Control Interface for Autonomous Robotic Brain Surgery Using Magnetically Stimulated Particles

Team Name: Team Nson

Team Members:
Elizabeth Ankrah, Bassam Mutawak, Minh Quan Do, Victor Huynh

Weinberg Medical Physics Sponsors:
Dr. Chad Ropp, Olivia Hale, Dr. Irving Weinberg Dr. Lamar Mair, Brad English

Faculty Advisors:
Dr. Nathalia Peixoto, Dr. Qi Wei

Date:
15 November 2018

Team Contact Information

Team Member	Email	Phone
Elizabeth Ankrah	eankrah@masonlive.gmu.edu	240-751-2793
Bassam Mutawak	bmutawak@masonlive.gmu.edu	703-889-0033
Victor Huynh	vhuynh8@masonlive.gmu.edu	703-887-0105
Minh Quan Do	mdo9@masonlive.gmu.edu	703-589-5392

1. Executive Summary:

This document provides the preliminary design review (PDR) for the Fall 2018 - Spring 2019 senior design project (Project Magneto) in collaboration with George Mason University (GMU) and Weinberg Medical Physics LLC (WMP). The highlight of the PDR is the evaluation of alternate design approaches to fulfill the project's technical requirements and the objective justification for the selected approach. In addition, this document contains several other areas of administrative and technical information.

Our project is the design and implementation of a software control interface for operating an electropermanent magnetic control system (MCS). This MCS is constructed by WMP to transport and activate magnetic particles loaded with therapeutic payloads inside the human body. Our project will operate the MCS to autonomously deliver these particles along a user-defined path.

The control interface will consist of 4 overarching modules coded in C++:

1) **Graphic User Interface (GUI)**
2) **Image Segmentation Module**
3) **Control Module**
4) **Physics Module**

Technical aspects for this project include, but are not limited to, a thorough understanding of magnetic fields and gradients, therapy carriers, MCS operational specifications, GUI design and functionality, software engineering, and C++ programming.

Document Contents

2. Problem Definition:

Despite recent advancements in research, major limitations are holding back clinical implementation of nose-to-brain magnetic particle delivery. These include physiological obstructions to particle transport, magnetic field strength attenuation as particles travel to deeper targets, and impracticality of manually operating magnet arrays in real-time. Weinberg Medical Physics is constructing an electropermanent magnetic control system to address many of these issues. To supplement their device's operation, we will design and implement a control interface allowing for the system to autonomously deliver magnetic particles from the nose to user-defined locations in the deep brain. Depending on the type of particle, various therapies such as drugs, genes, or heat could be delivered to these locations.

A. Objectives

1. Design software control modules for MCS operation
2. Design an interactive GUI for surgeon usage

B. Requirements

Control Modules (non-GUI)

- Must track and segment magnetic resonance (MR) images to determine particle locations
- Must guide particles along user-defined path via visual feedback
- Must be able to perform particle manipulations including: Translation, Rotation, Aggregation, Dispersion, and Activation
- Must include safety measures between every module-to-module interaction
- Must be coded in C++

GUI

- Must be able to receive, display, and process a three-dimensional (3D) path input from the user
- Must discretize user-specified path into a 3D point array
- Must be able to collect batches of MR images and display them intuitively on the GUI
- Must be coded in C++ via QT integrated development environment (The Qt Company, Espoo, Finland)
- Must include control panel with start/stop button(s) to begin and halt operations, settings button, status update of the operation, help button, etc.

3. Potential Application and Customers

The primary customer of Project Magneto is our sponsor, Weinberg Medical Physics. Our software system will support the company's research on magnetic particle delivery and activation through MRI image guidance. Upon further success in their testing, WMP plans to market their complete system (consisting of ultra-fast MRI machine, electropermanent magnetic control system, and software control interface) to potential clinical investors and customers. Below is a Q&A with WMP to express the significance of their research and the role of our project.

Interview Q&A with Weinberg Medical Physics, LLC.

1. What is WMP's driving purpose for their research and development of an electropermanent magnetic control system? What clinical applications do WMP foresee that utilize the finalized system, and what are the biggest implementation challenges that WMP will tackle?

   The electropermanent system is able to provide image guidance with MRI and also preserves many degrees of freedom in transporting and activating magnetic particles. Eventual clinical applications will include cancer treatment, otolaryngology, and neurological and psychiatric disease.

2. What is the significance of the senior design project (Control Interface for Autonomous Robotic Brain Surgery using Magnetically Stimulated Particles) for WMP's system?

   This will be a first step in developing a treatment planning system.

3. What sort of research/testing will WMP be performing with the finalized system (our software integrated with WMP's electropermanent hardware)?

   We will build upon it by adding features and bringing it to FDA-mandated quality levels.

4. What is WMP's vision beyond their research with the finalized system? Will the system be advocated or marketed towards potential customers, and if so, who are the customers in mind?

   We have a global market in mind. Based in our initial market research, we expect more than 1,000 users per year for the eventual therapeutic product.

5. Beyond the specified technical requirements (e.g. programming language, UI functions), what are WMP's measures of success for our senior design project? What criteria will WMP use to evaluate the project?

   It needs to work in guiding particles to a predetermined target. The software must be documented well enough so that somebody who is not familiar with the project can pick it up.

4. Design Alternatives

A. Engineering Characteristics

The following engineering characteristics are guiding principles for our project design. All proposed alternatives meet these engineering principles. The justification for the selected architecture will be discussed under the later section, Rationale for Selected Approach.

Engineering Characteristics	Description
Structural Division of Function	Software architecture is compartmentalized into functional components with distinct areas of task execution. No module's functions overlap with any others.
Safe System Operation	Safety checkpoints with sensitive error detection are placed after each module's operation within the software architecture. Effective error mitigation actions are provided to the user.
Modular Hardware Connection	Software system operation is optimized for multiple hardware components. This allows for seamless transition from one hardware system to another.
Platform Independent	Software can be operated on multiple operating systems for versatility. Functionality will be the same on all compatible operating systems.

B. Proposed Architecture Designs

Before discussing the proposed architecture designs, it is important to note that following engineering characteristics are fulfilled regardless of the architecture.

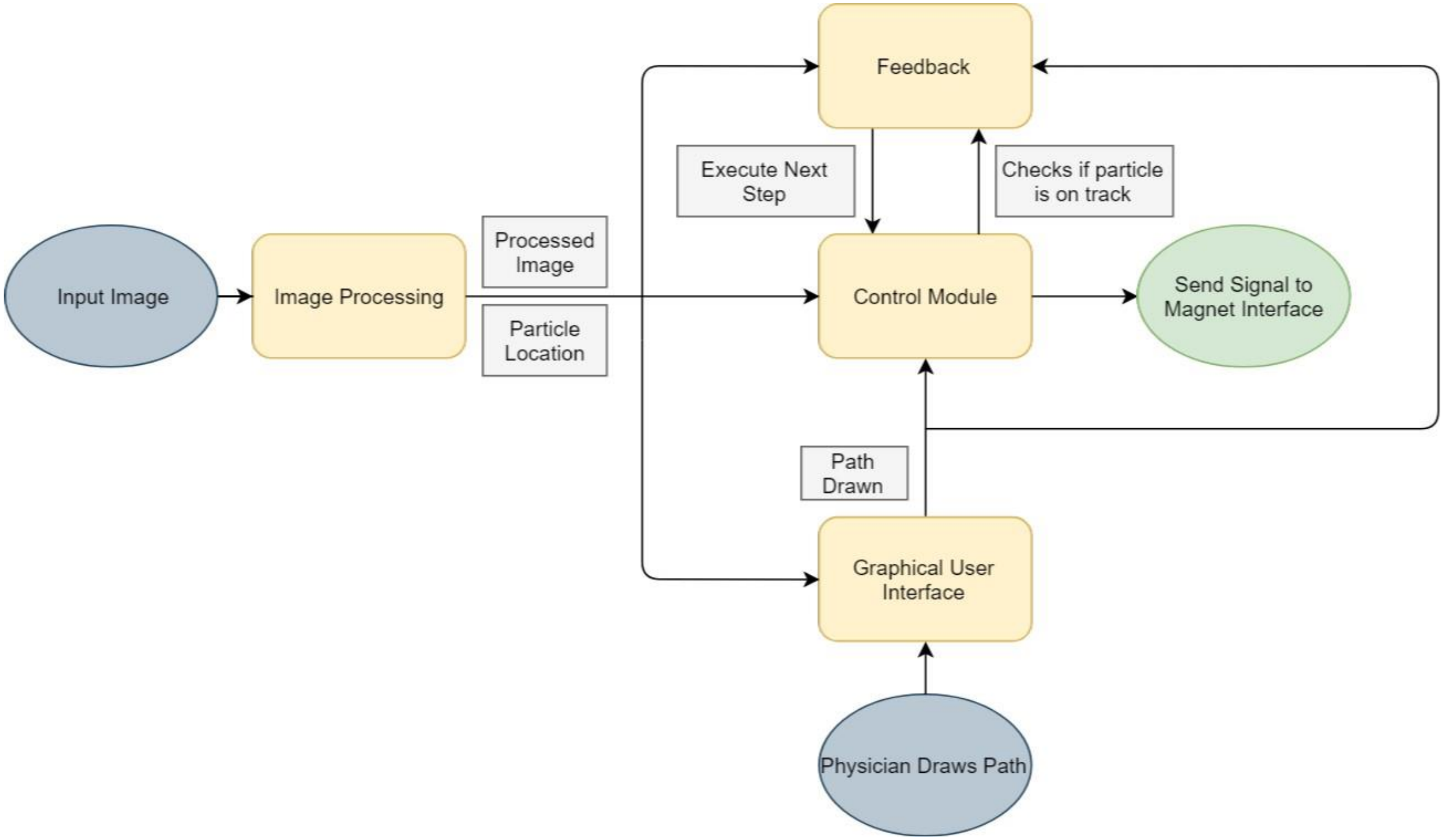
Fulfillment of "Modular Hardware Connection"

The programming language established for our project (C++) allows for access to many supporting libraries for robotics programming and operating systems development. This allows us to integrate our software with various hardware from WMP.

Fulfillment of "Platform Independent"

C++ is a programming language with the characteristics of both high- and low-level languages that has proven to be popular for operating systems development. Using C++ allows for development of software that can easily be made compatible with many different operating systems.

Approach #1: Control-Centric Design



Approach #1: Control-Centric Design

Overview:

This design is termed "control-centric" due to the multiple major responsibilities fulfilled by the control module (e.g. calculating magnetic physical parameters and outputting commands to hardware). The control module acts as the sole connection between the software and the magnetic hardware system.

General System Operation:

1. An input image array is segmented by the **Image Processing** module to obtain particle(s) location and create a 3D representation of their current location. The locations are then sent to the **Graphical User Interface (GUI)** for the user to see. At a separate time, the physician draws a 3D path on the GUI. The GUI discretizes this second input and sends it to the next module.

2. Prior to start of operation, the **Control** module waits to receive the discretized path and the current particle(s) location. The module computes the required magnetic field gradient and current to move the particle(s) along the user-defined path. This information is sent to the hardware microcontroller for execution.

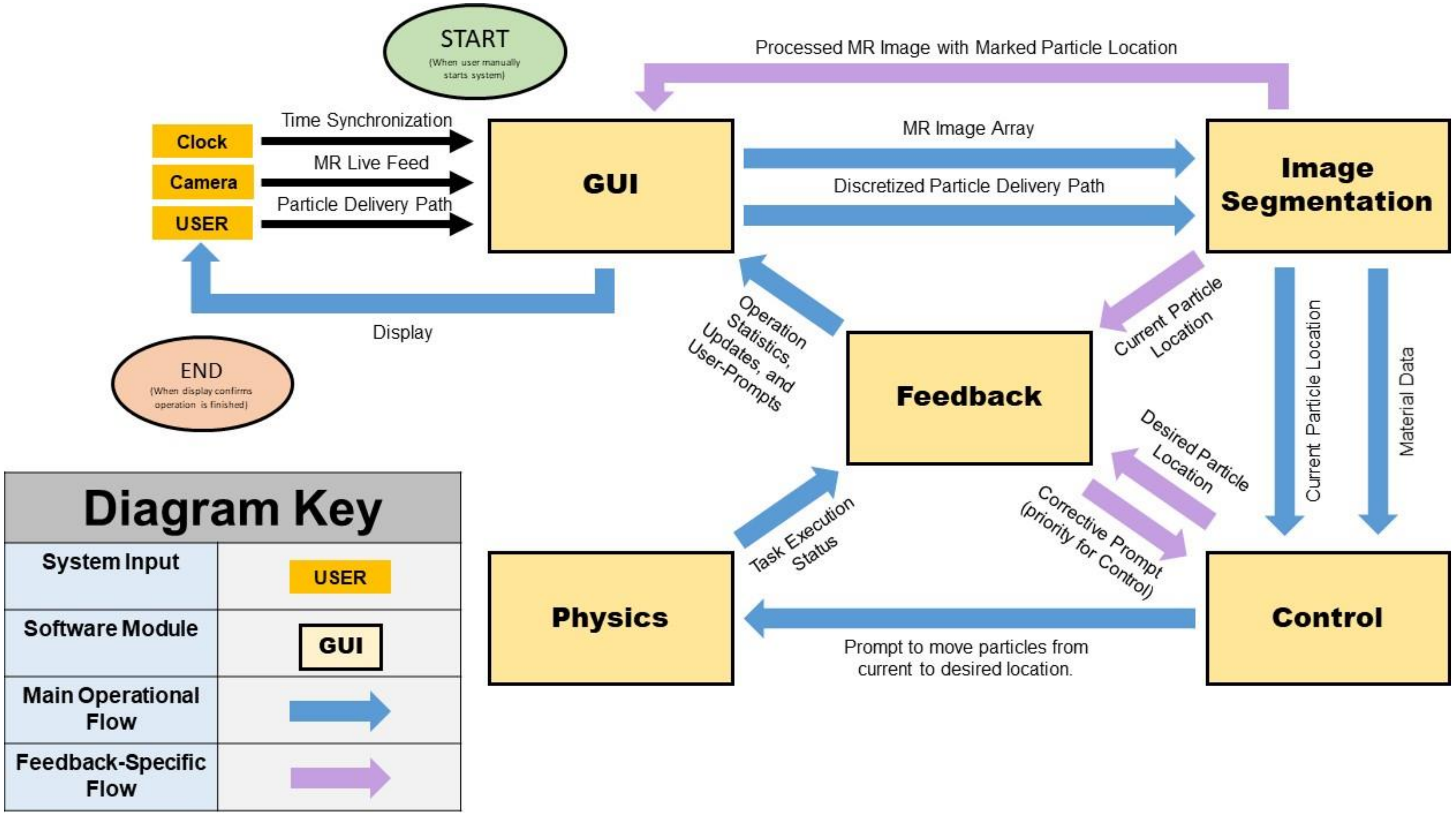
3. The **Feedback** module ensures that the particle(s) are on the user-defined path based on information send to it from the Control module. If the particle(s) deviate from the path, the module computes the necessary microcontroller command to return particle(s) on track. The computed command is sent back to the control module for execution. At the end of operation when the particles reach the target destination, the Control module confirms that system operation is complete.

Fulfillment of Engineering Characteristics:

Structural Division of Function	Design utilizes 4 modules (GUI, Image Processing, Control, and Physics) with distinct tasks.
Safe System Operation	As the Control module is the sole software-hardware link, any commands to terminate execution or software errors can be caught by the module before reaching the hardware.
Modular Hardware Connection	*See prior paragraphs under Proposed Architecture Design.
Platform Independent	*See prior paragraphs under Proposed Architecture Design.

Feedback from Sponsor:
Flow of information and timing of task execution for the modules is very ambiguous. System could work but would require a significant amount of multithreading and concurrent processes that would make development unnecessarily more complex and challenging.

Approach #2: Feedback-Centric Design


START
(When user manually starts system)
Clock
Camera
USER
Time Synchronization
MR Live Feed
Particle Delivery Path
GUI
Processed MR Image with Marked Particle Location
MR Image Array
Discretized Particle Delivery Path
Image Segmentation
Display
END
(When display confirms operation is finished)
Operation Statistics, Updates, and User-Prompts
Current Particle Location
Feedback
Current Particle Location
Material Data
Desired Particle Location
Corrective Prompt (priority for Control)
Task Execution Status
Physics
Control
Prompt to move particles from current to desired location.
Diagram Key
System Input
USER
Software Module
GUI
Main Operational Flow
Feedback-Specific Flow

Approach #2: Feedback-Centric Design

Overview:

This design is termed “feedback-centric” due to multiple interactions and high dependence of the other modules with the Feedback module. The Feedback module is at the “center” of system operation and provides safety checks for all modules.

General System Operation:

1. Prior to operation, the **Graphical User Interface (GUI)** discretizes a 3D path input from the user. The path is sent to the **Image Segmentation** module along with an MR image array. The images are segmented to obtain particle(s) location and material data. Then at this point, a composite image is created with marked particle(s) location to be relayed back to the GUI for the user visualization.

2. The **Control** module receives the current particle(s) location and material data. It sends the **Feedback** module the desired particle location (expected result of current translation). Control then sends a prompt to **Physics** module to move the particle(s) to the desired location. After Physics executes the translation, the execution status is sent to Feedback.

3. In the next iteration, Feedback obtains new current particle(s) location from Image Segmentation and compares it to the desired particle location that was previously obtained. If the locations do not match within reason, a special corrective prompt is sent to Control (takes priority) to pass onto Physics. Regardless of corrective prompt, statistics and other updates are relayed to the GUI for user discretion.

4. When Control determines that the end of the path has been reached, an update is sent to Feedback, which stops system operation. A final notification of operation completion is relayed back to the GUI.

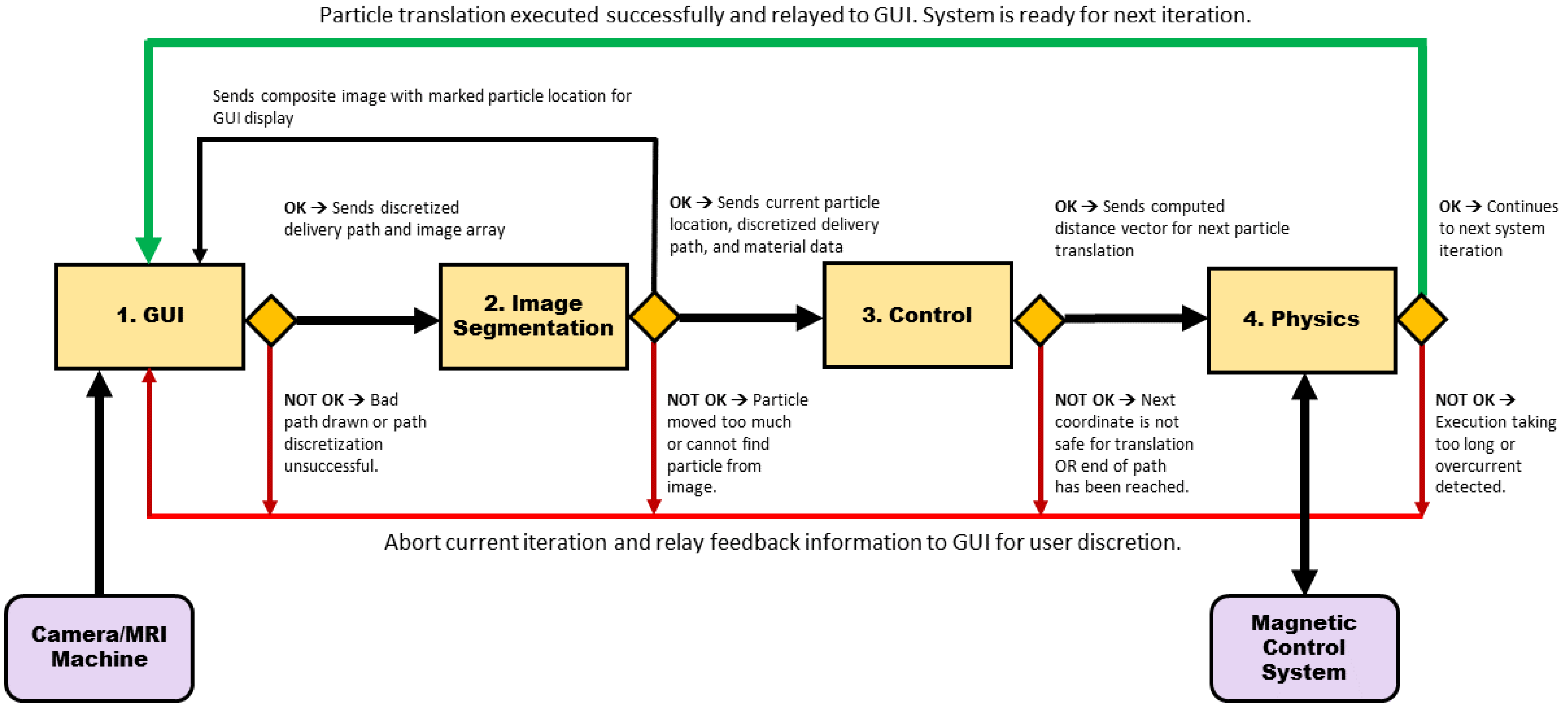
Fulfillment of Engineering Characteristics:

Structural Division of Function	Design utilizes 5 modules (GUI, Image Processing, Control, and Feedback, and Physics) with distinct tasks.
Safe System Operation	The Feedback module functions as a safety check by detecting anomalies in particle translation and providing corrective commands for output to the hardware. It also notifies the user of detected errors and allows for manual intervention.
Modular Hardware Connection	*See prior paragraphs under Proposed Architecture Design.
Platform Independent	*See prior paragraphs under Proposed Architecture Design.

Feedback from Sponsor

Feedback module is overloaded with too many tasks; very dangerous for operation. Timing of module actions and information flow is ambiguous, especially with regards to corrective prompts from Feedback module. Also, system structure is too complex.

Approach #3: Linear Design with Regular Safety Checkpoints


Particle translation executed successfully and relayed to GUI. System is ready for next iteration.
Sends composite image with marked particle location for GUI display
OK → Sends discretized delivery path and image array
OK → Sends current particle location, discretized delivery path, and material data
OK → Sends computed distance vector for next particle translation
OK → Continues to next system iteration
1. GUI
2. Image Segmentation
3. Control
4. Physics
NOT OK → Bad path drawn or path discretization unsuccessful.
NOT OK → Particle moved too much or cannot find particle from image.
NOT OK → Next coordinate is not safe for translation OR end of path has been reached.
NOT OK → Execution taking too long or overcurrent detected.
Abort current iteration and relay feedback information to GUI for user discretion.
Camera/MRI Machine
Magnetic Control System

Approach #3: Linear Design with Regular Safety Checkpoints

Overview:

This design operates in a linear fashion (modules execute in sequential order). A module cannot interact with the next module unless the attached safety checkpoint (one per module) is cleared. If the checkpoint is not cleared, the system breaks execution and alerts the user. The sequential design simplifies the information flow and the timing of execution for all modules.

General System Operation:

1. Prior to operation start, the **Graphical User Interface (GUI)** is fed an MR image array from the MRI machine. The user draws a 3D path on the GUI using the image array, which is then discretized into a set of coordinates. The drawn path is then processed through a safety checkpoint that evaluates the validity of the discretized path.

2. At operation start, the **Image Segmentation** module receives an MR image array and the discretized path from the GUI. The images are segmented to obtain the particle(s) location and determine the material data (i.e. bone or tissue). The particle location is checked by the safety checkpoint to confirm it is successfully located. A composite image with marked particle(s) location is sent back to the GUI for user visualization.

3. The **Control** module receives the particle(s) location, discretized path, and material data from the Image Segmentation module and uses them to compute a distance vector for particle translation. Before the particle is moved, the safety checkpoint confirms that the translation is safe.

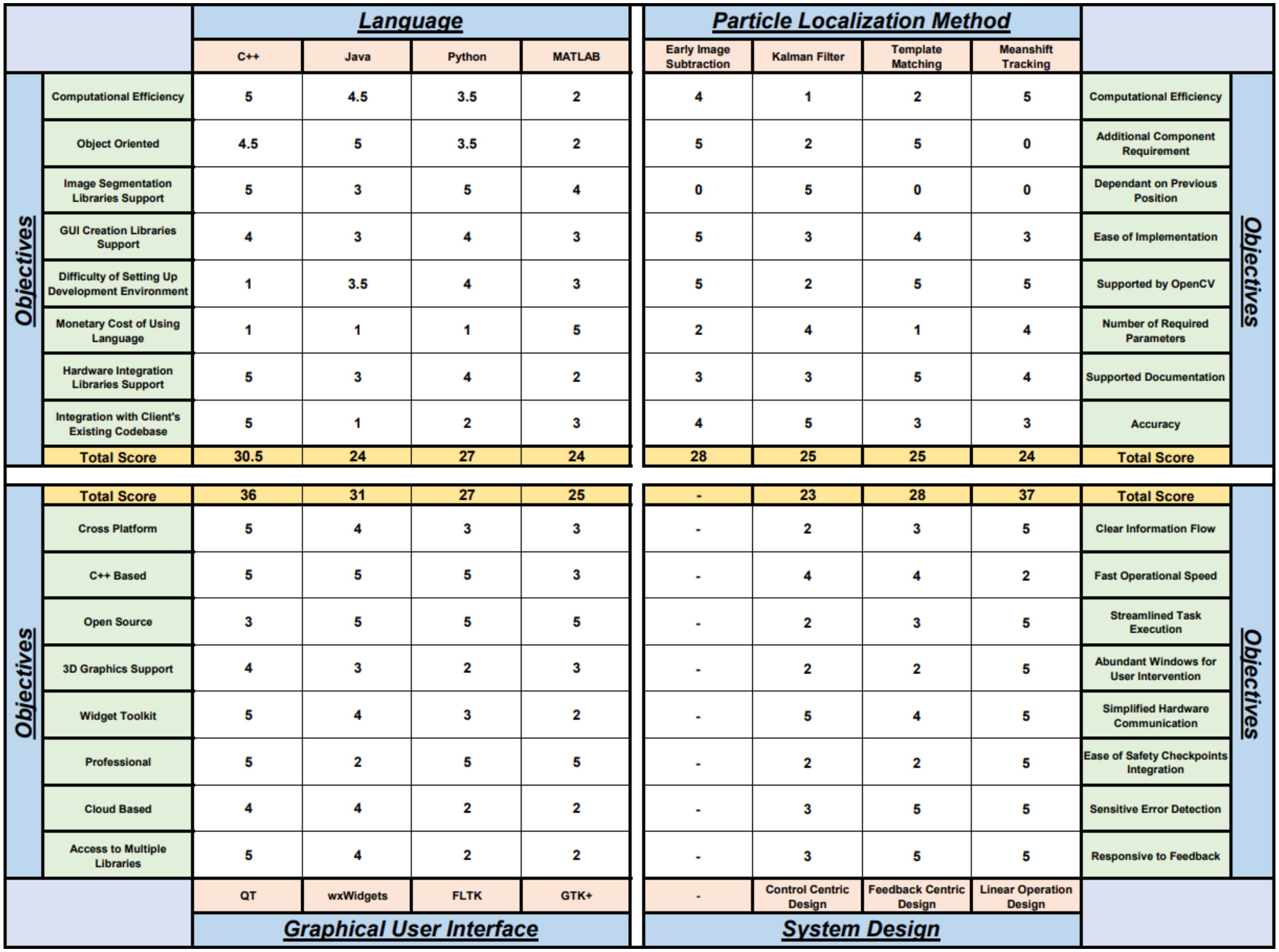
4. The **Physics** module receives the computed distance vector and material data from the control module. It then instructs WMP's magnetic control system (MCS) to execute the particle translation. Once the translation has been executed, the MCS relays this back to the physics module. If the attached safety checkpoint confirms no error has occurred, the translation is deemed successful and this status is relayed to the GUI. Then the system begins the next iteration.

Fulfillment of Engineering Characteristics:

Structural Division of Function	Design utilizes 5 modules (GUI, Image Processing, Control, and Feedback, and Physics) with distinct tasks.
Safe System Operation	Safety checkpoints are regularly placed throughout the system (one attached to each module) to cover error detection in various areas. When a checkpoint detects an error, it quickly breaks system execution to ensure safety and allows for user intervention.
Modular Hardware Connection	*See prior paragraphs under Proposed Architecture Design.
Platform Independent	*See prior paragraphs under Proposed Architecture Design.

Feedback from Sponsor
This design looks good. The information flow and timing of execution is easy to follow, the safety checkpoints are clearly defined, and the inputs and outputs are clearly defined.

C. Decision Matrix



Language

Objectives	C++	Java	Python	MATLAB
Computational Efficiency	5	4.5	3.5	2
Object Oriented	4.5	5	3.5	2
Image Segmentation Libraries Support	5	3	5	4
GUI Creation Libraries Support	4	3	4	3
Difficulty of Setting Up Development Environment	1	3.5	4	3
Monetary Cost of Using Language	1	1	1	5
Hardware Integration Libraries Support	5	3	4	2
Integration with Client's Existing Codebase	5	1	2	3
Total Score	30.5	24	27	24

Particle Localization Method

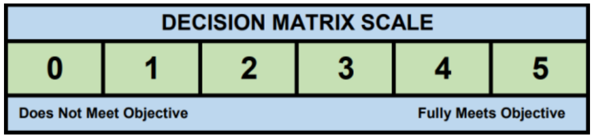
Objectives	Early Image Subtraction	Kalman Filter	Template Matching	Meanshift Tracking
Computational Efficiency	4	1	2	5
Additional Component Requirement	5	2	5	0
Dependant on Previous Position	0	5	0	0
Ease of Implementation	5	3	4	3
Supported by OpenCV	5	2	5	5
Number of Required Parameters	2	4	1	4
Supported Documentation	3	3	5	4
Accuracy	4	5	3	3
Total Score	28	25	25	24

Total Score	36	31	27	25
Cross Platform	5	4	3	3
C++ Based	5	5	5	3
Open Source	3	5	5	5
3D Graphics Support	4	3	2	3
Widget Toolkit	5	4	3	2
Professional	5	2	5	5
Cloud Based	4	4	2	2
Access to Multiple Libraries	5	4	2	2
Objectives	QT	wxWidgets	FLTK	GTK+

Graphical User Interface

Total Score	-	23	28	37
Clear Information Flow	-	2	3	5
Fast Operational Speed	-	4	4	2
Streamlined Task Execution	-	2	3	5
Abundant Windows for User Intervention	-	2	2	5
Simplified Hardware Communication	-	5	4	5
Ease of Safety Checkpoints Integration	-	2	2	5
Sensitive Error Detection	-	3	5	5
Responsive to Feedback	-	3	5	5
Objectives	-	Control Centric Design	Feedback Centric Design	Linear Operation Design

System Design



DECISION MATRIX SCALE					
0	1	2	3	4	5
Does Not Meet Objective					Fully Meets Objective

D. Rationale for Selected Approach

Programming Language

The eight factors (ordered from most important to least important) that we considered for choosing the project's programming language are: 1) integration with WMP's existing codebase, 2) good image processing support libraries, 3) good Graphical User Interface (GUI) development programs, 4) ease of hardware integration, 5) object-oriented based, 6) good computational efficiency, 7) ease of development environment set up, and 8) involved monetary costs.

As many of WMP's current magnetic hardware systems use Arduino microcontrollers, a large portion of their existing codebase is written in C/C++. Additionally, C++ has many supporting libraries for robotics programming. WMP has requested that our software integrate seamlessly with their existing codebase and their hardware. This points towards C++ as the prefered language of choice. To perform sophisticated image-processing techniques for particle tracking, the image-processing module requires high-quality supporting libraries. Java has libraries dedicated for image-processing such as OpenIMAJ and ImageIO. MATLAB also has dedicated libraries for image processing such as the famous Image Processing Toolbox. However since WMP requests that our software integrates well with their existing codebase, C++ has access to the robust computer vision library, OpenCV. In addition, the GUI should be built in a well-supported application development framework (API). The Qt API, which runs predominantly on C++, is well-known UI development program and has previously been used by WMP.

Because our system must execute its tasks quickly to enable well-paced magnetic particle delivery operation, computational efficiency is paramount. Using a low-level language (a language that is "close to the hardware") like C++ will help us improve our software's computational speed, unlike a slower language such as MATLAB. However, C++ also has the advantage of high-level features, namely object-oriented support. This would enable us to write code that can be structured into modules of functionally distinct tasks. As a result, code conceptual intuition improves and objects and functions can easily be reused throughout the software.

C++ has many free and well-supported Integrated Development Environment (IDEs) such as Microsoft Visual Studios, MinGW, and even Qt itself. The downside to OpenCV is that its setup is a complex task. In terms of monetary costs, all approaches are open-source and free to use except for MATLAB. Nonetheless, because C++ integrates well with WMP's existing codebase and hardware and excels in the factors mentioned above, we have chosen C++ as the optimal programming language for this project.

Particle Tracking Method (Image Segmentation Method)

Out of the four image tracking approaches considered for this application, we decided to use the early image subtraction method to determine particle location from image frames. This was mainly due to the unique circumstances with respect to the imaging procedure used for our project. An image frame without the particles ("early frame") can be procured prior to the particle delivery operation and subtracted from all subsequent captured image frames in which the particles are present. The resulting image contains a single area of high color intensity (so long as the field of view is kept constant) that indicates the general particle area. Further image filtering techniques (e.g. bilateral, dilation) can be implemented to narrow down exact particle location. While this technique is not the most computationally efficient, the segmentation time is sufficient for our tracking purposes (~20 ms for a single 750x750 image). Due to the immense image filtering and arithmetic support by various open-source computer vision libraries along with the circumstances regarding the imaging procedure and computational efficiency, this localization method was selected for development.

Template matching is a particle tracking technique similar to early image subtraction since it also requires setup prior to the particle delivery operation.. An image exclusively containing the particles at their initial location (“template frame”) is used to segment each incoming image frame. This technique is rather computationally inefficient as it requires metric calculation of each pixel within the template and incoming image frames (~100 ms). The template frame is also particle specific and would need to be replaced when imaging different particles. Additional computation is also required to account for particle rotation and orientation which would result in increased delay when segmenting an image batch.

Using attributes from a previously captured image frame (i.e. particle location and velocity), Kalman filtering can be used to predict and locate particles. Similarly to the early image subtraction method, a Kalman filter is not particle specific and can be used to locate multiple moving particles. Despite its robust performance, this technique was not selected due to increased computation required to compute particle velocity (which is variable). Kalman filtering is also not as well supported by computer vision libraries as the aforementioned segmentation methods, meaning that its implementation would be more difficult. While this method is accurate in a 2D imaging system, the application of a Kalman filter into a 3D environment will require additional computation because the particle is not moving along a 2D axis and may lead to inaccurate particle location. Due to these limitations, a Kalman filter was not our selected particle tracking method.

A mean-shift tracking algorithm can be used to track particle location through the use of histograms. This algorithm locates the area of maximum pixel density within an image frame by shifting the “area of focus” across the frame until certain user-defined criteria are satisfied (i.e. total pixel density). A major drawback to this method is the requirement for one area of large pixel density. Our project application includes imaging MRI slices of the entire cranium, meaning that image slices will consist of large areas of uniform pixel density. Thus, this method was not selected for development.

Graphic User Interface (GUI)

Four of the eight factors identified in the Decision Matrix for GUI program selection were deemed the most important (Commercial Popularity, Widget Toolkit, Available Libraries, Cross Platform Development) . Commercial popularity is significant as using a GUI software that is well-recognized by the commercial world can help us to meet FDA software standards, which has been one of WMP’s key goals for the project. It is also significant because more renowned UI programs tend to have stronger software and troubleshooting tools. Qt is the most popular of the four approaches, although wxWidgets was a close second.

The Widget Toolkit is defined as the available graphical control elements within the GUI design. Qt is strongest here because it has the largest amount of available widgets compared to the others. Qt’s in-software libraries are more robust than the other options, providing us with access to more UI-geared functions. Cross platform development is significant because our software should be transferable to several operating systems. Also as mentioned under “Programming Language,” Qt is a familiar software to WMP which means that our sponsors can more easily operate and tune the GUI to their preferences after project completion. Thus, Qt is the GUI API of choice.

System Architecture

The prior section Proposed Architecture Designs details the 3 system architectures that we considered for our software (Control-Centric Design, Feedback-Centric Design, and Linear Design with Regular Safety Checkpoints). The section also contains direct feedback from WMP on the designs. The eight factors listed in the Decision Matrix essentially emphasize operational clarity and safety.

The Control-Centric Design is attractive in theory due to the heightened role of the Control module compared to those in the other designs (e.g. calculating magnetic physical parameters and outputting commands to hardware). This is the design's greatest strength, allowing for very simplified hardware communication. However, it falls flat in many of the other areas. Information flow is incredibly ambiguous, as is the timing of task executions which lowers operational safety as a consequence. Also, it should be noted that simplified hardware communication is not unique to this design. It is a characteristic of the other designs too since Control module in each design is also the only link between software and hardware.

The Feedback-Centric Design is superior to the Control-Centric Design in that safety is the primary focus. Information flow and task execution are represented in a more intuitive, cyclical pattern. The Feedback module is at the literal center of the operation and provides inputs to most of the other modules. When a translation error is detected, Feedback has power over the Control module's output to the hardware. However, the weaknesses of this design are the still-vague information flow and timing of task executions. Also, because the Feedback module handles too many tasks, it will most likely not perform well in practice.

Unlike the other two designs, the Linear Design with Regular Safety Checkpoints excels in both operational clarity and safety. Modules execute linearly and must "pass" regular safety checkpoints to output their information to the next module down the pipeline. Any detected errors/failures break execution and notify the user to take corrective action. This reaction to error is a benefit on the development end. Errors can be more easily caught anywhere from start to finish and fixed to ensure smooth operation. The primary weakness of this design is the expected loss of operational speed due to the lack of high-level concurrency of execution. However, this drawback is something that we accept as a manageable tradeoff for increased safety and simplified design. Thus, this architecture is selected for our software.

E. System Components Description for Selected Approach

Note:
← indicates the module sending the input.
→ indicates the module receiving the output.

Graphical User Interface (GUI):

Input: User-Drawn Path Input (← **User**), MR Image Array (← **Camera/MRI Machine**), Clock (← **System**), Processed Image with Marked Particles (← **Image Segmentation**)
Output: Discretized Particle Delivery Path (→ **Image Segmentation**), GUI Execution Status (→ **GUI**)

- The GUI is used to obtain the path drawn by the physician.
- Displays the processed images, particle location, particle statistics, and operational status to update the physician on the progress of the magnetic particles.
- Prompts the user to draw a new path if necessary.
- Allows the user to terminate operation.

Image Segmentation Module:

Input: Image Array, Discretized Particle Delivery Path (← **GUI**)
Output: Current Particle(s) Location, Material Data, Image Segmentation Execution Status (→ **GUI**), Processed Image with Marked Particles (→ **GUI**)

- This module is used to determine the location of the magnetic particles.
- Creates a 3D model from 2D image array input.
- Uses image segmentation techniques to locate the magnetic particles in three dimensions.
- Determines material composition of local particle environment.

Control Module

Input: Current Particle(s) Location, Image Segmentation Execution Status (← **Image Segmentation**), User-Drawn Path(← **GUI**)
Output: Computed Distance Vector (→ **Physics**), Control Execution Status (→ **GUI**)

- Used to determine if the particles are still on the user-created path.
  - If particle(s) are not on track, computes distance vector to return particle(s) on track.
  - If particle(s) are on track, computes distance vector to move particle(s) to next coordinate along user-defined path.
- Ensures computed vector lies within the MCS's technical capabilities.
- Passes vector onto Physics module.

Physics Module:

Input: Computer Force Vector (← **Control**)
Output: Electrical Current Specifications For Next Particle Translation (→ **WMP MCS**), Physics Execution Status (→ **GUI**)

- Interfaces with hardware system to instantiate a magnetic field gradient to move particles in the direction of a vector supplied by the Control module
- Sends operational status to GUI. This includes if there are any issues moving the particle(s) and if the path needs to be redrawn by user.
- This module will be implemented by WMP.

F. Applicable Formulae and/or Algorithms

Below are engineering formulae that we expect to implement in our project. Note that this list is not rigid. We expect to utilize more formulae and algorithms as our project develops further and our strategy matures.

Formulae

- **Biot-Savart Law** - The magnetic field *B* at a position *r* generated by a current *I* in 3D-space over a can be computed using the Biot-Savart Law:

$$B(r) = \frac{\mu_0}{4\pi}\int_C \frac{Idl \times r'}{|r'|^3}$$

- **Solenoid Magnetic Field** - The magnetic field *B* for a solenoid with *n* turns and current *I* can be derived from Biot-Savart Law:

$$B = \mu_0 n I$$

- **Gauss's Law for Magnetism** - The magnetic field *B* out of a closed surface S can be calculated using Gauss's Law for Magnetism:

$$\oiint_S B \cdot dA = 0$$

Algorithms

- **Bilateral Filter Blurring** - The output "blurred" pixel value g(i,j) of input pixel f(i,j) using filter kernel h(k,l) can be computed using the following formula:

$$\boldsymbol{g(i,j) = \sum_{k,l} f(i+k, j+l)h(k,l)}$$

G. Applicable Standards and Codes

The following standards are issued by the Food and Drug Administration (FDA) as industry guidelines for documenting the development, clinical safety, and verification and validation of medical software devices intended for market submission. Note that the FDA standards define “medical software device” as any device that contains one or more software components, parts, or accessories to carry out or supplement medical procedures. A medical software device may take the form of firmware, stand-alone software, software accessories to medical devices, etc. Because our software device is intended for therapeutic applications, we will adhere to these standards. We will produce and expand upon the recommended documentation alongside the development of our software.

Guidance for Industry and FDA Staff: Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices (2005) [7]

This standard provides information regarding the FDA-recommended documentation for industry to provide in their pre-market submissions of medical device software. Recommended documentation covers software requirements and implementation, hazard analysis, verification and validation, development history, and other areas. Criteria for injury classification and the medical software level of concern are also clearly defined.

Key Areas

- **Level of Concern:** Criteria for establishing the appropriate level of concern (major, moderate, minor) to a medical software device. It is based upon device function and associated risk of injury.

- **Software Description:** Comprehensive overview of device features controlled by the software. Describes software operation and intended operational environment.

- **Device Hazard Analysis:** Analysis of hardware and software hazards associated with the device’s intended use. Includes corrective measures taken during hazards.

- **Software Requirements Specification (SRS):** Provides software requirements including areas such as functional, performance, interface, design, etc.

- **Architecture Design Chart:** Flowchart depicting relationships among major functional units in the software. Hardware interactions and data flow are typically included.

- **Software Design Specification (SDS):** Software requirements implementation. References other documents such as detailed software specifications.

- **Traceability Analysis:** Link between product design requirements, design specifications, and testing requirements. Hazards are coupled with mitigation testing.

- **Software Development Environment Description:** Software development life cycle plan. This may include a list or description of software coding standards, configuration management, and maintenance.

- **Verification and Validation Documentation:** Testing documentation used for vindicating that the medical software device performs as expected and meets stakeholder requirements.

- **Revision Level History:** Documentation that records and lists software device version changes.

Software as a Medical Device (SaMD): Clinical Evaluation Guidance for Industry and Food and Drug Administration Staff (2017) [8]

This standard discusses the International Medical Device Regulators Forum (IMDRF) process used to clinically evaluate medical device software. The FDA considers IMDRF as a significant forum to discuss future directions in medical device regulatory harmonization. The 4 main components of the evaluation process include 1) Clinical Evaluation, 2) Valid Clinical Association, 3) Analytical Validation, and 4) Clinical Validation.

Key Areas

- **Clinical Evaluation**: A comprehensive list of activities conducted in the assessment and analysis of a SaMD's clinical safety following the document's tenets will be upkept.

- **Clinical Association**: Statement regarding the extent to which the SaMD's output is clinically accepted or well-founded and corresponds accurately in the real world to the healthcare situation and condition will be established along with a valid association between the SaMD output and the SaMD's targeted clinical condition as recommended by the literature.

- **Analytical Validation**: Objective evidence demonstrating that the software was constructed properly will be provided following the document's template along with extensive documentation proving that all software specifications were met.

- **Clinical Validation**: Software functionality will be tested against existing data from studies conducted for the same intended use or for a different intended use following the guidelines listed in the documentation.

General Principles of Software Validation; Final Guidance for Industry and FDA Staff Document issued (2002) [9]

This standard outlines general validation principles offered by the FDA to be used in the design, development, or manufacturing of medical device software. Suggested guidelines include System and Software Requirements, Data Throughput, Risk Assessment, and Error Management.

Key Areas

- **Software Requirement Specifications**: A comprehensive summary of software requirements including performance, I/O, functionality, and user interaction among other critical specifications.

- **Design Review**: Proposed system designs will be systematically examined to evaluate the capability of the design to meet system requirements.

- **Testing**: An exhausted list of test cases will be created using the predefined tenets for software testing. Verification and validation procedures will be taken into consideration as noted in the document.

- **Maintenance and Software Changes**: Sufficient regression analysis will follow any perfective, corrective, or adaptive software maintenance performed as suggested by the literature.

5. Next Steps

A. Hardware Testing and Evaluation

For the Fall 2018 semester, Weinberg Medical Physics has provided us with a magnetic coil array consisting of two pairs of solenoid electromagnets (four total) to be used for initial project testing and evaluation. This system allows for particle manipulation in 2D across a ~5.08 cm^2 surface area and is powered by an independent voltage supply. Current through each solenoid is controlled through an onboard motor controllers capable of providing 60 Amperes of current through each channel (2 channels per motor controller, 2 motor controllers total). Communication with motor controllers will be achieved using microcontrollers such as Arduino or Raspberry Pi. Ice water is circulated through the coil array system to dissipate heat created by running current through each solenoid. A high-resolution optical camera will be used as the particle imaging system.

For the Spring 2019 semester, Weinberg Medical Physics will allow us to utilize a simple MRI machine capable of 3D particle manipulation. Towards the end of the Spring 2019 semester, we will validate our control system by performing tests using their electropermanent hardware system.

B. Criteria/Metrics

To objectively evaluate project success, we have determined the following testing criteria for each of the four major software components. ALL criteria must be met to ensure success.

Note: Testing procedures are subject to change next semester as new equipment will be made available by WMP for use. Project Magneto will also transition from 2D optical camera stream to 3D MR image arrays.

GUI Criteria

- All buttons successfully perform the specified functions. Button behavior is controlled so that the user cannot deviate from the operational procedure.
- Users is able to design a complete delivery path by creating path segments across image slices; path segments can be modified to make appropriate changes.
- Acquired particle location coordinates are accurate and can be represented as real-world distances.
- Particle location is displayed on the GUI as it is updated.

Image Segmentation Criteria

- Locates particles in image slices with minimum 95% accuracy.
- Processes and output images at a maximum period of 30 ms/frame.

Control Module Criteria

- Computes accurate distance vectors to move particles through delivery path.
- Determines corrective actions if particles are not clustered around desired point along path.
- Communicates properly with Physics Module.
- Provide timely feedback to physician.
- Executes operational commands from physician (i.e. pause/terminate operation, redraw path, etc.).

Physic Module Criteria
- Carries out correct particle translation using the control module outputs with 100% accuracy. - Begins particle translation with minimal delay (< 1s) after receiving control module outputs.

C. Budget

As of 11/15/2018, our project is primarily built through open-source software and total operational cost is $0. Any need for a budget will be discussed in future meetings with our sponsors and advisors. Hardware used for system testing and validation will be provided by Weinberg Medical Physics.

Item	Quantity	Cost
Qt Development Environment	4	$0
Microsoft Visual Studio 2017 Community	4	$0
OpenCV	4	$0
Sourcetree Version Control (AWS)	4	$0
ImageJ	4	$0
Total Operational Cost		$0

D. Progress-to-Date and Schedule for the Next Semester

Completed high-level tasks (to-date):

- **Image Segmentation Module**
  - Researched and implemented particle detection algorithm (early image subtraction method)

- **Graphic User Interface v0.01**
  - Designed layout for GUI
  - Implemented functionality for recorded videos, 2D image stacks, and 2D optical stream

- **Control Module**
  - Created function to compute distance vector for particle translation
  - Built functions used to determine if particles is on track/off track
  - Implemented functions to determine next point along path inputted by user
  - Integrate control module with GUI pathing input
  - Created classes describing points for user-input path, particle-location, and material data

- **Website**
  - Create project website
  - Uploaded completed project documentation (i.e. Project Proposal, Presentations, Gantt Chart, etc.)

Current high-level tasks (for remainder of Semester 1):

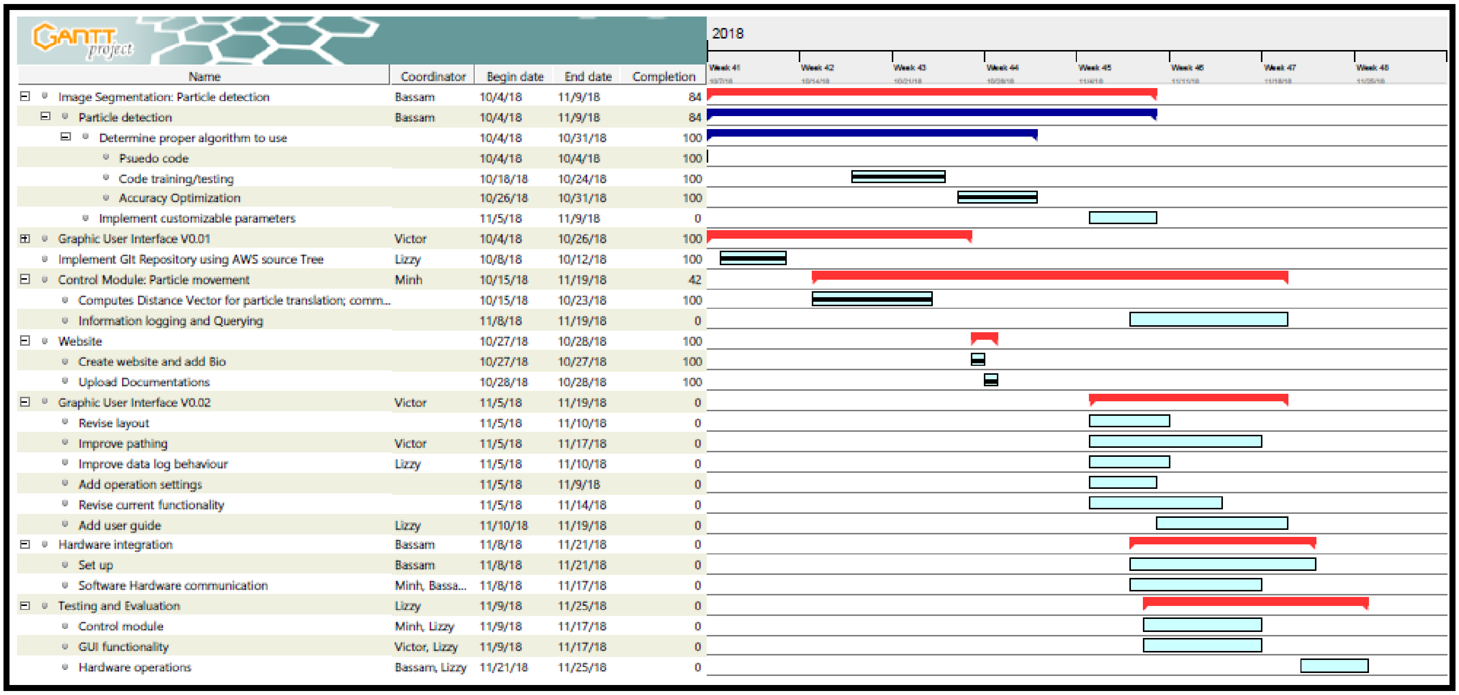
- **Image Segmentation Module**
  - Implement customizable parameters

- **Graphic User Interface v0.02**
  - Revise layout
  - Improve pathing
  - Improve data log behavior
  - Add operation settings
  - Revise current functionality
  - Add user guide

- **Hardware Integration**
  - Become familiarized with 4-coil array hardware setup
  - Implement Software-Hardware communication

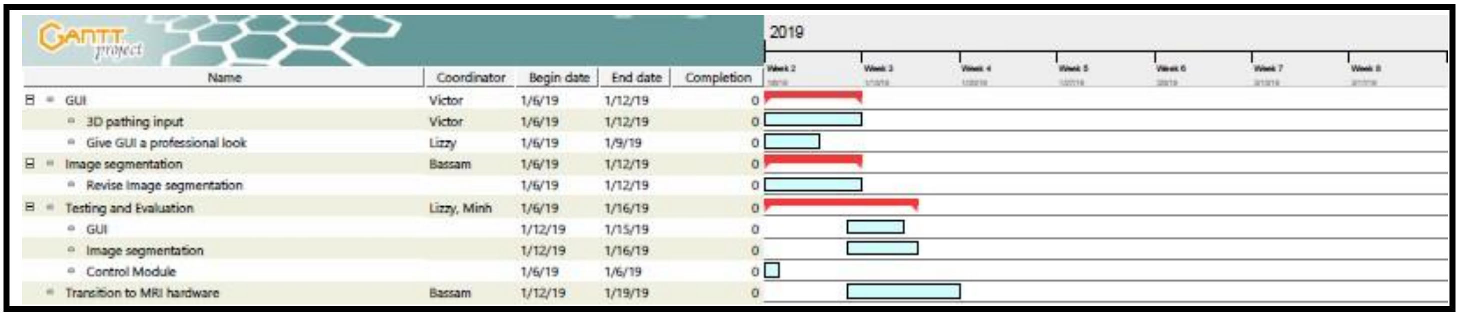
- **Testing and Evaluation**
  - Test Control Module performance
  - Test GUI performance
  - Evaluate hardware and software operations

Semester 1 Gantt Chart - Fall 2018



Name	Coordinator	Begin date	End date	Completion
⊟ Image Segmentation: Particle detection	Bassam	10/4/18	11/9/18	84
⊟ Particle detection	Bassam	10/4/18	11/9/18	84
⊟ Determine proper algorithm to use		10/4/18	10/31/18	100
Psuedo code		10/4/18	10/4/18	100
Code training/testing		10/18/18	10/24/18	100
Accuracy Optimization		10/26/18	10/31/18	100
Implement customizable parameters		11/5/18	11/9/18	0
⊞ Graphic User Interface V0.01	Victor	10/4/18	10/26/18	100
Implement Git Repository using AWS source Tree	Lizzy	10/8/18	10/12/18	100
⊟ Control Module: Particle movement	Minh	10/15/18	11/19/18	42
Computes Distance Vector for particle translation; comm...		10/15/18	10/23/18	100
Information logging and Querying		11/8/18	11/19/18	0
⊟ Website		10/27/18	10/28/18	100
Create website and add Bio		10/27/18	10/27/18	100
Upload Documentations		10/28/18	10/28/18	100
⊟ Graphic User Interface V0.02	Victor	11/5/18	11/19/18	0
Revise layout		11/5/18	11/10/18	0
Improve pathing	Victor	11/5/18	11/17/18	0
Improve data log behaviour	Lizzy	11/5/18	11/10/18	0
Add operation settings		11/5/18	11/9/18	0
Revise current functionality		11/5/18	11/14/18	0
Add user guide	Lizzy	11/10/18	11/19/18	0
⊟ Hardware integration	Bassam	11/8/18	11/21/18	0
Set up	Bassam	11/8/18	11/21/18	0
Software Hardware communication	Minh, Bassa...	11/8/18	11/17/18	0
⊟ Testing and Evaluation	Lizzy	11/9/18	11/25/18	0
Control module	Minh, Lizzy	11/9/18	11/17/18	0
GUI functionality	Victor, Lizzy	11/9/18	11/17/18	0
Hardware operations	Bassam, Lizzy	11/21/18	11/25/18	0

Semester 2 Gantt Chart - Spring 2019



Name	Coordinator	Begin date	End date	Completion
GUI	Victor	1/6/19	1/12/19	0
3D pathing input	Victor	1/6/19	1/12/19	0
Give GUI a professional look	Lizzy	1/6/19	1/9/19	0
Image segmentation	Bassam	1/6/19	1/12/19	0
Revise image segmentation		1/6/19	1/12/19	0
Testing and Evaluation	Lizzy, Minh	1/6/19	1/16/19	0
GUI		1/12/19	1/15/19	0
Image segmentation		1/12/19	1/16/19	0
Control Module		1/6/19	1/6/19	0
Transition to MRI hardware	Bassam	1/12/19	1/19/19	0