CSCI 1106
Lecture 3

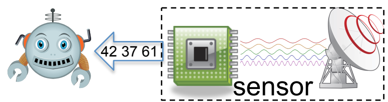
Characterizing Sensors

Announcements

- Quiz #1 is this Friday, January 18, in class
- Slides are now on our web page (http://projects.cs.dal.ca/hallab/CSCI1106)
- Today's Topics
  - What is a Sensor
  - How to Characterize a Sensor
  - Using Sensors
  - Sampling

What is a Sensor?

- A *sensor* measures physical properties in the environment
  - Sound
  - Light
  - Pressure
  - Temperature



- Convert a physical property into an electric (analog) signal and then into a digital signal.
- Have maximum and minimum values
- Values change with the physical property sometimes at a constant rate, sometimes not (response curves)
- Sensors are described by a variety of characteristics

Sensor Characteristics

- *Sensitivity* : the minimum change of input that will result in change in output
- *Range* : the minimum and maximum inputs that a sensor can handle
- *Response Time* : how quickly the sensor can change state as a result of a change of input
- *Precision* : the degree of reproducibility of the measurement
- *Accuracy* : the difference between the true and measured value
- *Bias* : the systemic error of the sensor
- *Variability* : the random deviation from the true value
- *etc*

Ultrasound Sensor

Bounces a high-pitched sound off of a surface and measures the distance as the time it takes for the sound to return

Range: 0 - 100cm

Problems:

- Assumptions?
- Sources of interference?

Sound Sensor

Measures the sound pressure (similar to sound intensity)

Range: 0-100

Problems:

- Assumptions?
- Sources of Interference?

Light Sensor

Reads light intensity. Is able to shine a light to illuminate a surface and help make better use of its range.

Range: 0-80ish

Problems:

- Assumptions?
- Sources of Interference?

Touch Sensor

With enough force, this button-like mechanism is displaced to close an electric circuit, representing a touch.

Range: > X Newtons

Problems:

- Assumptions?
- Sources of Interference?

Rotation Sensor

Converts the movement of a motor into an angle of rotation using an “optical encoder”.

Range: 0-?? (~1 unit per degree)

Problems:

- Assumptions?
- Sources of Interference?

Sensors are Imperfect

- Sensors have two kinds of errors
  - *Bias*: a systemic deviation from the true value
    - E.g., a clock that runs fast, or
    - A thermostat that thinks its warmer than it is.
  - *Variability*: random deviation from the true value
    - E.g., static on the radio and
    - Flickering low-oil sensor
- Key Ideas:
  - No matter how good a sensor is, it is imperfect
  - Imperfect sensors introduce *uncertainty*
  - Need to quantify the uncertainty
  - Need to quantify a sensor’s characteristics

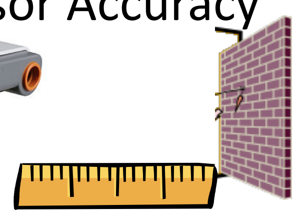
How to Characterize a Sensor

1. Identify the sensor we want to characterize
2. Identify the sensor characteristic we want to measure
3. Identify the possible variables of the characteristic
4. Fix all but one of the variables
5. Create a sequence of known ``actual'' values where the
   - One variable is varied and
   - All other variables are fixed
6. Perform a sequence of measurements (*multiple times)* on the ``actual'' values
7. Tabulate the results and compute means
8. Plot the results
9. Repeat steps 4 – 8, allowing a different variable to vary each time
10. Analyze the plot(s) to derive the sensor's characteristics

Questions:

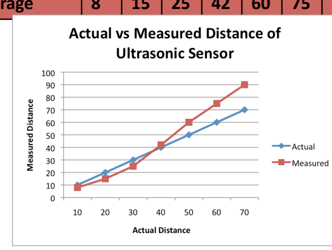
1. How do we get the "measured" values?
2. How do we get the "actual" values?
3. How do we ensure all other variables are fixed?

Example: Ultrasonic Sensor Accuracy



1. Sensor: Ultrasonic Sensor
2. Characteristic: Accuracy
3. Variables to consider:
   - Distance to target
   - Target size
   - Target material
   - Target shape
4. Fix all variables except "Distance to Target"
5. Create a sequence of known ``actual'' values
6. Perform a sequence of measurements on the ``actual'' values
7. Tabulate the results and compute means
8. Plot the results
9. Repeat steps 4 - 8
10. Analyze the plot to derive the sensor's characteristics

Actual (cm)	10	20	30	40	50	60	70
Measure 1	6	17	23	41	55	70	91
Measure 2	8	14	24	42	61	76	92
Measure 3	9	14	27	42	64	80	87
Average	**8**	**15**	**25**	**42**	**60**	**75**	**90**



Making Use of the Results

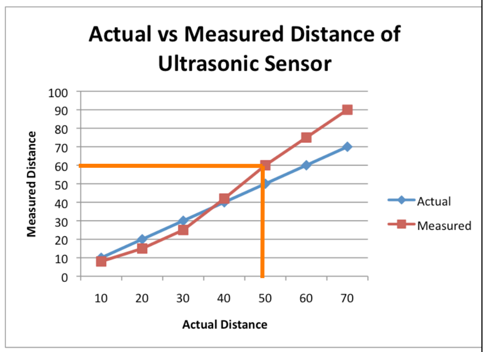
How do we use the results?

– Interpolate the plot

  Use plot as a rough guide to convert measured to actual values

– Observe that as the range increases, the accuracy decreases

  So … don't use sensor at long range



How reproducible is the result?

Using Sensors

- A program *polls* the sensor to get its current value
- It is the program's responsibility to *interpret* the value, i.e.,
  – Translate the value into a meaningful decision
- A sensor will not inform you when a property has changed
- The program must poll the sensor repeatedly to detect change
- Usually a program polls until a *threshold* is reached

Thresholds

- We are typically not interested in what the value of a measurement is.
- We are typically interested
  - when that value changes, or
  - when that value reaches a specific threshold
- For example,
  - We don't care if the car ahead of us is 50 meters away or 150 meters away.
  - We do care if
    - the car is getting closer, or
    - the car is less than 5 meters away!

Thresholds (cont.)

- A *threshold* is a fixed constant such that an event is triggered when a measurement from a sensor returns a value that is above (or below) the constant.
- E.g.,
  - Object too close: if distance < threshold, trigger event
  - Loud sound: if sound level > threshold, trigger event
  - Light/Dark threshold:
    - If light level > threshold, a light surface is detected
    - if light level < threshold, a dark surface is detected
- But … How often should we poll?

Polling Frequency

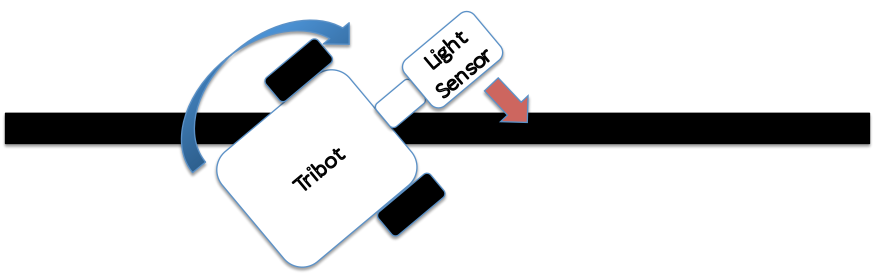
- Polling Frequency depends on
  - The response time of the sensor
  - The rate at which the environment changes
- Response time dictates the maximum useful polling rate
- The rate of change dictates the minimum rate needed to ensure that no events are missed
- Question: What if the maximum useful rate is less than the minimum required rate?

Polling Frequency vs Response Time

- Observation: There is no point in polling the sensor quickly if its response time is slow
  - Are we there yet? How about now? Now? Now?
- Polling the sensor too quickly does not hurt, but wastes CPU resources
- Our sensors have a fast response time (mostly)
- When the response time is slow, our programs need to take this into account

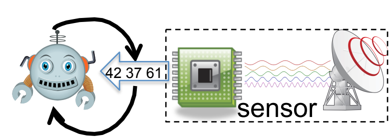
When Response Time Matters

- In Follow-The-Line
  - The angular velocity of the light sensor is quite fast
  - This could cause the sensor to move over the black line too quickly to pick it up
  - This would result in the Tribot losing the line
- How do we ensure that we don't miss the line?

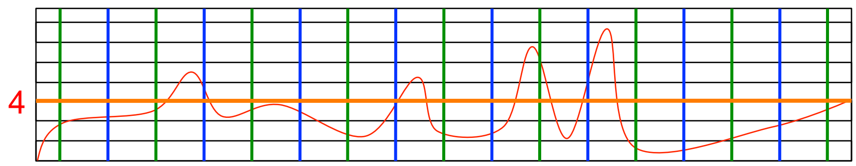


Sampling

- Sensors must be polled (sampled) for values
- The *sampling rate* is the frequency of the polls
- A higher rate means we are
  - Less likely to miss a change in inputs
  - Using more CPU time to poll the sensor
- If the rate is too high, there is no time to do anything else



Example: When is the Signal above 4?



S1 2 2.5 2.5 1.25 1.25 6 0.5 1 2.5

S2 2.25 5 2.5 4 1.25 5 0.5 2

S* 2 2.25 2.5 5 2.5 2.5 1.25 4 1.25 1.25 6 5 0.5 0.5 1 2 2.5

What If We Do Detect a Change?

- Suppose the sensor returns a different value.
- Does this mean that the environment has changed?
- Are you sure?