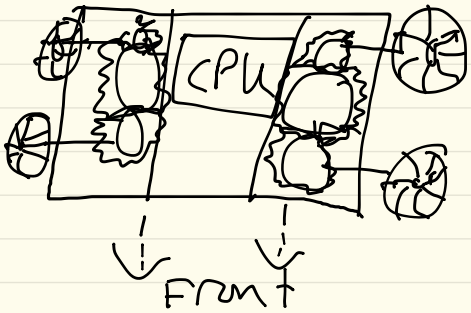
Design Notebook

Square Bot Day 1

01/20/2015

- We pulled out the major parts, but left the screws in our kit
- Simple assembly is just that... simple
- I like making my robot asthetically pleasing, but I have been told that "it's a robot. Function always trumps form."
- we created the outer and inner chasis first.
- then, we assembled the motors & connected them through the inner chasis to the gears
- the gears are connected to the wheels through the outer chasis.
- our group dynamic is great! Nitkin and Sin (?) are both very skilled & think well on their feet. There is no power struggle or disagreement.
- I think the robot will look something like this:

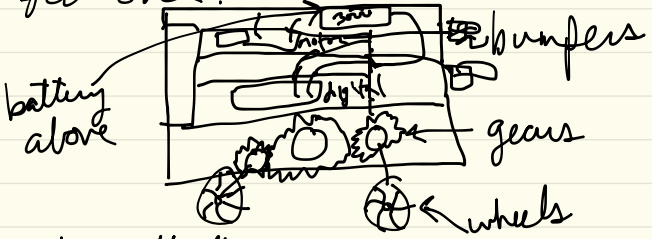


- the collars don't seem to be very effective
- step 5: Chasis subassembly with the screws is so frustrating

Square Bot Day 2

01/22/2015

- Since the chasis was already assembled, we simply had to assemble the canopy and wire the robot.
- The instructions became a bit confusing because we were not using the vex rechargeable battery.
- After assembly, our robot went crazy and started panic driving due to incorrect wiring
  - The instructions mentioned plugging the left & right motors into ports 2 & 3, but they needed to be in ports 1 & 10
- The teleoperation & autonomous mode steps were very easy.
- it's like giving life to metal
- the screws were so hard to install!
- I felt the instructions should have been reordered.
  - we should have plugged in the motors prior to the upper balcony
- A lot of the rules for building these robots seem to be logical & self-learned
  - e.g. using spacers & where to put screws & bringing in rubber bands
- when enabling autonomous mode, we lose all teleoperation
- the breaker serves as the logical switch for the bumpers
  - in the 2nd class demo, we were shown a robot that worked with a remote but could operate autonomously when it fell over.



- Rather easy since all the programs are predefined.

Microcontroller Programming Day 1

01/27/15

- Our goal this week is to construct the programs used to allow autonomous operation of the Squarebot
- No hardware this week, only code
- RobotC uses C programming language
- C is an object-oriented language that works much the same way as Java
  - in C, you use writeDebugStream() to print to the verbose debug log.
    - writeDebugStream() takes any value, but they must be specified with a type first
      - ↳ e.g. ("%d", 5) ("%s", "happy")
- We started by learning about C & connecting our robot w/ the program
- The first assignment is to write a function int hailstone(int n) that returns the next integer in the hailstone sequence
  - ↳ this sequence always returns to one (not proven)

$$f(n)\begin{cases} n_{i+1}=n_i/2 & \text{if } n_i \text{ is even} \\ n_{i+1}=3n_i+1 & \text{if } n_i \text{ is odd} \end{cases}$$

- This function was easily defined. We used if and else if statements.
- I had to rewrite the code a couple of times
  - ↳ 1) to get the sequence and recursion working
    2) to put a line between each element
    3) To properly run hailstone from the main method
    4) optimize the code & move writeDebugStream("%d \n", n); to the top of the code to run before the if statement
- RobotC programs need to wait upon running to account for the time needed by the firmware

Microcontroller Programming Day 1 (cont...) 01/27/15

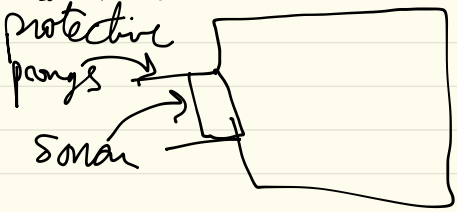
- The second assignment is to write a program that adjusts the speed of the motor to the amount of light received by the light sensor
- the actual definition of the program was made for easier by the example code given to us. The light sensor is imported via a #pragma config(Sensor, in1, lit, sensorReflective) command
- the rest works just like any OO programming language.
- the real trick is getting the motor to scale with the light.
  - The range of the sensor is 1023
  - The range of the motor is 127
  - Thus the scale is the range of motor / the range of sensor
    - in this case we divide the sensor_value by 8.
- However, we still had the motor running indirectly proportional to the light
  - ↳ this is because the darkest value is ~1023 while the slowest value is 0 for the motor
- We thus had to reverse the polarity of the light sensor. Because I noticed the light sensor going higher than 1023, I subtracted the value from 1100, but theoretically it should just be the max value.

  max value − SensorValue(lit)

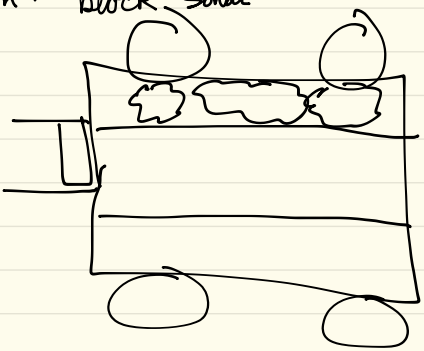
Microcontroller Programming Day 2

01/29/19

- today we have to add a sonar sensor
- For the design, we removed the bump sensors to make room for the sonar sensor
- We added two metal bars on each side of the sensor to protect it running into a wall



- next we started programming.
- I added the sensor to dgtl 7, so we imported the sensor to dgtl 7.
- we then encountered major problems!
- the robot wouldn't function when connected to the computer
- no idea what problem
  - ↳ could be debugger stopping the motor)
  - ↳ "habitual lean" - AI
- finally got it working
- Sensor checks mm distance
- So we used if statement with (sensor_value < 150)
- ends up working!
- metal prongs don't block sonar



Sensor Characteristics (Day 1)

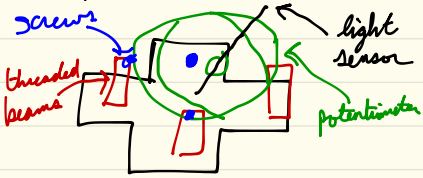
02/03/2015

1. First assignment requires a light sensor to scan for the greatest source of light & turn in that direction.
- We started off trying to get the entire robot to turn & save variables for each incremented step.
- huge margin of error
- Switched to servomotor & 2nd light sensor.
- Misunderstood that whole robot doesn't need to move. Only light sensor needs to move
- Once I understood the documentation for the Servo motor, it became quite easy
- Servo motors work under the range of [-127, 127].
  ↳ Better than other motors because it remembers & can go to specific location
- implemented a for loop w/ 2 variables: one for the light value, one for motor value
- Stopped working - unsure of hardware vs. software issue
- ended up needing += instead of + in for loop
- still not working
- needed to make it wait after moving back to highest light source
- didn't wait a full second - something to think about

Sensor Characteristics (Day 2)

02/05/2015

- Assignment 2 requires potentiometer mounting
- Very tight fit on the shaft
  - ↳ good, because you only want internal gears to move
- need to calibrate the sensor.
- changed up our design & put 3 threaded beams to allow more places for mounting



- potentiometer readings
  - ↳ (1500, 3900)
  - ↳ 2500 is middle
  - ↳ 1 degree
  - ↳ . . 180 degrees

measurements - close

45° - 1754
90 = 2600
135° = 3550

measurements - far

45° = 1750
90° = 2440
135° = 3800°

error propagation

angle	real	error
45° = 2613.9 =	1754	12.8%
90° = 2660.1 =	2600	2.26%
135° = 3306.6 =	3850	-7.36%

measurements - 1 in. - close

45° = 2613.9 =	1750	13.09%
90° = 2660.1 =	2440	8.27%
135° = 3306.6 =	3800	-14.92%

45° = 1750
90° = 2960
135° = 3260

45° = 2613.9 =	1758	13.69%
90° = 2660.1 =	2560	3.76%
135° = 3306.6 =	3260	1.41%

measurements - 1 in - far

45° = 1735
90° = 2330
135° = 3245

Sensor Characteristics (Day 3)

02/10/2015

45° = 2613.9 = 1735 13.83%
90° = 2660.1 = 2330 12.41%
135° = 3366.6 = 3245 1.86%

45° = 2613.9 = 1775 11.84%
90° = 2660.1 = 2515 5.45%
135° = 3366.6 = 3320 -.40

45° = 2613.9 = 1800 10.60%
90° = 2660.1 = 2720 -2.29%
135° = 3366.6 = 3339 -0.98%

measurements - 2in - close

45° = 1775
90° = 2515
135° = 3320

measurements - 2in - far

45° = 1800
90° = 2720
135° = 3339

- Today, catastrophe struck ...
- One group member didn't show (main hardware guy)
- We found out that we had incorrectly completed 2a. Thus, our theoretical data for 2b & 2c was off.
- We are back at 2a on the 4th day of work (including Friday)
- For 2a, we need to measure the pot readings compared to degrees at certain points given via the Servo motor.
- We took 10 readings, from -180 to 100, & plotted these points in excel
- We came up w/ a line of best fit

$y = .0696x - 95.142$

- This equation gives us an empirical formula for calculating pot results from degrees
- Hence, using our measurements from last Thursday we can show which aperture provides the least error.

degrees	servo	pot
140°	-100	3471
130°	-80	3258
118.5°	-60	3043
107°	-40	2860
95	-20	2662
85°	0	2467
82°	20	2466
74°	40	2462
64°	60	2358
50	80	2162
37	100	1955

Line of best fit = $y = -0.0696x + 95.142$

Sensor Characteristics (Day 2)

02/12/2015

- Finally onto part 3
- need to find how many sec. of turning = how many degrees
- once we calibrate, we can easily code the turn based on light

650 ms per 90°

left turn = L: 0 R: 127

right turn = L: -127 R: 0

- implemented code w/ simple if conditionals

Sensor Characteristics (Day 3)

02/13/2015

- Last day of lab
- I had a brainwave last night:

  instead of using half a dozen 'if' statements, I could use multiplication to scale the time the robot turns to the number of degrees

- At first, I tried 650/127

  650 = time for 90° turn; 127 = total motion of servo motor on one side

- However, this was horribly inaccurate since the servo readings do not relate well to degrees
- Thus, I tried 375/45

  375 = half the time for 90° turn & 45° turn; 45 = degrees

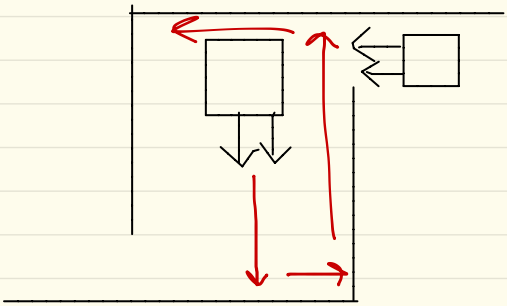
- This proved far more accurate
- In addition, I used an if statement to check if the light source is directly ahead of the light sensor. If so, the robot drives straight forward at full speed for 3 seconds.

- My first else case was if the robot sensed light to its right (from the back)
  ↳ if so, the robot moved (375/45) * the servo value to the right
- The last else case covered if the light was on the left side & functioned just like the right case, except it turned to the left.
- After coding, we ran into major issues.
- Thanks to the little tumble our robot took yesterday, our left motor clutch was no longer working. The end had been mangled.
- Hence, I removed the left side of the robot & replaced the clutch.
- After reassembly, the robot functioned properly.
- I had to adjust the first 'if' case to accept a bigger area as the "front" of the robot.

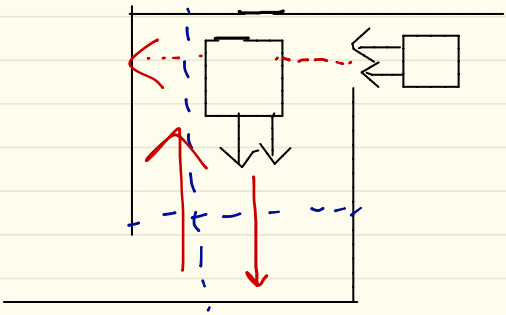
Maze Traversal

02/17/2015

- We were in lecture for most of today.
- Our task is to create a robot capable of traversing a random maze.
- There will be a light at the end of the maze.
- Perhaps I can install a bigger rod so that my current light sensor can see over the walls & find the exit...
  - ↳ I don't see how I will be able to get to my location in this manner. Perhaps this can serve as a way to choose the best option in the maze
- I was also thinking of a simple bump sensor setup. I could have the robot drive straight & automatically turn left when it bumps into a wall. & if the robot bumps into another wall soon after, then it makes 2 right turns
  - ↳ this won't work well if the maze is long
  - ↳ We have devised times for perfect left & right turns.
  - ↳ this idea would get stuck in a maze like this:



Without the double right turn, the robot will be stuck in a left turn loop



If the distances depicted by the blue lines are equal then the robot will make 2 right turns & fail

Maze Traversal

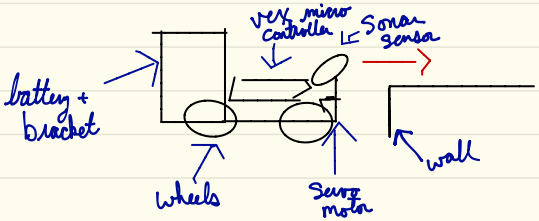
02/19/2015

- Our first day of real robot building
- we are allowed to completely reconfigure our robot.
  - ↳ I think we will keep the original squarebot design & just add the required sensors.
  - ↳ Don't fix what isn't broken.
- Our group has decided to add bumpers & use a sonar sensor attached to the servo motor from the last assignment
- We need the robot the scan how close it is to a wall and then turn if it is too close. The bump sensors are just a precaution.
- If scan all the angles in front of the robot, then we can travel in that direction for a set amount of time, or until we hit a wall. Then we can scan again.
- That's a lot of code!
- We need a scanner in a while loop.
  - ↳ We can use while(true).
- After a scan, we save the smallest servo angle to a variable & move the servo in that direction
- I have no clue how to get the robot in the same direction as the servo...
- I'll think about it tomorrow.

Maze Traversal

02/24/2015

- We're into the second week of robot building.
- I like that I have so much freedom with this project, but I wish I had a better direction.
- Our servo range - finder idea is not going to work out properly.
- I need a different idea. I looked up some robot projects online & I could only find maze-navigating algorithms for virtual robots. THAT DOESN'T HELP!
- I'm thinking we should scan the front & one side instead.
- We can then focus on going forward & not hitting a wall while still finding the best time to take a turn.
- If the sensor finds a wall in front & to the left, then it makes a right turn.
- If the robot finds only a wall to the front, it makes a left turn.
- I have started installing the sonar sensor onto the servo motor.
- I need a bracket to hold to sonar
- Nitin suggested the bracket
- The servo works with the sonar, but for some odd reason, it got switched to a different direction
- I couldn't figure out how to properly adjust the hardware so, I just set an adjustment for the left & front values in the code.
- Works well!
- I spoke too soon, it is peaking over the wall
- I will fix it on Thursday



Maze Traversal

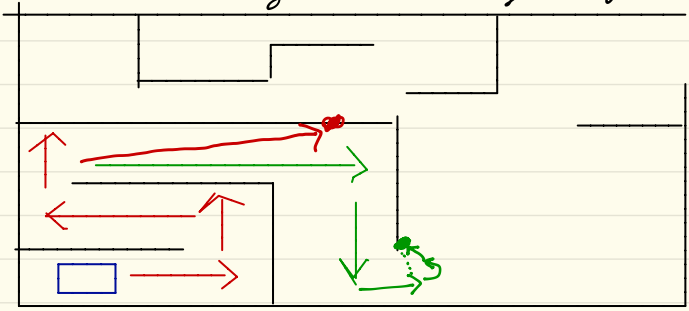
02/26/2015

- My tasks for today are to find a way to reduce the height of the sonar & finish the code for the robot
- Nitin is working on the height. I am fixing / writing the code.
- My current code has all of the import statements, followed by the main task, followed by a while(true) loop.
- Inside the while(true) loop, I have my sonar scanning the left side first
  - ↳ I move my servo to the left (exaggerated due to the weird hardware config).
  - ↳ If the value is less than 300 or so, I move on to the next loop by moving the servo to the front.
- The front loop checks if the robot is <300 from the wall.
  - ↳ if true, it turns right
  - ↳ if false, it moves outside the loop & checks if the front is <300.
    - ↳ if so, it goes left.
    - ↳ else, it goes straight
- Nitin fixed the height issue by removing some washers, but now the robot will fall apart if held upside down. Not good..
- We tested the code & immediately failed
  - ↳ we keep running into walls.
  - ↳ added a backup to the turns
  - ↳ removed from left turns
- The robot moves forward, sees a wall, backs up & turns.
- Hopefully, we'll figure out the rest next week & get the week off.

Maze Traversal

03/03/2015

- Tragedy finally struck our robot
- It won't do anything.
- We plugged it in to compile & download new code, and it stopped working.
- The result is the same whether or not it is plugged into the computer.
- We tried running start & stop on the computer & powering it on & off
- Our last resort was disconnecting the battery & charging it for a bit.
- Still not working ...
- Lo and behold, I called one of the instructors over & it started working. I hate this robot
- We are having a major tilting problem. Our robot keeps drifting left. I set all the motor commands higher for the left motor compared to the right.
- Takes forever to test.
- I have found that 64 left & 60 right makes the robot still drift left, while 65 left and 60 right makes the robot start drifting right. I can't find a midpoint.
- I tried adding weight to the right side of the robot & it worked!
- The robot actually drives straight!
- Now we're having issues with the edges hitting



My robot always finds a way to hit a wall. Sometimes it happens when it goes down long corridors, and other times it happens when making a left turn & accidentally spotting a corner. It then turns into that corner & the wall connected to it

Maze Traversal

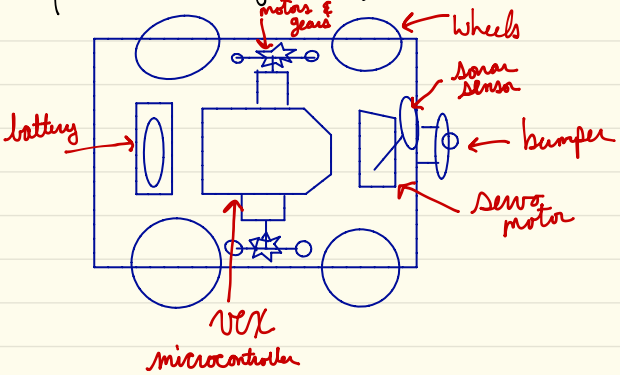
03/05/2015

- It's the last day of the 3rd week.
- I'm getting pretty nervous...
- Our robot is still struggling with corners. I don't know how to help it.
- Everyone else seems to be running their robot continuously. 2 groups are already done.
- I have no idea how they can keep running w/o scanning their distances.
- Our code is a mess as well, and I think it is causing issues.
- We have multiple while loops & if statements.
- Upon testing, we noticed the robot turns left randomly while going forward.
  - ↳ unable to diagnose the specific problem.
- We still haven't fixed the edge issue
- Sonar keeps falling off the robot
- maybe we should have worked on something simple first.
- In my 343 - Data Structures class, we talked about figuring out overarching solutions and then figuring out the smaller details
- I think I should redesign the robot & rewrite the code. But I only have 1 week.
- We're going to leave on time today & think about this over the weekend.
- I'm completely stuck

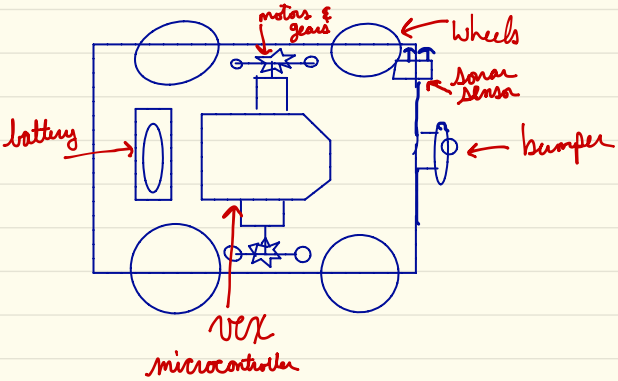
Maze Traversal

03/10/2015

- We learned the rules for the maze competition
- We have 7 minutes. Seems like more than enough time.
- My groupmates left, so it's all me.
- I have decided to take the robot apart & try to build a better robot that works w/ brand new code.
- We need a wall counter. The robot must follow the left wall all the way to the end of the maze
- It will always work!
- It will also be faster since the robot can drive until it hits a wall & then turn
- My first order of business is hardware:

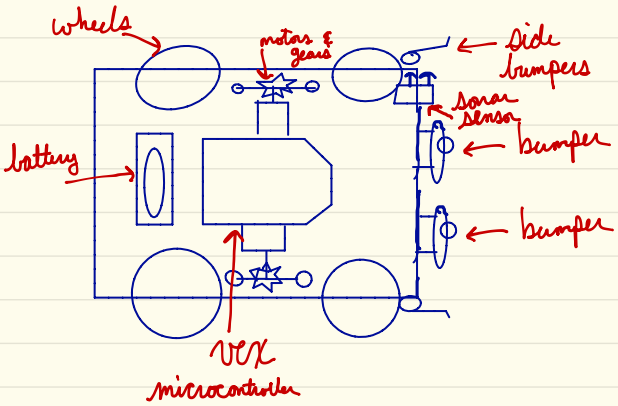


OLD



NEW

- The main change is a shift from a moving sensor to a stationary left sonar
- The code is insanely simple.
  - ↳ The inputs & left & right motor value remain
  - ↳ After the while loop, there is another while loop that deals with following the wall.
    - ↳ the conditions for the loop are that the bump sensor must not be triggered, & the sonar sensor must be <400 from the wall.
    - ↳ While the conditions are met, the robot drives forward w/ the left & right values
  - ↳ else, the robot checks if it has been bumped
    - ↳ if it has been bumped, it turns right
    - ↳ else, it turns left.
- The robot is still having issues with corners.
- I have decided to add side bumpers.
- The switches only work in one direction.
  - ↳ They don't latch on well in the opposite direction
    - ↳ I'm using lots of tape, but it doesn't hold properly.
- It keeps getting destroyed on the forward bumps.
- Attached string to keep them held. Decent Results.
- Added second front bumper to cover entire front.

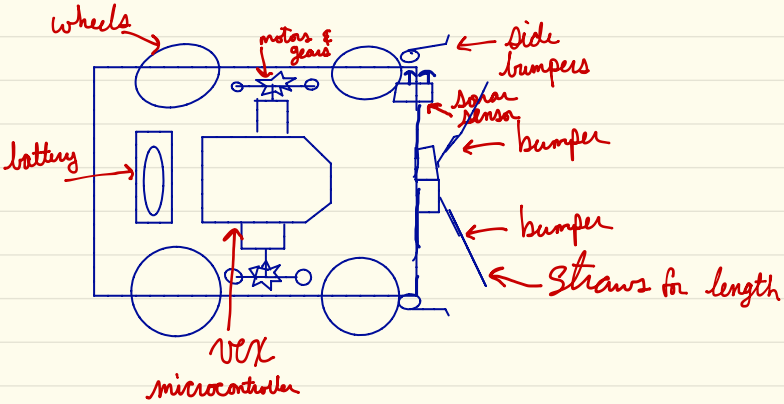


- Works almost always. Few issues w/ the edge.

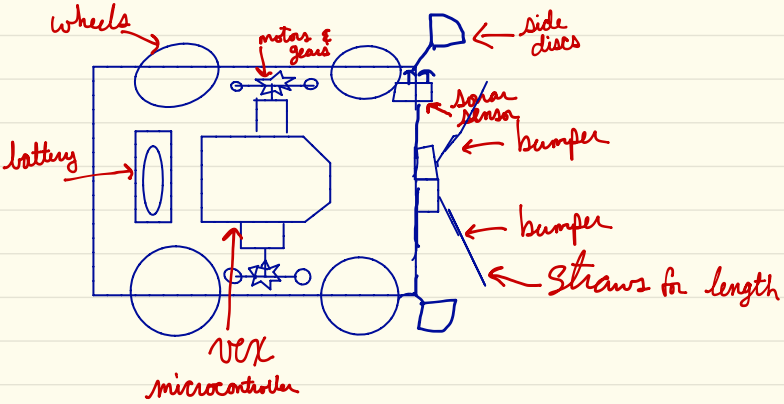
Maze Traversal

03/12/2015

- Last day for robot work!!!
- I've decided to fix my last edge issues
- I switched my 2 front bumpers for 2 front angle bumpers
  - like the ones on the side



- robot now working!
- I made it through the maze w/o adjustment
- The sensors keep getting busted
- I switched out the sensors for metal discs.

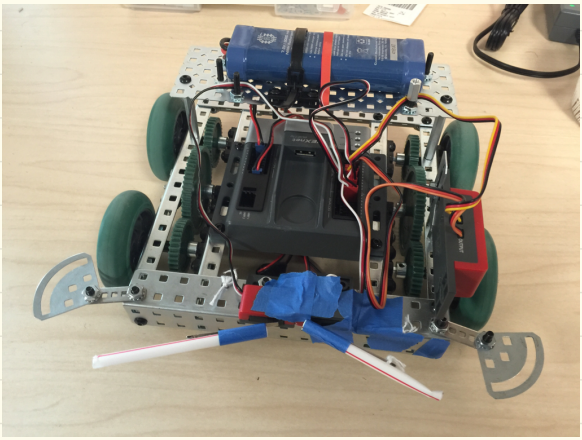


-Everything works.
-It has to be tightened a lot more so the discs don't fall back on the wheels
- Post-tightening, it's working!
- 2:05 run time w/ 21 bumps
- Ran 5 more times w/o issues.
-Time for the final run!

Maze Traversal

03/13/2015

- My robot failed today.
- There were no major adjustments to the maze
- The metal arcs kept falling back on the wheels
- It made it almost all the way through, but ran out of time
- I was very disappointed.
- I fixed it right after by adding a bracket w/ more screws



Coke Bot

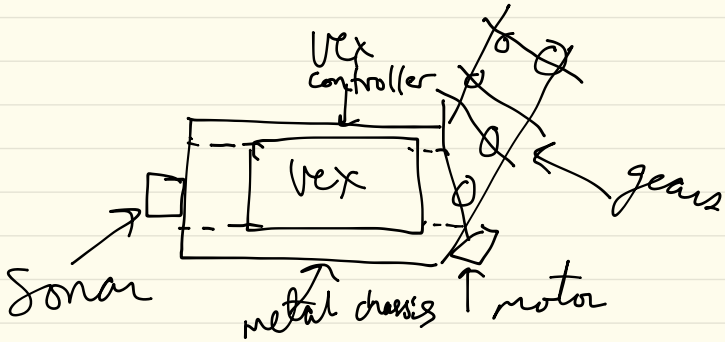
03/24/2015

- Did not get much time to work today.
- Sat through long lecture. Very confused as to where we will need to use cos formula
- We will start tomorrow by building a new base.

Coke Bot

03/26/2015

- Today we are starting with a new base.
- We no longer require wheels!
- Or motors for that matter.
- I'm thinking something simple that will simply support the crane & hold the Vex plus the battery
- My group members think making a battery holder is useless. They're probably right



- an awful picture, but it shows my idea sort of.
- The group behind us is doing a massive gear setup that is entirely along the side of the robot. Not up the crane like ours.
- We have 5 weeks left, so we'll leave it here, but I won't get caught by the time like last time.
- To review: we built the base today with 3 metal pieces & the vex controller.
- 2 more rods will lead into the crane

Coke Bot

03/31/2015

Presenting in Indianapolis

Coke Bot

04/02/2015

2nd Day Of Indy Presentation

Coke Bot

04/07/2015

To-do

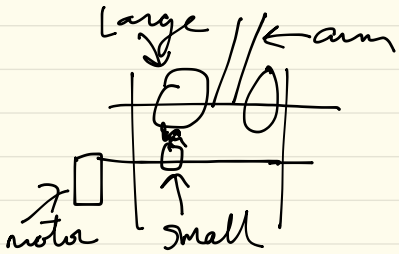
1. Add switch below the shoulder to prevent motion
2. Potentiometer / angle sensor
3. Add metal pieces to reduce noise

- While I was gone, Nitin got a fair bit done. He finished the poles for the crane & tried some gears out.
- We have yet to get our first check off
- We apparently need a hardcoded switch to stop the shoulder
  - ↳ I think soft coded is best, but Zach likes his hardware haha
- In addition, we need a pot sensor for finding angles
- Lastly, we need to reduce "noise"
  - ↳ fancy term for gear moving on its rod.
- Washers will fix that
- just finished putting on switch
- Nitin change the code to all methods
- I think Zach did it
  - ↳ looks very clean
- Added a failsafe method to prevent clawbot from smashing itself
- Nitin added washers & we got pot working
- Still no check because our tongue is apparently too low...
- I'll deal with it on Thurs.

Coke Bot

04/09/2015

- 3 weeks left
- We really need first check.

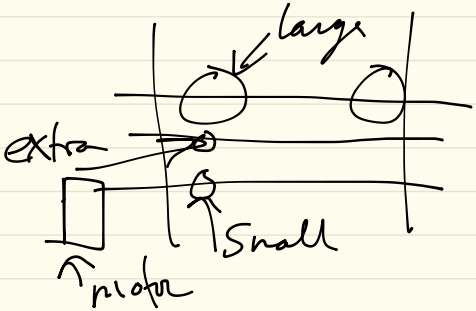


- Just read through the lecture slides.
- Apparently this is only 5x torque because of the gear ratio.
- 12-tooth : 60-tooth

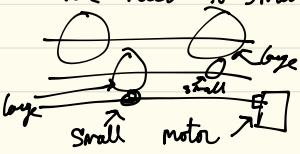
  12 : 60

  1 : 5

  5x
- So we need to add more gears...



- Shouldn't have wasted time drawing that. Turns out "idler" gears provide no torque.
- We need to make the cogs look like they do on the lecture slides.



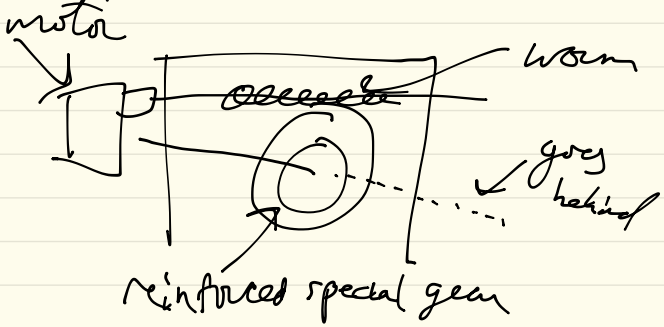
5x * 5x = 25x torque

- We got the robot to lift!
- Metal bar + tires, but those gears do not sound happy
- I'll come in tomorrow & get the last 5x attached.

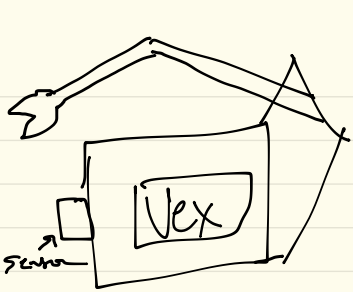
Coke Bot

04/14/2015

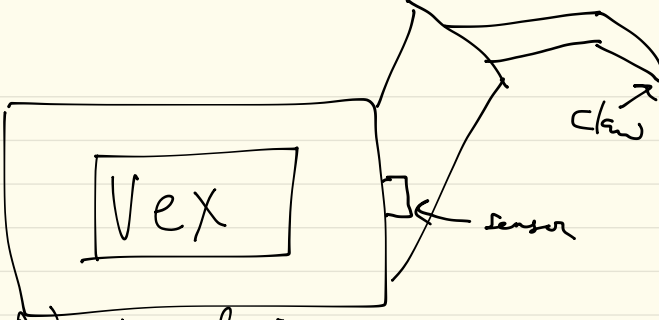
- I wasn't able to make it on Friday, but Nitin got the 125x working.
- It lifts like a pro (Ernest... hehe)
- Zach thinks we should flip it to the other side.
- I say we get the elbow working. Nitin already got the worm gear together from the photo I showed him.



- It sort of looks like this.
- The elbow is so much easier!
- Done!
- attached pot to worm gear & added a forearm
- We used multiple screws to attach elbow to the upper arm
- Ran into some issues with testing
- Shoulder is running into the gear train.
- If we're going to move gears anyways, might as well flip



Old design



New design

- Overall, a pretty easy implementation.
- Code required fixes. But easy enough.
  ↳ Just swapped negs & pos.
- Our robot keeps tipping over
- Added metal to back & books to support
- Nitin tried taping it down, but it didn't work.
- Good progress

Coke Bot

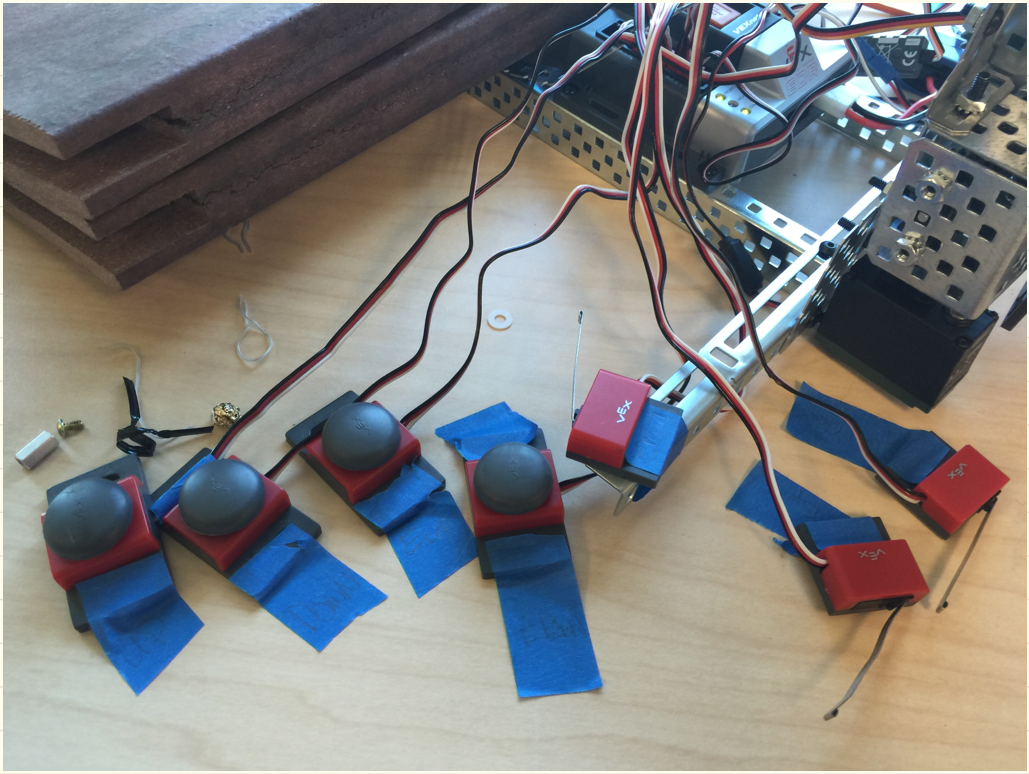
04/16/2015

Missed for Class Presentation

Coke Bot

04/21/2015

- We got part 1 & 2 checked off. Looking good. Still 2 weeks left.
- I have no idea how to go about picking up cans.
- We made & added the claw w/ a motor first, but Professor Beer recommended a servo for weight.
- Robot needs a light sensor (Sonar) on front.
- I attached w/o an issue.
- Also came up w/ genius (if I say so myself) way of moving robot around. Just implemented if statements in a method that point to bumpers. I attached the bumpers to the side of the robot with tape & now I control robot without recompiling. Just left the method inside a while(true) loop. Pretty cool.....
- Pic wouldn't fit so check next page!



- I talked to Professor Beer & I've decided to do a excel headline w/ empirical data like one of our previous projects.
- I'm not messing around w/ cos & math haha not w/ 1 week left after this week.
- I'll gather data Thurs. & finish up.

Coke Bot

04/23/2015

- An awful start today. Nitin & Ines didn't show & my robot just wouldn't work for 20 minutes! Wth!
- Ugh... I'll have to come in tomorrow and work as well. I've started adding data ↳ on next page.
- This is a pain
- SERVO BROKE! Why God!?
- Spent all day fixing. Attached a new claw that's not spring loaded. Idk if it will help at all. We'll see, I guess.
↳ Bonus, it's Maroon. Katelyn would be pleased
- Pot stopped working on elbow, but that was an easy fix.
- I need all day tomorrow.

Coke Bot

04/28/2015

- I forgot to come in & work on Friday.
- I had a massive lab due.
- Just moving & adding data today.
- *Knock on wood* no breaks today
- get this, the sonar freaks out if a can is bent. Light reverses it up or something
- Got all the data!
- Putting it into excel & graphing
- Messed up & had to regraph today
- Got my formulas!
  - ↳ Shoulder: $y = -3.6491x + 3262.7$
  - ↳ elbow: $y = -.917x + 1393.7$

  > X = Sonar value, y = pot value
- Nitin showed up.
- Put the formulas into a new method called finalPrgm(). Sounds ominous haha
- Testing...
- Bumped last can & failed. Fixed by adding if condition with (n > 436)
  ↳ if bumped out, the claw will go to same position I tested & grab can

Shoulder / Elbow Readings

pot	shoulder	elbow
200	1184	2900
209	1184	2490
221	1181	2490
229	1181	2427
243	1181	2400
251	1170	2410
264	1160	2290
272	1145	2298
284	1135	2230
293	1126	2180
303	1128	2134
313	1115	2120
331	1120	2100
343	1084	2008
356	1083	1985
366	1069	1942
380	1052	1875
395	1020	1815
411	1013	1780
421	990	1724
436	966	1640

~~elbow pot broken~~

- Success!!!
  - I'll call Zach over
  - It barely got the last 2, but we passed :) :) :)
  - Adding photos below: