Artificial Intelligence Qualifying Exam

Alli Nilles

October 3, 2017

University of Illinois at Urbana-Champaign

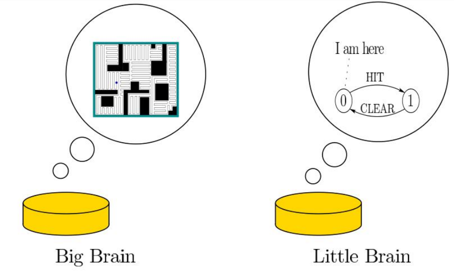
Outline

- Brief overview of my research projects
  - bouncing robots
  - improv: a high-level language for live-coding robot motion
  - morphogenesis through local cell reconfigurations
  - weaselballs (undergraduate-led project)
- *Understanding Black Box Predictions via Influence Functions*
  - deriving influence (sketch/intuition of proof)
  - validation
  - application domains
- *Generating Plans that Predict Themselves*
  - defining what makes a plan t-predictable
  - instantiation and experiments

My Research

Simple Mobile Robots

- Mobile robots can vacuum floors, transport goods in warehouses, act as security robots (patrol), etc
- We want to **minimize** sensing, computation, actuation
  - make robots less expensive, more energy efficient
- Often, robots can bump into things and be ok!
- How can we use **contact with the environment** as a strategy or source of information?



Blind, Bouncing Robots[1]

Abstract the robot as a point moving **in straight lines** in the plane, "bouncing" off the boundary at a **fixed angle** θ from the normal:

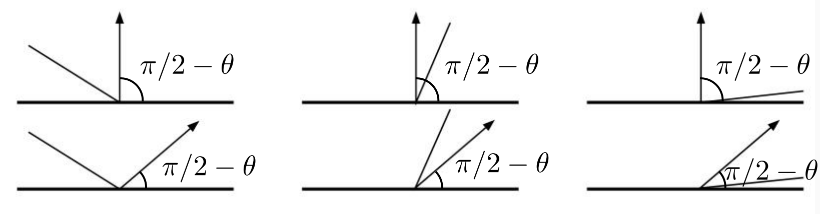


Figure 1. A point robot moving in the plane. The top row shows bounces at zero degrees from the normal. The second row shows bounces at 50 degrees clockwise from normal.

[1] (Erickson and LaValle 2013), ICRA

Research Questions

Given a constant control strategy, will the robot become "trapped" in part of the environment? Or in a certain motion pattern? We focus on **patrolling**: periodically orbiting the workspace.

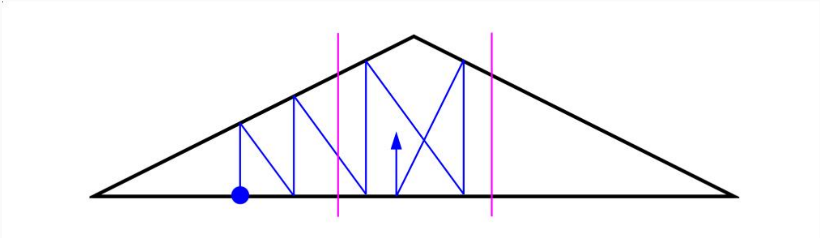
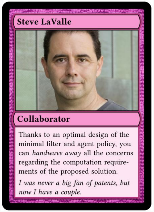


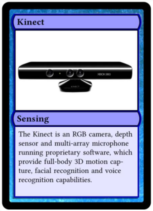
Figure 2. In this environment, bouncing at the normal, the robot will become trapped in the area between the purple lines.[3]

[3] (Erickson and LaValle 2013), ICRA

Related Work in Robotics

- Minimal sensing, actuation, computation requirements for mapping, navigating, localizing, patrolling, pursuit evasion[4]
- formalize tradeoffs between sensor and actuator power, computational complexity, energy use, etc
    - ICRA 1996 workshop, RSS ’08, ’16, ’17

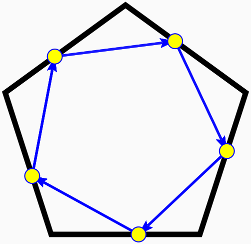




[4] Tovar, Guilamo, and LaValle (2005), Bilò et al. (2012), O’Kane and LaValle (2007), Disser (2011)

Results

- limit cycles in regular polygons
- limit cycles in convex polygons (Israel Becerra, postdoc)
- next steps: incorporate feedback control, and explore design space (other sensors, actuation strategies, etc)



Morphogenesis

With Yuliy Baryshnikov

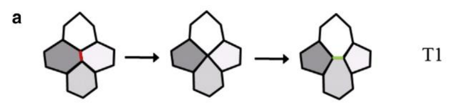


Figure 3. One type of epithelial cell reconfiguration (Fletcher et al. 2014).

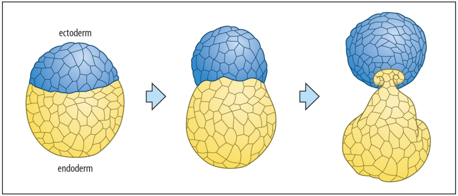


Figure 4. Morphogenesis in amphibian blastula (Staveley, n.d.).

Improv: a High-Level Language for Live-Coding Robot Motion

Weaselballs

- largely undergradute-led project
- related to *Asymmetric gear rectifies random robot motion* (Li and Zhang 2013) and *Bacterial Ratchet Motors* (Di Leonardo et al. 2010)

Common Themes?

- geometrical, topological, dynamical systems approaches
- exploiting dynamics to make simple models and controllers
- use abstractions to make better tools and programming languages for robotics
- Why AI qual?
    - context for making planners/controllers
    - need to reason about subsystems that use learning

Understanding Black Box Predictions via Influence Functions

Background

- Pang Wei Koh, and his advisor Percy Liang
- Stanford and Microsoft Research
- ICML 2017 Best Paper Award

“otherwise high-performing models are still difficult to debug and fail catastrophically in the presence of changing data distributions and adversaries. . . it is critical to build tools to help us make machine learning more reliable ‘in the wild.’ ” – Percy Liang

Problem Formulation

For a given learned model (with known loss function):

- How would the model's predictions change if we **omit** a specific training point?

Problem Formulation

For a given learned model (with known loss function):

- How would the model’s predictions change if we **omit** a specific training point?
- How would the model’s predictions change if we **perturb** a specific training point?

Problem Formulation

For a given learned model (with known loss function):

- How would the model's predictions change if we **omit** a specific training point?
- How would the model's predictions change if we **perturb** a specific training point?

Problem Formulation

For a given learned model (with known loss function):

- How would the model's predictions change if we **omit** a specific training point?
- How would the model's predictions change if we **perturb** a specific training point?

To approach these questions, study the *derivative* of the *optimal parameters*, or of the *loss*, with respect to different perturbations of a single training point.

Problem Formulation

For a given learned model (with known loss function):

- How would the model's predictions change if we **omit** a specific training point?
- How would the model's predictions change if we **perturb** a specific training point?

To approach these questions, study the *derivative* of the *optimal parameters*, or of the *loss*, with respect to different perturbations of a single training point.

When this value is larger, that training point is more *influential*.

Related Work

- statistics: Cook, Weisberg 1980: *Residuals and influence in regression*
    - focused on linear models, exact solutions
- ML:
- adversarial examples and training-set attacks

Definitions

predictor: $\mathcal{X} \to \mathcal{Y}$

Definitions

predictor: $\mathcal{X} \to \mathcal{Y}$

given training points $z_1, \ldots, z_n$, where $z_i \in \mathcal{X} \times \mathcal{Y}$

Definitions

predictor: $\mathcal{X} \to \mathcal{Y}$

given training points $z_1, \dots, z_n$, where $z_i \in \mathcal{X} \times \mathcal{Y}$

trained parameters $\theta \in \Theta$

Definitions

predictor: $\mathcal{X} \to \mathcal{Y}$

given training points $z_1, \ldots, z_n$, where $z_i \in \mathcal{X} \times \mathcal{Y}$

trained parameters $\theta \in \Theta$

loss $L(z, \theta)$ and empirical risk $R(\theta) = \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$

- approach is agnostic to loss (but assumes convex, twice-differentiable wrt θ)
- we will often use $H_{\hat{\theta}} \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} \nabla_{\theta}^{2} L(z_i, \hat{\theta})$

Definitions

predictor: $\mathcal{X} \to \mathcal{Y}$

given training points $z_1, \ldots, z_n$, where $z_i \in \mathcal{X} \times \mathcal{Y}$

trained parameters $\theta \in \Theta$

loss $L(z, \theta)$ and empirical risk $R(\theta) = \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$

- approach is agnostic to loss (but assumes convex, twice-differentiable wrt θ)
- we will often use $H_{\hat{\theta}} \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} \nabla_{\theta}^{2} L(z_i, \hat{\theta})$

empirical risk minimizer $\hat{\theta} = \arg\min_{\theta \in \Theta} R(\theta)$

Sketch of Derivation

We want to find change in model parameters if training point z is removed, but we don't want to retrain

Instead, weight z by ϵ:

$$\hat{\theta}_{\epsilon,z} = \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z, \theta)$$

Sketch of Derivation

We want to find change in model parameters if training point z is removed, but we don't want to retrain

Instead, weight z by ϵ:

$$\hat{\theta}_{\epsilon,z} = \arg\min_{\theta\in\Theta} \frac{1}{n}\sum_{i=1}^{n} L(z_i,\theta) + \epsilon L(z,\theta)$$

With $\Delta_\epsilon = \hat{\theta}_{\epsilon,z} - \hat{\theta}$, we can calculate influence as:

$$\mathcal{I}_{\text{up,params}}(z) \stackrel{\text{def}}{=} \frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon} = \frac{d\Delta_{\epsilon,z}}{d\epsilon}$$

Sketch of Derivation

$\hat{\theta}_{\epsilon,z}$ minimizes $R(\theta) + \epsilon L(z, \theta)$:

$$0 = \nabla R(\hat{\theta}_{\epsilon,z}) + \epsilon \nabla L(z, \hat{\theta}_{\epsilon,z})$$

Sketch of Derivation

$\hat{\theta}_{\epsilon,z}$ minimizes $R(\theta) + \epsilon L(z, \theta)$:

$$0 = \nabla R(\hat{\theta}_{\epsilon,z}) + \epsilon \nabla L(z, \hat{\theta}_{\epsilon,z})$$

Taylor expand the right hand side around $\hat{\theta}$

$$\begin{aligned} 0 \approx & \left[\nabla R(\hat{\theta}) + \epsilon \nabla L(z, \hat{\theta})\right] + \\ & \left[\nabla^2 R(\hat{\theta}) + \epsilon \nabla^2 L(z, \hat{\theta})\right] \Delta_\epsilon \end{aligned}$$

and solve for Δ_ϵ

$$\begin{aligned} \Delta_\epsilon \approx & - \left[\nabla^2 R(\hat{\theta}) + \epsilon \nabla^2 L(z, \hat{\theta})\right]^{-1} \\ & \left[\nabla R(\hat{\theta}) + \epsilon \nabla L(z, \hat{\theta})\right]. \end{aligned}$$

Sketch of Derivation

But $\nabla R(\hat{\theta}) = 0$. Keeping only $O(\epsilon)$ terms, we have

$$\Delta_\epsilon \approx -\nabla^2 R(\hat{\theta})^{-1} \nabla L(z, \hat{\theta})\epsilon.$$

We conclude that:

$$\begin{aligned} \frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\Big|_{\epsilon=0} &= -H_{\hat{\theta}}^{-1} \nabla L(z, \hat{\theta}) \\ &\stackrel{\text{def}}{=} \mathcal{I}_{\text{up,params}}(z). \end{aligned}$$

Removing and Perturbing Training Points

Similar methods can derive the following:

$$
\begin{aligned}
\mathcal{I}_{\text{up,loss}}(z, z_{test}) &\stackrel{\text{def}}{=} \left.\frac{dL(z_{test}, \hat{\theta}_{\epsilon,z})}{d\epsilon}\right|_{\epsilon=0} \\
&= -\nabla_\theta L(z_{\text{test}}, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta})
\end{aligned}
$$

which measures influence on the loss, not just the parameters.

Removing and Perturbing Training Points

Similar methods can derive the following:

$$\mathcal{I}_{\text{up,loss}}(z, z_{test}) \stackrel{\text{def}}{=} \frac{dL(z_{test}, \hat{\theta}_{\epsilon,z})}{d\epsilon}\Big|_{\epsilon=0}$$
$$= -\nabla_\theta L(z_{\text{test}}, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta})$$

which measures influence on the loss, not just the parameters.

We can also measure the influence of perturbing the **value** of a training input: $z_\delta = (x + \delta, y)$, which gives:

$$\frac{d\hat{\theta}_{\epsilon, z_\delta, -z}}{d\epsilon}\Big|_{\epsilon=0} = \mathcal{I}_{\text{up,params}}(z_\delta) - \mathcal{I}_{\text{up,params}}(z)$$
$$= -H_{\hat{\theta}}^{-1}\big(\nabla_\theta L(z_\delta, \hat{\theta}) - \nabla_\theta L(z, \hat{\theta})\big). \tag{1}$$

Analysis - Remove Terms from Influence

Let $p(y \mid x) = \sigma(y\theta^\top x)$, with $y \in \{-1, 1\}$ and $\sigma(t) = \frac{1}{1+\exp(-t)}$.

Analysis - Remove Terms from Influence

Let $p(y \mid x) = \sigma(y\theta^\top x)$, with $y \in \{-1, 1\}$ and $\sigma(t) = \frac{1}{1+\exp(-t)}$.

For a training point $z = (x, y)$,

$$
\begin{aligned}
L(z, \theta) &= \log(1 + \exp(-y\theta^\top x)) \\
\nabla_\theta L(z, \theta) &= -\sigma(-y\theta^\top x)yx \\
H_\theta &= \frac{1}{n}\sum_{i=1}^{n} \sigma(\theta^\top x_i)\sigma(-\theta^\top x_i)x_i x_i^\top
\end{aligned}
$$

Analysis - Remove Terms from Influence

Let $p(y \mid x) = \sigma(y\theta^\top x)$, with $y \in \{-1, 1\}$ and $\sigma(t) = \frac{1}{1+\exp(-t)}$.

For a training point $z = (x, y)$,

$$
\begin{aligned}
L(z, \theta) &= \log(1 + \exp(-y\theta^\top x)) \\
\nabla_\theta L(z, \theta) &= -\sigma(-y\theta^\top x)yx \\
H_\theta &= \frac{1}{n}\sum_{i=1}^{n} \sigma(\theta^\top x_i)\sigma(-\theta^\top x_i)x_i x_i^\top
\end{aligned}
$$

and $\mathcal{I}_{\text{up,loss}}(z, z_{\text{test}})$ is

$$
-y_{\text{test}}y \cdot \sigma(-y_{\text{test}}\theta^\top x_{\text{test}}) \cdot \sigma(-y\theta^\top x) \cdot x_{\text{test}}^\top H_{\hat{\theta}}^{-1} x.
$$

Analysis - Remove Terms from Influence

Let $p(y \mid x) = \sigma(y\theta^\top x)$, with $y \in \{-1, 1\}$ and $\sigma(t) = \frac{1}{1+\exp(-t)}$.

For a training point $z = (x, y)$,

$$L(z, \theta) = \log(1 + \exp(-y\theta^\top x))$$

$$\nabla_\theta L(z, \theta) = -\sigma(-y\theta^\top x)yx$$

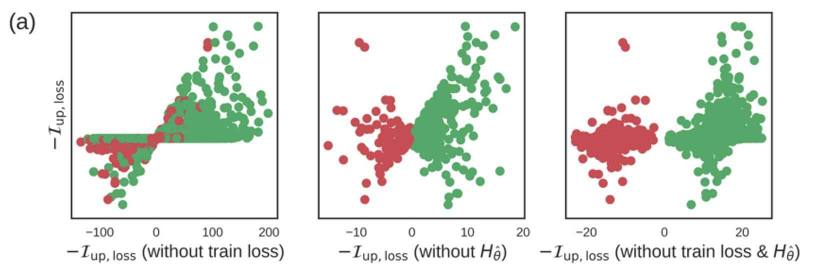
$$H_\theta = \frac{1}{n}\sum_{i=1}^{n} \sigma(\theta^\top x_i)\sigma(-\theta^\top x_i)x_i x_i^\top$$

and $\mathcal{I}_{\text{up,loss}}(z, z_{\text{test}})$ is

$$-y_{\text{test}}y \cdot \sigma(-y_{\text{test}}\theta^\top x_{\text{test}}) \cdot \sigma(-y\theta^\top x) \cdot \mathbf{x}_{\textbf{test}}^\top H_{\hat{\theta}}^{-1}\mathbf{x}.$$

Analysis - Remove Terms from Influence

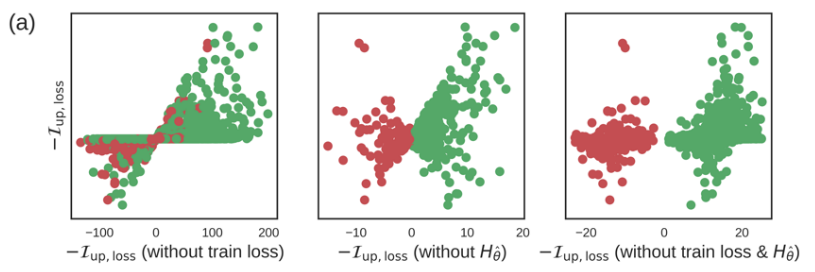
$$-y_{\text{test}} y \cdot \sigma(-y_{\text{test}} \theta^\top x_{\text{test}}) \cdot \sigma(-y \theta^\top x) \cdot x_{\text{test}}^\top H_{\hat{\theta}}^{-1} x.$$



left: $\sigma(-y\theta^\top x)$ gives points with high training loss more influence: without it, we overestimate the influence of training points

Analysis - Remove Terms from Influence

$$-y_{\text{test}} y \cdot \sigma(-y_{\text{test}} \theta^\top x_{\text{test}}) \cdot \sigma(-y \theta^\top x) \cdot x_{\text{test}}^\top H_{\hat{\theta}}^{-1} x.$$



middle/right: the weighted covariance matrix $H_{\hat{\theta}}^{-1}$ measures the “resistance” of the other training points to the removal of z. Without it, all same-label points are helpful, all opposite-label points are harmful.

Efficiency

Two challenges:

1. Forming and inverting $H_{\hat{\theta}} = \frac{1}{n} \sum_{i=1}^{n} \nabla_{\theta}^{2} L(z_i, \hat{\theta})$
    - n training points, $\theta \in \mathbb{R}^p$ requires $\mathcal{O}(np^2 + p^3)$ ops
2. Often want to calculate influence across all training points for a specific test point

How to Make Faster?

Overall approach:

- Efficiently approximate $s_{test} \stackrel{\text{def}}{=} H_{\hat{\theta}}^{-1} \nabla_{\theta} L(z_{test}, \hat{\theta})$

How to Make Faster?

Overall approach:

- Efficiently approximate $s_{test} \stackrel{\text{def}}{=} H_{\hat{\theta}}^{-1} \nabla_{\theta} L(z_{test}, \hat{\theta})$
- Use this to efficiently compute $\mathcal{I}_{\text{up,loss}}(z, z_{test})$ by just multiplying s_{test} by $\nabla_{\theta} L(z, \theta)$ as needed!

How to Make Faster?

Overall approach:

- Efficiently approximate $s_{test} \stackrel{\text{def}}{=} H_{\hat{\theta}}^{-1} \nabla_\theta L(z_{test}, \hat{\theta})$
- Use this to efficiently compute $\mathcal{I}_{\text{up,loss}}(z, z_{test})$ by just multiplying s_{test} by $\nabla_\theta L(z, \theta)$ as needed!

How to Make Faster?

Overall approach:

- Efficiently approximate $s_{test} \stackrel{\text{def}}{=} H_{\hat{\theta}}^{-1} \nabla_\theta L(z_{test}, \hat{\theta})$
- Use this to efficiently compute $\mathcal{I}_{\text{up,loss}}(z, z_{test})$ by just multiplying s_{test} by $\nabla_\theta L(z, \theta)$ as needed!

Conjugate Gradients

How to Make Faster?

Overall approach:

- Efficiently approximate $s_{test} \stackrel{\text{def}}{=} H_{\hat{\theta}}^{-1} \nabla_\theta L(z_{test}, \hat{\theta})$
- Use this to efficiently compute $\mathcal{I}_{\text{up,loss}}(z, z_{test})$ by just multiplying s_{test} by $\nabla_\theta L(z, \theta)$ as needed!

Conjugate Gradients

Stochastic Estimation

Validation: Influence matches leave-one-out retraining

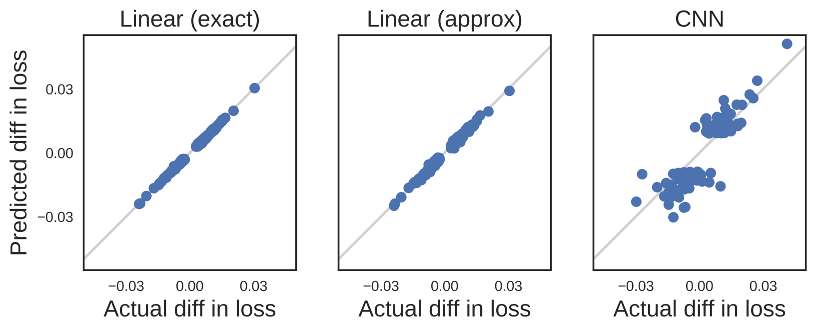
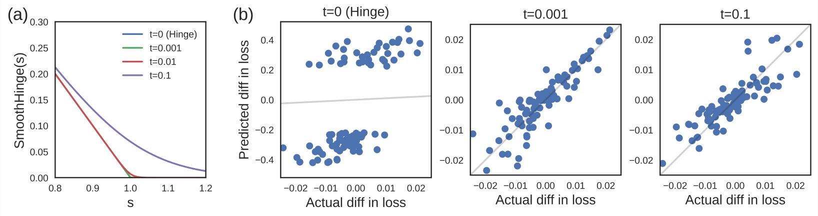


Figure 5. Left: For each of the 500 training points with largest influence, we plotted $-\frac{1}{n} \cdot \mathcal{I}_{\text{up,loss}}(z, z_{\text{test}})$ against the actual change in test loss after removing that point and retraining. The inverse HVP was solved exactly with CG. **Mid:** Same, but with the stochastic approximation. **Right:** The same plot for a CNN, computed on the 100 most influential points with CG. For the actual difference in loss, we removed each point and retrained from $\tilde{\theta}$ for 30k steps

Nonconvexity



- SVM with hinge loss
  - approximate with $smoothHinge(s, t) = t \log(1 + \exp(\frac{1-s}{t}))$
- set derivative at hinge to 0, lose second derivative information
- t=0.001, Pearson's R=0.95
- t=0.1, Pearson's R=0.91

Non-differentiable losses

Applications

Understanding Model Behavior

Adversarial Training Examples

Domain Mismatch

Fixing Mislabeled Examples

Thank you!

References

Bilò, Davide, Yann Disser, Matús Mihalák, Subhash Suri, Elias Vicari, and Peter Widmayer. 2012. "Reconstructing Visibility Graphs with Simple Robots." *Theoretical Computer Science* 444. Elsevier: 52–59.

Di Leonardo, R, L Angelani, D Dell'Arciprete, Giancarlo Ruocco, V Iebba, S Schippa, MP Conte, F Mecarini, F De Angelis, and E Di Fabrizio. 2010. "Bacterial Ratchet Motors." *Proceedings of the National Academy of Sciences* 107 (21). National Acad Sciences: 9541–5.

Disser, Yann. 2011. *Mapping Polygons*. Logos Verlag Berlin GmbH.

Erickson, L. H., and S. M. LaValle. 2013. "Toward the Design and Analysis of Blind, Bouncing Robots." In *IEEE International Conference on Robotics and Automation*.

Fletcher, Alexander G, Miriam Osterfield, Ruth E Baker, and Stanislav Y Shvartsman. 2014. "Vertex Models of Epithelial Morphogenesis." *Biophysical Journal* 106 (11). Elsevier: 2291–2304.

Li, He, and H. P. Zhang. 2013. "Asymmetric Gear Rectifies Random Robot Motion." *EPL (Europhysics Letters)* 102 (5).

O'Kane, J. M., and S. M. LaValle. 2007. "Localization with Limited Sensing." *IEEE Transactions on Robotics* 23 (4): 704–16.

Staveley, Brian E. n.d. "Molecular and Developmental Biology (Biol3530)." *Department of Biology, Memorial University of Newfoundland*. http://www.mun.ca/biology/desmid/brian/BIOL3530/DEVO_08/devo_08.html.

Tovar, Benjamin, Luis Guilamo, and Steven M LaValle. 2005. "Gap Navigation Trees: Minimal Representation for Visibility-Based Tasks." *Algorithmic Foundations of Robotics VI*. Springer, 425–40.