# Carnegie Mellon University

## Robotics Capstone Project

# Concept Evaluation

*Friction Force Explorers:*

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

supervised by
Dr. David Wettergreen

Version 1
October 27, 2016

# Contents

# List of Figures

# 1  System Description

The goal of our project is to develop a multi-agent robotic painting system. Use a team of homogenous robots, we can efficiently parallelize the dull and often dirty task of painting. Our system is motivated by the miles of paint needs to maintain runways and sport areas as well as the widespread use of sidewalk chalk. Below, in Fig.1, we model our overall system graphical, with a more detailed description following. We then enumerate each subsystem in Sec. 5.
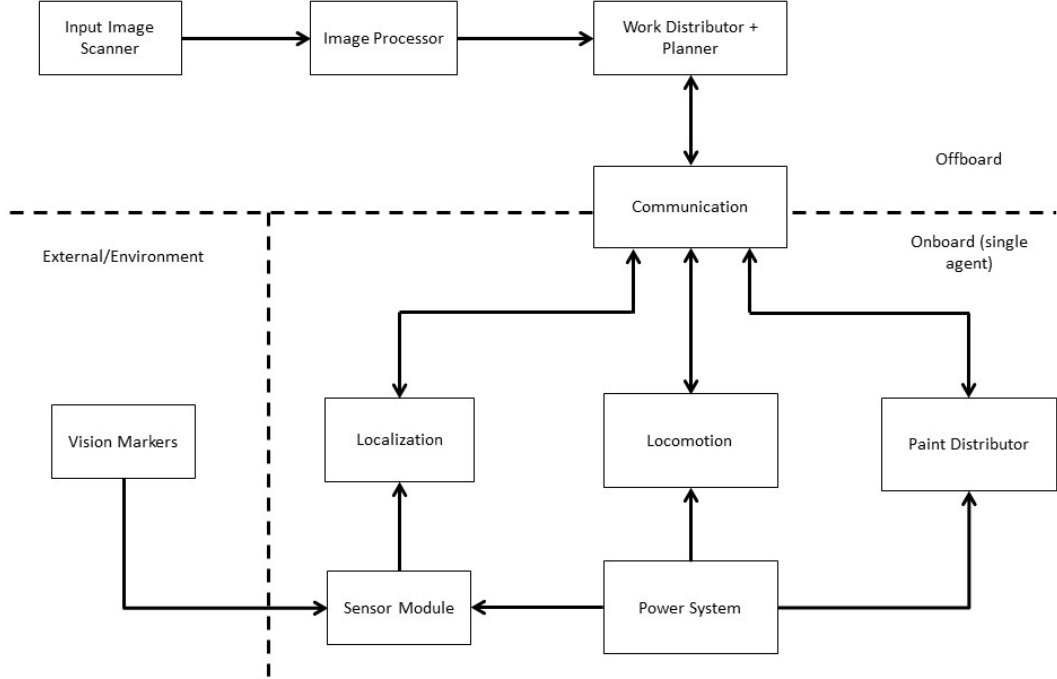


Figure 1: High Level State Machine to Model our System

Our system encapsulates three components, shown in Fig.1 through dotted lines. First, there is a single offboard unit that hands most of the coordination, planning and user interface. The initial drawing process begins in the offboard unit, where users will generate an input image and process it through a scanner. Image processing will then occur, where the input is processed into a format compatible by the planner and work distribution subsystem Sec. 5.6. Moving forward, the planner module will take in updates to current robot progress and use it to update commands it sends to robot agents.

Our second component is the robotic agents, marked in Fig.1 as "onboard". While the diagram shows one onboard agent, this schema is the same for each agent. Commands are sednt to these agents. Communication protocols are able to send information both to and from the work planner to robot agents (Sec. 5.7). From here, onboard systems are separated into three subsystems: Localization (Sec. 5.3), Locomotion (Sec. 5.2), and Paint Distribution (Sec. 5.1). The paint distributor is involved with all commands regarding engaging and disengaging the writing implement to output paint onto the writing surface. Locomotion is the drive system, which engages wheels and motors used for movement. This subsystem also includes any motion-related safety mechanisms, such as quickly stopping. The Localization subsystem works to determine the position and orientation of an individual robot agent. This subsystem takes input from the sensor module (Sec. 5.9), which provides data about the surroundings for the localization system to process.

The sensor module gets data from external vision markers (Sec. 5.11) for processing by localization. It is important to note that the sensor module, along with the locomotion and paint distribution subsystems are all powered by an individual subsystem Sec. 5.10.

Finally our third component comes from environmental setup, which is mainly the situational landmark tools need to localize the robot.
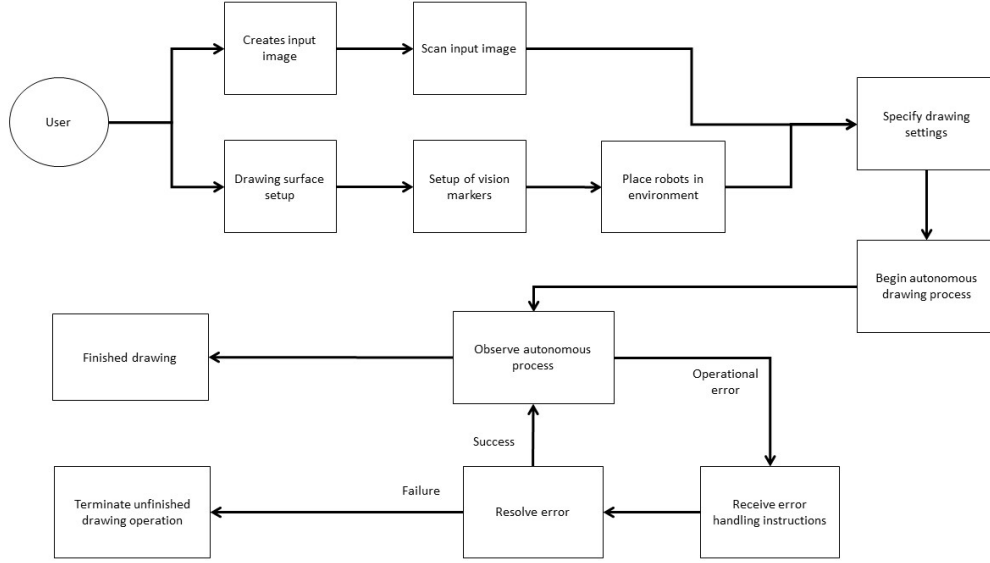
## 2 Concept Operation



Figure 2: State Machine of Concept of Operations

Above, in Fig.2 we outline the user experience with the system. This is detailed below and our user interface is fully explained in Sec. 5.8.

The initial setup contains two operations that can be performed simultaneously: adding the image to be drawn to the system, and setup of the drawing surface. Adding the input image involves generating the image, and then scanning it into the system. This satisfies requirements for input the drawing plan (Requirements Specification 5.1, FR9), and a user interface (Requirements Specification 5.1, FR12). Setup of the system involves placing the drawing surface, placing and calibrating vision markers, and finally placing the robot agents within the bounds of the drawing surface. Once both steps are done, the user can enter any required settings for their use, and begin operation. Processing of the input image is done automatically by the system and is invisible to the user (Sec. 5.5).

Once the autonomous drawing process begins, the user simply observes the robots complete the task. However, in the case of problems, error(s) will be reported to the user, who then has the option of fixing the issue to continue operation, or terminating the drawing process. The autonomous process satisfies functional requirement FR2 (Requirements Specification 5.1, FR2).

## 3 Use Cases

### 3.1 Load/Switch Writing Implement

**Summary:** A robot's main consumable, the writing implement, must be reloaded or replaced when empty or when a different implement (with different color stroke, or other properties) is desired.
**Actors:** Writing implement, human
**Precondition:** No writing tool in the system or the one that is currently loaded is no longer needed for the task.
**Post-condition:** System has desired writing tool loaded.
**Alternative:** If the robot is unable to load the implement then the system should report improper loading results to the user. Additionally, other robot agents should be informed that the robot to be reloaded is broken, and therefore cannot draw. Hence, in order to complete the picture the other robots must re-plan accordingly.
**Description:** When the robot is working on a large or intricate drawing that requires a large amount of consumable writing material, the robot may not be able to carry enough material to complete its allocation of the drawing. Additionally, some drawings may involve a variety of colors, strokes, or other properties that necessitate the use of different writing materials. Thus, the robot must be able to replace
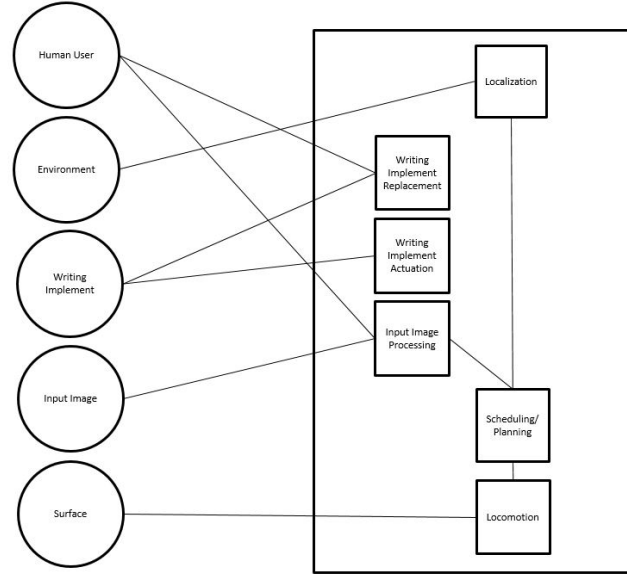
Figure 3: Diagram representing the relationship between use cases and actors in the system

its writing implement with human assistance. By using a modular writing implement actuator, the robot can use a variety of different writing utensils - for example, by switching out the actuator the robot can go from being equipped to use chalk to being equipped to use permanent marker. The user can simply pull the previous writing implement out of the actuator, and insert the new implement, which is held in place by friction. The robot must be able to recognize reloading failures and alert the human operator of their occurrences through the user interface (Sec. **??**). If the failure is not corrected, The robot must communicinpate its inability to perform to other robots in the system so they can re-plan drawing paths accordingly. For further details, see the Writing Implement Actuator subsystem description (Sec. 5.1).

## 3.2  Process Input Image

**Summary:** The system must take in a human-provided image and interpret what the robot system is required to draw.
**Actors:** Image, Human
**Precondition:** No existing image being actively drawn by the robot system.
**Post-condition:** The image is processed and ready for work distribution between robots.
**Alternative:** If this step fails the robotic system should report an image processing failure. The user must then address the error before the system can continue work.
**Description:** The system's task will be input using a human-produced image following specific guidelines. From the image, the system can determine where to place markings in the real world, and how the work should be distributed amongst the robot workers. Parameters such as the scale and orientation of the final drawing are also specified at this stage. For further details, see the Input Image Scanner and Image Processor subsystem descriptions (Sec. 5.4 and Sec. 5.5 respectively).

## 3.3  Localization

**Summary:** The robot must determine where it is in the working area.
**Actors:** Environment
**Precondition:** The robot is uncertain as to its position and orientation.
**Post-condition:** Following localization we hope to have minimal uncertainty about the current location and orientation of the robot.
**Alternative:** If the robot cannot localize then this should be reported to human user and the other robots should be alerted of the robot's inability to localize and orient itself. Additionally, any robot that cannot localize can potentially draw incorrectly or collide with another component of the system. Thus

any robot that fails to localize should halt all movement until localization capability is reestablished.

**Description:** In order to create an accurate reproduction of the input image, the robot needs to know how its location maps to a location on the input image. The system will use a beacon/marker system to determine a robot's position and orientation. If it is unable to do so with a particular robot, that robot cannot continue drawing and the system must alert other robots to the fact so that they can re-plan and reschedule the workload. For further details, see the Localization subsystem description (Sec. 5.3).

## 3.4  Scheduling and Robot Planning/Coordination

**Summary:** The robot workers must determine an efficient allocation of the work.
**Actors:** None
**Precondition:** No plan or schedule currently exists.
**Post-condition:** Each robot has an allocation of work and a planned path.
**Alternative:** If planning fails, the human user is alerted and the operation is aborted. The user must address the error before operation can continue.
**Description:** The work required must be determined from the input image, and analyzed to determine an efficient allocation of work, as well as a path schedule for each of the robots. Additionally, robots must communicate work completed and failure states to each other in case re-planning is required. For further details, see the Scheduler/Planner subsystem description (Sec. 5.6).

## 3.5  Move Robot

**Summary:** The robot moves across flat terrain.
**Actors:** Ground
**Precondition:** Robot is stationary.
**Post-condition:** Robot is moving as commanded.
**Alternative:** If the robot is unable to move, the the human user is alerted and work is redistributed between remaining robots.
**Summary:** The robot moves across the writing surface. By using a holonomic drive mechanism, the robot can make sharp turns and intricate designs. For further details, see the Locomotion subsystem description (Sec. 5.2).

## 3.6  Use Writing Implement

**Summary:** The robot creates a mark on the writing surface.
**Actors:** Writing surface, Writing implement
**Precondition:** The robot has a plan to draw an mark, but that mark has not currently be drawn yet.
**Post-condition:** The robot's planned mark is drawn on the writing surface.
**Alternative:** If the robot cannot write, the robot alerts user that the writing implement must be replaced (or inserted if one does not exist).
**Summary:** The robot creates markings on the surface with the given writing implement, and ensures that its movements do not disrupt drawing accuracy. For further details, see the Writing Implement Actuator subsystem description (Sec. 5.1).

# 4  Artistic Sketch of System
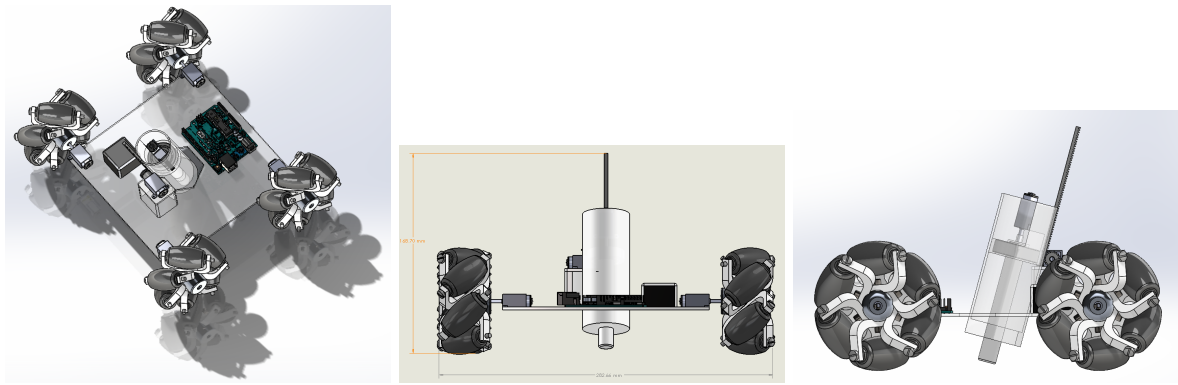
Figure 4: Top (left), front (center), and side (right) views of a individual robot worker.



Figure 5: Uncovered (left) and partially covered (right) views of the bearing that turns the chalk during writing.

Figure 6: The translational motor (left) and rotational motor (right) of the writing implement actuator.

Figure 7: Various views of the whole writing implement actuator.



Figure 8: Close-up view of the writing implement actuator as part of the whole assembly.

# 5 Subsystem Descriptions

## 5.1 Writing Implement Actuator

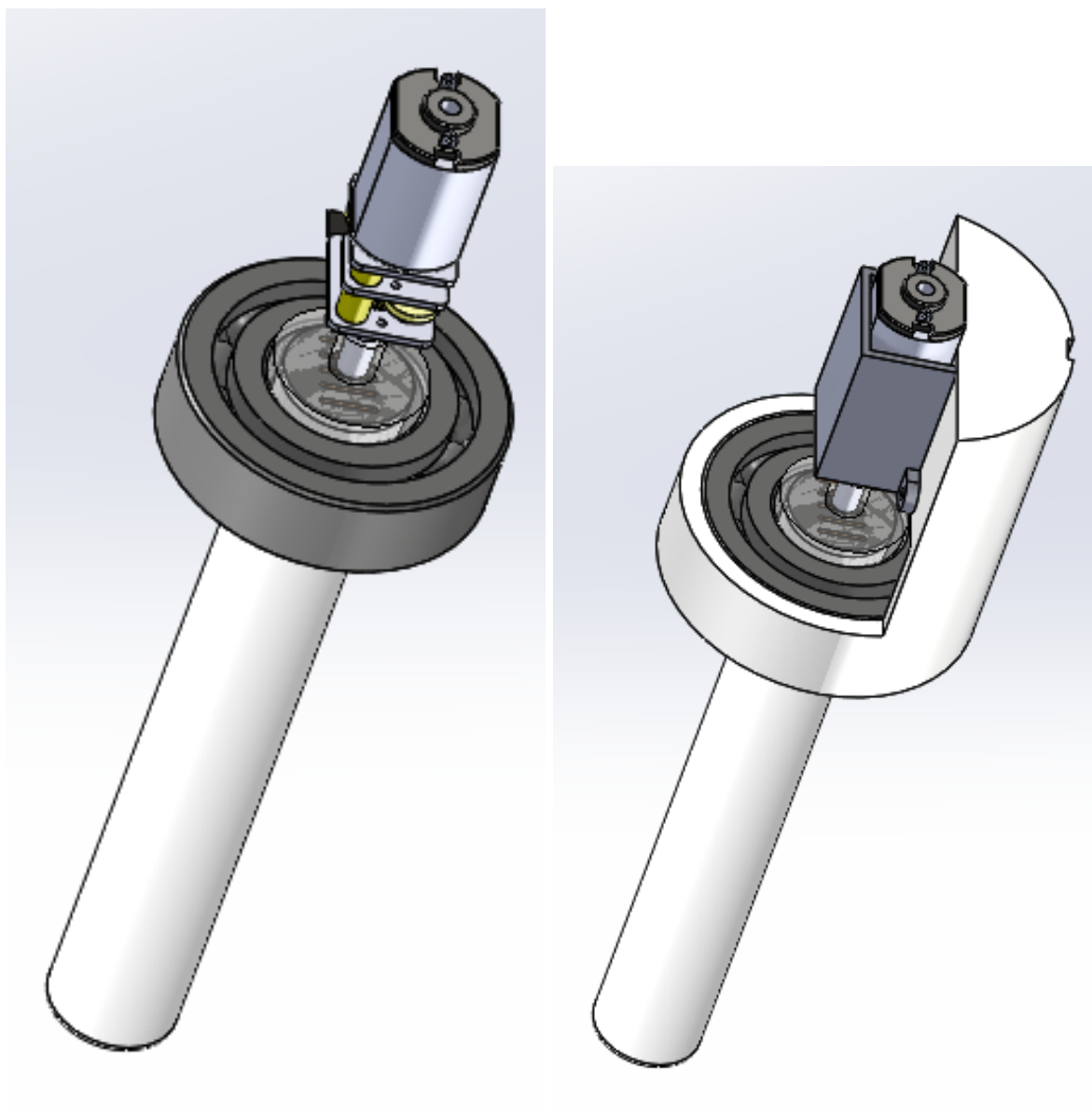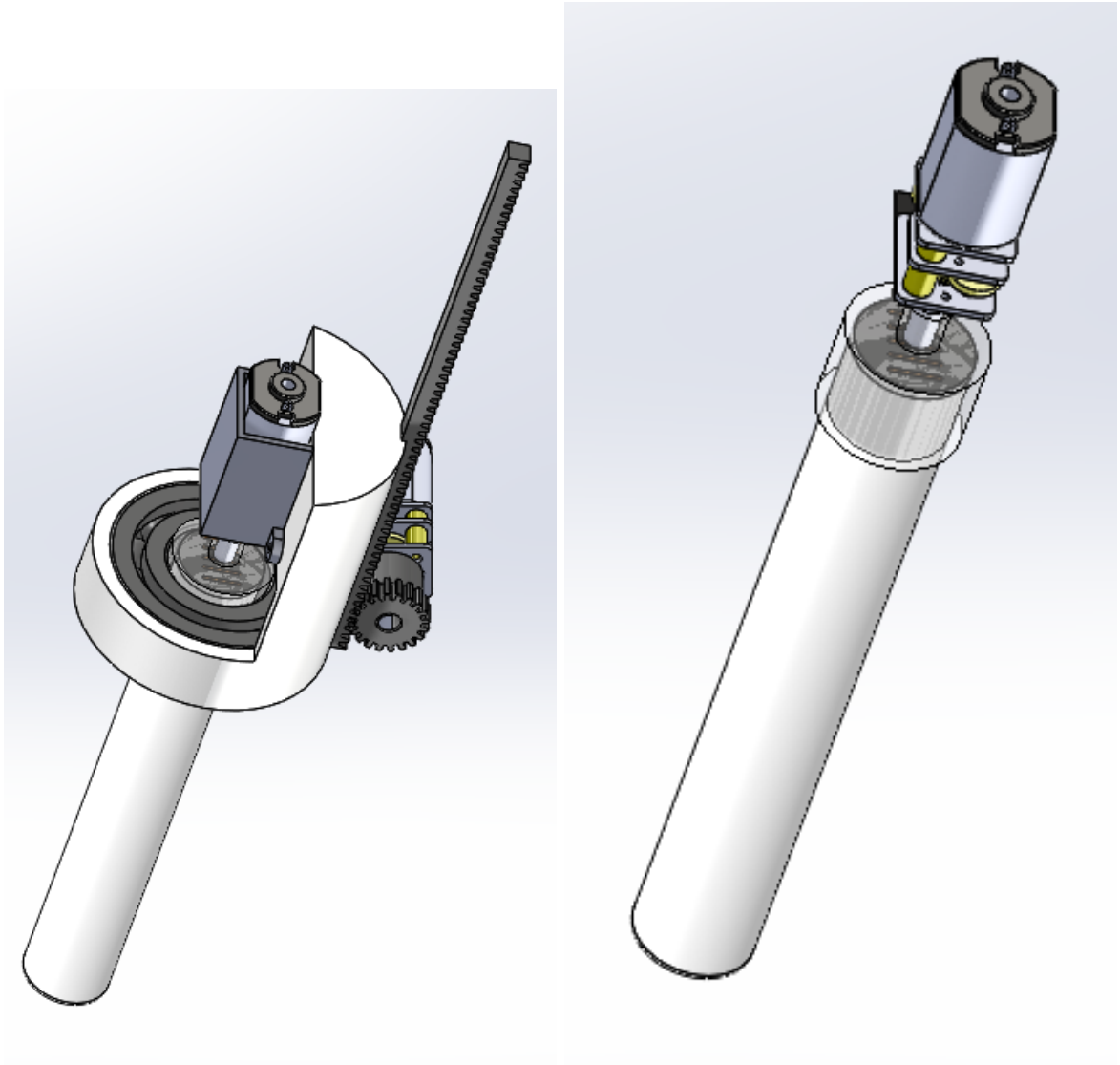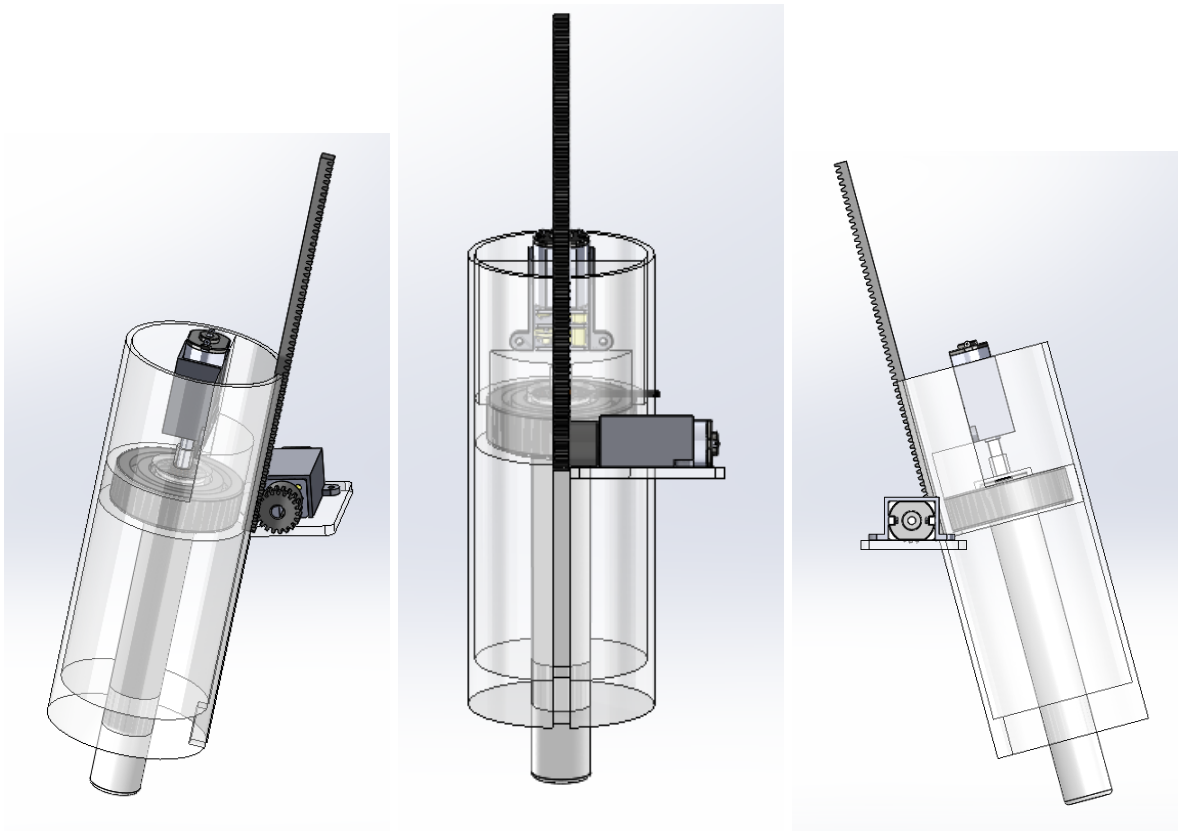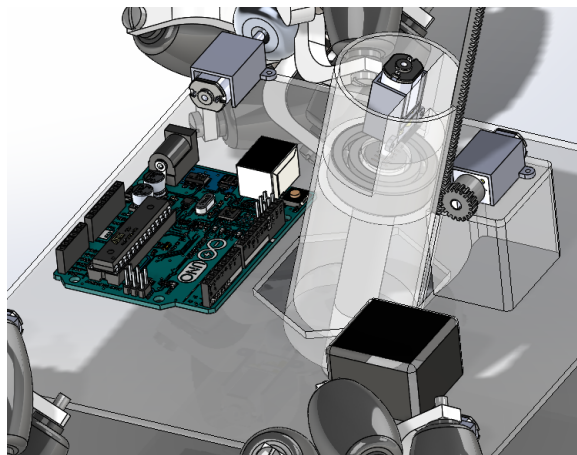**Description:** The writing implement actuator deposits writing material onto the working surface. It manages reloading of the writing implement and ensures that the implement is properly secured. Additionally, it can extend writing material closer to the ground as the material is shortened through use, and it can rotate the writing material for variation in stroke. In the event of writing material depletion, its sensors will detect the occurrence and alert the communication module (Sec. 5.7). This subsystem satisfies writing tool related requirements (Requirements Specification, 5.1, FR6, FR8).
The writing implement system is modular, in that various writing tools (such as chalk, markers, pens, etc.) can be easily inserted by users. This involves a fixed motor mechanism on the robot, with mounts for each tool that can be locked into place on the robot. The mounts can optionally connect to two motors: One for linear motion to raise and lower the writing implement, and another to rotate the writing material as described above.
**Critical Components:** Writing implement holder, actuation mechanism, writing material levels sensor, reloading mechanism.

## 5.2 Locomotion

**Description:** The robot's locomotion system propels each individual robot across the working surface. If the robots use a holonomic locomotion systems, they are able to move in any direction, ensuring their ability to execute intricate designs. They also provide enough clearance for the writing implement to work effectively. In the case of a tread-based locomotion system, individual robots can rotate in place to move in all directions. A driving system that allows for multidirectional movement satisfies motion-related requirements (Requirements Specification, 5.1, FR7). Additionally, the locomotion system contains encoders whose data is used by the localization module (Sec. 5.3).
**Critical Components:** Wheels or treads, motors, encoders.

## 5.3 Localization

**Description:** The localization system uses a hybrid infared and ultrasonic marker-based method (Sec. 6.1) to determine a robot's position and orientation. By using the delay between the infared and ultrasonic pulses detected by the sensor module (Sec. 5.9) it computes the its distance to each marker. After performing some trigonometry, it can then accurately determine position. Orientation is determined by dead reckoning through the wheel encoders, and is constantly corrected by measuring changes in position. It then communicates the positions and orientations of the robots to the scheduling module (Sec. 5.6). This module directly satisfies local and global localization requirements, as well as indirectly allows for safe motion from the robots (Requirements Specification, 5.1, FR3, FR4).
**Critical Components:** Localization algorithm, sensor module, vision markers.

## 5.4 Input Image Scanner

**Description:** The image scanner takes a user-provided image or generates one, and transfers it to the image processing module (Sec. 5.5). The image scanner, being a module requiring user interaction, will also incorporate a front-end user interface (Sec. 5.8). This subsystem satisfies requirements for adding a drawing plan (Requirements Specification, 5.1, FR9).
**Critical Components:** User interface, image sensor.

## 5.5 Image Processor

**Description:** The image processor takes input from the image scanner (Sec. 5.4) and produces information that is recognizable by the planning module (Sec. 5.6).
**Critical Components:** Image processing algorithm.

## 5.6 Work Scheduling, Distribution and Planning

**Description:** Given the output from the image processor (Sec. 5.5), and parameters such as the number of robots and the size of the workspace, the module determines the work that will be distributed to each robot. See software architecture (Sec. 7) for more information. Planning between robots allows for coordination and efficiency, satisfying requirements for both (Requirements Specification, 5.2, NFR2, NFR7).
**Critical Components:** Scheduling and distribution algorithm.

## 5.7 Communication

**Description:** The communication module is the link between the offboard system and the individual robots. To facilitate real-time changes in the working schedule, communication will be speedy and reliable. Potential communication protocols include WiFi and Bluetooth. Communication between the offboard system will allow individual robots to know their progress relative to the entire drawing (Requirements Specification, 5.1, FR10).
**Critical Components:** Antennae, wireless communication protocol.

## 5.8 User Interface

**Description:** The user interface provides a unified system for user input. This is the system with which the user specifies the input image and monitors the robots' progress. The system will also display any error messages or anomalies detected by the system, and how the user should address them. There will also be a system-wide kill switch in case of emergencies to satisfy requirement FR11 (Requirements Specification, 5.1, FR11).
**Critical Components:** Screen, input device.

## 5.9 Sensor Module

**Description:** The sensor module gathers data from environmental markers (Sec. 5.11). This data is used by the localization module (Sec. 5.3) to determine an individual robot's position and orientation in the workspace.
**Critical Components:** Environmental beacons, onboard sensors.

## 5.10 Power System

**Description:** The power system supplies power to the rest of the system. Each robot has an onboard battery that is small and light enough to satisfy the size and weight requirement, but also provides enough uptime to last a entire drawing session. The power system will satisfy battery life and contribute to portability requirements (Requirements Specification, 5.2, NFR1, NFR6).
**Critical Components:** Battery, voltage regulator modules.

## 5.11 Environmental Markers

**Description:** The environmental markers must be set up by the user before the system can begin drawing. The continuously output pings using both an infared light and ultrasonic transmitter. In doing so, they provide the data needed by the localization subsystem (Sec. 5.3) to determine the robot's position and orientation in the workspace. They will be mounted high enough as to be visible by every robot in the workspace at all times.
**Critical Components:** Beacons/markers, elevated stands.

# 6　Trade Studies

In creating our multi-agent drawing robot, we face design trade-offs and decisions for many of our major subsystems detailed in Sec. 5. We investigate three major design decisions in the following brief trade studies. Our discussions lead to a few technical solutions, when we feel we have enough data and strong reasoning to back them up. However, for some of our discussion points, further prototyping and testing are needed to fully evaluate all options. For these points we lay out brief prototyping plans below.

## 6.1　Localization Method

In multi-agent planning, it is important to accurately localize robots' positions and orientations. Keeping in mind limitations in price and ease of use, we come up with two major methods for localization: vision-based and marker based. They are described below.

Vision-based localization involves using cameras or other visual sensors to directly obtain information of the environment and localize the robots in the system based on found landmarks in that environment. One example of this is SLAM (Simultaneous Localization and Mapping), often used by autonomous vehicles to simultaneously build maps and localize  [1]. With this approach, robots could build small maps of their surroundings and match their locations to features they find in the environment. Benefits of this method include being location agnostic and requiring no additional parts or external setup. However, pure vision systems are difficult to calibrate and localization accuracy can depend heavily on static surroundings, which is not something this system can guarantee.

The other choice of methodology is marker-based. Using markers placed around the drawing surface, robot agents can quickly locate these markers and their positions relative to each marker, and consequently triangulate their positions and orientations. While requiring additional setup and more parts to calibrate than vision-based localization, existing technology makes it convenient and cheap to get marker-based localization working quickly. One example of a marker-based localization system is AprilTags [2], which can be described as 2D barcodes placed in a scene. Marker-based localization can be further classified into two subcategories: passive and active. Passive markers do not output any information and exist for the robot agent to observe and triangulate accordingly. AprilTags is an example of the passive marker system. On the other hand, active markers will look at robot agents to determine where the agents are, rather than the robots searching for markers. While less common, active systems behave well in conditions when the markers may not always be easily visible to robot agents [3].

Given the convenience and ease of use of marker-based localization, it is clearly the better choice. However, it is more difficult to determine whether using active or passive systems will be more effective. A prototype for each system should be made for further evaluation.

## 6.2　Locomotion Method

One of our functional requirements with the highest priority (FR1) is to be able to move in specified directions with a high degree of accuracy. Additionally, a medium high priority functional requirement (FR 9) is to have a drive control system that enables this movement. Therefore, it is clear from our requirements that locomotion is critical to our robot's performance. When considering locomotion options there are three large categories: aerial, legged and wheeled.

Our drawing occurs on the two dimensional plane. Therefore, any flight based system would have to constrain one translational degree of freedom and two rotational degrees of freedom. Doing so would require precise control algorithm and large power input. Practically speaking, this seems to over-complicate the problem with little additional gain.

Therefore, we are narrowed down to a legged or wheeled system. While a legged system would have the ability to traverse uneven terrain, such capability is not necessary for our project due to our assumption (A3) that the drawing surface is flat and homogenous. Furthermore, compared to wheeled systems, legged systems usually involve more complicated control algorithms and more complex electrical and mechanical components. We are inspired by the wide spread success of wheeled robotic systems and especially appreciate their stability, a critical concern when carefully drawing images.

Therefore, we have concluded that our agent's drive system should be wheeled. In our next analysis, we will further investigate different types of drive train.

## 6.3   Drive System

In continuing to prioritize our functional requirement (FR1) of being able to accurately move in specified directions, we further investigate possible drive systems.

Three similar drive systems first come to mind: differential drive, ackerman steering and four-wheel steering. Differential drive on a robot typically consists of two independently driven wheels and a non-driven wheel. Moving up in complexity and taking after some cars is ackerman steering, where the back pair of wheels is fixed in orientation but the front pair can pivot. In more complex is four-wheel steering, which is similar to ackerman steering but allows the back wheels to also pivot. However, these systems suffer from the fundamental issue of being nonholonomic. Therefore, while they could achieve full mobility, this might complicate drawing and path planning.

Therefore instead we want an omnidirectional base that is holonomic. This can be achieved with omniwheels, perhaps more commonly, with mecanum wheels.

# 7 Software Architecture

The software architecture can be separated into three sections: image processing, work planning and distribution, and communication between robot onboard systems and the central offboard system. These systems are described below.
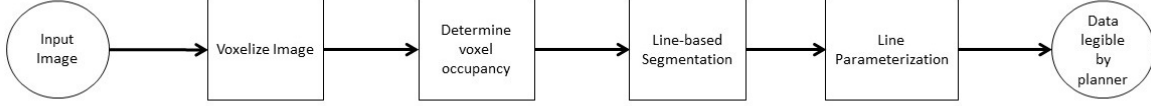
## 7.1 Image Processing



Figure 9: Image Processing Software Pipeline

The image processing pipeline (Fig.9) takes the image scanned by the user, and parses it into a form readable by the work planner. The goal of this software system is to create a series of lines that describes the image. To do this, the system first creates an occupancy grid of the input via voxelization. For a black and white image, the occupancy grid determines black or white. In the case of color, voxels are assigned color based on the image values inside of the voxel.

Once voxelization is complete, lines are formed from the voxels through a nearest-neighbor search. These lines are simply a series of voxel squares that pass through the image. In order to preserve curvature of lines from the input, the grid-delineated lines are reparamterized with splines into paths. These paths are then sent to the work planning module.
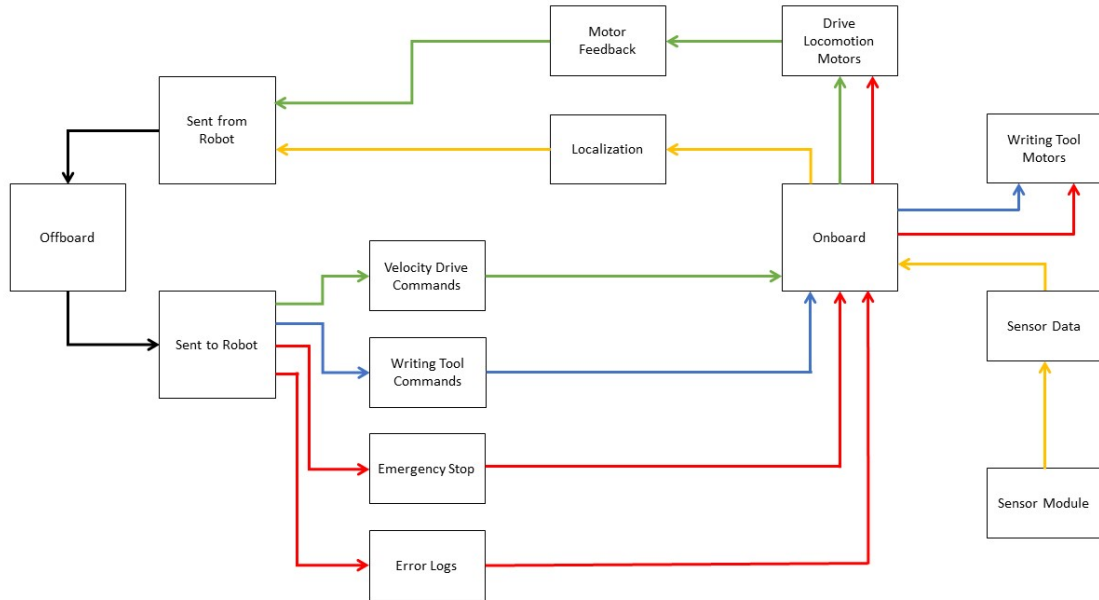
## 7.2 Communication



Figure 10: Communication Software Pipeline

The communication diagram (Fig.10) describes how an individual robot communicates with the offboard path planning system. The system can be described by communication in four categories: motion commands, writing tool commands, error handling, and localization.

The offboard system will use the planning algorithm to determine motion plans for the robot systems. It determines and then sends velocity commands to the onboard system. The onboard system passes these commands to the drive motors for motion.

To command the writing tool, the offboard system also decides how to move the writing implement based on localization of the robot systems. Similar to velocity commands, the onboard system receives commands for the writing tool, and commands the system accordingly.

Localization receives input from the sensor module, and computes position and orientation. This data is then sent back to the central offboard module to update planning and scheduling.

Error logs are generated by the offboard system based on localization and planning, and sent to each robot's onboard system for appropriate reaction. This includes the emergency stop, which will immediately shut down writing tool and drive motors.

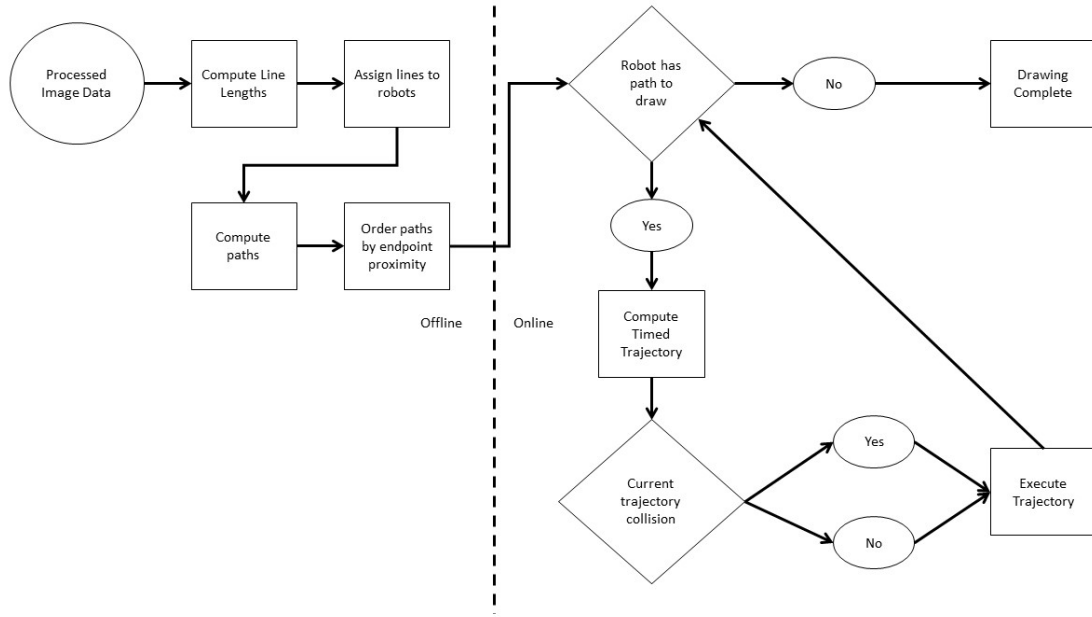## 7.3 Work Scheduling, Distribution and Planning



Figure 11: Planning and Scheduling Software Pipeline

A key aspect of our software system is distributing and planning the work to the robotic agents, modelled in Fig.11. Two of our nonfunctional requirements are to be efficient and have the robots coordinate with each other (NFR4, NFR9). This two objectives inform our planning pipeline. We split the planning and coordination into two separate problems, allowing us to frame coordination as a scheduling problem [4]. Hence our work distribution and planning can be handled offline while our coordinate occurs online.

From our image processing Sec. 7.1, we recieve processed image data. Using this data, we compute the length of every individual line. Guided by the assumption that line length correlates to amount of work done and the time to perform that work, we then use those lengths to load balance when assigning which lines should be drawn by each robot. Once each line has been assigned to each robot, each robot has a complete picture of their work and assign an ordering to their lines. To limit wasteful movement, the robots order lines greedily: having completed one lines, they pick their next lines (from their assigned set) based on which line has the closet endpoint to the lines they have just finished. Having ordered lines, we now can determine each robot's set of paths.

Next, we briefly describe a sketch of our coordination mechanism. Each robot has a queue of paths, based o n the set ordering, to execute. Given that a robot has not finished all of it's assigned paths, it removes a path from the queue. We then time that path, with uniform timing, to convert it to a trajectory. Now that we have timing, we can check if the path would collide, or have the possibility of collision, with any trajectories currently being executed. If the trajectory is not at risk of collision, then it can be executed - hence it is sent to the robot via the protocol mentioned in Sec. 7.2.

If the trajectory would be in collision with a currently executing trajectory, then the trajectory being processed defers to the one being executed. Hence we adjust the timing of the processing trajectory to have a pause to avoid collision. Given this modified timing we can then execute it.

Once all paths have been drawn, and hence the queues are empty, we have completed our drawing.

# 8 Installation

The system's setup requires a drawing surface and vision markers under specific conditions. Conditions for setup must be in line with the scope and assumptions made for successful operation of the system. Maintaining scope of the project, as described in the requirements specification (Requirements Specification, 2.3), the drawing surface must be placed indoors and in an obstacle-free area. For compliance with surface requirements (Requirements Specification, 2.4, A1), the surface chosen must be a flat and homogenous surface.

Once the drawing surface has been placed indoors on a flat surface, vision markers must be set up and calibrated. As per Sec. 6.1, localization markers must be placed around the edges of the drawing surface. These markers must then be calibrated by the user, who will input marker locations into the system. The localization markers serve as calibrated base points from which the robot agents will determine their position and orientation.

# References

[1] M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (slam) problem," *IEEE Transactions on robotics and automation*, vol. 17, no. 3, pp. 229–241, 2001.

[2] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *ICRA*, pp. 3400–3407, IEEE, 2011.

[3] R. Cassinis, F. Tampalini, and R. Fedrigotti, "Active markers for outdoor and indoor robot localization," *Proceedings of TAROS*, pp. 27–34, 2005.

[4] P. A. O'Donnell and T. Lozano-Pérez, "Deadlock-free and collision-free coordination of two robot manipulators," in *ICRA*, IEEE, 1989.