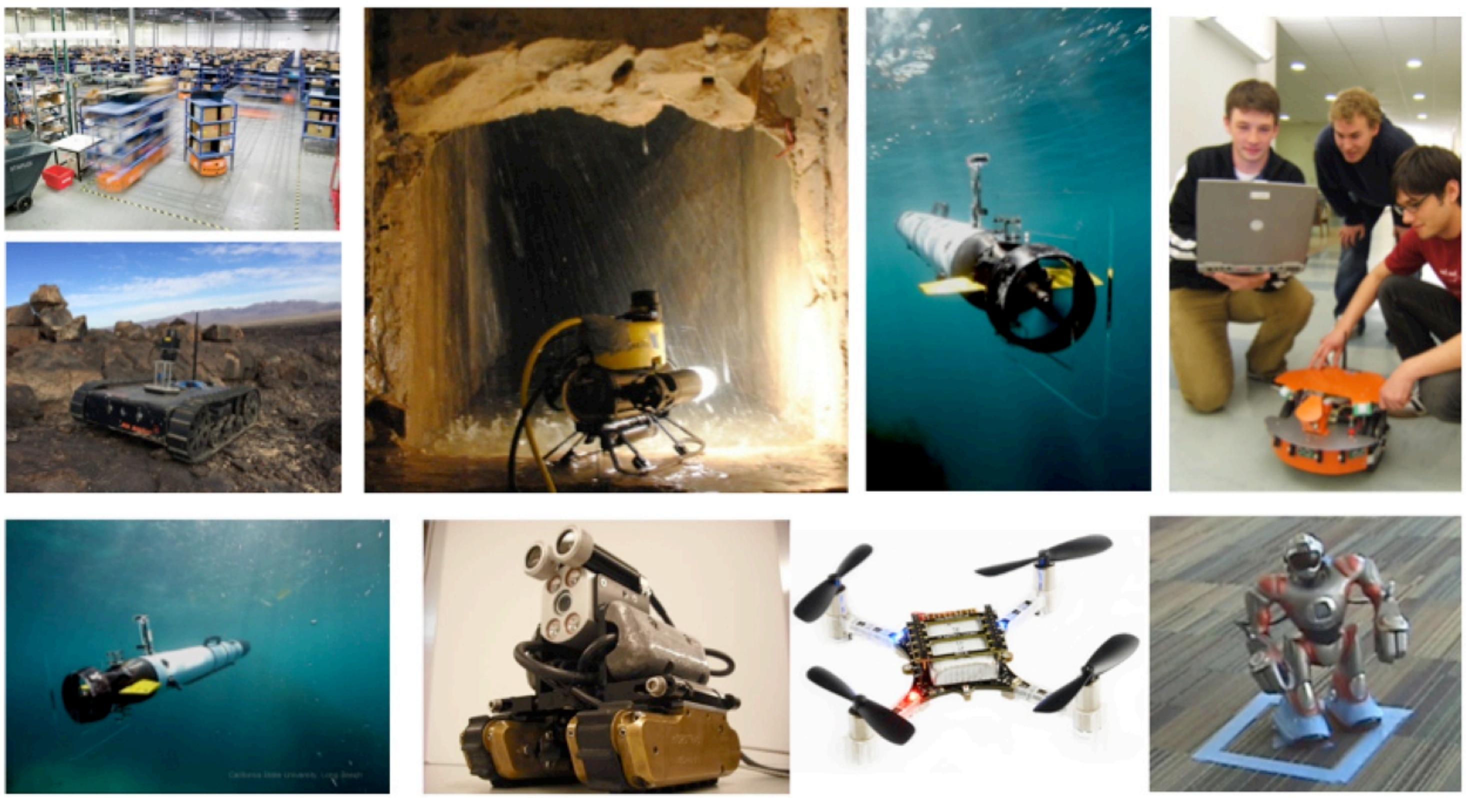
ARW – Lecture 01
Odometry Kinematics

Instructor: Chris Clark

Semester: Summer 2016

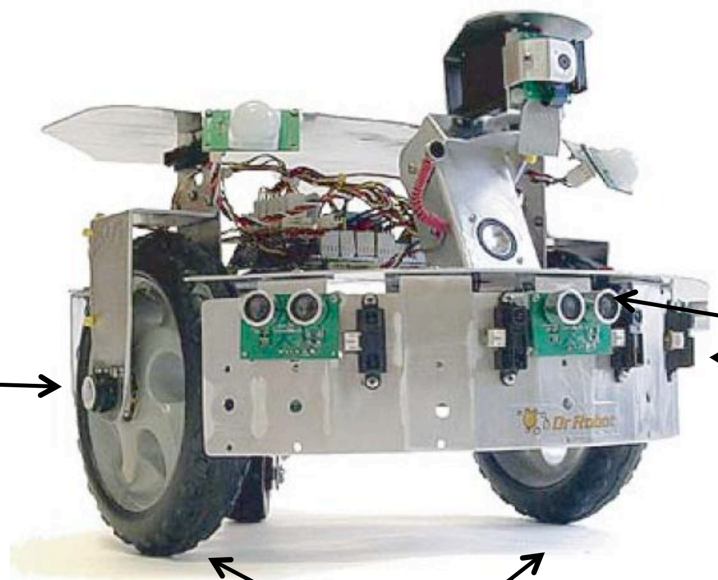
Figures courtesy of Siegwart & Nourbakhsh

Introduction



Different Bots

Wheel
Encoders

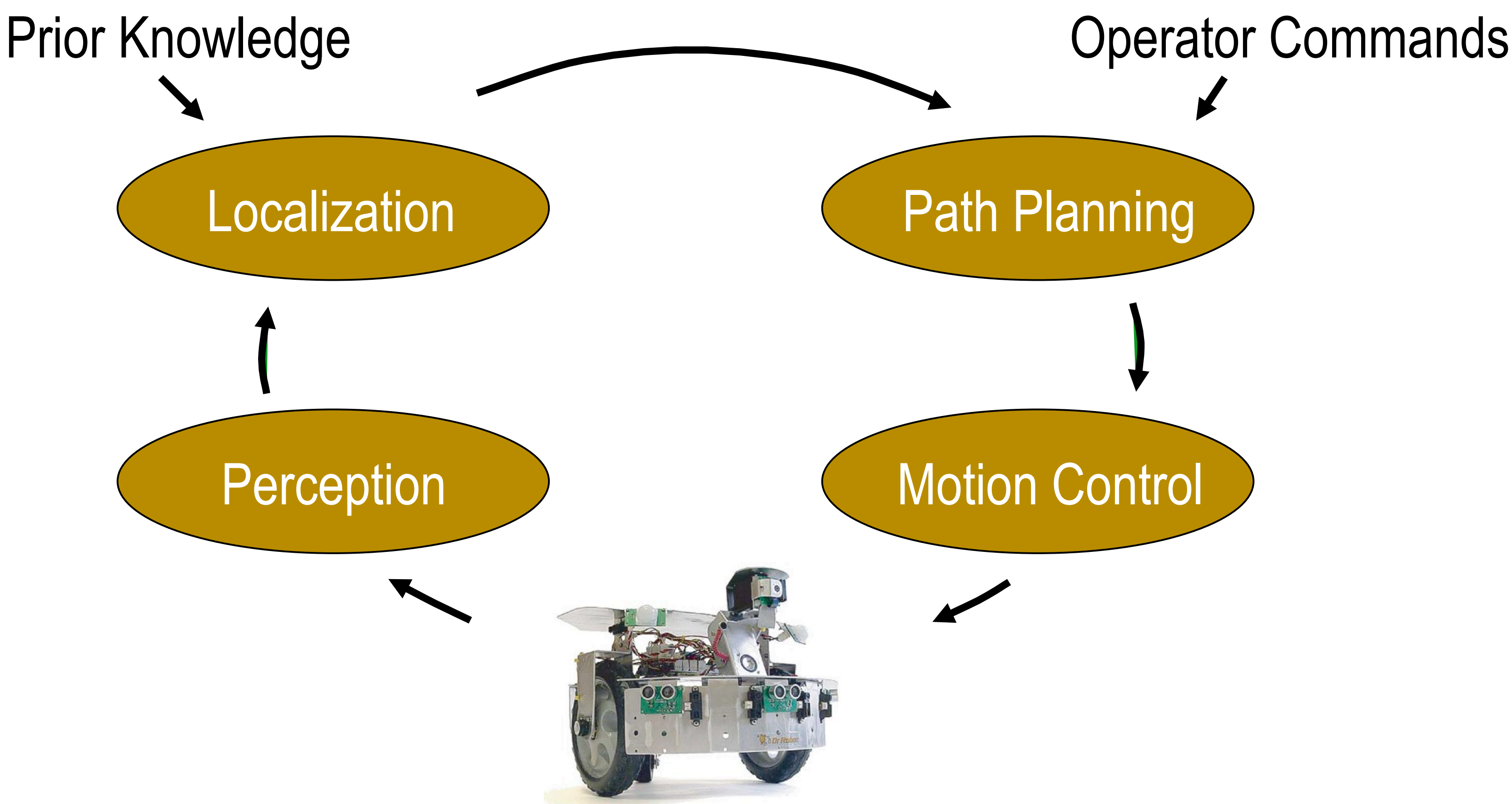


Range
Sensors

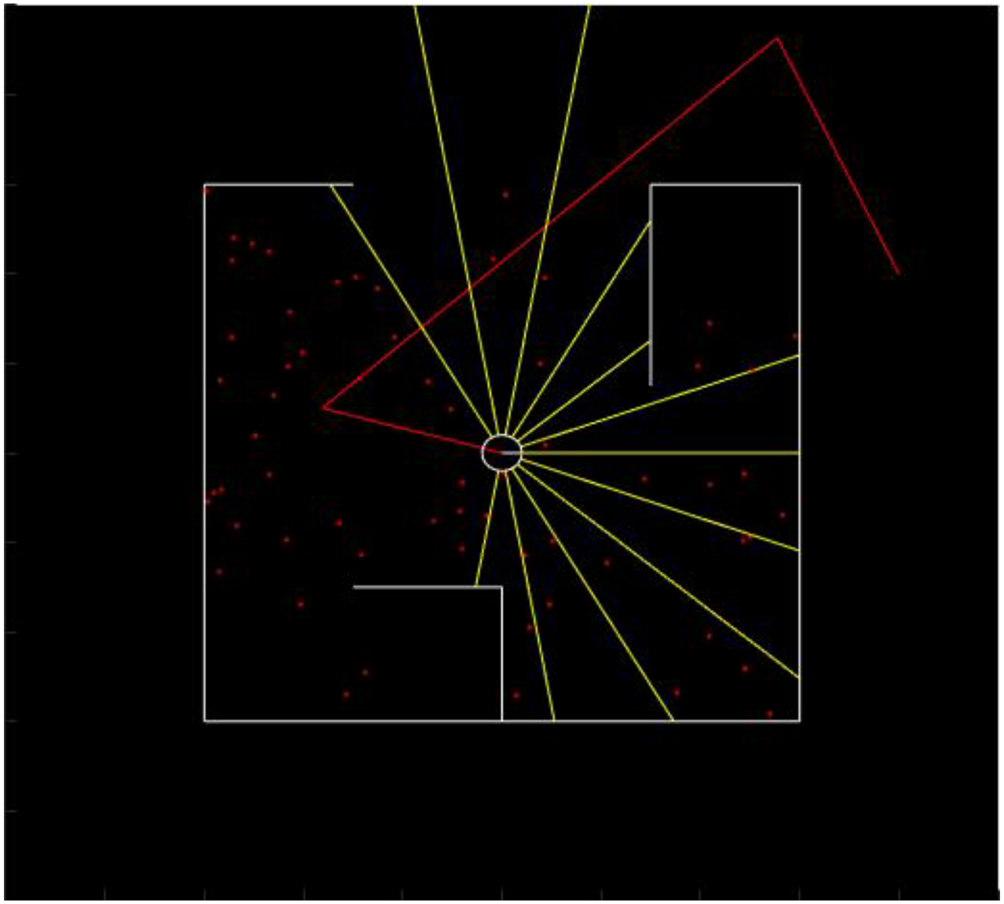
Differential Drive
Motor Configuration

Different Bots

Planning Based Control



ARW Goals



Odometry Kinematics

- Lecture Goal

  - Develop an equation that maps the previous robot state and wheel encoder measurements to the new robot state.

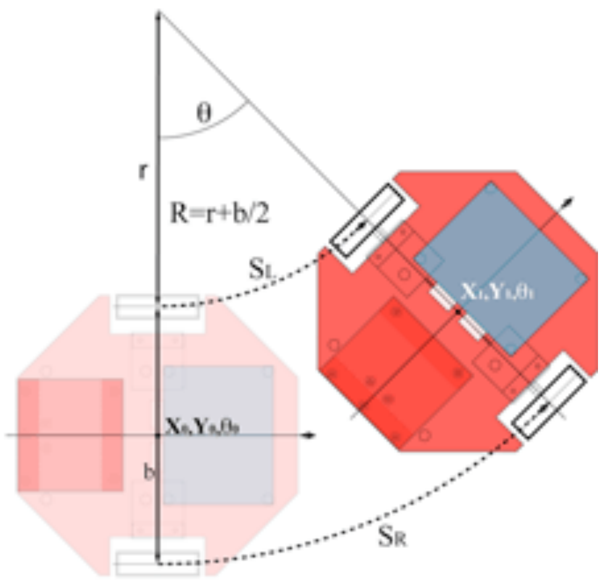
$$X_t = f(X_{t-1}, U_{t-1})$$

Odometry Kinematics

1. Odometry & Dead Reckoning
2. Modeling motion – The X80
3. Modeling motion – An ROV
4. Odometry in your Sim

Odometry & Dead Reckoning

- Odometry
  - Use wheel sensors to update position
- Dead Reckoning
  - Use wheel sensors and heading sensor to update position
- Straight forward to implement
- Errors are integrated, unbounded



http://www.guiott.com

Odometry & Dead Reckoning

- Odometry Error Sources?

Odometry & Dead Reckoning

- Odometry Error Sources?
  - Limited **resolution** during integration
  - Unequal **wheel diameter**
  - Variation in the **contact** point of the wheel
  - Unequal **floor** contact and variable friction can lead to slipping

Odometry & Dead Reckoning

- Odometry Error Sources?

Odometry & Dead Reckoning

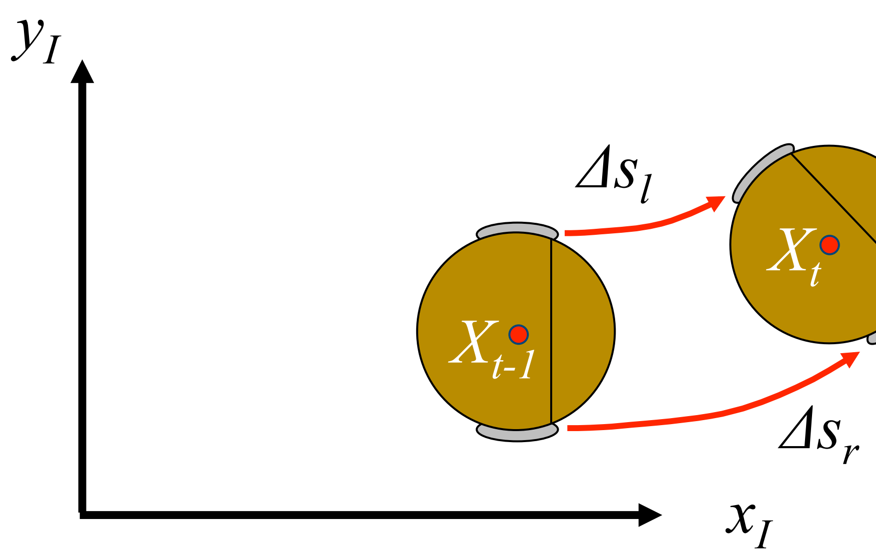
- Odometry Errors
  - **Deterministic** errors can be eliminated through proper calibration
  - **Non-deterministic** errors have to be described by error models and will always lead to uncertain position estimate.

Odometry Kinematics

1. Odometry & Dead Reckoning
2. Modeling motion – The X80
3. Modeling motion – An ROV
4. Odometry in your Sim

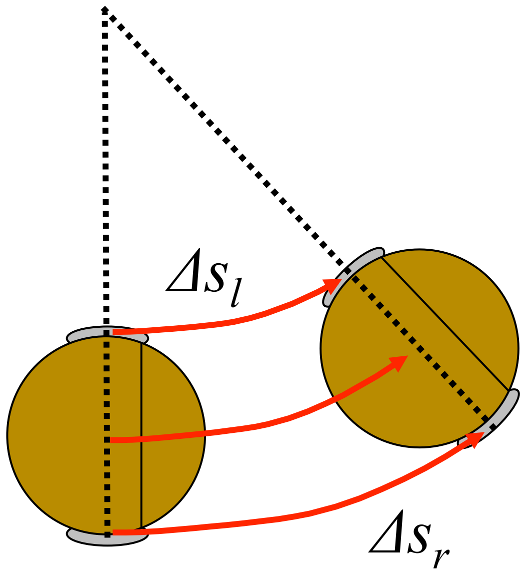
Modeling Motion

- If a robot starts from a position X_{t-1}, and the right and left wheels move respective distances Δs_r and Δs_l, what is the resulting new position X_t ?



Modeling Motion

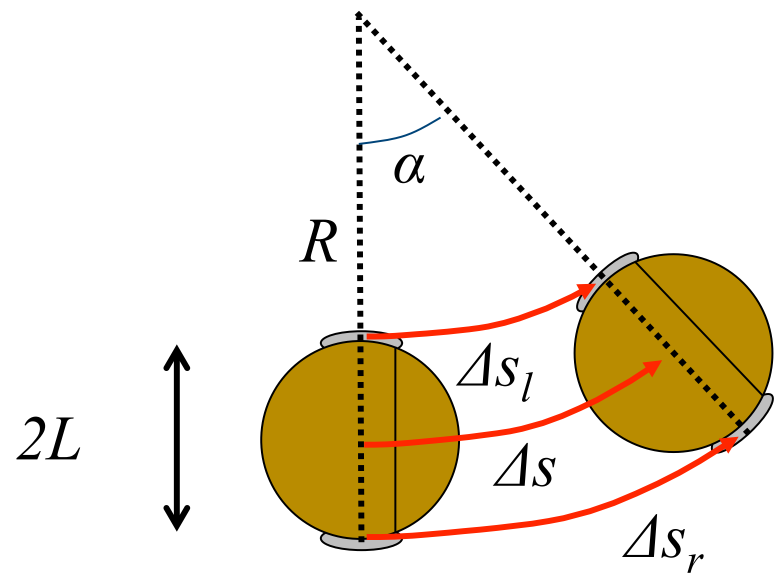
- To start, let's model the change in angle $\Delta\theta$ and distance travelled Δs by the robot.
  - Assume the robot is travelling on a circular arc of constant radius.



Modeling Motion

- Begin by noting the following holds for circular arcs:

$$\Delta s_l = R\alpha \qquad \Delta s_r = (R+2L)\alpha \qquad \Delta s = (R+L)\alpha$$



Modeling Motion

- Now manipulate first two equations:

$$\Delta s_l = R\alpha \qquad \Delta s_r = (R+2L)\alpha$$

To:

$$R\alpha = \Delta s_l$$

$$L\alpha = (\Delta s_r - R\alpha)/2$$

$$= \Delta s_r/2 - \Delta s_l/2$$

Modeling Motion

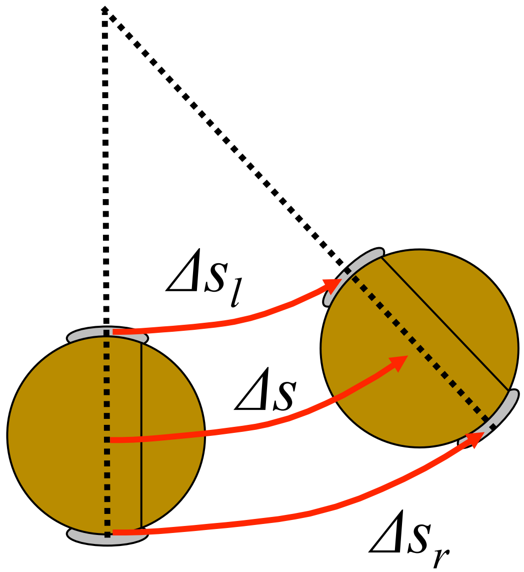
- Substitute this into last equation for Δs:

$$
\begin{aligned}
\Delta s &= (R+L)\alpha \\
&= R\,\alpha + L\alpha \\
&= \Delta s_l + \Delta s_r/2 - \Delta s_l/2 \\
&= \Delta s_l/2 + \Delta s_r/2 \\
&= \frac{\Delta s_l + \Delta s_r}{2}
\end{aligned}
$$

Modeling Motion

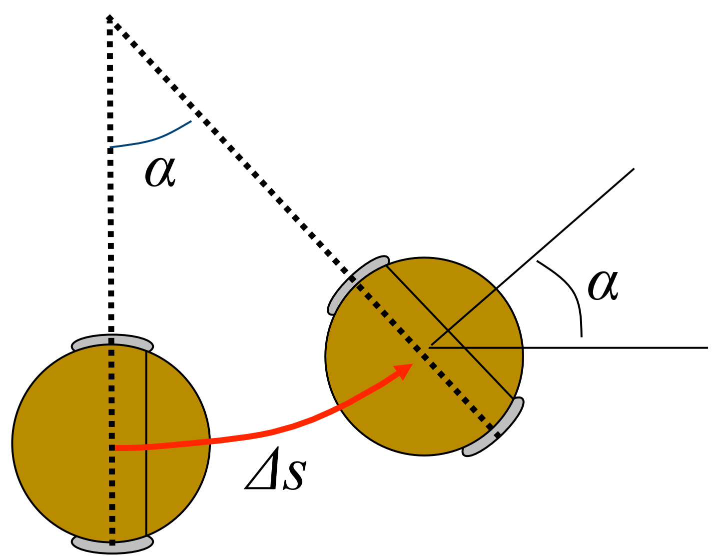
- Or, note the distance the center travelled is simply the average distance of each wheel:

$$\Delta s = \frac{\Delta s_r + \Delta s_l}{2}$$



Modeling Motion

- To calculate the change in angle $\Delta\theta$, observe that it equals the rotation about the circular arc's center point



$$\Delta\theta = \alpha$$

Modeling Motion

- So we solve for α by equating α from the first two equations*:*

$$\Delta s_l = R\alpha \qquad \Delta s_r = (R+2L)\alpha$$

This results in:

$$\Delta s_l / R = \Delta s_r / (R+2L)$$

$$(R+2L)\,\Delta s_l = R\,\Delta s_r$$

$$2L\,\Delta s_l = R\,(\Delta s_r - \Delta s_l)$$

$$\frac{2L\,\Delta s_l}{(\Delta s_r - \Delta s_l)} = R$$

Modeling Motion

- Substitute R into

$$\alpha = \Delta s_l / R$$

$$= \Delta s_l\, (\Delta s_r - \Delta s_l) / (2L\, \Delta s_l)$$

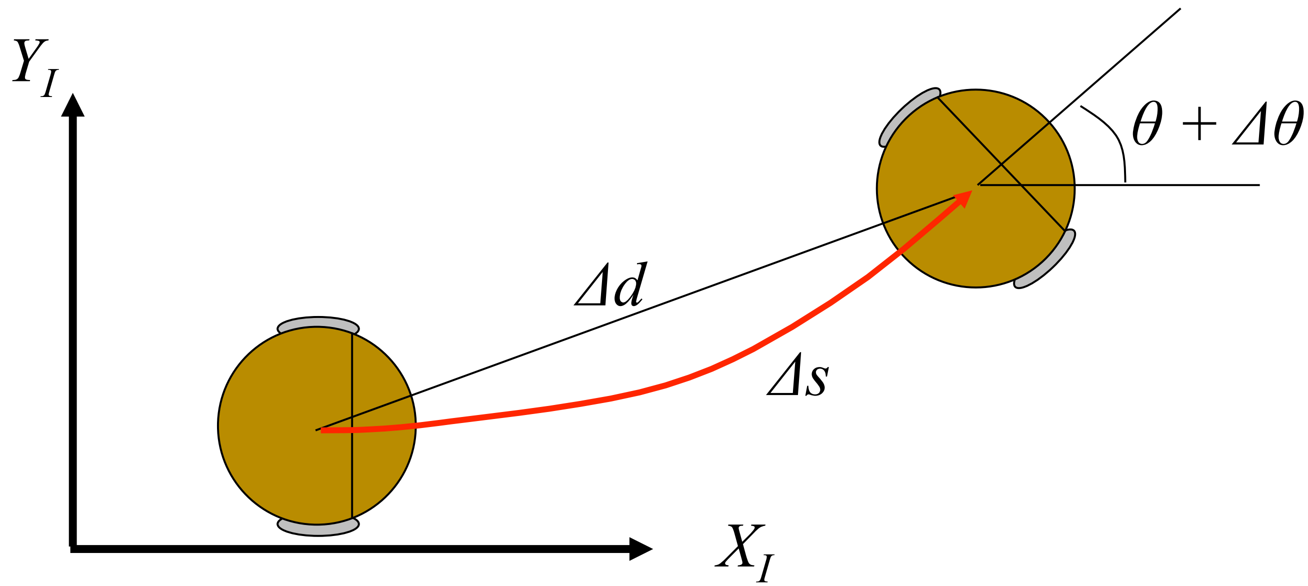
$$= \frac{(\Delta s_r - \Delta s_l)}{2L}$$

So…

$$\Delta\theta = \frac{(\Delta s_r - \Delta s_l)}{2L}$$

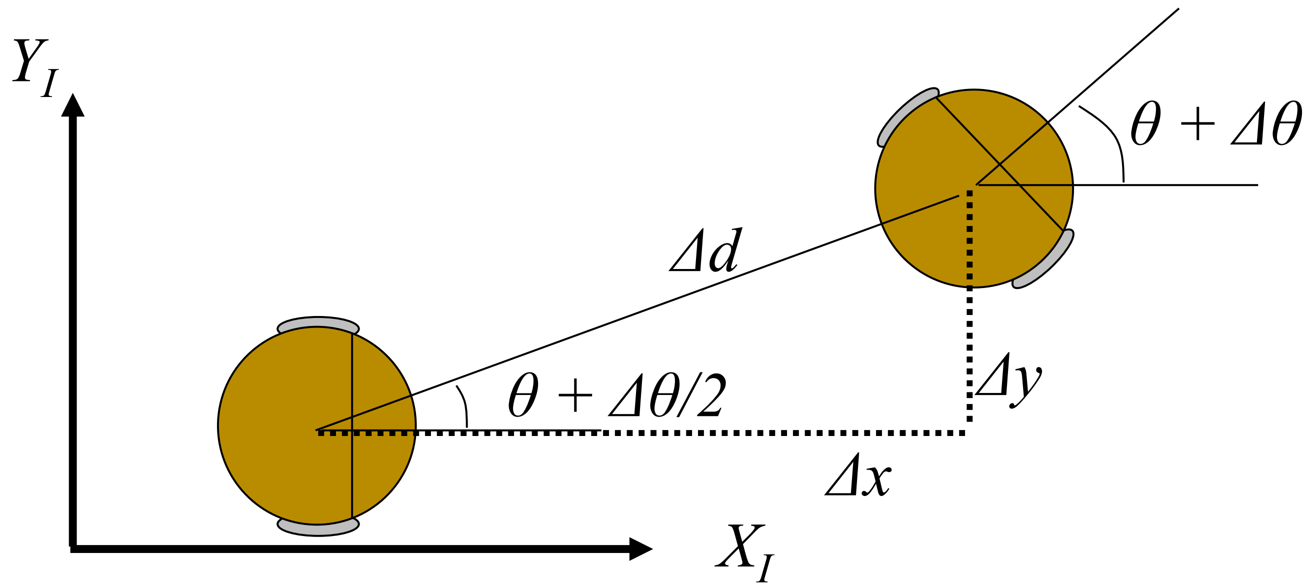
Modeling Motion

- Now that we have $\Delta\theta$ and Δs, we can calculate the position change in global coordinates.
  - We use a new segment of length Δd.



Modeling Motion

- Now calculate the change in position as a function of Δd.

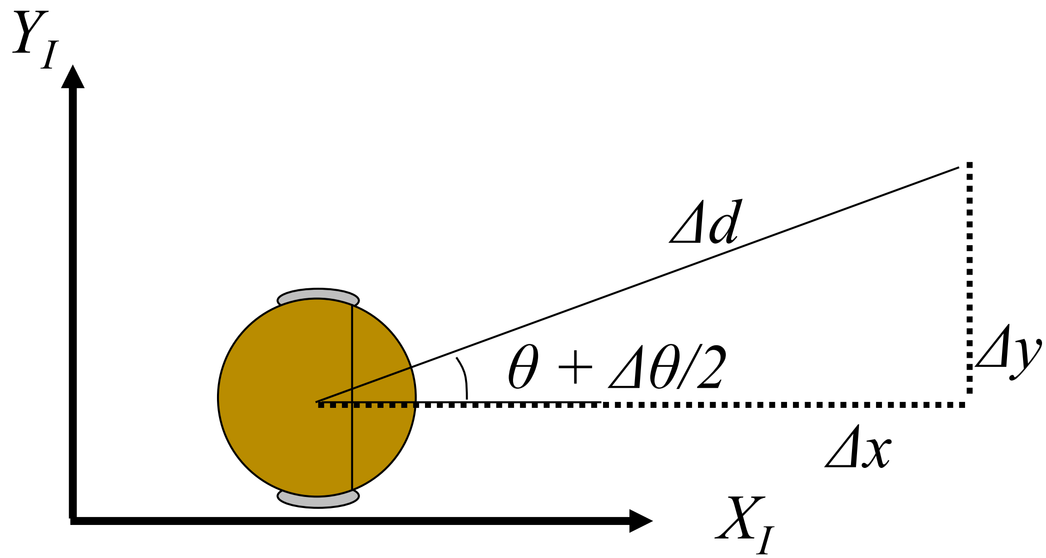


Modeling Motion

- Using Trig:

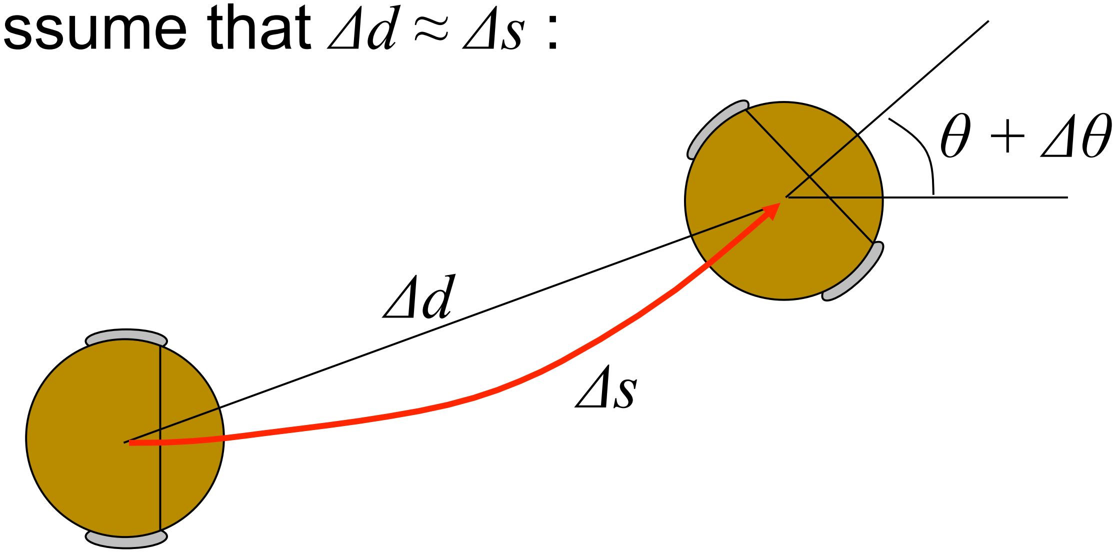
$$\Delta x = \Delta d \cos(\theta + \Delta\theta/2)$$

$$\Delta y = \Delta d \sin(\theta + \Delta\theta/2)$$



Modeling Motion

- Now if we assume that the motion is small, then we can assume that $\Delta d \approx \Delta s$:



- So…

$$\Delta x = \Delta s \cos(\theta + \Delta\theta/2)$$
$$\Delta y = \Delta s \sin(\theta + \Delta\theta/2)$$

Modeling Motion

- Summary:

$$\Delta x = \Delta s \cos(\theta + \Delta\theta / 2)$$

$$\Delta y = \Delta s \sin(\theta + \Delta\theta / 2)$$

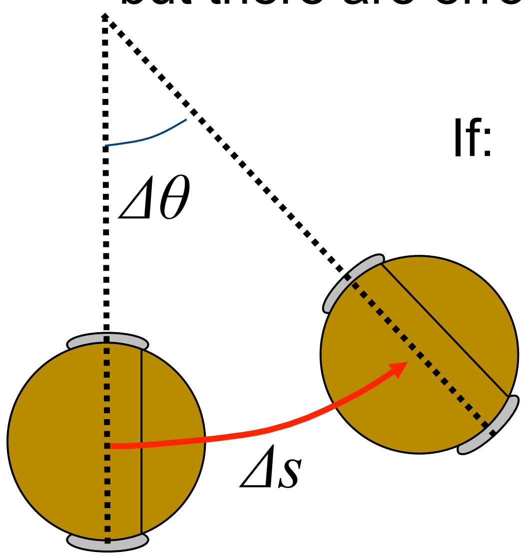
$$\Delta\theta = \frac{\Delta s_r - \Delta s_l}{2L}$$

$$\Delta s = \frac{\Delta s_r + \Delta s_l}{2}$$

$$X_t = f(x, y, \theta, \Delta s_r, \Delta s_l) = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \frac{\Delta s_r + \Delta s_l}{2} \cos\left(\theta + \frac{\Delta s_r - \Delta s_l}{4L}\right) \\ \frac{\Delta s_r + \Delta s_l}{2} \sin\left(\theta + \frac{\Delta s_r - \Delta s_l}{4L}\right) \\ \frac{\Delta s_r - \Delta s_l}{2L} \end{bmatrix}$$

Modeling Uncertainty in Motion

- Let's consider wheel rotation measurement errors, and see how they propagate into positioning errors.
  - Example: the robot actually moved forward 1 m on the x axis, but there are errors in measuring this.



If:

$$\Delta s = 1 + e_s$$

$$\Delta\theta = 0 + e_\theta$$

where e_s and e_θ are error terms

Modeling Uncertainty in Motion

- According to the following equations, the error $e_s = 0.001\text{m}$ produces errors in the direction of motion.

$$\Delta x = \Delta s \cos(\theta + \Delta\theta/2)$$

$$\Delta y = \Delta s \sin(\theta + \Delta\theta/2)$$

- However, the $\Delta\theta$ term affects each direction differently. If $e_\theta = 2$ deg and $e_s = 0$ meters, then:

$$\cos(\theta + \Delta\theta/2) = 0.9998$$

$$\sin(\theta + \Delta\theta/2) = 0.0175$$

Modeling Uncertainty in Motion

- So

$$\Delta x = 0.9998$$
$$\Delta y = 0.0175$$

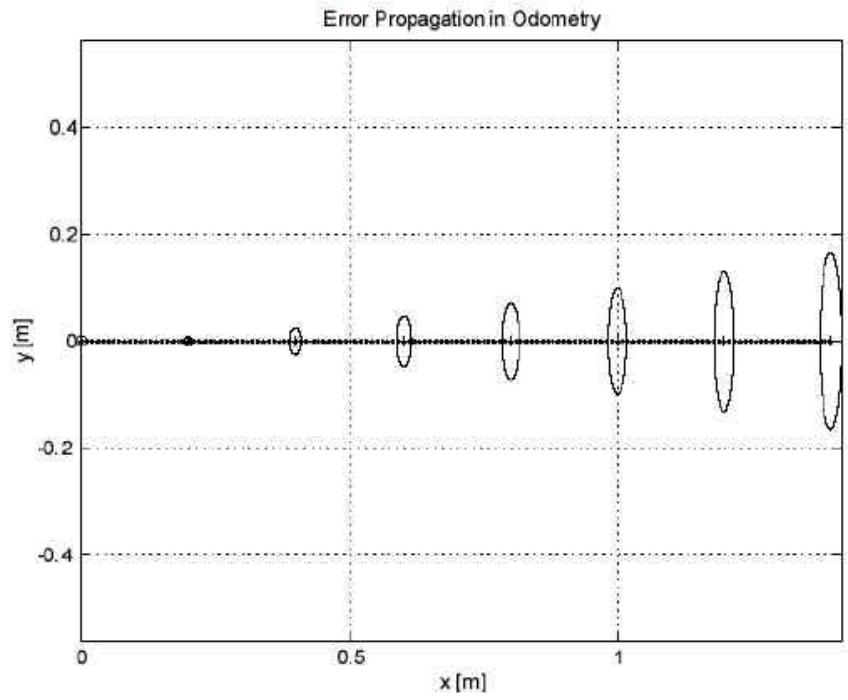
- But the robot actually went to $x = 1, y = 0$, so the errors in each direction are

$$\Delta x = +0.0002$$
$$\Delta y = -0.0175$$

- *THE ERROR IS BIGGER IN THE "Y" DIRECTION!*

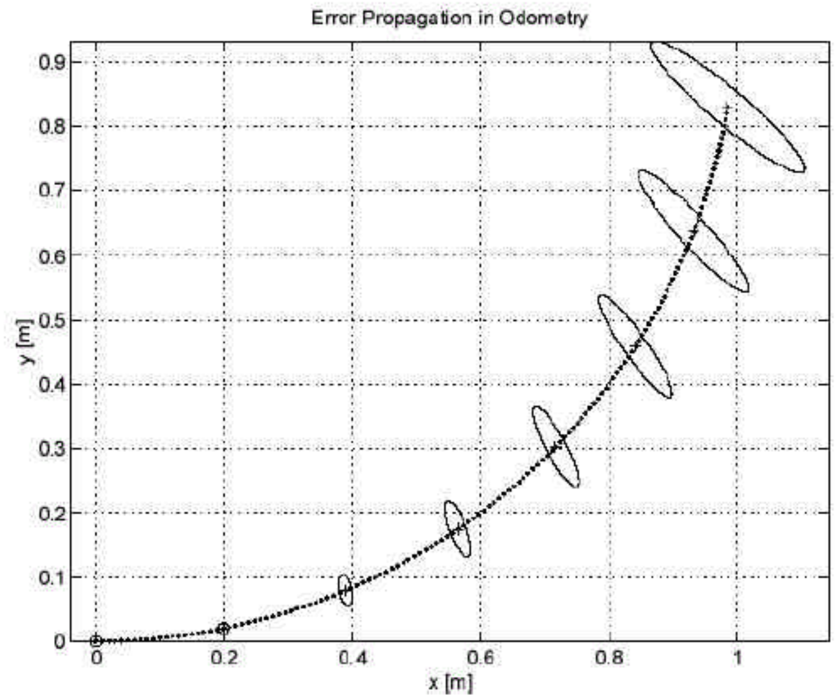
Modeling Uncertainty in Motion

- Errors perpendicular to the direction grow much larger.



Modeling Uncertainty in Motion

- Error ellipse does not remain perpendicular to direction.

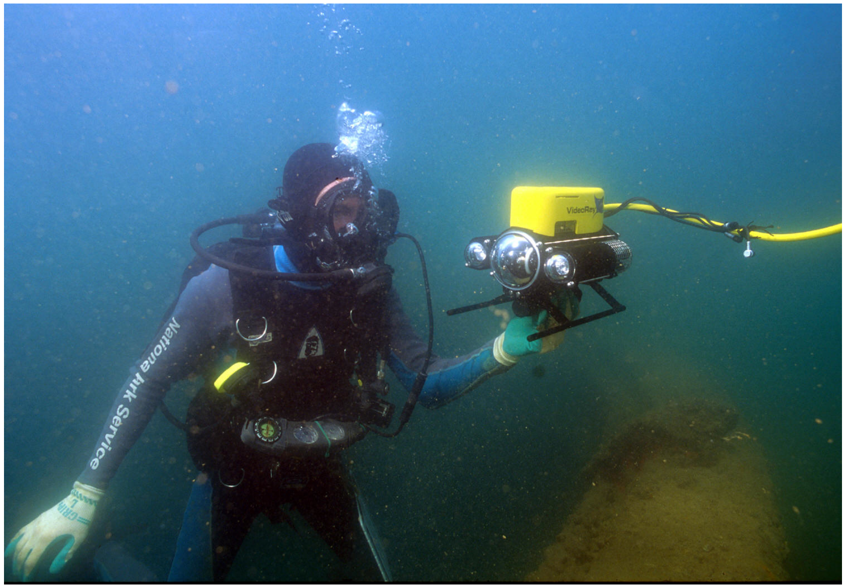


Odometry Kinematics

1. Odometry & Dead Reckoning
2. Modeling motion – The X80
3. Modeling motion – An ROV
4. Odometry in your Sim

The VideoRay MicroROV

- ROV Specs
  - Two horizontal thrusters, one vertical
  - Forward facing color camera
  - Rear facing B/W camera
  - 1.4 m/s (2.6 knots) speed
  - 152m depth rating
  - **Depth & Heading** sensors
  - SeaSprite Scanning **Sonar**



The VideoRay MicroROV

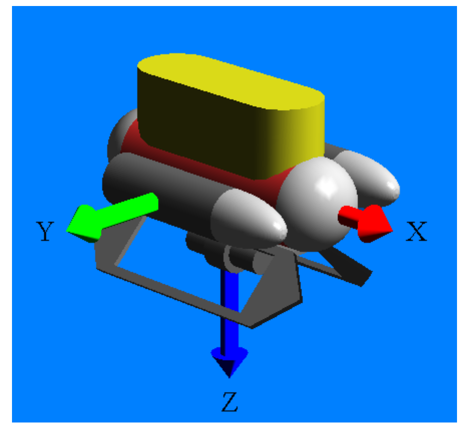
- ROV Modeling

$$
\begin{aligned}
m[\dot{u} - vr + wq - x_G(q^2 + r^2) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})] &= X \\
m[\dot{v} - wp + ur - y_G(r^2 + p^2) + z_G(qr - \dot{p}) + x_G(qp + \dot{r})] &= Y \\
m[\dot{w} - uq + vp - z_G(p^2 + q^2) + x_G(rp - \dot{q}) + y_G(rq + \dot{p})] &= Z \\
I_x\dot{p} + (I_z - I_y)qr - (\dot{r} + pq)I_{xz} + (r^2 - q^2)I_{yz} + (pr - \dot{q})I_{xy} & \\
+m[y_G(\dot{w} - uq + vp) - z_G(\dot{v} - wp + ur)] &= K \\
I_y\dot{q} + (I_x - I_z)rp - (\dot{p} + qr)I_{xy} + (p^2 - r^2)I_{zx} + (qp - \dot{r})I_{yz} & \\
+m[z_G(\dot{u} - vr + wq) - x_G(\dot{w} - uq + vp)] &= M \\
I_z\dot{r} + (I_y - I_x)pq - (\dot{q} + rp)I_{yz} + (q^2 - p^2)I_{xy} + (rq - \dot{p})I_{zx} & \\
+m[x_G(\dot{v} - wp + ur) - y_G(\dot{u} - vr + wq)] &= N
\end{aligned}
$$

Equations of Motion

- 6 degrees of freedom (DOF):
- State vectors:
  body-fixed velocity vector:
  earth-fixed pos. vector:

$$\nu = \left[\nu_1^T, \nu_2^T\right]^T = \left[u, v, w, p, q, r\right]^T$$
$$\eta = \left[\eta_1^T, \eta_1^T\right]^T = \left[x, y, z, \phi, \theta, \psi\right]^T$$



DOF	Surge	Sway	Heave	Roll	Pitch	Yaw
Velocities	u	v	w	p	q	r
Position & Attitude	x	y	z	ϕ	θ	ψ
Forces & Moments	X	Y	Z	K	M	N

Equations of Motion

- Initial Assumptions
  - The ROV will usually move with low velocity when on mission
  - Almost three planes of symmetry;
  - Vehicle is assumed to be performing non-coupled motions.

[W. Wang et al., 2006]

Equations of Motion

- Horizontal Plane:

$$
\begin{aligned}
m_{11}\dot{u} &= -m_{22}vr + X_u u + X_{u|u|}u|u| + X \\
m_{22}\dot{v} &= m_{11}ur + Y_v v + Y_{v|v|}v|v|, \\
I\dot{r} &= N_r r + N_{r|r|}r|r| + N,
\end{aligned}
$$

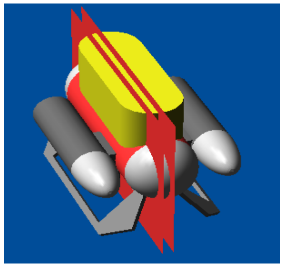
- Vertical Plan:

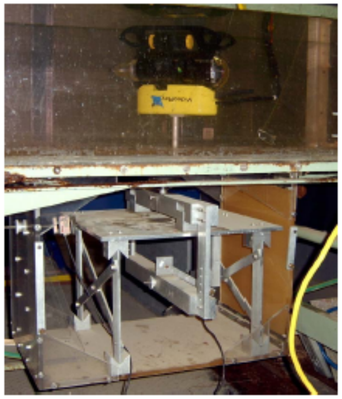
$$m_{33}\dot{w} = Z_w w + Z_{w|w|}w|w| + Z$$

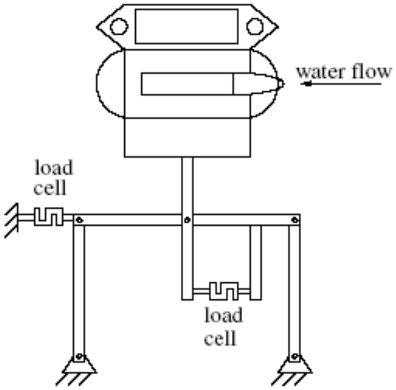
[W. Wang et al., 2006]

Theory vs. Experiment

- Coefficients for the dynamic model are pre-calculated using strip theory;
- A series of tests are carried out to validate the hydrodynamic coefficients, including
  - Propeller mapping
  - Added mass coefficients
  - Damping coefficients

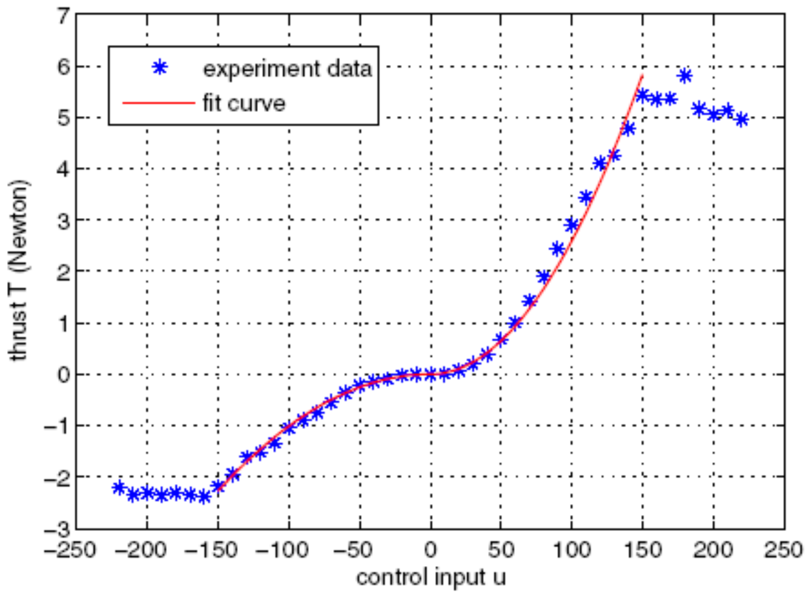






Propeller Thrust Mapping

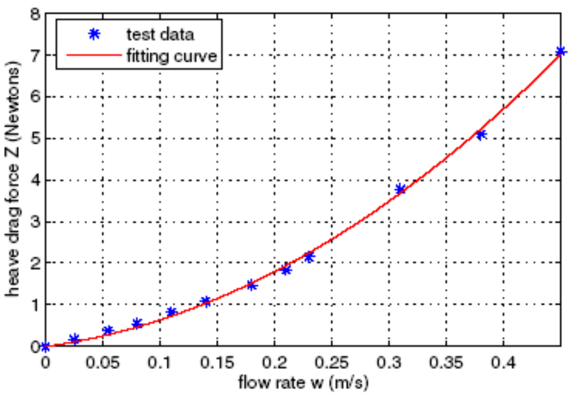
- The forward thrust can be represented as:



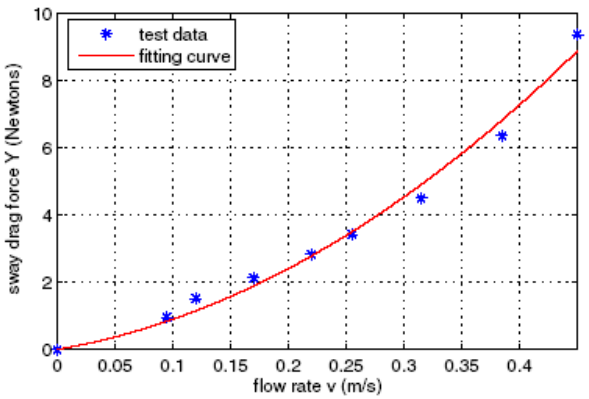
Direct Drag Forces

- The drag can be modeled as non linear functions

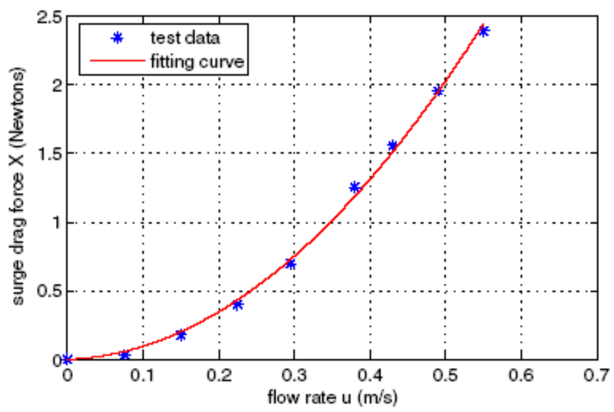
Drag in Heave (Z) Direction



Drag in Sway (Y) Direction

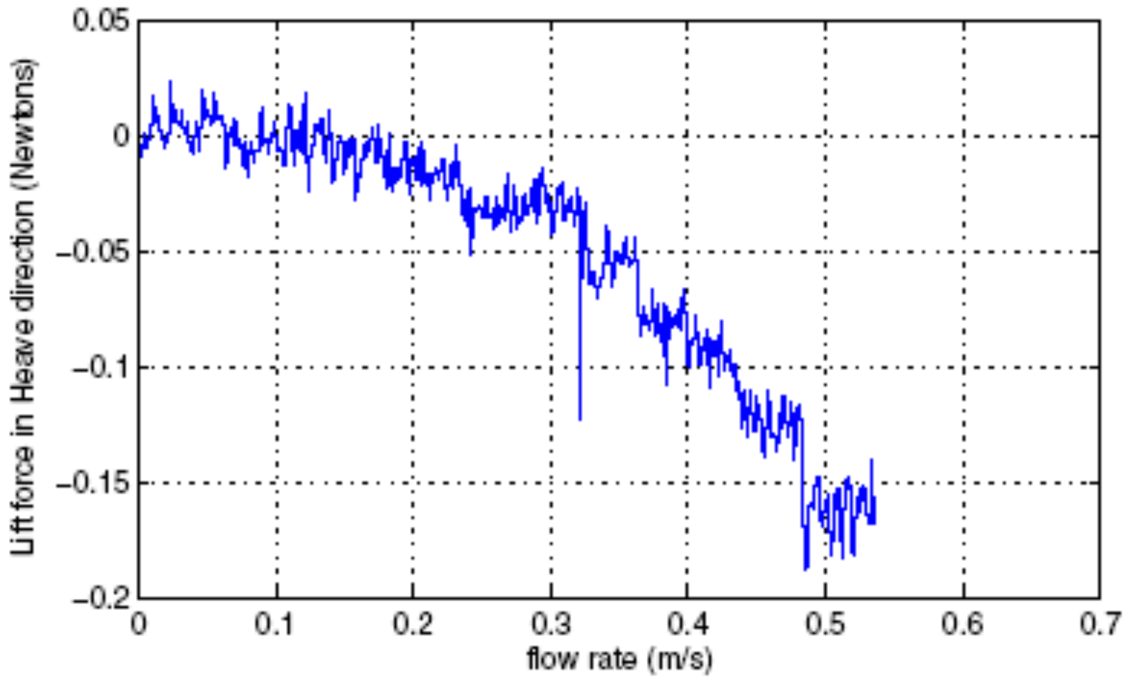


Drag in Surge (X) Direction



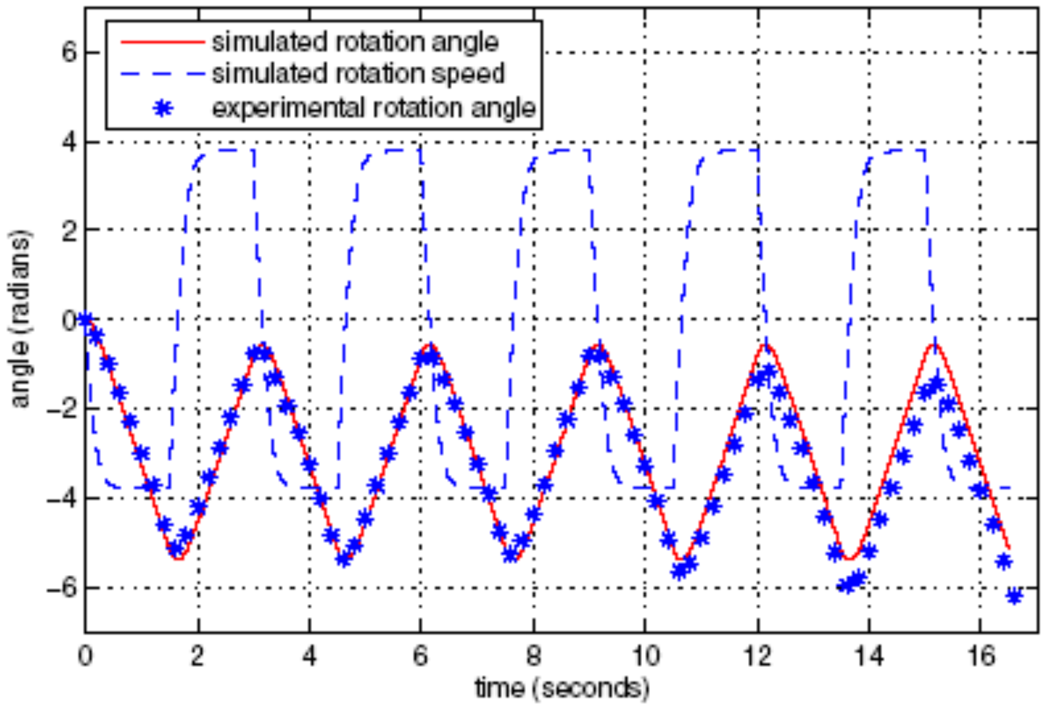
Perpendicular Drag Forces

- Heave (Z) drag from surge speed



Model Verification

- Yaw Verification



Model Verification

- Surge Verification

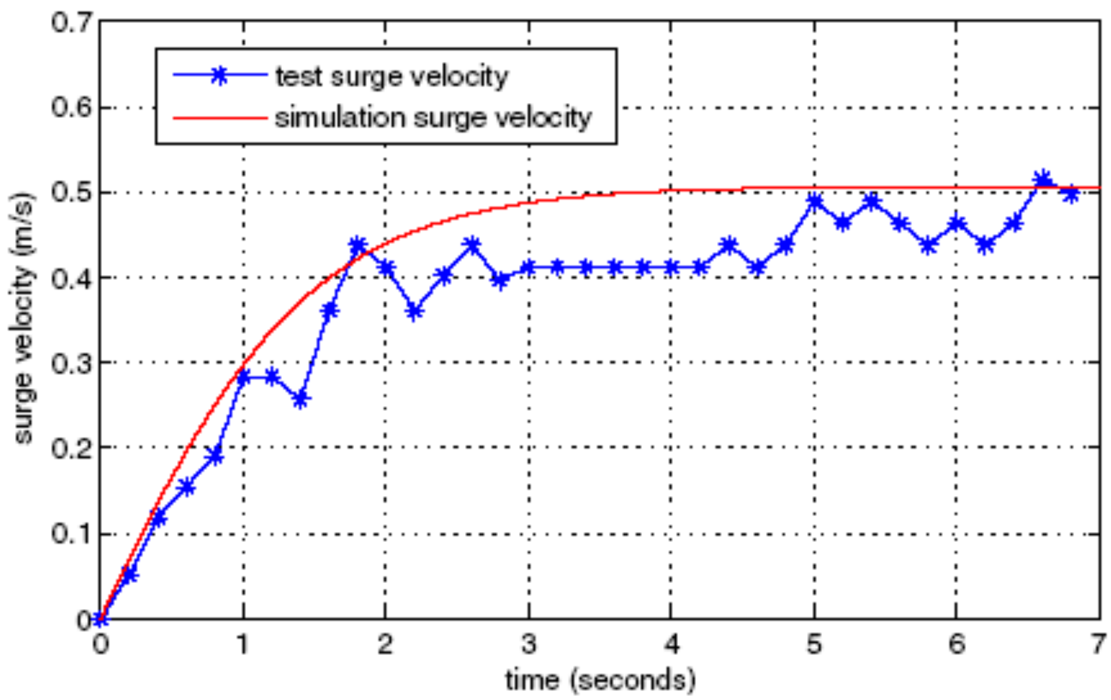


Fig. 14. Surge test experiment data and simulation result

Autonomous Control

Odometry Kinematics

1. Odometry & Dead Reckoning
2. Modeling motion – The X80
3. Modeling motion – An ROV
4. Odometry in your Sim

Odometry on the Jaguar

- Goals:
  - Calculate the resulting robot position and orientation from wheel encoder measurements.
  - Display them with the Matlab plot function

Odometry on the Jaguar

- Method cont':
  - Make use of the fact that your encoder has resolution of 4096 pulses per revolution. Be able to convert this to a distance travelled by the wheel.

$$r\varphi_r = \Delta s_r$$

  - Given the distance travelled by each wheel, we can calculate the change in the robot's distance and orientation.

$$\Delta s = \frac{\Delta s_r + \Delta s_l}{2} \qquad \Delta\theta = \frac{(\Delta s_r - \Delta s_l)}{2L}$$

Odometry on the Jaguar

- Method cont':
  - Now you should be able to update the position/ orientation in global coordinates.

$$\Delta x = \Delta s \cos(\theta + \Delta\theta/2)$$
$$\Delta y = \Delta s \sin(\theta + \Delta\theta/2)$$