Jumping Kinematics of the Chacma Baboon

Presented by:

Matthew Terblanche

Prepared for:

Dr. Amir Patel

November 7, 2021

Submitted to the Department of Electrical Engineering at the University of Cape Town in partial fulfilment of the academic requirements for the degree of Bsc Eng in Mechatronics.

Abstract

Very little is known about the 3D kinematics of Chacma baboons. This is due to previous studies only making use of GPS-IMU collars to track their movements. This project studies a manner of developing the 3D kinematics of baboons in a cost-effective and unobtrusive manner. This enables the 3D kinematic study of animals to become accessible to a much wider audience and accelerate the development of bio-inspired robots.

Acknowledgments

I would like to thank Dr. Amir Patel for all the help and guidance that he provided throughout this project. This was a challenging but very interesting project to be a part of.

Secondly, I would like to thank Professor Justin O' Riain, and Joselyn and Fanus from NCC Environmental Services for their help in capturing the video footage of the baboons.

I would also like to thank Daniel Joska, my mentor for this project, for his advice and timely responses to my questions.

Lastly, I would like to thank my family for their support throughout my degree. This has been a challenging couple of years and I wouldn't have been able to do it without them.

Plagiarism Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.

2. I have the IEEE convention for citation and referencing. Each contribution to, and quotation in, this final year project report from the work(s) of other people, has been attributed and has been cited and referenced.

3. This final year project report is my own work.

4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as there own work or part thereof.

Matthew Terblanche

November 7, 2021

Word count: 7024

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Bio-inspired robots are an exponentially growing field of study in robotics. Studying the agility and movements of four-legged animals, in particular, can aid in the development of some amazing robots.

To study these animals, an accurate and detailed way of tracking their movements is needed. There are many ways to do this but most involve very expensive equipment and are usually constrained to being done in a controlled environment like a laboratory. This also usually involves having to capture the animals and keep them in a cage.

This project presents a low-budget method of studying the 3D kinematics of a Chacma baboon in the wild. The method shown in this project is beneficial as it can be done in a lost-cost manner which makes it accessible to a wider range of people. It can be done in the wild without the need to cage or interact with the animals, ensuring the safety of the animal and the person studying them.

1.2 Motivation

Chacma baboons are very agile and athletic animals. They are excellent at moving around in difficult environments by running, jumping, and climbing. This makes them great candidates to study when developing bio-inspired robots. The goal of this project is to gather footage of these baboons in the wild and process the data by doing pose estimation and 3D reconstruction.

1.3 Problem Statement

Bio-inspired robotics is an exponentially growing field due to the advantages of mimicking biological systems [1]. To develop these robots, the animal's movements need to be studied thoroughly.

Very little is known about the whole-body kinematics of Chacma baboons due to previous studies only using GPS-IMU trackers. These will need to be worn by the baboon, usually in the form of a collar. GPS inertial measurement units, when worn as a collar, can track the position, velocity, and acceleration of the baboons as a whole but cannot track the detailed kinematics of their limbs [2].

There are many high-end and expensive devices and software that can aid in pose estimation and 3D reconstruction but are usually not accessible to most people.

To study animals in the wild in an ethical and safe way without disturbing them, a non-obtrusive manner for data collection is needed. Markerless motion tracking is very useful to achieve this.

1.4 Objectives

This project aims to develop a model of the 3D kinematics of the Chacma baboon in the wild. The main objectives to achieve this goal are to:

- Discuss the method of video and data capture that is needed for 3D markerless motion capture in the wild
- Investigate the theory behind the data capture system and post-processing of the data
- Develop a plan to build and test the system and 3D kinematics
- Document the implementation of data capture and processing
- Evaluate the accuracy of the system using ground truth data

1.5 Project Contributions

The main contributions of this project are as follows:

- A low-cost stereo camera rig system is designed and built
- Markerless pose estimation of the Chacma baboon is implemented
- Existing 3D reconstruction techniques are analyzed with real-life data

1.6 Terms of Reference

The requirements and functionality of the different objectives of the project are discussed in list form below:

1.6.1 Stereo Camera Rig

A stereo camera rig is needed to capture stereo footage of baboons in the wild. The functionality of the camera rig to meet the requirements are as follows :

1. Build a structure to hold the cameras in a fixed position relative to each other on a horizontal rotating bar;
2. Allow rotation of the cameras about the vertical axis;
3. Log the angle of rotation of the cameras;
4. Synchronize the cameras and data;
5. Calibrate the intrinsic and extrinsic parameters of the cameras

To test the timing of the data logging and the accuracy of the camera and data synchronization, a MATLAB script was created to plot the video from the left camera, video from the right camera, and the on\off state of an LED.

1.6.2 Markerless Pose Estimation

Markerless pose estimation is done using DeepLabCut, an open-source markerless pose estimation toolbox. To successfully implement pose estimation the following outcomes need to be met:

1. Extract frames from various videos of the baboons;
2. Label the body parts of the baboons;
3. Train the neural network;
4. Evaluate the robustness of the DeepLabCut model;

5. Analyze videos and evaluate tracking accuracy

To test, DeepLabCut provides functions to assess the accuracy of the trained model and the root-mean-square error of the tracked labels.

1.6.3 3D Reconstruction

3D reconstruction needs to be done to acquire a 3D model of the baboon. To develop a 3D model from the 2D videos the following needs to be done:

1. A ground truth model or simulation needs to be created by manually labeling frames

2. 2D pose estimation data containing 2D pixel coordinates from DeepLabCut needs to be extracted for processing

3. 2D data must be processed using triangulation to get 3D points

4. 2D data must be processed using sparse bundle adjustment (SBA) to get 3D points

5. To test the accuracy of the 3D reconstruction methods, the triangulation and SBA 3D reconstruction is compared to the ground truth reconstruction the find the percentage of correct keypoints (PCK).

1.7 Scope and Limitations

The outcome of this project is mainly to develop the 3D kinematics of a baboon in the wild with no prior data. The detailed scope is as follows:

- A low cost stereo camera rig would need to be built;
- The camera system must have the ability to rotate about the vertical axis;
- Only two GoPro Hero 5 Session cameras can be used;
- Markerless pose estimation is done using the open-source toolbox, DeepLabCut;
- Due to time constraints, 3D reconstruction is limited to triangulation and sparse bundle adjustment;
- Accuracy of the 3D reconstruction methods is evaluated using a ground truth simulation;
- Videos of wild baboons needs to be recorded whilst keeping a safe distance;
- Recording of baboons was restricted to availability of the NCC Environmental Services employees and the current location of the baboon troop;

1.8 Project Outline

The remainder of this thesis is organized as follows:

Chapter 2, Literature Review: Background and literature review

Chapter 3, Methodology: Project methodology

Chapter 4, System Design: System design and development

Chapter 5, Results and Discussion: Experimental results and discussion

Chapter 6, Conclusions: Conclusions and future work

Chapter 2

Literature Review

The following chapter discusses the current work in computer based motion capture and pose estimation. This is followed by details on stereo camera configuration and methods for 3D rectification. The chapter finishes with a summary of the main concepts discussed and main takeaways for this project.

2.1 3D Computer Vision Motion Capture

There are various methods for motion capture. These include optical-passive optical-active, video, and inertial motion tracking techniques [3].

These can be grouped into marker-based tracking, markerless tracking, and inertial tracking. Optical-active and optical-passive are marker-based tracking methods that make use of cameras and markers placed on the object of study. This enables highly accurate tracking results. Marker-based tracking is usually constrained to a controlled laboratory environment. As a result it is not possible to track animals in the wild using marker-based tracking, as the markers cannot be placed on a wild animal.

Markerless tracking is done using only video footage i.e there are no markers on

the object that is being tracked. The caveat of this method of tracking is that it is often not as accurate as marker-based tracking. The accuracy is highly dependent on the cameras, camera setup, and software used to process the video footage. Advanced image processing computer vision tools are used to track positions of the object across frames [4]. Markerless tracking is therefore a very good candidate for tracking animals in the wild.

Inertial motion tracking makes use of inertial measurement units (IMUs). These small devices are worn on the body of the object and consist of acceleroemeters and gyroscopes to track the linear displacement and angular orientation of the object. Previous studies of baboons mainly made use of GPS-IMU collars [5]. These collars are used to quantify major behavioural states of the baboon such as walking and running. This study is restricted to the general movement of the baboon as a whole. This leaves a lack of knowledge on the whole body kinematics of the baboon.

There are many commercially available 3D computer vision based motion tracking systems such as Vicon. Vicon contains a full motion capture package of high-end hardware and software that makes use of high precision marker-based tracking. This is generally not accessible to the average user therefore other low-end systems such as the Microsoft Kinect camera are used for motion capture. There is a trade off of accuracy when using a cheaper markerless solution as shown in Figure 2.1 below:

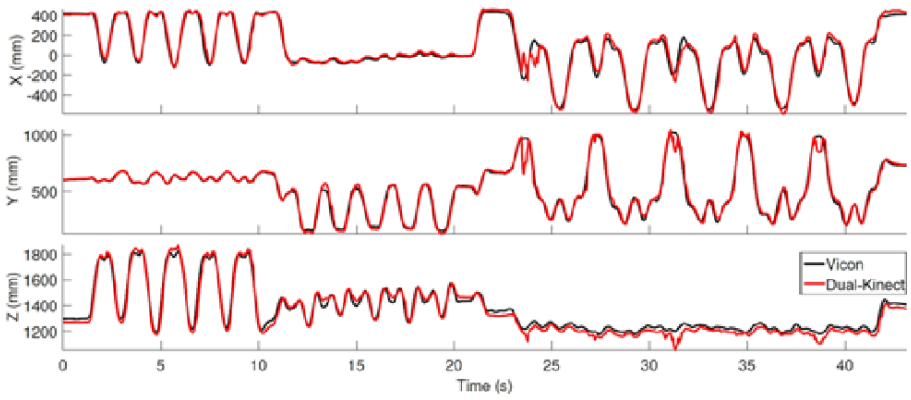


Figure 2.1: Joint trajectory of left wrist from Dual-Kinect cameras compared to Vicon system [6]

This minor error in accuracy is usually accepted due to the benefits of using a low-cost, portable, and unobtrusive motion capture system. The Kinect is made only for use indoors there a set of GoPro cameras will be used for recording video footage for this project as a low-cost means of video capture.

2.2 Neural Networks and Pose Estimation

Artificial Neural Networks (ANNs) are computational processing systems that work similar to how the brain operates. ANNs are designed to learn from an input in order to optimise the final output.

Convolutional Neural Networks work in a similar manner to ANNs. The main difference between the two is that CNNs are mostly used in computer vision such as pattern recognition within images [7].

Pose estimation is a computer vision task that is done to track the position and motion of an object over time. Pose estimation tracks specific keypoints. IN the case of this report, these keypoints are animal joint labels like wrist, ankle, and knee.

There are a vast amount of open source human deep learning methods available

with huge datasets such as OpenPose [8], AlphaPose [9], and DeepCut [10].

Unfortunately there are much fewer animal pose estimation data sets, especially for baboons. There are a lot of open source deep learning pose estimation toolkits that have been created over the last couple of years designed specifically for animal tracking such as DeepPoseKit [11], LEAP [12], AniPose [13], and DeepLabCut [14].

The general user work flow for using DeepLabCut is shown in Figure 2.2 below:

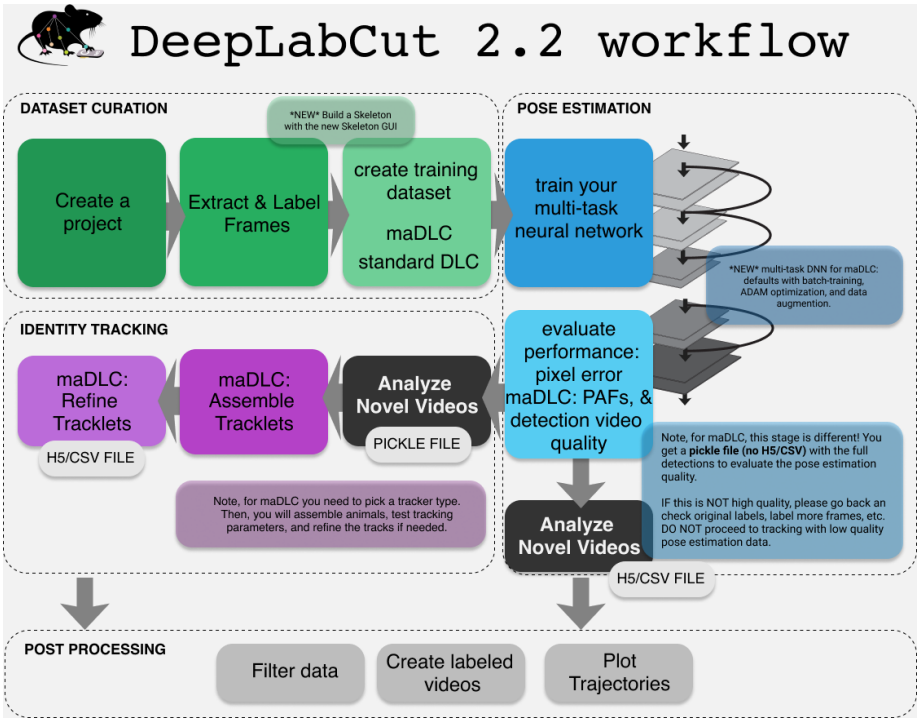


Figure 2.2: User workflow for the latest multi animal DeepLabCut (maDLC) [14]

These will hopefully result in the animal tracking datasets to grow exponentially over the coming years.

2.3 Stereo Camera

To obtain 3D coordinates, multiple view points are needed of the same scene. This can be achieved with a stereo camera setup. A stereo camera setup involves two or more cameras that have the same object of interest in their field of view (FOV). The characteristics of these camera models and the calibration of a stereo camera system is discussed below.

2.3.1 Pinhole Camera Model

A pinhole camera model is a simple lensless model with a single small aperture. As shown in Figure 2.3 below, when light rays pass through the aperture, they project an inverted image on the opposite side of the camera [15].

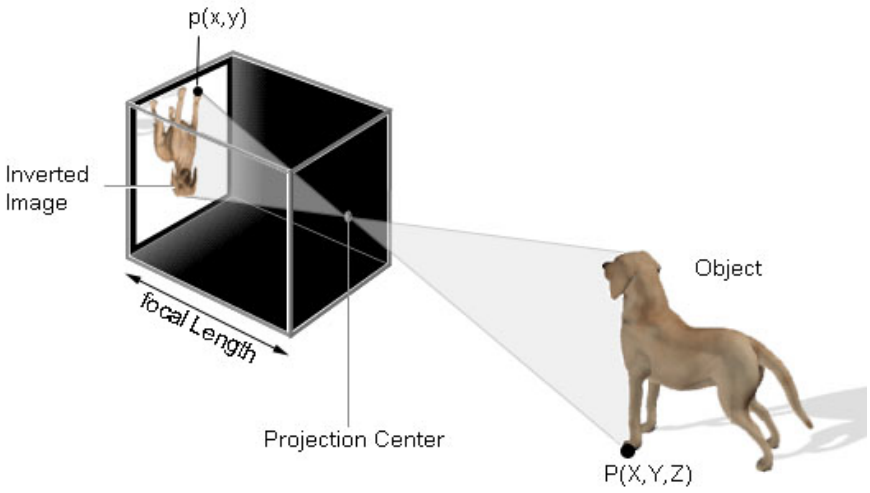


Figure 2.3: Visualization of a pinhole camera model [16]

The parameters of the pinhole camera are represented by a 4x3 camera matrix. The camera matrix maps the 3-D world scene onto the image plane using Equation 2.1.

$$\mathrm{s\ p} = \mathrm{A} \begin{bmatrix} R \\ t \end{bmatrix} \mathrm{P_w} \tag{2.1}$$

Where $\mathrm{P_w}$ is a 3D point in world coordinates, p is the 2D pixel in the image plane, A is the camera intrinsic matrix, R is the rotation and t is translation of the extrinsics.

The origin of the camera's coordinate system is at the optical centre with the x-axis pointing to the right, the y-axis pointing down, and the z-axis pointing away from the front of the camera.

The camera intrinsic matrix A is shown in Equation 2.2.

$$\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

Where $[\mathrm{c_x}\ \mathrm{c_y}]$ is the optical centre or principle point measured in pixels and $\mathrm{f_x}$ and $\mathrm{f_y}$ is the focal length in pixels.

Equation 2.1 assumes a no lens model and therefore does not take lens distortion into account.

When taking the radial and tangential distortion into account Equation 2.1 is extended as:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f_x x'' + c_x \\ f_y y'' + c_y \end{bmatrix} \tag{2.3}$$

where

$$\begin{bmatrix} x'' \\ y'' \end{bmatrix} = \begin{bmatrix} x' \frac{1+k_1 r^2 + k_2 r^4 + k_3 r^6}{1+k_4 r^2 + k_5 r^4 + k_6 r^6} + 2p_1 x' y' + p_2(r^2 + 2x'^2) + s_1 r^2 + s_2 r^4 \\ y' \frac{1+k_1 r^2 + k_2 r^4 + k_3 r^6}{1+k_4 r^2 + k_5 r^4 + k_6 r^6} + p_1(r^2 + 2y'^2) + 2p_2 x' y' + s_3 r^2 + s_4 r^4 \end{bmatrix} \tag{2.4}$$

with

$$r^2 = x'^2 + y'^2 \tag{2.5}$$

and

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} X_c/Z_c \\ Y_c/Z_c \end{bmatrix}, \tag{2.6}$$

if

$$Z_c \neq 0 \tag{2.7}$$

The radial coefficients k_1, k_2, k_3, k_4, k_5, and k_6 are the radial distortion parameters. The tangential distortion coefficients are p_1 and p_2. Whilst s_1, s_2, s_3, and s_4 are the thin prism distortion coefficients [15].

2.3.2 Fisheye Camera Model

The fisheye camera model uses an omnidirectional camera model. World points are transformed to camera coordinates using the extrinsic parameters and the intrinsic paramters are used to map the camera coordinates into the image plane [17]. A fisheye camera model is shown in Figure 2.4.

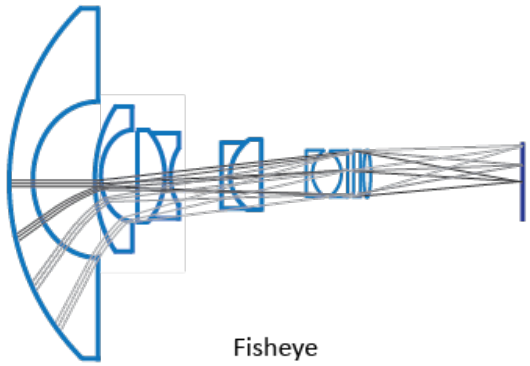


Figure 2.4: Visualization of a fisheye camera model [18]

The transformation of world points to camera points is shown in Equation 2.8

$$Xc = RX + T \tag{2.8}$$

If

$$\begin{aligned} x &= Xc_1 \\ y &= Xc_2 \\ z &= Xc_3, \end{aligned} \tag{2.9}$$

then the pinhole projection coordinates of a 3D point is [a;b] where

$$\begin{aligned} a = x/z \; and \; b = y/z \\ r^2 = a^2 + b^2 \\ \theta = atan(r) \end{aligned} \tag{2.10}$$

The fisheye distortion can be expressed as

$$\theta_d = \theta(1 + k_1\theta^2 + k_2\theta^4 + k_3\theta^6 + k_4\theta^8) \tag{2.11}$$

The fisheye distorted point coordinates are [x';y'] where

$$\begin{aligned} x' &= (\theta_d/r)a \\ y' &= (\theta_d/r)b \end{aligned} \tag{2.12}$$

Therefore the final pixel coordinates vector is [u;v] where [19]

$$\begin{aligned} u &= f_x(x' + \alpha y') + c_x \\ v &= f_y y' + c_y \end{aligned} \tag{2.13}$$

2.3.3 Calibration

Three-dimensional reference object-based camera calibration involves observing an object of known dimensions to calibrate the intrinsic and extrinsic parameters of a stereo camera system [15].

Using a checkerboard with known dimensions as shown in Figure 2.5, and the equations shown in Section 2.3.1 and 2.3.2 the unknown parameters of the camera system can be found. This is done using the known points in real world 3D coordinates and the 2D image pixel coordinates and solving for the unknown parameters. Images of the checkerboard need to be taken whilst the checkerboard is moved all around the FOV of the cameras. The orientation of the checkerboard should also be slightly altered during image capture to ensure a accurate and robust camera calibration.

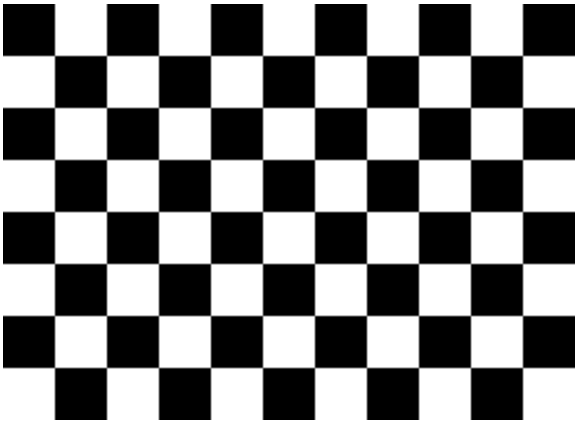


Figure 2.5: Black and white checkerboard used for camera calibration [20]:

There is widely available software to aid in successfuly completing camera calibration such as OpenCV [21] and the MATLAB Calibrator App [22].

2.4 3D Reconstruction

3D reconstruction is necessary to gather real world 3D information on objects recorded with a stereo camera system. Two methods of 3D reconstruction will be briefly discussed below.

2.4.1 Triangulation

Triangulation is the practice of finding the intersection of two rays in space from two images taken from calibrated cameras.

Due to the noise in the system and the rays not perfectly intersecting, the triangulation problem can be formulated as a least-squares minimization problem [24].

Using equations discussed in previous sections, the real world 3D point can be found when the 2D pixel location on the left and right image is known, and the intrinsic and extrinsic parameters of the stereo system is known.

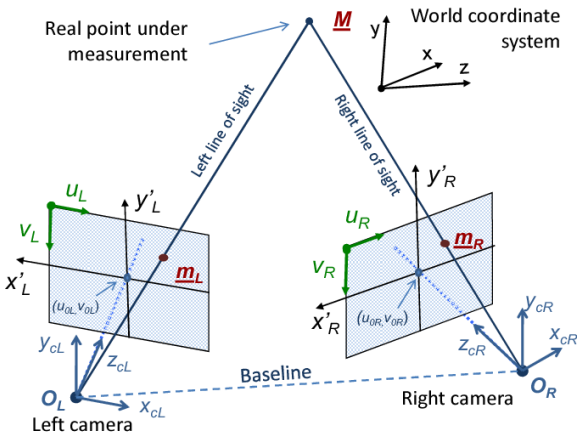


Figure 2.6: Coordinate systems of a stereo pair of cameras [23]:

Figure 2.6 shows how a 3D point is found using a stereo camera setup where the point is observable from both the left and right camera.

2.4.2 Sparse Bundle Adjustment

The goal of sparse bundle adjustment is to find the optimal projection matrices P_i and world coordinates X_j so that the summed square reprojection error is minimal i.e Equation 2.14 must be solved

$$\min_{P_i, X_j} \sum_{ij} d(P_i X_j, x_{ij})^2 \tag{2.14}$$

where,

P_i is the set of camera matrices, X_j is the set of world points, x_{ij} are the image coordinates, and d(x,y) is the Euclidean distance between image points x and y [25].

Sparse bundle adjustment can be used to refine the cameras matrices found when doing camera calibration.

2.5 Summary

From the literature review above, several key observations can be made. Markerless tracking is definitely the most suitable method of motion capture for tracking baboons in the wild. There is no need to interact with the baboons by placing markers on them and the tracking can be done outside in the baboons natural habitat. This ensures that the motion tracking is done in a safe and ethical manner whilst still achieving accurate motion capture.

Convolutional Neural Networks are a great tool for image object detection and pose estimation. The advances in deep learning technology have resulted in great markerless pose estimation software such as DeepLabCut.

GoPro Hero 5 Session cameras with a fisheye lens will be used for recording video of the baboons therefore the fisheye camera model will be needed. To calibrate the cameras a big checkerboard will be used as a large FOV needs to be covered.

And lastly for 3D reconstruction both methods, triangulation and sparse bundle adjustment will be used to develop a 3D pose estimation data set of the Chacma baboon.

Chapter 3

Methodology

3.1 Overview

To meet the requirements of this project a data capture system needs to be developed. The data then needs to be captured and processed. Once that is complete the results need to be analysed. An iterative approach to the project was used by completing tasks, analysing results, and then repeating the task if the results weren't sufficient.

3.2 Problem Definition

Due to previous research on the locomotion of Chacma baboons only making use of GPS-IMU collars, very little is known about there whole body kinematics. The aim of this project is to develop a method to track the kinematics of the baboons in the wild and to initialise a BaboonSet data model for future research to develop bio-inspired robots.

3.3 Design Approach

The steps followed to complete this project are as follows:

1. Build a stereo camera rig for data and video capture
2. Calibrate the cameras using a checkerboard of known size
3. Capture video footage of baboons in the wild
4. Analyse the video footage and data
5. Complete pose estimation of the baboons
6. Complete 3D reconstruction of the pose estimation data
7. Evaluate the results

3.4 Experiment Design

Various tests were needed to be done to ensure that the stereo camera rig captured accurate and reliable data. These can be grouped into data syncing and data logging tests.

3.4.1 Data Syncing

Data syncing tests were needed to ensure that the cameras were in sync and that the data was in sync with the videos. A flashing Light Emitting Diode (LED) and a beeping buzzer where used for synchronisation. The buzzer beeped at the start of data capture and the LED was held in view of both cameras. The on or off state of the LED was logged during data capture. The videos and LED state where then played frame by frame to ensure the cameras and data were in sync by visually inspecting them side-by-side.

3.4.2 Data Logging

To test the timing and therefore the accuracy of the data logging, the elapsed time was logged at the same time as the data. Data needed to be logged on a per frame basis therefore the elapsed time between data points was compared to the frame rate of the cameras to ensure they were capturing data at the same rate.

3.5 Data Collection, Analysis and Evaluation

Data was collected near the Tokai Forest in Cape Town, South Africa with guidance from members of NCC Environmental Services. Videos of the baboons in the wild were recorded as well as the rotation of the cameras.

Useful videos were then chosen and frames were extracted for labeling. These labeled frames were then used for training the DeepLabCut model for pose estimation. This processed data was then used for 3D reconstruction using triangulation and sparse based adjustment. The resulting 3D pose estimation data was evaluated.

Chapter 4

System Design

The system used to meet the goals of this study can be broken into four main components: stereo camera rig, camera calibration, 2D pose estimation, and 3D pose estimation. These components will be discussed below.

4.1 Stereo Camera Rig

4.1.1 Requirements and Constraints

The requirements of the system are to:

- Record stereo video footage
- Be portable
- Track rotation of the cameras
- Synchronise cameras and data

The constraints to the system are:

- Only two cameras can be used

- Cameras are restricted to GoPro Hero 5 Sessions
- Must be low-cost

4.1.2 Hardware Design

To meet the requirements of the system, the stereo rig needs to be able to flash an LED, log the rotational angle to an SD card, beep a buzzer, be battery powered and portable, hold two GoPro cameras at fixed positions relative to each other, and be triggered by a button.

An Arduino Nano was chosen as the device to interface all the required components as it is cheap, small and low powered. To track the angle of rotation an incremental encoder was used which resets to O every time it is powered on. To counteract this the encoder is rotated to a home position and reset to set every time it is started up to ensure all measurements are taking in the same reference frame.

Four buttons/switches are connected to the device. There is a toggle switch for power, a push button for resetting the Arduino for error handling, a push button to start data logging, and a push button to stop data logging. The device has two LEDs. A green LED is used to show the user that it is actively logging data and a red LED is used show the user when there is an error. The red LED is also used for syncing the cameras and data. When the red LED remains on this means that there is an error and the Arduino must be reset. When there is no error the LED is constantly flashing. At the start of video recording and data capture the LED is held in view of both the cameras. Whilst the cameras are recording the flashing of the LED, the on\off state of the LED is being logged to the SD card.

The SD card is interfaced with the Arduino via a SD Card Module. The module interfaces with the Arudino using the Serial Peripheral Interface protocol. Two files are stored in the SD card. A "dataset.csv" file is used to store the dataset number which relates to the order of video capture from the GoPros. This is done to ensure

the correct data is used with the correct video. The other file is a "encoder.csv" file. This file stores the dataset number, time in microseconds, rotation value from the encoder from 0 to 4095, and the state of the LED.

To power the device 3x 3.7V 9Ah size C lithium batteries were used which supplies a total of about 11.1V. To properly power the Arduino and all the components connected to it a voltage between 7V and 12V needs to be supplied to the Vin pin. If the usb connector is used to supply power, the SD card module will not work properly as it will be receiving less than 5V due to all the other components being powered by the Arduino. The maximum continuous discharge current for the battery is 200mA. A multimeter was used to measure the current draw of the system when it was operating at its peak (data logging, saving to SD card, rotating, flashing LED and beep sound playing) and it only reached a maximum of about 110mA therefore the batteries are an acceptable power supply for this system. The high Ah also means that there is no need to worry about the batteries needing to be changed any time soon.

The Arduino GPIO pins supply a voltage of 3.3V and have a maximum current rating of 40mA. With a voltage drop of approximately 1.7V, a 220Ω resistor was chosen to limit the current to approximately 7.3mA.

A handle was also designed and 3D printed to allow ease of rotation of the cameras.

4.1.3 Software Design

Figure 4.1 below shows a high-level design of the stereo rig system software. The loop continues at a rate of 60Hz, or 16.67ms time intervals until the stop button signal is received.

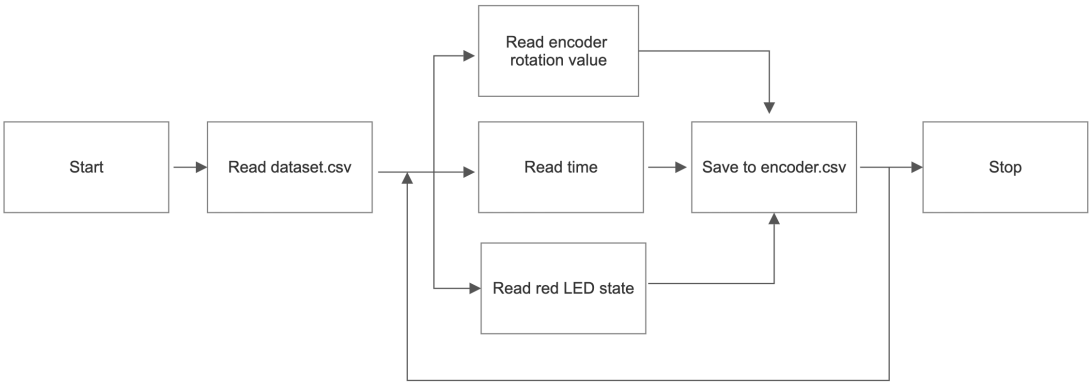


Figure 4.1: High level block diagram of the stereo rig software

A timer interrupt was used to log the angle of rotation and save the data to the SD card every frame. The video footage was recorded at 60fps therefore the timer interrupt would need to trigger at a rate of 60Hz or 60 times per second.

To reduce the latency when saving data to the SD card, the data was written to the SD card every cycle (or frame) but it was only saved after a full 512byte block had been sent as shown in Listing 4.1. This ensured that the latency during the save operation was minimized.

The data was saved in a table format with four columns consisting of the dataset number, the time in microseconds, the encoder rotation value, and the led state.

```
void saveEncoderData() // Save encoder position to sd card
{

  if (openFile == HIGH)
  {
    myFile = SD.open("encoder.csv", O_CREAT | O_APPEND | O_WRITE);
    openFile = LOW;
  }
  if (myFile)
  {
    myFile.print(dataSet);
    myFile.print(',');
    myFile.print(readMicros);
    myFile.print(',');
    myFile.print(newPosition);
    myFile.print(',');
    myFile.println(ledState);

    writeCount++;

    if (writeCount == 512)
    {
      Serial.println("FLUSH");
      myFile.flush();
      writeCount = 0;
    }
  }
  else
  {
    // if the file didn't open, print an error:
    Serial.println("SAVE error opening encoder.csv");
    error = HIGH;
  }
}
```

Listing 4.1: Function to save data to encoder.csv file

The angle in rad was calculated from the encoder rotation values in post processing using Equation 4.1

$$Angle = \frac{(\frac{Rotation}{4096})*2\pi}{3} \tag{4.1}$$

The Arduino was programmed using C++ in the Virtual Studio Code IDE. To read the encoder values the Encoder library by Paul Stroffregen was used [26]. For reading and writing data to the SD card the SdFat library by Bill Greiman was used [27].

4.1.4 System Implementation

(a) Rear view (b) Side view (c) Top view

Figure 4.2: Views of the stereo camera rig

Various views of the stereo camera rig are shown in Figure 4.2. The horizontal bar can rotate about the vertical axis whilst the rest of the rig is fixed. There is a gear ratio of 3:1 therefore for every full 360° rotation of the cameras, the encoder shaft is rotated three times. The gear ratio compensation was handled in the code.

The circuit for connecting the Arduino and various components was soldered onto a veroboard to ensure no connections came loose.

The two weights placed on the horizontal bar act as stabilizers to reduce minor vibrations from the camera rig shaking.

4.1.5 Evaluation Methods

To test the accuracy of the timing of the Arduino, a GPIO pin was probed by an oscilloscope and a pulse was sent for every trigger of the timer interrupt. The oscilloscope showed a steady 60Hz frequency which means that the timing is ideal.

4.2 Camera Calibration

4.2.1 Requirements and Constraints

The requirements of the system are to:

- Determine the intrinsic parameters of the cameras
- Determine the extrinsic parameters of the cameras

The constraints to the system are:

- Use a black and white checkerboard
- Cameras can not be moved relative to each other after calibration

4.2.2 System Implementation

To implement the camera calibration, a black and white checkerboard with 8x11 90mm squares was recorded with both cameras at the same time. The checkerboard

was moved around the field of view and rotated to create a robust set of calibration images.

Using calibration code that was modified from the AcinoSet dataset [28], the camera calibration for intrinsic and extrinsic parameters was completed.

4.2.3 Evaluation Methods

The calibration program produces a root mean square reprojection error for the intrinsic and extrinsic calibration. This can be used to analyse the accuracy of the collaboration.

4.3 2D Pose Estimation

4.3.1 Requirements and Constraints

The requirements of the system are to:

- Accurately track body parts across frames

The constraints to the system are:

- Limited amount of time for labeling and training
- Only one person labeling the frames which is very time consuming
- Done using DeepLabCut

4.3.2 System Implementation

The 2D pose estimation was implemented using DeepLabCut. First frames from videos of baboons are extracted and their body parts are labeled. The labels used are shown in Figure 4.3.

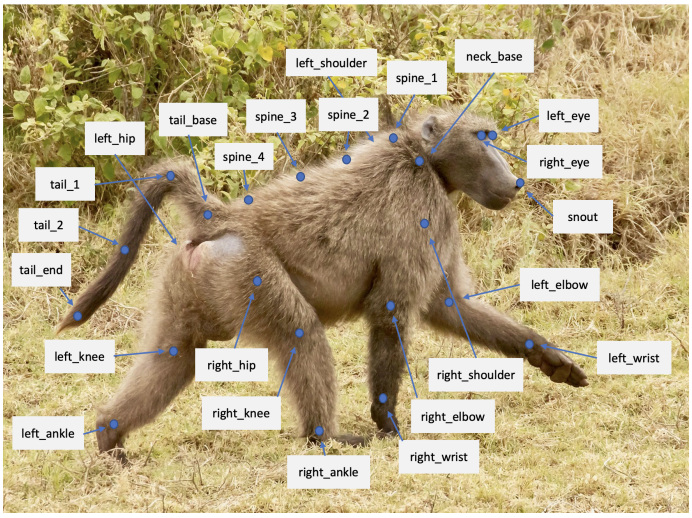


Figure 4.3: Labels locations used as a guide for DeepLabCut labelling [29]:

The labeled data is then fed into the Convolutional Neural Network where a DeepLabCut model is trained for 2D pose estimation.

4.3.3 Evaluation Methods

The network can be evaluated within the DeepLabCut software. This shows a train and test error in pixels for a chosen likelihood cutoff. Likelihood is the confidence that the network has that the marker has been placed in the correct location.

4.4 3D Pose Estimation

4.4.1 Requirements and Constraints

The requirements of the system are to:

- Reconstruct 2D points from left and right images into 3D real world coordinates

The constraints to the system are:

- Limited to triangulation and sparse bundle adjustment due to time constraints

4.4.2 System Implementation

The 3D pose estimation is implemented using the modified AcinoSet python code [28]. It uses OpenCV and the Jupyter Notebook IDE. The code can import the 2D pixel positions for each image from DeepLabCut and then implement either triangulation or sparse bundle adjustment 3D reconstruction using the cameras intrinsic and extrinsic parameters.

4.4.3 Evaluation Methods

The 3D reconstruction is tested by visual inspection and by comparing the 3D positions of each reconstruction method to a manually labelled ground truth model and calculating the percentage of correct keypoints (PCK) which evaluated the distance or offset of the predicted joint from the ground truth joint.

Chapter 5

Results and Discussion

5.1 Stereo Camera Rig

The purpose of the stereo camera rig was to enable low budget stereo data capture of baboons in the wild. The stereo camera rig was built with this in mind. The results of the effectiveness of the camera rig for stereo data capture are discussed below.

5.1.1 Results

One of the vital requirements of the stereo camera rig is the ability to to capture video from two cameras that contain the same object in their FOV and keep a record of the rotation of the cameras on a per frame basis. The stereo camera rig also needs to provide the capability of syncing the data and cameras together.

A MATLAB script was written to sync the cameras and data as well as provide a UI to verify the results. First the audio is extracted from each camera and the offset between the two audio files is found as shown in Listing 5.1.

```
%% Delay of the two videos

D = finddelay(y_audio,y_audio2);

syncOffset = ((D)/48000); % seconds
```

Listing 5.1: Function to find the offset between two audio files

Before the result can be used to sync up the videos the offset was found between the audio and the video of each camera by plotting the video, frame by frame, above a plot of the audio. A loud clap sound was made when recording. The video and audio were then plotted and it was found that the spike in the audio had an offset of 7 frames compared to the video. This offset was carried throughout the rest of the script.

When data logging is started, the buzzer plays two beep sounds at 2800Hz and three beep sounds at 2800Hz when logging is stop. To use this information to find the time in the video when data started and stopped logging, the audio file was transformed from the amplitude time domain to the frequency time domain using a spectrogram as show in Listing 5.2.

```
%% Frequency 1

[s,f,t] = spectrogram(y_audio,blackman(1000),200,5000,Fs);

[smx,trow] = max(abs(s),[],1);
    % Time Indices Of Maxim[U+FFFD]s[U+FFFD]
frqv = f(trow);
    % Frequencies For Each Time

startPos = find(frqv(1:round(length(frqv)/2))>(startBeepFreq - 100)
    & frqv(1:round(length(frqv)/2))<(startBeepFreq + 100));
startTime = t(startPos(1)) - 2/frameRate;

stopPos = find(frqv(round(length(frqv)/2):end)>(stopBeepFreq - 100)
    & frqv(round(length(frqv)/2):end)<(stopBeepFreq + 100)) + round(
    length(frqv)/2);
stopTime = t(stopPos(1)) - 2/frameRate;

plot(ax3, t(startPos(1):(stopPos(1))), frqv(startPos(1):(stopPos(1))
    ));
title(ax3,"Frequency");

% Fix the axes
ax3.XLim = [startTime (stopTime)];
ax3.YLim = [0 max(frqv)*1.2];
```

Listing 5.2: Function using the frequecny to find the start and stop times of data capture

Using the start and stop time found above, the video start time can be set for each camera and the video frames can be plot vs the state of the sync LED to visualize the sync offset of the cameras and the logged data, in this case the on\off state of an LED in the FOV of both cameras as shown in Figure 5.1.

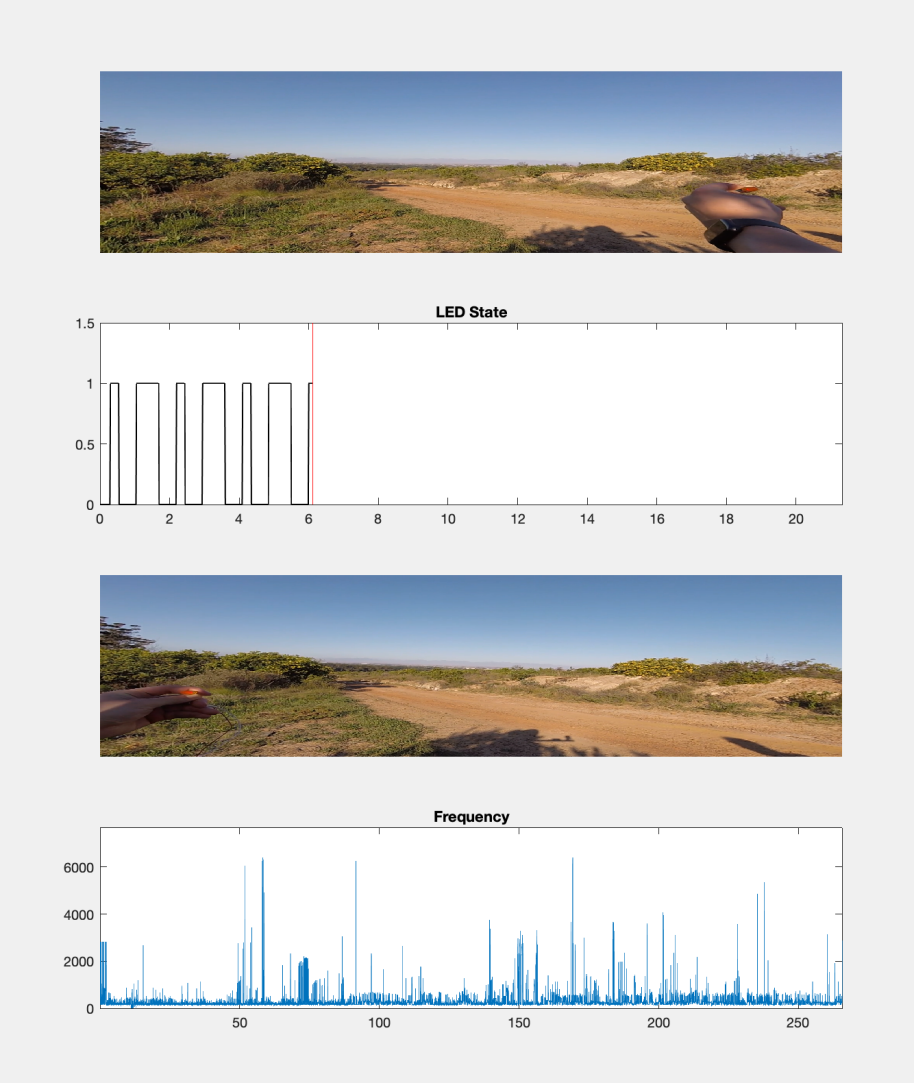


Figure 5.1: MATLAB plot to visualize camera and data syncing

If there is an offset found between the cameras and the data, the offset can be counted as the video is played at only 2 frames per second. The offset can be found by watching when the LEDs turn either on or off and when the state of the LED plot is high or low.

The left camera at the top and the right camera at the bottom can also be

visually inspected to ensure that the audio syncing done in Listing 5.1 was accurate by ensuring that the state of the LED changes at the same time.

The full MATLAB code for syncing the cameras and data can be found in Figure A.1 in Appendix A.

5.1.2 Discussion

As shown in the results section above the stereo camera rig system is successful in capturing stereo video footage and ensuring the data and cameras are in sync. The method followed to obtain the results and how the results were used was also discussed in depth.

5.2 Camera Calibration

The purpose of camera calibration is to find the intrinsic and extrinsic parameters of the cameras. This is used to handle lens distortion and to map the image 2D coordinate into 3D real world coordinates.

5.2.1 Results

A large black and white checkerboard with 8x11 90mm squares was used for calibration. The intrinsic and extrinsic calibration footage was recorded at the same time using the checkerboard.

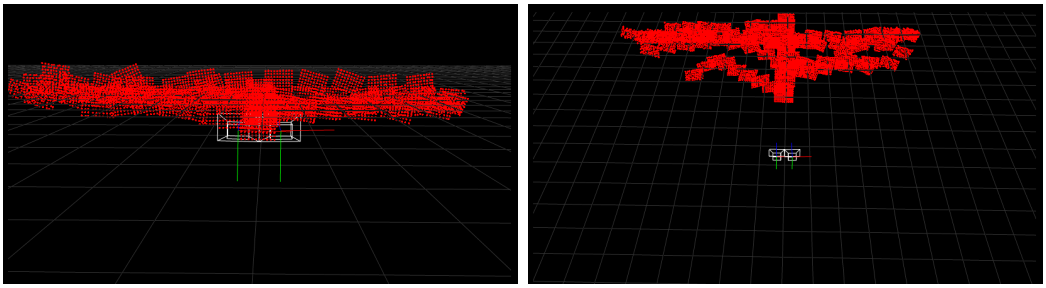
A 6min and 24sec video was recorded whilst moving the checkerboard around in the FOV of the two cameras. A frame was then extracted every second which resulted in a total of 384 calibration frames. Using OpenCV and the calibration code from AcinoSet [28], the coordinates of the points of the checkerboard squares were found. These points were used for calibration to find the instrinsic parameters of camera 1. Due to both of the cameras being GoPro Hero 5 Sessions it can be

assumed that they have the same camera intrinsics. A fisheye lens model is used due to the GoPro have a fisheye lens.

The root mean square (RMS) error of the intrinsic calibration was found to be 0.255 pixels which is acceptable.

Next the extrinsic calibration was done through pairwise calibration. Using the checkerboard points from both the left and right cameras the extrinsic parameters where found using 112 common frames between the left and right camera.

A RMS reprojection error of 0.37418 pixels was returned.



(a) Low angle view (b) High angle view

Figure 5.2: 3D scene reconstruction of the camera calibration using a checkerboard

Figure 5.2 shows two different views of the scene reconstruction of the camera calibration. It is clear that a large volume was covered to ensure a robust camera calibration was achieved.

5.2.2 Discussion

The RMS reprojection errors for the camera and stereo camera setup are shown in Table 5.1

By looking at the large volume covered by the calibration as shown in Figure 5.2 and the low RMS error in Table 5.1, it can be concluded that a robust calibration was found for the stereo camera setup.

Table 5.1: RMS Error Summary

Calibration	RMS Error (pixels)
Intrinsic	0.255
Extrinsic	0.374

Now that the camera calibration is complete it is vital that the cameras do not move relative to each other as a result of getting bumped or disturbed. If the stereo camera system does get disturbed the camera calibration for the extrinsic parameters would need to be repeated.

5.3 2D Pose Estimation

The video footage acquired near the Tokai Forest in Cape Town, South Africa, of the baboons in the wild was then loaded into DeepLabCut. Using DeepLabCut, frames were manually extracted due to large gaps of inactivity in the videos. 600 frames were then labeled as shown in Figure 4.3 in Chapter 4.3.2. Once all the frames were labeled the DeepLabCut training network was created and the model was trained up to 200 000 iterations.

5.3.1 Results

The evaluation of iteration 100K and 200K are shown in Table 5.2 and 5.3. It is shown that the network is most accurate when trained to 200K iterations.

Table 5.2: Training Network Evaluation Results with no p-cutoff

Training iterations	Train error(px)	Test error (px)
100 000	2.99	14.51
200 000	2.8	8.14

Table 5.3: Training Network Evaluation Results with p-cutoff of 0.6

Training iterations	Train error with p-cutoff	Test error with p-cutoff
100 000	2.81	9.73
200 000	2.66	6.14

Using this newly trained network, videos could be analysed for pose estimation. Due to time constraints, more frames could not have been labeled and the network be retrained to strengthen its tracking. As a result, no jumping videos could be used as the tracking was not good enough for such a high speed action. Therefore the focus for pose estimation will only be on a walking baboon.

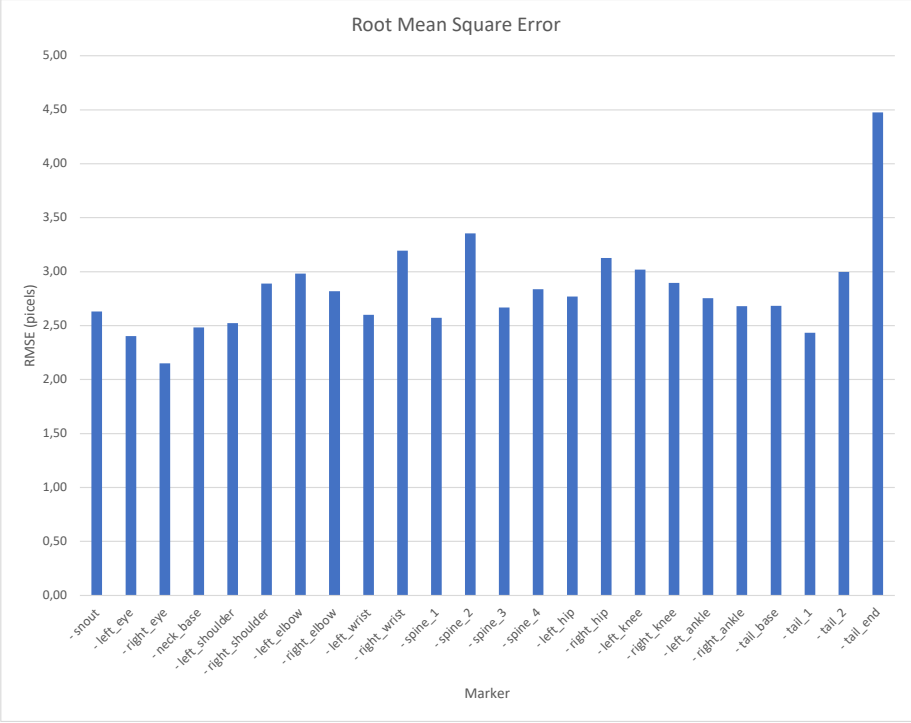


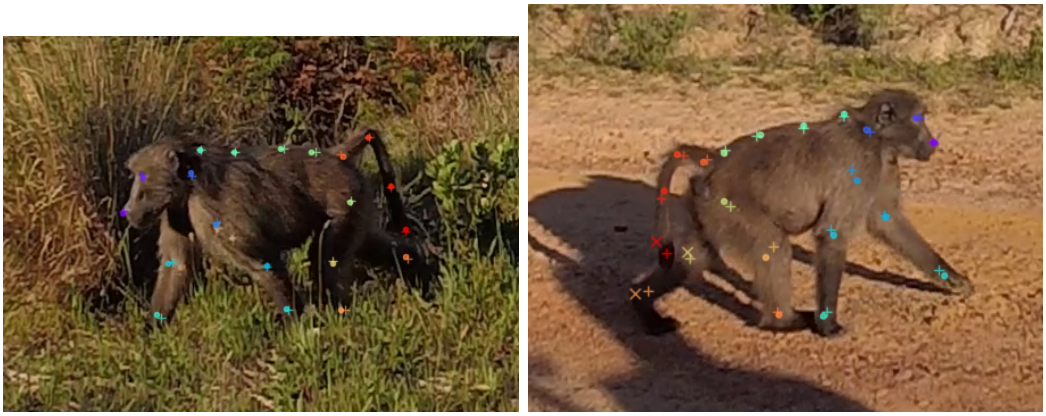
Figure 5.3: Root Mean Square Error of a tracked baboon

Figure 5.3 shows the mean RMSE per body part for a baboon across 600 frames. The RMSE error is the trained network error relative to the ground truth labeled

data.

5.3.2 Discussion

As shown in the section above, training the network to 200 000 iterations compared to 100 000, results in large decreases in test error. Train error is the difference in pixels of the training data compared to the labeled data whilst test error is the error in pixels of the new unseen data.



(a) Train error (b) Test error

Figure 5.4: Images showing detection of baboon body parts by DeepLabCut trained model

As seen in the images in Figure 5.4, DeepLabCut struggles to accurately detect body parts near the rear of the baboon. This coincides with the spike in RMSE shown for the tail end in Figure 5.3.

This is most likely caused by a lack of viewpoints from the rear side of the baboon in the labeled data. Other than that it seems that DeepLabCut is doing acceptable tracking of the baboon therefore there should be enough data to implement 3D reconstruction of a walking baboon.

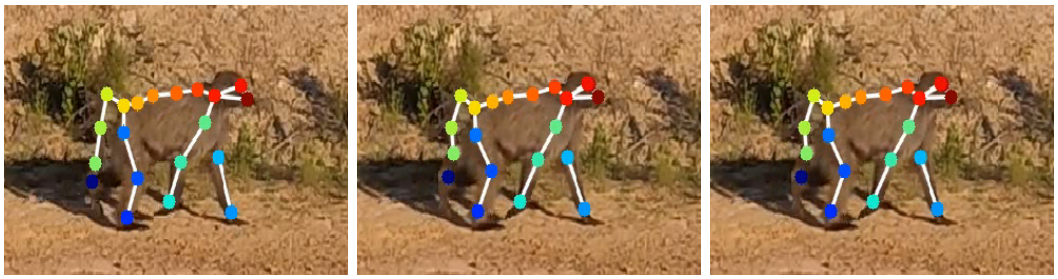
5.4 3D Pose Estimation

Two methods of 3D reconstruction were implemented to track the 3D kinematics of the baboon. Triangulation and sparse bundle adjustment (SBA). The results of these 3D reconstructions will be shown and their accuracy will be compared to ground truth data.

5.4.1 Results

To implement triangulation and SBA, the code from AcinoSet was modified to work for a 2 camera system and the increase of 24 joints compared to 20 of the cheetah model [28].

To evaluate the accuracy of 3D reconstruction a ground truth model was developed. Three frames were extracted from a video and the joints were manually labeled for the left and right camera. Triangulation was then used to represent the ground truth model in the real world 3D coordinates. The 3D reconstructed skeleton for frame 1 is shown in Figure 5.5. The ground truth model (a) was manually labelled in 2D and mapped to 3D using triangulation. Images (b) and (c) where obtained by doing 3D reconstruction of the pose estimation data from DeepLabCut.



(a) Ground Truth (b) Triangulation (c) SBA

Figure 5.5: Visualisation of triangulation and SBA compared to the ground truth model

To compare the accuracy of the two 3D reconstruction methods, the percentage of correct keypoints (pck) metric was used. The pck value represents the percentage of detections that fall within a certain distance, X, to the ground truth model.

The code used to calculate the pck is shown in Listing 5.3.

```
X = 10/100 % 10 cm
difference = frameTri - frameGT;

distance = sqrt(difference(1,:,1).^2 + difference(1,:,2).^2 +
    difference(1,:,3).^2);

correct = 0;
for i = 1:24

    if distance(i) < X
      correct = correct + 1;
    end
end

PCK = correct/20

PCK_per = correct/20 * 100
```

Listing 5.3: Code to calculate the pck value at X

The pck for a distance of 1cm, 5cm, 10cm, 20cm, and 25cm was calculated and the results shown in Table 5.4.

Table 5.4: 3D reconstruction analysis using pck

Method	**X=1cm**	**X=5cm**	**X=10cm**	**X=20cm**	**X=25cm**
Triangulation	0	30	60	75	95
SBA	0	30	60	75	95

5.4.2 Discussion

As seen in Table 5.4, triangulation and sparse bundle adjustment give the same level of accuracy in terms of 3D reconstruction.

It can also be noted that none of the keypoints are within a 1cm accuracy and that only 60% of them are within a 10cm accuracy. This could possibly be improved by using a better DeepLabCut model and adjusting the baseline of the cameras to change the disparity. A baseline of 60cm was used for this project.

One of the issues found with both the 3D reconstruction methods was that the baboon moved or rotated in the camera frame and not relative to the world frame. This was due to the rotation of the cameras constantly changing the inertial frame.

To rectify this issue the rotation matrix used to map the 2D image plane coordinates to 3D real word coordinated needs to be rotated about the vertical access on a per frame basis using the rotation data that was logged during recording. This was not done during these tests as a major flaw was noticed with the design of the stereo camera rig. When rotating the cameras, they rotate about the centre of the camera rig and not about the centre of each individual camera.

As a result it is not possible to rotate the cameras rotation matrix. One would normally first translate the coordinate frame, rotate it, and then translate it back but the rotation matrix of the cameras has no reference to points and therefore cannot be translated.

An improved stereo camera rig design that enables the cameras to rotate about their own vertical axis would fix this issue.

Chapter 6

Conclusions

6.1 Conclusions

Chacma baboons are a great candidate to study when building bio-inspired robots due to their agile and athletic movements. Very little is known about their whole body kinematics due to past studies only making use of GPS-IMU tracking collars which can't differentiate movements for each body part.

This project showed the development of a low cost stereo camera rig and the workflow of using the data collected to successfully implement pose estimation and 3D reconstruction to enable the study of the baboon's kinematics. This was done in a non-laboratory environment which results in the baboons not being disturbed. This also means that the study can be done in an ethical and safe manner without capturing the baboons to be studied in a lab.

A few limitations have been found during this project. The first one would be the inability to compensate for the rotation of the cameras whilst recording footage due to the cameras not rotating about their own vertical axis. This can be rectified by redesigning the camera rig.

Another limitation is that the two 3D reconstruction methods used are non-model

based. This means that the baboon's pose parameters and constraints were not taken into consideration. An improvement on this would be to use an Extended Kalman Filter (EKF) for 3D reconstruction. The EKF takes the baboon's rigid body model into account which results in a much more robust and accurate representation of the baboon's 3D kinematics.

Time was another limitation. There wasn't enough time to follow a reiterative process of retraining the DeepLabCut model to improve its tracking. As a result, the jumping motion of the baboon was not tracked properly and could not be analysed. The walking motion of the baboon was then used throughout the project.

Overall the project was successful at developing the 3D kinematics of the Chacma baboon in the wild in low-cost and easily accessible manner.

6.2 Future Work

This project is just the start of studying the 3D kinematics of the Chacma baboon. A much larger dataset will be needed to gather enough information to enable the development of bio-inspired robots.

A larger dataset will also help improve the pose estimation abilities of DeepLabCut.

This project can be improved upon by redesigning the stereo camera rig to enable the cameras to rotate about their own vertical axis. Using an Extended Kalman Filter will also help improve the 3D reconstruction tremendously.

The stereo camera rig and workflow presented in this projected can also be expanded for use with any other animals. As a result knowledge about animal's amazing athletic abilities can pave the way to amazing bio-inspired robots in the future.

Bibliography

[1] A. Sequeira, A. Usman, P. Oommen, Z. Tharakan, and Ali, "Biologically inspired robots into a new dimension- a review," *International Journal of Automation, Mechatronics Robotics*, vol. 3, 08 2016.

[2] "How are imus different to gps systems? — imeasureu," https://imeasureu.com/2020/02/06/how-are-imus-different-to-gps-systems/, (Accessed on 11/05/2021).

[3] J. S. Furtado, H. H. T. Liu, G. Lai, H. Lacheray, and J. Desouza-Coelho, "Comparative Analysis of OptiTrack Motion Capture Systems," in *Advances in Motion Sensing and Control for Robotic Applications*, F. Janabi-Shari[U+FB01] and W. Melek, Eds. Springer International Publishing, 2019, pp. 15–31.

[4] J. E. van Schaik and N. Dominici, "Motion tracking in developmental research: Methods, considerations, and applications," *Progress in Brain Research*, vol. 254, pp. 89–111, 2020.

[5] G. Fehlmann, M. J. O'Riain, P. W. Hopkins, J. O'Sullivan, M. D. Holton, E. L. Shepard, and A. J. King, "Identification of behaviours from accelerometer data in a wild social primate," *Animal Biotelemetry*, vol. 5, no. 1, pp. 1–11, 2017.

[6] F. Schlagenhauf, S. Sreeram, and W. E. Singhose, "Comparison of kinect and vicon motion capture of upper-body joint angle tracking," *2018 IEEE 14th International Conference on Control and Automation (ICCA)*, pp. 674–679, 2018.

[7] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," *arXiv preprint arXiv:1511.08458*, 2015.

[8] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," 2019.

[9] H.-S. Fang, S. Xie, Y.-W. Tai, and C. Lu, "Rmpe: Regional multi-person pose estimation," 2018.

[10] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. Gehler, and B. Schiele, "Deepcut: Joint subset partition and labeling for multi person pose estimation," 2016.

[11] J. M. Graving, D. Chae, H. Naik, L. Li, B. Koger, B. R. Costelloe, and I. D. Couzin, "DeepPoseKit, a software toolkit for fast and robust animal pose estimation using deep learning," *bioRxiv*, 2019, publisher: Cold Spring Harbor Laboratory _eprint: https://www.biorxiv.org/content/early/2019/09/04/620245.full.pdf. [Online]. Available: https://www.biorxiv.org/content/early/2019/09/04/620245

[12] T. Pereira, D. E. Aldarondo, L. Willmore, M. Kislin, S. S.-H. Wang, M. Murthy, and J. W. Shaevitz, "Fast animal pose estimation using deep neural networks," *bioRxiv*, 2018, publisher: Cold Spring Harbor Laboratory eprint: https://www.biorxiv.org/content/early/2018/05/30/331181.full.pdf. [Online]. Available: https://www.biorxiv.org/content/early/2018/05/30/331181

[13] P. Karashchuk, K. L. Rupp, E. S. Dickinson, S. Walling-Bell, E. Sanders, E. Azim, B. W. Brunton, and J. C. Tuthill, "Anipose: A toolkit for robust

markerless 3D pose estimation," *Cell Reports*, vol. 36, no. 13, p. 109730, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2211124721011797

[14] J. Lauer, M. Zhou, S. Ye, W. Menegas, T. Nath, M. M. Rahman, V. Di Santo, D. Soberanes, G. Feng, V. N. Murthy, G. Lauder, C. Dulac, M. W. Mathis, and A. Mathis, "Multi-animal pose estimation and tracking with DeepLabCut," *bioRxiv*, 2021, publisher: Cold Spring Harbor Laboratory _eprint: https://www.biorxiv.org/content/early/2021/04/30/2021.04.30.442096.full.pdf. [Online]. Available: https://www.biorxiv.org/content/early/2021/04/30/2021.04.30.442096

[15] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[16] "The pinhole camera - anagram engineering," https://www.anagram.at/diplomarbeit/the-pinhole-camera/, (Accessed on 11/06/2021).

[17] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A toolbox for easily calibrating omnidirectional cameras," 10 2006.

[18] "Fisheye calibration basics - matlab & simulink," https://www.mathworks.com/help/vision/ug/fisheye-calibration-basics.html#mw_f299f68d-403b-45e0-9de5-55203a09460d, (Accessed on 11/06/2021).

[19] "Opencv: Fisheye camera model," https://docs.opencv.org/4.5.3/db/d58/group__calib3d__fisheye.html, (Accessed on 11/06/2021).

[20] "Calibration checkerboard collection — mark hedley jones," https://markhedleyjones.com/projects/calibration-checkerboard-collection, (Accessed on 11/07/2021).

[21] "Opencv: Camera calibration," https://docs.opencv.org/3.4.15/dc/dbb/tutorial_py_calibration.html, (Accessed on 11/07/2021).

[22] "Estimate geometric parameters of a single camera - matlab," https://www.mathworks.com/help/vision/ref/cameracalibrator-app.html, (Accessed on 11/07/2021).

[23] G. D. Leo, C. Liguori, and A. Paolillo, "Propagation of uncertainty through stereo triangulation," *2010 IEEE Instrumentation & Measurement Technology Conference Proceedings*, pp. 12–17, 2010.

[24] R. I. Hartley and P. Sturm, "Triangulation," *Computer Vision and Image Understanding*, vol. 68, no. 2, pp. 146–157, 1997. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1077314297905476

[25] N. Sünderhauf, K. Konolige, S. Lacroix, and P. Protzel, "Visual odometry using sparse bundle adjustment on an autonomous outdoor vehicle," 01 2005, pp. 157–163.

[26] "Paulstoffregen/encoder: Quadrature encoder library for arduino," https://github.com/PaulStoffregen/Encoder, (Accessed on 11/07/2021).

[27] "greiman/sdfat: Arduino fat16/fat32 exfat library," https://github.com/greiman/SdFat, (Accessed on 11/07/2021).

[28] D. Joska, L. Clark, N. Muramatsu, R. Jericevich, F. Nicolls, A. Mathis, M. W. Mathis, and A. Patel, "Acinoset: A 3d pose estimation dataset and baseline models for cheetahs in the wild," 2021.

[29] "Chacma baboon (papio ursinus) — the chacma baboon (papio urs... — flickr," https://www.flickr.com/photos/mosesharold/49177644586, (Accessed on 11/07/2021).

[30] "Opencv: Camera calibration and 3d reconstruction," https://docs.opencv.org/4.5.3/d9/d0c/group__calib3d.html, (Accessed on 11/06/2021).

[31] "1511.08458.pdf," https://arxiv.org/pdf/1511.08458.pdf, (Accessed on 11/06/2021).

[32] J. Heikkila and O. Silvén, "A four-step camera calibration procedure with implicit image correction," in *Proceedings of IEEE computer society conference on computer vision and pattern recognition*. IEEE, 1997, pp. 1106–1112.

[33] J.-Y. Bouguet, "Camera calibration toolbox for matlab," *http://www. vision. caltech. edu/bouguetj/calib_doc/index. html*, 2004.

Appendix A

Code

```
%% Constants

video1FileName = "/Users/matthewterblanche/Downloads/Baboon Data/
    BaboonSet-Matthew Terblanche-2021-10-07/videos/CAM1-4.MP4";
video2FileName = "/Users/matthewterblanche/Downloads/Baboon Data/
    BaboonSet-Matthew Terblanche-2021-10-07/videos/CAM2-4.MP4";



datafile = "/Users/matthewterblanche/Desktop/UCT/5th Year 2021/2nd
    Semester/EEE4022S/Recording Day 2/Data/encoder4.csv";

audioVideoOffset = 7; % frames

startBeepFreq = 2800; % Hz
stopBeepFreq = 2800; % Hz

frameRate = 59.94; % fps / Hz

startOffset = 0;
```

```
%% Setup the subplots
ax1 = subplot(4,1,1); % For video
ax2 = subplot(4,1,2); % For sensor plot
ax3 = subplot(4,1,4); % For audio  plot
ax4 = subplot(4,1,3); % For video 2

%% Read audio1 data

[y_audio, Fs] = audioread(video1FileName);
 y_audio = y_audio(:,1);

dt = 1/Fs;

%% Read audio2 data

[y_audio2, Fs2] = audioread(video2FileName);
 y_audio2 = y_audio2(:,1);

dt2 = 1/Fs2;

%% Delay of the two videos

D = finddelay(y_audio,y_audio2);

syncOffset = ((D)/48000); % seconds

%% Frequency 1

[s,f,t] = spectrogram(y_audio,blackman(1000),200,5000,Fs);

[smx,trow] = max(abs(s),[],1);
   % Time Indices Of Maxim[U+FFFD]s[U+FFFD]
```

```
frqv = f(trow);
    % Frequencies For Each Time

startPos = find(frqv(1:round(length(frqv)/2))>(startBeepFreq - 100)
    & frqv(1:round(length(frqv)/2))<(startBeepFreq + 100));
startTime = t(startPos(1)) - 2/frameRate;

stopPos = find(frqv(round(length(frqv)/2):end)>(stopBeepFreq - 100)
    & frqv(round(length(frqv)/2):end)<(stopBeepFreq + 100)) + round(
    length(frqv)/2);
stopTime = t(stopPos(1)) - 2/frameRate;

plot(ax3, t(startPos(1):(stopPos(1))), frqv(startPos(1):(stopPos(1))
    ));
title(ax3,"Frequency");

% Fix the axes
ax3.XLim = [startTime (stopTime)];
ax3.YLim = [0 max(frqv)*1.2];

%% Setup Video1Reader object

v1 = VideoReader(video1FileName);

nFrames = floor(v1.NumFrames - (v1.NumFrames - stopTime*frameRate) -
     startTime*frameRate);

v1.CurrentTime = startTime + (audioVideoOffset)/v1.FrameRate +
    startOffset/frameRate
disp("Video 1 Start Logging Time (sec): ")
disp(v1.CurrentTime)

disp("Video 1 Start Logging Time (frames): ")
```

```
75 disp(v1.CurrentTime*v1.FrameRate)
76
77 % Display the first frame in CAM 1 plot
78 vidFrame1 = readFrame(v1);
79
80 image(vidFrame1, 'Parent', ax1);
81 ax1.Visible = 'off';
82
83 %% Setup Video2Reader object
84
85 v2 = VideoReader(video2FileName);
86 v2.CurrentTime = v1.CurrentTime + syncOffset
87
88 disp("Video 2 Start Logging Time (sec): ")
89 disp(v1.CurrentTime)
90
91
92 disp("Video 2 Start Logging Time (frames): ")
93 disp(v1.CurrentTime*v1.FrameRate)
94
95 % Display the first frame in CAM 2 plot
96 vidFrame2 = readFrame(v2);
97 image(vidFrame2, 'Parent', ax4);
98 ax4.Visible = 'off';
99
100 %% Load the sensor data
101
102 [Time, Rotation, LEDState] = importfile(datafile, [2,inf]);
103 tSensor = (Time - Time(1))./1000000;
104 y = LEDState;
105
106 %%
107 angle = ((Rotation / 4096) * 360) / 3;
```

```
108
109 %y = angle;
110 %%
111 nDataPoints = length(tSensor); % Number of data points
112 step = round((nFrames/nDataPoints));
113 index = 1:step:nDataPoints;
114
115 i = 2;
116 % Diplay the plot corresponds to the first frame in the bottom
    subplot
117 h = plot(ax2,tSensor(1:index(i)),y(1:index(i)),'-k', 'linewidth', 1)
    ;
118 title(ax2,"LED State");
119 % Fix the axes
120 ax2.XLim = [tSensor(1) tSensor(end)/50];
121 ax2.YLim = [min(y) max(y)*1.5];
122
123
124 %% Add timeline
125
126 y2 = ylim(ax2);
127 x2 = xlim(ax2);
128
129 l2=line(ax2,[x2(1),x2(1)],[y2(1),y2(end)],'color','r', 'linewidth',
    0.1);
130
131 %% Animate
132 while hasFrame(v1)
133
134 vidFrame1 = readFrame(v1);
135 image(vidFrame1, 'Parent', ax1);
136 ax1.Visible = 'off';
137
```

```
138 vidFrame2 = readFrame(v2);
139 image(vidFrame2, 'Parent', ax4);
140 ax4.Visible = 'off';
141
142
143 pause(0.5);
144
145 %pause(1/v1.Framerate);
146 disp(i)
147
148 set(l2, 'XData', tSensor(index(i))*[1,1])
149 set(h,'YData',y(1:index(i)), 'XData', tSensor(1:index(i)))
150
151 i = i + 1;
152
153
154 if (i == nFrames)
155     return;
156 end
157
158 end
```

Listing A.1: MATLAB code used for syncing stereo camera rig videos and data

Link to a Github repository containing code for the stereo camera rig

ETHICS 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		Matthew Terblanche
Department		Electrical Engineering
Preferred email address of applicant:		trbmat002@myuct.ac.za
If Student	Your Degree: e.g., MSc, PhD, etc.	BSc Eng in Mechatronics
	Credit Value of Research: e.g., 60/120/180/360 etc.	40
	Name of Supervisor (if supervised):	A/Prof. Amir Patel
If this is a researchcontract, indicate the source of funding/sponsorship		-
Project Title		Jumping Kinematics of the Chacma Baboon

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.

APPLICATION BY	Full name	Signature	Date
Principal Researcher/ Student/External applicant	Matthew Terblanche		16/8/2021
SUPPORTED BY	Full name	Signature	Date
Supervisor (where applicable)	A/Prof. Amir Patel		16/8/2021

APPROVED BY	Full name	Signature	Date
HOD (or delegated nominee) Final authority for all applicants who have answered NO to all questions in Section 1; and for all Undergraduate research (Including Honours).			
Chair: Faculty EIR Committee For applicants other than undergraduate students who have answered YES to any of the questions in Section 1.			