Carnegie Mellon University

Robotics Capstone Project

# Test Plan

*Friction Force Explorers:*

*Don Zheng*
*Neil Jassal*
*Yichu Jin*
*Rachel Holladay*

supervised by
Dr. David Wettergreen

Version 1.0
November 21, 2016

# Contents

# List of Figures

# 1 Design Verification

## 1.1 Writing Implement

### 1.1.1 Performance Test: Loading

**Test Question:** Is a human operator able to relaod the writing tool within the required tiem limit of 10s, and by how much?
**Operational Procedure:** With a writing tool already in the writing assembly, a human test subject will perform 3 reloads separately timed reloads of the tool.
**Metric:** Duration of the shortest reload time.
**Acceptance Criteria:** The shortest reload time is under 10s.
**Requirement(s) Verified:** NFR 14

### 1.1.2 Performance Test: Writing Quality

**Test Question:** Is the mark produced by the robot of acceptable quality?
**Operational Procedure:** Using a fully loaded writing tool, the robot attempts to draw along a route with at least 4ft of travel distance and 3 turns exceeding 50 degrees. Verify that the resulting drawing is of acceptable quality.
**Metric:** Percent thickness of the route at its narrowest point compared to the maximum thickness of a line created with the writing tool, and whether or not there are complete breaks in the line.
**Acceptance Criteria:** The percent thickness is at least 70%, and there are no complete breaks in the line.
**Requirement(s) Verified:** NFR 6

### 1.1.3 Functional Test: Replace Writing Tool

**Test Question:** Is a human user able to replace the writing tool?
**Operational Procedure:** A human test subject attempts to replace a writing tool already inside the robot. **Metric:** Whether or not the human succeeds in the task before giving up.
**Acceptance Criteria:** The human must successfully replace the writing tool without giving up.
**Requirement(s) Verified:** FR 7

### 1.1.4 Functional Test: Writing Pressure Control

**Test Question:** Can the writing tool be actuated to move up an down?
**Operational Procedure:** With a writing tool in the writing assembly, the motors for moving up and down attempt to move throughout their range.
**Metric:** Whether or not the writing tool moves.
**Acceptance Criteria:** The writing tool must move through the writing assembly's full vertical range.
**Requirement(s) Verified:** FR 10

### 1.1.5 Functional Test: Simultaneous Driving and Writing

**Test Question:** Does the writing tool make a mark when pushed down?
**Operational Procedure:** With a fully loaded writing implement, the robot will mark a 1ft line on the writing surface.
**Metric:** Whether or not any discernible mark is made and the full distance is travelled.
**Acceptance Criteria:** A discernible mark must be made and the full distance must be travelled without the robot becoming stuck or breaking.
**Requirement(s) Verified:** NFR 6

### 1.1.6 Functional Test: Marking

**Test Question:** Does the writing tool make a mark when pushed down?
**Operational Procedure:** The robot will press down on a writing surface with a fully loaded writing implement.
**Metric:** Whether or not a any discerible mark is made.

**Acceptance Criteria:** A discernible mark must be made.
**Requirement(s) Verified:** NFR 6

#### 1.1.7 Functional Test: Force Sensor

**Test Question:** Can the force sensor measure applied force?
**Operational Procedure:** The robot will press a writing implement down onto a writing surface with three different pressure settings: optimal, underactuated, and overactuated.
**Metric:** Whether or not the sensor can distinguish between the three different pressure settings.
**Acceptance Criteria:** The sensor must be able to distinguish between all three settings.
**Requirement(s) Verified:** NFR 6

#### 1.1.8 Failure Mode: Out of Writing Material

**Description:** This failure mode describes the situation when a writing implement is loaded inside a robot agent, and runs out of writing material.
**Cause:** Overuse of the writing implement.
**Effects:** The robot agent moves around and attempts to continue drawing, without making physical marks.
**Criticality:** This is a critical failure, as it requires the user to replace the implement before drawing can continue. If the user fails to replace the implement, lines will be missing from the drawing.
**Safety Hazards:** There are no safety hazards associated with this failure mode.

#### 1.1.9 Failure Mode: Writing Mechanism Failure

**Description:** This failure occurs when the mechanism that raises and lowers the writing implement does not work. This causes the robot to be incapable of either drawing a line, or moving on the writing surface without drawing a line.
**Cause:** This is caused by a communication failure, as described by Sec. **??**, or more likely, by the raise/lowering mechanism breaking.
**Effects:** The robot agent will be unable to alter whether or not it is writing as it moves. This can cause either missing lines or incorrect ones, depending on the state of the writing mechanism.
**Criticality:** This is a critical error as it directly affects the quality of the lines being drawn. Users can resolve this issue after receiving an error report to continue operation. However, it is possible the drawing has been compromised.
**Safety Hazards:** There are no safety hazards associated with this failure mode.

### 1.2 Locomotion

#### 1.2.1 Performance Test: Accuracy

**Test Question:** Is the robot able to drive with positional and rotational accuracy?
**Operational Procedure:** The robot drives along a predetermined testing route consisting of at least 3 feet of linear distance and 90 degrees of turn.
**Metric:** The difference of the robot's final position and orientation from the intended position and orientation.
**Acceptance Criteria:** The robot's position must be less than 1 inch away from the intended position, and its orientation must be within 10 degrees of the intended orientation.
**Requirement(s) Verified:** NFR 12

#### 1.2.2 Functional Test: Speed

**Test Question:** Can the robot reach a desired speed?
**Operational Procedure:** The robot will drive along a straight line for 5ft during a timed trial.
**Metric:** The time required for the robot to reach the end of the line.
**Acceptance Criteria:** The robot must reach the end of the 5ft line in 20 seconds.
**Requirement(s) Verified:** NFR 5

#### 1.2.3 Failure Mode: Inaccurate Motion

**Description:** This failure mode describes the situation in which a robot agent is unable to accurately follow motion commands. An example of this is if the robot is commanded to move 10 inches, but localization detects it only moving 4.
**Cause:** Inaccurate motion could be a result of slippage of the wheels or failed motor encoders.
**Effects:** Inability to move accurately can cause a user-reported error, which the user can resolve to continue operation.
**Criticality:** This is a minor failure, as it does not end system operation and can be resolved by the user.
**Safety Hazards:** The only safety hazard is with regard to the drawing surface; slippage could cause minor destruction of the surface.

#### 1.2.4 Failure Mode: Failure to Move Omnidirectionally

**Description:** Failure to make omnidirectional movement means a robot agent cannot move in an arbitrary direction on the flat plane represented by the drawing surface.
**Cause:** Similar to Sec. 1.2.3, this could be caused by wheels causing slippage in some directions. Alternatively, a broken wheel or motor could have this effect as well.
**Effects:** Failling to move omnidirectionally could result in incorrect drawings - the robot agent can no longer make arbitrarily sharp curves to faithfully recreate the input image. When detected, robot operation should halt and report to the user to be fixed.
**Criticality:** This is a critical error, as it has a direct relation on the quality of the drawing.
**Safety Hazards:** As with Sec. 1.2.3, the only safety hazard is that a broken wheel could damage the drawing surface.

### 1.3 Localization

#### 1.3.1 Performance Test: Robot Position Accuracy

**Test Question:** Is the localization system able to accurately determine the position of each robot?
**Operational Procedure:** Both robots sit stationary within the working bounds, and the localization system attempts to determine their locations.
**Metric:** The difference of the robots' actual position from the position reported by the localization system.
**Acceptance Criteria:** The reported position must be within 1/10in of the actual position.
**Requirement(s) Verified:** NFR 6, FR 4

#### 1.3.2 Performance Test: Bounds Accuracy

**Test Question:** Is the localization system able to accurately determine the boundaries of the workspace?
**Operational Procedure:** The localization system attempts to determine the bounds of the workspace.
**Metric:** The total difference in distance between the reported corners of the workspace and the actual corners.
**Acceptance Criteria:** The total difference must not exceed 1in.
**Requirement(s) Verified:** FR 4

#### 1.3.3 Functional Test: Robot Position

**Test Question:** Can the localization system find the robot?
**Operational Procedure:** With a single robot within the working bounds, the localization system attempts to determine the robot's location.
**Metric:** Whether or not a location is returned by the localization system.
**Acceptance Criteria:** The system must return a location for the robot.
**Requirement(s) Verified:** FR 4, NFR 6

#### 1.3.4 Functional Test: Bounds

**Test Question:** Can the localization system find the working bounds?
**Operational Procedure:** The localization system attempts to find all four corners of the working bounds while they are all in its field of view.
**Metric:** Whether or not locations are returned for all four corners.
**Acceptance Criteria:** The system must return locations for all four corners of the working bounds.
**Requirement(s) Verified:** FR 4, NFR 6

#### 1.3.5 Failure Mode: Camera Failure

**Description:** A camera failure occurs when the localization camera, mounted above the drawing surface, is incapable of gathering and/or sending data to the off-board processor for localization.
**Cause:** The two potential causes for a camera failure are insufficient power being supplied to the camera, or improper mounting. Improper mounting can cause the camera to fall or hang, which results in skewed and miscalibrated camera data.
**Effects:** The effect of camera failure results in localization being poor or impossible, which can halt operation. This can be temporary, as a user-reported error would be generated to resolve the issue and continue operation.
**Criticality:** This failure is of medium importance, as while it halts operation, it can be resolved by the user to continue the drawing process.
**Safety Hazards:** The only safety hazard exists if the camera falls entirely from its mount, in which case it may fall on a person below.

#### 1.3.6 Failure Mode: Unusable Localization Data

**Description:** This failure mode exists when the offboard processing system is unable to localize.
**Cause:** Causes include miscalibrated camera data, or incorrectly placed bounds tags. For example, the bounds tags could be placed in a shape that does not reflect the drawing surface accurately, resulting in incorrect localization. Blurry data could also result in misreading localization tags.
**Effects:** If localization cannot be completed, robot operation will halt to avoid performing undefined actions. This error can be resolved by the user recalibrating or fixing the issue causing bad data.
**Criticality:** Similar to Sec. 1.3.5, this failure is of medium criticality and can be resolved by the user.
**Safety Hazards:** There are no safety hazards that result from this failure mode.

#### 1.3.7 Failure Mode: Vision Tag Occlusion

**Description:** Occlusion of the vision tags is when the camera does not have direct line-of-sight of any vision tag used for robot and bounds localization.
**Cause:** Tag occlusion is likely the result of an obstacle unexpectedly entering the scene. This could be a person walking over the drawing surface, or over the edges of the camera view, where the vision tags represnting the surface bounds are.
**Effects:** Inability to find a tag results in incomplete localization, and will pause operation until the user can resolve the issue. This guarantees all robots are tracked continually during operation, as well as staying within bounds of the drawing surface.
**Criticality:** This is a minor failure, as robot operation can easily be corrected and operation can continue.
**Safety Hazards:** There are no safety hazards that result from this failure mode.

### 1.4 Image Processing

#### 1.4.1 Performance Test: Accuracy

**Test Question:** How closely does the image processor output resemble the original image?
**Operational Procedure:** The image processor takes in a valid input image and produces a result.
**Metric:** The percentage of lines that were accurately captured by the image processing system.

**Acceptance Criteria:** The system must succeed in capturing 95% of the lines in the input image.
**Requirement(s) Verified:** NFR 6

#### 1.4.2 Functional Test: Return Data

**Test Question:** Does the image processor return data usable by the planner?
**Operational Procedure:** The image processor takes in a valid input image and attempts to produce a series of lines from it.
**Metric:** Whether or not usable output is produced.
**Acceptance Criteria:** The image processor must be able to produce a series of lines for the planner.
**Requirement(s) Verified:** FR 11

#### 1.4.3 Functional Test: Reject Improper Input

**Test Question:** Is the image processor able to detect and reject improper input?
**Operational Procedure:** The image processor takes in an invalid input image (not an image file, or is not only consisted of lines).
**Metric:** Whether or not input is rejected.
**Acceptance Criteria:** The invalid input must be rejected.
**Requirement(s) Verified:** FR 11

#### 1.4.4 Functional Test: Keep Lines Within Bounds

**Test Question:** Does the image processor keep lines to be drawn within the working bounds?
**Operational Procedure:** The image processor takes in a valid input and produces an output in the context of the bounds.
**Metric:** Whether or not all lines lie within the working bounds.
**Acceptance Criteria:** All lines must lie within the working bounds.
**Requirement(s) Verified:** FR 4, FR 11

#### 1.4.5 Failure Mode: Unable to Process Input Image

**Description:** Inability to process user input refers to the image processing subsystem failing to determine a set of lines usable for work distribution, planning, and scheduling.
**Cause:** This could be caused by an unreadable input, or input drawings that do not conform to requirements. An example of this would be an input that contains background noise, making it unsuitable for processing and drawing.
**Effects:** The effect is that another input, or a corrected version of the initial input, will have to be supplied for the system to continue operation.
**Criticality:** This failure is critical to system operation, as no drawing can be made until the input can be properly processed.
**Safety Hazards:** There are no safety hazards involved in this failure mode.

### 1.5 Work Scheduling, Distribution and Planning

#### 1.5.1 Performance Test: Executable Plans

**Test Question:** How consistent is the planner at generating executable plans, ie those that avoid collision and stay within bounds?
**Operational Procedure:** Given a set of example drawing inputs, run each input and check the plan for potential robot-robot collisions and out of bounds driving.
**Metric:** Ratio of number of unacceptable plans, those that would involve collision or driving out of bounds, over the total number of plans.
**Acceptance Criteria:** Almost all, 99% of plans would not involve collision or out-of-bounds if executed.
**Requirement(s) Verified:** FR 4, NFR 11

#### 1.5.2 Performance Test: Execution Distribution

**Test Question:** How efficiently is execution time, i.e. the total time robots spend moving, distributed?
**Operational Procedure:** Given a set of example drawing inputs, run each input and record the total

time each robot spends moving.
**Metric:** We define execution efficiency as $\frac{\min(execution(R_0), execution(R_1))}{\max(execution(R_0), execution(R_1))}$ where execution($R_0$) refers to the execution time of robot 0 and execution($R_0$) refers to the execution time of robot 1
**Acceptance Criteria:** Execution efficiency of 0.75.
**Requirement(s) Verified:** NFR 5, NFR 9

#### 1.5.3 Performance Test: Drawing Distribution

**Test Question:** How efficiently is drawing time, i.e. the total time robots spend drawing, distributed?
**Operational Procedure:** Given a set of example drawing inputs, run each input and record the total time each robot spends drawing.
**Metric:** We define drawing efficiency as $\frac{\min(draw(R_0), draw(R_1))}{\max(draw(R_0), draw(R_1))}$ where draw($R_0$) refers to the drawing time of robot 0 and draw($R_0$) refers to the drawing time of robot 1
**Acceptance Criteria:** Drawing efficiency of 0.75.
**Requirement(s) Verified:** NFR 5, NFR 9

#### 1.5.4 Performance Test: Speedup

**Test Question:** What speedup is achieved by using two robots instead of one?
**Operational Procedure:** Given a set of example drawing inputs, run each input first with one robot and then with two. Time the execution time of each variant.
**Metric:** The comparison of duration, i.e. $\frac{executiontimewith2robots}{executiontimewith1robot}$.
**Acceptance Criteria:** According to our requirements we expect a speedup of 2x.
**Requirement(s) Verified:** NFR 5

#### 1.5.5 Functional Test: Collision Free

**Test Question:** Does the planner and executor generate collision free plans?
**Operational Procedure:** Given a set of example drawing inputs, run each input and check for any robot-robot collisions during execution.
**Metric:** Boolean across each plans on whether a collision occurred.
**Acceptance Criteria:** We only accept if collisions were avoided on 95% of our test cases.
**Requirement(s) Verified:** NFR 11

#### 1.5.6 Functional Test: Autonomy

**Test Question:** Does the system require no user input beyond adding the image to be drawn (except for error handling)?
**Operational Procedure:** After having input a plan, press "Run" on the system and observe if the system requires user input to finish the drawing.
**Metric:** Boolean on whether user input was required, excluding input relating to errors.
**Acceptance Criteria:** Accept only if no input was required.
**Requirement(s) Verified:** FR 2

#### 1.5.7 Failure Mode: Fail to Plan

**Description:** This failure mode occurs when the offboard system is unable to generate a valid plan for the robot agents. This means the main controller is unable to command the robots to successfully complete the input drawing.
**Cause:** Failure to create a valid plan could arise from an out of bounds drawing. Other reasons include the image processing result being incorrect, which forces the work planner to incorrectly assign and generate plans.
**Effect:** The system is unable to complete an invalid drawing, and cannot begin autonomous operation.
**Criticality:** This is a system-critical failure due to the fact that the system cannot recreate the drawing if it cannot generate a robot motion plan to do so.
**Safety Hazards:** There are no safety hazards associated with this failure mode, given that it is entirely software-based.

## 1.6 Communication

### 1.6.1 Performance Test: Uptime

**Test Question:** What is the uptime on our ability to communicate data to between the robots and the offboard system?
**Operational Procedure:** Run the system for a significant period of time (several hours) and record any communication downtime or data loss during communication.
**Metric:** Time duration of down communication and packet loss.
**Acceptance Criteria:** Operational 95% of the time.
**Requirement(s) Verified:** FR 8, FR 12

### 1.6.2 Functional Test: Sending and Receiving Data

**Test Question:** Can the robot send and receive data from the off-board device and can the off-board device send and receive data to the robot?
**Operational Procedure:** Send data from the off-board device to the robot and verify the robot received it. Send data from the robot to the off-board device and verify the off-board device received it.
**Metric:** Four booleans on whether the data is successfully sent and recieved on both ends.
**Acceptance Criteria:** We must succeed on all four accounts.
**Requirement(s) Verified:** FR 8

### 1.6.3 Functional Test: Data Parsing

**Test Question:** Can the data on each side (robot, off-board device) be parsed by each other?
**Operational Procedure:** Send data from the off-board device to the robot and verify the robot received it and can execute it. Send data from the robot to the off-board device and verify the off-board device received it and can respond to it.
**Metric:** Check whether the data was successfully parsed on all sides.
**Acceptance Criteria:** We require all data be parsable.
**Requirement(s) Verified:** FR 8

### 1.6.4 Failure Mode: Loss of Connection

**Description:** A loss of connection occurs when a robot agent and the off-board processing unit are unable to send data between each other. As per Sec. 8, the robot system expects consistent communication. This means that a failure resulting in intermittent or sparse connection will be treated equivalently to no connection.
**Cause:** This failure mode is the result of a robot agent and the off-board unit being unable to connect. This could be the result of a hardware failure, in which the either of the robot or off-board device's transmitter fail. Other causes could be loss of signal due to distance between the two devices, or obstacles that attenuate or disturb communication.
**Effects:** In the case that the off-board device cannot communicate with the robot, robots should be aware of a dropped connection and cease all locomotion. This will prevent robots from moving out of bounds or into collision, as without connection they can no longer localize. If the robot cannot communicate with the off-board device, locomotion should also end. The robot cannot report errors or sensor information to the off-board device for planning and scheduling, which then risks incorrect drawing and motion.
**Criticality:** Loss of connection is high-risk with regard to completing the drawing task. Requirements do not specify the ability to recover a connection, so processing and drawing will end on signal loss.
**Safety Hazards:** The only risk is the robots going out of bounds, or colliding with each other. Both of these pose little hazard to bystanders, as the robots are designed to be safe in the event of human-robot collision (Sec. 11).

### 1.6.5 Failure Mode: Incorrect Data

**Description:** This failure mode occurs when the robot receives bad data from the off-board device, or when the off-board device receives bad data from a robot agent. Bad data here refers to data that cannot be parsed by either end.
**Cause:** Garbage data could be the result of a low-quality connection with high noise, or if data being

sent becomes corrupted. It could also occur due to controller inability to parse the data being sent.
**Effects:** Invalid and incorrect commands and information should be ignored by the robot agent or the off-board processor.
**Criticality:** Incorrect data is noncritical to task completion as incorrect commands will be reacted to accordingly and resent. For example, if the off-board device sends a locomotion command to a robot, but the command becomes corrupt. The localization system will see the robot not move, and attempt to send a similar motion command again.
**Safety Hazards:** There is no risk to unparseable data being sent between robot and off-board device.

## 1.7 User Interface

### 1.7.1 Performance Test: Emergency Stop Speed

**Test Question:** How fast does the emergency stop shut down the system?
**Operational Procedure:** While the system is in use, press the emergency stop button and time how long it takes for everything to completely shut down.
**Metric:** Elapsed time.
**Acceptance Criteria:** It is vital to safety that our emergency stop shuts everything down within a second.
**Requirement(s) Verified:** NFR 11, FR 13

### 1.7.2 Performance Test: Error Reporting Delay

**Test Question:** What is the delay between an error occurring and that error being reported to the user?
**Operational Procedure:** Given a list of known operational errors, intentionally trigger each error within the system and report the time between causing the error and it being reported to the user.
**Metric:** Averaged elapsed time across error reporting.
**Acceptance Criteria:** The average time to detect and report an error should be within 3 seconds.
**Requirement(s) Verified:** NFR 11, NFR 2

### 1.7.3 Performance Test: Error Understandability

**Test Question:** How understandable and informative or are the user errors?
**Operational Procedure:** Given a list of known operational errors, intentionally trigger each error while a non-developer user is using the system (while masking the error cause) and evaluate how well the user can determine the error. For example, while the system is drawing the user could be in a different room with only the error reporting device, making the user unable to see what errors the robots are facing.
**Metric:** Determine if the user can determine the error and knows how to react to or correct the error.
**Acceptance Criteria:** The user should be able to determine and react effectively for 90% of the errors.
**Requirement(s) Verified:** NFR 2, NFR 1, FR 14

### 1.7.4 Functional Test: Emergency Stop

**Test Question:** Does the emergency stop fully stop the system?
**Operational Procedure:** While the system is in use, press the emergency stop button and check if all systems halt their operation.
**Metric:** Boolean on whether every subsystem stops or not.
**Acceptance Criteria:** It is only successful if the boolean metric is true.
**Requirement(s) Verified:** FR 13

### 1.7.5 Functional Test: Error Reporting

**Test Question:** Is each operational error reported to the user?
**Operational Procedure:** Given a list of known operational errors, intentionally trigger each error within the system and report whether the error caused it reported to the user.
**Metric:** Each error must be reported correctly. Hence we can divide the number of correctly reported errors by the number of total errors caused to determine an error-reporting score.

**Acceptance Criteria:** Considering error handling is critical to performance, our system should have an error-reporting score of 90%.
**Requirement(s) Verified:** NFR 2, NFR 1, FR 14

#### 1.7.6 Failure Mode: UI Navigation

**Description:** This failure occurs when a user is unable to navigate the UI to setup and begin the autonomous drawing process.
**Cause:** Causes of this effect could be an unintuitive user interface, a UI that lacks features necessary to run the system, or lack of user training to use the interface properly.
**Effects:** The only effect is that the system is unable to begin the drawing process.
**Criticality:** UI failure is noncritical to system operation, as the system can run without a graphical interface. However, it is critical for demo purposes as a demo user must be able to begin system operation.
**Safety Hazards:** No safety hazards are posed by this failure mode.

### 1.8 Power System

#### 1.8.1 Performance Test: Battery Duration

**Test Question:** How long can an individual robot run for on a single battery charge?
**Operational Procedure:** Charge a robot fully. Given some example drawing inputs, continue to input drawings until the robot is fully drained of power. Time how long this takes.
**Metric:** The duration of operational time given one charge
**Acceptance Criteria:** We accept this if the operational time exceeds the necessary duration time of 90% of our test drawing inputs.
**Requirement(s) Verified:** FR 15

#### 1.8.2 Functional Test: Battery Life

**Test Question:** Can the robots complete a drawing from a single charge?
**Operational Procedure:** Given a set of example drawing inputs, we want to test the robots ability. For each input, fully charge each robot, send the input and keep track of whether the drawing is fully complete before the battery on either robot is fully drained.
**Metric:** The ratio of the number of completed drawings to the total number of drawings.
**Acceptance Criteria:** We want to be able to successfully draw 90% of the drawings in our example drawing input set.
**Requirement(s) Verified:** FR 15

#### 1.8.3 Failure Mode: Insufficient Battery

**Description:** This failure mode arises when the battery for an individual robot agent is low or out of power.
**Cause:** Insufficient battery power is a result of overuse of the battery from autonomously drawing for too long.
**Effect:** The result is a robot agent being unable to move, draw, or communicate with the offboard system, as it has no subsystems being powered.
**Criticality:** This failure is critical, as it can pause and/or end robot operation. Robot agent battery must be replaced before operation can continue.
**Safety Hazards:** There are no safety hazards that result from the battery being low or out of power.

## 2 Full System Validation

Test: do the drive commands respect physical robot capabilities. This relates to whether they robot can drive/draw at the same time w/o marker getting stuck. Also, is there a speed threshold at which drawing quality degrades - involves writing implement, locomotion

Test: does an individual robot draw accurately relative to the commanded motion. This involves making sure line aren't jagged/squiggly, etc. come up with better words for this

Test: does the robot fulfill size requirements. CHECK REQ SPEC FOR VALUES

Test: does the robot fulfill weight requirements. CHECK REQ SPEC FOR VALUES

Test: is the robot safe, aka does it not have pointy (sharp) edges on the outside

Test/verification: Was robot system price within budget metric: dollars acceptance criteria: 2500 dollars

**NJ: Fail: out of bounds - uses localization, locomotion NJ: Fail: robot makes incorrect mark on ground - uses locomotion, localization, writing implement NJ: Fail: collision between robots - uses locomotion, localization**

### 2.0.1 Failure Mode: Out of Bounds

**Description:** This failure describes the situation when any robot agents move beyond the bounds of the drawing surface, as described by the boundary vision markers.
**Cause:** This error can be caused by either poor localization, or poor locomotion. If localization software believes the robot to be somewhere it is not, it may command the robot out of bounds. If the robot motors or wheels are not working properly, it may move out of bounds, despite being given correct motion commands.
**Effect:** The robot moving out of bounds introduces undefined behavior that could result in collision, incorrect drawings, or making marks with the writing tool that are not on the appropriate writing surface.
**Criticality:** This is a critical error, as it results in incorrect localization, drawing marks, and system operation. Entering this failure mode results in system operation ending.
**Safety Hazards:** This failure poses a safety hazard of collision, as robots that exist the bounds may collide with objects or people that it is not expecting.

### 2.0.2 Failure Mode: Incorrect Markings

**Description:** Incorrect markings are made when a robot agent lowers the writing implement, and makes a mark in a location that does not match with the input drawing.
**Cause:** The cause of incorrect markings could be a result of the writing implement, locomotion, or localization subsystems. The writing implement subsystem may malfunction and lower the tool at an incorrect time. The locomotion subsystem may move the robot incorrectly while the writing implement is lowered, making markings where they are not expected. Localization error can cause markings to be in places that the system thinks are correct, but do not line up with the input drawing.
**Effect:** The effect of this failure is that the output drawing by the system is incorrect.
**Criticality:** Given that the goal of this robot subsystem is to accurately recreate an input drawing, failure to do so is a critical error.
**Safety Hazards:** There are no safety hazards associated with incorrectly marking the drawing surface.

### 2.0.3 Failure Mode: Robot Collision

**Description:** This failure exists when robot agents collide with any obstacle, including each other or external obstacles.
**Cause:** Robot collision is a result of the locomotion, localization, or work scheduling subsystems failing. Locomotion failure could cause undefined motions by a robot agent, causing it to hit another robot, or move out of bounds (Sec. **??**) and collide with an obstacle. Localization failure could result in incorrect motion commands, resulting in collision with unintended objects. Finally, poor motion planning has potential to require the robots to move into collision with each other.
**Effect:** Robot collision could damage the robot agents, or hurt human observers who are hit.
**Criticality:** This failure is of medium importance. The robots are designed to be safe (Sec. 11), and therefore are unlikely to hurt or be significantly damaged by a collision.
**Safety Hazards:** There is a hazard of minor human injury, but as stated above the robots are designed to minimize human injury in the case of collision.

# 3 Risk Management

## 3.1 Battery Explosion

Battery explores

## 3.2 Intruder Collision

Robots hit intruders

## 3.3 Camera Collision

Camera fall

## 3.4 Finger Jam

User's fingers somehow get stuck in rotating wheels when install and uninstall tools or get stuck in motors for lifting chalk

# 4 Requirements Traceability Matrix

| Requirement | Test |
|---|---|
| FR 1 | T?? |
| FR 2 | T?? |
| FR 3 | T?? |
| FR 4 | T1.3.1, T1.3.2, T1.3.3, T1.3.4, test:image$_{f}t_{b}ounds$ |
| FR 5 | T?? |
| FR 6 | T?? |
| FR 7 | T1.1.3 |
| FR 8 | T?? |
| FR 9 | T?? |
| FR 10 | T1.1.4 |
| FR 11 | T1.4.2, T1.4.3, T1.4.4 |
| FR 12 | T?? |
| FR 13 | T?? |
| FR 14 | T?? |
| FR 15 | T?? |
| NFR 1 | T?? |
| NFR 2 | T?? |
| NFR 3 | T?? |
| NFR 4 | T?? |
| NFR 5 | T1.2.2 |
| NFR 6 | T1.1.2, T1.1.5, T1.1.6, T1.1.7, T1.3.1, T1.3.3, T1.3.4, T1.4.1 |
| NFR 7 | T?? |
| NFR 8 | T?? |
| NFR 9 | T?? |
| NFR 10 | T?? |
| NFR 11 | T?? |
| NFR 12 | T1.2.1 |
| NFR 13 | T?? |
| NFR 14 | T1.1.1 |