



12-2017

Dynamic *In Vivo* Skeletal Feature Tracking Via Fluoroscopy Using a Human Gait Model

William Patrick Anderson
University of Tennessee, Knoxville, wander21@vols.utk.edu

Follow this and additional works at: https://trace.tennessee.edu/utk_graddiss



Part of the [Applied Mechanics Commons](#), [Biomechanical Engineering Commons](#), [Electro-Mechanical Systems Commons](#), and the [Robotics Commons](#)

Recommended Citation

Anderson, William Patrick, "Dynamic *In Vivo* Skeletal Feature Tracking Via Fluoroscopy Using a Human Gait Model. " PhD diss., University of Tennessee, 2017.
https://trace.tennessee.edu/utk_graddiss/4730

This Dissertation is brought to you for free and open access by the Graduate School at TRACE: Tennessee Research and Creative Exchange. It has been accepted for inclusion in Doctoral Dissertations by an authorized administrator of TRACE: Tennessee Research and Creative Exchange. For more information, please contact trace@utk.edu.

To the Graduate Council:

I am submitting herewith a dissertation written by William Patrick Anderson entitled "Dynamic *In Vivo* Skeletal Feature Tracking Via Fluoroscopy Using a Human Gait Model." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Mechanical Engineering.

William R. Hamel, Major Professor

We have read this dissertation and recommend its acceptance:

Jindong Tan, D. Caleb Rucker, Hairong Qi

Accepted for the Council:

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Dynamic *In Vivo* Skeletal Feature Tracking Via Fluoroscopy Using a Human Gait Model

A Dissertation Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

William Patrick Anderson
December 2017

Copyright © 2017 by William Patrick Anderson
All rights reserved.

DEDICATION

This dissertation is dedicated to my parents who supported me, my siblings who commiserated with me, and my friends who distracted me.

ACKNOWLEDGEMENTS

I would like to express my thanks and appreciation to my advisor, Dr. William R. Hamel, for his encouragement, support, and guidance. In addition, I wish to also thank my committee, Dr. Jindong Tan, Dr. D. Caleb Rucker, and Dr. Hairong Qi, for their understanding and flexibility during this lengthy processes. I would not have been able to complete this work without the aid, ideas, and even occasional distractions of those in the BioRobotics Lab at the University of Tennessee. Thank you everyone!

I would also like to thank my entire family and close friends for giving me encouragement when things seemed to start going wrong, which felt all too frequent. I do not believe I would have been able to see this project through to completion without them.

ABSTRACT

The Tracking Fluoroscope System II, a mobile robotic fluoroscopy platform, developed and built at the University of Tennessee, Knoxville, presently employs a pattern matching algorithm in order to identify and track a marker placed upon a subject's knee joint of interest. The purpose of this research is to generate a new tracking algorithm based around the human gait cycle for prediction and improving the overall accuracy of joint tracking.

This research centers around processing the acquired x-ray images of the desired knee joint obtained during standard clinical operation in order to identify and track directly through the acquired image. Due to the inability for tracking through x-ray imaging during knee crossovers (when both knees enter and align within the x-ray image), a form of prediction is developed around the kinematics of human gait motion. This gait model is designed to consider the natural swinging motion of the knee during walking in order to predict path for the x-ray system to follow when active tracking is not possible. During the later stages of research, modifications were made in the setup and testing in order to accommodate changes put in place upon the research environment.

Individually, the processing of the x-ray images and the prediction ability of the gait model have shown decent success. The overall controlling algorithm which manages the tracking system has demonstrated some downfalls, however, which have been attributed to the modified setup of the testing. Therefore, while the final results of this research demonstrated some shortcomings, it has confirmed its usability in a real-time environment with the capability of tracking the complete joint implant, and the human gait model developed provides a means of accounting for the natural swing motion of the knee joints during leg motion. The end results

provide evidence for a feasible system should it be possible to test and employ it in the scenario to which it was first intended, i.e. in conjunction with x-ray images.

TABLE OF CONTENTS

CHAPTER ONE – INTRODUCTION	1
CHAPTER TWO – BACKGROUND AND PRIOR RESEARCH	3
ROBOTICS	3
THE TRACKING FLUOROSCOPE SYSTEM.....	4
TRACKING.....	5
HUMAN GAIT MODELING	7
CHAPTER THREE – FUNDAMENTAL CONTRIBUTIONS	10
MOTIVATION.....	10
CONTRIBUTIONS	10
CHAPTER FOUR – ANTICIPATED PROGRESSION OF RESEARCH	12
PHASE ONE – IMAGE PROCESSING.....	12
<i>Edge Detection</i>	12
<i>Particle Analysis</i>	15
PHASE TWO – HUMAN GAIT MODELING	18
PHASE THREE – CONTROL ALGORITHM	20
CHAPTER FIVE – EXPERIMENTAL PLATFORM AND COMPONENTS	22
EXPERIMENTAL PLATFORM	22
SYSTEM COMPONENTS	23
<i>LASER Scanners</i>	23
<i>X-Ray System</i>	24
<i>Schunk Axis Drives</i>	27
<i>ASUS Wireless Router</i>	28
<i>Coding Platform</i>	29
<i>Control Computer</i>	29
CHAPTER SIX – EXPERIENCED PROGRESSION OF RESEARCH	31

PHASE ONE – IMAGE PROCESSING.....	31
<i>Edge Detection Method</i>	32
<i>Particle Analysis Method</i>	35
PHASE TWO – HUMAN GAIT MODELING	48
<i>Change in Expected Model’s Development</i>	53
PHASE THREE – CONTROL ALGORITHM: SUPERVISORY KNEE TRACKING ALGORITHM.....	57
<i>Data Communication</i>	59
<i>Image Analysis</i>	60
<i>Positioning Commands/Transitioning</i>	63
RGB-D SENSOR USAGE.....	64
<i>Hardware</i>	65
<i>Software</i>	66
MAJOR CHANGE UPON EXPERIMENTAL PLAN	68
<i>Testing of the Human Gait Model</i>	68
<i>Emulating X-Ray Images for SKTA Testing</i>	70
<i>SKTA Testing</i>	74
CHAPTER SEVEN – RESULTS OF RESEARCH	76
IMAGE PROCESSING	76
PREDICTION: NONE VERSUS HEURISTIC VERSUS HUMAN GAIT MODELING	82
SKTA PERFORMANCE	88
CHAPTER EIGHT – CONCLUSIONS AND FUTURE WORK	101
IMAGE PROCESSING	101
HUMAN GAIT MODELING	102
SUPERVISORY KNEE TRACKING ALGORITHM	103
DIFFERENCES BETWEEN VISUAL AND X-RAY PARTICLE ANALYSIS	104
FUTURE RECOMMENDATIONS.....	105

LIST OF REFERENCES	106
APPENDICES	110
APPENDIX A – ADDITIONAL PHASE ONE RESULTS’ FIGURES	111
APPENDIX B – TFS CODE	113
APPENDIX C – SPECIFIC TESTING CODE	149
VITA	160

LIST OF TABLES

Table 1: Comparison of Curve, HOG, and LBP feature extractions for execution speeds.	33
Table 2: Comparison of execution speeds and curve extraction count between three operations as performed on a basic x-ray sample image.	36
Table 3: Comparison of execution speeds and curve extraction count between three operations as performed on a smoothed x-ray sample image.	36
Table 4: Summary of execution speed testing between the Version 1 pattern matching (V1 PM), the Version 2 (V2 PM), and the particle analysis (PA). All speeds are given in milliseconds.	77
Table 5: Percentage difference in identified locations between pattern matching image frames for the three prediction methods. Demonstrates the general magnitude of the required axis motion between individual frames that were studied.	85
Table 6: Percentage difference in identified locations between particle analysis image frames for the three prediction methods. Demonstrates the general magnitude of the required axis motion between individual frames that were studied.	87

LIST OF FIGURES

Figure 1: Tracking Fluoroscope System as seen from multiple angles.	22
Figure 2: Diagram of the TFS setup illustrating the connections between systems of interest.	23
Figure 3: LASER Scanners (blue boxes) mounted onto a bar to allow for height adjustments to each subject.	24
Figure 4: Toshiba X-Ray Emitter with distance standoff rod.	25
Figure 5: PerkinElmer Dexela Digital Flat Panel Imager with a Basler RGB camera mounted beneath.	26
Figure 6: Tap configuration of the Dexela Flat Panel Imager's output pixel stream. N.B. – Taps III and IV are flipped 180 degrees respective to taps I and II.	27
Figure 7: Horizontal and Vertical Schunk Axis Drives for x-ray system motion.	28
Figure 8: Edge detection operation with curve extraction of the edge data. A) Original sample image. B) Canny ED curve extraction results. C) Prewitt ED curve extraction results. D) Sobel ED curve extraction results.	37
Figure 9: Edge detection operation after a single-pass Gaussian smoothing operation. A) Gaussian smoothed original image. B) Canny ED with extracted curves. C) Prewitt ED with extracted curves. D) Sobel ED with extracted curves.	38
Figure 10: Final sequence of image processing steps. A) Original image. B) Borders removed. C) Size reduced by half. D) Thresholding operation performed. E) Pixel Cleaning operation performed. F) POpen operation performed, final operation before analysis.	39
Figure 11: A) X-ray image received from Flat Panel (972 x 768 pixels). B) Image with 50 pixel borders extracted (922 x 718 pixels). C) Reduced image size (461 x 359 pixels).	42
Figure 12: A) Thresholded x-ray image. B) Pixel Cleaning operation. C) POpen operation.	47
Figure 13: Original subject representation for angle and segment lengths of equations. N.B. - Positive for x and y-directions are right and down, respectively.	51
Figure 14: Original TFS representation for direction and positioning. N.B. - Positive directions for x and y-directions are right and down, respectively.	52

Figure 15: Modified subject representation for angle and segment lengths of equations. N.B. – Only change between present and prior diagram is the positive direction of the vertical axis which is now upwards..... 55

Figure 16: Modified TFS representation for direction and positioning. N.B. – Changes from prior diagram are the positive vertical direction, upwards, and the perspective of the TFS machine. 55

Figure 17: Diagram displaying positioning values and variables for modeling equations..... 56

Figure 18: System diagram demonstrating the flow of data from the Flat Panel (top-left), through the image processing and analysis, updating of the gait model (and prediction calculations if necessary), and output of the motion commands to the axis control to obtain motion of the x-ray system. 58

Figure 19: System diagram for the Human Gait Model's active tracking versus prediction. 59

Figure 20: Image energy profile of a single gait cycle. Spikes in the energy level represent the moments of crossover. 61

Figure 21: Sample sequence of binary x-ray images demonstrating the increase in energy during a crossover event. The auxiliary knee moves through the images in an upward vertical direction. 62

Figure 22: The removed FP bracket with the Basler black-and-white camera rigged to the central position within the bracket..... 69

Figure 23: White sheet draped over x-ray emitter axis to simulate the light background of an x-ray image..... 72

Figure 24: Back view of the white sheet draped over the x-ray emitter axis..... 72

Figure 25: Example differences between x-ray and visual images in terms of indistinct edges after binary conversion. A) Obtained x-ray image. B) Thresholded x-ray image. C) Obtained visual image. D) Thresholded visual image. 73

Figure 26: System diagram of the alternate setup using visual images rather than x-ray. 75

Figure 27: Positioning comparison on the first Gait Set between the three tracking methods in image coordinates. A) Image chosen for testing, frame 11 of 210, with visually identified

positioning commands from image center. B) TFS Version 1 pattern matching results. C) TFS Version 2 pattern matching results. D) Particle Analysis results.....	78
Figure 28: Positioning comparison on the second Gait Set between the three tracking methods in image coordinates. A) Image chosen for testing, frame 9 of 150, with visually identified positioning commands from image center. B) TFS Version 1 pattern matching results. C) TFS Version 2 pattern matching results. D) Particle analysis results.....	78
Figure 29: Positioning comparison on the third Gait Set between the three tracking methods in image coordinates. A) Image chosen for testing, frame 111 of 140 with visually identified positioning commands from image center. B) TFS Version 1 pattern matching results. C) TFS Version 2 pattern matching results. D) Particle analysis results.....	79
Figure 30: First test sequence comparing the pattern matching and the particle analysis algorithms to identify crossovers. N.B. - Lower lines represent time spent predicting motion.	81
Figure 31: Second test sequence comparing the pattern matching and particle analysis algorithms to identify crossovers. N.B. - Lower lines represent time spent predicting motion.	81
Figure 32: Third test sequence comparing the pattern matching and particle analysis algorithms to identify crossovers. N.B. - Lower lines represent time spent predicting motion.	82
Figure 33: Image sequence depicting the motion results of not having any prediction algorithm. The image(s) listed in red represents an instant without direct tracking.	84
Figure 34: Image sequence depicting the motion results for using the basic heuristic prediction method.....	84
Figure 35: Image sequence depicting the motion results for using the Human Gait Model for prediction.	84
Figure 36: Particle analysis testing results for having no predictive algorithm.....	86
Figure 37: Particle analysis testing results for using the heuristic prediction method.....	86
Figure 38: Particle analysis testing results for using the gait model prediction method.	87

Figure 39: Full image sequence demonstrating the behaviour of the overall algorithm created for particle analysis tracking. The images listed on the table in red represent instances without direct tracking (considered part of the crossover).....	89
Figure 40: Encoder recorded motion of the three prediction methods (Gait Model, Heuristic, and No Prediction) employing Pattern Matching for tracking.	91
Figure 41: Closer look at the encoder recorded motion of the three prediction methods employing Pattern Matching for tracking. Looks at three regions of prediction between the 170 ms mark and the 300 ms mark.....	92
Figure 42: Closer look at the encoder recorded motion of the three prediction methods employing Pattern Matching for tracking. Looks at the three regions of predicted motion between the 300 ms mark and the 460 ms mark.	93
Figure 43: Encoder recorded motion of the three prediction methods (Gait Model, Heuristic, and No Prediction) employing Particle Analysis method for tracking without inclusion of the auxiliary knee within the binary image.....	95
Figure 44: Closer look at the encoder recorded motion of the three prediction methods employing Particle Analysis for tracking. Looks at the three regions of predicted motion between the 430 mS mark and the 700 mS mark.....	96
Figure 45: Closer look at the encoder recorded motion of the three prediction methods employing Particle Analysis for tracking. Looks at the three regions of predicted motion between the 900 ms mark and the 1010 ms mark.	97
Figure 46: Encoder recorded motion of the three prediction methods (Gait Model, Heuristic, and No Prediction) employing the Particle Analysis methodology.....	99
Figure 47: Closer look at the initial predictive regions of the Particle Analysis method. Displays the first three regions between the 130 ms mark and the 224 ms mark.....	100
Figure 48: Processing speed results for the pattern matching and particle analysis algorithms on the first sequence set of x-ray images.	111
Figure 49: Processing speed results for the pattern matching algorithms and the particle analysis algorithm on the second sequence set of x-ray images.	111

Figure 50: Processing speed results for the pattern matching and particle analysis algorithms on the third sequence set of x-ray images.....	112
Figure 51: Main Control interface for the Tracking Fluoroscope System.....	113
Figure 52: Main Control code for the TFS. Panel 1-1.....	114
Figure 53: Main Control code for the TFS. Panel 1-2.....	114
Figure 54: Main Control code for the TFS. Panel 1-3.....	115
Figure 55: Main Control code for the TFS. Panel 1-4.....	115
Figure 56: Main Control code for the TFS. Panel 2-1.....	116
Figure 57: Main Control code for the TFS. Panel 2-2.....	116
Figure 58: Main Control code for the TFS. Panel 2-3.....	117
Figure 59: Main Control code for the TFS. Panel 2-4.....	117
Figure 60: Tracking Code interface for the TFS.....	118
Figure 61: Tracking code for the TFS.....	118
Figure 62: Camera Tracking interface.....	119
Figure 63: Camera Tracking code. Panel 1-1.....	119
Figure 64: Camera Tracking code. Panel 1-2.....	120
Figure 65: Camera Tracking code. Panel 2-1.....	120
Figure 66: Camera Tracking code. Panel 2-2.....	120
Figure 67: Pattern Matching interface.....	121
Figure 68: Pattern Matching code. Panel 1-1.....	121
Figure 69: Pattern Matching code. Panel 1-2.....	121
Figure 70: X-Ray Image Display interface.....	122
Figure 71: X-Ray Image Display code. Panel 1.....	122
Figure 72: X-Ray Safety & Timer interface.....	123

Figure 73: X-Ray Safety & Timer code. Panel 1-1.	123
Figure 74: X-Ray Safety & Timer code. Panel 1-2.	124
Figure 75: Axis Command interface.....	125
Figure 76: Axis Command code. Panel 1-1.....	125
Figure 77: Axis Command code. Panel 1-2.....	126
Figure 78: Axis Command code. Panel 2-1.....	126
Figure 79: Axis Command code. Panel 2-2.....	127
Figure 80: Axis Command code. Panel 3-1.....	127
Figure 81: Axis Command code. Panel 3-2.....	128
Figure 82: Axis Command code. Panel 3-3.....	128
Figure 83: Homing Sequence interface.....	129
Figure 84: Homing Sequence code. Panel 1.	129
Figure 85: MIWD Command interface.....	130
Figure 86: MIWD Command code. Panel 1.	130
Figure 87: MIWD Command code. Panel 2-1.....	130
Figure 88: MIWD Command code. Panel 2-2.....	131
Figure 89: MIWD Command code. Panel 3.	131
Figure 90: Data Communication interface.....	132
Figure 91: Data Communication code. Panel 1.	133
Figure 92: Data Communication code. Panel 2-1.....	134
Figure 93: Data Communication code. Panel 2-2.....	135
Figure 94: Laser Processing interface.....	136
Figure 95: Laser Processing code. Panel 1-1.....	136
Figure 96: Laser Processing code. Panel 1-2.....	136

Figure 97: Laser Processing code. Panel 2.	137
Figure 98: Laser Processing code. Panel 3-1.	137
Figure 99: Laser Processing code. Panel 3-2.	137
Figure 100: Laser Processing code. Panel 4-1.	138
Figure 101: Laser Processing code. Panel 4-2.	138
Figure 102: Laser Processing code. Panel 5-1.	138
Figure 103: Laser Processing code. Panel 5-2.	139
Figure 104: Flat Panel interface.	139
Figure 105: Flat Panel code. Panel 1A.	139
Figure 106: Flat Panel code. Panel 1B-1.	140
Figure 107: Flat Panel code. Panel 1B-2.	140
Figure 108: Flat Panel code. Panel 1C-1.	141
Figure 109: Flat Panel code. Panel 1C-2.	141
Figure 110: Flat Panel code. Panel 1C-3.	142
Figure 111: Flat Panel code. Panel 1D-1.	142
Figure 112: Flat Panel code. Panel 1D-2.	143
Figure 113: Flat Panel code. Panel 1D-3.	143
Figure 114: 2nd Computer Communications/Processing interface.	144
Figure 115: 2nd Computer Communications/Processing code. Panel 1.	145
Figure 116: 2nd Computer Image Processing interface.	146
Figure 117: 2nd Computer Image Processing code. Panel 1.	146
Figure 118: Operator Computer Data Receiving interface.	147
Figure 119: Operator Computer Data Receiving code. Panel 1-1.	147
Figure 120: Operator Computer Data Receiving code. Panel 1-2.	148

Figure 121: Modified Tracking code for testing.....	149
Figure 122: Modified Camera Tracking code for testing. Replaces code in Panel 2-2.....	149
Figure 123: Prediction Control interface.....	150
Figure 124: Prediction Control code. Panel 1.....	150
Figure 125: Prediction Control code. Panel 2-1.....	150
Figure 126: Prediction Control code. Panel 2-2.....	151
Figure 127: Knee Location Calibrator interface.....	151
Figure 128: Knee Location Calibrator code. Panel 1.....	152
Figure 129: Knee Locator interface.....	152
Figure 130: Knee Locator code. Panel 1-1.....	153
Figure 131: Knee Locator code. Panel 1-2.....	153
Figure 132: Modified X-Ray Image Display code for testing. Replaces code in Panel 1.....	154
Figure 133: Data Collection code added to Main Control code. Panel 1-1.....	155
Figure 134: Data Collection code added to Main Control code. Panel 1-2.....	156
Figure 135: Data Collection code added to Main Control code. Panel 1-3.....	157
Figure 136: Operator Computer Data Receiving interface for testing.....	158
Figure 137: Modified Operator Computer Data Receiving code for testing. Replaces code in Panel 1-1.....	158
Figure 138: Modified Operator Computer Data Receiving code for testing. Replaces code in Panel 1-2.....	159

LIST OF ABBREVIATIONS

TFS	– Tracking Fluoroscope System
FPGA	– Field Programmable Gate Array
ED	– Edge Detection
OS	– Operating System
RTOS	– Real-Time Operating System
NI	– National Instruments
VI	– Virtual Instrument
IMAQ	– IMage AcQuisition
HOG	– History of Oriented Gradients
LBP	– Linear Binary Patterns
LUT	– Look-Up Table
DLL	– Dynamic Link Library
SKTA	– Supervisory Knee Tracking Algorithm
TCP/IP	– Transmission Control Protocol/Internet Protocol
NS	– Network Stream
ToF	– Time-of-Flight

CHAPTER ONE – INTRODUCTION

Over the past several decades, robotics research has continued to evolve and grow, expanding into new areas of interest. While science fiction literature and cinema have painted many a variety of robots existing within our future, we still have not yet achieved such artificial independence from human cognition. However, with each new area unlocked, another potential avenue opens up for the future of this vast field in engineering.

One particular area of robotics that is of great interest is a robot's ability to perceive different forms of data and utilize it for further decision-making processes. This ability would allow a robot to interact and adapt to their situation just as a human can. Given the correct data and algorithms, a robot would even be able to make judgements about probable outcomes based on the perceived data. This is comparable to the way humans can make "judgement calls" about various situations.

In the capacity of medical diagnostics and research, a machine's ability to make such decisions based upon perceived data could not only prove invaluable, but critical. A computer (robot) would be able to detect and adjust itself courtesy of such algorithms on the order of micro to millisecond scale as opposed to a human operator likely requiring seconds. This rapid, and ideally accurate, response could signify the difference between ongoing correct data acquisition and errors of some form developing (non-usable data, computer control issues, etc.).

A machine called the Tracking Fluoroscope System, also referred to as the TFS, is a robot that is able to provide fluoroscopic video of a subject's hip or knee joint while they perform various dynamic activities by tracking the joint in question. When the TFS is not able to actively find the target, however, the system must have a method that would allow it to reacquire the target while ideally continuing to obtain correct data. At present, it does not contain such a

method. The purpose of this research is to generate a new tracking algorithm based around the human gait cycle that accounts for this missing component and, ideally, improving the overall accuracy of joint tracking.

This generated algorithm will need to be capable of managing and processing a continuous stream of x-ray images immediately to obtain target tracking information. Such processing must occur on the order of micro to milliseconds in order to prevent the target from moving out of the x-ray frame, and it will need to identify when the image frames are no longer viable for direct tracking and instead provide alternate tracking information to continue the necessary motion for data acquisition.

In the ensuing chapters, pertinent background and prior research is explained followed by the contributions this research will provide. Then, the design concepts will be explained in Chapter Four, and Chapter Five provides more detailed information about the research platform's setup. Finally, Chapters Six, Seven, and Eight provide information about executing the conceptual designs, the results from it, and the conclusions pulled from them, respectively. A list of references and the Appendix can be found at the end of this work.

CHAPTER TWO – BACKGROUND AND PRIOR RESEARCH

In this chapter, information is provided into the background of the four primary areas of interest for this dissertation. The first provided is concerned with robotics in general while the second will provide a brief explanation of the Tracking Fluoroscope System that is the central platform for this research. The additional two sections are devoted to sensor-based tracking and gait modeling. Incorporated into these background synopses are brief summaries of previously performed research which was reviewed in the given areas.

Robotics

The word “robot” was first introduced to science fiction in Rossumovi Univerzální Roboti, [1], a play written by Karel Čapek in 1920. The word’s original implication of *roboti*, referencing the Czech word *robota* or forced laborers, was used to mean artificial people that would labor on behalf of humanity. Since this play was released, the meaning of this word, *robot*, has evolved to match the changing technology of the age. As defined by Sciavicco and Siciliano nearing the end of the twentieth century in [2], a robot is a machine that can take the place of a human and perform various tasks either repetitive, menial, or both. Present day provides a further broadening of the term’s usage to also include the idea of an artificial machine capable of performing decision-making processes about the tasks it is executing.

The use of robots in medical scenarios, [3], has grown due to the high degree of accuracy that a computer controlled apparatus can offer over a human hand and eye. This does not mean, however, that a robot alone is capable of performing complex surgery; instead, they are being used to assist doctors and medical practitioners with their surgical procedures. [4-8] are all vision-based controlling methodologies developed to aid in surgery bringing a higher precision to said techniques. These methods provide controlling algorithms such as auto-positioning of

needles or other tools by computer usage of x-ray/fluoroscopy imaging or camera lenses to perform visual servo control. However, these forms of visual servoing are designed more on the interests of precision rather than speed and so can be permitted to take a second or two to process and calculate before continuing motion. For the interests of this research, visual servoing is a central component but execution and control will need to be on the magnitude of micro to milliseconds rather than seconds in order to maintain awareness of the target joint while in motion.

The Tracking Fluoroscope System

The conceptual design for the robotic machine called the Tracking Fluoroscope System (TFS) was first written out and reviewed in [9] with a U.S. patent [10] being awarded in 2013. Its premise is to provide fluoroscopic data of a subject's lower limb joints (knees will be the focus for this dissertation) while the subject performs activities and maneuvers reminiscent of everyday living. [11] is a detailing of its construction while [12-14] provide explanations regarding several control methodologies employed on the original robot. [13] focused on the overall motion control of the TFS, and [14] focused on modeling and then controlling the dynamics of the machine. [12] touched more in-depth into the visual servoing control for the TFS' joint tracking system.

There are two forms of tracking built into the TFS, only one of which is employed with visual servoing. Both of these tracking systems work concurrently to provide overall tracking of the subject within the robot, but it is only the visual servo tracking that is of interest for this dissertation. [15] provides more information regarding the concepts for the non-visual servoing form if desired while [16] covers its implementation. More importantly, however, it also covers the concept and implementation of the visual servoing-based tracking.

This second form is a way of tracking the subject's knee joint through the x-ray images. In January of 2014, the Tracking Fluoroscope System was upgraded with new, more powerful components and restructured to broaden some of its capabilities. Among the new equipment introduced with this upgrade were more powerful linear drive motors for the axis positioning as well as a flat panel digital imaging device for the x-ray system. To accommodate them, alternate intermediary components such as amplifiers and FPGA computer boards were also included requiring modifications to several of the controlling code sequences for proper interfacing. And so, while the knowledge and understanding obtained from the work done in [16] endures, the specifics are no longer applicable.

Tracking

Tracking is the act of following something either whole or in part. A well-known example of tracking would be a hunter following prey, and this analogy holds comparatively for tracking applications with technology. A software-based example would be following the motion of a target in a video while a physical-based one would be a robot performing self-adjustments to its orientation to keep a target within its sensor range. Both of these forms use a method commonly referred to as *object tracking*. Security applications [17], gait modeling studies [18], and even medical research [19] are a few additional examples that employ this form of tracking.

Object tracking is the application of software and sensors, commonly camera-based sensors, to first locate and then follow a target of interest through the sequence of images making up a video stream. The software algorithms are commonly a set of different image processing techniques such as those explained in [20] applied in sequence that will identify and indicate the location of the target. For non-stationary visual sensors, this identified location can further be

used for servo controlling the camera's position/orientation, [21]. The performance requirements for this form of tracking will be dependent upon how this tracking is to be employed.

Most known applications for object tracking allow for it to be done in what is known as "off-line" processing or after the video has been captured and stored. [22-24] each use a set of techniques that allow them to follow a desired target from one frame to the next for a stationary camera. [22] is based almost exclusively around image processing while [23] uses a technique to "teach" the tracking algorithm what target to identify and follow. [24] takes advantage of stereovision's dual camera system to provide 3-dimensional data for target identification from one frame to the next. These techniques would be limited in their applicability for this research, however, as a non-stationary camera will be employed. The greatest downside to all of them is their inability to perform in real-time. Their algorithms and processes are either too intensive for "online" application, or they are specifically designed to work on pre-existing video files.

On the other hand, the capability to execute in real-time can be a necessity for such things as security and data collection, but this capability will also limit what can be done computationally. [25-28] developed methodologies that can manage the real-time processing; however, they are restricted by the fact that their visual sensors are stationary. [26] actually has both the target being tracked and the robot using the tracking information in the same view field. [27] circumvents the stationary sensor constraint by utilizing a network of stationary cameras to continue tracking the target of interest, but a camera network is not feasible for the research being performed in this dissertation work. Of greater interest are [29-32] which operate both in real-time and with non-stationary camera sensors similar to the platform setup that will be used in this research.

[29, 30] both present techniques for tracking motion within image sequences while the camera sensor itself is in motion. Through continuous adjustments to the camera sensor's position and orientation, their techniques stabilize the image's background using prominent features between subsequent frames providing a relatively constant "landscape" such to detect motion constrained within said background's field. On the other hand, [31] takes advantage of the Microsoft Kinect to perform 3-dimensional detection of the target and its motion while also on a mobile platform itself. Lastly, [32] developed a technique for tracking airborne targets that relies on depth detection from the camera for accuracy. While each of these techniques presents potential avenues, they are not directly applicable due to this research's scenario being centered around tracking through x-ray images.

More detailed information regarding x-ray techniques and this research's setup can be found in Chapter Five, but in regards to applying these aforementioned tracking techniques, there are a couple factors that prevent their usage. The first is a lack of distinguishing background features in an x-ray image while the second is a need for limited distance from the imaging device. Despite the extensive research performed in object tracking and its apparent straight forward applications, this research has presented a scenario more complex than expected due to image content (or lack thereof), camera distance, and camera motion. This has led to the need for augmenting the object tracking.

Human Gait Modeling

Gait modeling is the simulation of leg-based locomotion, commonly through the use of a system of mathematical equations. There have been many studies done regarding this particular aspect of biomechanics concerned with the musculoskeletal system of both animals and the human body. Dr. Gerald Loeb provides a nice introduction to the mechanics of human gait in [33].

Bipedal locomotion (human walking) has been studied considerably in the biomechanics and medical fields with the aim of learning how a person should be able to move, and, especially for medical interests, what might cause problems with this natural motion. While [18, 34, 35] all studied the gaits of people walking in the interests of modeling the gait patterns being observed, [35] also looked at the differences between various groups (natural versus metronomic walking, child and adult, and aging/neurodegenerative diseases). A more complex gait model was deemed necessary after a review of these works demonstrated the simplistic nature of the gait pattern.

Additionally, a program called OpenSIM, [36], which is a full simulation program modeling musculoskeletal motion designed to aid researchers in analysis, was investigated for potential direct inclusion to the generated tracking algorithm. While this program provides the ability for complete modeling of the human body's motion capabilities, as a fully stand-alone program it would prove problematic attempting to connect it into a control algorithm and manage running in a real-time environment.

Of the set of gait modeling studies found, most concluded that a thorough modeling of the human gait would require understanding both the musculoskeletal system as well as the parts of the nervous system managing locomotion. [35, 37-39] focus on the interaction between the nervous system and the muscle control of the skeletal frame to generate motion. Although their results demonstrated greater accuracy in their modeling techniques, the computational requirements for tracking both neural control and the skeletal motion with the present technological capabilities of the TFS would exclude the possibility of performing this in real-time on its current embedded computer system.

In the interests of real-time application, more simplified equations centered around the basic kinematics would be needed which lead to the older [40, 41]. [41] focused on using

markers to identify key points on a walking subject and then calculates the extremities' kinematics from the video while [40] used anthropometric data to determine the inertial properties of the different human body segments. [42] is a more recent study performed in the interests of ergonomics that provided the anthropometric height-to-limb length ratio equations used in this research.

While the earlier studies performed on gait modeling centered primarily around simply analyzing human gait, the more recent studies have been more concerned with being able to simulate the human gait. Because of this shift in focus, the kinematic results produced from these later studies are fairly in-depth representations which commonly include the nervous system in their calculations. These forms of modeling provide equations detailing both the neuro-muscular responses as well as the kinematic motions of the skeletal frame. However, due to the intricacy of their entwinement, it is not possible to separate the two from each other to use their kinematics. This resulted in a need for developing a basic kinematic-level modeling rather than the complexities of a simulation-level modeling.

CHAPTER THREE – FUNDAMENTAL CONTRIBUTIONS

Motivation

The driving force behind this research was the need for an accurate and robust alternative form of tracking on the Tracking Fluoroscope System. The present form, which relies upon pattern matching, has a high potential for losing the target of interest within the x-ray image. During a knee crossover event (where the auxiliary knee moves past the primary knee being studied), there are multiple objects potentially similar enough to each other within the same image that the pattern matching algorithm is unable to obtain a positive match. Additionally, the bone of the primary joint of interest will become obstructed by the auxiliary's bone as the two align and overlap each other. Furthermore, should the pattern used for the matching algorithm not be precise enough for specific image frames, the pattern matching algorithms may not be able to acquire a match for the tracking to work.

Contributions

1. Development and evaluation of robust and high frame rate identification of skeletal features of interest suitable for tracking control purposes from medium scale resolution x-ray images.
2. Development and evaluation of dynamic tracking methods for skeletal features of interest, during natural movements, based on:
 - a. Feature identification tracking directly from fluoroscope image frames.
 - b. Human gait/maneuver modeling and observation for both primary and auxiliary knees.
3. Development of feature identification algorithms suitable for vision-based servo-tracking control that do not require specific templates.

4. New schemes to minimize leg crossover effects on vision-based servo-tracking control through medium scale resolution x-ray images:

- a. Robust feature identification
- b. “Opposite” leg kinematic awareness
 - Kinematic model plus state estimation
 - Direct kinematic sensing plus state estimation

CHAPTER FOUR – ANTICIPATED PROGRESSION OF RESEARCH

There will be three phases of this research. The first phase will focus on the image processing portion to generate usable images while the second phase will focus on generating a mathematical model for the human gait cycle. Finally, the third phase will be generating a control algorithm to allow the image processing and gait model to interact and provide servo commands for the x-ray system. This chapter will provide an overview of the expectations for each of these three phases in this research.

Phase One – Image Processing

The image processing will revolve around one of two avenues. The first involves edge detection and classification of the x-ray images to locate the target of interest while the second concerns using particle analysis with a potential logic tree for object identification. Their beginning stages will be investigated simultaneously in order to determine the most promising candidate among these two for further study and inclusion in the final algorithm. The metric for this decision will fall first to their execution speed. If they cannot process fast enough to employ in real-time control, then even the most accurate information will be inapplicable as it would be outdated. After measuring their speed, the next criteria will be their accuracy.

Edge Detection

The first to be covered here will be the edge detection and classification. The method in which edge detection algorithms operate is by identifying significant changes in neighbouring pixel intensity values. These significant changes are then highlighted under the assumption that they represent an edge within the image's content. There are several edge detection algorithms available, but amongst them the most promising found during initial investigations were the

Prewitt, Sobel¹, and Canny² Edge Detections. The mathematics of these operations can be found after this section.

The reasons for these three are their more favorable results to sample x-ray images. With darkened backgrounds against lighter lines, there will be less data to manage, and while various settings can be manipulated to achieve different levels of results, they produced the most promising images in terms of outlining the target implant.

Therefore, the next steps will be to further test with changing settings in order to obtain as refined an image as possible. The goal being to attain an image with as clear a definition of the target implant within the x-ray image. From here, we perform some comparisons with the particle analysis testing, and if the edge detection method is not excluded yet, we will move into creating this method's classifier.

Under the assumption that Edge Detection has not been eliminated at this point from further investigation, a means of classifying the identified outlines within the x-ray image will need to be generated. This classifier can take multiple forms, but the main two of interest here will be one using a metric of dimensions to identify objects and another performing an abbreviated form of matching to locate specific shapes within the processed image. The deciding factor in which classifier to employ (or an alternative to these all together) will be the very criteria used for the overall methods as well, first their speed of execution followed by the classifier's accuracy.

¹ http://zone.ni.com/reference/en-XX/help/370281AC-01/imaqvision/imaq_edgedetection;
October, 2017

² http://zone.ni.com/reference/en-XX/help/370281AC-01/imaqvision/imaq_cannyedgedetection;
October, 2017

Prewitt, Sobel, and Canny Edge Detection

The Prewitt and Sobel edge detection operations are performed using the convolution equation with different kernels.

$$P_{out}(x, y) = \frac{1}{m^2} \sum_{u=0}^{m-1} \sum_{v=0}^{m-1} P_{in}(x + k - u, y + k - v) \cdot G(u, v) \quad (1)$$

$$k = \frac{m-1}{2}$$

P_{out} is the output pixel located in the image's x-y coordinate frame. LabVIEW defaults m (the size of the convolving kernel) to equal 3 which additionally defaults k as 1. G represents the kernel being convolved into the input image, P_{in} , with matrix coordinates u and v .

The difference between the Prewitt and Sobel edge detections mathematically is simply the kernel being used. Each possess two kernels, one to obtain horizontal edges and one for vertical. The Prewitt ED's kernel for the x- and y-directions are shown in (2).

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \quad G_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \quad (2)$$

Unlike the Prewitt, the Sobel ED kernel uses a two for its middle numbers within the sets as seen in (3).

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \quad G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad (3)$$

Convolving these kernels with the image array will result in two image, one from each of the kernels. To obtain a final image with all edges present, the magnitude between them is calculated:

$$P(x, y) = \sqrt{P_x(x, y)^2 + P_y(x, y)^2} \quad (4)$$

The end result of (4), P , is the completed image for the given edge detection operation.

Unlike the Prewitt or Sobel, the Canny Edge Detection is actually a sequence of steps developed in 1986 by John Canny in [43]. This sequence is approximated here into four steps:

1. Gaussian Smoothing operation
2. Edge Detection operation
3. Non-Maxima Suppression operation
4. Double Gradient Threshold/Hysteresis operation

The initial Smoothing operation is meant to mitigate out any noise that might cause edges to be generated. The Edge Detection operation is performed with any operator desired such as the Prewitt or Sobel among others. While the particle ED kernel used by LabVIEW was not identifiable due to LabVIEW's black box coding, what information was found hinted strongly towards the use of the Sobel. The Suppression operation is a method of "thinning" the obtained edges to roughly a single pixel in width transforming them into rough lines for aiding the final step. The Gradient Threshold measures the "strength" of the particular gradients by comparing them to specified limits while Hysteresis will completely suppress any pixel with a weak gradient (as determined by the Gradient Threshold operation) that does not contain a strong gradient pixel near it. In this way, edges expected to be caused by noise can be further filtered out.

Particle Analysis

While the edge detection process will be concerned with manipulating the ED's criteria to generate as ideal an image as possible, the particle analysis method will center around manipulating the image itself to generate a result focused primarily on the target joint of interest. This method is centered around the use of binary images which contain pixel values of either zeros or ones. The "particles" in this analysis are the groupings of three or more active pixels

(pixels with values of one). It is expected that this method will be capable of higher speeds of computation due to the use of binary images which possess far less data to process as compared to a standard greyscale image of the same size.

In order to obtain this binary image from the greyscale x-ray images, a thresholding technique will be employed that will convert the pixel values to either true (one) or false (zero) depending upon the original pixel value being compared to a target value. This is written as an equation in (5).

$$P_{out}(x, y) = \{P_{in}(x, y) \leq n\} \quad (5)$$

The result of the comparison operation between the input image's pixel value, P_{in} , and the threshold value, n , will either be 1 (True) or 0 (False), and this result will be assigned as the value for the output image's pixel, P_{out} .

To aid the thresholding process in translating as little of the target joint's surrounding tissue and superfluous background as much as possible, some prior processing of the x-ray images will be considered. Chief among these prior processes will be a manipulation of the greyscale image's pixel intensity values. By broadening the gap between the dark values of the implant from the darker grey values of the surrounding hard tissue as well as the lighter grey values of the softer tissue, it should aid the thresholding technique's ability to remove most of the non-implant material in the greyscale image.

There are several post thresholding operations that can also be performed to further "clean up" the binary image and refine it down to the target implant. One operation, Erode, is designed to remove a set amount of pixels from the edges of particles, completely removing any particle that is too small. This operation could prove invaluable in taking care of outlier pixels

that survive the thresholding operation as it should have only a minute impact upon the target itself. Another set of operations which may lend itself very well to this method, is the Particle Analysis operations set provided by LabVIEW's Vision Development code suite. This particular code set, which as the name implies is geared specifically for performing particle analysis operations, possesses several built-in functions which could aid in further cleaning up the binary image. Such operations include the ability to remove particles that do not meet specified criteria, as well as the ability to segment/label each of the identified particles within the image.

At this point, a comparison to the edge detection method will provide some insight into which method is showing greater promise for continued use. If the particle analysis method has not been excluded from continued investigation, the next steps would center around actually locating the target within the image. Depending upon the abilities of the Particle Analysis operations to segment/label the individual particles within the binary image, this particular step could already be complete. However, there are certain techniques that could be employed should this not prove the case.

The first is through the use of a logic tree to identify the particular particle of interest (the target joint's implant). This logic tree would classify the various particles based upon their general size, their location within the binary x-ray image, as well as their relative location to other particles within the image. This identification would then allow for determining the coordinates of the particle of interest.

On the other hand, there is a certain operation that might allow the direct computation of the target's location within the binary image. The operation in question, Centroid Calculation, is designed to calculate the center-of-mass of the image's active pixels. If the image could be refined enough to possess only (or near enough to) the target, then this operation could be

performed to directly determine a coordinate within the image frame of the target's central mass and therefore the location of the target for tracking.

Phase Two – Human Gait Modeling

The purpose of the human gait modeling is to provide a means of virtually tracking the desired target of interest. In this manner, should the actual target ever be lost due to an image ambiguity (any event in which the target can no longer be clearly identified), the system could switch to the virtual tracking and be capable of maintaining a high probability of reacquiring the target after the ambiguity ends. This means that the model will provide the expected path of the knee joint during the standard walking motion, and the computer will essentially be capable of simultaneously tracking the target in real-space with the actual subject as well as in virtual-space within the computer's processor with the model. Should the tracking algorithm ever lose the target, when the two knees cross each other for instance, the tracking algorithm could switch to the virtual tracking to continue the correct cyclical motion of walking and be able to switch back immediately after the target is reacquired with little discrepancy between the virtually predicted location and the actual location.

However, preliminary investigations into human gait modeling, as previously explained in Chapter Two, have shown that a set of simple equations, while possible, are not a likely outcome. Instead, what is more probable will be a series of equations that are constantly updating themselves using information provided by the real-time tracking. Therefore, the first step in developing this gait model will be to identify or develop a set of limb proportionality equations in terms of a limited number of input parameters, the most probable being the subject's height. This will allow for a degree of customization towards individual subjects. A person who is six

feet and three inches tall will have a broader gait than someone who is only five feet and six inches tall.

With these equations identified, the next step will be in determining which equations, if any, from prior research can be utilized and how. The in-depth equations provided by the research done involving both the musculoskeletal system as well as the nervous system possess the potential for more accurate predictions of the target location. Yet, they would only be usable if the two systems involved could have been separated from each other. As explained in the Chapter Two review, the TFS' computer system does not currently possess the computational capacity to perform such extensive calculations alongside the original control and management algorithms already taking place for system operations all in real-time. Thus, while these equations are not directly applicable, they may provide guidance in generating equations for use.

An alternative would be to investigate the OpenSIM software referenced in Chapter Two. OpenSIM is a full software program that was designed to allow a complete simulation of human motion. While the software itself is too extensive for use directly within a real-time tracking algorithm, it could potentially provide a means of generating equations which could be used by the system. Another potential would be to acquire a series of sample points using this software, and then developing equations to work with these saved points.

If the OpenSIM software proves unusable, the final course will be to construct the necessary equations directly. This avenue will have two potential options of its own. The first option will be to develop a set of pure kinematic equations to define the motion of the human body, and then input these equations into the tracking algorithm, relating them to the TFS' motion where necessary. This avenue provides the advantage of more fully defining the human

motion itself and increasing its probability of accuracy. On the other hand, as with the equations found in the literature, this carries the potential for a higher computational cost.

The other avenue to attempt will require identifying a common measurement point to use between the subject and the TFS. Once this is done, then two kinematic equations can be written. One will be the motion of the subject's target knee with respect to this given point while the second would be for the motion of the TFS' components to match that of the target. This avenue will have the potential for a lower computational cost, but it also carries the possibility of lower accuracy from the potential for developing errors.

Phase Three – Control Algorithm

The final phase of this research will be centered around an overall control system that will contain both the image processing sequence designed in Phase One as well as the gait modeling equations created in Phase Two. The first and most basic function of this algorithm will be to provide the necessary information needed for both the image processing and the gait modeling to execute, as well as transferring the results of each to where they are required for further operations of the TFS. This controlling algorithm will also contain the necessary logic to determine when to switch between the actual tracking of the subject and employing the gait model for predictive motion control of the x-ray system.

The criteria for switching can be a multitude of options, primarily dictated by the progression of the previous phases of the research. As an example, if edge detection is employed for the image processing, the logic check could be based upon how many items the classifier has identified within the image. Too many identified items, and it could be assumed that a crossover (the most likely cause for losing tracking) is beginning. On the other hand, if particle analysis is pursued than a total count of active pixels may provide the needed trigger for identifying a

crossover occurring. Therefore, the first steps to creating this control system will simply be to ensure that all necessary information is being passed to where it should as well as continuously checking against the criteria for ambiguity for transitioning to the gait model.

Whether the gait model is being employed alongside a filter such as the Kalman Filter for prediction purposes or it is a standalone predictor, it will need the real-time information of the target's location in order to provide as accurate a prediction as possible. There may also need to be a sub-system to aid in switching between the different forms of tracking (active versus predictive). It will be expected that the gait model will not provide perfect location coordinates as compared to the real target. There is the potential for "jumping" to occur when transitioning back from the prediction tracking to the actual tracking of the target. To minimize this jump in locations, a safety mechanism will likely be needed to help smooth this transition over several milliseconds.

Additional to the image processing and gait modeling, the control algorithm may also host a supplementary 3D sensor such as the Microsoft Kinect, a time-of-flight camera, or even stereovision cameras. The purpose of this extra sensor would be to provide a third perspective of the target to allow for additional comparisons with the actual versus predicted locations. Should this sensor be included, the controlling algorithm will also need to be capable of managing the data from this sensor and provide the means by which to compare all three of the data streams.

CHAPTER FIVE – EXPERIMENTAL PLATFORM AND COMPONENTS

Experimental Platform

The platform for this research will be the Tracking Fluoroscope System (TFS), and it will center around a few specific systems contained onboard. While the TFS (refer to Figure 1) requires many systems in order to successfully operate, only those of interest to this research will be introduced. The TFS will be placed into its Knee Configuration for the purposes of this experimentation. This entails *a priori* knowledge of specific operational settings and component configurations that will provide a basis for measurement.

A diagram of the system setup is provided in Figure 2 for ease of reference.



Figure 1: Tracking Fluoroscope System as seen from multiple angles.

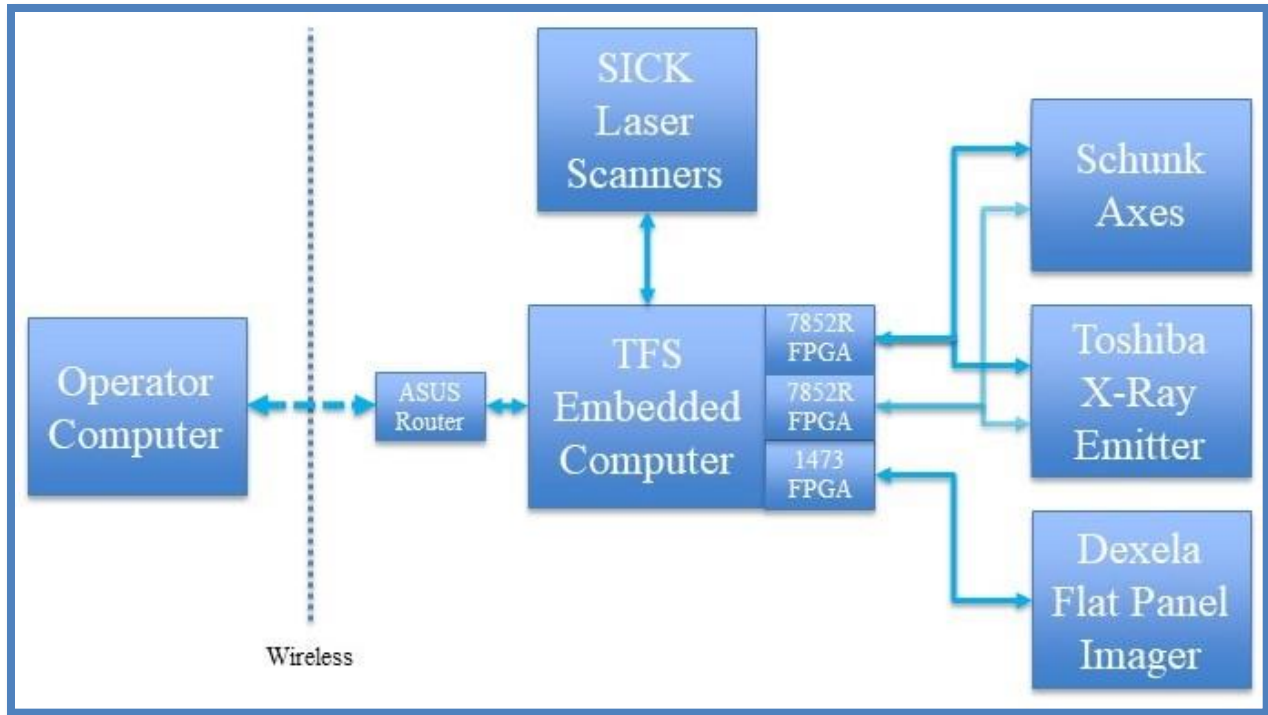


Figure 2: Diagram of the TFS setup illustrating the connections between systems of interest.

System Components

Listed below are the systems necessary for a tracking algorithm to operate. Each of these systems are shown and explained below:

LASER Scanners

The distance between the subject walking within the Tracking Fluoroscope System and the TFS itself must be measured to determine if and when the TFS should move. Manufactured by SICK Sensor Intelligence³, two ML400 Series sensors (blue boxes seen in Figure 3) are used on the TFS. Networked together using TCP/IP and connected to the TFS' main computer via dedicated

³ <https://www.sick.com/us/en/>; February 2017

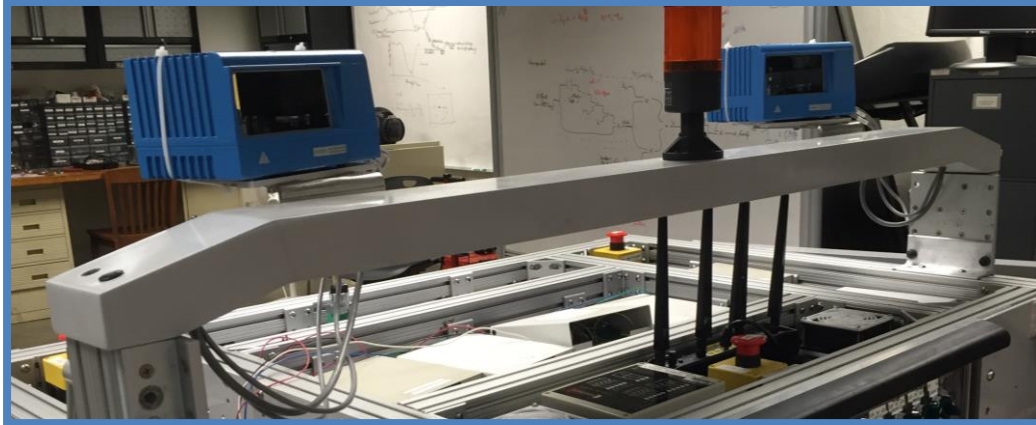


Figure 3: LASER Scanners (blue boxes) mounted onto a bar to allow for height adjustments to each subject.

router, the lasers output the subject's position with respect to the TFS carriage. This measured distance is compared to the desired distance, and the difference is relayed through the TFS' wheel control algorithm governing the carriage motion.

X-Ray System

There are two components to the TFS' onboard x-ray system. The x-ray emitter generates and directs the x-ray energy into a guided beam while the x-ray imager takes this energy and converts it into a viewable image. While the first version of the TFS was constructed using components of a traditional C-Arm hospital-style unit, the current version of the TFS that will be used in this research possesses a more customized system.

There is a particular factor that must be mentioned in terms of the x-ray system and its utilization concerned with producing usable images. An x-ray image is created through a shadowing effect with the generated x-ray energy. The denser the material being x-rayed, the less energy can pass through and hit the imager, and therefore the darker the associated pixels.

Because of this, if a subject stands directly in the middle between the emitter and the imager, the resultant image will be different than if the subject were to stand closer to one or the other. For this reason, standard x-ray techniques require the subject to be placed as close to the imaging device as possible. This allows the consistent creation of clear, crisp x-ray images.

Toshiba X-Ray Emitter

The first component of this customized setup is a Toshiba X-Ray Emitter (see Figure 4). Capable of utilizing up to 5 mA and 150 kVp to penetrate material, our system is limited by the TFS' onboard power supply. This restricts the emitter to a maximum 2.5 mA and 120 kVp settings.



Figure 4: Toshiba X-Ray Emitter with distance standoff rod.

Dexela Flat Panel Digital Imager

In order to transform the emitted x-ray energy into usable images, the second component of the x-ray system is a PerkinElmer⁴ Dexela series Flat Panel Digital Imager (please refer to Figure 5). The Dexela CMOS 2923 utilizes a four tap configuration demonstrated in Figure 6 to output the digital pixel information through a Camera Link (CL) connection. The digital imager streams this pixel data through the CL connection at rates reaching as high as eighty-six frames per second when using the smallest binning configuration. The images are then reconstructed primarily upon the Imaging FPGA Computer card (refer to the *Control Computer* section below)

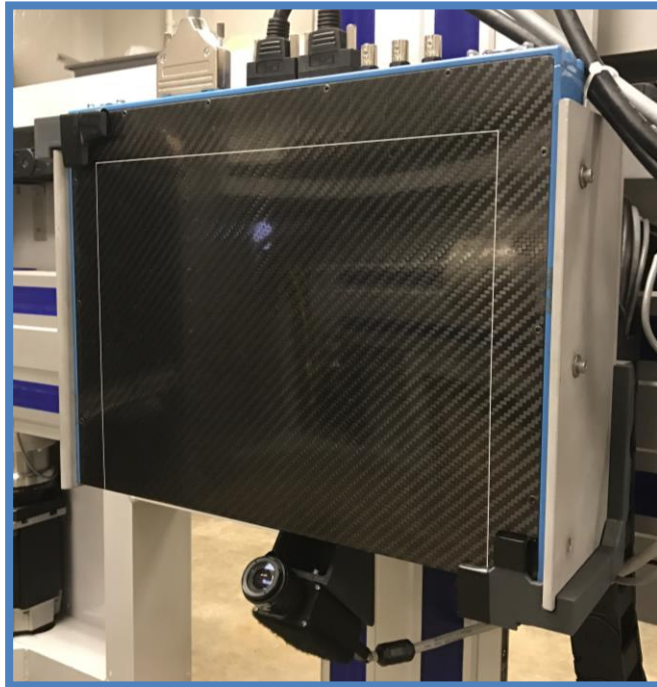


Figure 5: PerkinElmer Dexela Digital Flat Panel Imager with a Basler RGB camera mounted beneath.

⁴ <http://www.perkinelmer.com>; February 2017

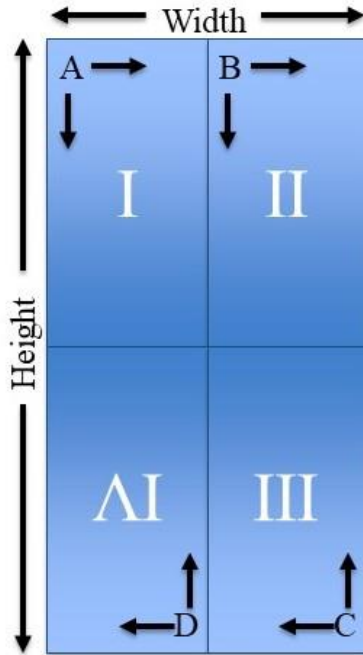


Figure 6: Tap configuration of the Dexela Flat Panel Imager's output pixel stream. N.B. – Taps III and IV are flipped 180 degrees respective to taps I and II.

as well as within the main code of the TFS.

Schunk Axis Drives

In order to maintain positioning of the x-ray system upon the target of interest during the trial maneuvers, vertical and horizontal axis drives, as shown in Figure 7, are used to provide the necessary motion. A combination of the HSB Automation⁵ Delta 145-SSS series linear belt-driven axes and Parker Hanifin⁶ MPP Rotary Servo series motors, these linear drive systems

⁵ <http://www.hsb-automation.de/en.html>; February 2017

⁶ <http://www.parker.com>; February 2017



Figure 7: Horizontal and Vertical Schunk Axis Drives for x-ray system motion.

provide the X and Z-motion for the x-ray emitter and the digital flat panel imager.

ASUS Wireless Router

A wireless router is used to transmit operator commands and data between the TFS and the Operator's computer. This router is an Asus AC 2400 RT-AC87U gigabit router. It is a dual band router capable of both a 2.4 gigahertz and a 5 gigahertz bandwidth. This router is mounted upon the TFS and hardwired into the TFS' embedded computer.

Coding Platform

The entirety of the TFS' control code is written in the LabVIEW Programming code language characterized by its ease in programming for parallel processing. This programming environment was created, offered, and is maintained by the National Instruments Corporation⁷, based out of Austin, Texas. It is designed around the concept of graphically representing code as opposed to the iconic linear, text-based form. NI offers several modules to tailor the LabVIEW programming environment to specific needs, such as the Vision Development suite mentioned in Chapter Three.

Control Computer

The control computer for the TFS is a Dell Precision T7600 Workstation. Possessing a total of sixteen cores, 256 gigabytes of solid-state hard drive space, and 16 gigabytes of RAM, the workstation computer also possesses three field-programmable-gate-array (FPGA) cards. Two of these FPGA cards are generic interaction forms while the third is specialized for Visual/Image processing. All three of these cards were purchased from National Instruments.

To execute the control code, the original operating system that came with the computer was replaced. The LabVIEW Real-Time Operating System (RTOS) is designed to provide the programmer with the ability to control execution priority of different code segments, capitalizing on LabVIEW's multi-threading capability, and it allows programmers to easily deploy and run written LabVIEW code.

⁷ <http://www.ni.com>; February 2017

On-Board FPGA Cards

All FPGA coding was done using the LabVIEW programming language and its FPGA code suite.

PCIe-7852R FPGA cards (2)

These two cards used together manage the majority of the signal control for the TFS. They provide the primary interface between the TFS' control code and the amplifiers and wiring/hardware. Among these components, the axes and the x-ray system are routed through these cards.

PCIe-1473-LX110 FPGA card

This card is used for interfacing with the Dexela Flat Panel Imager. While passing command sequences between the control computer and the imager, it also contains preliminary re-stitching operations to transform the incoming pixel stream into coherent images.

CHAPTER SIX – EXPERIENCED PROGRESSION OF RESEARCH

In this chapter is the development of the research presented in Chapter Four. While there was no distinct separation between completing one phase and starting the next, the three phase presentation used in Chapter Four will be maintained here for the purpose of clarity.

Phase One – Image Processing

Having prior experience focused mainly in the mechanical discipline, the image processing phase had been anticipated to be the quickest and easiest portion of this research. LabVIEW offered an entire suite of Image Processing code pieces, so it should ideally have been a simple matter of figuring out which code pieces were needed and in what order, connect them in the proper sequence, and the image processing phase of the research would be complete. Experience to the contrary, this segment took the longest to reach a usable sequence.

The reason for this is that, even with the limitations on what could be expected within a given x-ray image, there is still enough variation between images that what might work well for one image will not necessarily work for the next image. To best accommodate every x-ray frame that must be processed for tracking, two things had to occur. The resultant image processing sequence would need to work on a slightly more general scale than originally desired, and the x-ray images would need to be “normalized” as much as feasibly possible to provide more consistency between the individual image frames.

The first matter was to finish the initial testing of the Edge Detection and the Particle Analysis methods. Once one was determined to be the more viable option, the more thorough investigation could then begin.

Edge Detection Method

As mentioned previously, the Prewitt, Sobel, and Canny Edge Detection algorithms demonstrated an end result with a more visually distinguishable implant within the x-ray image. The goal from here was to determine if it would be possible to quickly and accurately identify the target implant within the image. This would require image segmentation to separate out the different parts of the image.

In LabVIEW, this operation would best be accomplished using *IMAQ Extract Curves* VI⁸, or potentially with either *IMAQ Extract HOG Feature Vector*⁹ or *IMAQ Extract LBP Feature Vector*¹⁰ VIs. The Extract Curves VI would be the most comprehensive one to use as it provides extensive data regarding every curve that is identified and quantified within the image. However, because this measures only identified curves rather than full features, the image would need to be filtered down to just the desired curve or curves (those of the target implant) or the results would need to be sorted through to identify the curves associated with the target of interest. The two Feature Vector VIs provide a 1D array of identified features' motion vectors based upon their specific designs (History of Oriented Gradients and Linear Binary Patterns, respectively) and input design criteria. The idea for the Feature Vector operations would be to identify and then minimize any motion actually occurring regarding the target of interest

⁸ http://zone.ni.com/reference/en-XX/help/370281AC-01/imaqvision/imaq_extract_curves;
October, 2017

⁹ http://zone.ni.com/reference/en-XX/help/370281AC-01/imaqvision/imaq_extract_hog_feature_vector;
October, 2017

¹⁰ http://zone.ni.com/reference/en-XX/help/370281AC-01/imaqvision/imaq_extract_lbp_feature_vector;
October, 2017

(keeping the target within the images' center). A comparison test was performed between these three VIs regarding execution speed, and their results can be found in Table 1

The first downfall, for the two Feature Vector VIs, is that they require several milliseconds to complete execution. The HOG Feature VI would range from 25 milliseconds to 86 milliseconds making it unusable. In contrast, the LBP Feature VI was much faster with a range from 7 to 22 milliseconds, averaging 13 milliseconds. As 16 milliseconds is the maximum usable execution speed for near real-time, even the *IMAQ Extract LBP Feature Vector* VI is pushing the boundary of usability for real-time tracking purposes.

The next downfall also belongs to the Feature Vector VIs. As mentioned previously, they output a 1D array of motion vectors for the extracted features, but they do not provide explicit image-frame coordinates for the locations of the individual motion vectors calculated. Instead, it is a list of integer values. This prevents an easy means of identifying which particular values in

Table 1: Comparison of Curve, HOG, and LBP feature extractions for execution speeds.

Gait	Operation	Speed (ms)		
		Min	Med	Max
1	<i>Curves</i>	2	3	8
	<i>HOG</i>	28	53.5	86
	<i>LBP</i>	7	13	22
2	<i>Curves</i>	2	3	6
	<i>HOG</i>	25	56.5	78
	<i>LBP</i>	7	13	20
3	<i>Curves</i>	2	3	6
	<i>HOG</i>	33	55	78
	<i>LBP</i>	7	12	22

the list correspond to a given feature and, in turn, which specific values will correspond to the desired feature(s) representing the target of interest, not without going through additional, potentially extensive, processing.

Finally, the Extract Curves VI outputs a 1D array of clusters (packaged groups of data) with each element of the array concerned with a single extracted curve. Theoretically, this would allow for the operator to cycle through the output array until the desired curve is found. For this application, the notion would be to cycle through the array searching for the longest curve which could “ideally” be the edges of the joint implant. Not, however, a definite likelihood. Testing this VI operation with the three different edge detection algorithms demonstrated some key drawbacks such as the speed of the Canny ED being too slow overall (the quickest execution time being 191 milliseconds). The other drawbacks were either too few curves identified, too many curves identified, and/or broken or erroneous curves that would not be usable. Table 2 presents these results being performed upon an x-ray image without any attempt at pre-processing while Figure 8 provides a sample image with the results of the curve extraction operation for each edge detection. To reduce the amount of inter-pixel discrepancies which generate miniature edges within given colour regions, these tests were performed upon the same image with a Gaussian Smoothing operation having been performed. This operation is a convolution operation (1), explained in Chapter 4, executed with LabVIEW’s *IMAQ Convolute* VI¹¹ and a 3×3 Gaussian Smoothing kernel:

$$G = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

¹¹ http://zone.ni.com/reference/en-XX/help/370281AC-01/imaqvision/imaq_convolute; October, 2017

The results of this second analysis can be found in Table 3 with accompanying sample images and curves in Figure 9.

These results led to further investigation of the Particle Analysis method.

Particle Analysis Method

The primary focus of the particle analysis method is to reduce the image's information down to a minimum for just the target of interest. The final sequence of processing steps established from this investigation can be seen in Figure 10.

After several weeks of testing and achieving less than desired results for speed, an alteration to the approach was considered. This alteration is designed to reduce the amount of data that must be processed in the hopes of increasing processing speed. This decreases the amount of overall data that will need to be processed. Therefore, a crucial step is a reduction of the image size. While this step was also considered for increasing the Edge Detection operations execution times, the loss of image data and feature clarity from downsizing the image would reduce the ED operations' ability to accurately identifying feature edges, and therefore it was not pursued.

From here, the goal is a reduction in the image's information to that of the target itself. This will be accomplished with a thresholding technique using a VI of the same name provided among LabVIEW's Vision Development suite. The purpose of this VI is to convert any image from multi-scale pixel intensity values down to binary values. If a particular pixel's intensity value equals or exceeds/remains under (as determined by the user) the specified thresholding value, the corresponding pixel in the binary image will be set to a value of one. If the pixel's intensity value fails this threshold, it's corresponding pixel value will be zero in the associated binary image. This technique could be performed upon a simple array of values as well using a

Table 2: Comparison of execution speeds and curve extraction count between three operations as performed on a basic x-ray sample image.

Gait	Operation	Basic			w/Curves Extraction					
		Speed (ms)			Curve Count			Speed (ms)		
		Min	Med	Max	Min	Med	Max	Min	Med	Max
1	<i>Canny</i>	195	208	250	599	1105	1808	245	286	372
	<i>Prewitt</i>	2	3	5	0	5	13	5	6	8
	<i>Sobel</i>	1	1	2	0	4	11	4	5	6
2	<i>Canny</i>	191	212	261	1190	1959	1114	236	293	377
	<i>Prewitt</i>	2	3	5	3	10	5	5	6	7
	<i>Sobel</i>	1	1	3	2	8	3	4	4	6
3	<i>Canny</i>	199	211	249	705	1110	1937	261	286	383
	<i>Prewitt</i>	2	3	5	0	4	10	5	6	7
	<i>Sobel</i>	1	2	3	0	4	11	4	5	8

Table 3: Comparison of execution speeds and curve extraction count between three operations as performed on a smoothed x-ray sample image.

Gait	Operation	Basic			w/Curves Extraction					
		Speed (ms)			Curve Count			Speed (ms)		
		Min	Med	Max	Min	Med	Max	Min	Med	Max
1	<i>Canny</i>	192	205	261	459	931	1593	226	263	343
	<i>Prewitt</i>	2	3	28	0	1	6	5	6	95
	<i>Sobel</i>	1	2	48	0	0	5	4	5	99
2	<i>Canny</i>	188	208	247	936	1791	948	226	265	353
	<i>Prewitt</i>	2	3	28	0	5	2	5	6	66
	<i>Sobel</i>	1	2	61	0	3	2	4	5	39
3	<i>Canny</i>	196	205	253	504	938	1812	233	263	353
	<i>Prewitt</i>	3	3	51	0	0	6	5	6	33
	<i>Sobel</i>	1	2	10	0	0	4	4	5	95

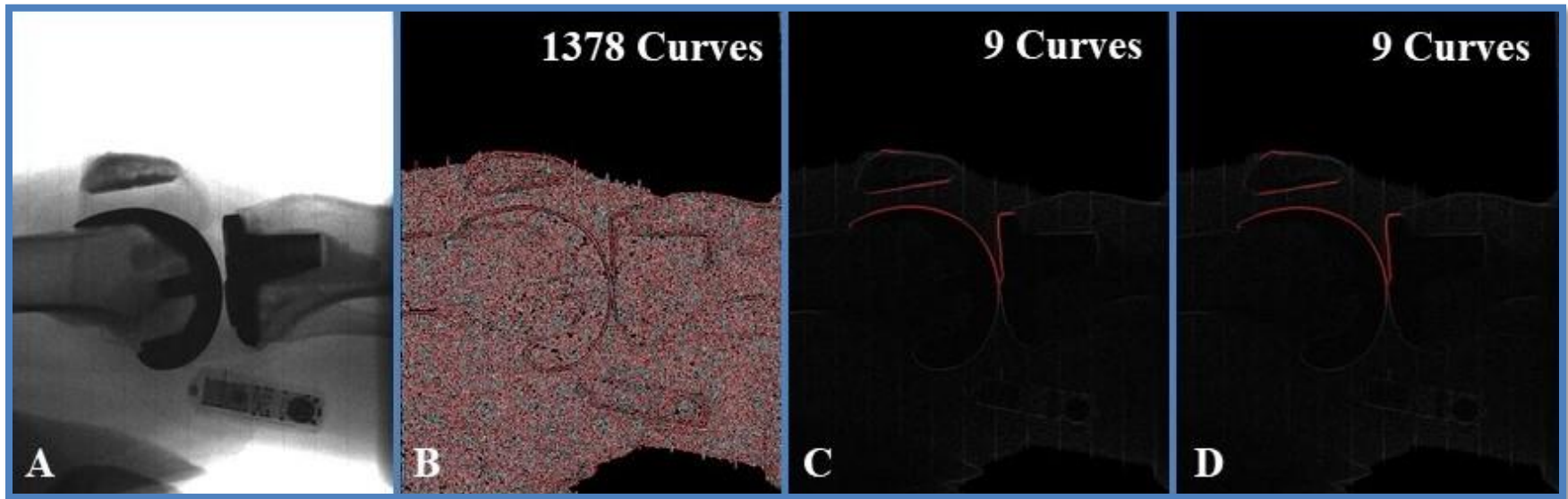


Figure 8: Edge detection operation with curve extraction of the edge data. A) Original sample image. B) Canny ED curve extraction results. C) Prewitt ED curve extraction results. D) Sobel ED curve extraction results.

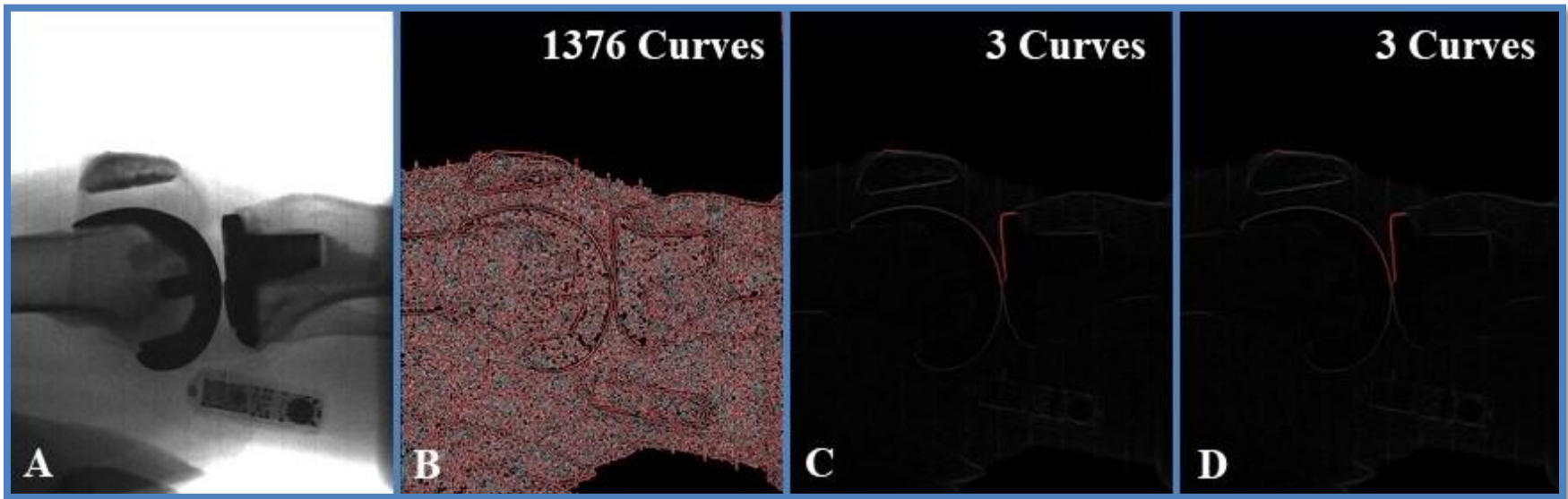


Figure 9: Edge detection operation after a single-pass Gaussian smoothing operation. A) Gaussian smoothed original image. B) Canny ED with extracted curves. C) Prewitt ED with extracted curves. D) Sobel ED with extracted curves.

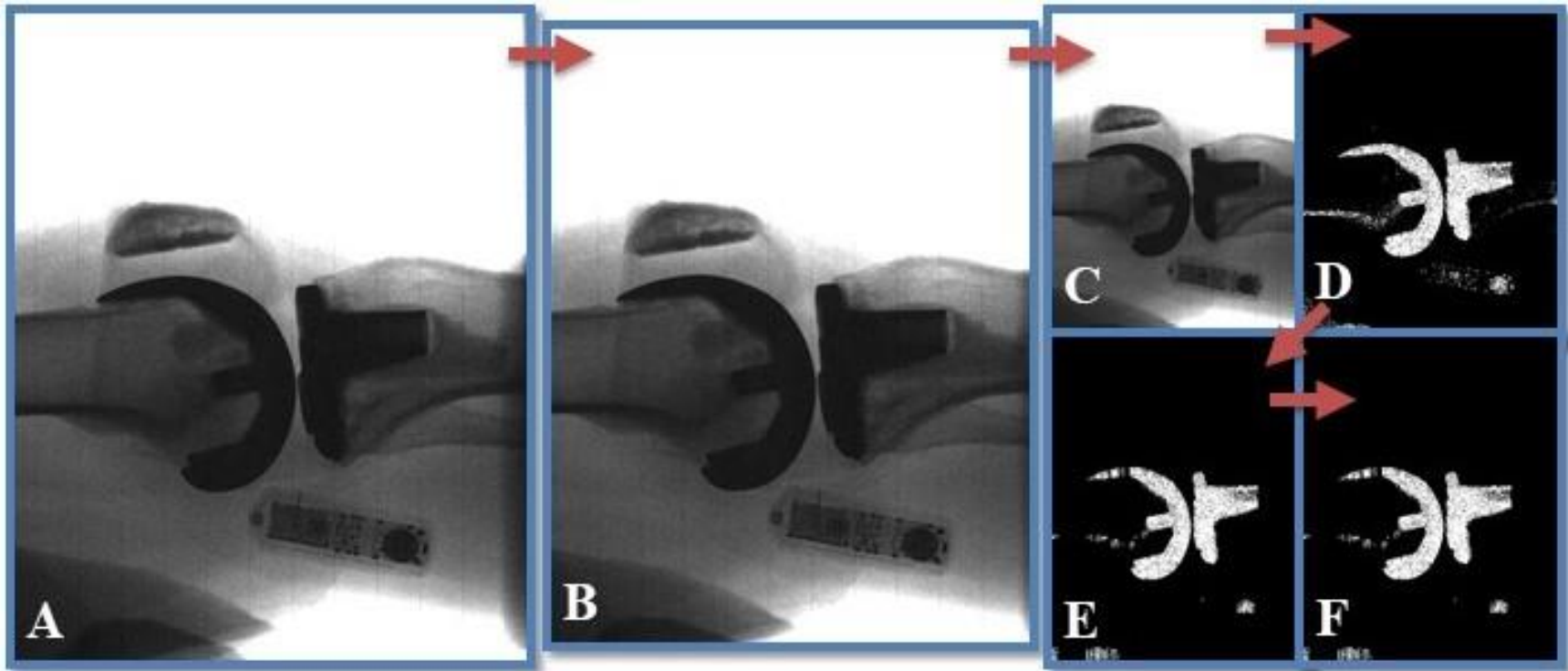


Figure 10: Final sequence of image processing steps. A) Original image. B) Borders removed. C) Size reduced by half. D) Thresholding operation performed. E) Pixel Cleaning operation performed. F) POpen operation performed, final operation before analysis.

simple comparison operator. However, LabVIEW's *IMAQ Threshold VI*¹² is an optimized form for working with the image data-type, and so it will be used here. For this research, if the pixel intensity *is or below* a certain value then the resulting value is a one. This form of Thresholding is referred to as Dark-Object Thresholding.

Before the Thresholding is applied, it would be beneficial to adjust the image to remove those regions of x-ray image that are expected to have low intensity values not associated with the target of interest. Due to the collimation design of the x-ray system, the borders of the resultant images are expected to contain a sliver or more of darkness representing the boundaries of the x-ray beam. To accommodate this, the *IMAQ Extract 2 VI*¹³ is used to “extract,” or pull out, the inner portion of the image, thereby removing the outer twenty-five pixels of the image.

This is done through the use of referencing the specific “sub-array” of the desired image for extraction. By defining the two coordinates for the portion of the original image desired, the operation identifies the associated pixel elements contained between them and places them into a new image array.

$$P_{out} = P_{in}(a:m, b:n) \tag{6}$$

The output image of (6), P_{out} , is the rectangular image array defined by the upper-left and lower-right pixel coordinates (a, b) and (m, n) , respectively.

¹² http://zone.ni.com/reference/en-XX/help/370281AC-01/imaqvision/imaq_threshold; October, 2017

¹³ http://zone.ni.com/reference/en-XX/help/370281AC-01/imaqvision/imaq_extract_2; October, 2017

While this operation could just as easily be performed after the reduction in image size, it was decided to perform it before this in order to keep a set range at the image's borders for this expected dark region. From here, the *IMAQ Resample* VI¹⁴ is used to reduce the image's size by half. LabVIEW does not provide the specific code sequences that take place for this operation; however, some inferencing can be done.

The first step for resampling is to determine the sampling step.

$$step = Round\left(\frac{Resolution_{old}}{Resolution_{new}}\right) \quad (7)$$

Because this particular usage of the Resampling operation is to down-sample, the step value obtained in (7) is then used to index and pull the pixel values, P , from the input image for the output image as shown in (8).

$$P_{out}(x, y) = P_{in}(x * step, y * step) \quad (8)$$

Therefore, the x-ray image is first extracted down from 972×768 to 922×718 and then resampled from there to 461×359 pixels. This pre-thresholding sequence is illustrated in Figure 11.

The inclusion of a Look-Up Table (LUT) operation was considered for pre-Threshold enhancement of the image. An LUT operation utilizes an input array matching in size to the integer scale of the input image array. Since our image input is an 8-bit greyscale, that means an input array containing 256 elements. Each element's position corresponds to a pixel intensity value for the greyscale image. When the VI reads the input image, each pixel value is replaced

¹⁴ http://zone.ni.com/reference/en-XX/help/370281AC-01/imaqvision/imaq_resample; October, 2017

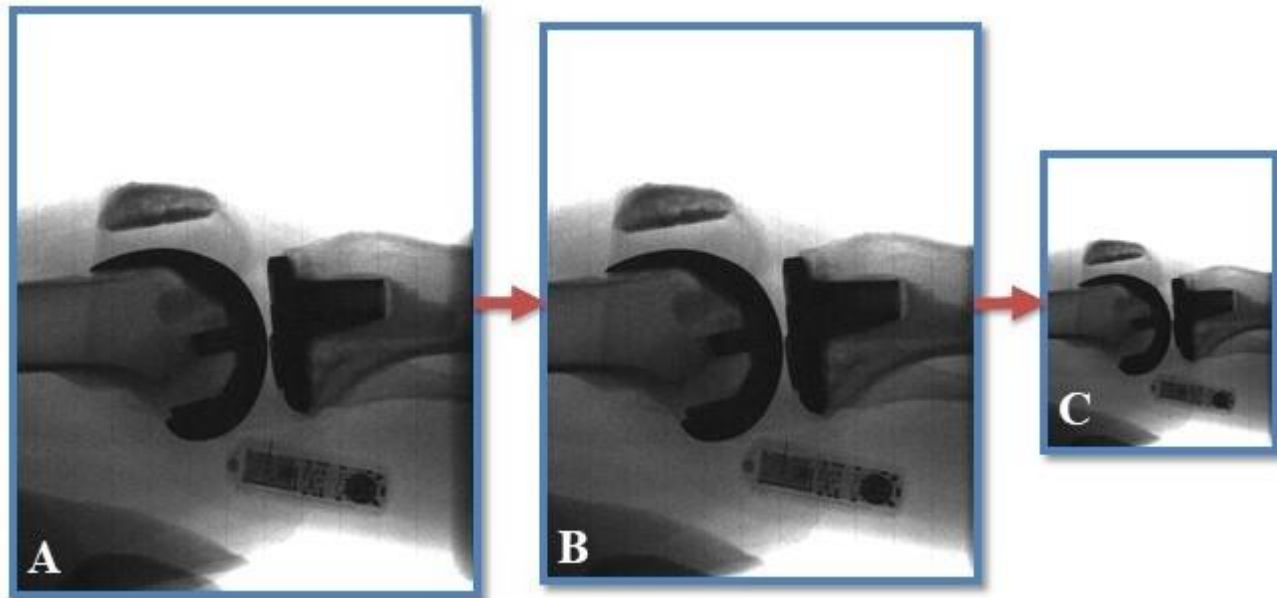


Figure 11: A) X-ray image received from Flat Panel (972 x 768 pixels). B) Image with 50 pixel borders extracted (922 x 718 pixels). C) Reduced image size (461 x 359 pixels).

with the stored value from the array element that matches that pixel's intensity value. This would have allowed an adjustment of the pixel intensity values through measuring original intensity values rather than attempting to adjust regions, which could unintentionally alter the target in unexpected ways. However, after spending several days attempting to determine an ideal spread for the re-adjustment, it was determined that this operation would be akin to the Thresholding operation already as it calls for determining the ideal place in the intensity scale for altering the darkness/lightness of the pixels. Considering that execution speed of this overall process was crucial, the addition of this operation would be a redundancy alongside the *IMAQ Threshold* operation, and it was subsequently discarded from further consideration.

Even with the application of the border removal before thresholding, there are still "outlier" pixels within the x-ray image itself that survive the operation into binary. These are usually parts of the greyscale image that contain the denser bone material or where the softer tissues may overlap (moving into/out of a crossover). The existence of these outliers means that the resultant binary image must also go through a clean-up step before further calculations should take place. Initial investigations lead to LabVIEW's Particle Analysis VI that would allow for the retention or redaction of various particles from the binary image based upon specified criteria. However, it was discovered that these particular operations were not ideal as they would remove entire particles, including segments of the target itself within the binary x-ray image, should some portion or all of it fit the indicated criteria. As this is exactly how these operations were designed, a different approach was needed for this stage of x-ray image clean-up.

In order to remove the outlier pixels without completely removing the main particles, a unique operation, dubbed Pixel Cleaning, was created to operate on binary images. This

operation is designed to calculate the total amount of pixels in each row and column of the input image and compare this row/column total to a specified value (separate values for rows versus columns). If the amount of pixels is fewer than the specified value, the entire row or column of pixels is set to values of zero.

$$\sum_{x=0}^{m-1} P(x, y) < a \leftrightarrow P_x(0: m - 1, y) = 0 \quad (9)$$

$$\sum_{y=0}^{n-1} P(x, y) < b \leftrightarrow P_y(x, 0: n - 1) = 0 \quad (10)$$

$$P_{out} = P_x \wedge P_y \quad (11)$$

The comparison value for rows, a , and columns, b , are provided by the user. The variables m and n represent the max width and height of the image, respectively. As the results of (9) and (10) are two separate binary images, the final cleaned image is obtained with (11) which performs a logical AND operation between the two.

Because this is a custom operation, there was no specific VI set that could be used from LabVIEW. Additionally, when originally tested using LabVIEW coding, it proved to require too much time for processing. To by-pass this issue, this particular code segment was written in C and compiled into a dynamic-link library (DLL), and then this DLL was loaded into LabVIEW via its *Call Library Function* VI¹⁵.

However, by using this custom operation in such a form, a DLL, this introduced a rather unexpected and complicated factor into this research's design scenario. The original plan was to have the entirety of the image processing being performed on the TFS' main computer which is

¹⁵ http://zone.ni.com/reference/en-XX/help/371361N-01/glang/call_library_function; October, 2017

configured using a Real-Time Operating System (RTOS). Because an RTOS does not possess all of the standard software libraries and supporting files that a standard OS like Windows or Apple OS, there are limits to which programs can be supported by such an environment. It was discovered that, while the compiler for Pixel Cleaning's code could be set to restrict the amount of libraries auto-linked to it, there are still certain libraries that are connected which limit its abilities for use on an RTOS. This meant that a drastic change in the execution environment would be needed if this operation would be employed. To accomplish this, a secondary computer was added to the TFS which possesses a Windows OS. This secondary computer would execute the Pixel Cleaning and any subsequent Image Processing algorithms before transmitting their results back for actual analysis on the TFS' main computer (the RTOS).

The final operation that takes place before the analysis is a Binary Morphology operation in LabVIEW called Proper-Open, or POpen, executed using the *IMAQ Morphology VI*¹⁶. This is a rapid succession of Opening and Closing operations performed a total of seven times (four Open operations and three Close operations, alternating). Opening and Closing operations are, in turn, combinations of Erosion and Dilation operations.

An erosion operation scans a given pixels neighbours to determine the minimum value, using it as the output pixel value. This erosion will trim down the borders of large particles and potentially remove small particles completely.

$$P_{out} = \min [P_{in(x-1,y+1)}, P_{in(x,y+1)}, P_{in(x+1,y+1)}, P_{in(x-1,y)}, P_{in(x+1,y)}, P_{in(x-1,y-1)}, P_{in(x,y-1)}, P_{in(x+1,y-1)}] \quad (12)$$

¹⁶ http://zone.ni.com/reference/en-XX/help/370281AC-01/imaqvision/imaq_morphology;

October, 2017

A Dilation operation is the reverse of the erosion. It searches for the maximum value among a given input pixel's neighbouring pixels and uses it as the out pixel value. This operation thickens particle borders, and it can often close small holes in binary image particles.

$$P_{out} = \max [P_{in(x-1,y+1)}, P_{in(x,y+1)}, P_{in(x+1,y+1)}, P_{in(x-1,y)}, P_{in(x+1,y)}, P_{in(x-1,y-1)}, P_{in(x,y-1)}, P_{in(x+1,y-1)}] \quad (13)$$

Equations (12) and (13) are written for a default 3×3 window size which was used with this research. Additional odd-numbered sizes can also be implemented at the user's discretion.

An Opening operation is a Dilation operation followed by an Erosion operation, and the Closing operation occurs in reverse order. What these particular operations offer, combined together into the POpen operation, is a means of cleaning the edges of the larger particles within the binary images as well as potentially removing any remaining smaller particles completely. A sample of both the Pixel Cleaning operation and the POpen operation can be found in Figure 12.

The Centroid calculation, performed with *IMAQ Centroid VI*¹⁷, is the culmination of the Particle Analysis Method, and it is the root of the analysis portion of this research phase. This calculation is a way of determining the location of the largest concentration of high pixel intensity values which should be the implanted knee joint within a binary x-ray image. In mathematical terms, the x- and y-coordinate for the centroids are calculated using weighted summations along each dimension, divided by the sum of all pixel values.

$$C_x = \frac{\sum_{y=0}^{n-1} (y+1) [\sum_{x=0}^{m-1} P(x,y)]}{\sum_{x=0}^{m-1} \sum_{y=0}^{n-1} P(x,y)} \quad (14)$$

¹⁷ http://zone.ni.com/reference/en-XX/help/370281AC-01/imaqvision/imaq_centroid; October, 2017

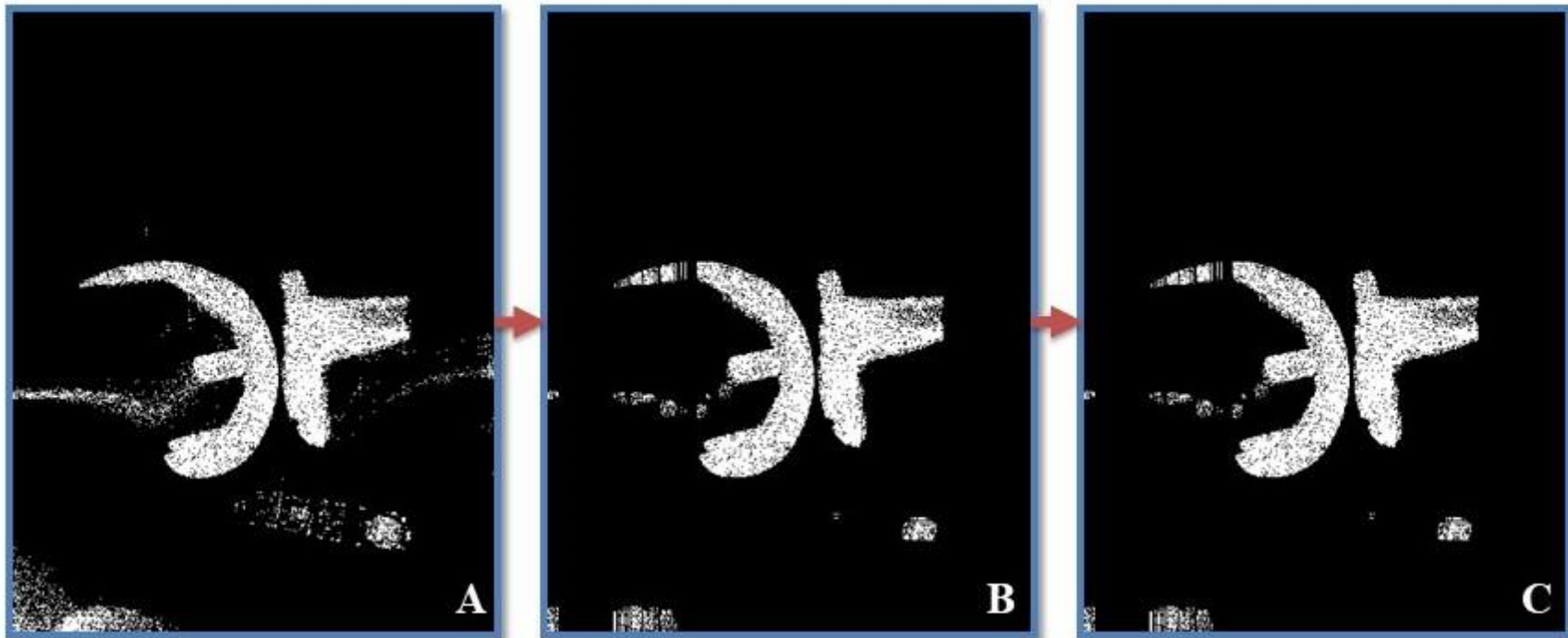


Figure 12: A) Thresholded x-ray image. B) Pixel Cleaning operation. C) POpen operation.

$$C_y = \frac{\sum_{x=0}^{m-1} (x+1) [\sum_{y=0}^{n-1} P(x,y)]}{\sum_{x=0}^{m-1} \sum_{y=0}^{n-1} P(x,y)} \quad (15)$$

C_x represents the x-coordinate along the image's width of the centroid. Likewise, C_y represents the centroid's y-coordinate. The maximum width and height are represented by m and n , respectively.

This can be performed upon a standard greyscale image; however, the determined centroid of the image would be influenced by each pixel within the image. As this would not be a very accurate estimate of where our desired target, the joint implant, is, we use the previously explained binary conversion and clean-up operations. With the binary image, we reduce the image's information down and provide a higher degree of accuracy regarding the target's calculated location within the image. This calculated location, (C_x, C_y) , obtained from (14) and (15) is then used as servo commands for the x-ray system's motion.

Phase Two – Human Gait Modeling

In developing the Human Gait Model, it invariably merged the Gait Modeling research with that of the state estimation for feature tracking. While the original design expected the modeling to be developed separately from but used within the state estimation, the end design merged these into a single setup that would estimate future positions based upon current and previous positions.

The first step of this phase was to identify within literature or develop a set of equations that would relate a subject's height to their limb dimensions. In any equations that are used for the gait modeling, there is a high probability of needing the lengths of both the torso, the thigh, and possibly the shank as well. Anthropometric data became the basis for this segment as the desire was to obtain equations that would be applicable for the majority of the population. A publication on ergonomics, [42], yielded the following equation results:

$$\text{Shoulder Height} = 0.81 * (\text{Patient Height})$$

$$\text{Lower Extremity Height} = 0.52 * (\text{Patient Height}) \quad (16)$$

$$\text{Thigh Length} = 0.232 * (\text{Patient Height}) \quad (17)$$

The Laser Read Height, (18), is an estimation of the ideal height on the subject for reading their distance from the TFS with the SICK Laser Scanners. Placed just below the shoulders, the lasers would have a broader span to register a distance measurement from without the potential of interference from the subject's chest or arm motion.

$$\text{Laser Read Height} = 0.8 * (\text{Patient Height}) \quad (18)$$

By subtracting (16) from (18), the torso length value is found to be:

$$\text{Torso Length} = 0.28 * (\text{Patient Height}) \quad (19)$$

The next step in generating the gait model was to find or develop the kinematics for motion. A starting point for this was believed to be the neuro-musculoskeletal modeling equations discussed in Chapter Two. As stated previously, these equations as they were presented were too computationally expensive for running in real-time, and so the desire was to separate out the skeletal-motion portion of these equations from the neuro-muscular control portions. However, despite some optimism and as expected, they were too intertwined with the neuro-muscular portions to be of use.

Therefore, it was decided that a purely kinematic approach would be needed. As prior research revealed, more recent works in this area revolved around simulation-level kinematics (meaning that comprehensiveness rather than execution time held priority) while earlier works were concerned with the physical properties of the motion, such as inertia. Consequently, the necessary equations would need to be developed rather than researched.

There are two points on the subject that are of import. The first is the obvious, the joint of interest, while the other is a second point of commonality connecting the subject and the TFS, the Laser Read Height/measured distance. This second point will establish a reference for developing a set of equations to the joint of interest. One equation will be for the subject going from this common point near the shoulders down to the knee, and another equation for the TFS also going from this common point to the x-ray imager (the knee's position).

Equations (17) and (19) provide the length of the thigh and torso sections connecting the Laser Read Height to the knee's location. The point of calculation comes in the rotation around the hip joint. By assuming that a person will consistently lean forward to the same degree when walking in their natural gait, the torso's positioning can be considered a constant. While this angle can be left as an input constant to account for each person's individual gait style, this will allow the equations to be reduced to a single dependent variable, the angle at the hip joint. With this, the knee joint's positioning in a Cartesian format as shown in Figure 13 for use within the x-ray images coordinate frame are given in (20) and (19).

$$x = -(0.28 * Height) \cos(\varphi) + (0.232 * Height) \cos(\theta) \quad (20)$$

$$z = -(0.28 * Height) \sin(\varphi) + (0.232 * Height) \sin(\theta) \quad (21)$$

Equation (20) provides the positioning of the knee joint with respect to the subject's torso at the Laser Read Height in the x-coordinate. The z-position is likewise provided in (21).

On the machine side, the TFS' configuration would remain consistent within the Knee Mode from subject to subject which would allow for some inferring from its dimensions and controls to develop the desired kinematics equations of motion. One would be the expected distance of the subject from the TFS machine registered using the SICK Laser Scanners while a

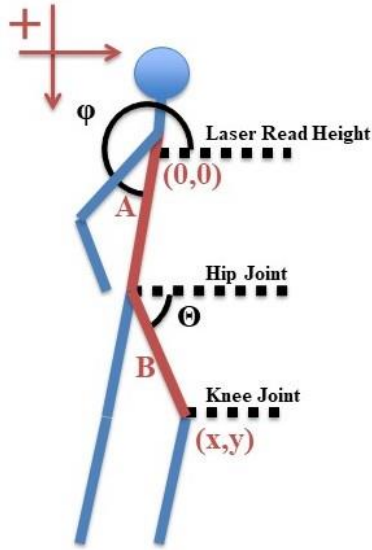


Figure 13: Original subject representation for angle and segment lengths of equations. N.B. - Positive for x and y-directions are right and down, respectively.

second is the homing positions of the axes used for zeroing the encoders. The first, the Laser Read position, establishes the connection point from the TFS to the subject. The second allows for the use of the axis encoder readings for positioning data of the joint through the machine.

These points will act as the connections for the TFS to the subject as demonstrated in Figure 14, the following equations are generated using simple Cartesian positioning:

$$x = x_{Laser} + x_{Home} + \Delta x - x_{Encoder} - 1477.96 \text{ mm} \quad (20)$$

$$z = 453 \text{ mm} + z_{Home} + \Delta z + z_{Encoder} - 0.8H (-114.3 \text{ mm}) \quad (21)$$

The associated terms of (20) are, in order, the read distance to the subject, the homing position for the x-axis, the change in position between current and next executions, and the encoder

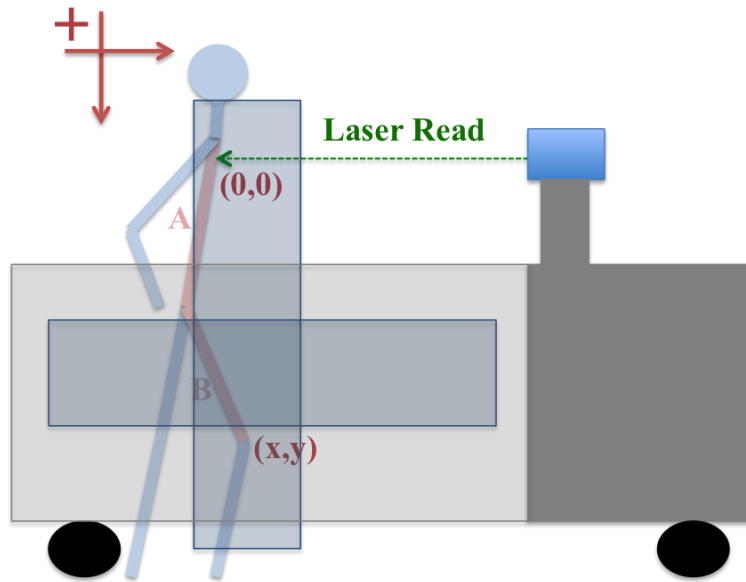


Figure 14: Original TFS representation for direction and positioning.
 N.B. - Positive directions for x and y-directions are right and down, respectively.

position reading (this value is subtracted due to reversed encoder readings from assumed positive), and lastly the fixed distance from the Laser Scanners source point to the end of the axis encoders in millimeters. Likewise, the first term in (21) is the distance from the ground to the bottom of the z-axis encoders while the intermediate terms are z-components similar to those in (20). The last two terms are the Laser Read Height, (18), and an optional addition should the subject be walking on a 114.3 millimeter platform instead of directly on the ground. Lastly, the H used represents the subject's height in millimeters.

By equating these two sets of equations, (20-23), they can be written in terms of the x and z-angle from an identified coordinate position or in reverse (coordinate position from an angle). These equations are the basis of the gait modeling:

$$0.232H \cos \theta - 0.28H \cos \varphi = x_{Laser} + x_{Home} + \Delta x - x_{Encoder} - 1477.96 \text{ mm} \quad (24)$$

$$0.232H \sin \theta - 0.28H \sin \varphi = 453 \text{ mm} + z_{Home} + \Delta z + z_{Encoder} - 0.8H (-114.3 \text{ mm}) \quad (25)$$

Equations (24) and (25) can then be written to become the equations in use: one to calculate the present angle from the identified x- and z-coordinates, (26), and two equations to calculate a predicated position change based upon an extrapolated angle, (27) and (28).

$$\theta = \tan^{-1} \left[\frac{453 \text{ mm} + z_{Home} + \Delta z + z_{Encoder} - 0.8H (-114.3 \text{ mm}) + 0.28H \sin \varphi}{x_{Laser} + x_{Home} + \Delta x - x_{Encoder} - 1477.96 \text{ mm} + 0.28H \cos \varphi} \right] \quad (26)$$

$$\Delta x = x_{Laser} + x_{Home} - x_{Encoder} - 1477.96 \text{ mm} - 0.232H \cos \theta + 0.28H \cos \varphi \quad (27)$$

$$\Delta z = 453 \text{ mm} + z_{Home} + z_{Encoder} - 0.8H (-114.3 \text{ mm}) - 0.232H \sin \theta + 0.28H \sin \varphi \quad (28)$$

Change in Expected Model's Development

While these equations appeared both logically and mathematically sound, there proved to be difficulties in their implementation. The first difficulty demonstrated to be a disparity with the positive value directions within the coordinate frames established. In specific, the reference frame for the z-direction conflicted with the inherent reference frame of the trigonometric functions. While adjustments could easily be worked into an equation to “flip” the coordinate frame’s direction, this could introduce the potential for further difficulties at a later point from an unnecessary negation. A second came in an apparent disparity between the calculated angle and

its associated point. In essence, to verify the validity of the equations, a sample point was taken and used to calculate its angle, θ , in radians. From there, this angle was then placed back into the equations to determine the associated point. This calculated point was not equaling that of the original point.

To determine where the potential issue is located, samples were taken using pre-determined positions, primarily with the two angles, φ and θ , set for -90 degrees. In this way, the Laser distance should approximately equal that of the x-direction positioning while the subjects Torso and Thigh lengths should be equal to the z-direction positioning from the Laser's height. This demonstrated that there were discrepancies within the calculations. Corrective values were introduced into (24) and (25) in an attempt to correct the equations.

$$0.232H \cos \theta - 0.28H \cos \varphi = x_{correction} + x_{Laser} + x_{Home} + \Delta x - x_{Encoder} - 1477.96 \text{ mm}$$

$$z_{correction} + 0.232H \sin \theta - 0.28H \sin \varphi = 453 \text{ mm} + z_{Home} + \Delta z + z_{Encoder} - 0.8H (-114.3 \text{ mm})$$

Through the use of a beginning calibration pose, these correction values could be determined for the individual subjects and included within subsequent calculations.

This method showed some promise in terms of the x-direction. However, there were still issues present within the z-direction calculations. The error encountered in the z-direction fluctuated enough that it could not strictly be considered as compounding nor did it display a recognizable pattern of behaviour that would lend towards diagnosing its cause. Therefore, it was decided to restructure the equations, eliminating any implicit positioning calculations. The first was to change the direction of the subject within the machine (see Figure 15 and Figure 16). By

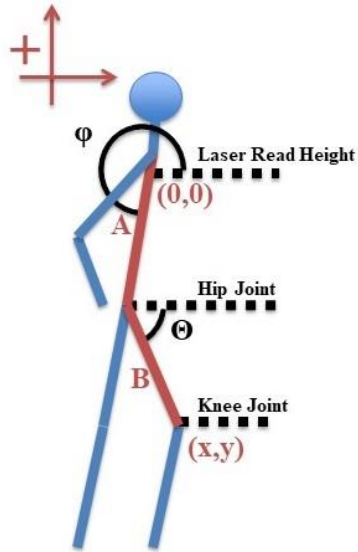


Figure 15: Modified subject representation for angle and segment lengths of equations. N.B. – Only change between present and prior diagram is the positive direction of the vertical axis which is now upwards.

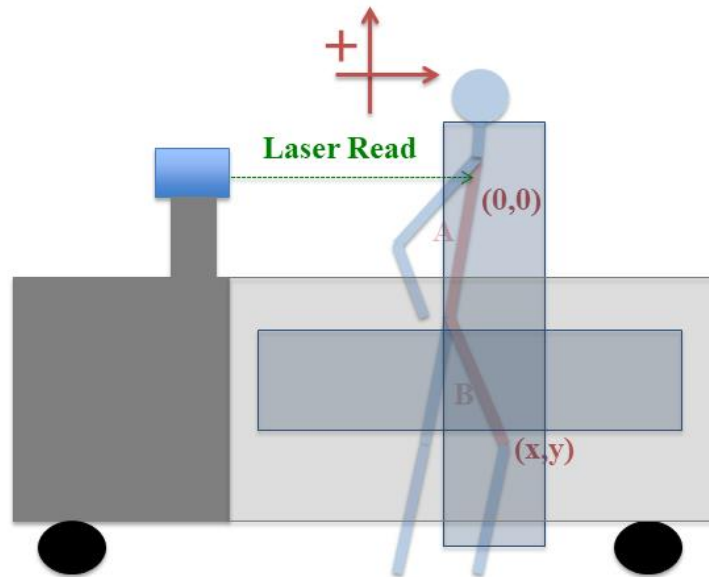


Figure 16: Modified TFS representation for direction and positioning. N.B. – Changes from prior diagram are the positive vertical direction, upwards, and the perspective of the TFS machine.

having the subject face into the machine, the depth of the subject's chest could vary significantly based upon where the laser is actively hitting at any given instant. From a slim to muscular and male to female, the contour of the subject's chest was a potential source of the error being introduced into the gait model. A change in the direction that the subject is facing provides the ability to track the subject across the back were there will not be as great a potential for changes in depth from one moment to the next during a walking cycle.

In addition, by including the calibration pose directly into this restructuring presents the opportunity to further let the system customize itself to each individual subject beyond simply using their height. By modifying the method of determining Laser Read Height, an explicit value can be measured rather than implicitly calculated from estimating position as displayed in Figure 17. This approach also has the added benefit of removing any need for accommodating potential platforms the subject stands upon.

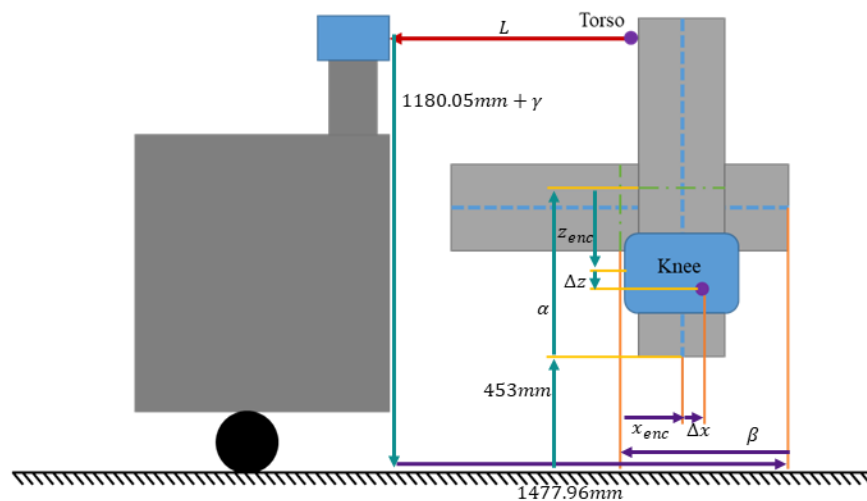


Figure 17: Diagram displaying positioning values and variables for modeling equations.

$$x_{correction} = 1477.96 \text{ mm} - x_{Laser} - x_{Home} - \Delta x + x_{Encoder} \quad (29)$$

$$A = 727.05 \text{ mm} + \gamma - z_{Home} - \Delta z - z_{Encoder} \quad (30)$$

Equation (30) provides the length of the torso from the laser read position on the subject's back to their hip joint. The 727.05 millimeters term represents a fixed portion of the distance for the Laser Read Height minus the 453 millimeters from the z-axis base to the ground while γ is the changeable distance physically measured for each subject. Reworking (24) and (25) to include the new values obtained in (29) and (30). The three equations for the recreated human gait model are as follows:

$$\theta = \tan^{-1} \left[\frac{z_{Home} + \Delta z + z_{Encoder} - 727.05 \text{ mm} + A \sin \varphi}{x_{correction} + x_{Laser} + x_{Home} + \Delta x - x_{Encoder} - 1477.96 \text{ mm} + A \cos \varphi} \right] \quad (31)$$

$$\Delta x = x_{correction} + x_{Laser} + x_{Home} - x_{Encoder} - 1477.96 \text{ mm} - 0.232H \cos \theta + A \cos \varphi \quad (32)$$

$$\Delta z = z_{Home} + z_{Encoder} - 727.05 \text{ mm} - \gamma - 0.232H \sin \theta + A \sin \varphi \quad (33)$$

The continued inclusion of the corrective X term accounts for the depth of the subject's torso in (31) and (32) while the z-direction correction is now built into the calculated A term within (31-33).

Phase Three – Control Algorithm: Supervisory Knee Tracking Algorithm

This phase is essentially software development and testing resulting in the creation of a Supervisory Knee Tracking Algorithm (SKTA). As mentioned under Phase One of this section, the original plan was to have both the image processing operations and the gait model located on the main TFS computer, and, in this way, it would be a simple matter of transferring the necessary data between the processing loops of the code. With part of the image processing

occurring on a secondary computer rather than the main, slight modifications to this expected design needed to occur.

A diagram of the SKTA's final organization is shown in Figure 18. The received x-ray image is sent through the image processing stages on both the Main and 2nd computers before being analyzed with Image Energy Profiling. This profiling will indicate whether the results of the centroid calculation can be sent directly on to the axis control code or if the gait model needs to provide a predicted command as shown in Figure 19.

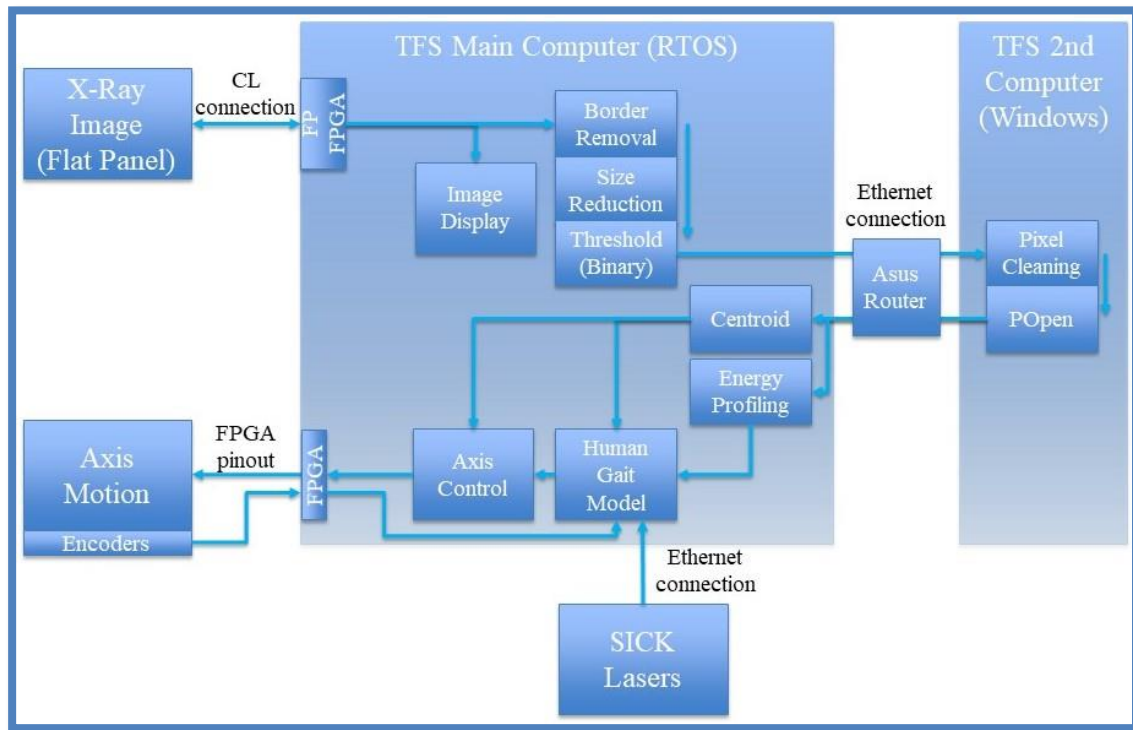


Figure 18: System diagram demonstrating the flow of data from the Flat Panel (top-left), through the image processing and analysis, updating of the gait model (and prediction calculations if necessary), and output of the motion commands to the axis control to obtain motion of the x-ray system.

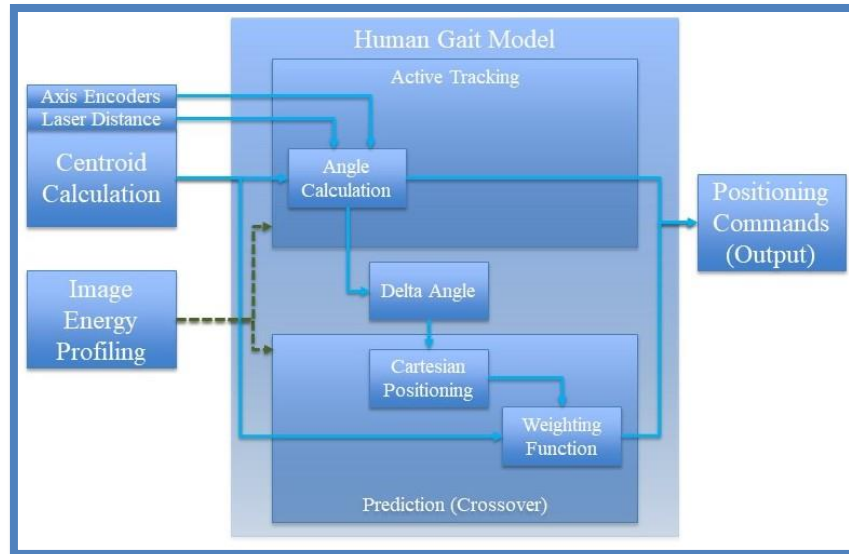


Figure 19: System diagram for the Human Gait Model's active tracking versus prediction.

Data Communication

To account for moving the data between two different computers, high speeds would be even more of a necessity to maintain real-time execution. To accomplish this, the basic TCP/IP communication protocols were investigated first.

While TCP/IP is among the fastest protocols to transfer data across a network, in LabVIEW programming it has a tendency of proving inconsistent in terms of establishing and maintaining this connection between points. LabVIEW's Network Stream (NS) protocols, on the other hand, are simpler to write and use. In addition, the NS protocols are a completely lossless form of communication across a network which reduces the risk of dropping or corrupting data being sent. It accomplishes this by using what it has named the Network Stream Engine.

Unlike communication links established directly using TCP/IP, Network Streams are a unidirectional form of computer communication with established Writer and Reader Endpoints that are connected to each other with the NS Engine. This engine establishes buffers at both endpoints and manages the actual transference of data from the writer's buffer to the reader's without any user or programmer input/management. As a side note, this also means that no user or programmer can modify these internal operations should they desire to.

The NS Communication would allow for the reading and writing of the data at approximately 65 Hertz writing/100 Hertz reading. In theory, faster writing speeds are possible, however, they would ultimately introduce a lag in the transference of the data through the NS Engine. Because the NS Engine is not something that can easily be manipulated by a user of LabVIEW, it can only be speculated that the faster write speeds would cause an overflow of the write-side buffer and therefore cause delays in the data transference. For both this reason and the fact that the minimum speed desired was only 60 Hertz, there was no need for further pursuit of increasing the transference rates.

Image Analysis

Once the processed data is transferred back across to the main computer, the SKTA simultaneously performs the centroid calculation as well as an analysis of the resulting binary image information. If this analysis indicates valid target data, the centroid location information is sent to both the axis command control for servo-motion as well as the gait modeling for use in updating present location. If it is not, it switches the gait model from update mode to prediction mode.

Image Energy Profiling

The analysis spoken of uses image energy profiling to determine the occurrence of crossover events. Image energy profiling is the distribution of active pixels in a binary image over time. By applying this concept to x-ray images of knees passing each other, the total pixel count will increase as the auxiliary knee begins moving into the x-ray frames and subsequently decrease as it moves out once more. This is calculated using:

$$E = \sum_{x=0}^{m-1} \sum_{y=0}^{n-1} P(x, y) \quad (34)$$

The image's energy level, E , is a unit-less number, and m and n represent the total width and height, respectively, of the image in pixels. Recording the results of (34) overtime demonstrates a spike in the total pixel count during crossover events as seen in Figure 20. Figure 21 provides a

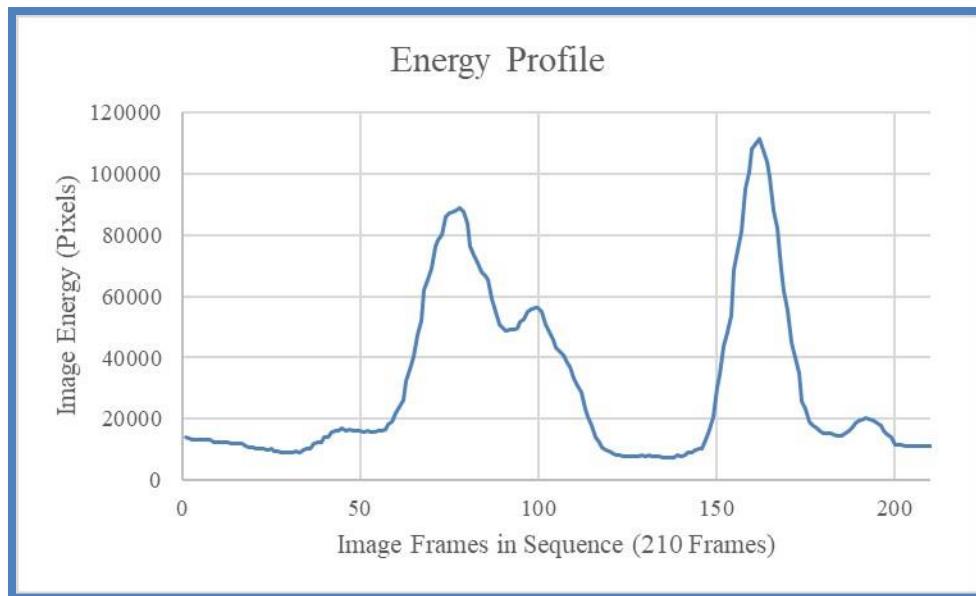


Figure 20: Image energy profile of a single gait cycle. Spikes in the energy level represent the moments of crossover.

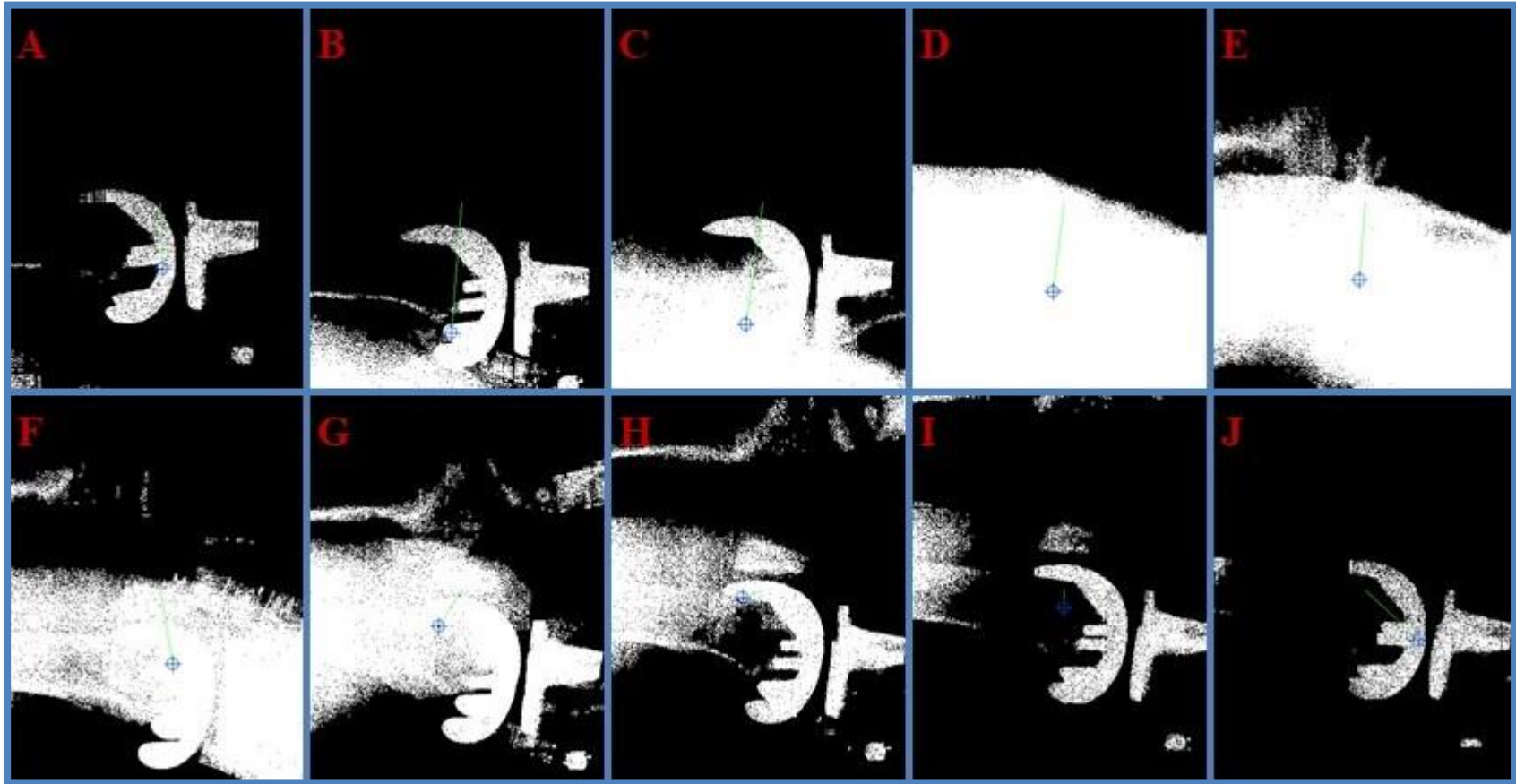


Figure 21: Sample sequence of binary x-ray images demonstrating the increase in energy during a crossover event. The auxiliary knee moves through the images in an upward vertical direction.

sample sequence of binary x-ray images demonstrating a single crossover. Please note that the auxiliary knee moves across these images in a vertical direction.

While there is no specific determination to definitively say that a crossover has begun or ended, trial and error testing has at least provided a general approximation to use. Therefore, using this methodology, when the image's energy level reaches a certain point, a crossover is assumed to be occurring, and the tracking algorithm is switched to using the Gait Model's predicted location. This predictive tracking will begin from the last known coordinate positioning allowing for a smooth transition from active to predictive tracking.

Positioning Commands/Transitioning

Due to the potentially significant changes in positioning between individual image frames during active tracking, there is the possibility for moderately large angle changes to be recorded. These large angles can cause the predictive tracking to rapidly progress beyond the bounds of the image frame. To prevent this from occurring, angle value boundaries are included within the angle's prediction calculation to prevent it from outputting an angle position too far forward or too far backward of the subject, i.e. anatomical limitations for the expected rotation about the hip joint.

In addition, the SKTA employs a weighting function between the predicted location and the identified centroid of the x-ray image's energy. This weighting function is designed to act as an anchor for the predicted location keeping it closer to the overall center of pixel mass within the image. Because the auxiliary knee's presence within a binary x-ray image is caused by the overlapping of tissue, the center of the image's pixel activity during the crossover remains close to the target knee of interest. The center of the image had been considered for the anchor point of this function, but by instead anchoring it to the centroid of the pixel activity, it has allowed for the predictive motion to remain close to the actual point of image activity and, consequently, the

target of interest. This weighting function applies seventy-five percent of the weight upon the calculated centroid to better maintain this anchor for the predicted location.

Once the crossover ends, the system needs to switch back from predictive to active tracking. The predictive motion was not planned to be perfect despite efforts to make it so. Thus, it is expected that a (potentially large) gap will exist between the final predicted position and the next active tracking position. Therefore, during this return from predictive tracking, a simple intermediary calculation is used to prevent the tracking from “jumping” the distance between its present location and the newly identified target’s location. This transitioning segment is designed to simply move half the distance between the present location and the identified location until there is less than a five millimeter gap between them. Once this distance is five millimeters or less, direct tracking is resumed.

The final output of both the image processing and the gait modeling is the change in position of the target in millimeters. This position change is then sent to the axis control code for final processing to adjust the x-ray image axis to be closer to the measured joint implant position.

RGB-D Sensor Usage

An original idea for this research had been to include an additional 3D sensor, an RGB-D style camera sensor. The concept had been to use this sensor as a third method for identifying and tracking the location of the desired target, the knee joint. In this manner, there would be three separate methodologies of tracking occurring simultaneously: active tracking using particle analysis, predictive tracking with the human gait model, and then visual tracking using the RGB-D sensor. At any given point in time, these three tracking forms would be able to compare their individual results to determine a realistic location for the target, and in such a way be able to mitigate any potential errors within the individual methods.

However, when the research for inclusion of such a sensor was beginning, several difficulties presented themselves. These difficulties lay in both the sensor's needed hardware, and the sensor's specific software alongside any interacting software necessary for integrate the sensor into the TFS' system.

Hardware

The most prevalent issue with the sensor hardware was the field-of-view for the given sensor. Amongst those 3D camera sensors investigated were the prominent Kinect V2 by Microsoft and Xtion by Asus. Their view fields would allow for a user to stand within a half meter to five meters for the sensors' software to identify them. However, through brief testing, it was determined that the sensor would require approximately 1.7 meters to generate a stable skeletal representation of the person for motion identification. While the subject could approach by another step to approximately 1.25 meters and continue to maintain some stability, any closer than this and the skeletal representation would not be stable. The resultant motion tracking could not be ensured of any accuracy due to "glitching" of the skeleton's joints (inaccurate position representations due to incorrect interpretations of what data is received).

This minimum distance was a concern for the desired application of this sensor with the TFS. To ensure its integration with the TFS' tracking methodology and to keep the subject within its sensor range at all times, it would need to be mounted directly onto the TFS. This means that the subject would be standing fairly close to the sensor, quite probably limiting the camera's view to around the subject's thighs and upwards or from their chest/midsection downwards. While tracking of the lower limb joints is the purpose of the TFS, to develop the stable skeleton's within the RGB-D sensor's software would require more reference points than these views could provide.

In an attempt to broaden the sensor's field-of-view, two options were considered. The first was simply the addition of a second RGB-D sensor to work in concert with the first while the second was to use fish-eye lenses to broaden and shorten the individual sensor's detection range. Unfortunately, the additional sensor option was quickly discarded due to the interferences overlapping infrared (IR) fields would cause. The depth of RGB-D sensors is most prevalently measured using IR cameras, and should these fields overlap it would cause distortions in each sensors' registered depth.

Therefore, the fish-eye lens option was investigated. The idea for these types of lenses were presented with Microsoft's Kinect V1 curtesy of the Nokia Company. These lenses, designed to expand a camera's view field by using sharply convex surfaces to bend in more light, would expand the region to which the Kinect V1 could see and thereby shift the range it could establish a stable software skeleton closer to the sensor. Unfortunately, a version of these lenses were never released for the Kinect V2 (which utilizes a different placement configuration for its RGB and Infrared cameras than the V1), nor was any found for other 3D sensors. If such lenses were to be used then, they could need to have been specially crafted for this research. This situation would normally have indicated that, instead of trying to work with the Kinect V2, the V1 would prove better hardware wise, and it potentially would have been. However, this would then move into the software issue for both the sensor itself and its inclusion with the TFS' existing software infrastructure.

Software

The TFS' software is written in the LabVIEW programming language and executes on a Real-Time Operating System. These present issues regarding both the sensors' individual controlling codes as well as incorporating these codes into the TFS' controlling infrastructure.

Only one sensor was found to have any LabVIEW coding pre-created for it. The Kinect V2's LabVIEW software suite was written and provided by the third-party HaroTek LLC working with National Instruments. An older version of this software suit, for the Kinect V1, was created, but with the debut of the V2 further development of the V1's code suit was never pursued.

While LabVIEW can incorporate outside code such as C or C++, it is a process in itself to do so. As the standard software packages that come with almost all sensors sold now are for the more common programming languages (C++, Python, Java, etc.), none of the sensors investigated possessed pre-generated LabVIEW code for their usage. In fact, almost all hardware that does include LabVIEW code is commonly manufactured under National Instruments, the designers of LabVIEW. The only RGB-D sensor that was found with LabVIEW code was the Kinect V2, mentioned previous, which was crafted by a third-party programmer. All of the other sensors would have needed to have their controlling software written in C or C++, compiled into a DLL, and imported into LabVIEW using its Call Library Function VI.

If these imported codes, packaged as DLLs, were to be run on a Windows-based OS, there would not be any further issues to prevent the sensors' inclusion within this research. However, this is not the case. The TFS runs on an RTOS as explained in a previous section, and so this prevents their easy inclusion into the TFS' software infrastructure. The compilation process for the DLLs write in the need for certain basic DLLs to be located upon the executing computer. As a RTOS is a stripped down operating system, these basic DLLs are not present, and so the loaded DLL will fail to run.

Major Change Upon Experimental Plan

Before final testing of the Human Gait Model research could be performed, unexpected events necessitated investigating alternative approaches for obtaining results. Restrictions, placed upon the usage of the TFS, prevented the ability to perform x-ray testing regarding the human gait model and, consequently, of the SKTA for overall performance of this research.

Devised workarounds allowed for continued testing, they did limit the ability to determine performance under the original conditions this research was developed for. The first workaround was designed to allow for testing the generated gait model. The second was aimed to emulate the acquisition of x-ray images using visual images in order to further evaluate the gait model as well as the performance of the SKTA as a whole.

Testing of the Human Gait Model

In its original setup design, the Human Gait Model would be tested using the image processed x-ray images devised in Phase One. This was expected to reduce the processing load prior to moving into the modeling/prediction stage, thus allowing for more latitude in the processing speed of the Human Gait Model.

Due to the inability to receive x-ray images from the fluoroscope, the alternative devised was to route the visual images from the TFS' pattern matching setup through the gait model. The methodology that the pattern matching employs requires the tracking of an external marker, and so this will introduce the potential for human error in terms of the target's placement during model calculations. In addition, this change in setup necessitated minor alterations to the code in order to allow the pattern matching technique to be sent into the modeling.

A physical adjustment of the TFS was also performed to aid in simulation of the x-ray images' positioning perspective. By removing the Dexela Flat Panel from its mounting and

rigging the Basler camera into the center of the mounting bracket, the center of the captured images from the Basler camera would better simulate the center of captured x-ray images (see Figure 22).

To provide a means of emulating the auxiliary knee obscuring the tracking target, the marker was placed upon the inside of the leg farthest from the camera. During actual x-ray, the knee closer to the Flat Panel would be the primary, but, by switching the knees for this testing, some of the behaviour inherent to the fluoroscope images is retained.

Testing demonstrated behaviour within this alternate setup that would not be present within the fluoroscopy, however. Due to the image energy profiling, the point of crossover

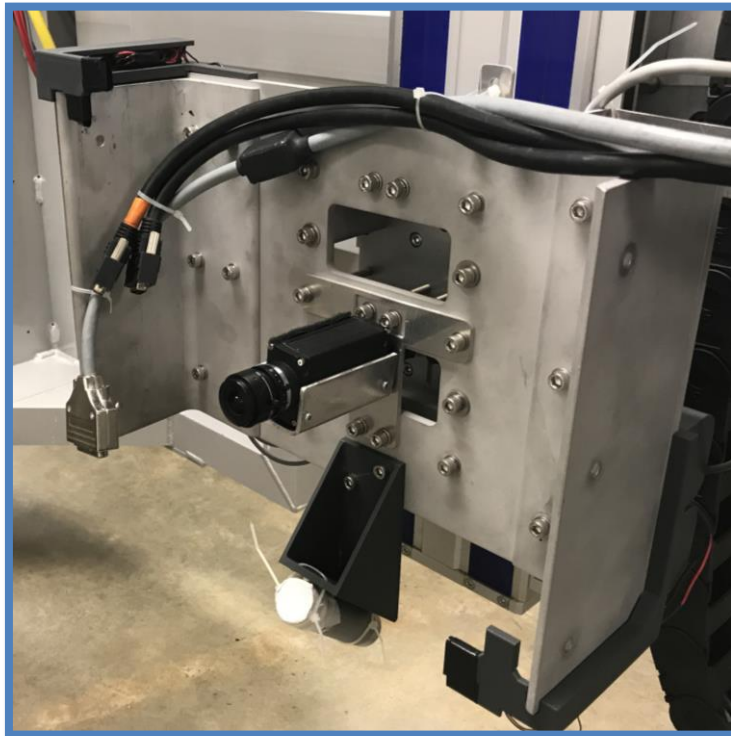


Figure 22: The removed FP bracket with the Basler black-and-white camera rigged to the central position within the bracket.

would be identified as the auxiliary knee began “entering” the field of view and end at the point it is exiting (please refer back to Figure 21). This is not the case with the pattern matching operation. The pattern is maintained through more of the image sequence resulting in smaller segments of time requiring predictive motion. In the original particle analysis approach with x-ray imaging, it is expected that between twenty to twenty-five images would require predictive tracking. However, it has usually been less than ten images requiring it for the pattern matching. With this smaller timeframe, results showed little difference between the use of the heuristic algorithm for predictive motion and the human gait model.

In an effort to better demonstrate the Human Gait Model’s use for prediction, emulated x-ray imaging (detailed in the proceeding section) was used to employ the image energy profiling for crossover detection. The usage of the image energy profiling method indeed extended the period of time requiring predictive motion commands while also having the added bonus of being closer to the original overall scenario. Again, this provided a better means of testing the behaviour of the three different predictive methods.

Emulating X-Ray Images for SKTA Testing

After testing the Human Gait Model, the same approach was decided upon for testing the entire system. The primary concern then revolved around emulating the acquisition of x-ray images for processing through the SKTA. Traditional images obtained through the Basler camera contain a significantly broader distribution of objects and potential pixel intensity values. With the image processing’s dark-object thresholding to binary, a lighter and more uniform background would need to be obtained through the Basler camera. Therefore, the x-ray emitter side was locked down, and a sheet of white cloth was draped over that axis. This can be seen in Figure 23 and Figure 24.

Using this rigging, it was then possible to bring in Phase One's Particle Analysis image processing to test its processing ability in conjunction with Phase Two's gait model and the performance of the Supervisory Knee Tracking Algorithm. The desire was to determine the SKTA's ability for transitioning between active and predictive tracking as well as to obtain a general idea of the overall performance of the algorithm as a complete tracking system.

Also, some changes were required in the tracking code to accommodate the new setup. In the code's design, the Basler camera fed directly into the Pattern Matching algorithm with the results then being fed into the Axis Command for motion control or into the Human Gait Model placed right before it. In order to route the acquired images through the Particle Analysis' image processing, this segment needed to be restructured in such a way that the acquired image could be sent to the processing code with the results then going through the gait model and on to the Axis Command.

However, another concern then presented itself at this point. To be more precise, this concern reintroduced itself from previous investigations into computer-to-computer communications options. The potential for communication lag was briefly mentioned prior, and after the secondary computer was reintroduced into the test setup to allow for image processing, this lag was discovered. While no alterations knowingly occurred upon either the secondary computer or the primary computer's communications code, this lag (approximately two seconds) had serious repercussions in the performance ability of the overall system.

It was determined that, with the alterations already in place to use the Basler camera for image acquisition rather than the Dexela FP, some minor adjustments of the Particle Analysis' image processing could take place without significantly impacting its performance. The limited operations being executed upon the secondary computer were the custom code segment as well



Figure 23: White sheet draped over x-ray emitter axis to simulate the light background of an x-ray image.



Figure 24: Back view of the white sheet draped over the x-ray emitter axis.

as a single follow-up operation. The purpose of the custom code segment was to further clean up and refine the binary x-ray images customarily containing outlier pixels and “fuzzy” edges from the indistinct nature of tissue densities as seen in Image B of Figure 25. With the obtained visual images now possessing a more distinct demarcation between image features, a moderately cleaner image is obtained. The possibilities of these outliers and indistinct edges occurring drop significantly as demonstrated when comparing Images B and D of Figure 25.

This meant that the custom clean-up operation, while still desirable for binary x-ray images, was no longer a necessity in the alternate setup. As the follow-up POpen operation could

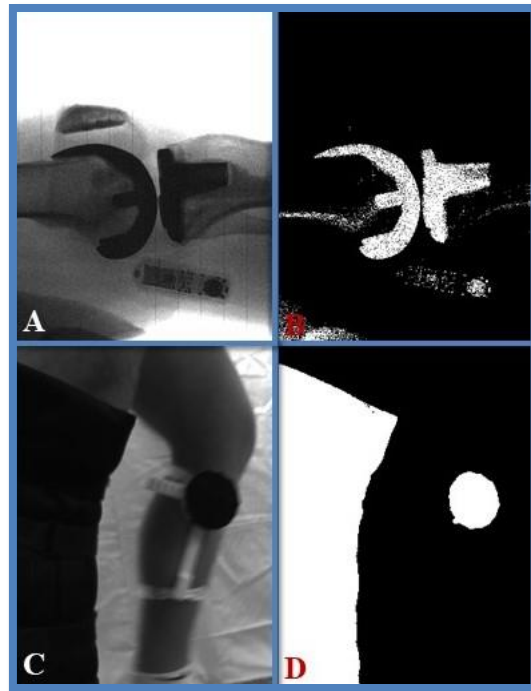


Figure 25: Example differences between x-ray and visual images in terms of indistinct edges after binary conversion. A) Obtained x-ray image. B) Thresholded x-ray image. C) Obtained visual image. D) Thresholded visual image.

easily be transferred to the primary TFS computer, the secondary computer could then be removed from the process altogether. This resulted in a minor slowdown of the image processing's execution as it no longer had the available power of two computers, but by operating on a single computer the concerns regarding lag in data transmission speeds were bypassed.

SKTA Testing

From this point, it was now possible to perform tests of the SKTA in operation. The system diagram for the alternative setup is shown in Figure 26. For reference comparison, Figure 18 previously illustrates the original system diagram.

The overall performance of the Supervisory Knee Tracking Algorithm demonstrated its feasibility for use in its intended scenario. While some issues arose within the image processing portion caused by the change in image source, primarily environmental lighting and image shadowing effects, it was possible to mitigate their influence to some degree using both modifications to the environment and manual adjustments within the code. Despite these undesired influences, the SKTA's analysis operations easily handled switching between the active tracking and prediction within the Human Gait Model. In addition, no significant jumps in axis positioning was observed indicating that the transitioning control worked as expected.

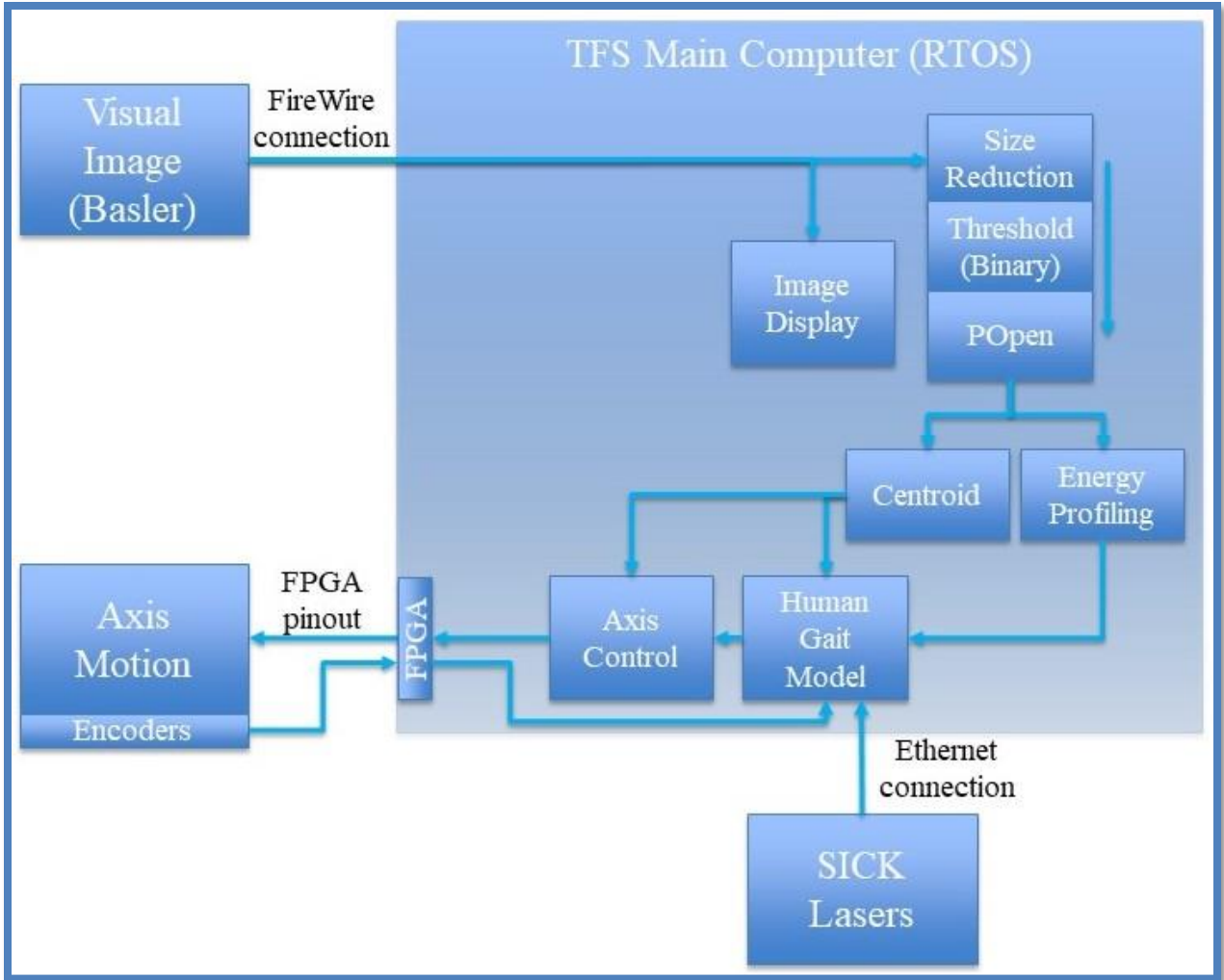


Figure 26: System diagram of the alternate setup using visual images rather than x-ray.

CHAPTER SEVEN – RESULTS OF RESEARCH

Over the course of this research, the Tracking Fluoroscope System underwent a full reconstruction. This reconstruction was designed to make improvements upon the TFS' x-ray motion driving abilities, x-ray imaging, and overall machine capabilities. Between the original TFS' tracking methodology and the unforeseen limitations introduced with the new x-ray digital imager, the course of this research took on new meaning.

In the original TFS' tracking design, it was possible to track through the x-ray images as desired using pattern matching and a simple heuristic algorithm for crossover handling. The more refined version of the TFS' pattern matching that was designed during the reconstruction aided in speeding up the pattern matching tracking to a degree. However, paired with the more data intensive x-ray images, it still could not process as quickly as desired, not even to allow the inclusion of additional calculations (for prediction). For this reason, the majority of the assessments performed in evaluating the work for this research were made between both versions of the Pattern Matching method (TFS Version 1 and Version 2) and the newly researched Particle Analysis method.

In addition, tests were performed for comparing the original tracking method's heuristic prediction with the current tracking method's lack of prediction and the new human gait modeling's prediction. This also provided the opportunity for a general assessment of the overall potential of the proposed Particle Analysis Method.

Image Processing

Of primary interest was the execution speed of the Particle Analysis. As this method is supposed to replace the much slower pattern matching, whatever new method was devised would first need to be fast enough to be of use. This means that any method would need to have a faster execution

rate than 60 Hertz, or 16 milliseconds. Table 4 provides a summary of the speed comparisons between the three methods. Comparison graphs can also be found in Appendix A.

In a close second to the execution speed, the accuracy of the determined positions is important. If the new method does not provide an accurate determination of the target within the image frame, it is just as impractical as a highly accurate method that is too slow for real-time usage. Figure 27-29 demonstrate a comparison between the identified locations of the target joint implant within the image coordinate frame by human visual inspection with the two versions of the pattern matching algorithm and the particle analysis. Please note that, for image processing, the image coordinate frame's origin is in the upper-left corner of the frame and positive directions are to the right and down.

The first image in each group is the original image used for analysis and visually inspected for the joint implant's center of rotation (desired, ideal point of tracking for maintaining both portions of the implant within the image frame), followed by the three images from each of the tracking methods. Within the remaining three, the red boxes and the blue reticle

Table 4: Summary of execution speed testing between the Version 1 pattern matching (V1 PM), the Version 2 (V2 PM), and the particle analysis (PA). All speeds are given in milliseconds.

	Test Set 1 (ms)			Test Set 2 (ms)			Test Set 3 (ms)		
	V1 PM	V2 PM	PA	V1 PM	V2 PM	PA	V1 PM	V2 PM	PA
Min	43	22	8	40	19	8	43	21	8
Max	65	35	11	63	34	11	63	75	12
Mean	52.58	27.69	9.16	52.01	26.85	9.38	53.40	26.79	9.24
Std Dev	5.74	3.58	0.78	5.69	4.08	0.91	4.79	5.36	0.95

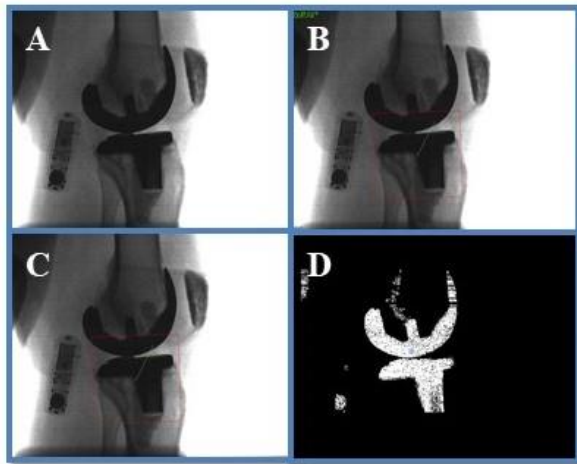


Image	X-Position (Pixels)	Error from Img A (%)	Y-Position (Pixels)	Error from Img A (%)
A	431	-	422	-
B	427.25	0.87	499.87	18.45
C	428.14	0.66	499.65	18.40
D	403.87	6.29	399.10	5.43

Figure 27: Positioning comparison on the first Gait Set between the three tracking methods in image coordinates. A) Image chosen for testing, frame 11 of 210, with visually identified positioning commands from image center. B) TFS Version 1 pattern matching results. C) TFS Version 2 pattern matching results. D) Particle Analysis results.

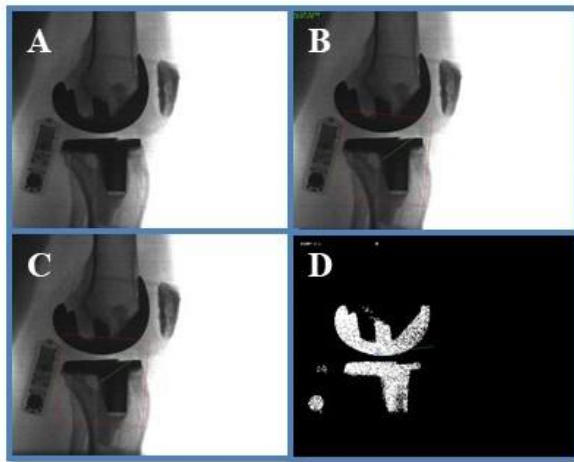


Image	X-Position (Pixels)	Error from Img A (%)	Y-Position (Pixels)	Error from Img Center (%)
A	317	-	443	-
B	315.00	0.63	505.82	14.18
C	311.87	1.62	507.57	14.58
D	292.60	7.70	408.22	7.85

Figure 28: Positioning comparison on the second Gait Set between the three tracking methods in image coordinates. A) Image chosen for testing, frame 9 of 150, with visually identified positioning commands from image center. B) TFS Version 1 pattern matching results. C) TFS Version 2 pattern matching results. D) Particle analysis results.

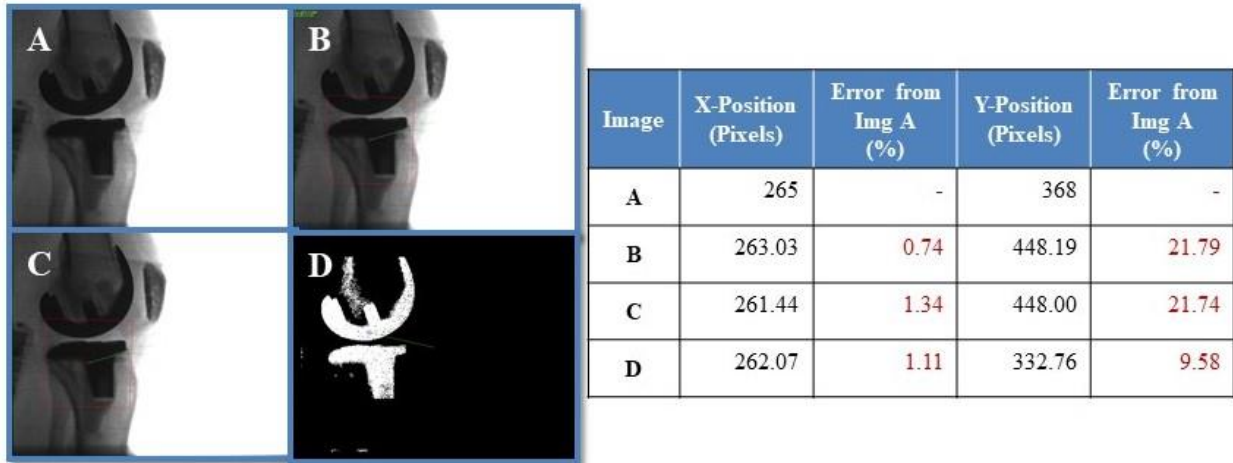


Figure 29: Positioning comparison on the third Gait Set between the three tracking methods in image coordinates. A) Image chosen for testing, frame 111 of 140 with visually identified positioning commands from image center. B) TFS Version 1 pattern matching results. C) TFS Version 2 pattern matching results. D) Particle analysis results.

indicate the identified location for target tracking. The accompanying tables provide the positions in pixel coordinates of the identified target within the image frame. Beside the position values are percentages for how far from the visually identified point (desired point for target tracking) the given coordinate is. Higher percentage values translate to a further position from the ideal.

It is important to note that the pattern matching algorithms required a fairly consistent feature for tracking purposes. Therefore, the tibial portion of the knee joint implants were the focus of the pattern matching operations. Because of this, the identified targets in frames B and C are the lower portions of the joint implant. This can be concerning when, during active tracking with pattern matching, there is the possibility of maintaining tracking of the “target” while part of the upper femoral portion of the implant could potentially be out of the x-ray field-of-view. In contrast, the particle analysis demonstrates the ability to track the implant closer to its point of rotation, thereby reducing such a possibility from occurring.

The coordinates from the visual inspection were “eye-balled” and so possess the potential for some human uncertainty in the exact positioning. In terms of the x-positioning behaviour, all three of the tracking algorithms demonstrated that they routinely calculated values similar to each other, with the particle analysis’ results occasionally being a few percentages more than the pattern matching. The z-positioning behaviour is the biggest tell, however, as the pattern matching is consistently further off. While this behaviour can be attributed to the necessity of tracking the tibial implant segment, it provides a clear distinction between the pattern matching and the particle analysis methodologies.

The final point of interest for examining the particle analysis approach to tracking revolves around the crossover of the knees within the x-ray images. When the pattern matching algorithm cannot identify the target within the image frame, this is when prediction must begin. The same holds for when the image analysis portion of the particle analysis method indicates a crossover using image energy profiling. On the other side of these, once the crossover completes, the algorithm needs to switch back. Depending upon how much time is spent in prediction, the final position before returning to tracking can potentially vary a great deal. The longer the prediction continues, the greater chance of positioning errors building up.

Therefore, what was of interest for this assessment was when and for how long these prediction periods occur. Having a tracking method that can maintain identification for as long as possible will reduce the amount of time for potential positioning error to buildup. Each of the three tracking methods were used on three sets of x-ray images containing crossovers recording which images could be tracked from and which the algorithms could not. Figure 30-32 present

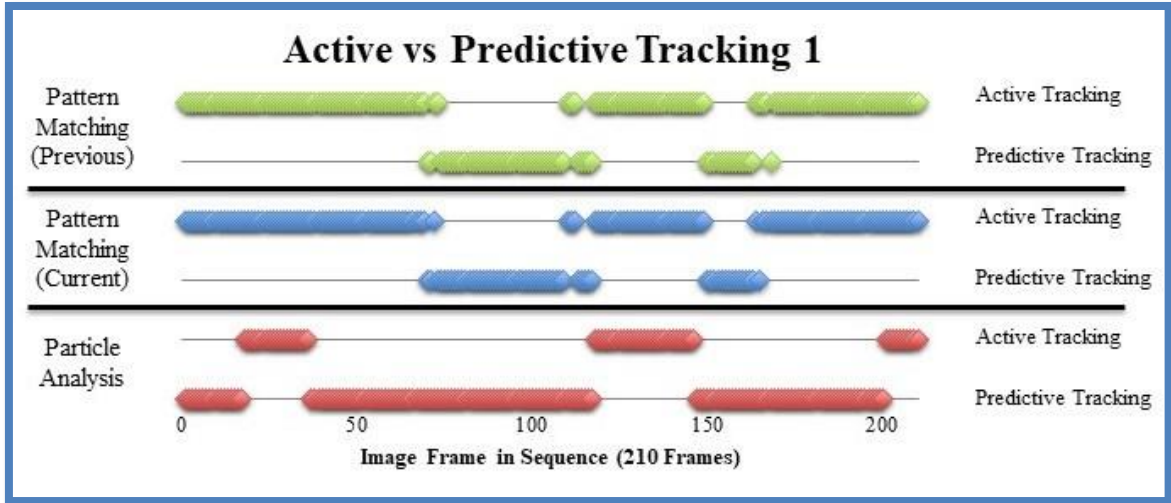


Figure 30: First test sequence comparing the pattern matching and the particle analysis algorithms to identify crossovers. N.B. - Lower lines represent time spent predicting motion.

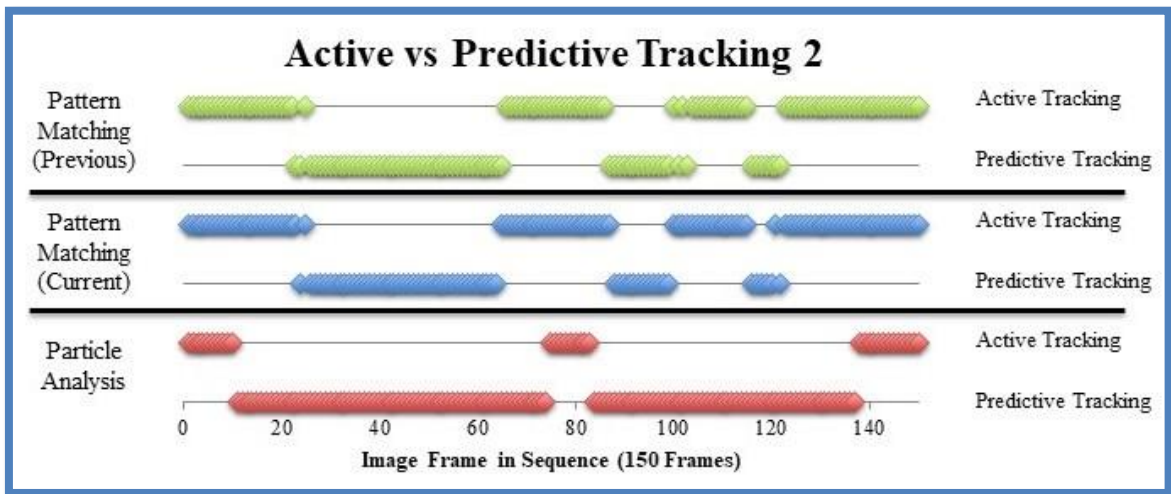


Figure 31: Second test sequence comparing the pattern matching and particle analysis algorithms to identify crossovers. N.B. - Lower lines represent time spent predicting motion.

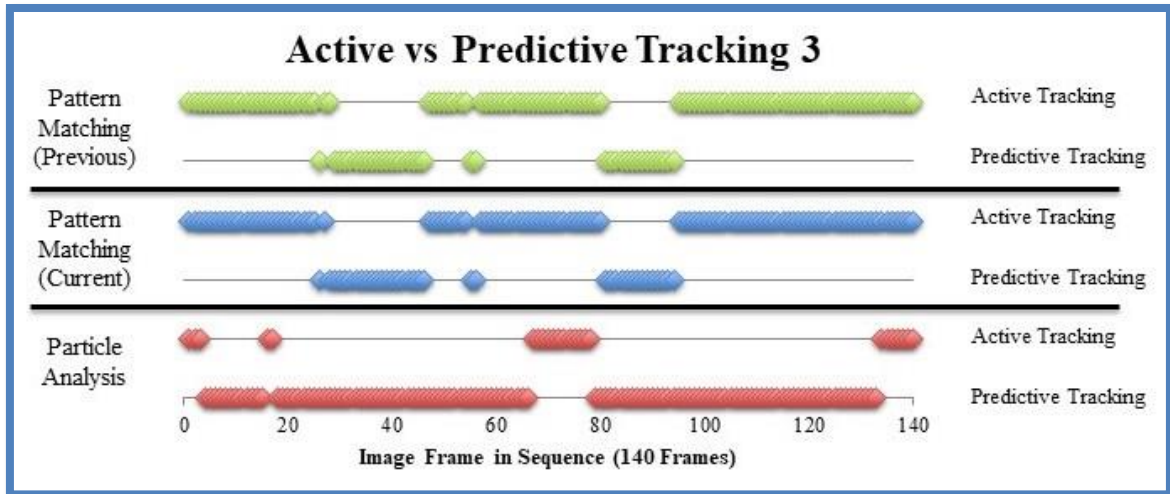


Figure 32: Third test sequence comparing the pattern matching and particle analysis algorithms to identify crossovers. N.B. - Lower lines represent time spent predicting motion.

these results. To better compare them, they have been stacked upon each other: the previous form of pattern matching, the current (refined) form of pattern matching, and the created particle analysis. The first line (upper) represents those frames that the algorithms successfully identified a target to track while the second (lower) line represents what would require prediction.

Prediction: None versus Heuristic versus Human Gait Modeling

The TFS version 1 employed the heuristic algorithm to determine motion during crossovers and other losses in tracking. On the other hand, the second version does not contain any method for prediction. Pattern matching is not currently used with the x-ray system but a visual camera, and so there are no crossover occurrences although the occasional tracking loss still occurs. Programmatically, the second version simply stops all axis motion in such cases until the target is identified once more. With particle analysis, the Human Gait Model provides the predictive motion for these instances.

The assessments performed to test the gait model is an overall behavioural comparison between having no form of prediction (Figure 33), the original heuristic algorithm (Figure 34), and the Human Gait Model (Figure 35). This was done by applying it in a testing scenario and recording the resulting motion for offline comparison. The differences between the methods are seen within Figures 33-35 in their output motion commands during prediction. While initial intentions were to perform preliminary testing on the three different sets of image sequences used in testing the image processing prior, it was quickly realized that the real-time encoder feedback is a necessary component for accurate results of this predictive motion.

Table 5 provides a summary in percentages of the difference in axis location from one image frame to the next for each of these prediction methods as shown in their respective figures prior. The location difference from Image A to B was the result of the prediction tracking while going from Image B to C was transitioning back from prediction.

Without prediction capabilities, the first prediction case had zero change in positioning, while the second case (the Heuristic Prediction) possessed only a small amount of motion. The Gait Model case provided the highest difference in position moving from A to B. During prediction, a high predicted motion is not necessarily good nor bad, but once the next step is taken into account, this case is seen as having been a benefit. Transitioning from Image B to Image C (moving out of prediction), the largest jump is from the situation without any form of prediction. The No Prediction case demonstrated a nearly 150 percent in required x-motion to re-center the x-ray axes on the target. The Heuristic Prediction still required an over 100 percent change in position to move from Image B's position to the identified position in Image C. On the other hand, due to the Gait Model's predictive motion between Images A and B, the location difference from Image B to Image C was just under 80 percent.

Img	Identified Location (mm)	Motion Commands (mm)
A	(-160.49, -260.18)	(25.10, 0.10)
B	(-160.49, -260.18)	(0, 0)
C	(65.87, -247.76)	(226.36, 12.42)

Figure 33: Image sequence depicting the motion results of not having any prediction algorithm. Please note, Motion Commands for Image A are from previous to Image A. The image(s) listed in red represents an instant requiring prediction.

Img	Heuristic Location (mm)	Motion Commands (mm)
A	(-238.57, -251.47)	(8.6, 1.10)
B	(-229.97, -250.37)	(8.6, 1.10)
C	(39.13, -247.82)	(269.10, 2.55)

Figure 34: Image sequence depicting the motion results for using the basic heuristic prediction method.

Img	Gait Model Location (mm)	Motion Commands (mm)
A	(-271.96, -250.48)	(16.85, 1.86)
B	(-195.63, -264.24)	(76.33, -13.76)
C	(-40.95, -252.36)	(154.68, 11.88)

Figure 35: Image sequence depicting the motion results for using the Human Gait Model for prediction.

Table 5: Change in motion between pattern matching image frames for the three prediction methods. Demonstrates the general magnitude of the required axis motion between individual frames that were studied.

Image Step	Change in motion (mm)					
	None		Heuristic		Gait Model	
	X	Z	X	Z	X	Z
A → B	0	0	3.60	2.76	28.07	5.49
B → C	141.04	4.77	117.02	1.02	79.07	4.50

A second series of tests were performed using the emulated x-ray imaging approach in order to test the gait model with the particle analysis setup. This was performed with the intent to emulate the longer prediction period produced by the image energy profiling. Initial tests with Particle Analysis only possessed the dark target for tracking. However, the observed performance was similar to that of Pattern Matching due to the “disappearance” of the target during crossover while it passed behind the auxiliary knee, obscured from view. To more closely emulate the x-ray imaging behaviour, a dark cloth was secured to the near side of the auxiliary knee to provide the increase in dark pixels from the addition of the auxiliary knee’s dense tissue in x-ray images.

The case with no prediction is shown in Figure 36 where it can be seen that the target is never re-acquired. Figure 37 provides the heuristic algorithm’s results which, again, do not regain the target for tracking. Lastly, the gait model’s results, Figure 38, demonstrate that it was able to recover tracking. Table 6 summarizes the percent differences in location comparable to the magnitude of motion needed to go from one image to the next.

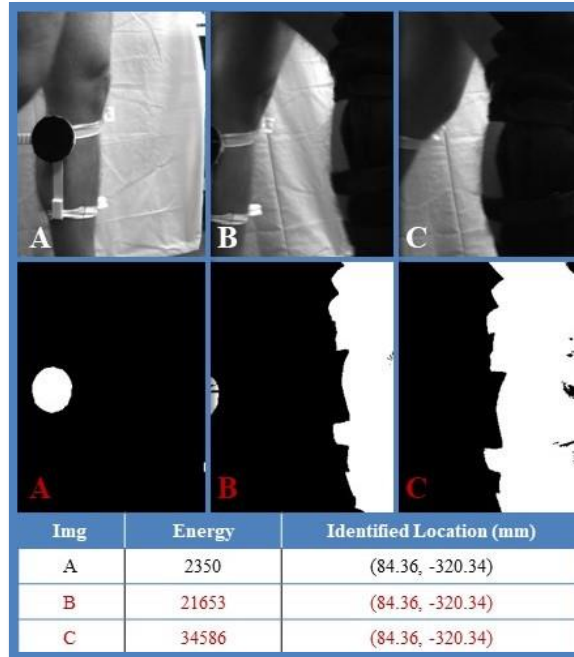


Figure 36: Particle analysis testing results for having no predictive algorithm.

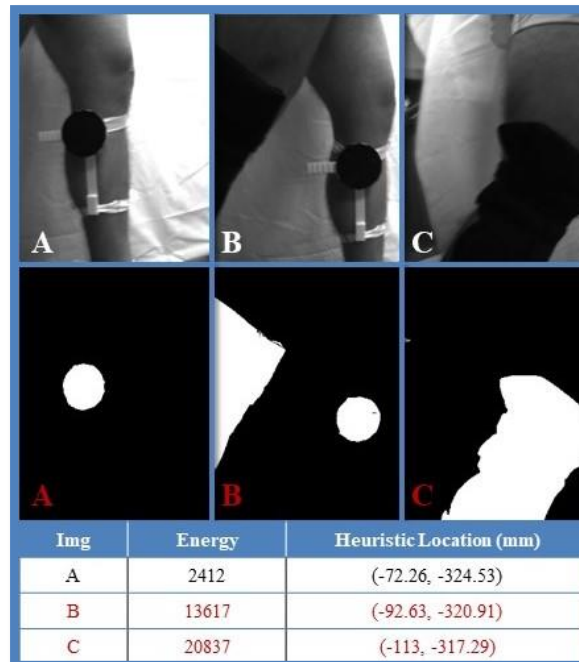


Figure 37: Particle analysis testing results for using the heuristic prediction method.

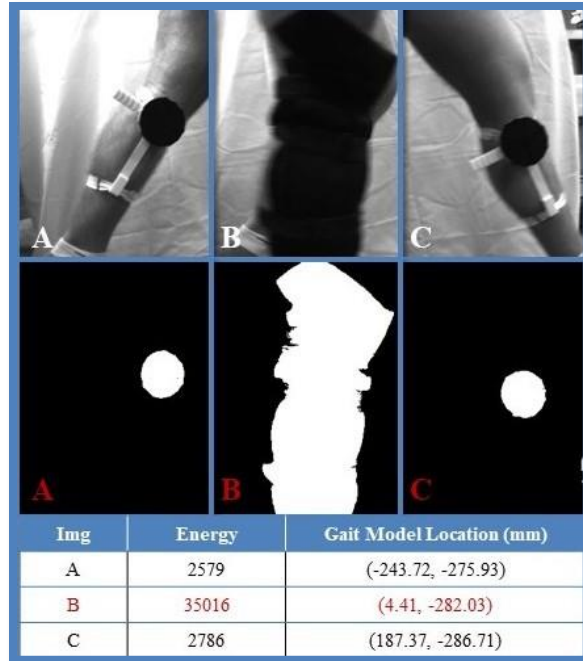


Figure 38: Particle analysis testing results for using the gait model prediction method.

Table 6: Percentage difference in identified locations between particle analysis image frames for the three prediction methods. Demonstrates the general magnitude of the required axis motion between individual frames that were studied.

Image Step	Difference in Location (%)					
	None		Heuristic		Gait Model	
	X	Z	X	Z	X	Z
A → B	0	0	28.19	1.12	101.81	2.21
B → C	0	0	21.99	1.13	41.49	1.66

The first row in each of these figures displays the image as seen through the Basler camera, while the second row displays what the particle analysis process obtains and operates with. The energy level is also provided in the accompanying tables which is used in Image Energy Profiling.

SKTA Performance

By expanding the timescale over which the tracking is monitored, it is possible to see the general behaviour of the Supervisory Knee Tracking Algorithm as a whole. Pulling from the same data that was obtained while testing the Gait Model within the Particle Analysis operation, it is possible to have some understanding of how the combined sections of this research are performing. It is worthwhile to reiterate that this performance testing is occurring under simulated x-ray conditions rather than the actual x-ray conditions that it was intended for.

Figure 39 provides this broader look through a sequence of images employing the Human Gait Model for predictive tracking. The first and third rows depict the images the Basler camera is actually registering while the second and fourth are the processed images used with the particle tracking. The bounds for Image Energy Profiling were 1,000 minimum and 11,500 maximum. Any image with an energy level outside these bounds would trigger the predictive tracking as indicated in the figure's attached table with red listings.

It is interesting to note that the motion commands required to move from one frame to the next are spread over a range rather than being fairly consistent. Direct tracking provided commands as small as six millimeters and as large as over one hundred millimeters in motion. Likewise, the prediction motion commands are just as varied. While this sequence possesses some image segments where the primary leg is not within the field-of-view at all (not even

























Img	Energy	Gait Model Location (mm)	Motion Commands (mm)
 A	2007	(56.06, -302.28)	(6.49, 4.81)
 B	2174	(162.99,-252.78)	(106.93, 49.50)
 C	2490	(227.19, -303.64)	(64.21, -50.86)
 D	2709	(220.65, -333.25)	(-6.54, -29.61)
 E	33965	(99.96, -284.76)	(-120.69, 48.49)
 F	13288	(114.72, -286.82)	(14.76, -2.07)
 A			
 B			
 C			
 D			
 E			
 F			
 G	34748	(90.62, -289.59)	(-24.11, -2.77)
 H	25375	(-77.67, -291.18)	(-168.29, -1.59)
 I	14103	(-112.21, -287.79)	(-34.54, 3.38)
 J	23624	(-106.24, -244.98)	(5.97, 42.81)
 K	33871	(-144.75, -284.60)	(-38.51, -39.61)
 L	1821	(-230.26, -303.25)	(-85.51, -18.65)
 G			
 H			
 I			
 J			
 K			
 L			

Figure 39: Full image sequence demonstrating the behaviour of the overall algorithm created for particle analysis tracking. The images listed on the table in red represent instances without direct tracking (considered part of the crossover).

behind the auxiliary leg), it does demonstrate that the SKTA has the ability to regain lost tracking using the Human Gait Model's predictions.

In an effort to better compare the behaviour of the Human Gait Model alongside the Particle Analysis approach, the actual motion of the axes during operation was compiled to demonstrate the overall performance of the Gait Model alongside that of the TFS version 1's Heuristic Prediction and the currently used No Prediction method in the TFS version 2. Figure 40-42 display the motion of the X- and Z-Axes over a timescale in milliseconds for tracking with Pattern Matching.

The first figure, Figure 40, is the complete timeline while the following two are closer views of the predictive regions (shaded areas of the graphs). Please note that the Heuristic Prediction's motion and the No Prediction's motion were calculated from the recorded encoder readings rather than recorded directly. This was done in order to show the effects of the three different prediction approaches side-by-side. The Heuristic's values were determined using the same method for calculating the prediction method's velocity vector (last two known points determined the direction and speed of motion). As no motion would take place at all during the No Prediction method, the relevant encoder values were held constant for these corresponding locations upon the graphs.

What these demonstrate is that, for the small time segments of prediction for pattern matching, there is only some difference between the heuristic approach and the gait modeling approach. Only two instance in the x-direction might have resulted in unrecoverable tracking, but as the largest of them is only a span of approximately twenty millimeters, the possibility of regaining tracking would be present. The z-direction, however, demonstrates several points of concern for the heuristic to regain tracking. Particularly, the third as well as the final periods of

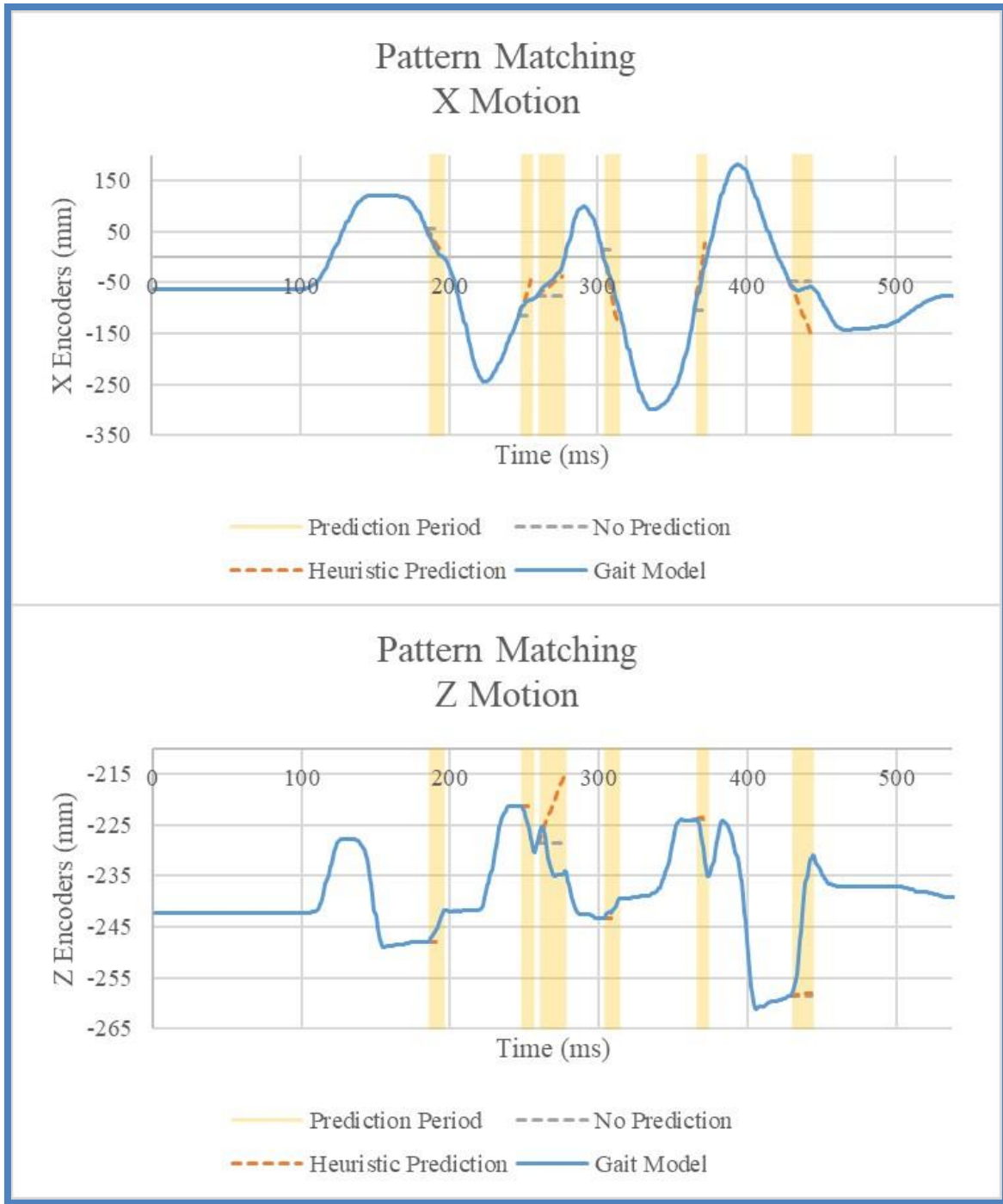


Figure 40: Encoder recorded motion of the three prediction methods (Gait Model, Heuristic, and No Prediction) employing Pattern Matching for tracking.

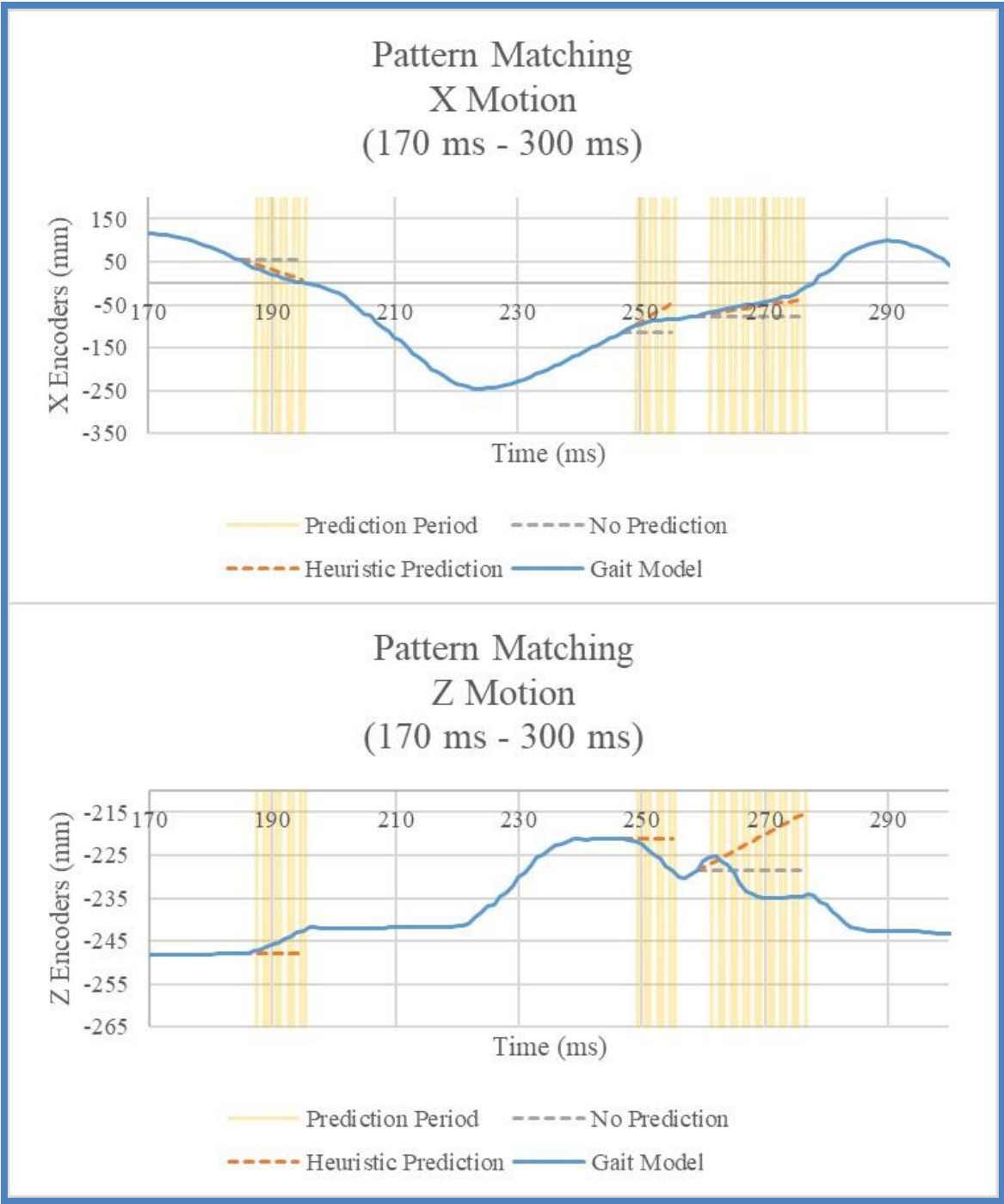


Figure 41: Closer look at the encoder recorded motion of the three prediction methods employing Pattern Matching for tracking. Looks at three regions of prediction between the 170 ms mark and the 300 ms mark.

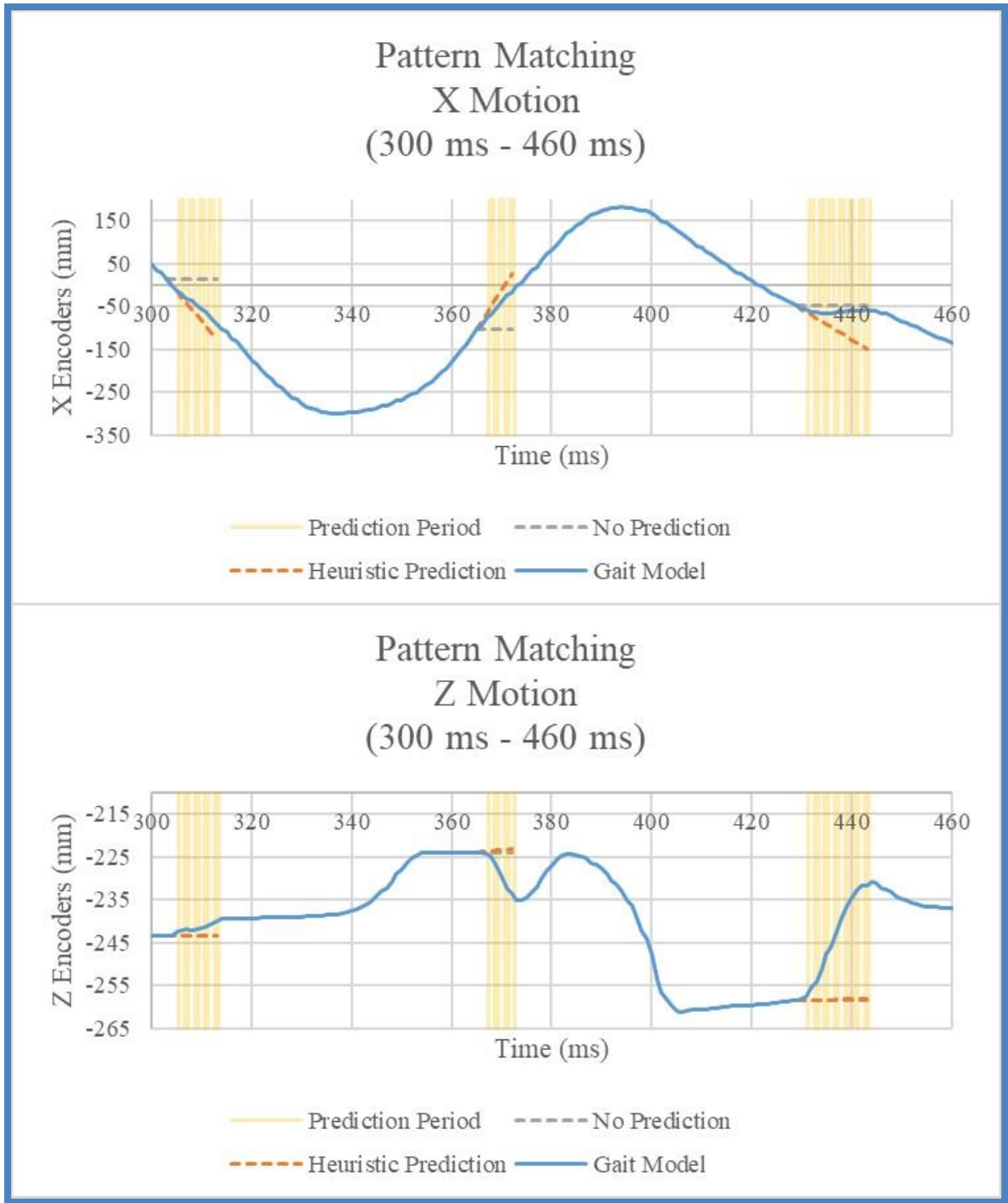


Figure 42: Closer look at the encoder recorded motion of the three prediction methods employing Pattern Matching for tracking. Looks at the three regions of predicted motion between the 300 ms mark and the 460 ms mark.

prediction demonstrate the probability of a significant difference between where the prediction would have taken the axes and where the target was actually reacquired. On the opposing hand, the gait model was capable of continually recapturing the target to continue tracking.

When Particle Analysis was similarly tested, it was first performed without the auxiliary knee affecting the binary image directly. Instead, it simply obscured the primary target from view as it passes during crossover making the target appear to “vanish” from the binary image.

Figure 43-45 are the recorded motions of the axes during this stage of testing the Particle Analysis approach. Once again, the behaviour observed is similar enough to the pattern matching results that the simpler heuristic approach to prediction might prove just as usable as the gait model. With the short prediction periods, the gait model’s x-direction motion almost coincides with that of the heuristic’s motion. On the other hand, the gait model does provide a more accurate motion control in the z-direction as demonstrated in the figures.

Lastly, the emulated x-ray imaging setup discussed in Chapter 6 using a dark cloth around the auxiliary knee was employed allowing it to directly affect the binary images and more closely emulate the original behaviour of x-ray images. The effect of the auxiliary knee’s inclusion within the binary images caused a drastic change in the outcome of tracking (see Figure 46). Rather than a continue cycle of active and predictive tracking previously observed, the results showed an eventual failure of the gait model and SKTA.

It is speculated that this failure was caused by the nature of the simulated x-ray images. During actual x-ray imaging of the knee crossovers, the auxiliary knee does not simply appear within the binary image as a solid mass when entering the image frame. Instead, it is a gradual build of the pixel mass caused by the gradual overlapping of tissue, increasing the density within the x-ray image. These trials utilized a method that, while still providing a means of representing

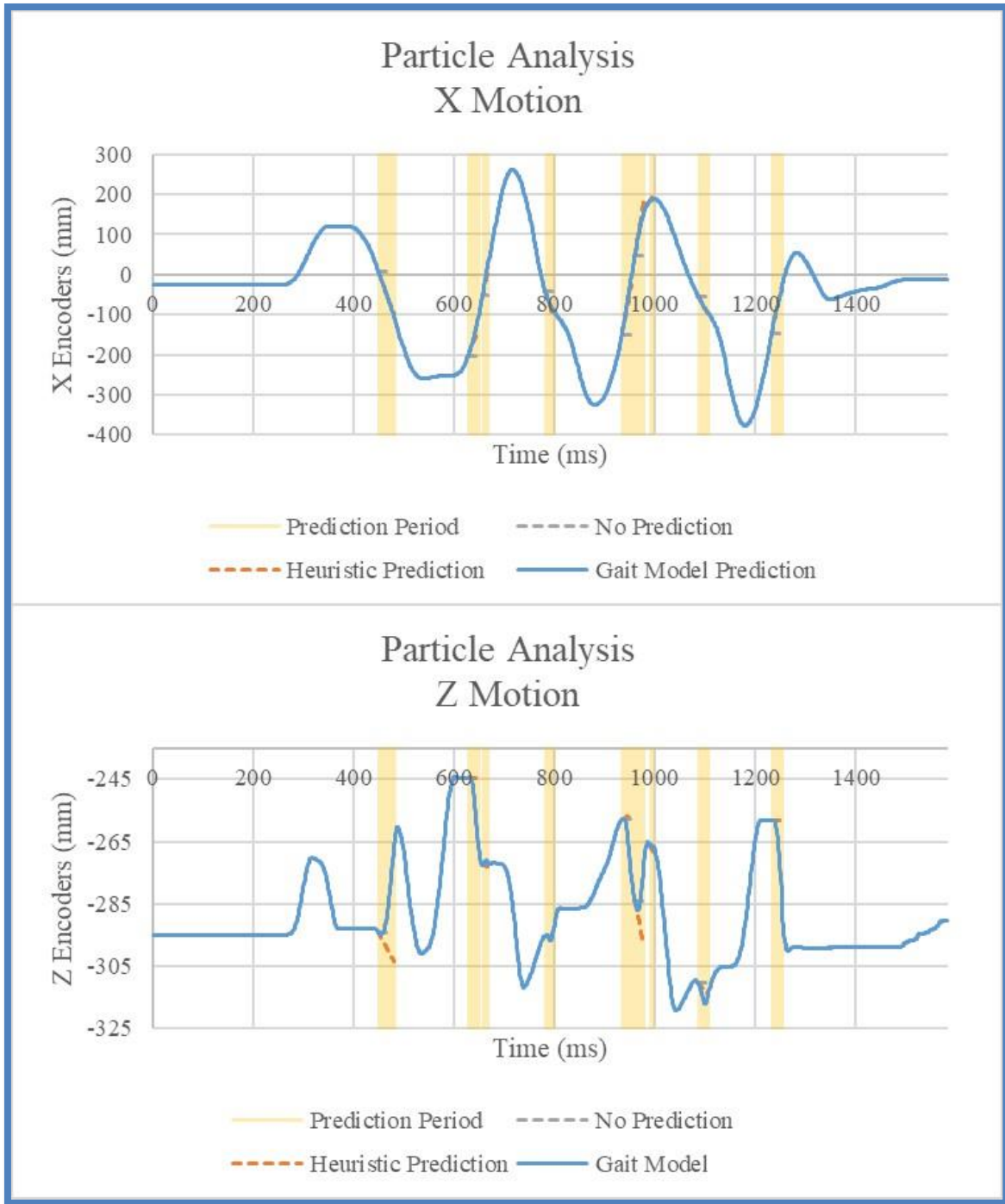


Figure 43: Encoder recorded motion of the three prediction methods (Gait Model, Heuristic, and No Prediction) employing Particle Analysis method for tracking without inclusion of the auxiliary knee within the binary image.

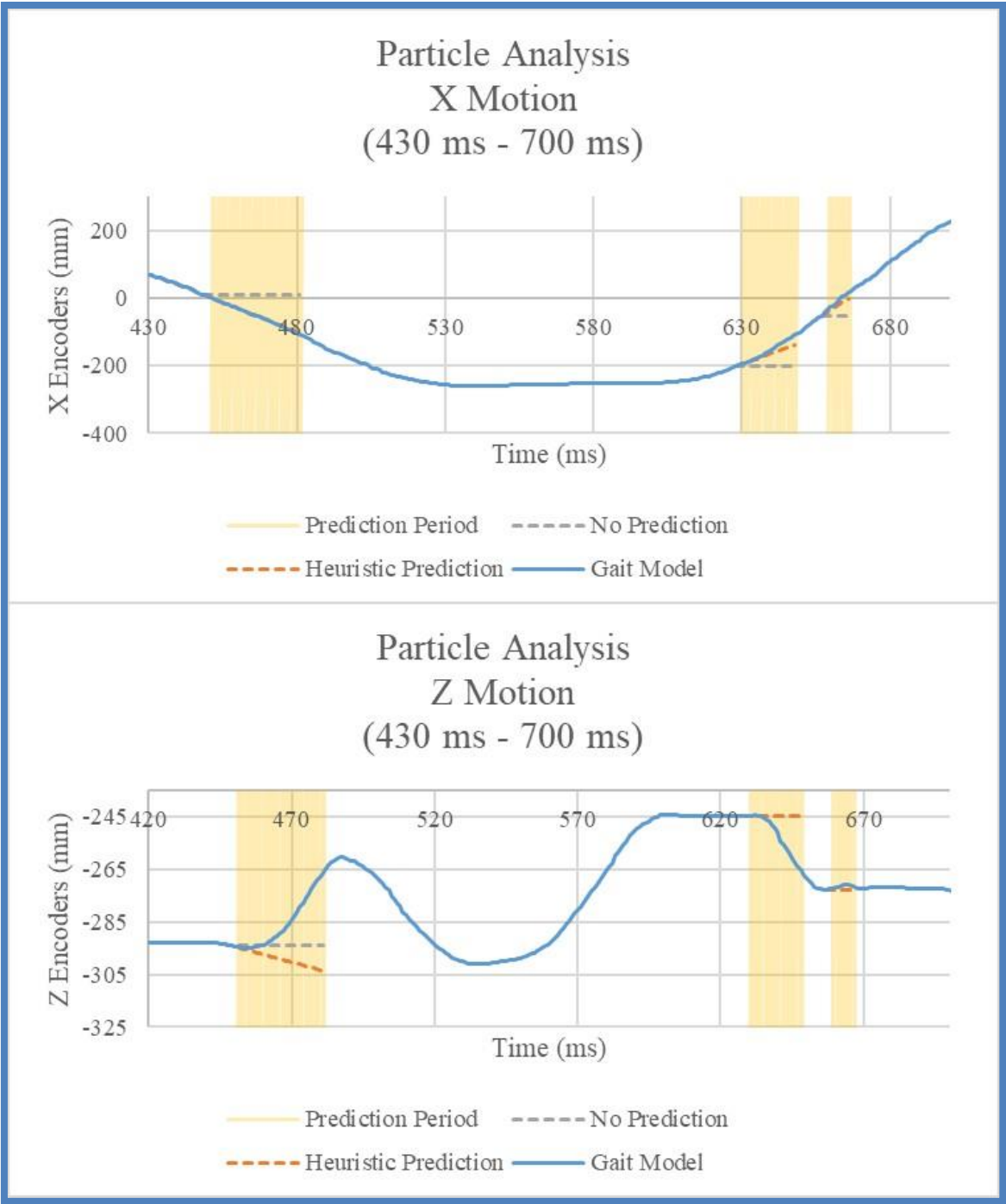


Figure 44: Closer look at the encoder recorded motion of the three prediction methods employing Particle Analysis for tracking. Looks at the three regions of predicted motion between the 430 mS mark and the 700 mS mark.

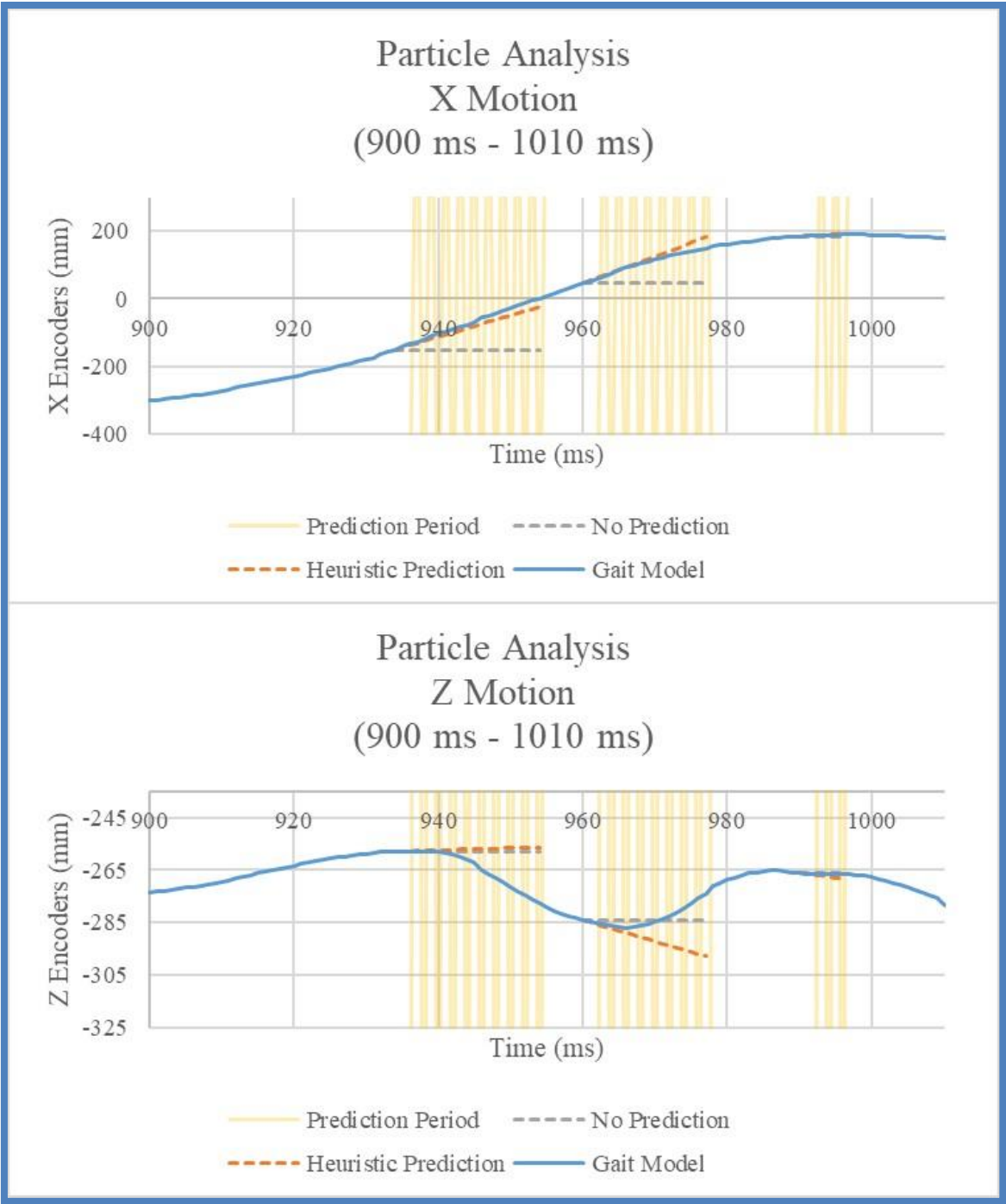


Figure 45: Closer look at the encoder recorded motion of the three prediction methods employing Particle Analysis for tracking. Looks at the three regions of predicted motion between the 900 ms mark and the 1010 ms mark.

this large mass of pixels that would occur during a crossover, does not precisely behave like the x-ray images would.

However, while eventually failing, what this test also demonstrates is that the inclusion of a human gait model is a potentially sound strategy for enhancing tracking. In the initial moments of tracking, the gait model was able to handle prediction fairly well. Figure 47 provides this closer look that demonstrates how the gait model provided means of moving through the crossovers without resulting in significant jumps. In addition, it shows how both the Heuristic Prediction and the No Prediction approaches would have either required said significant jumps or potentially failed altogether.

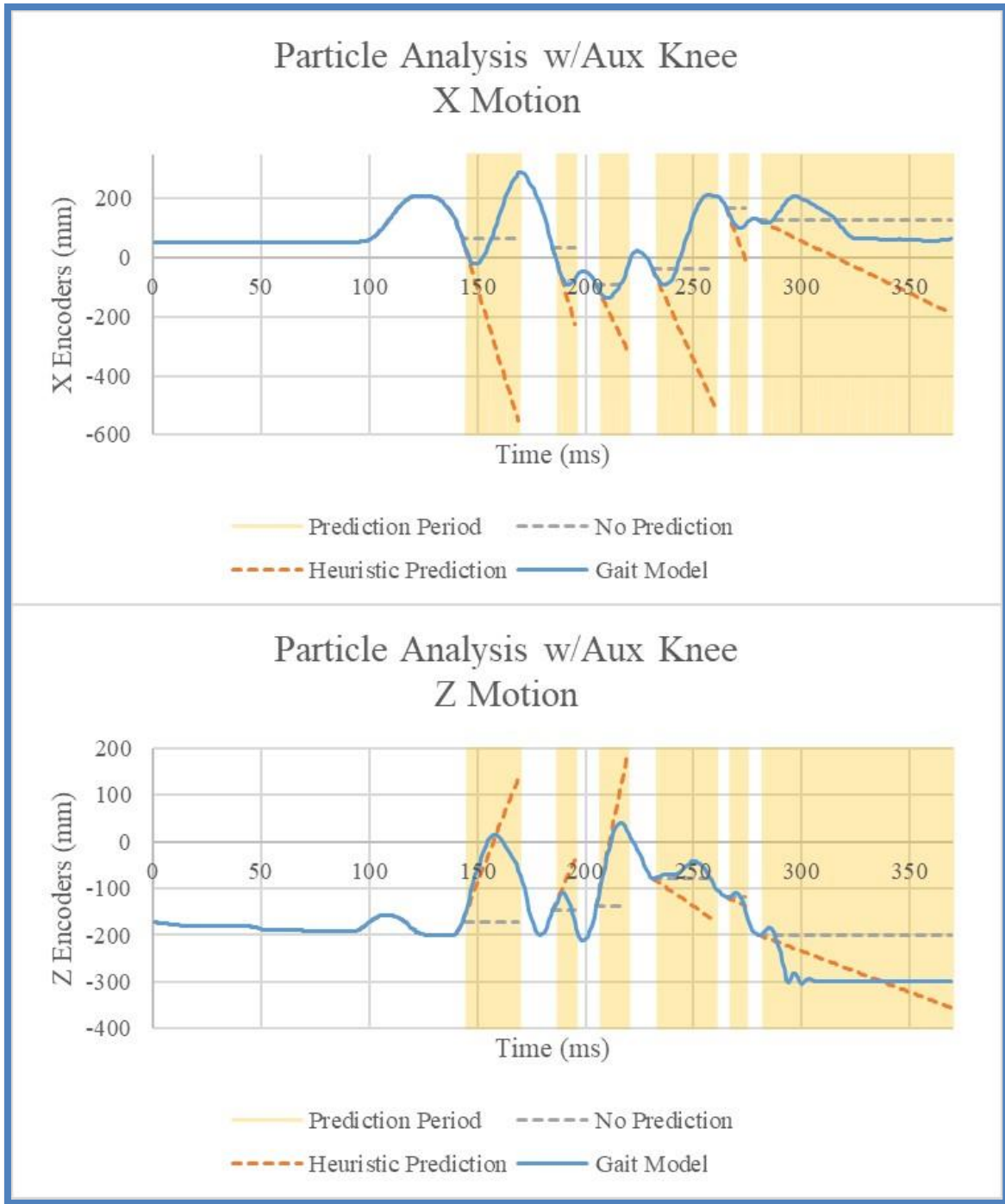


Figure 46: Encoder recorded motion of the three prediction methods (Gait Model, Heuristic, and No Prediction) employing the Particle Analysis methodology.

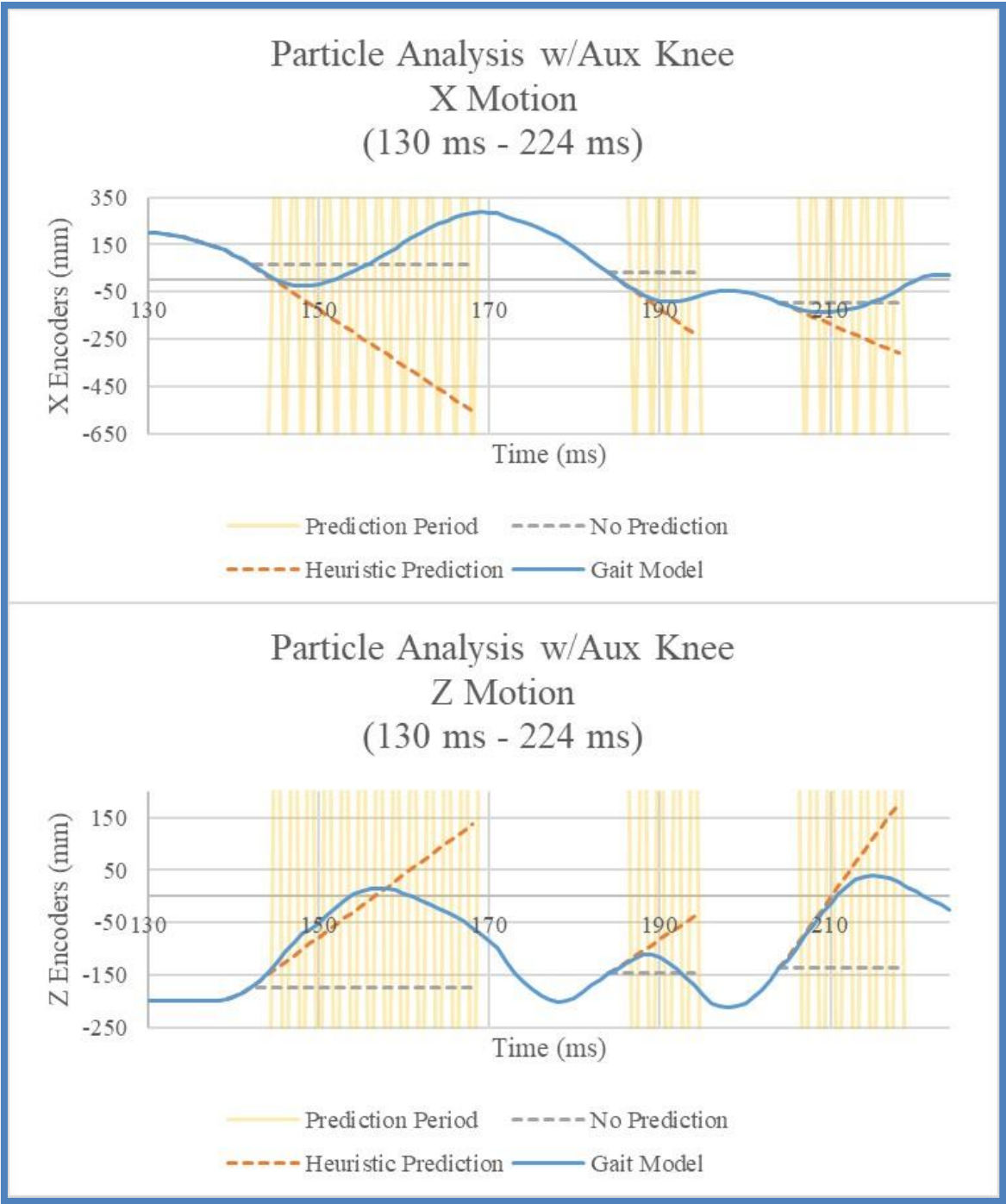


Figure 47: Closer look at the initial predictive regions of the Particle Analysis method. Displays the first three regions between the 130 ms mark and the 224 ms mark.

CHAPTER EIGHT – CONCLUSIONS AND FUTURE WORK

While the Pattern Matching algorithm is usable on the original Tracking Fluoroscope System's imaging system (which uses smaller image intensifier x-ray images), the updates that took place to the TFS replaced the smaller x-ray images with larger, more data intensive images. This situation led to the need for an alternative tracking method for x-ray servo tracking. The tracking approach presented in this work has demonstrated several possible advantages over the original pattern matching that has been in present use on the TFS.

Image Processing

The most prominent advantage the new Particle Analysis method has demonstrated is its processing speed. Requiring an average of only nine milliseconds per image frame, the particle analysis would only consume just over half of the sixteen milliseconds allotted between acquiring the image and outputting commands to the axes. It is approximately five times faster than the original Version 1 Pattern Matching algorithm, and it is approximately three times faster than the updated Pattern Matching algorithm in the TFS Version 2's code.

In addition, the particle analysis does not rely on user generated templates which introduce the potential for error, but it instead uses simply what is provided within the obtained image. It is also capable of targeting the entire implant, unlike the pattern matching method. Without being restricted to a single component of the implant, the possibility of actively tracking with part of a component outside the x-ray field-of-view is drastically reduced.

The particle method has been shown to carry a single disadvantage in its image processing however. The Image Energy Profiling, which controls when to switch between active and predictive tracking, is sensitive enough to trigger a crossover occurrence the moment the auxiliary knee begins to enter the x-ray image frame when the pixel count rises. While this

situation does provide the means for accurately identifying the opposite leg's motion in relation to the primary leg, it forces the SKTA to switch from active tracking to prediction for longer periods of time than the pattern matching would.

In summary, while the image processing segment requires longer periods of prediction, this is vastly outweighed by the drastic increase in processing speed, the ability to track the implant as a whole, and the independence from user generated templates.

Human Gait Modeling

The heuristic algorithm from the original Tracking Fluoroscope System is a basic calculation of speed and direction using the last two known locations of the target. On the other hand, the Human Gait Model developed for this research takes into account the slight curvature in the knee's path as it rotates about the hip joint. Comparing the two predictive methods has demonstrated both advantages and disadvantages.

Firstly, the key advantage to the gait model is its ability to accommodate the knee's more pendulum-like motion over the heuristic algorithm's direct path assumption. The image processing designed in the first phase of this research demonstrated longer periods of predictive tracking which is the downfall of the heuristic approach. On miniscule scales, the small-angle assumption provides a basis for the aforementioned heuristic algorithm to work as it is only a matter of several frames to predict over. With the image energy profiling extending this timeframe to a broader range, the actual curvature motion of the knee is now more prominent, and therefore the ability of the Gait Model to account for this different motion is important.

On the other hand, the singular disadvantage observed is the complexity of the gait model. While great strides were taken to build a model with as little processing intensity as possible, it is still significant when compared to the simplicity of the heuristic algorithm. To

implement the heuristic algorithm within the code, it is a simply matter of repeating the previously used motion commands (calculated from the last two known points). In contrast, the Human Gait Model requires several ongoing calculations in order to update its current awareness of the knee's position, and then a different set of calculations in order to provide the predicted motion commands. This means that, for small distances such as experienced with the pattern matching, the increase in processing time may come at more of a cost than the associated benefit in accuracy of position prediction.

In general, the heuristic algorithm is adequate for use with pattern matching, but the Human Gait Model's accommodation of the knee's curved path provides a basis for handling the longer prediction periods likely to occur with the Particle Analysis.

Supervisory Knee Tracking Algorithm

Observing the behaviour demonstrated in the encoder recorded motion figures of Chapter 7, it is clear that the Human Gait Model's prediction is more accurate than the simple heuristic algorithm. Without the auxiliary knee actually appearing within the tracking image frame (during pattern matching and the initial particle analysis tests), the scale of predicted motion is small enough that it requires a much closer inspection to see any definitive differences. On the other hand, once the auxiliary knee's inclusion to the processed image is actually accounted for as it would be during x-ray, the behaviour seen is much more prominent.

Reconsidering the close-up of the initial predictive segments (please refer back to Figure 47, Chapter 7), it is clear that there would have been some difficulties for the system to regain tracking on the other side of the crossover. However, after these first few successful moments of tracking and prediction, the system quickly lost accurate tracking altogether. Part of this tracking loss is believed to be due to environmental lighting casting down shadows as well as difficulties

in maintaining a light background for the acquired visual images. A large part of the loss is also believed to be attributed to the manner in which the auxiliary knee would enter the binary images. As previously mentioned, the “sudden” appearance of a mass at the borders of an image is not behaviour reminiscent to that of an actual binary x-ray image. This observed behaviour would often cause sudden large jumps in the identified centroid’s location, going from on-target to a median point in between it and the image frame’s border before the Image Energy Profiling could trigger prediction.

Overall, the entire particle analysis method for x-ray joint tracking would appear to be a usable, if not yet perfect, approach to tracking a knee implant through the data intensive digital x-ray images obtained on the Tracking Fluoroscope System. The image processing possesses the speed required for real-time usage and the ability to track from the entire implant rather than only part, and the human gait model can predict the more realistic curved path of the target during a subject’s walking cycle. Its demonstrated downside is its greater reliance on predictive tracking due to earlier identification and prolonged periods of crossovers.

Differences Between Visual and X-Ray Particle Analysis

In order to complete this research, a compromise was needed. This being the fact that the entire particle analysis setup was modified to operate with a monochrome visual camera rather than an x-ray imager. This modification resulted in adjustments for the image processing setup and a point of note for the implementation of the gait model.

The adjustment made to the image processing setup centered around the computational architecture for the machine. The exclusion of the custom code allowed for the removal of the secondary computer. This custom code should be re-introduced for x-ray particle analysis which will necessitate the restructuring of the TFS’ architecture once again.

The gait model itself was not modified and is independent of the method used to obtain the tracking data it employs. The consideration, however, is its requirement for an initial calibration. This calibration will require a brief burst of x-ray at the beginning of subject testing, potentially anywhere from two to five seconds.

Future Recommendations

The inclusion of an RGB-D style sensor would be a potentially great benefit to this work. A 3-D sensor could provide an additional avenue for target tracking as well as the ability to verify the particle analysis tracking and gait model prediction positions. In this, it could also offer a means of real-time algorithm correction in order to “learn” each patient during testing.

Another point of interest for potential revisiting lay in terms of real-time updating of the gait model’s parameters. Schwarts and Rozumalski presented in [44] a method of estimating joint parameters by analyzing motion data. After tracking has begun, perhaps this technique could be employed to improve the dimensional estimations within the tracking program and thereby further refine the model’s accuracy.

LIST OF REFERENCES

1. Capek, K., et al., *Rossum's universal robots*. Prague, CZ, 1920.
2. Sciavicco, L. and B. Siciliano, *Modeling and control of robot manipulators*. Vol. 8. 1996: McGraw-Hill New York.
3. Dario, P., et al., *Robotics for medical applications*. IEEE Robotics & Automation Magazine, 1996. **3**(3): p. 44-56.
4. Bascle, B., et al. *Needle placement under X-ray fluoroscopy using perspective invariants*. in *Proceedings IEEE Workshop on Mathematical Methods in Biomedical Image Analysis. MMBIA-2000 (Cat. No.PR00737)*. 2000.
5. Hofstetter, R., et al., *Fluoroscopy as an imaging means for computer-assisted surgical navigation*. Computer Aided Surgery, 2010.
6. Schweikard, A. *Robotic Radiosurgery*. in *The Institution of Engineering and Technology Seminar on Robotic Surgery: The Kindest Cut of All, 2006. (Ref. No. 2006/11372)*. 2006.
7. Wang, Y.-F., D.R. Uecker, and Y. Wang, *A new framework for vision-enabled and robotically assisted minimally invasive surgery*. Computerized Medical Imaging and Graphics, 1998. **22**(6): p. 429-437.
8. Wen, R., et al. *Intraoperative visual guidance and control interface for augmented reality robotic surgery*. in *IEEE ICCA 2010*. 2010.
9. Cunningham, R.B., *Mechanical Design and Integration of a Tracking Fluoroscope System*. 2005.
10. Komistek, R.D., W.R. Hamel, and D.A. Dennis, *Method and apparatus for imaging tracking*. 2013, Google Patents.
11. Nycz, A. and W.R. Hamel, *The development of the Tracking Fluoroscope System*. IFAC Proceedings Volumes, 2009. **42**(13): p. 168-173.
12. Nycz, A., *Vision-Based Robot Control in the Context of Human-Machine Interactions*. 2012.
13. Preliasco, G.R., *Motion control for a tracking fluoroscope system*. 2005.
14. Young, M.A., *Modeling, Analysis, and Control of a Mobile Robot for In Vivo Fluoroscopy of Human Joints during Natural Movements*. 2014.
15. Nycz, A. and W.R. Hamel. *Active tracking control between a bio-robot and a human subject*. in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2009.
16. Nycz, A., M.A. Young, and W.R. Hamel. *A bio-robotics approach to real-time skeletal joint fluoroscopy during natural movements*. in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. 2011.
17. Zhang, R., C. Vogler, and D. Metaxas. *Human gait recognition*. in *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW'04. Conference on*. 2004. IEEE.
18. Wagg, D.K. and M.S. Nixon. *On automated model-based extraction and analysis of gait*. in *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*. 2004. IEEE.
19. Hamel, W., R. Komistek, and D. Dennis. *Results From a New Method for Obtaining in Vivo Fluoroscopic Arthroplasty Evaluations With Normal Patient Movement*. in *Orthopaedic Proceedings*. 2012. Orthopaedic Proceedings.
20. Burger, W., *Principles of Digital Image Processing : Core Algorithms*. 1.. ed, ed. M.J. Burge and SpringerLink. 2009, London: London : Springer-Verlag London.
21. Hutchinson, S., G.D. Hager, and P.I. Corke, *A tutorial on visual servo control*. IEEE Transactions on Robotics and Automation, 1996. **12**(5): p. 651-670.

22. Berryman, J.G., *Measurement of spatial correlation functions using image processing techniques*. Journal of Applied Physics, 1985. **57**(7): p. 2374-2384.
23. Jemilda, G. and S. Baulkani, *Capturing moving objects in video using Gabor and local spatial context model*. Asian Journal of Information Technology, 2016. **15**(5): p. 846-854.
24. Lara, L.S., *Bone tracking from X-Ray sequences*. 2009.
25. Cevher, V., et al., *Target tracking using a joint acoustic video system*. IEEE Transactions on Multimedia, 2007. **9**(4): p. 715-727.
26. Cipolla, R. and N.J. Hollinghurst, *Human-robot interface by pointing with uncalibrated stereo vision*. Image and Vision Computing, 1996. **14**(3): p. 171-178.
27. Ercan, A.O., A. El Gamal, and L.J. Guibas. *Object tracking in the presence of occlusions via a camera network*. in *IPSN 2007: Proceedings of the Sixth International Symposium on Information Processing in Sensor Networks*. 2007.
28. Verma, S., J. Kofman, and W. Xianghai. *Application of markerless image-based arm tracking to robot-manipulator teleoperation*. in *First Canadian Conference on Computer and Robot Vision, 2004. Proceedings*. 2004.
29. Burt, P.J., et al., *Object tracking with a moving camera*. Visual Motion, 1989., Proceedings. Workshop on, 1989: p. 2-12.
30. Cohen, I. and G. Medioni, *Detecting and tracking moving objects for video surveillance*. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1999. **2**: p. 319-325.
31. Machida, E., et al. *Human motion tracking of mobile robot with Kinect 3D sensor*. in *SICE Annual Conference (SICE), 2012 Proceedings of*. 2012. IEEE.
32. Wang, T., G. Liu, and W. Xie. *Visual servoing control of video tracking system for tracking a flying target*. in *Advanced Intelligent Mechatronics (AIM), 2011 IEEE/ASME International Conference on*. 2011. IEEE.
33. Loeb, G.E. and R. Davoodi, *Musculoskeletal Mechanics and Modeling*. Scholarpedia, 2016. **11**: p. 12389.
34. Begg, R.K. and R. Wytch. *A television-based system for recording the kinematics of human gait*. in *Proceedings of 17th International Conference of the Engineering in Medicine and Biology Society*. 1995.
35. Scafetta, N., D. Marchi, and B.J. West, *Understanding the complexity of human gait dynamics*. Chaos: An Interdisciplinary Journal of Nonlinear Science, 2009. **19**(2): p. 026108.
36. Seth, A., et al., *OpenSim: a musculoskeletal modeling and simulation framework for in silico investigations and exchange*. Procedia Iutam, 2011. **2**: p. 212-232.
37. Hatze, H., *A comprehensive model for human motion simulation and its application to the take-off phase of the long jump*. Journal of biomechanics, 1981. **14**(3): p. 135-142.
38. Taga, G., *A model of the neuro-musculo-skeletal system for human locomotion*. Biological cybernetics, 1995. **73**(2): p. 97-111.
39. West, B.J. and N. Scafetta, *Nonlinear dynamical model of human gait*. Physical Review E, 2003. **67**(5): p. 051917.
40. Hanavan Jr, E.P., *A mathematical model of the human body*. 1964, DTIC Document.
41. Kadaba, M.P., H. Ramakrishnan, and M. Wootten, *Measurement of lower extremity kinematics during level walking*. Journal of orthopaedic research, 1990. **8**(3): p. 383-392.

42. Qianxiang, Z., et al., *Geometric dimension model of virtual human for ergonomic analysis of man-machine system*, in *Advances in Physical Ergonomics and Safety*. 2012, CRC Press. p. 290-300.
43. Canny, J., *A Computational Approach to Edge Detection*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1986. **PAMI-8**(6): p. 679-698.
44. Schwartz, M.H. and A. Rozumalski, *A new method for estimating joint parameters from motion data*. Journal of biomechanics, 2005. **38**(1): p. 107-116.

APPENDICES

Appendix A – Additional Phase One Results' Figures

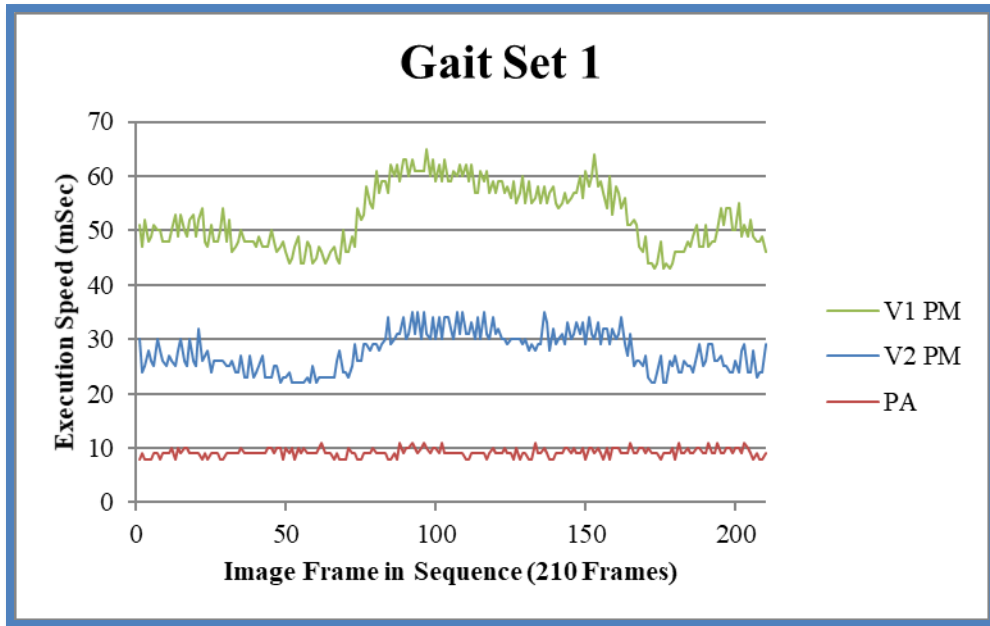


Figure 48: Processing speed results for the pattern matching and particle analysis algorithms on the first sequence set of x-ray images.

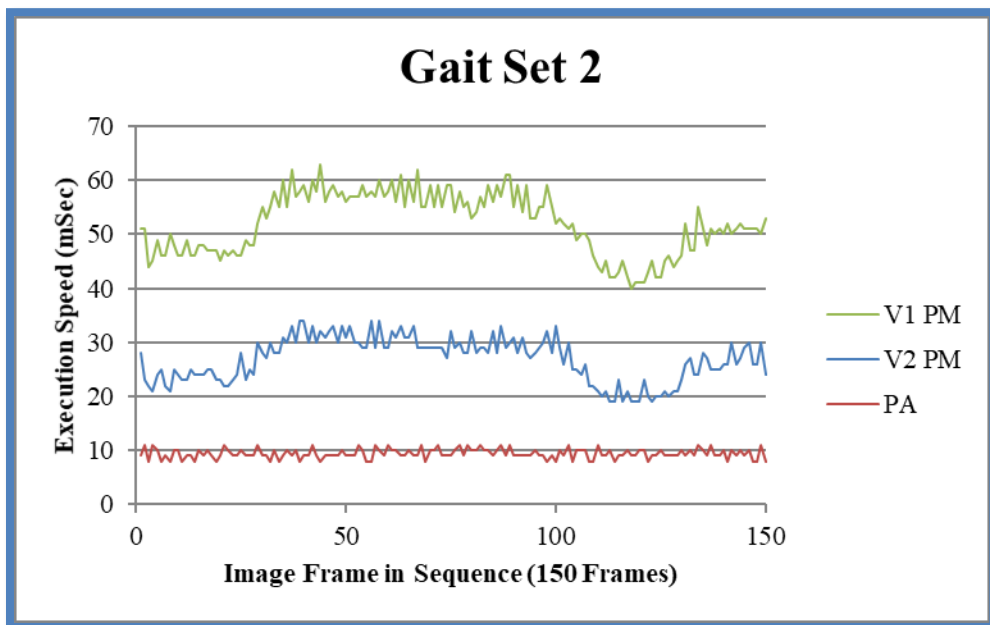


Figure 49: Processing speed results for the pattern matching algorithms and the particle analysis algorithm on the second sequence set of x-ray images.

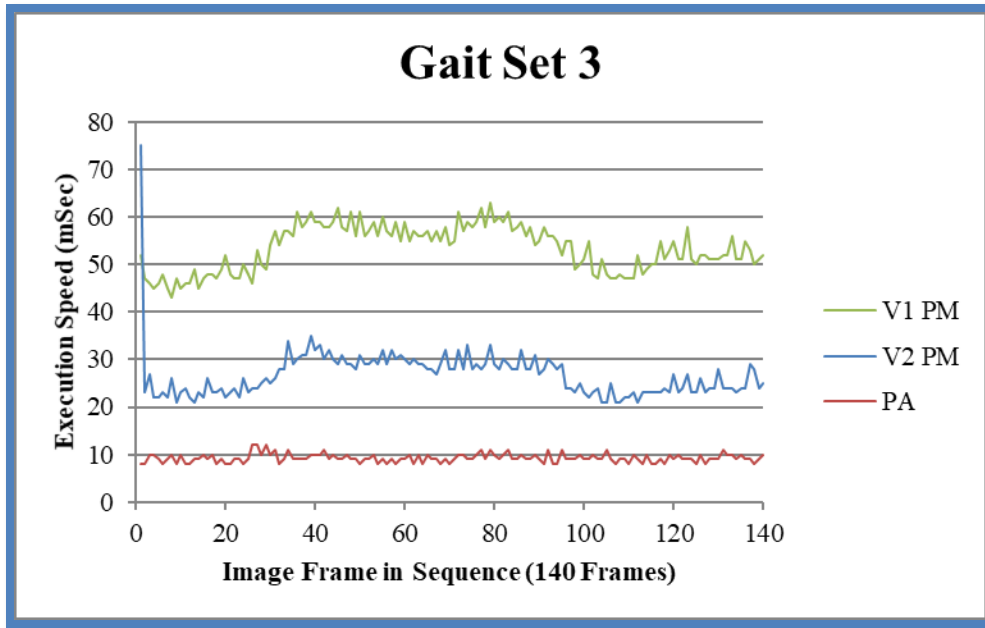


Figure 50: Processing speed results for the pattern matching and particle analysis algorithms on the third sequence set of x-ray images.

Appendix B – TFS Code

Main Control

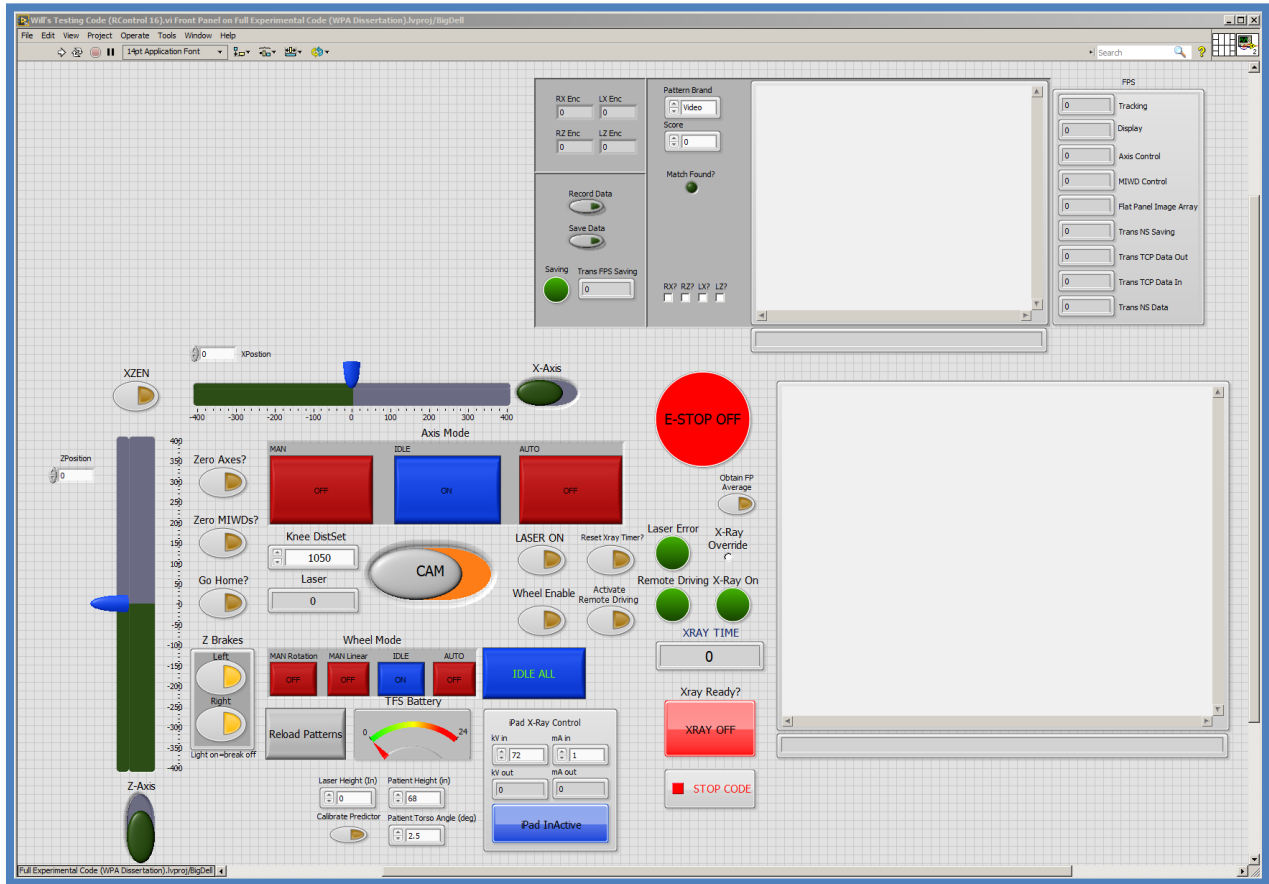


Figure 51: Main Control interface for the Tracking Fluoroscope System.

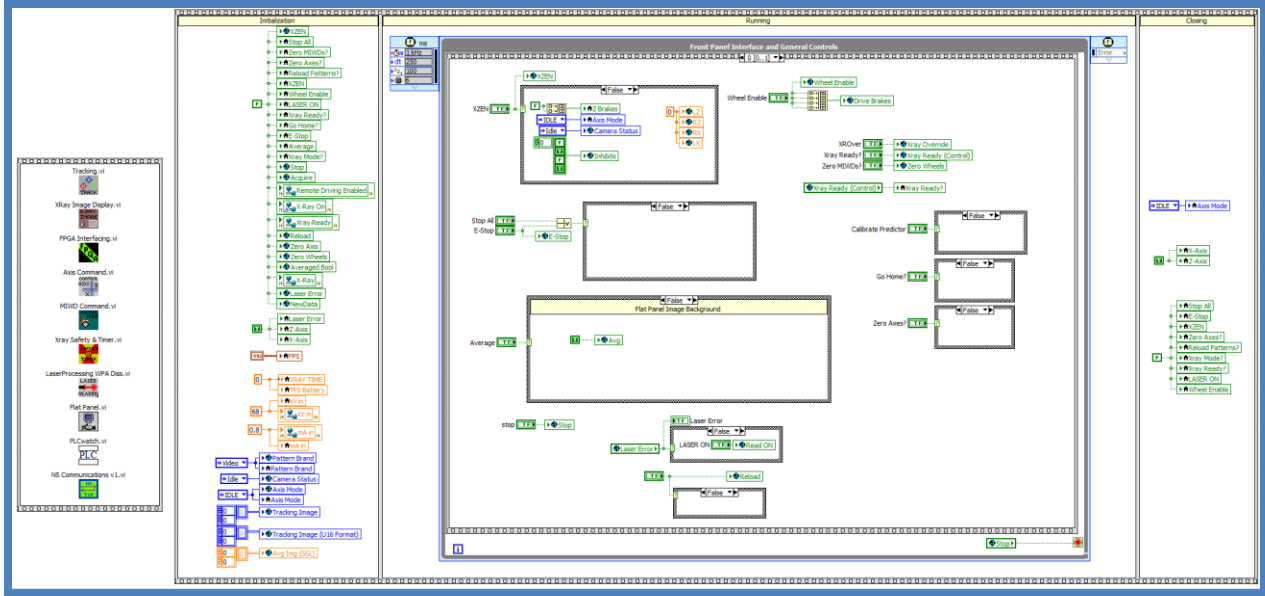


Figure 52: Main Control code for the TFS. Panel 1-1.

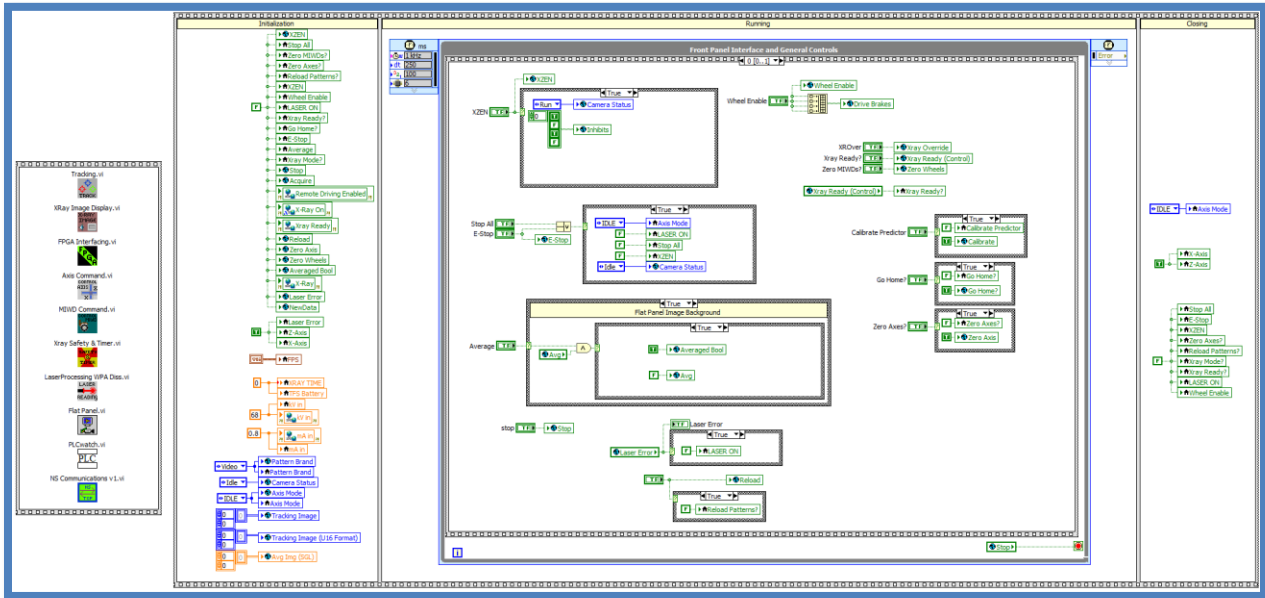


Figure 53: Main Control code for the TFS. Panel 1-2

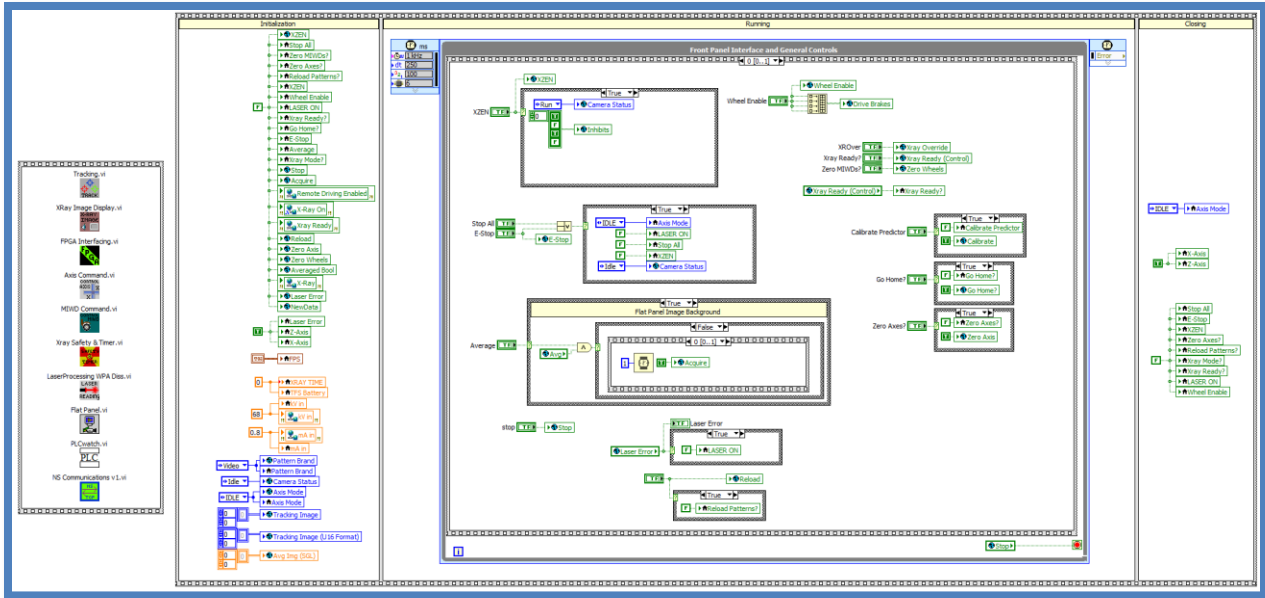


Figure 54: Main Control code for the TFS. Panel 1-3.

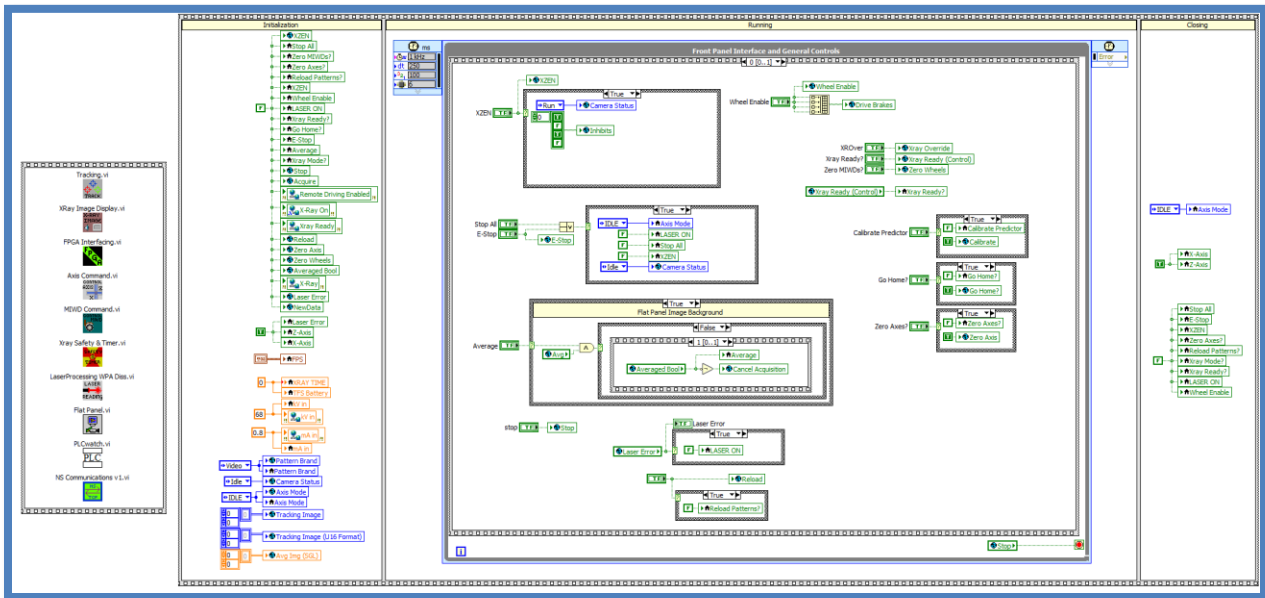


Figure 55: Main Control code for the TFS. Panel 1-4.

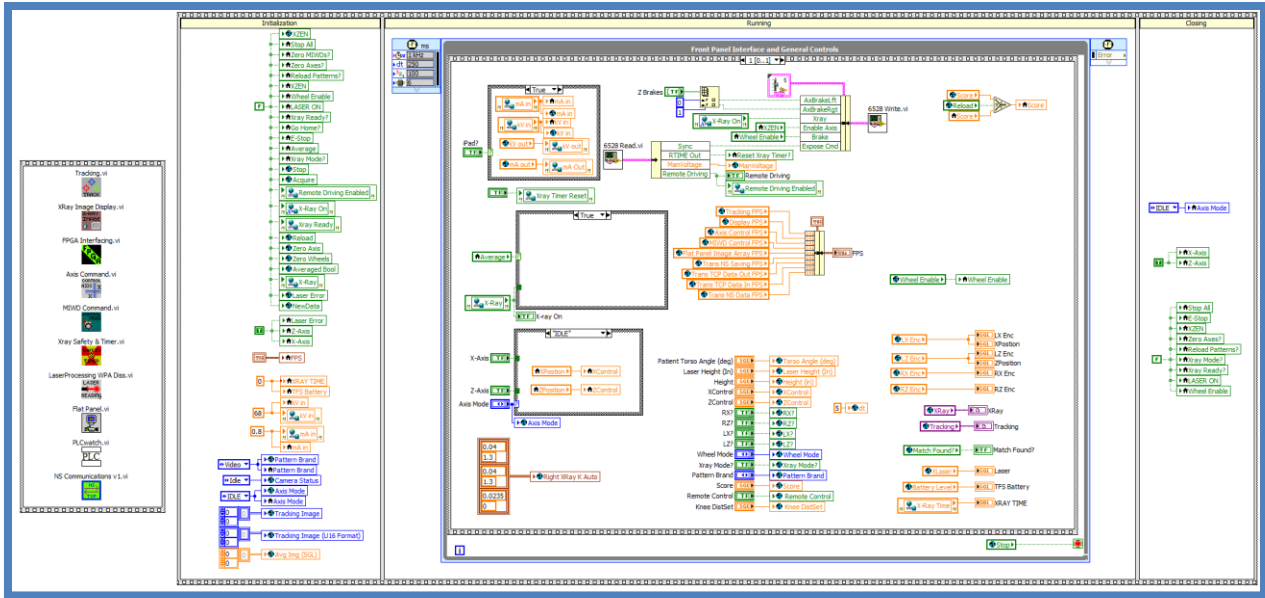


Figure 56: Main Control code for the TFS. Panel 2-1.

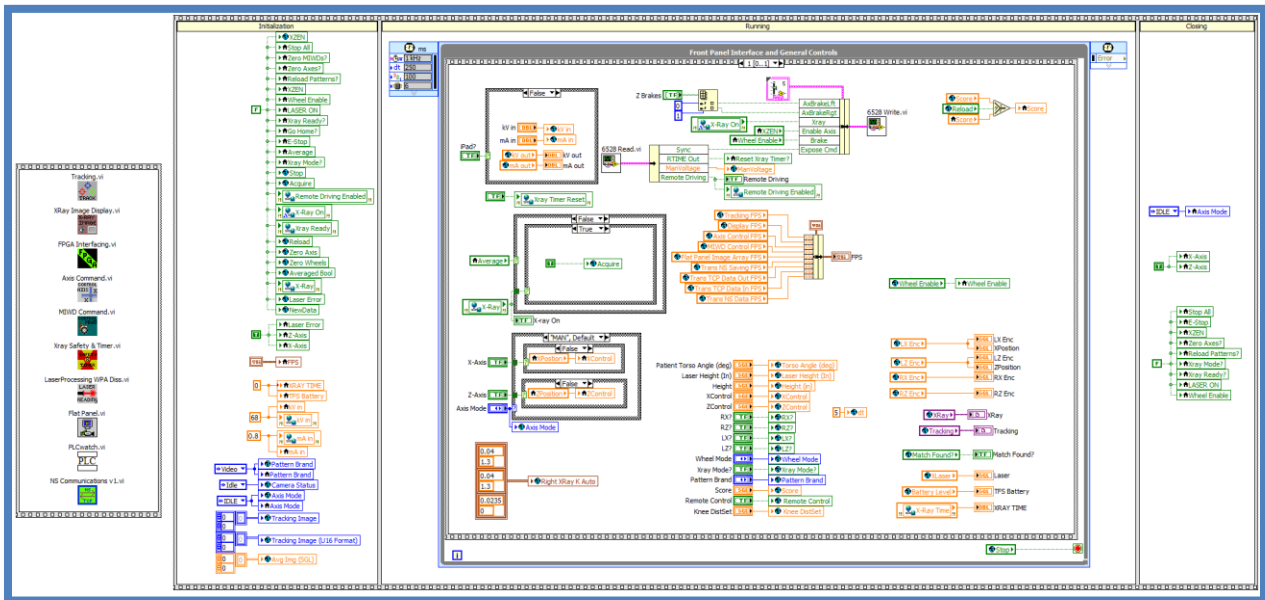


Figure 57: Main Control code for the TFS. Panel 2-2.

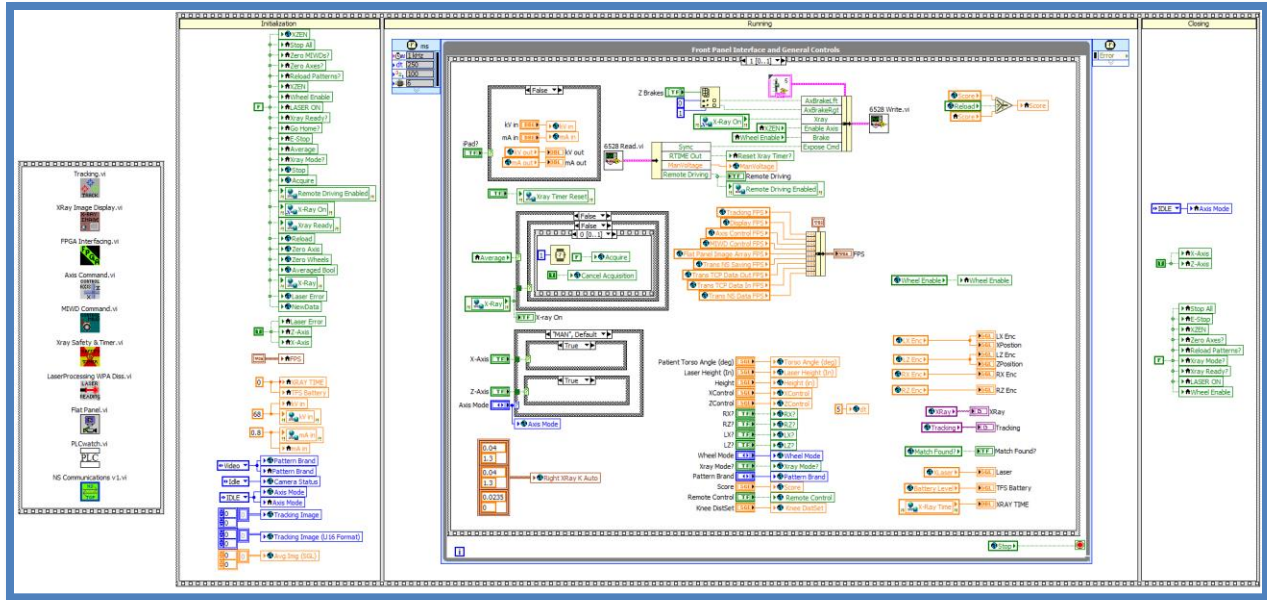


Figure 58: Main Control code for the TFS. Panel 2-3.

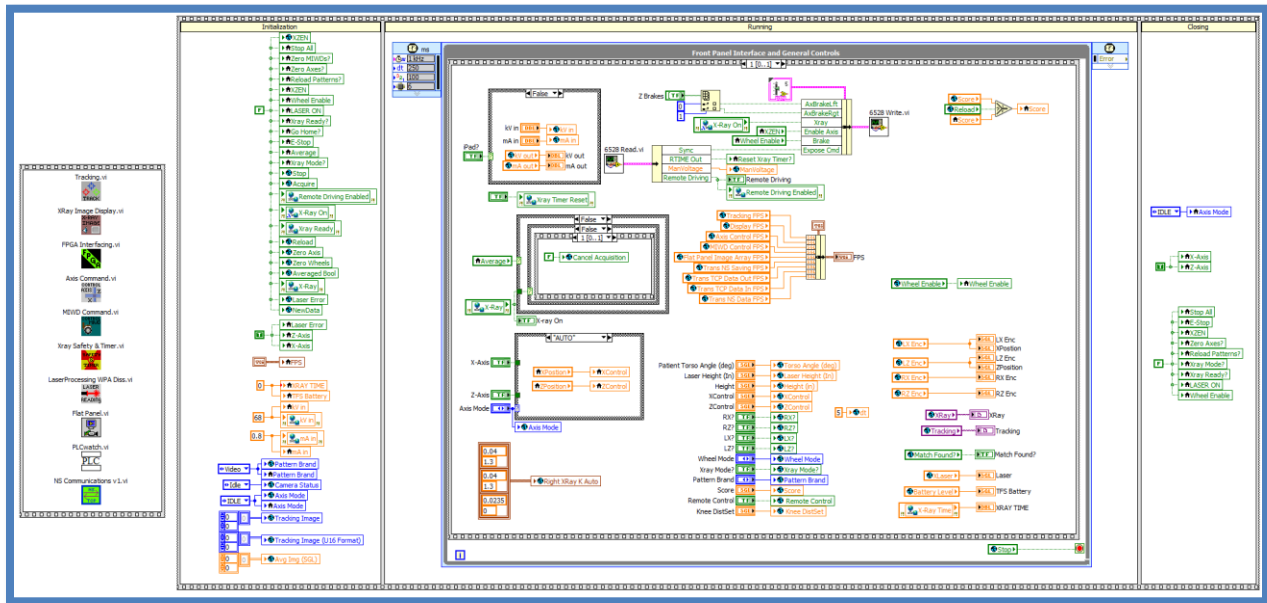


Figure 59: Main Control code for the TFS. Panel 2-4.

Tracking

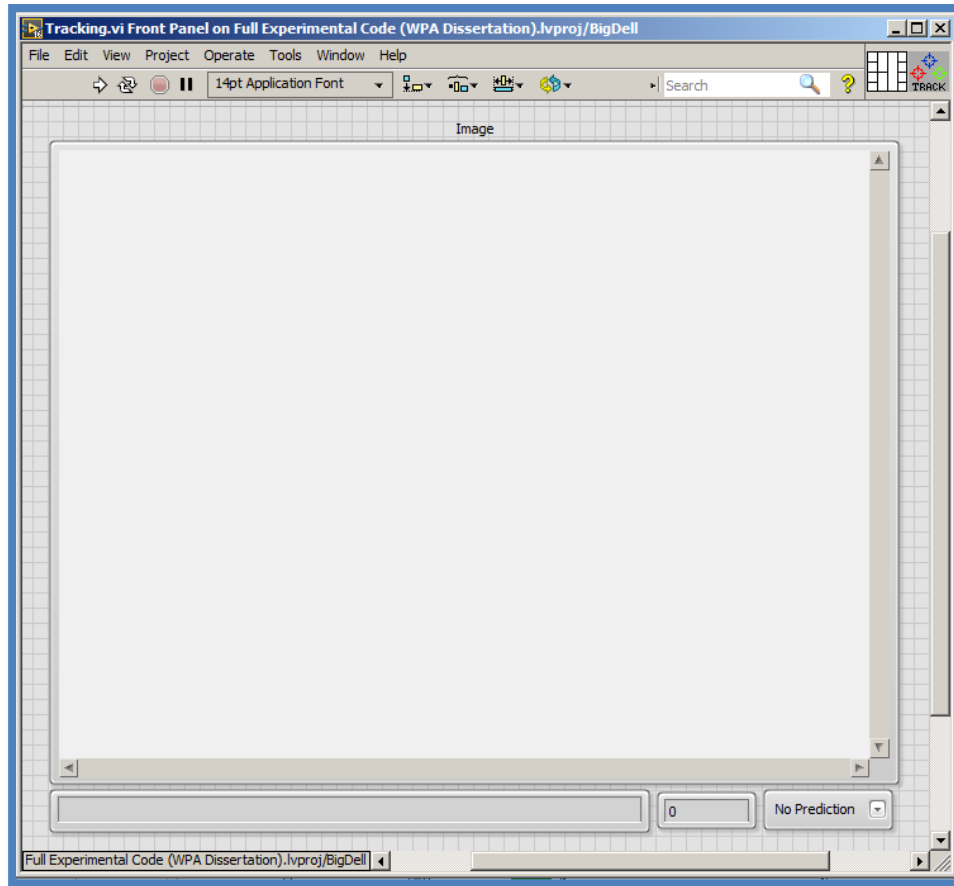


Figure 60: Tracking Code interface for the TFS.

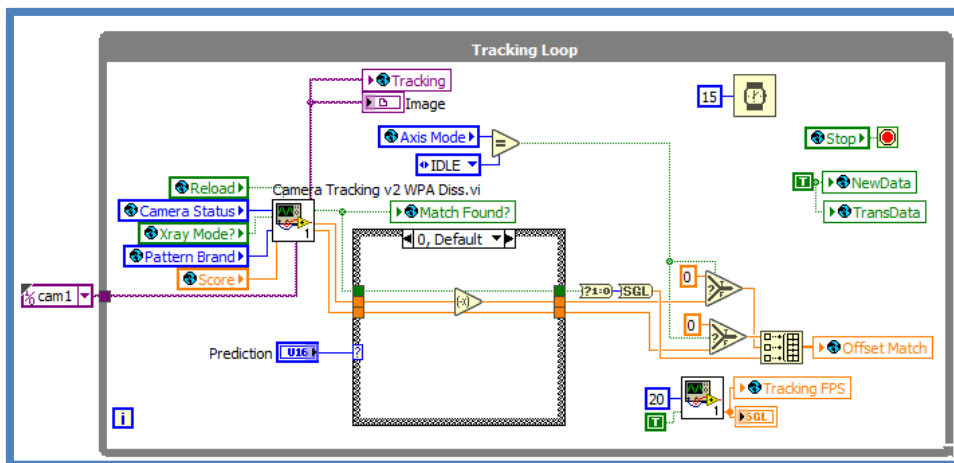


Figure 61: Tracking code for the TFS.

Camera Tracking

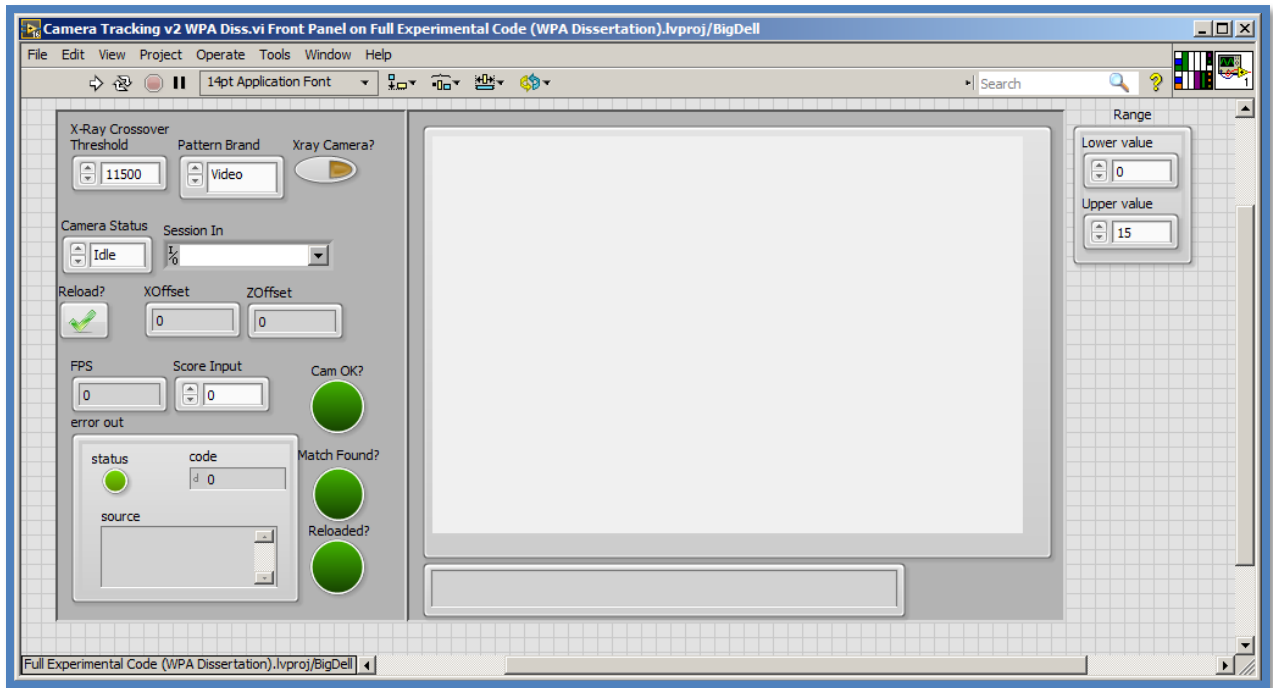


Figure 62: Camera Tracking interface.

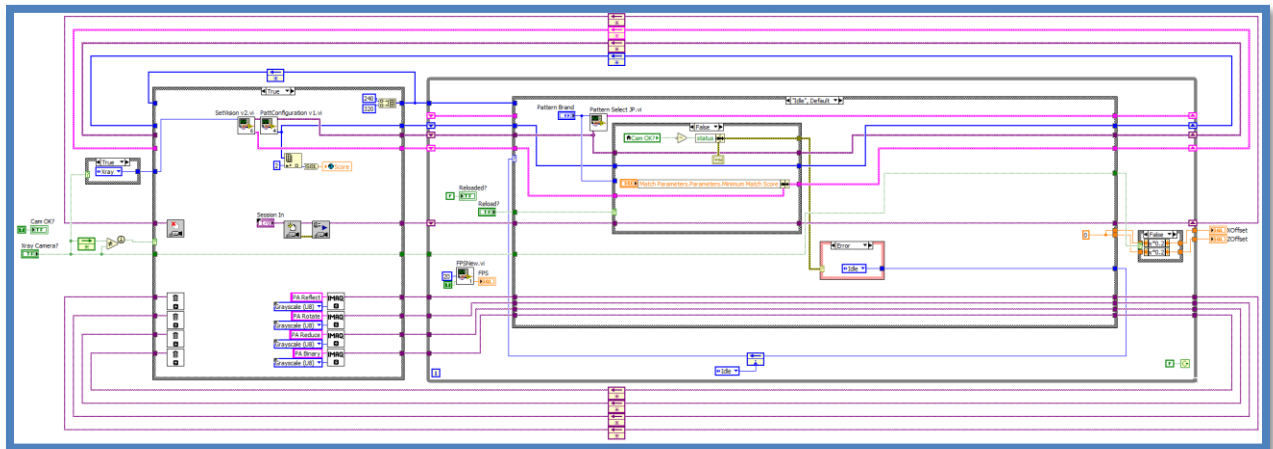


Figure 63: Camera Tracking code. Panel 1-1.

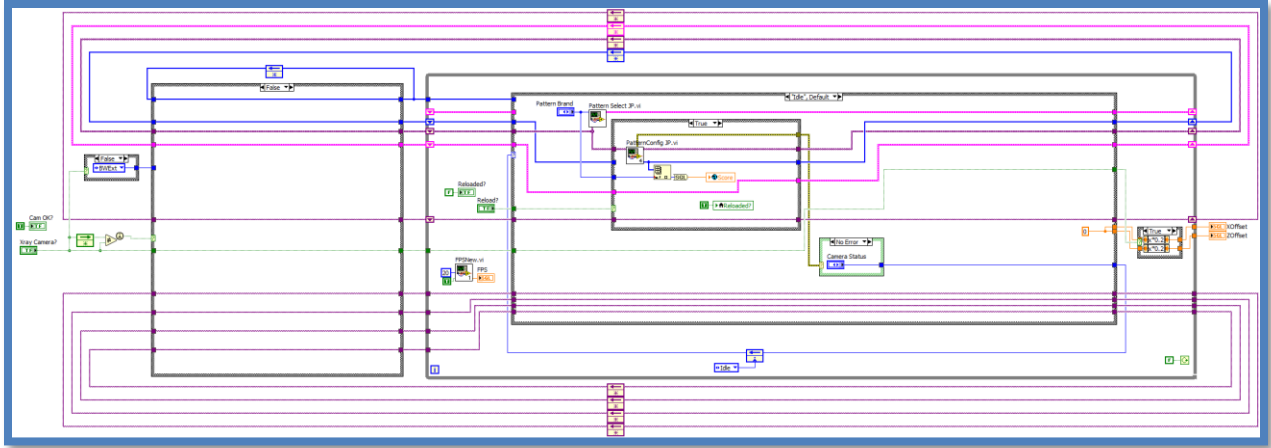


Figure 64: Camera Tracking code. Panel 1-2.

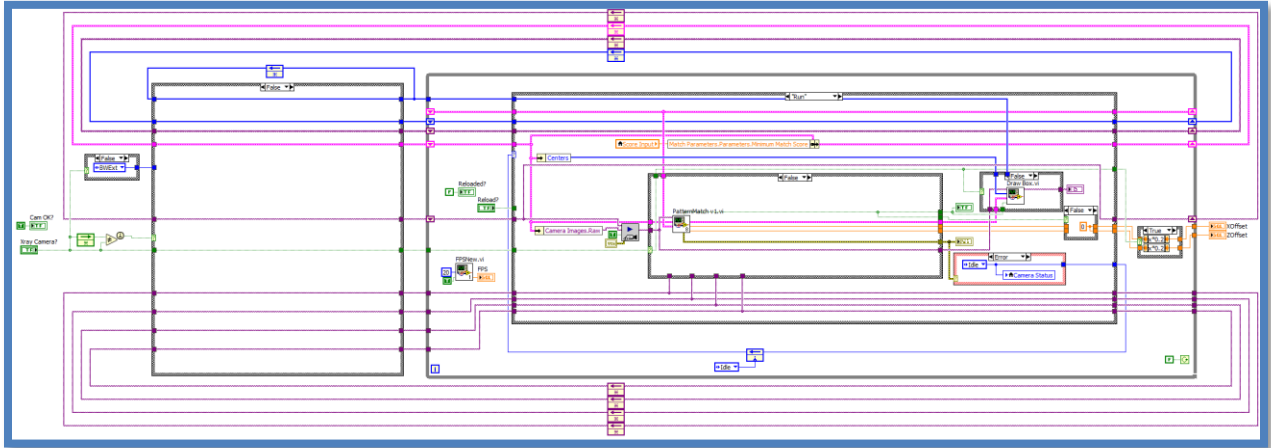


Figure 65: Camera Tracking code. Panel 2-1.

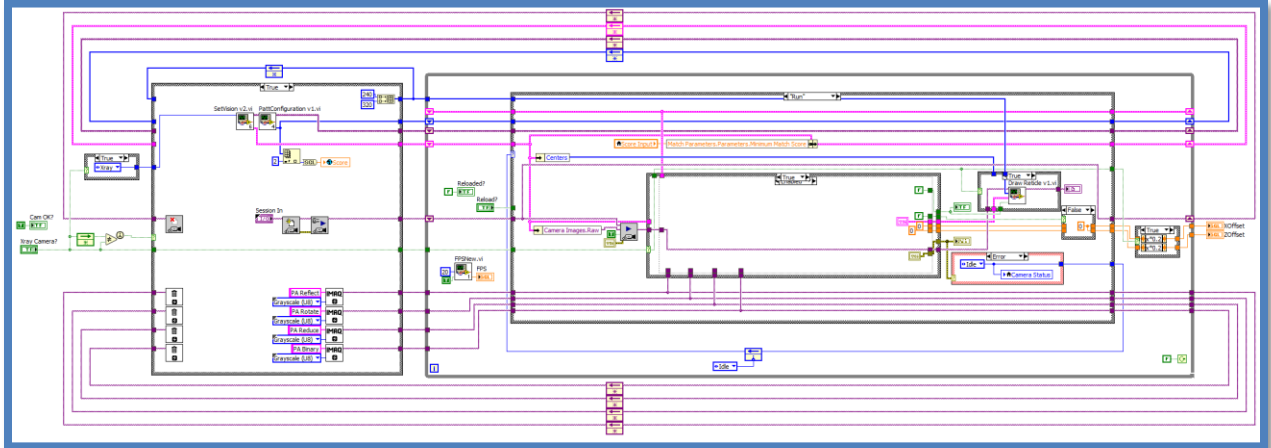


Figure 66: Camera Tracking code. Panel 2-2.

Pattern Matching

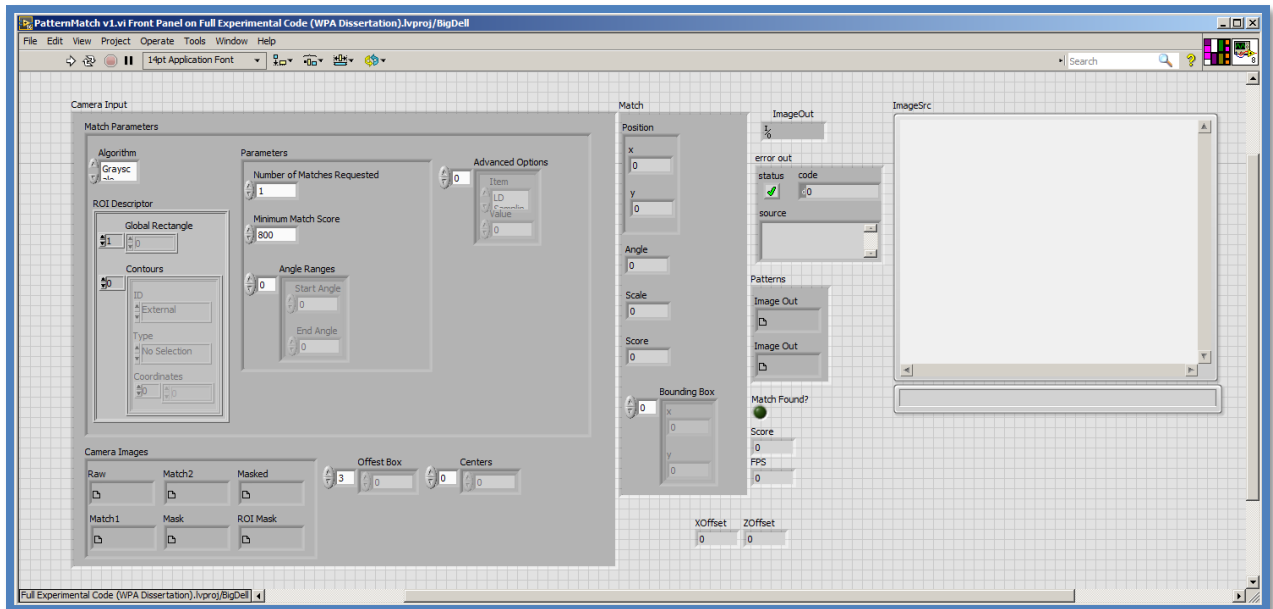


Figure 67: Pattern Matching interface.

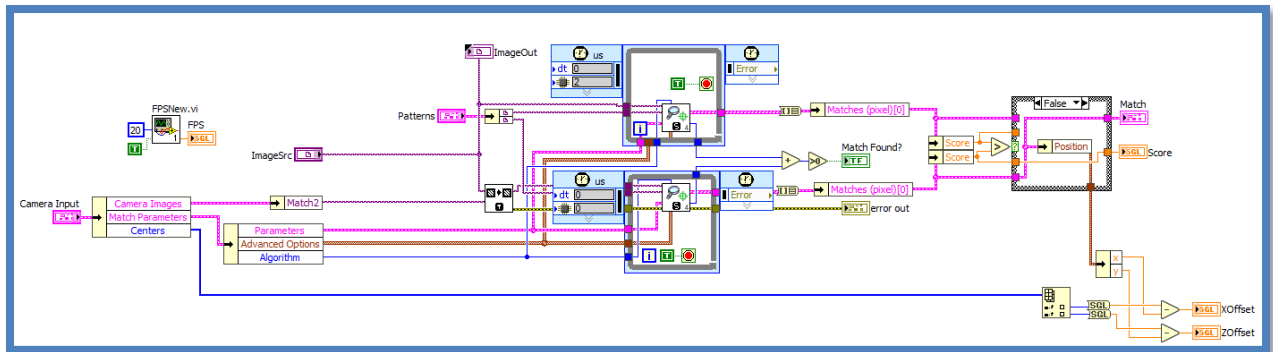


Figure 68: Pattern Matching code. Panel 1-1.

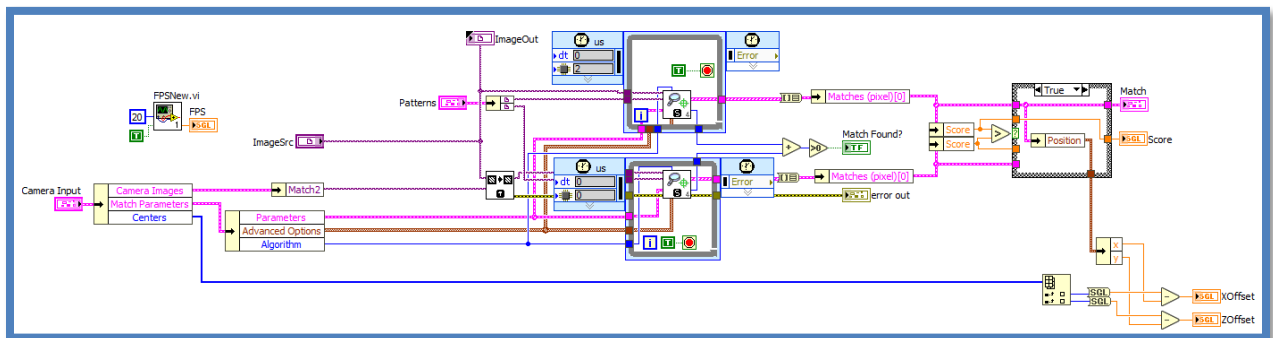


Figure 69: Pattern Matching code. Panel 1-2.

X-Ray Display

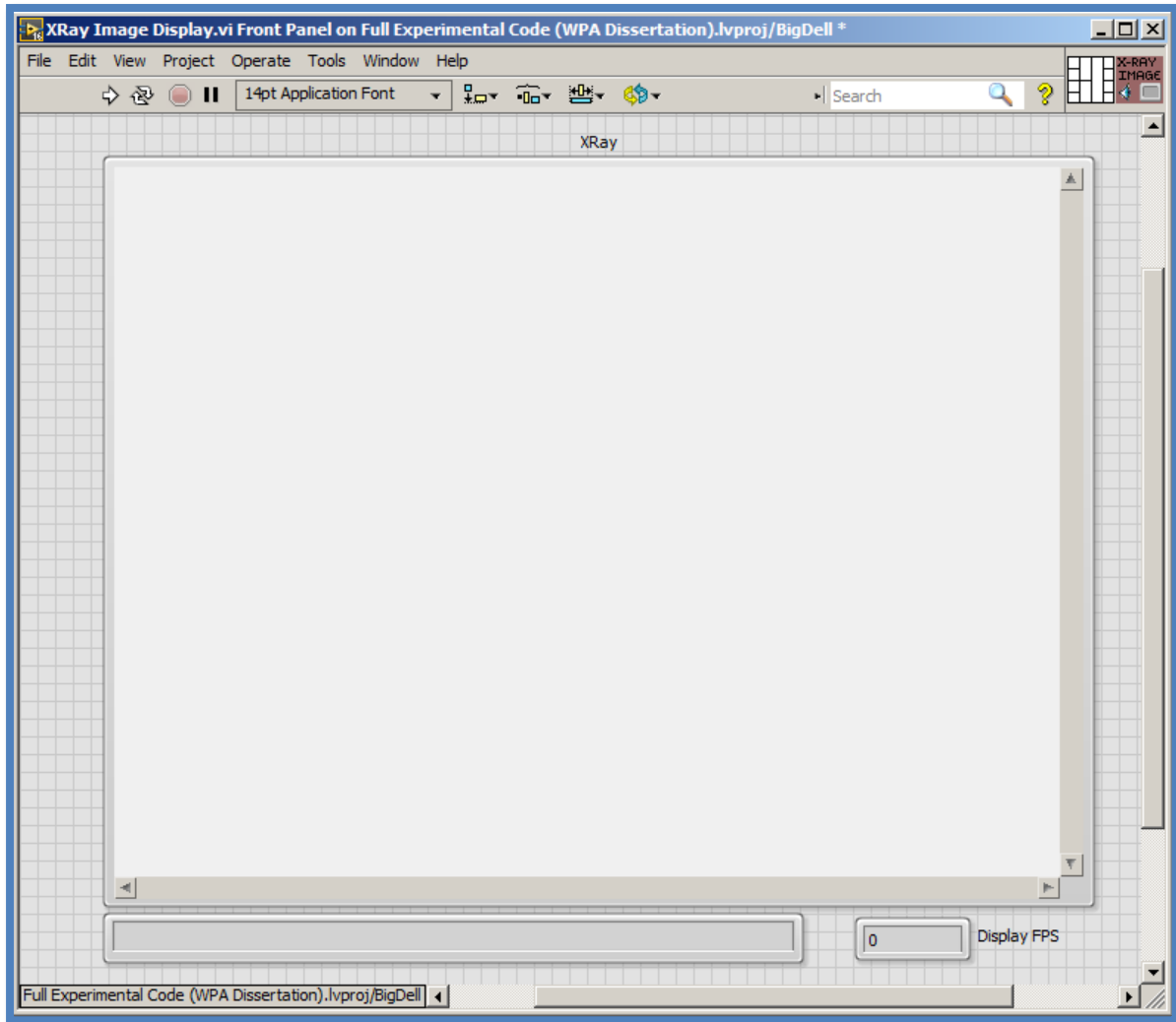


Figure 70: X-Ray Image Display interface.

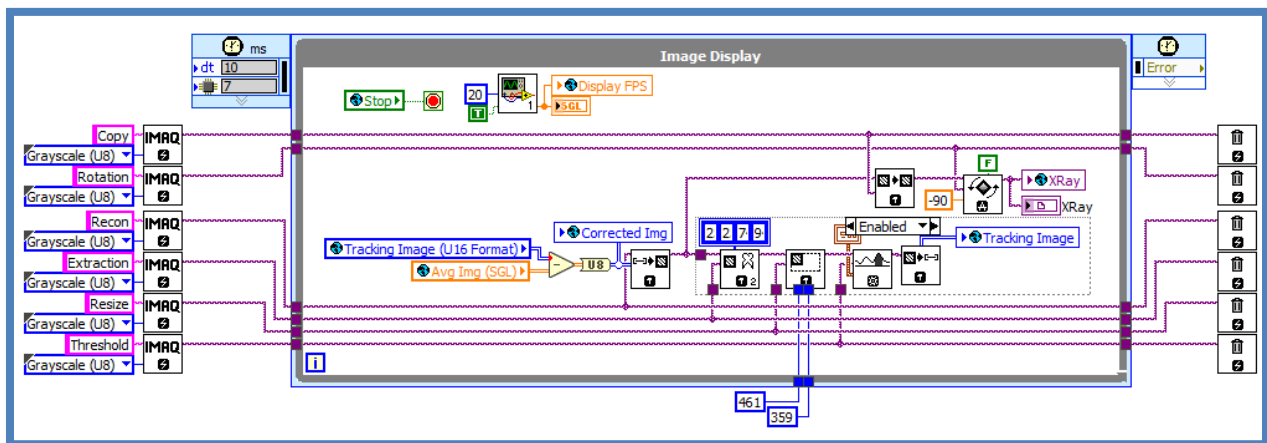


Figure 71: X-Ray Image Display code. Panel 1.

X-Ray Safety & Timer

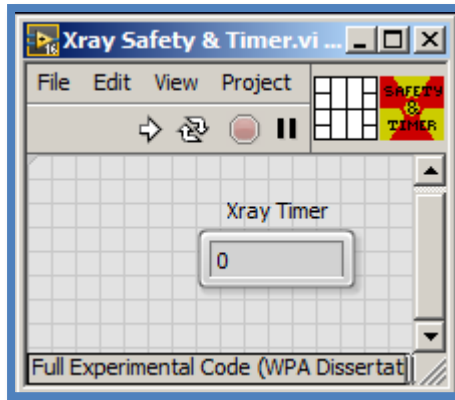


Figure 72: X-Ray Safety & Timer interface.

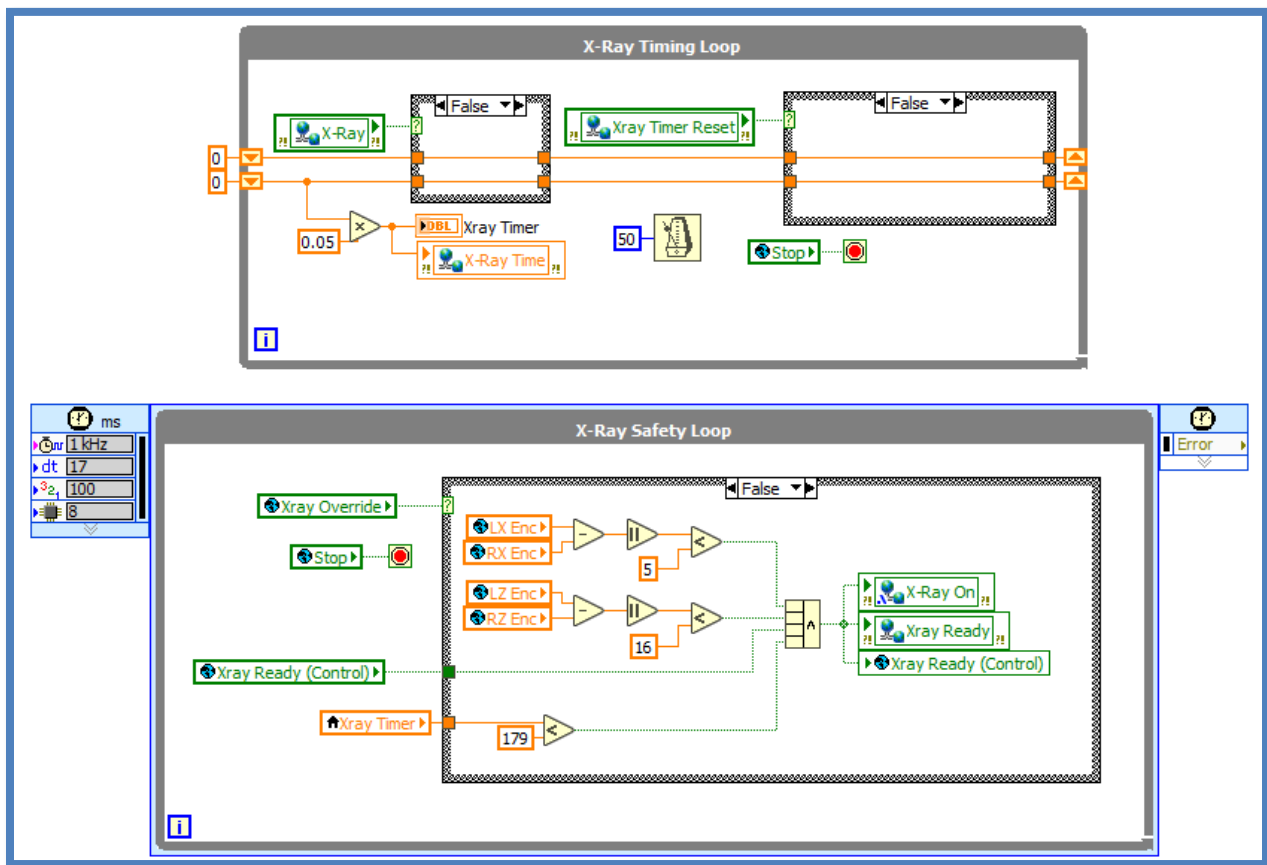


Figure 73: X-Ray Safety & Timer code. Panel 1-1.

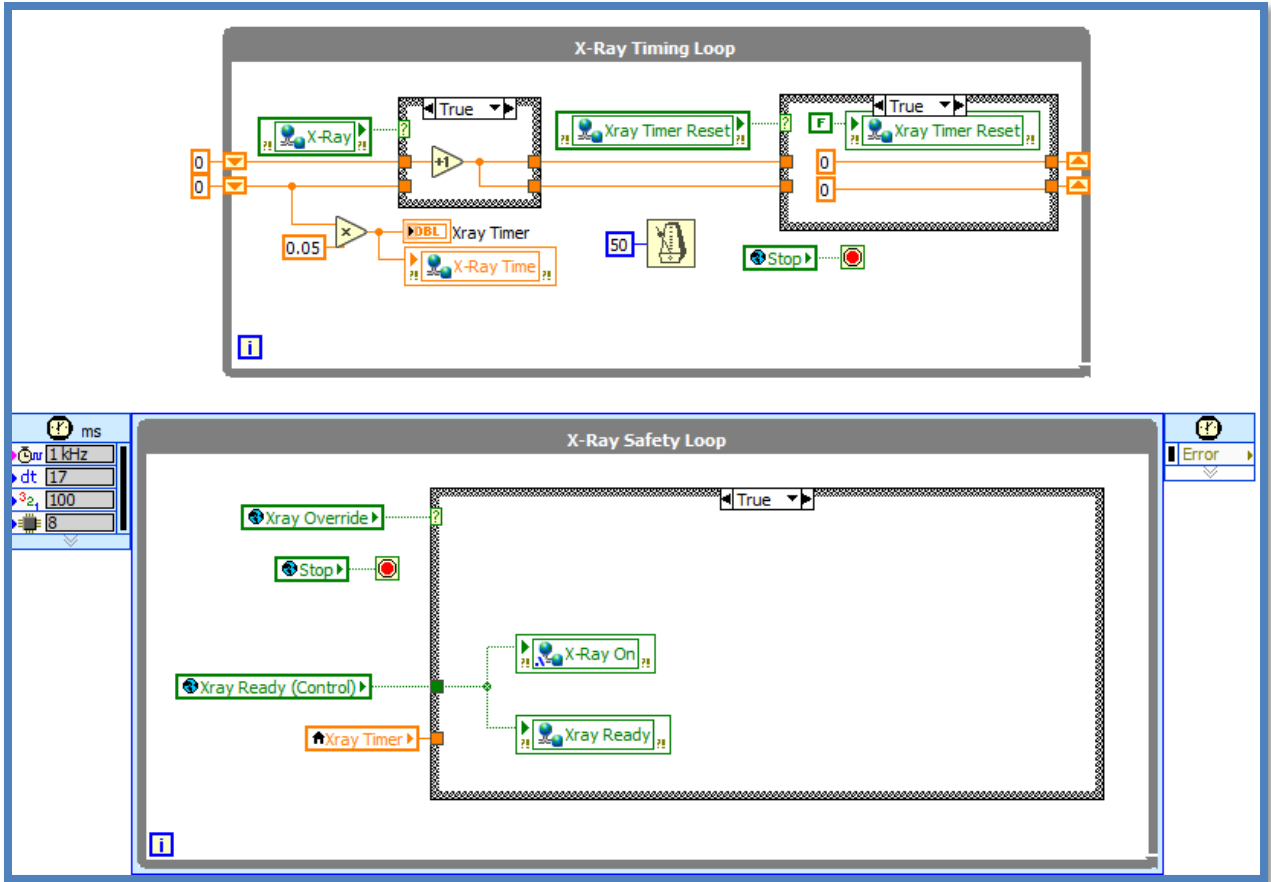


Figure 74: X-Ray Safety & Timer code. Panel 1-2.

Axis Command

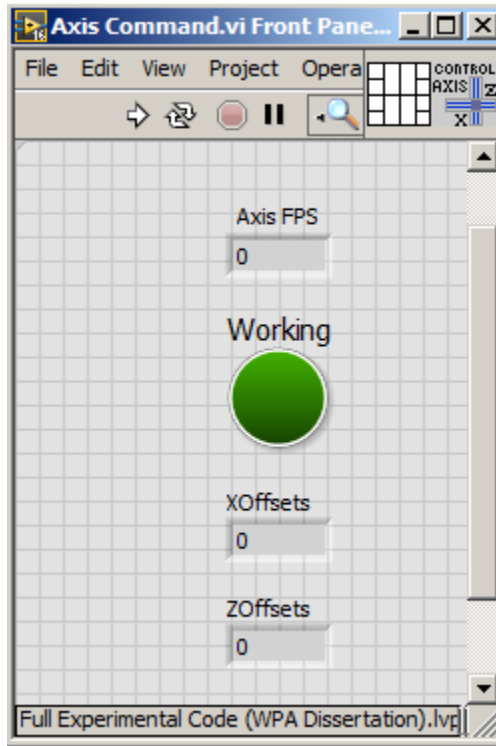


Figure 75: Axis Command interface.

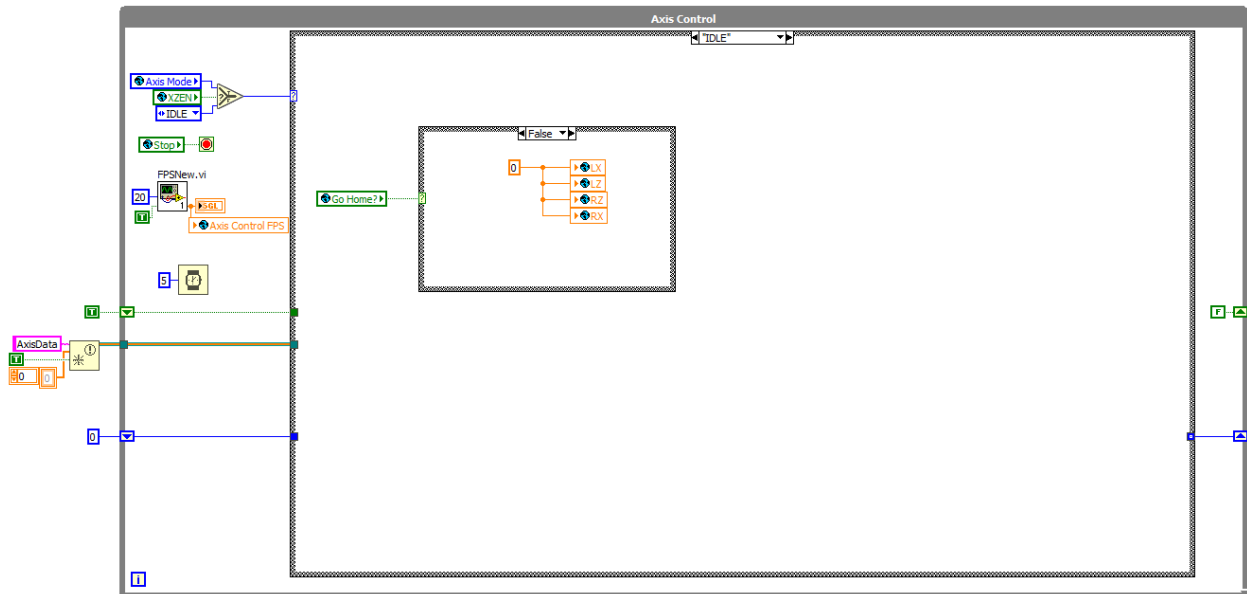


Figure 76: Axis Command code. Panel 1-1.

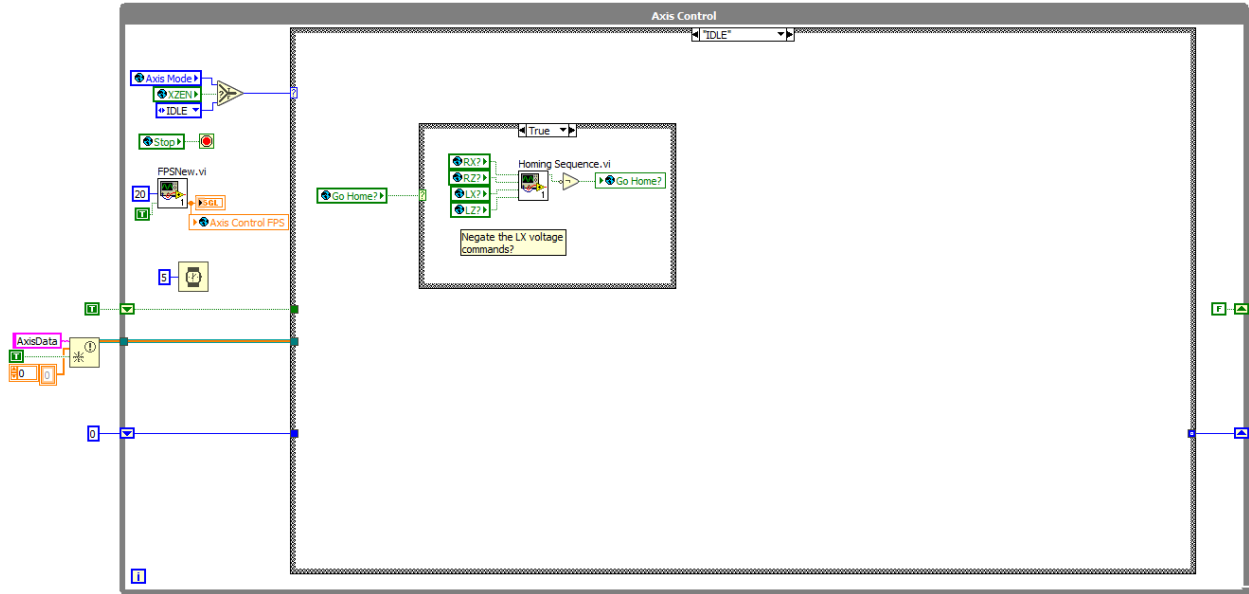


Figure 77: Axis Command code. Panel 1-2.

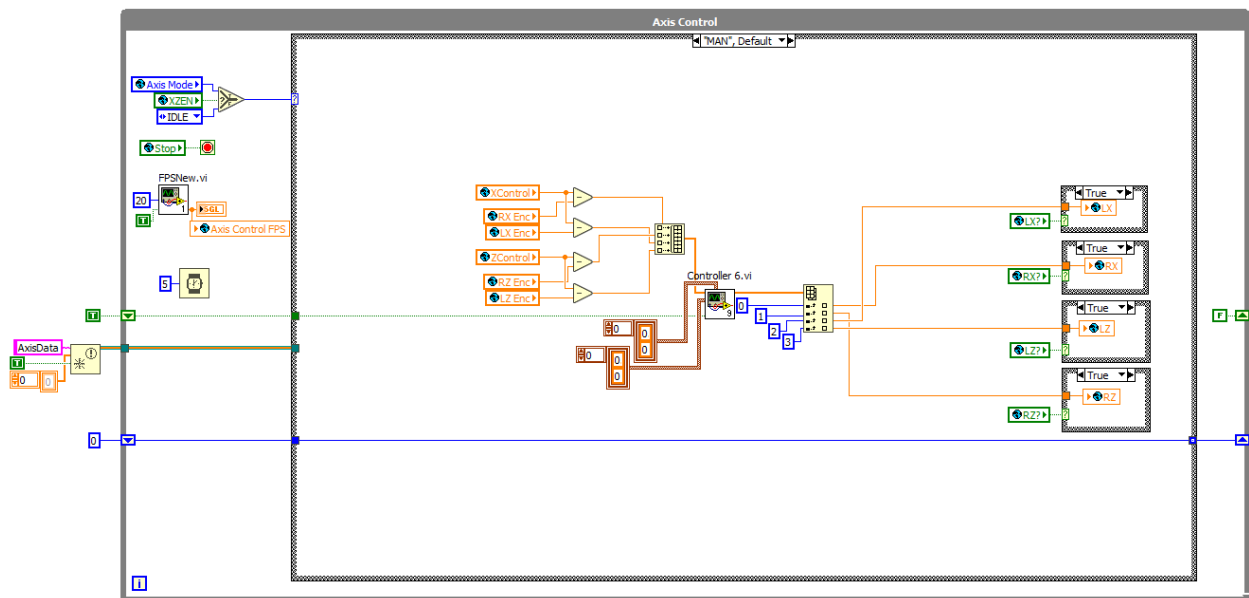


Figure 78: Axis Command code. Panel 2-1.

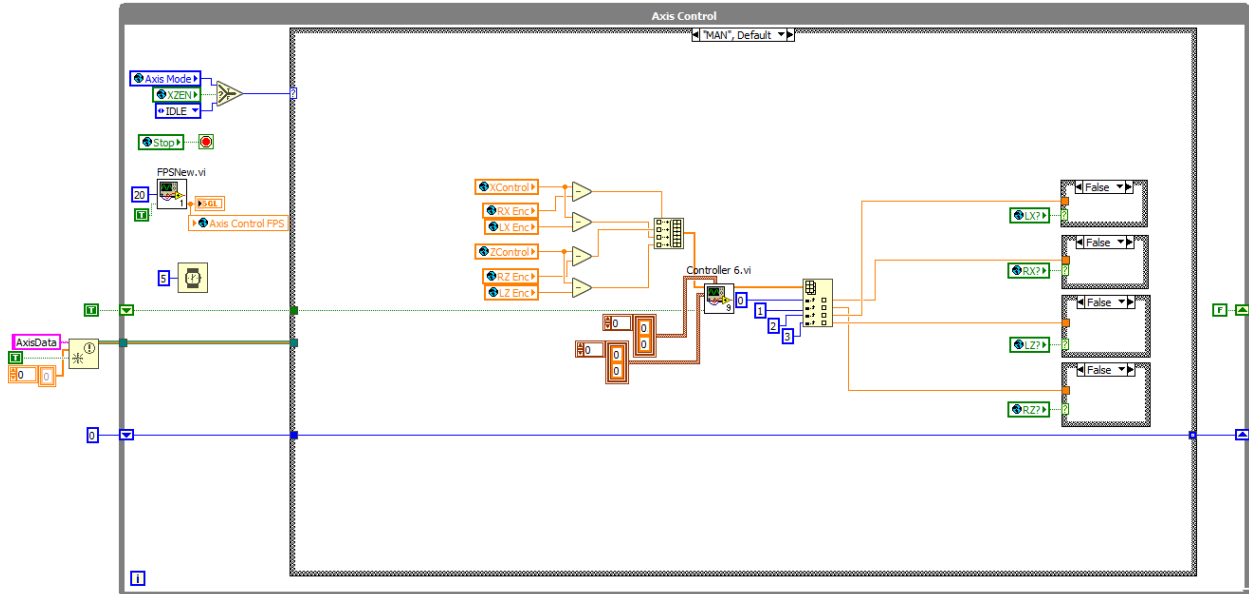


Figure 79: Axis Command code. Panel 2-2.

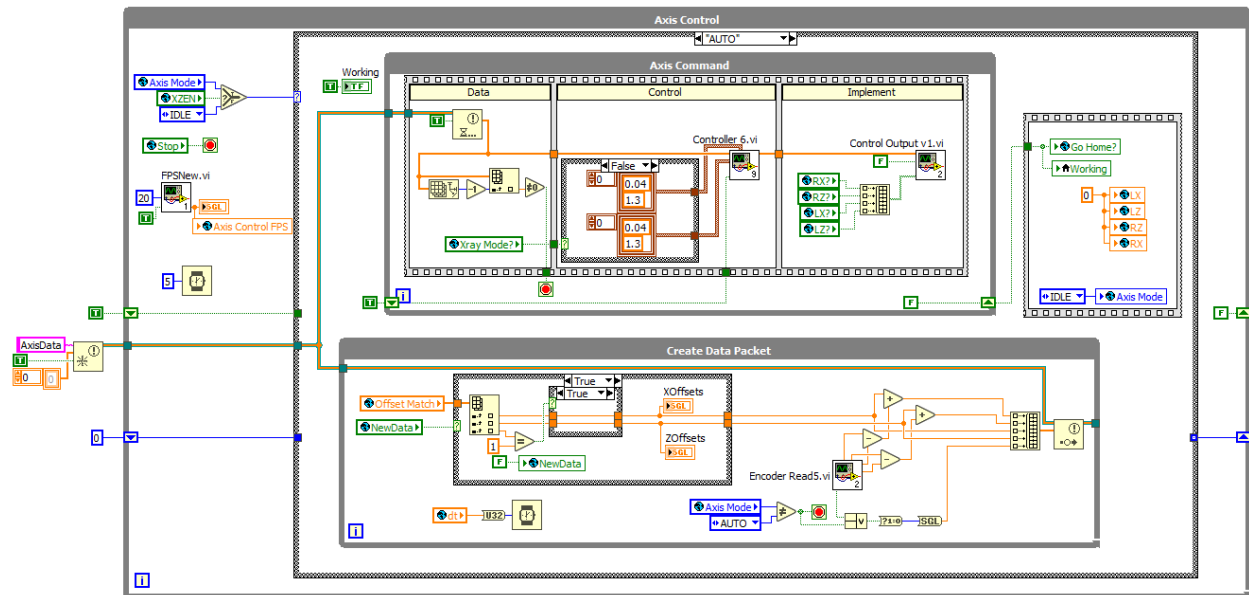


Figure 80: Axis Command code. Panel 3-1.

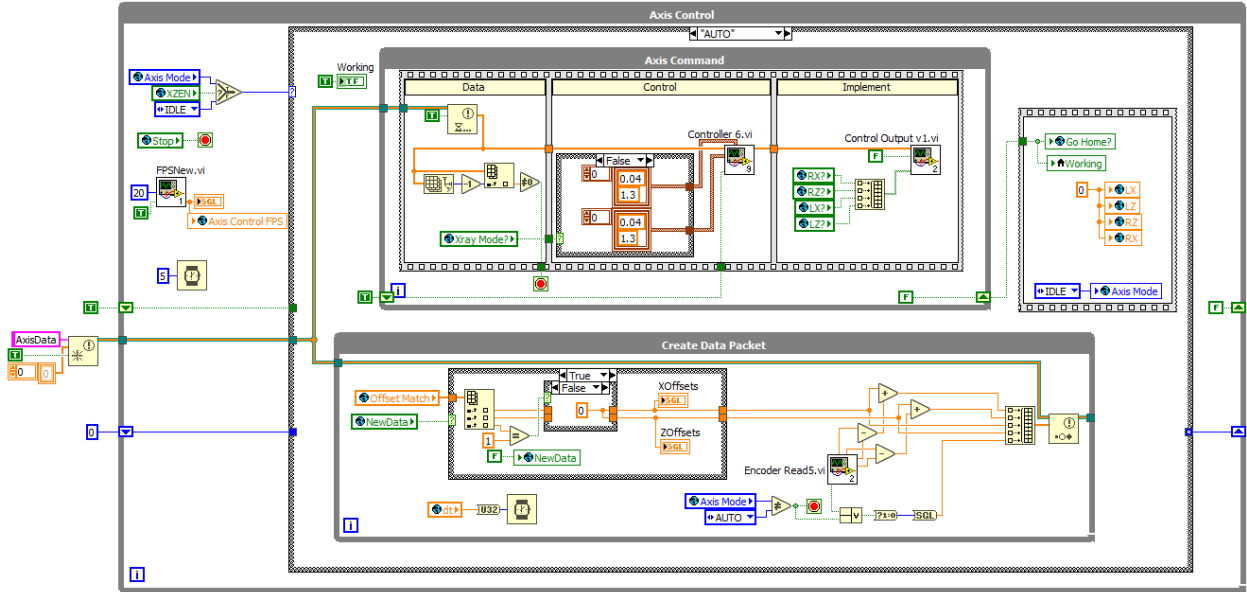


Figure 81: Axis Command code. Panel 3-2.

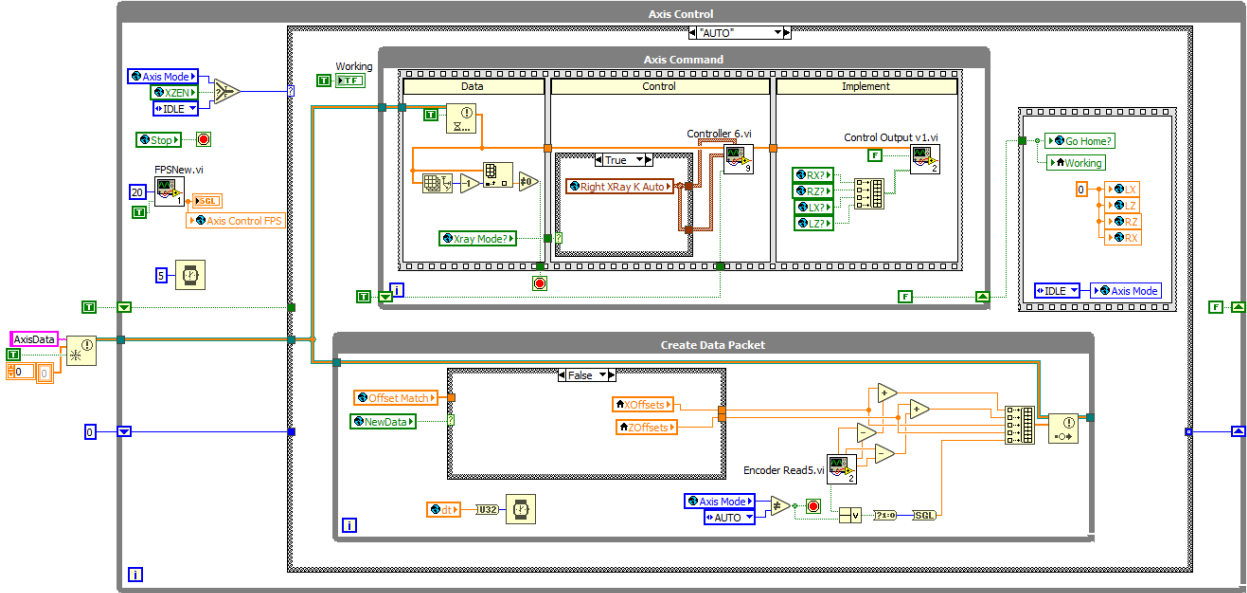


Figure 82: Axis Command code. Panel 3-3.

Homing Sequence

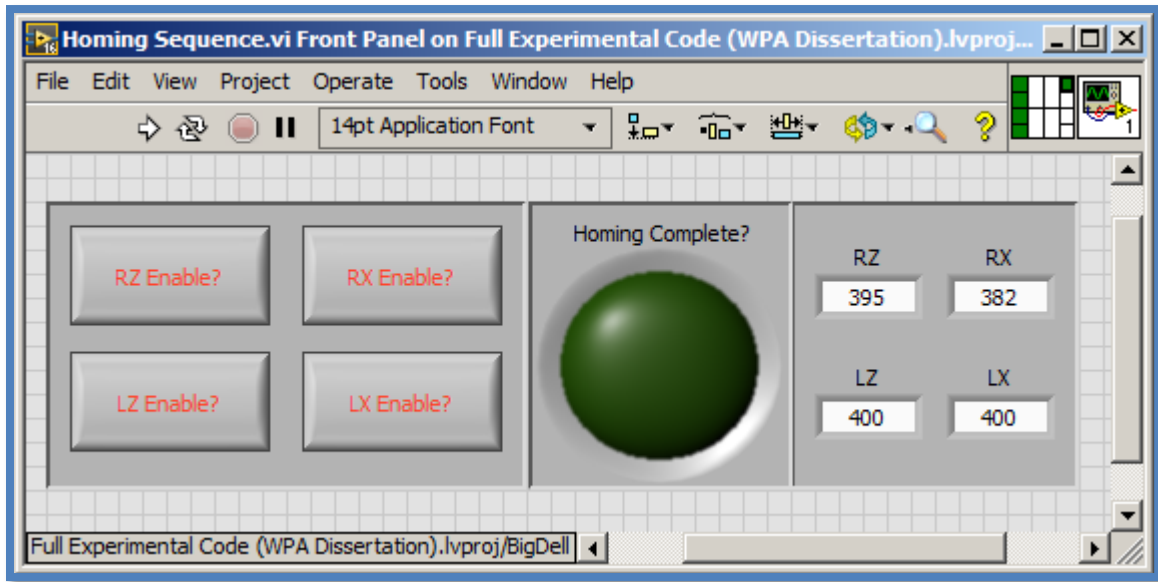


Figure 83: Homing Sequence interface.

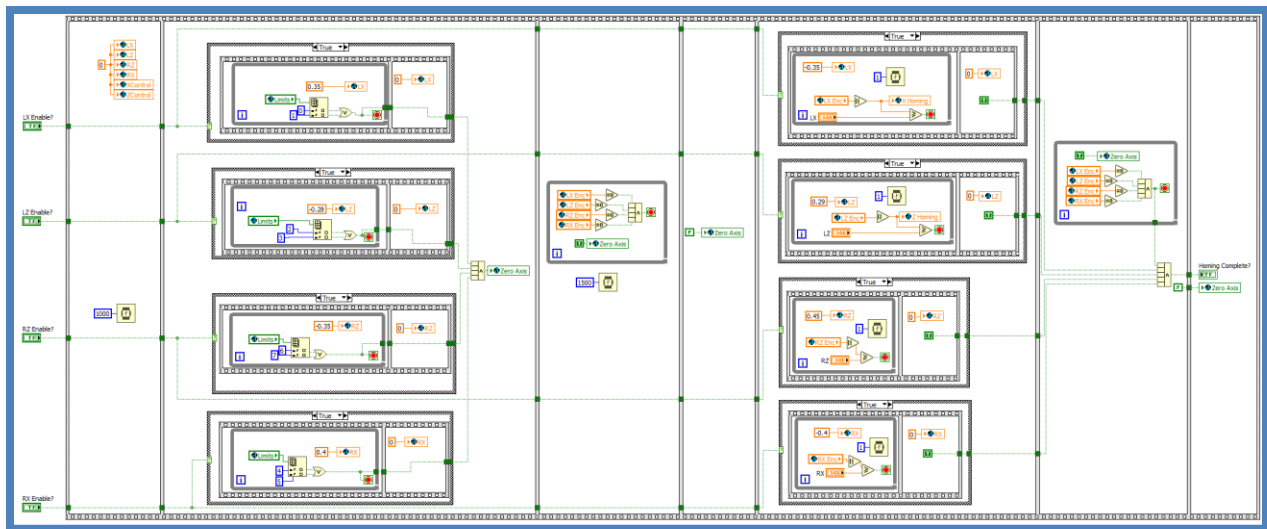


Figure 84: Homing Sequence code. Panel 1.

MIWD Command

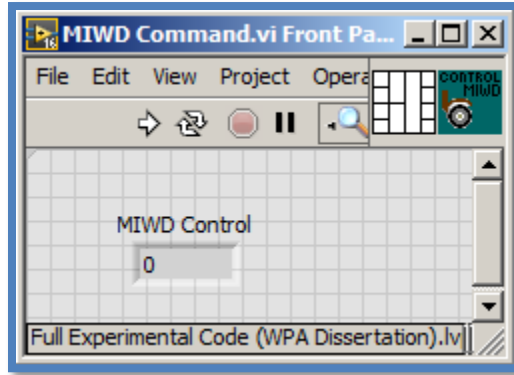


Figure 85: MIWD Command interface.

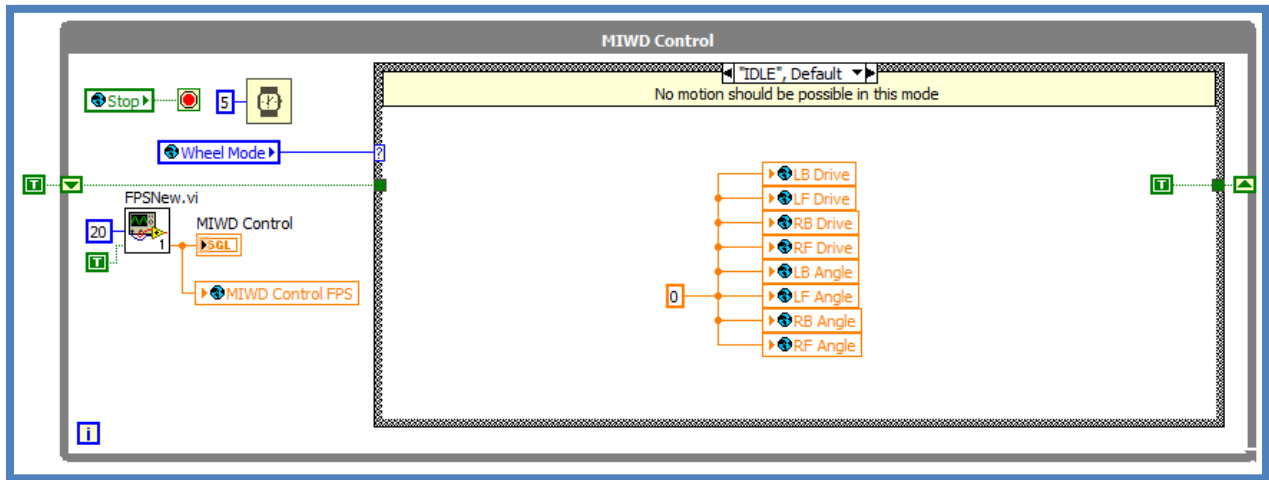


Figure 86: MIWD Command code. Panel 1.

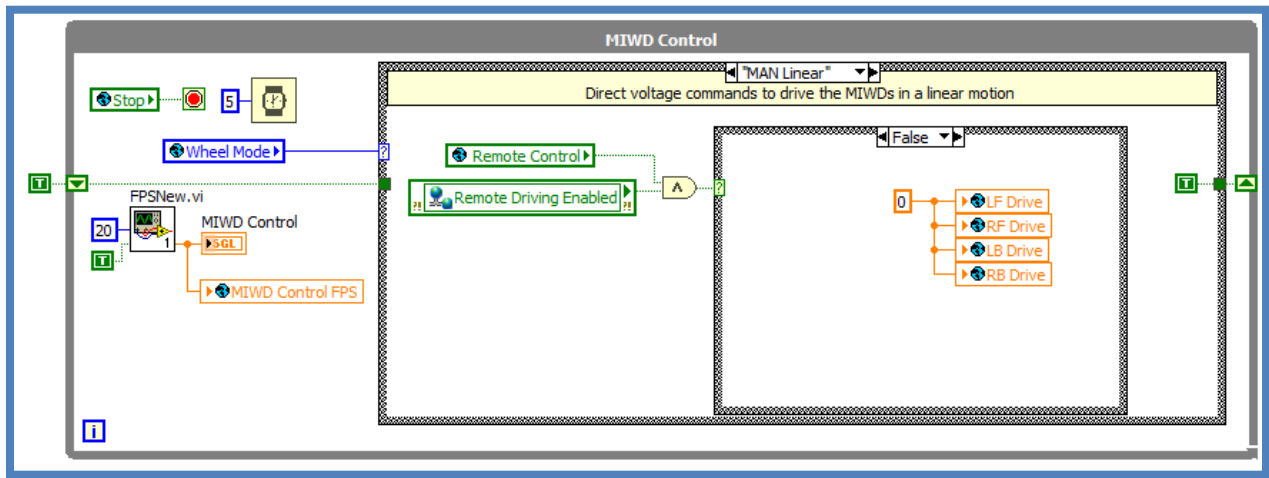


Figure 87: MIWD Command code. Panel 2-1.

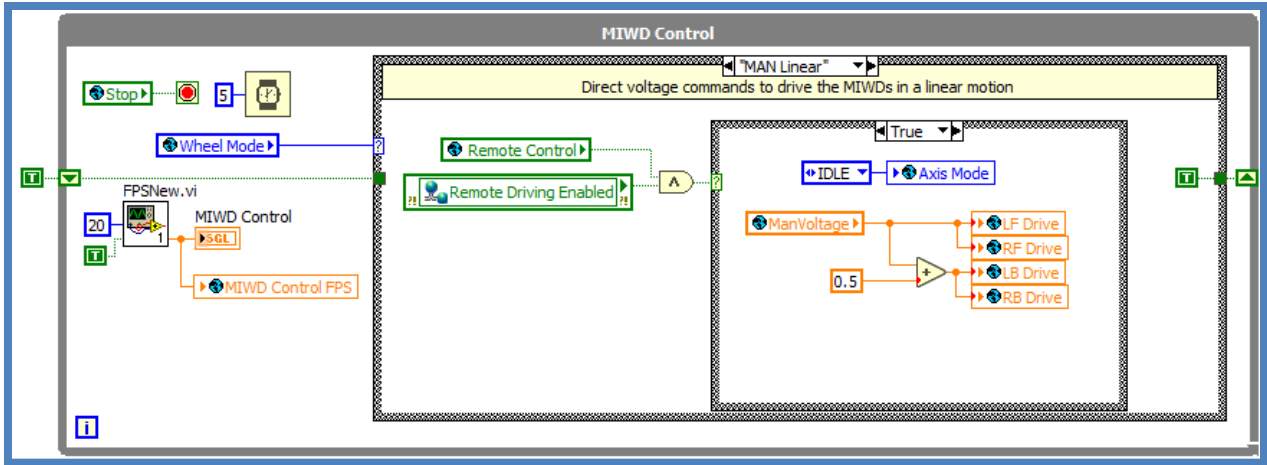


Figure 88: MIWD Command code. Panel 2-2.

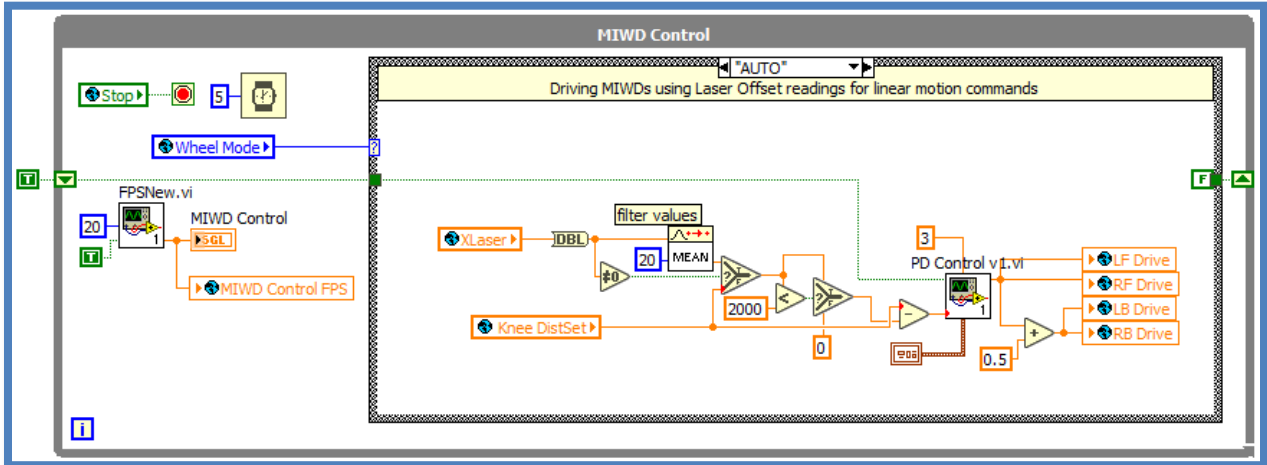


Figure 89: MIWD Command code. Panel 3.

Data Communication

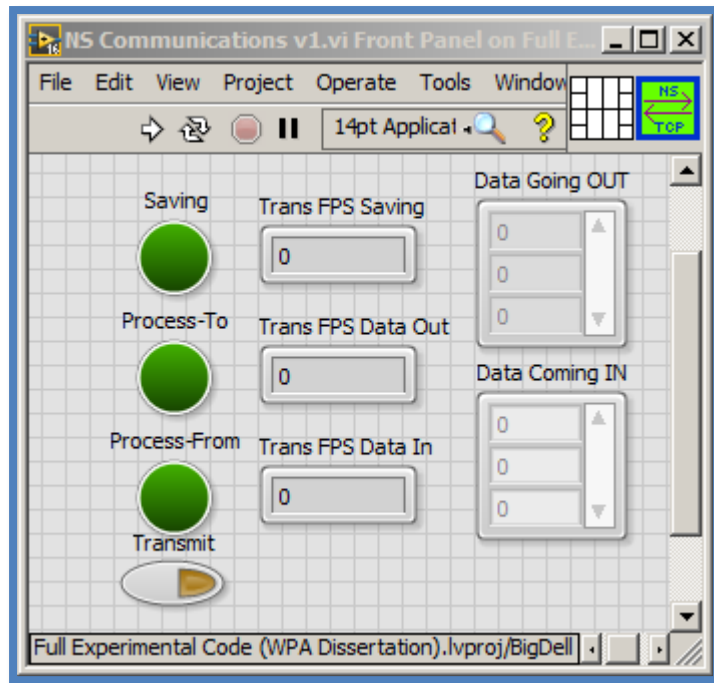


Figure 90: Data Communication interface.

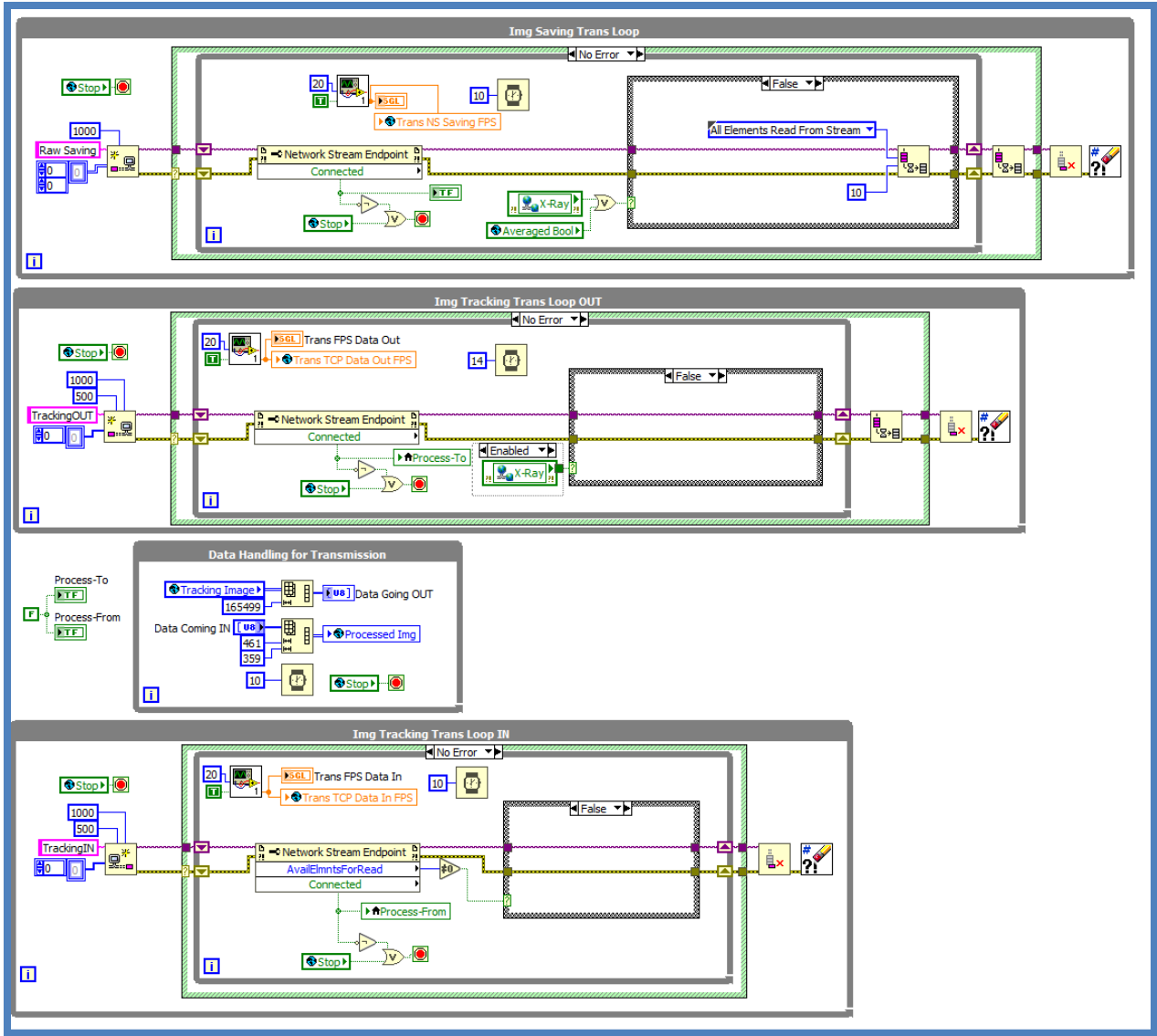


Figure 91: Data Communication code. Panel 1.

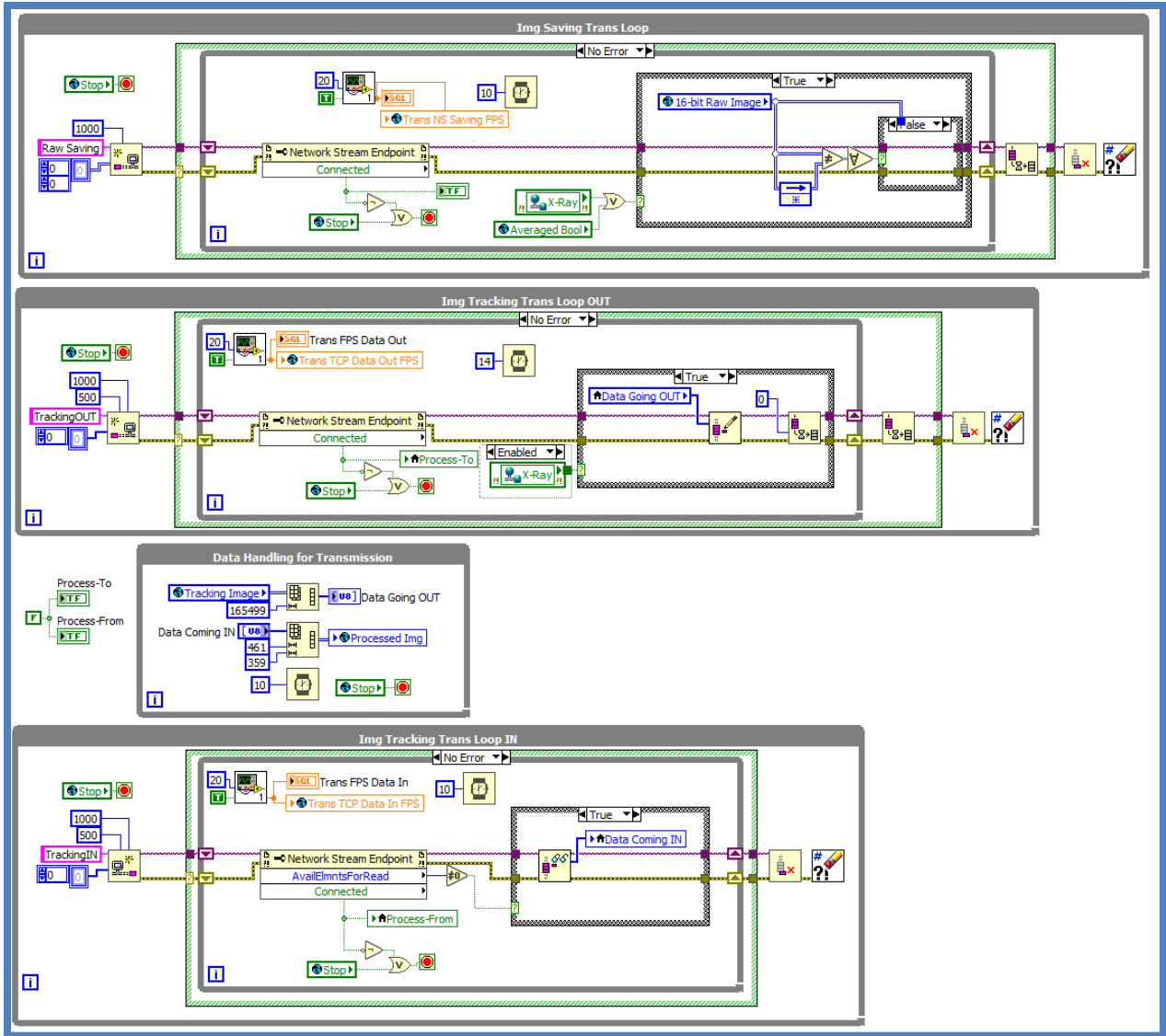


Figure 92: Data Communication code. Panel 2-1

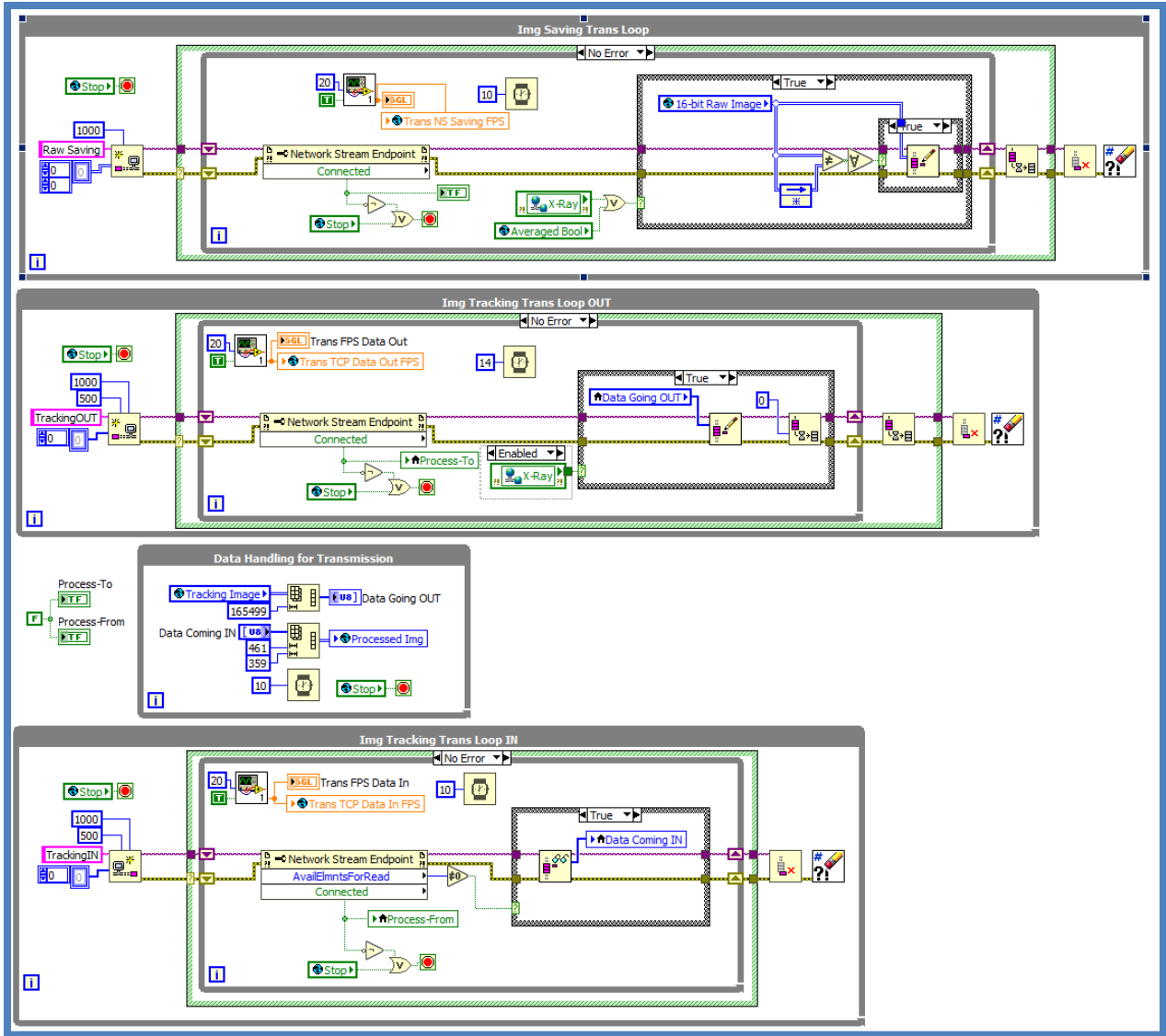


Figure 93: Data Communication code. Panel 2-2.

Laser Processing

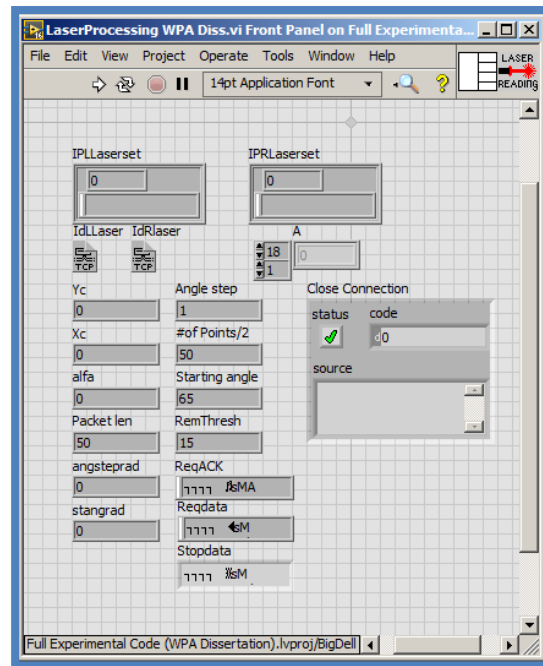


Figure 94: Laser Processing interface.

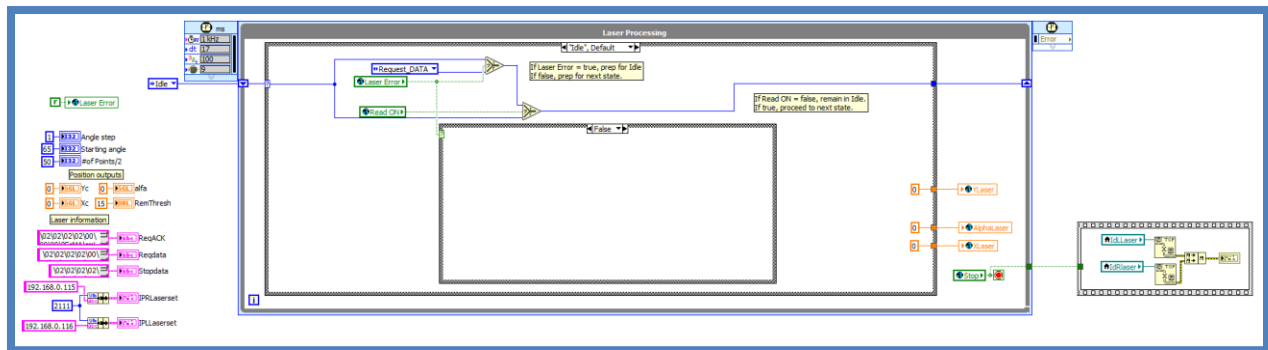


Figure 95: Laser Processing code. Panel 1-1.

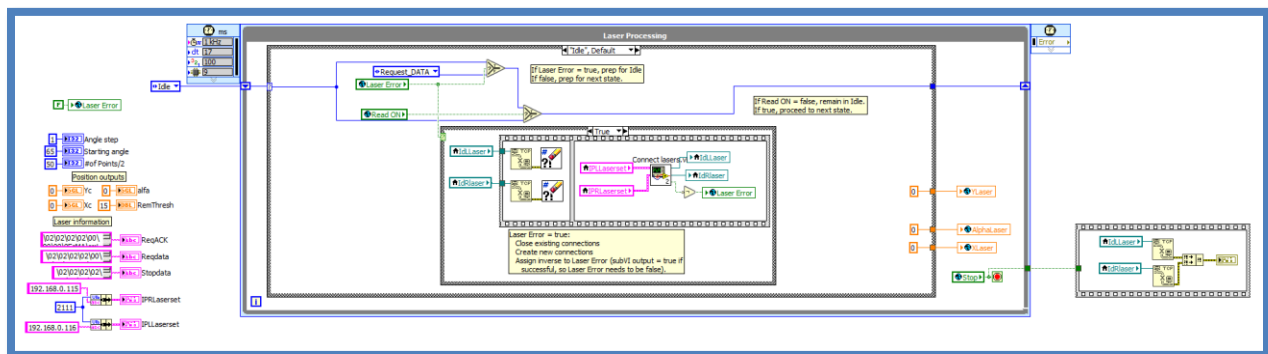


Figure 96: Laser Processing code. Panel 1-2.

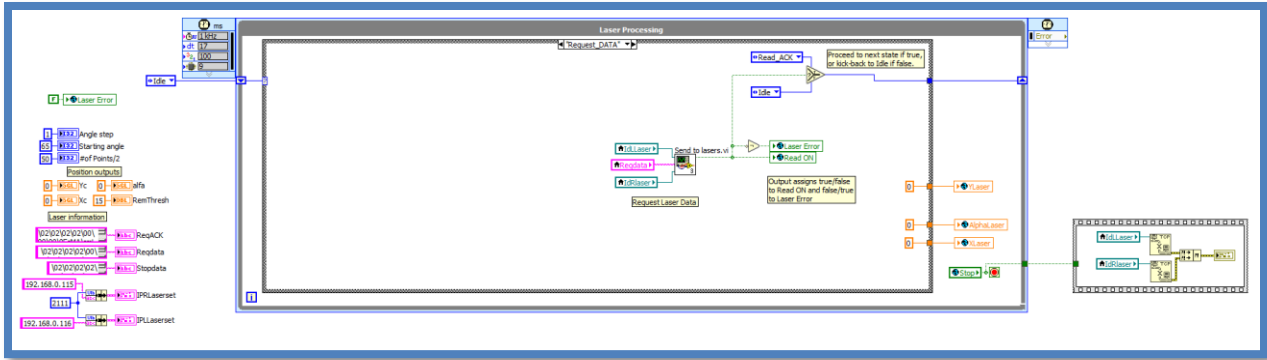


Figure 97: Laser Processing code. Panel 2.

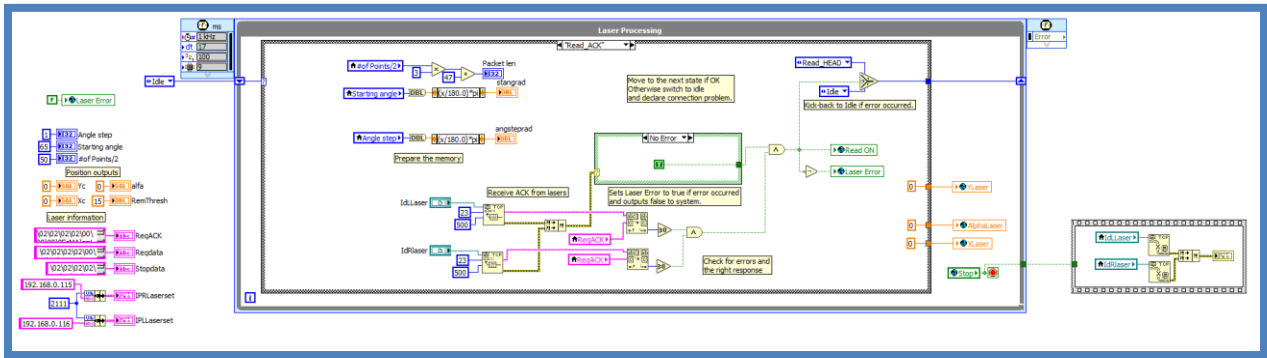


Figure 98: Laser Processing code. Panel 3-1.

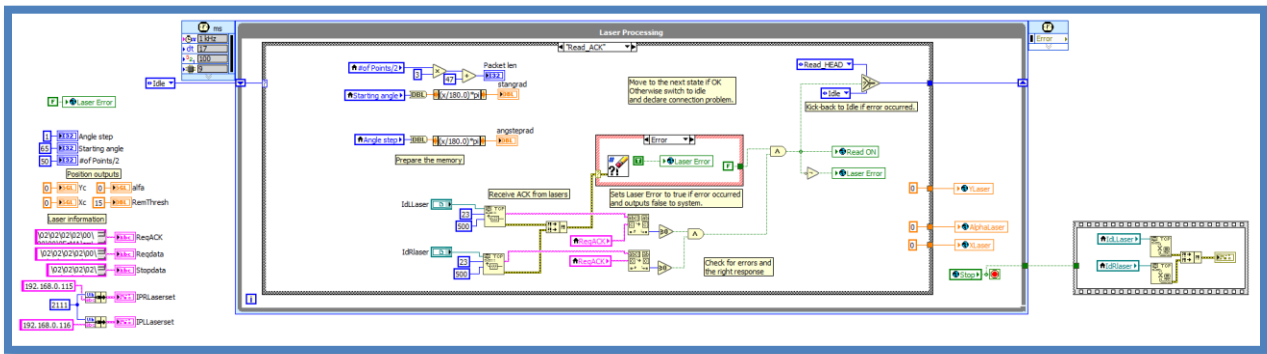


Figure 99: Laser Processing code. Panel 3-2.

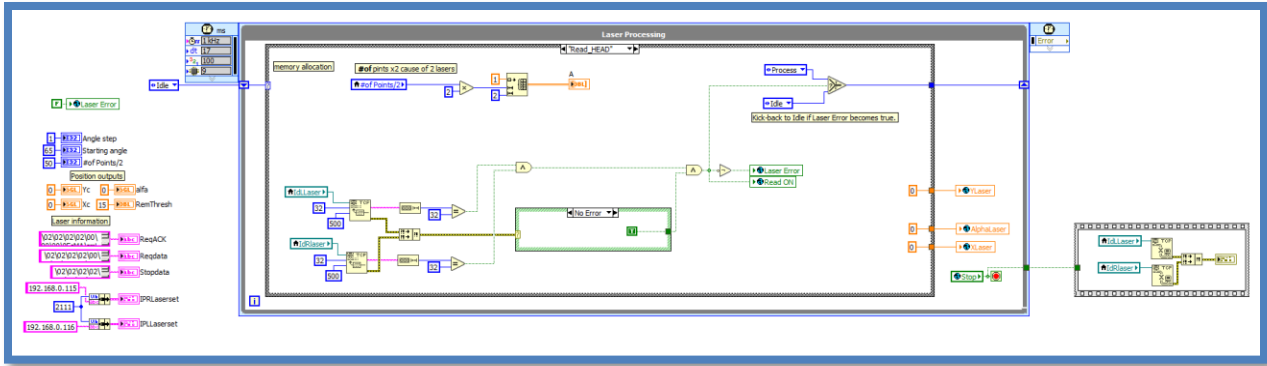


Figure 100: Laser Processing code. Panel 4-1.

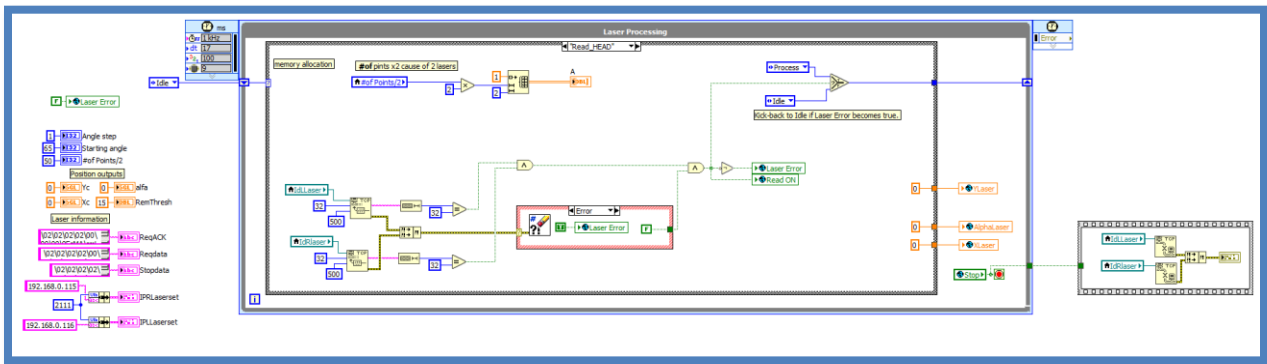


Figure 101: Laser Processing code. Panel 4-2.

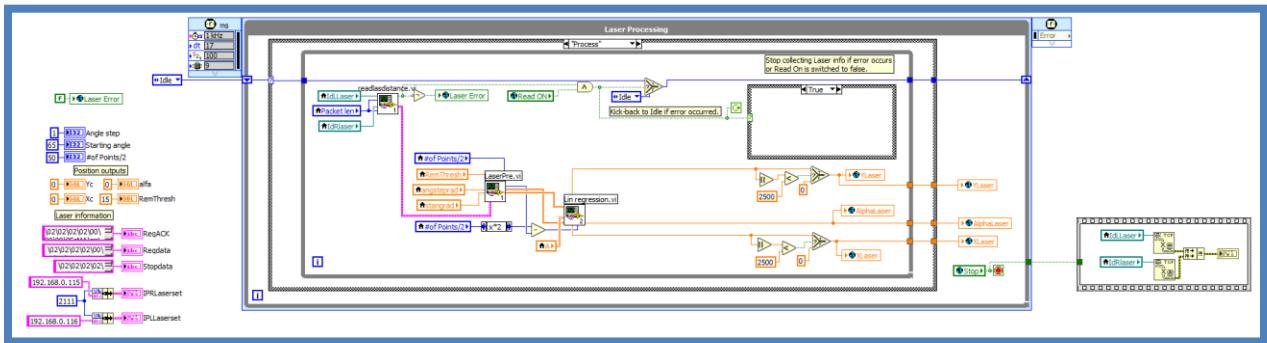


Figure 102: Laser Processing code. Panel 5-1.

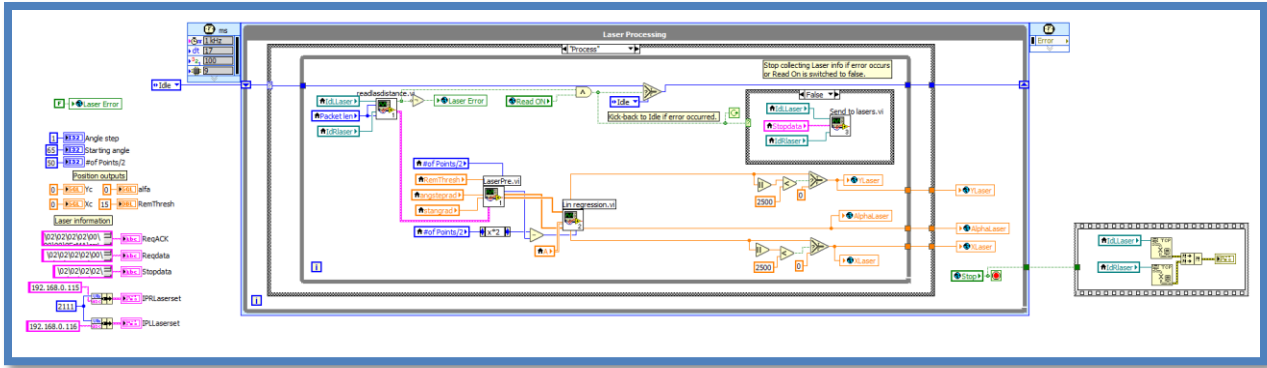


Figure 103: Laser Processing code. Panel 5-2.

Flat Panel

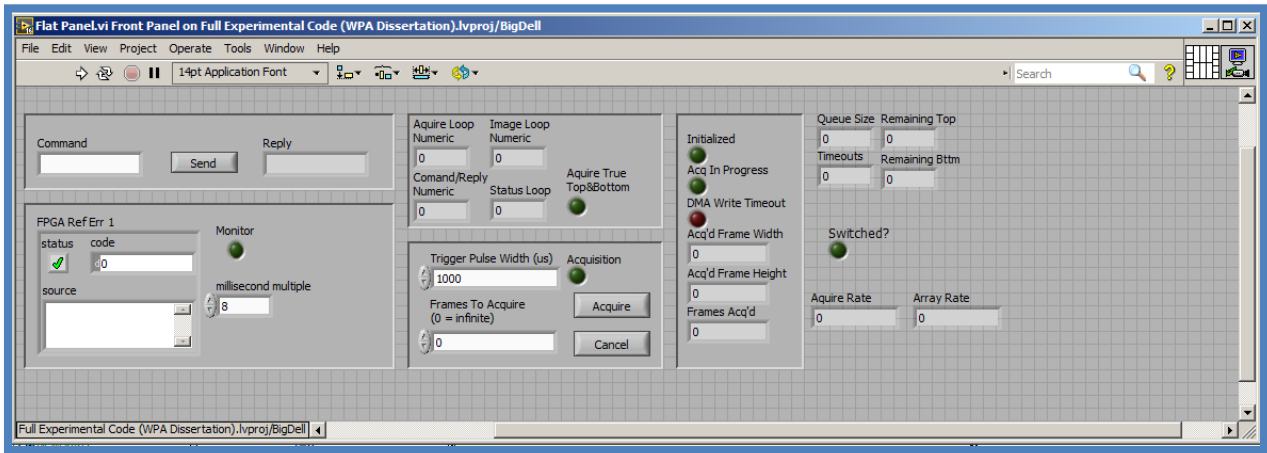


Figure 104: Flat Panel interface.

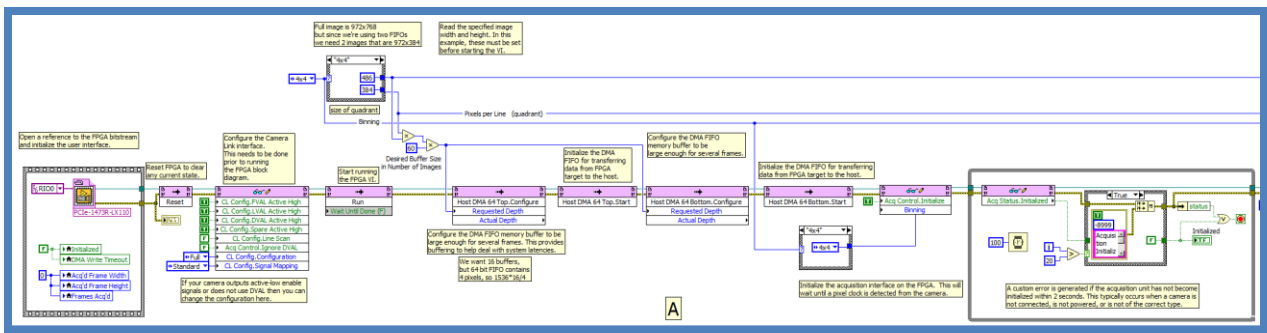


Figure 105: Flat Panel code. Panel 1A.

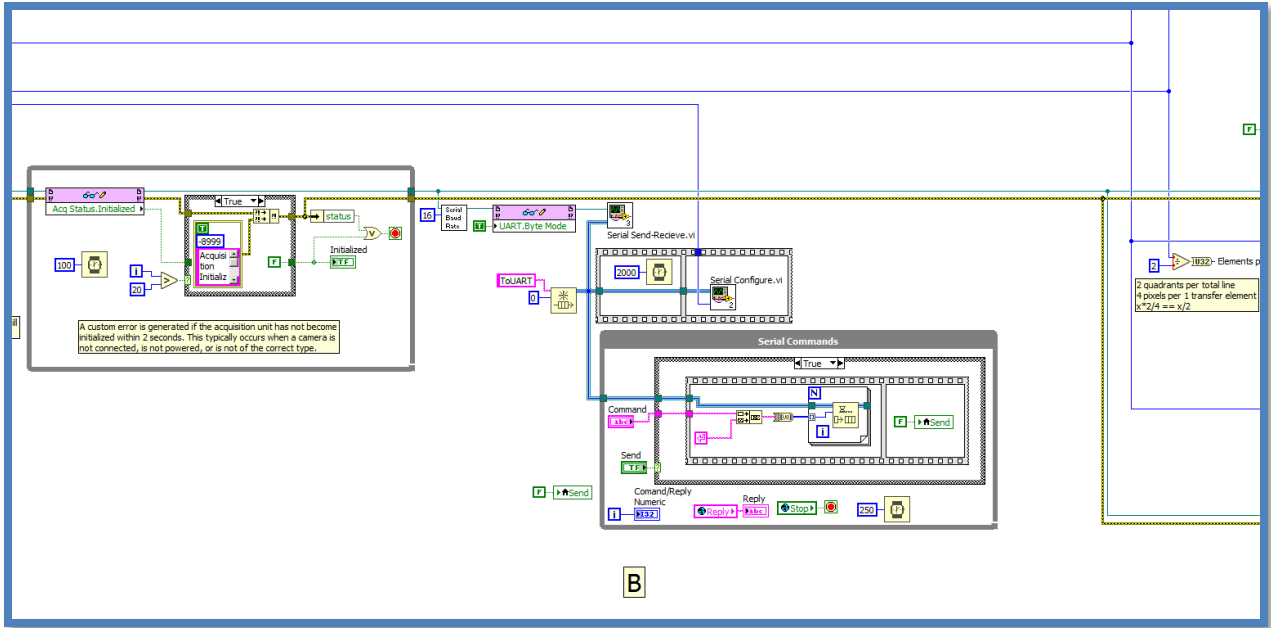


Figure 106: Flat Panel code. Panel 1B-1.

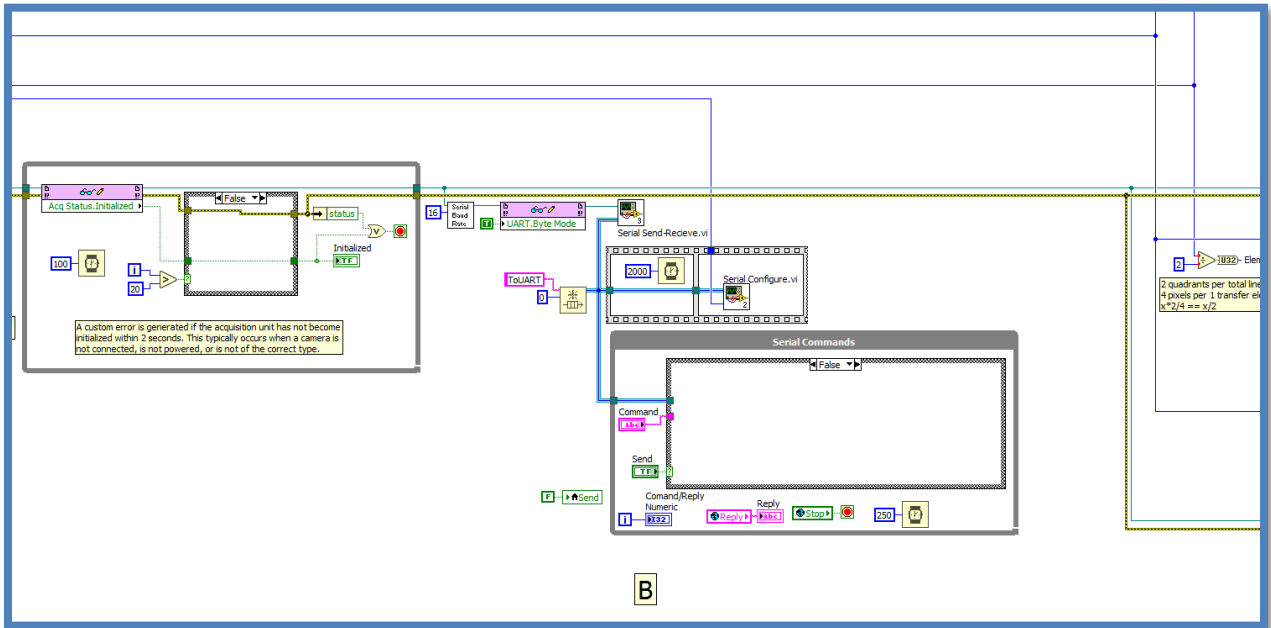


Figure 107: Flat Panel code. Panel 1B-2.

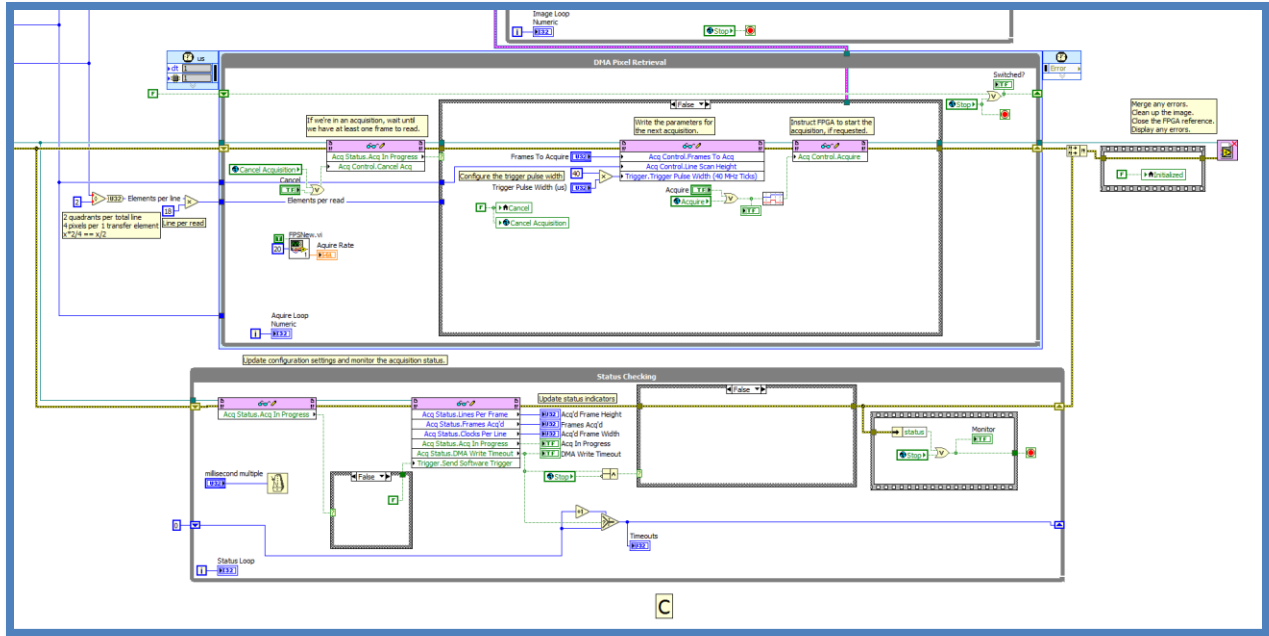


Figure 108: Flat Panel code. Panel 1C-1.

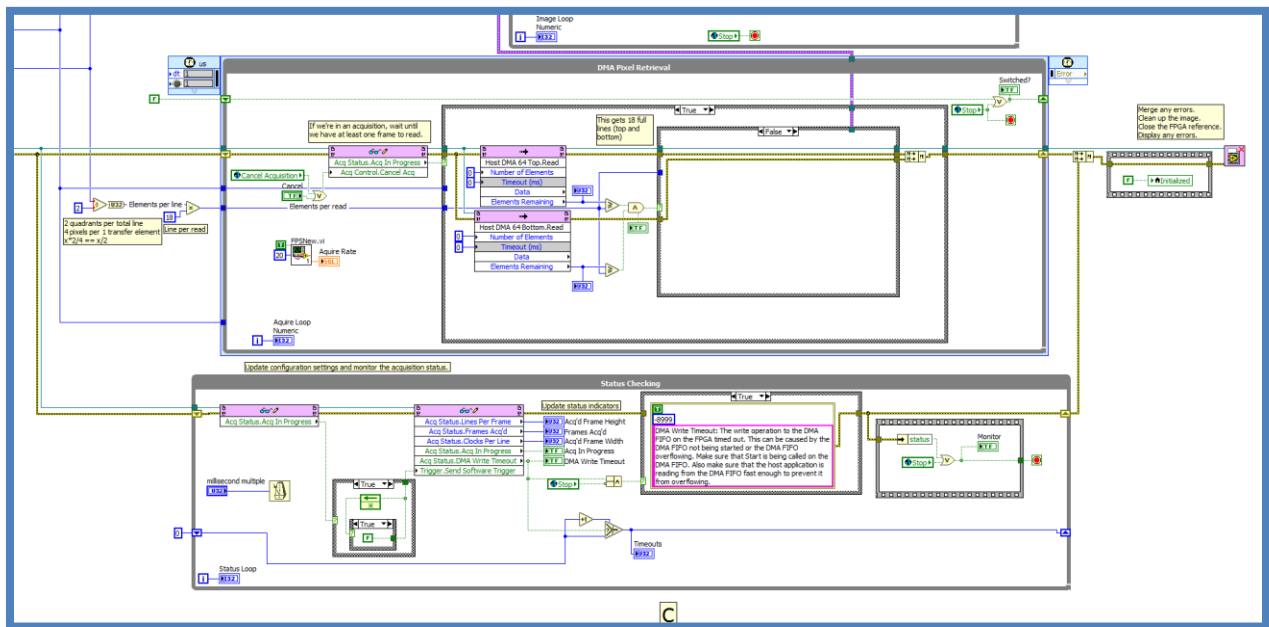


Figure 109: Flat Panel code. Panel 1C-2.

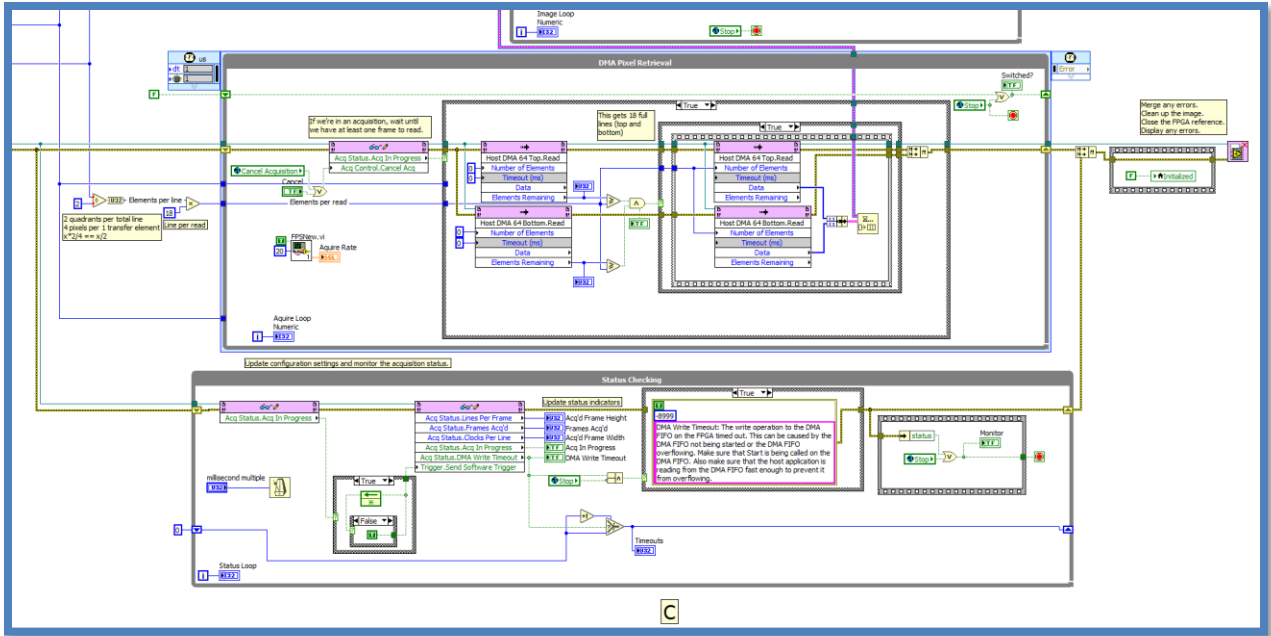


Figure 110: Flat Panel code. Panel 1C-3.

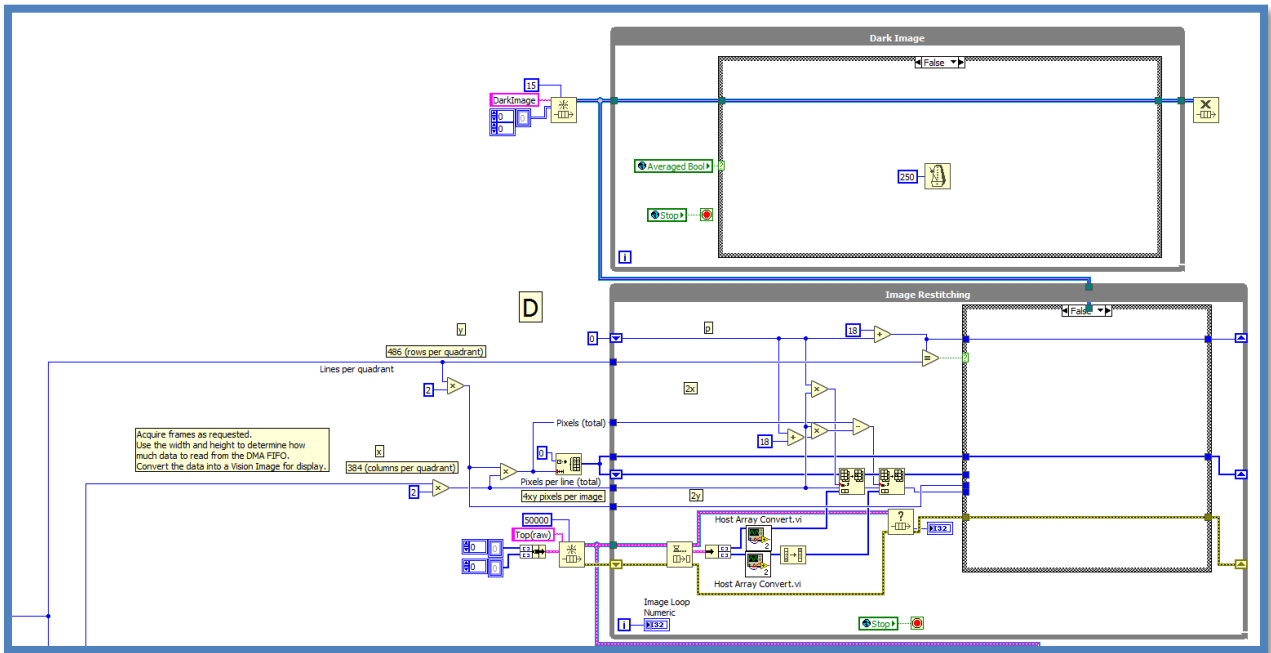


Figure 111: Flat Panel code. Panel 1D-1.

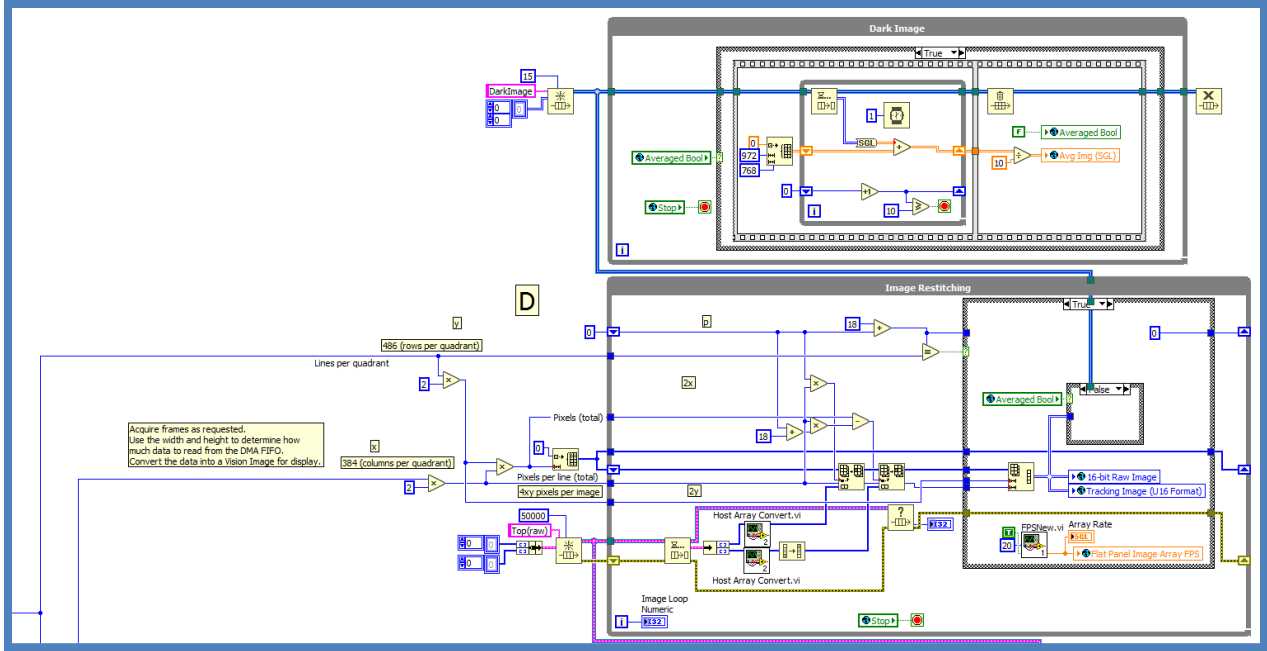


Figure 112: Flat Panel code. Panel 1D-2.

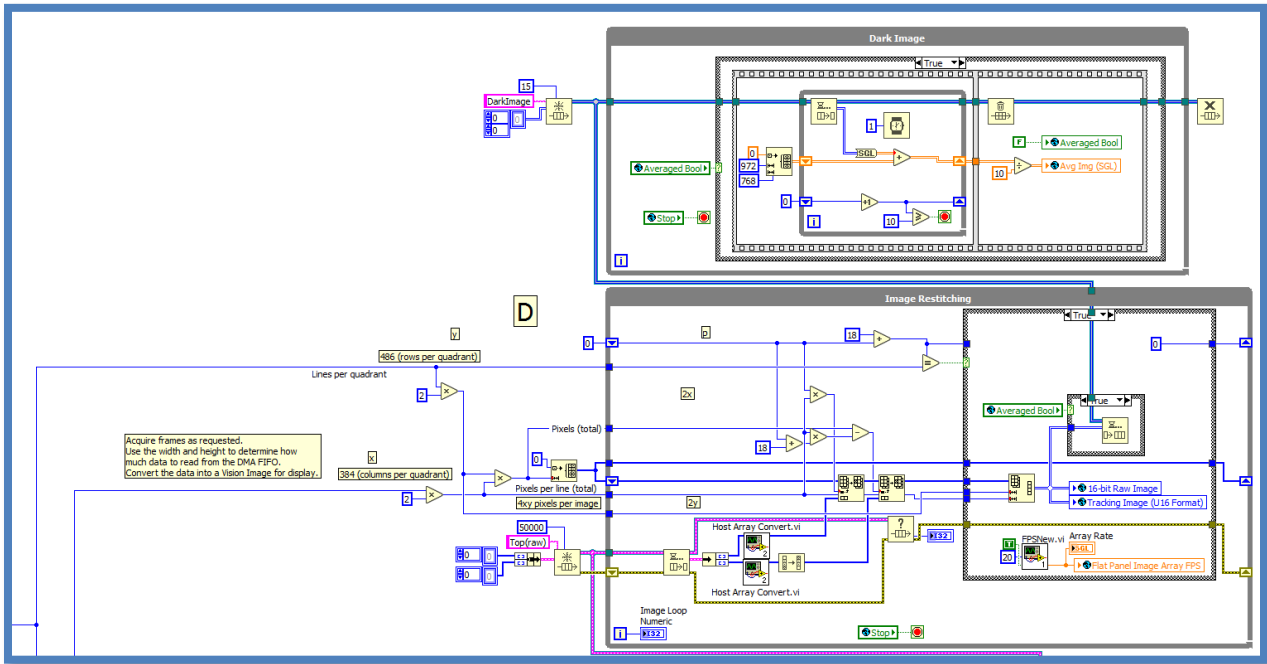


Figure 113: Flat Panel code. Panel 1D-3.

2nd Comp Code

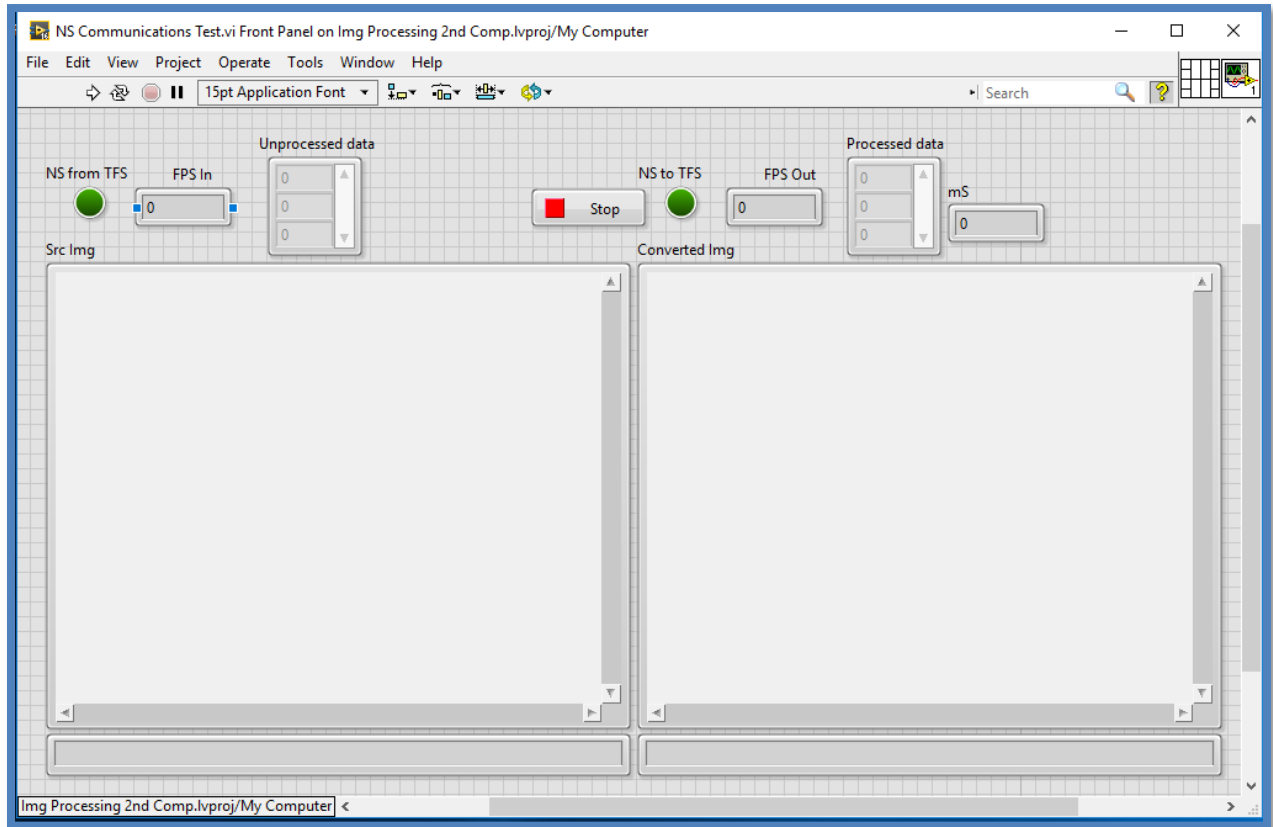


Figure 114: 2nd Computer Communications/Processing interface.

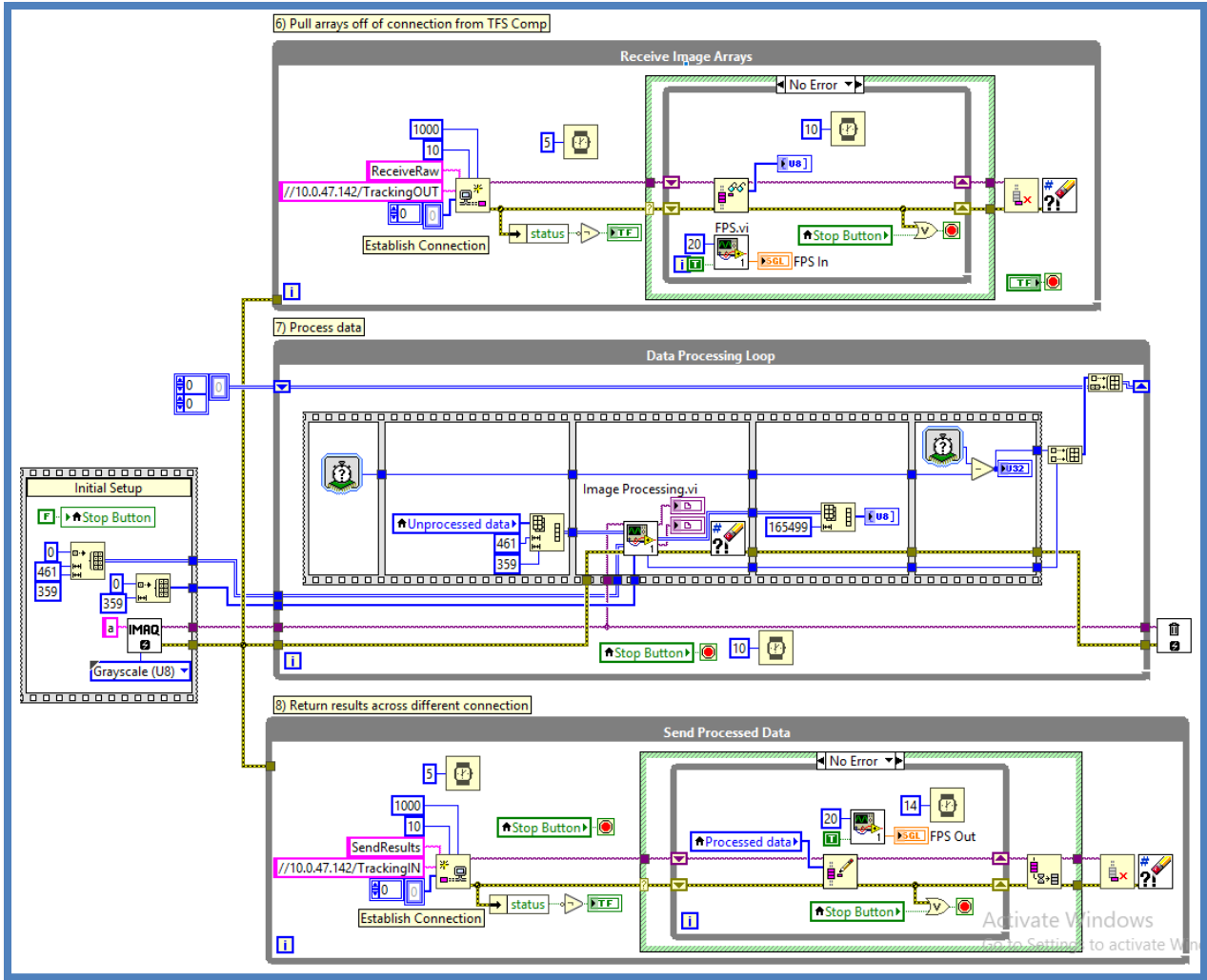


Figure 115: 2nd Computer Communications/Processing code. Panel 1.

Image Processing

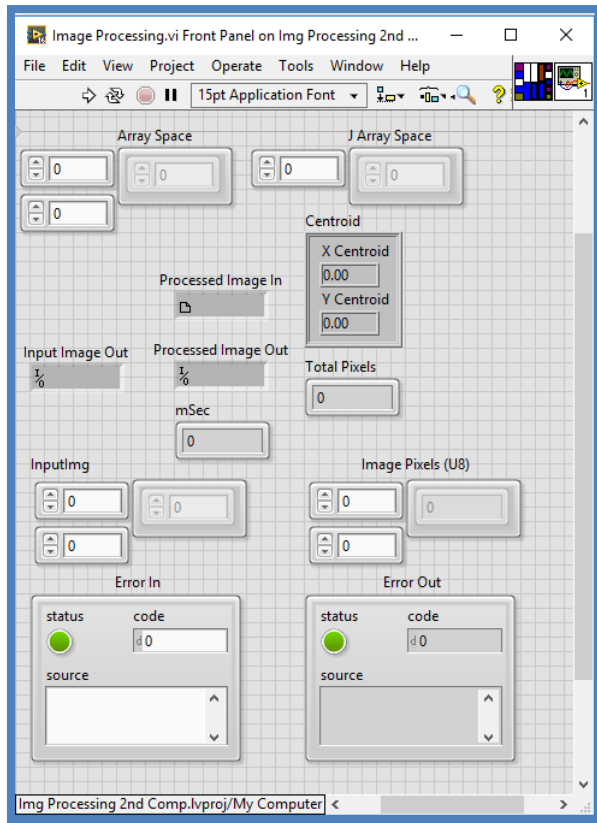


Figure 116: 2nd Computer Image Processing interface.

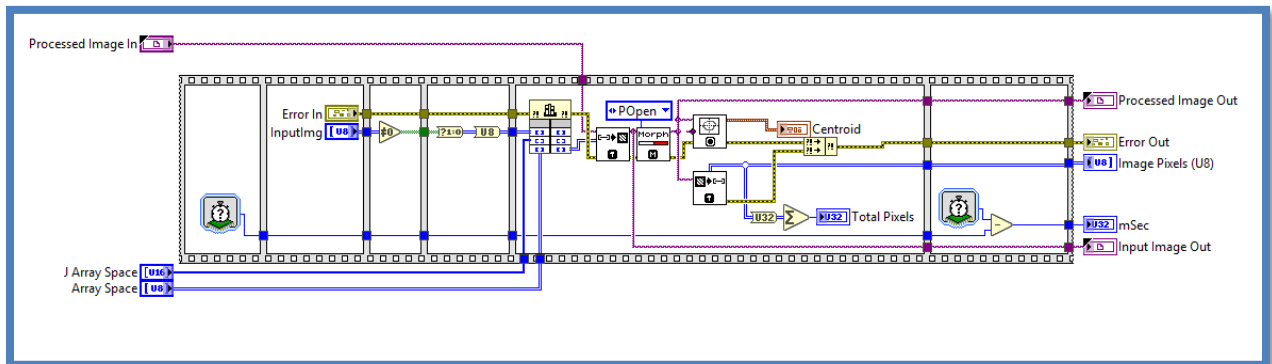


Figure 117: 2nd Computer Image Processing code. Panel 1.

Operator Comp Code

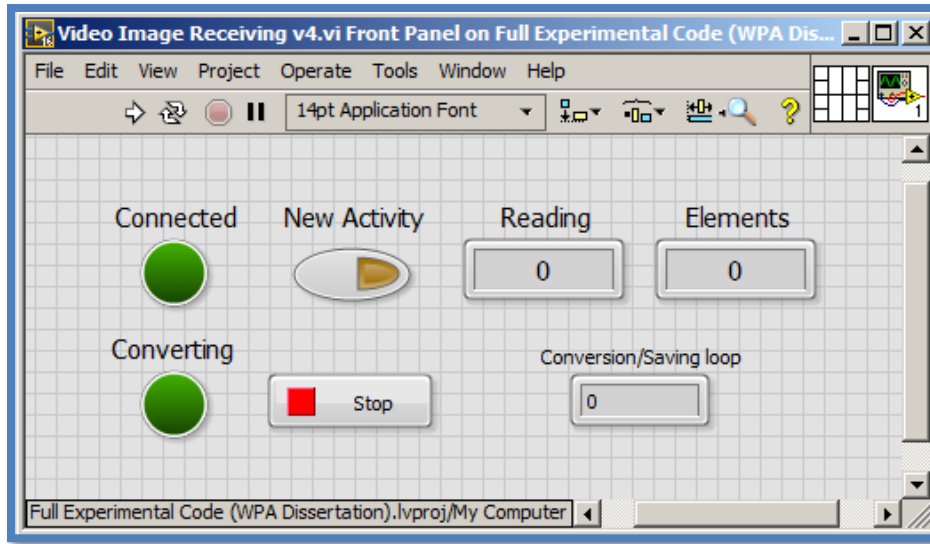


Figure 118: Operator Computer Data Receiving interface.

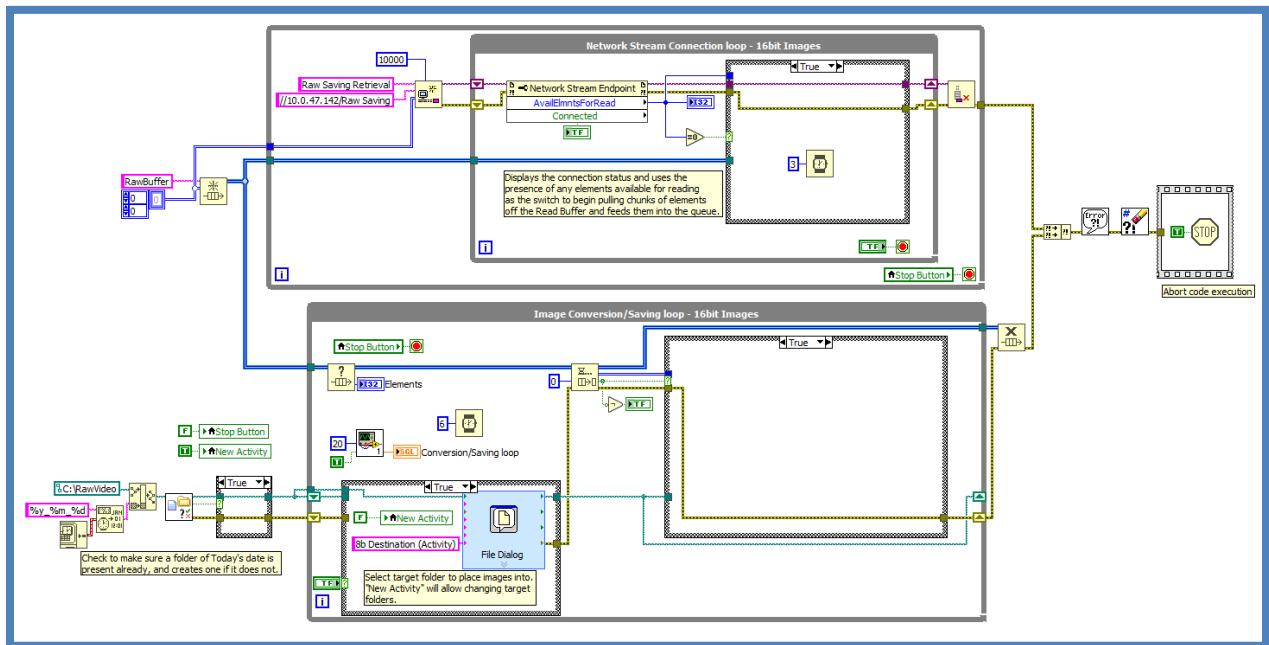


Figure 119: Operator Computer Data Receiving code. Panel 1-1.

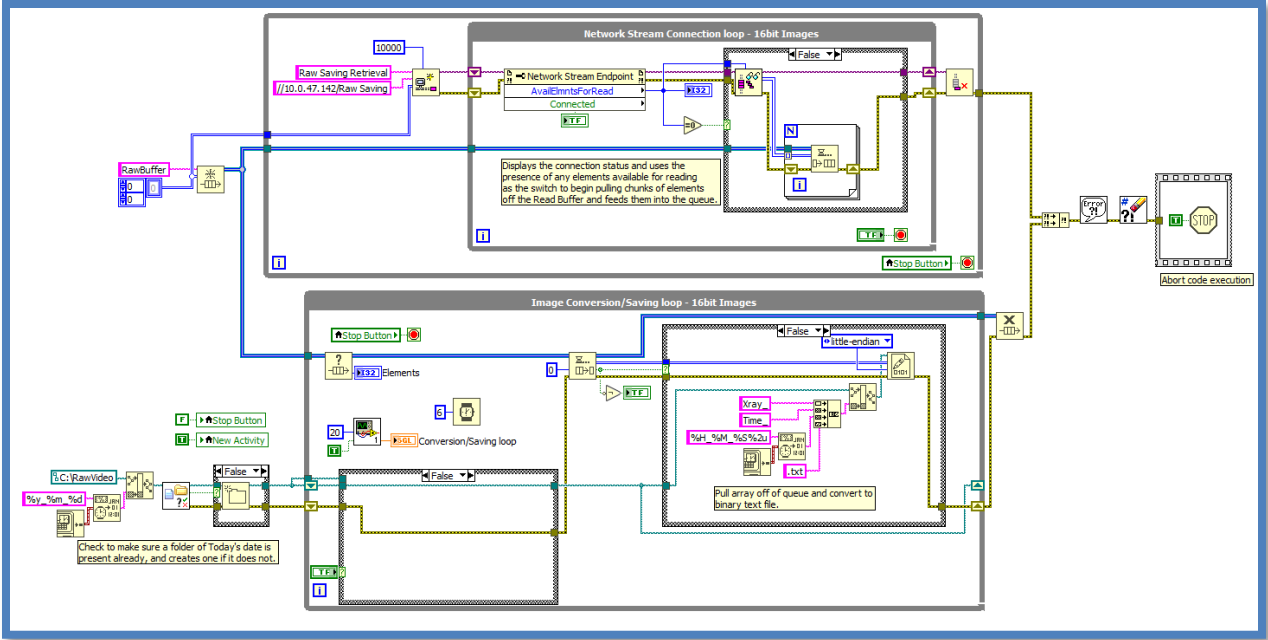


Figure 120: Operator Computer Data Receiving code. Panel 1-2.

Appendix C – Specific Testing Code

Tracking

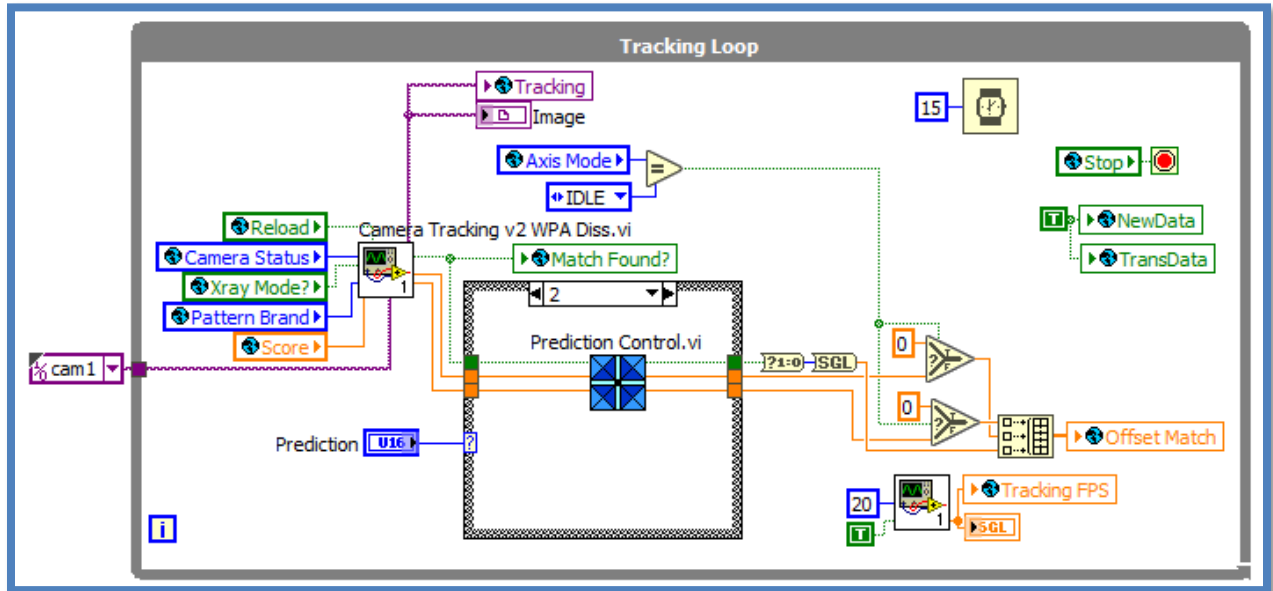


Figure 121: Modified Tracking code for testing.

Particle Analysis

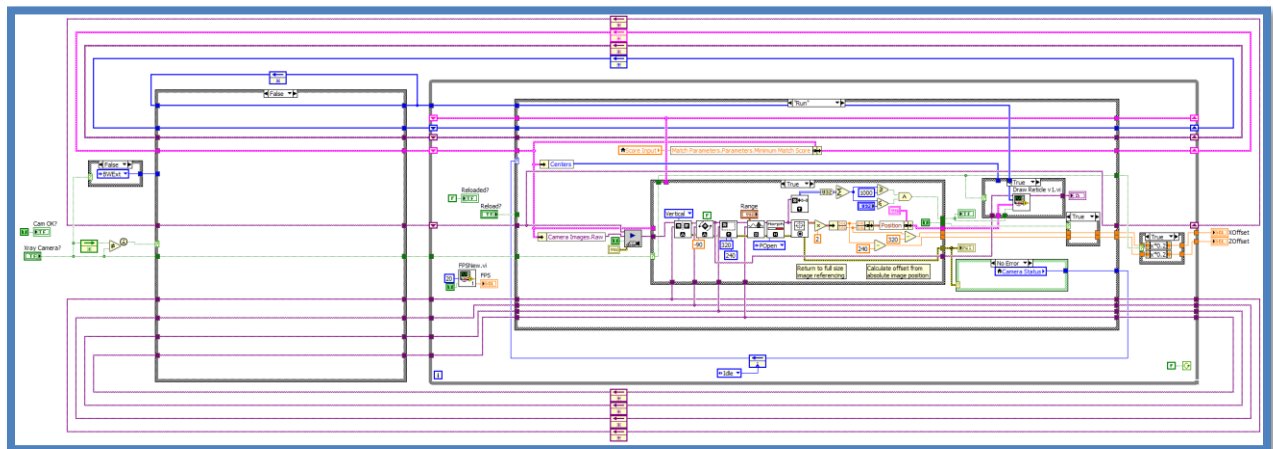


Figure 122: Modified Camera Tracking code for testing. Replaces code in Panel 2-2.

Knee Prediction

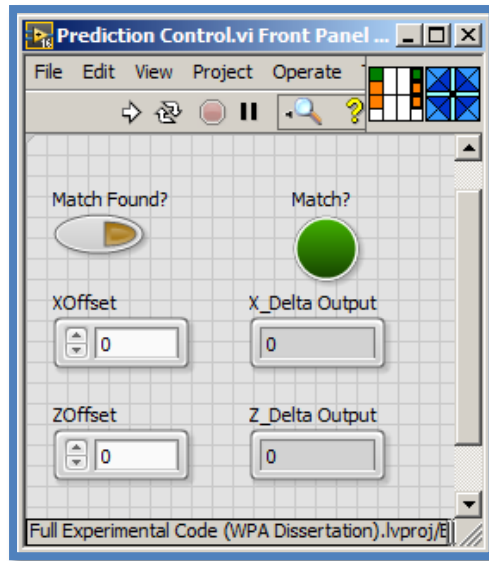


Figure 123: Prediction Control interface.

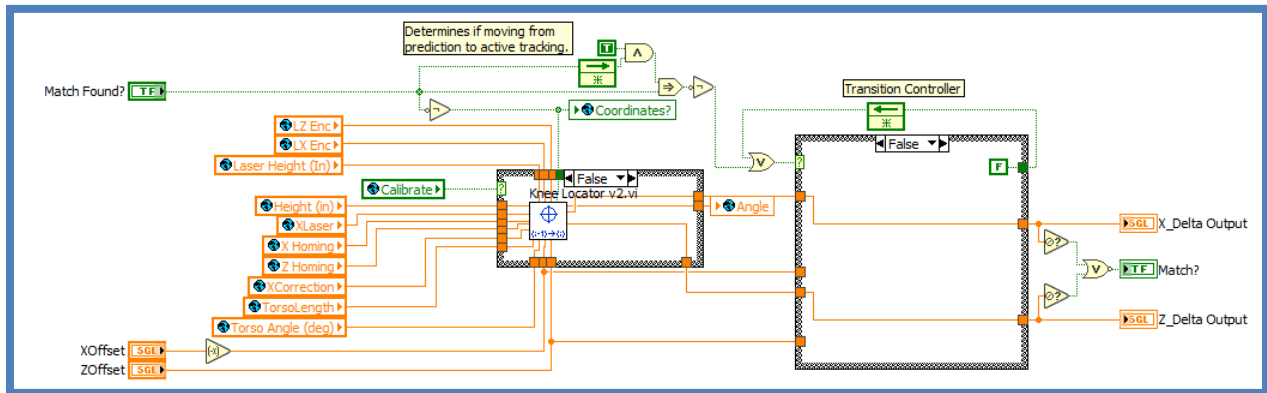


Figure 124: Prediction Control code. Panel 1.

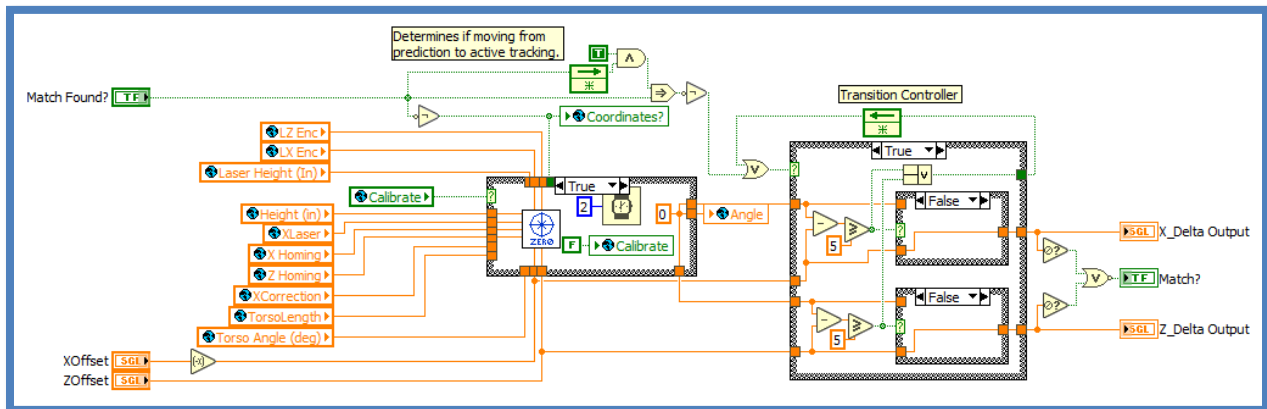


Figure 125: Prediction Control code. Panel 2-1.

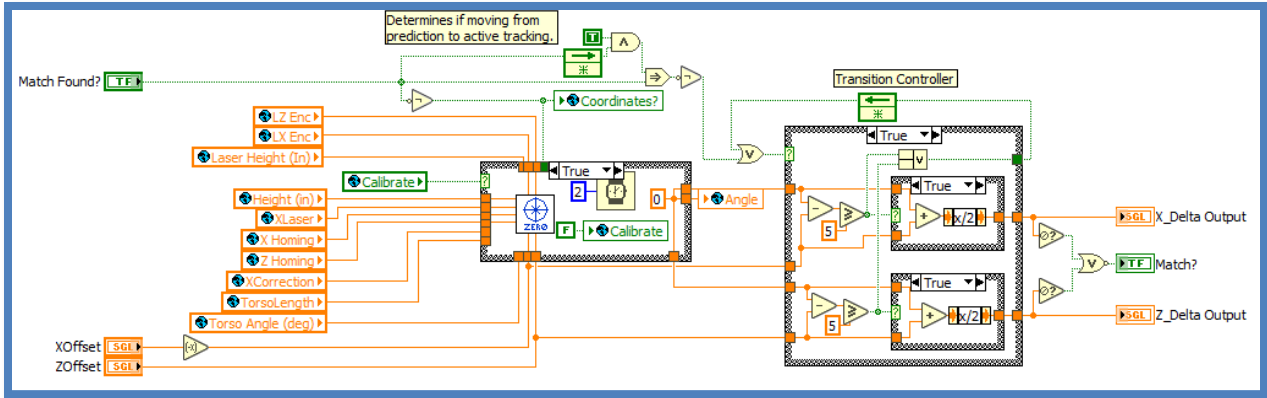


Figure 126: Prediction Control code. Panel 2-2.

Knee Calibrator

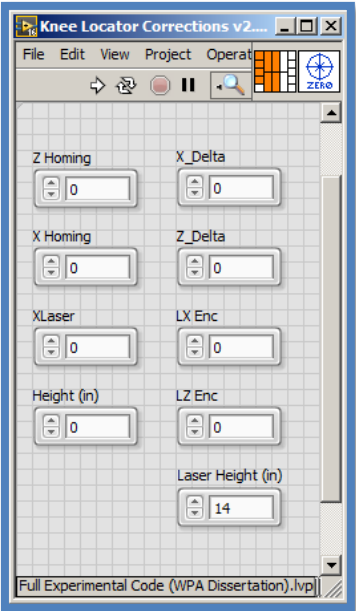


Figure 127: Knee Location Calibrator interface.

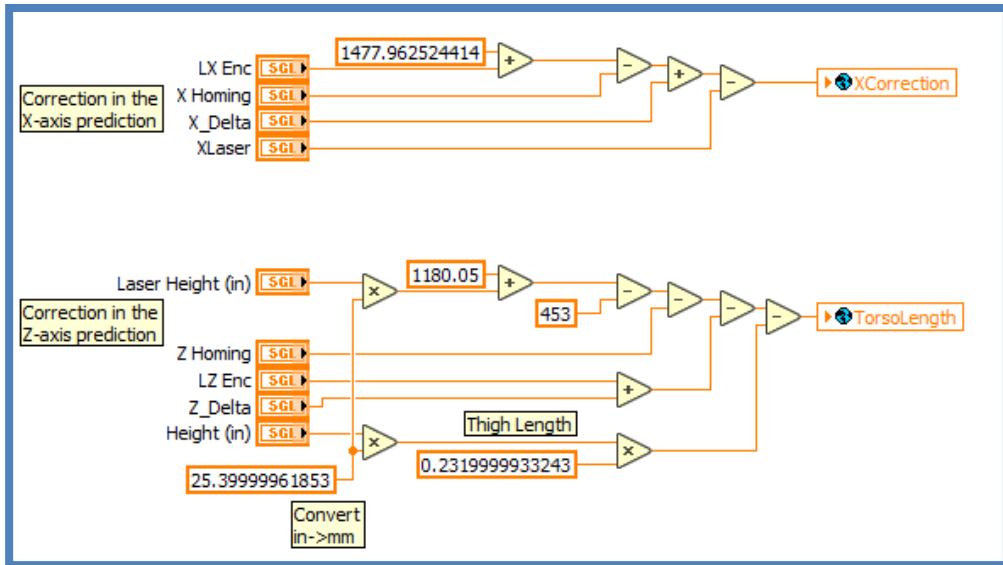


Figure 128: Knee Location Calibrator code. Panel 1.

Knee Locator

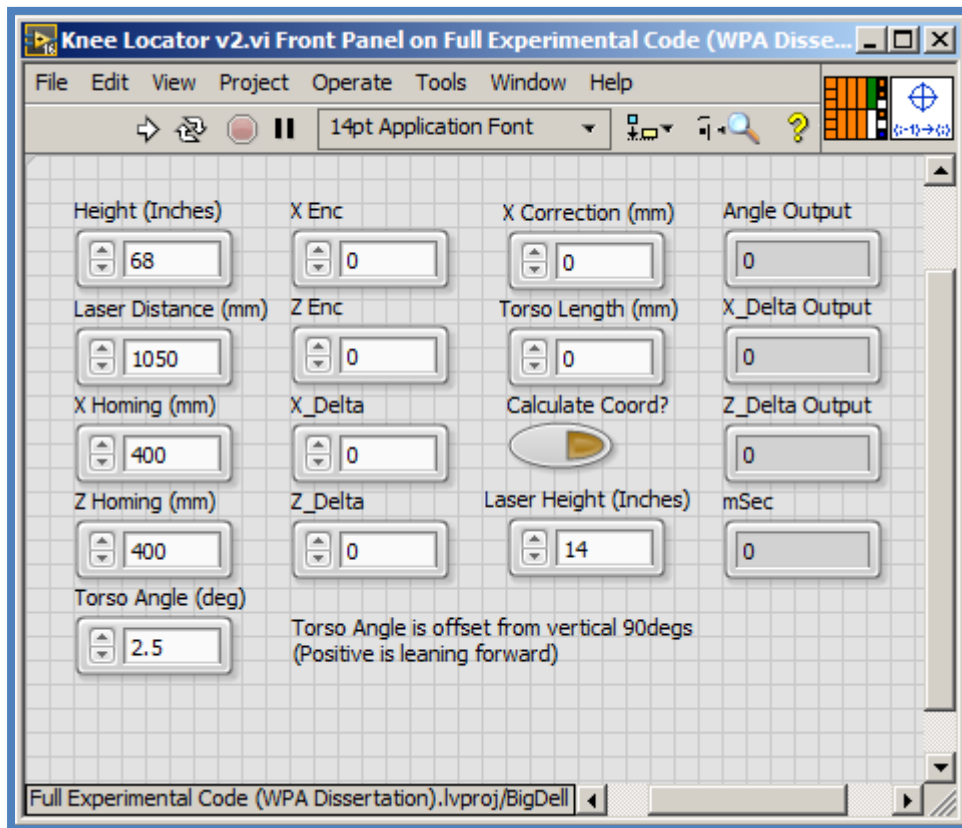


Figure 129: Knee Locator interface.

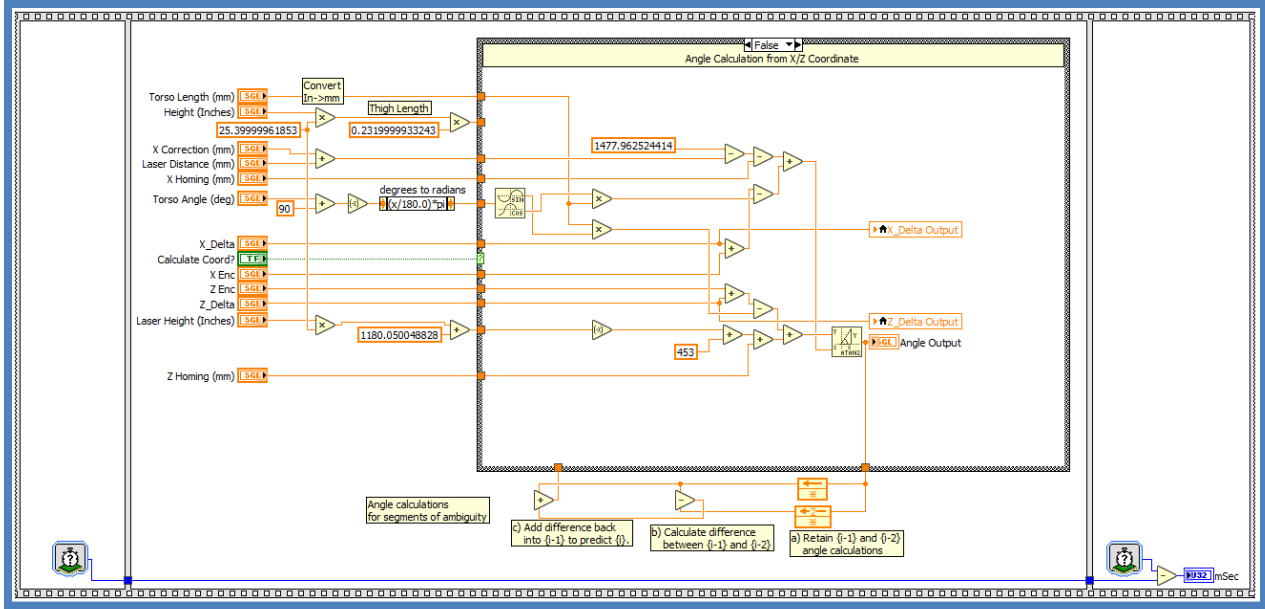


Figure 130: Knee Locator code. Panel 1-1.

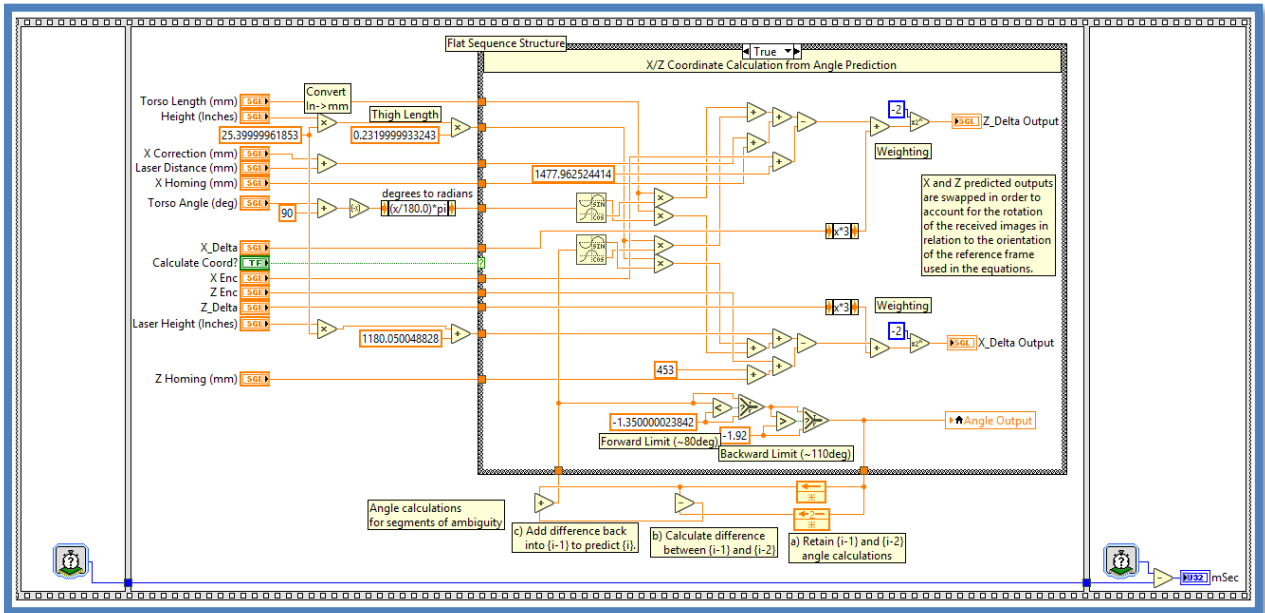


Figure 131: Knee Locator code. Panel 1-2.

X-Ray Display

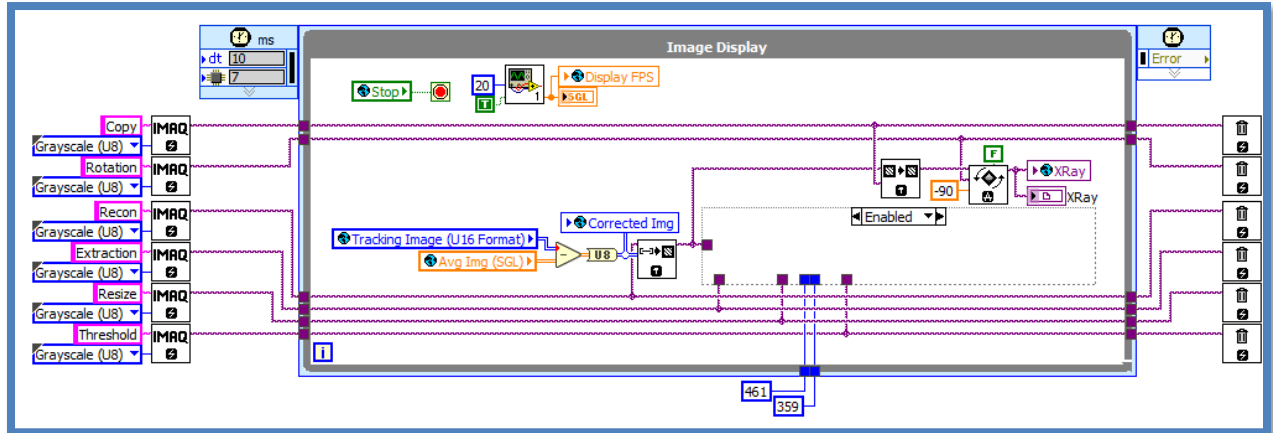


Figure 132: Modified X-Ray Image Display code for testing. Replaces code in Panel 1.

Data Collection

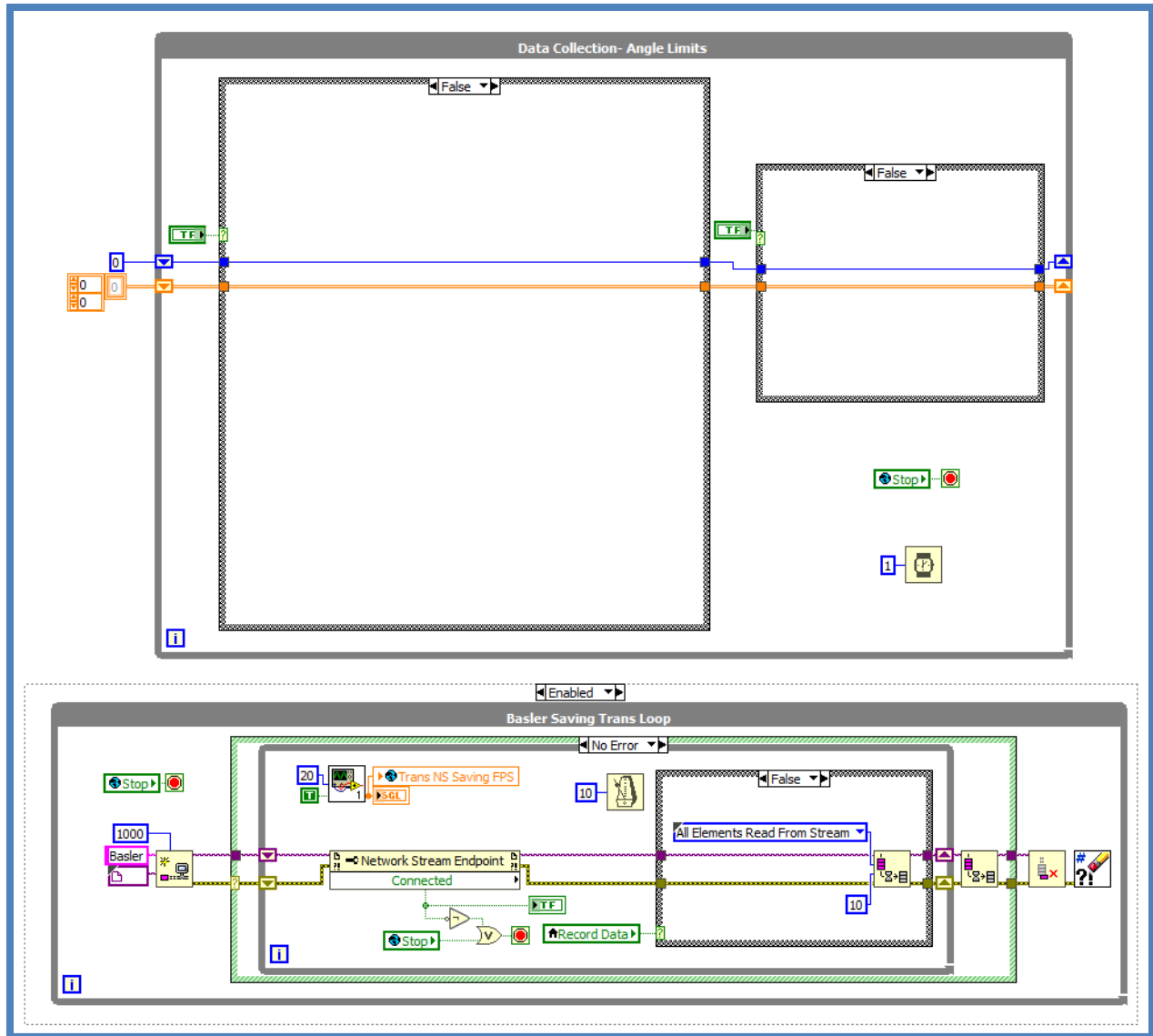


Figure 133: Data Collection code added to Main Control code. Panel 1-1.

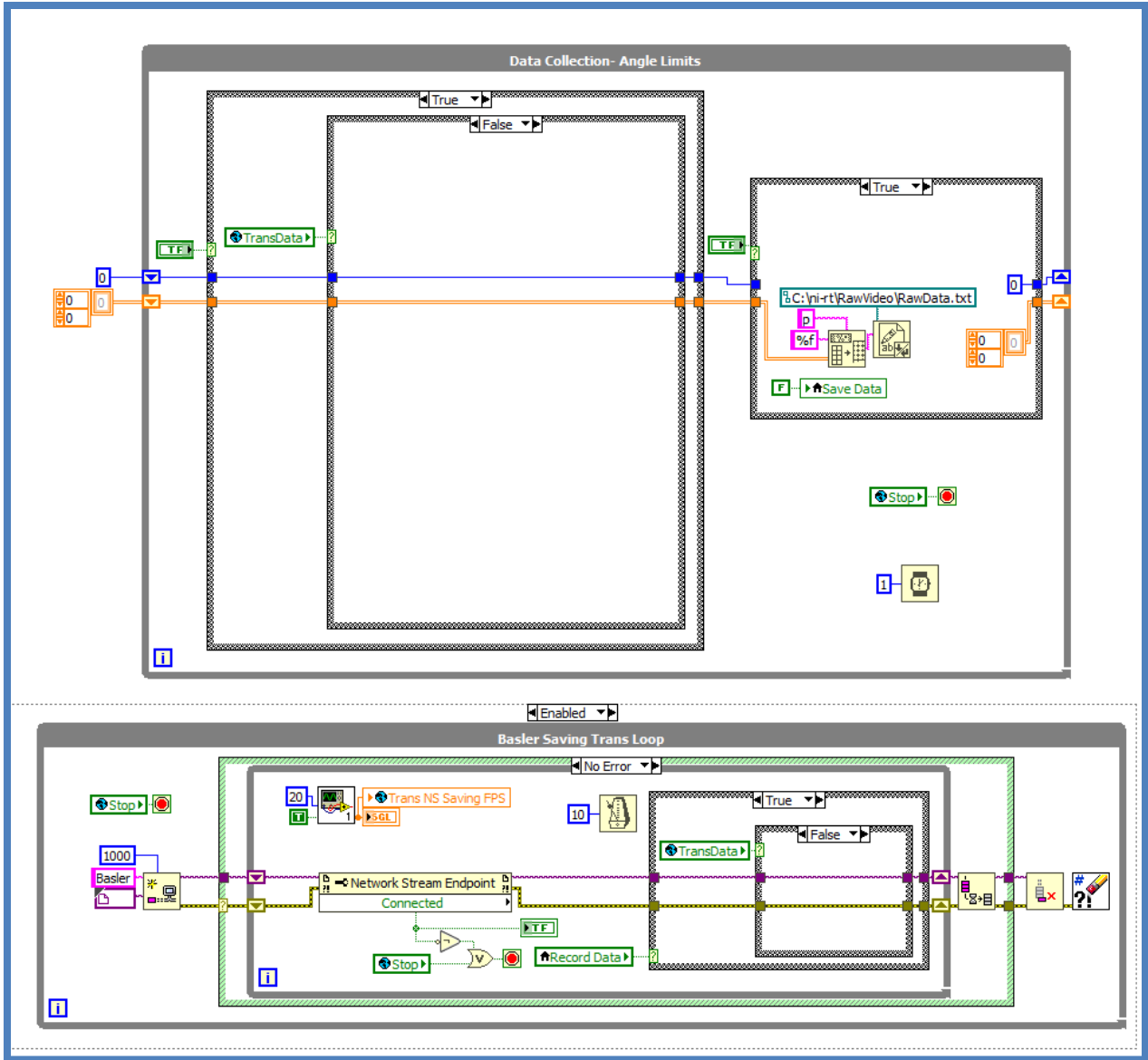


Figure 134: Data Collection code added to Main Control code. Panel 1-2.

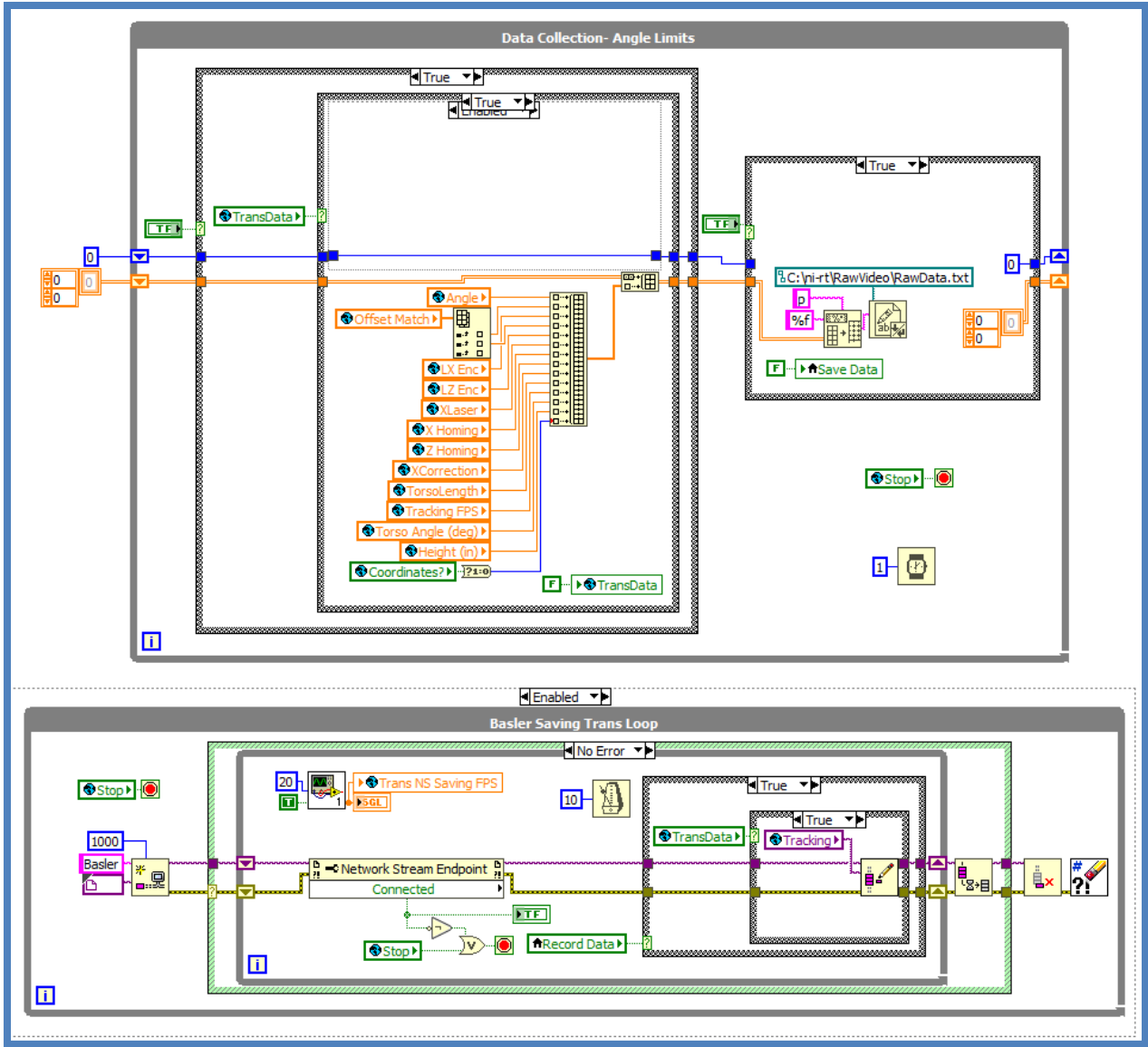


Figure 135: Data Collection code added to Main Control code. Panel 1-3.

Operator Comp Code

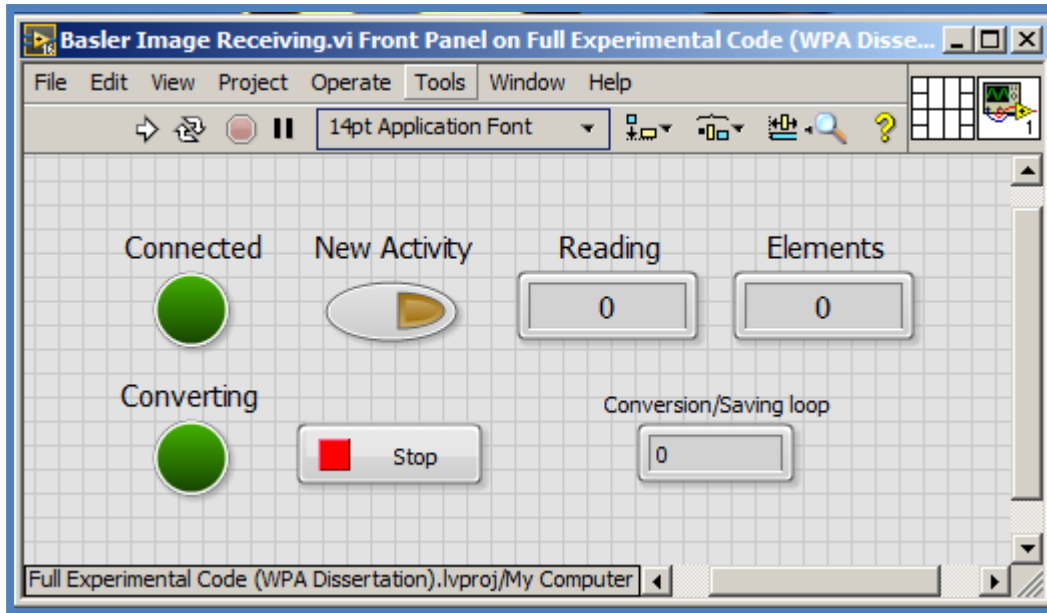


Figure 136: Operator Computer Data Receiving interface for testing.

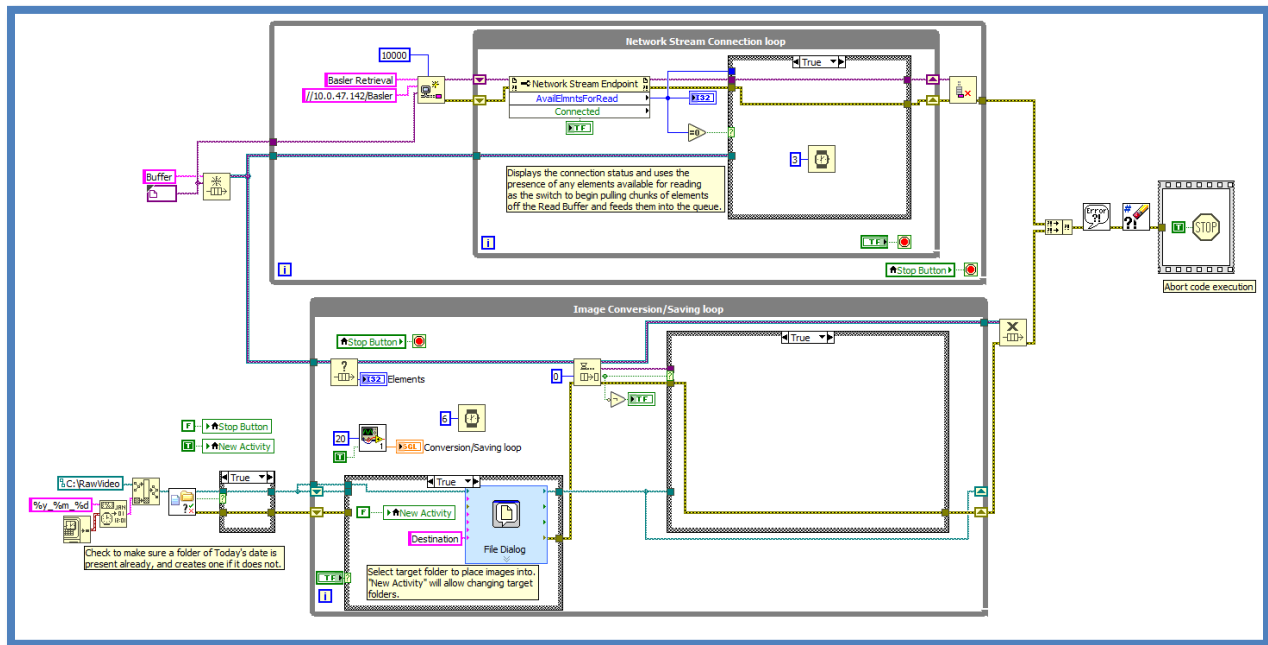


Figure 137: Modified Operator Computer Data Receiving code for testing. Replaces code in Panel 1-1.

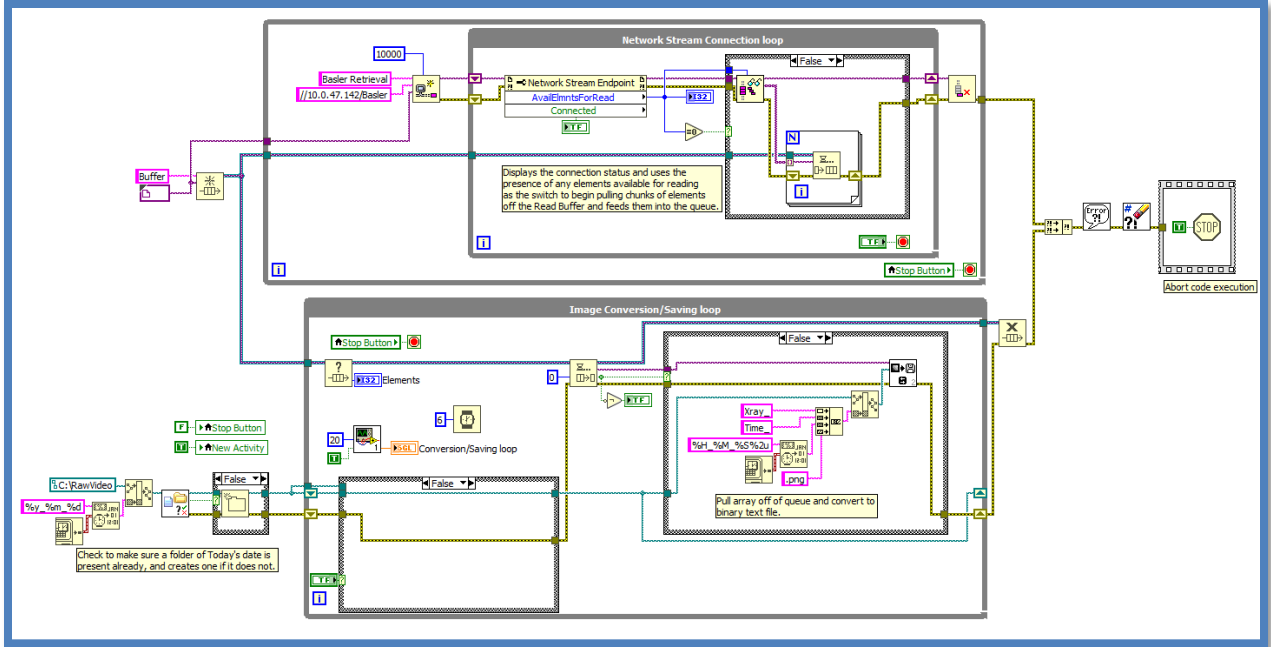


Figure 138: Modified Operator Computer Data Receiving code for testing. Replaces code in Panel 1-2.

VITA

William Patrick Anderson was born in San Diego, CA, to William and Lori Anderson. Being the middle child, he has two sisters: Christine and Patricia. After graduating from Munford High School in Atoka, TN, he went on to attend the University of Memphis. He received his Bachelor of Science in Mechanical Engineering from the university in December of 2010, and he continued on obtaining his Master of Science degree also in Mechanical Engineering from the University of Memphis in May of 2012.

Wanting to continue his education, he applied for and was accepted to the University of Tennessee's Doctoral Program. He joined the BioRobotic's Lab at the University of Tennessee, Knoxville in August of 2012 as a Graduate Research Assistant. His research to date has focused on vision-based servoing and robotic controls and operations.