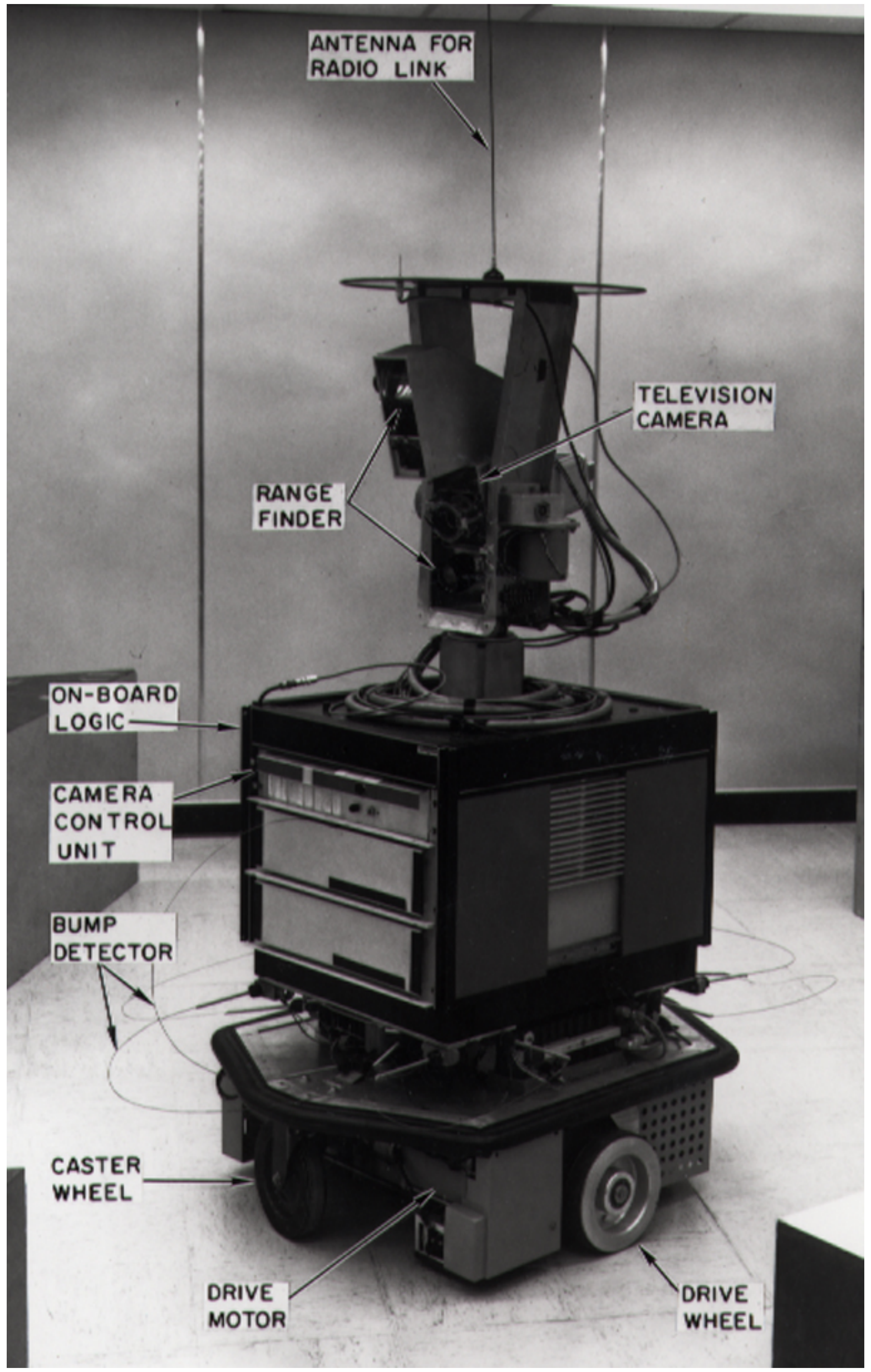
Sensors

Fast Robots, ECE4160/5160, MAE 4190/5190

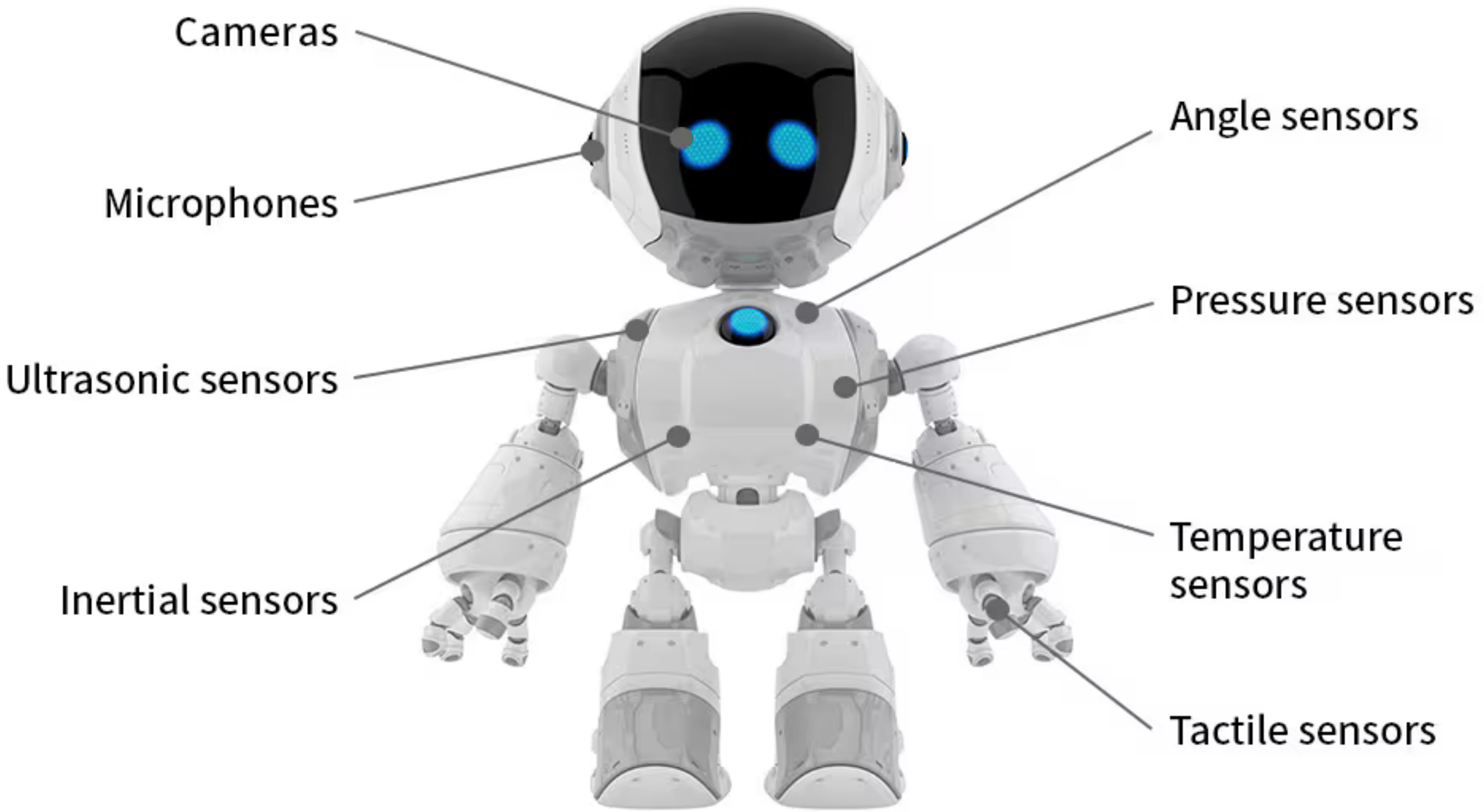
E. Farrell Helbling, 1/28/25

Sensing History



Shakey: Experiments in Robot Planning and Learning (1972), SRI

Sensor Classification



TDK Corporation

Sensor Classification

- Proprioceptive: sense robot's own state
  - Motor speed, wheel load, joint angles, battery voltage
- Exteroceptive: sense the surrounding environment
  - Distance measurements, light intensity, sound amplitude
- Passive sensors: measure ambient environmental energy
  - Temperature probes, microphones, light sensors
- Active sensors: senses reaction to emitted energy
  - Wheel quadrature encoders, ultrasonic sensors, laser rangefinders

Classification

Type	Sensor	Prop/Extero	Passive/Active
Tactile (contact/ closeness)	Contact switches, bumpers, Break beams, proximity Capacitive	Exteroceptive Exteroceptive Exteroceptive	Passive Active Both
Wheel/motor	Brush encoders Potentiometers Optical encoders Magnetic/inductive/capacitive encoders	Proprioceptive Proprioceptive Proprioceptive Proprioceptive	Passive Passive Active Active
Active ranging	Reflectivitiy sensors, ultrasonic, laser rangefinders, optical triangulation, etc.	Exteroceptive	Active
Heading	Compass Gyroscopes	Exteroceptive Proprioceptive	Passive Passive
Ground based	GPS, RF, reflective beacons	Exteroceptive	Active
Motion/speed	Doppler radar, sound	Exteroceptive	Active
Vision	CCD/CMOS	Exteroceptive	Passive

Sensor Characteristics

Name some examples

Distance Sensors

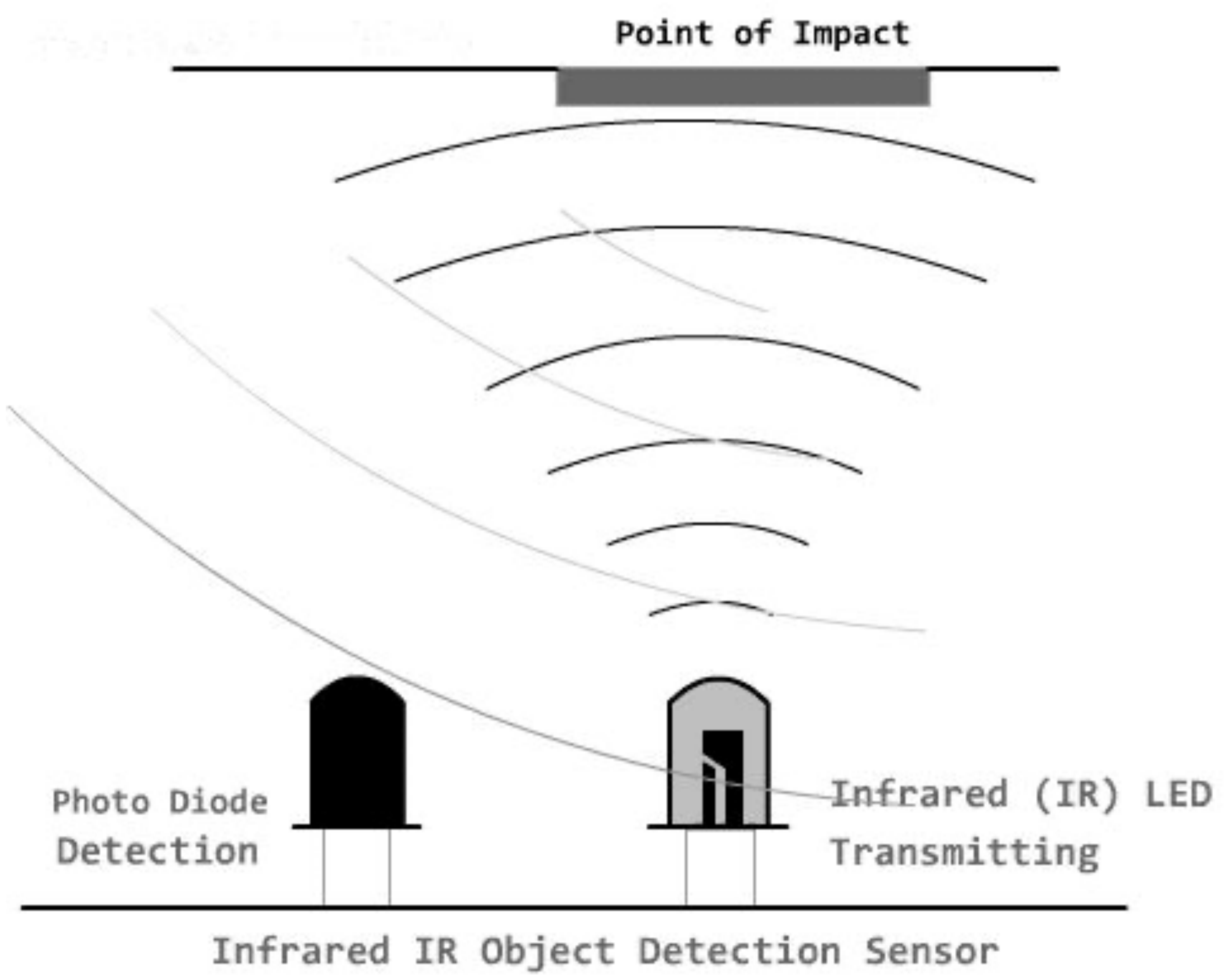
Hobby Distance Sensors

Technology	Application	Pros	Cons
Amplitude-based IR	< 10cm	• ~0.50 USD • Small form factor • High sample rate (4kHz)	• Depends on target reflectivity • Does not work in high ambient light
IR triangulation	< 1m	• Insensitive to surface color/ texture	• ~10 USD • Does not work in high ambient light • Bulky (1.75 x 0.75 x 0.53 in^3) • Low sample rate (26 Hz)
IR ToF	0.1 - 4m	• Small form factor • Insensitive to surface color/ texture/ ambient light	• ~6.50 USD • Complicated Processing • Low sample rate (7 - 30 Hz)
Ultrasonic	0.2 - 10m	• Low cost • Insensitive to surface color and ambient light • Works in rain and fog	• ~4 USD • Complicated processing • Resolution tradeoff with max range • Output depends on surface/ geometry/ humidity • Bulky, high sample time (~10s ms) • Hard to achieve a narrow FOV

The Electromagnetic Spectrum

Amplitude-based IR

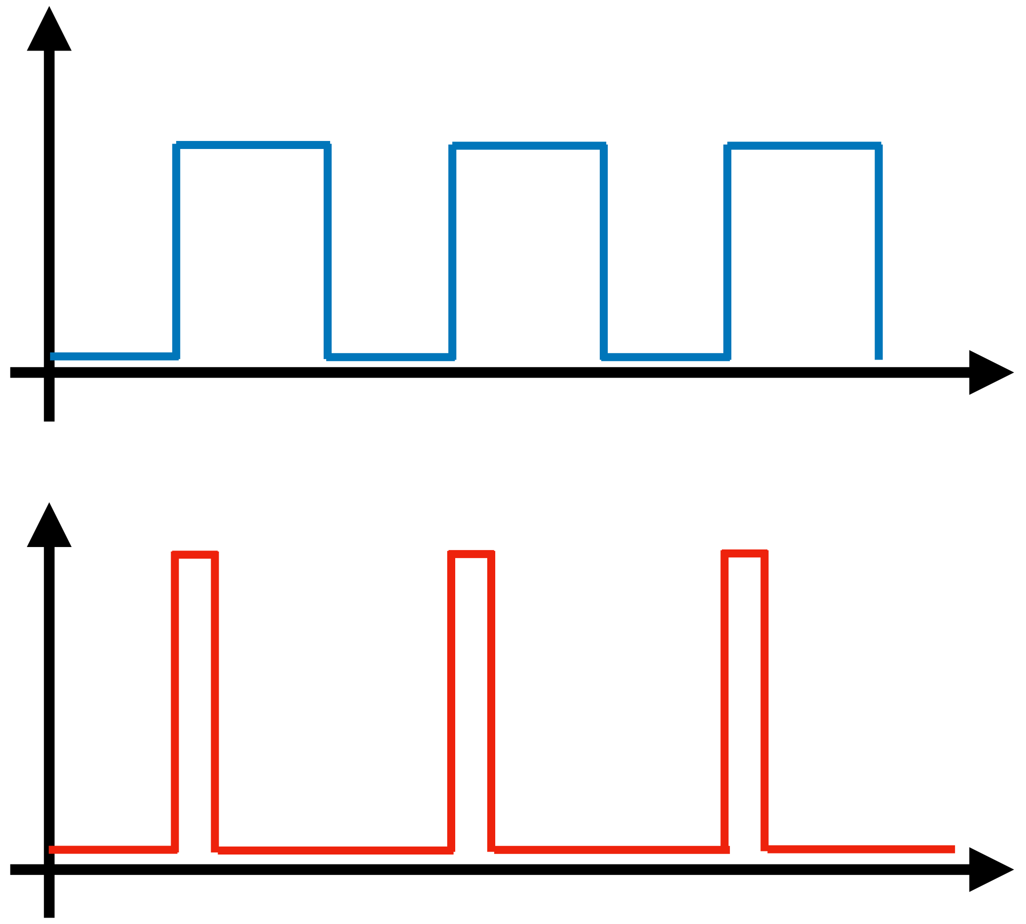
- Very cheap
- Very simple circuitry
- Works reasonably well for:
  - Object detection
  - Break beam sensors
  - Classifying grayscale intensity at a fixed distance
  - Short-range distance sensor
- Range < 0.5m
- Sensitive to surface color, texture and ambient light

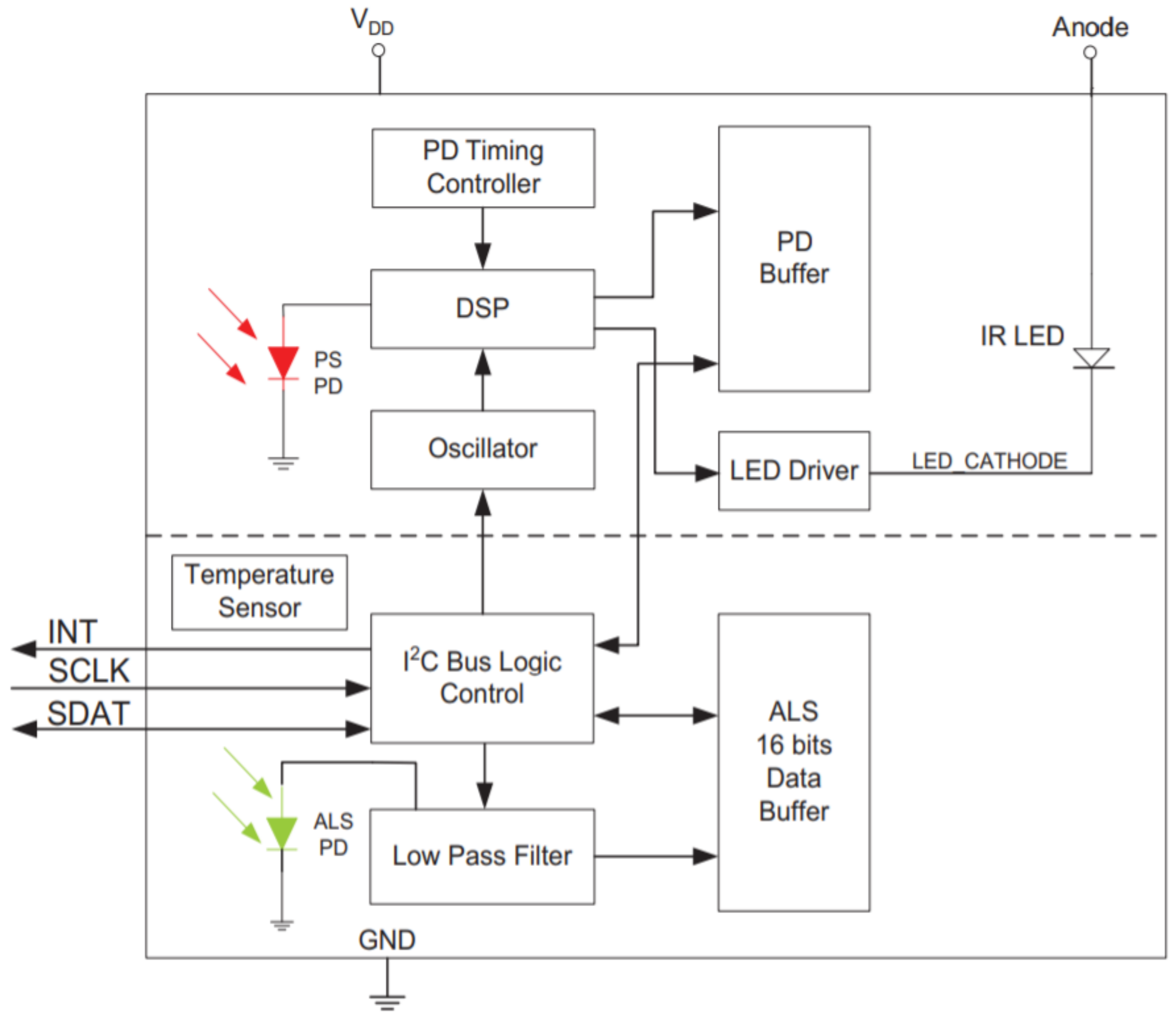


VCNL4040
- $3.34
- Range 20cm
- Ambient light sensor
- Programmable DC

Amplitude-based IR

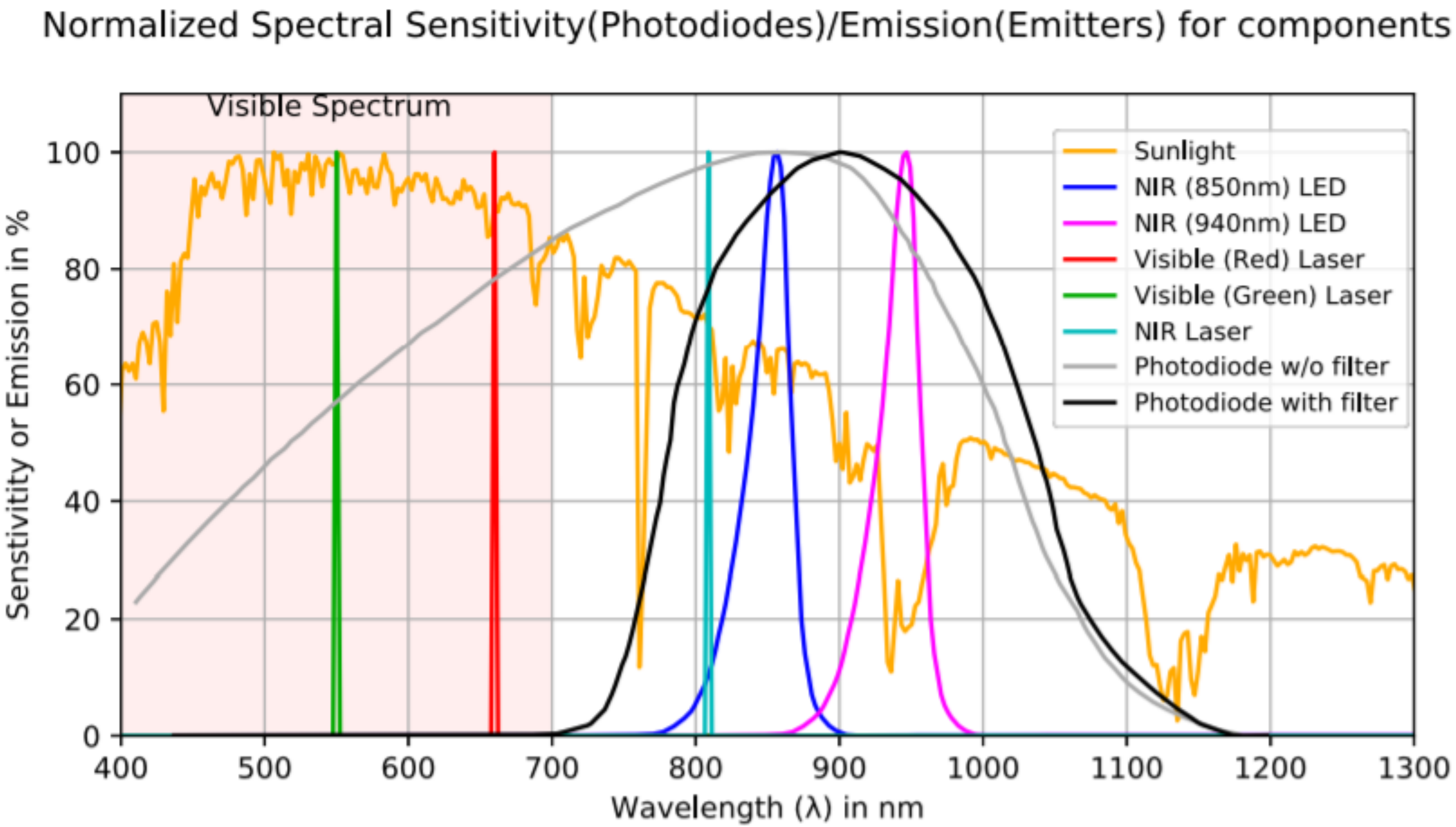
- Very cheap
- Very simple circuitry
- Works reasonably well for:
  - Object detection
  - Break beam sensors
  - Classifying grayscale intensity at a fixed distance
  - Short-range distance sensor
- Range < 0.5m
- Sensitive to surface color, texture and ambient light





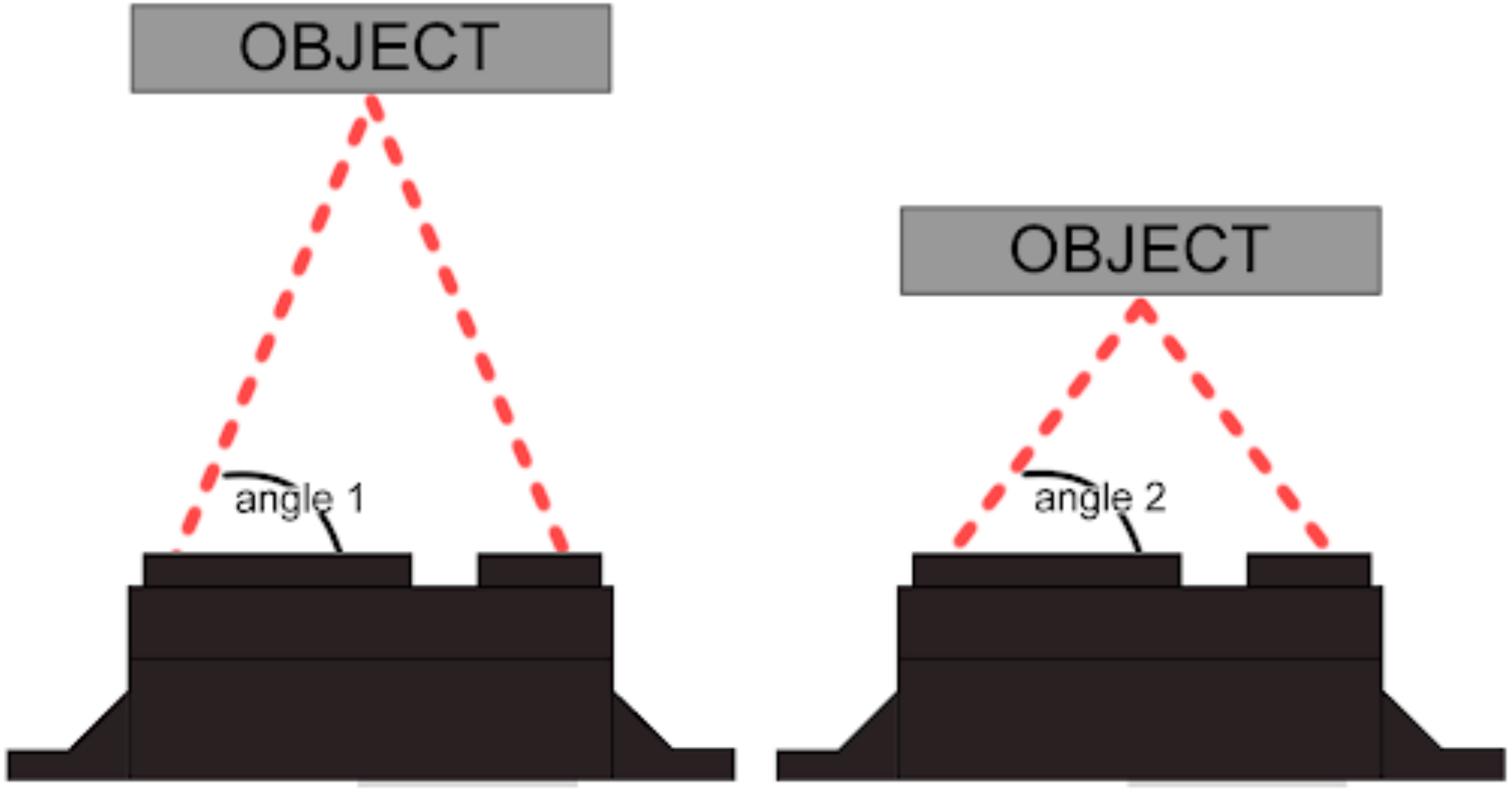
VCNL4040
- $3.34
- Range 20cm
- Ambient light sensor
- Programmable DC

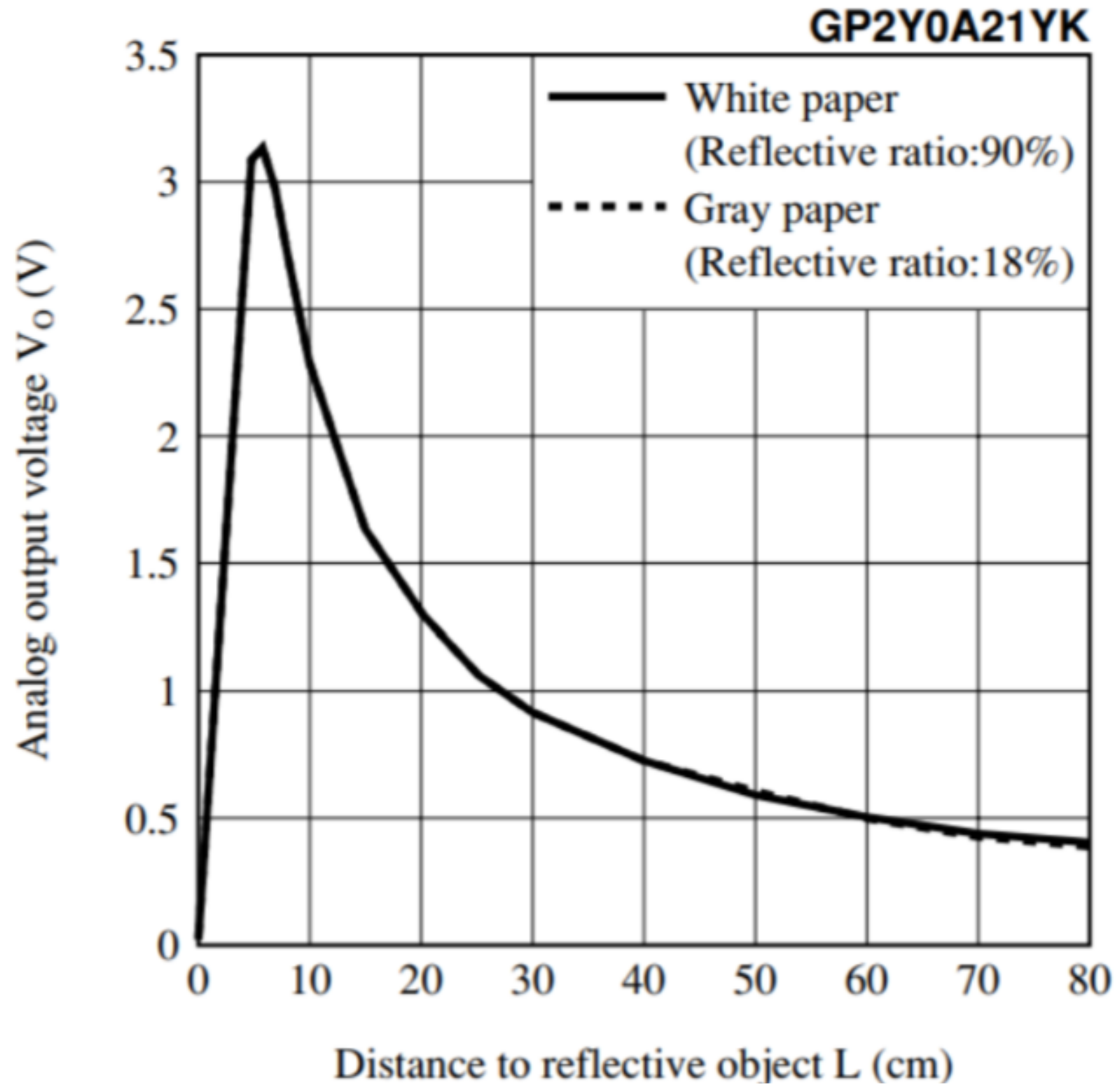
Amplitude-based IR



Triangulation-based IR

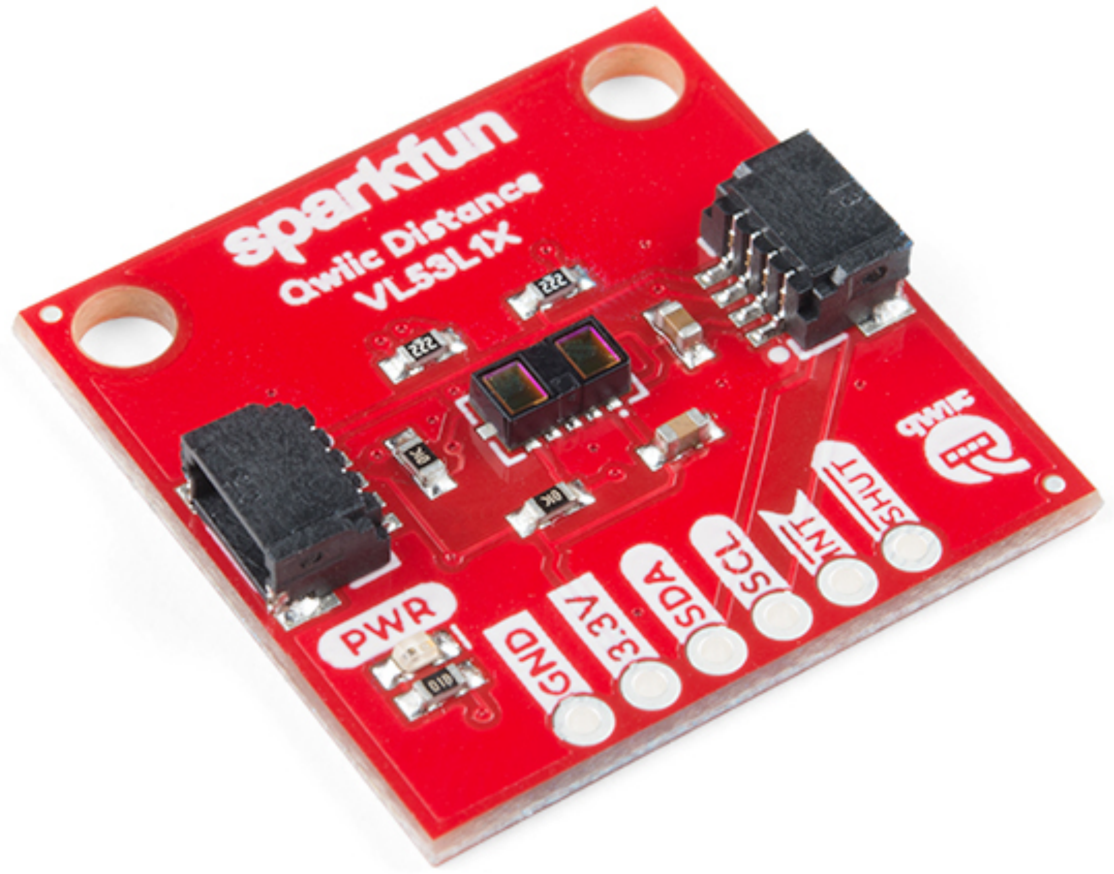
- Very simple circuitry
- Less sensitive to color, texture, ambient light
- Medium range (0.05 - 1m)

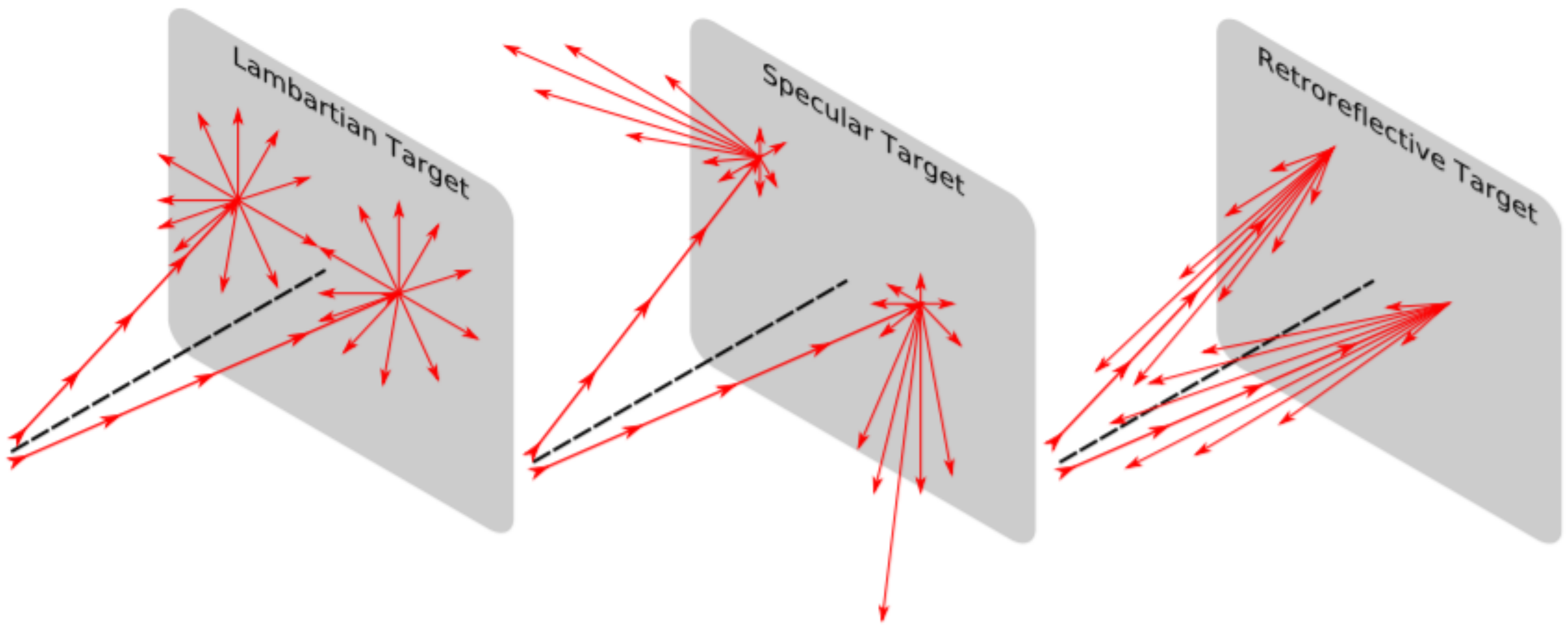


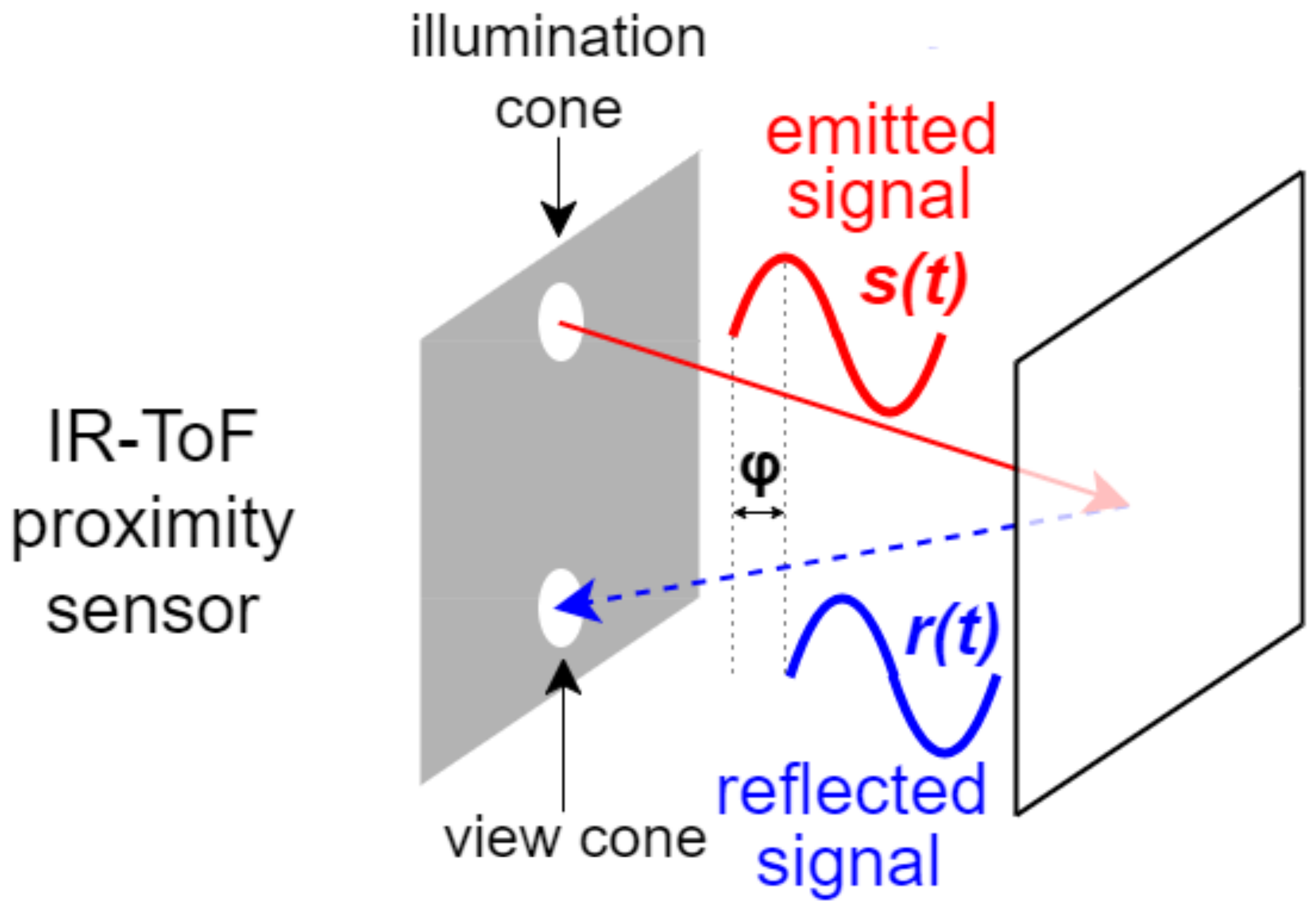


Time of Flight IR

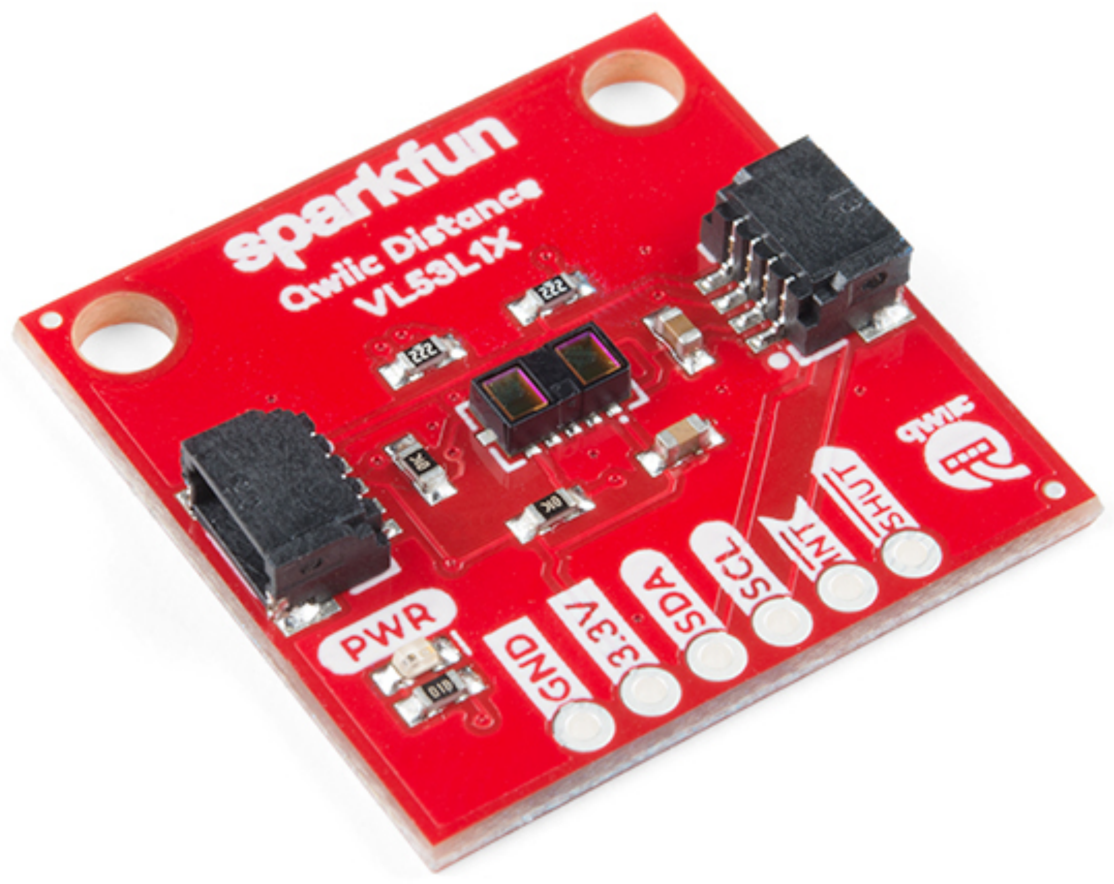
- Emits a pulse-modulated signal, record time *t* until return
  - $r = t \times c/2$, c = *speed of light*, 3×10^8 *m/s*
- Mostly insensitive to texture, color, ambient light



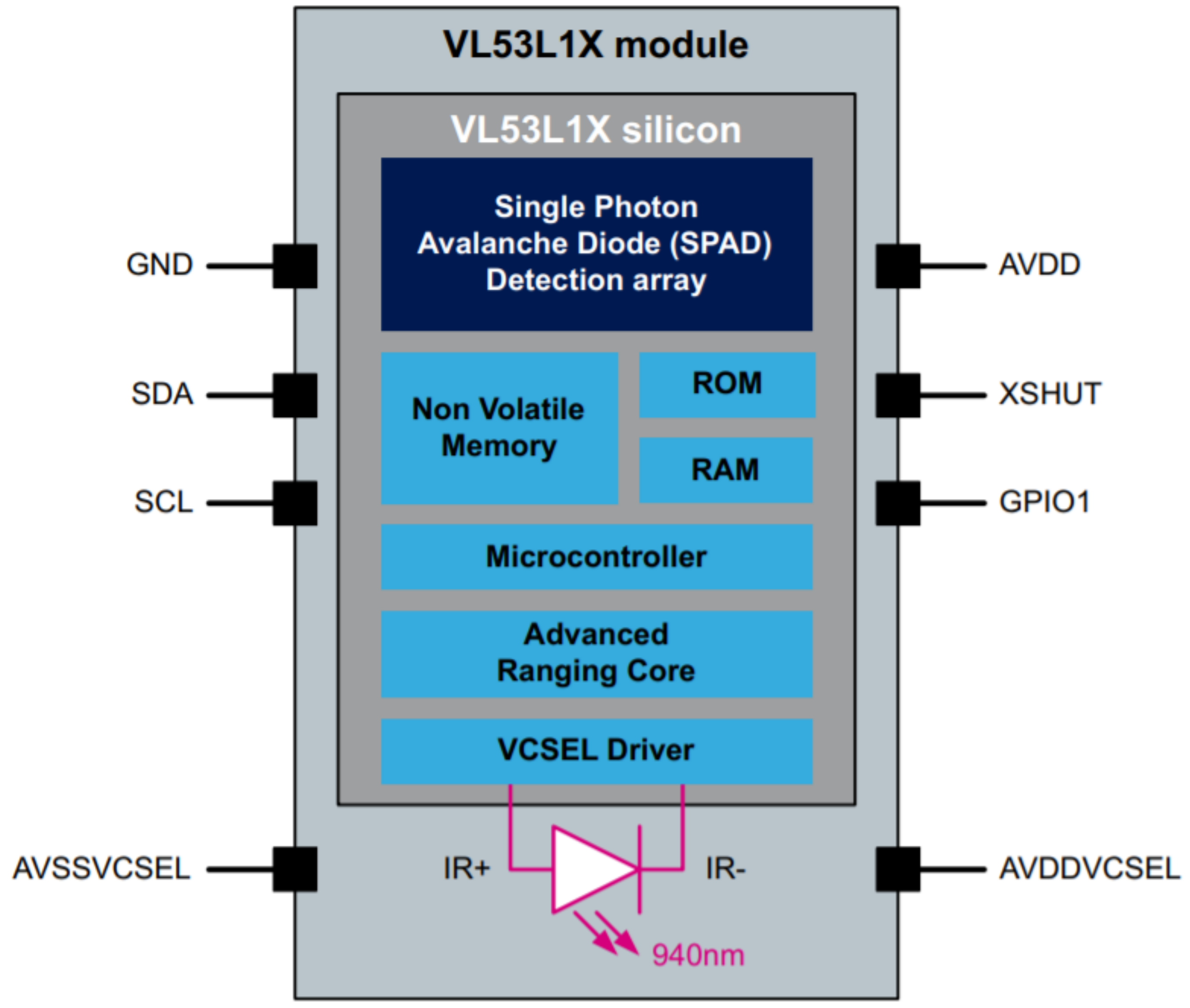


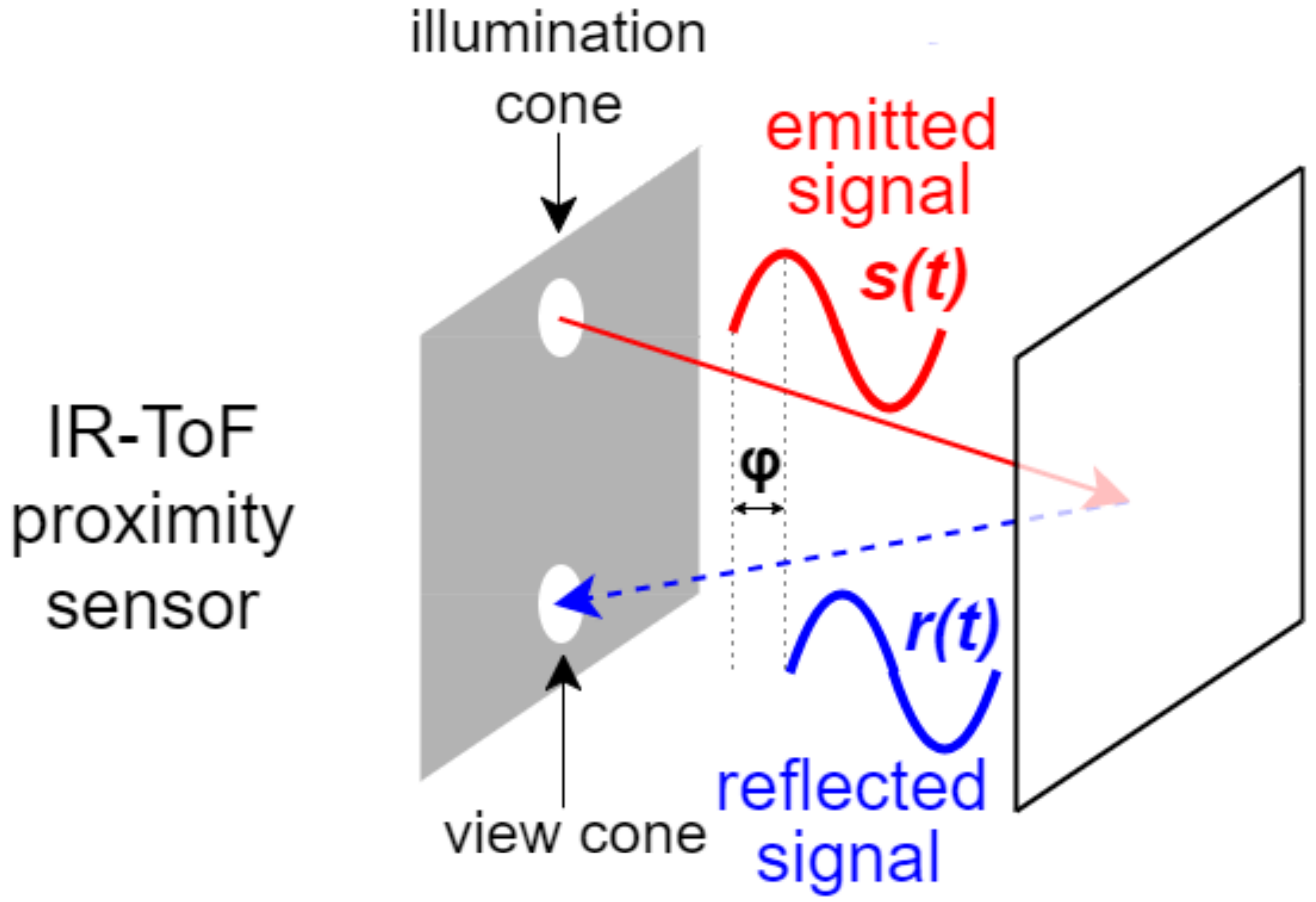


Time of Flight IR



- Emits a pulse-modulated signal, record time *t* until return
  - *$r = t \times c/2$, c = speed of light, 3×10^8 m/s*
- Mostly insensitive to texture, color, ambient light
- Outputs: distance (mm), return signal rate, ambient signal rate, range status





Time of Flight IR

- Emits a pulse-modulated signal, record time *t* until return
  - *$r = t \times c/2$, c = speed of light, 3×10^8 m/s*
- Mostly insensitive to texture, color, ambient light
- Outputs: distance (mm), return signal rate, ambient signal rate, range status
- Programmable FOV
- Distance mode (~1, 2, 4m)
- Timing budget
  - 20ms: short distance mode (0.05 - 1.3m)
  - 33ms: all distance modes (0.05 - 3.6m)
  - 140ms: improved reliability errors
- Newest developments: ToF imager (64 pixels)

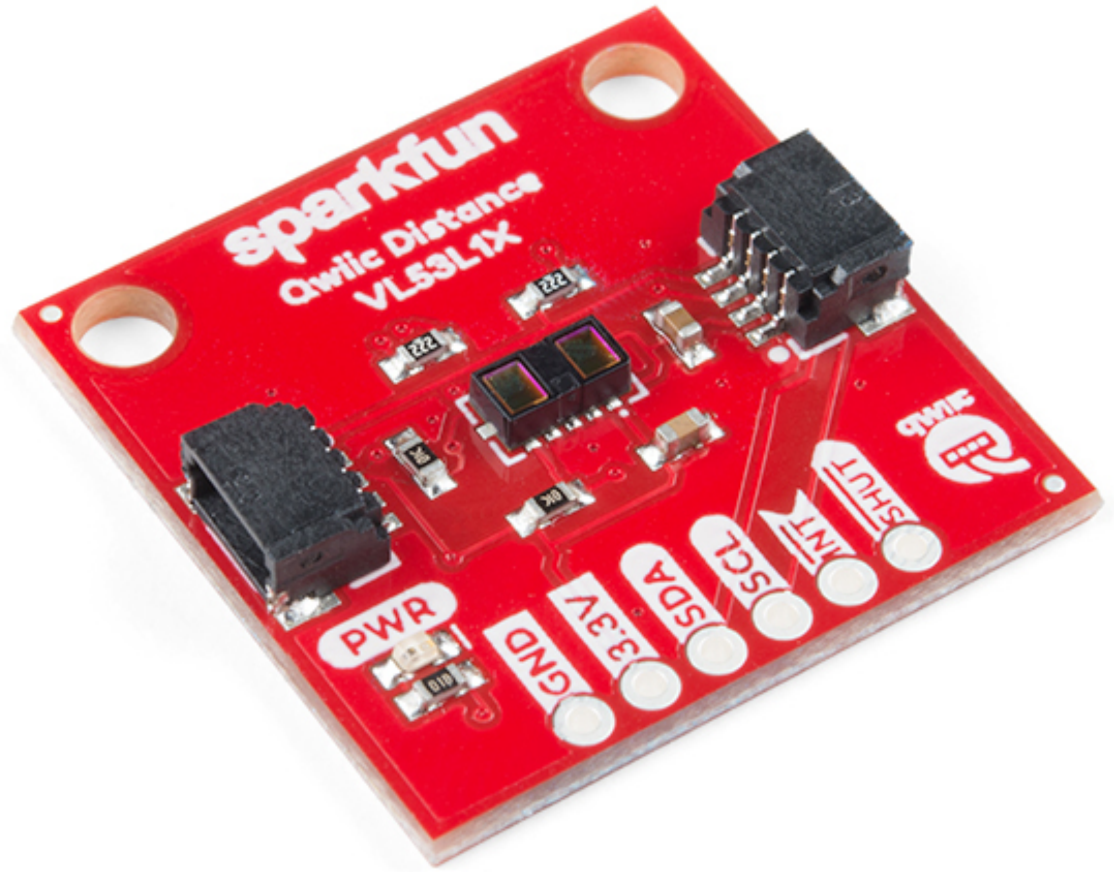
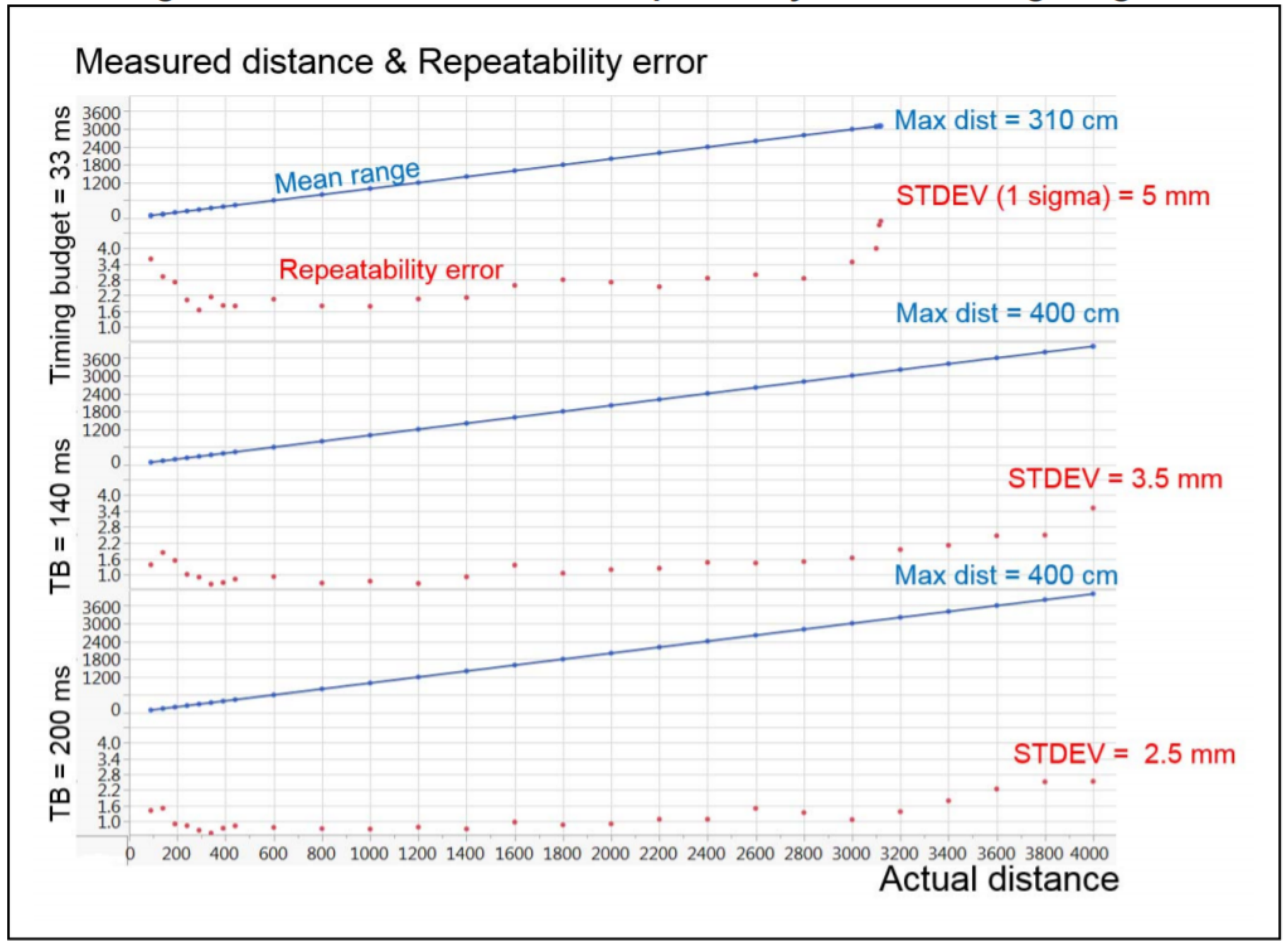


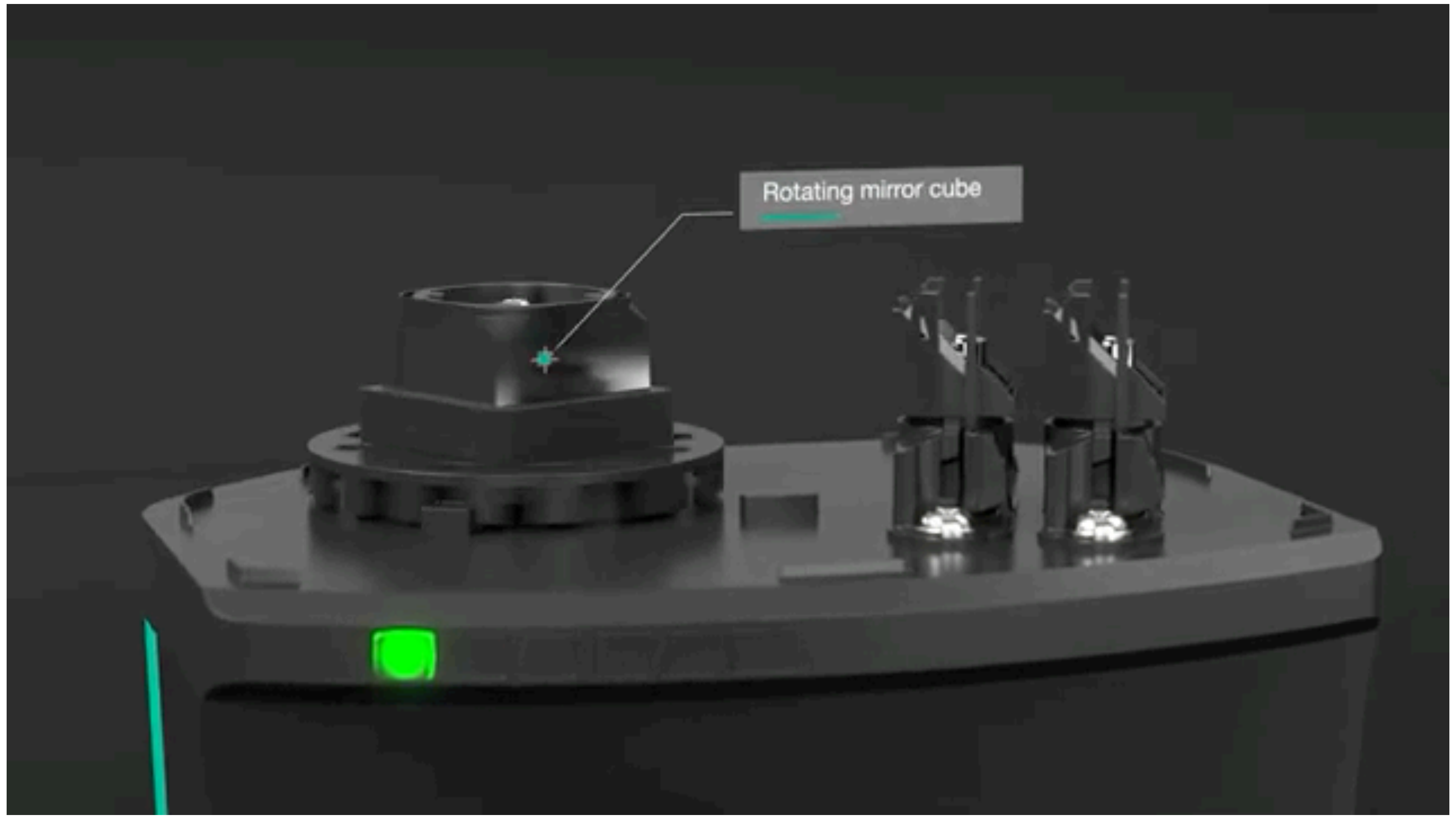
Figure 6. Maximum distance and repeatability error vs. timing budget

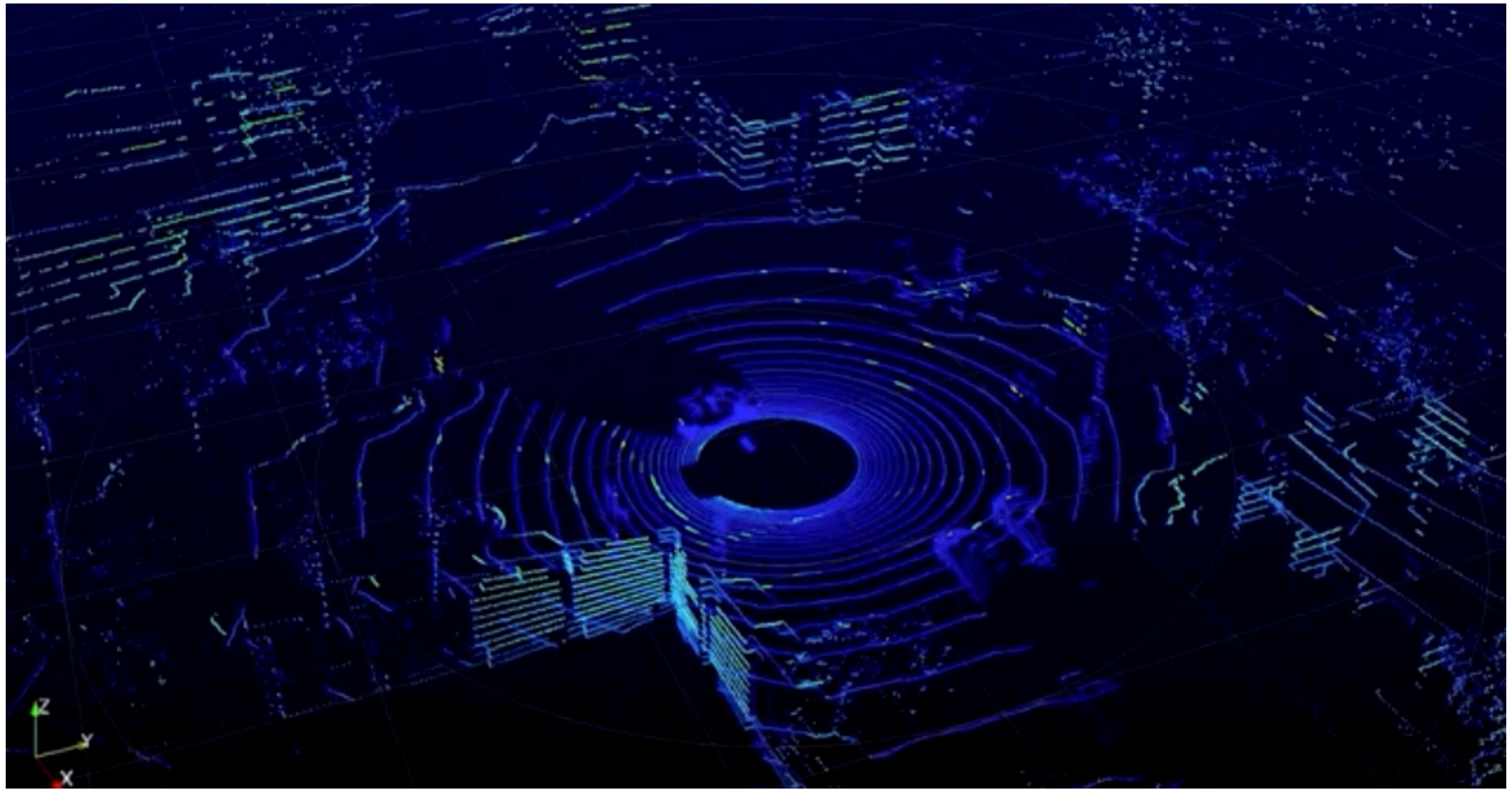


Light Detection and Ranging

- Most common sensors on autonomous cars and robots
- Single points, line scans, full 3D
- Expensive

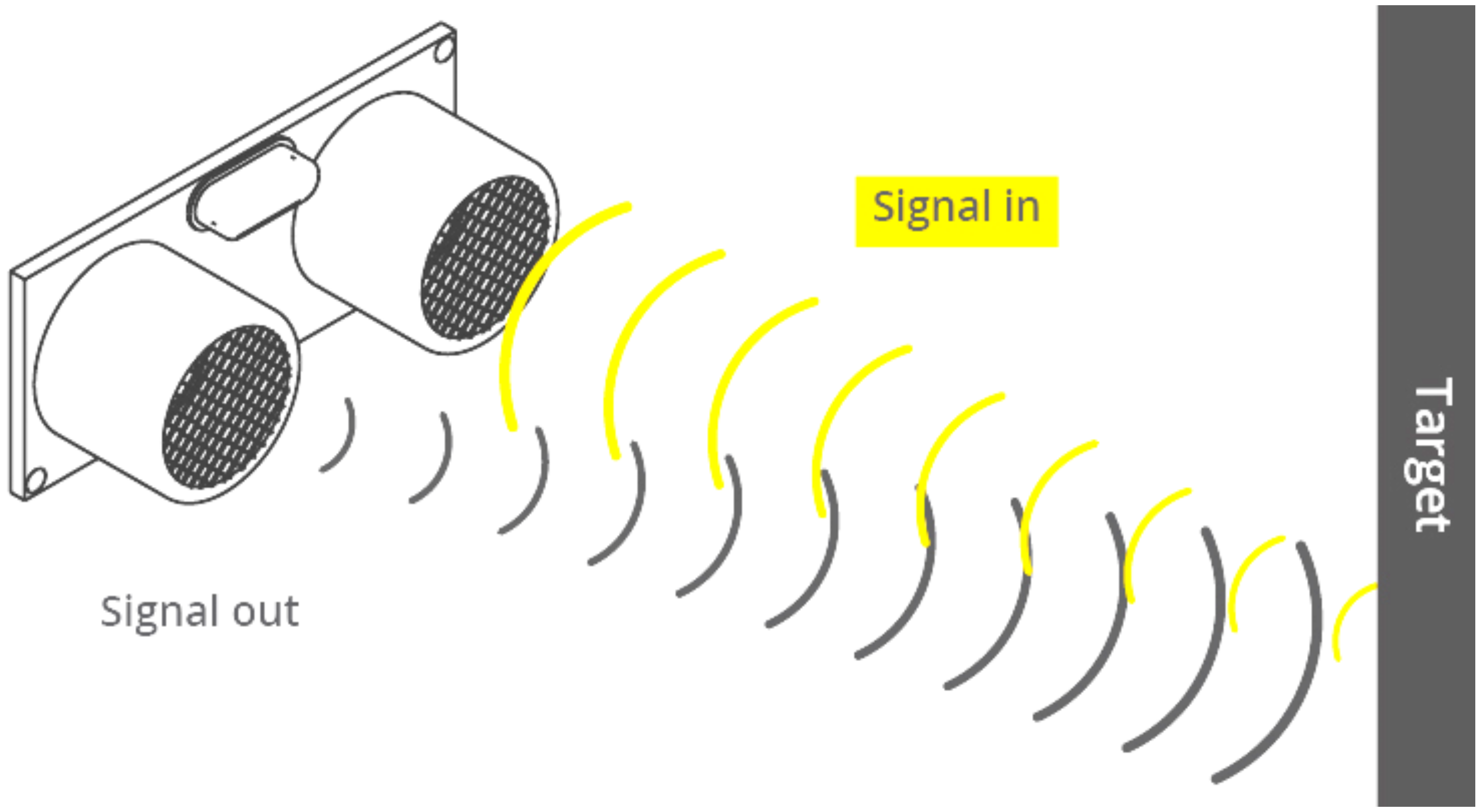
What do the colors represent?

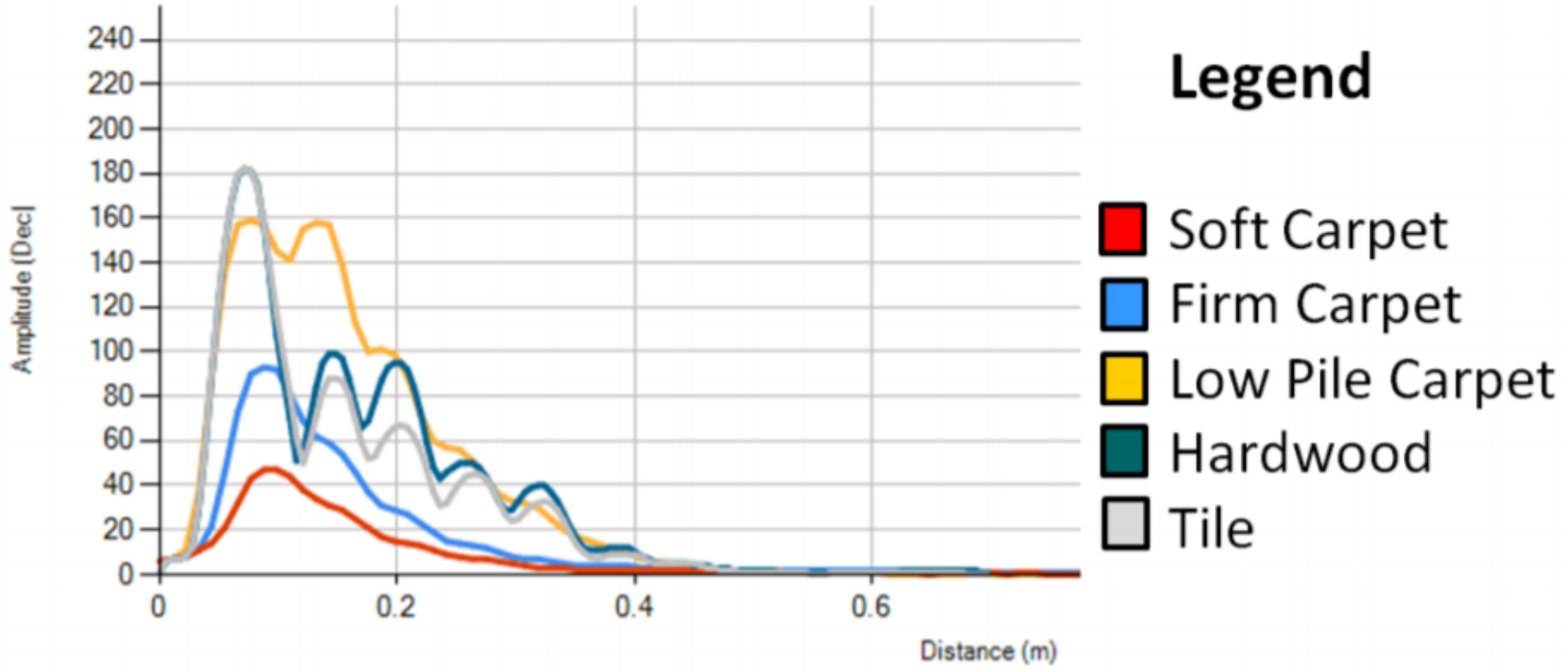


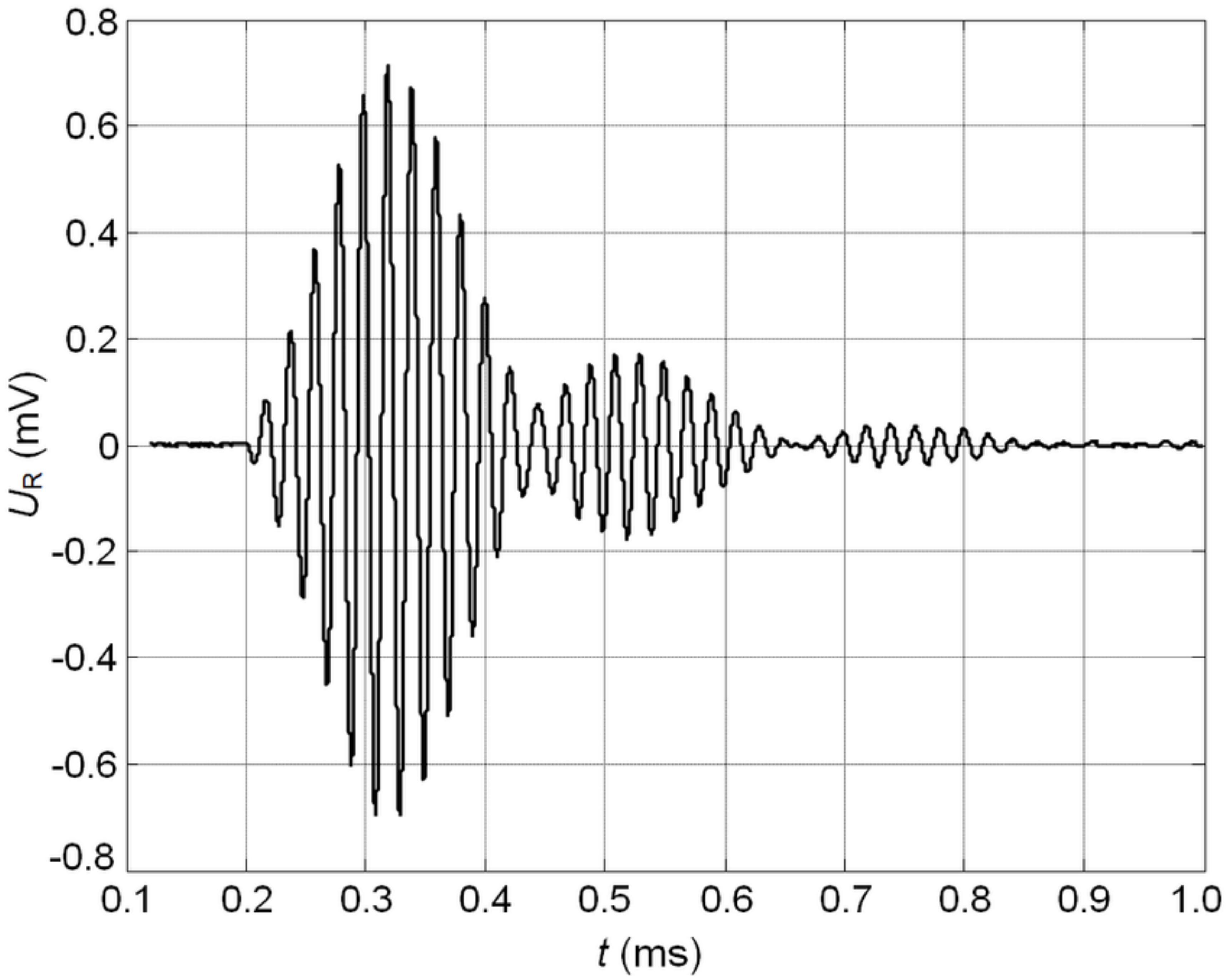


Time of Flight Ultrasonic

- Emits a sound wave, record time *t* until return
  - *r = t x c/2, c = speed of sound, 343 m/s*
- Resolution and range depend on frequency
  - 58kHz: cm-resolution, range < 11m
  - 300kHz: mm-resolution, range < 0.3m
- Low cost (4-12 USD, Sparkfun)
- Insensitive to color, texture, glass, fog, dust, etc.
- Sensitive to humidity, temp, audible noise, and geometry







Time of Flight Ultrasonic

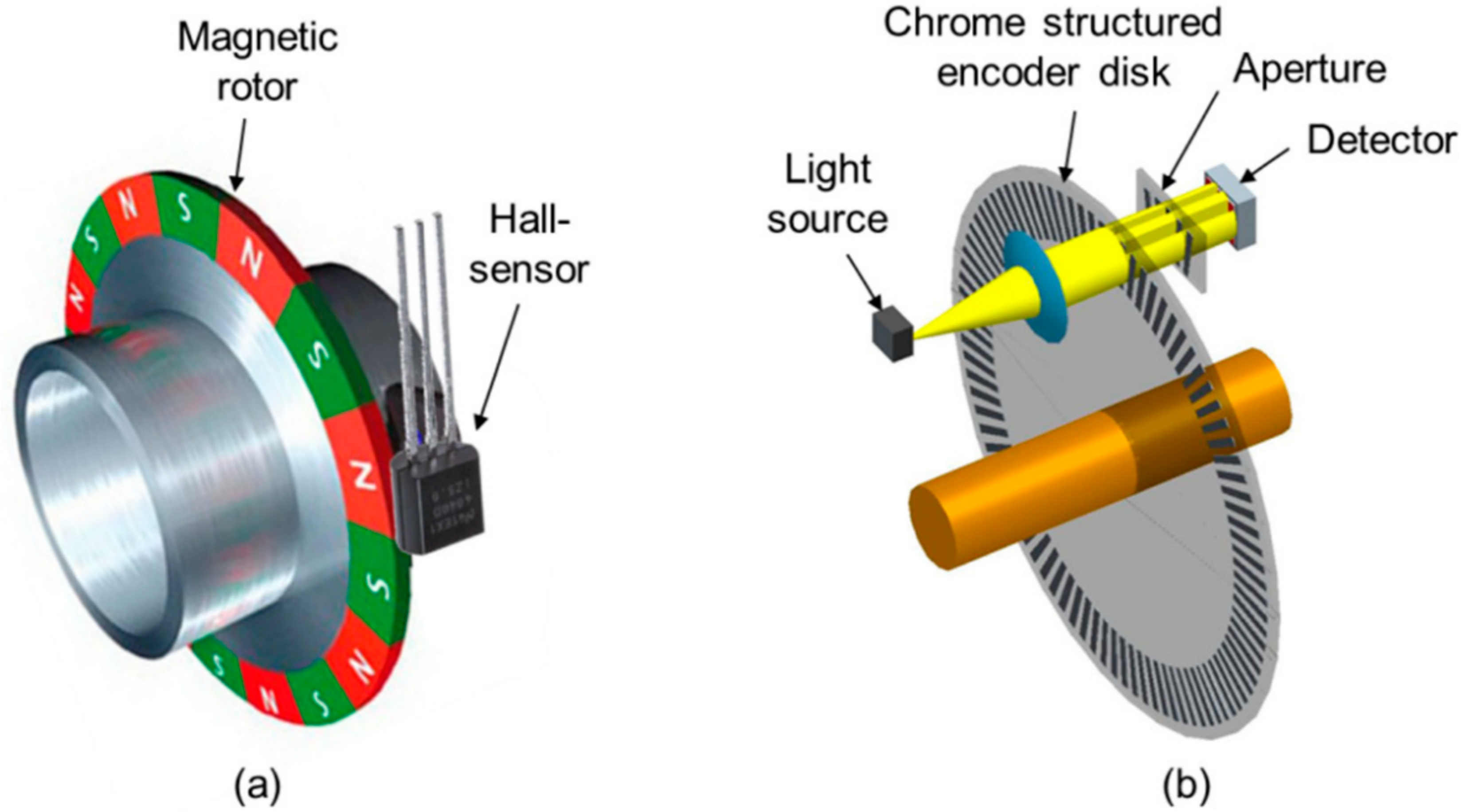
Hobby Distance Sensors

Technology	Application	Pros	Cons
Amplitude-based IR	< 10cm	• ~0.50 USD • Small form factor • High sample rate (4kHz)	• Depends on target reflectivity • Does not work in high ambient light
IR triangulation	< 1m	• Insensitive to surface color/ texture/ ambient light	• ~10 USD • Does not work in high ambient light • Bulky (1.75 x 0.75 x 0.53 in^3) • Low sample rate (26 Hz)
IR ToF	0.1 - 4m	• Small form factor • Insensitive to surface color/ texture/ ambient light	• ~6.50 USD • Complicated Processing • Low sample rate (7 - 30 Hz)
Ultrasonic	0.2 - 10m	• Low cost • Insensitive to surface color and ambient light • Works in rain and fog	• ~4 USD • Complicated processing • Resolution tradeoff with max range • Output depends on surface/ geometry/ humidity • Bulky, high sample time (~10s ms) • Hard to achieve a narrow FOV

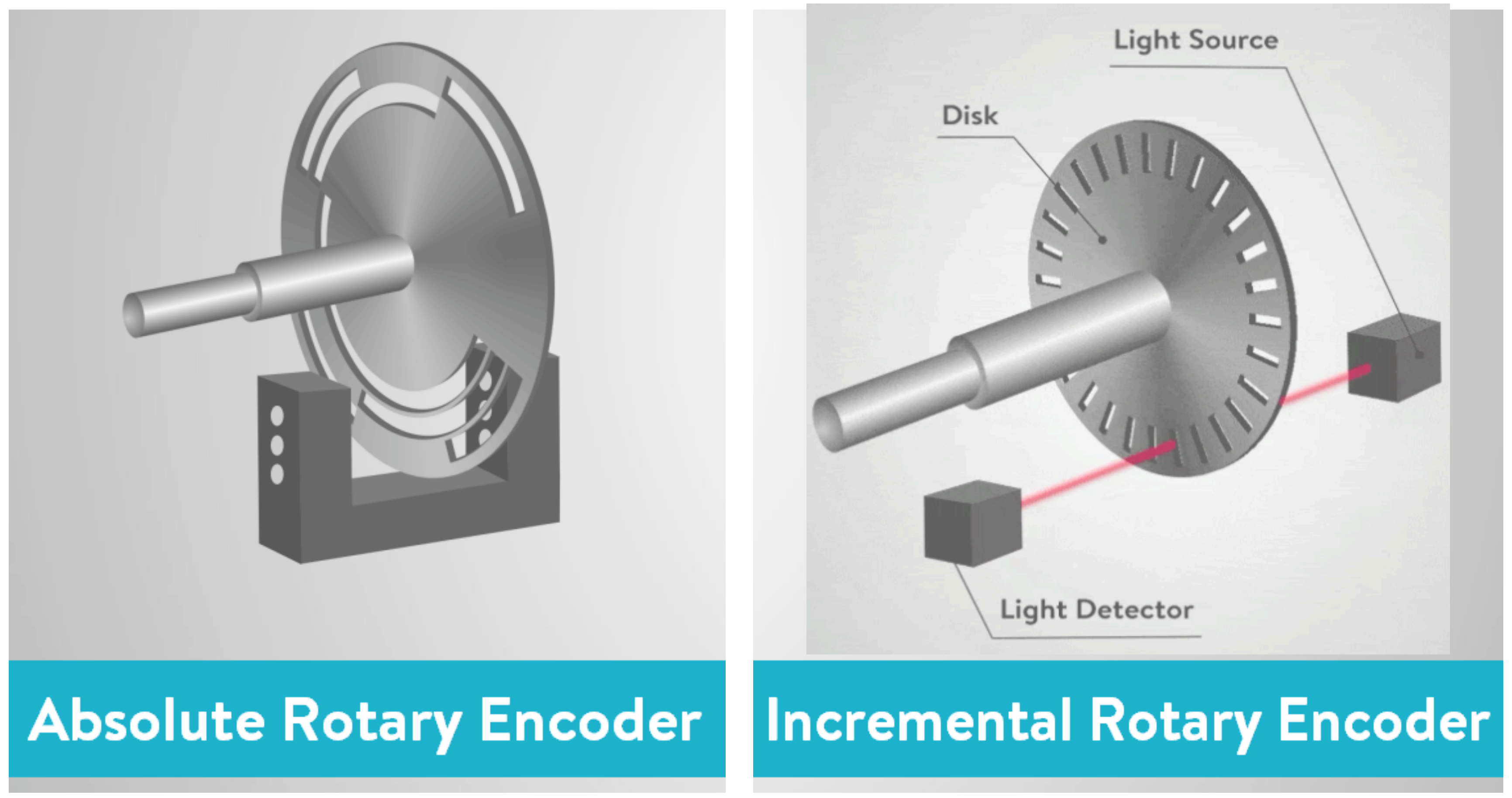
Odometry Sensors

Encoders

- Magnetic, optical, inductive, capacitive, or laser
- Rotary (shaft) encoders
  - Absolute rotary encoder (angular position)
  - Incremental rotary encoders (distance, speed, position)

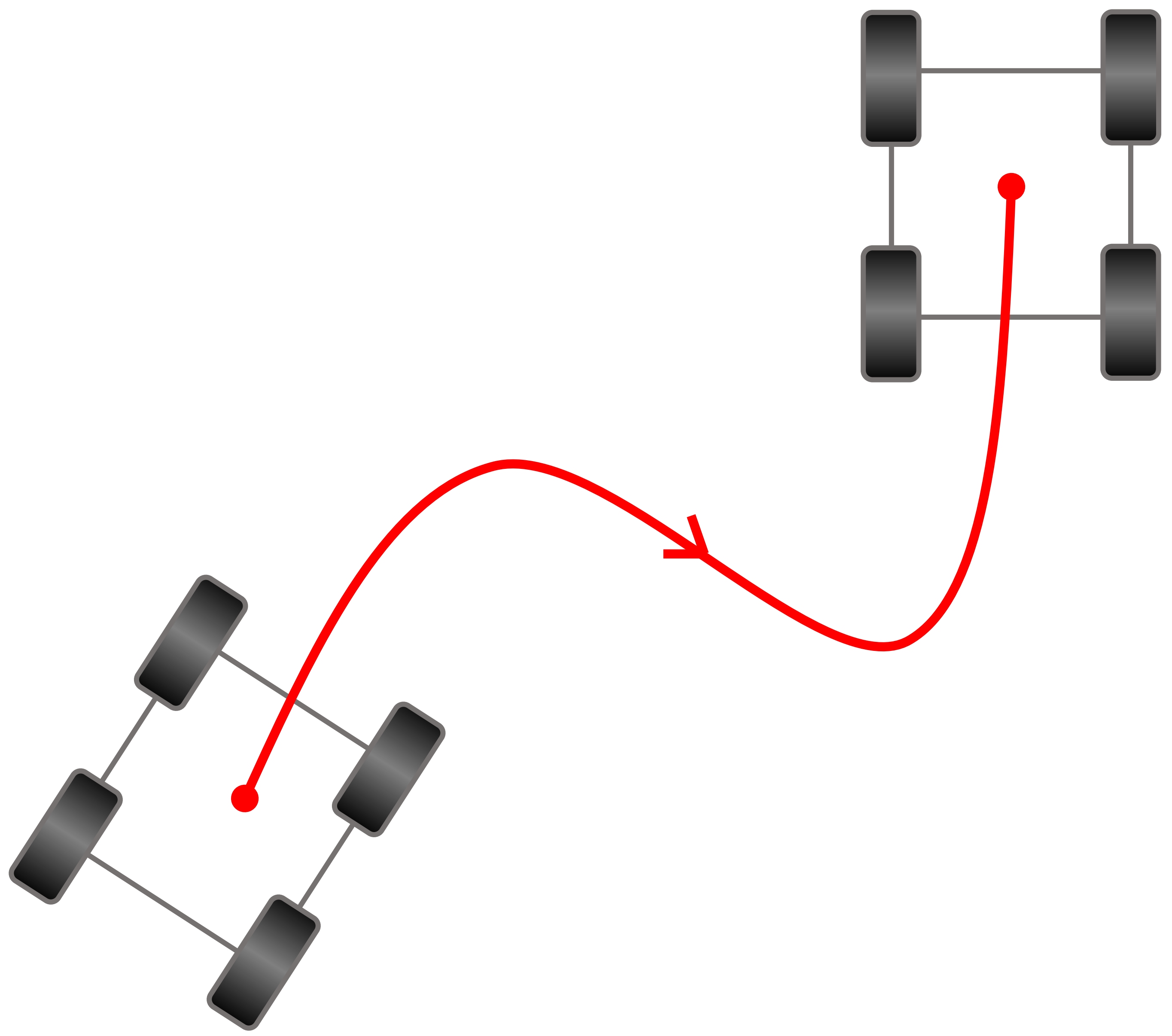


How can you add encoders to your robot?



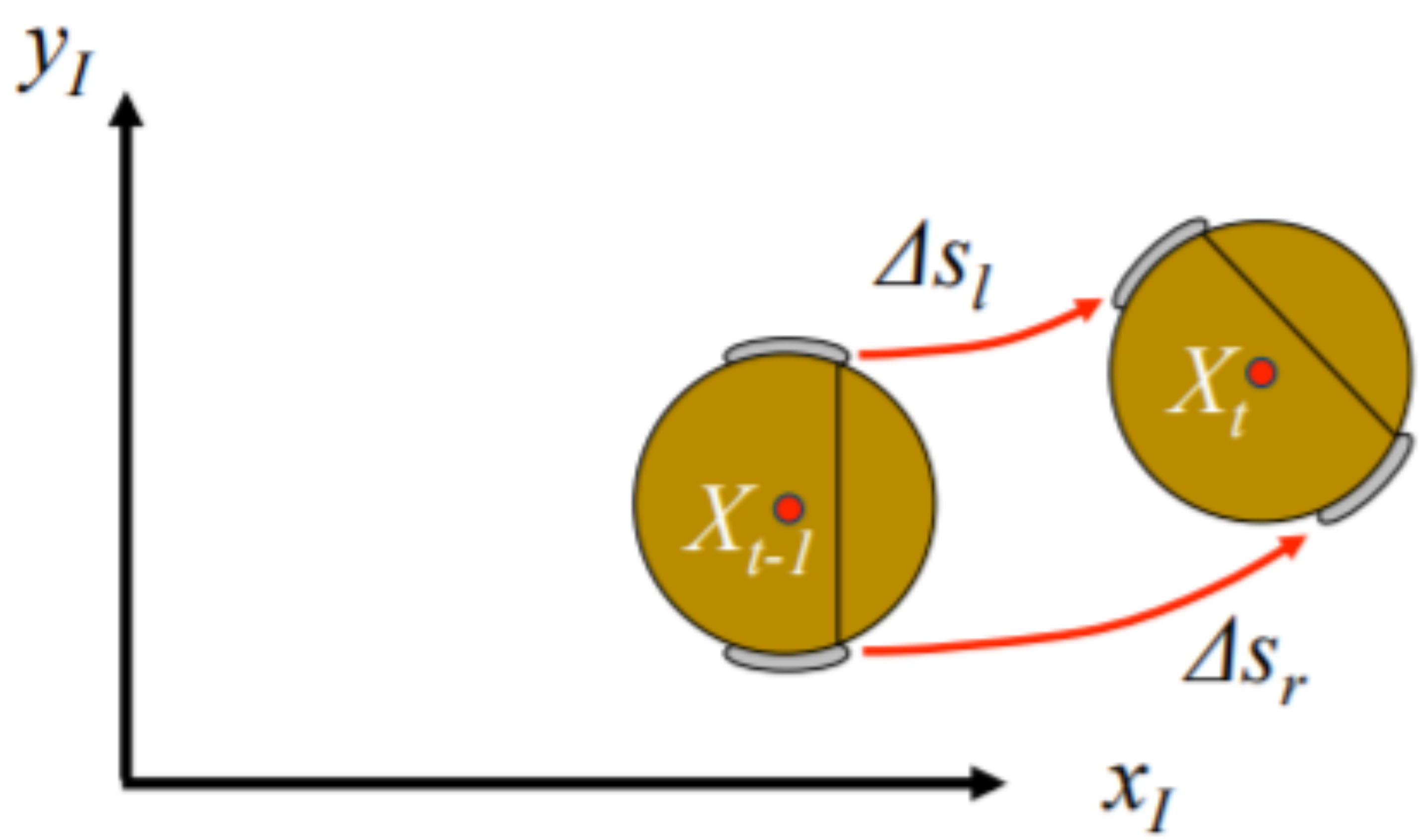
Dead Reckoning

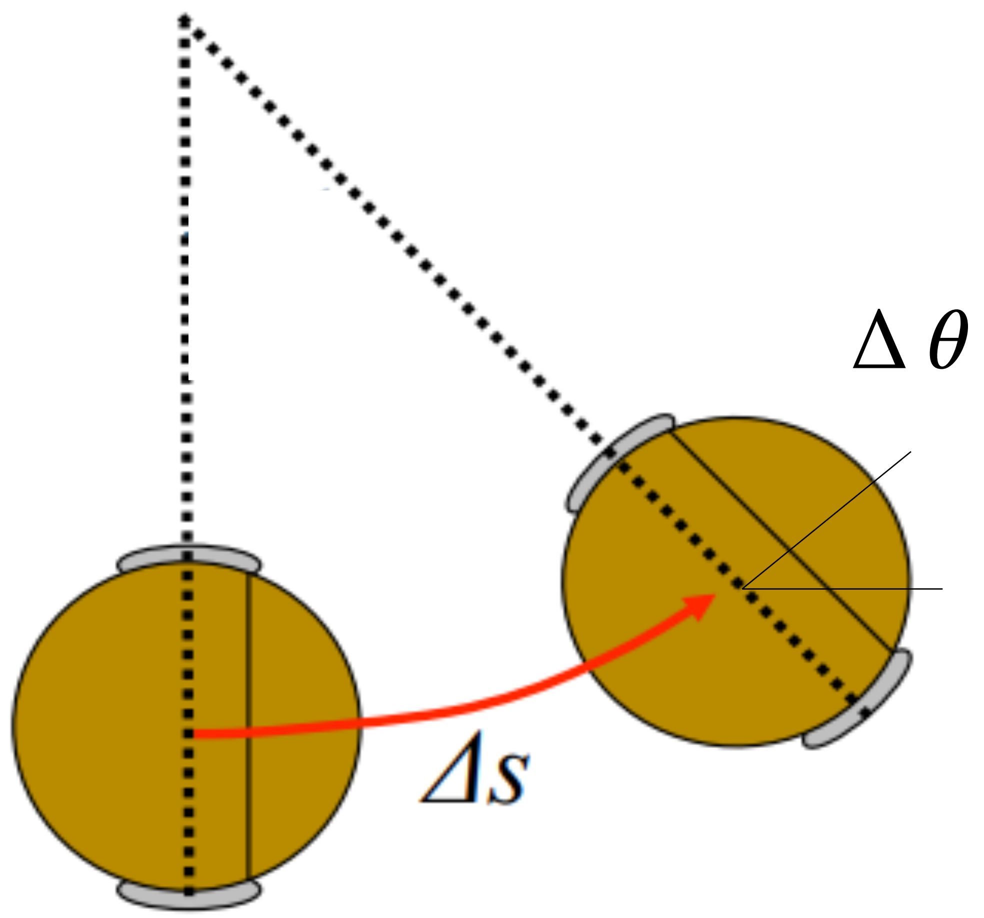
- Map the present state and wheel encoder measurements to the new robot state
  - $X_t = f(X_{t-1}, U_{t-1})$
  - Pro: easy to implement
  - Con: errors integrate and grow unbounded
- **Sources of error**
  - Limited resolution during integration
  - Unequal wheel diameter
  - Variation in the contact point of the wheel
  - Variable friction > slipping
  - Drift or noise in sensors
- How do wheel rotation errors propagate into positioning errors?



Modeling Motion

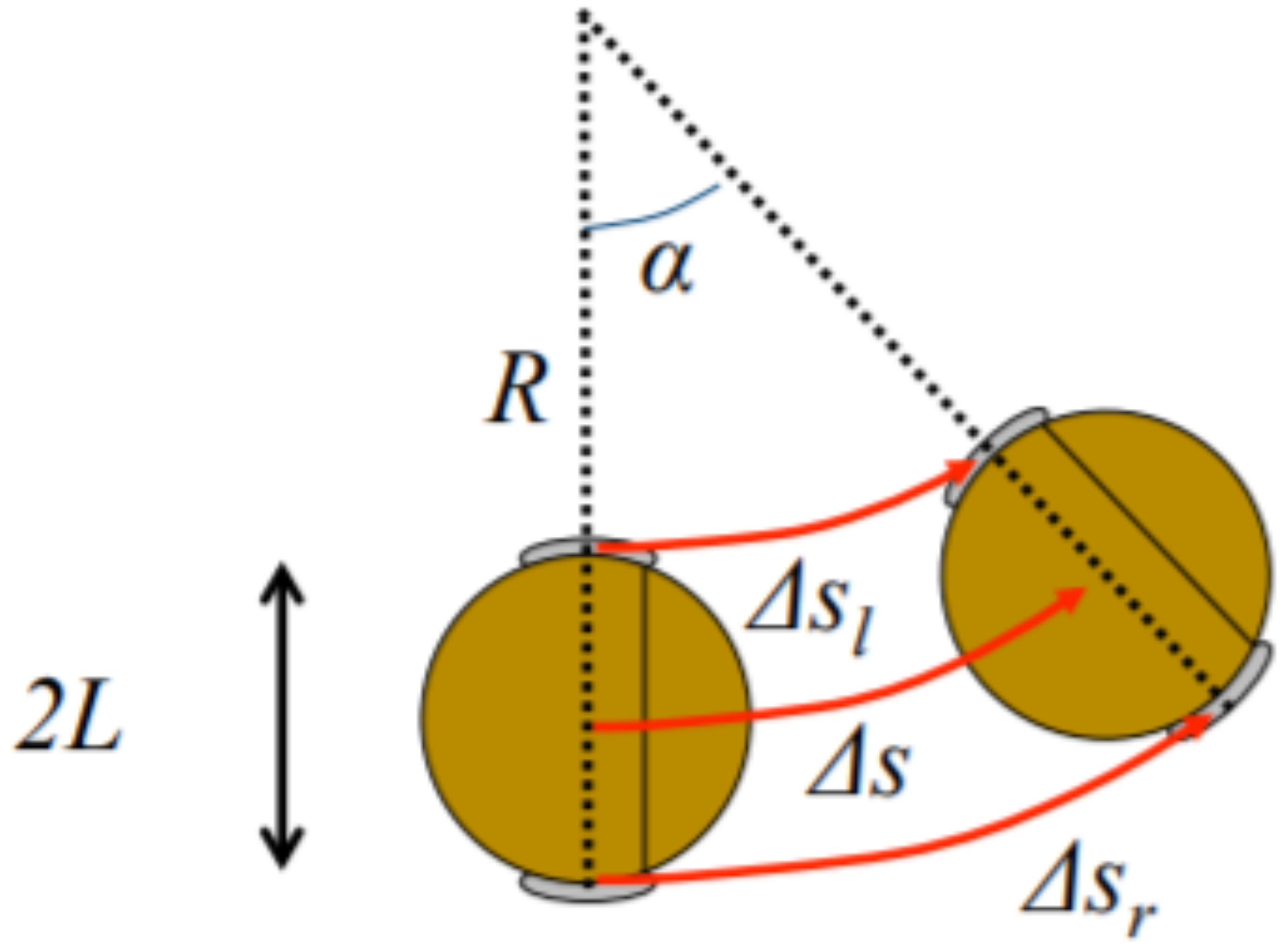
- Start at post X_{t-1}, move right/ left wheel by Δs_r and Δs_l what is pose X_t?
- Model the change in angle $\Delta\theta$ and the distance travelled Δs
  - Assume that the robot is traveling on a circular arc of constant radius





Modeling Motion

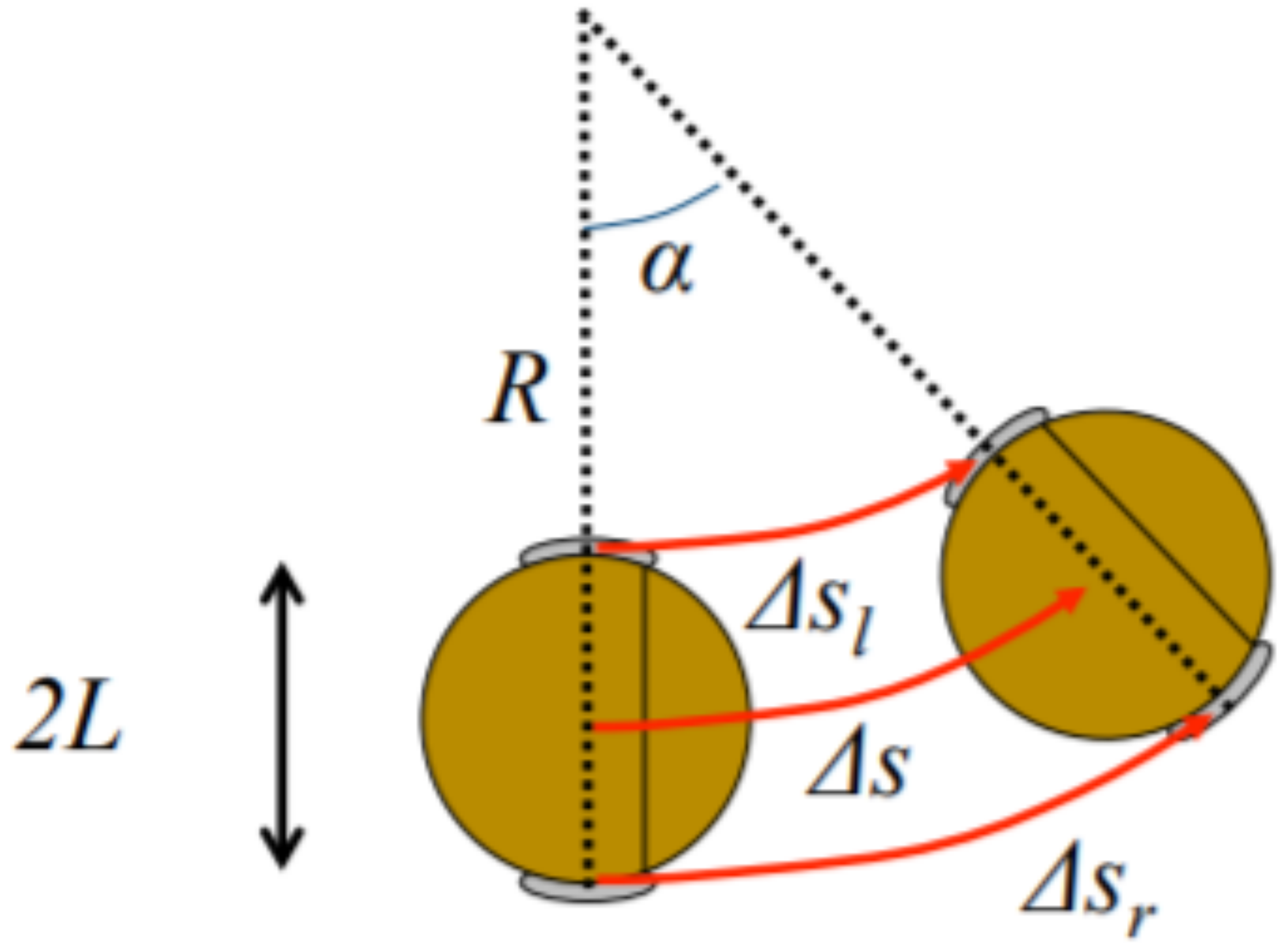
- Start at post X_{t-1}, move right/ left wheel by Δs_r and Δs_l what is pose X_t?
- Model the change in angle $\Delta\theta$ and the distance travelled Δs
  - Assume that the robot is traveling on a circular arc of constant radius



- For circular arcs:
  - (1) $\Delta s_l = R\alpha$
  - (2) $\Delta s_r = (R + 2L)\alpha$
  - (3) $\Delta s = (R + L)\alpha$
  - Use (1) and (2): $L\alpha = \dfrac{\Delta s_r - R\alpha}{2} = \dfrac{\Delta s_r}{2} - \dfrac{\Delta s_l}{2}$
  - Insert into (3) to compute (4): $\Delta s = \dfrac{\Delta s_l + \Delta s_r}{2}$

Modeling Motion

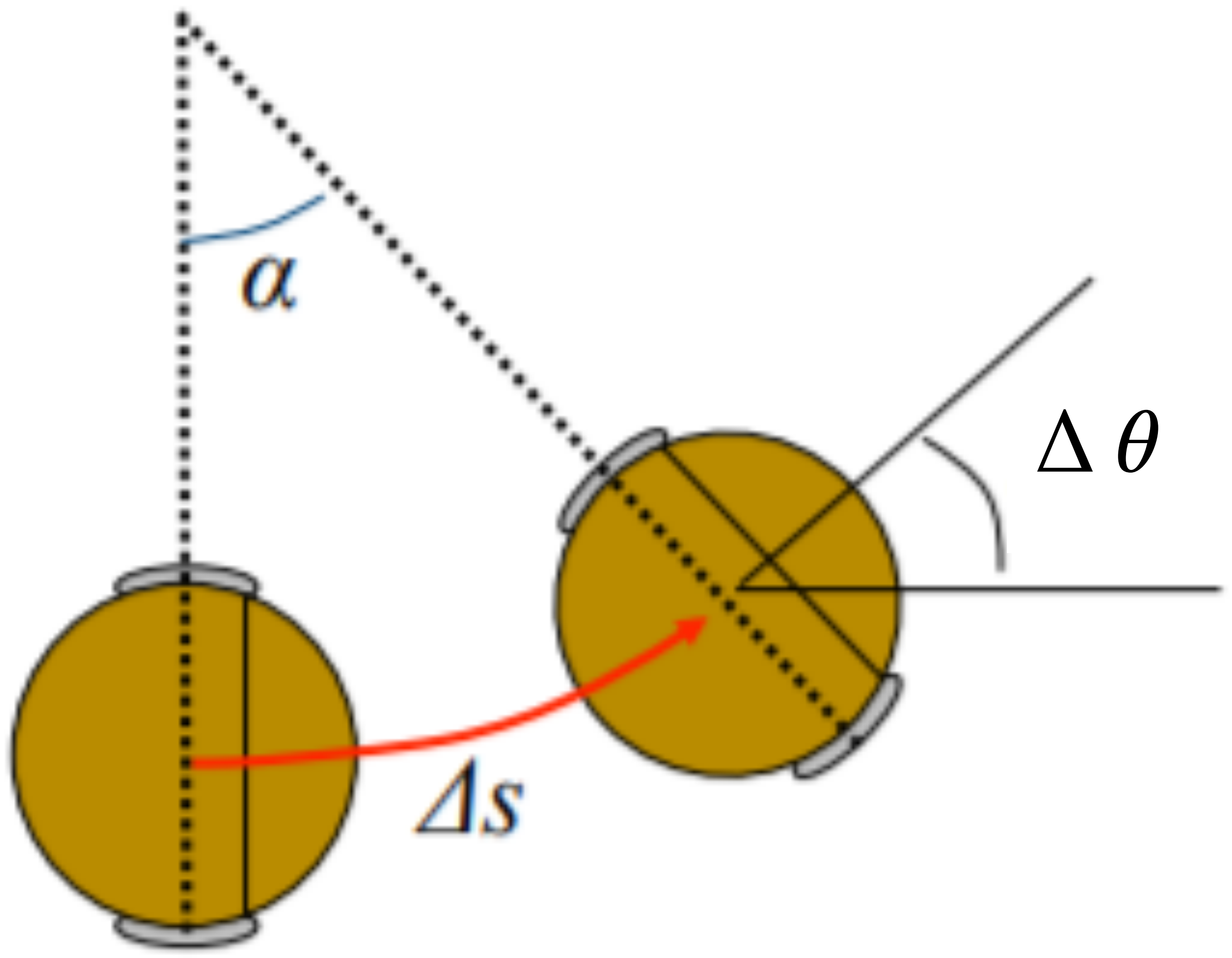
- Start at post X_{t-1}, move right/ left wheel by Δs_r and Δs_l what is pose X_t?
- Model the change in angle $\Delta\theta$ and the distance travelled Δs
  - Assume that the robot is traveling on a circular arc of constant radius



- For circular arcs:
  - (1) $\Delta s_l = R\alpha$
  - (2) $\Delta s_r = (R + 2L)\alpha$
  - (3) $\Delta s = (R + L)\alpha$
- (4) $\Delta s = \dfrac{\Delta s_l + \Delta s_r}{2}$
- Or you can note that the distance traveled by the robot center, is simply the average distance traveled by each wheel

Modeling Motion

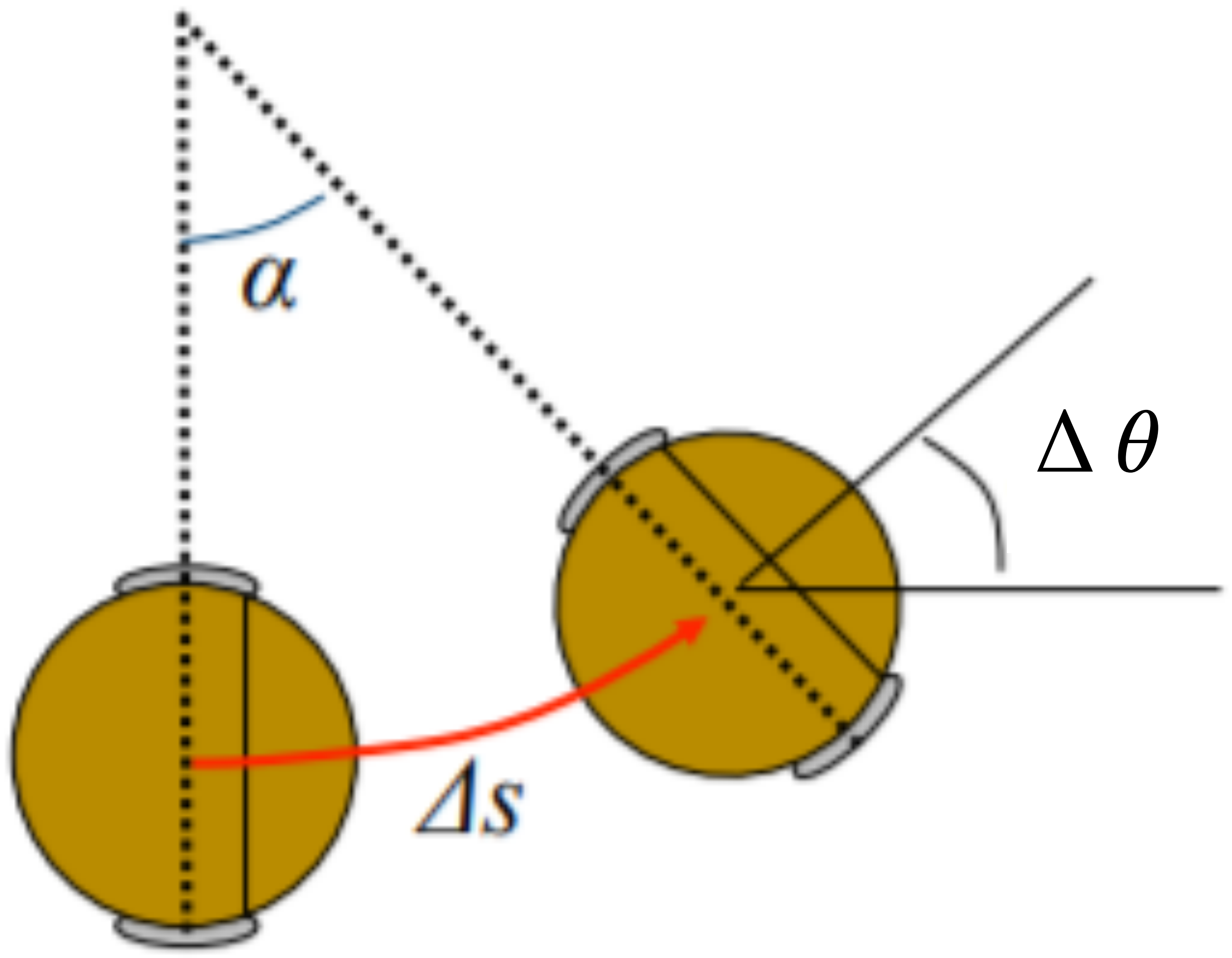
- Start at post X_{t-1}, move right/ left wheel by Δs_r and Δs_l what is pose X_t?
- Model the change in angle $\Delta\theta$ and the distance travelled Δs
  - Assume that the robot is traveling on a circular arc of constant radius



- For circular arcs:
  - (1) $\Delta s_l = R\alpha$
  - (2) $\Delta s_r = (R + 2L)\alpha$
  - (3) $\Delta s = (R + L)\alpha$
- (4) $\Delta s = \dfrac{\Delta s_l + \Delta s_r}{2}$
- The change in angle, $\Delta\theta = \alpha$

Modeling Motion

- Start at post X_{t-1}, move right/ left wheel by Δs_r and Δs_l what is pose X_t?
- Model the change in angle $\Delta\theta$ and the distance travelled Δs
  - Assume that the robot is traveling on a circular arc of constant radius

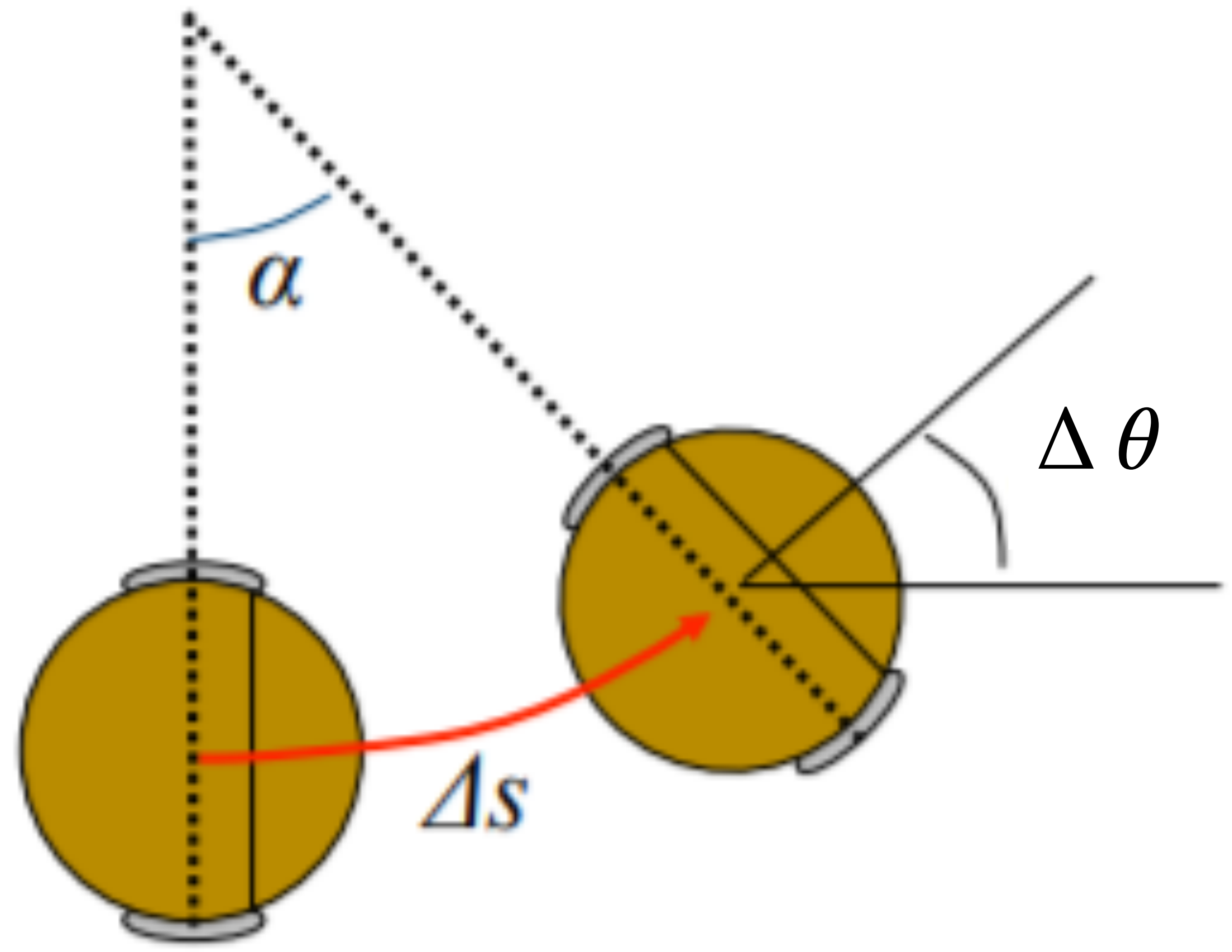


- For circular arcs:
  - (1) $\Delta s_l = R\alpha$
  - (2) $\Delta s_r = (R + 2L)\alpha$
  - (3) $\Delta s = (R + L)\alpha$
- Use α in (1) and (2):
  - $\frac{\Delta s_l}{R} = \frac{\Delta s_r}{R + 2L} \leftrightarrow (R + 2L)\Delta s_l = R\Delta s_r$

- (4) $\Delta s = \frac{\Delta s_l + \Delta s_r}{2}$
- (5) $R = \frac{2L\Delta s_l}{\Delta s_r - \Delta s_l}$

Modeling Motion

- Start at post X_{t-1}, move right/ left wheel by Δs_r and Δs_l what is pose X_t?
- Model the change in angle $\Delta\theta$ and the distance travelled Δs
  - Assume that the robot is traveling on a circular arc of constant radius

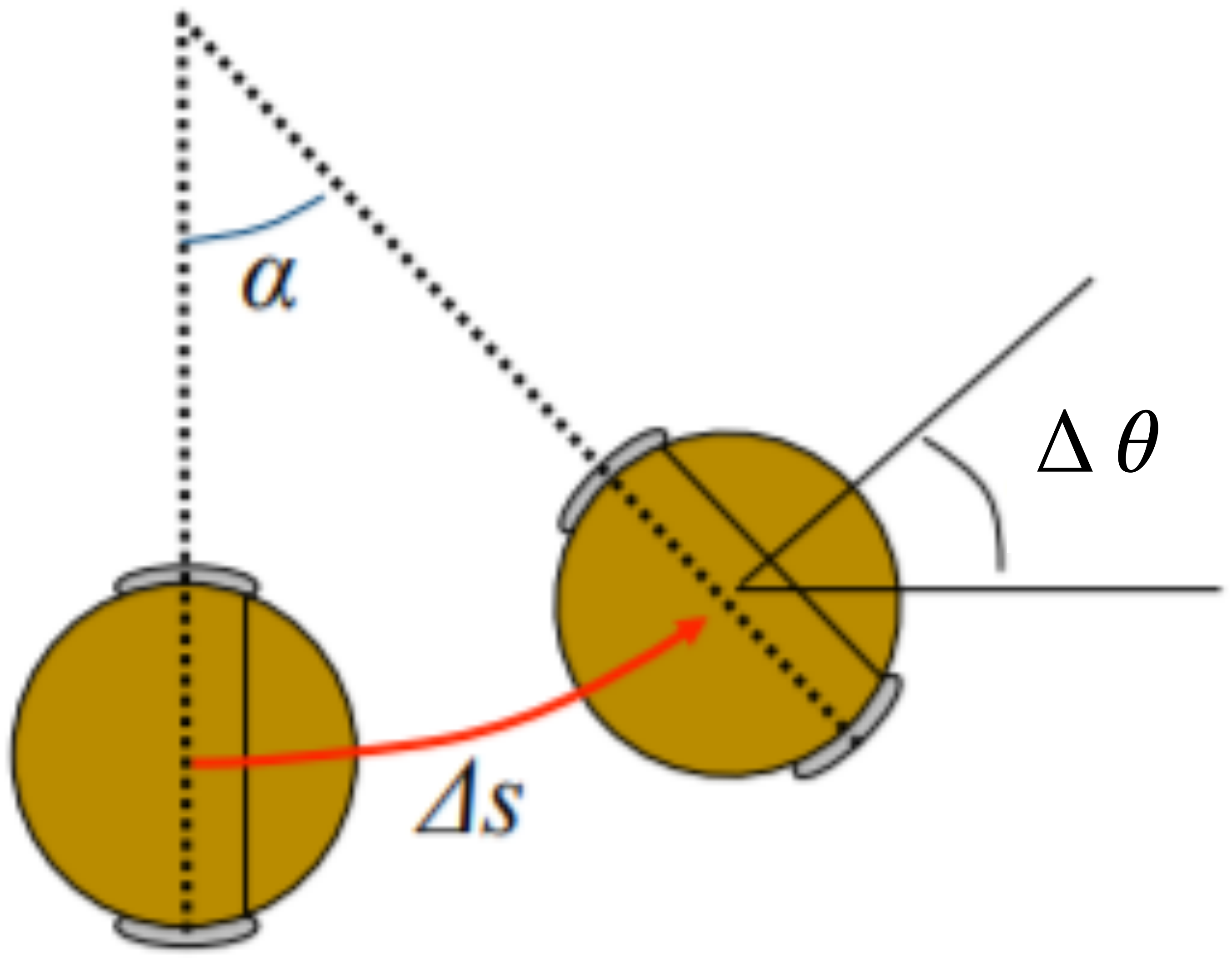


- For circular arcs:
  - (1) $\Delta s_l = R\alpha$
  - (2) $\Delta s_r = (R + 2L)\alpha$
  - (3) $\Delta s = (R + L)\alpha$
- Use (5) in (1):
  - $\alpha = \dfrac{\Delta s_l}{R} = \dfrac{\Delta s_l(\Delta s_r - \Delta s_l)}{2L\Delta s_l} = \dfrac{(\Delta s_r - \Delta s_l)}{2L} = \Delta\theta$

- (4) $\Delta s = \dfrac{\Delta s_l + \Delta s_r}{2}$
- (5) $R = \dfrac{2L\Delta s_l}{\Delta s_r - \Delta s_l}$

Modeling Motion

- Start at post X_{t-1}, move right/ left wheel by Δs_r and Δs_l what is pose X_t?
- Model the change in angle $\Delta\theta$ and the distance travelled Δs
  - Assume that the robot is traveling on a circular arc of constant radius
  - Assume that the motion is small ($\Delta d = \Delta s$)

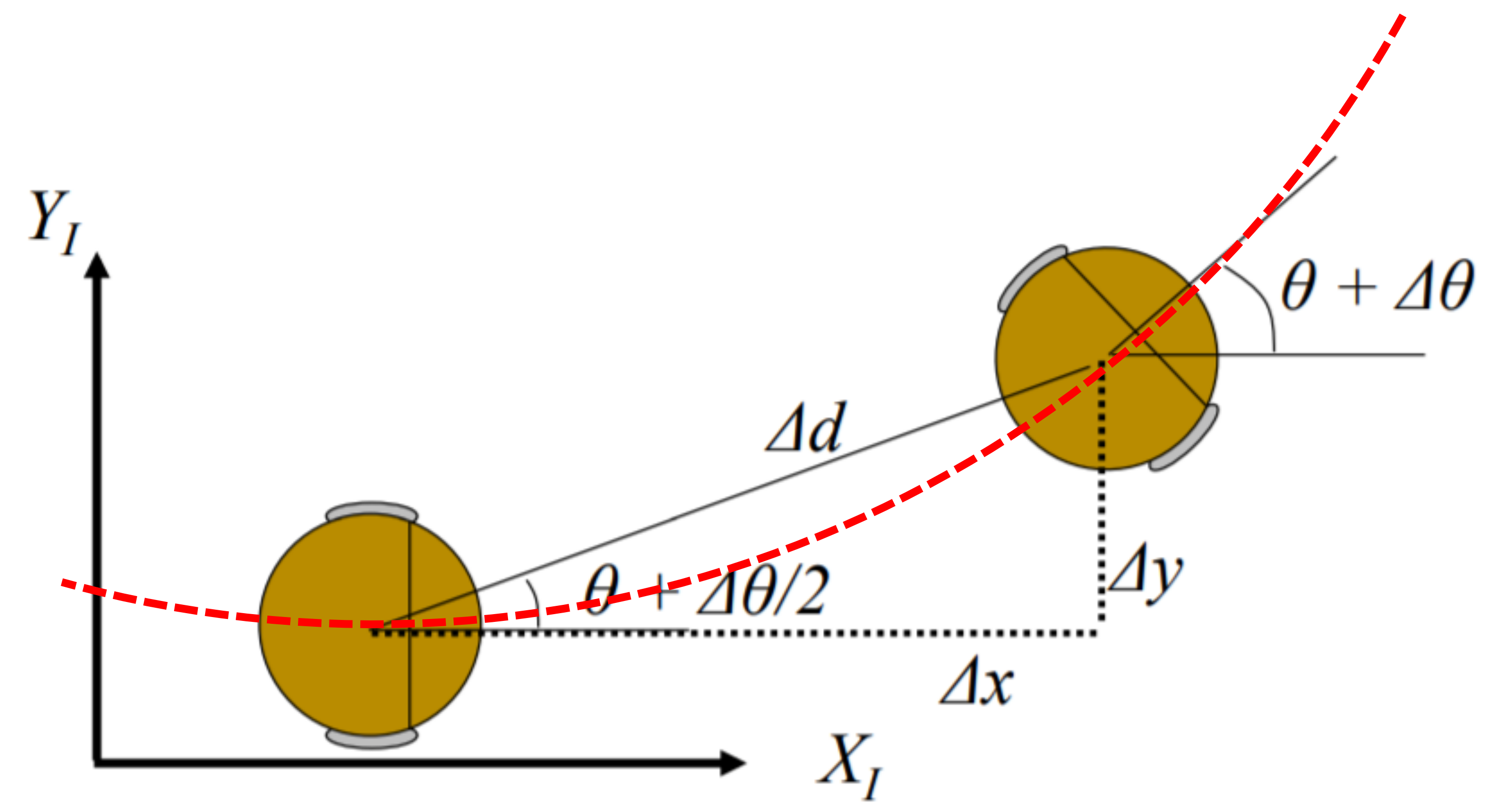


- (1) $\Delta s_l = R\alpha$
- (2) $\Delta s_r = (R + 2L)\alpha$
- (3) $\Delta s = (R + L)\alpha$

- (4) $\Delta s = \dfrac{\Delta s_l + \Delta s_r}{2}$
- (5) $R = \dfrac{2L\Delta s_l}{\Delta s_r - \Delta s_l}$
- (6) $\Delta\theta = \dfrac{(\Delta s_r - \Delta s_l)}{2L}$

Modeling Motion

- Start at post X_{t-1}, move right/ left wheel by Δs_r and Δs_l what is pose X_t?
- Model the change in angle $\Delta\theta$ and the distance travelled Δs
  - Assume that the robot is traveling on a circular arc of constant radius
  - Assume that the motion is small ($\Delta d = \Delta s$)

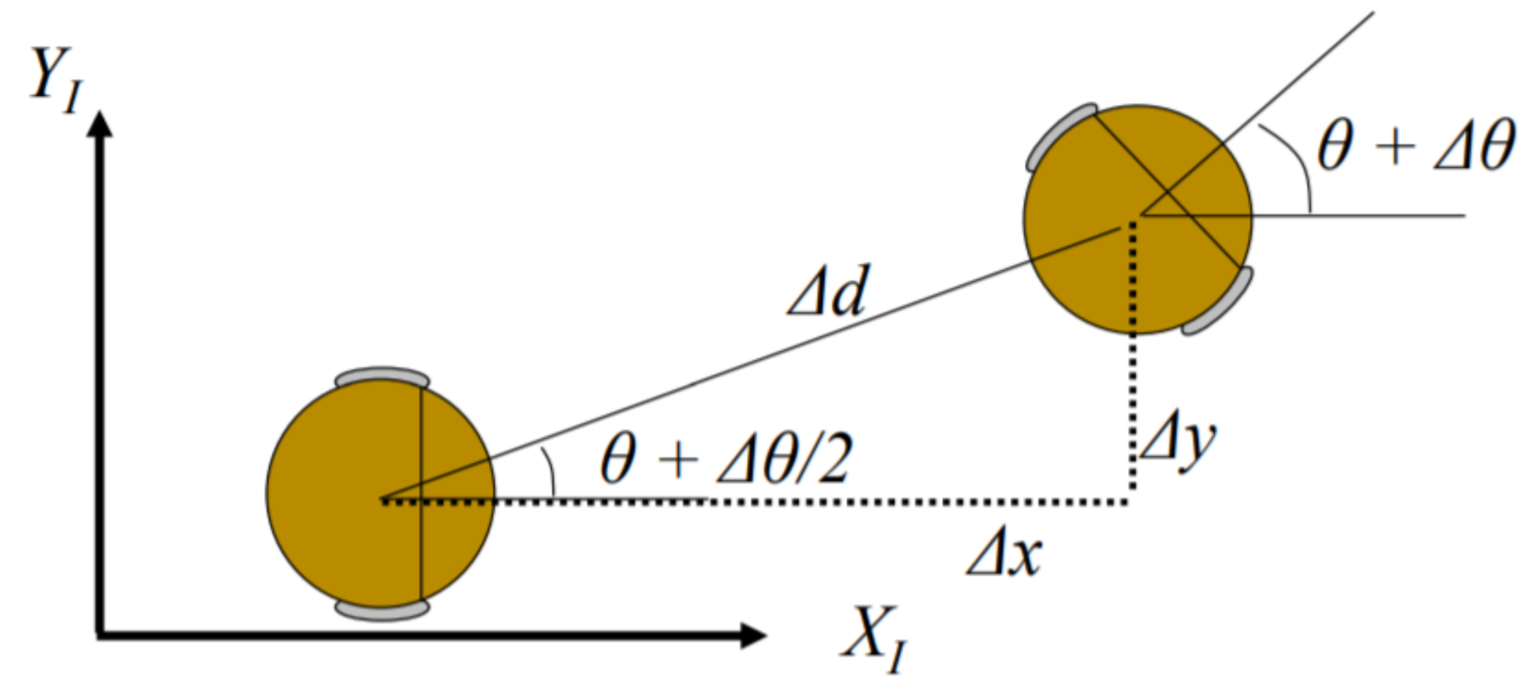


- (1) $\Delta s_l = R\alpha$
- (2) $\Delta s_r = (R + 2L)\alpha$
- (3) $\Delta s = (R + L)\alpha$
- (7) $\Delta x = \Delta s \cos(\theta + \Delta\theta/2)$
- (8) $\Delta y = \Delta s \sin(\theta + \Delta\theta/2)$

- (4) $\Delta s = \dfrac{\Delta s_l + \Delta s_r}{2}$
- (5) $R = \dfrac{2L\Delta s_l}{\Delta s_r - \Delta s_l}$
- (6) $\Delta\theta = \dfrac{(\Delta s_r - \Delta s_l)}{2L}$

Modeling Motion

- Start at post X_{t-1}, move right/ left wheel by Δs_r and Δs_l what is pose X_t?
- Model the change in angle $\Delta\theta$ and the distance travelled Δs
  - Assume that the robot is traveling on a circular arc of constant radius
  - Assume that the motion is small ($\Delta d = \Delta s$)



- (4) $\Delta s = \dfrac{\Delta s_l + \Delta s_r}{2}$
- (6) $\Delta\theta = \dfrac{(\Delta s_r - \Delta s_l)}{2L}$
- (7) $\Delta x = \Delta s \cos(\theta + \Delta\theta/2)$
- (8) $\Delta y = \Delta s \sin(\theta + \Delta\theta/2)$

$$X_t = f(x, y, \theta, \Delta s_r, \Delta s_l)$$

$$X_t = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \boxed{\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{bmatrix}}$$

$$\boxed{\begin{bmatrix} \dfrac{\Delta s_l + \Delta s_r}{2} \cos\left(\theta + \dfrac{\Delta s_r - \Delta s_l}{4L}\right) \\ \dfrac{\Delta s_l + \Delta s_r}{2} \sin\left(\theta + \dfrac{\Delta s_r - \Delta s_l}{4L}\right) \\ \dfrac{\Delta s_r - \Delta s_l}{2L} \end{bmatrix}}$$

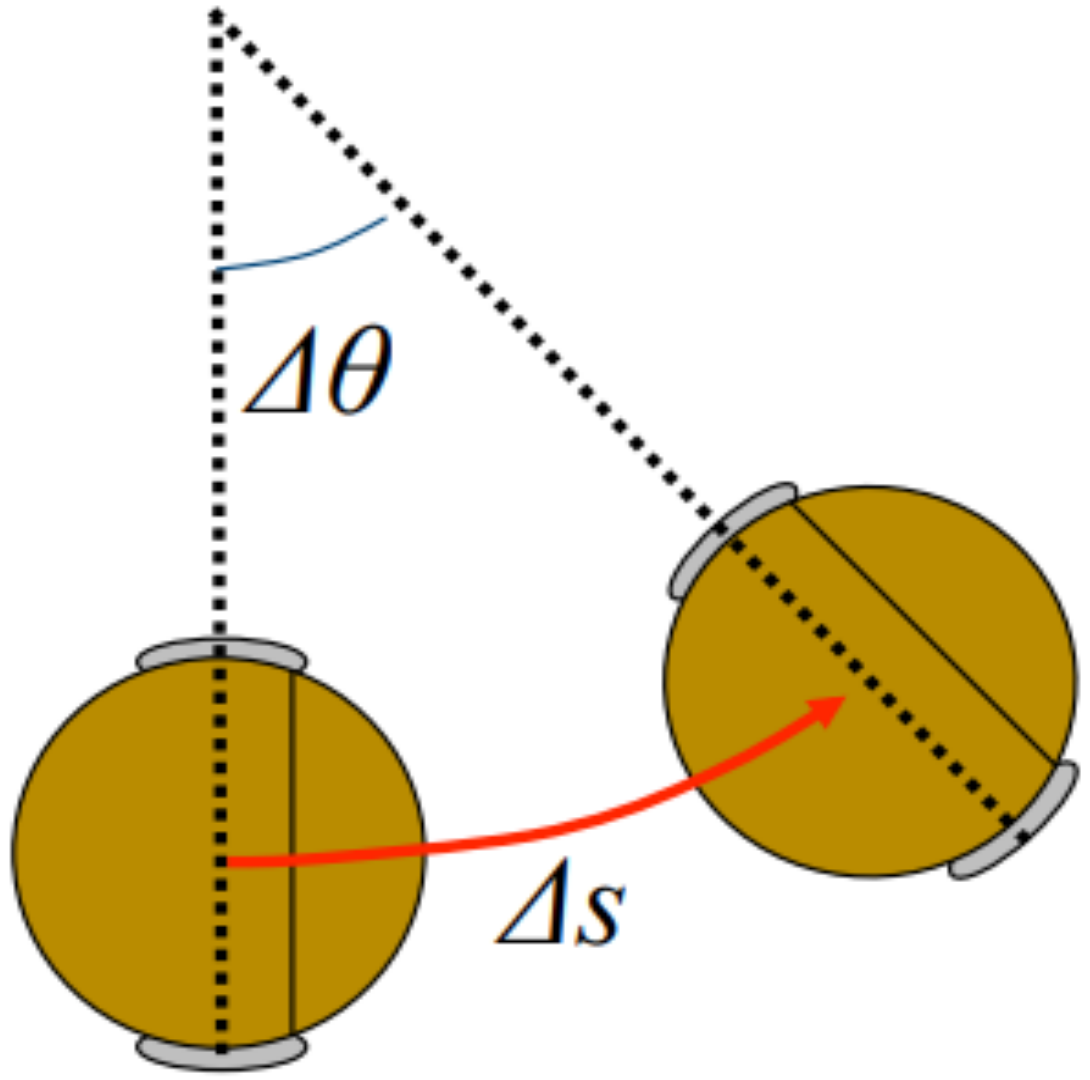
Modeling Motion

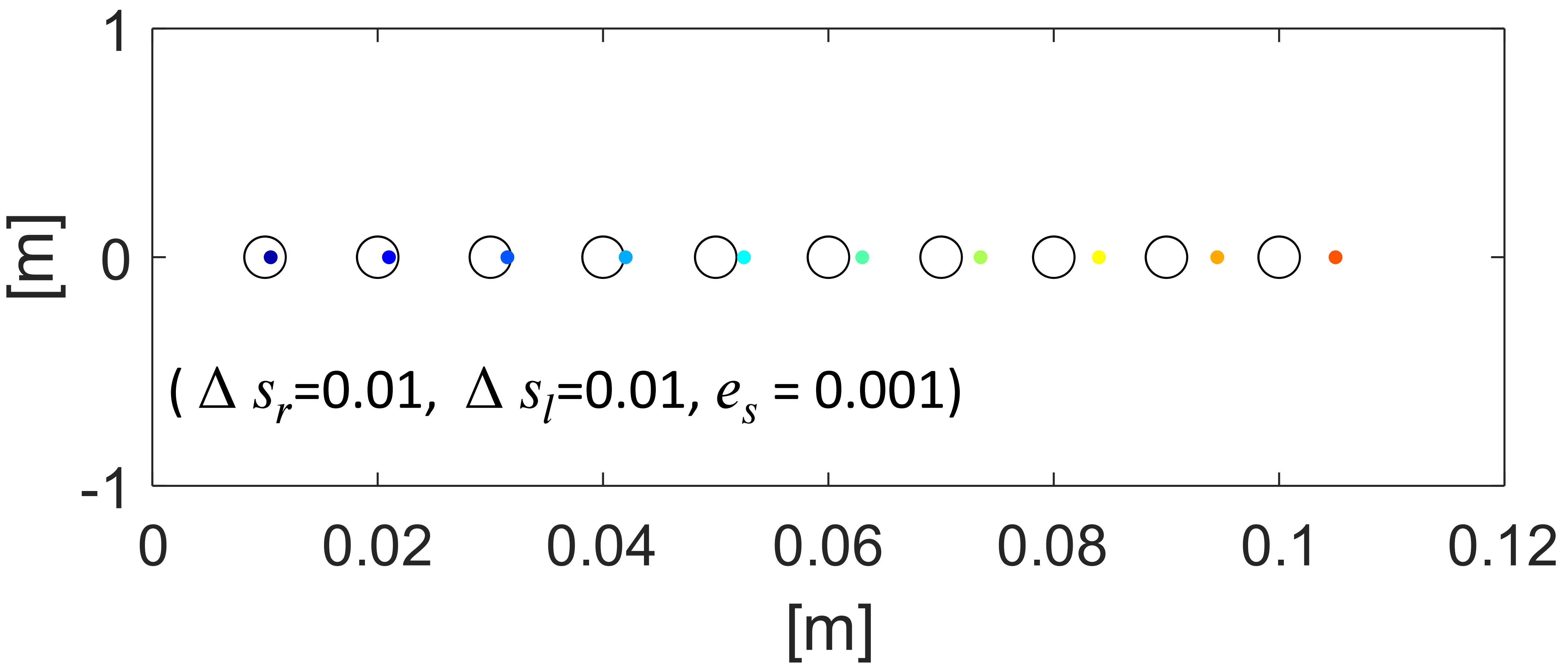
- How do wheel rotation errors propagate into positioning errors?

$$\Delta s = d + e_s$$

$$\Delta x = \frac{\Delta s_l + \Delta s_r + e_s}{2}\cos\left(\theta + \frac{\Delta s_r - \Delta s_l}{4L}\right)$$

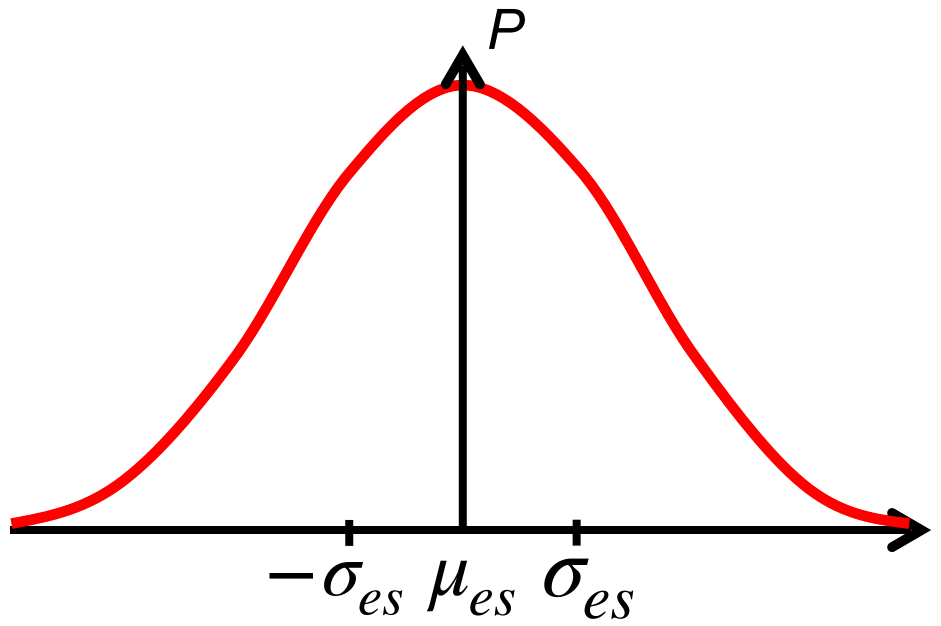
$$\Delta y = \frac{\Delta s_l + \Delta s_r + e_s}{2}\sin\left(\theta + \frac{\Delta s_r - \Delta s_l}{4L}\right)$$





Modeling Motion

- How do wheel rotation errors propagate into positioning errors?

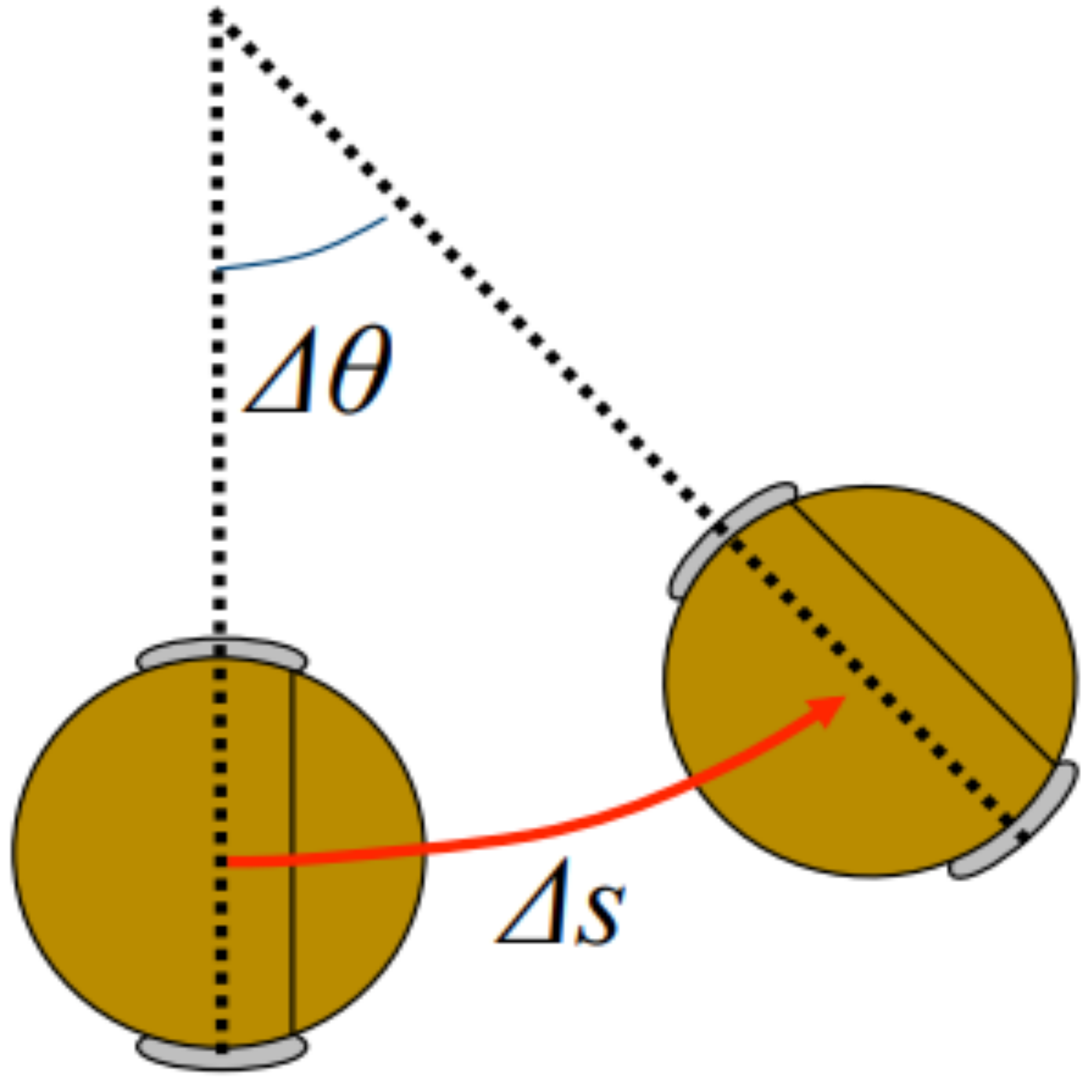


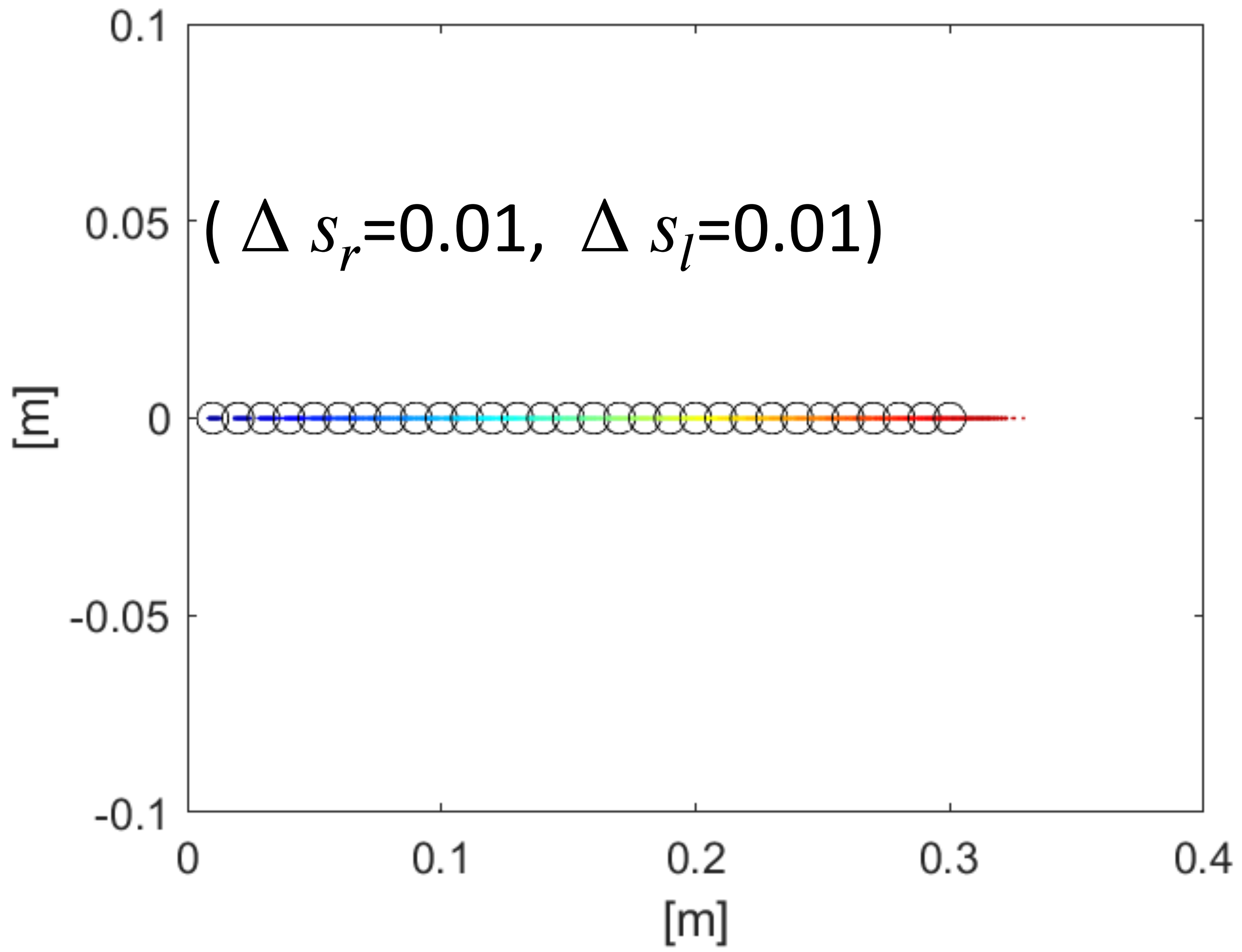
$$\Delta s = d + e_s$$

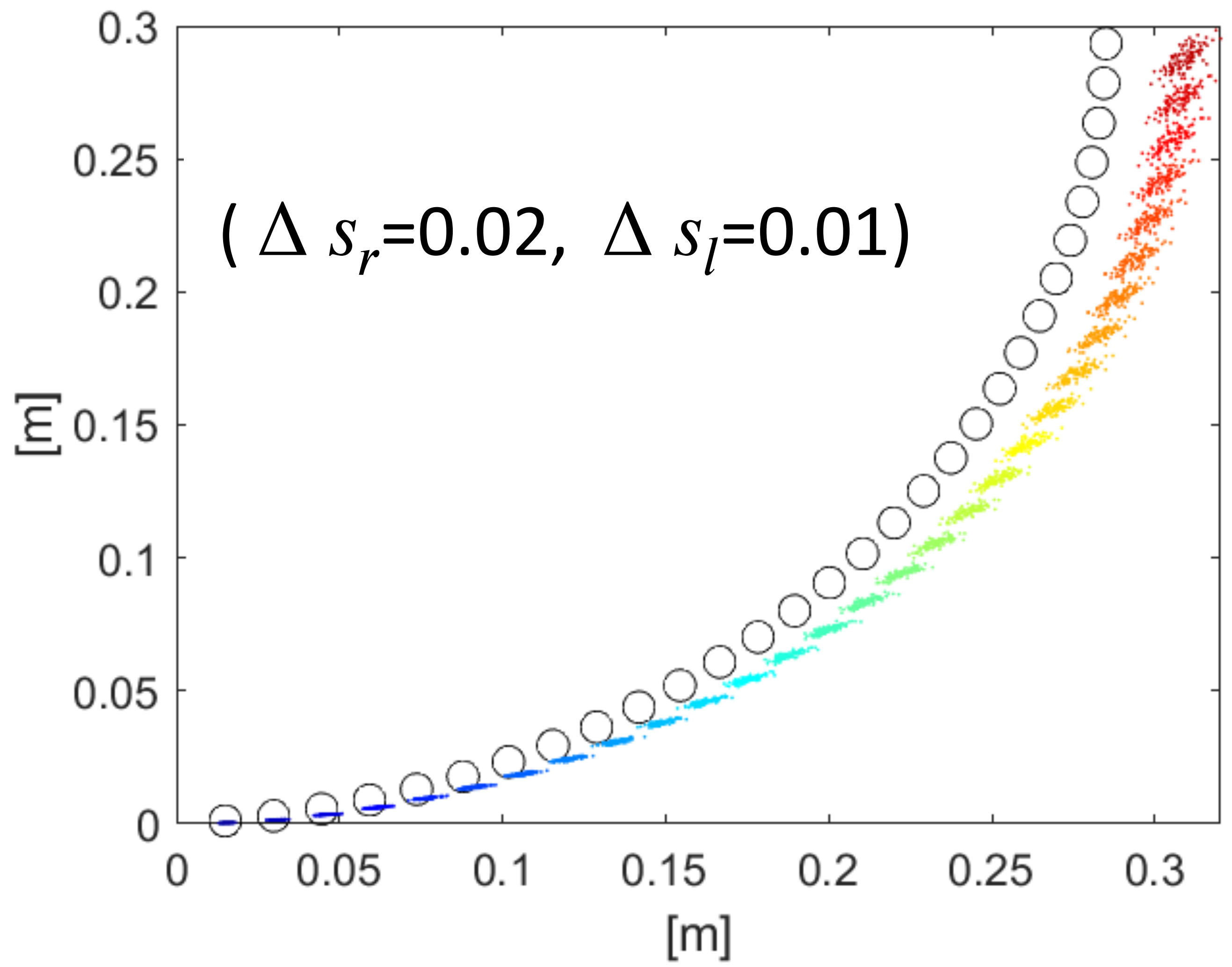
$$e_s\ (\mu_{es} = 1\text{mm},\ \sigma_{es} = 2mm)$$

$$\Delta x = \frac{\Delta s_l + \Delta s_r + e_s}{2} \cos\left(\theta + \frac{\Delta s_r - \Delta s_l}{4L}\right)$$

$$\Delta y = \frac{\Delta s_l + \Delta s_r + e_s}{2} \sin\left(\theta + \frac{\Delta s_r - \Delta s_l}{4L}\right)$$





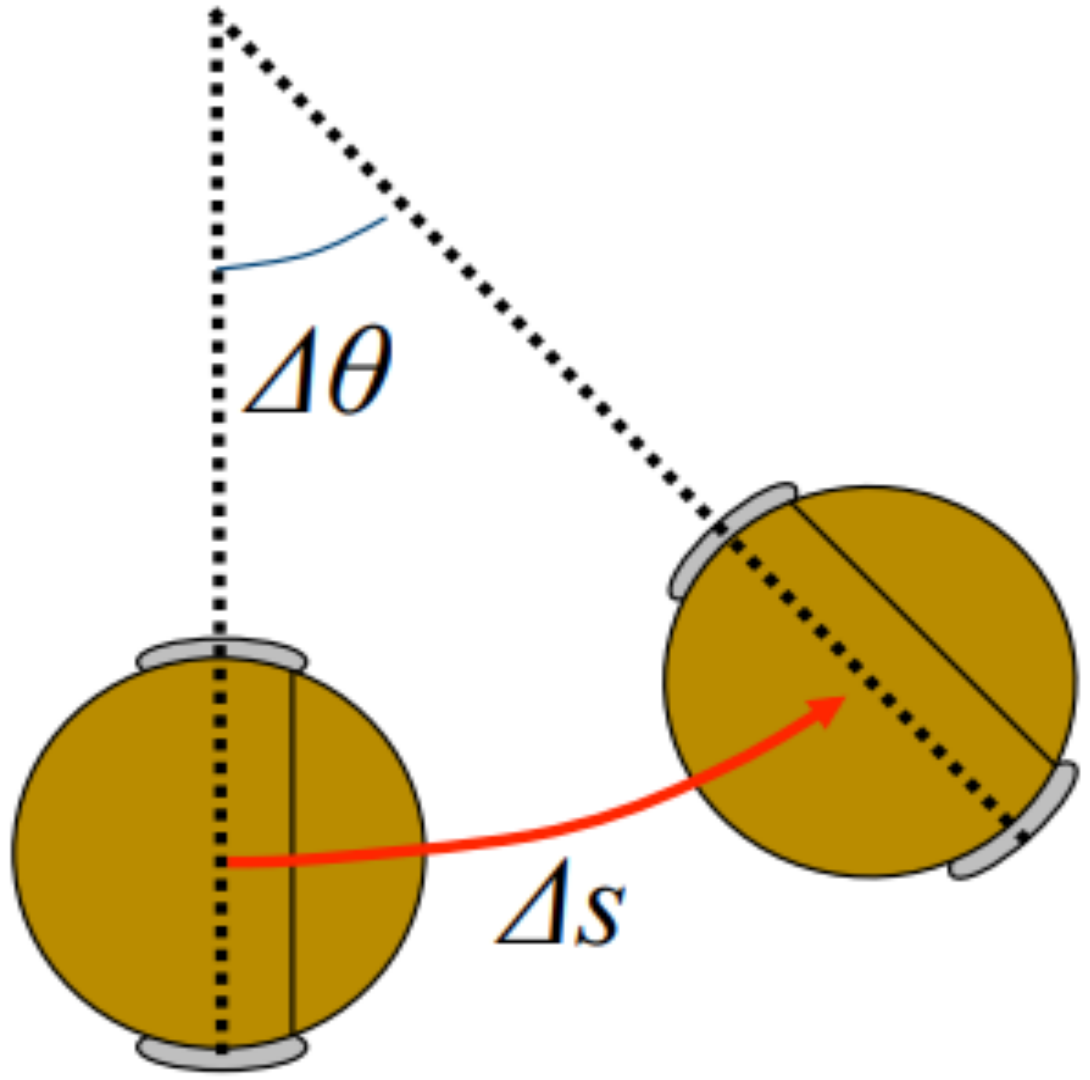


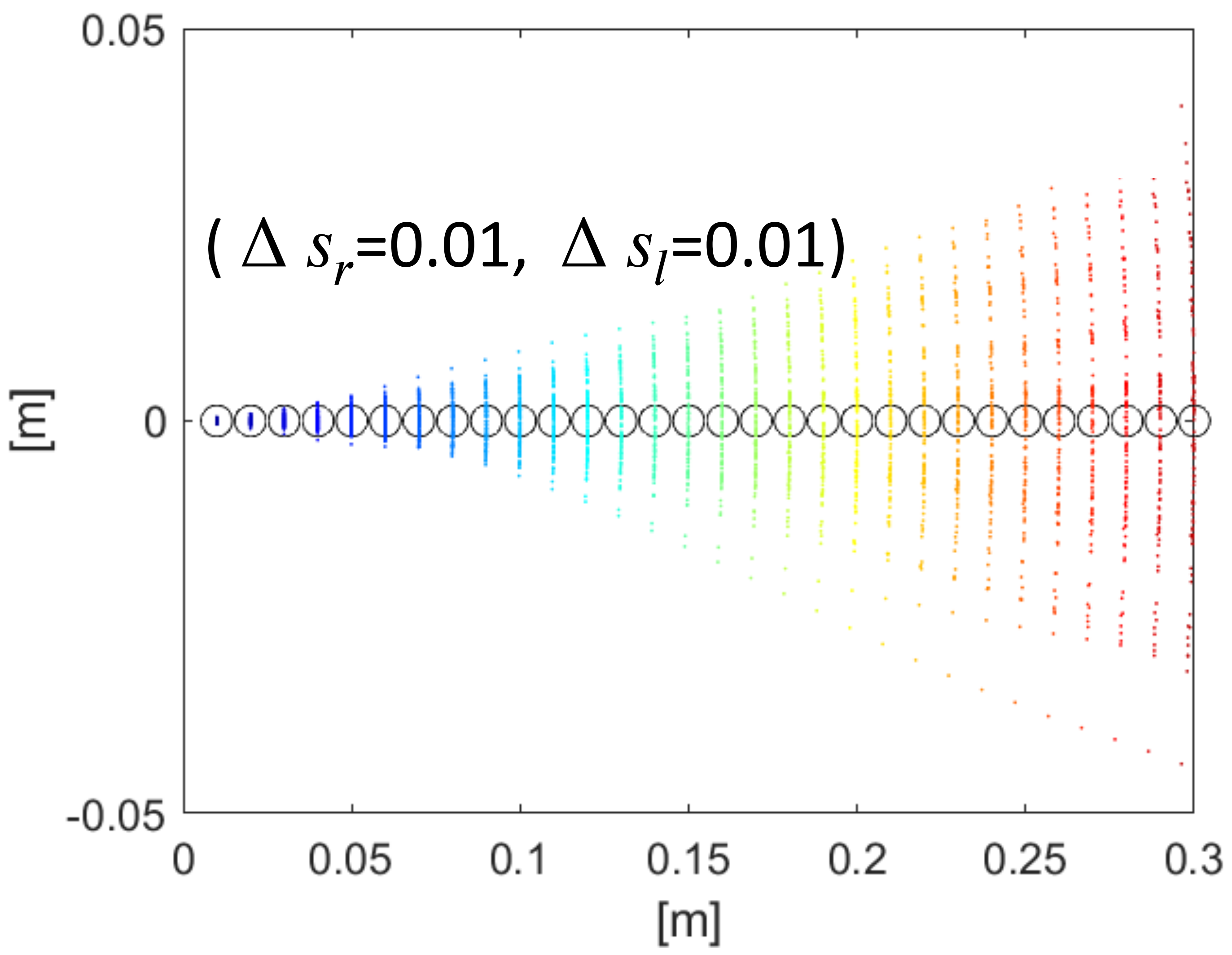
Modeling Motion

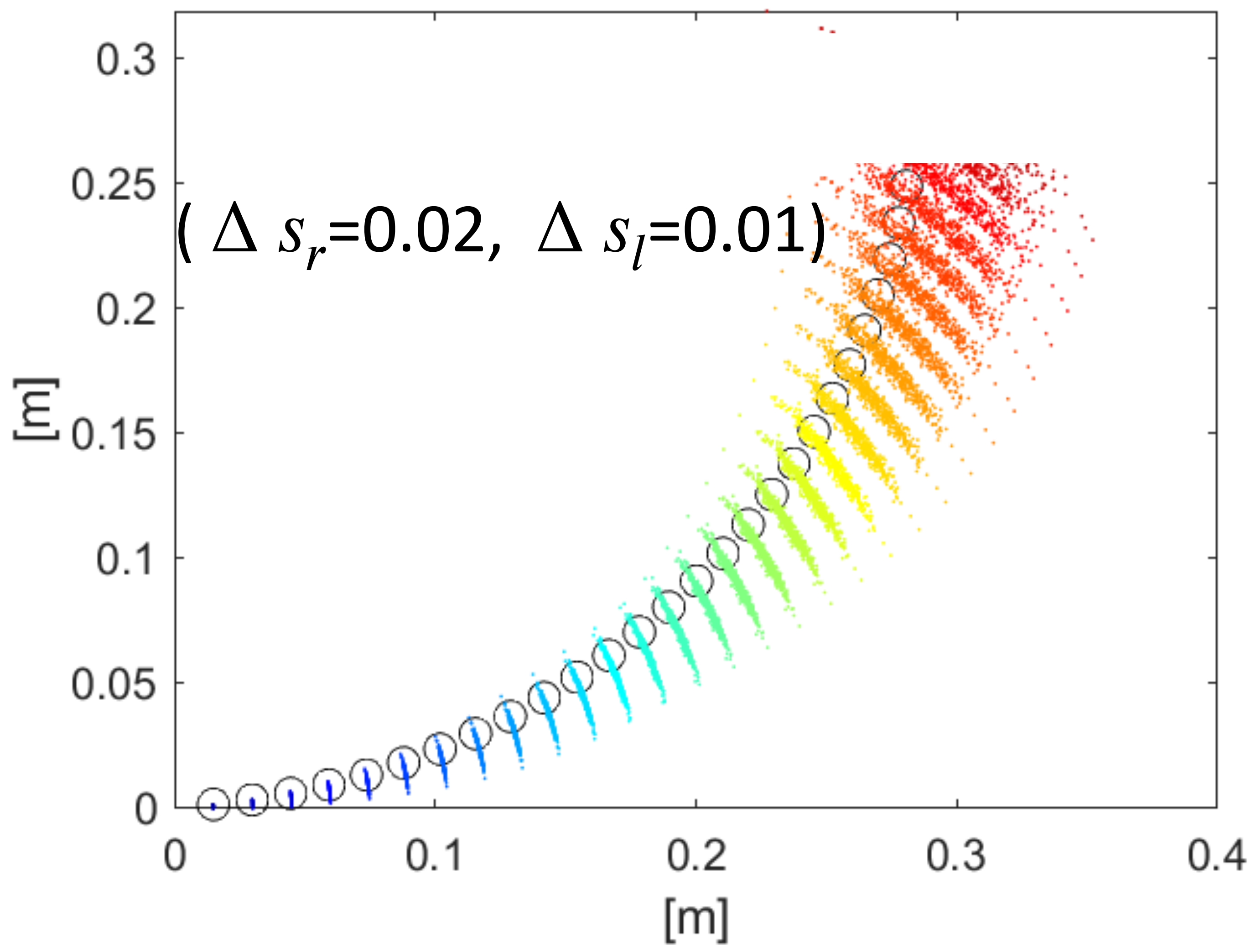
- How do wheel rotation errors propagate into positioning errors?

$$\Delta\theta = \beta + e_\theta,\ e_\theta = 1^\circ \qquad \Delta x = \Delta s \cos\left(\theta + \frac{\beta}{2} + e_\theta\right) = 0.1000\text{m}$$

$$e_\theta\ (\mu_{e\theta} = 0^\circ,\ \sigma_{e\theta} = 1^\circ) \qquad \Delta y = \Delta s \sin\left(\theta + \frac{\beta + e_\theta}{2} + e_\theta\right) = 0.0175\text{m}$$







Sources and references

- http://www.cs.cmu.edu/~rasc/Download/AMRobots4.pdf
- https://www.ti.com/lit/ug/sbau305b/sbau305b.pdf?ts=1599417595209&ref_url=https%253A%252F%252Fwww.google.com%252F
- https://hmc.edu/lair/ARW/ARW-Lecture01-Odometry.pdf
- Matlab Tech Talks on Sensor Fusion (https://www.youtube.com/watch?v=6qV3YjFppuc)
- Prof. Kirstin Petersen

Lab 1B: Programming

Class Action Items

- If you want to drop the class, please let me know **ASAP** and return your lab kits!
- **January 31st, midnight**: Make a GitHub repository and build your Github page
  - Include: name, photo, a small introduction, and the class number
  - Share **the page link** in the canvas assignment
- **February 4th (8am)** for Lab 401, and **February 5th (8am)** for Labs 402 & 403: Lab 1A and Lab 1B write-ups are due!