Optimal Coverage for an Arctic Sensor Network

Holly Dinkel

Stanford University

Abstract

Robotic sensing has the capability of amassing large amounts of data about environments unreachable to humans. Wireless sensor networks are of popular interest for sensing climate changes in light of the National Science Foundation's announcement of "Navigating the New Arctic" as one of its Ten Big Ideas for the next decade. This project implements two location optimization methods, the cross-entropy method and Voronoi cell decomposition, to maximize coverage of sensors placed in the Arctic over a region of observation.

Introduction

Observable environmental changes such as a rise in temperature, decreased salinity, and recession of Arctic sea ice are taking place in the Arctic region. These changes have regional and global significance: there is growing concern about the implications of ice-free seasons in the Arctic in coming decades as interest in human access to natural resources and new trade routes increases [6]. However, evidence of these changes is impaired by partial geographic sensor coverage, making descriptions of the state of the Arctic myopic and an understanding of the impact of these changes on the rest of the Earth incomplete. Many of the knowledge voids exist because of technological limitations created by extreme cold and remoteness, with some areas decommissioning observatories due to cost of upkeep [2]. At the same time, many organizations acknowledge the value of autonomous collection of data by sensors placed in the environment.

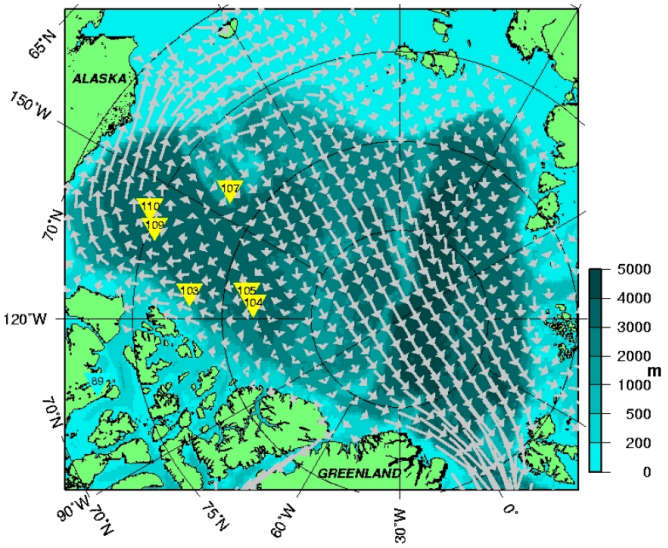


Figure 1: Active Ice Tethered Profiler Locations (Yellow Triangles) as of June 6, 2019 [4]

The International Polar Year program supported the development of Ice-Tethered Profilers (ITPs), buoys designed to sample the temperature and salinity of the Arctic Ocean for an operating life of three years [4]. While a

multi-agent sensor network offers improved confidence in data and robustness to agent failure, the ITP network deployed suffers from poor sensor coverage. Figure 1 shows the six active ITPs occupying just over 30^o of longitude of the Arctic Circle.

A central design problem in multi-robot control is determining how to deploy a network of sensors to cover an area of interest while minimizing sensing cost. This study aims to explore the optimization of the deployment of a network of sensors in the Arctic region.

Methodology

In this project, optimization of a coverage cost function via the population-based cross-entropy method is compared against the Voronoi cell location optimization obtaining the true minimum. The cross-entropy method is implemented in this project because gradient-based methods have been implemented widely in optimization of multi-agent deployment and control in the literature [1], [5], whereas population methods have been studied comparatively less. Population methods are also convenient because they do not require knowledge of gradient information, which becomes particularly useful in coverage optimization when importance region weighting is discontinuous.

Population-Based Optimization

The optimization of deployment of a network of sensors is framed as minimzation of a cost function, $H(\mathbf{x})$, subject to the constraint that the solution must lie within a sensing region, $\mathbf{Q}$ [3]. This optimization framework is represented in Equation (1).

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & H(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x} \in \mathbf{Q} \end{aligned} \tag{1}$$

For n sensors in an environment with m sensing targets, sensor i has state $\mathbf{x}_i \in \mathbf{X} \subset \mathbb{R}^2$ where $\mathbf{X}$ is the design space. The combined array of the states of each sensor in the network is $\mathbf{X} = [\mathbf{x}_1^T, ..., \mathbf{x}_n^T]$. The sensors are deployed to cover a bounded region of the Arctic, $\mathbf{Q}$. When all desired sensing locations carry equal weights of importance, the cost of sensing all points $\mathbf{q}_j \in \mathbf{Q}$ is represented by f_1 in Equation (2).

$$f_1(\mathbf{x}, \mathbf{q}) = \sum_{i=1}^{n} \sum_{j=1}^{m} || \mathbf{x}_i - \mathbf{q}_j || \tag{2}$$

The coverage cost depends both on the distances of each sensor from its desired sensing location (Equation (2)) and on the distance of each sensor from every other sensor in the network, shown in Equation (3).

$$f_2(\mathbf{x}) = \sum_{i=1}^{n} \sum_{k=1}^{n} || \mathbf{x}_i - \mathbf{x}_k || \tag{3}$$

Optimal sensor coverage minimizes the distance between $\mathbf{x}_i$ and $\mathbf{q}$ while maximizing the amount of space between each sensor. Thus, the coverage cost function from Equation (1) is represented as Equation (4)

$$H(\mathbf{x}) = f_1(\mathbf{x}, \mathbf{q}) - f_2(\mathbf{x}) \tag{4}$$

The constraint requiring the optimal design to lie within $\mathbf{Q}$ is added to the coverage cost function $H(\mathbf{x})$ by a counting penalty. The coverage cost function is optimized using the cross-entropy method. The cross-entropy method maintains a probability distribution over the design space. Upon every iteration k of the cross-entropy method, a proposal distribution is used to generate new samples for the next iteration. The new samples are distributed as $\mathbf{x}^{(k+1)} \sim \mathcal{N}(\boldsymbol{\mu}^{(k+1)}, \boldsymbol{\Sigma}^{(k+1)})$. The set of new samples is pruned to the top ten samples that yield the lowest objective function value through specifying a number of elite samples. The parameters of the normal

distribution used for the subsequent iteration are updated by fitting a normal distribution to the elite samples, as shown in Equations (5, 6) [3].

$$\boldsymbol{\mu}^{(k+1)} = \frac{1}{m_{elite}} \sum_{i=1}^{m_{elite}} \mathbf{x}^{(i)} \tag{5}$$

$$\boldsymbol{\Sigma}^{(k+1)} = \frac{1}{m_{elite}} \sum_{i=1}^{m_{elite}} (\mathbf{x}^{(i)} - \boldsymbol{\mu}^{(k+1)})(\mathbf{x}^{(i)} - \boldsymbol{\mu}^{(k+1)})^{\top} \tag{6}$$

Voronoi Cell Decomposition

One existing solution to coverage optimization is Voronoi cell decomposition. Where $\phi(\mathbf{q})$ is a distribution density function weighting the importance of $\mathbf{q}_j \in \mathbf{Q}$, a cost function for Voronoi coverage is presented in [1] as Equation (7)

$$H(\mathbf{x}, \mathbf{q}) = \sum_{i=1}^{n} \int_{\mathbf{Q}} f(|| \mathbf{x}_i - \mathbf{q} ||)\phi(\mathbf{q})d\mathbf{q} \tag{7}$$

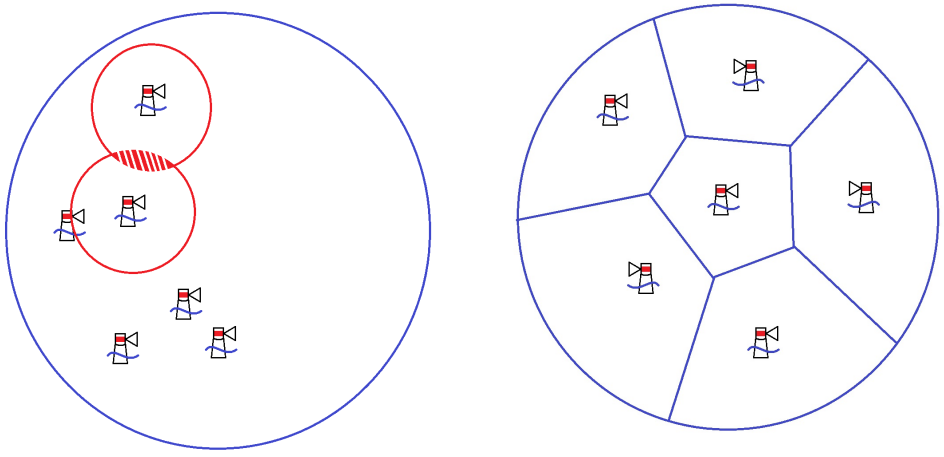
This cost function ins minimized with respect to both sensor location and assignment over the region $\mathbf{Q}$. At fixed sensor locations, the optimal partition of $\mathbf{Q}$ is the Voronoi cell, represented compactly as Equation (8).

$$V_i = \{q \in Q| \; || \; q - x_i \; || \leq || \; q - x_j \; ||, \forall i \neq j\} \tag{8}$$

The not-necessarily-unique local minimum points for the location optimization function $H(\mathbf{p})$ are the centroids, C_i of their Voronoi cells. This can be represented as Equation (9)

$$C_i = \underset{x_i}{\text{argmin}} \quad H(\mathbf{x}) \tag{9}$$

Centroidal Voronoi coverage optimization is depicted in Figure 2.



(a) Sub-optimal observation of the Arctic Circle with ITPs.

(b) Optimal multi-agent observation of the Arctic Circle with ITPs using Voronoi cell decomposition.

Figure 2: Sensor Coverage of the Arctic Circle With and Without Optimization.

Results

In the initialization of the arctic region simulation, a user-specified number of sensors and sensing targets is input. Random initial locations for each sensor, $\mathbf{x}_i$ and random locations for each sensing target, $\mathbf{q}$ are generated within the bounded region $\mathbf{Q} = [0:20, 0:20]^T$.

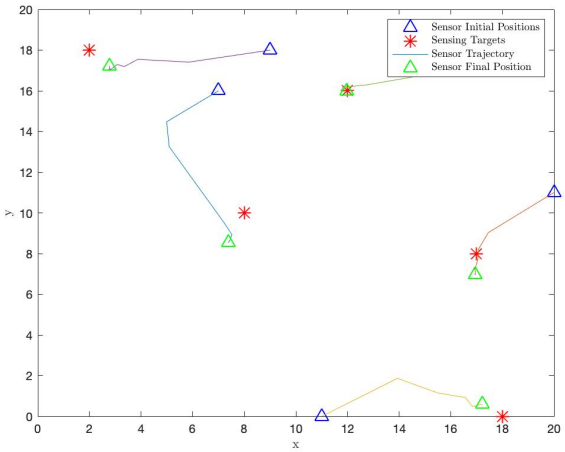


Figure 3: Map with five targets, five initial sensor locations, five final sensor locations, and the path each sensor took to its final location over the course of five iterations of cross-entropy coverage cost optimization.

In optimization of the coverage cost function, the total number of samples and the total number of elite samples used for each iteration of the cross-entropy method was 150 and 10, respectively. Figure 3 plots these initial conditions along with the optimized final sensor locations and the path the sensors followed to reach their final locations for a case with five sensors and sensing targets. The plot shows that the sensors did not take straight-line paths towards their sensing target. This phenomenon can be interpreted as the result of two factors. The coverage cost optimization is carried out as optimizing each sensor location individually rather than simultaneously. After the first sensor location is optimized, the algorithm then optimizes the second sensor location, the third, and so on. After a sensor position is optimized, the sensor is permanently placed in the environment and its position cannot be updated. The i^{th} sensor's optimal position is used for the optimization of the $i + 1^{th}$ sensor.

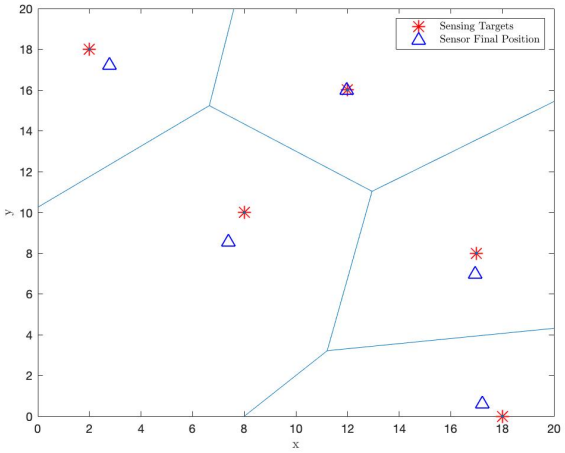


Figure 4: Map with five targets, five cross-entropy optimized sensor locations, and Voronoi cell decomposition juxtaposed.

The optimization architecture was designed this way rather than optimizing all of the sensor positions simultaneously because it allows freedom for the positions of targets that have not yet been sensed to change without the requirement of re-optimizing the position of every sensor in the network in a more accurate simulation of the sensor deployment process.

In Voronoi cell decomposition, the coverage cost is minimized when final sensor positions are at the sensing target position. The final positions of cross-entropy method coverage optimization are compared to the results of Voronoi cell decomposition in Figure 4. The final positions all fall very closely to the sensing targets, with the final position for all of the sensors less than 1.5 units away from the sensing targets. Perhaps the better metric for performance of the cross-entropy method is that every sensor found a final position in a unique Voronoi cell, demonstrating that the final sensor positions do not overlap and good coverage has been achieved.

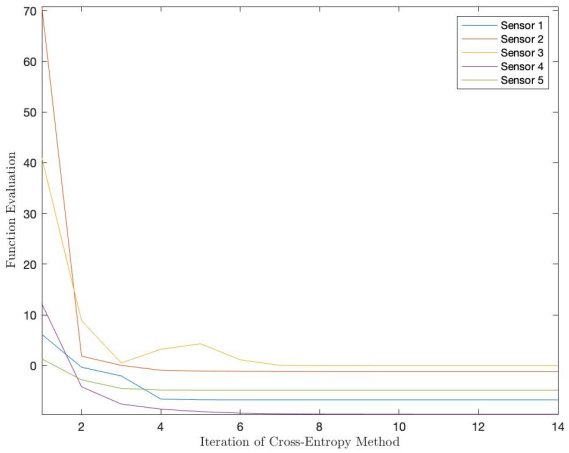


Figure 5: Coverage cost minimization with the cross-entropy method yields decreasing function evaluations for each new iteration.

The initial objective function evaluations for each sensor are highly dependent on the proximity of each sensor to targets and other sensors in the environment, and depend less on sensor order. Figure 5 shows that each iteration of the cross-entropy method results in a lower evaluation of the objective function for each sensor; after approximately six iterations of the cross-entropy method, the sensors have found their optimal position.

Conclusions

This project presented a use-case for cross-entropy optimization in the deployment of a network of sensors to maximize coverage over sensing targets in the Arctic. Optimal positions obtained from cross-entropy optimization compared with local minima obtained from Voronoi cell decomposition showed that the cross-entropy method worked well to minimize coverage cost.

This project leaves important extensions open for further study. Literature surrounding coverage optimization is saturated with the implementation of gradient-based optimization methods. Literature would benefit from the comparison of a numerical gradient-based method with the population-based method and Voronoi cell decomposition implemented in this project to quantify the robustness of the gradient-based approach when the objective function is highly nonlinear or discontinuous. This project also simplified the Arctic landscape, and the simulation of the environment is low-fidelity. The simulation could be further improved by using the MATLAB Arctic Mapping Toolbox to plot the current sensor positions, to incorporate more feasible starting points for sensors (such as at Arctic ports), and to incorporate constraints governing which sensing targets could be reached by helicopters or icebreaker ships.

References

[1] J. Cortés, S. Martinez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20:243–255, 2004.

[2] National Research Council. *Toward an Integrated Arctic Observing Network*. The National Academies Press, Washington, DC, 2006. ISBN 978-0-309-10052-6. doi: 10.17226/11607.

[3] M. Kochenderfer and T. Wheeler. *Algorithms for Optimization*. MIT Press, Cambridge, 2019. ISBN 978-0-262-03942-0.

[4] R. Krishfield, K. Doherty, D. Frye, T. Hammar, J. Kemp, D. Peters, A. Proshutinsky, J. Toole, and K. von der Heydt. Design and operation of automated ice-tethered profilers for real-time seawater observations in the polar oceans. Technical Report WHOI-2006-11, Woods Hole Oceanographic Institute, 01 2006.

[5] M. Schwager, D. Rus, and J. Slotine. Unifying geometric, probabilistic, and potential field approaches to multi-robot deployment. *The International Journal of Robotics Research*, 0:1–12, 08 2010. doi: 10.1177/0278364910383444.

[6] Congressional Research Service. Changes in the arctic: Background and issues for congress. Technical Report R41153, Members and Committees of Congress, 03 2019.

Appendix: MATLAB Code

Objective Function

```
function [f] = objective(x,Target,Sensor)
f1 = zeros(1,size(Target,2)); f2 = zeros(1,size(Sensor,2)); pcount =
   zeros(1,4);
p1 = 1; %Weighting coefficient for distances between sensors and
   target locations
p2 = 1; %Weighting coefficient for distances between sensors and all
   other sensors
    for i = 1:size(Target,2)
        f1(i) = norm(x-Target(:,i));
    end
f1 = sum(f1);
    for i = 1:size(Sensor,2)
        f2(i) = norm(x-Sensor(:,i));
    end
f2 = sum(f2);
%Include penalty for sensors leaving the map defined by [0 20 0 20]
constraint = [x(1)-20,x(2)-20,-x(1),-x(2)];
    for j = 1:length(constraint)
        pcount(j) = max(constraint(j),0);
    end
penalty=sum(pcount)^3;
if penalty > 0
    p2 = 0;
end
f = (p1*f1-p2*f2)+penalty;
end
```

Cross-Entropy Function

```
function [mu,mu_feval] = crossentropy_method(mu,sigma,Target,Sensor)
%f = function to optimize
%P = (initial and final) sampling distribution
%k_max = number of iterations
%m = number of samples in the distribution per iteration
%m_elite = samples used to define the proposal distribution
m = 150;
k_max = 15;
m_elite = 10;
eval = zeros(1,m);
mu_feval = zeros(1,k_max);
for k = 2:k_max
    samples = mvnrnd(mu(:,k-1),sigma,m)';
        for i = 1:m
            eval(i) = objective(samples(:,i),Target,Sensor);
        end
```

```
    [order, I] = sort(eval);
    I = I(1:m_elite);
    A = samples(1,I); B = samples(2,I);
    mu_feval(k) = mean(eval(1:I));
    mu(:,k)=[mean(A) mean(B)];
    sigma = cov(A,B);
end
end
```

Optimization Function

```
function [x,Sensor, feval] = optimize(x,Target,Sensor,i)
% Adjusts x/mu, the position of the ith sensor, to minimize coverage
% cost using the cross entropy method. Updates Sensor, the matrix of
   all
% sensor locations, with the optimized position of x.
d = length(x);
A = rand(d);
A = 0.5*(A+A'); %random positive definite matrix
sigma = A + d*eye(d);
mu = x;
[mu mu_feval] = crossentropy_method(mu, sigma, Target,Sensor);
x = mu; feval = mu_feval;
Sensor(:,i) = mu(:,end);
end
```

Generating the Voronoi Centroids and Partitions

```
voronoi(Target(1,:),Target(2,:))
```

Calling the Optimization Function

```
%% Optimization
f = zeros(1,size(M,2));
for i = 1:n
    [mu Sensor mu_feval] = optimize(M(:,i),Target,M,i);
    x(i) = {mu};
    M = Sensor;
    f(i) = objective(M(:,i),Target,M);
    feval(i,:) = mu_feval;
end
funEval = sum(f);
```