, due to ultra-fine particulates on other worlds, could pose as serious performance inhibitors. The Pneupard is an experimental quadruped, developed by Osaka University, that uses pneumatics to effectively power synthetic 'muscle' structures that allow deformation on contact and thus energy absorption with less 'stiff' oscillation characteristics than those using rigid metal links powered by electric motors or hydraulics. The inclusion of damping models could serve as a refinement to describe the inherent limitations on accelerated motion, while the advantages of free-swinging (and thus possibly chaotic) motions could be investigated further. However, this is a challenging task since unpowered joints highly complicate the stability investigation of such mechanisms as noted by [16]. The self-stabilisation of robotic systems under optimum energy and torque conditions is crucial to furthering efficient operation, especially in highly variable terrain that is able to increase the dynamic complexities and which may require highly adaptive control.

Finally, the importance of non-linear systems analysis tools is crucial to locating (unwanted) chaotic regimes and for underactuated joints the stability investigation is highly complex. In addition, solutions to the self stabilisation and that achieve this with minimal energy is a core research area for the future.

Bibliography

[1] A. Werner, W. Turlej, C. Ott, "Generation of Locomotion Trajectories for Series Elastic and Viscoelastic Bipedal Robots", *IEEE International Conference on Intelligent Robots and Systems*, 2017.

[2] T-W. Lu, C-F. Chang, "Biomechanics of Human Movement and its Clinical Applications", *Kaohsiung Journal 0f Medical Sciences*, 2011.

[3] M.Edgington, Y.Kassahun, F.Kirchner, "Dynamic Motion Modelling for Legged Robots", *IEEE International Conference on Intelligent Robots and Systems*, October 11-15,2009.

[4] X. Zang, S.Iqbal, Y. Zhu, X. Liu, J. Zhao, "Applications of Chaotics in Robotics", *International Journal of Advanced Robotic Systems*, March 2016.

[5] J. Ching, J.L. Beck, K.A. Porter, "Bayesian State and Parameter Estimation of Uncertain Dynamical Systems", *Probabilistic Engineering Mechanics*, August 2005.

[6] B. Leela Kumari, K. Padma Raju, V.Y.V. Chandan, V.M.J. Rao, "Application of Extended Kalman Filter for a free Falling Body towards Earth", *IJACSA International Journal of Advanced Computer Science and Applications*, Vol.2,No.4, 2011.

[7] K. Tchoń, "Optimal Extended Jacobian Inverse Kinematics Algorithms for Robotic Manipulators", *IEEE Transactions on Robotics*, January 2009.

[8] D. Rowell, "Analysis and Design of Feedback Control Systems: State Space Representation of LTI Systems", MIT Course/Lecture Notes: *http://web.mit.edu/2.14/www/Handouts/StateSpace.pdf* ,pp.1-18, October 2002.

[9] "Filtering and State Estimation", MIT Course Work, Source URL: *http://fab.cba.mit.edu/classes/864.17/text/filt*, pp.237-254.

[10] M. Ondera, "Matlab Based Tools for Nonlinear Systems", Department of Automation and Control, University of Technology, Bratislava.

[11] J. Shen, A.K. Sanyal, N.A. Chaturvedi, D. Bernstein, H. McClamroch, "Dynamics and Control of a 3D Pendulum",Department of Aerospace Engineering, University of Michigan, Ann Arbor.

[12] R. Kandepu, B. Foss, L. Imsland, "Applying the Unscented Kalman Filter for Nonlinear State Estimation", *Journal of Process Control*, July 2007.

[13] A. Belyaev, "Vector Fields, Winding Number and Index.Poincare Theorem", *http://www.mpi-sb.mpg.de/ belyaev.*

[14] C. RunBin, C. YangZheng, L. Lin, W. Jian, M. Hong Xu, "Inverse Kinematics of a new Quadruped Robot Control Method", *International Journal of Advanced Robotic Systems*, November 2012.

[15] J. McBride, S. Zhang, M. Wortley, M. Paquette, G. Klippe, E. Byrd, L. Baumgartner, X. Zhao, "Neural Network Analysis of Gait Biomechanical Data for Classification of Knee Osteoarthritis", *Biomedical Sciences and Engineering Conference (BSEC), IEEE*, April 2011.

[16] S. Iqbal, Z. Xi Zhe, Z. Yan He, Z. Jie, "Bifurcations and Chaos in Passive Dynamic Walking: A Review", *Robotics and Autonomous Systems*, 2014.

[17] R.N. Kirkwood, H. de Alencar Gomes, R. Ferreira Sampaio, E. Culham, P. Costigan, "Biomechanical Analysis of Hip and Knee Joints during Gait in Elderly Subjects", *Acta ortop.bras*, Vol.15, no.5, 2007.

[18] J. Barreto, A. Trigo, P. Menezes, J. Dias, A.T. de Almeida, "Kinematic and Dynamic Modeling of a Six Legged Robot", Institute of Systems and Robotics, Leiria & Department of Electrical Engineering, University of Coimbra, July 1998.

[19] M. Tarokh, M. Lee, "Systematic Method for Kinematics Modeling of Legged Robots on Uneven Terrain", *International Journal of Control and Automation*,Vol.2,No.2, June 2009.

[20] L. Righetti, J. Buchli, M. Mistry, S. Schaal, "Control of Legged Robots with Optimal Distribution of Contact Forces",*11th IEEE-RAS International Conference on Humanoid Robots, IEE*, October 26-28, 2011.

[21] O. Ruf, "Dynamic Modeling of Robots with Kinematic Loops", School of Electrical Engineering, Aalto University and Lulea University of Technology, pp.1-14, August 2014.

[22] O. Ruf, "Dynamic Modeling of Robots with Kinematic Loops", School of Electrical Engineering, Aalto University and Lulea University of Technology, pp.15-26, August 2014.

[23] J.S. Lam, "Control of an Inverted Pendulum",*Journal of Computer Science*, 2004.

[24] C. Yang, H. Ma, M. Fu, "Advanced Technologies in Modern Robotic Applications", Source URL: *http://www.springer.com/978-981-10-0829-0* pp.27-43, 2016 .

[25] V. Lertpiriyasuwat, M.C.Berg, K.W. Buffinton, "Extended Kalman Filtering Applied to a Two-Axis Robotic Arm with Flexible Links", *The International Journal of Robotics Research*,Vol.19,No.3,pp.254-270, March 2000.

[26] C. Li, H. Li, Y. Tong, "Analysis of a Novel Three-Dimensional Chaotic System", *Optik Journal*, April 2012.

[27] S. Vaidyanathan, C.K. Volos, I.M. Kyprianidis, I.N. Stouboulos, K. Rajagopal, P. Alexander, "A Seven-Term Novel 3-D Chaotic System with Three Quadratic Nonlinearities and its LABVIEW Implementation", *Latest Trends on Systems*, Vol.1, pp.117-122.

[28] H. Joshi, N. Paulose, "Discrete Time Model Predictive Control Approach for Inverted Pendulum with Input Constraints", *IJEE International Journal of Electrical and Electronics Engineering*,Vol.3,Issue 1, pp.105-110, 2013.

[29] "Nonlinear Control System, Lecture-11, Pendulum on a Cart", Course Notes, Division of Electrical Engineering, NPTEL, pp.31-33.

[30] S. Lynch, "MATLAB Programming for Engineers", School of Computing, Mathematics and Digital Technology, Manchester Metropolitan University.

[31] J. Hannah, R.C. Stephens, "Mechanics of Machines - Advanced Theory and Examples", *2nd edition*, 1997.

[32] J.L. Meriam, L.G. Kraige, "Engineering Mechanics Dynamics", *John Wiley & Sons, Inc., 4th edition*, 1998.

[33] "System Identification, Estimation and Learning", MIT, Course Work: *Lecture Notes No.8, Chapter 4, Extended Kalman Filter*,pp.1-7 March 2006.

[34] J.H. Park, K.T. Chang, N. Kazantzis, A.G. Parlos, "Time-Discretization of Non-Affine Nonlinear System with Delayed Input using Taylor-Series", *KSME International Journal*, Vol.18, no.8, pp.1297-1305, 2004.

[35] G.H. Patel, A.K. Samantary, "Fault-tolerant Control of a Compliant Legged Quadruped Robot for Free Swinging Failure", *Journal of Systems and Control Engineering*, November 2017.

[36] J-J.E. Slotine, W. Li, "Applied Nonlinear Control", *Prentice Hall, Inc.*, 1991.

[37] A. Kaw, "Runge-Kutta 2^{nd} Order Method for Ordinary Differential Equations", Holistic Numerical Methods Institute, University of South Florida.

[38] T. Nath, A. Mathis, A-C. Chen, A. Patel, M. Bethge, M.W. Mathis, "Using DeepLabCut for 3D Markerless Pose Estimation across Species and Behaviours", *http://dx.doi.org/10.1101/476531*, November 2018.

[39] "Lyapunov Exponent", Source URL: *http://systems-sciences.uni-graz.at /etextbook/sw2/lyapunov.html*, November 2018.

[40] E. Vlasbom, "Nonlinear State Estimation for a Bipedal Robot", MSc Thesis Project, Delft Center for Systems and Control, Delft University of Technology, June 2014.

[41] L. Cucu, "Applying Kalman Filtering on a Quadruped Robot", EPFL, Biorobotics Laboratory, Federal Polytech Institute, Lausanne, Switzerland.

[42] X. Xinjilefu, C.G. Atkeson, "State Estimation of a Walking Humanoid Robot", The Robotics Institute, Carnegie Mellon University, Pittsburgh.

[43] X. Xinjilefu, "State Estimation for Humanoid Robots", PhD Thesis Document, The Robotics Institute, Carnegie Mellon University, Pittsburgh.

[44] CW. Sul, SK. Jung, K. Wohn, "Synthesis of Human Motion using Kalman Filter", Department of Computer Science, KAIST & Department of Computer Engineering, Kyungpook National University.

[45] L. Aguiar, M.R.O.A. Maximo, T. Yoneyama, S. Pinto, "Kalman Filtering for Differential Drive Robots Tracking", SBAI, Porto Alegre, 4 October 2017.

[46] M. Burke, J. Lasenby, "Single Camera Pose Estimation using Bayesian Filtering and Kinect Motion Priors", Department of Engineering, University of Cambridge, UK, 18 June 2014.

[47] K. Jankowski, "Dynamics of Double Pendulum with Parametric Vertical Excitation", MSc(Eng) Project, Department of Mechanical Engineering and Applied Computer Science, Technical University of Lodz, 11 July 2017.

[48] AC. Chen, "3D Markerless Body Motion Capture for the Cheetah", BSc(Eng) Thesis Project, Department of Electrical Engineering, University of Cape Town, 22 October 2018.

[49] J. Cushway, "Whole-body Motion Tracking of the Cheetah using Vision Data", BSc(Eng) Thesis Project, Department of Electrical Engineering, University of Cape Town, 13 November 2017.

[50] R. Nilsson, "Inverse Kinematics", Master's Thesis in Engineering, Department of Computer Science and Electrical Engineering, Luleǎ University of Technology, 2009.

[51] S. Schaal, "Jacobian Methods for Inverse Kinematics and Planning" (slides), Max Planck Institute, University of Southern California & E.Todorov, "What makes control hard" (lecture notes), Department of Applied Mathematics, Computer Science and Engineering, University of Washington.

[52] J.Ratajczak, "Design of Inverse Kinematics Algorithms:extended Jacobian approximation of the dynamically consistent Jacobian inverse",*Archives of Control Sciences , Vol 25(LXI),No.1, pp.35-50*, 2015

[53] C. Sprunk, B. Lau, W. Burgard, "Improved Non-linear Spline Fitting for Teaching Trajectories to Mobile Robots", *IEEE International Conference on Robotics and Automation*, May 2012.

[54] H.H Asada, "Introduction to Robotics", Department of Mechanical Engineering, MIT, Cambridge, USA.

[55] K. Tchoń, "Optimal Extended Jacobian Inverse Kinematics Algorithms for Robotic Manipulators", *IEEE Transactions on Robotics*, January 2009.

[56] H. Simas, D. Martins, A. Dias, "Extended Jacobian for Redundant Robots obtained from the Kinematics Constraints", University of Santa Catarina , Published by Researchgate: *publication no.260354295*, January 2012.

[57] N.P.G. Salau, J.O. Trierweiler, A.R. Secchi, "State Estimation of Chemical Engineering Systems tending to Multiple Solutions", *Brazilian Journal of Chemical Engineering, Vol.31, No.3*, pp.771-785, July-September 2014.

[58] R. Hauser, "Kalman Filtering with Equality and Inequality Constraints", *Report No.07/18*, Oxford University Computing Laboratory, Numerical Analysis Group, September 2017.

[59] X. Jian, L. Zushu, "Dynamic Model and Motion Control Analysis of Three-link Gymnastics Robot on Horizontal Bar", *IEEE International Conference on Robotics, Intelligent Systems and Signal Processing*, October, 2003.

[60] M. Kelemen, I. Virgala, T. Lipták, L. Miková, F. Filakovský, V. Bulej, "A Novel Approach for a Inverse Kinematics Solution of a Redundant Manipulator", MDPI, Applied Sciences, 12 November 2018.

[61] P. Boscariol, D. Richiedei, "Trajectory Design for Energy Savings in Redundant Robotic Cells", MDPI, Robotics, 20 February 2019.

[62] M. Neunert, F. Farshidian, J. Buchli, "Efficient Whole-Body Trajectory Optimization Using Contact Constraint Relaxation", Agile & Dexterous Robotics Laboratory, ETH, Zurich.

[63] S.R. Buss, "Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares Method", Department of Mathematics, University of California, San Diego, 7 October 2009.

[64] Alain Muraro, Christine Chevallereau, Yannick Aoustin. Optimal trajectories for a Quadruped Robot with Trot, Amble, Curvet Gaits for Two Energetic Criteria. Multibody System Dynamic, 2003, 9 (1), pp.39-62. hal-00794871

[65] S.R. Buss, J-S. Kim, "Selectively Damped Least Squares for Inverse Kinematics", University of California, San Diego, 25 October 2004.

[66] C. Yang, H. Ma, M.Fu, "Advanced Technologies in Modern Robotic Applications", Science Press and Springer Science+Business Singapore, 2016, pp.27-48.

[67] Matlab Version R2018a, Documentation, Keywords: Nonlinear Least Squares, lsqnonlin, Levenberg-Marquardt Method, Output Arguments.

Appendix A

Joint Space Trajectories

The joint space trajectories as estimated by the inverse kinematics models in θ and ϕ phase states where applicable.

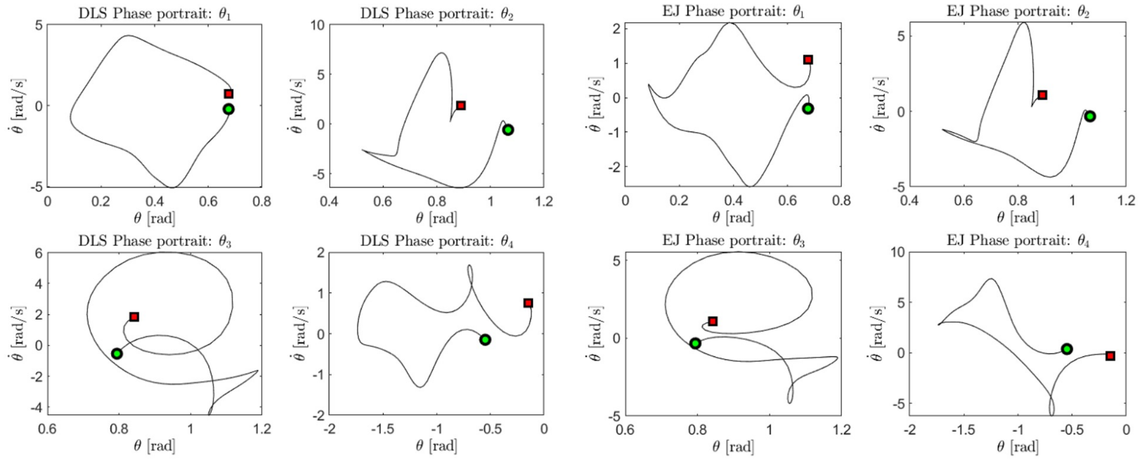


Figure 76: Joint trajectory estimates in θ phase space by the gradient projection method.

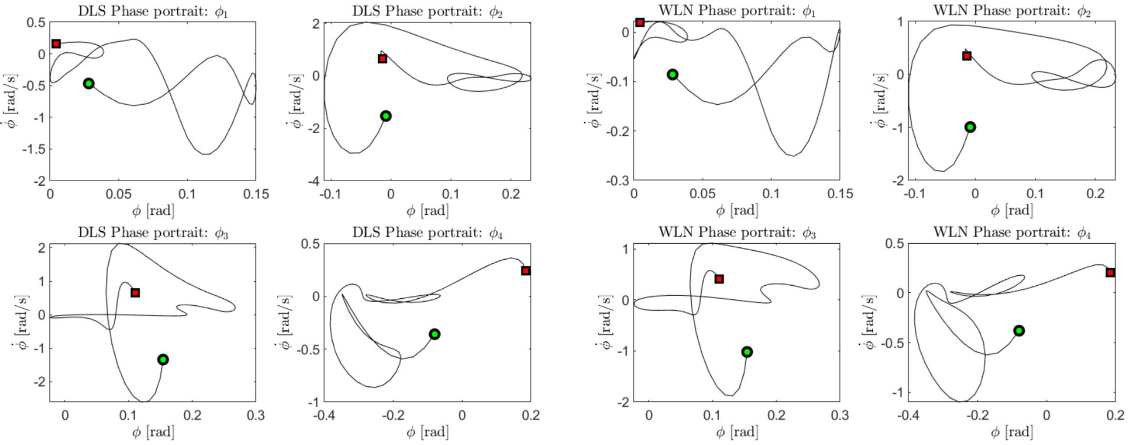


Figure 77: Joint trajectory estimates in ϕ phase space by the damped least squares method.

Appendix B

Optimisation Functions

Extended Jacobian optimisation functions $g(\boldsymbol{\theta}) = \mathrm{g}(\theta_9, \theta_1, \theta_2, \theta_3, \theta_4)$ and results:
(i) Optimisation of $g(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ in the nullspace of the Jacobian matrix using: relative angle velocities.

$g = (\theta_2 - \theta_1)^2$: convergence to target achieved.
$g = (\theta_4 - \theta_3)^2$: matrix singular !
$g = (\theta_3 - \theta_2)^2$: matrix singular !
$g = (\theta_3 - \theta_1)^2$: chaotic trajectory !
$g = (\theta_3 - \theta_9)^2 + (\theta_2 - \theta_9)^2$: chaotic trajectory !
$g = (\theta_3 - \theta_1)^2 + (\theta_2 - \theta_1)^2$: chaotic trajectory !
$g = (\theta_4 - \theta_1)^2 + (\theta_3 - \theta_1)^2 + (\theta_2 - \theta_1)^2$: chaotic trajectory !

(ii) Optimisation of $g(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ in the null space of the Jacobian matrix using : rotational energies.

$g = (r_5\theta_9)^2 + (r_5\theta_1)^2 + (r_5\theta_2)^2 + (r_5\theta_3)^2 + (r_5\theta_4)^2$: convergence to target achieved.

(iii) Optimisation of $g(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ in the null space of the Jacobian matrix using : augmenting harmnonic function, $g = x_2(ax_1+b)\sqrt{1+x_1^2}$. $x_1 = (\theta_2-\theta_3), x_2 = (\theta_4 - \theta_3), a = 1, b = 0$: optimal convergence to target achieved.

Appendix C

Non Convergent Trajectories

Unsuitable optimisation functions in the extended Jacobian method resulted in trajectories not converging to the target point. Some of these results are shown here together with the applied g($\boldsymbol{\theta}$) function.

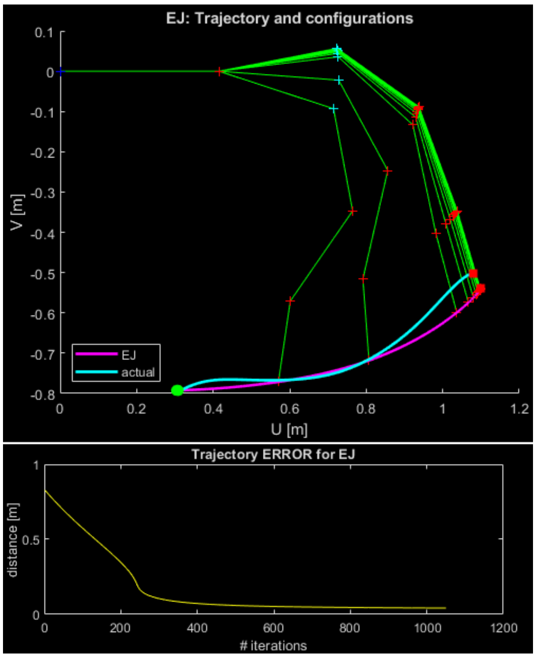


Figure 78: Non convergent trajectory resulting from an unsuitable cost function: $g = (\theta_2 - \theta_1)^2(\theta_1 - \theta_3)^2 + (\theta_3 - \theta_4)^2(\theta_4 - \theta_1)^2$. The graphic below it shows the distance between the current and target points confirming failure to converge.

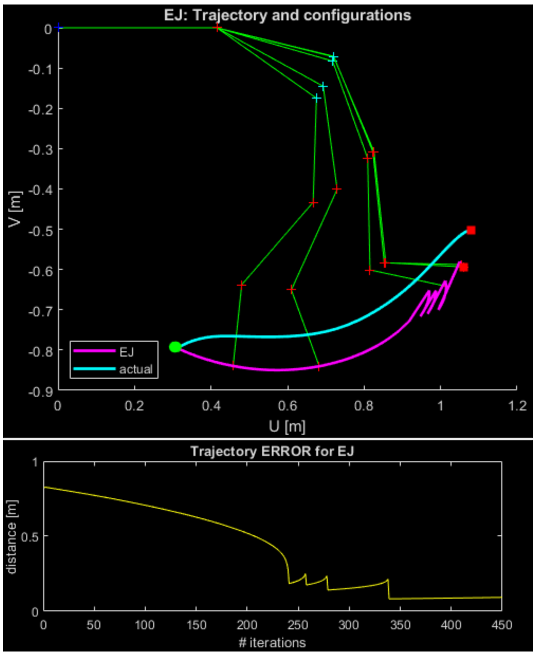


Figure 79: Chaotic, non convergent trajectory resulting from an unsuitable cost function: $g = (\theta_2 - \theta_1)(\theta_2 - \theta_3) + (\theta_3 - \theta_4)(\theta_4 - \theta_1)$. The graphic below it shows the distance between the current and target points confirming failure to converge.

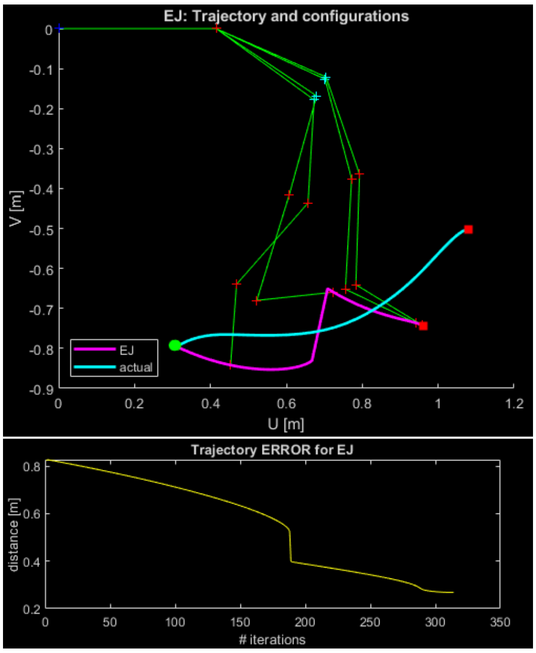


Figure 80: Discontinuous, non convergent trajectory resulting from an unsuitable cost function: $g = (\theta_3 - \theta_2)(\theta_2 - \theta_3) + (\theta_3 - \theta_4)(\theta_4 - \theta_1)$. The graphic below it shows the distance between the current and target points confirming failure to converge.

Appendix D

Vector Fields

The vector fields for all the links as determined by the optimisation. The vector reference level is a scaling of the magnitudes so that comparatives may be established over the entire time interval.

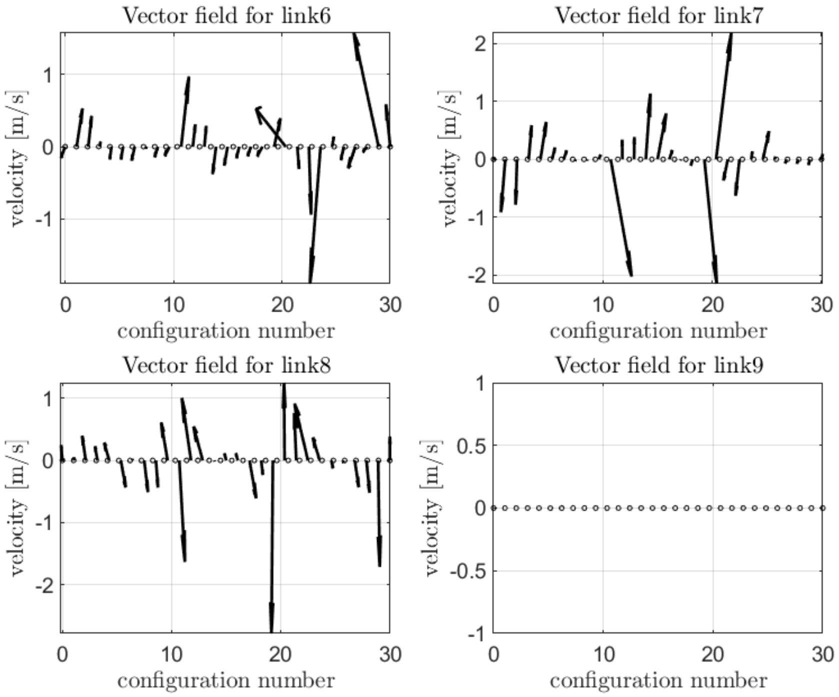


Figure 81: Vector Field I.

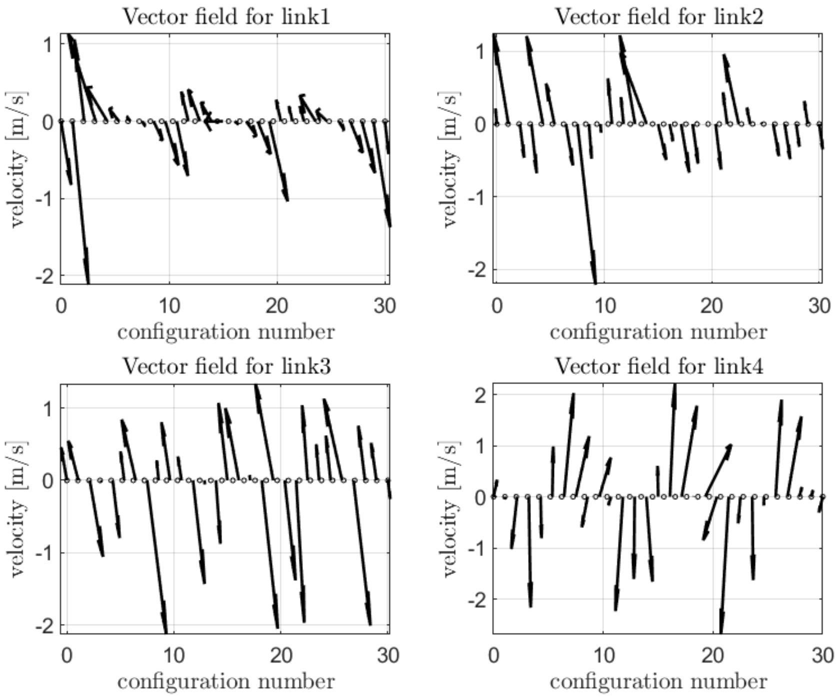


Figure 82: Vector Field II.

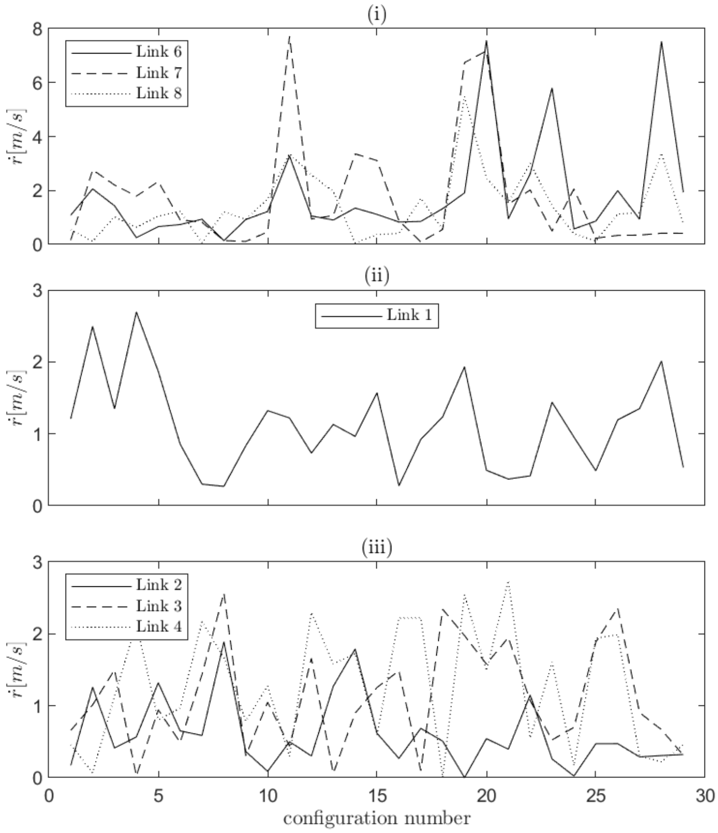


Figure 83: Polar velocity magnitudes for (i) front leg, (ii) spine and (iii) back leg links for all of the configurations.

Appendix E

Quadruped Configurations

The full results of all of the image and sagittal (UV) plane configurations of the quadruped as determined by the non-linear least squares optimisation function. The Robot Configuration (RC) number is shown together with each of their respective image and UV plane configurations.

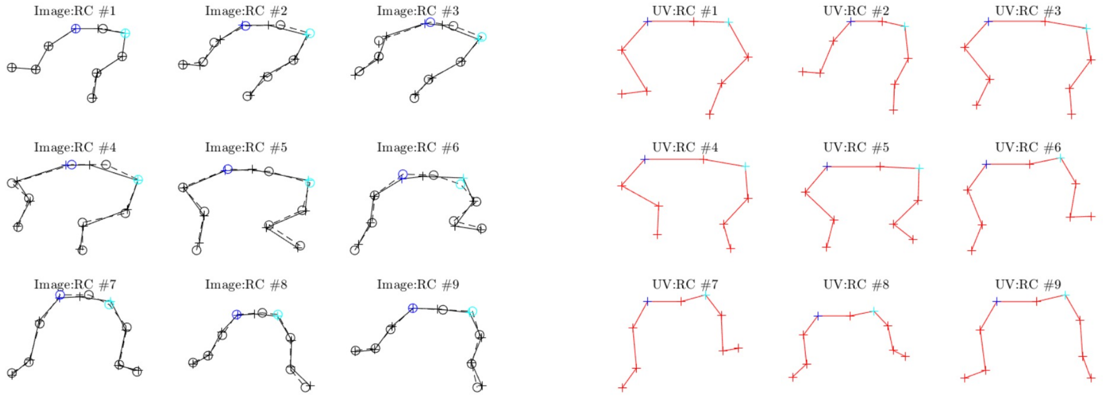


Figure 84: The first set of 9 configurations (out of 30). At left are the composite image configurations showing the truth (black dashed line) and reprojected (black solid line) images and at right the computed quadruped configurations (red) in the sagittal plane.

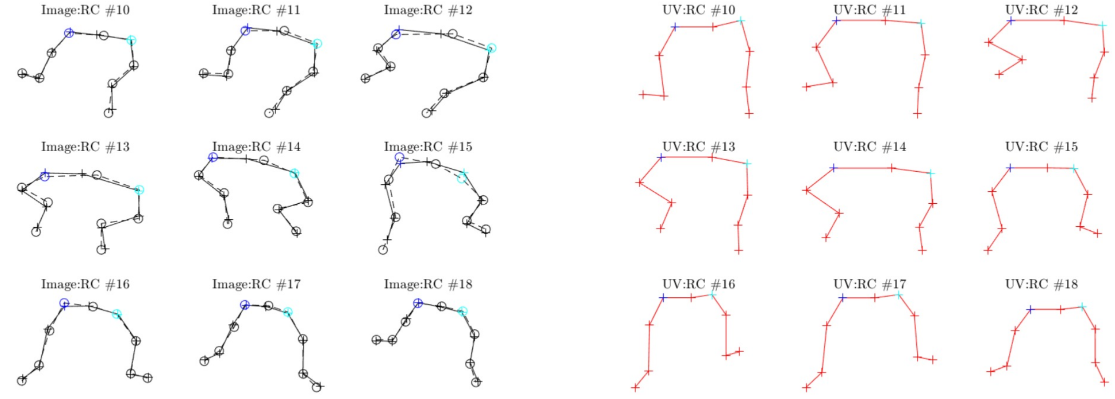


Figure 85: The second set of 9 configurations (out of 30). At left are the composite image configurations showing the truth (black dashed line) and reprojected (black solid line) images and at right the computed quadruped configurations (red) in the sagittal plane.

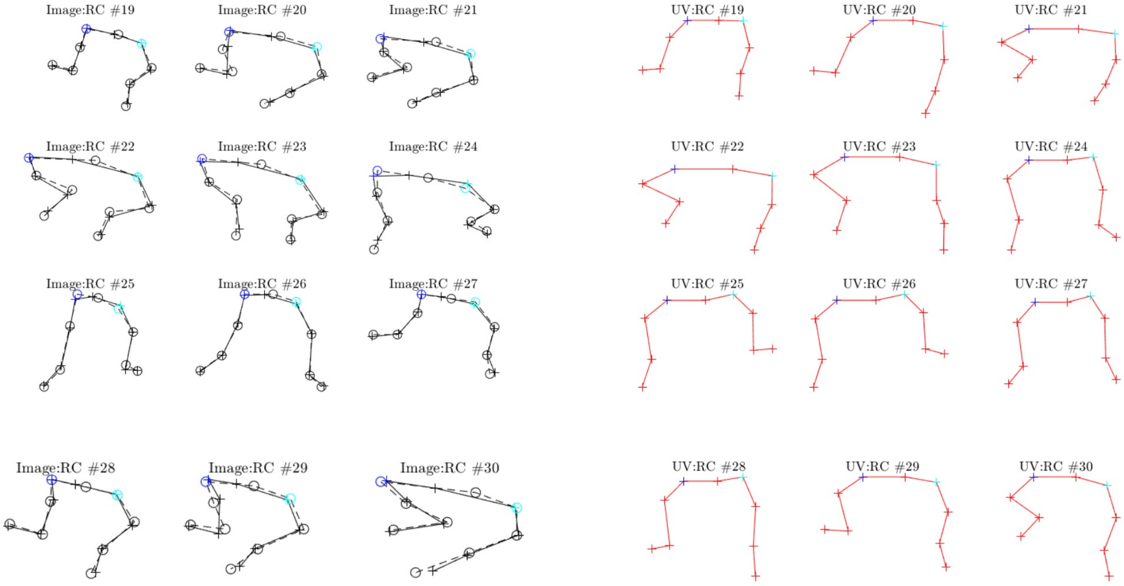
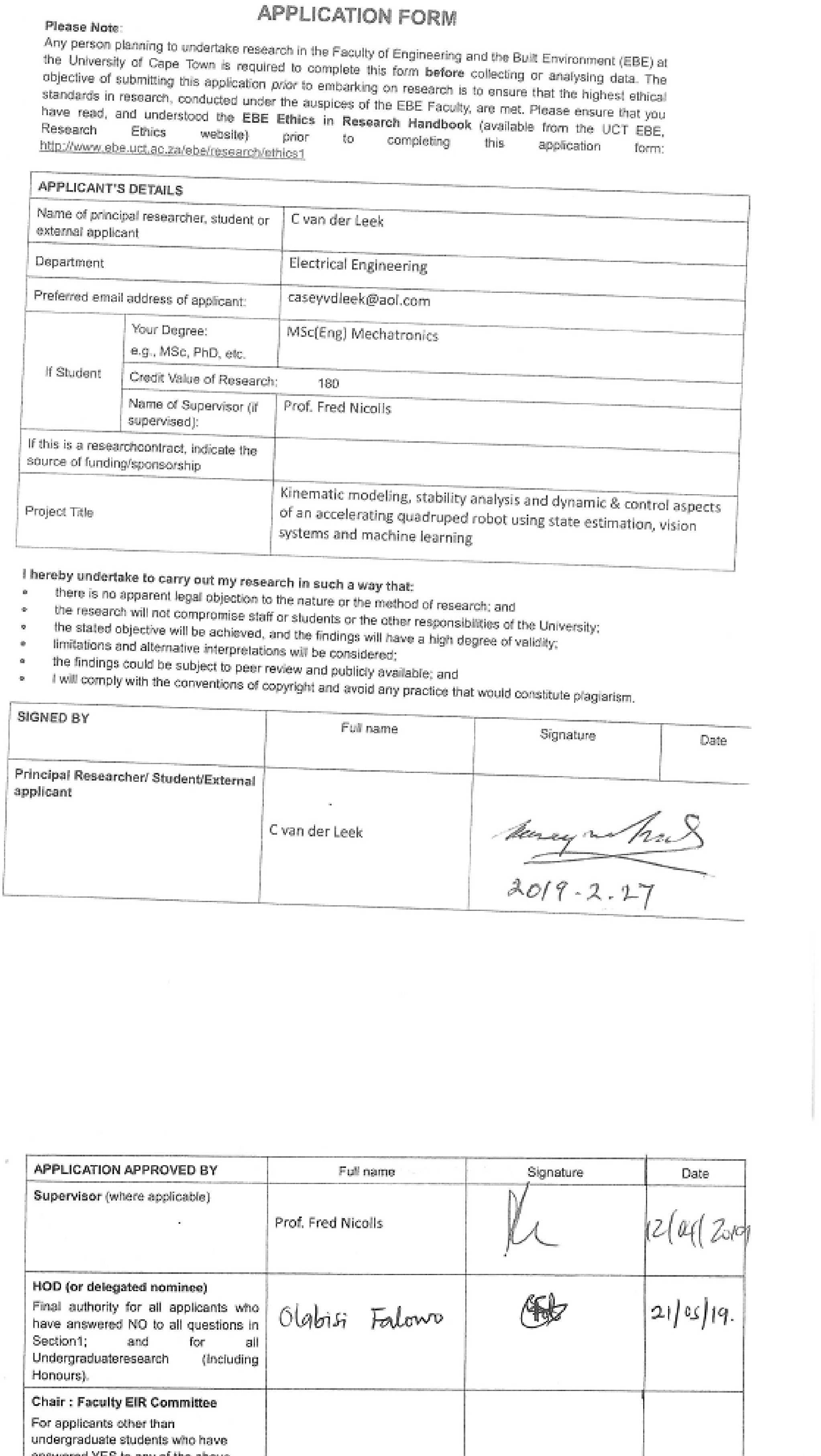


Figure 86: The last set of 12 configurations (out of 30). At left are the composite image configurations showing the truth (black dashed line) and reprojected (black solid line) images and at right the computed quadruped configurations (red) in the sagittal plane.



APPLICATION FORM

Please Note:

Any person planning to undertake research in the Faculty of Engineering and the Built Environment (EBE) at the University of Cape Town is required to complete this form **before** collecting or analysing data. The objective of submitting this application *prior* to embarking on research is to ensure that the highest ethical standards in research, conducted under the auspices of the EBE Faculty, are met. Please ensure that you have read, and understood the **EBE Ethics in Research Handbook** (available from the UCT EBE, Research Ethics website) prior to completing this application form: http://www.ebe.uct.ac.za/ebe/research/ethics1

APPLICANT'S DETAILS		
Name of principal researcher, student or external applicant		C van der Leek
Department		Electrical Engineering
Preferred email address of applicant:		caseyvdleek@aol.com
If Student	Your Degree: e.g., MSc, PhD, etc.	MSc(Eng) Mechatronics
	Credit Value of Research:	180
	Name of Supervisor (if supervised):	Prof. Fred Nicolls
If this is a researchcontract, indicate the source of funding/sponsorship		
Project Title		Kinematic modeling, stability analysis and dynamic & control aspects of an accelerating quadruped robot using state estimation, vision systems and machine learning

I hereby undertake to carry out my research in such a way that:

- there is no apparent legal objection to the nature or the method of research; and
- the research will not compromise staff or students or the other responsibilities of the University;
- the stated objective will be achieved, and the findings will have a high degree of validity;
- limitations and alternative interpretations will be considered;
- the findings could be subject to peer review and publicly available; and
- I will comply with the conventions of copyright and avoid any practice that would constitute plagiarism.

SIGNED BY	Full name	Signature	Date
Principal Researcher/ Student/External applicant	C van der Leek	2019-2-27	

APPLICATION APPROVED BY	Full name	Signature	Date
Supervisor (where applicable)	Prof. Fred Nicolls		12/04/2019
HOD (or delegated nominee) Final authority for all applicants who have answered NO to all questions in Section1; and for all Undergraduateresearch (Including Honours).	Olabisi Falowo		21/05/19.
Chair : Faculty EIR Committee For applicants other than undergraduate students who have			