2018

Reducing Uncertainty in Head and Neck Radiotherapy with Plastic Robotics

Mark R. Ostyn
Virginia Commonwealth University

REDUCING UNCERTAINTY IN HEAD AND NECK RADIOTHERAPY WITH PLASTIC ROBOTICS

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University.

by

Mark R. Ostyn, M.S., Indiana University, 2014

B.S., Northwest Nazarene University, 2011

Director: Siyong Kim, Professor, Radiation Oncology

Virginia Commonwealth University

Richmond, Virginia

July 2018

Acknowledgments

No scientific research is ever done in a void. This work too stands on the shoulders of those who came before me. Even in the work described here within, I cannot begin to take singular credit. In addition to generous support of time and effort from each member of this committee, I am grateful for the assistance of the numerous students and residents who have given their time and energy. The aid given by Daniel Martinez, Melvin Rosario, Rachel Sacks, Matt Miller, Paden King, and Siqiu Wang has been instrumental in the fulfillment of this work. I must also thank Thomas Dwyer and Ross Cruikshank for their substantial efforts in the development of the robotics and foundational mathematics.

For Dr. Hong Yeo, I am grateful for providing an avenue for fostering leadership. His willingness to help support me as a student advisor led to some of the most valuable lessons in my time in this program. For Dr. Laura Padilla, I am grateful for her enthusiasm for novel projects, and her persistent willingness to help whenever the situation required. For Dr. Taeho Kim, I am grateful for providing keen academic feedback to ensure that the forest was never lost in the trees. And for Dr. Siyong Kim, I am ever grateful for selecting me for the opportunities within this program, and for entrusting me with these crazy ideas.

I am also grateful to the VCU Quest Innovation fund, the VCU School of Engineering Foundation, and the VCU Health department of radiation oncology for their financial support. The material support by The Workshop of the VCU Library has also been an invaluable resource.

In memory of Leliana

April 15, 2014 - August 28, 2017

She was just a cat, but one who showed purest love, and was lost too soon to the very death this seeks to heal.

Abstract

REDUCING UNCERTAINTY IN HEAD AND NECK RADIOTHERAPY WITH PLASTIC ROBOTICS

Mark R. Ostyn, M.S.

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University.

Virginia Commonwealth University, 2018.

Major Advisor: Siyong Kim, Professor, Radiation Oncology

One of the greatest challenges in achieving accurate positioning in head and neck radiotherapy is that the anatomy at and above the cervical spine does not act as a single, mechanically rigid body. Current immobilization techniques contain residual uncertainties that are especially present in the lower neck that cannot be reduced by setting up to any single landmark. The work presented describes the development of a radiotherapy-friendly mostly-plastic 6D robotic platform for positioning independent landmarks, (i.e., allowing remote, independent positioning of the skull relative to landmarks in the thorax), including analysis of kinematics, stress, radiographic compatibility, trajectory planning, physical construction, and phantom measurements of correction accuracy. No major component of the system within the field of imaging or treatment had a measured attenuation value greater than 250 HU, showing compatibility with x-ray-based imaging techniques. Relative to arbitrary overall setup errors of the head (min = 1.1 mm, max = 5.2 mm vector

error) the robotic platform corrected the position down to a residual overall error of 0.75 mm $\pm$ 0.33 mm over 15 cases as measured with optical tracking. This device shows the potential for providing reductions to dose margins in head and neck therapy cases, while also reducing setup time and effort.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The fundamental challenge in achieving accurate positioning in head and neck radiotherapy is that the anatomy at and above the cervical spine does not act as a single, mechanically rigid body. Instead, the anatomy is comprised of individually articulated structures between the thorax and the skull that together are mechanically flexible. This flexibility complicates the execution of identical patient positioning from the first simulation scan to the end of the last delivery of treatment. Although thermoplastic immobilization masks have helped reduce the uncertainty in this positioning, some residual errors remain [1][2]. The flexible nature of the spine has long been identified as a source of error in patient setup, but technical limitations reduced the importance of achieving perfect alignment. However, fast and accurate imaging technology and conformal treatment protocols are now commonplace, making complete and accurate alignment more important than ever.

1.1 Background and understanding the flexibility of the neck

To reliably determine the magnitude of uncertainty, reported data from computed tomography (CT) studies is preferable. As recently as the late 1990's, portal imaging was the primary tool used to verify setup alignment of the head in radiotherapy. While portal imag-

ing could be used for setting up bony anatomy, it was also difficult to repeatably interpret. Lacking visualization of even the outlines of soft-tissue structures, setting up the disease to the treatment isocenter using portal imaging was highly subjective [3][4]. Thus, the beginning of the contemporary era is marked by the introduction of integrated kilovoltage on-board imaging (OBI) devices, and their use to acquire cone-beam CT, into the radiotherapy process. Because the technology is volumetric, localization of individual anatomic structures was less ambiguous when compared with portal imaging. Visualization of the complete interior anatomy was possible; the soft photon energies ($<$500 keV) used by CT enabled soft tissue to be visualized, though individual delineation between soft tissue structures would require magnetic resonance imaging. Due to these factors, measurements of setup errors taken by portal imaging are incomparable to those taken with CT.

One of the first CT studies to measure the magnitude of setup errors of specific anatomical landmarks was completed by Zhang *et al* in 2006 [4]. In the research, 6D setup errors were measured for C2 and C6 vertebrae and the palatine process of the maxilla (PPM). Patients were immobilized using thermoplastic masks. While all of the masks used in the study fully covered the face, only a sub-group of patients were assigned masks extending over the shoulders. Setup errors were measured with respect to a reference point defined as the centroid between three radio-opaque markers on the masks. The researchers investigated whether setup errors for each region of interest (ROI) were correlated with errors occurring in the other ROI's. They found that displacements between C6 and C2 and between the PPM and C2 were neither well correlated in the left-right direction nor in the superior-inferior (SI) direction, and concluded that these indicated that angular displacements were present as well, especially pitch (angular displacements about the ML axis) and roll (angular displacements about the SI axis) rotations. The authors note that no observable improvement in setup accuracy between patients immobilized with shoulder-covering masks compared to the face-only masks, citing similar observations from other authors [5]. Zhang *et al* were also among the first to note that the uncertainty in positioning

in head and neck treatments tends to worsen over the course of treatment due to significant tumor shrinkage, especially in the lower neck. Together, these observations form a common theme in understanding the issues caused by the flexibility of the neck: accurate positioning of one anatomical landmark is insufficient to provide proper positioning for the anatomy as a whole.

Similar investigation on the positioning impact of flexibility was done a year later (2007) by Polat *et al* [6]. Polat's study differed from Zhang's by including a greater number regions of interest (Skull, mandible, C1-C3, and C4-C6). The observed rotations were similar to those seen by Zhang, Polat explicitly noted that 3D errors tended to increase in the caudal direction, and recommended differential margins, increasing in size in more inferior locations. Compared to the work by Zhang, Polat did not observe increased uncertainty throughout the treatment course, but did acknowledge that the problem could exist in patients with an initially large tumor, or patients who experience significant weight loss. Additionally, Polat concluded that complete correction of setup errors is not possible with whole-body 3D correction due to the relative motions between the regions of interest.

As the use of OBI CBCT increased, the community-at-large began to investigate the potential improvement to positioning by using imaging on a daily basis, such as that seen in the work by Djordjevic *et al* [10]. In the experiment protocol, patients would be imaged and setup with orthogonal kV images on a daily schedule, and the residual setup errors were compared to controls using various frequencies of setup imaging. The control group were off-line simulations of the resultant dose distributions on the treated patients generated by applying the same dose plan onto the geometry at uncorrected positions. The group looked at the residual setup errors in landmarks including the maxilla, mandible, C1, C2, C4, and C5. Djordjevic's observations concurred with Polat's in observing that larger errors occured more frequently in the caudal direction. Djordjevic went a step further than Polat by calculating the necessary margin size according to the vertebral location, and according to the frequency of image guidance. In the cases without image guidance,

Table 1.1: Reported values of uncertainty for multiple ROI in the neck from [4][6][7][8][9][10][11][12]

Author	Immobilization	Registration Reference	Observation
Zhang	Some mask to shoulders and some mask alone	6D isocenter	Random setup error in maxilla 1.2, 1.0, 2.0 mm for AP, ML, SI. Standard deviations for C6 were 1.9, 2.3, 2.3 mm.
Polat	Mask to chin	Whole image	Standard deviations in skull, 1.4 mm for vector error. Standard deviation of net vector for C4-C6 was 2.4 mm.
Ove	Mask to shoulders	C2	Random setup error in C6 was 3.9, 3.3, 2.6 mm for AP, ML, and SI.
Graff	Unspecified	C2	Standard deviation in vector error for C7-T1 was 1.2 mm.
Piotrowski	Mask to shoulders	Whole image	Standard deviations in C1-C2, 1.4, 0.9, 1.1 mm for AP, ML, SI. Standard deviations for C6-T1 were 2.7, 1.3, 2.9 mm.
Djordjevic	Mask to shoulders	Whole image	Standard deviations in maxilla, 1.4, 1.9, 1.8 mm for AP, ML, SI. Standard deviations for C5 were 1.9, 2.4, 1.8 mm.
Park	Mask to shoulders	Whole image	Standard deviations in in C1-C2 14.6, 3.0, 1.0 mm for AP, ML, SI. Standard deviations for C6-C7 were 14.4, 2.9, 0.8 mm.
Cheo	Mask to shoulders	Whole image	Random setup error in clivus 0.7, 0.4, 0.4 mm for AP, ML, SI. Standard deviations for C7 were 1.3, 1.5, 0.7 mm.

the authors found that the required size for sufficient margins on tumors near the vertebrae would have ranged between 4.5 mm and 9.3 mm, with increasing size toward C5. Comparatively, with daily image guidance, margins near the vertebral structures were between 2.3 mm and 5.0 mm, with little variation along the spine. Similarly to Polat, Djordjevic demonstrated that image guidance alone cannot be used to reduce the uncertainty, since the relative mobility between vertebral structures will contribute some residual uncertainty.

Since setup to a single landmark is does not guarantee alignment of the entire anatomy, some research has been done to determine the best choice of reference structure. Graff *et al* identified six different bony landmarks including the sphenoid, maxillary sinuses, mandible, C1-C2, and C7-T1 as candidates for setup [8]. In a retrospective analysis, Graff simulated dose delivery to a group of patients using each of these substructures and compared the perturbations to the dose distribution to the choice of landmark. On average, the least amount of dose variation occurred when setting up to C1-C2 while the greatest amount of dose variation occurred when setting up to the sphenoid sinus. However, large errors were still observed when using C1-C2 as a reference point; errors between 1.0 mm and 4.7 mm were observed in both the extreme cranial caudal locations. While no subregion was perfect, the authors note that using C1-C2 as a reference structure for setup produced less anatomical variation and less dose perturbation relative to using automatic registration of the entire image.

Observations from these and other similar studies are compiled in Table 1.1. Based on these measurements, one can estimate that the residual uncertainty in the setup of the lower neck is approximately 3 mm per axis, if the patient is set with either upper neck landmarks or with whole-image registration, and if the patient is setup with shoulder-covering immobilization masks. While some reports suggest that the greatest uncertainties exist in a particular direction, no such conclusion can be drawn from looking at the data as a whole. Together through these this exploration of setup uncertainties, one can observe the pervasive theme that since no singular landmark is perfect, other measures must be

explored to further reduce uncertainty in setup.

1.2 Other uncertainties in head and neck treatments

While inter-fractional setup errors are a significant source of uncertainty, further deviations from the planned position can also occur after commencing delivery of a single fraction. Intra-fractional errors tend to occur slowly, often unobservable second to second, but cumulate over minutes, becoming a significant source of uncertainty given enough time [13]. Intra-fractional errors also tend to be random [14], making prediction of their direction difficult, and vary significantly between patients [15].

Early measurements of intra-fraction motion for treatments to the head and neck came from studies of swallowing as measured by video fluoroscopy [16]. Swallowing is common to occur during treatment, occurring in 70% of a cohort studied by van Asslen, so the effect of swallowing on positioning must be considered. Displacements of the larynx of up to 8.1 mm have been observed along a single axis, though the anatomy approximately returned to the start position in nearly every instance, and only accounted for 0.45% of the total treatment delivery time. Still, given that these magnitudes are so large, clinicians should watch for large deviations[1] and be ready to reposition if the post-swallowing position has deviated past some tolerance in hypofractionated treatment paradigms where single displacements could have a much greater impact on the total dose delivery.

Intra-fraction motion of the skull has been measured by an infrared marker-based optical system (BrainLab AG, Munich, Germany) [13]. Kim *et al* attached markers to a bite block, and observed patients for 15-minute fractions. The bite block ensures that motions captured by the optical system corresponded to true movement of the skull because the maxilla is rigidly connected to the skull. This method has greatly improved precision in both time and space relative to portal films [17]. While portal films have been cited as mostly being subjective below 0.5 mm [18], Kim reports infrared marker-based systems

[1] Perhaps with optical guidance, or an electronic sensor.

capable of accurately detecting 0.03 mm - 0.04 mm per axis.

In the study of 6 patients, patient motion was recorded for the full duration of every fraction, and the data was analyzed through a few perspectives. One of the most useful tools used for understanding the impact of intra-fraction motion is the uncertainty time histogram (UTH), like the one seen in Figure 1.1(b). In a UTH, the magnitude of motions are plotted against the probability of that magntitude or less occuring. Therefore, a reader can quickly observe what magnitude of motion or less may occur for a given probability metric.

Kim reports a few significant observations in this study. First, for the cohort, the overall measured vector displacement was less than 2 mm for all of time for all patients, was less than 1 mm for 90% of all timepoints for all but 1 patient ($\approx$70%), but greater than 0.25 mm for at least 80% of time points for all patients (Figure 1.1). This suggests that while large net displacements are relatively uncommon, small errors (between 0.1 mm and 0.5 mm) are present for the overwhelming majority of fraction duration. This finding implies that even with perfect image guidance, margins are still necessary to ensure adequate dose coverage, unless the motion could somehow be suppressed. Second, Kim observes that the cumulative displacement due to intra-fraction motion tends to increase with treatment duration from setup (Figure 1.1(b)). This effect becomes more important in complex plans that cannot be delivered quickly, but emphasizes the benefit of rapid delivery made possible by flattening filter free machines [19][20].

While these errors are nearly negligible when compared to the setup errors described earlier, they do contribute to the overall uncertainty considerations when prescribing dose margins. Therefore, any attempt to reduce their impact is still valuable, although reducing setup errors is more so.

Especially when compared to earlier studies, the observations in Kim's report were more limited by the mechanical attachment between the observable objects and the corresponding disease, rather than on physical limitations of the measuring equipment. The

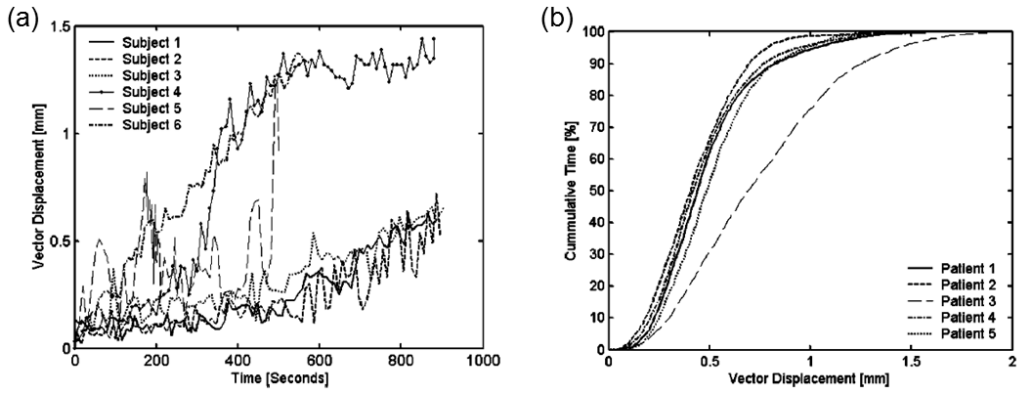


Figure 1.1: (a) Head and neck intra-fraction motion and (b) Time histogram of intra-fraction motion

greatest uncertainty was the correspondence of motion between the bite block and the patient's skull, with the author reporting 0.5 mm $\pm$ 0.4 mm for patients with full dentition. This trend is also present in other modern motion-monitoring paradigms, such as the uncertain correspondence between skin motion measured by surface imaging and the motion of underlying interior anatomy [21][22][23].

Contemporaneous work by Linthout *et al* [15] verify Kim's estimations of intra-fraction motion. In Linthout's study, patients were immobilized with five point immobilization masks covering the shoulders, and an infrared reflective marker system was used to monitor patient motion in real-time. Reflective markers were attached onto the immobilization mask, leaving some ambiguity about the correlation between marker motion and target motion. In addition to using the IR system, stereoscopic x-ray images were taken at beginning and end of each treatment session to measure the change from setup position. Using this data, the bony anatomy had moved by similar, if not greater displacements than the IR data would suggest, including rotational displacements up to $\approx 6^{\circ}$, (Figure 1.2(a)). While most of the maximum deviations per patient were below 2 mm, some were as large as 4 mm (Figure 1.2(b)). Like other studies [3][10], Linthout's detection of 6D errors was based on rigid registration of the entire visible bony anatomy.

Surface imaging has also seen widespread use for improving patient setup [24] includ-

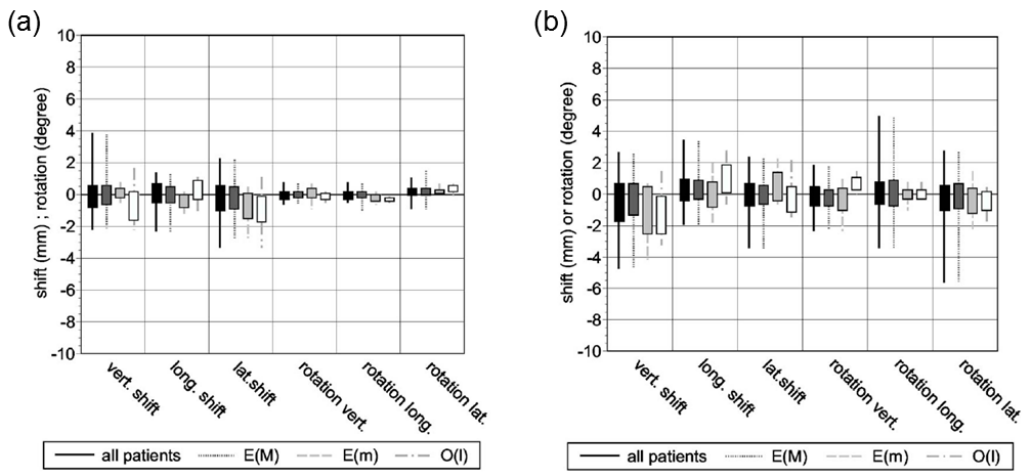


Figure 1.2: (a) Maximum intra-fraction displacements patients measured by IR markers, where each color is a different immobilization system, and (b) net intra-fraction displacements as measured by stereoscopic x-ray.

ing head and neck sites [25]. Popular surface imaging systems like AlignRT (Vision RT, London, UK) have been shown to be accurate to within 0.2 mm $\pm$ 0.1 mm or degrees per axis of translation when compared to infrared marker systems [26], though the accuracy of the system varies significantly depending on the parameters of the reference surface. For example, studies have observed that the accuracy of surface imaging can be affected by the shape and color of the surface being monitored [27], and others have observed that while the measured surface can be registered to a surface extracted from CT data, superior results are obtained by matching to a reference surface taken by the camera system [26]. While surface imaging also experiences the same ambiguities about the correspondence of surface features to internal anatomy, surface imaging systems can measure displacements of rigid objects with reasonable accuracy [28][29].

Techniques other than infrared marker monitoring of measuring intra-fraction motion have also been explored, but most others have not reached widespread use. For safety, radiographic imaging is rarely used to perform real-time monitoring of patient motion, though movement of the prostate has been monitored with x-ray using on-board imaging equipment [30]. Therefore, other techniques must be explored. One strategy was explored by Farahmand, who developed a novel solution using accelerometers, though only pre-

liminary phantom development occurred. The technique could be further explored, but verification of the time sensitivity is required, since only very small, slow motions are expected in real patients [31]. Inata developed a pressure sensor-based system for head and neck monitoring, but is also only in the early stages of development. At the most recent report, Inata describes the system as capable of detecting 1-mm motions with 67% accuracy [32]. Like surface imaging, each of these technologies are limited to explicitly measuring external features.

Very recently, the availability of magnetic resonance imaging (MRI)-guided treatment machines have given the most precise picture of the actual motion of the disease during treatment. Relative to CT, MRI provides the superior soft tissue contrast needed to delineate disease from healthy tissue [33][34]. Since MRI does not utilize ionizing radiation, it may be used to continuously image a patient without the risk of causing harm. In radiotherapy purposes, MRI has been limited to diagnostic usage until the past few years due to the technical challenges in integrating strong magnets into radiotherapy delivery devices. But, these technological challenges are being overcome. With the introduction of commercial radiotherapy systems with incorporated MRI though, accurate measurements of internal anatomy in a treatment setting could be potentially be made in the near future. However, no literature could be found that used MRI to measure real-time motion of the head at this point in time.

The only non-imaging technology developed to directly measure the position of internal target anatomy was the Calypso system (Varian Medical Systems, Inc., Palo Alto, CA), that uses a series of implanted RF transponder to calculate real-time positions of internal features, using a nearby electromagnetic array. Although the technology is proprietary, the likely method of localization is by lateration of received signal strength [35], where an RF pulse is emitted by an array element, received, and rebroadcast by the transponders. The return strength is then measured by the array. With enough measurements, a position estimate of the transponder is possible to sub-millimeter resolution with fast time resolution

[36][24]. However, the necessity of the array limits treatment delivery options, since x-ray radiation cannot pass through the array without being attenuated or scattered. A review of the literature found no studies that used the Calypso system to measure intra-fraction motion of the head and neck.

Geometric uncertainties affect decisions about how best to deliver a radiation dose to the patient. In an ideal photon external beam treatment, the target would exist as a fixed region of space in a volume of perfectly specified shape, location, and material, $V(x, \rho)$, repeatable for every fraction of delivery. However, due inherent uncertainties, the requirements of repeatable position and shape are not achievable. Therefore, dose margins are used to account for this uncertainty.

Van Herk provided the influential analysis for prescribing dose margins, treating uncertainties as a weighted combination of random and systematic variables [37]

$$m = \alpha\Sigma + \gamma\sigma \tag{1.1}$$

The symbols α and γ are weighting factors derived from confidence intervals, and Σ and σ are the systematic and random variations, respectively. The goal of this model is to provide the necessary margins to probabilistically ensure that a certain percentage of the patient population has a given probability of a certain volume of target receiving the prescribed dose. This relationship creates the obvious trade-off between the volume of healthy tissue treated by increased margin size and the certainty of ensuring that the target regions receive sufficient dose. Van Herk assumes an acceptable compromise between margin size and level of confidence as ensuring that all of the target region receives at least 95% of the prescribed dose in 90% of all patients treated. However, van Herk's relationship also implies the clear implication that the safest way to treat less healthy tissue is to reduce the uncertainty in delivery.

Van Herk's recommended prescriptions on dose margins has had significant influence

on many of the studies presented here. Cheo used the margin recipe to calculate a differential margin for each level of the neck, ranging from 2.33 mm at the skull (SI) to 6.52 mm at C7 (ML) [12]. Using a similar approach, Djordjevic prescribes margins ranging from 2.3 mm at C2 with image guidance (SI) to 9.3 mm at C5 with no image guidance (AP), and margins as large as 5.0 mm for C5 (AP), and 5.9 mm for the maxilla (AP) [10].

As the cumulative dose to healthy tissue increases, so does the risk of developing early or late effects, including permanent tissue damage or additional cancers [38], [39]. Bressan reports that treatment of cancers in the head and neck with radiation are associated with patients developing swallowing disorders, dysgeusia, oral mucositis, and xerostomia [18]. All these have direct impact on patients' ability to comfortably eat, damaging the patients' quality of life. Unsurprisingly, the occurances of these side effects are also associated with patient weight loss, which can also negatively affect the positioning accuracy, necessitating even larger margins [40][41], though some studies indicate the effect is more ambiguous [42]. Therefore, steps must be taken to reduce the size of prescribed margins without compromising confidence of treatment efficacy.

1.3 Robotic attempts to improve positioning

Robotics have long been used for reliably moving patients into specific setup positions, and could be used to provide active motion compensation. Seppenwoold *et al* were some of the first researchers to examine the feasibility of robotic compensation. The authors simulated the control of a robotic-mounted linear accelerator (CyberKnife, Accuray, Sunnyvale, CA) to follow the respiratory motion of extra-cranial tumors in real-time [43]. In the simulation, the 3D position and trajectory of the tumors were periodically measured (1 image per 1-5 minutes) with implanted gold markers via x-ray. Simultaneously, the respiratory motion was tracked with an optical surrogate, and the data from both methods were combined to build a prediction model for each patient. The robotic arm was instructed to perform

motion vectors in compensation to the predicted trajectories, and the residual error was measured. In all cases with compensation, the RMS error was reduced to less than ± 2 mm for each direction, with uncompensated displacements as large as ± 6 mm, indicating that at least from a conceptual level, active motion compensation was possible.

Of course, most radiotherapy clinics do not have such a mobile radiation source, so more universal solutions are preferred. Alternatively to moving the treatment source, Haas *et al* proposed using a robotic treatment couch to compensate for patient repiratory motion as observed based on optical guidance [44] for 3D correction. Haas's examination was exhaustive, including design of control algorithms, modeling of couch deflection from loading, and experimental validation. The control algorithms developed relied on proportional, integrating, and differentiating (PID) tracking circuits to determine if and how much the couch should move in response to patient motion. In a PID system, the circuitry attempts to increase or decrease an input parameter to make the output match some specified fixed point. In Haas's work, each axis actuates independently of the others using a DC motor, and the output actuation of the treatment couch experiences mostly linear motion relative to the input voltage on the motors. In a simplified model, the calculation would only need to determine the required time on and the applied power to the motors to move to any arbitrary position on a linear axis. However, Haas recognized that friction forces and applied loads can make the system behave non-linearly, and thus experimented with a variety of models in the control algorithms. With these tracking system, Haas reports vector errors between 0.5 mm and 2.0 mm, varying by breathing phase and axis. The overall displacements were about 20 mm along the longitudinal and vertical axes.

Stewart platform 6D couches have appeared in clinics [45][46][47]. A Stewart-platform is a specific type of mechanical configuration comprised of six extending linkages commonly used for 6D control applications (Figure 1.3). They are also frequently referred to as "HexaPOD" platforms. While the use of 3D couches has been investigated for use of motion compensation [48], Hermann *et al* investigated using a Stewart platform treatment

Figure 1.3: A prototypical Stewart platform

couch to perform 6D motion compensation of lung tumors [45], a 6D couch could reliably reduce surrogate motion to less than ± 0.5 mm per axis (original motion up to ± 1.5 mm)

Hermann also compared two control schemes: model-predictive control versus velocity-based positioning. Control of Stewart platforms is non-linear and the control axes are coupled. This configuration makes time-based positioning more complicated because the positioning response in a treatment coordinate system is not proportional to any single signal from the control system. The model-predictive and velocity-based systems attempt to find efficient non-linear control vectors for adaptive positioning. In the model-predictive formulation, a cost function was used to find the minimal effort trajectory change to make accurate 3D motion compensation. Comparatively, in the velocity-based position control, the velocity was constrained to be less than an artificial limit while best matching the velocity vector of the target and where the new 6D target position was calculated on an instant-to-instant basis. In a simulation comparing these two methods of tracking a sinusoidal target, the RMS and maximum error were significantly lower in all cases for the velocity-based position control compared to the model predictive control.

Finally, work has been done by Wiersma investigating active motion compensation for head and neck radiotherapy with independent positioning of the head. Wiersma developed a 3D motion control stage for improving stereotactic radio surgery (SRS) [49]. Wiersma sought to create an innovative head immobilization technique as an alternate to cranial

frames and thermoplastic masks, hypothesizing that gravity could suppress most rapid motions while slow drifts could be corrected by a robotic system.

Whereas Seppenwoold's and Herrman's motion compensation relied on predictive models, Wiersma's used a simplistic correction scheme based on optical tracking: if a three-second average showed greater displacement than a 0.2-mm threshold, a command was issued to move the surrogate back to the reference position. Wiersma noted that tracking these slow-occuring small motions is well-suited for head and neck treatments because motion is limited from both cardiac and respiratory sources. Motion was tracked in a sample of healthy volunteers using infra-red markers attached to a bite block. With motion compensation, the net positioning error was less than 0.5 mm for greater than 95% of the observation period for all but one patient. Without correction, some patients were measured as having moved 2.5 mm over a 15-minute observation time, as determined by back-tracing the motion corrections. Although the timescales were similar, the magnitude of intra-fraction motion in this sample group was significantly greater than that described by Kim [13] likely due to the author's decision to not immobilize the volunteers with thermoplastic masks.

Wiersma's system would be later utilized in a 2016 study, as a ground-truth reference for a new implementation of a 4D robotic motion compensation system, allowing for translational corrections and pitch corrections [50]. The group tested the system with the 6D motion phantom previously developed and with a human volunteer without immobilization, measured with an infrared reflector based optical system. For the test with the motion phantom, the platform was moved so as to simulate a human subject with motion in the range of ± 2 mm and $\pm 0.45^{\circ}$ for each axis over a 15 minute simulated treatment. With motion compensation, the measured position was within the range of the ± 0.2 mm and 0.2° noise of the measurement system for both translations and rotations. With the human volunteer, the results were similar. The group also tested having the human volunteer make a severe displacement by simulating a cough. The simulated cough created a 4-mm

and 4° displacement that took approximately 6 seconds to transpire and the position was successfully corrected within ±0.2 mm and 0.2° 14 seconds later. Since position deviations at each second in time each contribute toward the RMS error, the response time is worth considering.

The group also developed a 6D motion phantom using a Stewart platform [51]. Liu measured the positioning accuracy of the device using an infrared reflective optical system. Compared to the programmed positions, the motion of the device was accurate to less than 0.04 mm, and less than 0.04° for rotations (RMSE per axis). The device's range of motion was ±20 mm for all horizontal translations, ±10 mm for vertical motions, and ±10° for all types of rotation.

While the research demonstrated by Wiersma, Belcher, and Liu show significant strides toward robotic motion compensation, each system developed also contained significant amounts of metal components within the treatment region, which could attenuate or scatter treatment fluence. Photon and electron attenuation and scatter could also impede the utility of x-ray based imaging systems important for accurate setup. Likewise, although robotic couches have been shown to achieve fast and precise compensation for respiratory motion, single point of interest correction cannot guarantee perfect setup for flexible bodies. Although the work described by Wiersma could be used to investigate this point, Wiersma's group has not performed any studies with internal imaging.

The work described in this dissertation consists of the development of a mostly-plastic robotic system which may be used for reducing positioning uncertainty in radiotherapy treatment of the head and neck. The robotic system is designed both for allowing accurate mechanical setup on the scale of residual setup errors reported in the literature, and for reducing the vector drift of target disease caused by intra-fraction patient motion. This research has focused on two major improvements over existing technologies. First, the system developed does not significantly interfere with x-ray-based imaging, nor with external beam delivery. Second, compared to traditional setup, this work investigates whether the

uncertainty present in the lower neck may be reduced by independently positioning the head from the body inferior to C7. This dissertation is organized into four chapters after this introduction. Chapter 2 provides a complete description of the development process of the robotic system, up to and including mechanical validation. Chapter 3 expands the robotic system described in Chapter 2, and shows the creation of a motion compensating control system deployable to the developed robotics. Topics will shift in Chapter 4, which contains descriptions of work done to create a novel radio-based position tracking system. Finally, the last chapter forms a conclusion to this work, and a brief glimpse into the potential for further work.

1.4 Specific aims

Specific Aim 1

To design, construct, and evaluate a radiotherapy-friendly robotic system for accurate positioning in head and neck radiotherapy

The development of this robotic system is multi-faceted. Even after selecting a prototypical geometry, many aspects remained before construction could occur. Presented are investigations into the kinematics, geometric optimization, stress and torque analysis, control systems, electronics, and feedback-loop elements required to build a robotic system out of radiotherapy-friendly materials.

The complete set of target engineering design specifications are as follows:

- Capability of motion in 6D
- Translation motion extending ± 2 cm
- Rotation motion extending $\pm 5^{\circ}$
- Mandatory motion accuracy of 0.5 mm and 0.5°

- Ideal motion accuracy of 0.1 mm and 0.1°
- Real-time control with response rate of 0.25 s
- No metal components within 30 cm radius from center of device
- As little material near the device as possible

Specific Aim 2

To determine whether the setup uncertainty in the lower neck may be reduced by independent positioning of the head relative to the body

This aim addresses the uncertainties described in section 1.1. As many studies have observed, the flexibility of the neck makes perfect alignment of the neck from C1-C7 very difficult, even with head-to-shoulder immobilization [8][9][12][52]. As observed by Djordjevic, the choice of landmark for registration has significant impact on the overall accuracy of the placement of the tumor, and as seen by Polat, no simple 3D correction can be used to completely align the spine [6]. Therefore, it is my hypothesis that this effect is due to the flexible nature of the spine, and that this effect may be reduced by providing accurate, independent placement of the head and of the body. To test this hypothesis, I created the following protocol for head-first supine patients:

1. Treatment Simulation
    (a) Position patient and 6D positioning system on simulation couch, with partial face immobilization affixed to robotic platform and arms at sides
    (b) With robot in rest position, simulate patient as normal
    (c) Identify treatment isocenter and designate with AP and RL fiducial markers on patient skin and on immobilization mask
    (d) Use surface imaging to obtain reference image

(e) Designate cranial region of interest on reference image, including bridge of nose

(f) Designate thoracic region of interest on reference image, including shoulders and excluding neck

2. Treatment

(a) Set up patient and 6D positioning on treatment couch in the same position as done in treatment simulation

(b) Using surface imaging, align thoracic region of interest with couch and manual adjustment

(c) Using surface imaging, align cranial region with 6D robotic system

(d) Use stereo kV imaging to align T1 bony anatomy in thorax

(e) Once thoracic anatomy is in proper position, use kV imaging to align cranial anatomy using 6D robotic system to match C2

3. Hypothesis Testing

(a) Offline, compare registration between simulation and treatment of multiple regions of interest including skull, C2, C5, T1. Registration errors between imaging sets for each region of interest are metrics of interest. The entire process should be repeated as a control, but omitting the steps involving any thoracic alignment. Hypothesis test will compare registration errors between control and test protocols.

Due to regulatory limitations, this test procedure has not been tested on live, human subjects. Instead, a comparable test is performed using an anthropomorphic skeletal phantom. This work differs from the work done by Graff, Djordjevic, and others by being the first to test whether independent alignment of multiple regions of interest can reduce the

positioning errors found in the lower neck. The 6D platform is necessary to allow for highly precise 6D positioning of the craninum without affecting the thoracic alignment.

Other aims

In addition to developing a robotic system for position correction, a novel, RF-based localization system is described in Chapter 4. The RF system was designed to overcome the limitations of optical tracking systems, including requirements of line-of-sight, better separation of regions of interest and generally improved specification of target regions of interest. Although this system was never physically constructed, much of the underlying mathematical theory was investigated, and is presented here as a potential future localization technology.

Chapter 2

Development of a Plastic Robot

Robotics have long been utilized in radiotherapy to provide fast and accurate positioning of treatment surfaces, linear accelerators, radioactive sources, or imaging equipment [53][54][53]. However, in all of these cases, radiation therapists still manually manipulated body segments of patients to reach accurate alignment with the treatment plan. A typical treatment process may include the iterative steps seen in the workflow featured in Figure 2.1. Manual manipulation is a time-consuming, iterative process. Human operators have limited abilities to recognize and manipulate small displacements in both translation and rotation space. This problem is exacerbated by the fact that human operators are also blind to patients' interior anatomy when using manual manipulation. Since transitions between the inside and outside of the treatment room can be the most time-consuming processes, robotic correction may provide a significant time-saving advantage by allowing therapists to remotely adjust a body segment (i.e., a patient's head) from outside the treatment room.

A robotic solution could reduce the setup uncertainty in the lower neck related to the flexibility of the spine through independent, mechanical positioning of the head separate from the body. This type of robotic correction has been explored before, but clinical implementation has been constrained by the inclusion of high-Z materials within the treatment field by the proposed systems. Therefore, a mostly non-metal robotic platform has been

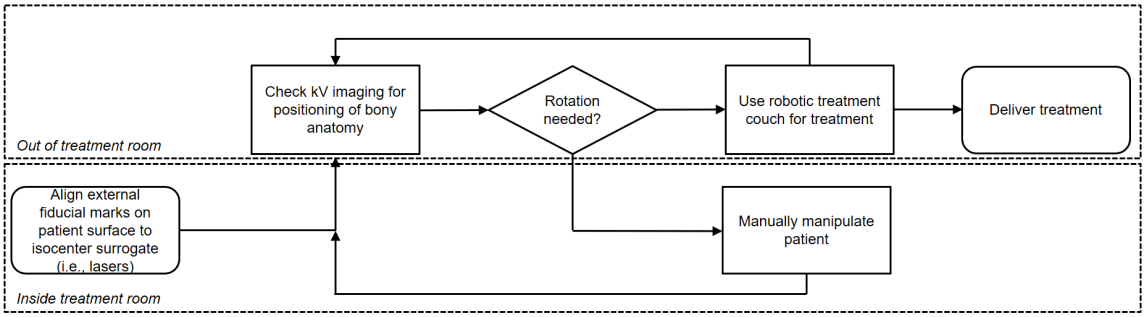


Figure 2.1: Typical workflow for setting up head and neck patients [12][8]

designed as a means of remotely and accurately positioning the head.

Three generations of prototypes were constructed. The first poof-of-concept prototype was designed by Mark Ostyn together with a team of senior engineering students (Thomas Dwyer, Ross Cruikshank, Melvin Rosario, and Daniel Martinez). The second prototype was designed by Mark Ostyn assisted by a team of junior engineering students (Rachel Sacks, Paden Kind, and Matt Miller) as a commercialized version. The last prototype was designed entirely by Mark Ostyn re-appropriating several core components from the second generation model to overcome major design failures in the second generation prototype.

2.1 The first generation prototype

The prototypical mechanical configuration capable of 6D motion was found by the first undergraduate engineering team. The system consists of a moving positioning plate connected by linkages to three pairs of parallel, linearly actuating sliding bases (often referred to as *sliders*). The 6D position of the plate is derived by the relative positions of the sliders. Together, the plate, linkages, sliders, and connecting components are often referred to as the *core positioning system* throughout this dissertation, due to their geometric importance. The design was selected because it allows 6D control with the capacity for reconfiguration to allow remote actuation with the powered elements could eventually be moved away from the irradiated region. This is a marjor improvement over previous works, because it

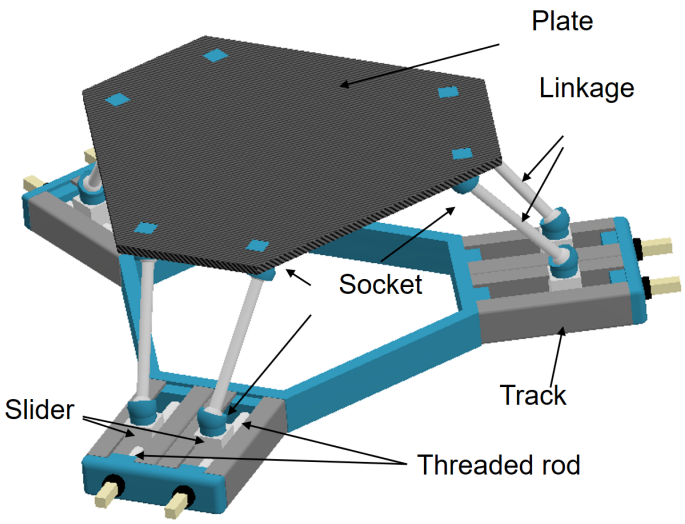


Figure 2.2: The first generation prototype

enables mechanical correction without overly constraining typical treatment processes.

2.1.1 Inverse kinematic analysis

The robotic system is designed to allow movement of the plate in six independent degrees of freedom: translation along and rotation about two orthogonal horizontal axes and one vertical axis. The axes were defined by the following: the y-axis lies in the inferior-to-superior direction and rotations about it are referred to as *roll*, the x-axis lies in the left-to-right direction and rotations about it are referred to as *pitch*, and the z-axis lies in the posterior-to-anterior direction and rotations about it are referred to as *yaw*, assuming that the patient is in head-first, supine position.

Inverse kinematic analysis was developed by Mark Ostyn and Thomas Dwyer. In inverse kinematics, the required set of positions of the actuating elements (the sliders) is calculated based on a specified position of the end effector (the plate). This contrasts to forward kinametics that would calculate the resultant position of the end effector based on the specified configurations of the articulating elements. Inverse kinematics are useful for situations where the effector is required to be positioned at a target location and the underlying kinematics are nonlinear (where there is no one-to-one correspondence between

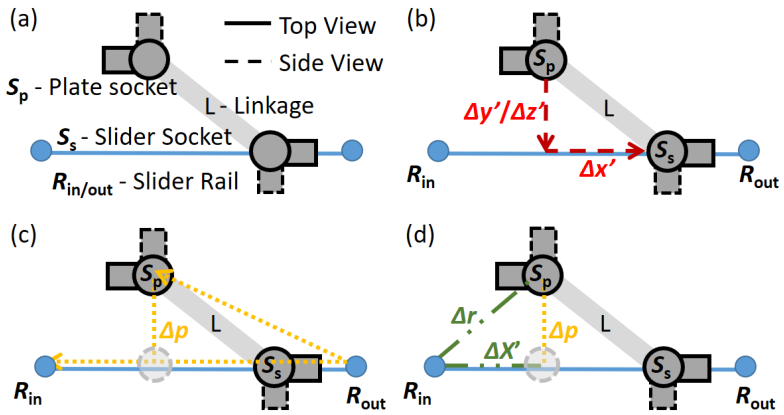


Figure 2.3: Geometric description of the inverse kinematic algorithm. Both side views and top views are shown. (a) Description of variables. (b) The length of the linkage is separated into orthogonal components in a single plane. (c) The cross product is taken between a direction vector of the path of motion for the lower joint and one of the components found in the previous step. (d) The lower joint's distance from a set reference point is calculated by Pythagorean's theorem.

changes in signal input and changes in mechanical output).

The calculation begins by defining the position of the plate in a reference coordinate system. The plate is specified by the fixed geometric relationship between the six attached socket elements. In the simplest operation, the origin is defined as the center point of the top of the plate, and the sockets are defined as a set of x, y, z coordinates each at point $\boldsymbol{S_{p,0,i}}$, each at a specified displacement from the reference origin (Figure 2.3). For convenience, each socket will be specified as $\boldsymbol{S_{p,0}}$, rather than including the index for each socket, because the process is identical between sockets. Thus, for a given transformation of a plate by a set of translations (x, y, z) and rotations (α, β, γ), the transformed position for each socket $\boldsymbol{S_p}$, is

$$\boldsymbol{S_p} = R_z(\gamma)R_y(\beta)R_x(\alpha)\boldsymbol{S_{p,0}} + \boldsymbol{T(x, y, z)} \tag{2.1}$$

where R_x, R_y, and R_z are standard rotation matrices[1]. Note that the choice of origin is the pivot point of the rotation coordinate system. Any arbitrary pivot point $\boldsymbol{P(x_p, y_p, z_p)}$

[1] An examination of rotation matricies is given in Appendix A.1.

may be selected by the following modification to the transformation equation.

$$\boldsymbol{S_p} = R_z(\gamma)R_y(\beta)R_x(\alpha)(\boldsymbol{S_{p,0}} + \boldsymbol{P(x_p, y_p, z_p)}) + \boldsymbol{T(x, y, z)} - \boldsymbol{P(x_p, y_p, z_p)} \quad (2.2)$$

The length of the linkage connecting the sockets on the plate and the sockets in the sliders is of known length, but is also defined by a composition of component vectors in an arbitrary reference frame

$$\|\boldsymbol{S_p} - \boldsymbol{S_s}\| = L = \sqrt{\Delta \boldsymbol{x'}^2 + \Delta \boldsymbol{y'}^2 + \Delta \boldsymbol{z'}^2} \quad (2.3)$$

where $\boldsymbol{S_s}$ is the set 3D coordinates (x, y, z) for each of the six sockets attached to sliders. By construction, the vector along the x' direction represents the projected displacement between the plate and slider sockets along the slider's path of motion on its rail, while the vectors in the y' and z' directions represent the projected displacements in the orthogonal horizontal and the vertical directions, respectively. The $\Delta \boldsymbol{y'}$ and $\Delta \boldsymbol{z'}$ vectors may then be combined into a single vector that represents the projected displacement between the sockets in the plane perpendicular to the direction of the slider's rail, and may be represented by

$$\Delta \boldsymbol{p}^2 = \Delta \boldsymbol{y'}^2 + \Delta \boldsymbol{z'}^2 \quad (2.4)$$

Let $\boldsymbol{R_{out}}$ represent an arbitrary reference point on the rail distal to the slider, and $\boldsymbol{R_{in}}$ represent a similar reference point on the rail medial to the slider. The vector difference between these two points represents the line on which the slider may travel, or its "rail vector". The magnitude of the perpendicular vector,$\boldsymbol{\Delta p}$ may be found through the directed area product of two known vectors: the rail vector, and the difference vector between a known reference point on the slider's rail, $\boldsymbol{R_{out}}$, and the corresponding plate socket, $\boldsymbol{S_p}$,

both of which are specified in the initial geometry

$$\|\Delta \boldsymbol{p}^2\| = \frac{\|(\boldsymbol{R_{out}} - \boldsymbol{R_{in}}) \times (\boldsymbol{S_p} - \boldsymbol{R_{out}})\|}{\|(\boldsymbol{R_{out}} - \boldsymbol{R_{in}})\|} \tag{2.5}$$

From this construction, the projected displacement along the rail between the sockets, $\Delta \boldsymbol{x'}$, may be found by rearranging equation 2.3. The geometric relationship between the rail and the corresponding plate socket is specified by

$$\Delta \boldsymbol{r} = \boldsymbol{S_p} - \boldsymbol{R_{in}} \tag{2.6}$$

Using the magnitude $\Delta \boldsymbol{r}$, the distance between the inner reference point and the start of the original $\Delta \boldsymbol{x'}$ may be calculated using the Pythagorean Theorem

$$\|\Delta X'\| = \sqrt{\Delta \boldsymbol{r^2} - \|\Delta \boldsymbol{p}^2\|} \tag{2.7}$$

Therefore, the required position of the slider, P, along the rail for a given transformation of the plate is then found by the addition of these distances

$$P(x, y, z, \alpha, \beta, \gamma) = \Delta X' - \|\Delta \boldsymbol{x'}\| \tag{2.8}$$

Note that because this math is separable to each socket-linkage pair, it is scalable to any configuration of platforms, sliders, and linkages, so long as the number of degrees of freedom in the sliders matches the necessary number of degrees of freedom of the end effector. (i.e., this math may be utilized to calculate the kinematics for a 3D positioning platform using three sliders and three linkages).

Table 2.1: Mechanical properties of used materials

Material Name	ρ (g/cm^3)	Modulus of Elasticity (GPa)	Yield Strength (MPa)
PLA	1.25	1.280	70.0
ABS	1.05	2.240	20.0
Nylon	1.13	2.930	82.7
CFRP	1.44	89.14	300.0
POM	1.41	2.900	68.2
PMMA	1.19	2.740	48.9
PTFE	2.20	0.500	23.0
PVC	1.41	3.400	46.5
Polycarbonate	1.13	2.300	62.0

2.1.2 Stress analysis

The structural safety of the prototype was estimated using computational modeling using a commercial software package (Inventor, Autodesk, San Rafael, CA) to conduct structural finite element analysis on the CAD model of the designed system. The loading effect was primarily investigated from a estimate weight of a typical patient's head, using a maximum simulated loading of 200 N applied directly onto the plate. Note that the maximum load is about three times the average weight of a human head to account for a factor of safety. Fixed, rigid constraints were placed on the bottom surfaces of the slider channels. The device was simulated in two positions, one with all sliders at their most proximal positions to the center of the plate, and one with all sliders at their most distant settings, 5 cm away. The von Mises stress was calculated to compare with the yield strength of the materials. The displacement from deformation was also calculated to investigate material deformation's effect on positioning accuracy. Table 2.1 summarizes the mechanical properties of selected device components.

Figure 2.4 summarizes the finite element analysis (FEA) results of the patient positioning mechanical system. The highest loading condition, 200 N of force, was applied directly to the top plate to calculate the maximum associated stress on the structure. The expected applied stress on the plate and linkages is only about 1 MPa (Figures 2.4a and

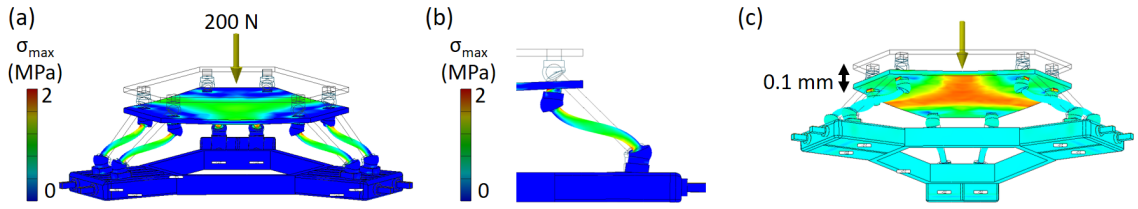


Figure 2.4: (a) Finite element structural analysis demonstrating von Mises stress of the device under load of 200 N. The maximum stress of 2.0 MPa was below the yield stress of 40 MPa for the material used. (b) Side view of Von Mises stress in linkage bar under load. (c) First principle stress in the device under load.

2.4b), however the maximum stress (approximately 2 MPa) occurs at the joints between the sockets and the plate, and was well within material limits. Even though the exaggerated loading condition may cause the plate to sag by about 0.1 mm (Figure 2.4c), the stress does not make any permanent deformation of the system considering the allowable yield strength of the material (ABS plastic) of 40 MPa. While the estimated deformation at the joints contributes toward the upper bound of the motion accuracy, the magnitude of sag is at the lower limit of physical detection, indicating that the sag would not strongly affect any possible dose distribution.

The stress limit was selected as such to reduce the risk of catastrophic structural failure. The deformation limit was selected so that deformation under load did not become a major component of the positioning uncertainty. The strain limit was selected both to maximize the mechanical accuracy as well as to reduce the risk of radiation-induced failure. A study from Hassan showed that the radiation exposure to ABS plastics lowers the maximum elongation before fracture [55]. An exponential fit of the data provided by Hassan shows the approximate relationship:

$$E = 10.1\mathrm{e}^{-0.012D} \tag{2.9}$$

Where E is the percent elongation at breaking and D is the dose received in kGy for a pure ABS sample. Since the linkage components are likely to receive upwards of 12.5 kGy per year (2 Gy per fraction for 25 fractions per day for 250 days per year), the fracture strain is estimated to reach the maximum strain in this prototype after approximately 300 years,

indicating long term stability.

2.1.3 Construction

The first-generation system was constructed from a combination of off-the-shelf components and 3D printing. Components which were 3D printed included the sliders, ball and socket joints, slider endcaps, central frame, linkages, and rod endcaps. Threaded nylon rods were purchased off-the-shelf, as were extruded polyvinylchloride tracks. Each slider track was held in place with a plastic zip-tie. The main actuating end was a 0.25-in thick hexagonal plate of cut carbon fiber-reinforced polymer (CFRP). The dimensions of some components were selected arbitrarily, though the length of the linkages were determined by a simulation process carried out by Thomas Dwyer. The inverse kinematic algorithm was applied to a list of points taken from an approximate Wiener process in 3D as a surrogate for patient motion[2]. The length of the linkages was selected by whatever linkage length provided the most geometrically achievable positions from the list, to the nearest 10 mm.

2.1.4 Step calculation algorithm

The first generation of control software was written by Thomas Dwyer in C for an Arduino-based control board, and the software was later revised into LabVIEW by Mark Ostyn. The step calculation algorithm is used to convert continuous motion vectors of the positioning platform into discrete pulse trains for the driving stepper motors. The algorithm operates by first discretizing a planned 6D continuous motion vector into positioning checkpoints. The number of checkpoints is selected by finding the smallest possible divisor n such that no single displacement between checkpoints on any axis results in vector segments longer than the step length (the distance a slider would move with one step of the stepper motor). The inverse kinematic algorithm is then applied to every positioning checkpoint,

[2]More on Wiener processes in Chapter 3

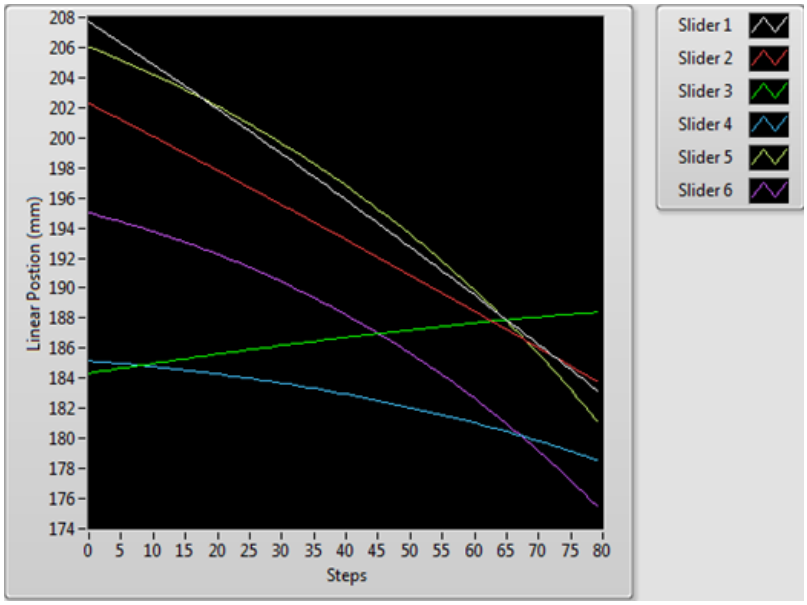


Figure 2.5: Sample slider trajectories for 6D linear motion

giving a table of the necessary slider positions for every checkpoint (Figure 2.5). Next, each column of the slider position table is converted into discrete step commands for the stepper motors, found by the algorithm shown in Figure 2.6

Because the step size is discrete, some small continuous motions cannot be realized into actual motions, but still need to be considered for accurate positioning of the sliders. The stepping algorithm helps determine when to take a step based on cumulations of small displacements. Functionally, this algorithm stores variables of cumulative motion for each slider. Once the absolute value of the cumulative motion is greater than the threshold step size (determined by the resolution of the stepper motor, the radius or pitch of the force applicator, and the net gear ratio), a step command is issued, along with a corresponding command of direction. The relevant step sizes for the first and third generation prototypes are given in equations 2.10 and 2.11.

The resultant table of step commands is then filtered for any instances where no steps are taken by any motor to increase the output speed of the platform. The finalized table of step commands is then sent row by row to the stepper motor controller with a delay of t milliseconds between commands. Operation without the delay can either overflow

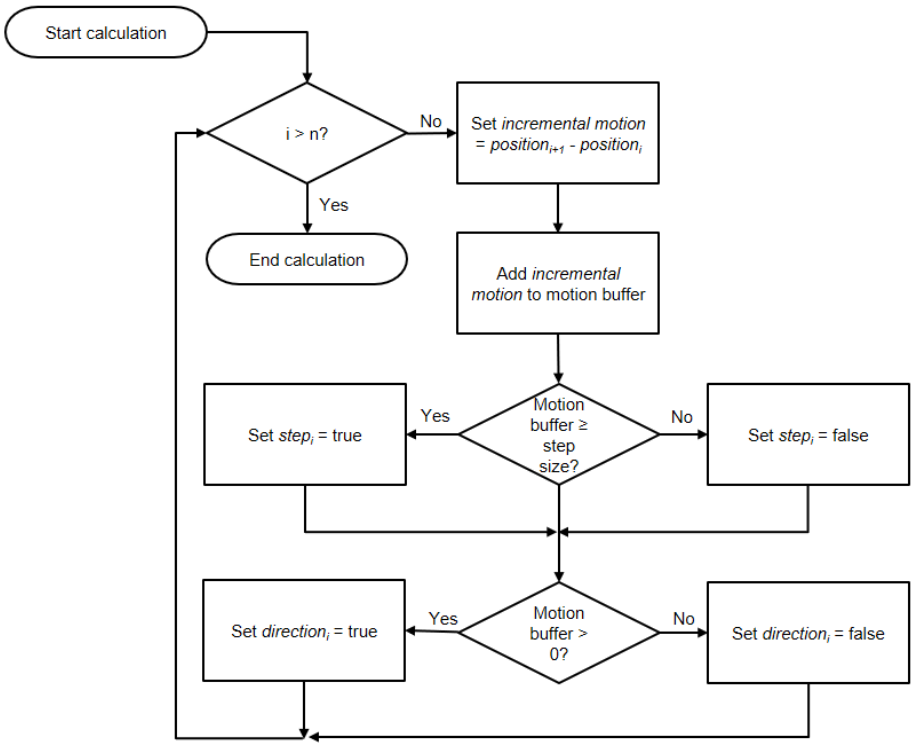


Figure 2.6: Flowchart of stepping algorithm

the stepper motor controller buffer (resulting in inaccurate positioning) or can overheat the stepper motor driver chips (resulting in their terminal failure). The value of t was selected based on the specifications of the motion controller, such as the National Instruments myRIO 1900 used as with Pololu A4988 stepper motor driver chips in the first generation prototype, or Pololu DRV8825 stepper motor driver chips used in the third generation prototype. Although the motors were placed at less-than-convenient locations[3] in this generation of prototype, the electronic control hardware was always kept at least 50 cm away from the center of the plate. Later iterations of the system would focus on creating solutions to keep all metal components out of the treatment field.

$$\text{Step size}_{\text{G1}} = \frac{1 \text{ revolution}}{200 \text{ steps}} \times \frac{1.25\text{mm thread}}{\text{revolution}} = 6.25 \times 10^{-3} \text{ mm/step} \tag{2.10}$$

[3]They extened over the sides of the treatment couch, and were 27 mm away from the center of the plate in the horizontal plane

$$\text{Step size}_{\text{G3}} = \frac{1 \text{ rev}}{200 \text{ steps}} \times \frac{0.25\text{in sprocket pitch}}{0.1 \text{ rev}} \times 10 \times \frac{25.4\text{mm}}{1\text{in}} = \frac{127}{1350} = 9.\overline{407} \times 10^{-2}\text{mm/step} \tag{2.11}$$

Initially, the platform trajectories were limited to linear paths, but the algorithm is adaptable to any given trajectory path, even following a parametric path with respect to time. A 20-millisecond delay between leading pulse edges is highly recommended when functioning in this time-parameterized capacity, which is the fastest that the motors can respond with an even integer divisor into seconds (e.g., a 20-millisecond delay between leading edges corresponds to exactly 50 Hz refresh rate).

2.1.5 Radiographic imaging compatibility

The imaging and treatment compatibility of the prototype was tested by measuring the attenuation of the major components using a Philips Brilliance computed tomography (CT) scanner configured for head and neck scan profile settings. The average Hounsfield Unit (HU) of each major component was measured along with the HU noise. Cortical bone was used as a control for comparison of HU values, since cortical bone does not greatly degrade x-ray or CT imaging quality compared to metal materials. Metal components were excluded from this study.

CT measurements of HU demonstrate the compatibility of the device within x-ray-based imaging and linear accelerator output within a radiotherapy environment. The mean measured HU of most components fell between 1050-1150 HU, which is between typical values of cancellous and cortical bone, positively indicates the potential use of these materials in clinical radiotherapy settings. The noticeable outliers included the linkages, which had a measured value of 384 ± 195 HU, and the tracks, which had a measured value of 1867 ± 33 HU. The low attenuation and large variance of the values of the linkages are easily explained by the low 3D printing infill, so much of the component is air by volume. Conversely, the relatively high attenuation of the track is a result of the chemical composition of polyvinylchloride, which is 53% chlorine by mass and therefore is a much stronger

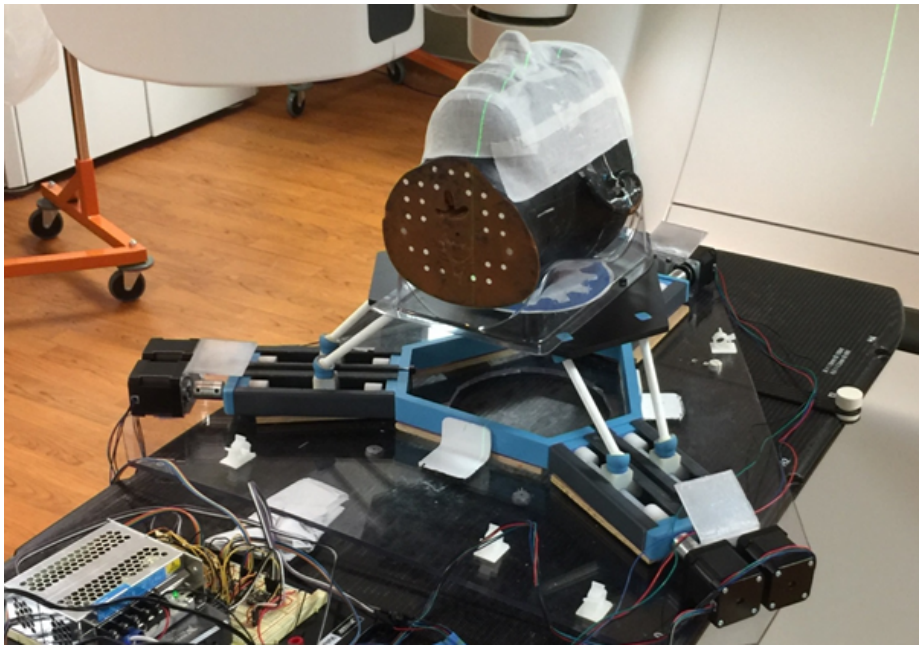


Figure 2.7: Setup of accuracy test for first generation prototype

attenuator than carbon or hydrogen at diagnostic energies, which are more strongly present in the other materials. Nevertheless, the measured attenuation of the tracks is still on the low end of typical HU values reported for cortical bone at soft energies. Note that since the attenuation values of all materials were measured as similar to or less than that of cortical bone at diagnostic energies, they are unlikely to interfere with the relatively more penetrating therapeutic megavoltage photon energies.

2.1.6 Positioning accuracy and analysis

Preliminary measurements of the 6D positioning accuracy of the first-generation system was tested by using the platform to correct manual displacements of the head potion of an anthropomorphic phantom (RANDO phantom, Supertech Inc, Elkhart, IN), as measured with an Align RT camera system (Vision RT Ltd., London, UK), as seen in Figure 2.7. In this process, an anthropomorphic phantom was set into a standard headrest fixed to the moving positioning plate, and manually aligned to a reference 6D configuration using the optical surface tracking system.

Table 2.2: Summary of displacements for accuracy test of first generation prototype

Direction	Mean absolute positioning error (mm/°)
Longitudinal	1.4 ± 1.0
Lateral	2.6 ± 1.3
Vertical	0.5 ± 0.3
3D Vector	3.2 ± 1.4
Pitch	0.7 ± 0.5
Roll	1.5 ± 0.9
Yaw	1.8 ± 1.0

Once in position, a new reference image was taken to set the reference coordinates for subsequent measurements. The phantom was then shifted by hand within the headrest to a new, displaced orientation. Negated values of the 3D rotation coordinates were then input to the control electronics of the positioning system to move the phantom back into the same angular orientation as the initial reference position. Since the rotation pivot point of the prototype device and the isocenter of the camera system were not the same in this experiment, a displacement vector in translation space develops, the negative of which is taken as the necessary translation correction vector. The absolute values of the positioning measurement from the optical system are taken as a measure of the positioning accuracy of the system. This process was repeated for 15 positions, moving back to the rest position after the residual 6D displacements of the translation correction were measured, and resetting the reference frame with each iteration. A summary of the rotation and translation displacements is provided in Table 2.2. The robotics were not recalibrated during this process.

The mechanical system was fixed to a typical treatment couch using a Bionix SecureFit bar (Bionix Radiation Therapy, Toledo, OH). The headrest was attached to the plate of the positioning system by plastic pins through the standard indexing holes. The resting state of the device was defined as a specific configuration in which all linear sliders are the same, specific calibration distance from the central frame of the mechanical positioning system, as measured with digital calipers. The initial angular orientation of the positioning

Table 2.3: Positioning errors of first generation prototype

Direction	Mean absolute positioning error (mm/°)
Longitudinal	0.3 ± 0.2
Lateral	0.3 ± 0.2
Vertical	0.1 ± 0.1
3D Vector	0.4 ± 0.2
Pitch	0.1 ± 0.1
Roll	0.3 ± 0.4
Yaw	0.3 ± 0.4

platform was aligned with the axes of measurement of the AlignRT system within $\pm 0.1^\circ$, as measured using a digital level.

Table 2.3 summarizes the results from the positioning accuracy testing, and Figure 2.8 provides a statistical view. Over the course of all movements, the mean absolute error (MAE) of 3D positioning was 0.4 ± 0.2 mm, and was less than 1 mm in all positions tested, with an average vector displacement of 3.2 ± 1.4 mm. A single-tailed t-test indicates that the real mean positioning error was less than 1 mm with p-value > 0.001. The MAE in rotations was also low. For each axis, the MAE was $0.3^\circ \pm 0.4^\circ$ for roll, $0.1^\circ \pm 0.1^\circ$ for pitch, and $0.3^\circ \pm 0.4^\circ$ for yaw. The rotation error for each axis was less than or equal to 0.5° in 80% of all positions, and less than 1° for 90% of all positions.

2.1.7 Conclusions about the first generation prototype

Some issues with this device were targeted for specific areas of improvement in later prototypes. First, the length of track for each slider was greater than what could conceivably fit on a treatment couch. This would give rise to a more comprehensive geometric analysis and optimization procedure found used in the design process for the second and third generation systems. Second, as a proof-of-concept device, the prominent high-Z materials were not completely out of the radiation field. Lastly, although the high resolution of the threaded rod is ideal for high-accuracy positioning, increased resolution also leads to decreased speed, making the speed of the positioning plate approximately 10 mm/min,

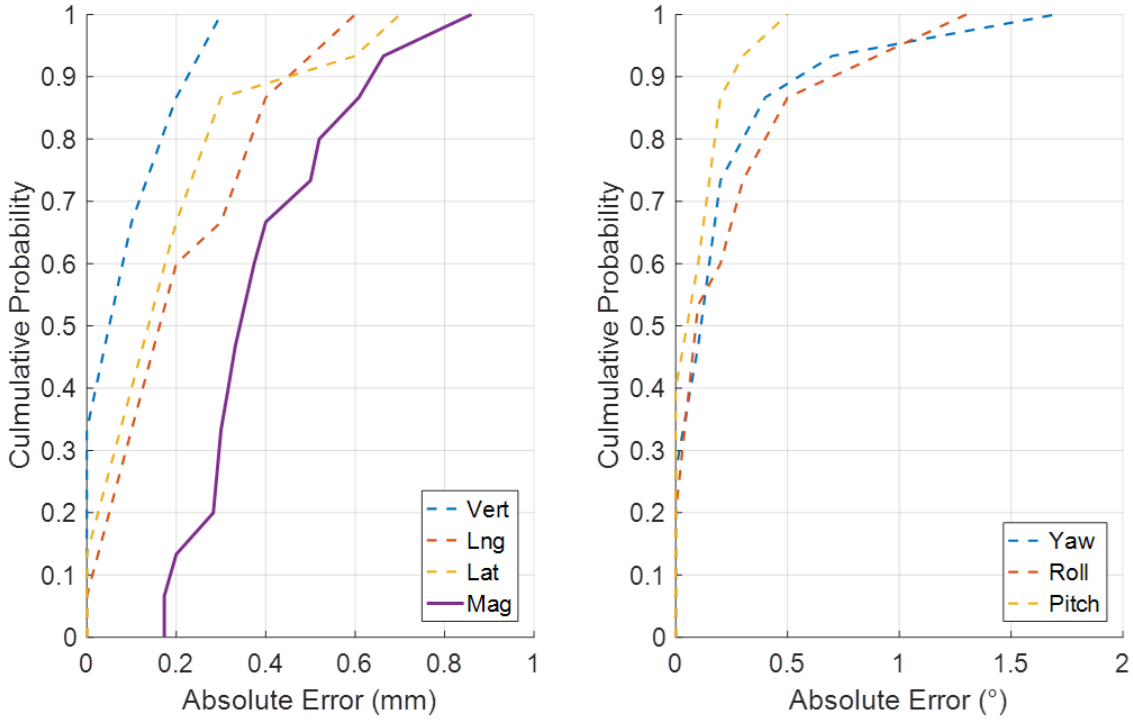


Figure 2.8: Cumulative histogram of error in accuracy test of first generation prototype

making for very slow clinical operations.

This prototype was useful to demonstrate the proof-of-concept for testing the electronic components and corresponding code. After significant testing, functional code and electronics were generalizable to future versions of the system. Additionally, this prototype also was useful for establishing methodologies of testing the accuracy of future prototypes. Although the physical components in later versions differed significantly, the most valuable experience with the first generation prototype was gaining familiarity with the most common points of failure in a plastic robotic system, so that issues in later systems could be diagnosed more quickly.

2.2 The second generation prototype

After construction of the first-generation prototype was completed, a second-generation prototype was developed for commercialization. The primary goals of this device were

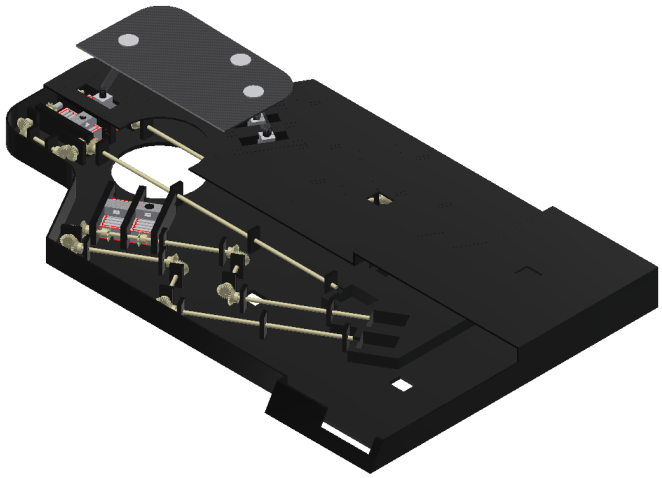


Figure 2.9: Cutaway view of the designed second generation prototype

to overcome the limitations of the first prototype, especially moving the high-Z materials out of the region of end effector and to consolidate the package into a convenient form factor for commercialization. Early in this stage of development, it became clear that most feasible location for the high-Z components (stepper motors and electronics) was placement beneath the patient in the caudal direction relative to the positioning platform. The decision of the development team was to create an all-plastic gearbox under the patient to transmit rotational energy to the sliders. This gearbox would use a series of non-metal gears and shafts to transmit rotational energy from the motors to the sliders. The goal was to keep the all metal components 40 cm away from the positioning plate origin, or approximately the distance from the sphenoid to the sternum in a large patient. The patient would be supported over the gearbox by a sturdy support system, often referred to as a *patient standoff*.

The layout for the gearbox was designed by Paden King (Figure 2.10), using schematics for off-the-shelf 90° bevel gears as a means of changing the direction of the rotational energy. The robotic system also underwent significant changes to the core positioning system in this generation. Rather than relying on rotation of a threaded rod to move the sliders, they would instead be moved by a roller chain and sprockets. This design change was in response to one source of uncertainty seen in the first generation prototype, where

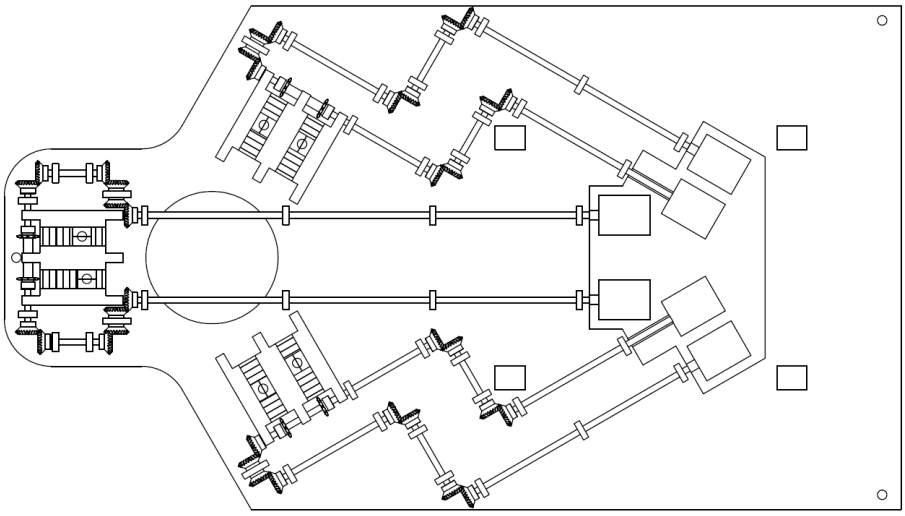


Figure 2.10: Layout gearbox of the second generation prototype

the sliders tended to pivot about the threaded rod when under load. This new design introduced new geometric constraints on the system. A 7-cm-radius exclusion region was designated at the origin of the horizontal reference frame where placement of most components was forbidden. This was done to minimize the the system's appearance on AP x-ray imaging. Additionally, to keep the system in a clinically-feasible size, components were restricted to stay within a 10.5-in (26.7-cm) boundary in the ML direction (measurements of a simulation couch from a Brilliance CT scanner was measured as 21-in wide).

2.2.1 Geometric optimization of the second generation prototype

A primitive geometric optimization sequence was devised to find the geometric relationships that granted the largest range of motion while staying within the necessary geometric constraints. The optimization process used the inverse kinematic analysis to simulate the mechanical system at various configurations to search for mechanically possible dimensional parameters. In this context, a *configuration* refers to the both the specified 6D plate position going into the inverse kinematic analysis and the corresponding 3D positions of the set of sliders. The dimensional or geometric *parameters* refers to the specific spacing between mechanical joints of the system, shown in Figures 2.11b and 2.11c. These

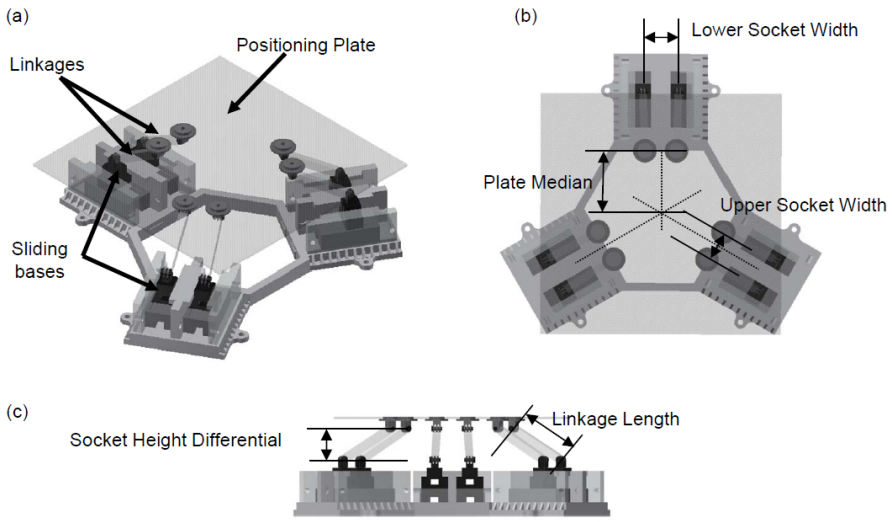


Figure 2.11: Major geometric components of core positioning system

parameters include the lengths of the linkages, the spacing of the sockets on the plate (determined by each pair's distance from the center of the plate and the horizontal distance from a median line), and the spacing between the horizontal tracks for the sliders. Constraints were also placed on the simulation, limiting how far any slider may be allowed to move inward along their tracks and how far any slider may move in the lateral direction.

Exhaustively examining every possible configuration of the device is a task much too large to perform realistically in simulation, even with a relatively small required range of motion. On an Intel Core2 Quad CPU computer (2.67 GHz), approximately 480 configurations may be tested per second. If the required range of motion was $\pm$20.0 mm along each translation axis and $\pm$5.0° of rotation freedom about each rotation axis with a spatial resolution of $\pm$ 0.1mm/0.1°. Checking every possible ($(401)^3(101)^3 \approx 6.6\times10^{13}$) configurations would require 1.4×10^{11} seconds (≈4400 years). Even with a constrained range of motion of $\pm$10.0 mm along each translation axis and $\pm$3.0° of rotation freedom along each rotation axis, there are still $(201)^3(61)^3 \approx 1.8\times10^{12}$ possible configurations, which would require approximately 3.8×10^{9} seconds (≈20 years) to exhaustively confirm the mechanical feasibility of each position. Clearly, this exhaustive approach is not possible.

Means of narrowing the search space were developed by simulating only the extreme

possible configurations of the platform in 6D space. An extreme position is defined any permutation of coordinate positions for which the platform was positioned at either end of its respective range of motion or in the center for each of the 6 axes, (i.e., (0,0,0,0,0,-3); (0,0,0,0,0,3); (10,10,10,3,3,3); (-10,-10,-10,-3,-3,-3); etc...). All three types of positions were included to ensure that all possible permutations of moving along or about single axes were included. This results in 3^6 = 729 possible configurations, which together could be simulated in only 1.5 s.

The most primitive method of searching for feasible geometry was to intuitively guess the dimensions of the input geometric parameters. In this methodology, the platform was simulated at all 729 possible configurations using the guessed parameters and the required positions of the sliders were compared to a series of constraints including a 15.0-cm limit of inward motion and a 19.0-cm limit of lateral travel from the sagittal mid-plane axis. If 99% of the configurations tested resulted in real (as opposed to imaginary) positions for all sliders on their specified tracks, and if the sliders were within set geometric constraints, the geometric parameters were accepted as mechanically feasible for the device. From this solutions were found that allowed a maximum range of motion of $\pm$10.0 mm along each translation axis and $\pm 3.0^\circ$ of rotation freedom about each rotation axis. The plate median set to 120 mm, the linkages set to 60 mm in length, the horizontal separation between upper sockets as 50 mm, and the resting height between socket sets as 40 mm. Note that these are all round numbers. This is a deliberate choice to simplify verification in manufacturing. The processes of confirming dimensions is substantially easier if the dimensions are near regular measurement increments.

2.2.2 Patient body support

The body support platform was constructed using bulk CFRP. The legs were made from 0.125-in thick strips of CFRP held together by epoxy while the platform was a solid slab of 0.25-in thick CFRP. The initial plan was to attach the legs to the plate by epoxying both

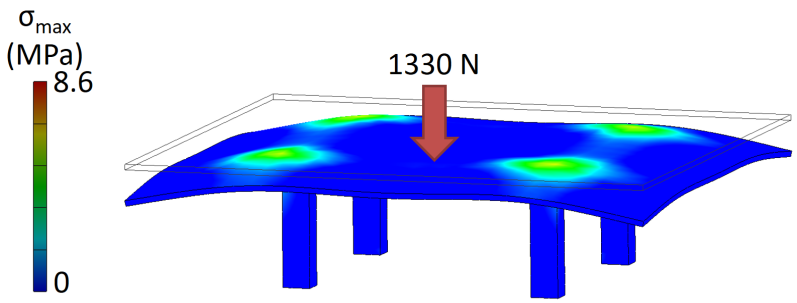


Figure 2.12: Stress analysis of the patient body support

components to a plastic hinge to allow collapsibility, but the epoxy failed to reliably hold the hinges onto the plate. The system was redesigned to use 3D printed plastic rectangular sockets bolted to the plate with nylon fasteners to hold the legs in place.

Structural integrity of the assembly was examined in finite element analysis using a loading of 300 lbs (1330 N). Examination of the expected von Mises stress revealed very little concern for breaking under load.

2.2.3 Construction of the second generation prototype

In the design phase, most core positioning components were identified as good candidates for 3D printing, while also purchasing the roller chains, sprockets, gears, and shafts off-the-shelf. The major 3D-printed components included new sliders, tracks for the sliders, pins to hold the sliders onto the roller chains, mounts for the motors, and a new socket and linkage assembly. Most 3D-printed components were printed with a Formlabs (Formlabs, Somerville, MA) Form 2 stereolithography printer (GPBK02, GPW02,GPCL02 resin), with notable exceptions of the mounts for the motors, which was printed with a MakerBot Replicator (fifth generation).

To simplify construction and cut costs, many structural components were designed to be constructed from laser-cut sheets of bulk acrylonitrile butadiene styrene (ABS) plastic. These components included the mounting plate and mounts for power-transmission CFRP shafts. The design of the shaft mounts was a simple rectangle with a pass through hole

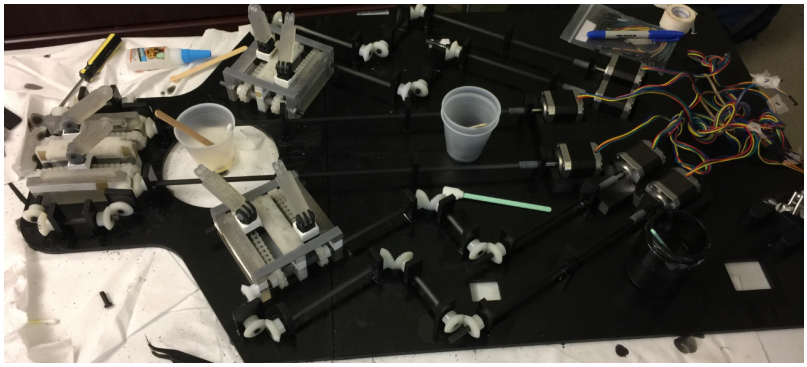


Figure 2.13: The assembled second generation prototype

that the shafts would be passed through before gluing the off-the-shelf bevel gears onto the ends. The shaft mounts would be inserted into laser-cut sockets on the mounting plate and welded in place with acetone. Since plastic welding works best for like materials, components made of disparate materials, such as the track walls and motor mounts, would be attached to the mounting plate by a strong adhesive.

Theoretically, since the laser-cut holes would match the dimensions of the shaft mounts, perfect alignment of the shafts was guaranteed. However, the cutting process was not fully explored, and disparities existed between the physically cut components and the planned components in the schematics; the sockets for the shaft mounts were larger than the base of the shaft mounts by approximately 1 mm, leaving significant clearance. This led to uncertainty in the proper placement of the power transmission shafts, creating significant lashing in the gears, with some gears failing to mesh entirely. The problem of uncertainty became compounded by the fact that adhesives were used to attach the gears onto the shafts: any assembly was irreversible. Several other issues also presented in the construction process. Inadequte mounting was provided for the motors, inadequate clearance was provided for their end effector, and inadequate angular tolerance was provided for attaching the motors to the drive rods. Later analysis also revealed other errors in the design of the plate. These factors led to the decision to cease development on this generation prototype, and instead develop a final, third-generation prototype.

2.2.4 Conclusions about the second generation prototype

This prototype was constructed with the philosophy of keeping costs low. While noble, this was misguided; the cheaper construction methods turned out to be more expensive in the long run when nearly the entire assembly needed to be re-manufactured from the ground up. However, the experience was fundamental in highlighting the principle design challenge when constructing with plastics: the need to account for uncertainty in the manufacturing process. Plastics are very difficult to manufacture with the same precision as metal components, often varying by 0.1-0.5 mm along straight edges depending on the manufacturing process. While nearly impossible to eliminate completely, clever design can mitigate the issues caused by the uncertainty.

2.3 The third generation prototype

The third generation of prototypes was designed to be durable in construction when using imprecise construction techniques, and to allow each assembly step to be reversible until the mechanics were fully verified. Reversible assembly was achieved by designing components to articulate with fasteners rather than welding or gluing. The re-engineering process also allowed for improvements to some underlying systems, such as making improvements to the geometric optimization process and improving slider positioning accuracy by introducing a feedback loop as opposed to relying on dead reckoning.

2.3.1 Geometric optimization

The decision to construct a third-generation prototype presented an opportunity to revisit the issue of geometric optimization to ensure that the optimal dimensions for the plate were being considered. A search program was designed to autonomously find acceptable combinations of dimension parameters. As discussed in early sections, the geometric

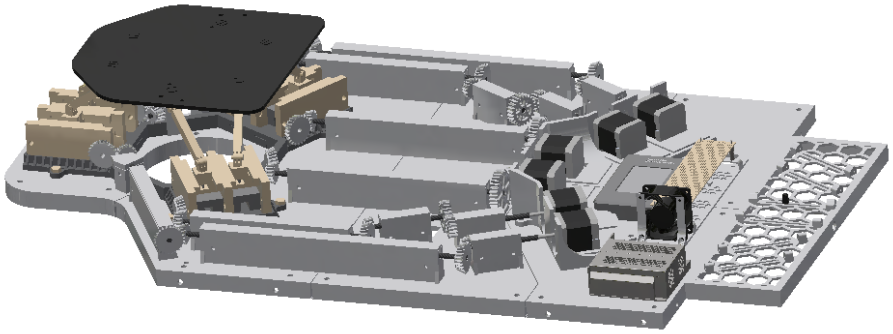


Figure 2.14: The third-generation prototype (without enclosure)

search processes relied on specifying the limitations and desired range of motion as inputs along with the best-guesses for feasible geometric dimensions, and the algorithms returned a pass rate for the guessed dimensions achieving the limits of the specified range of motion within the given restriction set. The process relied heavily on intuition and manual guesswork, and viable geometric sets became increasingly sparse with larger ranges of desired motion. Due to these factors, one could never be sure that the passing geometry was achieving the maximum range of motion. In addition, parameters were often guessed in convenient increments to keep the size of the search space manageable[4]. This intuitive approach is problematic because combinations of mechanically feasible parameters may not exist on such convenient size intervals, and the probability of manually finding feasible geometry for large ranges of motion is very low. Therefore, the major goals in creating a robust search algorithm was to increase the resolution of the search space while also decreasing the amount of time required to perform a search.

The first revision to the search program was to autonomously and incrementally step through all possible combinations of various dimensions for the four independent variables (linkage length, plate median, separation of socket pairs, and resting height). While this approach was better than manually guessing, it quickly encountered the problem of multivariate complexity. Since each of the 729 geometric configurations must be simulated for

[4]Creating components in convenient increments also simplifies the process of confirming manufacturing fidelity

each combination of values for the four independent variables . In example, if each of the four variables has a possible search space of 20 possible values, (such as searching with a 10-cm range in 5-mm increments), this leads to $729 \times 20^4 \approx 1.16 \times 10^8$ configurations to test, which would take approximately 67 hours to simulate, for a specified range of motion. After running several searches in this way, it became clear that nearly all tested combinations of geometric parameters were not mechanically feasible.

However, the combination of this property and the property that fewer geometrically feasible geometries existed for increased ranges of motion leads to the conclusion that only a small number of positions need to be tested before attempting the entire set of extreme configurations. This was implemented by adding a preliminary failure-check filter to the start of the search process that selects n 6D configurations at random from within the specified range of motion for the program to simulate the required slider positions. Because most combinations of geometric parameters are not mechanically possible, the probability is high that these few scout configurations will also fail, unless the tested geometric parameters happens to be a valid setting. Conversely, if the tested parameter combination was feasible for these arbitrary scout configurations, the probability was much higher that the extreme configurations point would also pass. This meant that it was possible to test n such that $n << 729$.

The most significant factors that affect the program runtime are the choice of n, the size of the increments being stepped through, and the range of dimensions being checked. Because there are four independent variables, increasing the resolution by a factor of two increases the runtime by a factor of sixteen. Alternatively, decreasing the search range by a factor of two decreases the runtime by a factor of sixteen. The impact on the choice of n is less intuitive. Choosing a small number of scout configurations does decrease the runtime for the scout program, but also results in a higher rate of false positives that must also be tested in the full extreme configuration check.

If 100% of the scout configurations are deemed as mechanically feasible, the com-

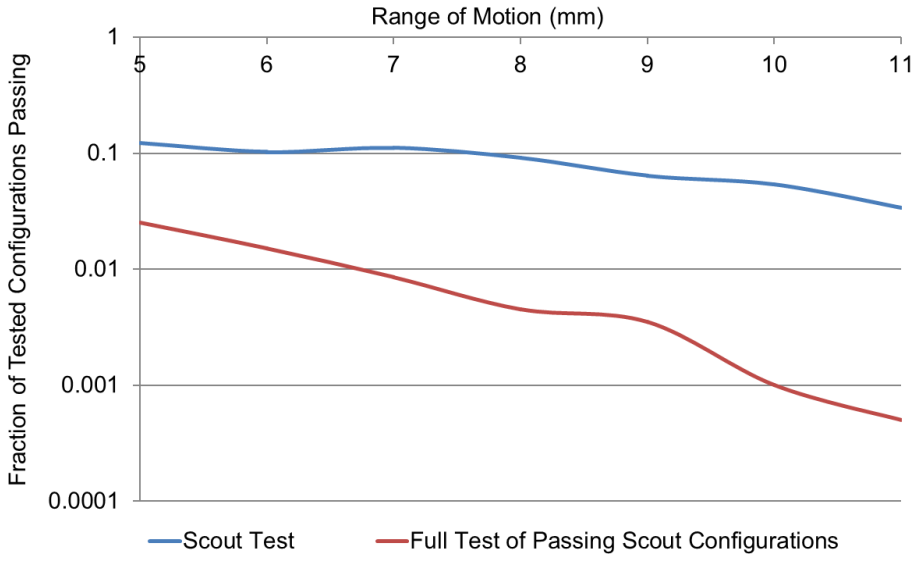


Figure 2.15: Relationship between geometry pass rate and range of motion

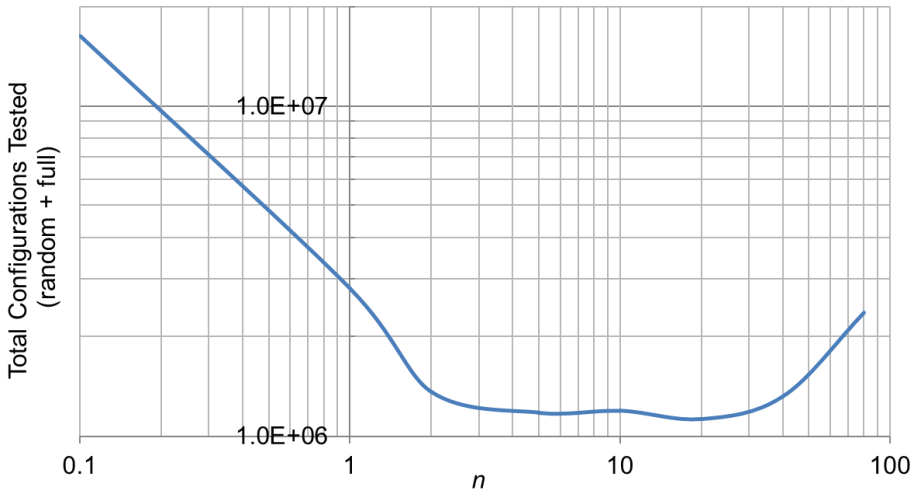


Figure 2.16: Relationship between total configurations tested and number of scout points. Note that the value for n = 0.1 is really 0, but included to show power of scout function.

Table 2.4: Output of optimization program showing valid values for the selected geometric parameters

Upper Socket Half Width (mm)	Plate Median (mm)	Linkage Length (mm)	Socket Height Differential (mm)
20	85	80	43
20	85	80	44
20	85	80	45
20	85	80	46
20	85	80	47

bination of parameters is passed along for a more thorough inspection of the extreme configurations. This filter drastically decreased the total runtime of the search program. The passing parameters for a 1-mm increement search are seen in Table 2.4, and allowed for a guaranteed range of motion of any arbitrary position within ± 8 mm and $\pm 3^{\circ}$, the largest range achievable while keeping within the constraints. While all results offer similar ranges of motion, more detailed searching [5] revealed that by allowing variation in the resting height, the range of guaranteed motion could be increased to ± 8.7 mm by selecting a resting height of 46.25 mm, rather than then nominal 45 mm, which allowed up to ± 8.5. However, this advantage was deemed as a near non-detectable increase in performance, so the height differential was arbitrarily selected to be 45 mm.

While this search function was written specifically for the device described in this dissertation, it is applicable to any systems based on the same prototypical geometric design, including those that may be later built as complex motion phantoms, including dimensional freedom within any combination of translations and rotations, so long as the basic mechanical functionality is the same: control of an end effector by translation of connected linkages.

[5] by reducing the search range and increasing the search resolution

2.3.2 Stress analysis

Rather than relying on laser cutting again, 3D printing was chosen to fabricate nearly all major components. This allowed much greater freedom in the design process. Objects that required fine detail printing (components with features smaller than 1 mm) were selected for printing with the Formlabs Form 2 printer or with a Polyjet Objet Eden 260VS. Components which did not require high detail were printed with a combination of fused deposition modeling printers, including uPrint SE Prototyper printers, MakerBot Z18 printers, and MakerBot Replicator printers (fifth generation). The materials selected were acrylonitrile butadiene styrene (ABS) when using the uPrint printer, polylactic acid (PLA) for any MakerBot-printed parts, the Formlabs "standard" resin, or VeroWhitePlus (RGD835) polymer when using the Objet Eden. Some components required high strength and mechanical accuracy. For these components, machined carbon-fiber-reinforced plastic was selected.

Finite-element modeling was used to help determine whether the designed components were strong enough to bear the weight of a patient using the materials in question. The primary dimension of concern was the thickness of the bar in the linkages, together which serve as a combination of structural pillars and load-bearing cantilevers, depending on the particular configuration of the the plate. Since the length of the linkage is set from the geometric optimization process, the thickness of the linkage is the only independent variable which may be used to augment the strength of the linkage[6].

Four different plate configurations were examined to ensure stability of the system for a variety of situations, listed in Table 2.5. The configurations were chosen. In all configurations, a CAD model head of weight 75 N (approximately the weight of a human head) was simulated on top of a durable headrest with equivalent dimensions to commercially available plastic headrests (Figure 2.18) to ensure that the load distribution was as close to reality as possible. The headrest was fixed to the plate using plastic indexing pins. Only

[6]The linkages would be printed with a 1-mm shell thickness with linear infill.

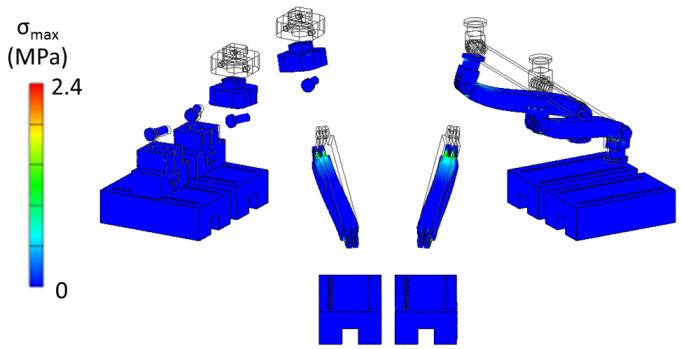


Figure 2.17: FEA of the linkage in the third-generation prototype

Table 2.5: Estimated stresses

Position (mm/°) (x, y, z, α, β, γ)	Max First Principle Stress (MPa)	Max von Mises Stress (MPa)	Max Equivalent Strain (%)	Max Deformation (mm)
(0, 0, 0, 0, 0, 0)	1.35	2.66	0.112	0.091
(0, 0,-10, 0, 0, 0)	1.86	3.52	0.146	0.124
(0, 0, 10, 0, 0, 0)	1.05	2.04	0.085	0.067
(10,-10,-10, 0, 0, 3)	2.17	3.71	0.155	0.138

load-bearing or connecting components were included in the analysis. For each configuration, the first principle stress, von Mises stress, equivalent strain, and deformation were examined (results based on final geometry summarized in Table 2.5). Based on maximizing the strength of the linkage bar and the angular freedom, a width of 10 mm was selected (Figure 2.17). FEA was also used to ensure that the supporting mounting plate segments could support the load of the patient on the positioning equipment, shown in Figure 2.19. Based on the simulations, no structural failures are expected in the plate, sockets, linkages, sliders, slider tracks, or mounting plates. One caveat though is that the sockets for the hinge components and sliders were designed as joining halves, which may not be as strong as the simulation implies.

Computational fatigue analysis was also utilized to determine the system's long term resilience against repetitive loading, using the Nastran add-on package for Autodesk Inventor. In this analysis, a simplified geometry of the linkages, hinges, sliders, and positioning plate was used, where the entire assembly was simulated as ABS plastic. In reality,

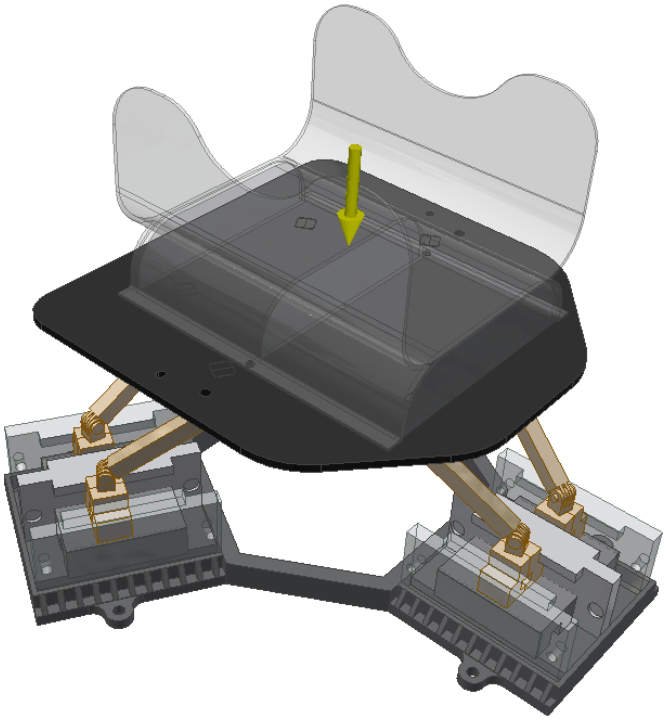


Figure 2.18: Free-body diagram for core positioning system FEA

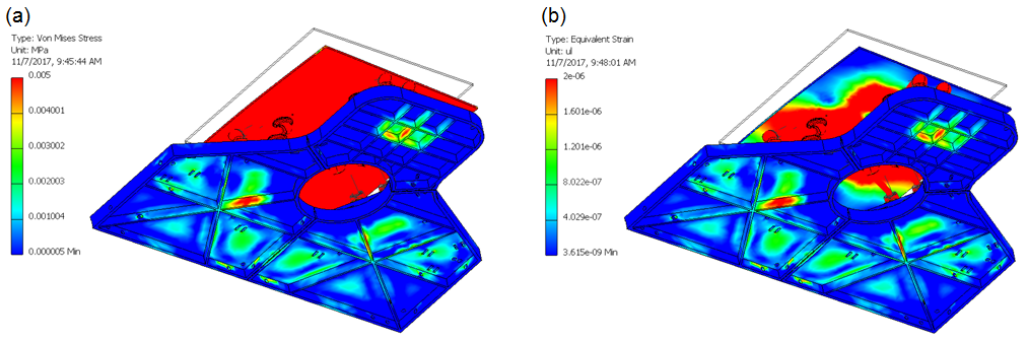


Figure 2.19: FEA of mounting plates

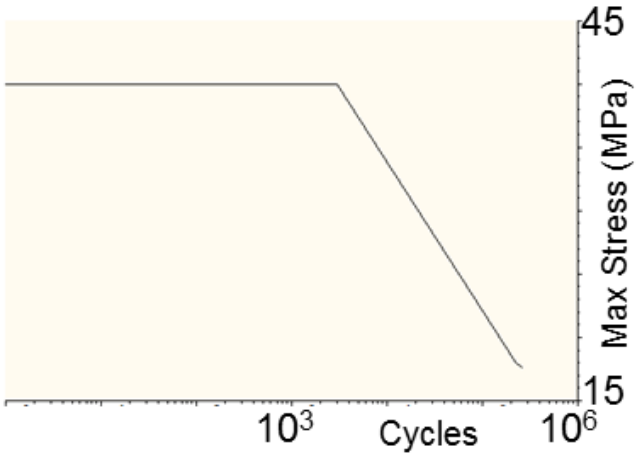


Figure 2.20: S-N curve programmed into Nastran, based on data from [56]

the plate was fabricated from PMMA, a stronger material. The material properties for the S-N (Stress-cycles) curve were estimated based on data from Park *et al* [56] (Figure 2.20).

The analysis simulated a load of 75 N dynamically loaded normally onto and off of the positioning plate each with a 1-s duration. Nastran estimated that the linkages were the most likely part to fail, after approximately 10^9 cycles. Although the real failure could be earlier, because 3D-printed components may contain inherent stress points not seen in conventionally manufactured parts, the analysis indicates that there is a significant stress margin in the component.

2.3.3 Torque analysis

The layout for the shaft mountings was developed in CAD software and employed custom bevel gears to allow for a reduced number of gears in each drivetrain. Transmission between well-fitting gears is expected to be around 90% [57], and the total power transmission approximately follows:

$$P_f = P_0(\text{efficiency})^n \tag{2.12}$$

where P_0 is the initial power and n is the number of gear pairs in the train. The gears

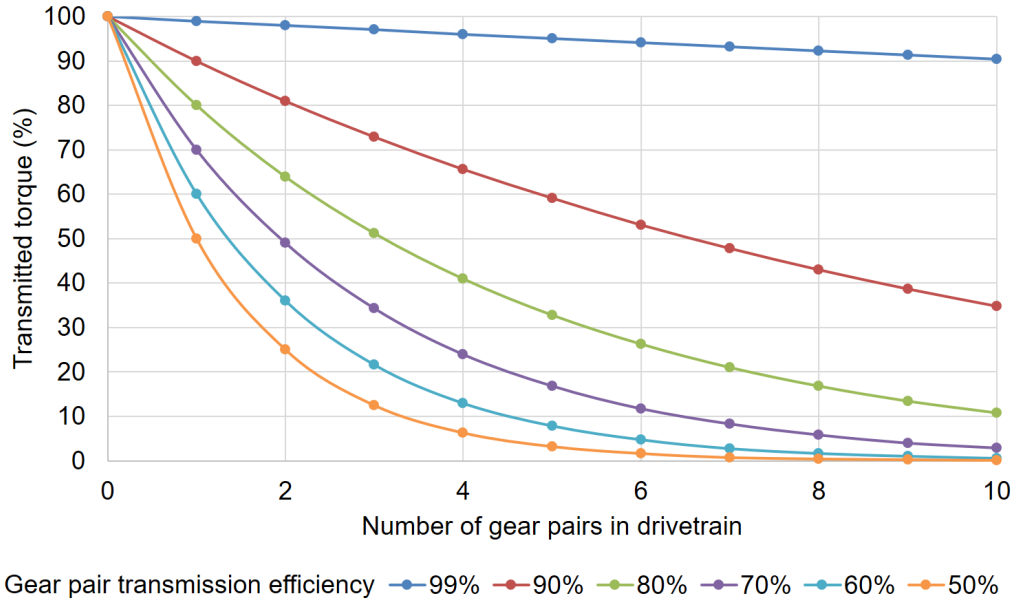


Figure 2.21: Torque transmission analysis

were created using the ”Design Accelerator” feature in Autodesk Inventor. Two types of bevel gears were designed: one type was used to create a 30° change in axle direction, and a separate type to create a 60° change in axle direction. This was done to minimize the number of gear pairs required to connect each motor to their respective slider since each pair of gears along the drivetrain reduces the power transmission efficiency. The layout was generated by carefully considering the optimal position for each bevel gear pair. A bevel gear pair may be considered as a pair of partial cones with intersecting vertices. Perfectly fitting pairs of each type of gear were created in CAD. These pairs were placed along each type of drivetrain, such that the drive gears were coaxial with the shafts of the motors, the gears sharing a common shaft were always coaxial, and the end gears sharing shafts with the sprockets were also coaxial. In the design process, the height of the full cone of each is known, so the layout may be created such as that shown in Figure 2.23.

Once custom gears were considered, a torque step-up in gear ratio was also implemented to reduce the required input torque from the stepper motors, and therefore reduce the weight of the system. Two different stepper motor models were considered for this system, NEMA 17 1.2 Amp motors and NEMA 23 2.8 Amp motors. Comparatively, the

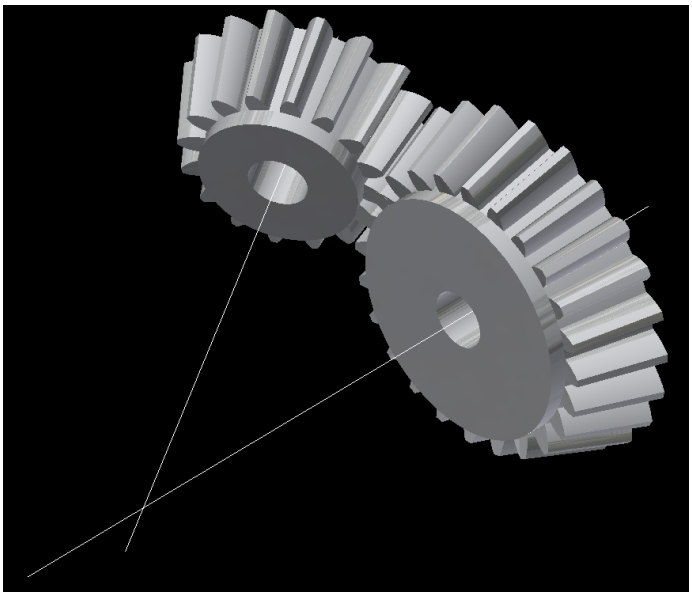


Figure 2.22: 30° bevel gears

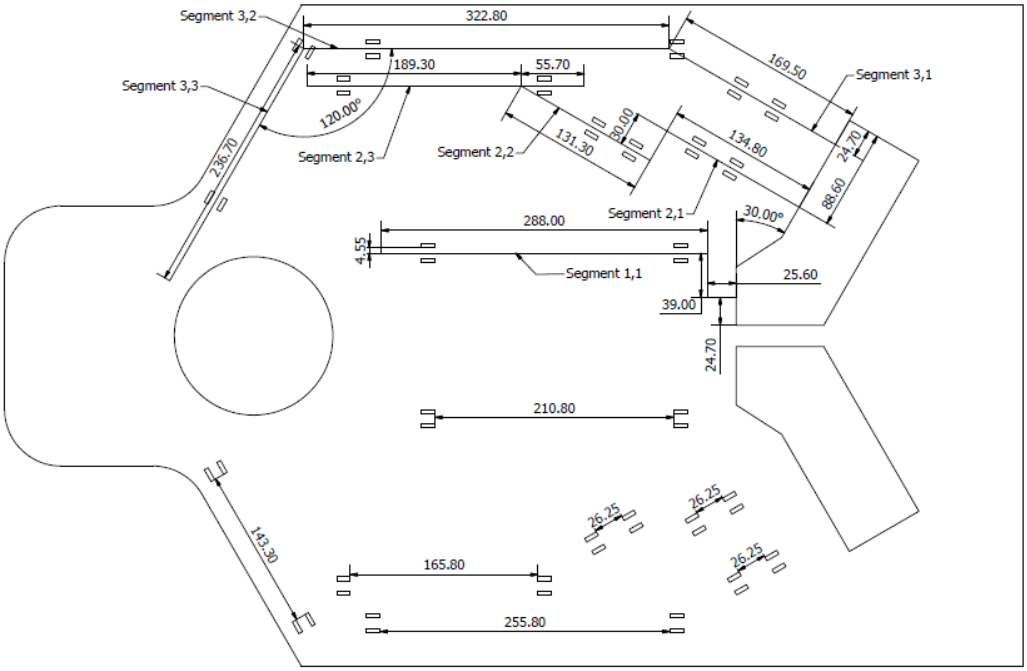


Figure 2.23: Layout of the gears and axles

NEMA 23 motors would provide significantly higher input torque, 270 oz-in compared to 44 oz-in by the NEMA 17 motors. However, the weight of each motor was also significantly greater, with a mass of 1.0 kg compared to 0.35 kg. Therefore, the low-weight NEMA 17 motors appear in the final device, resulting in an input torque of 44 oz-in.

Snap fit joints were developed to securely lock into place, ensuring proper alignment of the shaft mounts. Once snapped in, the mounting could be moved by $\approx \pm 0.5$ mm laterally with respect to the axle for final positioning and permanent attachment with a strong adhesive. Each shaft was held in place within the mounting by a pair of 3D-printed smooth bore solid bearing. The use of custom 3D-printed ball bearings was explored, and some prototype examples were constructed, but material strength became an issue at the size considered. Most notably, the races in the bearings was constrained to submilimeter thicknesses due to size limitations, and was not strong enough to support the stresses caused by torque from the shafts.

A lid was designed for the shaft mounts to hold these bearings in place, held in place by zip-ties as a reversible construction process. Longitudinally, the axles were held in place within the mount by a series of 3D-printed spacers, each 1 mm in cylindrical height. As mentioned in section 3.1.6, the axles mounts were designed with mostly hollow space, consisting of a 2-mm thick shell and 1-mm ribs. The mount and lid were printed using "Tough PLA", to allow elastic deflection of the snap joint fittings.

There is no one to one correspondence between linear motion of the sliders and achieved motion of the platform. Instead, there is a complex, nonlinear relationship between the sum positions of the sliders and the resultant platform position. Therefore, rather than using direct mechanics to model the necessary applied force to a slider to move the plate, an analogy may be used instead. Ultimately, the applied force to the slider is dominated by overcoming the friction force between the slider and its track. Show that the force required is dominated by friction. In this simplification, the force required to move any

slider is simply the force required to overcome static friction:

$$F_{f,s} = \mu F_N \tag{2.13}$$

Where $F_{f,s}$ represents the static friction threshold, μ represents the coefficient of static friction between the slider and the track, and F_N is the normal force of the slider. Note that the normal force of the slider is the result of the composite weight of the entire assembly, including the load applied to the platform.

$$F_N = g(m_{slider} + m_{lowersocket} + m_{uppersocket} + m_{linkage} + \frac{m_{platform} + m_{load}}{6}) \tag{2.14}$$

In order to calculate this required force, the coefficient of friction must be known. Therefore, empirical analysis was performed to determine this value. From rest, the force required to overcome static friction between the slider in its track was 20 mN, compared to the measured weight of 80 mN, which can be used to conclude the coefficient of static friction between these two substances as 0.25. Using this value, and the known masses of the components, and using a maximum load of 75 N, the required force to move the slider in the full assembly with 75 N load is 3.25 N.

A roller chain and sprocket assembly was utilized to move the slider. Since the effective nominal diameter of the sprocket was 0.92 inches, this gives a required torque of 62 N-mm. A drive train of gears were used to increase the applied torque of the motors from their maximum of 5.5 kgf-cm (539 N-mm) to a maximum torque of 1820 N-mm using a net gear ratio of 8:27. In certain configurations, the gear ratio was achieved using three instances of a 2:3 gear while in other configurations, the gear ratio was achieved using a single instance of a 2:3 gear and a single instance of a 4:9 gear. Note that the mechanics are substantially simpler if the same gear ratio is present in driving all six sliders.

This gear ratio also gave the advantage of increasing the effective step resolution of the stepper motors. Since each stepper motor used had a step resolution of 200 steps/rotation,

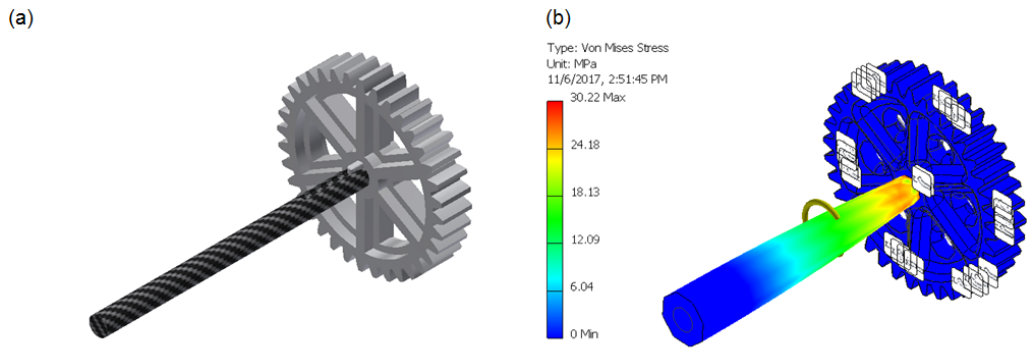


Figure 2.24: (a) Diagram of key and keyway, and (b) torque analysis of drive rods.

a 1:1 gear ratio would have moved the sliders 1.063 mm/step. However, with the applied gear ratio, this reduces to 0.315 mm/step. A more comprehensive analysis of the step resolution is explored in 2.1.4.

Finite element analysis was used to determine the stress present at the interfaces between drive rods and the attached gears or sprockets using the torque calculated in the previous section. The maximum torque in the drivetrain was used for all interfaces, regardless of the expected torque at any single interface. In the analysis, a prototypical shaft with a gear attached was simulated with the gear held fixed and 1050 N-mm torque was applied to the shaft (Figure 2.24b). In this analysis, the maximum stress present at the interface was found to be 48 MPa, directed around the surface of the shaft. In researching various epoxies for holding the gears onto the shaft, this was found to be significantly greater than their yield strength (typically around 14 MPa), indicating their poor candidacy for holding strength. Therefore, a keyway and key were instead selected as a means of transmitting the rotational energy from the shaft to the attached gears (Figure 2.24a). In the finite element analysis of the keyway, the key was found to be more than strong enough to rigidly fix the components together.

2.3.4 Construction

Relative to the second-generation prototype, the third-generation prototype heavily used 3D printing to create structural and mechanical components. The third-generation prototype repurposed the slider mechanism from the second-generation prototype. In the second generation prototype, assembly of the mounting plate was impeded by the maximum size of material slab that could fit in the laser cutter, 12×24 in^2. This led to assembling the mounting plate from several individual segments welded together on their adjacent edges. In practice, this process was imprecise, and caused some flexure and misalignments. In the third generation prototype, the 3D printer used for creating the mounting plate also had a build volume limited to $30 \times 30.5 \times 45.7$ cm^3, necessitating segmentation. However, in this configuration, plastic nuts and bolts were used to secure the plates together ensuring much more precise component alignment. Additionally, carbon fiber rods were used beneath the mounting plates to give the structure extra rigidity.

2.3.5 Radiographic imaging compatibility

The materials used in the positioning system were carefully selected to minimize the effects of scatter and attenuation on incident radiation. All components within 40 cm horizontally of the positioning system's isocenter were constructed using fully plastic materials. Two metrics were used for assessing the compatibility of the system with radiographic imaging. First, no component could possess a measured Hounsfield Unit value greater than that of cortical bone at diagnostic energies. Second, when imaged, no obvious scattering or attenuating-linked artifacts could be visible.

An estimate of the Hounsfield Unit for each material used is shown in Table 2.7. The estimate was generated using the basic definition of Hounsfield Units:

$$HU_{material}(E) = 1000 \times \frac{\mu_{material}(E) - \mu_{water}(E)}{\mu_{water}(E) - \mu_{air}(E)} \tag{2.15}$$

Table 2.6: Estimated radiographic properties of the used materials

Matieral	Chemical Composition	$\rho(g/cm^3)$	μ/ρ(40keV) (cm^2/g)	Estimated Attenuation (HU)
PLA	$C_3H_4O_2$[58]	1.25	0.214	-2.0
Tough PLA	$C_3H_4O_2$[58]	1.50	0.214	198
ABS	$C_{15}H_{17}N$[59]	1.05	0.220	-138
Nylon	$C_{12}O_2N_2H_{22}$[60]	1.13	0.231	-26
CFRP	67%C; 33%$C_{21}H_{25}ClO_5$ [61]	1.44	0.241	295
POM	CH_2O[62]	1.41	0.244	285
PMMA	$C_5H_8O_2$[63]	1.19	0.235	44
PTFE	C_2F_4[64]	2.20	0.265	1180
Water	H_2O[65]	1.00	0.268	0
Air	78%N; 21%O; 1%Ar[66]	1.29×10^{-3}	0.246	-1000
Bone	-	-	-	2500

The net linear attenuation coefficient for each material was calculated using the method specified by Attix for combining linear attenuation coefficients in a mixture [67]:

$$(\frac{\mu}{\rho})_{mixture}(E) \cong (\frac{\mu}{\rho})_A(E)f_A + (\frac{\mu}{\rho})_B(E)f_B + ... \tag{2.16}$$

The chemical compositions and densities of the materials used are shown in Table 2.6. The composition of most used materials are well known, but some materials used are less established. The Carbon fiber-reinforced polymer is a mixture of carbon fibers and a reinforcing epoxy resin. While the exact ratios of carbon fiber to epoxy are unknown for the materials used, an educated guess is approximately a 2:1 mix. The chemical composition for carbon fibers is simply pure carbon, but many possible forms of epoxy could have been utilized in the formation of the materials used in this project. Since the purpose of this exercise is to obtain an estimate of the attenuating properties, one possible chemical composition for epoxy is $(C_{21}H_{25}ClO_5)_n$. Likewise, while the photopolymer used in printing with the stereolithography printer is a proprietary secret, some sources have indicated that photopolymer materials may be similar to that of poly(methyl methacrylate) [68], which provides a reasonably close estimate of the actual attenuation. In all, in no case was the

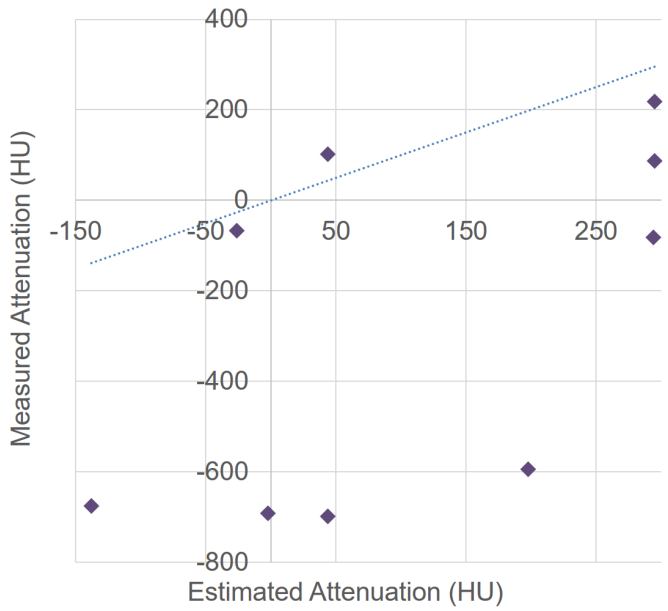


Figure 2.25: Relation between estimated and measured radiographic attenuation. Note that in all but one case (the track walls), the estimated attenuation was greater than the actual attenuation, showing better-than-expected compatibility.

estimated attenuation greater than the attenuation values listed by Attix for cortical bone.

After construction of the prototype, the device was placed in a CT simulation unit for measurement of HU values. While some of the analytic estimates of attenuation were close to the measured values, some differed greatly due to the small physical thickness of many plastic components. In many cases, non-load bearing components were designed to utilize a minimal amount of plastic to reduce production costs and production time. These components were often printed as plastic shells with reinforcing ribs using industry recommended thicknesses between 2 mm and 3 mm for the shells and 1-mm thickness ribs [69]. Due to this design practice, some components lacked volumetric regions spatially larger than the voxel size of the CT scanner, resulting in significant volume averaging between the air and the plastic components. Further, some non-shelled components were created from thin materials on the same scale as the slice thickness to reduce attenuation, (i.e., the positioning plate was a 4.76 mm thick PMMA plate) and thus also experienced some volume averaging as well. The measured HU values for each component was nearly

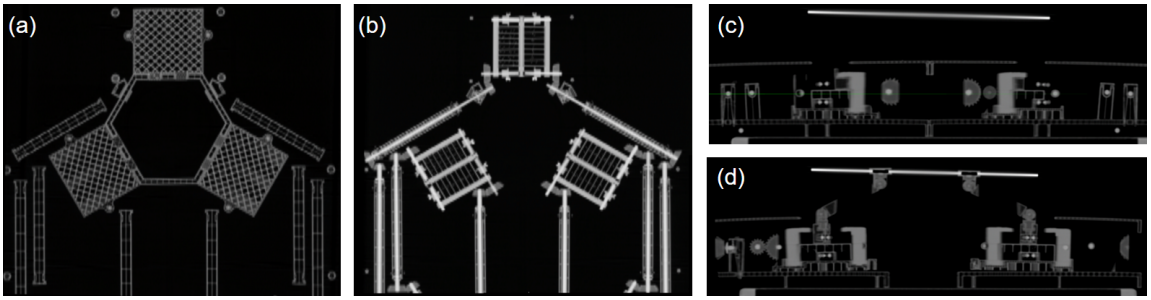


Figure 2.26: CT scans of the third generation prototype including (a) horizontal slice view at base level of supports, (b) horizontal slice at level of gears and shafts, (c) vertical slice through slider assembly, and (d), vertical slice through linkage attachment points.

Table 2.7: Comparison between the estimated and measured radiographic properties

Part Name	Material	Mean Measured Attenuation (HU)	Estimated Attenuation (HU)
Track Walls	PMMA	102	44
Slider Supports	PMMA	-699	44
Shaft Mounts	Tough PLA	-594	198
Mounting Plate	PLA	-692	-2
Chains	POM	-82	284
Postitioning Plate	CFRP	218	295
Shafts	CFRP	87	295
Bolts	Nylon	-67	-26
Sliders	ABS	-675	-138

always lower than the estimations (Figure 2.25), and no artifacts were visible, implying full compatibility with radiographic imaging (Figure 2.26).

2.3.6 Slider positioning feedback loop

Even with the careful shaft layout and including a key and keyway to fix the gears to the shafts, some lashing remained in the gearbox. While much if this could be reduced by tightening the attachment and articulation of components, some of it seemed to stem from the inherent flexibility of the plastic materials used in construction. The lashing created a serious issue in relying on dead reckoning with the step calculation as a means of determining the true locations of the sliders; in some test cases, the true motion of the sliders differed by more than 3 mm from the programmed motion. This magnitude of error

would cause significant errors to accumulate in the resultant plate positioning. With some experimentation, the backlash also seemed to vary from day to day, precluding the option of accounting for a consistent backlash per gear within software.

As an alternative to relying on dead reckoning navigation for positioning the sliders, a feedback loop could instead. In this scheme, the sliders would still follow the prescribed motion trajectory as calculated by the stepping algorithm, but following the conclusion of the motion, a physical measurement could be used to detect residual errors and determine the necessary steps to achieve the requisite position. Using an optical reflectance measurement as a surrogate for position measurements was briefly considered, but a few flaws contraindicated its use. The optical conditions were too variable, and integrating the hardware into the existing prototype would have required extensive overhauls, and would have required using fiber optic cables to maintain the prohibition on high-Z components near the positioning core.

Eventually, a Bowden cable system was devised to create a mechanical connection between each slider and a sliding potentiometer (Bourns linear slide potentiometer, 250Ω, 60-mm travel) located far outside the treatment field (Figure 2.27). A Bowden cable is a flexible tube and wire assembly in which the ends of a flexible tube are fixed onto discrete reference points while the wire is ran through the tube and fixed on either end to an actuating component and an end effector. They are commonly seen in bicycle handbrake assemblies as the method of transmitting mechanical power from the handle to the brake. Custom Bowden cable assemblies were created using nylon tubing and PTFE 2-mm diameter wires. PTFE was selected due to its low coefficient of friction to minimize drag against the tubing.

The ends of the wires were fixed to the sliders using cyanoacrylate adhesive (Figure 2.27b). This was achieved by first creating a rough texture on the outer surface of the wires with needle-nose pliers, forming a drop of adhesive on the tip of the wire, and then inserting the end into a shallow hole drilled into the face of the slider. The tubes were

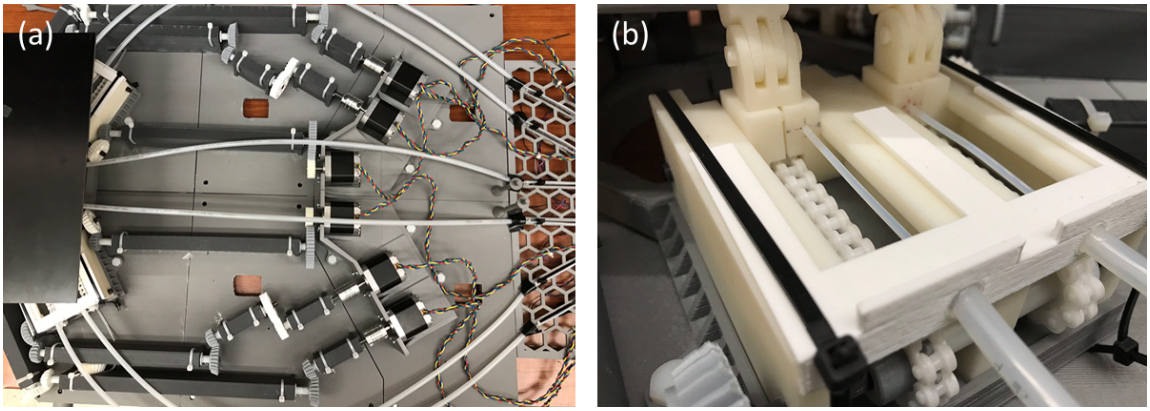


Figure 2.27: Bowden cable assembly, showing (a) the general layout of the cable system and sliding potentiometers, and (b) the attachment point onto the sliders.

mounted between attachment points on the tracks and near the potentiometers.

The myRIO hardware was used to power and read the potentiometers by applying an +5 DC voltage and measuring the potential difference across the resistor. The voltage is then converted into a distance by simple ratios. In the control software, the measurement of the voltage was very noisy when the stepper motors were powered. Therefore, the true distance is based on a 100-point moving average of the voltage. This process did introduce a 1-second latency period, however.

The feedback loop functions by performing periodic checks on the measured position of the sliders and compares them to the expected positions. After every given number of trajectory-based step commands are sent to the motors, the system waits for the 1-second averaging period, compares the measured positions against the expected trajectory-based positions, and provides a constant-velocity command to make up the difference. This process ensures that the achieved trajectories for each slider matches the planned trajectories as faithfully as possible.

The feedback loop contains several points of adjustment for sensitivity, and room for optimization. More frequent comparisons will ensure greater trajectory replication, but will also increase the time required to move from start to end position. Additionally, while each correction step is built to compensate for lashing in the gears, this is done indirectly

by measuring discrepancies in the expected positions, which can lead to either under-corrections or over-corrections especially in the early segments of the motion. This could be better tuned by more frequent comparisons, but the existence current latency duration discourages such practices by dramatically lengthening the required time for a start to finish motion.

2.3.7 Construction of a flexible anthropomorphic phantom

An anthropomorphic phantom with a flexible neck was constructed for testing whether independent positioning of the head and body results in proper neck alignment, affectionately referred to as "Jack". The phantom was constructed primarily from a plastic classroom model skeleton (Anatomy Warehouse). While the skeleton came with many small metal components to articulate each bony joint, the vast majority of these components were removed, along with many bones, leaving the phantom to contain the skull, ribs, scapulae, clavicles, and vertebrae C1-T12, and rubber vertebral discs. Two thin wires were routed through the vertebral column and through the occipital bone near the foramen magnum to provide flexible articulation. The wires were held in place to the skull by wrapping the wires through the foramen transversarium of the C1 vertebrae and through the added holes near the foramen magnum and the jugular foramen.

The phantom was wrapped in silk tape and then covered with three layers of a flexible nylon mesh as a skin surrogate to facilitate the use of surface-based optical positioning. An air cavity made from compressed paper representing nasal cartilage was also added beneath the skin surrogate to remove symmetry ambiguities.

2.3.8 Positioning accuracy and analysis

The positioning accuracy of the prototype was tested in a similar manner to that used for the first generation prototype, as described in section 2.1.6. In this instance, the skeletal phantom was used as the patient surrogate for correction. Because the skull of the phantom

Table 2.8: Summary of displacements used in accuracy test of third generation prototype

Direction	Mean absolute positioning displacement (mm/°)
Longitudinal	1.73 ± 1.02 [0.4, 4.5]
Lateral	0.73 ± 0.48 [0.1, 2.0]
Vertical	1.41 ± 0.99 [0.1, 3.2]
3D Vector	2.52 ± 1.16 [1.1, 5.2]
Pitch	0.73 ± 0.44 [0.1, 1.5]
Roll	1.57 ± 0.72 [0.5, 2.6]
Yaw	0.91 ± 0.59 [0.3, 2.7]

was smaller than the Rando phantom, and showed noticeably less friction between its surface and that of Q-Fix headrests, a rubberized grip ring was used as a headrest. This was also done to help ensure that the load distribution remained within the points of stability on the plate by keeping the weight from the neck of the edge of the plate.

The test consisted of 15 independent correction trials, where the head was set an arbitrary reference position within the headrest, followed by a reference capture by AlignRT, then creating an arbitrary misalignment by manual manipulation. Negations of the detected displacements were used as correction vectors, first applying a correction for rotations, followed by the resultant measured translation displacements. This resolved the discrepancies between pivot points used in calculation by the surface tracking system and that used in the plate positioning software. In the tests performed with the first generation prototype, the pivot point of the positioning software was located at the default position at the center of the anterior surface of the plate. In these tests, the movable pivot point functionality was implemented to reduce alias translations. Therefore, the pivot point was placed approximately at the location of the C2 vertebrae (90 mm inferior to the center of the plate, 60 mm above the anterior surface of the plate), based on the Graff's observation that the rigid registrations of C2 provided the best alignments of bony landmarks.

The results of the manual errors are summarized in Table 2.8. Over 15 trials, the mean absolute vector displacements for correction ranged from 1.1 to 5.2 mm (distribution of 2.52 ± 1.16 mm). The rotations used were all below 3° to ensure that the position would

Table 2.9: Summary of residual errors from accuracy test of third generation prototype

Direction	Mean absolute positioning residual error (mm/°)
Longitudinal	0.47 ± 0.28 [0.1, 1.1]
Lateral	0.21 ± 0.14 [0.0, 0.5]
Vertical	0.43 ± 0.34 [0.1, 0.9]
3D Vector	0.74 ± 0.33 [0.2, 1.5]
Pitch	0.41 ± 0.27 [0.1, 1.0]
Roll	0.63 ± 0.58 [0.0, 2.0]
Yaw	0.31 ± 0.24 [0.0, 0.8]

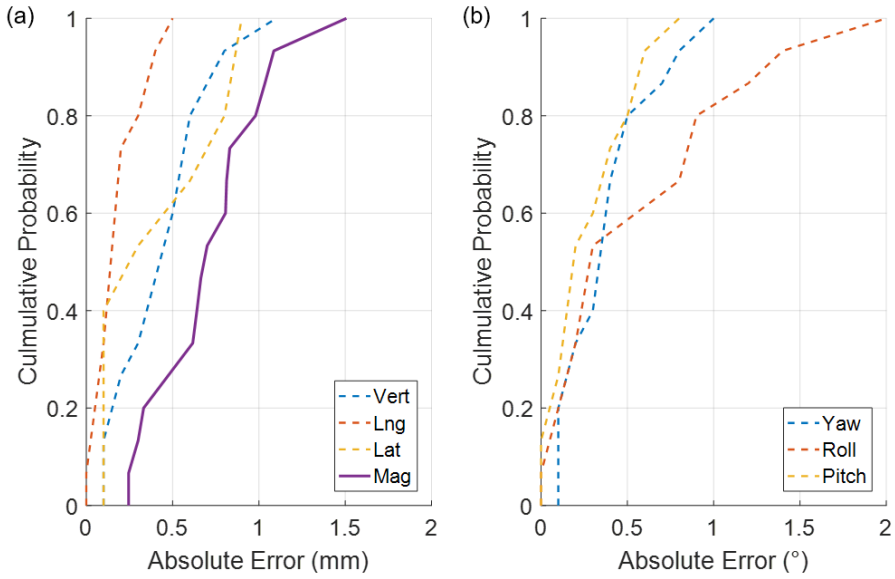


Figure 2.28: Cumulative histogram of residual errors for third generation prototype for (a) translations and (b) rotations.

always be achievable. The residual errors post-correction are summarized in Table 2.9 and seen in cumulative histogram in Figure 2.28. The mean absolute vector displacements for correction was 0.74 ± 0.33 mm, indicating the prototype's likely accuracy as sub-millimeter. The residual rotation error also showed significant improvement relative to the initial offset, although a larger ratio of residual errors greater than 1° occurred relative to the first generation prototype.

Relative to the first generation prototype, the system was slightly less accurate overall (≈0.3-mm increase in mean absolute residual vector error, with an additional ≈0.1 mm in-

crease to the standard deviation). This is likely due to the increased complexity in the drive system in the third generation prototype relative to that in the first generation prototype, and the increase in step size. In the first generation system, the drive system consisted of stepper motors rigidly attached to threaded rods that directly actuated the sliders by 6.25 μm per step, guided by dead-reckoning (counting steps). In contrast, in the third generation prototype, the motors attach to train of gears and axles, (where each gear has some backlash), that ultimately turn sprockets that pull a chain that actuate the sliders, with approximately 0.1 mm of slider actuation per motor step, guided by a Bowden cable mechanism. Therefore, because the third generation system contains a greater number of systems (designed to keep the motors out of possible treatment fields), each system with its own uncertainty, the greater uncertainty is not surprising.

2.3.9 Neck alignment by separate alignment of the head and body

A second set of mechanical correction trials was also conducted to investigate whether independent setup of the head relative to the body improved the alignment of the cervical spine. This series followed the same basic process described in the previous sections concerning positioning accuracy measurement procedure, but with additional steps added. Before any mechanical correction was attempted, the entire length of the anthropomorphic skeleton was set up to be as close as possible to the planning CT image as measured by CBCT. Once in alignment, a reference CBCT image was taken to be used as the source image for registrations of misaligned and corrected images. After this, a series of 3 trials were performed to correct the manually created positioning errors as guided by AlignRT where the rotations were corrected first, followed by correcting the translations, each step with only a single correction vector. After the manual misalignment and after each correction step, CBCT images were taken. These images were then rigidly registered to the reference CBCT image set at three separate regions of interest encapsulating C2, C4, and T1 (Figure 2.29). To reduce confusion about alias transformations (rotation-induced trans-

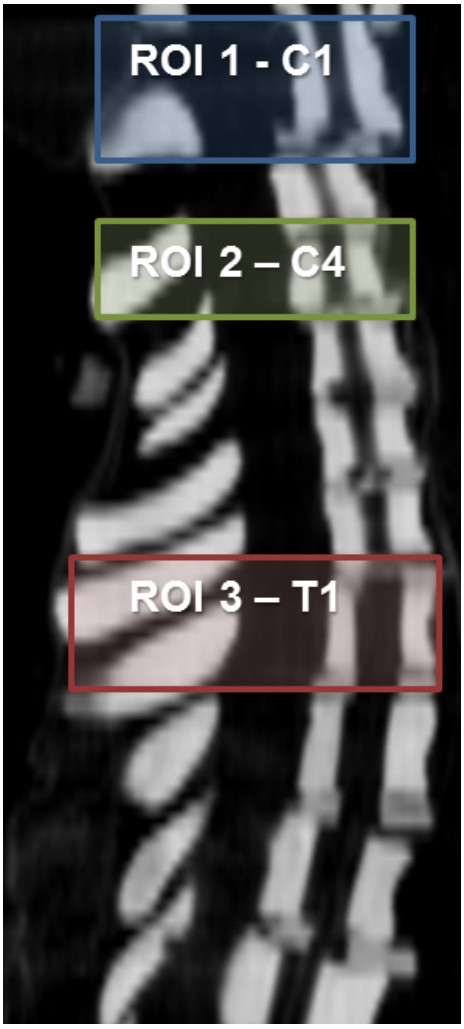


Figure 2.29: Regions of interest used for registration in neck alignment study

lations), registrations allowing translations and rotations were performed separately.

With the mechanical corrections, significant reductions in positioning error relative to the reference setup were observed in each region of interest, as summarized in Table 2.10. Relative to the initial rotation offset, the total rotation (pitch + roll + yaw) decreased from a mean absolute average offset of $2.7^\circ \pm 1.2^\circ$ to $1.2^\circ \pm 0.5^\circ$ at C2 compared to a decrease from $0.7^\circ \pm 1.1^\circ$ to $0.5^\circ \pm 0.3^\circ$ at T1. In both of these instances, not only did the mean error decrease, but so did the standard deviation. In comparing the translation vector error correction (from after the rotation correction to after the translation correction), the residual error at C2 decreased from 3.4 ± 2.4 mm to 1.3 ± 0.7 mm and from 1.0 ± 0.8

Table 2.10: Summary of mean absolute errors in neck levels after correction by third generation prototype as measured by CBCT.

	Anatomical Region	Vector Translation Error (mm/°)	Total Rotation Error (°)
Initial	C2	1.3 ± 0.9	2.7 ± 1.2
Simulated	C4	1.2 ± 1.0	1.6 ± 0.8
Setup Error	T1	0.7 ± 0.7	1.1 ± 0.9
Step 1 Fix	C2	3.4 ± 2.4	1.9 ± 1.1
(Fixed	C4	2.5 ± 2.1	1.7 ± 0.8
Rotations)	T1	1.0 ± 0.8	1.7 ± 1.2
Step 2 Fix	C2	1.3 ± 0.7	1.2 ± 0.3
(Final	C4	1.0 ± 0.4	0.5 ± 0.3
Position)	T1	0.8 ± 0.2	0.5 ± 0.3

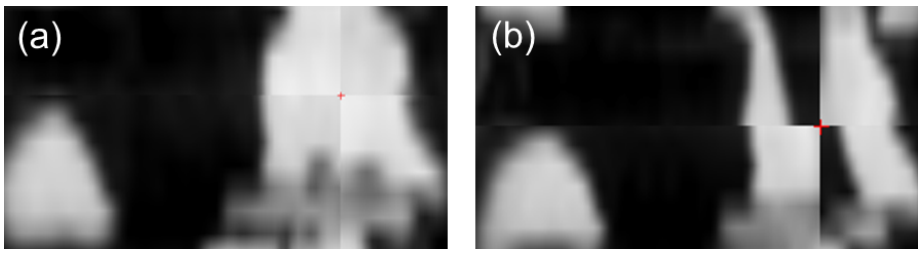


Figure 2.30: Overlaid CT images of C2 vertebrae with T1 registered between images (a) at manual misalignment and (b) after mechanical correction.

mm to 0.8 ± 0.2 mm at T1.

In contrast to the studies presented in the introduction of this dissertation, the greatest uncertainty occured in the upper neck (near C2) rather than the lower neck (near C7). This is primarily due to the fact that the body was specifically set up to match the planning CT, while the head was set up mechanically, which did not appear to be a concern in the studies done by other authors. A possible explanation for the increased uncertainty in the lower neck seen in the literature is that normal treatment workflows for head and neck radiotherapy primarily utilize immobilization systems that restrict misalignments of the head to a greater degree than that of the body. Most multi-ROI studies have observed residual vector errors of approximately 1 cm near the C7-T1, but do not report residual errors more inferior to these landmarks. By extrapolation, landmarks far outside the treatment area could be several centimeters away from the simulation position, leading to the uncertainty

in the lower neck reported in literature.

2.4 Conclusions about the robotic system

Through this process, a mechanical correction system for use in a radiotherapy environment has been produced. Analyses were performed to ensure that the prototype would possess the maximum mechanical freedom within the geometric constrains, possess the required strength to actuate the intended load, maintain structural integrity when powered and under load, sustain durability after receiving radiographic dose, and interfere negligibly with diagnostic and therapeutic radiation. In construction, the prototype was built using methods designed to best ensure accurate motion production when using inaccurate manufacturing techniques. Even so, the prototype is not fully optimized yet. Goals marked for future improvement are reducing acoustic noise, reducing physical size, and increasing the actuation speed.

The noise produced by the prototype could be reduced by adding a more complete enclosure including side panels and perhaps the addition of other sound-dampening hardware. These efforts would be aimed at increasing comfort for patients.

Reducing the physical size of the device would reduce the risk of gantry-patient clearance issues. The third generation prototype is approximately 19 cm tall and 56 cm wide. While these dimensions were necessary given the employed construction techniques and goal of maximizing the range of motion, they could be reduced in future constructions, especially if the range of motion was also modestly reduced. With more accurate and precise manufacturing and construction techniques, the largest gears could be reduced in diameter by perhaps 30%, which could lead to a 2-cm height reduction. Reconfiguring the slider actuation from a chain-and-sprocket design to a toothed-belt could also lead to a further 2-cm of height reduction, though at the expense of decreased torque (although a toothed-belt system may be more efficient, negating the power loss). Better manufacturing could

also allow for a shorter mounting plate, especially if strong materials like CFRP were employed. Most significantly, geometric optimization simulations show that by reducing the guaranteed range of motion from the $\pm$9-cm/$\pm3^\circ$ range down to a $\pm$6-cm/$\pm3^\circ$ range, the width could be reduced by at least 2 cm and the height could be reduced by 3 cm. With all of these revisions in mind, the total height could be reduced by at least 7 cm to 12 cm tall (37% reduction).

In the construction as of the time of this writing, the feedback loop system introduces a 1-second latency period in between each correction step due to required signal averaging to reduce electrical noise from the motors. The inclusion of the latency period can lengthen a start to finish motion by over 100% compared to the required time without it. Therefore, by reducing the electrical noise, less signal averaging would be required, and motions could be completed significantly faster. This would also allow for more frequent feedback loop comparisons, increasing the trajectory fidelity.

Most importantly, the constructed robotic system has been demonstrated to be accurate in positioning, with likely sub-millimeter accuracy, with a MAE of 0.8 $\pm$0.3 mm for correction vectors ranging between 1.1 and 5.2 mm. Additionally, experiments performed with an anthropomorphic skeleton demonstrate the technology's potential to improve setup alignment throughout the neck by independent positioning of the head and body.

Chapter 3

Development of a Motion Compensation System

The *raison d'être* for this robotic system is to provide *active motion compensation*: using mechanical robotic systems to counteract random intra-fraction patient motion in real-time. In the previous chapter, considerations were discussed for designing a robotic system compatible with a radiotherapy environment. However, unlike designing a robotic system for static positioning, one capable of real-time correction must also take into account movement speed, latency, and signal analysis.

3.1 Simulated patient motion

In van Herk's paper on prescribing dose margin sizes, van Herk reminds the reader that the sum of multiple independent random processes approaches a Gaussian distribution [37][1]. This model of random motion is also verified in closer examinations done by Murphy [14] and later by Ballhausen [70]. Ballhausen specifically notes that intra-fraction motion of the prostate is well characterized by a mathematical random walk: consecutive measurements of displacement are not well-correlated in any dimension.

Ballhausen's terminology is likely mistaken, and more probably refers to a *Wiener process* [71][72]. The differences between the two processes are shown in Figures 3.1 and

[1] Van Herk also mentions that respiratory motion is the most prominent exception to this assumption, where sinusoidal models are more appropriate; but respiratory motion is not significantly present in cranial regions [49]

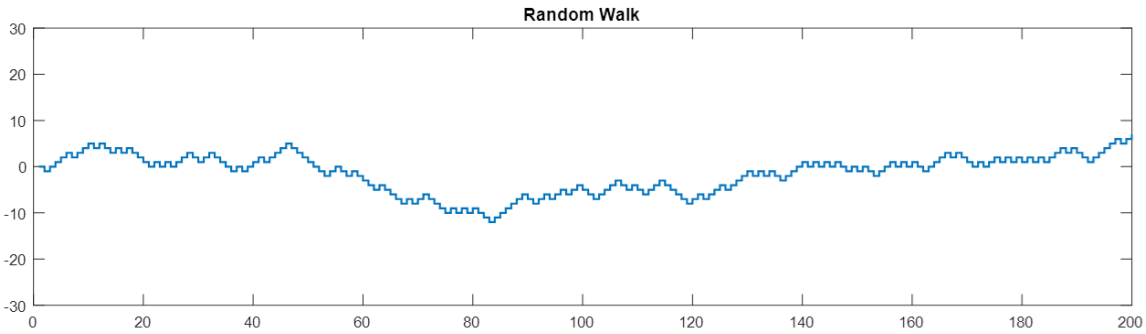


Figure 3.1: A 1D random walk containing 200 steps starting from the origin, generated by taking step size = 1 in a random direction.

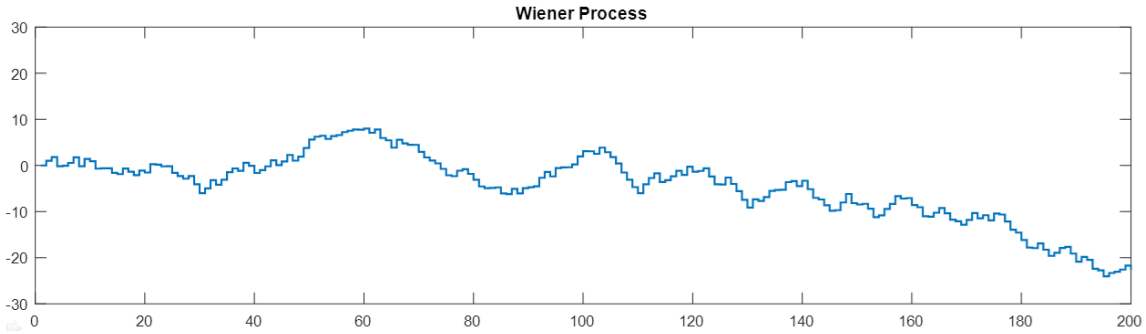
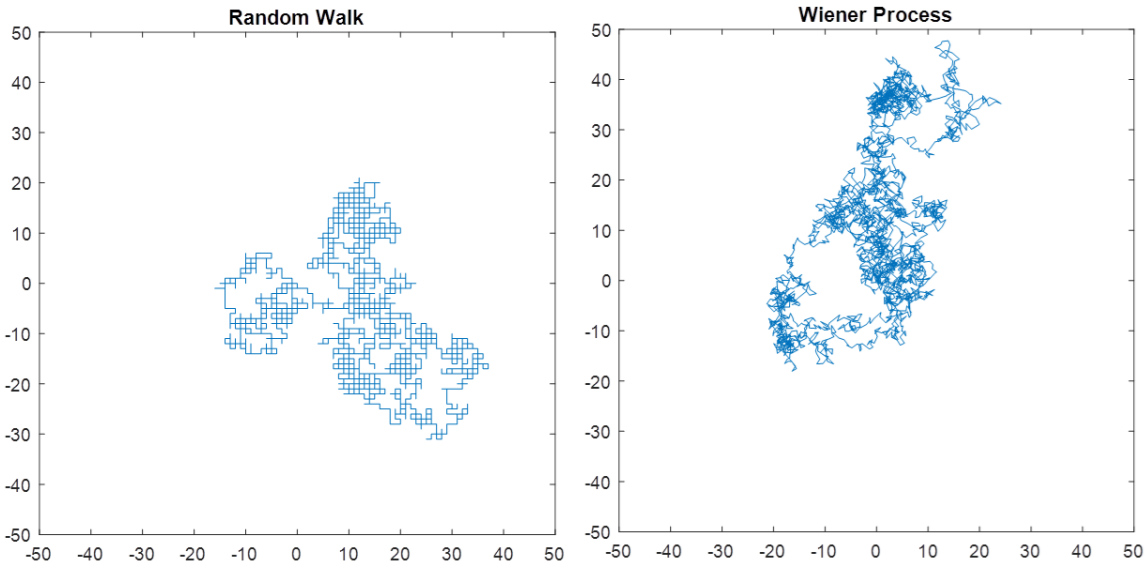


Figure 3.2: A 1D Wiener process containing 200 steps, starting from the origin, generated by taking a step of random size between $[-2, 2]$, so that the mean absolute step size is comparable to that in the random walk example. Note that the step size is variable, compared to the identical step size found in the random walk. The Wiener process is a better representation of real patient motion.

3.2 for the 1D case and in 3.3 for the 2D case. In a true random walk, motion occurs on a mathematical lattice, where each step is random in direction, but restricted to a single dimension per step, and commonly, each step is assumed to be of identical length (Figures 3.1 and 3.3a). This gives rise to rotational dependency obviously not seen in anatomy. Comparatively, a Wiener process is like a random walk, but each step is random in direction and length, and vector combinations of displacements are permitted within a single step (Figures 3.2 and 3.3b). The Wiener process has long historical association as the model of Brownian motion that, like the probable model of patient motion, is rotationally invariant.

To experiment with modes of correcting patient motion with the robotic system de-



(a) (b)

Figure 3.3: Comparison between (a) a 2D random walk and (b) a 2D Wiener process. Both contain 3000 random steps starting from the origin. The random walk was generated by taking step size = 1 in a random 2D direction. The Wiener process was generated by taking a step of random size between $\left[-\sqrt{2}, \sqrt{2}\right]$ in both orthogonal directions.

scribed in the previous chapter, a mathematical model was created to simulate intra-fraction patient motion. For this purpose, this distinction between the random walk and the Wiener proceess is important to ensure accurate replication of patient motion. The simulation was created to simulate a rigid body that follows a Wiener process in 6D for later use to simulate whether the positioning system can accurately correct that motion in real-time. The amplitude of displacement was controllable for each axis, so that the amplitudes could be set to mimic displacements found in the literature. In the study done by Kim *et al* [73], the net displacement for six different patients after a 15-minute treatment ranged between 0.5 mm and 1.5 mm (seen in Figure 1.1). Comparatively, the motions given by Linthout *et al* [15] show much greater volatility, though the treatment duration was not given, and therefore, cannot directly be compared to those reported by Kim. However Linthout does give the range of measured values between each given axes, which can be used to establish approximate ratios of amplitudes for the 6D Wiener process. In Linthout's data, vertical shifts ranged by 6 mm, longitudinal shifts by 3.6 mm, lateral shifts by 5.5 mm, vertical

rotations by 1.6°, longitudinal rotations by 1.7°, and lateral rotations by 2.4°. Therefore, as a rough approximation, the AP amplitude should make up 40% of the translational variations, the SI should make up 25%, and the lateral amplitude should make up the remaining 35% of the translational motion. Likewise, the yaw should make up 30% of the rotational motion, the roll should make up another 30%, and the pitch should make up the other 40% of the rotational motion. Further, the total rotational displacements were approximately 40% of the total of the translational displacements. The simulation was then adjusted so that the motion approximated the findings by Kim and Linthout, resulting in the amplitudes shown in Table 3.1.

Table 3.1: Simulated patient amplitude

Direction	Amplitude (mm/min or $^\circ/min$)	Contribution to type of motion
Longitudinal	2.25	25%
Lateral	3.15	35%
Vertical	3.6	40%
Pitch	1.4	40%
Roll	1.1	30%
Yaw	1.1	30%

The model also contained a degree of variance between patients, taken from a normal distribution. A scaling factor of (0.57 + X$\sim$N(0,.075)) was multiplied by the amplitudes shown in Table 3.1 to obtain the individual amplitudes for each patient, used as a component of the individual shifts between each simulated timepoint. Therefore, each i^{th} displacement, d_i, for each axis, j was taken from a normal distribution, such that

$$d_{ij} = Y \sim N(0, A_j(c_1 + X \sim N(0, c_2))) \tag{3.1}$$

Where A_j is the amplitude estimate taken from Table 3.1, and c_1 and c_2 are constants. In this specific case, the equation becomes

$$d_{ij} = Y \sim N(0, \frac{A_j(0.57 + X \sim N(0, 0.075))}{s60}) \tag{3.2}$$

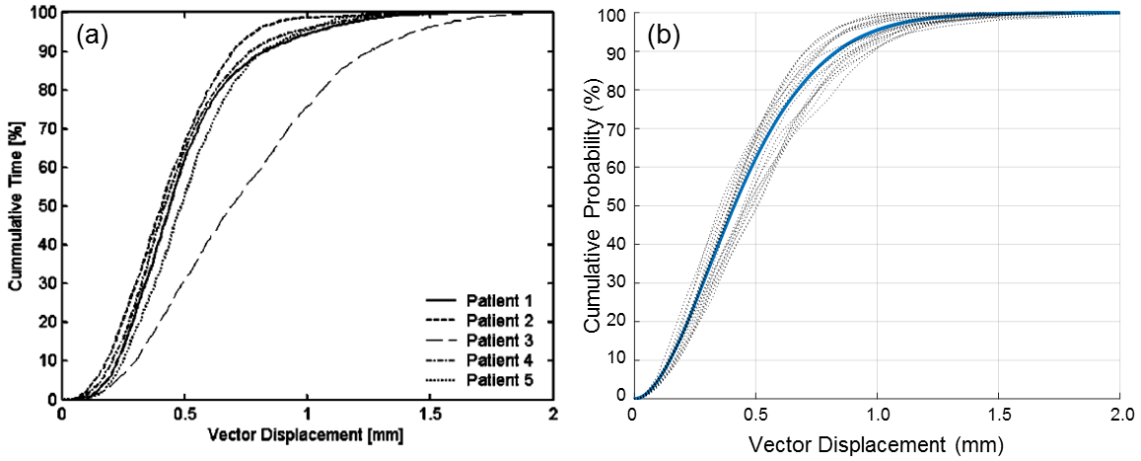


Figure 3.4: Comparison between (a) Time uncertainty histogram from real patients from [73]. Also seen in Figure 1.1, and (b) Simulated patient UTH for 20 patients. Dark line is total cdf for 200 patients over 35 fractions each. Simulation was done with parameters shown in Table 3.1.

Where s is the sample rate of 4 Hz.

To validate the motion model, and to tune the constants in the above equation, the results of the simulated patients with comparable parameters were compared to the results reported by Kim (Figure 3.4). The tuning was accomplished by iteratively adjusting the constant parameters, and minimizing the mean squared error between the 10^{th}, 20^{th}, ..., and 90^{th} percentiles of the median patient's uncertainty time histogram to the equivalent percentiles in a large number (300) simulated patients. The mean squared error between these benchmarks was 0.0034 mm.

3.2 Robotic error analysis

This model was then duplicated into LabVIEW and used as an input into a the basic robotic positioning program, simulating the system response to realistic, time-parameterized displacements. In this case, robotic correction was made without regard to realistic timing of motor signals. The purpose of this simulation was twofold: First, this simulation was performed as a fast method of determining whether the robotic system could achieve any

arbitrary configuration that might be required over the course of a single 15-minute fraction. In some ways, this is similar to a validation of the geometric optimization program described in Chapter 2. By looking at multiple simulations, the upper limits of necessary motion could be identified used to determine whether the robotic was designed with enough mechanical freedom. Second, this simulation investigates something equivalent to mechanical resolution, the inherent uncertainty caused by the discretization of control. In a linear system, this resolution would be linearly related to the resolution of the control system. In example, if a stepper motor with 200 steps per revolution was connected to a gear and rack system so that every rotation of the stepper motor translated the rack by 10 cm, the resolution of the system would be 0.5 mm. Since control of this robot is non-linear, linear displacements of the sliders do not correspond to linear displacements of the plate. Therefore, the residual errors of the slider positions and the plate were investigated. The resolution was defined as:

$$\text{Resolution} = \text{RMS}(\text{Mechanical} - \text{Theoretical}) \tag{3.3}$$

The statistics of resolution were taken from 1000 simulated patients by taking the mean and standard deviation of the RMS, shown in Tables 3.2 and 3.3, giving the range of errors that can be expected for typical patient. Certain types of motion are more likely than others in the simulation generated by the parameters in Table 3.1. Additionally, the simulation began patients with a starting position within a uniformly distibuted space of $\pm$ 5 mm and $\pm$ 3° with a pivot point placed randomly within a uniformly distributed hemi-ellipsoid of $\pm$ 10 cm along the medial-lateral axis, $\pm$ 10 cm along the superior-inferior axis, 0-15 cm in the vertical axis. Given that the stepping algorithm only commands a step to be taken if the absolute displacement for a single slider increases past the step size (as opposed to treating of steps as bins centered at discrete steps), it is not surprising that the achieved resolution is smaller than half the step size because the algorithm is sacrificing some accuracy of the

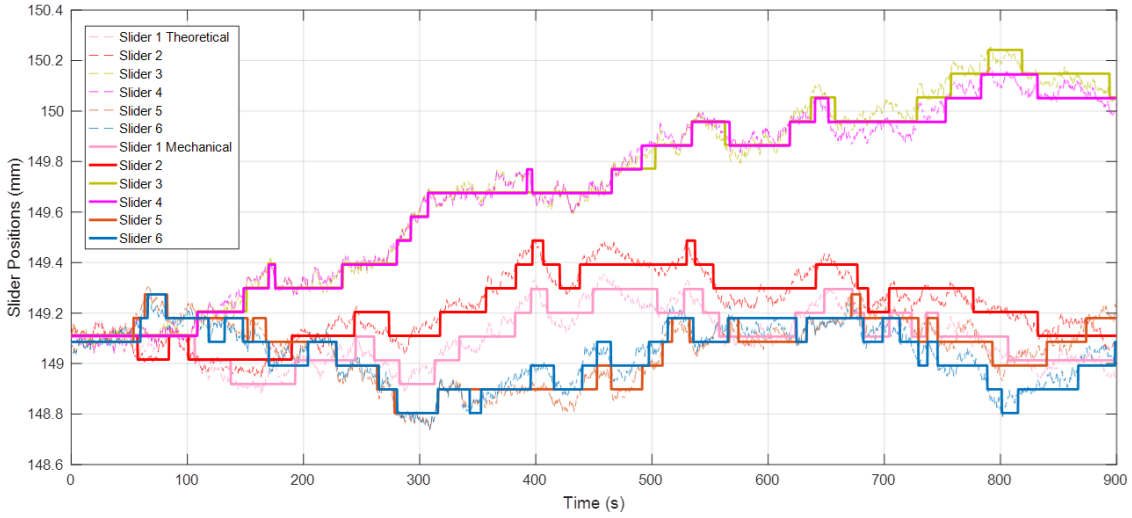


Figure 3.5: Simulated slider output for real-time correction, comparing theoretical and maximum accuracy mechanical output.

plate in favor of computational efficiency. While a slightly more accurate algorithm would use a binning model, the theoretical mechanical resolution of the plate is already smaller than the detection limit of most available localization tools.

Table 3.2: Mechanical resolution of sliders

Slider	Resolution (μm)
1	40.3 ± 1.7
2	40.2 ± 1.6
3	40.4 ± 1.5
4	40.5 ± 1.4
5	40.6 ± 1.4
6	40.6 ± 1.4
Half step size	$\approx$47.0

These simulations can also demonstrate that the required mechanical freedom of the sliders for correcting expected patient motion is very small. While the sliders have a range of motion of 49 mm, on average, the maximum slider displacements to correct real-time motion for each simulated fraction was 0.76 ± 0.33 mm, and the overall maximum displacement for a single slider was 2.60 mm. This discrepancy is the result of designing the system so that very extreme 6D configurations could be achievable. The robotic platform is

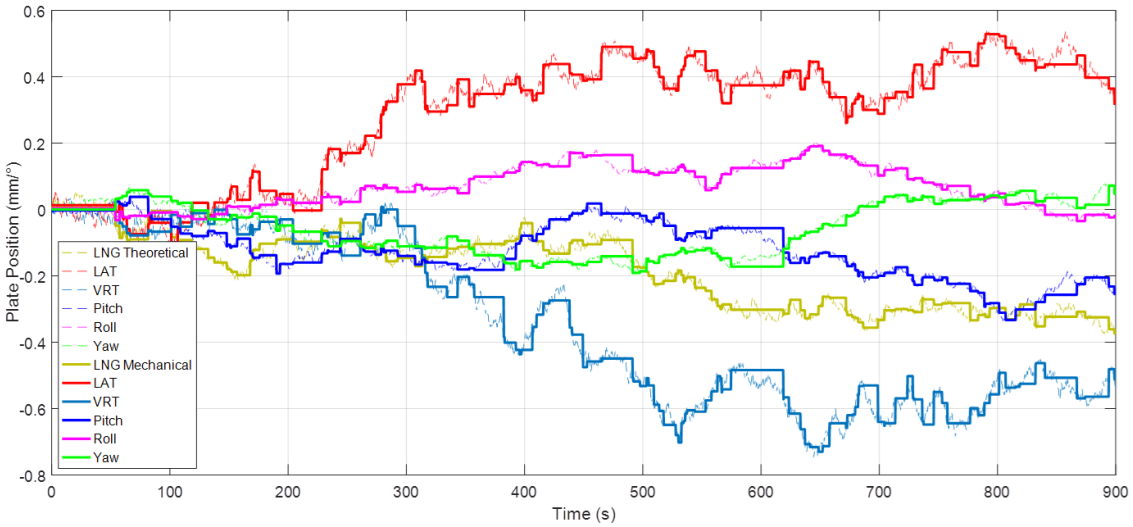


Figure 3.6: Simulated plate output for real-time correction, comparing theoretical and maximum accuracy mechanical output.

Table 3.3: Mechanical resolution of positioning plate

Direction	Resolution (μm or arc-seconds)
Longitudinal	20.4 ± 3.0
Lateral	26.3 ± 3.4
Vertical	30.0 ± 4.3
Pitch	49.0 ± 10.4
Roll	38.9 ± 7.9
Yaw	38.9 ± 7.6

guaranteed to be able to move to any position within ± 9 mm for any combination of axes, while simultaneously achieving any combination of rotations, all within $\pm 3°$. Looking at Figure 3.4(b), one can observe that the expected net displacement from intra-fraction motion is less than 1.25 mm for 95% of all patients, far less than the 17.3 mm that the plate is capable of moving. However, this extra range of motion is available for mechanical patient setup, where displacements are much larger [4][6][7][8][9][10][11][12]. As discussed in Chapter 1, the setup error in C2 compared with the whole body may be up to 5 mm per axis plus rotations. This extra range allows for the real-time correction to occur at any setup position, and with greater freedom in choice of pivot point.

3.3 Motion compensation simulation

The simulations described in the previous sections were used as the starting basis for developing a real-time intra-fraction motion compensation system. The motion compensation software begins by taking the measured position of the patient and running the position through a monitoring algorithm that decides whether a correction needs to be made. The monitoring algorithm makes three different types of checks to decide whether a correction is necessary (Figure 3.7). The first check determines whether the absolute position of the patient for any axis or for the net vector has moved beyond a set threshold, t, similar to the method described by Wiersma [49]. The second check determines whether a slow drift is occurring by comparing the best-fit velocity to a given speed threshold, s. This linear regression of the best-fit velocity is taken from a 5-second buffer and is performed for each axis and for the net vector position. The last check determines if any rapid motion occur by watching for any situation where n consecutive motions appear in the same direction for any axis or if the vector distance increases in consecutive measurements. If any of these checks occur for a given step, a command for a correction is issued.

The software then attempts to calculate the correction vector by setting the destination of the plate to a negation of the measured 6D position, relative to the plate's current position, and then uses the stepping algorithm described in section 2.1.4 to convert the vector into discretized commands for the stepper motors. To avoid chasing high-frequency noise in position measurements, the monitoring algorithm is disabled once a correction command is issued to the motors, until the end of the physical correction, but to ensure that motions that occur during the correction do not cumulate to very large displacements, a time limit, T, is placed on the duration of a correction vector. Once either the time limit elapses (or once the motion completes within the time limit)), the monitoring algorithm resumes. If further correction is required, another correction command is issued.

As an initial test of functionality, the software was tested with the simulated motion

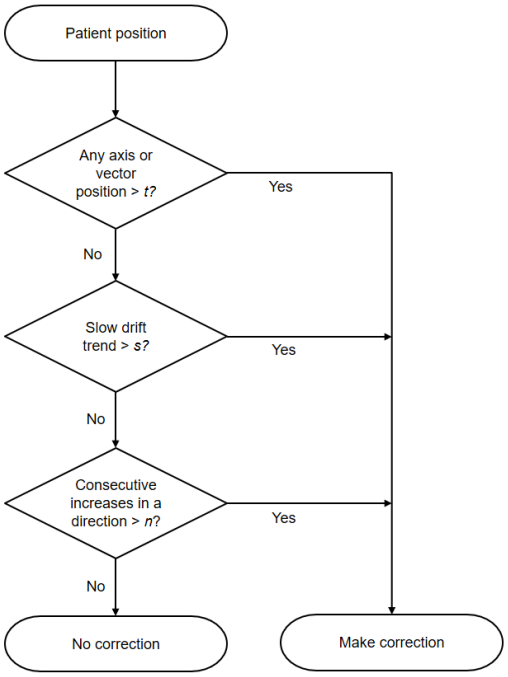


Figure 3.7: Decision flowchart for motion correction

model. The motion compensation generated real-time steps in 250 ms increments, similar to a typical refresh rate found in some optical tracking software. In contrast to the continuous nature of the Weiner process, motion monitoring software has a discrete resolution, which creates some uncertainty. To replicate this effect, the 6D position was always rounded to the nearest tenth of a millimeter or degree when sent to the monitoring algorithm. This rounding did not affect the stored 6D position of the patient.

The correction algorithm was put through trials in this simulation to ensure the end position could be acheived within slider limits. Many 15-minute simulations were performed using the parameters described in Table 3.1. The tests were performed in three stages: first, the simulations were executed with the simulated patient starting at the origin, with the pivot point set to the center of the top of the positioning platform. Next, the simulations were ran again, but with the starting position taken from a range of ± 10 mm and $\pm 3^{\circ}$, also with the pivot point in the default location. Finally, the simulations were performed again, with the same range of starting positions, but also with the pivot point taken from a range of 15 cm in the vertical direction, ± 10 cm in the longitudinal direction,

and ±10 cm in the lateral direction.

Table 3.4: Mechanical resolution of positioning plate from real-time simulation

	Starting at origin	Starting with offset	Starting with floating pivot point
Number of patients simulated	5	5	5
RMS vector error with correction (mm)	0.23±0.02	0.24±0.03	0.24±0.02
RMS vector error without correction (mm)	0.75±0.39	0.68±0.28	0.51±0.18
Correction ratio	3.26	2.84	2.14
Number of steps required in fraction	80±13	69±26	70±33
Average change in slider position (mm)	0.53±0.26	0.41±0.28	0.30±0.26

As seen in Table 3.4, the choice of starting position and choice of pivot point makes very little difference in the compensation ability. The RMS of the vector error was indistinguishable between the three cases, as was the average net change in slider position over the course of a fraction. The number of correction steps may have some slight dependency on the geometry, but because the standard deviations overlapped, no firm conclusion may be drawn. No case requried the sliders to move outside of their ranges of motion.

3.4 Conclusions about real-time motion compensation

A realistic model of intra-fraction patient motion was created to validate a motion compensation algorithm and to estimate the resolution of the mechanical system. Models of the sliders in response to realistic real-time motion have shown that the discretization of the sliders' motion are unlikely to significantly affect the accuracy of plate positioning. The motion compensation algorithm demonstrated that the mechanical system potentially can reduce RMS vector displacement by a factor of two.

Further testing with live subjects will be required to fully assess the compensation ability of the robotic system. While the simulation results show very high degrees of accuracy, the mechanical uncertainties in construction will likely reduce the overall accuracy of the mechanics. If the robotic system mechanics are verified to be accurate to the nearest 1

mm, and the expected net displacement for a typical patient is only about 1.5 mm by the end of a single fraction, the value of the robotic platform to compensate in real-time is questionable. On the other hand, commercial construction techniques could reduce the uncertainty in the mechanics of the robot and more sensitive motion validation tests could reduce the lower bound of the verified uncertainty.

Chapter 4

RF Localization System

Although the robotic system in Chapters 2 and 3 has been described as functioning in cooperation with an optical tracking system, any accurate patient positioning information may be used instead. Therefore, in the early development of the robotic system, alternate methods of tracking position were investigated. In addition to exploring conventional commercial solutions, a novel, radio-frequency (RF)-based positioning solution was conceived as well. The RF system was designed to potentially overcome many of the inherent weaknesses found in popular optical tracking systems, such as requiring line-of-sight, not being robust against changes in patient size and shape, and can usually only track a single rigid-body.

4.1 Background and description of localization system

The localization system uses an RF sensor network to identify the positions of skin-wearable transmitters on a patient (Figure 4.1). The technique presented was inspired by the global positioning system (GPS) by satellite array. In GPS technology, a series of orbiting satellites with known positions in time emit periodic pulses of RF waves containing timestamps of transmission. On the ground, portable receiving hardware compares the time of signal reception to the time of signal arrival to calculate a time difference between transmission and reception. Using the speed of light (and other correction factors),

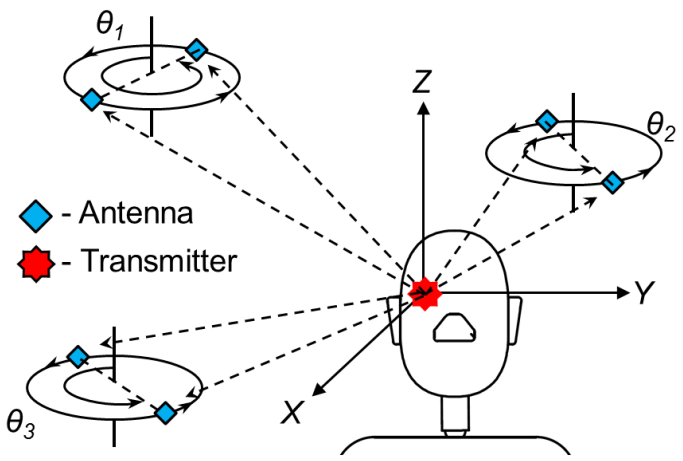
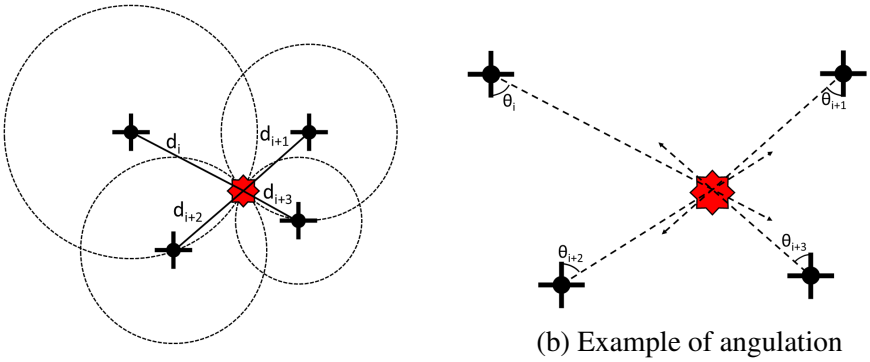


Figure 4.1: Generic configuration of multiple sensors used to localize a transmitter affixed to a patient

the distance from the receiver to each satellite may be estimated. The relative position of the receiving hardware may be calculated using lateration, which combines the estimated distances to each reference point with the known positions to the satellites to create an estimate of the position of the receiver [35]. This mathematical process finds the intersection point of several circles in a 2D plane or spheres in a 3D volume (Figure 4.2a).

The precision of RF-based lateration techniques is strongly dependent on the ability to accurately and precisely measure the times of signal transmission and reception with synchronized clocks. An RF-wave moves at the speed of light traveling about 30 meters in 1 microsecond, 30 centimeters in 1 nanosecond, and 0.3 millimeters in 1 picosecond. Therefore, to obtain distance measurements with sub-millimeter precision required for accurate clinical positioning, detection and resolution of signals would be require sub-picosecond (faster than 1THz), and would require just as precise synchronization between the transmitter and detector. While this technology currently exists, it is very expensive and not widely available.

Fortunately angulation exists as an alternative mathematical localization technique. In contrast to the distance measurements used by lateration, angulation uses the intersection of direction vectors between known points toward the point of interest to calculate the



(a) Example of lateration

(b) Example of angulation

Figure 4.2: Contrasting localization techniques

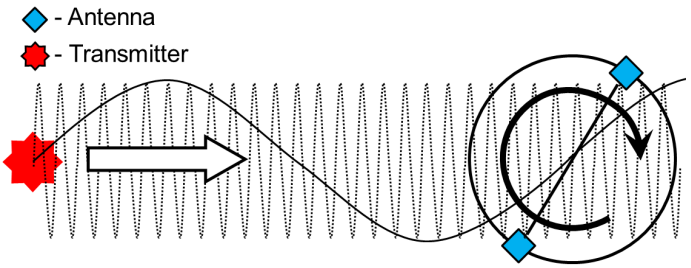


Figure 4.3: A prototypical direction of arrival interferometer prototype

relative position of that point (Figure 4.2b) [35]. In an RF-based system, a transmitting point may be localized if the direction to the transmitter could be measured by a receiving sensor (also called the *direction of arrival* at the receiver). Relative to the time-difference of arrival techniques, this method does not require explicit synchronization between the transmitter and the receiver. In addition, the technique does not require information to be sent via the transmission signal, so any signal may be used, rather than one that must be modulated to contain information.

Several approaches currently exist to measure the direction of arrival of an RF signal by comparing the phase difference of an RF wave between elements in static antenna arrays [74][75][76]. The initial envisioning of this system is similar, but with two major variations: a rotating array platform and operating at two significantly different frequencies, seen in Figure 4.3.

4.2 Monte Carlo simulation of uncertain angulation

Multiple sensors are required to localize each transmitter's signal in 3D space. Figure 4.4 displays the general layout of a practical scenario including transmitters on a patient and an array of sensors surrounded in a treatment room. Since the sensors have an inherent uncertainty in measurement, a least squares calculation is necessary to obtain a localization estimate. For this RF system to have the same utility as conventional optical systems, the localization estimates must have errors less than 1 mm, and preferably less than 0.1 mm. A Monte Carlo simulation was created to relate the uncertainty in the direction measurements to the expected error in the localization. The impact of varying the number of measurements included in the estimation was also investigated.

Because the geometries of real treatment rooms vary considerably from clinic to clinic, the distribution of sensors was chosen from a random distribution in a room-sized hemispherical shell, centered at the simulated transmitter. The polar and azimuthal coordinates of each sensor was selected in a uniform 2π space. The radial distance from the transmitter to each receiver was taken from a uniform distribution between 3.5 and 4.5 meters; the approximate size of a treatment room. The relationship between the angular resolution of the sensors and the accuracy of the position estimate was investigated by performing multiple iterations of estimation based on the same physical setup but varying the magnitude of the uncertainty in the angular measurement (Figure 4.5a).

The relationship between the number of sensors and their physical arrangement were also investigated. The initial 3D direction vectors between each sensor and transmitter was calculated as

$$\boldsymbol{V_i} = \frac{\boldsymbol{s_i} - \boldsymbol{p}}{\|\boldsymbol{s_i} - \boldsymbol{p}\|} \tag{4.1}$$

where the position $\boldsymbol{s_i}$ represents the position of the i^{th} sensor and $\boldsymbol{p}$ represents the position of the transmitter. In order to simulate uncertainty in the direction of arrival measurement spatial variation, $\boldsymbol{\Delta X_i}$ was added to the transmitter's position for direction of arrival mea-

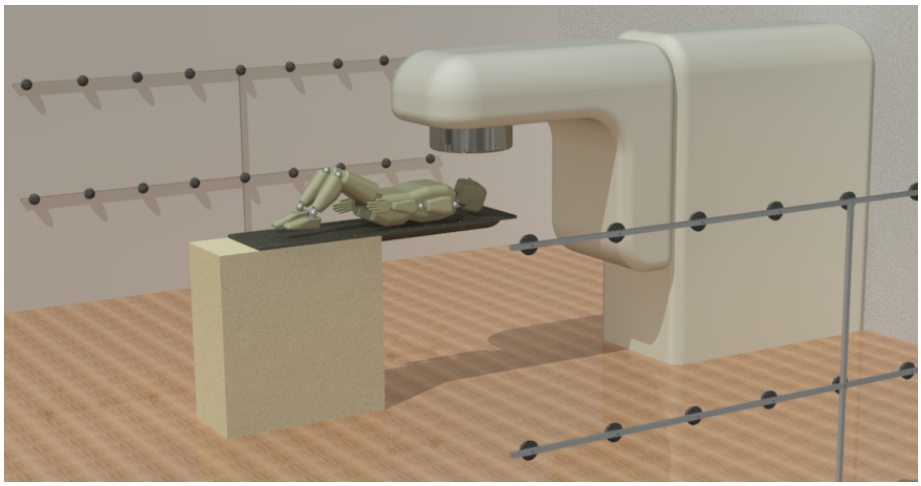


Figure 4.4: Example layout of direction of arrival sensors

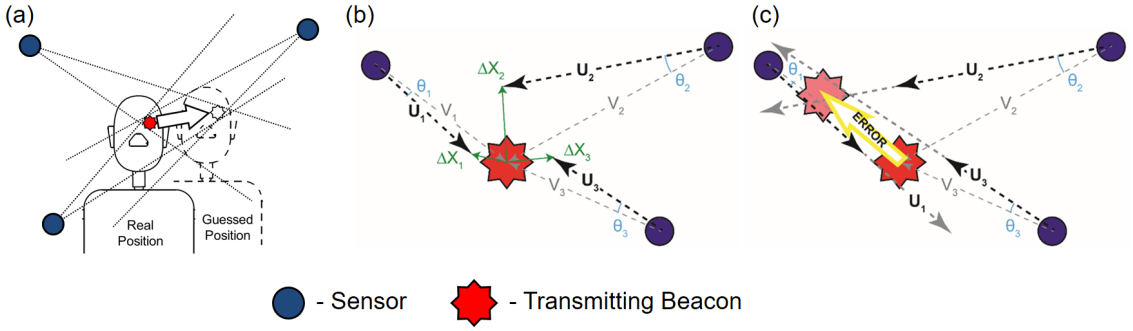


Figure 4.5: (a) Rationale for the Monte Carlo simulation, which attempts to find the relationship between uncertainty in direction of arrival measurements and the resultant error in the angulation estimation. (b) Process of the Monte Carlo simulation: angular uncertainty θ is created by adding a vector $\boldsymbol{\Delta X}$ to the true vector $\boldsymbol{V}$ between sensor and transmitter, resulting in a simulated mis-measurement of the direction vector $\boldsymbol{U}$. The vectors $\boldsymbol{\Delta X_i}$ are randomly selected from a 3D Gaussian distribution; (d) Continued process of the Monte Carlo simulation: a least squares calculation is used to find the best fit intersection of the mis-measured direction vectors, $\boldsymbol{U_i}$, and the error is taken as the difference between the intersections of the mis-measured vectors $\boldsymbol{U_i}$ and the intersection of the true vectors $\boldsymbol{V_i}$. The error is compared to the average angular uncertainty $\overline{\theta}$.

surement according to a 3D Gaussian distribution of width σ_j and direction vectors were calculated again (Figure 4.5b). Values of σ_j were varied exponentially between 1.5^2 and 1.5^{28} to observe a wide range of angular resolutions.

$$\boldsymbol{U_{ij}} = \frac{\boldsymbol{s_i} - (\boldsymbol{p} + \boldsymbol{\Delta X_{ij}}(\boldsymbol{\sigma_j}))}{\|\boldsymbol{s_i} - (\boldsymbol{p} + \boldsymbol{\Delta X_{ij}}(\boldsymbol{\sigma_j}))\|} \tag{4.2}$$

A least squares estimate of the position, $\boldsymbol{t_j}$, was calculated based on these new direction vectors for each magnitude of angular resolution σ_j.

$$\boldsymbol{t_j} = \left(\sum_{\boldsymbol{i}} (\boldsymbol{I} - \boldsymbol{U_{ij}}\boldsymbol{U_{ij}^T})\right)^{-1} \left(\sum_{\boldsymbol{i}} (\boldsymbol{I} - \boldsymbol{U_{ij}}\boldsymbol{U_{ij}^T})\boldsymbol{s_i}\right) \tag{4.3}$$

where $\boldsymbol{I}$ represents a 3×3 identity matrix. The magnitude of the error in the estimated position was then found for each level of σ_j.

$$\text{Error}_j = \|\boldsymbol{t_j} - \boldsymbol{p}\| \tag{4.4}$$

For each level of σ_j, the average angular miss was calculated by taking the dot product between the true direction vectors from the transmitter and sensors and the simulated miss.

$$\overline{\theta_j} = \overline{\cos^{-1}(\boldsymbol{U_{ij}} \cdot \boldsymbol{V_i})} \tag{4.5}$$

The mean angular miss θ serves as a surrogate for angular uncertainty and was compared to the error in estimated position for each σ_j (Figure 4.5c). This process had 5000 iterations for each σ_j to obtain statistically acceptable ranges of error for a given angular uncertainty. The number of receivers sensors in each hemisphere varied by 5, 50, and 500 in the network. This calculation was performed to observe the effect of increasing the number of vectors available for angulation analysis.

The relationship between the direction of signal arrival measurement angulation uncer-

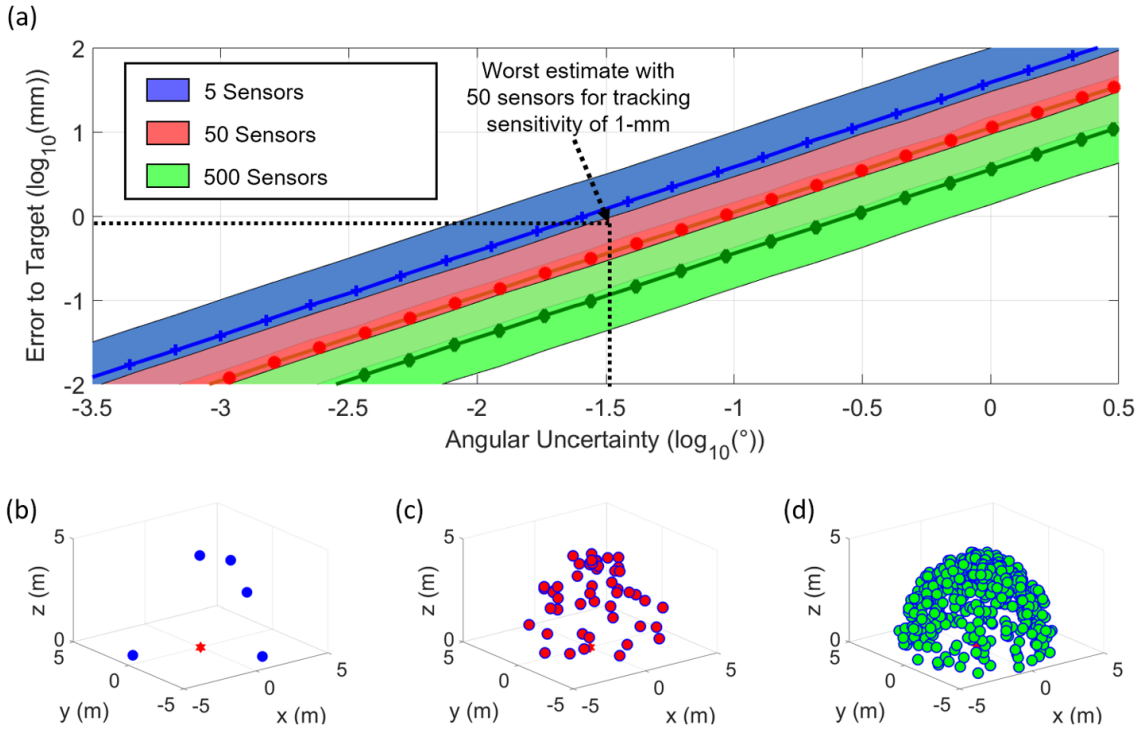


Figure 4.6: (a) Relationship between angular precision and error in least squares estimation of the transmitter position. Filled area represents two standard deviations of the Monte Carlo simulation results. Better results are closer to the bottom of the graph; therefore worst-case scenarios for a given number of sensors is the upper bound for each filled area; (b-d) Spatial distribution of sensors for the 5-sensor case (b), 50-sensor case (c), and 500-sensor case (d), respectively.

tainty of direction of signal arrival and the mean error of the position estimate presents the linear patterns (Figure 4.6a). The result shows that increased number of receivers sensors improves the position estimation for a given angular resolution, but follows a logarithmic rate of improvement. The plot in (Figure 4.6a) shows that the slopes of lines are parallel, which means that the rate of estimation improvement with respect to angular resolution depends is proportional to the number of direction estimates. This system aims to achieve tracking sensitivity less than 1 mm, considering which is about the same scale as the typical dose margins in conformal radiotherapy. Therefore, the direction of arrival sensors requires an angular resolution near $10^{-1.4}$ degrees ($\approx 0.04^{\circ}$) to achieve the ideal accuracy, if the final design includes 50 sensors. The criterion of consistency is based on maintaining 1-mm sensitivity exceeding the worst-case scenario with 50 sensors in 97% of all estimates.

4.3 Monte Carlo simulation of a dual-frequency direction of arrival sensor

The sensor uses radio interferometry to determine the direction of signal arrival. In the most simple description, the sensor consists of two antennas that rotate about a common axis. For most of the rotation, the antennas are at different distances from the transmitting beacons, so different phases of the transmitted wave arrive at the antennas at simultaneous moments in time. By careful selection of sensor dimensions and choice of signal frequency, the relative phase difference may be related to the angular direction of the signal's arrival for a given angular setting of the sensor. By selecting a frequency component that has a wavelength equal to twice the length of the sensor's diameter, the resultant interfered signal oscillates at the rate of the sensor's rotation, and the angular position of the peak corresponds to the direction of arrival.

In the preliminary attempts to simulate this sensor, the signal became difficult to in-

terpret when noise was added. Therefore, a secondary frequency was added that creates a second signal within a single rotation of the sensor. If this secondary frequency is an order of magnitude greater than the fundamental frequency, multiple oscillations appear in the received signal (Figure 4.7). With this pair of oscillation types, the fundamental frequency signal could be used as a course search for the signal peak, while the high frequency signal could be used to better identify the exact angular location of the direction of arrival.

Figure 4.3 illustrates the primary geometry of a given sensor. Each transmitter emits a sine wave, $x(t)$, for a given duration, consisting of two frequencies, (1.5 GHz and 48 GHz). These frequencies were selected as a compromise between antenna spatial constraints and available spectra. Oscillators operating at 1.5 GHz are widely available, and since the wavelength of the 1.5 GHz wave is approximately 20 cm, the sensor would require a conveniently-sized 10-cm diameter that could be placed discretely about a clinical treatment room. The 48 GHz frequency was selected because the spectra is currently not allocated by the Federal Communications Commission (47 CFR 2.106, Dec 2017 revision). As the sensor system rotates, the signals from two antennas are multiplied together and the resultant amplitude is measured.

A Monte Carlo simulation was created in MATLAB to serve as a testbed for experimenting with signal analysis techniques and to investigate the potential resolution and accuracy of hypothetical sensor. In the program, the ground truth geometric relation between the transmitter and the sensor are specified. A simulated pulse is the transmitted to the receiving antennas for a number of angular configurations of the sensor. Uniform Gaussian white noise is added to the pulse before reception to model multipath reflections and interference. Upon signal reception, the signals from each antenna are interfered and the amplitude of the interfered signal is measured and stored for that angular position of the sensor. Note that the entire wave is simulated as being received for each angular position because the electromagnetic waves oscillate at substantially higher frequencies versus any reasonable angular speed of the rotating sensor (1.5 GHz for the slow RF wave vs.

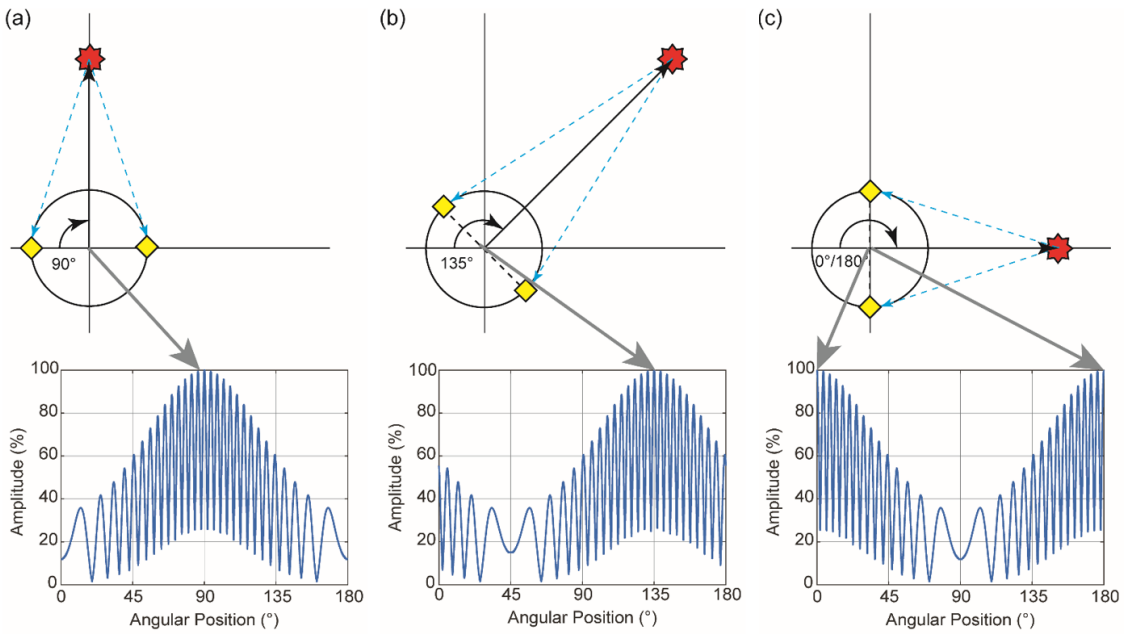


Figure 4.7: Simple example of RF-interferometer

less than 100 Hz for the rotating sensor). After an interfered measurement is obtained for a sweep of sensor angular configurations, the direction of arrival is estimated.

In mathematical terms, the following equation creates the time-dynamic transmitted wave

$$x(t) = A_1 \sin(2\pi f_1 + \phi_1) + A_2 \sin(2\pi f_2 + \phi_2) \tag{4.6}$$

where A is the amplitude of each of the two waves, f_1 and f_2 are the frequency components of each wave, t is the sampled point in time, and ϕ is the initial phase of each wave. The sine wave was sampled at 150 THz over a 33 nS pulse, corresponding to 25 wavelengths of the lower frequency (2.5×10^5 points). The starting phases of the wave components are chosen from a random uniform distribution between 0 and 2π, and the amplitudes of both waves are assumed as equal in strength. The wavelength of the lower frequency component is calculated according to the speed of light in air, and termed as the *carrier wavelength*, λ_1. The carrier wavelength is later used for calculating the phase offset at signal reception.

Vector analysis defines the location of the transmitter as point $\boldsymbol{t}$ in 3D space. The

center of a given sensor is defined as point $\boldsymbol{s}$, and the angular position of the j^{th} antenna in the plane of the sensor's rotation is defined as θ_j. The vector $\boldsymbol{V}$ between the transmitter's location and the center of the sensor is defined as

$$\boldsymbol{V} = \boldsymbol{s} - \boldsymbol{t} \tag{4.7}$$

The vectors from the center of the sensor to the j^{th} antenna receiver is then calculated for i angular positions, for a given sensor radius R

$$\boldsymbol{C_{ij}} = R \begin{bmatrix} \cos\theta_{ij} & \sin\theta_{ij} & 0 \end{bmatrix} \tag{4.8}$$

The distance between the transmitter and each receiving antenna at every sampled angular position is then found as the magnitude of the vector connecting between the transmitter and each receiver

$$d_{ij} = \|\boldsymbol{V} + \boldsymbol{C_{ij}}\| \tag{4.9}$$

This distance is then converted to fractions of the carrier wavelength, m_{ij}.

$$m_{ij} = \frac{d_{ij}}{\lambda_1} \tag{4.10}$$

At each angular position i, each receiver reads the transmitted signal after the respective number of wavelengths of the transmitted signal, $m_{ij} + M$, for n wavelengths, where M represents the minimum number of wavelengths to wait before the signal is read, ensuring that receiving antennas did not attempt to read the signal before the first index in the transmitted signal. M = 5 and n = 10 were used in the simulation. Effectively, this forces the received wave at each antenna to begin with a $\Delta\phi_{ij}$, such that

$$\Delta\phi_{ij} = d_{ij} \mod \lambda_1 \tag{4.11}$$

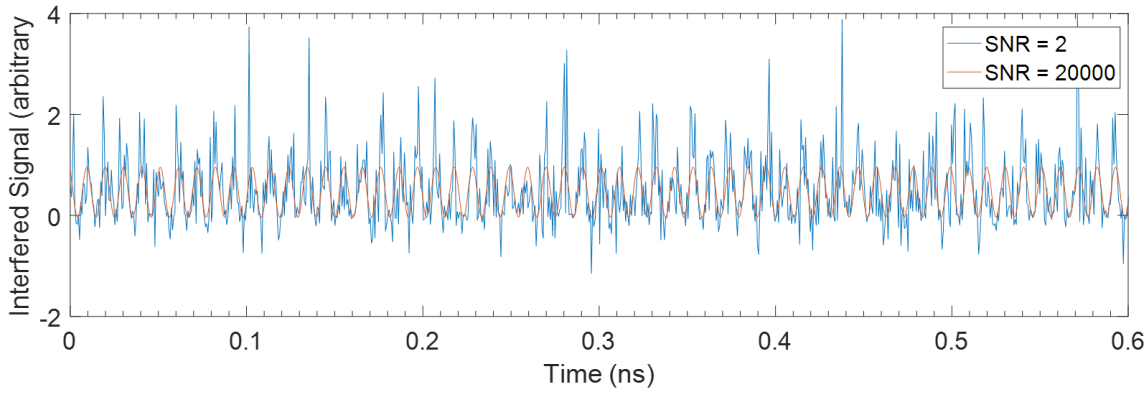


Figure 4.8: Example of interfered signal for dual frequency sensor for one angular measurement

Simulated noise was added at the moment of reception with an estimated signal to noise (SNR) of 2.

$$S(\Delta\phi_{ij}) = x(t_{read}, \Delta\phi_{ij}) + \frac{X_{ij}(t_{read})}{SNR} \tag{4.12}$$

Where $X_{ij}(t_{read})$ is a random number from a normal distribution of $\mu = 0$ and $\sigma = 1$ sampled for each receiving antenna at each angular position of the sensor. Note that the time indicies change at this step, representing the truncated read from the original signal. The signals are then multiplied and the resultant amplitude is recorded against each sensor angle θ_i, (Figure 4.8). A median filter is applied to smooth the recorded data.

$$I_i = \text{median filter}|\max\left(S(\Delta\phi_{i1}) * S(\Delta\phi_{i2})\right)| \tag{4.13}$$

A peak-finding algorithm is used to identify the angular position corresponding to the signal peaks. This process was repeated over 5000 iterations in a 4° search space of the simulated transmitter's known position to judge the accuracy of the direction finding algorithm.

Figure 4.9 shows simulated results of the direction of signal arrival. The measured data demonstrates that the product of signals from two rotating receivers antennas in the external system forms a signal that may be easily identified. If there were only the fun-

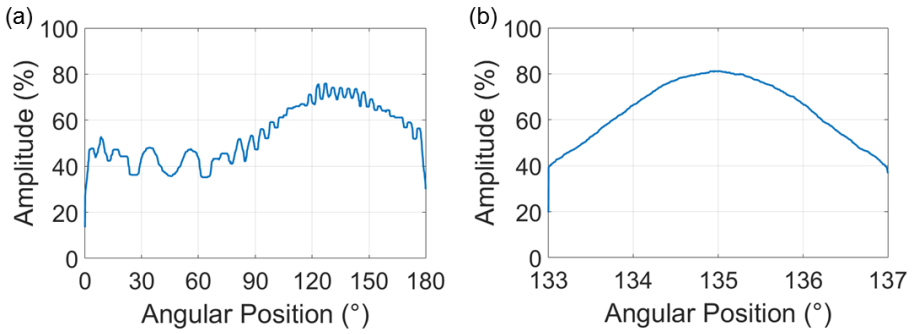


Figure 4.9: Simulated results of dual frequency direction of arrival sensor, showing (a), the entire half rotation search, and (b) the narrow search space. The ground truth direction was at 135°.

damental frequency, the signal would rise from a minimal value when the antennas are in line with the signal path to a maximum value when the vector between the antennas is perpendicular to the signal direction. The signal then returns to a minimal value at 180°. The pattern repeats every 180° because the sensor has 2-fold rotational symmetry. The interference between the mixed-frequency signals (1.5 + 48 GHz) creates a pattern as the antenna array platform rotates. In a noise-free environment, the angular locations of maximum constructive and destructive interference would correspond to the direction of signal arrival, and thus, must be found by further analysis. Using the dual-frequency paradigm, the number of peaks in a complete sensor rotation is known based on the approximate geometry and the ratio of the component frequencies. Therefore, the peak of maximum interference may be used to identify the correct peak in a coarse search with a peak-finding algorithm. Once the peak of maximum interference is identified, a narrow angular search may begin (repeating the algorithm in a 4° window, compared to a 180° window). The center of the peak is found with the same peak-finding algorithm.

Based on 5,000 iterations of this process with a given spatial configuration, the program found the direction of the transmitter within $0.01° \pm 0.07°$ compared to ground-truth. Comparing this result to the results of the least-squares simulation shown in Figure 4.6, the 2σ confidence interval for the expected error of the localization is on the order of 1 cm if 5 sensors are used, and on the order of 1 mm if 500 sensors are used. This error is

not quite within the clinically acceptable range, but further improvements to the technique could be made that could push the uncertainty to within an acceptable range. Note that while the average was on the order of 0.01°, the standard deviation was almost an order of magnitude greater, indicating that better estimates of the actual direction of arrival (and therefore, also localization) could be obtained with iterative measurements and averaging.

4.4 Monte Carlo simulation of single-frequency direction of arrival sensor

For many reasons (such as cost or hardware simplicity), it was desirable for the direction of arrival sensor to operate using only the 1.5 GHz frequency, rather than operating in a dual-frequency mode. After recognizing several areas for improvement in the signal analysis software, an alternate configuration was realized that did not need to include the secondary frequency. In simulation, the wave was again simulated as a sinusoid wave y of 1.5 GHz, with a sampling rate of 3 THz

$$y(t) = A\sin\left(2\pi(ft+\phi)\right) \tag{4.14}$$

The process of simulating signal reception at a distance is identical to the dual frequency simulation, as is found in equations 4.7 though 4.12. However, upon simulating the interference of the signals, the signal analysis technique is substantially changed.

Rather than multiplying the two received waveforms together, they are instead added. This mathematical process represents creating a superposition of the two waveforms, as could be done with a physical RF mixer. This better represents the likely physical hardware path for interfering two analog signals compared to pure mathematical operations between two discretized signals used in simulation. The method of measuring the amplitude of the interfered signal was also improved. In the dual frequency formulation, the amplitude of

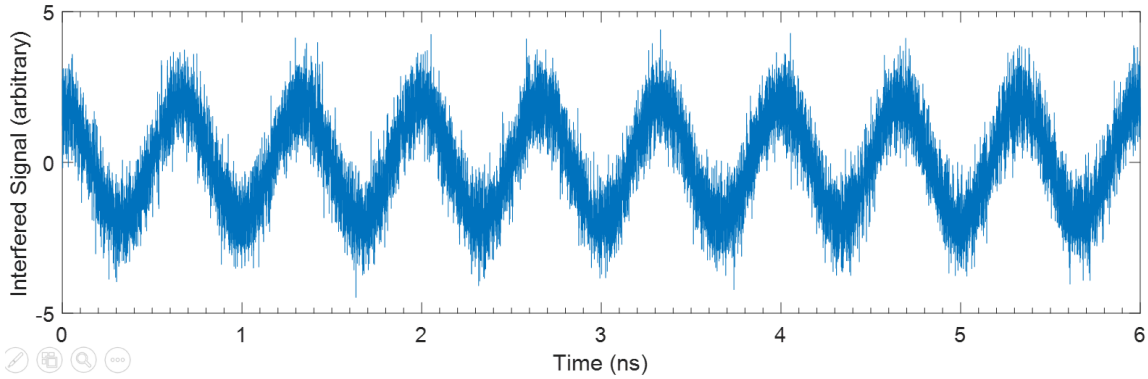


Figure 4.10: Example of interfered signal for single frequency sensor for one angular measurement

the interfered signal was measured by taking the signal maximum. This process led to noisy results when multipath noise was added to the received signal, and really was not a good way to measure amplitude. Instead, a fast Fourier transform applied to the interfered signal I produces a much better the amplitude of the interefered signal after filtering out the noise. The maximum response of the FFT is assumed to be the transmitted frequency.

$$|I_i| = \max(FFT\left(S(\Delta\phi_{i1}) + S(\Delta\phi_{i2})\right)) \tag{4.15}$$

The $|I_i|$ terms are then organized into a single $|I|$ array according to the swept phase positions of the sensor.

In the dual frequency paradigm, the direction of arrival was found using MATLAB's built-in peak finding algorithm on the narrow search of the maximum interference peak. This method was problematic for a few reasons. First, the algorithm could only find peaks that corresponded to the discretely searched angular locations. Since true direction of arrival may be not lie so conveniently on one of these points in angular space, the true direction could only be found through iterative searching and taking an average of the estimates. Secondly, the algorithm was very sensitive to noise, as it located the peaks by identifying the most prominent points for a given metric of prominence. An ideal technique would instead eliminate these two weaknesses: requiring fewer iterations and

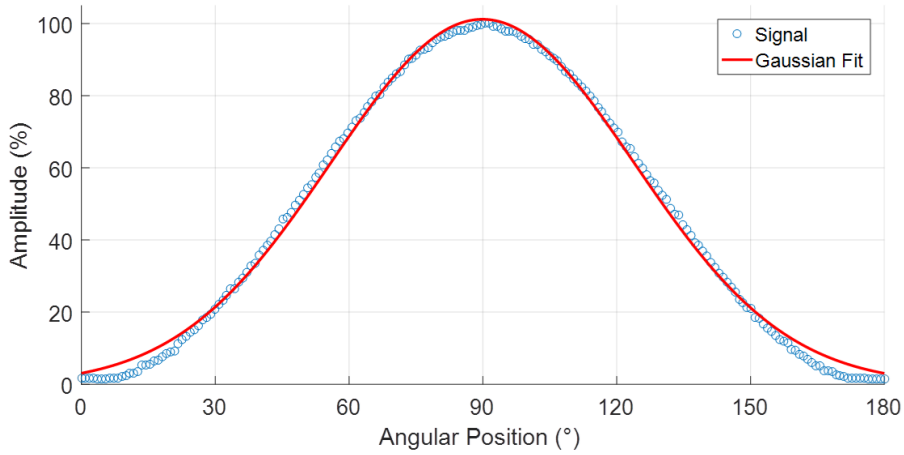


Figure 4.11: Simulated results of single frequency direction of arrival sensor for a ground truth direction was at 90°.

experiencing less variation in response to noise.

If only a single frequency is used, then only one oscillation from minima to maxima occurs in a half rotation, and the peak may instead be found by an appropriate fitting function, such as a Gaussian (Figure 4.11).

$$f(\theta) = A\mathrm{e}^{\left(\frac{\theta-\mu}{\sigma}\right)^2} \tag{4.16}$$

The drawback of using a this fitting technique is that the difference between μ and the true direction of arrival can vary significantly if the data is significantly offset from 90°; when the oscillation begins to wrap around. Thus, to improve the fit for a wide search range (>90°), a moving exclusion window was used to filter out signal response more than ±45° away from a rough estimate of the peak location (found with the find peaks function).

Compared with the dual frequency model, the single frequency system showed comparable accuracy and significantly better precision. For the same geometric setup and SNR, the average error was 0.017° ± 0.021°. Using Figure 4.6 the estimated localization error will range between about 1 mm if a few sensors are used and about 0.100 mm if a few hundred sensors are used, demonstrating clear clinical acceptability.

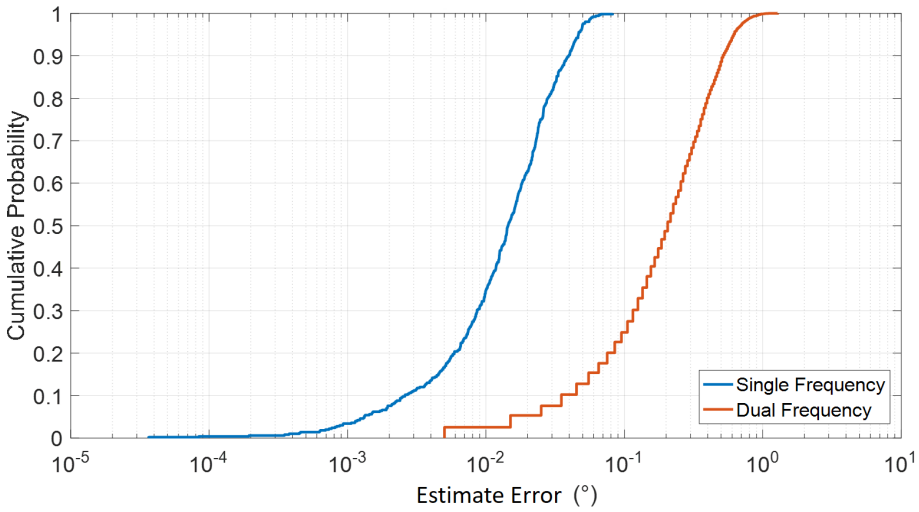


Figure 4.12: Comparison between single frequency and dual frequency accuracy

4.5 Conclusions about the RF-based localization sensors

In a clinical environment, RF-based tracking equipment is largely untested. Potential RF reflectors include radiotherapy linear accelerators and simulation CT scanners. This introduces a RF localization system for accurate tracking of patient motion in radiotherapy, which would provide high accuracy and robustness even in poor SNR setting even in situations where the true signal may be difficult to distinguish from the noise using RF waves. This system also calls for the use of components that operate in the *K*-band of the electromagnetic spectrum, which are not widely available in normal commercial channels and less investigated in scientific and engineering literature. RF waves in this spectral range may have different propagation characteristics, which could introduce multi-path noise.

Three parameters can reduce the error in the positioning estimation: improving the angular resolution, increasing the number of receivers sensors, and reducing the distance between the each transmitter on a patient and receiver in the external system. Due to practicality reasons, the total number of receivers sensors is constrained to be less than 100, and the receiver-transmitter distance is constrained by the treatment room geometry. Therefore, a great amount of effort has been focused into methods that can improve the

angular resolution of direction of arrival analysis software. Based on standard room sizes and assumption that the about 50 receivers sensors will be used, an angular resolution near 0.04° is necessary to achieve position estimate errors less than 1 mm. Optimistically, achieving the resolution of 0.01° is possible with better peak detection algorithms, even in poor SNR environments.

However, despite the estimated accuracy, several challenges remain which could limit the practicality of this system. First, the direction-finding prototype requires very accurate knowledge of the wavelengths in use, assumes that the wavelengths are always constant, and assumes that the waves have traveled in exactly straight lines. In reality, when electromagnetic radiation propagates through material, the wavelength of the wave is changed by the index of refraction, (even assuming attenuation is negligible).

$$n_{medium} = \frac{\lambda_{vac}}{\lambda_{med}} = \{\mu\varepsilon(\omega)\} \tag{4.17}$$

Tissue is a complex, composite structure, and the electric susceptibility, χ_e, of tissue along any straight-line vector path is nonzero.

$$\chi_e(\omega) = \frac{\varepsilon(\omega)}{\varepsilon_0} - 1 \tag{4.18}$$

Therefore, as any RF waves travel through tissue, they scatter, refract, and most importantly, undergo many changes in wavelength, which cannot be easily predicted or modeled without explicit knowledge of the position and composition of the tissue in question. The sensors explicitly use interferometry of simultaneously received amplitudes to calculate the direction of arrival. If these amplitudes are shifted in receiving time by changes in wave properties along the way by even a partial oscillation ($\approx$1 ps) the direction estimate will be extremely inaccurate. Therefore, this technology is expected to have best accuracy within line-of-sight conditions.

However, other challenges also exist. The requirement simultaneous comparison of

signals requires very precise electronics manufacturing to ensure that the signals arrive at the mixer within approximately 1 fs of each other. Although construction may be possible in the current era, they are beyond the scope of this research. Based on these challenges, this technology was deemed to have lost its most significant advantage over optical techniques.

Chapter 5

Conclusions

The work presented in this dissertation has demonstrated the feasibility of plastic mechanical positioning systems for use in radiotherapy. This research sought two specific aims: to develop, construct, and evaluate a radiotherapy-friendly system capable of accurate mechanical positioning in 6D space, and to use the developed system to investigate whether independent setup of the body and head produces superior setup alignment compared to single-body positioning. The process of pursuing the first aim required creating and refining a framework for design, construction, control, and evaluation. The key milestones required in this development are listed below:

- Identification of project goals and constraints
- Creation of forward and/or inverse kinematic models
- Material selection
- Estimation of radiographic properties
- Structural strength analysis (including behavior with radiation dose)
- Power analysis
- Physical design

- Electronic control design (including redundant guidance control)
- Control software design
- Validation of radiographic properties
- Validation of strength analysis
- Validation of mechanics
- Validation of purpose

These independent analysis together demonstrate fulfillment of the first specific aim. Once the fully-functional system was constructed and evaluated, the second objective could be fulfilled. While the second objective did not require the mechanical system to verify (since independent alignment can always be achieved by manual alignment) our measurements are evidence of veracity for our hypothesis that independent setup of the head and body ensures accurate alignment of the neck. The developed mechanical system is important because it significantly simplifies the process of achieving this superior alignment by providing means of remote manipulation without significantly compromising the tools used in typical treatment workflows.

Future Work

Now that the basic functionality of the prototype system has been demonstrated, future efforts will be focused on optimization and *in vivo* testing. As stated in Chapter 2, the height of the third generation prototype could create clearance issues with linear accelerator gantries if used with treatment of real patients. Therefore, revised versions will be designed that aim to reduce the height without sacrificing performance.

Significant work also remains for finding the optimal method of mechanically registering a patient's head with the positioning platform. In phantom testing, the head was

secured with a re-purposed rubberized ring originally meant as a simple handhold for patients during treatment. This method seemed in line with that described by Wiersma, who argued that with a feedback loop system, open geometry without a mask was possible because gravity and friction forces suppressed large amplitude motion, while the feedback loop-driven mechanical system worked to reduce small amplitude motion [77]. Our method could be improved by indexing the headrest to the positioning plate for repeatable setup. This process of headrest optimization could also provide a chance to optimize the weight distribution of the head onto the positioning platform.

In vivo testing is also necessary to completely validate the system's ability to accurately set up a human subject before live patients may be used. Although our anthropomorphic phantom contained similar bony anatomy, it lacks soft and connective tissue, which could change the force requirements necessary for repositioning a human subject. In a real neck, elastic forces present in tendons, muscles, nerves, and blood vessels could affect the mechanical strength necessary to move the head as an entire anatomic assembly.

Bibliography

1. Amelio, D. *et al.* Analysis of inter- and intrafraction accuracy of a commercial thermoplastic mask system used for image-guided particle radiation therapy. *Journal of Radiation Research* **54,** 69–76. ISSN: 0449-3060 (2013).

2. Kang, H. J., Grelewicz, Z. & Wiersma, R. D. Development of an automated region of interest selection method for 3D surface monitoring of head motion. *Medical Physics* **39,** 3270–3282. ISSN: 0094-2405 (2012).

3. Chen, A. M. *et al.* Utility of daily image guidance with intensity-modulated radiotherapy for tumors of the base of skull. *Head Neck* **34,** 763–70. ISSN: 1097-0347 (Electronic) 1043-3074 (Linking) (2012).

4. Zhang, L. *et al.* Multiple regions-of-interest analysis of setup uncertainties for head-and-neck cancer radiotherapy. *Int J Radiat Oncol Biol Phys* **64,** 1559–69. ISSN: 0360-3016 (Print) 0360-3016 (Linking) (2006).

5. Sharp, L. *et al.* Randomized trial on two types of thermoplastic masks for patient immobilization during radiation therapy for head-and-neck cancer. *Int J Radiat Oncol Biol Phys* **61,** 250–6. ISSN: 0360-3016 (Print) 0360-3016 (Linking) (2005).

6. Polat, B., Wilbert, J., Baier, K., Flentje, M. & Guckenberger, M. Nonrigid patient setup errors in the head-and-neck region. *Strahlenther Onkol* **183,** 506–11. ISSN: 0179-7158 (Print) 0179-7158 (Linking) (2007).

7. Ove, R., Cavalieri, R., Noble, D. & Russo, S. M. Variation of neck position with image-guided radiotherapy for head and neck cancer. *Am J Clin Oncol* **35,** 1–5. ISSN: 1537-453X (Electronic) 0277-3732 (Linking) (2012).

8. Graff, P. *et al.* The residual setup errors of different IGRT alignment procedures for head and neck IMRT and the resulting dosimetric impact. *Int J Radiat Oncol Biol Phys* **86,** 170–6. ISSN: 1879-355X (Electronic) 0360-3016 (Linking) (2013).

9. Piotrowski, T. *et al.* Impact of the spinal cord position uncertainty on the dose received during head and neck helical tomotherapy. *J Med Imaging Radiat Oncol* **57,** 503–11. ISSN: 1754-9485 (Electronic) 1754-9477 (Linking) (2013).

10. Djordjevic, M., Sjoholm, E., Tullgren, O. & Sorcini, B. Assessment of residual setup errors for anatomical sub-structures in image-guided head-and-neck cancer radiotherapy. *Acta Oncol* **53,** 646–53. ISSN: 1651-226X (Electronic) 0284-186X (Linking) (2014).

11. Park, E. T. & Park, S. K. Setup uncertainties for inter-fractional head and neck cancer in radiotherapy. *Oncotarget* **7,** 46662–46667. ISSN: 1949-2553 (Electronic) 1949-2553 (Linking) (2016).

12. Cheo, T., Loh, Y., Chen, D., Lee, K. M. & Tham, I. Measuring radiotherapy setup errors at multiple neck levels in nasopharyngeal cancer (NPC): A case for differential PTV expansion. *Radiother Oncol* **117,** 419–24. ISSN: 1879-0887 (Electronic) 0167-8140 (Linking) (2015).

13. Kim, S., Jin, H., Yang, H. & Amdur, R. J. A study on target positioning error and its impact on dose variation in image-guided stereotactic body radiotherapy for the spine. *Int J Radiat Oncol Biol Phys* **73,** 1574–9. ISSN: 1879-355X (Electronic) 0360-3016 (Linking) (2009).

14. Murphy, M. J. *et al.* Patterns of patient movement during frameless image-guided radiosurgery. *Int J Radiat Oncol Biol Phys* **55,** 1400–8. ISSN: 0360-3016 (Print) 0360-3016 (Linking) (2003).

15. Linthout, N., Verellen, D., Tournel, K. & Storme, G. Six dimensional analysis with daily stereoscopic x-ray imaging of intrafraction patient motion in head and neck treatments using five points fixation masks. *Medical Physics* **33,** 504–513 (2006).

16. Van Asselen, B., Raaijmakers, C. P., Lagendijk, J. J. & Terhaard, C. H. Intrafraction motions of the larynx during radiotherapy. *Int J Radiat Oncol Biol Phys* **56,** 384–90. ISSN: 0360-3016 (Print) 0360-3016 (Linking) (2003).

17. Perera, T., Moseley, J. & Munro, P. Subjectivity in interpretation of portal films. *Int J Radiat Oncol Biol Phys* **45,** 529–34. ISSN: 0360-3016 (Print) 0360-3016 (Linking) (1999).

18. Bressan, V. *et al.* The effects of swallowing disorders, dysgeusia, oral mucositis and xerostomia on nutritional status, oral intake and weight loss in head and neck cancer patients: A systematic review. *Cancer Treat Rev* **45,** 105–19. ISSN: 1532-1967 (Electronic) 0305-7372 (Linking) (2016).

19. Georg, D., Knoos, T. & McClean, B. Current status and future perspective of flattening filter free photon beams. *Med Phys* **38,** 1280–93. ISSN: 0094-2405 (Print) 0094-2405 (Linking) (2011).

20. Vassilev, O. *et al.* Dosimetric properties of photon beams from a flattening filter free clinical accelerator. *Phys Med Biol* **51,** 1907–17. ISSN: 0031-9155 (Print) 0031-9155 (Linking) (2006).

21. Chi, P. C., Balter, P., Luo, D., Mohan, R. & Pan, T. Relation of external surface to internal tumor motion studied with cine CT. *Med Phys* **33,** 3116–23. ISSN: 0094-2405 (Print) 0094-2405 (Linking) (2006).

22. Fassi, A. *et al.* Tumor tracking method based on a deformable 4D CT breathing motion model driven by an external surface surrogate. *Int J Radiat Oncol Biol Phys* **88,** 182–8. ISSN: 1879-355X (Electronic) 0360-3016 (Linking) (2014).

23. Berbeco, R. I., Nishioka, S., Shirato, H., Chen, G. T. & Jiang, S. B. Residual motion of lung tumours in gated radiotherapy with external respiratory surrogates. *Phys Med Biol* **50,** 3655–67. ISSN: 0031-9155 (Print) 0031-9155 (Linking) (2005).

24. Kupelian, P. *et al.* Multi-institutional clinical experience with the Calypso System in localization and continuous, real-time monitoring of the prostate gland during external radiotherapy. *Int J Radiat Oncol Biol Phys* **67,** 1088–98. ISSN: 0360-3016 (Print) 0360-3016 (Linking) (2007).

25. Gopan, O. & Wu, Q. Evaluation of the accuracy of a 3D surface imaging system for patient setup in head and neck cancer radiotherapy. *Int J Radiat Oncol Biol Phys* **84,** 547–52. ISSN: 1879-355X (Electronic) 0360-3016 (Linking) (2012).

26. Peng, J. L. *et al.* Characterization of a real-time surface image-guided stereotactic positioning system. *Medical Physics* **37,** 5421–5433. ISSN: 0094-2405 (2010).

27. Stieler, F., Wenz, F., Shi, M. & Lohr, F. A novel surface imaging system for patient positioning and surveillance during radiotherapy. A phantom study and clinical evaluation. *Strahlenther Onkol* **189,** 938–44. ISSN: 1439-099X (Electronic) 0179-7158 (Linking) (2013).

28. Bert, C., Metheany, K. G., Doppke, K. & Chen, G. T. A phantom evaluation of a stereo-vision surface imaging system for radiotherapy patient setup. *Med Phys* **32,** 2753–62. ISSN: 0094-2405 (Print) 0094-2405 (Linking) (2005).

29. Bert, C. *et al.* Clinical experience with a 3D surface patient setup system for alignment of partial-breast irradiation patients. *Int J Radiat Oncol Biol Phys* **64,** 1265–74. ISSN: 0360-3016 (Print) 0360-3016 (Linking) (2006).

30. Keall, P. J. *et al.* The first clinical treatment with kilovoltage intrafraction monitoring (KIM): a real-time image guidance method. *Med Phys* **42,** 354–8. ISSN: 2473-4209 (Electronic) 0094-2405 (Linking) (2015).

31. Farahmand, F., Khadivi, K. O. & Rodrigues, J. J. Detecting intra-fraction motion in patients undergoing radiation treatment using a low-cost wireless accelerometer. *Sensors (Basel)* **9,** 6715–29. ISSN: 1424-8220 (Print) 1424-8220 (Linking) (2009).

32. Inata, H. *et al.* Development of a real-time monitoring system for intra-fractional motion in intracranial treatment using pressure sensors. *Physics in Medicine and Biology* **60,** 7229–7243. ISSN: 0031-9155 (2015).

33. McRobbie, D., Moore, E., Graves, M. & Prince, M. *MRI: From Picture to Proton* (Cambridge University Press, Cambridge, UK, 2007).

34. Bushberg, J., Seibert, J. A. & Leidholt Edwinand Boone, J. *The Essential Physics of Medical Imaging* Third Edition (Lippincott Williams and Wilkins, Philadelphia, Pennsylvania, 2012).

35. Dardari, D., Falletti, E. & Luise, M. *Satellite and Terrestrial Radio Positioning Techniques: A Signal Processing Perspective* (Elsevier Ltd., Oxford, 2012).

36. Willoughby, T. R. *et al.* Target localization and real-time tracking using the Calypso 4D localization system in patients with localized prostate cancer. *Int J Radiat Oncol Biol Phys* **65,** 528–34. ISSN: 0360-3016 (Print) 0360-3016 (Linking) (2006).

37. Van Herk, M., Remeijer, P., Rasch, C. & Lebesque, J. V. The probability of correct target dosage: dose-population histograms for deriving treatment margins in radiotherapy. *Int J Radiat Oncol Biol Phys* **47,** 1121–35. ISSN: 0360-3016 (Print) 0360-3016 (Linking) (2000).

38. Alpen, E. *Radiation Biophysics* Second Edition (Academic Press, San Diego, California, 1990).

39. Hall, E. & Giaccia, A. *Radiobiology for the Radiologist* Seventh Edition (Lippincott Williams and Wilkins, Philadelphia, Pennsylvania, 2012).

40. Zia, S. *et al.* The impact of weight loss on setup accuracy for head and neck cancer patients in the era of image guided radiation therapy. *J Radiat Oncol* **5,** 359–362. ISSN: 1948-7894 (2016).

41. Hou, W. H., Wang, C. W., Tsai, C. L., Hsu, F. M. & Cheng, J. C. The ratio of weight loss to planning target volume significantly impacts setup errors in nasopharyngeal cancer patients undergoing helical tomotherapy with daily megavoltage computed tomography. *Radiol Oncol* **50,** 427–432. ISSN: 1318-2099 (Print) 1318-2099 (Linking) (2016).

42. De Bari, B. *et al.* Does weight loss predict accuracy of setup in head and neck cancer patients treated with Intensity-Modulated Radiation Therapy? *Radiol Med* **117,** 885–91. ISSN: 1826-6983 (Electronic) 0033-8362 (Linking) (2012).

43. Seppenwoolde, Y., Berbeco, R. I., Nishioka, S., Shirato, H. & Heijmen, B. Accuracy of tumor motion compensation algorithm from a robotic respiratory tracking system: a simulation study. *Med Phys* **34,** 2774–84. ISSN: 0094-2405 (Print) 0094-2405 (Linking) (2007).

44. Haas, O. C. *et al.* Couch-based motion compensation: modelling, simulation and real-time experiments. *Phys Med Biol* **57,** 5787–807. ISSN: 1361-6560 (Electronic) 0031-9155 (Linking) (2012).

45. Hermann, C., Ma, L., Wilbert, J., Baier, K. & Schilling, K. Control of a HexaPOD treatment couch for robot-assisted radiotherapy. *Biomed Tech (Berl)* **57,** 333–51. ISSN: 1862-278X (Electronic) 0013-5585 (Linking) (2012).

46. Mancosu, P. *et al.* Are pitch and roll compensations required in all pathologies? A data analysis of 2945 fractions. *Br J Radiol* **88,** 20150468. ISSN: 1748-880X (Electronic) 0007-1285 (Linking) (2015).

47. Linthout, N. *et al.* Assessment of secondary patient motion induced by automated couch movement during on-line 6 dimensional repositioning in prostate cancer treatment. *Radiother Oncol* **83,** 168–74. ISSN: 0167-8140 (Print) 0167-8140 (Linking) (2007).

48. Lee, S. *et al.* Evaluation of mechanical accuracy for couch-based tracking system (CBTS). *J Appl Clin Med Phys* **13,** 3818. ISSN: 1526-9914 (Electronic) 1526-9914 (Linking) (2012).

49. Wiersma, R. D., Wen, Z., Sadinski, M. & Kang, H. Development of Frameless SRS using Real-Time 6D Facial Surface Monitoring and Continuous Adaptive Head Motion Correction. *International Journal of Radiation Oncology Biology Physics* **78,** S793–S793. ISSN: 0360-3016 (2010).

50. Liu, X., Belcher, A. H., Grelewicz, Z. & Wiersma, R. D. Robotic real-time translational and rotational head motion correction during frameless stereotactic radiosurgery. *Med Phys* **42,** 2757–63. ISSN: 0094-2405 (Print) 0094-2405 (Linking) (2015).

51. Belcher, A. H. & Wiersma, R. D. Simulation and Design of a Real-Time 6D Head Motion Compensation Platform Based On a Stewart Platform Approach. *Medical Physics* **39,** 3688–3689. ISSN: 0094-2405 (2012).

52. Van Kranen, S., van Beek, S., Rasch, C., van Herk, M. & Sonke, J. J. Setup uncertainties of anatomical sub-regions in head-and-neck cancer patients after offline CBCT guidance. *Int J Radiat Oncol Biol Phys* **73,** 1566–73. ISSN: 1879-355X (Electronic) 0360-3016 (Linking) (2009).

53. Fonesca, G. *et al.* HDR 192Ir source speed measurements using a high speed video camera. *Med Phys* **42,** 412–115. ISSN: 0094-2405 (Print) 0094-2405 (Linking) (1 2015).

54. Gevaert, T. *et al.* Setup accuracy of the Eovalis ExacTrac 6DOF system for frameless radiosurgey. *International Journal of Radiation Oncology Biology Physics* **82,** 1627–1635. ISSN: 0360-3016 (2012).

55. Hassan, M. M. Mechanical, thermal, and morphological behavior of the polyamide 6/acrylonitrile-butadiene-styrene blends irradiated with gamma rays. *Polym Eng Sci* **48,** 373–80 (2008).

56. Park, S., Park, C., Kim, J. & Kim, T. Derivation of Fatigue Properties of Plastics and Life Prediction for Plastic Parts. *The 2015 World Congress on Advances in Civil, Environmental, and Materials Research* (2015).

57. Stanisic, M. *Mechanisms and Machines: Kinematics, Dynamics, and Synthesis* (Cengage Learning, Stamford, Connecticut, 2015).

58. *PLA* <http://www.rsc.org/Merck-Index/monograph/m6656/>.

59. *ABS Plastic* <http://pubchem.ncbi.nlm.nih.gov/compound/24756>.

60. *Nylon-66* <http://pubchem.ncbi.nlm.nih.gov/compound/36070>.

61. *Epoxy Resin* <http://pubchem.ncbi.nlm.nih.gov/compound/169944>.

62. *Formaldehyde* <http://www.rsc.org/Merck-Index/monograph/m5533/>.

63. *PMMA* <http://pubchem.ncbi.nlm.nih.gov/compound/6658>.

64. *Ethene, 1,1,2,2-tetrafluoro-,homopolymer* <http://webbook.nist.gov/cgi/cbook.cgi?ID=9002-84-0>.

65. *Water* <http://www.rsc.org/Merck-Index/monograph/m11507/>.

66. *Earth Fact Sheet* <http://nssdc.gsfc.nasa.gov/planetary/factsheet/earthfact.html>.

67. Attix, F. *Introduction to Radiological Physics and Radiation Dosimetry* (John Wiley and Sons, Inc, United States of America, 1986).

68. Gear, J. I. *et al.* Abdo-Man: a 3D-printed anthropomorphic phantom for validating quantitative SIRT. *EJNMMI Phys* **3,** 17. ISSN: 2197-7364 (Print) 2197-7364 (Linking) (2016).

69. Maier, C. Design Guides for Plastics. *Plastics and Rubbers Weekly* (2009).

70. Ballhausen, H., Li, M., Hegemann, N. S., Ganswindt, U. & Belka, C. Intra-fraction motion of the prostate is a random walk. *Phys Med Biol* **60,** 549–63. ISSN: 1361-6560 (Electronic) 0031-9155 (Linking) (2015).

71. Venkatesh, S. *The Theory of Probability: Explorations and Applications* (Cambridge University Press, Cambridge, UK, 2013).

72. Suhir, E. *Applied Probability for Engineers and Scientists* (McGraw-Hill, New York, 1997).

73. Kim, S. *et al.* Evaluation of intrafraction patient movement for CNS and head and neck IMRT. *Med Phys* **31,** 500–6. ISSN: 0094-2405 (Print) 0094-2405 (Linking) (2004).

74. Amundson, I., Sallai, J., Koutsoukos, X. & Ledeczi, A. Mobile Sensor Waypoint Navigation via RF-Based Angle of Arrival Localization. *International Journal of Distributed Sensor Networks,* 1–15. ISSN: 1550-1329 (2012).

75. Dagefu, F. T., Oh, J. & Sarabandi, K. A Sub-Wavelength RF Source Tracking System for GPS-Denied Environments. *Ieee Transactions on Antennas and Propagation* **61,** 2252–2262. ISSN: 0018-926x (2013).

76. Gorcin, A. & Arslan, H. A Two-Antenna Single RF Front-End DOA Estimation System for Wireless Communications Signals. *Ieee Transactions on Antennas and Propagation* **62,** 5321–5333. ISSN: 0018-926x (2014).

77. Wiersma, R. D., Wen, Z. F., Sadinski, M., Farrey, K. & Yenice, K. M. Development of a frameless stereotactic radiosurgery system based on real-time 6D position monitoring and adaptive head motion compensation. *Physics in Medicine and Biology* **55,** 389–401. ISSN: 0031-9155 (2010).

Appendix A

Useful Mathematics

A.1 Descriptions of time-based 6D motion

When discussing the accuracy of a multi-dimensional system, one should strive to use the most appropriate mathematics to describe the measurements taken. In this work, the two primary statistical methods employed are the root-mean-square (RMS) and the mean absolute error (MAE), both described formally below. These two metrics are important because they accurately create a sense of the typical errors present on a signed axis. In most cases within this work, RMS is used in any measurements that are part of a time series while MAE is used in any discrete measurements.

$$\text{RMS} = \sqrt{\frac{1}{n}\left(x_1^2 + x_2^2 + ... + x_n^2\right)} \tag{A.1}$$

$$\text{MAE} = \frac{\sum_{i=1}^{n} |x_{\text{measured}} - x_{\text{target}}|}{n} \tag{A.2}$$

The RMS is especially useful in cases of motion compensation, such as in the compensation ratio described by Hermann [55] is the compensation ratio [48].

$$CR = \frac{RMS(p_{actual})}{RMS(p_{desired})} \tag{A.3}$$

In testing the 6D motion phantom developed by Wiersma's group, Belcher identified that changes to rotations also affect positioning along different axes as well, depending on the distance between the point of interest and the pivot point [51]. For the specific case where rotations are only about a single axis (i.e., pitch if about the lateral axis) and where the changes are small, the group identified a linear approximation of the change in positioning.

$$\begin{aligned} x &= x \\ y &= y - a\theta \\ z &= z + b\theta \end{aligned} \tag{A.4}$$

where a and b are the distances between the pivot point and the point of interest along the z (AP) and y (SI) axes, respectively. This model is actually taken from rotation matrices, which may be used to make a rotational transformation on any set of (x, y, z) points about the origin. In standard form, the matricies are defined as:

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{A.5}$$

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{A.6}$$

$$R_z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.7}$$

about the x, y, and z axes, respectively. Using matrix multiplication, these may multiplied

onto the column vector of any single, 3D point. To accurately describe a complex rotation (rotations described by a combination of more than one reference axis), the matricies are each multiplied onto each other in a set order, usually R_x onto R_y, R_{xy} onto R_z.

These matrices may be preemptively multiplied together with the initial (x, y, z) coordinates of a single point for faster computation time, resulting in one comprehensive matrix, $R(x, y, z, \alpha, \beta, \gamma)$:

$$R(x,y,z,\alpha,\beta,\gamma) = \begin{bmatrix} x(\cos\beta\cos\gamma) + y(\sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma) + z(\cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma) \\ x(\cos\beta\sin\gamma) + y(\sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma) + z(\cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma) \\ -x\sin\beta + y(\sin\alpha\cos\beta) + z(\cos\alpha\cos\beta) \end{bmatrix} \tag{A.8}$$

This matrix was used in the C program. In the LabVIEW program, pre-built functions were used instead, since computation time was not much of an issue. However, future versions of the program may incorporate this matrix to speed up computation.

Appendix B

Physical Schematics

These are complete schematics for all custom components produced as part of this project. Each of these parts were either custom machined or 3D-printed. A description of materials used for each part is provided in Chapter 2, in subsections 2.3.5.

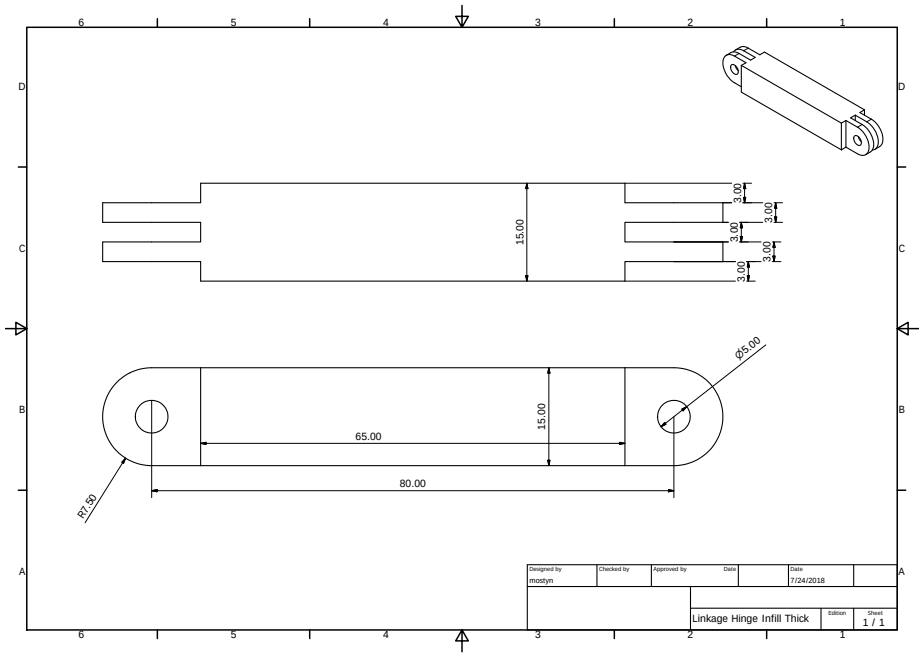


Figure B.1: Hinge type linkage

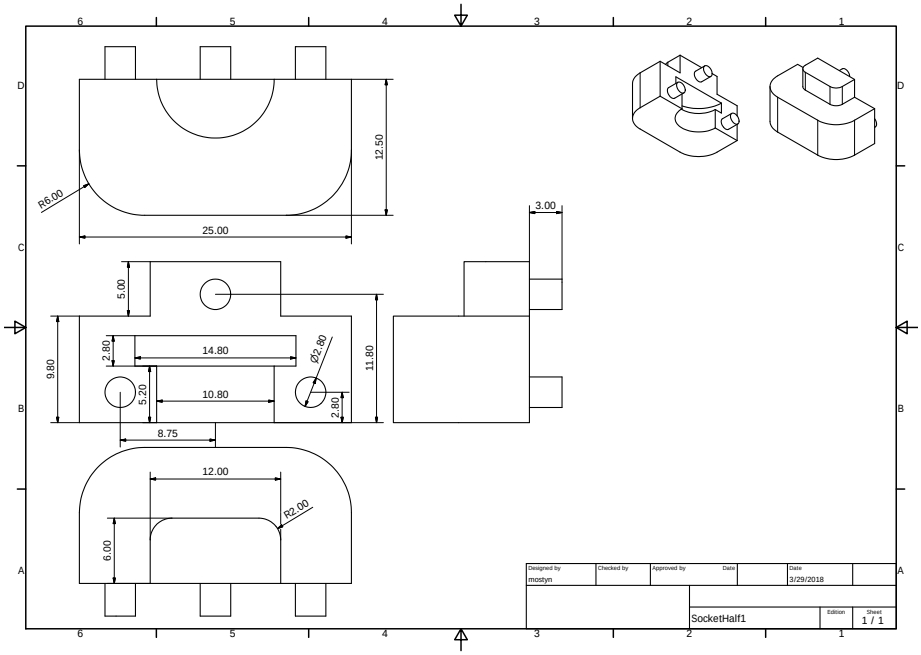


Figure B.2: Socket half 1

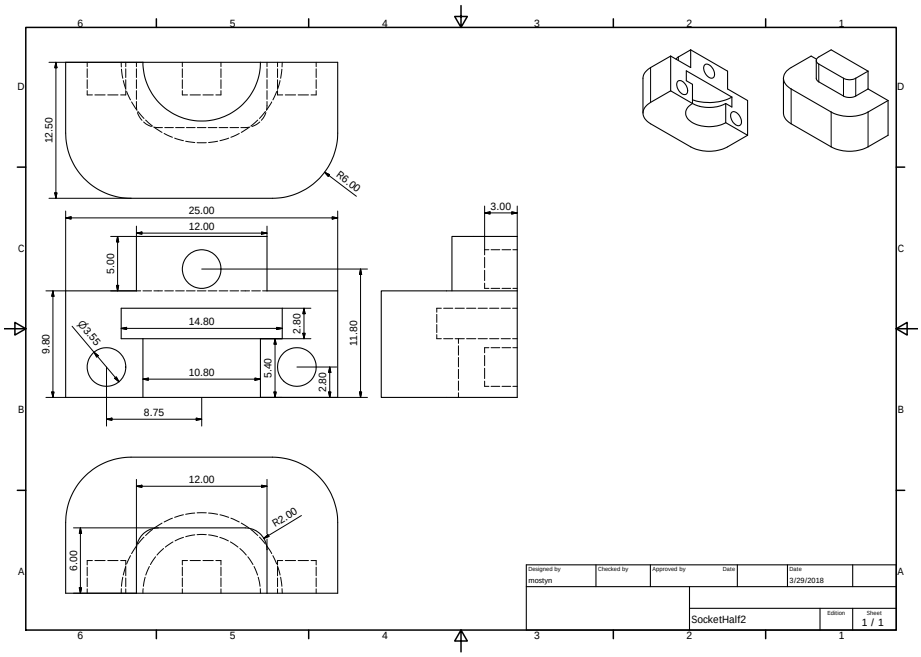


Figure B.3: Socket half 2

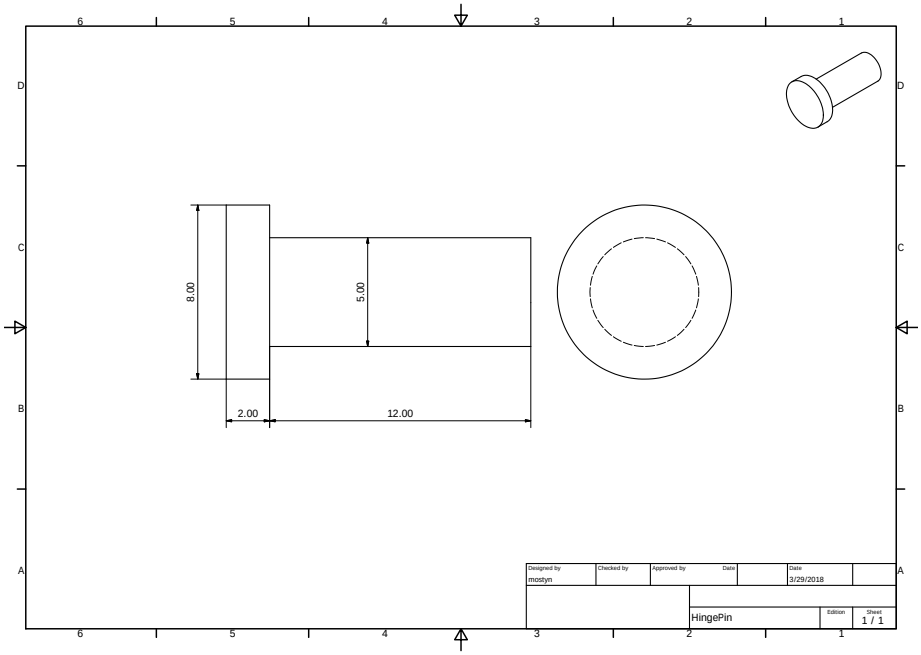


Figure B.4: Hinge pin

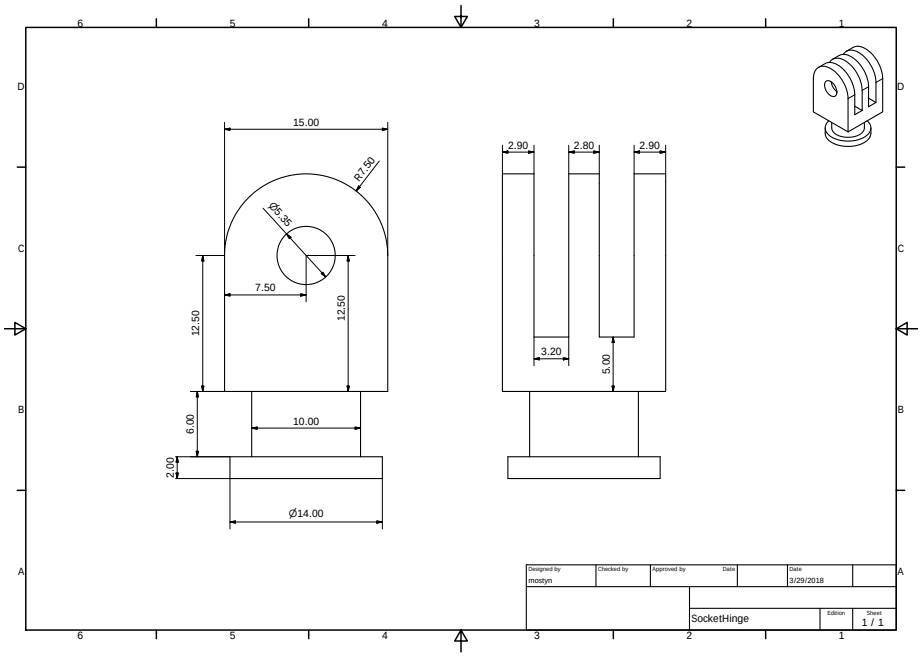


Figure B.5: Pivoting hinge

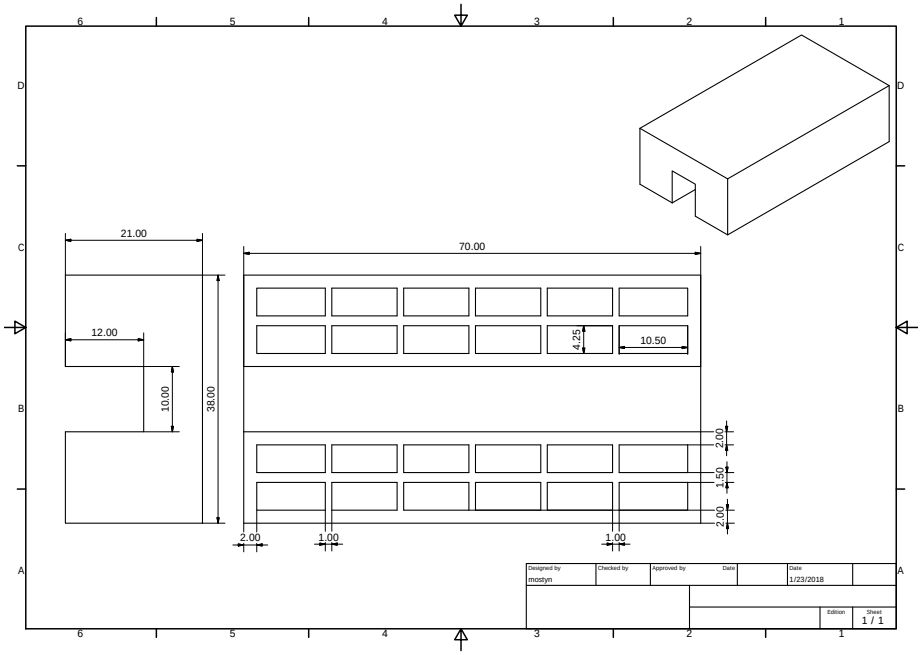


Figure B.6: Support for slider

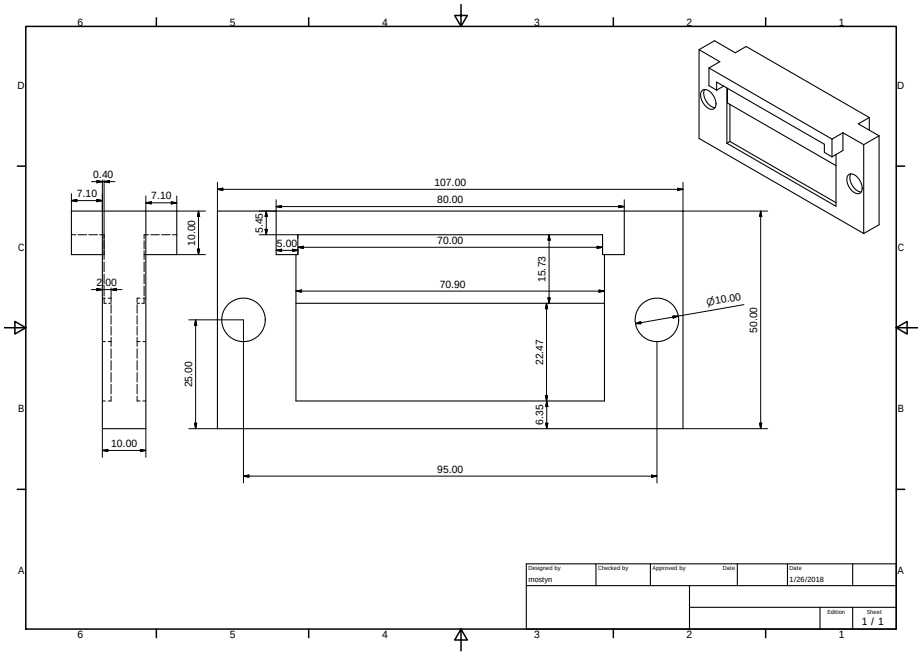


Figure B.7: Center Wall for Slider Track

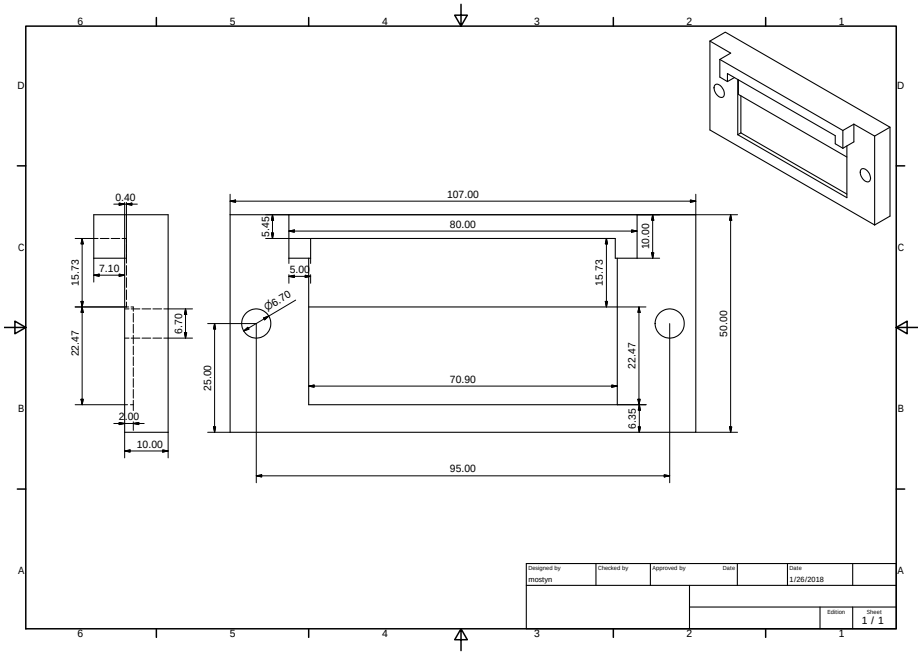


Figure B.8: Outer wall for slider track

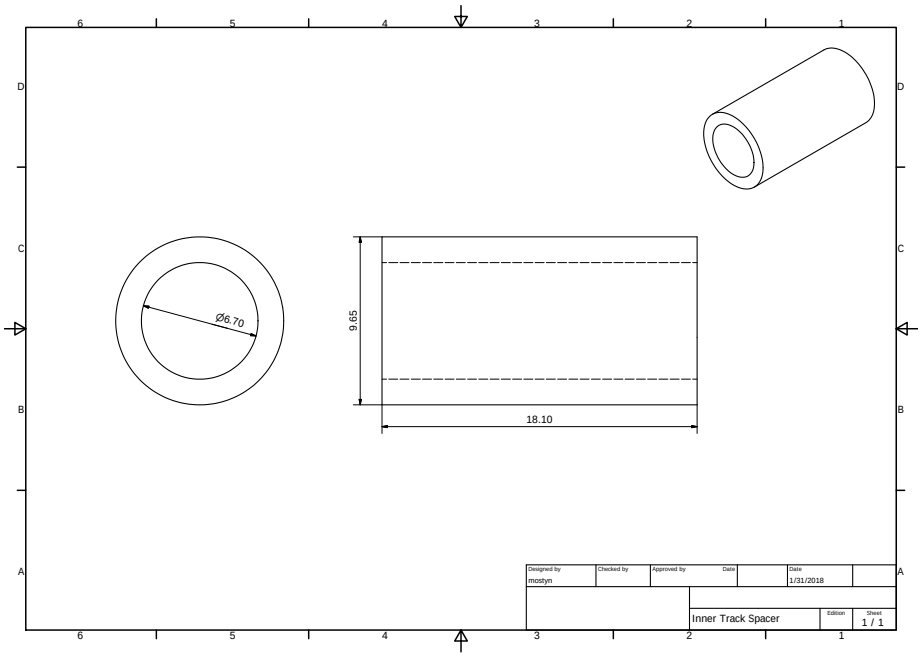


Figure B.9: Spacer to hold sprocket in place in slider assembly

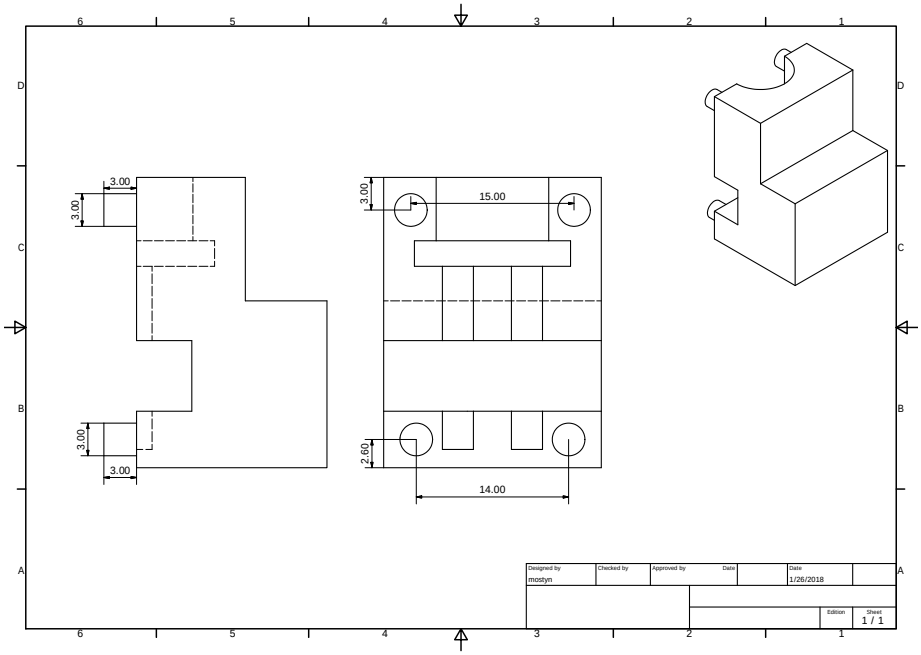


Figure B.10: Slider half 1

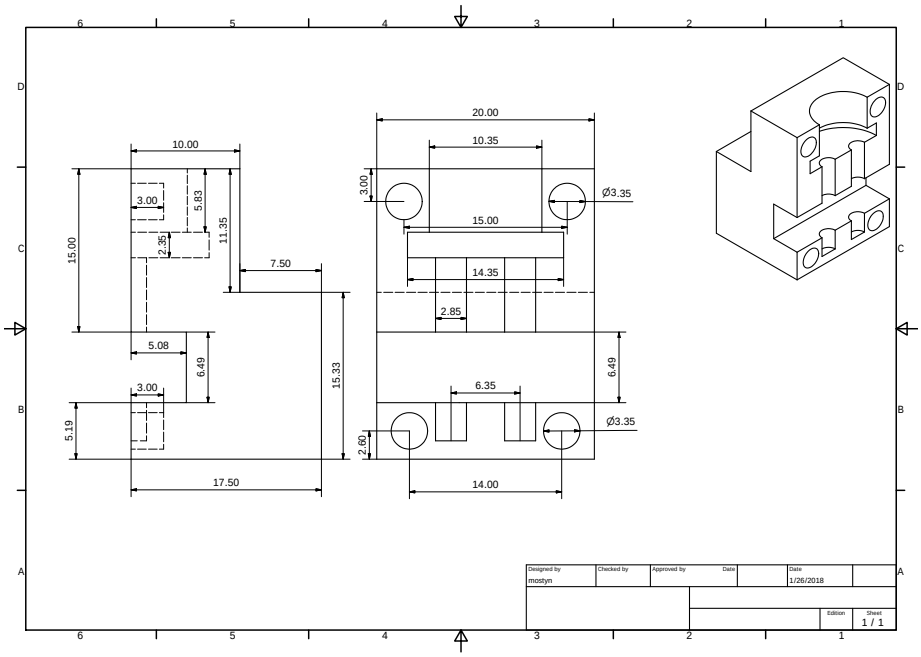


Figure B.11: Slider half 2

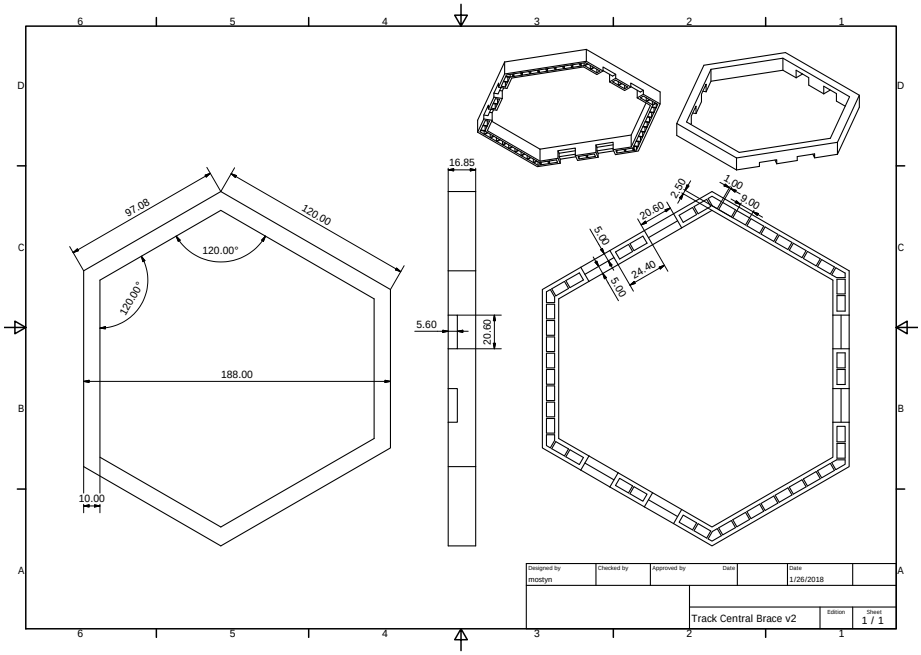


Figure B.12: Central brace for slider tracks

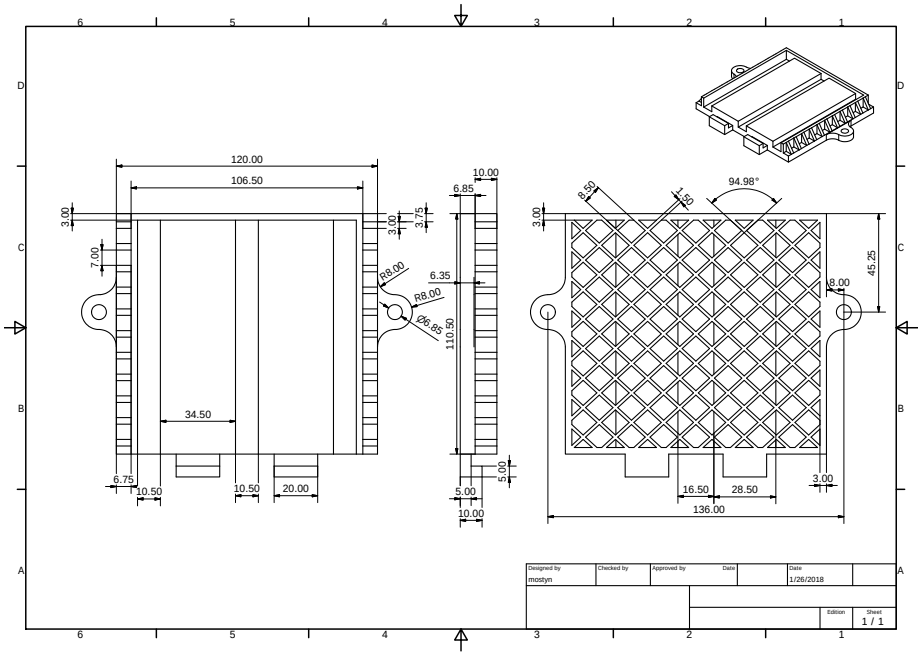


Figure B.13: Base for slider tracks

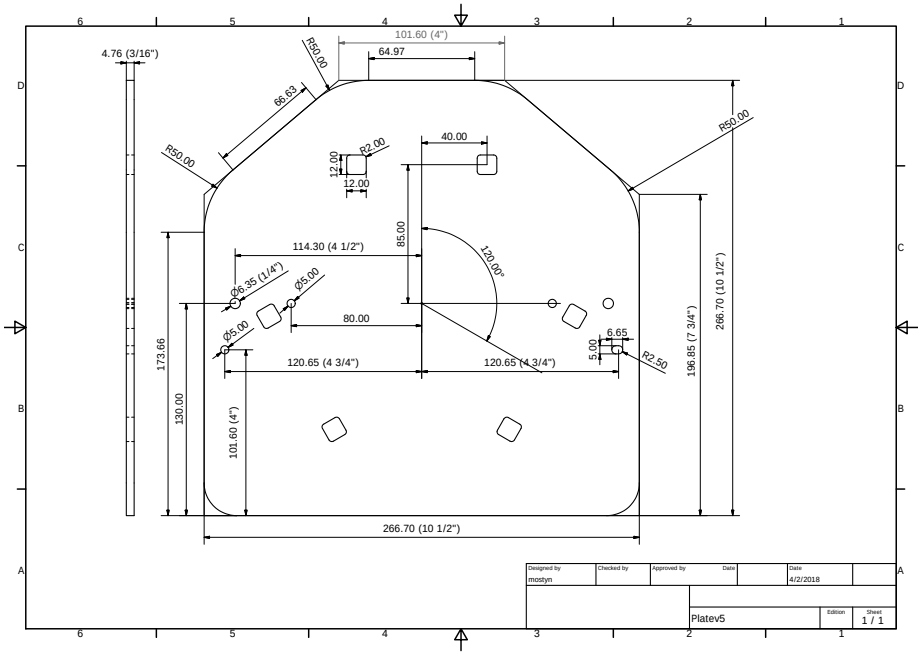


Figure B.14: Positioning plate

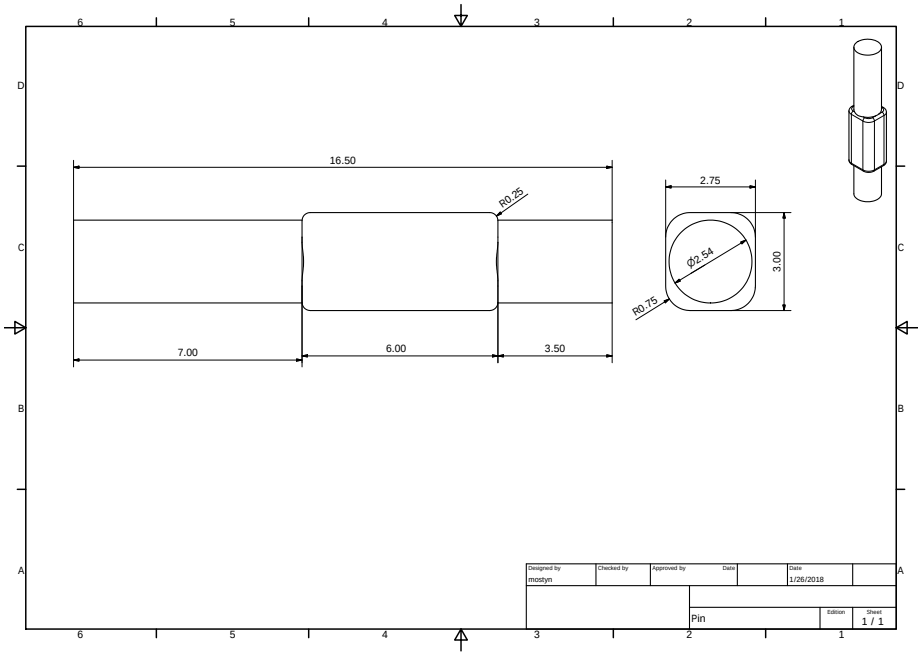


Figure B.15: Pin for holding chains to Sliders

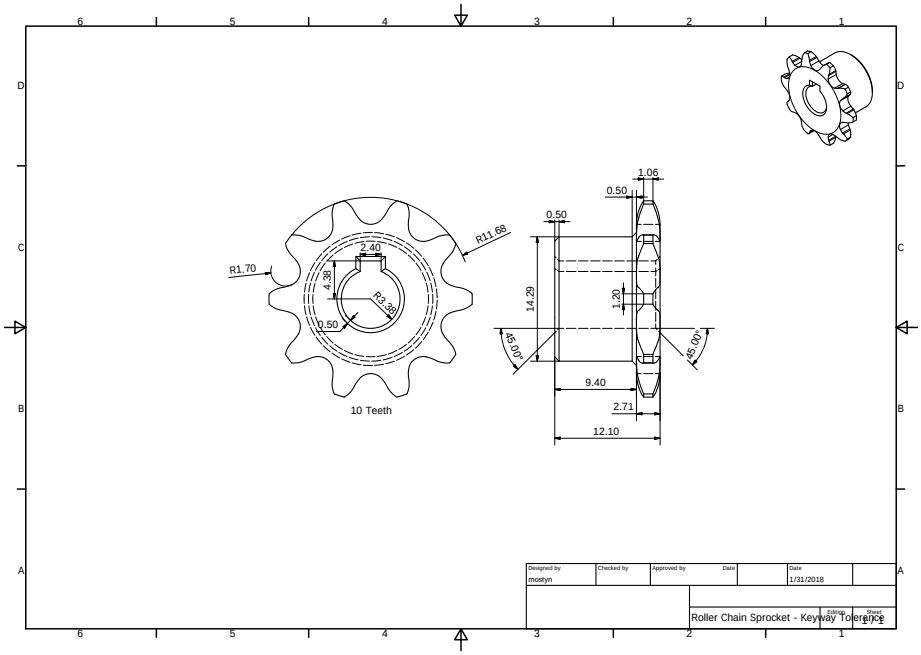


Figure B.16: Sprocket for chain assembly

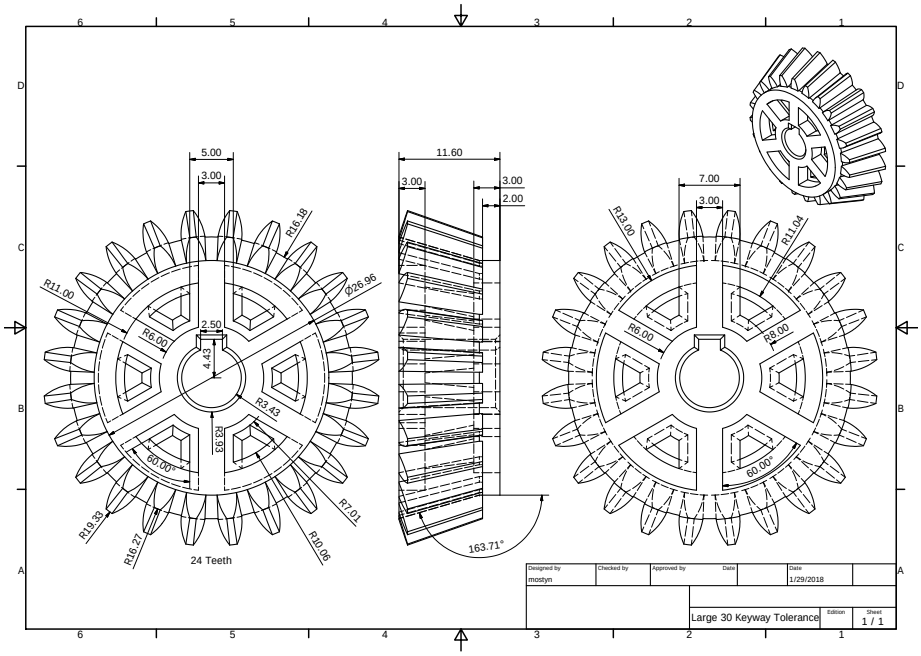


Figure B.17: Large gear of 30° gear pair

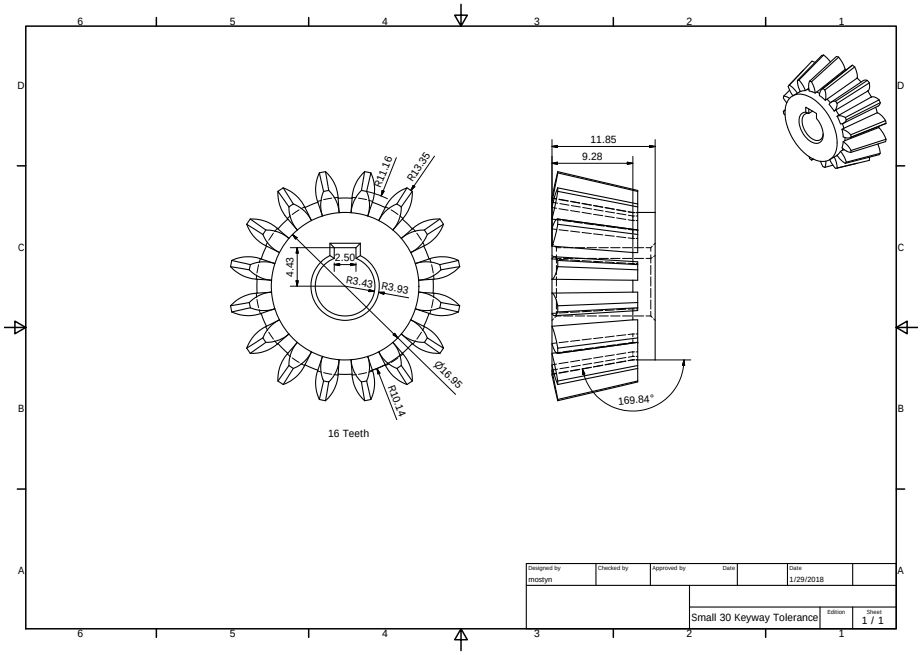


Figure B.18: Small gear of 30° gear pair

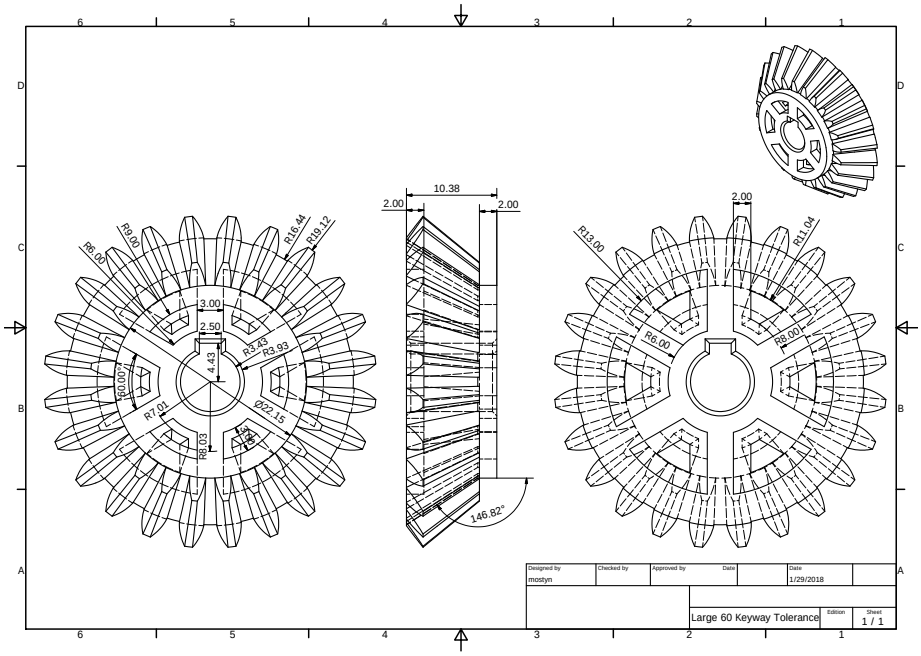


Figure B.19: Large gear of 60° gear pair

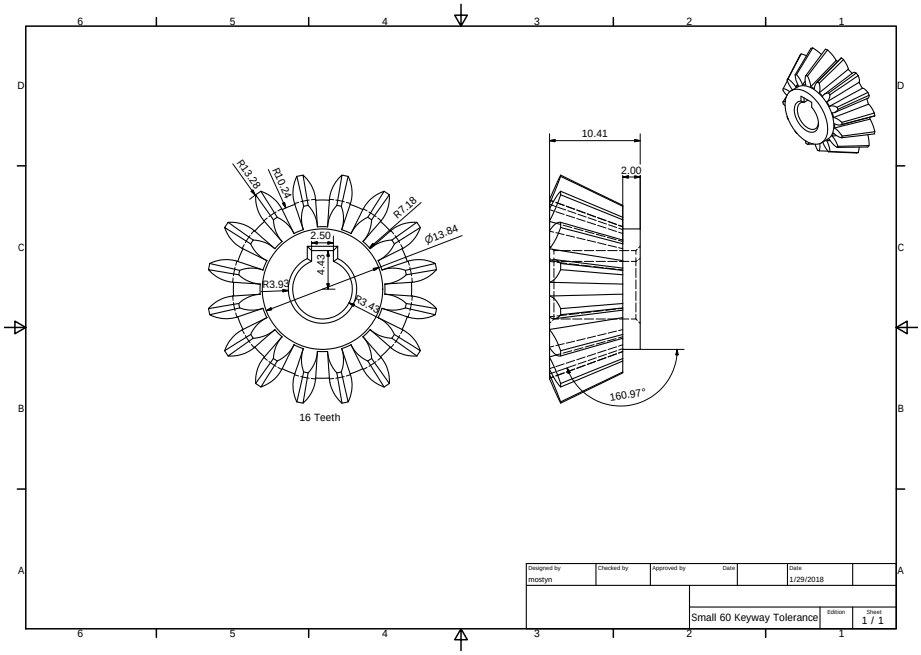


Figure B.20: Small gear of 60° gear pair

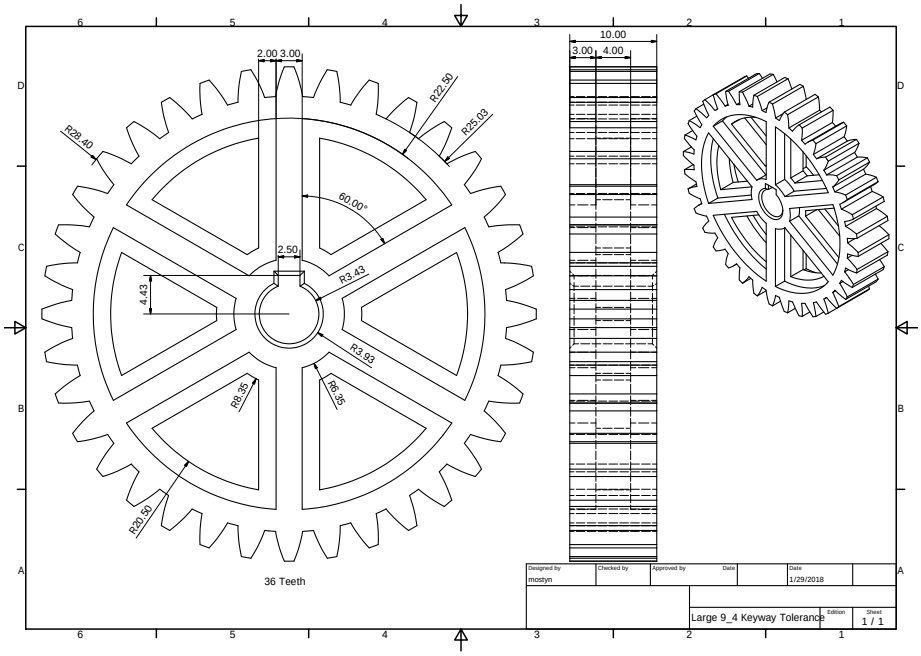


Figure B.21: Large 4:9 gear

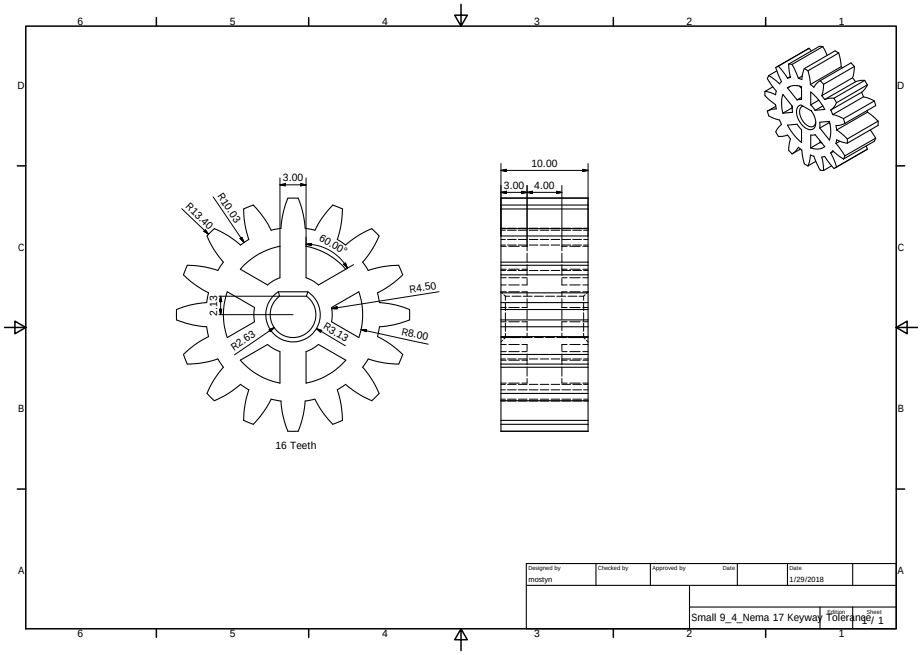


Figure B.22: Small 4:9 gear with D-shaft connection

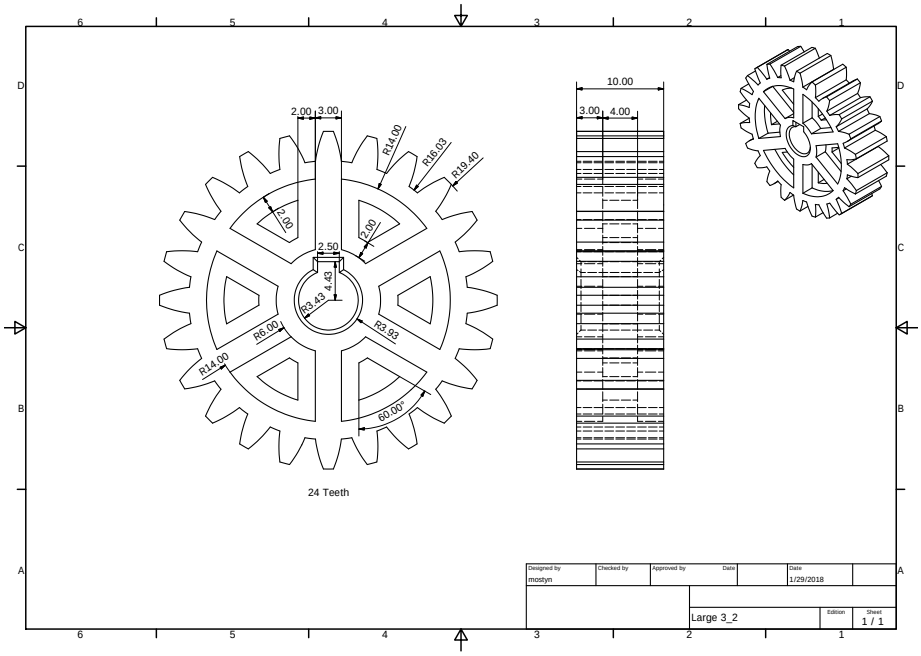


Figure B.23: Large 2:3 gear pair

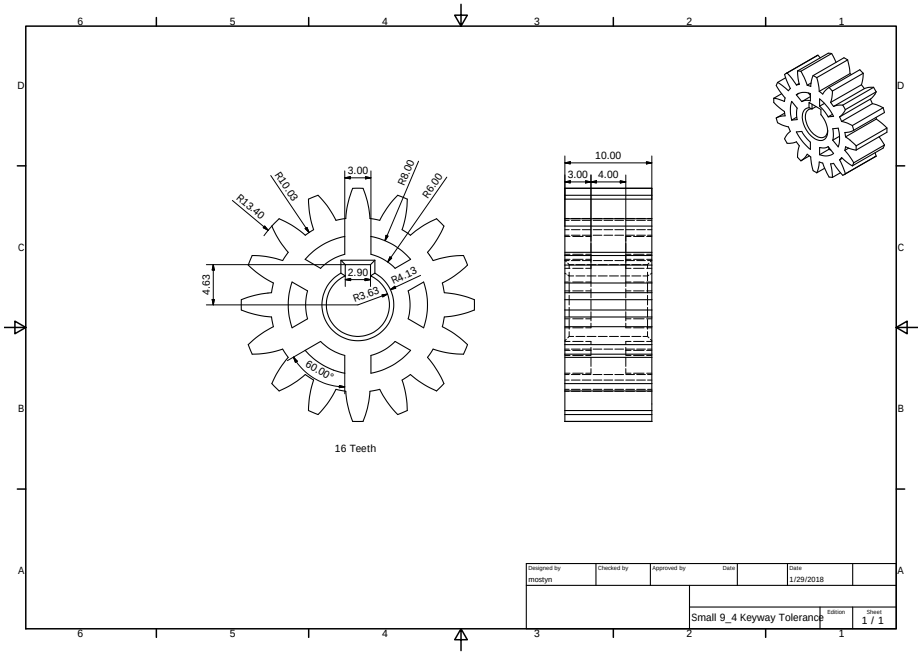


Figure B.24: Small for 2:3 gear pair

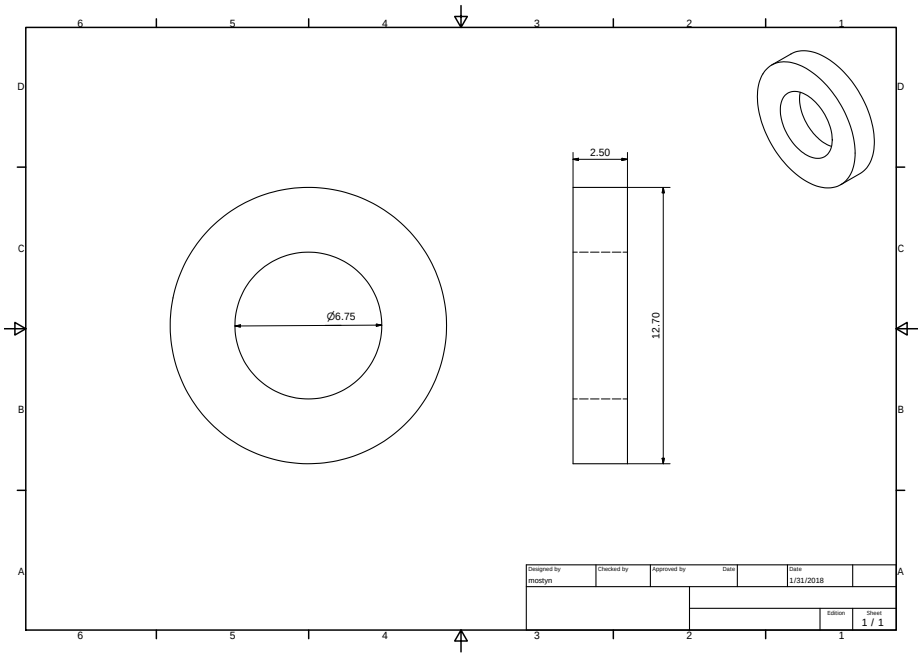


Figure B.25: Bearing for holding drive shafts in place

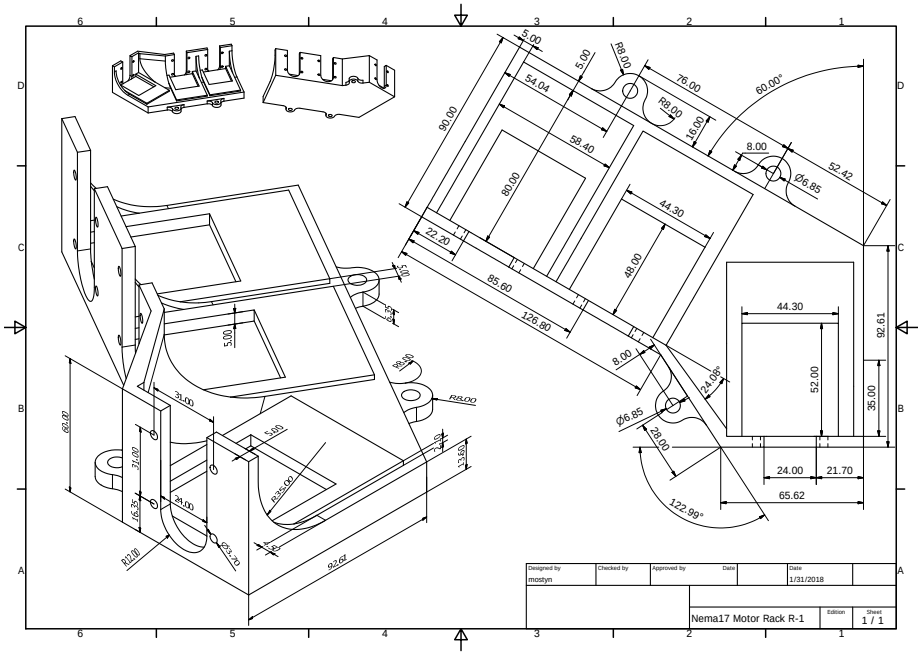


Figure B.26: One side of the mounts for NEMA 17 motors. Other half is symmetric

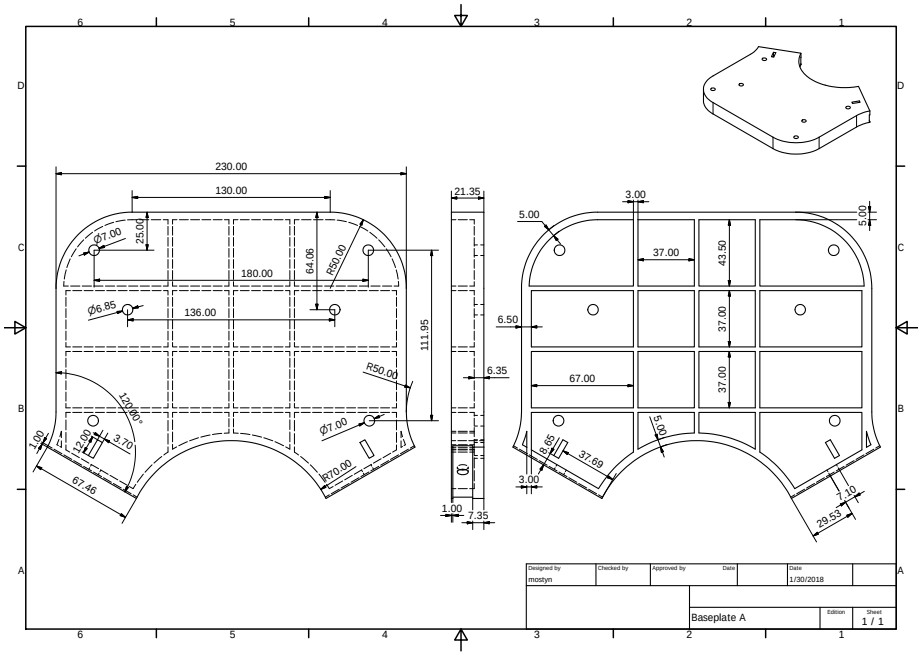


Figure B.27: Segment ”A” of mounting board

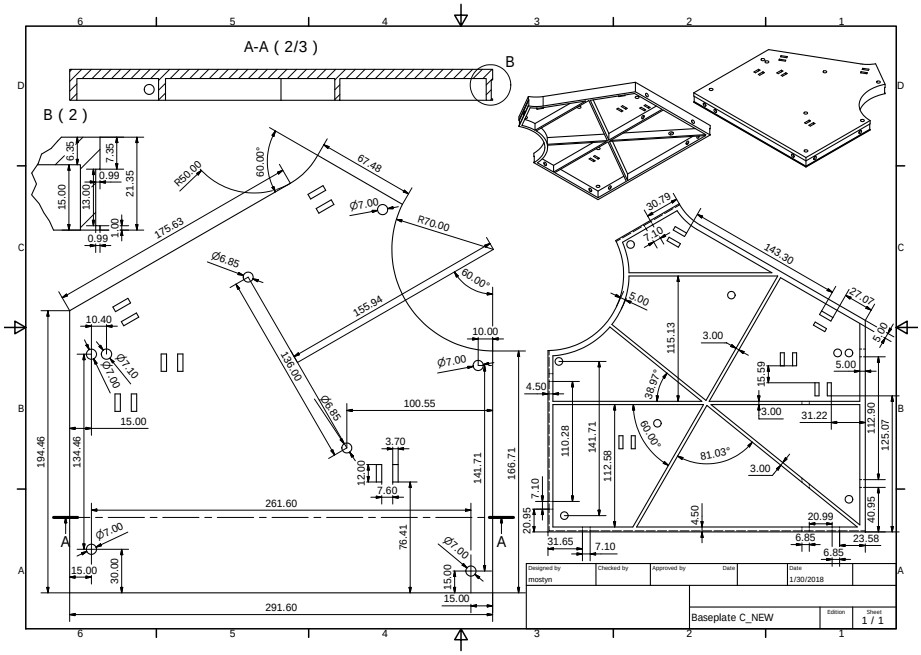


Figure B.28: Segment ”C” of mounting board. Segment ”B” is symmetric

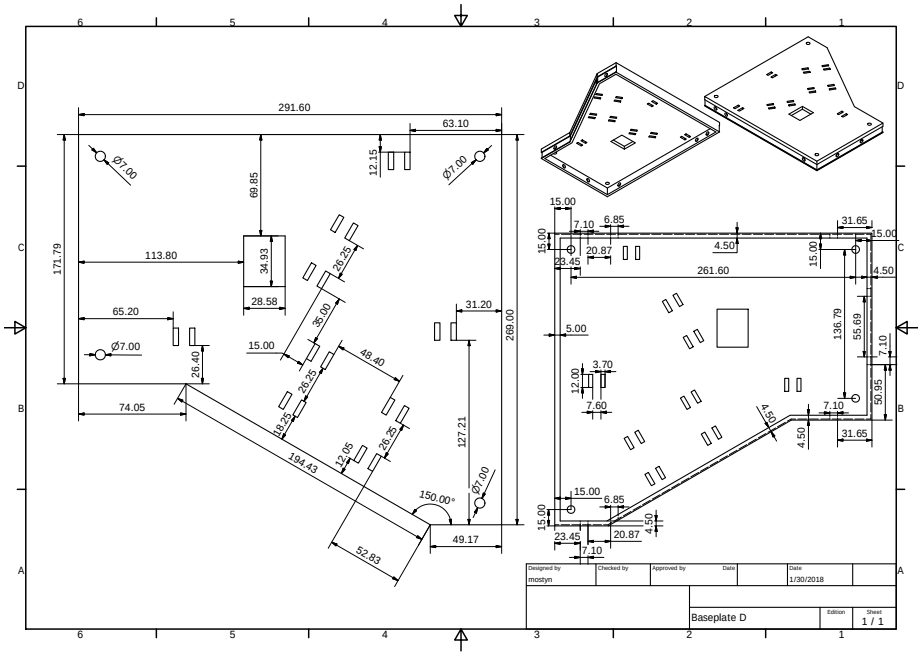


Figure B.29: Segment ”E” of mounting board. Segment ”D” is symmetric

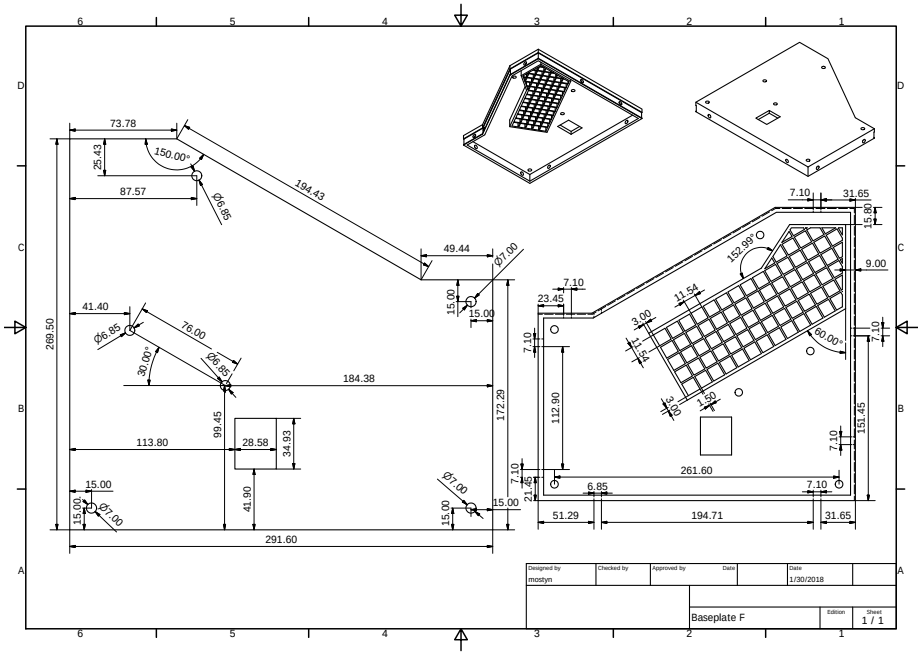


Figure B.30: Segment ”G” of mounting board. Segment ”F” is symmetric

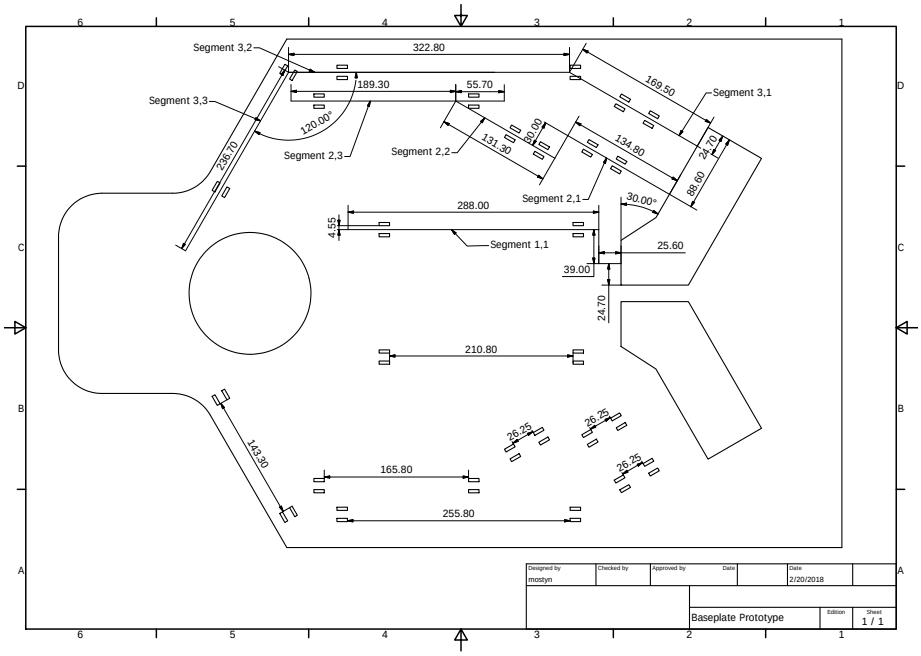


Figure B.31: Prototypical baseplate with layout for rod mounts

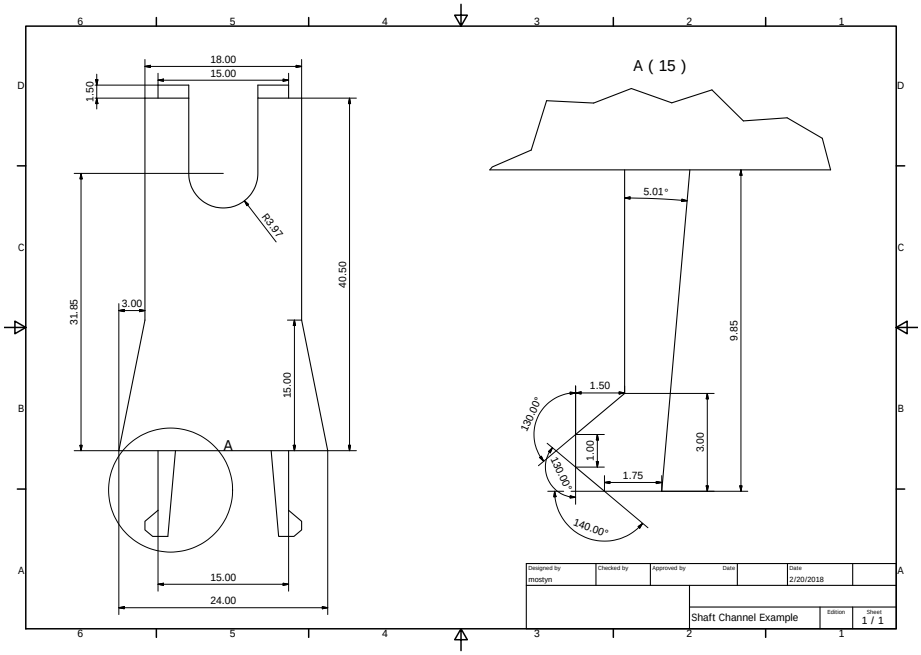


Figure B.32: Profile view of rod mounts

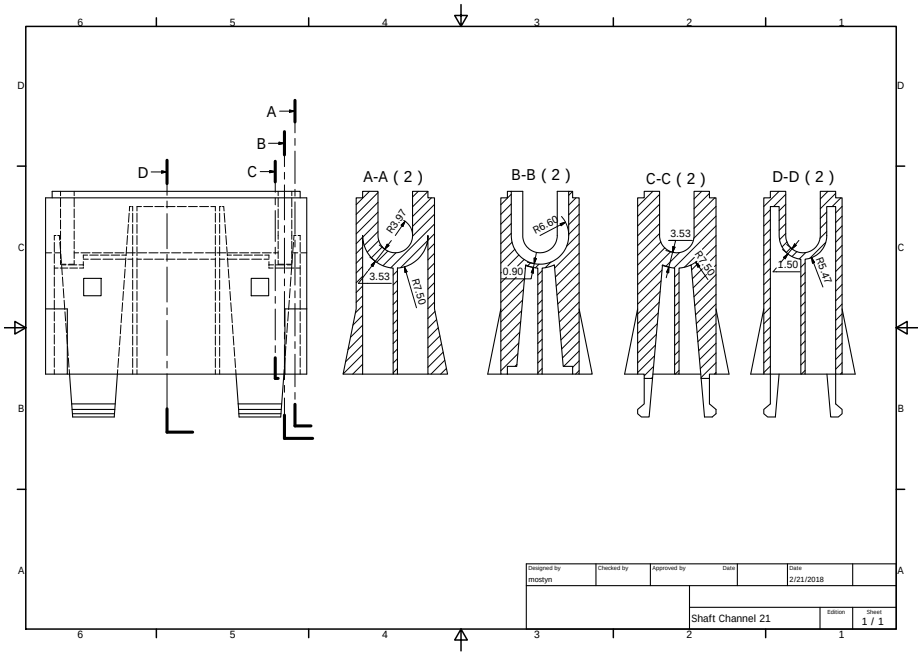


Figure B.33: Significant cross sections of rod mounts

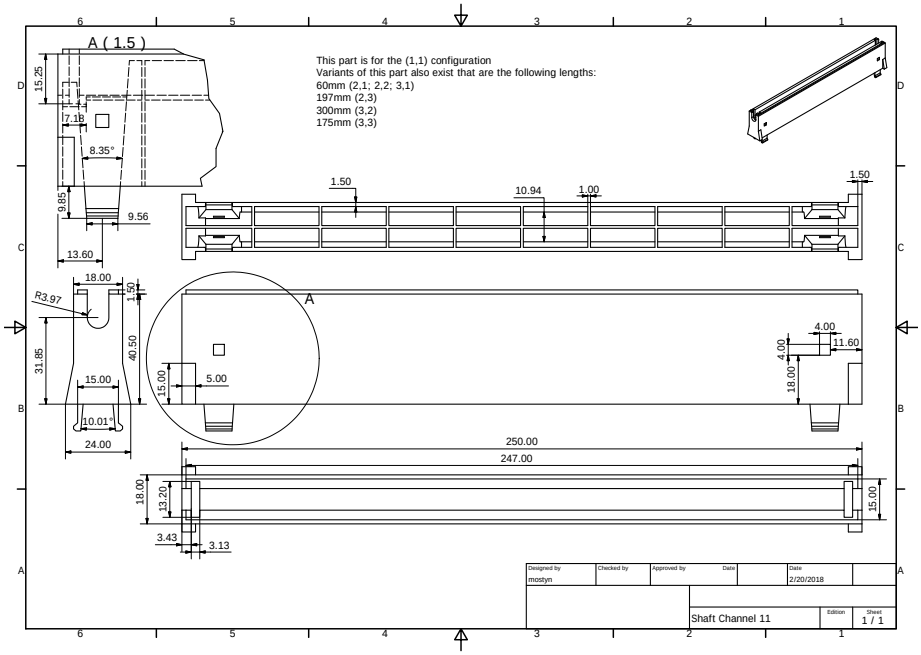


Figure B.34: Rod mounts

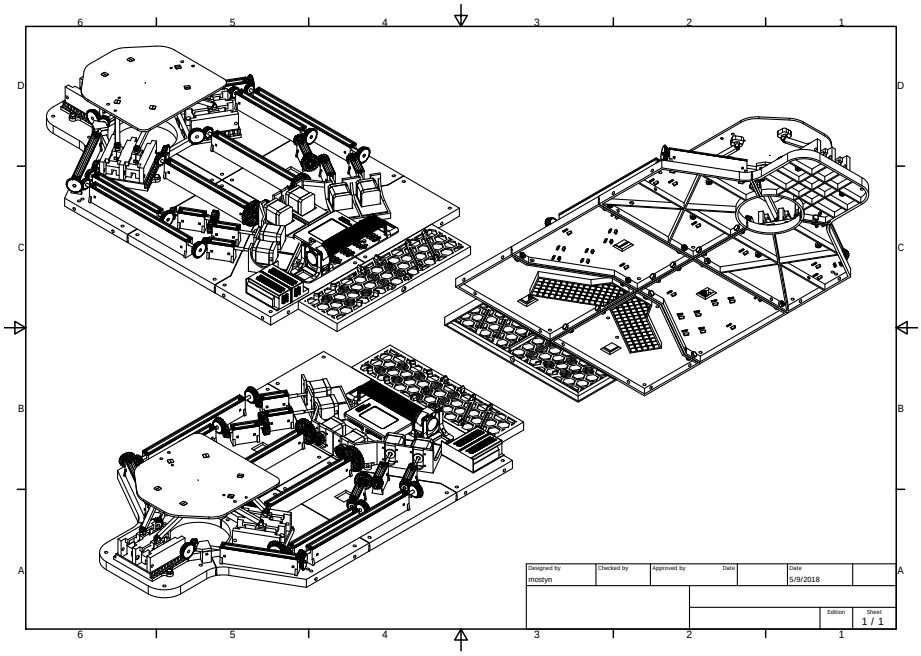


Figure B.35: Assembled third generation prototype

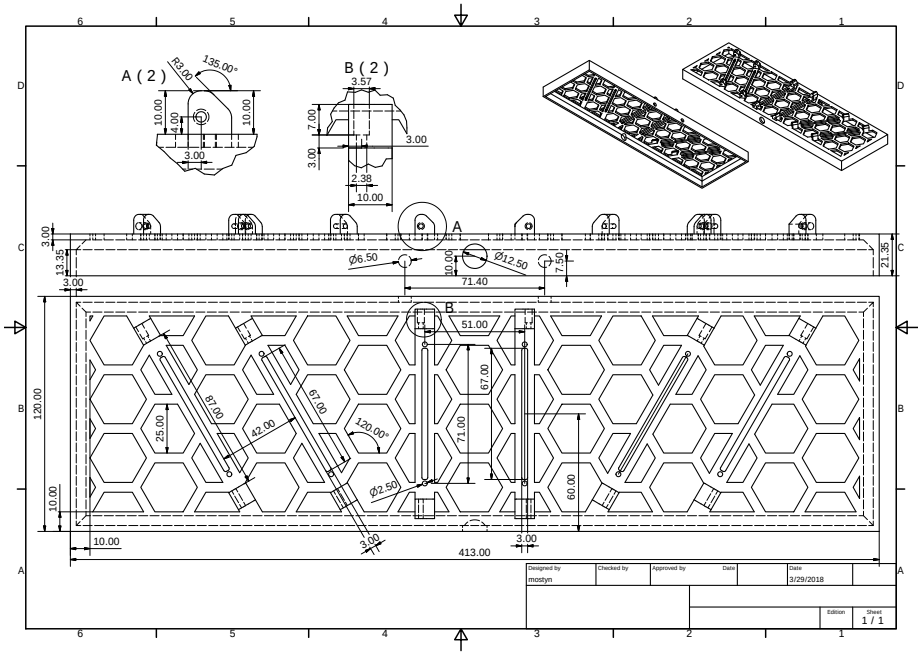


Figure B.36: Mount for potentiometers

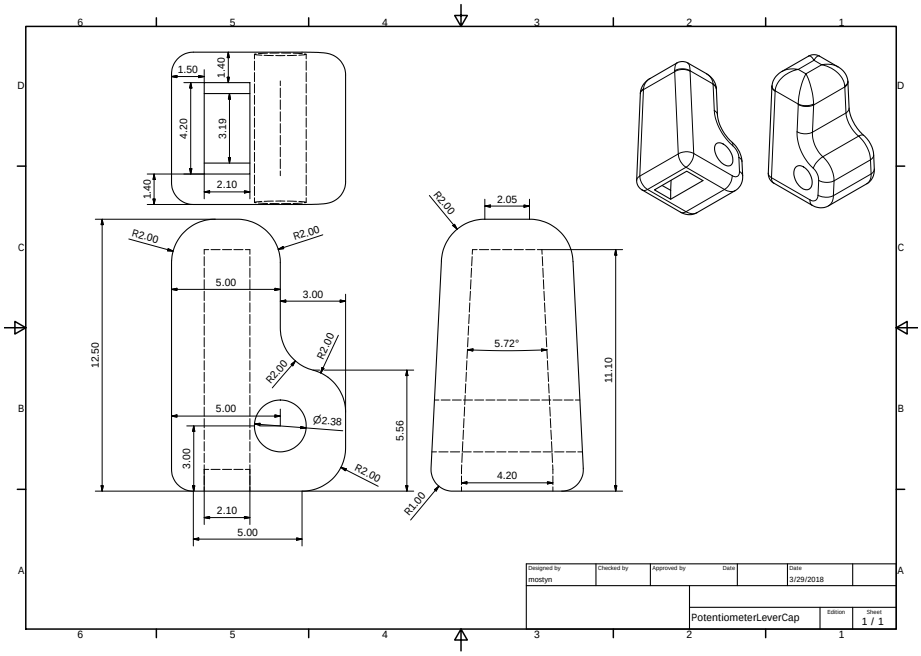


Figure B.37: Cap attachment for potentiometer

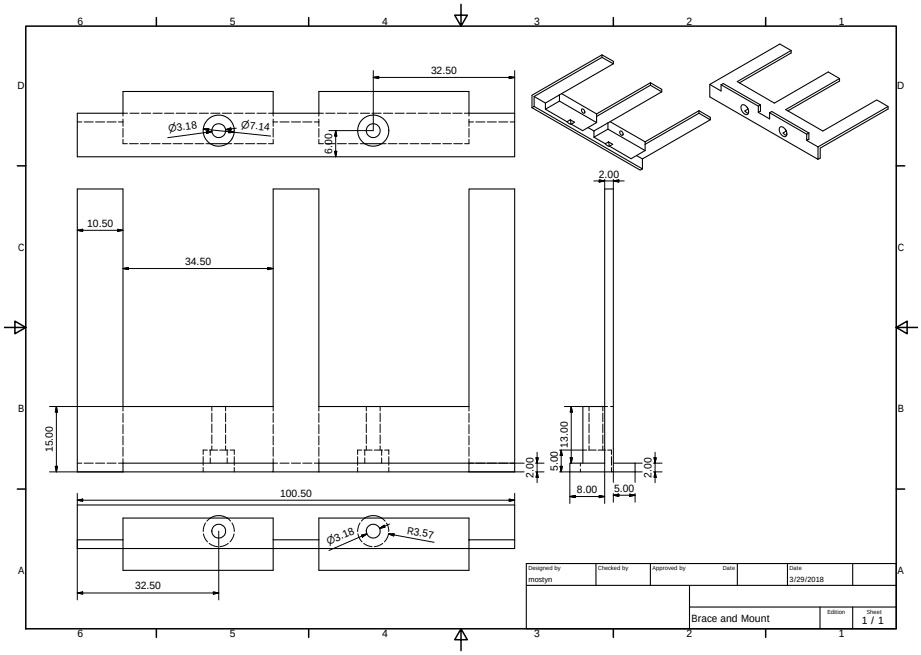


Figure B.38: Core positioning attachment for bowden cable

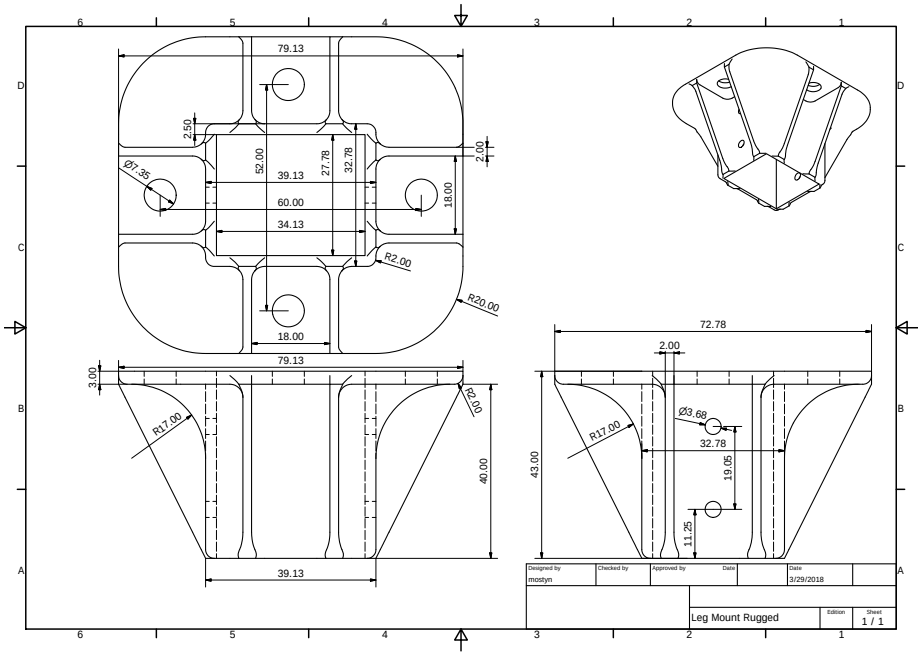


Figure B.39: Sockets for legs on patient standoff

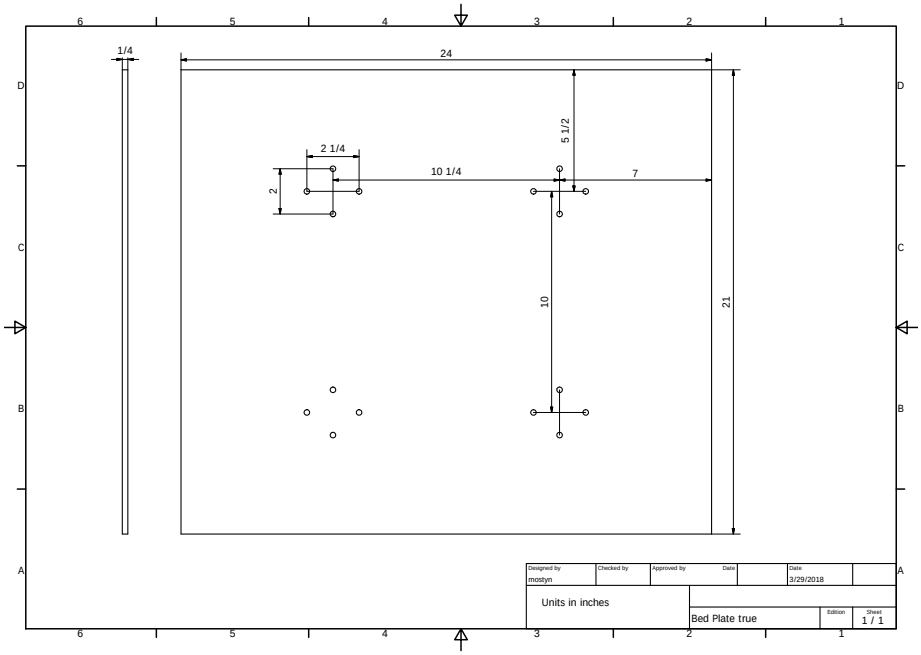


Figure B.40: Support surface for patient standoff

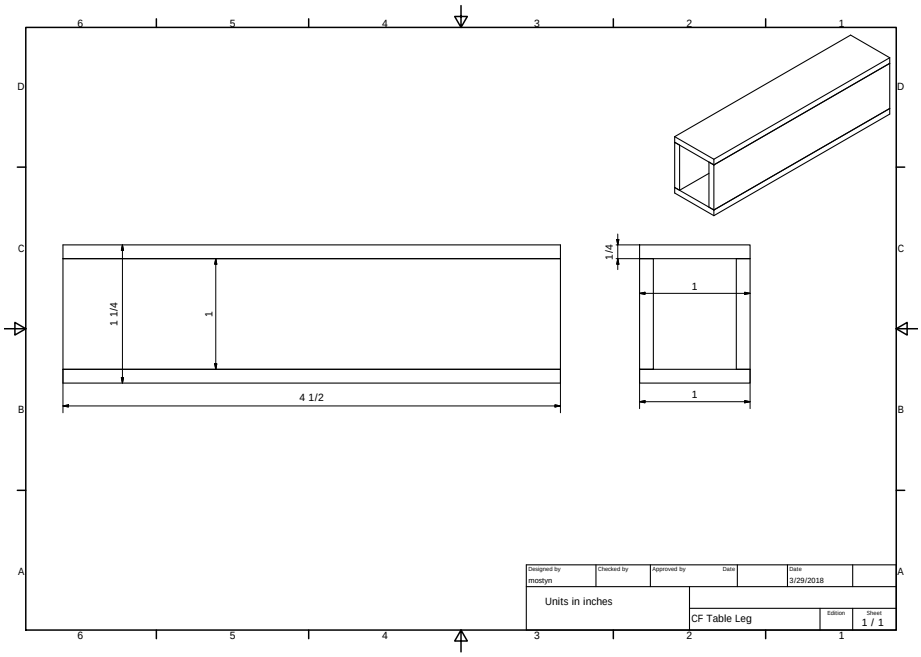


Figure B.41: Leg assembled from 1” $\times$ 1/4” carbon fiber strips

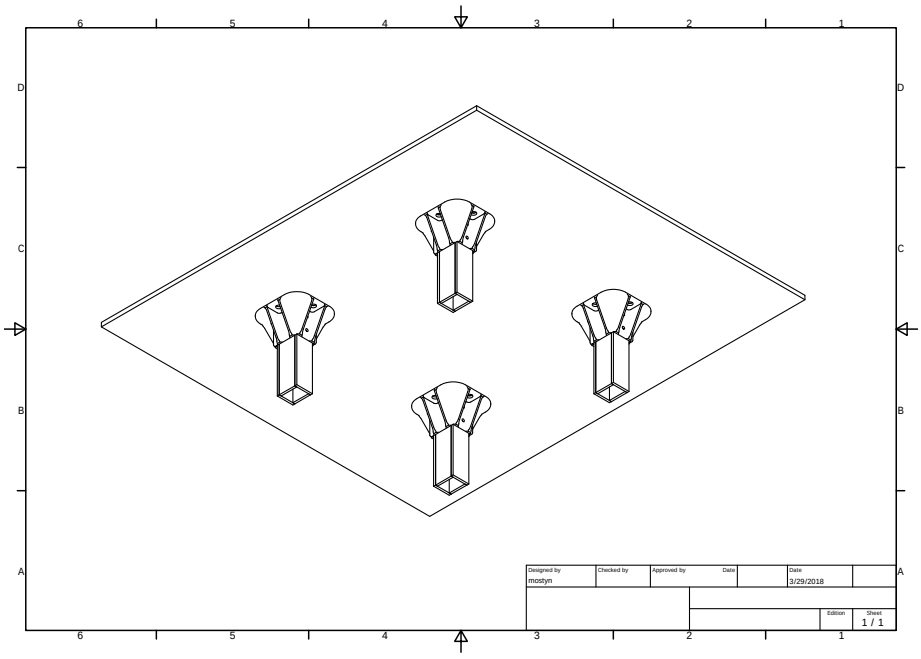


Figure B.42: Assembly of patient standoff

Appendix C

Electronic Schematics

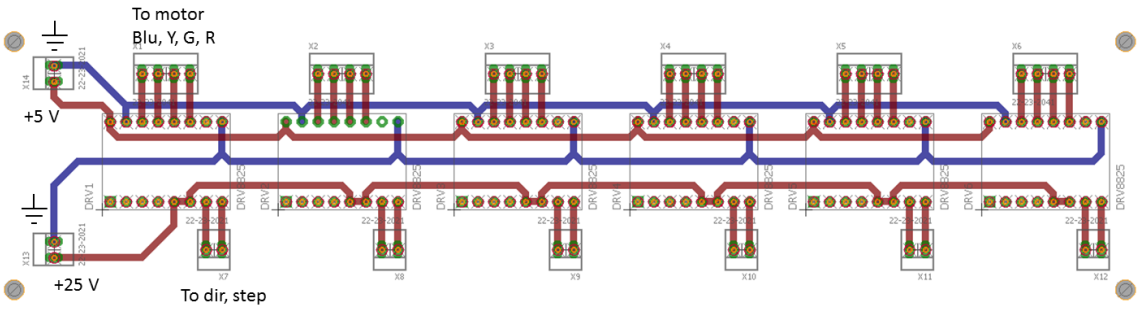
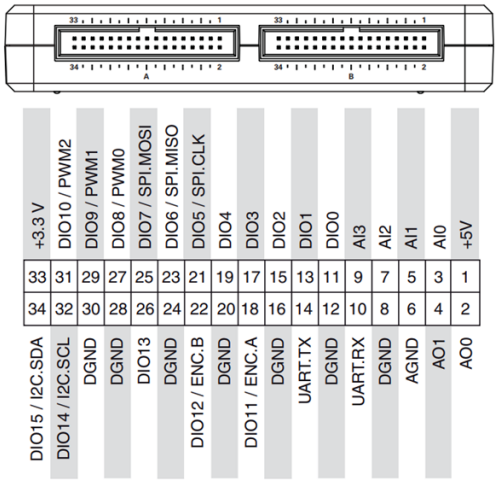


Figure C.1: Control board



Pin (A/B)	Connected to
1	Potentiometer +V
2	Potentiometer 2/6
3	Potentiometer 3/4
5	Potentiometer 1/5
11/23	Motor 1/4 step
13/25	Motor 1/4 direction
15/27	Motor 2/5 step
17/29	Motor 2/5 direction
19/31	Motor 3/6 step
21/34	Motor 3/6 direction
33	Motor digital VCC
33	Motor digital GND

Figure C.2: Pinout diagram for myRIO electronics

Appendix D

Code

D.1 MATLAB

D.1.1 Rotation matrix calculation

```
function [R] = Rot(Theta,Axis)
%Rot creates the rotation matrix about an axis
%  Input roation angle in degrees and axis, axis must be x,y, or z

  %X-Axis
  if Axis == 1
    R = [1, 0, 0;
    0, cosd(Theta), -sind(Theta);
    0, sind(Theta), cosd(Theta)];
  end

  %Y-Axis
  if Axis == 2
    R = [cosd(Theta), 0, sind(Theta);
    0, 1, 0;
    -sind(Theta), 0, cosd(Theta)];
  end

  %Z-Axis
  if Axis == 3

  end
end
```

D.1.2 6D position calculation

```
% 6D Plate Control
% Commercialization Project

% This program calculates the necessary slider positions to acheive
   any 6D
% orientation of the plate. The plate is assumed to be inline with the
% sockets in this iteration of the program.


function Return = SixDSliderPositions(Coords,Geometry)

Coords = [0,0,0,0,0,0];
Geometry = [20,80,80,40];

X = Coords(1);
Y = Coords(2);
Z = Coords(3);
alpha = Coords(4);
beta = Coords(5);
gamma = Coords(6);

%% Inputs
% This section turns the inputs into vectors to be used later

% [mm] Translational coordinates of the plate
% X and Y are horizontal displacements
% Z is vertical displacement
PlateTrans = [X, Y, Z];

% [deg] Rotational coordinates of the plate
% alpha and beta are tilts of the plate
% gamma is a rotation about the vertical axis
PlateRot = [alpha, beta, gamma];

%% Constants
% This section contains the device dimensions

% [mm] Half of the distance between a pair of plate sockets
SocketUHalfWidth = Geometry(1);
```

```
% [mm] The median of the plate, the distance between the the center of
    the
% plate and the sockets, along bisecting line
PlateMedian = Geometry(2);

% [mm] The height of the upper sockets from the top of the plate at
    rest
%SocketUHeight = -13.175;
%SocketUHeight = -19;

SocketUHeight = -13.175;

% [mm] The height of the lower sockets from the top of the plate at
    rest
SocketLHeight = SocketUHeight - Geometry(4);
%SocketLHeight = 83;


% [mm] The length of the rails, defined along the medians
RailLength = 400;

% [mm] The start of the rails, defined along the medians
RailStart = 0;

% [mm] Half of the distance between a pair of threaded rods
SocketLHalfWidth = 25;

% [mm] Length of linkages, from socket center to socket center
Linkage = Geometry(3);


%% Socket initial coordinate geometry
% This section calculates the socket positions based on the input
    constants

% [deg] Angular separation between the median vectors and the plate
    sockets
theta = atand(SocketUHalfWidth/PlateMedian);


% [mm] The distance from the center of the plate the upper sockets
SocketDist = norm([SocketUHalfWidth, PlateMedian]);
```

```
% [deg] Array of angular positions of each upper socket
Thirds = (ceil((1:6)/2)-1)*(360/3);
Halves = ((-1).^(1:6));
SocketUAngles = Thirds + Halves*theta;


% [mm] XYZ Coordinates of each socket
SocketU = [SocketDist*cosd(SocketUAngles);...
SocketDist*sind(SocketUAngles);...
repmat(SocketUHeight,1,6)]';

%% Transformations of sockets
% This section applies the input transformations to the socket
   positions,
% first with rotations, then translations

SocketUN = zeros(6,3);

for i = 1:6
  % [mm] 3D positions of each socket after applying the rotations
  SA = sind(alpha);
  CA = cosd(alpha);
  SB = sind(beta);
  CB = cosd(beta);
  SG = sind(gamma);
  CG = cosd(gamma);

  SocketUN(i,:) = [SocketU(i,1)*CB*CG + SocketU(i,2)*(SA*SB*CG-CA*SG)
     + SocketU(i,3)*(CA*SB*CG+SA*SG),...
  SocketU(i,1)*CB*SG + SocketU(i,2)*(SA*SB*SG+CA*CG) +
     SocketU(i,3)*(CA*SB*SG-SA*CG),...
  -SocketU(i,1)*SB + SocketU(i,2)*(SA*CB) + SocketU(i,3)*(CA*CB)];



  SocketU(i,:) =
     (Rot(gamma,3)*(Rot(beta,2)*(Rot(alpha,1)*SocketU(i,:)')))';

end


% [mm] 3D positions of each socket after applying translations
```

```
SocketU = SocketU + repmat(PlateTrans,6,1);


%% Construction of slider tracks
% This section calculates the limits of the slider positions

% [deg] Angular separation between the median vectors inner ends of
   rails
phi1 = atand(SocketLHalfWidth/RailStart);
phi2 = atand(SocketLHalfWidth/RailLength);


% [deg] Arrays of angular positions of rail limits
RailInnerAngles = Thirds + Halves*phi1;
RailOuterAngles = Thirds + Halves*phi2;


% [mm] Distance from origin to rail limits
RailInnerDist = norm([SocketLHalfWidth,RailStart]);
RailOuterDist = norm([SocketLHalfWidth,RailLength]);


% [mm] XYZ Coordinates of rail limits
RailInner = [RailInnerDist*cosd(RailInnerAngles);...
RailInnerDist*sind(RailInnerAngles);...
repmat(SocketLHeight,1,6)]';

RailOuter = [RailOuterDist*cosd(RailOuterAngles);...
RailOuterDist*sind(RailOuterAngles);...
repmat(SocketLHeight,1,6)]';


%% Inverse Kinematics
% This section computes the positions of the sockets on the rails
   based on
% the positions of the transformed plate sockets

% [mm] Vector defining the rail tracks
RailVect = RailOuter - RailInner;


% [mm] Shortest distance between each upper socket and each rail
```

```
% This is distance is the radius of a cylinder coaxial with the rail
   where
% the shell intersects through the center of the upper socket
SocketURailDist = sqrt(sum((cross(RailVect,SocketU -
   RailInner)).^2,2))/...
RailLength;


% [mm] Track distance from shortest distance intersect and slider
   position
dx = sqrt(repmat(Linkage^2,6,1) - SocketURailDist.^2);


% [mm] Distance between upper sockets and inner reference position
SocketURailInner = sqrt(sum((SocketU - RailInner).^2,2));


% [mm] Distance from start of track to shortest distance intersect
RadiusRailPosition = sqrt(SocketURailInner.^2 - SocketURailDist.^2);


% [mm] Net position of sliders
Slider = dx + RadiusRailPosition;


% [mm] Slider positions for graphing
SliderGlobal = [RailInner(:,1) + Slider.*cosd(Thirds'),...
RailInner(:,2) + Slider.*sind(Thirds'),...
repmat(SocketLHeight,6,1)];


Return = [SliderGlobal(1), SliderGlobal(2), SliderGlobal(3),...
SliderGlobal(4), SliderGlobal(5), SliderGlobal(6);...
SliderGlobal(7), SliderGlobal(8), SliderGlobal(9),...
SliderGlobal(10), SliderGlobal(11), SliderGlobal(12);
SliderGlobal(13), SliderGlobal(14), SliderGlobal(15),...
SliderGlobal(16), SliderGlobal(17), SliderGlobal(18);
Slider(1),Slider(2),Slider(3),Slider(4),Slider(5),Slider(6)];
```

D.1.3 Intra-fraction motion simulation using wiener process

```
clear

% number of histories
n = 1e4;

% [s] duration of the treatment
t = 900;

% [Hz] sample rate, approximately the same as Align RT
s = 4;

% Setting up arrays
X = zeros(n,t*s);
Y = zeros(n,t*s);
Z = zeros(n,t*s);
A = zeros(n,t*s);
B = zeros(n,t*s);
G = zeros(n,t*s);

% [mm/min or deg/min] Amplitudes
XA = 2.25;
YA = 3.15;
ZA = 3.6;
AA = 1.4;
BA = 1.1;
GA = 1.1;

% A coefficient to increase the amplitude of the random motions
AmpControl = 1;
XA = XA*AmpControl;
YA = YA*AmpControl;
ZA = ZA*AmpControl;
AA = AA*AmpControl;
BA = BA*AmpControl;
GA = GA*AmpControl;

% Generation of the random numbers
XR = 2*rand(n,t*s)-1;
YR = 2*rand(n,t*s)-1;
ZR = 2*rand(n,t*s)-1;
AR = 2*rand(n,t*s)-1;
```

```
BR = 2*rand(n,t*s)-1;
GR = 2*rand(n,t*s)-1;

% Scaling the amplitude coefficient to the correct units of time
XR = (XA/s/60)*XR;
YR = (YA/s/60)*YR;
ZR = (ZA/s/60)*ZR;
AR = (AA/s/60)*AR;
BR = (BA/s/60)*BR;
GR = (GA/s/60)*GR;

% Generation of the motions
for j = 2:s*t
  for i = 1:n;
    X(i,j) = X(i,j-1)+XR(i,j);
    Y(i,j) = Y(i,j-1)+YR(i,j);
    Z(i,j) = Z(i,j-1)+ZR(i,j);
    A(i,j) = A(i,j-1)+AR(i,j);
    B(i,j) = B(i,j-1)+BR(i,j);
    G(i,j) = G(i,j-1)+GR(i,j);

    Mag(i,j) = norm([X(i,j),Y(i,j),Z(i,j)]);
  end
end

% Simulating 20 fractions for 5 patients
P1 = reshape(Mag(1:20,:),[1,t*s*20]);
P2 = reshape(Mag(21:40,:),[1,t*s*20]);
P3 = reshape(Mag(41:60,:),[1,t*s*20]);
P4 = reshape(Mag(61:80,:),[1,t*s*20]);
P5 = reshape(Mag(81:100,:),[1,t*s*20]);

% Statistical measurements of the simulation
SXmu = mean(X(:,s*t));
SYmu = mean(Y(:,s*t));
SZmu = mean(Z(:,s*t));
SAmu = mean(A(:,s*t));
SBmu = mean(B(:,s*t));
SGmu = mean(G(:,s*t));

SXsd = std(X(:,s*t));
SYsd = std(Y(:,s*t));
SZsd = std(Z(:,s*t));
```

```
SAsd = std(A(:,s*t));
SBsd = std(B(:,s*t));
SGsd = std(G(:,s*t));

font = 16;
%subplot(2,2,1)
% hold 'on'
% cdfplot(P1)
% cdfplot(P2)
% cdfplot(P3)
% cdfplot(P4)
% cdfplot(P5)
% xlabel('Net vector displacement (mm)')
% ylabel('Probability CDF','FontSize',font)
% set(gca,'Ytick',[0:0.1:1])
% set(gca,'FontSize',font)
% axis([0,2,0,1])
% grid 'off'
% title('Simulated Patient Motion CDF')
% legend('Patient 1','Patient 2','Patient 3','Patient 4','Patient
   5','Location','southeast')
% subplot(2,2,1)
% histogram(Mag(:,j))
% subplot(2,2,2)
% histogram(X(:,j))
% subplot(2,2,3)
% histogram(Y(:,j))
% subplot(2,2,4)
% histogram(Z(:,j))
% hold 'off'

cdfplot(Mag(:,j))
xlabel('Net␣vector␣displacement␣(mm)','FontSize',font)
ylabel('Probability␣CDF','FontSize',font)
set(gca,'Ytick',[0:0.1:1])
set(gca,'FontSize',font)
```

D.1.4 Geometric optimization

```
%% Define limits

% [no unit] Number of random configurations to attempt for a given
    geometry
n = 75;

Increment = 1;

%SocketUHalf = 20:Increment:40;
%PlateMedian = 80:Increment:120;

SocketUHalf = 20;
PlateMedian = 80;
Linkage = 80;

%Linkage = 60:Increment:160;
Height = 40:Increment:120;

LengthSocketUHalf = length(SocketUHalf);
LengthPlateMedian = length(PlateMedian);
LengthLinkage = length(Linkage);
LengthHeight = length(Height);

N = LengthSocketUHalf*LengthPlateMedian*LengthLinkage*LengthHeight;

% [mm] The farthest outward that the plate should move in any one
    direction
TransLim = 10;

% [deg] The farthest that a plate may rotate about any single axis
RotLim = 6;

% [mm] The farthest distance in lateral directions that the sliders
    may move
XLimit = 158;

% [mm] The farthest distance inward that the sliders may come
SLimit = 119;

%% Define the space
% This sections generates the all 729 limiting cases
```

```
% These are permutations of max/min/0 for all six variables of position
% This counts 0,1,2 as min,0,max, and then scales



Space = rand(n,6);

% Scaling to desired translational limits
Space(:,1:3) = TransLim*Space(:,1:3) - TransLim;

% Scaling to desired rotational limits
Space(:,4:6) = RotLim*Space(:,4:6) - RotLim;

t0 = clock;



%% Limit Testing - Random Points

sets = {SocketUHalf,PlateMedian,Linkage,Height};
[w x y z] = ndgrid(sets{:});
GeomSpace = [w(:), x(:),y(:),z(:)];

clear x y z w;

GeometryTest = zeros(1,N);


for k = 1:N

  tic

  for i = 1:n
    SliderPositions(k,i,:,:) = ...
    SixDSliderPositions_Optimizer(Space(i,:),GeomSpace(k,:));

    % This counts how many of the tested positions are physically
       possible
    PossibleSliderLims(i) = isreal(SliderPositions(k,i,:,:));
    for j = 1:6

      % This tests whether position is within the X limit
      if SliderPositions(k,i,2,j) < XLimit &&...
```

```
    SliderPositions(k,i,2,j) > -XLimit
      Test(i,j) = 1;
    else
      Test(i,j) = 0;
    end

    if sum(Test(i,:)) == 6
      WithinXLimits(i) = 1;
    else
      WithinXLimits(i) = 0;
    end

    % This tests whether a position requires a slider to be closer
       in than the
    % specified limit
    if SliderPositions(k,i,4,j) > SLimit
      TestS(i,j) = 1;
    else
      TestS(i,j) = 0;
    end

    if sum(TestS(i,:)) == 6
      WithinSLimits(i) = 1;
    else
      WithinSLimits(i) = 0;
    end

  end

end

% WithinXLimits = logical(WithinXLimits);
% WithinXLimits = logical(WithinXLimits);

PercentPotential(k) = sum(PossibleSliderLims)*100/n;
PercentWithinXLims(k) = sum(WithinXLimits)*100/n;
PercentWithinSLims(k) = sum(WithinSLimits)*100/n;

if PercentPotential(k) > 99 &&...
PercentWithinXLims(k) > 99 && ...
PercentWithinSLims(k) > 99

  GeometryTest1(k) = 1;
```

```
  else
    GeometryTest1(k) = 0;
  end

  t2 = toc;

  Message = Progress(k,N,t2,t0);

  disp(Message)

end

%% Check-in

Potential = [GeometryTest1.*GeomSpace(:,1)';...
GeometryTest1.*GeomSpace(:,2)';...
GeometryTest1.*GeomSpace(:,3)';...
GeometryTest1.*GeomSpace(:,4)'];
Potential(all(Potential==0,2),:)=[];

N = size(Potential,1);

disp(strcat([num2str(N),' possible configurations found']))

%% Redefine the space - Narrowed down

n = 729;

Space = zeros(n,6);

x = 0;
y = 0;
z = 0;
a = 0;
b = 0;
c = 0;

for i = 1:729;

  c = c + 1;

  if c >= 3
  c = 0;
```

```
  b = b + 1;
  end

  if b >= 3
  b = 0;
  a = a + 1;
  end

  if a >= 3
  a = 0;
  z = z + 1;
  end

  if z >= 3
  z = 0;
  y = y + 1;
  end

  if y >= 3
  y = 0;
  x = x + 1;
  end

  Space(i,:) = [x,y,z,a,b,c];


end

%Space = rand(n,6);

clear x y z a b c

% Scaling to desired translational limits
Space(:,1:3) = TransLim*Space(:,1:3) - TransLim;

% Scaling to desired rotational limits
Space(:,4:6) = RotLim*Space(:,4:6) - RotLim;

% Fixing Counting Error
Space(729,:) = [-TransLim,-TransLim,-TransLim,...
-RotLim,-RotLim,-RotLim];

t0 = clock;
```

```
%% Limit Testing - Narrow results

for k = 1:N

  tic

  for i = 1:n
    SliderPositions(k,i,:,:) = ...
    SixDSliderPositions_Optimizer(Space(i,:),Potential(k,:));

    % This counts how many of the tested positions are physically
       possible
    PossibleSliderLims(i) = isreal(SliderPositions(k,i,:,:));
    for j = 1:6

      % This tests whether position is within the X limit
      if SliderPositions(k,i,2,j) < XLimit &&...
      SliderPositions(k,i,2,j) > -XLimit
        Test(i,j) = 1;
      else
        Test(i,j) = 0;
      end

      if sum(Test(i,:)) == 6
        WithinXLimits(i) = 1;
      else
        WithinXLimits(i) = 0;
      end

      % This tests whether a position requires a slider to be closer
         in than the
      % specified limit
      if SliderPositions(k,i,4,j) > SLimit
        TestS(i,j) = 1;
      else
        TestS(i,j) = 0;
      end

      if sum(TestS(i,:)) == 6
        WithinSLimits(i) = 1;
      else
```

```
        WithinSLimits(i) = 0;
      end

    end

  end

  % WithinXLimits = logical(WithinXLimits);
  % WithinXLimits = logical(WithinXLimits);

  PercentPossible(k) = sum(PossibleSliderLims)*100/n;
  PercentWithinXLims(k) = sum(WithinXLimits)*100/n;
  PercentWithinSLims(k) = sum(WithinSLimits)*100/n;

  if PercentPossible(k) > 99 &&...
  PercentWithinXLims(k) > 99 && ...
  PercentWithinSLims(k) > 99

    GeometryTest2(k) = 1;
  else
    GeometryTest2(k) = 0;
  end

  t2 = toc;

  Message = Progress(k,N,t2,t0);

  disp(Message)

end


%% Finalization

% The list of slider configurations are Nx4x6, where N is the number
   of not
% passing conditions. The rows are as follows:
% Row 1: Slider positions on the longitudinal axis (head to toe)
% Row 2: Slider positions on the lateral axis (left to right)
% Row 3: Slider positions on the vertical axis (back to front)
% Row 4: Slider positions on the slider axes, from the center of the
   tracks
% All in mm
```

```
% % This generates a list of slider configurations which are not
   physically
% % possible
% NotPossible = repmat(~PossibleSliderLims,6,1)'.*Space;
% NotPossible( ~any(NotPossible,2), : ) = [];
%
% % This generates a list of slider configurations which are not
   within the
% % limits in the lateral directions
% NotWithinXLims = repmat(~WithinXLimits,6,1)'.*Space;
% NotWithinXLims( ~any(NotWithinXLims,2), : ) = [];
%
% % This generates a list of slider configurations which are not
   within the
% % inner slider limits
% NotWithinSLims = repmat(~WithinSLimits,6,1)'.*Space;
% NotWithinSLims( ~any(NotWithinSLims,2), : ) = [];
%
% % [mm] This is the maximum travel distance of any of the sliders
% %TrackLength =
   max(max(SliderPositions(:,:,4))-min(SliderPositions(:,:,4)));


clear XLimit SLimit a b c i j x y z Test TestS w sets
clear PercentPossible PercentWithinSLims PercentWithinXLims
clear WithinSLims WithinXLims Increment
clear Possible SliderLims SliderPositions Space Message
clear LengthHeight LengthLinkage LengthPlateMedian LengthSocketUHalf
clear PossibleSliderLims


Possible = [GeometryTest2.*Potential(:,1)';...
GeometryTest2.*Potential(:,2)';...
GeometryTest2.*Potential(:,3)';...
GeometryTest2.*Potential(:,4)']';
Possible(all(Possible==0,2),:)=[];


NumPossible = sum(GeometryTest2);

disp(NumPossible)
```

```
clear GeomSpace GeometryTest


c = clock();

filename = strcat('C:\Users\mostyn\Documents\MATLAB\6D␣Mechanical␣
   Couch\Optimization␣Results\',...
num2str(c(1)),'_',...
num2str(c(2)),'_',...
num2str(c(3)),'_',...
num2str(c(4)),...
num2str(c(5)),'_',...
num2str(TransLim),'mm','_',...
num2str(RotLim),'deg');

clear c

save(filename)
```

D.1.5 DOA simulation - dual frequency

This code contains the simulation of the direction of arrival sensor based on a dual 1.5 GHz and 48 GHz model. The code simulates a dual-frequency waveform at a given sampling rate and how the wave would be received at the sensor for a set number of phase configurations for the sensor. Based on the amplitude measurements according to the phase, a direction of arrival for the signal is estimated.

```
%% Rotating Pair Antenna Triangulation
% This program simulates the behavior of a 2D DOA sensor with a target
    in
% 3D space. This version sets out to find the direction of arrival
    with a
% wide search first, and then narrows down to a fine search.

%% Plot Option
Plot = 1;

%% Geometry setup

% [deg] Angular positions used for the sensor
S1.phaselims = [0,180];
S2.phaselims = [133,137];

% Number of phase steps
S1.steps = 500;
S2.steps = 1;

% Number of repetitions
S1.reps = 500;
S2.reps = 1;

% [deg] Direction in longitude

%Long = 90 + 3.5260;
Long = 90 + 0.05*randn(1,1);

% [m] Location of the center of a single sensor
SensLoc = [0,0,0];

% [m] Location of the transmitting beacon
```

```
TransLoc = 2*[cosd(Long+90),sind(Long+90),0];

% [m] Vector connecting the two
ST = SensLoc - TransLoc;

% [m] Distance connecting the center of the sensor to each antenna
SR = 0.05;

%% Wave Properties

% [Hz] Frequency of the waves used for the wide search
S1.freq = 1.4986e9;
S2.freq = 48e9;

% [Arb] Amplitude of the wave used for the wide search
S1.amp = 1;
S2.amp = 1;

% [Amp] Signal to Noise ratio for the wide search
S1.snr = 500;
S2.snr = 1;

% [Hz] Sample Rate of the waves
S1.rate = 2e3*S1.freq;
S2.rate = 2e3*S2.freq;

% [s] Duration of the waves
S1.duration = 100/S1.freq;
S2.duration = 100/S2.freq;

% [s] Period of the waves
S1.period = 1/S1.freq;
S2.period = 1/S2.freq;

% [m/s] Speed of the wave in air, found using the index of refraction
Speed = 2.99792e8/1.00028;

% Number of samples simulated
S1.samplesperwave = S1.period*S1.rate;
S2.samplesperwave = S2.period*S2.rate;

% [m] The wavelength of each wave
S1.wavelength = Speed/S1.freq;
```

```
S2.wavelength = Speed/S2.freq;

% For conviniece of comparison
S = [S1,S2];

%% Vector math for first wave

% This creates the phase arrays, representing the position of each
   antenna
S1.phaseA = linspace(S1.phaselims(1),S1.phaselims(2),S1.steps);
S1.phaseB = S1.phaseA - 180;

% This creates the vectors connecting the center of the sensor to each
% antenna, for every angle specified
SAV = [SR*cosd(S1.phaseA);SR*sind(S1.phaseA);zeros(1,S1.steps)]';
SBV = [SR*cosd(S1.phaseB);SR*sind(S1.phaseB);zeros(1,S1.steps)]';

ST = CoordRepeat(ST,S1.steps);

% This creates the vectors from the transmitter to each antenna for
   every
% angle specified, and finds the resultant distance
TA = ST + SAV;
TB = ST + SBV;

TAd = Magnitude(TA);
TBd = Magnitude(TB);

%clear ST SAV SBV TA TB

%% Wave mechanics
% This section deals with the broadcast and reception of wavelengths

% This finds the minimum number of wavelengths between the transmitter
   and
% the receiver antennas, which determines how much of the wave should
   be
% simulated at each angular position - The number is arbitrary
MinWavesBetween = min([min(TAd),min(TBd)])/S1.wavelength;
WavesAcquired = floor(MinWavesBetween);

% This cuts the number of samples simulated down
WavesToWait = floor(WavesAcquired/2);
```

```
% This determines the statrting point for each wave
WavesToWaitRA = Dist2Waves(TAd,S1.wavelength)+WavesToWait;
WavesToWaitRB = Dist2Waves(TBd,S1.wavelength)+WavesToWait;

% This finds the indicies needed for acquisition
RecAL = ceil(WavesToWaitRA*S1.samplesperwave+1);
RecAU = ceil((WavesAcquired+WavesToWaitRA)*S1.samplesperwave+1);

RecBL = ceil(WavesToWaitRB*S1.samplesperwave+1);
RecBU = ceil((WavesAcquired+WavesToWaitRB)*S1.samplesperwave+1);


% This is the number of samples actually acquired for each position
window = length(RecAL:RecAU-1);

% preallocation of memory
RecSigA = zeros(S1.steps,window);
RecSigB = zeros(S1.steps,window);

% [s] Creation of timespace
t = linspace(0,S1.duration,S1.duration*S1.rate);

% Easy Constant
tau = 2*pi;

t0 = clock;

%% Beginning of random numbers territory

for X = 1:S1.reps

tic;

  % Creation of waveform
  TransSig = S1.amp*sin(t*tau*S1.freq+rand()*tau);

  for j = 1:S1.steps
    RecSigA(j,:) = TransSig(RecAL(j):(RecAL(j) + window - 1));
  end

  for j = 1:S1.steps
    RecSigB(j,:) = TransSig(RecBL(j):(RecBL(j) + window - 1));
```

```
end


% Addition of noise
RecSigA = RecSigA + (1/S1.snr)*randn(size(RecSigA));
RecSigB = RecSigB + (1/S1.snr)*randn(size(RecSigB));

% Interference - simple addition
Interfered = RecSigA + RecSigB;



% These next few lines make an amplitude measurement of the
   interfered
% signal and then massage the data for curve purposes
Fs = S1.rate;

InterferedAmp = zeros(1,S1.steps);
InterferedAmpOld = zeros(1,S1.steps);

InterferedAmpOld = (max((Interfered'),[],1))';

for i = 1:S1.steps;
  I = fft(Interfered(i,:));
  I = I(1:length(Interfered(i,:))/2+1);
  I = I/length(Interfered(i,:));
  I(2:end-1) = 2*I(2:end-1);

  InterferedAmp(i) = max(abs(I));
end


InterferedAmp = medfilt1(InterferedAmp,3);
InterferedAmp = 100*InterferedAmp/max(InterferedAmp);

InterferedAmpOld = medfilt1(InterferedAmpOld,100);
InterferedAmpOld = 100*InterferedAmpOld/max(InterferedAmpOld);


% These lines find the peak of the signal for a single iteration of
   the
% program, and records the mean and standard deviation
[A,B,C,D] = findpeaks(InterferedAmpOld,'MinPeakProminence',10);
```

```
  [dummy,GuessIndex] = max(D);

  if isempty(A)
    Guess(X) = NaN;
  end

  if ~isempty(A)
    Guess(X) = S1.phaseA(B(GuessIndex));
  end


  gaussEqn = 'a*exp(-((x-b)/c)^2)';
  startPoints = [1 90 1];

  if Guess(X) < 45
    fitcurve = fit(S1.phaseA',InterferedAmp',gaussEqn,...
    'Start',startPoints, 'Exclude', S1.phaseA > 90);
  elseif Guess(X) > 135
    fitcurve = fit(S1.phaseA',InterferedAmp',gaussEqn,...
    'Start',startPoints, 'Exclude', S1.phaseA < 90);
  else
    fitcurve = fit(S1.phaseA',InterferedAmp',gaussEqn,...
    'Start',startPoints);
  end

  x(X,:) = coeffvalues(fitcurve);

  Guessnew(X) = x(X,2);

  t2 = toc;


  Message = Progress(X,S1.reps,t2,t0);
  disp(Message);
end


% G1 = abs(mean(Guess-Long));
% G1a = std(Guess);

G2 = mean(abs(Guessnew-Long));
G2a = std(Guessnew);
```

```
% SimAccuracy1 = G1;
% SimPrecision1 = G1a;

SimAccuracy = G2;
SimPrecision = G2a;

% disp(strcat(['The old accuracy was ',...
%    num2str(SimAccuracy1,4),char(176)]))
% disp(strcat(['The old precision was ',...
%    num2str(SimPrecision1,4),char(176)]))

disp(strcat(['The␣accuracy␣was␣␣',...
num2str(SimAccuracy,4),char(176)]))
disp(strcat(['The␣precision␣was␣',...
num2str(SimPrecision,4),char(176)]))


if any(Plot)

  font = 16;

  subplot(2,1,1)
  hold 'off'
  scatter(S1.phaseA,InterferedAmp,'*')
  hold 'on'
  scatter(S1.phaseA,InterferedAmpOld)
  p = plot(fitcurve);
  set(p,'LineWidth',2)

  xlabel(strcat('Angular␣Position␣(',char(176),')'),'FontSize',font)
  ylabel('Amplitude␣(%)','FontSize',font)
  grid('on')
  legend('New␣Signal','Old␣Signal','Gaussian␣Fit')
  axis([S1.phaselims(1),S1.phaselims(2),0,100])

  set(gca,'FontSize',font)
  %set(gca,'XTick',[S1.phaselims(1):tickincrement:S1.phaselims(2)])
  set(gca,'Ytick',[0:20:100])


  subplot(2,1,2)
  hold 'off'
  histogram(Guess,20)
```

```
  hold 'on'
  subplot(2,1,2)
  histogram(Guessnew,20)
  axis([S1.phaselims(1) S1.phaselims(2) 0 S1.reps/4])
  legend('Old Method','New Method')
  xlabel(strcat('Guessed Direction (',char(176),')'),'FontSize',font)
  ylabel('Counts','FontSize',font)
  set(gca,'FontSize',font)
  grid('on')

end

% Big vectors to clear memory
clear RecSigA RecSigB t TransSig I RecAL RecAU RecBL RecBU TAd TBd
clear WavesToWaitRA WavesToWaitRB
clear Interfered x

clear SAV SBV ST TA TB

% Unneeded scalars
clear Fs gaussEqn i j MinWavesBetween Message Plot Speed startPoints
clear SR tau WavesAcquired WavesToWait window X

clear G1 G1a G2 G2a

filename =
   strcat('C:\Users\Mark\Documents\MATLAB\Results\RotPairV6\',...
num2str(c(1)),'_',...
num2str(c(2)),'_',...
num2str(c(3)),'_',...
num2str(c(4)),...
num2str(c(5)));

save(filename)
```

D.1.6 DOA simulation - single frequency

This code is similar to that in the previous section, but obtains comparable results using only a single emitted frequency by using more sophisticated signal analysis techniques. The code simulates a single-frequency waveform at a given sampling rate and how the wave would be received at the sensor for a set number of phase configurations for the sensor. Based on the amplitude measurements according to the phase, a direction of arrival for the signal is estimated. This code uses more sophisticated signal analysis techniques than those given in the dual frequency code for comparable error.

```
%% Rotating Pair Antenna Triangulation
% This program simulates the behavior of a 2D DOA sensor with a target
    in
% 3D space. This version sets out to find the direction of arrival
    with a
% wide search first, and then narrows down to a fine search.

t0 = clock;

%% Plot Option

% 1 - Angle Signal Response
% 2 - Histogram of Guesses
% 3 - Transmitter Angle Signal Response

plottype = 2;

%% Setup of Monte Carlo

% [deg] Angular limits used for the sensor
S.phaselims = [90,100];

% Number of phase steps
S.steps = 200;

% Number of repetitions
S.reps = 500;

% [Amp] Signal to noise ratio
S.snr = 2000;
```

```
% [deg] Window of fitting
Window = 180;

% [deg] Center of fitting window
Direction = [90];

% Arbitrary Counter
k = 0;

%% Wave Properties

% [Hz] Frequency of the waves used for the wide search
S.freq = 1.4986e9;

% [Arb] Amplitude of the wave used for the wide search
S.amp = 1;

% [Hz] Sample Rate of the waves
S.rate = 2e3*S.freq;

% [s] Duration of the waves
S.duration = 100/S.freq;

% [s] Period of the waves
S.period = 1/S.freq;

% [m/s] Speed of the wave in air, found using the index of refraction
Speed = 2.99792e8/1.00028;

% Number of samples simulated
S.samplesperwave = S.period*S.rate;

% [m] The wavelength of each wave
S.wavelength = Speed/S.freq;


%% Geometry setup

for Y = Direction
  k = k+1;

  tic;
```

```
for X = 1:S.reps
  % [deg] Direction in longitude

  Long(X) = 90 + 3.5260;
  %Long(X) = 90 + 5*randn(1,1);

  % [m] Location of the center of a single sensor
  SensLoc = [0,0,0];

  % [m] Location of the transmitting beacon
  TransLoc(X,:) = 2*[cosd(Long(X)+90),sind(Long(X)+90),0];

  % [m] Vector connecting the two
  ST = SensLoc - TransLoc(X,:);

  % [m] Distance connecting the center of the sensor to each antenna
  SR = 0.05;


  %% Vector math for first wave


  % This creates the phase arrays, representing the position of each
     antenna
  S.phaseA = linspace(S.phaselims(1),S.phaselims(2),S.steps);
  S.phaseB = S.phaseA - 180;

  % This creates the vectors connecting the center of the sensor to
     each
  % antenna, for every angle specified
  SAV = [SR*cosd(S.phaseA);SR*sind(S.phaseA);zeros(1,S.steps)]';
  SBV = [SR*cosd(S.phaseB);SR*sind(S.phaseB);zeros(1,S.steps)]';

  ST = CoordRepeat(ST,S.steps);

  % This creates the vectors from the transmitter to each antenna for
     every
  % angle specified, and finds the resultant distance
  TA = ST + SAV;
  TB = ST + SBV;

  TAd = Magnitude(TA);
```

```
TBd = Magnitude(TB);

%clear ST SAV SBV TA TB

%% Wave mechanics
% This section deals with the broadcast and reception of wavelengths

% This finds the minimum number of wavelengths between the
   transmitter and
% the receiver antennas, which determines how much of the wave
   should be
% simulated at each angular position - The number is arbitrary
MinWavesBetween = min([min(TAd),min(TBd)])/S.wavelength;
WavesAcquired = floor(MinWavesBetween);

% This cuts the number of samples simulated down
WavesToWait = floor(WavesAcquired/2);

% This determines the statrting point for each wave
WavesToWaitRA = Dist2Waves(TAd,S.wavelength)+WavesToWait;
WavesToWaitRB = Dist2Waves(TBd,S.wavelength)+WavesToWait;

% This finds the indicies needed for acquisition
RecAL = ceil(WavesToWaitRA*S.samplesperwave+1);
RecAU = ceil((WavesAcquired+WavesToWaitRA)*S.samplesperwave+1);

RecBL = ceil(WavesToWaitRB*S.samplesperwave+1);
RecBU = ceil((WavesAcquired+WavesToWaitRB)*S.samplesperwave+1);


% This is the number of samples actually acquired for each position
window = length(RecAL:RecAU-1);

% preallocation of memory
RecSigA = zeros(S.steps,window);
RecSigB = zeros(S.steps,window);

% [s] Creation of timespace
t = linspace(0,S.duration,S.duration*S.rate);

% Easy Constant
tau = 2*pi;
```

```
%% Beginning of signal transmissions

% Creation of waveform
TransSig = S.amp*sin(t*tau*S.freq+rand()*tau);

for j = 1:S.steps
  RecSigA(j,:) = TransSig(RecAL(j):(RecAL(j) + window - 1));
end

for j = 1:S.steps
  RecSigB(j,:) = TransSig(RecBL(j):(RecBL(j) + window - 1));
end


% Addition of noise
RecSigA = RecSigA + (1/S.snr)*randn(size(RecSigA));
RecSigB = RecSigB + (1/S.snr)*randn(size(RecSigB));

% Interference - simple addition
Interfered = RecSigA + RecSigB;


% These next few lines make an amplitude measurement of the
   interfered
% signal and then massage the data for curve purposes
Fs = S.rate;

InterferedAmp = zeros(1,S.steps);

for i = 1:S.steps;
  I = fft(Interfered(i,:));
  I = I(1:length(Interfered(i,:))/2+1);
  I = I/length(Interfered(i,:));
  I(2:end-1) = 2*I(2:end-1);

  InterferedAmp(i) = max(abs(I));
end


InterferedAmp = medfilt1(InterferedAmp,3);
InterferedAmp = 100*InterferedAmp/max(InterferedAmp);
```

```
  % These lines find the peak of the signal for a single iteration of
     the
  % program, and records the mean and standard deviation
  [A,B,C,D] = findpeaks(InterferedAmp,'MinPeakProminence',10);
  [dummy,GuessIndex] = max(D);

  if isempty(A)
    Guess(X) = NaN;
  end

  if ~isempty(A)
    Guess(X) = S.phaseA(B(GuessIndex));
  end


  gaussEqn = 'a*exp(-((x-b)/c)^2)';
  startPoints = [100 90 50];

  if Guess(X) < 45
    fitcurve = fit(S.phaseA',InterferedAmp',gaussEqn,...
    'Start',startPoints, 'Exclude', S.phaseA > 90);
  elseif Guess(X) > 135
    fitcurve = fit(S.phaseA',InterferedAmp',gaussEqn,...
    'Start',startPoints, 'Exclude', S.phaseA < 90);
  else
    fitcurve = fit(S.phaseA',InterferedAmp',gaussEqn,...
    'Start',startPoints);
  end


  %       fitcurve = fit(S.phaseA',InterferedAmp',gaussEqn,...
  %       'Start',startPoints, 'Exclude', S.phaseA > Guess(X) +
     Window/2,...
  %       'Exclude', S.phaseA < Guess(X) - Window/2);

  x(X,:) = coeffvalues(fitcurve);

  Guessnew(X) = x(X,2);
end

t2 = toc;
```

```
Message = Progress(k,length(Direction),t2,t0);
disp(Message);


SimAccuracy(k) = mean(abs(Guessnew-Long));
SimPrecision(k) = std(Guessnew-Long);

end

% disp(strcat(['The accuracy was ',...
%    num2str(SimAccuracy,4),char(176)]))
% disp(strcat(['The precision was ',...
%    num2str(SimPrecision,4),char(176)]))


font = 16;

switch plottype

case 1
  hold 'off'
  scatter(S.phaseA,InterferedAmp)
  hold 'on'
  p = plot(fitcurve);
  set(p,'LineWidth',2)

  xlabel(strcat('Angular␣Position␣(',char(176),')'),'FontSize',font)
  ylabel('Amplitude␣(%)','FontSize',font)
  grid('on')
  legend('Signal','Gaussian␣Fit','Location','southeast')
  axis([S.phaselims(1),S.phaselims(2),0,120])
  set(gca,'FontSize',font)
  set(gca,'Ytick',[0:20:120])
  set(gca,'Xtick',[0:30:180])

case 2
  cdfplot(abs(Guessnew-Long))
  %axis([0 ceil(max(Guessnew-Long))/5 0 1])
  xlabel(strcat('Estimate␣Accuracy␣(',char(176),')'),'FontSize',font)
  ylabel('Probability␣CDF','FontSize',font)
  set(gca,'Ytick',[0:0.1:1])
  set(gca,'FontSize',font)
```

```
  grid('on')

case 3
  semilogy(Direction,SimAccuracy,Direction,SimPrecision)
  xlabel(strcat('Trans␣Position␣(',char(176),')'),'FontSize',font)
  ylabel(strcat('Quantity␣(',char(176),')'),'FontSize',font)
  grid('on')
  set(gca,'FontSize',font)
  legend('Accuracy','Precision')

end

% Big vectors to clear memory
clear RecSigA RecSigB t TransSig I RecAL RecAU RecBL RecBU TAd TBd
clear WavesToWaitRA WavesToWaitRB
clear Interfered

clear SAV SBV ST TA TB

% Unneeded scalars
clear Fs gaussEqn i j MinWavesBetween Message Plot Speed startPoints
clear SR tau WavesAcquired WavesToWait window X

clear G1 G1a G2 G2a

c = clock();

filename = strcat('C:\Users\mostyn\Documents\MATLAB\RF␣
   System\Results\',...
num2str(c(1)),'_',...
num2str(c(2)),'_',...
num2str(c(3)),'_',...
num2str(c(4)),...
num2str(c(5)));

save(filename)
```

D.1.7 RF angulation

This code was used to estimate the relationship between uncertainty of the direction of arrival sensors, the number of sensors used to make a localization estimate, compared to the uncertainty of the localization estimate. It simulates n randomly positioned direction of arrival sensors in a hemispherical shell around a fixed point of reference. The sensors are then simulated to have made an error of measurement the direction of arrival. The mean angular uncertainty is then compared to the error in the least-squares fit for the localization estimate based on angulation. This is done for $n \times 10^(x)$ sensors to find the relationship between number of sensors and localization error for various degrees of angular measurement uncertainty.

```
%% Angulation in 3D
% This simulation finds the error in position estimation of a
   transmitter
% as a function of angular uncertainty in 3D space. The position is
% estimated by a least-squares of the intersection of lines in 3D
   space.
% This serves as a surrogate to angulation.

disp('Start Time')
c = clock;
d = [c(4), c(5), c(6)];
disp(d)


NumRecs = [5,50,500];

for Recs = NumRecs

% [m] The true position of the transmitter
Origin = [0,0,0];

% [m] This specifies the locations of the receivers in a particular
   setting
MeanDist = 2;
StartCoord = [0,0,MeanDist];
RecLoc = CoordRepeat(StartCoord,NumRecs);
```

```
R = MeanDist+0.1*rand(NumRecs,1)-.05;
Theta = rand(NumRecs,1)*180;
Phi = rand(NumRecs,1)*360;

RecLoc(:,1) = R.*sind(Theta).*cosd(Phi);
RecLoc(:,2) = R.*sind(Theta).*sind(Phi);
RecLoc(:,3) = R.*cosd(Theta);

% This is used for creating simulated misses, by varying the width of
% the normal distribution about the origin
TolArray = ceil(1.5.^(2:28));

for Tol = TolArray

  % This is the number of repetitions to average over
  for Rep = 1:5000
    % This is the creation of the transmitter's location as
    % measured by each receiver. A gaussian distribution of varying
    % width is added to the origin for n receivers the size is 3xn
    MissedArray = (1/Tol)*randn(size(RecLoc));
    MissedArray = [Origin(1)+MissedArray(:,1),...
    Origin(2)+MissedArray(:,2),...
    Origin(3)+MissedArray(:,3)];

    % This creates a 3xn array of the transmitter's true position
    % for comparison later
    TrueArray = [Origin(1)+zeros(length(MissedArray(:,1)),1),...
    Origin(2)+zeros(length(MissedArray(:,2)),1),...
    Origin(3)+zeros(length(MissedArray(:,3)),1)];

    % This represents the vector from the transmitter to the
    % receiver, or RecLoc to the origin. This is created for both
    % the true transmitter position and the flawed measurement. The
    % next lines are used to normalize these vectors. The size of
    % these vectors is 3xn
    D = MissedArray-RecLoc;
    Dt = TrueArray-RecLoc;

    Dmag = [sqrt(sum(D.^2,2)),sqrt(sum(D.^2,2)),sqrt(sum(D.^2,2))];
    Dtmag = [sqrt(sum(Dt.^2,2)),sqrt(sum(Dt.^2,2)),sqrt(sum(Dt.^2,2))];

    D = D./Dmag;
    Dt = Dt./Dtmag;
```

```
  % Preallocation of memory
  x = zeros(3,3,length(Dmag));
  y = zeros(3,length(Dmag));

  % This creates the two halves of the least squares estimate
  % x represents I-vv', where I is an identity matrix, and vv' is
  % the unit vector for each line times its transpose. The size
  % is 3x3xn.
  % y represents (I-vv')p, where p is the end point of the
  % vector. It works best for this to be the MissedArray
  for i = 1:length(Dmag)
  x(:,:,i) = eye(3) - D(i,:)*(D(i,:))';
  y(:,i) = x(:,:,i)*(MissedArray(i,:))';
  end

  % [m] This sums over all of these terms and completes the least
  % squares estimate, finding the closest point of intersection
  % for n lines
  TransLoc = inv(sum(x,3))*sum(y,2);

  % [mm] This is the vector shift of the estimated position from
  % least squares to the true origin position.
  Error(Tol,Rep) = 1000*sqrt(sum((TransLoc-Origin').^2));

  % [deg] This compares the angle of the true vector to the
  % missed vector and takes the average over all n lines
  for i = 1:length(Dmag)
  Diff(i) = acosd((D(i,:)*Dt(i,:)'));
  end

  Miss(Tol,Rep) = mean(Diff);

end

Error(~any(Error,2),:) = [];
Miss(~any(Miss,2),:) = [];

%    % [deg] This is the mean angular error averaged over each
    normal width
%    MeanMiss1 = mean(Miss,2);
%    % [mm] This is the mean magnitude error of the position estimate
%    % averaged over each width of the normal distribution
```

```
    %    MeanError1 = mean(Error,2);

    disp(floor(log(Tol)/log(1.5)))
    c = clock;
    d = [c(4), c(5), c(6)];
    disp(d)

end
Error(~any(Error,2),:) = [];
Miss(~any(Miss,2),:) = [];

Error = log10(Error);
Miss = log10(Miss);

% [deg] This is the mean angular error averaged over each normal width
MeanMiss(:,Recs) = mean(Miss,2);
% [mm] This is the mean magnitude error of the position estimate
% averaged over each width of the normal distribution
MeanError(:,Recs) = mean(Error,2);
STDError(:,Recs) = std(Error,1,2);

clear Miss Error

MeanMiss( ~any(MeanMiss,2),:) = [];
MeanError( ~any(MeanError,2),:) = [];
STDError( ~any(STDError,2),:) = [];

disp(Recs)
c = clock;
d = [c(4), c(5), c(6)];
disp(d)

end

MeanMiss = real(MeanMiss)';
MeanError = real(MeanError)';
STDError = STDError';

MeanMiss( ~any(MeanMiss,2),:) = [];
MeanError( ~any(MeanError,2),:) = [];
STDError( ~any(STDError,2),:) = [];

MeanMiss = MeanMiss';
```

```
MeanError = MeanError';
STDError = STDError';

Bounds = [MeanError+2*STDError, MeanError-2*STDError];

plot(MeanMiss,MeanError,'LineWidth',3)
grid('on')
xlabel('Angle uncertainty (log(deg))')
ylabel('Error to target (log(mm))')
axis([-3 1 -2 2])
legend(num2str(NumRecs(:)))

hold on

plot(MeanMiss(:,1),Bounds(:,1),'b:',MeanMiss(:,1),Bounds(:,4),'b:','LineWidth',2)
plot(MeanMiss(:,2),Bounds(:,2),'r:',MeanMiss(:,2),Bounds(:,5),'r:','LineWidth',2)
plot(MeanMiss(:,3),Bounds(:,3),'g:',MeanMiss(:,3),Bounds(:,6),'g:','LineWidth',2)
```

D.2 C

D.2.1 Motor calibration

This code is used for calibrating the home position of the first-generation 6D platform via Arduino board. The rest position was specified the point when the medial face of every slider 55 mm from the track brace. Reseting the origin resets the 6D location value stored on the on-board memory.

```
/*
Slider Calibration Program
Commericialization Project 2015-1016
Thomas Dwyer
Heavily Modified by Mark Ostyn

Units in mm
Degrees in Rads
Requires LiquidCrystal, Encoder, and EEPROM Library
*/

//Libraries
#include <LiquidCrystal.h>
#include <Encoder.h>
#include <EEPROM.h>

//Pinouts
int allOutputs[] =
    {17,16,48,54,47,55,57,56,62,23,22,24,26,25,27,28,29,39,7,44,19,42,18,38,41,40,11,12,

#define M1_step 17 //X
#define M1_dir 16
#define M1_en 48
```

```
#define M2_step 54 //Y
#define M2_dir 47
#define M2_en 55

#define M3_step 57 //Z
#define M3_dir 56
#define M3_en 62

#define M4_step 23 //E0
#define M4_dir 22
#define M4_en 24

#define M5_step 26 //E1
#define M5_dir 25
#define M5_en 27

#define M6_step 29 //E2
#define M6_dir 28
#define M6_en 39

#define FanToggle 7

#define BEEPER   44
#define LCD_RS   19
#define LCD_ENABLE 42
#define LCD_D4   18
#define LCD_D5   38
#define LCD_D6   41
#define LCD_D7   40
#define ButtonLeft 11
#define ButtonRight 12
#define ButtonClick 43
```

```
LiquidCrystal lcd(LCD_RS, LCD_ENABLE, LCD_D4, LCD_D5, LCD_D6, LCD_D7);
Encoder myEnc(ButtonLeft, ButtonRight);

//LCD Vars
int LastPosition = 0;
int CursorPosition = 0;
int CursorPositionLast = 0;
boolean UpdateLCD = false;
boolean clicked = false;
int TurnDirection = 1;

void setup(){
Serial.begin(9600);

//Prepare LCD GUI
lcd.begin(20,4);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("6D Robotic Platform");
lcd.setCursor(0,1);
lcd.print("Calibration Program");

delay(1000);

lcd.clear();
lcd.setCursor(0,0);
lcd.print("WARNING!");
lcd.setCursor(0,1);
lcd.print("Stepping from M6 to");
lcd.setCursor(0,2);
lcd.print("M1 resets internal");
lcd.setCursor(0,3);
```

```
lcd.print("position memory");

delay(5000);

lcd.clear();
lcd.setCursor(0,0);
lcd.print("Adjust Motor #:");

lcd.setCursor(0,2);
lcd.print("1");

//Set motor pins to Outputs
for(int m = 0; m < sizeof(allOutputs); m++){
pinMode(allOutputs[m], OUTPUT);
digitalWrite(allOutputs[m], LOW);}

//Encoder inputs
pinMode(ButtonLeft, INPUT_PULLUP);
pinMode(ButtonRight, INPUT_PULLUP);
pinMode(ButtonClick, INPUT_PULLUP);
digitalWrite(FanToggle, HIGH); }

void loop(){
  //LCD Control
  double Position = myEnc.read();

  if(Position > LastPosition + 1){
    TurnDirection = 1;
    ActivateStepperMotors();
    delay(100);
    LastPosition = Position;}

  if(Position < LastPosition - 1){
```

```
  TurnDirection = 0;
  ActivateStepperMotors();
  delay(100);
  LastPosition = Position;}

if(UpdateLCD){
  lcd.setCursor(0,2);
  lcd.print((CursorPosition+1));
  UpdateLCD = false;}

if(!digitalRead(ButtonClick) && !clicked){
  clicked = true;
  CursorPosition++;
  UpdateLCD = true;}

if(digitalRead(ButtonClick) && clicked){
  clicked = false;}

if(CursorPosition > 5){
  CursorPosition = 0;

  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Setting Internal");
  lcd.setCursor(0,1);
  lcd.print("Position");

  EEPROM.write(0, 127); // X Position [mm]
  EEPROM.write(8, 127); // Y Position [mm]
  EEPROM.write(16, 127); // Z Position [mm]

  EEPROM.write(24, 127); // alpha [deg]
  EEPROM.write(32, 127); // beta [deg]
```

```
    EEPROM.write(40, 127); // gamma [deg]

    delay(1000);

    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Adjust Motor #:");
    lcd.setCursor(0,2);
    lcd.print("1");}}

void ActivateStepperMotors(){

  int motorArray1[] =
     {M1_step,M2_step,M3_step,M4_step,M5_step,M6_step};
  int motorDirArray[] = {M1_dir,M2_dir,M3_dir,M4_dir,M5_dir,M6_dir};
  Serial.println("stepping");

  for(int s = 0; s < 20; s++){
    Serial.println(s);
    digitalWrite(motorArray1[CursorPosition], HIGH);
    digitalWrite(motorDirArray[CursorPosition], TurnDirection);
    delay(1);
    digitalWrite(motorArray1[CursorPosition], LOW);}}
```

D.2.2 6D platform positioning

This code is used for controlling the first-generation 6D platform via Arduino board with output to an LCD monitor. The program was controlled with a two-direction dial and single button press. The code allows movement to 20 preset 6D positions. The program assumes that the platform has been calibrated using the previous program. The code stores the current position in the on-board non-volatile memory, and is updated with each motor step.

```
/*
Plate Position Control V5
Senior Design 2015-1016 / Commercialization Project
Thomas Dwyer and Mark Ostyn

Length in mm
Arcs in Rads
Requires LiquidCrystal, Encoder, and EEPROM Library
*/

// Libraries
#include <LiquidCrystal.h>
#include <Encoder.h>
#include <EEPROM.h>

// Pinouts
int allOutputs[] =
    {17,16,48,54,47,55,57,56,62,23,22,24,26,25,27,28,29,39,7,44,19,42,18,38,41,40,11,12,



#define M1_step 17 //X
#define M1_dir 16
#define M1_en 48
```

```
#define M2_step 54 //Y
#define M2_dir 47
#define M2_en 55

#define M3_step 57 //Z
#define M3_dir 56
#define M3_en 62

#define M4_step 23 //E0
#define M4_dir 22
#define M4_en 24

#define M5_step 26 //E1
#define M5_dir 25
#define M5_en 27

#define M6_step 29 //E2
#define M6_dir 28
#define M6_en 39

#define fan0 7

#define BEEPER   44
#define LCD_RS   19
#define LCD_ENABLE 42
#define LCD_D4   18
#define LCD_D5   38
#define LCD_D6   41
#define LCD_D7   40
#define BTN_EN1  11
#define BTN_EN2  12
#define BTN_ENC  43
```

```
LiquidCrystal lcd(LCD_RS, LCD_ENABLE, LCD_D4, LCD_D5, LCD_D6, LCD_D7);
Encoder myEnc(BTN_EN1, BTN_EN2);


// Global Vars
  const double pi = 3.1415926535898;
  const double degrad = pi/180;
  const double Height = -83;

  const double SocketU[6][3] = {{120, -40,       -19},
                   {120,        40,       -19},
                   {-25.35898385, 123.9230485, -19},
                   { -94.64101615, 83.92304845, -19},
                   { -94.64101615, -83.92304845, -19},
                   {-25.35898385, -123.9230485, -19}};

  const double InnerRef[6][3] = {{ 0, -25, Height},
                   { 0,        25,  Height},
                   { 21.65063509, 12.5, Height},
                   {-21.65063509, -12.5, Height},
                   {-21.65063509, 12.5, Height},
                   { 21.65063509, -12.5, Height}};

  const double OuterRef[6][3] = {{ 250, -25,   Height},
                   { 250,      25,       Height},
                   {-103.3493649, 229.0063509, Height},
                   {-146.6506351, 204.0063509, Height},
                   {-146.6506351, -204.0063509, Height},
                   {-103.3493649, -229.0063509, Height}};

  const double RailRefVect[6][3] = {{ 250, 0, 0},
                   { 250, 0,       0},
```

```
                    {-125, 216.5063509, 0},
                    {-125, 216.5063509, 0},
                    {-125, -216.5063509, 0},
                    {-125, -216.5063509, 0}};

// Read internally stored position and set as current position
int TempCurrentPosition[6] = {EEPROM.read(0),
                  EEPROM.read(8),
                  EEPROM.read(16),
                  EEPROM.read(24),
                  EEPROM.read(32),
                  EEPROM.read(40)};

double CurrentPosition[6] = {0,0,0,0,0,0};
double CurrentSldPos[6] = {0,0,0,0,0,0};
double TargetPosition[] = {0, 0, 0, 0, 0, 0};
double TargetDist[6] = {0,0,0,0,0,0};
double NewPos[6] = {0,0,0,0,0,0};
double NewSldPos[6] = {0,0,0,0,0,0};
double TargetMaxDist = 0;
double MaxStep = 0.05; //Maximum plate displacement per step (mm,
   smaller = higher resolution)
double StepFactor = 0;
double P[6] = {0,0,0,0,0,0};

// LCD Vars
int lastPos = 0;
int cursorPos = 0;
int cursorPosLast = 0;
boolean lcdUpdate = false;
boolean clicked = false;

void setup(){
```

```
Serial.begin(9600);

//Prepare LCD GUI
lcd.begin(20,4);
lcd.begin(20,4);
lcd.setCursor(0,0);
lcd.print("6D Robotic Platform");
lcd.setCursor(0,1);
lcd.print("VCU");

delay(1000);

lcd.clear();
lcd.setCursor(0,0);
lcd.print("Current Position:");

lcd.setCursor(0,2);
lcd.print("*");
lcd.setCursor(0,3);
lcd.print("0");
lcd.setCursor(1,3);
lcd.print("1");
lcd.setCursor(2,3);
lcd.print("2");
lcd.setCursor(3,3);
lcd.print("3");
lcd.setCursor(4,3);
lcd.print("4");
lcd.setCursor(5,3);
lcd.print("5");
lcd.setCursor(6,3);
lcd.print("6");
lcd.setCursor(7,3);
```

```
  lcd.print("7");
  lcd.setCursor(8,3);
  lcd.print("8");
  lcd.setCursor(9,3);
  lcd.print("9");
  lcd.setCursor(10,3);
  lcd.print("A");
  lcd.setCursor(11,3);
  lcd.print("B");
  lcd.setCursor(12,3);
  lcd.print("C");
  lcd.setCursor(13,3);
  lcd.print("D");
  lcd.setCursor(14,3);
  lcd.print("E");
  lcd.setCursor(15,3);
  lcd.print("F");
  lcd.setCursor(16,3);
  lcd.print("G");
  lcd.setCursor(17,3);
  lcd.print("H");
  lcd.setCursor(18,3);
  lcd.print("I");
  lcd.setCursor(19,3);
  lcd.print("J");


// Set motor pins to Outputs
for(int m = 0; m < sizeof(allOutputs); m++){
  pinMode(allOutputs[m], OUTPUT);
  digitalWrite(allOutputs[m], LOW);}

// Encoder inputs
```

```
pinMode(BTN_EN1, INPUT_PULLUP);
pinMode(BTN_EN2, INPUT_PULLUP);
pinMode(BTN_ENC, INPUT_PULLUP);

// Start Fan
digitalWrite(fan0, HIGH);

// Convert the single byte stored position to mm and deg
for(int i=0; i<3; i++){
  CurrentPosition[i] = round((TempCurrentPosition[i]-127)/8.53);}

for(int i=3; i<6; i++){
  CurrentPosition[i] = round((TempCurrentPosition[i]-127)/4.26)/10;}

calculate(CurrentPosition[0],CurrentPosition[1],CurrentPosition[2],CurrentPosition[3],C

for(int i = 0; i < 6; i++){
  CurrentSldPos[i] = P[i];}}


void loop(){
// Get distance and maximum between current and target position
TargetMaxDist = 0;

for(int i = 0; i < 6; i++){
  TargetDist[i] = TargetPosition[i]-CurrentPosition[i];

  if(abs(TargetDist[i]) > abs(TargetMaxDist)){
    TargetMaxDist = abs(TargetDist[i]);}}


// Find intermediate position between current and target positions
   with change <= to MaxStep
```

```
if(TargetMaxDist != 0){
  StepFactor = MaxStep/TargetMaxDist;
  for(int i = 0; i < 6; i++){
    if(abs(TargetDist[i]) < MaxStep){
      NewPos[i] = TargetPosition[i];}
    else{
      NewPos[i] = CurrentPosition[i] + (TargetDist[i]*StepFactor);}}

  int TempPos[6];

  calculate(NewPos[0],NewPos[1],NewPos[2],NewPos[3],NewPos[4],NewPos[5]);

  for(int i = 0; i < 3; i++){
    TempPos[i] = round(NewPos[i]*8.53)+127; }

  for(int i = 3; i < 6; i++){
    TempPos[i] = round(NewPos[i]*10*4.26)+127;}

  for(int i = 0; i < 5; i++){
    EEPROM.write(8*i, TempPos[i]);}

  for(int i = 0; i < 6; i++){
    NewSldPos[i] = P[i];}

  ActivateStepperMotors();}

//LCD Control
double pos = myEnc.read();

if(pos > lastPos + 1){
if(cursorPos < 19){
cursorPosLast = cursorPos;
cursorPos++;
```

```
lcdUpdate = true;}

lastPos = pos;}

if(pos < lastPos - 1){
  if(cursorPos > 0){
    cursorPosLast = cursorPos;
    cursorPos--;
    lcdUpdate = true;}
  lastPos = pos;}

if(lcdUpdate){
  lcd.setCursor(cursorPosLast,2);
  lcd.print(" ");
  lcd.setCursor(cursorPos,2);
  lcd.print("*");
  lcdUpdate = false;}

if(!digitalRead(BTN_ENC) && !clicked){
  clicked = true;

  // Stored example positions
  if(cursorPos == 0){
    TargetPosition[0] = 0; TargetPosition[1] = 0; TargetPosition[2] =
       0; TargetPosition[3] = 0; TargetPosition[4] = 0;
       TargetPosition[5] = 0;}

  if(cursorPos == 1){
    TargetPosition[0] = 10; TargetPosition[1] = 0; TargetPosition[2] =
       0; TargetPosition[3] = 0; TargetPosition[4] = 0;
       TargetPosition[5] = 0;}

  if(cursorPos == 2){
```

```
  TargetPosition[0] = 0.9; TargetPosition[1] = -3.9;
     TargetPosition[2] = -5.8; TargetPosition[3] = 1.9;
     TargetPosition[4] = -1.6; TargetPosition[5] = 0.5;}

if(cursorPos == 3){
  TargetPosition[0] = -4.1; TargetPosition[1] = 0.2;
     TargetPosition[2] = -3.9; TargetPosition[3] = -0.2;
     TargetPosition[4] = -1.0; TargetPosition[5] = 0.7;}

if(cursorPos == 4){
  TargetPosition[0] = 4.9; TargetPosition[1] = 0.2;
     TargetPosition[2] = -0.6; TargetPosition[3] = -1.5;
     TargetPosition[4] = 1.2; TargetPosition[5] = -0.4;}

if(cursorPos == 5){
  TargetPosition[0] = -6.2; TargetPosition[1] = 6.4;
     TargetPosition[2] = -5.4; TargetPosition[3] = -1.0;
     TargetPosition[4] = -1.9; TargetPosition[5] = -0.5;}

if(cursorPos == 6){
  TargetPosition[0] = 3.7; TargetPosition[1] = 5.9;
     TargetPosition[2] = 6.9; TargetPosition[3] = -0.4;
     TargetPosition[4] = 1.7; TargetPosition[5] = 2.0;}

if(cursorPos == 7){
  TargetPosition[0] = -6.3; TargetPosition[1] = 2.9;
     TargetPosition[2] = -6.1; TargetPosition[3] = 0.4;
     TargetPosition[4] = 0.9; TargetPosition[5] = -1.9;}

if(cursorPos == 8){
  TargetPosition[0] = -2.6; TargetPosition[1] = 2.4;
     TargetPosition[2] = -5.5; TargetPosition[3] = -0.9;
     TargetPosition[4] = 0; TargetPosition[5] = 1.5;}
```

```
if(cursorPos == 9){
  TargetPosition[0] = 2.5; TargetPosition[1] = 6.2;
     TargetPosition[2] = -6.6; TargetPosition[3] = 0.4;
     TargetPosition[4] = 0.3; TargetPosition[5] = 1.7;}

if(cursorPos == 9){
  TargetPosition[0] = 5.6; TargetPosition[1] = 0.7;
     TargetPosition[2] = -5.5; TargetPosition[3] = 0.8;
     TargetPosition[4] = -1.1; TargetPosition[5] = 1.2;}

if(cursorPos == 10){
  TargetPosition[0] = -8.3; TargetPosition[1] = -3.0;
     TargetPosition[2] = -1.3; TargetPosition[3] = -1.1;
     TargetPosition[4] = -0.2; TargetPosition[5] = -1.6;}

if(cursorPos == 11){
  TargetPosition[0] = 8.6; TargetPosition[1] = 8.8;
     TargetPosition[2] = -3.8; TargetPosition[3] = -1.5;
     TargetPosition[4] = 1.9; TargetPosition[5] = -0.9;}

if(cursorPos == 12){
  TargetPosition[0] = 5.5; TargetPosition[1] = 7.5;
     TargetPosition[2] = 8.5; TargetPosition[3] = -0.8;
     TargetPosition[4] = 0.2; TargetPosition[5] = 0.7;}

if(cursorPos == 13){
  TargetPosition[0] =-0.3; TargetPosition[1] = 1.0;
     TargetPosition[2] = -1.4; TargetPosition[3] = -0.7;
     TargetPosition[4] = 0.1; TargetPosition[5] = 0.7;}

if(cursorPos == 14){
  TargetPosition[0] = -0.3; TargetPosition[1] = 1.0;
```

```
        TargetPosition[2] = -1.4; TargetPosition[3] = -0.7;
        TargetPosition[4] = 0.1; TargetPosition[5] = -0.7;}

  if(cursorPos == 15){
    TargetPosition[0] = -1.3; TargetPosition[1] = 2.5;
        TargetPosition[2] = -6.3; TargetPosition[3] = -0.3;
        TargetPosition[4] = -1.1; TargetPosition[5] = -1.5;}

  if(cursorPos == 16){
    TargetPosition[0] = -1.0; TargetPosition[1] = 1.7;
        TargetPosition[2] = 8.1; TargetPosition[3] = 0;
        TargetPosition[4] = 0; TargetPosition[5] = 0.9;}

  if(cursorPos == 17){
    TargetPosition[0] = 1; TargetPosition[1] = 1; TargetPosition[2] =
        1; TargetPosition[3] = .2; TargetPosition[4] = .2;
        TargetPosition[5] = .2;}

  if(cursorPos == 18){
    TargetPosition[0] = 0; TargetPosition[1] = 0; TargetPosition[2] =
        0; TargetPosition[3] = 0; TargetPosition[4] = 0;
        TargetPosition[5] = 0;}

  if(cursorPos == 19){
    TargetPosition[0] = 5; TargetPosition[1] = 5; TargetPosition[2] =
        5; TargetPosition[3] = 1; TargetPosition[4] = 1;
        TargetPosition[5] = 1;}}

  if(digitalRead(BTN_ENC) && clicked){
    clicked = false;}}

// Calculates necessary slider positions - expects mm and deg
void calculate(double x_i, double y_i, double z_i, double theta_i,
```

```
 double phi_i, double gamma_i) {
// Initialize

double Translate[] = {x_i, y_i, z_i};
double FinalPosition[6][3] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
   0, 0, 0, 0, 0};
// The 3D positions of every socket after transformations
double RM3[6][3] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
   0, 0};
double dx[6] = {0, 0, 0, 0, 0, 0};
double SocketUInnerRef[6][3] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
   0, 0, 0, 0, 0, 0};
double SocketUInnerCross[6][3] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
   0, 0, 0, 0, 0, 0, 0};
double SocketURailDist[6] = {0, 0, 0, 0, 0, 0};
double SocketUInnerDist[6] = {0, 0, 0, 0, 0, 0};
double SocketUProjectionDist[6] = {0, 0, 0, 0, 0, 0};
double Slider[6] = {0, 0, 0, 0, 0, 0};

// Dimensions of the robot
double H = -55;  //Rail height from ground
double W = 25;   //Half Width of Rail
double L = 100;  //Length of Linkages
double SL = 250; //Slider Length

double SA = sin(theta_i*degrad);
double CA = cos(theta_i*degrad);
double SB = sin(phi_i*degrad);
double CB = cos(phi_i*degrad);
double SG = sin(gamma_i*degrad);
double CG = cos(gamma_i*degrad);

double RM11 = CB*CG;
```

```
double RM12 = SA*SB*CG-CA*SG;
double RM13 = CA*SB*CG+SA*SG;
double RM21 = CB*SG;
double RM22 = SA*SB*SG+CA*CG;
double RM23 = CA*SB*SG-SA*CG;
double RM31 = -SB;
double RM32 = SA*CB;
double RM33 = CA*CB;


// Apply net rotation matricies - RM3 is the transformed SocketU;

for(int i = 0; i < 6; i++){
  RM3[i][0] = SocketU[i][0]*RM11 + SocketU[i][1]*RM12 +
     SocketU[i][2]*RM13;
  RM3[i][1] = SocketU[i][0]*RM21 + SocketU[i][1]*RM22 +
     SocketU[i][2]*RM23;
  RM3[i][2] = SocketU[i][0]*RM31 + SocketU[i][1]*RM32 +
     SocketU[i][2]*RM33;}

for(int i = 0; i < 6; i ++){
// Add rotated plate to base position
  FinalPosition[i][0] = Translate[0] + RM3[i][0];
  FinalPosition[i][1] = Translate[1] + RM3[i][1];
  FinalPosition[i][2] = Translate[2] + RM3[i][2];}


for(int i = 0; i < 6; i ++){

  for(int j = 0; j < 3; j ++){
    SocketUInnerRef[i][j] = FinalPosition[i][j] - InnerRef[i][j];}

  SocketUInnerCross[i][0] = RailRefVect[i][1]*SocketUInnerRef[i][2]
```

```
    - RailRefVect[i][2]*SocketUInnerRef[i][1];

  SocketUInnerCross[i][1] = RailRefVect[i][2]*SocketUInnerRef[i][0]
    - RailRefVect[i][0]*SocketUInnerRef[i][2];

  SocketUInnerCross[i][2] = RailRefVect[i][0]*SocketUInnerRef[i][1]
    - RailRefVect[i][1]*SocketUInnerRef[i][0];

  SocketURailDist[i] =
    sqrt(SocketUInnerCross[i][0]*SocketUInnerCross[i][0]
  +SocketUInnerCross[i][1]*SocketUInnerCross[i][1]
  +SocketUInnerCross[i][2]*SocketUInnerCross[i][2])
  /SL;

  dx[i] = sqrt(L*L - SocketURailDist[i]*SocketURailDist[i]);

  SocketUInnerDist[i] =
    sqrt(SocketUInnerRef[i][0]*SocketUInnerRef[i][0]
  +SocketUInnerRef[i][1]*SocketUInnerRef[i][1]
  +SocketUInnerRef[i][2]*SocketUInnerRef[i][2]);

  SocketUProjectionDist[i] =
    sqrt(SocketUInnerDist[i]*SocketUInnerDist[i] -
    SocketURailDist[i]*SocketURailDist[i]);

  Slider[i] = dx[i] + SocketUProjectionDist[i];
  Serial.println(Slider[i]);

  P[i] = Slider[i];}}

void ActivateStepperMotors(){
// Motor and thread specs
double pitch = 1.25;
```

```
int stepsPerRot = 215;
int motorSpeed = 4; // = 300/RPM, must be int > 0. Max 4, 75RPM

double SldDist[6];
double SldMaxDist = 0;
int stepsNeeded;
float stepsPerMotor[6];
int stepDir[6];
double stepProgress = 0;
double SliderPositions[6];

// Get slider displacements and the maximum
for(int i = 0; i < 6; i++){
  SldDist[i] = abs(CurrentSldPos[i] - NewSldPos[i]);

  //Detects rotation direction

  if(CurrentSldPos[i]-NewSldPos[i] >= 0){
    stepDir[i] = 0;}
  else{
    stepDir[i] = 1;}

  //How many steps each motor must make in the next operation
  stepsPerMotor[i] = stepsPerRot*SldDist[i]/pitch;

  //Largest of those
  if(abs(SldDist[i]) > abs(SldMaxDist)) {
    SldMaxDist = SldDist[i];}}

//Number of steps needed
stepsNeeded = stepsPerRot*SldMaxDist/pitch;

// Array for holding step timings
```

```
float stepIndex[6][stepsNeeded];
for(int i = 0; i < 6; i++){
  for(int j = 0; j < stepsNeeded; j++){
    stepIndex[i][j] = 0;}}

// Spreads each step sequence evenly over the max steps needed
for(int m = 0; m < 6; m++){
  for(int s = 0; s < stepsPerMotor[m]; s++){
    //Catch overflow
    double skipSize = stepsNeeded/stepsPerMotor[m];
    if(round((s+1)*skipSize) > stepsNeeded){
      stepIndex[m][stepsNeeded] = 1;}
    else{
      stepIndex[m][round((s)*skipSize)] = 1;}}}


int stepCountTest = 0;
for(int s = 0; s < stepsNeeded; s++){
  stepCountTest++;
  digitalWrite(M1_step, stepIndex[1][s]);
  digitalWrite(M1_dir, stepDir[1]);
  digitalWrite(M2_step, stepIndex[2][s]);
  digitalWrite(M2_dir, stepDir[2]);
  digitalWrite(M3_step, stepIndex[3][s]);
  digitalWrite(M3_dir, stepDir[3]);
  digitalWrite(M4_step, stepIndex[4][s]);
  digitalWrite(M4_dir, stepDir[4]);
  digitalWrite(M5_step, stepIndex[5][s]);
  digitalWrite(M5_dir, stepDir[5]);
  digitalWrite(M6_step, stepIndex[0][s]);
  digitalWrite(M6_dir, stepDir[0]);
  delay(motorSpeed);
  digitalWrite(M1_step, LOW);
```

```
  digitalWrite(M2_step, LOW);
  digitalWrite(M3_step, LOW);
  digitalWrite(M4_step, LOW);
  digitalWrite(M5_step, LOW);
  digitalWrite(M6_step, LOW);}


// Define new current pos
calculate(NewPos[0],NewPos[1],NewPos[2],NewPos[3],NewPos[4],NewPos[5]);
for(int i = 0; i < 6; i++){
  CurrentSldPos[i] = P[i];
  CurrentPosition[i] = NewPos[i];}

// Display info
lcd.setCursor(0,1);
lcd.print("            ");
lcd.setCursor(0,1);
lcd.print(round(CurrentPosition[0]));
lcd.print(" ");
lcd.print(round(CurrentPosition[1]));
lcd.print(" ");
lcd.print(round(CurrentPosition[2]));
lcd.print(" ");
lcd.print(round(CurrentPosition[3]));
lcd.print(" ");
lcd.print(round(CurrentPosition[4]));
lcd.print(" ");
lcd.print(round(CurrentPosition[5]));}
```

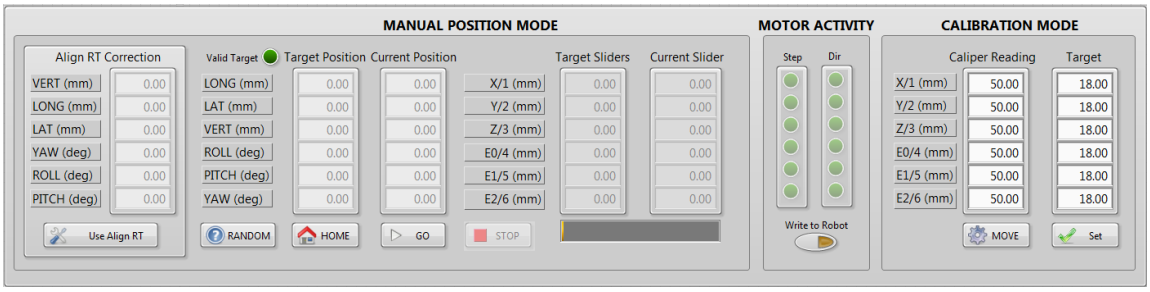


Figure D.1: User interface of LabVIEW positioning system

D.3 LabVIEW

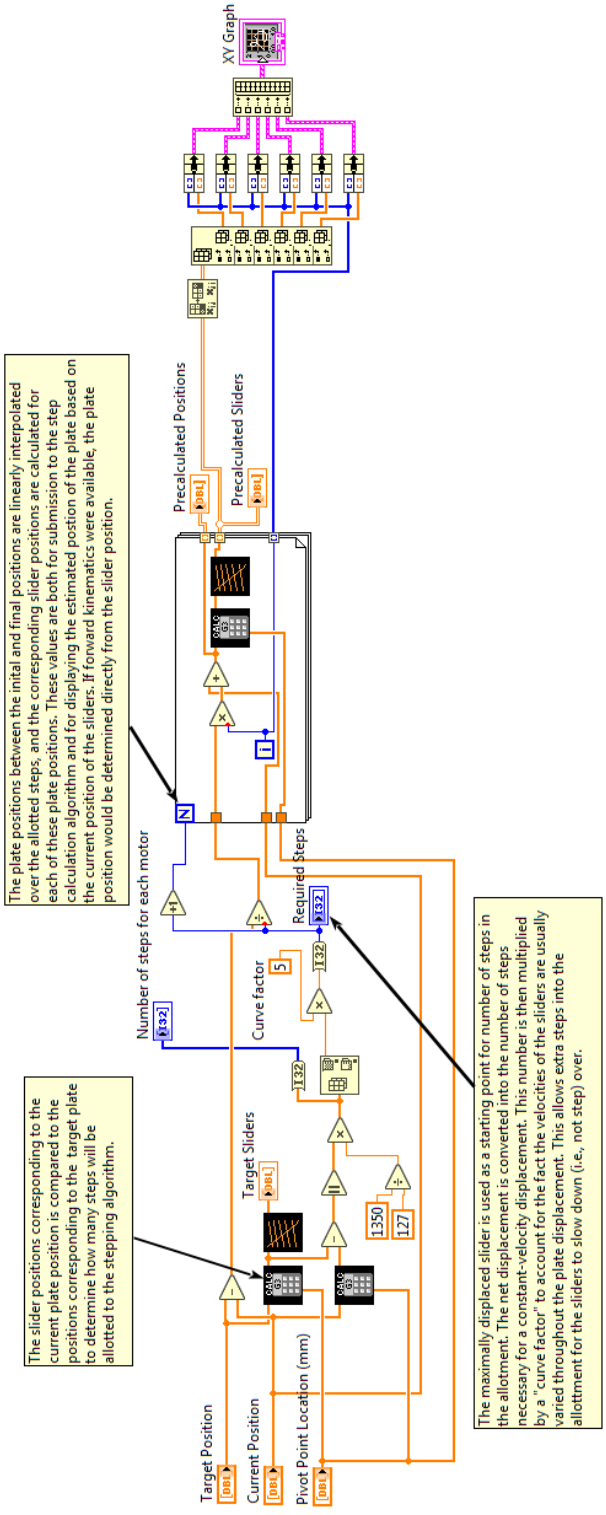


Figure D.2: Block diagram of slider precalculation

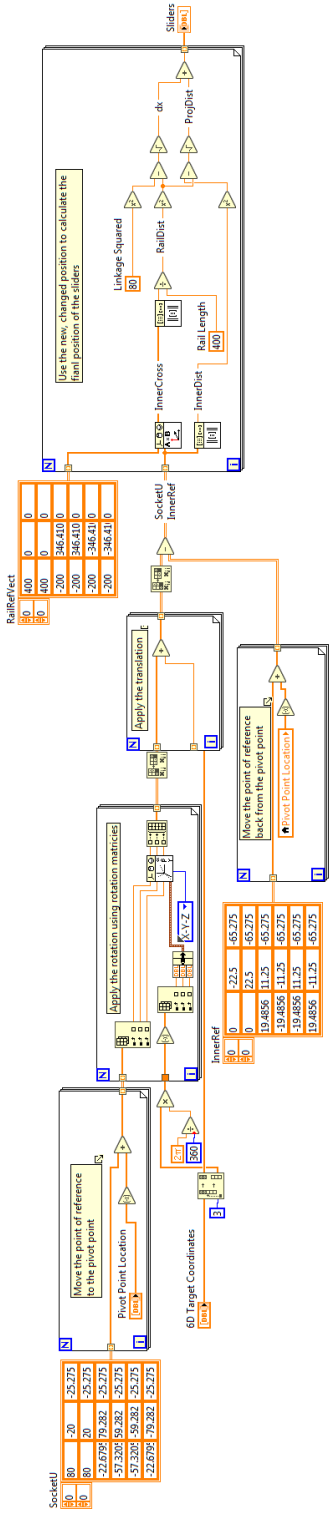


Figure D.3: Block diagram of slider position calculation

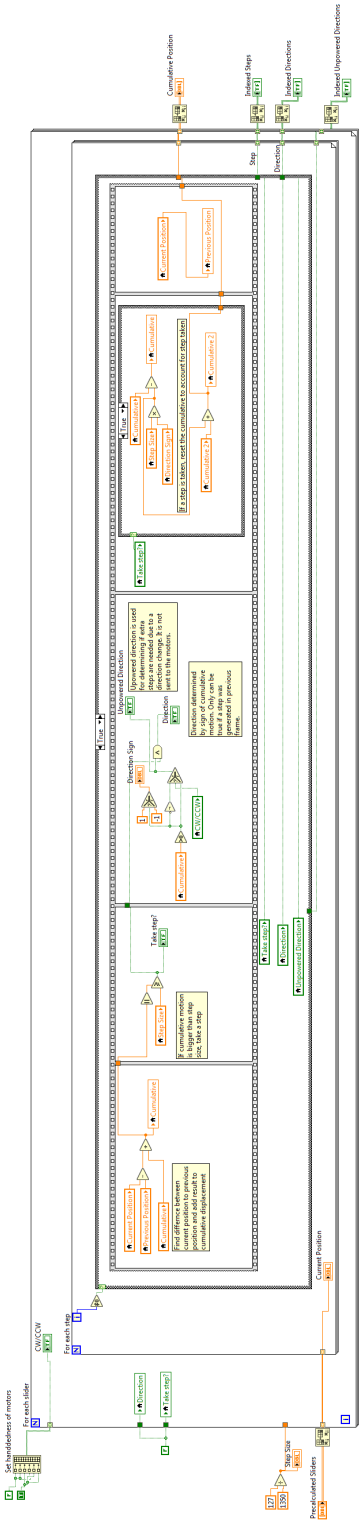


Figure D.4: Block diagram of stepping algorithm. For the major false case, the cumulative motion is set to zero, the previous and secondary cumulative are set to the current position. This ensures that the first step is always off. For the minor false case in frame 4, no step is generated, and the value for cumulative 2 is reported for the cumulative position.

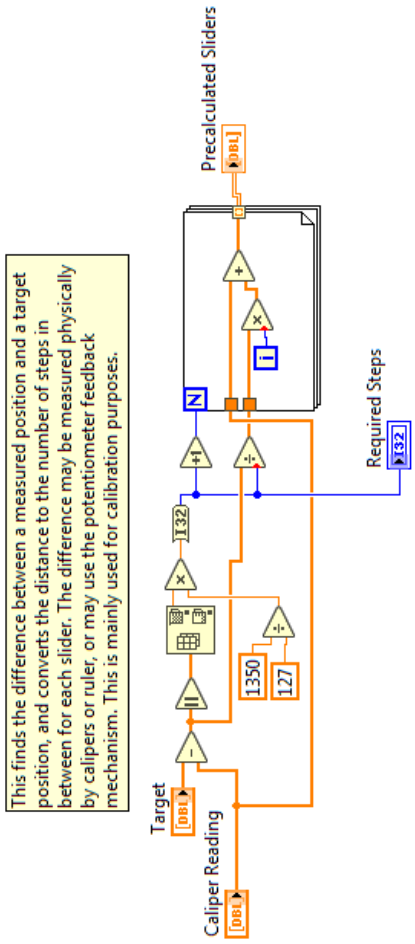


Figure D.5: Block diagram of simple slider translation program

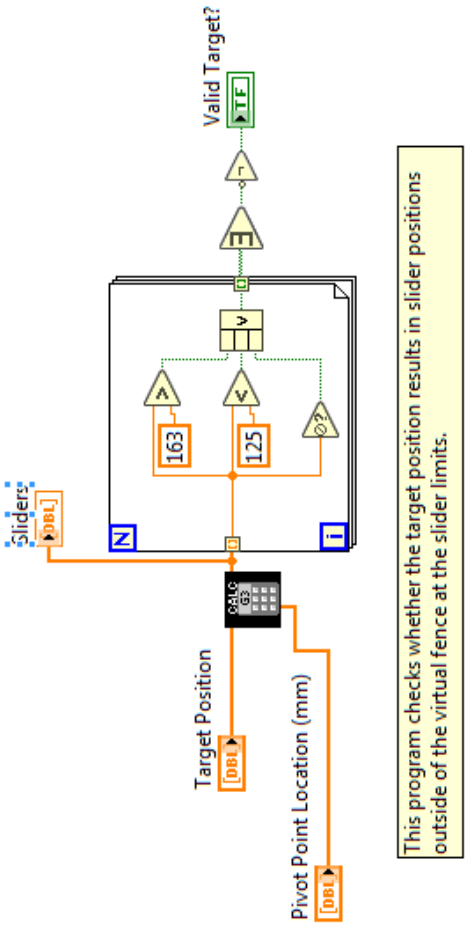


Figure D.6: Block diagram of target validity check

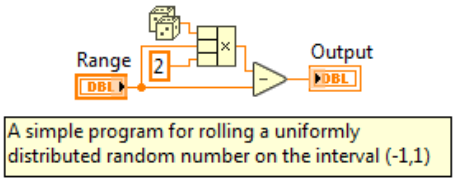


Figure D.7: Block diagram of symmetric uniform random selection

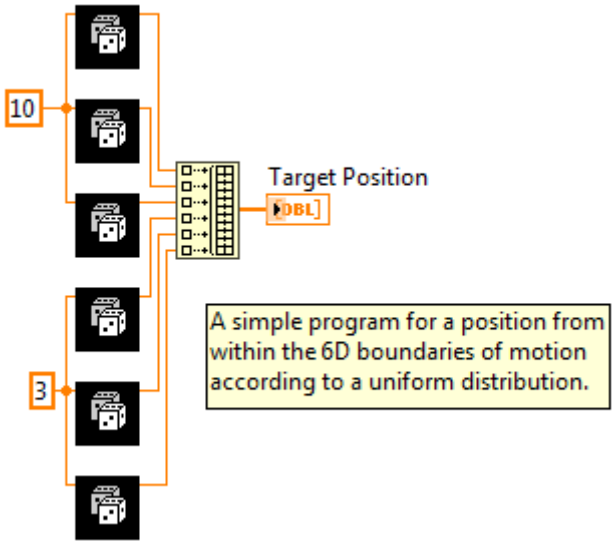


Figure D.8: Block diagram of random target selection

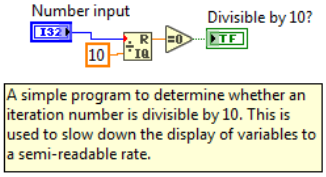


Figure D.9: Block diagram of display speed reduction

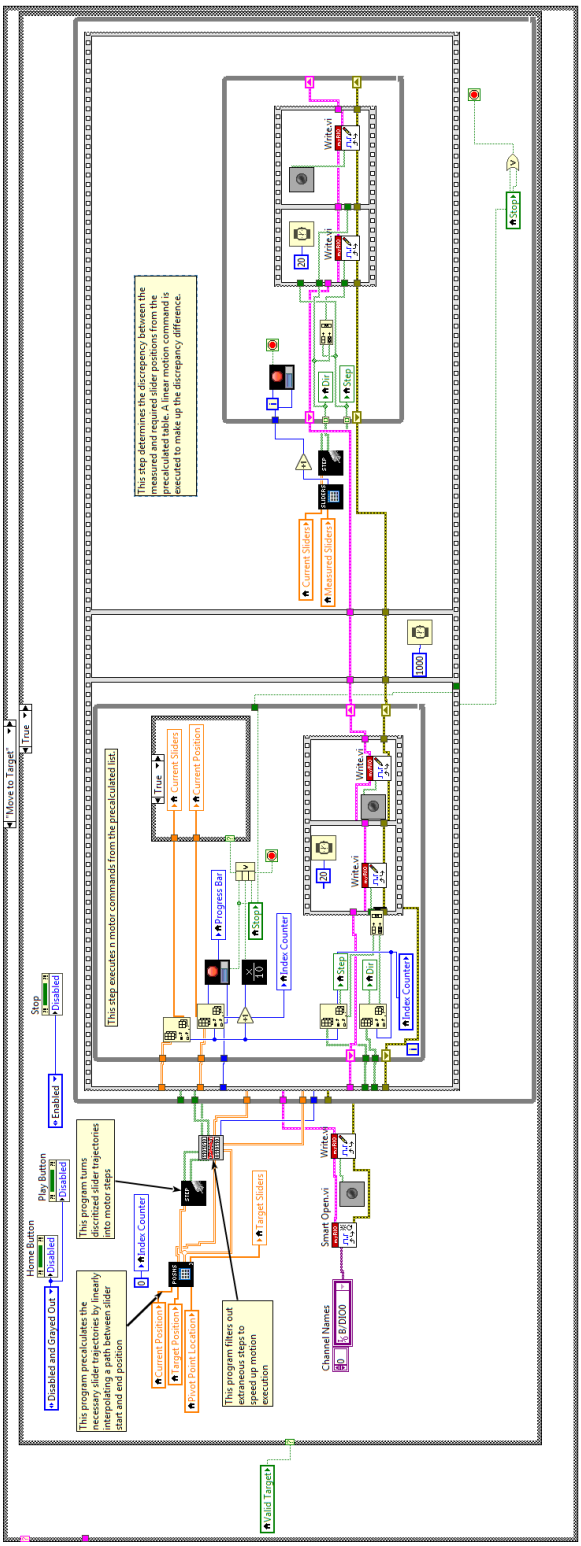


Figure D.10: Block diagram of target calculation and execution program

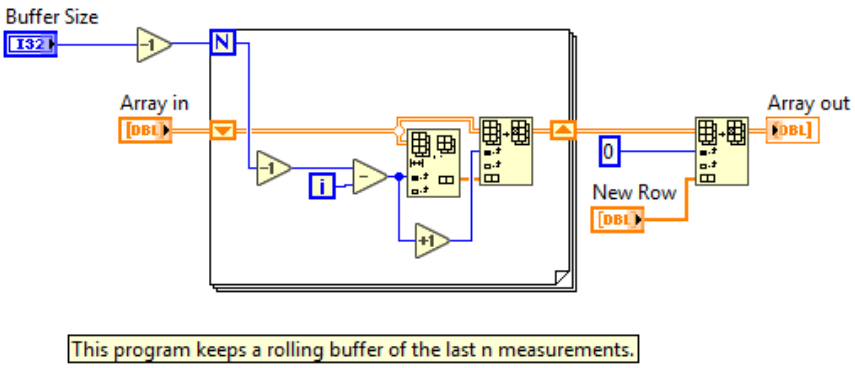


Figure D.11: Block diagram of buffering program.

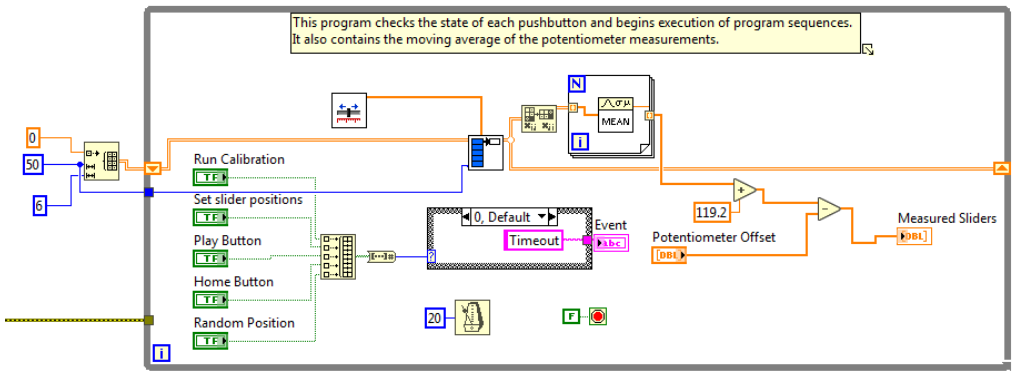


Figure D.12: Block diagram of execution software.

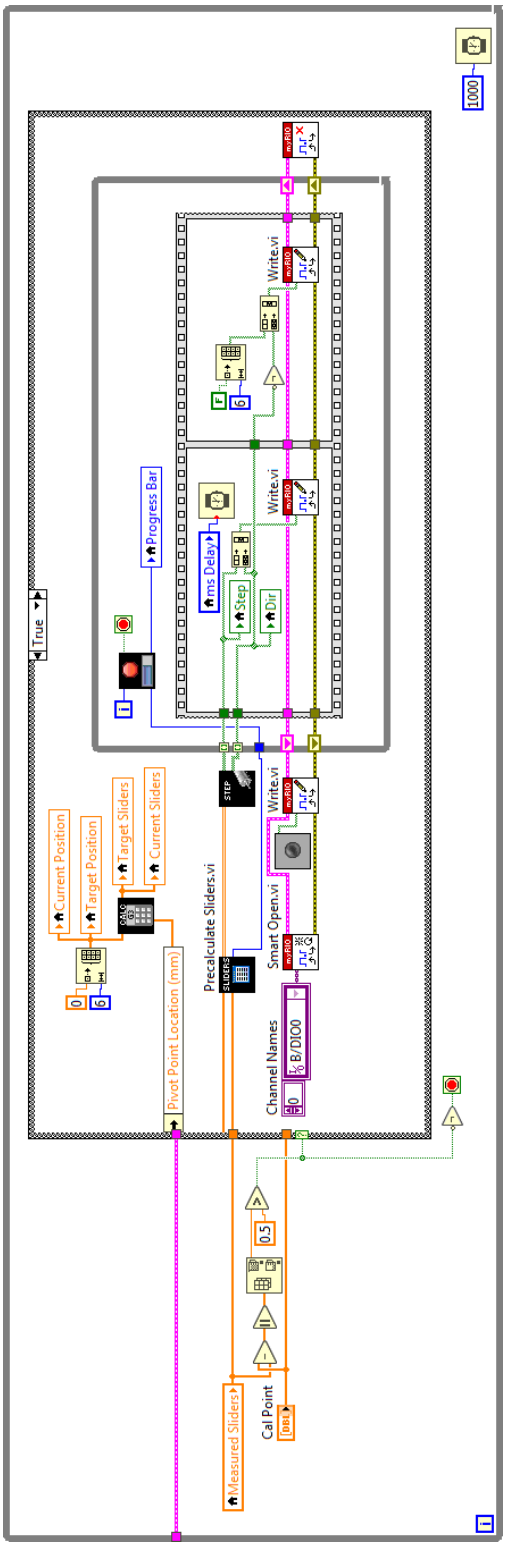


Figure D.13: Block diagram of calibration program.

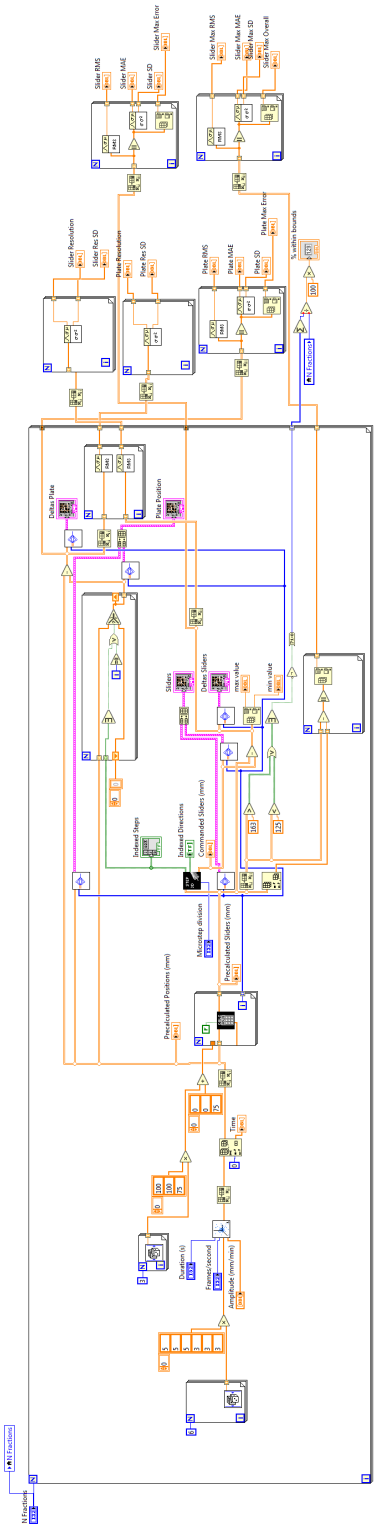


Figure D.14: Block diagram of non-realtime motion correction simulation statistical analysis.

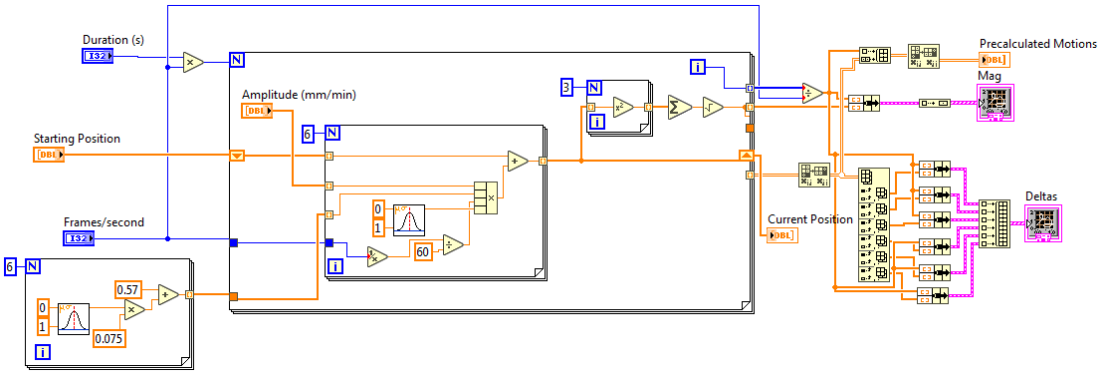


Figure D.15: Block diagram of LabVIEW non-realtime simulated patient.

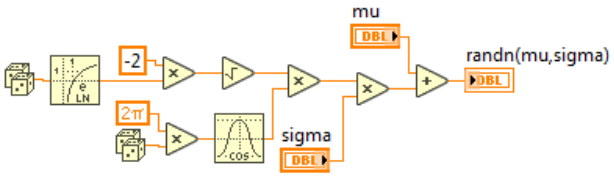


Figure D.16: Block diagram of Box-Muller transform used to obtain normally-distributed random numbers.

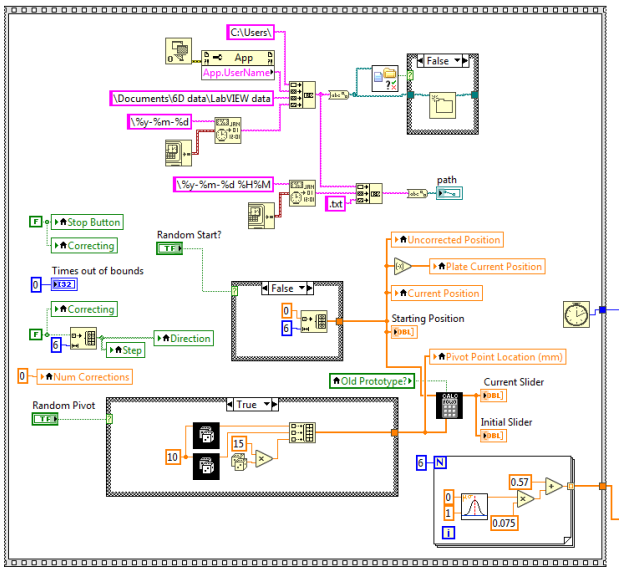


Figure D.17: Block diagram of initialization of the real-time motion monitoring program.

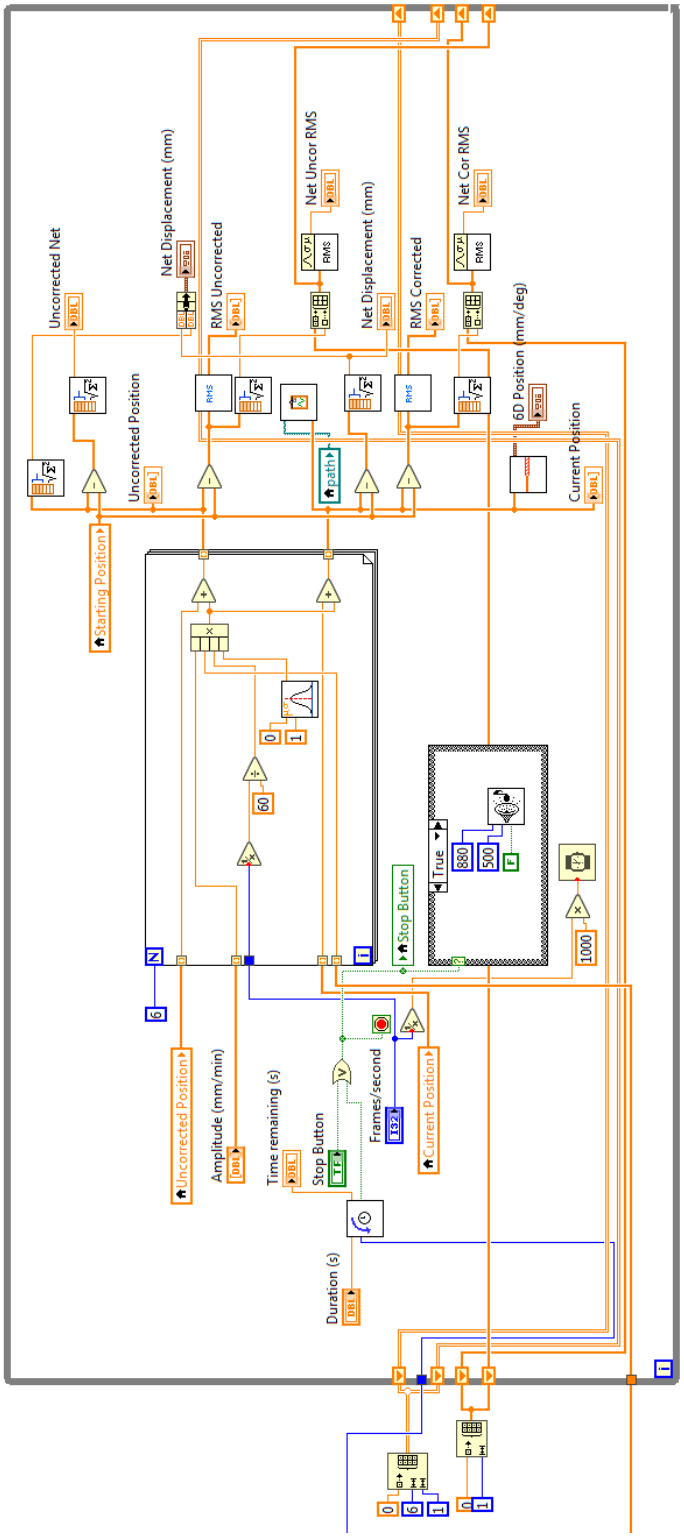


Figure D.18: Block diagram of the simulated real-time patient.

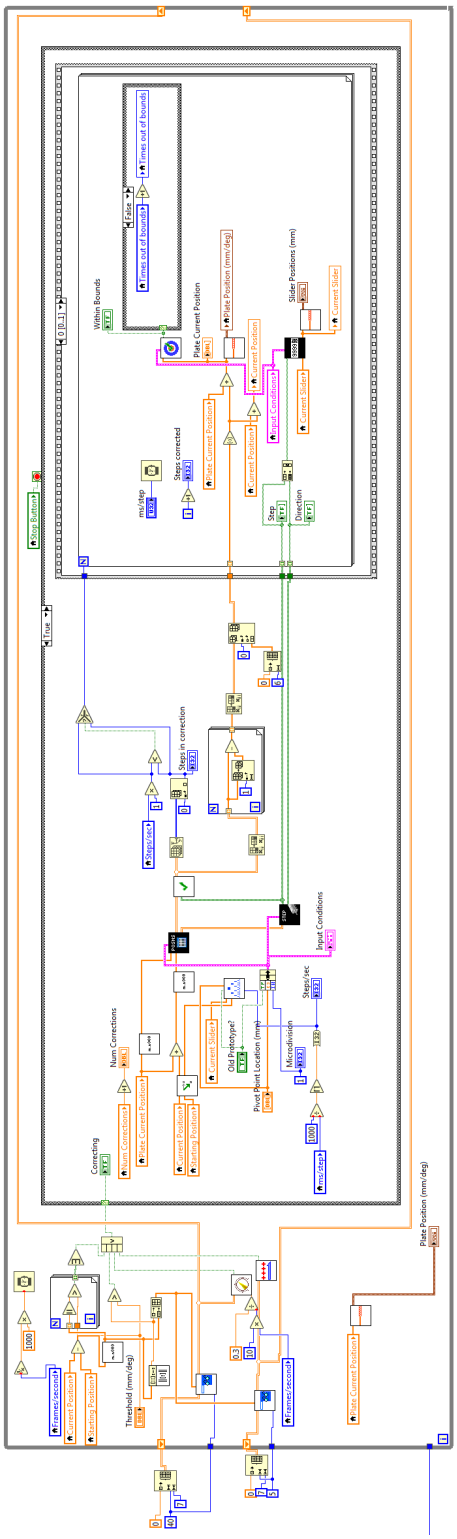


Figure D.19: Block diagram of monitoring program for the real-time motion monitoring program.

Vita

Mark Raymond Ostyn was born on November 28, 1988, in Nampa, Idaho, and is still an American citizen. He graduated from Nampa High School in 2007, and then attended Northwest Nazarene University in Nampa to receive a Bachelor of Arts in Physics in 2011, and then also a Bachelor of Science in Engineering Physics in 2013. He then attened Indiana University in Bloomington and received a Master of Science degree in Medical Physics 2014. Before attending Virginia Commonwealth University, Mark worked for ProNova Medical Solutions as a programmer. At VCU, Mark has published "A Simulation Study of Radiofrequency Localization System for Tracking Patient Motion in Radiotherapy" in *Sensors* and "A Novel Radiotherapy-friendly, Mechanical Patient Positioning System for Head and Neck Radiotherapy" in *Physics in Medicine and Biology*. Mark has also presented his research at the *Virginia Academy of Sciences* in 2016, winning an honorable mention in oral presentations, at the 2017 annual meeting of the American Association of Physicists in Medicine, and at the 2017 mid-Atlantic chapter meeting of the AAPM, winning second place in the Young Investigator's Symposium. Most recently, Mark has been selected to represent the mid-Atlantic chapter of the AAPM at the 2018 annual meeting of the AAPM in the first *Physics Slam* event, after winning the chapter-wide competition.