Chapter 1

Designing the Data Capture System

To obtain data for the Extended Kalman Filter, a data-capture system needed to be designed. Since the data sources have been identified as multiple video streams and various sensors from a smartphone these devices must be mounted to a wearable harness. The following specifications are highly modular as any camera source and any smartphone with sufficient sensors and sufficient data capture rate would suffice.

1.1 GoPro Hero Session Camera

Due to the availability of GoPro Hero Session cameras the wearable motion capture system was designed with these in mind. These cameras only take up a volume of $250cm^3$ and have a square housing measuring $6.3cm$ on all sides. The following figure presents a 3D rendered model of the camera.

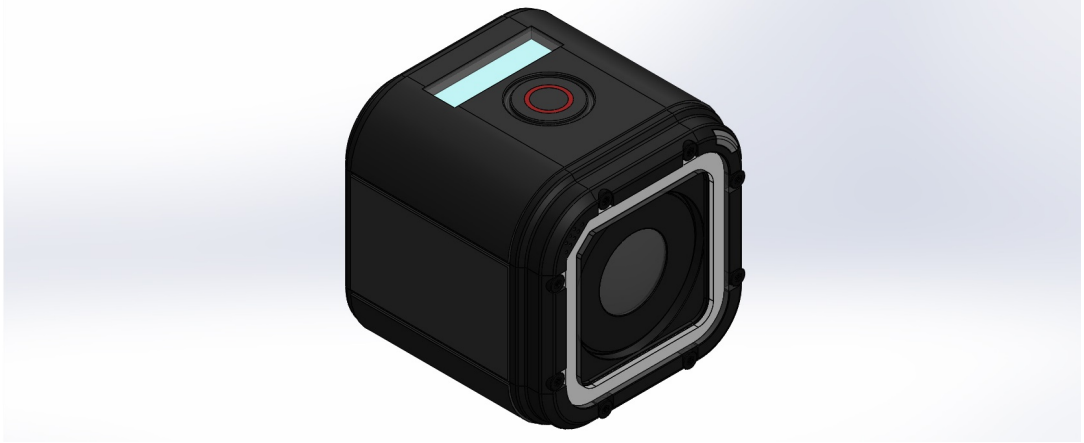

Figure 1.1: 3D CAD rendering of the GoPro Hero Session action camera from [1]

The camera can capture videos at a variety of frame rates and a variety of resolutions. These settings are limited as the software that controls the camera is proprietary. The set frame rates and resolutions are presented in the table below.

Resolution	Frame Rates
1920 x 1440	30 fps, 25 fps
1920 x 1080	60 fps, 50 fps, 48 fps, 30 fps, 25 fps
1280 x 960	60 fps, 50 fps, 30 fps, 25 fps
1280 x 720	100 fps, 60 fps, 50 fps, 30 fps, 25 fps
848 x 480	120 fps, 100 fps

Table 1.1: Possible frame rate and resolution combinations on the GPHS camera

The relative motion of the lower limbs appeared to move rapidly when viewing test footage and therefore the highest possible frame rate with the best resolution was chosen. The camera was configured to record at 100Hz and at a resolution of 1280 x 720 pixels. This was chosen as the quality of the 848 x 480 video was assumed to distorted to identify the marker centres with computer algorithms.

The camera also has the ability to record using a normal lens or a wide angle lens. The field of view of the camera greatly increases with the wide angle lens but its focal length decreases proportionally. The wide lens also produces more distortion when compared to the normal lens. [2]. Due to the relatively narrow area of capture needed the camera was configured to use the normal lens as it would decrease distortion without compromising the area of interest.

Naturally, working with proprietary hardware presents some difficulties. One of these difficulties is created by the GoPro camera regulating its exposure automatically. In darker environments the GoPro will automatically change to low light mode, causing inconsistent light levels in videos taken in different conditions. This causes uncertainty when trying to use feature detection since the varying light levels change the relative colors of the markers.

Another difficulty is the output video files generated by the GoPro. These files have a .MP4 file extension implying that they have already been compressed. This compression causes a loss of precision opposed to the raw video data being recorded. Compression had been implemented to save memory on the GoPro's micro SD memory card. Decompressing this video data will be discussed in the following chapter.

1.2 Camera Mount Design

Some initial work on modelling a housing for the camera was completed by the Mechatronics Research Lab. This was a 2 part 3D printable enclosure with no mounting points or control access. The enclosure is pictured below.

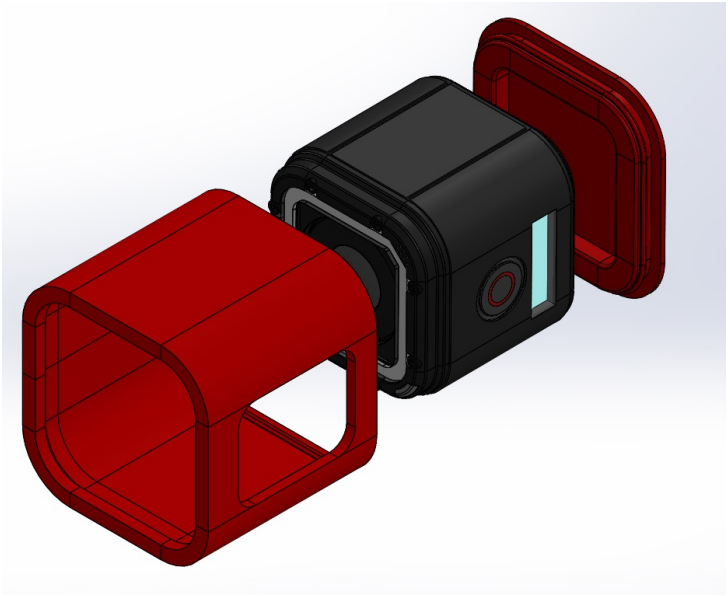

Figure 1.2: Initial camera enclosure designed by the mechatronics lab

This model was heavily modified using Dassault Systems SOLIDWORKS software to enclose 2 cameras mounted side by side. The bracket also needed a mounting point to join to the chest mount. Finally the bracket needed to be lightweight, provide access to the camera controls and not obscure the built in status screen of the cameras. The following figure shows the final dual camera bracket that was 3D printed. This bracket can be found on the accompanying CD.

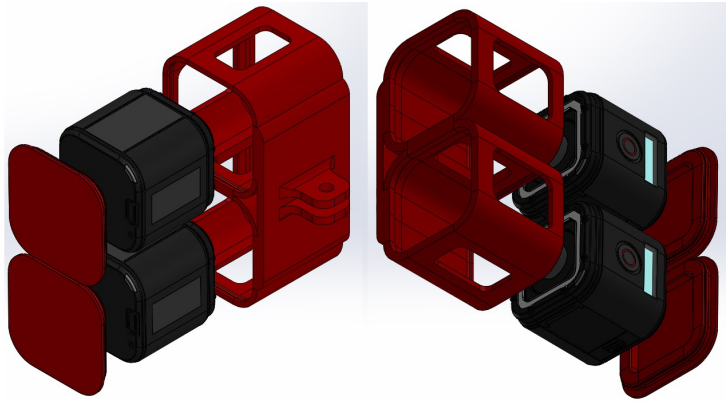

Figure 1.3: Final dual camera enclosure designed by the author

As seen in the figure above the sides of the housing was opened to reduce the overall weight of the mount. These opening also serve as access to the controls and status screen on the GoPros. The mounting piece of the housing was designed to mate with standard GoPro screw connectors allowing it to be used on a variety of GoPro harnesses.

This bracket was mounted to the Action Mounts Chest mount. One bracket was mounted to the front of the chest mount and using the included GoPro mounting pads the chest harness was modified to carry a bracket on the back plate as well. The back plate of the Chest Harness was relatively small and when examining the footage taken during test runs the rear camera pair was found to be very unstable. The following diagram shows the chest harness.

Figure 1.4: Action Mounts chest mount showing the front mounting plate

The need to increase the stability without hindering wearability introduced a new specification to be incorporated into the design process. The part had to comfortably fit to the back of a runner while providing a larger surface area for the camera to mount too. The part shown in the following figure was created and tested to fulfil this specification.

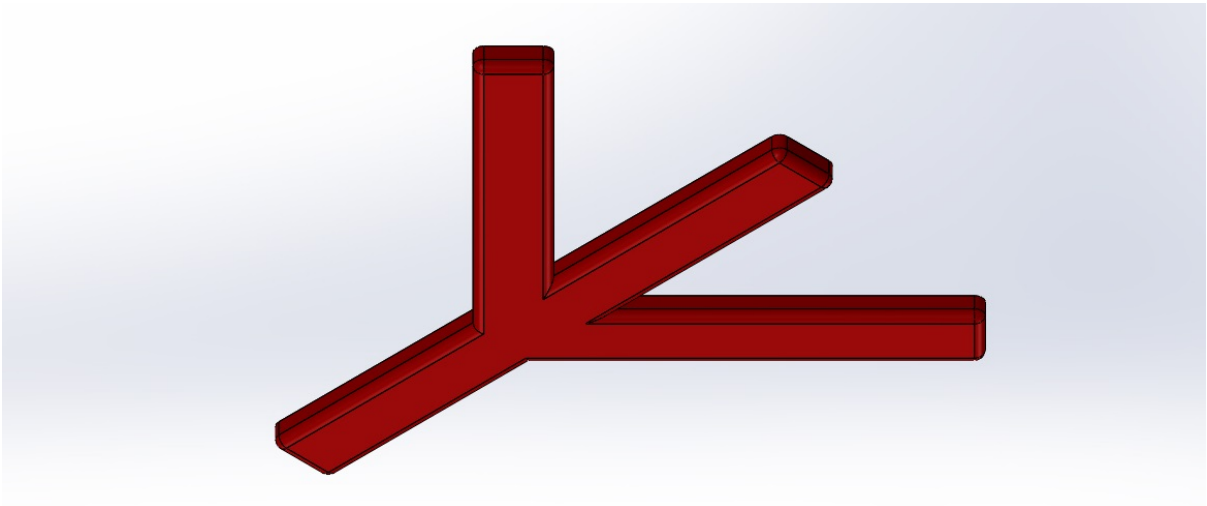

Figure 1.5: Stabilizer mounted to the rear camera

The stabilizer was fastened to the back of the harness backplate. The following figure shows stabilizer mounted below the rear camera.

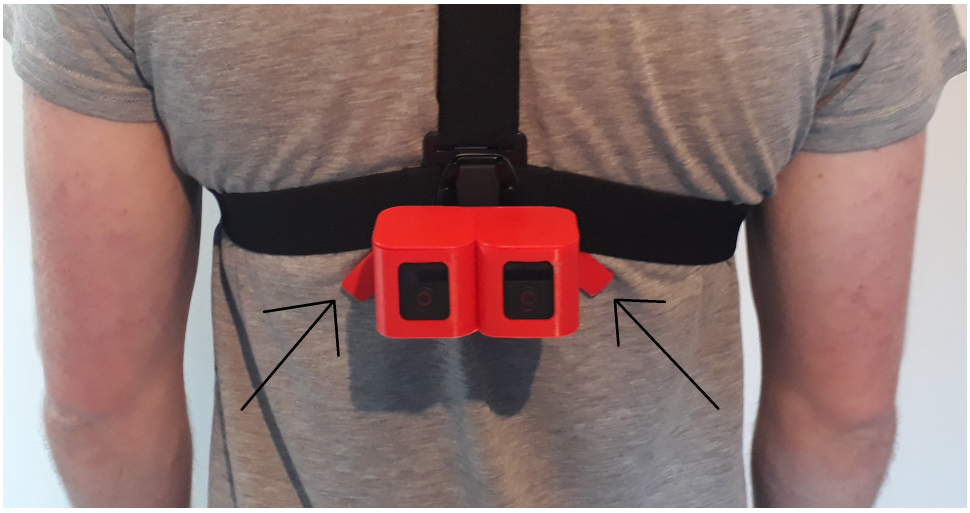

Figure 1.6: Stabilizer mounted to the rear camera

When comparing footage of the back cameras before and after the stabilizer was attached a clear difference is noted. The videos before contained excessive movement to the extent that the image blurred the lower limbs. This would prove problematic for attempts at image processing as the relatively low resolution and blurry images would introduce large uncertainties in the marker locations. After interviewing subjects wearing the data capture system did not reduce the comfortability of the system.

1.3 Sony Xperia Z3 Compact Smartphone

The Sony Xperia Z3 Compact has a complex sensor system that includes an accelerometer, gyroscope, magnetometer, light sensor, pressure sensor, and proximity sensor. These sensors are combined locally to also create artificial sensors for orientation, gravity ad linear acceleration. A built in GPS is also available to provide positional data.

To log the different datapoints generated by the sensors a software application for the smartphone was needed. Many free applications are available on the Sndroid Marketplace to fulfil this purpose; largely due to the open source approach the Android operating system is built on. By checking user reviews and ratings the following applications were considered.

1.3.1 AndroSensor

Androsensor was created by Fiv Asim and is availible freely from the Android marketplace or [3]. The source code for this application is not available but the author details the various underlying methods on [3]. The following are screenshots of the application running on the Z3 Compact.

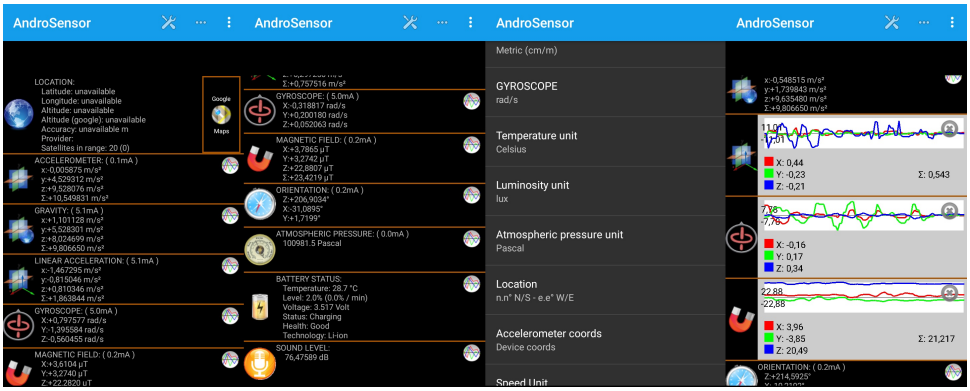

Figure 1.7: Stabilizer mounted to the rear camera

One of the advantages of AndroSensor is that it is highly configurable. A user can select which sensors are monitored and logged and the rate at which they are updated. This allows us to log important data and eases pre processing efforts. The application can also log the magnitude of sound allowing us to use this to synchronize the entire system. The output file type and units of the sensors can also be configured also easing pre processing

The application has some drawbacks

1.3.2 SensorLab

Sensor

advantages disadvantages

1.3.3 Custom Software

A custom software solution was developed by Stocks in [4] to log the various sensors of the smartphone. This application was developed to include readings from external sensors as well as the internal sensors of the smartphone. The source code for the application is available and

Figure 1.8: Stabilizer mounted to the rear camera

Figure 1.9: Stabilizer mounted to the rear camera

When the source code of the application was compiled using android studio the application did not work

an attemp was made to modeify the source code to correct the application but proved impossible due to the large codebase and lack of documentation.

1.3.4 Final Selection

AndroSensor was selected over the over application due to the above mentioned advantages. The application was configured to log the accelerometer, gyroscope, magnetometer, gps location, and sound level. The logging rate was set at 100Hz to have uniformity in the

Heading	Fields
ACCELEROMETER	X,Y,Z
GRAVITY	X,Y,Z
LINEAR ACCELERATION	X,Y,Z
GYROSCOPE	X,Y,Z
MAGNETIC FIELD	X,Y,Z
ORIENTATION	X,Y,Z
ATMOSPHERIC PRESSURE	
LOCATION Latitude	
LOCATION Longitude	
LOCATION Speed	
LOCATION ORIENTATION	
VOLUME	

Table 1.2: This table shows the different headings of the output IMU file

system.

1.4 Smartphone Mount Design

The design specification was to rigidly mount the smart phone to the chest of the subject. To complete this objective a rubber smartphone housing for the Z3 compact was fastened to the ActionMounts chest mount above the front duel camera. The following image illustrates the smartphone as connected to the harness.

At if the smartphone had been mounted on top of the camera we could have used pose estimation to understand exactly the orientation of the camera

this is a big assumption and problem

Figure 1.10: phone mount

1.5 Critical Point Markers

To increase the accuracy of the image processing elements of this methodology various colourful markers were used to identify critical points on the subjects lower extremities. The following picture shows the location of the markers.

Figure 1.11: Subject wearing Green and Pink Markers to identify critical points

From the image we can see that green markers were used to identify the toe edge of the runner in the front camera frame and pinks marker to identify the end of the thigh in the same frame. In the rear mounted camera frame pink markers were used to identify the heel of the subject and green markers to identify the centre of the calf. These markers in conjunction with the model is used to identify the lower limb orientation during the fusion stage.

These luminous green and pink markers were chosen because they are bright in most lighting conditions and offer a stark contrast to both the subjects clothing and skin colour. They also contrast a typical black tar road surface where the runs were performed. This contrast makes them easy to detect using feature detection as they cause a spike in the color decomposition of the image.

Chapter 2

Processing the Captured Data

This chapter is dedicated to the process of extracting critical data from the video files and IMU .csv file. This process will take the raw captured data and transform it to quantified values that we can feed the Extended Kalman Filter.

2.1 Processing the IMU Data

The following flowcharts depicts the procedural processing of IMU data gathered from the chest mounted smartphone running the AndroSensor application. The following flow diagram shows the different steps in preprocessing.

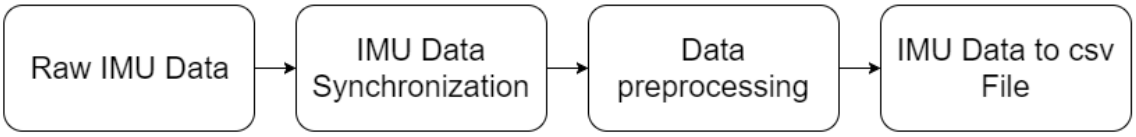

Figure 2.1: Diagram showing the progression and dependence of the major stages of video processing in the project

From the diagram we can see that our input is a large unprocessed dataset captured by the Smartphnone and logged at 100Hz. From this critical data must be extracted and synchronized with the rest of the system. The data must also undergo pre processing to simplify the implementation of the EKF.

2.1.1 Obtaining IMU Data

To use the smartphone as an IMU a free application "AndroSensor" [3] was installed. This application could log different parameters from all the different sensors that the smartphone contains. The application also gives the user the ability to change the output data file, log rate and what sensors are logged.

The sampling rate and logging rate of the application was configured to 100Hz as this was the maximum.

THe following image is a screen shot taken from within the AndroSensor application. It shows the main screen of the application and the ability to see live plots of the active sensors.

The IMU data logged by the smartphone was saved as a .csv file with the first row containing the headings of various variables. These headings are summarized in the table below.

All these variables have been recorded with respect the smartphone frame of reference as shown below

2.1.2 Synchronizing IMU Data

The AndroSense application can record the the magnitude of sound that the microphone is experiencing. Using this magnitude we can see at what sample the spike from the clap was recorded. This sample will be a common point between the IMU and the cameras and can thus be used to synchronize the different hardware elements.

2.1.3 Preprocessing IMU Data

Before we can apply the IMU data directly to the EKF we need to make some minor modifications to the data. This is critical in removing any bias from the sensors. as discussed in the Sensor calibration section.

in order to extract the linear acceleration from the data we can simply take the measure acceleration and subtract the gravitational acceleration vector

2.1.4 Exporting IMU Data

The IMU data was finally exported as a csv final and imported into MATLAB as different vectors. This allows for faster processing and better data processing in matlab.

2.2 Processing the Video Data

To once again simplify the design and implementation of the EKF it is important to pre-process teh video data to a leess data heavy format. The following digram shows the process of converting a data heavy video file to a more lightweight .csv (Comma Separated Values) file whilst amintaining all critical information.

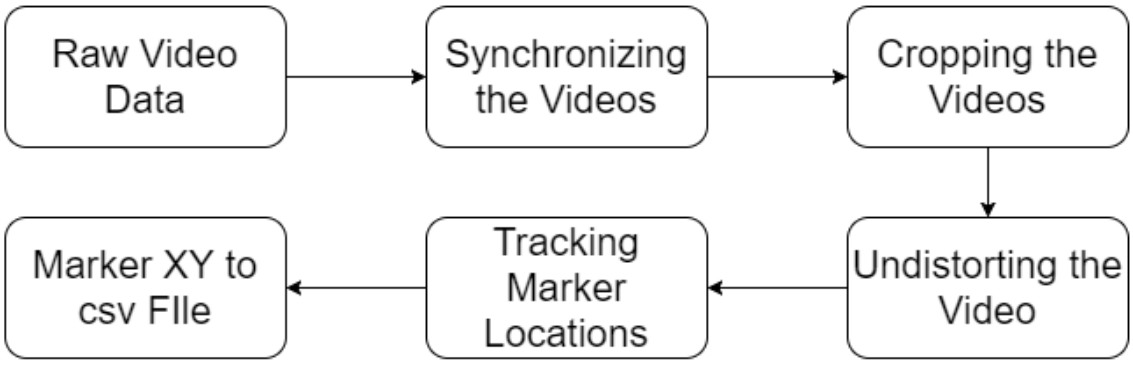

Figure 2.2: Diagram showing the progression and dependence of the major stages of video processing in the project

2.2.1 Obtaining Video Data

Using the chest mounted cameras detailed in the previous chapter we can generate raw video data. The GPHS cameras can be configured to record at different frame rates and resolutions as discussed in the previous chapter. The video files where stored in an .MP4 format. This meant that during recording the video was compressed and the

2.2.2 Synchronizing Video Sources

I typical problem faced when working with different sources of data is that of synchronization. Since this project used 4 different cameras, synchronizing the video sources are critical to generate accurate stereo vision data.

The problem of synchronization was overcome by using a audio cue to align the video data post capturing. With all systems recording, a simple hand clap can serve as a spiking audio input easily identified in the audio track of the video streams. The frame associated with this audio spike can be identified using SVP (Sony Vegas Pro) video editing software as shown in the figure below.

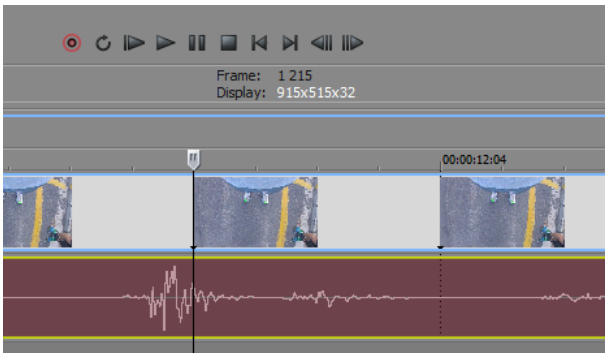

Figure 2.3: Figure showing the user interface of SVP video editing software

The red track in the above figure shows the recorded audio stream while the corresponding frames are displayed in the blue track above that. The cursor is aligned with the audio spike caused by the clap with the corresponding frame number displayed below the playback controls.

This method was repeated for every video stream such that a common starting point was generated.

2.2.3 Cutting Critical Video Data

With the video data synchronized the next step was to generate a subset of video demonstrating a transient period and steady state period of running. From accelerometer readings we can easily determine the gait cycle period of our subject; that is the amount of time take between the same foot impacting the ground. These impacts are visible as spikes as seen in the accelerometer data.

2.2.4 Undistorting the Video Data

To generate accurate distances using stereo vision the video frames need to be undistorted.

Distortion of the frames is a result of the

To gain further understanding of undistorting video files [2] served as a reference. In this

work Hartley describes various methods of undistorted images. These distortions are due to various lens effects.

2.2.5 Tracking and Exporting Marker Positions in the Frame

This section discusses the different methods of feature detection subdividing them into two main methodologies: automated and semi-automated. Each of these approaches offer advantages and disadvantages. To understand the approaches considered for this work it is important to visualize the input image data to the system. The following picture shows the various frames from all the cameras.

Figure 2.4: 4 Frames from the different cameras combined

The top row of images have been generated by the front mounted cameras and the bottom row by the rear mounted cameras. The left images were produced by the left cameras and the right images by the right cameras. From this image we can get a visual idea of the data our image processing system needs to manipulate.

Feature detection is a methodology in image processing that allows critical points of an image to be detected. This extracts an XY coordinate of the point of interest that can then be used to track the movement and position of object in the image. This methodology can be applied with frames with more than one point of interest as shown in this work. Multiple points of interest in multiple video sources does however increase the complexity of feature detection.

An initial approach of using automated detection was considered and three possible system were considered. Using a trained neural network, using an edge detection algoritm with a panning search algorithm and finally using a colour identifying algorithm paired with a local search algorithm.

In theory a well trained **neural network** will provide the most robust and accurate system to identify features in varying lighting condition. It would also be the most accurate methodology for a system without the markers. This is due to the *understanding* developed by the neural network after sufficient training data is processed.

This however presents a key difficulty in setting up and training a neural network. The most important element of a neural network is the training data used to teach the. This training data needs to have annotated images with metadata fields to

A neural network therefore cannot generate training data and a method of generating training data needs to be considered in any case.

The second approach taken was that of **edge detection**. This method

The final approach considered and used was to **semi automatically** label critical point in the image using a toolbox created by Hedrick et al. [5]. This software allows for a semi automatic tracking of points of interest in the video frames. While this method is labour intensive it is arguably more accurate than the previously investigated methodologies. The following figure shows the functionality of the toolbox within MATLAB.

By using this toolbox we can generate a dataset of (x, y) pixel coordinates by identifying the different markers on each frame and assigning their corresponding points to them. The following tables shows the markers and their corresponding point.

Front Cameras	
Point 1	Right Knee
Point 2	Left Knee
Point 3	Right Foot
Point 4	Left Foot
Rear Cameras	
Point 1	Right Calf
Point 2	Left Calf
Point 3	Right Heel
Point 4	Left Heel

Table 2.1: Table showing the relation between points and markers

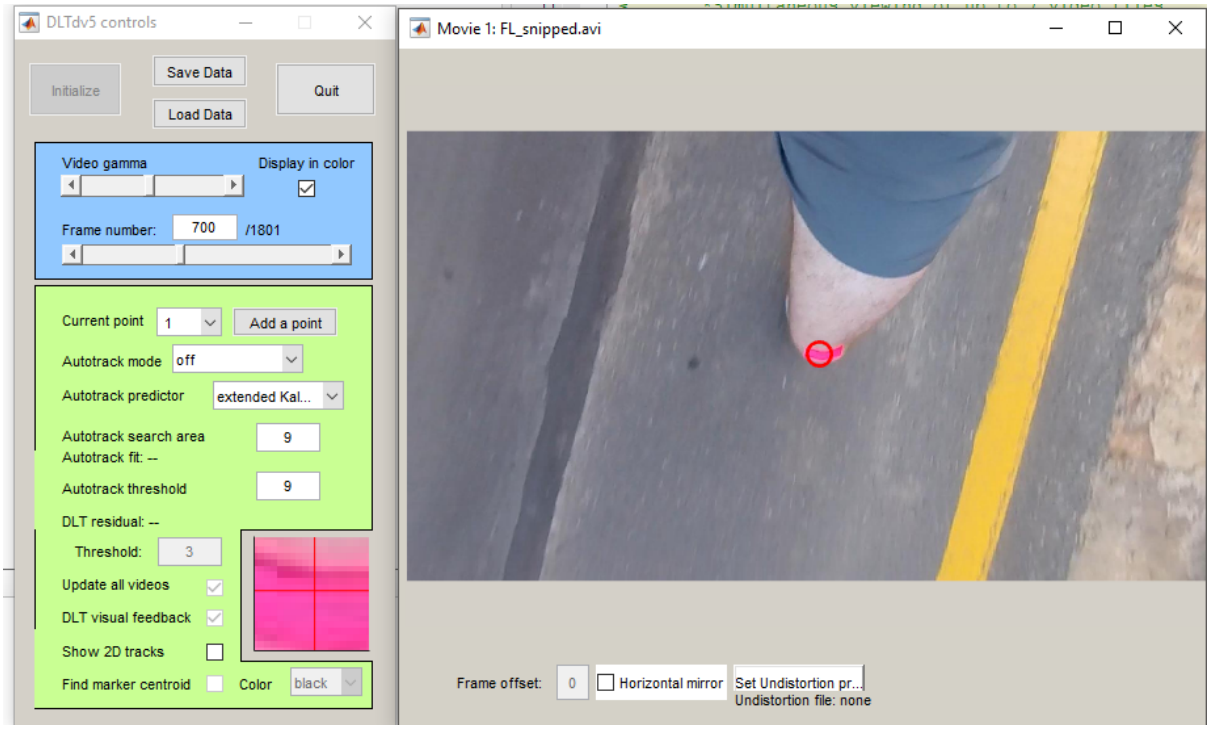

Figure 2.5: Using the dltdv toolbox in MATLAB to export marker coordinates

2.3 On units

In order to assure consistency between the different sources of data considering a general set of units is of critical importance. It was decided to implement the system using SI units such that all lengths was given in meters and all angles in radians.

gyroscope was logged as radians per second,

accelerometer was logged as meters per second .

imgae data was recorded as m

2.4 On Frames of Reference

we need to developed some commonalities among the various sensors and their individual frames of reference. The body frame

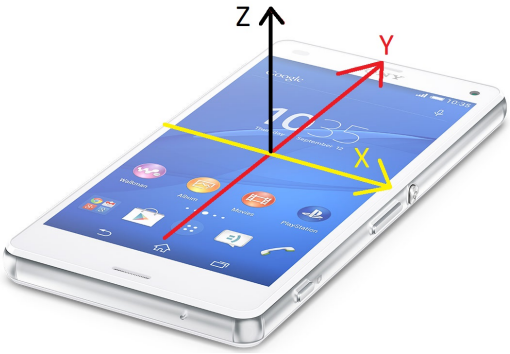

Figure 2.6: Figure demonstrating the frame of reference of the smartphone

Chapter 3

Data Fusion and State Estimation

This chapter is dedicated the explaining the mathematical methods and models used to fuse data generated by the cameras and IMU.

A good starting point for this chapter would be to define the states of interest of our system. These states are predicted by EKF using various measurements relating directly and indirectly to them. The following tables specifies the parameters used to symbolize the states of the system.

State	Description
x_{body}	x Position of body w.r.t. the inertial frame
y_{body}	y Position of body w.r.t. the inertial frame
z_{body}	z Position of body w.r.t. the inertial frame
ϕ_{body}	Roll of body w.r.t. the inertial frame
θ_{body}	Pitch of body w.r.t. the inertial frame
ψ_{body}	Yaw of body w.r.t. the inertial frame
θ_{LH}	Pitch of left thigh w.r.t. left hip
ψ_{LH}	Yaw of left thigh w.r.t. left hip
θ_{LK}	Pitch of left calf w.r.t. left knee
θ_{LA}	Pitch of left foot w.r.t. left ankle
θ_{RH}	Pitch of right thigh w.r.t. right hip
ψ_{RH}	Yaw of right thigh w.r.t. right hip
θ_{RK}	Pitch of the right calf w.r.t. right knee
θ_{RA}	Pitch of the right foot w.r.t. the right ankle

Table 3.1: Table showing the different states of the model to be determined by the EKF

Our system is not only concerned with these positional and angular elements, but also how they change over time. These rates where defined as the derivative of the states with respect to time. The first derivative serving as velocity and angular velocity, while the second derivative serving as acceleration and angular acceleration. Here the vector x

serves as a element of our state vector X.

$$\mathbf{x} = [x_{body}\ y_{body}\ z_{body}\ \phi_{body}\ \theta_{body}\ \psi_{body}\ \theta_{LH}\ \psi_{LH}\ \theta_{LK}\ \theta_{LA}\ \theta_{RH}\ \psi_{RH}\ \theta_{RK}\ \theta_{RA}]$$

The vector x contains 14 elements.

$$\mathbf{X} = [\mathbf{x}\ \dot{\mathbf{x}}\ \ddot{\mathbf{x}}]$$

From this it is clear that our state vector X contains 42 elements.

3.1 Prediction Equations

The fundamental assumption made when deriving the prediction equations for our states was that the acceleration (both linear and angular) was constant between sampling intervals. It would therefore stand that for the positional states of the filter ($x_{body}\ y_{body}\ z_{body}$) and the angular states (θ_{body} ... θ_{RA}) could be predicted using:

$$\ddot{p}_{k+1} = \ddot{p}_k + \sigma_{\ddot{p}}^2$$
$$\dot{p}_{k+1} = \dot{p}_k + \ddot{p}_k T + \sigma_{\dot{p}}^2$$
$$p_{k+1} = p_k + \dot{p}_k T + \sigma_p^2$$

for the positional states and:

$$\ddot{\alpha}_{k+1} = \ddot{\alpha}_k + \sigma_{\ddot{\alpha}}^2$$
$$\dot{\alpha}_{k+1} = \dot{\alpha}_k + \ddot{\alpha}_k T + \sigma_{\dot{\alpha}}^2$$
$$\alpha_{k+1} = \alpha_k + \dot{\alpha}_k T + \sigma_\alpha^2$$

for the angular states.

These prediction equations where created in MATLAB using symbolic functions. The sigma associated each equations takes into account the prediction uncertainties contained in the Q matrix. By adjusting these values we can increase the filter performance. This is discussed further in a later section.

3.2 Measurement Equations

The measurement equations for the lower limbs where generated using inverse kinematics. This requires a model

3.2.1 Euler Matrices

The following matrices are the rotational matrices for rotating a point in 3D space along a certain axis.

$$Roll(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

$$Pitch(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$Yaw(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3.2.2 Direct Cosine Matrix

3.2.3 Inverse kinematics

front

right knee

$$p1xyz = bodyY + bodyZ + R1 * Thigh$$

left knee

$$p2xyz = bodyY + bodyZ + R1 * Thigh$$

right foot

$$p3xyz = bodyY + bodyZ + R1 * Thigh + R2 * Calf + R3 * Foot$$

left foot

$$p4xyz = bodyY + bodyZ + R1 * Thigh + R2 * Calf + R3 * Foot$$

back
right calf

$$p1xyz = bodyY + bodyZ + R1 * Thigh + R2 * 0.5 * Calf$$

left calf

$$p2xyz = bodyY + bodyZ + R1 * Thigh + R2 * 0.5 * Calf$$

right heel

$$p2xyz = bodyY + bodyZ + R1 * Thigh + R2 * Calf$$

left heel

$$p2xyz = bodyY + bodyZ + R1 * Thigh + R2 * Calf$$

3.3 Understanding the Data Sources

It is important to understand the different parameters that mathematically quantify cameras. These parameters can be devided into *extrinsic* and *extrinsic*. Extrinsic camera variables related to the cameras position in the inertial frame and the direction the camera is facing. These can be summarized by the extrinsic camera matrix

$$[R \,|\, \boldsymbol{t}] = \left[\begin{array}{ccc|c} r_{1,1} & r_{1,2} & r_{1,3} & t_1 \\ r_{2,1} & r_{2,2} & r_{2,3} & t_2 \\ r_{3,1} & r_{3,2} & r_{3,3} & t_3 \end{array}\right]$$

3.4 State Estimation

This section will mathematically explain the Kalman filter and its implementation in this project.

Process equation of the kalam filter. from states to emasurements

$$X_{k+1} = FX_k + w_k$$

our state, contained in the vector X can be estimated by applying the process matrix F to our current known state. the term w is the noise variable that accounts for process noise.

Measurement equations from measurements to states.

$$Y_k = H_k X_k + v_k$$

w will be contained int he matrix Q

while v will be contained in the matrix R1

linearizing nonlinear system we get the EKF

3.5 Q Matrix, R Matrix and Initialization

This section will discuss the final components of the EKF namely the Q matrix containing the various process noise variations, R matrix containing the various measurement noise variances and the initial state values.

3.5.1 Defining the Q Matrix

From the above equations we can see that the Q matrix must have dimensions of $n * n$, where n in this equation is the total amount of states. As previously defined in this section our filter operates over 42 states, giving Q a size of $42 * 42$. All the variance parameters will be contained on the diagonal of the matrix with all other entries being zero.

To find the initial values for these uncertainties the the derivative of the various accelerations where taken. The maximum element from that set was selected as the uncertainty parameter.

3.5.2 Defining the R Matrix

The R matrix relates to the measurement variables and must therefore have size of $m*m$, where m is the amount of inputs the EKF. These are elements of the sensors themselves

and can be found by researching the relative data sheets for the smartphone IMU. As for the cameras a relatively large uncertainty of about 5 pixels was assumed.

3.5.3 Choosing Initial States

The subject was stationary during the initial stages of the run. This allows us to initialize our state vector with all states initially zero. The filter will therefore track the transience and steady state estimation of a running subject, giving insight into the filters performance under different conditions.

3.5.4 Initializing the Covariance Matrix P

following on from the previous section zeroing the initial states of the system allows us to have a relatively small initial covariance matrix due to the relative certainty we have that the states are truly zero.

Bibliography

[1] M. Christensen, "Gopro hero4 session," https://grabcad.com/library/gopro-hero4-session-1, [Online; accessed 6-October-2017].

[2] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[3] F. Asim, "Androsensor," https://play.google.com/store/apps/details?id=com.fivasim.androsensor&hl=en, [Online; accessed 10-October-2017].

[4] B. Stocks, "Cheetah motion analysis," Master's thesis, University of Cape TOwn, South Africa, 2016.

[5] T. L. Hedrick, "Software techniques for two-and three-dimensional kinematic measurements of biological and biomimetic systems," *Bioinspiration & biomimetics*, vol. 3, no. 3, p. 034001, 2008.