Learning how to talk robot.

Katherine Scott

Computer Vision Engineer *katherine.a.scott@gmail.com* *http://www.kscottz.com*

July 15, 2013

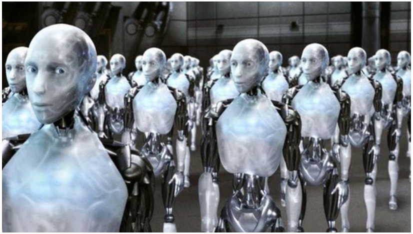
What is the most important language in robotics?

- C++?
- Java?
- Python?
- Lisp?
- Assembly?

What is the most important language in robotics?

English!

So what is the most important skills of a roboticist?

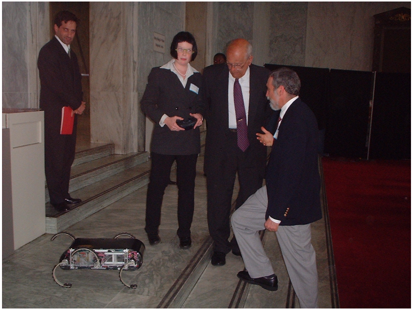


- Mechanical Engineering?
- Electrical Engineering?
- Computer Science?
- Management?
- Protecting humans from the robot forthcoming robot apocalypse?

So what is the most important skills of a roboticist?

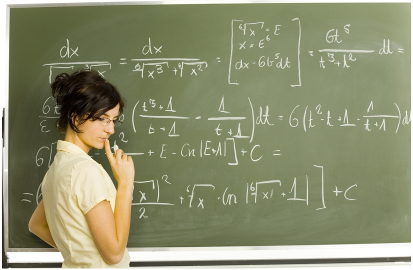
- Asking the right question in the correct way.
- Finding and reading about a solution.
- Not being afraid to give it a shot.

What I've learned.



- Words have **specific** meaning. Learn the meaning.
- With these words you can ask (google) better questions.
- These words encode scientific papers that you can read.
- You start to sound like a pro. People will respect your opinion.

What I've learned... about math

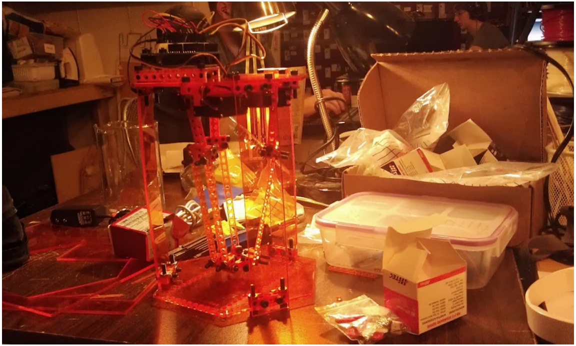


- **Learn to skim scientific papers**.
- Math is just another language. Learn the symbols to unlock the meaning.
- Remember, you don't have to do the math (proof, derivation, etc), you just need to translate it to code or English.

And another thing!

- **DO NOT PANIC**
- **RTFM** READ THE FRAKING MANUAL. Really read it. Twice.
- Break problems/solutions/papers down to the individual words, and work back up.
- Ask for help.

I brought my friend tapsterbot to help us.

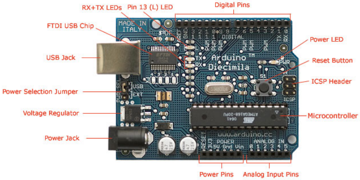


- Tapsterbot is a free and open-source parallel robot.
- These types of robots are used for sorting tasks.
- Tapsterbot is used to automatically test smart phones.
- Cheap and easy to build. Just an arduino and a few servos.

All Robots Have Three Basic Parts

- **Sensors**
  - Sense the world around the robot.
  - Just like your eyes, ears, nose, and skin.
- **Actuators**
  - Move the robot around. Motors, gears, levers, cams, etc.
  - Just like your muscles and bones.
- **Controllers**
  - Take input from sensors, reason about it, and decide what to do.
  - Just like your brain.

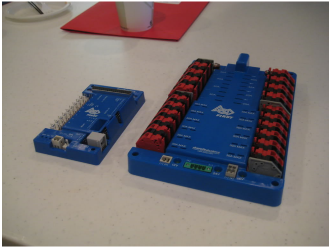
Let's look at tapsterbot



Photograph by SparkFun Electronics. Used under the Creative Commons Attribution Share-Alike 3.0 license.

- **Sensors**
  - Eventually a camera on top.
  - Each servo has an encoder.
- **Actuators**
  - Hobby servos (servos have built in sensors).
- **Controllers**
  - Arduino connected to my computer.

Other things robots usually have...

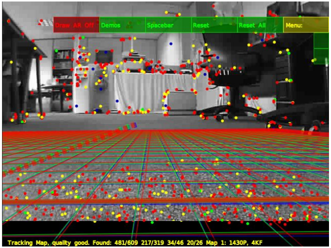


- **Power Distribution**
  - Different parts take different voltages, current but come from one battery.
- **Digital IO**
  - This board usually translates (talks) in different digital and analog formats.
- **Communications**
  - How do we control the robot remotely. Usually wifi.
- On tapsterbot the Arduino does most of this stuff.

Common Sensors

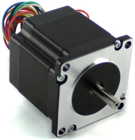
- **Encoders** - count how far something has moved (wheels).
- **Cameras** - see the world, stereo cameras give depth.
- **LIDAR** - Laser RADAR high fidelity 2D/3D maps.
- **Limit Switch** - Just a switch. Off or On.
- **Accelerometer** - Measures motion, can find gravity (down).
- **Gyroscope** - Measure rotation.
- **Magnetometers** - Can find North, metal stuff.

Sensor Concepts



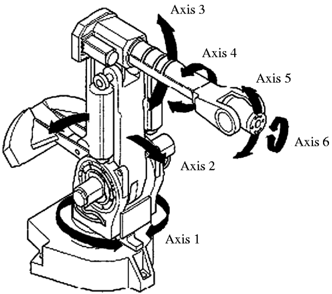
- **SLAM** - imultaneous localization and mapping. Where am I?
- **Pose Tracking** - Figure out x,y,z location and orientation.
- **Sample Rate** - How fast? Measured in hertz (Hz).
- **State** - What is the current pose of the robot.
- **Format** - What language does the sensor talk.
- **Calibration** - Does the sensor value match the real world.

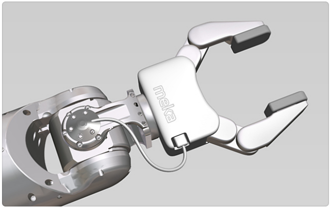
Types of Actuators



- **Things that look like motors**
  - **Motor** - A regular motor, might add an encoder.
  - **Stepper** - A motor with an encoder that let's you do precise rotation.
  - **Servo** - A motor with an encoder that turns a set number of degrees.
- **Linear Actuator** - Motor that moves in a straight line.
  - **Pneumatics** - Linear actuators that move with air.
  - **Hydraulics** - Linear actuators that move with oil or water.

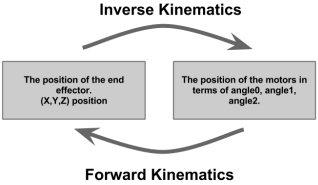
Actuator Concepts





- Robots that move around are classified by how they move.
- **Degrees of Freedom - DOF** robots are also described by the number of things that move on them.
- **End Effector** is a fancy word for a robotic hand.
- How many degrees of freedom in tapsterbot?

Controllers - This is where the magic happens.



- We use kinematic equations to relate points in the world to actuator positions.
- **Forward Kinematics** - Forward kinematics tell us where the robot will be if we move each motor a certain amount.
- **Inverse Kinematics** - Inverse Kinematics tell us where to put the motors to get the robot to a desired position.
- We often use physics and linear algebra (matrices) to figure this stuff out.

Controllers - Tapsterbot

$$\begin{cases} x^2 + (y - y_1)^2 + (z - z_1)^2 = r_e^2 \\ (x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2 = r_e^2 \\ (x - x_3)^2 + (y - y_3)^2 + (z - z_3)^2 = r_e^2 \end{cases} \Rightarrow \begin{cases} x^2 + y^2 + z^2 - 2y_1 y - 2z_1 z = r_e^2 - y_1^2 - z_1^2 & (1) \\ x^2 + y^2 + z^2 - 2x_2 x - 2y_2 y - 2z_2 z = r_e^2 - x_2^2 - y_2^2 - z_2^2 & (2) \\ x^2 + y^2 + z^2 - 2x_3 x - 2y_3 y - 2z_3 z = r_e^2 - x_3^2 - y_3^2 - z_3^2 & (3) \end{cases}$$

$$w_i = x_i^2 + y_i^2 + z_i^2$$

$$\begin{cases} x_2 x + (y_1 - y_2) y + (z_1 - z_2) z = (w_1 - w_2)/2 & (4) = (1) - (2) \\ x_3 x + (y_1 - y_3) y + (z_1 - z_3) z = (w_1 - w_3)/2 & (5) = (1) - (3) \\ (x_2 - x_3) x + (y_2 - y_3) y + (z_2 - z_3) z = (w_2 - w_3)/2 & (6) = (2) - (3) \end{cases}$$

From (4)-(5):

$$x = a_1 z + b_1 \quad (7) \qquad y = a_2 z + b_2 \quad (8)$$

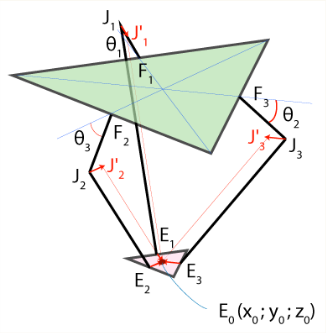
$$a_1 = \frac{1}{d}\left[(z_2 - z_1)(y_3 - y_1) - (z_3 - z_1)(y_2 - y_1)\right] \qquad a_2 = -\frac{1}{d}\left[(z_2 - z_1)x_3 - (z_3 - z_1)x_2\right]$$

$$b_1 = -\frac{1}{2d}\left[(w_2 - w_1)(y_3 - y_1) - (w_3 - w_1)(y_2 - y_1)\right] \qquad b_2 = \frac{1}{2d}\left[(w_2 - w_1)x_3 - (w_3 - w_1)x_2\right]$$

$$d = (y_2 - y_1)x_3 - (y_3 - y_1)x_2$$

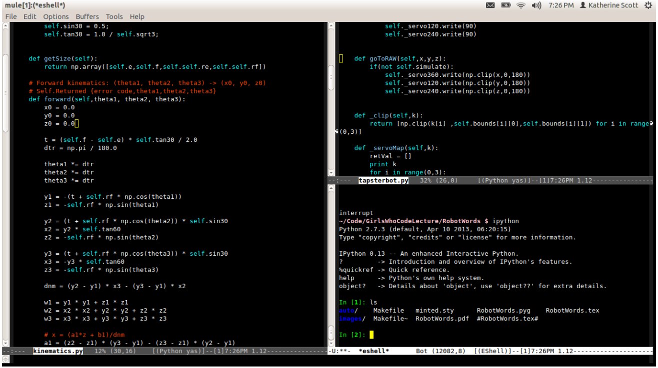
Now we can substitute (7) and (8) in (1):

$$(a_1^2 + a_2^2 + 1)z^2 + 2(a_1 + a_2(b_2 - y1) - z_1)z + (b_1^2 + (b_2 - y_1)^2 + z_1^2 - r_e{}^2) = 0$$

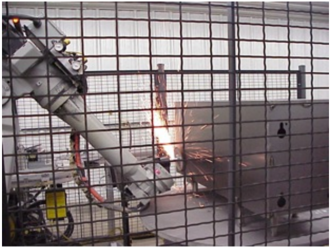


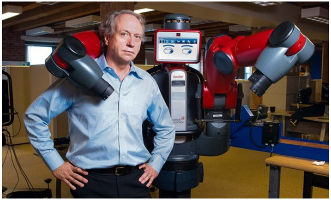
REMEMBER! DO NOT PANIC!

But let's see how that really works!



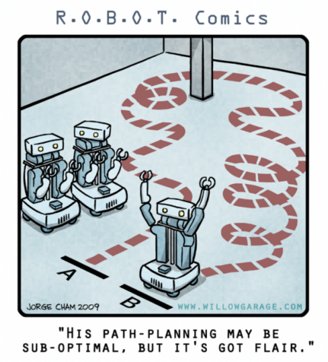
Controllers - More Fancy Words





- **Closed-Loop Control** - We move the robot a bit, we check encoders, we move again.
- **Open-Loop Control** - We just move the actuators. If they slip or we hit something too bad.
- **PID Controller** - Proportional Integral Derivative. An algorithm that uses calculus to do closed loop control.
- **Kalman Filter** a way of estimating "state" given noisy measurements.

Map Building and Path Planning



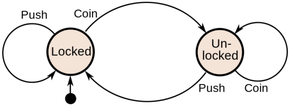
- **Mapping** is what allows a robot to plan a path. Map data comes from sensors or knowledge.
- **Path Planning** - is the general name for the algorithms that help robots go from one point to another.

Mapping / Path Planning

Roomba Path Planning

- The best way to get from A to B is not always a line. Stuff gets in your way.
- Path planning might be done to avoid “singularies” where our math does weird stuff.
- **Dead Reckoning** is a simple path planning algorithm. Basically keep a list of heading and distance traveled.
- What happens when we move one tapsterbot motor at a time versus all three at once?

High Level Behavior



A turnstile state machine,two states, two inputs.

- For beginners **finite state machines** are a good way to build up complex behaviors.
- State Machines have **states** where the robot performs one set of behaviors.
- State Machines also have **inputs** that cause **transitions** between states.
- State Machines are a great way to break up and think about problems.

Hey, Let's Write Some Python Code for Tapsterbot

Let's create three states

- **WAIT** - Do nothing.
- **DANCE** - Swing around.
- **PULL-UP** - Do some pull ups.

And tie those states to some inputs from a leap motion.

- Hand is open, let's do pull up.
- Hand is closed, let's dance.
- Hand is not there, wait

go hug a robot