**Date: September 27, 2016**

This serves as an e-introduction among all of us—Magnus and I are interested to see if his swarm work can be integrated with our ant robot work to learn if collisions among densely packed task oriented collectives of robots can be used to ensure good flow, etc. I suggest that all the students and postdocs cc’d on this email self-organize to trade info on what we (they) are doing and then in the coming weeks we can all meet to see if anything emerges,

The new research direction on using ant-inspired collisions in swarm robot control/localization seems very interesting.

**Date: October 4, 2016**

Grad Groups meeting today featured a woman from the library who overviewed the different resources available to graduate students. For research in engineering, she recommended compendex and inspec as good databases.

**Date: October 5, 2016**

I was supposed to have a Skype call with JSP today to review the robotic ant systems, but he was not able to get online. Instead, we chatted on the phone, and I took down the following notes:

Mechanically it is sort of sorted out

* Two of them are not working well (Echo and Charlie?)

When you connect the batteries, you need to separate the batteries completely off of the wires

Issues are not commented within the code

I also spent a good amount of time today studying the codebase and learning its architecture. I think I have a good understanding of it, and I love the thorough documentation. I am not sure I understand the Watch Dog Timers (WTD), and I haven’t found a satisfactory explanation of them online. I would love an explanation of them from JSP.

**Date: October 6, 2016**

I continued reviewing the code, and Dr. Goldman showed me how to turn on the robots. There is a black switch on the side which turns it on and off, as well as a button on the main board which can be used to reset the robot.

All of the batteries were dead, so there are two things that I need to do to get started:

1. Learn how to charge the batteries
2. Make a power cable so that the robots can run off of a power supply (3.6V).

**Date: October 6, 2016 - Update**

I made a power cable to hook up to the power supply, but the power supply did not provide a high enough current to power all of the systems on the robots. I had asked Will to show me how to charge the Li-Ion batteries so I could charge them while I worked on the cable, so we were able to plug them into the battery cartridge. I also tested each of the robots with fresh set of batteries to see if they were operational, and have started a [Google Sheet](https://docs.google.com/spreadsheets/d/13Cs-2KYUY0CCcL2nHpb5zxYH1IgCLRBWxw0L3XnAVaY/edit#gid=0) to track the health of each robot over time.

**Date: October 7, 2016**

11:00AM - Will mentioned to me yesterday that the recharging process while the ants are running is not a straightforward process, and there are several files floating around on the lab desktop with tables I think might be necessary for regulating the charging method. See “C:\Users\vlinevich3\Desktop\Charging MPR121 Program vals.xlsx”.

12:00PM - Continued going through the documentation left by JSP and fleshing out the [Robotic Ant Status](https://docs.google.com/spreadsheets/d/13Cs-2KYUY0CCcL2nHpb5zxYH1IgCLRBWxw0L3XnAVaY/edit#gid=0) to get a better feel for the architecture and pre-existing problems. I also started up a [Trello Board](https://trello.com/b/C5Ynd05Q/georgia-tech-research) to track proposed changes from JSP. The Trello board is populated with the “List of things that need to be done in the future” in the [Ant Robot Mk.2 Software Documentation](https://docs.google.com/document/d/1-CU7JS0_f5LQ8yZ_QIc0lyrAXqRvDMj-JCnnrbyMqyo/edit)

4:00PM - Calling it a day. Feeling better about navigating the code. There seems to be some extraneous stuff left over from previous iterations, but I’d prefer to just get things running for now before going back and modifying working code to basically just look pretty. Best practices can be implemented after I have proven that I can get data with what we have right now.

I was hoping to hear from JSP today, but I might just shoot him an email with all of my documented questions later tonight.

**Date: October 10, 2016**

Notes from my conversation with JSP and Bahni.

*Questions for JSP:*

1. Where is the correct code?
   1. In JSP folder like Bahni showed you.
2. Upload process?
   1. Straightforward
3. Where is data stored and how do I retrieve it?
   1. Need to communicate with the Fio board, due to the lack of available pins on the Due. Talk to Bahni about this
4. Will mentioned to me yesterday that the recharging process while the ants are running is not a straightforward process, can you tell me about the recharging procedure and what I need to do to make sure everything is properly tuned?
   1. Bahni knows how to do this.
5. Debugging process? I am guessing I can get a lot from the data logging capabilities of the system
   1. Write data to Fio
6. I was going through the code, and I found a few lines pertaining to an [LSM9DS0](https://learn.sparkfun.com/tutorials/lsm9ds0-hookup-guide/introduction), which wasn’t on the inventory list, and I wasn’t able to find anything on the robots which looked like the board in its documentation. Is this code from a previous version...have I even been looking at the right code?
   1. You have, there is just a lot of old code leftover from previous iterations
7. Watch Dog Timer? Is this a deprecated thing now?
   1. Still needed. Maybe a bit messy, but it gets the job done.
8. *Recommended steps from JSP:*
   1. *Turning behavior needs the most work*
   2. *Focus on getting the IMU working*
      1. *Two modes:*
         1. *Magnetic field tells heading*
         2. *Using gyroscope to tell degrees of rotation*
            1. *Used to turn, but neither seems to work*
   3. *And get to know the hardware*
   4. *Resetting behavior is a minor issue*

Methods

TurnHeading - mainly for turning from one direction to another (180 deg). If preferGyro is true, then it will use the gyro readings to turn. If it is false, then it will use the magnetic fields heading to determine the heading of the robot as it turns. Neither of them seem to work.

Set charging station to 4.2 volts

PID needs some work, but Vadim is a better source than JSP.

PixyCam doesnt have a large enough field of vision. Modify the mount to make this better.

CapacitiveSensors should be reliable. The point is that we want to distinguish the wall from the other ant robots. Capacitive sensors charge and discharge the copper pads on the robot. If the time to charge changes, it can detect if it’s touching, and if it is, what type of material it is touching. When you connect it to a common ground, then the readings from the sensors will be totally different.

Antcomm lets you communicate with integers *fiowriteint*

repository/antorobots/FioWirelessV4

Clicking motion is a reset indication.

**Date: October 11, 2016**

10:00AM - Working on re-uploading the code to the robots which stall in *Going In* mode (Alpha and Echo). Method given below.

Uploading software to Robot Ants

[Uploading to the Fio Board](https://www.arduino.cc/en/Main/ArduinoBoardFioProgramming)

1. Unplug the 3.3V wire which goes to the LCD from the FTDI pins on the left side of the robot.

There are two arduino board on the robot ant platform (Due and Fio). Open the FioWirelessV4.ino in >> repository/antorobots/FioWirelessV4 in the Arduino UI and make sure that Tools>Board>Arduino Fio is selected. Verify the sketch.

*What is the intended reset switch behavior?*

11:30AM - Uploaded software to both non-operational robots and their behavior did not change. Next step is checking the constants files for each robot.

Both robots which do not work have low Kp values (Alpha = 1.5 and Echo = 0.75) compared to the working ones (Bravo = 2.5, Charlie = 2.5, Delta = 2.0). I increased the constantsA.h Kp values up to 2.5 and 3.5 and did not see any changes. Afterwards I returned the Kp values to the original values.

In constantsA.h, I updated OUT\_DIRECTION from 270 to 291 and updated STARBOARD\_DIRECTION from 170 to 200.

2:00PM - For Charlie, I am looking at the issues for the IMU.

dog.my ranges from 3100 to 9200 (always positive)

dog.mx takes on values in the range of -1500 to 200

To get familiar with the hardware, I am going to run through all of the testing methods in [Robotic Ant Status](https://docs.google.com/spreadsheets/d/13Cs-2KYUY0CCcL2nHpb5zxYH1IgCLRBWxw0L3XnAVaY/edit#gid=1335105529).

3:00PM - TestSwitches is not producing any output for Charlie.

3:45PM - After a bit of tinkering I was not able to produce any output, but this may be correct. JSP showed Bahni and I some slides yesterday which demonstrated that the capacitive charges vary depending on whether or not the robot is powered by USB (something about common ground?). I removed all test cases and ran Charlie in the test bed, and the Fio serial output reflected all touches I introduced.

4:00PM - TestSwitches looks like it is deprecated and I should have been using TEST\_CAP instead. Whoops :/

5:00PM - Calling it a day. I need someone to explain getDetectedContacts to me. I think I get the idea behind the if/else statements, but the meaning of switchState and the bitwise OR operations is lost on me.

**Date: October 12, 2016**

I have two big exams tomorrow, so I am not planning to do much today. Vadim gave us some input on the PixyCam regarding whether or not we should move to a different camera. We will need more processing power to use a webcam, because we will be required to handle the image processing onboard as opposed to on the camera itself like on the Pixy. He also recommended we use some sort of version control, which I am super happy about. When I asked the other members of the group why they didn’t use it, I think they were just scared to change over and opted to just keep making copies on the desktop. So the main task of the day is getting the files on my computer and then starting a git repository using [GT Github Enterprise](https://support.cc.gatech.edu/support-tools/faq/what-gt-github-enterprise).

There is a single repository for this project, with two directories, one for the Due and one for the Fio. In addition, there are a number of dependencies on external libraries found in the Documents/Arduino/libraries directory. The best solution I can think of it just having everyone install these themselves, due to the fact that I am not sure how a repository of libraries would affect the other libraries other people already have.

[Private repository is set up here.](https://github.gatech.edu/rwarkentin3/ant-robots)

**Date: October 13, 2016**

After the exams I came in for 3 hours and was working on getting the Github enterprise thing set up. It was a little messy with the security things in place, but I think it is all sorted out. I have the software running on both my laptop and the lab machine. *Version controlled code on the lab machine is in Documents/Github/ant-robots*

**Date: October 14, 2016**

I believe JSP said that the IMU issues popped up all at once, which makes me think that a software bug popped up during development that led to some IMU sensor data being interpreted incorrectly. There are several backup files saved to the desktop, and I am going to go through each one to see if there are any changes in the files dealing with the IMU that may be the source of this incorrect behavior.

There is lots of IR sensor stuff in the code, but it is all commented out. Are there even IR sensors on these robots anymore? Ask bahni…

Got side tracked and ended up reformatting a lot of the test code. Now the variables which activate the test loops are in a file called TestVals.h. I think it would be a good idea to remove the actual test methods from the .ino files and move them into their own .cpp file. In addition, it might be a good idea to move the TestVals.h data into RobotSelector.h and renaming to something like RobotSetup.h so that you only have to modify a single value prior to compiling your code.

I found an infinite loop in due\_digger\_mk2.ino that looks like it is intentionally forcing WDT to restart? Doesn’t seem like the best way to handle a hang.

numOfConsequitiveBackwardKicks is initialized all over the place but never used.

**Date: October 17, 2016**

Populated the Issues section of the ant-robots repository on GitHub, so the Trello board is a bit obsolete and out of the way.

10:30AM - Added parts documentation files. Eventually would like to add stuff like this to a Wiki to make it easier to access.

11:00AM - Added testing directions to the GHE wiki.

1:00PM - fioWrite() vs fioWriteInt() in due\_digger\_mk2/LinkArduinosI2C.h?

2:30PM - Committed a change to the test\_branch repo that uses the enumerated testing method.

4:45PM - Bravo works well. The screws which attach the claw to the servo motor fell out, so I had to do some tinkering to get everything attached. The claw seems to be operational now, and I mounted the claw on Bravo so that they cannot fall out again.

**Date: October 18, 2016**

Running lots of tests with TEST\_TURN\_HEADING

**Date: October 19, 2016**

Look up code reference on line 2217. dof.gz is the rotational acceleration about the z-axis?

preferGyro is set to true on line 906 if there is something in the payload. Is this why turn heading seems to work dring going in but not during going out?

This might explain the issue with the LSM9DS0: <https://forums.adafruit.com/viewtopic.php?f=19&t=66571>

So the LSM9DS0 on Bravo stopped working for a little while. I found that I was able to fix it by uploading code with all power supplies unplugged, and then by...honestly, I don’t know. Resetting seems to help sometimes. I think there is an issue with power.

**Date: October 20, 2016**

Things I want to understand:

* checkWrongDirections() method
  + Returns a boolean value which is not used...see comments added above method in code
* where is current\_target\_heading used
* why is turnHeading() present?
  + As best I can figure, it is a hard coded method that will turn the robots around 180 degrees.
* What is the constant numOf*Consequitive*BackwardKicks?
* FollowLane()
  + Is a method in driveMethods.h. It polls the camera for data using getDetectedSigs() and moves forward if the size of the pheromone area is greater than a certain threshold (fixed at 150 in the method).
* getDetectedSigs()
  + Is a method in visionMethods.h. It is a method that declares storage space and clears previous information, and performs filtering of incoming detected color data where only the biggest block of each color signature is detected  
    returns pointer to an array struct.
* checkHeadSensor()
* Is exitTunnel() ever called? What is the difference between this method and goingOut()?
* Is nextMode variable variable necessary (see line 922)?

Added a serial output in FollowLane() with indicated if the area of the pheromone trail is not large enough to move forward.

**Date: October 21, 2016**

Problem with IMU didnt’t exist until capacitive sensors were implemented. Consider turning this sensing system off and checking IMU functionality again.

Car is dead. Got to leave early to work on it.

**Date: October 23, 2016**

Spoke with JSP:

* 1. C:\Users\vlinevich3\Desktop\JSP - Back\_UP\_051916\DiggerMk2\_vision\_new\_PhysicsLabSetUp
  2. Draining out the the batteries seems to help
  3. Turn on comm line, then turn on robot.

**Date: October 24, 2016**

IMU readings for robot are different depending on whether or not the USB is plugged in?

Raised the IMU higher off of the robot and the IMU testing gets better. Dropped it down and now it seems fine. Maybe one of the wires was loose?

Pulled enough cotton out of the wheels of Bravo to make a troll doll.

TurnHeadingRoss() is working. ExitTunnel() -> GoingOutMode() and handlingContacts is a little weird. Work on this stuff.

**Date: October 26, 2016**

The robot can dump!!! The second time it goes to dig, the robot seems to undershoot the bed when it attempts to grab the granular material. Not sure what that is.

Dr. Goldman and I observed a digging/leaving cycle, and he made a comment about the PID controller looking poorly tuned in during goingOutMode. I agree with his observation, but the robot works fine during goingInMode and I don’t think that the PID parameters change. Dr. Goldman suggested I reach out to Vadim and ask him about the PixyCam and look at the PixyCam software, and mentioned that I might need to re-tune the camera on Bravo (I think this is necessary for one of the other robots as well).

Lastly, Dr. Goldman asked about the power supply. I have only been testing the robots with fully charged batteries (~4.2V) and the robots seems to operate underpowered. It may be a circuitry thing, but I think there is just a lot more hardware on the robots than was originally planned for.

**Date: October 27, 2016**

<http://cmucam.org/projects/cmucam5/wiki/Porting_Guide>

<http://cmucam.org/projects/cmucam5/wiki/Hooking_up_Pixy_to_a_Microcontroller_like_an_Arduino>

Says Data out port should be set to 2, not 0 or 4 like you had…

So I found some files in Desktop/antrobots\_2.0 that are config files for the firmware on the Pixy. I uploaded the most recent version “roboJSP.prm” and the robot started working again. Part of this sets the Data out port to 3…yeah...that goes against everything in the documentation, but whatever works.

Car battery needs to be replaced and I have a RoboJackets meeting in the early afternoon, so I gotta leave early and probably won't get back in time.

**Date: October 28, 2016**

Ended up having to go in this morning to get my car battery replaced, so I came in late today. Yesterday I found that the documentation for the Pixy does not line up with the software leftover from JSP which allows the robot to receive input from the the Pixy.

The Out data port is 3, when UART comm should be on 2. This could be the result of several issues:

1. The Pixy is hooked up incorrectly to the Due.
   1. Checked this. The Pixy pins are correctly connected to the Arduino according to the UART setup described [here](http://cmucam.org/projects/cmucam5/wiki/Porting_Guide).
   2. If you open up PixyMon and try to configure a device, you can hover over “Data out port” on the Interface tab. This will display a comment which indicates that 3 is for UART.
2. The firmware for the Pixy has changed and the documentation does not reflect it.

I have downloaded all of the released firmwares, and want to test each firmware release with each of the .prm files I found to see which one operates best. This may allow me to observe the effects of firmware change on parameter processing.

In addition, the lens on the PixyCam can be focused. I may look into playing with this today after the firmware evaluation.

|  |  |  |  |
| --- | --- | --- | --- |
|  | pixycam.prm | roboVadim.prm | roboJSP.prm |
| firmware-1.0.2beta | Not tested | Not tested | Not tested |
| pixy\_firmware-2.0.5 | Not tested | Not tested | Not tested |
| pixy\_firmware-2.0.8 | Digs well, but gets frozen in Going Out mode. | Digs well, but gets frozen in Going Out mode. | Digs well, and performs the best of the parameters for pixy\_firmware-2.0.8 |
| pixy\_firmware-2.0.12 | Not tested | Not tested | Not tested |
| pixy\_firmware-2.0.17-general | Not tested | Not tested | Digs well, and perf  orms the best. Currently on board. |
| pixy\_firmware-2.0.19-general | Not tested | Not tested | Not tested |

I have opted to use the most current firmware (pixy\_firmware-2.0.17-general) with roboJSP.prm

I think that the robot freezes when there are not any pink blocks registered in the camera. I can “unfreeze” the robot by waving my rather pink hand in front of it...I should get outside more…

The first thing I want to do is see if the PID parameters are different for the GoingIn and GoingOut states. The GoingIn path following is very good, while the path following for GoingOut is poor.

**Date: October 30, 2016**

GoingOutModeRoss is the same as GoingInMode (the original method), with some slight modifications to handle behavior switching, but all in all everything is the same. I developed a test for the PID controller and will work on evaluating the output of the camera readings.

**Date: October 31, 2016**

Trying to figure out what is going on with the PID controller.

I have disabled the checkHeading method in goingOut mode, so it should just follow the lane and handle contacts, but it still exhibits overshoots and pivoting.

Test 1. So I disabled the checkHeading method in goingInMode, and initialized in goingInMode and it was able to follow the lane in both directions with the .prm file I made. Yay!...why?

Test 2. So I copied goingInMode into goingOutModeRoss and disabled the checkHeading method. I initialized in goingInMode, but in loop(), if goingIn is true, we go to goingOutModeRoss instead of GoingInMode. This approach also tracks the pheromone trail both directions!

Test 3. Initialize the robot in GoingOutMode.

Everything is the same except for if we start with enable\_GoingInMode() or enable\_GoingOutMode(). Both switch cases call GoingOutModeRoss() with the checkWrongDirections commented out.

enable\_GoingInMode() with checkPayload() if-statement commented out works.

enable\_GoingOutMode() with checkPayload() if-statement commented out does not work.

enable\_GoingInMode() with CONTACT if-statement commented out works.

enable\_GoingOutMode() with CONTACT if-statement commented out does not work.

enable\_GoingInMode() with DUMPING\_SWITCH if-statement commented out works

enable\_GoingOutMode() with DUMPING\_SWITCH if-statement commented out does not work

Everything but followLane commented out:

enable\_GoingInMode() … doesn’t work! That is sort of progress

enable\_GoingOutMode() … doesn’t work!

enable\_GoingInMode() with CHARGER if-statement commented out works

enable\_GoingOutMode() with CHARGER if-statement commented out does not work

**enable\_GoingInMode() with checkHeadSensor if-statement commented out does not work!!!**

enable\_GoingOutMode() with checkHeadSensor if-statement commented out

void DriveForward(uint16\_t x){ is weird...

Wait...ALLOW\_USELESS\_RUNS???

**checkHeadSensor is definitely an issue**. I have modified the bit relevant to goingOut mode that is reflective of the behavior for goingIn mode.

**checkHeadSensor** for goingIn mode looks like it has a 1 second check in it, but it really doesn’t. The syntax is wrong and the compiler isn’t catching it for some reason. However, for the goingOut mode, there is a proper 1 second delay, during which the PID feedback dominates the behavior of the robot.

**Date: November 1, 2016**

Went to check the IMU with an oscilloscope with Andy and he shorted the power distribution board immediately :( Calling it a day. Will try to take the Power Distribution board from...some other robot i guess?

[Arduino Due](https://www.arduino.cc/en/Main/ArduinoBoardDue) page says “The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.”

I think this could be a source of the issues regarding the PixyCam power, as we supply 5V, which is beneath the recommended range given above.

We are also only using 220uF capacitors, when all of the documentation left by JSP and Vadim indicates that they should be 470uF.

Static memory allocation for the PixyCam data.

Power Distribution Board Review:

Battery 3.6V goes to current sensor, and GND goes to GND.

**Date: November 2, 2016**

Might need to order more of the Crydom DM0063

Accidentally snipped the wires on Charlie instead of Bravo...oops.

**Date: November 3, 2016**

I did some more reading on the recommended capacitors to include in the circuit near the voltage regulator [here](https://www.pololu.com/docs/0J16), and found that while the minimum recommendation the provide points to a 33 microfarad capacitor with a 35V rating, the bigger the capacitor the better so long as the dimensions of the circuit allow it. This leads me to believe that the 220 microfarad 25V capacitors will be sufficient for our purpose.

**Date: November 4, 2016**

The maintenance guys are working at my desk, so I am sitting in Andi’s spot for the time being. To do today:

* Recheck the breadboard I made and make sure it is in line with the documentation left by JSP.
  + I am comfortable with the breadboard implementation -- I just need to add some wires which will enable the hookup.
* See if the software can be sufficiently tested on another robot. One of the robots wipes it’s PixyData after driving for a bit, likely to insufficient power. I believe I have already demonstrated that this is not an issue when it is plugged into USB (therefore receiving sufficient power), but I would like to check it again.

Things to check:

1. Voltage across Vin and GND when battery plugged in and system on
2. Voltage across Vin and GND when battery plugged in, system on, and touching charger
3. Voltage across 3.3V and GND when battery plugged in and system on
4. Voltage across 3.3V and GND when battery plugged in, system on, and touching charger
5. Voltage across 5V and GND when battery plugged in and system on
6. Voltage across 5V and GND when battery plugged in, system on, and touching charger
7. Voltage on line going to AnalogA1 on Due -- check that the voltage divider is set up right.

One of my concerns is that the 3.3V which goes to the power distribution board might actually just be whatever the battery voltage is. We should check the range of allowable input voltages on the components on the 3.3V rail.

**Date: November 7, 2016**

There is a 2V potential across the Crydom mosfet...why? New ones are on the way.

**Date: November 9, 2016**

Regulators are here and I have been struggling with the breadboard. Initial observation: if all of the systems are plugged in, then the input and output voltages from the 5V regulator are inconsistent. If I unplug the 5V rail on the distribution board then the 3.3V line goes to the battery voltage.

**Date: November 10, 2016**

More electronics work. I want to swap out the Power Board tomorrow to see if it is an arduino problem or a Power Board problem.

**Date: November 11, 2016**

I took the power board from Charlie, one of the other non-working robots, and put it on Bravo, which was otherwise operational. Bravo is now working fine.

Bravo run started at approximately 12:05 and ran until around 12:25

* Several intervention was necessary when the charging rod was caught on the charging wire in the corner. This corner is a bad spot to be. The table is uneven which means the robots can bottom out, and the frame holding the spring can hand on the robots’ charging rod.

Working on the power board and modified it a little bit to make the board run on a separate 5v regulator. The results were the same. It turns out one of the 5V regulators I have been working with is toast :(

All robots have power! Still have not deployed a board with the 12V regulator on it.

Started working on Delta and the power is wonky now? I have taken an osilloscope to the regulators and the power going to the Power Distribution Board is not working. Try swapping with another robot.

**Date: November 13, 2016**

JSP got back to me about the voltage regulator swap-out I have been proposing. Apparently he tried something like this in the past, and came to the conclusion that the 5V regulators are sufficient. He even had a box full of 9V and 12V regulators tucked away in a red box marked “Do Not Use”. Lesson learned.

Goals for the day: I believe I have 3 robots with operational hardware as of now, with only some minor tweaks to get all of them totally operational. During goingOut mode, there is a software issue which causes the robots to pause at approximately 1 second intervals which needs to be resolved, but in the grand scheme of things this is not too big an issue.

* Find cause of goingOut mode pausing behavior: I think I got it. There was a Forward(BASE\_SPEED); Delay(1000); command in goingOutModeRoss() which I think might have been causing the issue. Now resolved.
* Tune PID parameters of Alpha: Done - Reduced BASE\_SPEED and increased Kd
* Tune goingIn mode direction of Bravo: Done
* Check magnetometer on Bravo: Done - I think it is fine. There was some weird code in the method regarding dumping that I think was causing some issues.

**Date: November 14, 2016**

* Tune the PID parameters for Echo: Done
* Make a second battery pack so we can run at least two robots at once: Done

Also note that when you want to actually start running trials, you should take the USB cables off! The wheels get caught on the cables of other robots very easily.

Ran the two robots at the same time semi-successfully! Definitely need to increase the size of the capacitive panels on the sides by a pretty good amount.

Realized that I have a homework due tomorrow, not next week, so I have to leave the lab a little early today :(

**Date: November 15, 2016**

Fixed the capacitive panel on bravo, and got everything running. Checked the capacitive sensor values, and they had hardly changed, which means I might be able to increase the size of the capacitor areas on all of the robots without having to do much tuning. I will say that I think there is a bug in the method which write/reads the capacitive sensor readings to the LCD screen which I hypothesize is losing the last (least significant) bit of the reading.

Working on replacing the left motor on Charlie. Done. Now checking.

**Date: November 16, 2016**

Something is wrong with the IMU on Charlie? When the sensor is horizontal, the values are stuck between 270-360, but when the sensor is turned vertically, the sensor returns all values. I think I have the IMU on Charlie working now. The PixyCam is also acting weird. When I plug in the Pixy to the USB connect, the entire robot can power up, which does not happen for all of the other robots.

I found that the 5V and 3.3V lines on the Power Distribution Board of Charlie had been crossed (bad), but fixing them (and corrected the RX/TX lines) made the Pixy work. Next is the Force sensors of the gripper.

**Date: November 17, 2016**

Started looking into why Robot Charlie always thought there was something inside the gripper. It turns out that while the wiring on the robot matches that of the documentation, it does not match any of the other robots. The documentation for the Force Sensitive Resistor Board was wrong, and the 5V and GND wires should be flipped in order for the voltage divider to work with the current implementation of the code.

Broke out the breadboard to start experimenting with the IMU. By itself, the range of values is between 245 degrees and 360 degrees, so that is a bit disappointing. I was able to freeze it and unfreesze it, and now the ranges are totally different.

I can break/reconnect either of the SCL/SDA wires and operation continues. Breaking the ground connection hangs the system, but resetting gets everything going again.

There 5V regulator for the lines going to the PDB is broken - IMU is fine.

* IMU not working? Check the connections and then check the power supply.

**Date: November 18, 2016**

Took the oscilloscope to the PDB on Charlie and realized that the testing sketch would not be activating the relay supplying power to the IMU. Hooked up the relay control to 5V on the Due and everything is working fine now. The IMU range is still limited to 275-360 degrees, but at least it is communicating.

The IMU only starts returning incorrect values when plugged into the SCL and SDA lines on the PDB.

SDA line alone doesn’t work. SLC line alone doesn’t work.

So now let’s look at what else is connected to the SDA and SLC lines. Maybe something is accidentally grounded?

All of the actual pins on the PDB are good. I checked this by plugging in the SCL and SDA wires from the IMU to each of them with the only other connection being the jumpers going to the SCL/SDA pins on the Due. When I plug in the SCL/SDA wires from the capacitive sensor the IMU hangs, but when I plug in the SCL/SDA wires from the MAG, the IMU still works.

The ground wasn’t connected for the capacitive sensor. So it seems that when any of the boards on the SCL/SDA line hang, all of them will. Specifically, if the GROUND connection is lost, the SDA/SCL communication will hang.

**Date: November 28, 2016**

Back from exams/holidays! Something on Charlie’s Pixy is weird. The data doesn’t make sense to me. Perhaps it is an issue with syncing? I’m not sure, but I want to table this and work on the other robot for now. When I come back to this, I was to write a script that it Pixy-specific to see what exactly is wrong with the communication protocol.

I took the PDB off of Charlie and put it on Delta, and noticed that the 5V regulator which goes to the PB started heating up just like the original board’s 5V regulator. Is there something wrong with the circuitry on Delta? I put the PDB back on Charlie and it is working fine now. So something is wrong on Delta.

Narrowed it down to the 5V line on PB of Delta. Look here tomorrow.

**Date: November 29, 2016**

Going to try to figure out what is wrong with the 5V line on the PB of Delta.

PixyCam does not seem to be the source of the issue.

The Hall effect sensor does not seem to be the source of the issue.

The Force Sensitive Resistor Board is also okay.

The Gripper Servo seems to be the issue. When I unplug it, the 5V line returns to 5V.

I wrote a script to test the servo motors and they seem to be working fine on their own. Not sure what is up.

I think everything is working on Delta now. May need to replace the Gripper Servo sometime, but in the meantime see if you can make it work by tweaking the software.

**Date: November 30, 2016**

I unplugged the gripper servo and everything on Delta works fine now. I e-mailed JSP asking if he has experienced this issue before.

**Date: December 1, 2016**

Plugged the Gripper Servo into a different set of pins along the 5V regulator line and everything seems to be fine now? Why can’t things start working before I e-mail someone??

Robot Delta has been having an issue with tilting back on the motor mounts when it accelerates. This is because the battery pack is not placed in a very smart place. Additionally, Delta had the 3D printed battery pack on it, which made it weigh approximately 10 grams more than the original design. I replaced the 3D printed version with one of the older packs, and the robot moves much better now. It still rides on the back mounts a bit, but there is less weight on it now and it moves more like the other robots.

Glued the magnetometer to the robot, and now I need to recalibrate the IMU.

After I tuned the IMU on Delta, everything seems to be working fine!

Now onto Charlie. I am thinking there might just be a PID issue going on that I mis-diagnosed as a PixyCam issue. Now to go back and comment out all of those print-statements I put in the Pixy files.

Everything is working! Tomorrow I will work on calibrating the capacitors!

**Date: December 2, 2016**

Working on getting the capacitive sensors working today. I have to leave early today, but Keith worked out the issues with my building access so I can come in on weekends by myself now!

Plan of attack for capacitors:

First of all, I think there is an issue with the connection between the Due and the Fio, such that the last (least-significant) digit of the write-out is lost. I will test this by writing to serial and to Fio and comparing the output.

Control flow in fio\_wireless.ino was a little bit wonky. I interlocked and if-statement with a switch-case statement, but the if-statement conditions were not being enforced. Never run into this issue before. It might be an artifact of the Arduino compiler, or just C++ behavior and I was not using a good practice.

Started calibrating the capacitive sensors.

**Date: December 4, 2016**

Got sick this weekend – feeling better, but not quite 100%.

Looking at the code in Capacitive Sensor.cpp, the WallDoubleThresh implementation is not sufficient. We need to replace the single WallDoubleThresh variable with two variables WallDoubleThresh1 and WallDoubleThresh2 to store the low and high capacitive thresholds, respectively.

Furthermore, AntThresh is not used anywhere in the code.

**Date: December 5, 2016**

Tuning capacitors.

**Date: December 6, 2016**

Sometimes in TurnHeadingRoss the robots will reverse, but not turn. Not sure why this happens, but a reset seems to return the behavior to normal.

Charlie’s IMU is not one-to-one. Not sure what the best solution for this is.

**Date: December 7, 2016**

Regarding the issue for the robots getting stuck in turn reversal, but not turning. Maybe there is a bug in the capacitive sensor methods which does not call anything? Such that the robot just stays still until it times out?

Working on finding two robot for the demo.

Alpha – Working well enough. Left gripper sensor keeps disconnecting and the one of the Hall Effect sensors has a broken connection.

Quick note: When the robot spins out, it is in the process of switching turn\_heading\_direction.

Bravo – Works well. Need to tune the capacitive sensors a bit better because of the expanded panel reading false positives.

Charlie – IMU reads 315 twice, one of which is at the IN\_DIRECTION point, so half of the time it does not end up in the right direction. Try playing with the IMU a bit. Also need to tune the gripper parameters.

Delta – Works well.

Echo – Working well enough

Spoke with Will about making a resistive touch sensor. Might be worth pursuing.

There are no methods used to handle touching another ant, only walls.

I changed the wiring to the IMU so that it avoids passing near the PDB, and saw an improvement in IMU readings.

**Date: December 12, 2016**

Weather/stress got me sick, but I’m feeling good enough to be in the lab.

Target for the day is extending the size of the copper panels on Bravo and calibrating them.

Be sure to clean up the lab before you leave for the tour that is coming through tomorrow.

Look for non-conducting materials for the wall material.

Trim up the sides.

So Dr. Goldman had a good insight today. There is an issue with the capacitive sensor in that if two panels are connected via a conducting material, the circuit is essentially shorted such that the reading more or less goes to zero. This is okay for a single material, but if we want to distinguish between two materials touching multiple panels, we need some non-conducting materials in the mix.

Red latex band is okay ~10 drop

Clear Scotch Packaging tape is okay ~10 drop  
Duct tape is good ~ 15 drop

Cardboard is bad

Petri dish is okay ~15 drop

Red Brick is good ~35-40 drop

CD material is also good ~ 50 drop

Copper is 712 – 430 = 282

Cooper w/ tape Single Touch = 719 – 625 =

Cooper w/ tape Double Touch = 719 – 427 =

Aluminum w/ tape Single Touch = 711 – 580 = 131  
Aluminum w/ tape Double Touch = 724 – 450 = 274

I have found that I can apply a layer of packing tape on the copper and aluminum I can create a sort of composite with unique capacitive readings that do not short when several contacts are made.

**Date: December 13, 2016**

I added a layer of tape on the aluminum siding of the tunnel, and tuned capacitive parameters of Bravo for both the taped aluminum and the copper panels on the ants. It can sense each of them well.

Next I will increase the size of the panels on all of the other ants.

Finished increasing the panel sizes. Now to tune them.

Also need to find out why the robots reset in exitTunnelMode. I think I got this figured out. There are a lot of cases that need to be implemented. For example, there are some left-side cases in goingOut mode that are not being caught. Whatever the case was, I think I found it. Be sure to implement something similar tomorrow on the right side.

Bravo wires going to the left force resistive sensor are broken.

**Date: December 14, 2016**

I implemented some new cases that were not being handled in the goingIn/goingOut/goingCharging case that I think were causing some problems. I also changed the default case to reverse for 0.5 seconds, so that at least something happens instead of the previous implementation where Stop() was called. That isn’t a good solution for getting out of a jam.

For Bravo, I moved the power source from the PDB to the Due 3.3V supply, which seems to be better at maintaining the supply when it touches the charging wires.

Check the IMU settings on Delta and extend the top cathode on the charging rod down another couple millimeters.

Found that the I2C connections on Delta were not sound. Reconnected and all issues were resolved.

Tomorrow, let’s work on fitting battery packs onto all of the robots, tuning the IMUs on each robot and making sure all of the wiring is good. From there on, I think there will just be software stuff to tinker with:

Need a way to ensure that the WDT is called in turnheading scenarios.

Perhaps add a check for the IMU WHO\_I\_AM?

**Date: December 15, 2016**

Need to tune the PID parameters of Bravo, and figure out why it doesn’t dig well. Also need to tune the IMU a little.

Finished tuning the PID parameters for Bravo. Still need to tune the IMU.

Also need to tune the IMU on Charlie.

Rewrite the implementation of checkWrongDirections so that instead of checking all of the other directions, we only check the direction we are actively moving towards.

Spent all day working on getting battery packs prepared and tuning some PID stuff. Work on getting all robots running together tomorrow.

**Date: December 16, 2016**

Robots start to seem underpowered at around 3.8V.

goingIn and goingOut are not symmetric in their behavior – working on determining why. Got it sorted out, the checkHeadSensor was modifying the motion a little bit. I thought that I had addressed this issue before, but it seems I hadn’t completely fixed the issue.

I got all 5 robots running today. The ones with the newer battery packs don’t run as well – they are just too heavy. Replace them tomorrow with the older iteration and see if their performance improves. Some wires on the motors broke while I was working on Charlie’s drive train, so that is the first order of business tomorrow.

Afterwards, let’s talk with Bahni about what sort of stuff we need to look at for the reversal/lorentzian implementations.

**Date: December 19, 2016**

Working on fixing the motor connections on Charlie. I tried a lot of different configurations to address the issue of the robot’s being back-heavy, but did not find a suitable solution. Redesign location of battery pack?

I changed all of the 3D printed battery packs I installed on the robots (Delta and Charlie) back to the insertable kind that was originally used. They are lighter weight, and the reducing of mass on the rear of the robots improves the tracking of the pheromone trail in a significant way.

I because the range of aluminum contacts on the robots is now a single value, but the copper panels still require two ranges, I will just adjust the double contact aluminum values to be sufficiently large values that will never be returned by the capacitive sensor. This is definitely only a temporary solution for the time being, but I think it gets the job done without introducing in sort of glaring potential sources for error.

Temporarily pause? Dr. Goldman and Jeff suggested that another behavior that they have seen is an ant that observes a clog ahead of it and backs up a bit, without totally giving up. It will wait, and then proceed after some given waiting period.

Trying to figure out what is wrong with the IMU:

Added to setup:

dof.setMagScale(dof.M\_SCALE\_2GS);

dof.setMagODR(dof.M\_ODR\_125);

Changed in isWantedHeading

float heading=getHeading((float) dof.mx, (float) dof.my);

to:

float heading=getHeading((float) dof.calcMag(dof.mx), (float) dof.calcMag(dof.my));

I have found that the metal screws holding the IMUs in place have a significant effect on the IMU readings after changing the code slightly. This solution worked on robots Alpha and Delta. I discovered better results on Echo, but noted that the hy reading never becomes negative, even after replacing the IMU. I noted that the wiring also has an effect on the outputs, so I need to investigate this a bit more.

**Date: December 20, 2016**

Trying to figure out why the Echo IMU won’t return a negative hy value – determined I had just been testing with two bad IMUs :/

I think the IMUs just need to be tuned. Create a mapping function?

I wrote a matlab script that may be useful for calibrating IMUs with a bias. I think it works well. Work on porting it to C++ tomorrow.

**Date: December 21, 2016**

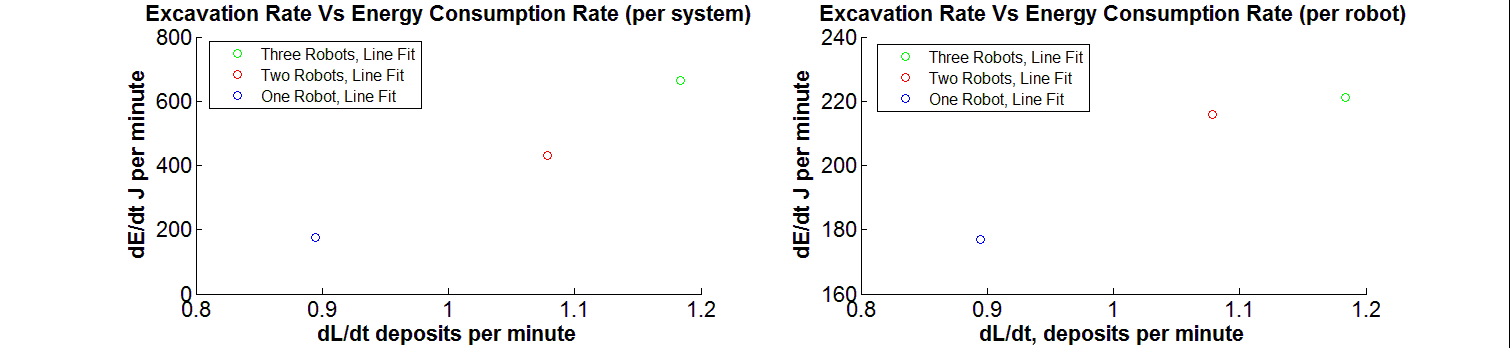
Added in the mapping correction to the getHeading method. It seems to work well on Bravo, which had a pretty bad IMU bias. The bias can drift by around 10 degrees over 30 minutes, so that might be an issue for longer trials. Sometime it seems like a reset can change the IMU bias, but all day today I did not observe that on a robot that ran more or less for ~3 hours, during which it reset many times. I also went through and cleaned up the pickDirection method and verified that everything works with the updated getHeading method. A few things needed tweaking, but for the most part it was all fine. Things to do when I get back:

* Perform the IMU bias correction on all robots.
* Check ellipses? Plot x vs angle to demonstrate deviation in y.
* Clarify what sort of reversal behaviors we want
* Probably need to rework Lorenz implementation?

**Date: January 2, 2017**

Getting back into things and I noticed this plot in:

C:\Users\vlinevich3\Documents\DiggerDataProcessingProgram



Super super super speculative, but it looks like the plot on the right might not be increasing linearly?

First thing to do is correct the IMU biases on each of the robots:

Alpha – Done

Bravo – Done

Charlie – Done

Delta – Done

Echo – Done

**Date: January 3, 2017**

I spoke with Bahni at the end of the day yesterday and if I understood her correctly, there is a behavior that she thinks is dominant in the videos she has reviewed:

Ants are likely to reverse as they go into the tunnel if they bump into another ant which is currently holding something.

In terms of implementing this, Bahni and I think an additional hardware system would be required. At the moment, there is no way for ants to share information with one another, even locally. Bahni suggested an LED might be useful, because we can use the PixyCam on the robots to observe the states of nearby robots without it necessarily being globally networked information.

In the meantime, now that the IMUs have been calibrated, I need to obtain their IN\_/OUT\_DIRECTION parameters, and work on the methods which allow the robots to give-up and perform the Lorenz trials.

Removed preferGyro functionality from the code.

Changed a recursive call in the dumping method to a do-while loop. This can be modified with a counter in the future to prevent the robots from getting stuck in the dumping state if there is an issue with the cohesive material being stuck in the gripper.

Possible changes for tomorrow:

DUMPING\_SWITCH is now longer useful in goingIn mode.

CheckPayload is not necessary in GoingOutModeRoss.

For goingIn method, enable\_turnReversalMode(7) redirects the flow of control to exitTunnel, instead of goingOut mode.

Besides these minor changes, I think I understand the code for the useless-run behavior well. Next is the Lorenz runs.

leaveDumpingSite() has bumpDelay() in it…not sure it works.

Is charging always allowed?...this might be what had been causing the robots to restart?

Charging stuff is pretty messy.

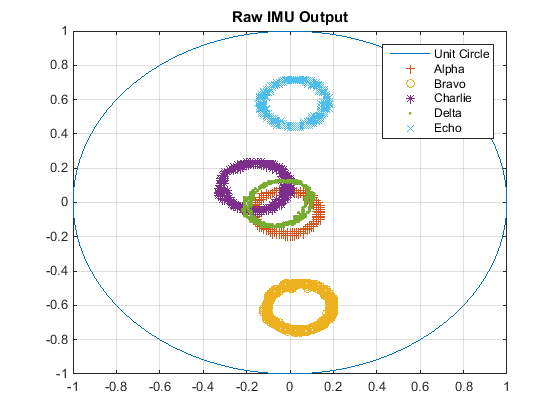
Spoke with Will about when Lorenz-evaluations take place. Currently the code only checks after a dumping event takes place, which is not in line with what Will has done. Instead, it might be better to have the Lorenz check in the goingIn portion of the control flow.

**Date: January 4, 2017**

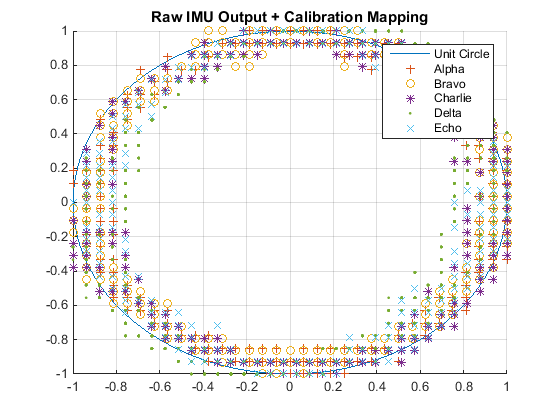
I think it might be best to move the Lorenz check from the DumpingMode() code to the if(goingIn)-statement in loop() such that it is the first thing that is checked. This is a change that I suggest because the current implementation does not actually begin to exhibit Lorenzian behavior until after the first dig/dump cycle has taken place, during which it is what we would classify as “active”.

If PROBABILITY\_DIG statement in evaluates to true within the goingIn if-statement, change the current\_target\_heading to OUT\_DIRECTION, call enable\_RestingMode(), and skip the rest of the functionality which sets up a goingIn run.

Before going into the details and testing the code, I want to give a little explanation for what I believe the main issue with the IMUs is and the solution I have come up with. It seems that these IMUs can have a significant bias in them which will affect the sign of their readings. Ideally, the IMU readings will be centered at zero, but in reality we see something like



I have written a matlab script ant-robots/debug/imu/imuCal.m and an accompanying ant-robots/debug/imu/imuCalDisp.m which will print out the parameters necessary for a mapping-technique I have added to the getHeading() function in the due\*.ino code. The result of the mapping is not perfect, as there is some ellipsoidal deformations the current mapping cannot correct for, but it is a definite improvement.



It does not really matter what the radius of the circle is, or even that it is the same across all robots -- I chose to make the unit circle a standard for the sake of simplicity.

Alpha seems to be running well with the calibrated IMU implementation.

Some quick modifications:

DUMPING\_SWITCH is now longer useful in goingIn mode. – Commented out

CheckPayload is not necessary in GoingOutModeRoss. – Commented out

For goingIn method, enable\_turnReversalMode(7) redirects the flow of control to exitTunnel, instead of goingOut mode. – Changed to enable\_turnReversalMode(3).

***Further notes***

*Dumping mode is a little weird*

*leaveDumpingSite() has bumpDelay() in it…not sure it works.*

*Is charging always allowed?...this might be what had been causing the robots to restart?*

*Charging stuff is pretty messy.*

I ended up gutting checkWrongDirections until it was more or less useless. Too much flag/timestamp stuff going on for something that is a rather simple sensor check. A lot of it was folded into some simple stuff tacked on after an isWantedHeading() check. This was motivated by the fact that I wanted to keep tracking the time we were goingIn, even if we needed to handle wrong direction turning and contacts. The way that checkWrongDirections was implemented introduced a break in the control flow that prevented this. Since removing this, the timer should be continuous.

At the moment, the robots will always dump, even if they are performing a useless run. I found a Boolean variable called justSurveying that is not being used for anything in DumpingMode(). This works well. I have renamed justSurveying to uselessRun.

Tomorrow work on getting several robots running with useless runs possible and begin digging into the Lorenz stuff.

What is Lorenz behavior? Right now, if there is a robot which dumps and Lorenz is on, it automatically goes into resting mode. Every resting period, the robot will go digging based on some probability.

Instead, should we roll first to decide if we go dig or go rest?

**Date: January 5, 2017**

Questions for Will:

* In reversal behavior, does the robot just touch the tunnel and turn around?
  + Will suggested adding a bit of a pause for all cases
* What is Lorenz behavior? Right now, if there is a robot which dumps and Lorenz

is on, it automatically goes into resting mode. Every resting period, the robot will go digging based on some probability.

* Instead, should we roll first to decide if we go dig or go rest?

RandomGenerator.h has some enable\_\*Mode() stuff in it that should probably be removed.

Need to update software on Fio, as the MasterSlaveProtocol.h file on the Fio was not up to date with that on the Due.

For the time being, there is not a LorenzMode-method. Just a check-loop that can be turned on or off within the goingIn setup. This is sufficient for now, but if we want the behavior to become more complex, such as incorporating charging behavior, then it will probably be smarter to break the Lorenz stuff out into a separate mode.

So I started implementing the Lorenz behavior, and when the robot “wakes up” it would go into goingIn mode and then immediately go into digging mode. I checked the readout from the mag and it was reading the max value. I think I have found the solution, which is a re-FSensor.setup() call. I am guessing this will be necessary for all of the devices on the I2C communication.

Wrote a wakeUp method that takes care of turning the relay back on an re-establishing comms with all of the I2C devices.

The PID on Alpha is a little weird. Trying to figure out if it is a code thing or an Alpha issue.

Code delay issue. The contacts loop can take up to ~500 ms, which is a pretty long time. Trying to incorporate FollowLane into the contact loop. *There is a 100 ms delay in there. Maybe we could make it call FollowLane for that period of time?*