Robotic Courier System

Autonomous mobile robot for indoor package delivery.

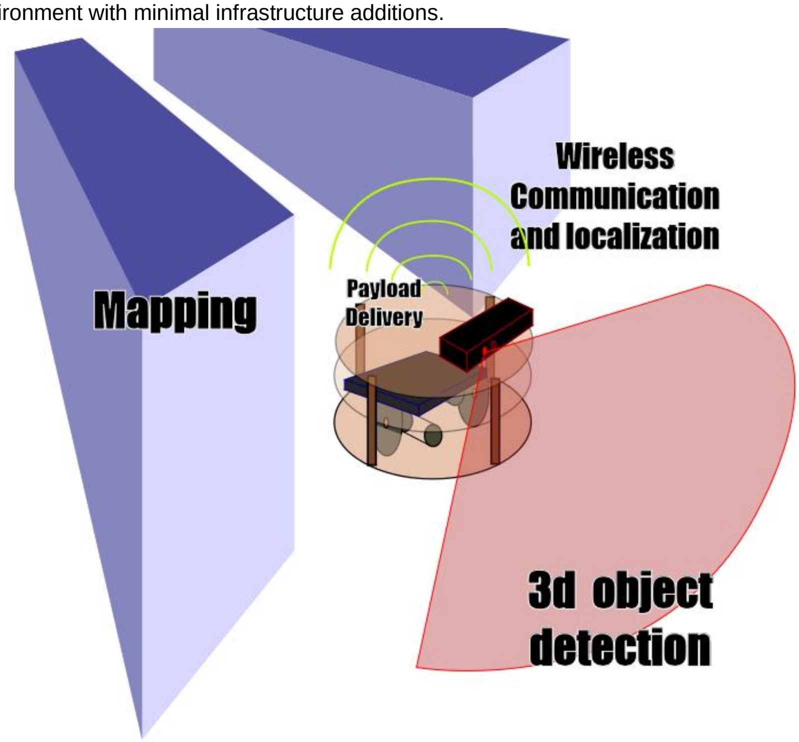
Alex Crawford
Andrew LeCain
Russell Martin
Ryan Cramer

02/22/2012

I. Overview

Needs Statement: Despite the advent of electronic mail, certain materials such as packages and original documents must be delivered physically. While the infrastructure to deliver these packages regionally and internationally is largely automated, the last mile delivery is almost always done by hand. Universities and large corporations employ many couriers for inter- and intra-departmental delivery at great expense. Some large office buildings may have primative interdepartmental mail through a series of pneumatic tubes, but this requires a large infrastructure investment and drastically limits the form factor and weight of the payload, requiring any large or heavy delivery to be made by a person. There is a need for an affordable and robust system to automate the delivery of such messages, packages, and assemblies without incurring this large infrastructure cost.

Objective: Our goal is to develop an automated vehicle that is able to autonomously deliver such payloads over moderate distances indoors, automatically mapping and navigating through its environment with minimal infrastructure additions.



Description: The vehicle will consist of a moving two wheeled base, with each wheel containing a separate motor. A series of sensors will be installed on the the vehicle, including a Microsoft Kinect depth camera sensor, as well as several ultrasonic rangefinders. A laptop, mounted on the robot, will handle the mapping and navigation using the Robot Operating System(ROS) framework. Additional location and navigation information will be generated through scanning wireless access points in the vicinity of the robot. The location of these access points will be estimated based on relative signal strength, allowing the generation of electronic landmarks detectable during navigation. A stationary server will do offloaded data processing and serve as the web interface, allowing users to send delivery commands to the robot.

II. Requirements Specification

Needs

1. Deliver small payloads typical of an office environment. (0 - 25lbs)
2. Map surroundings without user intervention
3. Plan route and navigate to a user provided waypoints
4. Follow an itinerary
5. Avoid all collisions with obstacles and people in its path
6. Allow users to view the robot’s map and status

Engineering Specifications

1. The vehicle will be able to receive a given location and autonomously drive itself to a specified location (1,3,4,5).
2. The vehicle will move at a top speed of at least 5mph, and able to brake to a stop within 2 meters. (5).
3. The vehicle will be able to determine its location with enough precision to enable navigation (3)
4. The vehicle will have a secure platform for placing payloads. (1)
5. The vehicle will be able to navigate to a destination and return back to base station without needing to be recharged (1,3,5).
6. A successful delivery in ideal conditions must be completed in no more than three times the time it would take a human courier. (1,3,5)
7. The vehicle will be able to take a series of destinations and automatically drive itself to all of these locations (3,4).
8. Users will be able to select a destination(s) for the vehicle via a web interface, as well as view the current status of the vehicle, such as its local map and delivery queue (1, 2, 3, 4, 6).
9. The vehicle will not move towards any objects detected within 1 meter of the sensors (5).
10. The vehicle’s range of destinations will be one floor at a time, and the vehicle will be prevented from driving near any stairs or elevators (2,3).
11. The vehicle will terminate all power to motors in the event of any collision with an object (5).

Comment [1]: rfm6038:
Again, This is just "Brain vomit"

The main goal for the vehicle is the ability to navigate to a specific destination. Subsequent goals are mainly expansions of this goal with additional features. The battery life will be the limiting factor in how many deliveries can be made in a row without a need for recharging. In the worst case, the vehicle will need to be able to navigate itself to a point on the map, without any previous mapping or navigation information of the route, and return to base station within one charge.

The vehicle will have limits on speed and distance thresholds to ensure safety and prevent collisions with any objects or people. By having the limit on the speed, the robot will be able to sufficiently brake in time to prevent a collision with any close objects in it's path. Additionally, the robot will not move forward if it does detect an object immediately in front of it. Finally, a fail safe will be added, terminating all power to the wheel motors when it detects a collision or when instructed via an administrative control. This termination of power drive power is needed to prevent any further damage in the event of a collision.

The limit on the vehicle's range for one floor is a limitation imposed only by the limited scope of this prototype. On the RIT campus where the vehicle will be designed and tested, the only means of transportation between floors are elevators and stairs. Interaction with elevators would be a complex and custom task, which prohibits their use in the context of this project. While this does restrict the robot to local deliveries, it is still useful and can easily deliver payloads between offices in the same department or floor, or manually moved from floor to floor. As multiple floors are mapped, the system will build separate maps for each.

III. Concept Selection

Existing Products

There is a similar system used in medical applications called HelpMate. HelpMate is used primarily in hospitals. *"With a 200-pound payload and various lockable compartments for storage, the robotic courier is capable of smoothly transporting pharmaceuticals, laboratory specimens, equipment and supplies, meals, medical records, and radiology films back and forth between support departments and nursing floors."* HelpMate weighs 600 pounds and is on a much larger scale than the proposed project. HelpMate determines the vehicle's orientation by looking at the ceiling lights. (In the hospital that uses HelpMate, the lights are rectangular, and oriented in a certain fashion. The implementation for this project uses more advanced localization technology, and does not make an assumption based on lighting.

Another existing system is TUG, created by the company Aethon. This system also seems to be primarily aimed at medical applications. As these systems already exist, it shows that there is demand for a robotic courier system.

Several existing platforms implement similar ROS systems with LIDAR like scanning to accomplish navigation. The most notable of these systems is Willow Garage's "PR2." This $400,000 robot is far more advanced, including two high-degree-of-freedom arms. This system is prohibitively expensive for office courier applications. A more suitable system is their "Tturtlebot", a $1,500 platform based on the iRobot create and Microsoft Kinect. While the price is more reasonable, the payload weight capacity is not nearly high enough to meet the needs of a shop courier. Our system uses techniques proven by the above platforms from Willow Garage, while improving payload capacity and lowering the price-point.

Navigation Scheme:

The core feature of the platform is the ability to navigate between waypoints autonomously. There are several different systems to provide location information to the platform to allow it to properly form a route and navigate to a location. To objectively record the quality of the solution we must specify the criteria of judgement for the subsystem. The system must be accurate for both location and orientation, robust, deterministic (noise levels), and able to work in any delivery domain encountered by the platform without special tailoring. Several proposed solutions are as follows:

Guide Wire: A "Buried" guide wire carrying HF signal used as a virtual track that goes around the delivery domain. Fairly robust, however destination location would be very difficult. Heavy modification of the environment is required, making this nearly unusable.
RFID tags: RFID tags will be placed at regular locations on the floor/baseboards. Absolute location is very accurate when they in range, however the system does not robustly handle getting lost.

IR Beacons: Strategically located modulated infrared beacons broadcasting UUIDs would be placed at all destinations. Less modification of the environment is required than the previous two systems, but absolute location is difficult to obtain,
Linear Scanner+ SLAM: The platform would mount a linear distance scanner (LIDAR, scanning sonar, scanning infrared rangefinder, Microsoft Kinect camera, etc.) paired with a Simultaneous Location And Mapping (SLAM) particle filter to internally represent environment and determine location.
WiFi triangulation: Map and reference a building's wireless network infrastructure to provide coarse location very reliably, however precision is not ideal.

	Location precision	Orientation	Robustness	Determinism	Domain ability	Tot.
Linear scanner+SLAM	**.8**	**.8**	**.8**	**.7**	**.7**	**3.8**
Wifi-Triangulation	**.6**	**.2**	**.9**	**.8**	**.6**	**3.1**
Guide wire	**.4**	**.8**	**.6**	**.6**	**0**	**2.4**
IR Beacons	**.2**	**.5**	**.4**	**.9**	**2**	**2.2**
RFID tags	**.9**	**.1**	**.2**	**1.0**	**.1**	**2.3**

OS:

An integrated robotic software platform will be used as the primary control software. It will integrate the various sensor data as well as sending commands to the robot based on the intended navigation. Various platforms were evaluated to determine which platform best meets these requirements. Several criteria were used to determine the best platform. The platform must provide support for all sensors used by the robot. The main sensors to be supported are the Kinect and Wi-Fi. The platforms were evaluated on cost, availability of source code, development language and licensing. The table below summarizes the results. If a square is left blank, the platform had already been ruled out before determining the result of that square.

	Kinect Support	Wi-Fi Support	Free	Open Source	Languages
ROS	Yes	Yes	Yes	Yes	C++, Python
Robotics Developer Studio	Yes		Yes (Education and Research)	No	C#/.net based
Urbi	Yes	No	Partially	Yes	Urbiscript, C++

ERSP			No	No	
The Player Project	Yes	Yes	Yes	Yes	C, C++, Python, Ruby

Based on the selection criteria, Robot Operating System (ROS.org) was chosen. The main compute module will consist of a small network of laptop computers running Ubuntu Linux with ROS running on top. Each computer will be responsible for processing different operations. Depending on the computing resources required, more or fewer machines can be added to the system.

3D Scanner:

The requirements of the 3D scanner were fairly vague, so a number of solutions were proposed. These solutions included the Kinect, laser radar (LIDAR), a network of sonar sensors, and a series of "bump" push-button sensors. Each of these four potential solutions were compared by expense, field of view, operating range, ease of integration, and resolution. The comparison is shown in the following table.

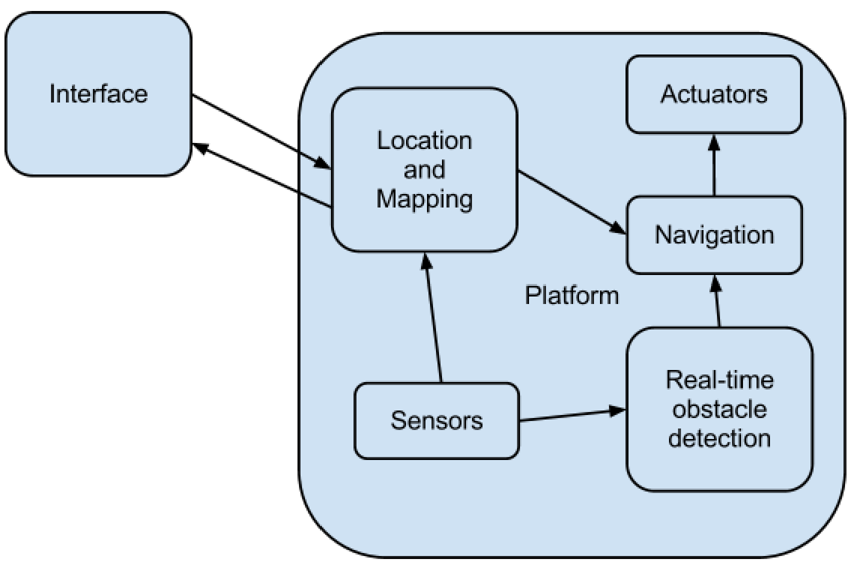
	Kinect	LIDAR	Sonar Sensors	Bump Sensors
Expense	0	-3	+1	+1
Field of view	0	+1	-1	-1
Operating range	+1	+2	0	-2
Ease of integration	-1	-2	0	+1
Resolution	+1	+1	0	0
Totals	**+1**	**-1**	**0**	**-1**

The “bump sensors” were ruled out mainly because they can only detect objects once they come into contact with them. This severely limits the speed that the robot can travel and nearly eliminates the mapping capabilities. The LIDAR was ruled out because of its extremely high cost and low availability. The Kinect was ultimately deemed most capable due to its relatively high resolution, large operating range, and the reasonable cost.

IV. System Design

The system will consist of a wheeled delivery platform. This platform will map its surroundings using a tiered sensor network to provide increasingly fine grain location and obstacle information. Using this map, the platform will navigate between different points, avoiding any new or transient obstacles along the way in real time with supplementary data provided by the sensors. Requests for delivery will be sent to the platform via radio, and the platform will transmit supplementary and diagnostic information.

This system is made of several subsystems that each perform an integral part of the working robot. The first is the mechanical platform including all mounting points and actuators. Closely coupled with this system is the sensor network and actuator control subsystem that manages all auxiliary sensors and actuator control. This sensor network is made available to the control algorithms through the operating system running on the main compute platform. On this platform, the mapping and localization subsystem determines location. The navigation subsystem long-term long and short-term route planning as well as obstacle detection and avoidance.



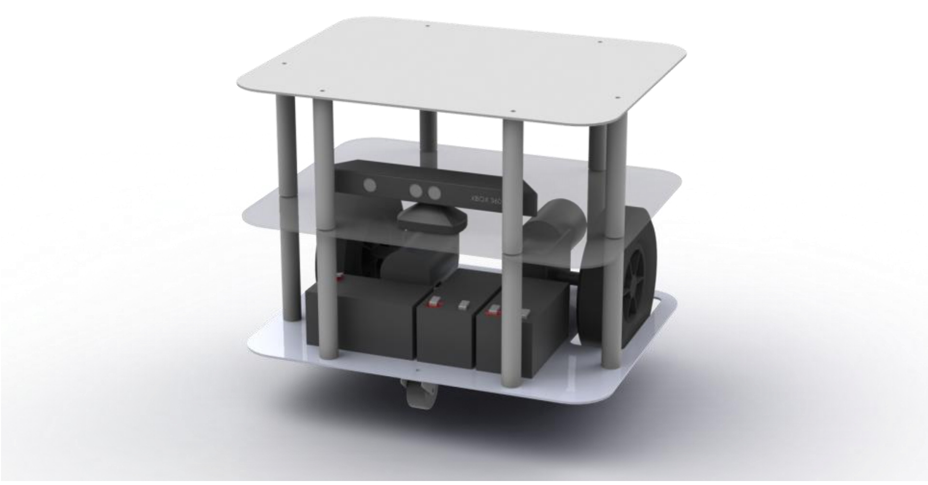
Autonomous mobile robot for indoor package delivery.

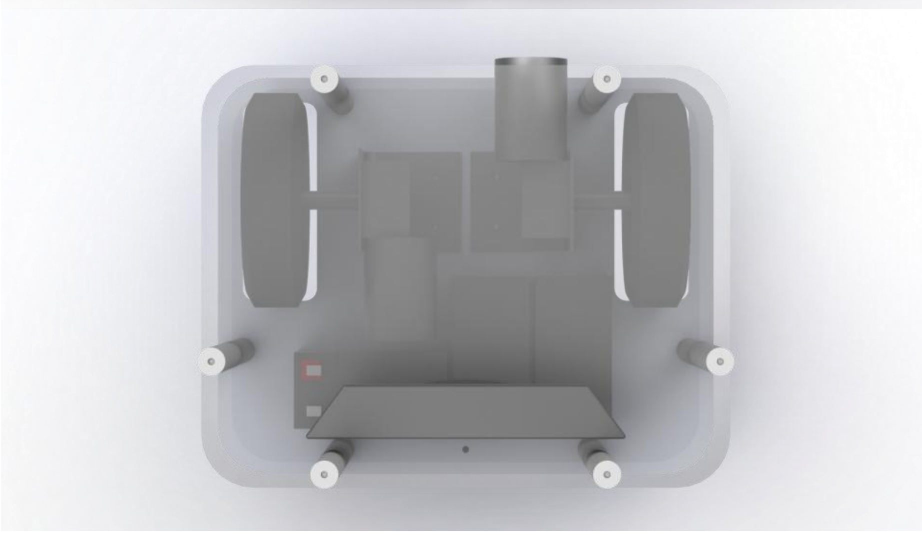
Mechanical Platform:

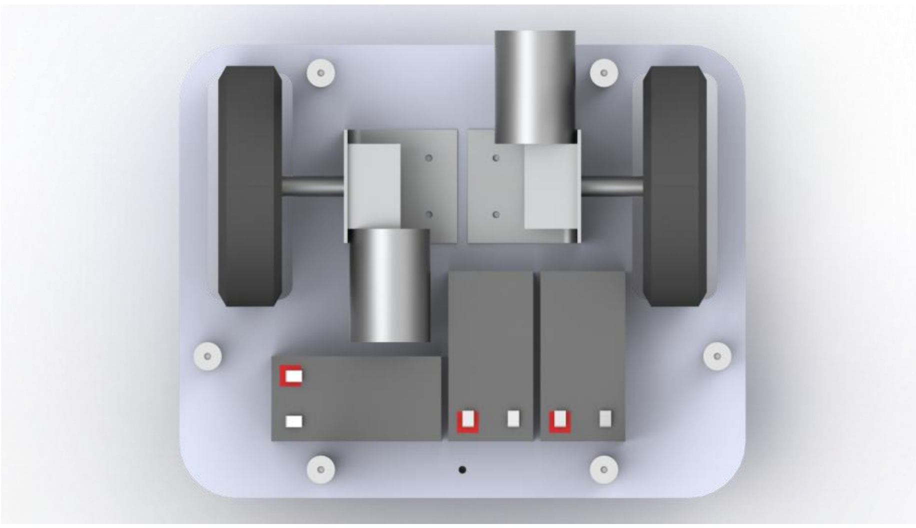
The mechanical platform will be a circular differential drive base with near zero turn radius. The drivetrain will consist of two 12-V DC motors with wormgear drives, and small 10 inch wheels. A dual channel 200-W motor controller with current feedback will be the primary actuator controller, interfacing via a serial connection to the main control computer. Rechargeable sealed lead acid batteries will provide power for both actuation and computation.

Actuators:

The primary actuators for the system are two surplus electric scooter style wheelchair motors with worm-gear drives. These were chosen for their high output torque, reasonable nominal speed, and relatively low operating voltage for the output power, simplifying power supply concerns. These motors are driven by the Pololu Robotics Dual VNHSP30 Motor driver board, which amplifies the logic control and speed signals to high power outputs for the DC motors. This board was chosen because of the simple interface, ability to handle the high output current at low voltages that we expect from our motor choice, and current sense feedback to detect stalls and wheel slippage.

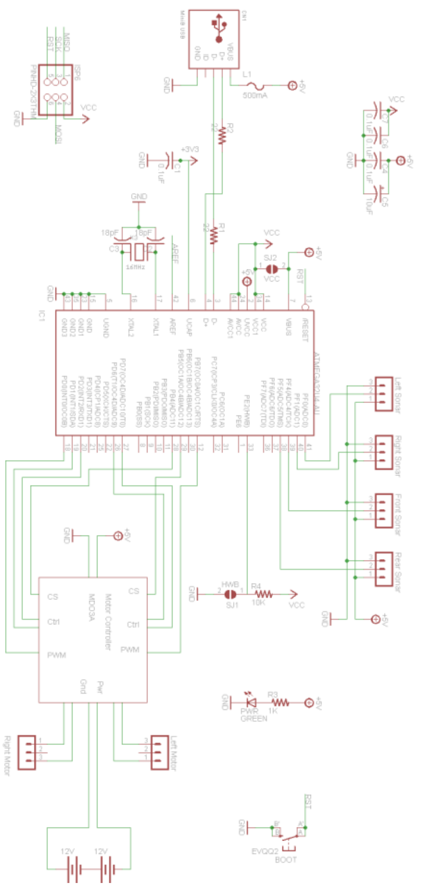






Sensors:

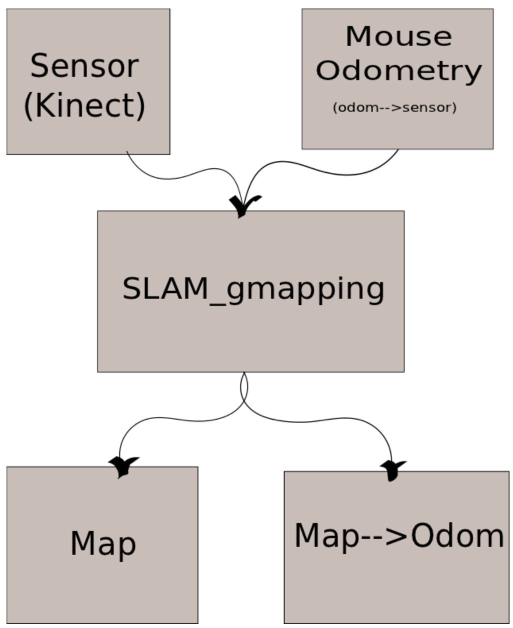
Several different types of sensors will be used in order to acquire data regarding the vehicle's motion and the environment surrounding it. The main visual sensor will be a Kinect sensor, able to provide a 3d depth map, giving the robot the ability to detect objects in its path. This sensor will be supplemented with four ultrasonic rangefinders mounted at 90 degrees from each other to detect glass objects and windows. An optical computer mouse mounted to the underside of the chassis will provide odometric feedback for the platform's movements. The Kinect data will be superimposed over this movement data for the generation of an internal map of the environment. All sensors will communicate over a custom USB device and enumerated in the host filesystem.



Sensor and motor interface device schematic.

Mapping and Localization:

Simultaneous Mapping and Localization (SLAM) will be the principle navigation mechanism used by the robot. The principle behind the mapping component of SLAM is a continuous update of a position by known relative increments via an odometer system (odom-->sensor transform) superimposed with sensor data. The position information is further updated by the localization component, which accomplishes its goal using the current sensor view to determine possible locations. Over multiple passes, the map becomes more precise and the location data more deterministic. From this map very accurate position and orientation data can be obtained. Robustness is obtained by a probabilistic model of position, allowing a lost platform to second-guess and recalculate position. The approach only relies on visual features, does not require any modification of the delivery domain, and will work in all but the most hectic of environments.

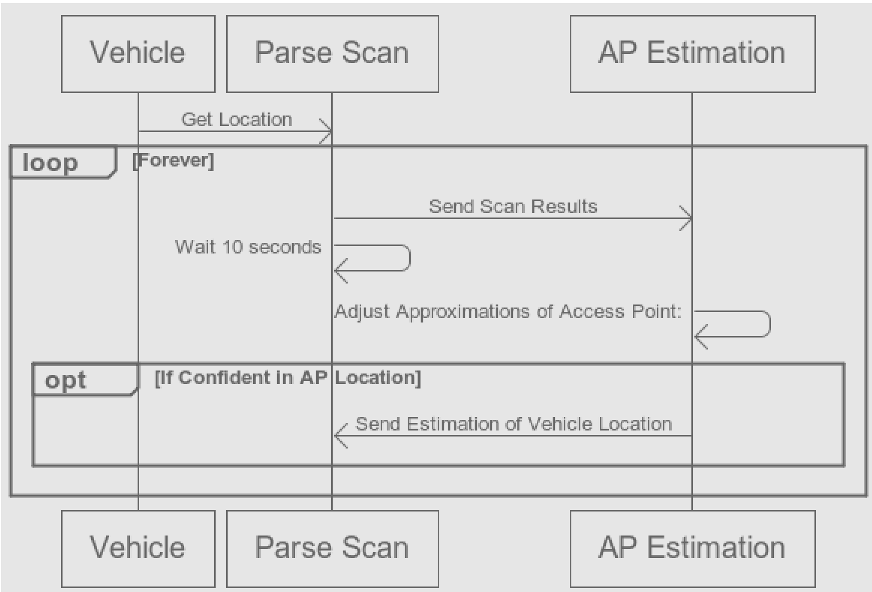


To achieve the performance required to deliver mail, packages, and parts, the system must provide precision on the order of +-6 inches, enough to successfully navigate the platform through doorways. The range of the sensor must be enough to detect structures at distances as large as the largest empty space in the delivery domain before resorting to more coarse location systems. This renders the some sensors unsuitable for large, flat spaces, but ideal for intra- and inter-departmental mailings and vice-versa. While not limiting, the maximum operating speed of the platform is a function of the update frequency of this system, the speed of the platform, the range of the linear sensor, and the level of activity in the domain. For office environments, a rate of 1-2 Hz with a 12 foot sensor range and 3mph robot should be sufficient, with slower speeds and longer sensor ranges allowing for even slower updates.

Wireless Triangulation:

As another layer of the sensor network, a wireless card will be periodically scanning the wireless networks in range of the vehicle. Over time, as scans accumulate, this will produce more and more accurate approximations of location through trilateration. This is useful in two different ways. The first is the fact that each of these wireless access points is unlikely to change, so it will provide a series of landmarks to guide the robot in navigation. Once access point locations are determined with sufficient confidence they can be used to produce an estimate of the robot's location. This location can be filtered with the data from the other sensors and map to increase location accuracy.

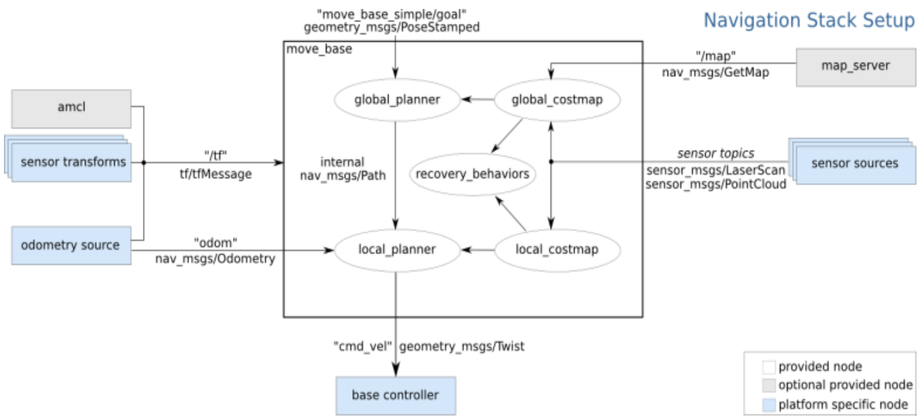
This subsystem will scan for all wireless access points in range every ten seconds. The period of this length was chosen because the robot will need to move an appreciable distance before there is a notable change in signal strength of the access points. Once these measures of signal strength are obtained, the robot will then parse out the needed information (MAC Address and Signal Strength) and write the information out to an external file. This file will be monitored by another program which will sort all of the scanned information by MAC address and use the signal strength and robot position in order to estimate the locations of the access points. Geometric calculations may be too computationally intensive to perform on the computer located on the vehicle, and may need to be calculated on another computer, such as the base station. ROS topics are ideal for this type of offloading. The delay introduced here is acceptable, as the data is not time-sensitive. The flow of this subsystem is shown below.



One of the issues with wireless scanning is irregular wireless propagation, especially with walls, floors, and other hazards that may block the wireless signal. This is mitigated on the RIT campus due to the way the broadcasting is done. Each access point broadcasts three separate networks, each with a different security scheme and wireless channel. Because of the channel difference, each will propagate slightly differently, allowing the irregularities to be averaged out. Detection of multiple networks from the same access point is possible due to the fact that the MAC addresses of each network on the same access point only varies in the last four bits.

Navigation and Object Detection:

The ROS navigation stack will provide local and global route planning for deliveries. Global route planning determines the general route through the static map as provided by the SLAM subsystem. Object detection will supplement the map with transient obstacle data to proceed with local route planning, or planning within the current field of view of the sensor. The navigation stack allows for customization of both these algorithms, and it is estimated that much of the configuration time for the project will be on optimizing this subsystem.

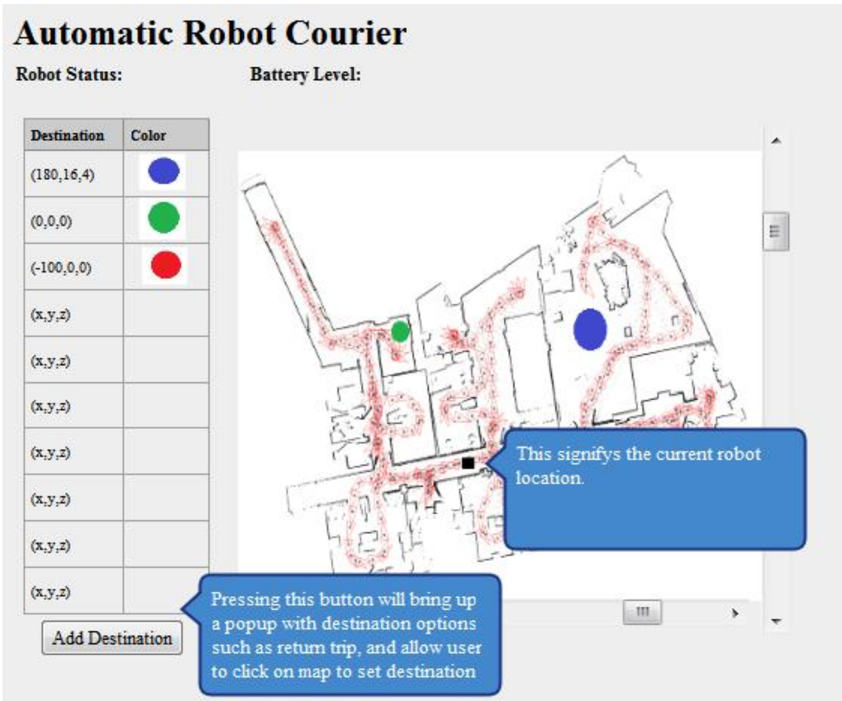


To progress towards a goal the UI will supply a simple goal position to navigate to. The “map_server” supplies the map gathered previously to the global route planner, while the sensors supply obstacle data to both the global and local planners in the form of point cloud messages. This sensor information is transformed by the corresponding sensor transform and made absolute via the root to base transform.

The global planner provides a basic route, and the local planner navigates through the current local ‘scene’, or recent sensor data, to realize this goal. This route is then translated into requested velocity and rotation vectors, or “geometry_msgs/Twist” messages sent to the base controller to be output to the motors. Progress is tracked by feedback from the localization components of the SLAM system, and the route is updated dynamically. In the event that either routing algorithm fails there are customizable recovery behaviors, such as waiting for a clear path, that will need to be tuned to the application environment.

User Interface:

The main portion of the interface will be a simple web page consisting of an image of the map as generated by the SLAM subsystem. In order to direct the robot to make a delivery, users will click a location on the map. By clicking on a specific point on the map, the user will trigger a simple goal to be enqueued in the navigation subsystem. The robot will automatically navigate to the first waypoint in the navigation queue, and wait for confirmation from the user before proceeding to the next. If there is extra time before the project deadline, additional efforts can be made to clean improve the interface by matching the map to a building blueprint, generate recurring routes, support for advanced scheduling of deliveries, etc. The following image shows a mockup of the initial version of this webpage, including the most recent map obtained by the robot, as well as the delivery queue and the locations of the robot’s destinations on the map.



Extensibility:

The Robotic courier system is designed to be extensible. Active payloads (such as a camera, gripper, etc) could easily be mounted to the chassis and added through the ROS system. The USB physical interface is expandable to add control for these payloads opening the platform to such tasks as surveillance, autonomous pickup and delivery without user intervention, tours, etc. One readily available extension of the robot is the automatic mapping of access points and wireless AP coverage, which could could be expanded to rogue access detection and localization, coverage maps, and Quality Of Service (QoS) monitoring.

Manufacturability

Because of the prototype nature of the project a redesign would be recommended before manufacturing this product. The system is primarily made from Commercial Off The Shelf (COTS) components to simplify design and maximize expandability at the expense of cost. The chassis construction should be optimized to increase aesthetic appeal as well as durability. The motors used were over-specified to accommodate a very large payload, as the size of the robot was as of yet undetermined. The expense of this drivetrain could be reduced significantly if a new drive-train was matched to the loaded weight of the final product. The electrical system was also designed to optimize expandability and development rather than for reduced cost, and should be optimized for the final product.

Reliability

Through the multi-tier sensor network, if one tier of sensors fail (such as the kinect or wireless scanning) the robot will still be able to navigate, albeit at a much less efficient level. In the event of a more complex failure the robot will at the very least know its approximate location and where base station is, allowing the vehicle to retrace its steps in order to return to the base station. Because the system is built around discovery and mapping, the system is naturally robust as any deviation will be handled as new exploration.

Background

Most of the background knowledge for designing our vehicle has been obtained through co-op work experience. Andrew worked at a company that used ROS on dismounted class mobile robots, giving him an introduction to the program which helped the initial learning curve. Russ had a co-op with a company that did wireless access point location through wardriving, allowing him to learn methods that worked out best for the wireless triangulation, as well as making him the best team member to work on that subsystem. Additionally, classes in real-time systems and interface and digital electronics helped improve the team's background knowledge for this project, producing experience in sensor interfacing as well as real-time communication schemes.

Any robotics project is a multidisciplinary interaction between computer engineering, electrical engineering, and mechanical engineering. Given our backgrounds, the focus is primarily on computer and electrical engineering, but the chassis construction and drive-train involve mechanical engineering. The power system for the robot, as well as motor control and sensor interfacing require a large amount of electrical engineering work. The software controlling the system is applied computer science.

V. Considerations

If the robot is successful it will reduce the number of courier jobs available. However, newer job opportunities could be created in the manufacture and repair of courier robots. The main economic benefit and purpose of the robot is the increased productivity, as workers can now use a robot to asynchronously pick up and deliver payloads. The productivity gain is expected to outweigh the economic loss of unskilled couriers.

Depending on the type of packages being delivered, there could be potential security risks with the robot. If the payload contains sensitive information and is not secured, the potential for theft exists. The chosen design does not have any security for the payload. This places a restriction on the types of packages that can be delivered with this robot. Future implementations could provide a cabinet with an electronic clasp that unlocks upon reaching its destination to solve this issue.

Whenever preexisting software is being used, the question of licensing arises. The chosen software platform was Robot Operating System (ROS), which uses a BSD license. The BSD license is ideal for this project, as it allows us to freely use and modify the software to suit the specific needs of the robot.

While the robot does use some materials that are not ideal from a sustainability perspective, it is a result of the material practices of the component manufacturers. The main consumable components of the robot are the sealed lead acid batteries, which are almost entirely recyclable. With limited budget and time, sustainability was not a major factor considered in the design process. Materials were chosen largely on ease of use for this project. During re-engineering for manufacture, more environmentally responsible materials could be sourced.

VI. Cost Estimates

Part	Quantity	Unit Cost
Microsoft Kinect	1	$99.99
Wireless Card	1	$49.99
Laptop	1	$500.00
Motors	2	$517.00
Sealed lead acid battery (17 A hour)	3	$48.69
Ultrasonic sensors	4	$24.95
Chassis materials	1	$300.00
Chassis fabrication	1	$200.00
Motor driver	1	$59.95
Optical mouse	1	$24.95

VII. Testing Strategy

Comment [2]: rfm6038:
Just a placeholder for me to do the grammatical corrections from the design, as well as make the adjustements to this section.

Subsystem Testing Strategy

Mapping Subsystem

The mapping subsystem is a critical component in the navigation of the platform. It must provide accurate and repeatable data for the navigation subsystem. Many components influence the quality of the measurements taken, so each must be tested individually to insure proper functionality.

Camera Transform and Mounting

The mapping system gets its data from the on-board depth camera. A single horizontal slice is taken from this depth map to provide the two-dimensional data for the map. This algorithm requires precise mounting-position data to ensure that the slice is exactly horizontal. Any deviation from this may cause obstacles to leave or come into view as distance changes.

To test this system the camera will be installed and configured as per normal use, with the robot facing a large, flat space. An adjustable tape measure will be extended into the field of view of the camera at varying distances from the sensor until the sensor detects it crossing the horizontal threshold. The angle offset will be calculated based on the measurement at these points.

Odometry

To determine the relative location of the different measurements while the platform is moving, the system requires a robust odometry system to provide the offset between different samplings. Long term variance can be compensated by the SLAM particle filter; however any drift should be minimized to the extent possible.

Verification of performance will first test the position feedback in a straight line at nominal operating speeds. The robot will be controlled to drive directly forward a known distance, and then rotated a set angle. The output from the odometry system will be calibrated based on the measured actual distances. To test accumulated drift and hysteresis, more complicated test patterns will be used, including driving forward and backwards several times, coming to rest at a final location, and, finally for a complicated route through the building, driving “blind” with teleoperative control.

Integration with the sensor will also need to be calibrated without interference from the particle filter. To perform this test, the simple tests listed above will be performed with the sensor data being recorded. These data will be overlaid in the visualizer with long decay time. Any variations should then manifest themselves as artifacts in the visualizer (thickened walls for translation noise, skewed lines for angular noise). Because of the visualizer and map system’s proportional coordinate system, any noise should be measurable within the visualizer.

SLAM

The final test will be the production of a floorplan map using the particle filter and the supplied data. The map will be inspected over time, to ensure that all transient obstacles are being removed and sensor noise is being accounted for. After the ploatform maps a room, the room will be physically measured to determine the accuracy of the generated map. Additionally, the room will be examined for any permanent or transient objects that may interfere with movement of platform. The generated map will be checked to make sure all such objects are included. All mapping will conform to the ROS map format to minimize integration errors with the navigation system.

Wireless Data Acquisition

There are several tests that can be done on the wireless data acquisition and measurement system. The system must be tested to ensure that it is able to determine wireless access points and get the signal strengths associated with these access points. The system must also ensure that it properly reads and writes access point data to and from its data store. The final set of tests for this component will verify that the data obtained can be used to accurately estimate the location of the access points. This will be done in two different ways: the first of which is taking all data points known and averaging them together in order to determine pre-existing information from the vehicle's access point history. The second method will be done iteratively as new data points are obtained. The goal of these tests is to ensure that the estimations obtained from both of the methods are very similar (within 1 meter) for the same data points used in the robot. Note also that for these tests, since the subsystem will be on its own, all location data will need to be manually entered. Additionally, some way of knowing the exact locations of the access points will also be needed, to help determine the accuracy of the estimations.

Tests

Parsing Software

1. Detect Nearby Access Points
    - Parse and determine if values are reasonable
2. Write into Data Structure
    -Test against known database

Access Point Calculation Software

3. Read from Data Structure
    -Test against known database
4. Estimate Location of Access Points
    -Use all known data points at once
    -Use iterative approach, see if results are similar
5. Merge Access Points Together
    -Because of RIT's network, three networks will be picked up simultaneously from each physical access points. The MAC addresses will be the same aside from the last 4 bits. The software must determine this by having all 3 calculations converge into a single physical point.

Platform Control

The platform control subsystem is responsible for the control and interfacing of the robot's peripherals from the PC. This system uses a USB link to communicate between the PC and the host controller. The USB link depends on a kernel device driver on the PC and USB protocol support on the controller. Both of these components have to be tested for compatibility and robustness. The controller must additionally process the motor speed commands sent from the PC. The controller should ensure that both motors actually turn at the same speed when set to operate at the same speed. When the sensor data feeding the controller changes, the controller needs to forward the new values to the PC. One of the more important features of the platform controller is the drive-train fail safe mechanism. If any of the range sensors reads that the robot is too close to an object, the controller will shut off the motors to prevent a collision.

Tests

USB Communication:

1. Loadable Kernel Module (LKM) loads/unloads from kernel properly
2. LKM attaches to controller and provides /dev interfaces
3. LKM recovers from invalid messages from controller
4. Controller allows enumeration, USB class identification, and pipe creation with host PC
5. Controller recovers from invalid message from LKM

Controller Operation:

1. Responds to motor set commands (adjusts motor speeds)
2. Adjusts motor output based on current feedback
    a. Platform will drive straight when motors are given identical speed values
3. Sensor values are sent to the PC when their output changes
4. Motors shutoff when the robot gets too close to obstacles
    a. This is an arbitrary value that can be set based on physical constraints

Navigation

The navigation system has to be able to successfully calculate a path to the destination and send the appropriate signals to tell the robot how and when to move. The testing covers several paths that contain conditions that can cause the robot to not make it to the desired destination. The main conditions to test are the presence of obstacles and the layout of the walls and doorways. The success or failure of these tests will be based on the avoidance of walls, the avoidance of obstacles, and making it to the destination using a reasonable path. If the above criteria are met, the test is considered passed. If any one of the criteria are not met, the test fails.

Tests

Robot can navigate to the specified destination given:

1. A path that contains no obstacles.
2. A path that contains one obstacle.
3. A path that contains multiple obstacles.
4. A path that used to contain obstacles, but no longer contains obstacles.

5. A path that has obstacles that have been moved since the last run.
6. A path that has the destination in a different room. (Successful navigation through doorways)
7. A path in open space. (Successful navigation without walls nearby.)
8. A path that contains a moving obstacle.

Integration Testing

Wireless Data Acquisition

The integration of the wireless system is a coupling with the moving vehicle. The wireless subsystem needs to be able to obtain location data from the robot, and use this in its calculations of access point locations. It will then report back the locations of the access points estimated. The method of communication between these two systems will be a ROS topic, a real-time 'pipe' like structure that can be configured to automatically respond to obtained data. The ROS topic read by the wireless subsystem will include the location of the vehicle, as well as the transform used to determine the offset between the vehicle and the wireless card that is obtaining the data. The wireless subsystem will then use the coordinates obtained by this topic and perform estimations in the same x,y,z coordinate space that is used by both the vehicle and the base station. It will then write out to a second ROS topic all of the access points and the estimated locations of the access points. The tests will ensure that the wireless subsystem is able both to read and to write these measurements in real time and use the real time data to perform the calculations.

Tests

Parsing Software

1. Read in from a ROS topic the x,y,z coordinates of the wireless card prior to every wireless measurement.
    - Using preset values and comparing
2. Writing the up-to-date coordinates along with the wireless data to the data structure used

Access Point Estimation

3. Create a ROS topic and write out real time data of access point estimations to the topic

Overall Robot Testing

Overall integration is expected to take the majority of development time. Once the individual systems are developed we will need to assure that each operates well together, and that the performance of the system in its entirety is able to navigate an office or shop environment successfully. After the components are tested individually, any components that directly interact will be tested together. This includes the mapping and navigation systems, navigation and the platform control systems, mapping and platform control system, and finally the wireless and mapping system. Once these tests are completed satisfactorily, testing of the complete system can be conducted. Any performance issues observed in the final tests will be narrowed down, and new, more rigorous subsystem and integration testing will be performed.

For system testing of the completed robot we will simulate several different use cases of the vehicle. The first test will create a single waypoint, and ask the vehicle to navigate automatically and drive to that waypoint. The next tests expand on that, adding return trips and multiple waypoints to the navigation queue of the robot. Advanced tests will integrate other more complex components, such as being able to take the floorplan obtained from the robot, allowing users to then order a “delivery” from the robot by clicking on a location on the floorplan. The location will then be added to the robot’s queue, and when all previous locations have been reached, the robot will automatically navigate itself to the given location. The criteria for passing all of these tests is the robot will autonomously navigate itself to the given locations/waypoints without any collisions or running out of battery power. In order to verify these tests, one person will need to follow the robot to determine that the locations specified are the locations the vehicle drives to, as there is no way to automatically detect that the robot’s location is exactly as it specifies when it claims it reaches a waypoint.

Tests

1. Navigate from Base Station to a waypoint designated from the robot’s computer itself
2. Navigate from Base Station to a waypoint and make a return trip.
3. Navigate to two waypoints designated at once then return to base station
4. Navigate to two waypoints designated at once, returning to base station after each.
5. Repeat tests 3 and 4 for many waypoints at once
6. Repeat tests 1-4 for waypoints designated from the user interface
7. Repeat tests 1-4 with random obstacles placed in the way after the mapping stage is done.
8. Repeat tests 1-4 with people walking in the path of the robot.

VIII. Risks

There are many risks to managing a fully autonomous device; specifically having it interact with its environment. The system needs to minimize the risk of having the robot becoming stranded due to power loss or due to theft. Lack of power can be caused by a depletion of battery power, or damage to the device from a collision with an obstacle. This risk can be minimized by adding a power sensor to the device, allowing it to monitor the remaining charge. Given this data, the vehicle can decide to return to its home base/charging station when low on power, and wait until it has sufficient charge to complete the next delivery.

The robot must avoid collisions with all objects that may come into the path of the vehicle, such as people or walls. Extreme care must be taken in order to ensure that no collisions occur, and if a collision does occur, then power will be cut immediately to the motors, to prevent it from moving with any possibly damaged components. Other issues with safety occur with the details of using an optical sensor that is unable to detect glass surfaces This risk is mitigated by using sensors of multiple types to ensure detection.

The risk of damage resulting from a collision can be mitigated by making the system more resilient to collisions. A more resilient system can be achieved by attaching bumpers to the outer edge of the vehicle and making the interconnections more robust. Falling is one of the most dangerous situations for the vehicle (i.e. falling down stairs, off of a ledge, etc.). To minimize this, the vehicle can utilize a forward sensor which detects the sudden drop in terrain and brake/turn accordingly. As for theft, a simple way of detecting this could be to implement a sensor that detects when the vehicle is picked up. Once triggered, the system would sound an alarm to draw attention and track its position.

In order to make sure that the sensors are providing accurate data, the system will need to implement self-tests on the sensors. If a sensor is found to have failed, it will enter a specific mode which either limits the functionality of the device to safe motions, or prevents the vehicle from leaving to make a delivery.

IX. Milestone chart

Task	**Scheduled Completion**	**Primary Team Member(s)**
Chassis construction	January 1st	Andrew
Teleoperation of Vehicle	March 10	Andrew, Alex
Auxiliary sensor installation and configuration	March 17	Alex
Mapping Framework Completed	March 24	Andrew
Wireless Data Acquisition & Estimation	March 31	Russ
Mechanical and Electrical Completion	March 31	Alex
Navigation Framework Completed	April 7	Andrew
User Interface Design Completed	April 7	Ryan
Wireless Data Acquisition on Vehicle	April 15	Russ
Wireless Location Integration	April 22	Russ, Andrew
User Interface Communicating With Vehicle	April 22	Ryan
Completed Assembly & Integration Testing	April 29	All
Completed Project & Testing	May 7	All