# MIT 16.412 - Cognitive Robotics

# Exploiting Submodularity in Information Gathering: *An Application to Spacecraft Berthing*

Garrett, M., Santana, P., and Schaffner, M.

May 15th 2013

## Abstract

The recent retirement of the Space Shuttle program has prompted NASA to begin outsourcing development of future technologies for transporting supplies and humans to the International Space Station (ISS). Any spacecraft approaching the ISS for docking or berthing is required to estimate its own position relative to the Space Station. We examine a problem of placing sensors on the ISS to aid in position estimation of spacecraft during the berthing process. We assume that NASA has a budget of 5 sensors available to be placed, which leads to an optimization problem of maximizing the sensor coverage of the planar area of the spacecraft operation. With some modest assumptions, this problem can be treated as an information gathering problem and is therefore submodular. By exploiting the submodularity of this problem, greedy algorithms can be used to find approximate solutions with guarantees on lower bounds of performance relative to the optimal solution. We implement a lazy greedy algorithm to solve the coverage maximization problem and discuss the resulting solution quality. We then model the spacecraft position estimation as a Hidden Markov Model, and implement an online greedy algorithm to sequentially probe the sensors based on expected reduction in entropy to update the HMM belief. The results of this algorithm are compared with a random sampling strategy and some implications are discussed. Finally, we describe several lessons that we learned throughout the course of this project.

## 1 Motivation

The recent retirement of the Space Shuttle program has prompted NASA to begin outsourcing development of future technologies for transporting supplies and humans into space, specifically to the International Space Station (ISS). As a result, private companies such as SpaceX and Orbital Sciences have begun developments under contracts from NASA's Commercial Orbital Transportation Services (COTS) program. Notably, on May 25, 2012, SpaceX's Dragon spacecraft became the only private space vehicle in history to berth with the ISS.

For this operation, involving maneuvers in close proximity to the ISS (within its 200-meter Keep Out Sphere), the Dragon spacecraft was required to use multiple sensors to estimate its own position relative to that of the ISS. This requirement is standard for any spacecraft involved in such operations with the ISS. To comply with this requirement, the Dragon spacecraft used a DragonEye sensing package, shown in Figure 1. The DragonEye consists of both LIDAR and thermal imaging components, which provide data that are processed by machine vision algorithms to perform the position estimation for the spacecraft.

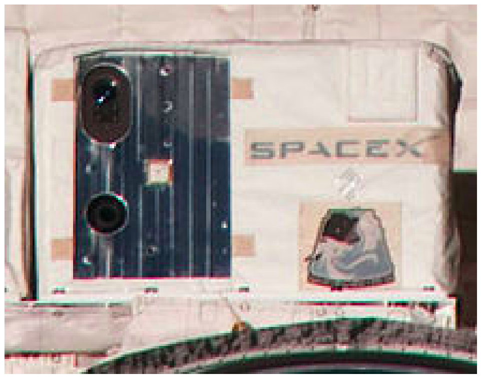


Figure 1: The DragonEye package, consisting of LIDAR and thermal sensors.

Additional sensors with potential use for spacecraft approaches and automated docking are rangefinders (LIDAR being a specific case) and fiducial markers near the docking port of a spacecraft. Rangefinders can provide a high degree of accuracy in the direction of heading. The use of fiducial markers is also well established for the problem of position estimation, and is particularly applicable to vision-based spacecraft docking [1]. Even a small number of such feature points can be used to calculate the estimated attitude and position translation parameters in the final translational phase of docking [2]. As space travel becomes more distributed among spacecraft manufactured by various private companies, and as humans become more frequent cargo, it will become increasingly important that manufacturers have standard physical and sensor interfaces for docking and berthing with other spacecraft in situ.

## 2 Problem Statement

For Dragon's operations in close proximity to the ISS, NASA has strict requirements on the bounds in which the spacecraft must stay in order to continue its berthing maneuver. These bounds minimize the risk of damage to the ISS by constraining the acceptable spacecraft position to a small region in front of the docking port. A two-dimensional representation of this region is shown below in Figure 2. The "cost" to NASA of a spacecraft approaching the ISS is lowest at the center of this region, and increases as the capsule position increases its distance from the center of this region. As the Dragon spacecraft updates estimates of its own position, its autonomous control loop will move the spacecraft toward the region of lowest cost (i.e., the center of the region). Doing so will allow the ISS's Canadarm robotic arm to successfully manipulate the Flight-Releasable Grapple Fixture on the Dragon in preparation for berthing. Failing to do so will result in severe delays and in the worst case, a mission abort.

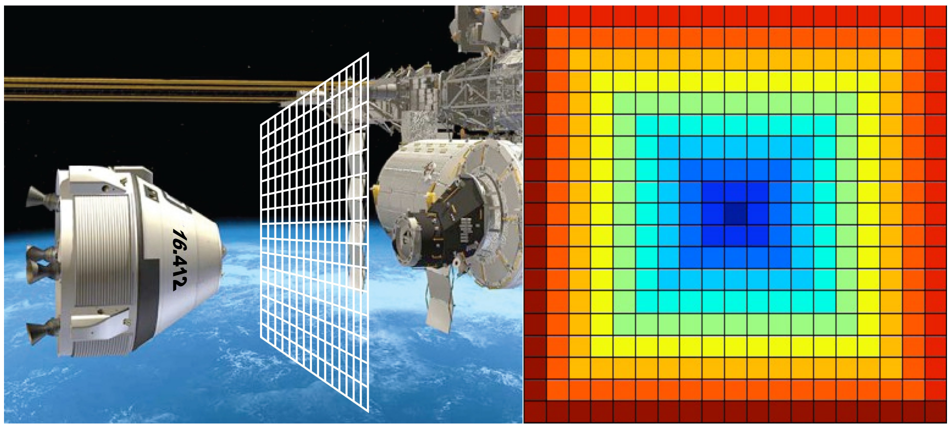


Figure 2: Left, a rendering of a Dragon-like capsule approaching the ISS, with area of operation as a white grid. Right, the two-dimensional area of operation, colored by the amount of risk "cost" to NASA. Blue is low, red is high.

To aid the approaching spacecraft in its position estimation, several sensors are provided by the ISS, on which the Dragon will be assumed to rely in addition to its own DragonEye. These sensors include radar range-finders, which are very accurate for distance measurements to the approaching spacecraft, IR beacons, which are very accurate for lateral measurements, and several fiducial markers, which provide adequate accuracy for the spacecraft's visual guidance system in both directions. These sensors are to be placed at various locations around the docking port of the ISS to maximize the sensors' coverage area of the plane on which the capsule's position is to be estimated.

A notional top-view diagram of possible sensor locations and accuracies is shown below in Figure 3. We assume that NASA prefers to limit the number of sensors to be placed on the ISS due to the monetary and opportunity costs of astronaut spacewalks, as well as the energy cost of operating the sensors. In addition, we assume that SpaceX desires to limit its number of requests for sensor information from the ISS (the Dragon can request data from these sensors at any time; however, each sample costs a certain amount of energy and bandwidth in the spacecraft's communication with the ISS).

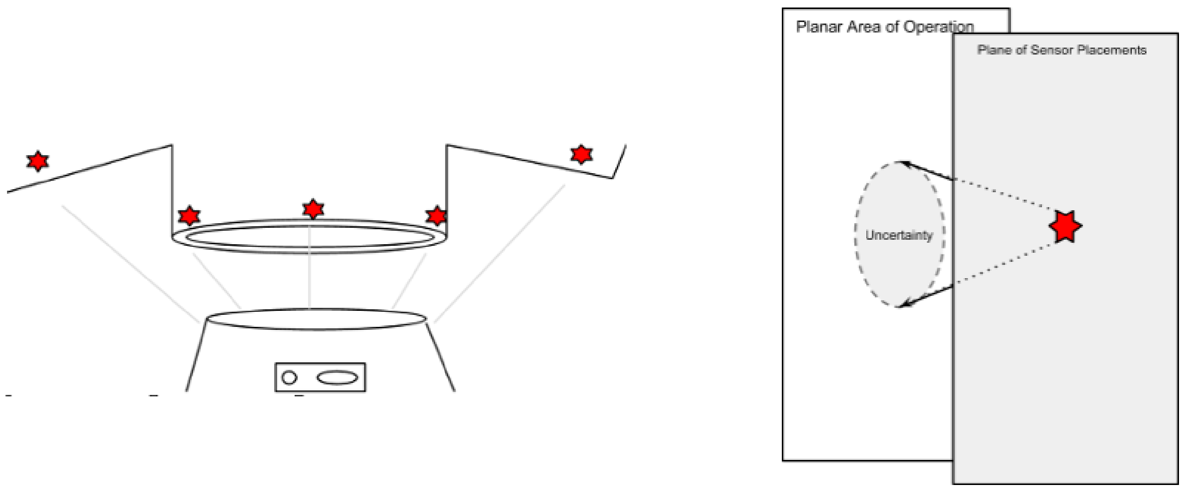


Figure 3: Left, a notional top-view diagram of the sensor configuration on the docking port of the ISS. (Red stars represent sensors. Right, an offset front view diagram of a range-finding sensor with its associated accuracy in detecting the spacecraft position in the center of the planar operating region.)

The limits on the number of sensors to be placed outside the ISS leads to an optimization problem of maximizing the sensor coverage of the planar area described above. Each sensor's observation is assumed to be conditionally independent of other sensors' observations of the area. The problem of position estimation from the multiple sensors' coverage of the area can then be considered an information gathering problem, the general form of which is summed up by Lambrecht as follows: "...given a world model, a set of information sources, a mapping of information sources to the world model, and a query on the world model, return information contained in the information sources that answer the query on the model" [3]. Each one of these parts of a general information gathering problem maps directly to the problem of sensor coverage for the ISS berthing maneuver. The set of information sources includes the sensors that NASA will place around the space station's docking port. The mapping of those information sources to the world model is represented by each sensor's coverage area, in relation to the planar area of spacecraft operation. The query on the world model is the query of spacecraft position on the planar area.

The maximization of sensor coverage, given a strict budget, forms the first problem that the current work addresses. The approach to solve this problem exploits the submodularity of certain information gathering problems to greedily generate a near-optimal offline solution for sensor placement. A second problem that arises, however, is that of online selection of sensors to query during the berthing maneuver. As previously stated, SpaceX and NASA both have budgets of energy and bandwidth usage during normal operations; as a result, they desire to actively sense as little as possible while still maintaining sufficient confidence in the location of the spacecraft at any given time. Two approaches will be compared to solve this problem of conserving measurements: one approach applies an online greedy algorithm adapted from [4] to minimize the entropy in the system after the selection of each sensor to probe. The other approach randomly chooses sensors to probe until it has reached the online operating budget. The results from these two approaches will be compared. Finally, general analysis of the results will be given, followed by a description of lessons learned.

# 3 Technical approach

This section provides a formal discussion of the technical aspects of the two kinds of information gathering tasks addressed in this final project. In Section 3.1, we start by modeling our spacecraft docking problem as an instance of belief state estimation for Hidden Markov Models (HMM) under noisy sensor measurements. Next, Section 3.2 shows how the submodularity of sensor coverage can be exploited to find a good placement of sensors on the ISS by means of a greedy strategy. Finally, given these static sensor placements and observation models for each one of the sensors on the ISS, Section 3.3 presents an algorithm based on the results of [4] that chooses which sensors to sample, based on a minimum conditional entropy criterion. A numerical comparison between this method and a random sensor sampling strategy is shown later in Section 4.

## 3.1 Modeling

We restrict the position of a reference point on the spacecraft (center of mass, for example) to a discrete ($X$,$Y$) grid. The same is done for the possible sensor locations on the ISS. Concerning the reasons behind this modeling assumption, it is worthwhile to first mention that submodularity is a concept defined for functions over sets of discrete elements [5]. Therefore, if we want to exploit submodularity in our sensor design process, it makes sense to restrict the sensor locations to a number of specific mounting points. Second, for an application of docking in close proximity to the ISS, the approximation error introduced by a reasonably-sized grid over the docking area might be comparable to the intrinsic uncertainty about the system's hidden state (in this case, the spacecraft's position). Hence, a discretized position approximation should not degrade our performance significantly, if at all. Finally, considering a discrete grid of positions facilitates the belief propagation step in our simulation with arbitrary measurement likelihood models, since we replace a series of numerical integrations by simple sums of values over a grid. It also facilitates the computation of probabilistic entropies for the exact same reasons. Thus, given the reasons behind our discretization, we proceed with the modeling of our system as an instance of an HMM.

We start by defining our set of discrete states $\mathbb{S} = \{s_1, s_2, ...s_N\}$, where $s_i = (x_i, y_i)$ is a particular position on the grid. We also define the observation set to be $\mathbb{O} = \mathbb{S}$, since we assume that the sensors output position information directly.

In the applications shown in this project, we assume that the spacecraft remains virtually static as a sequence of sensor probings are performed. Therefore, the transition function $T : \mathbb{S} \times \mathbb{S} \to \mathbb{R}$, where $T(s_i, s_j)$ denotes the probability of performing a transition $s_i \to s_j$, becomes

$$T(s_i, s_j) = \begin{cases} 1, & \text{if } s_i = s_j, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

If seen as a matrix, we see that $T = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix of appropriate dimensions.

We finish by defining a family of observation models $O : \mathbb{S} \times \mathbb{O} \times \mathbb{M} \to \mathbb{R}$, where $\mathbb{M} \subset \mathbb{S}$ is a set of mounting positions for the sensors projected over the grid $\mathbb{S}$. The function $O(s_i, z_i, m_i)$ returns the probability of position $z_i$ being reported by the sensor mounted at $m_i$ given that the spacecraft is at $s_i$. In other words, we have $O(s_i, z_i, m_i) = \Pr(z_i | s_i, m_i)$. The reason why we need this additional argument $m_i$ is because we wanted to model sensor degradation as the spacecraft moves away from the "preferred detection area". For example, sensing systems based on off-the-shelf or wide-angle cameras usually suffer from severe performance degradation as the target approaches the boundary of the image, where distortion is a significant factor even after calibration[1]. Therefore, we model sensor outputs by

$$z_i = s_i + \varepsilon(m_i), \varepsilon(m_i) \sim N(0, R(m_i)), \tag{2}$$

where $R(m_i){=}R_{center} \times \alpha^{\|s_i - m_i\|_{grid}}$, $\alpha \geq 1$, is a function that penalizes the covariance of measurements from the sensor at $m_i$ based on the distance $\|.\|_{grid}$ of $s_i$ to the projected mounting point $m_i$ on the grid $\mathbb{S}$. With these definitions, our HMM model is given by the tuple $\langle \mathbb{S}, \mathbb{O}, T, O \rangle$.

[1]Based on the authors' experience with visual tracking systems for robotic applications.

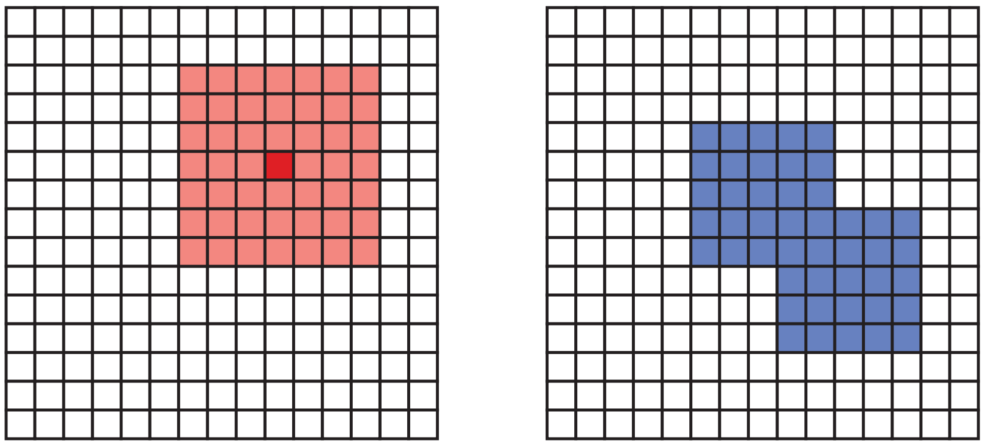

(a) Single sensor coverage (b) Coverage of two overlapping sensors.

Figure 4: Sensor coverage examples. In Figure 4(a), the darker cell represents the projection of the mounting point ($m_i$) of a sensor with coverage value $w = 3$. Figure 4(b) shows the coverage of two overlapping sensors.

## 3.2 Optimizing sensor coverage

The first part of our final project exploited the submodularity of the MAX-COVER problem [5–7] in order to efficiently choose a set of sensor mounting points that maximizes the coverage of the docking grid. For that, we defined the function $COVER(m,w)$ that takes two arguments: $m \subseteq \mathbb{M}$ is a set of sensor mounting locations and $w \in \mathbb{R}^{|m|}$ is a vector of real numbers, where $w_i$ denotes the coverage of a sensor mounted at $m_i$, $i \in \{1, 2, \ldots, |m|\}$. Figure 4(a) shows an example of a sensor with coverage $w = 3$, where $w$ is the number of squares from the center that the sensor covers in any direction.

Algorithm 1 shows a greedy approach for solving the MAX-COVER problem whose performance is guaranteed never to be below a factor of $1 - 1/e$ from the optimal solution [5]. In order to find an assignment of sensor mounting points that maximizes sensor coverage, or at least approximates the optimal solution well, we used Algorithm 2, a lazy version of Algorithm 1 originally proposed in [8] and pedagogically shown in our advanced lecture. Section 4 shows the numerical results for a budget of 5 sensors obtained using this lazy greedy strategy, whose implementation can be found in the attached Matlab code.

**Algorithm 1** Greedy algorithm for maximizing sensor coverage (adapted from [5])

```
Input: Coverage function $C(m, W)$, coverage map $W$, mounting points $\mathbb{M}$, sensor budget $d$.
Output: Selected sensor locations $A$.
1: $A \leftarrow \emptyset$
2: for $k = 1$ to $d$ do
3:     $a^* = \underset{a \in M \setminus A}{\text{argmax}}\ C(A \cup \{a\}, W) - C(A, W)$
4:     $A \leftarrow A \cup \{a^*\}$
5: return $A$
```

## 3.3 Adaptive probing under communication constraints

The results of Section 3.2 tell us what would be a good placement of sensors, on average, for spacecraft docking missions on the ISS. For the second part of our final project, we wanted to investigate how we could leverage submodularity once again to come up with a good probing sequence for these sensors; the goal is to reduce uncertainty about our hidden state $S$ as much as possible if we have a limited budget on the number of sensors that we could measure. It has been shown that information entropy $H(S|Z)$, where $S$ is the hidden state and $Z$ is a set of observations, is a submodular function of $Z$ [9]. Therefore, if we applied the greedy algorithm from [5] or its lazy version in [8] to choose a set of probing points offline, we would be able to guarantee a performance of at least $(1 - 1/e)$ with respect to the optimal final conditional entropy [7]. However, it is somewhat unrealistic to prevent a spacecraft from incorporating real-time sensing information into its decision making. Therefore, instead of using the offline greedy algorithm from [5], we implemented an adapted version of the greedy online probing algorithm by de Kleer and Williams [4] shown in Algorithm 3.

Algorithm 3 is short and simple, but Step 3 places heavy demands on the processor even for small-sized problems due to the large amount of averaging required. In the minimization for Step 3, we need to repeatedly compute

$$H(S|Z) = \sum_{z} \Pr(Z = z) H(S|Z = z), \tag{3}$$

**Algorithm 2** Lazy greedy algorithm for maximizing sensor coverage (adapted from [8])

**Input:** Coverage function $C(m, W)$, coverage map $W$, mounting points $\mathbb{M}$, sensor budget $d$.

**Output:** Selected sensor locations $A$.

```
1: if $d \geq |\mathbb{M}|$ then
2:     return $\mathbb{M}$
3: $q \leftarrow$ LIST($\emptyset$)
4: for $a \in \mathbb{M}$ do PUSH($q$,$\langle a, C(\{a\}, W)\rangle$)
5: SORT($q$); $A \leftarrow$ SENSOR(POP($q$)); $N \leftarrow 1$
6: while $N \leq d$ do
7:     $a \leftarrow$ SENSOR(POP($q$))
8:     $v = C(A \cup \{a\}, W) - C(A, W)$
9:     if $v \geq$ VALUE(TOP($q$)) then
10:        $A \leftarrow A \cup \{a\}$
11:        $N \leftarrow N + 1$
12:    else
13:        PUSH($q$,$\langle a, v\rangle$)
14:        SORT($q$)
15: return $A$
```

where

$$H(S|Z=z) = -\sum_s \Pr(S=s|Z=z)\log_2 \Pr(S=s|Z=z),$$

$$\Pr(Z=z) = \sum_s \Pr(Z=z, S=s) = \sum_s \Pr(Z=z|S=s)\Pr(S=s). \tag{4}$$

It is easy to see from (3) and (4) that, if $|S|$ or $|Z|$ are large, the number of operations required to compute Step 3 exactly, starts to become impractical. For example, for the $20 \times 20$ grid used in Section 4, we would need approximately $400^2$ operations for each possible choice $Z \in A$.

The last step in the FOR loop, Step 5, updates the belief state $b$ with a new measurement $z$ using the well-known Baye's rule for computing *a posteriori* distributions.

**Algorithm 3** Online greedy sensor probing (adapted from [4])

```
Input: HMM $H=\langle \mathbb{S}, \mathbb{O}, T, O \rangle$, belief state $b_{\text{prior}}$, available sensors $A$, sensing budget $d$
Output: Updated belief $b_{\text{up}}$
1: $b \leftarrow b_{\text{prior}}$
2: for $k = 1$ to $d$ do
3:     $Z^* = \operatorname*{argmin}_{Z \in A} H(b|Z)$
4:     $z \leftarrow \mathit{Sample}(Z^*)$
5:     $b \leftarrow \mathit{HMM\text{-}Update\text{-}Belief}(H, b, z)$
6: return $b$
```

For each discrete state $s \in \mathbb{S}$, we have

$$\Pr(s|Z = z) \propto \Pr(Z = z|s) \Pr(s). \tag{5}$$

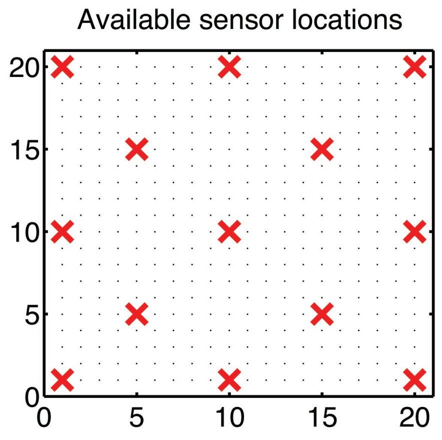
The first term on the right-hand side of (5) is given by the observation model of our HMM, while the second term is our prior belief about $\Pr(S = s)$ before incorporating the new measurement. Once all terms in (5) are computed, we then proceed to the normalization of their sum.

## 4 Numerical results

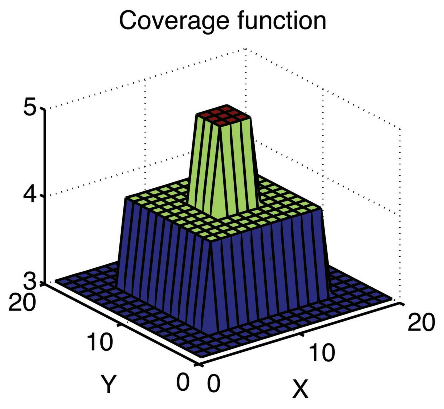
This section is a discussion of our results from both the offline sensor design and online adaptive probing sections of this final project. Our first main goal was to apply a greedy strategy to the offline selection of sensor placements outside the ISS, to enable confidence in the Dragon spacecraft's estimates of its own position. The second main goal was to compare the performance of two online algorithms for conserving energy in sampling a subset of the sensors installed. We begin with a brief review of the problem setup.

### 4.1 Problem Review

As discussed in Section 3.1, we discretized the region of operation (and the corresponding area on the docking port surface) into 400 cells arranged in a $20 \times 20$ grid. The potential sensor locations approved by NASA for placement on the ISS are shown in Figure 5(a). The coverage function used to evaluate the benefit of a sensor in a given cell is shown in Figure 5(b).



(a) Possible sensor mounting locations.



(b) Coverage function.

Figure 5: (a) potential locations approved by NASA to place sensors on the docking port surface of the ISS. (b) coverage function value for each cell describing the "radius" width of coverage from placing a sensor in that cell. Sensors placed toward the center of the grid cover the most (5 cell radius) area of the spacecraft operating grid.

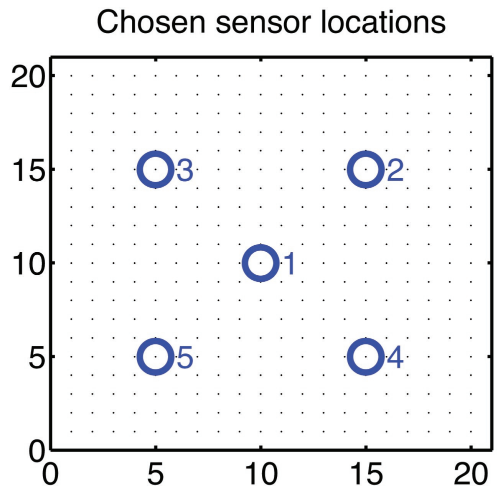
The coverage function value for each cell is the number of cells in any direction that a sensor would cover with sufficiently good quality (for an example of the coverage provided by a sensor with a coverage function value of 3, see Figure 4(a)). Outside of this area of coverage for each sensor, the noise increases exponentially by a factor greater than one. In addition, we assume that sensor coverage overlap, or redundant coverage of cells, adds no value to the effective coverage (see Figure 4(b) for an example of the coverage resulting from placing two sensors with overlapping coverage).

Section 4.2 presents the greedily chosen locations for these sensors, as well as the resulting coverage from the solution set. We then discuss the optimality of the solution with differing budget values. Once the offline design problem is solved, and NASA's 5 sensors are placed around the docking port of the ISS, we examine the problem of online probing of sensors. Probing each sensor for information takes a certain amount of energy and bandwidth, both of which are invaluable resources in space applications. It is desirable, therefore, for the Dragon capsule to sequentially probe one to three sensors before issuing its control commands to navigate toward the center of the operating region (the area of lowest risk cost in Figure 2). Section 4.3 presents two methods of selection, the resulting HMM simulation results, and the respective reductions in overall entropy.

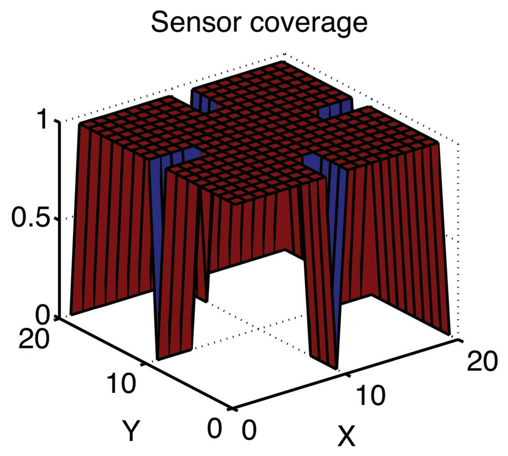
## 4.2 Offline Design

Our first attempt at using a lazy greedy approach to solve our problem involved the Matlab Submodular Function Optimization (SFO) toolbox. This toolbox is a collection of Matlab functions written by Andreas Krause and freely available from `http://www.submodularity.org`. After a few creative attempts at basic functionality with it, followed by extensive examination of the code provided in the toolbox, we determined that the toolbox was specifically written for the univariate problems described in Krause's many publications, which was not suitable for our application. This motivated our decision to write our own functions in Matlab for submodular function optimization.

Using our own implementations in our second attempt, we created a 20×20 matrix to represent possible spacecraft positions, a vector of the 13 possible sensor locations in the corresponding cells on the ISS docking port shown in Figure 5(a), and the sensor coverage function shown above in Figure 5(b). We then implemented Algorithm 2 (refer to the attached code), a lazy greedy optimizer of sensor coverage. At its first step, it computes the coverage for each one of the sensors and stores them in a list sorted in descending order of coverage. In subsequent iterations, the algorithm begins by evaluating only the first element in this list, then either adding it to the solution set or placing it back in the sorted list. This continues until the budget has been reached (in our demonstration problem for this project, NASA's budget is 5 total sensors to be placed).



(a) Chosen sensor locations.

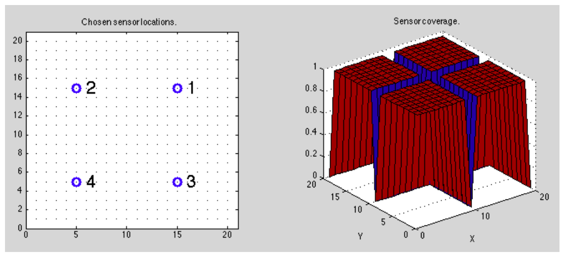


(b) Resulting grid coverage.

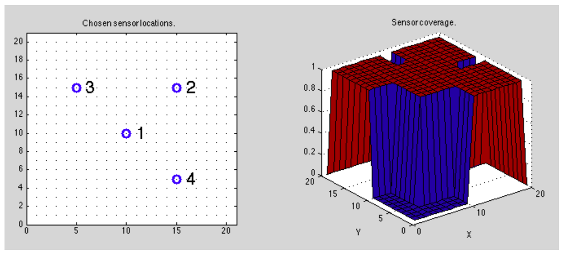
Figure 6: (a) sensor locations chosen by the lazy greedy algorithm, labeled according to order chosen. (b) coverage resulting from the lazy greedy's solution to the problem of maximum coverage with 5 sensors.

The locations chosen by the greedy algorithm are shown in Figure 6(a). Due to the simplifying assumptions of this problem, we conclude that the lazy greedy indeed chose the optimal set of sensors, given its budget of 5 total sensors. The resulting coverage of the Dragon's area of operation is shown in Figure 6(b). It should be noted that coverage value of "1" for a cell refers strictly to whether that cell is ideally covered by a sensor (see Section 3.2); the coverage value is not a binary indicator of whether the spacecraft can be detected in a cell or not.

It should be obvious from some short calculations that the lazy greedy indeed chose the optimal solution to place 5 sensors for this problem. But what if NASA's budget was only 4 sensors? The optimal solution in this case is depicted in Figure 7(a). The lazy greedy algorithm would still choose the first 4 sensors in the same order, leaving one corner of the grid largely uncovered, shown in Figure 7(b). The solution quality in terms of total coverage function value is around 89.2% of the optimal placement's value, which is well above the theoretical lower bound of $\sim 63\%$ discussed previously, but the solution itself would not look very favorable to even a casual observer. The budget for sensor placement, then, turns out to be a deciding factor in the relative quality of the solution found by the lazy greedy; and a higher budget does not mean a higher quality solution relative to the optimal placement.



(a) Optimal coverage.



(b) Lazy coverage.

Figure 7: (a) optimal sensor coverage with a budget of 4 sensors. (b) lazy greedy's sensor coverage with a budget of 4 sensors.

## 4.3 Online Probing

Once the sensors are installed on the ISS at the locations chosen in Figure 6(a), an online problem of sensor selection arises for spacecraft docking with the ISS. Choosing a sensor at random is a very fast method of selection, but there is a chance that the sampled sensor may not provide that much information to the spacecraft. This is primarily due to the increased variance and degradation of sensor quality outside of each sensor's coverage area. We implemented the random selection algorithm as a baseline, plotting the resulting HMM belief after each step. The initial belief and results after each step are shown below in Figure 8.

As Figure 8 shows, the initial belief of the HMM has quite a spread across the space, centered around the (10,10) coordinate. The true position of the spacecraft, not shown, is at (16, 16). After randomly sampling one sensor, the belief is updated, shown in the

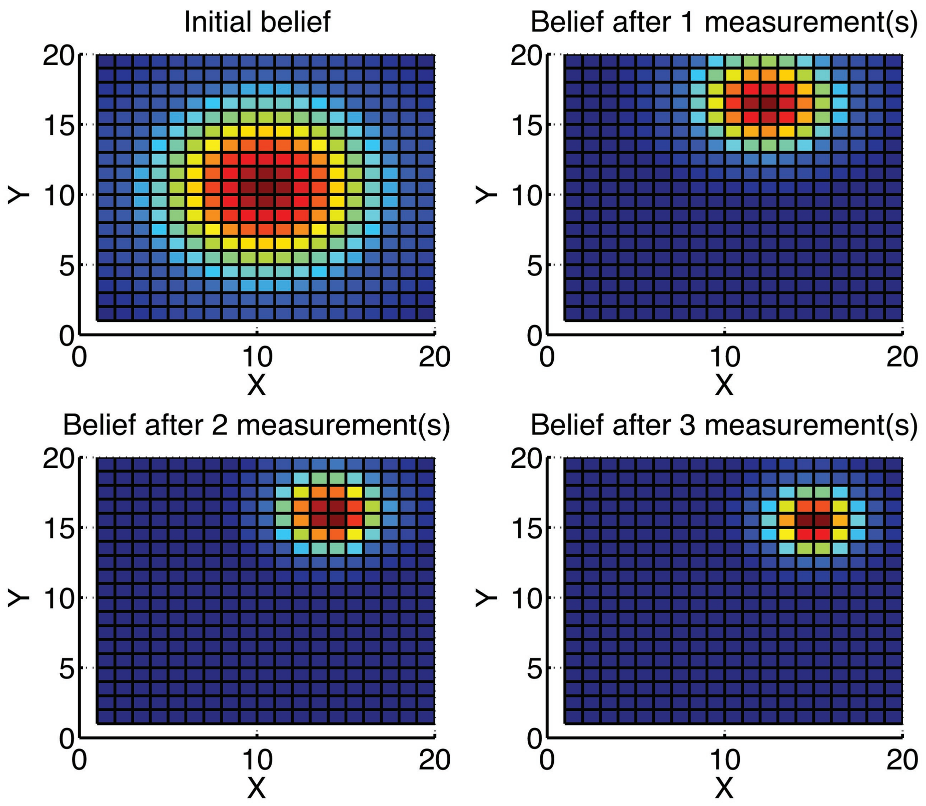


Figure 8: A plot of the HMM belief over 4 steps, with updates after sampling a randomly selected sensor at each step.

upper right-hand corner of the figure. The second and third updates are shown along the bottom of the figure.

After obtaining the results above from the random selection algorithm, we proceeded to implement an adapted version of the online sampling algorithm presented in [4]. Our hypothesis was that greedy selection of the sensor most likely to reduce the average entropy in the system would provide a belief more concentrated around the true spacecraft position. The results are shown below in Figure 9.

Figure 9 clearly shows the adaptive sampling algorithm did not improve much over the random algorithm. From the plots shown, especially after the first udpate, it actually appears that it performs worse than the random sampling. This result is not surprising, as the random selection algorithm is bound to exhibit good behavior at some point due to its random nature. To better understand the difference in average performance between these two algorithms, we simulated 100 trials of 3 sequential

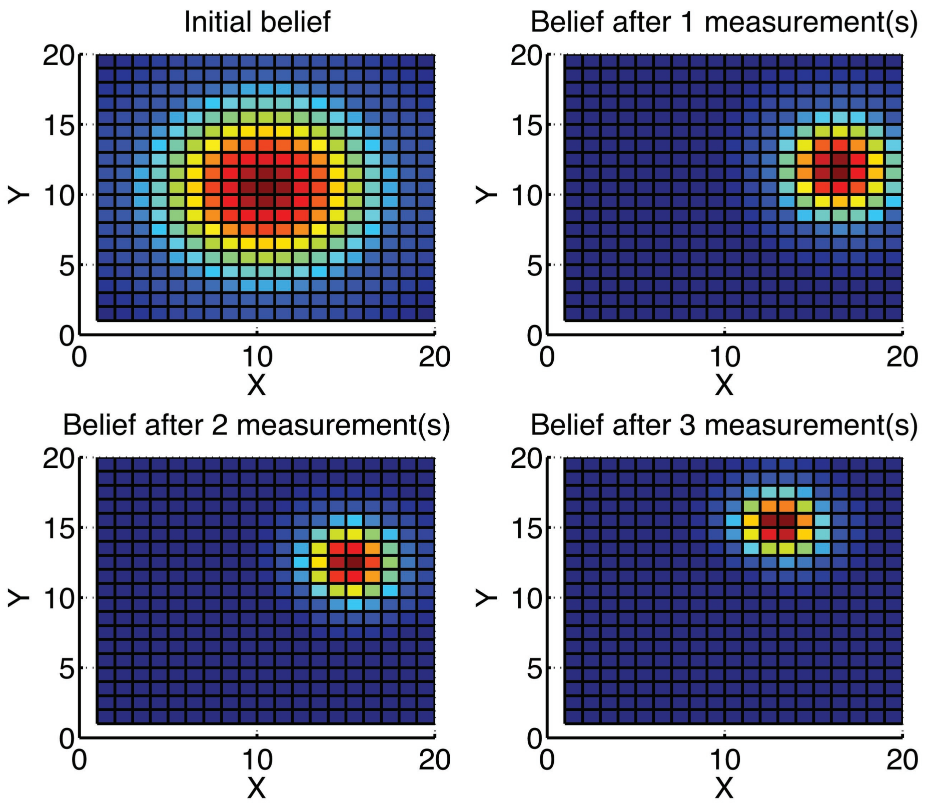


Figure 9: A plot of the HMM belief over 4 steps, with updates after sampling a greedily-chosen selected sensor at each step.

probing steps each and averaged the resulting entropy after each step. The results are shown below in Figure 10.

As Figure 10 shows, the conditional entropy of the system decreases with each measurement. The adaptive algorithm does indeed dominate the random sampling in decreasing the system entropy with each step, and the difference becomes more apparent with each successive choice of sensor to sample. The vertical axis of the figure is non-zero-based to better show this difference. However, the small average difference between the results does raise the question of whether the extra calculations for the adaptive sampling is worth it. The Dragon spacecraft will no doubt have limited computational power, and the greedy algorithm's evaluation of the conditional entropy is a non-trivial calculation, requiring approximation to complete in polynomial time (see Section 3 for a discussion of the computational complexity of computing conditional entropy). Therefore, with a small sensor set, such as the 5 total sensors in our problem, random

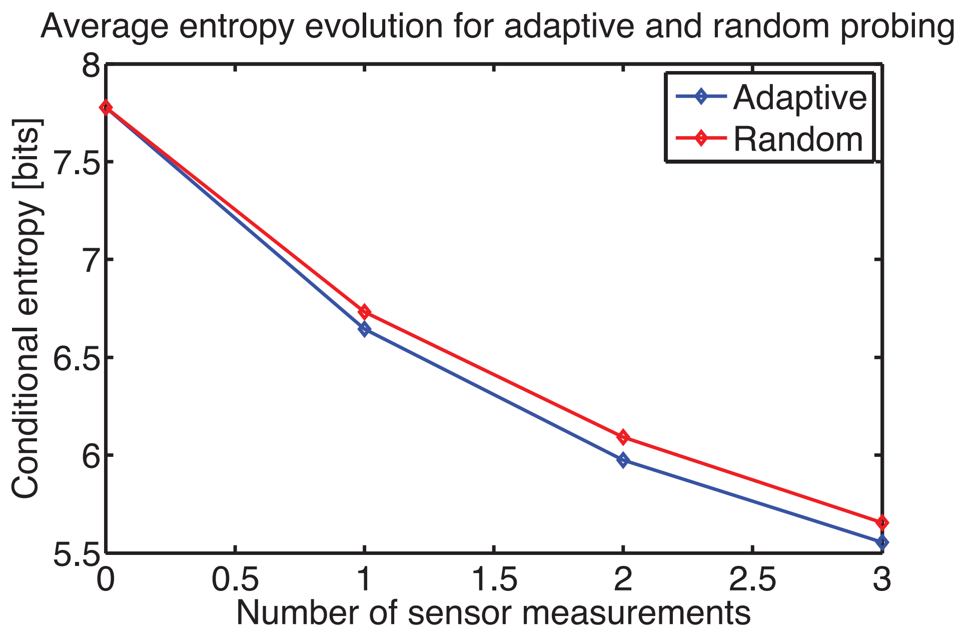


Figure 10: A comparison of the average entropy reduction obtained by adaptive and random sampling methods in 100 trials of online sensor probing.

selection may be the best algorithm to use for reasons of speed, simplicity, and raw CPU cycles.

## 5 Lessons Learned

We take away a couple of key observations from this project. First, it is far easier to dream up an interesting problem that might be related to a theoretical concept than it is to apply the theory to that problem. We went through several iterations of brainstorming, problem selection, and scoping of goals before landing on our current application of offline and online spacecraft position sensing.

For instance, we considered a problem from the automated diagnosis field, such as locations to put "probing points" in a circuit that would be most informative for online diagnostics later. We thought that this application would be an interesting alternative to the online greedy probing described in [4], in a case where probing were prohibitively expensive (or physically impossible) once the system was in operation. We noted that the observations of such a circuit are conditionally independent, and that the conditional entropy is non-decreasing, meaning that we should be able to exploit the submodularity of the problem to find a near-optimal solution within any given budget of $N$ probing points. However, in order to be able to represent interesting circuits that were significantly different from the very simple examples seen in class, we would need a way to describe a large number of circuit component classes with different types of connections between them. This led us to begin considering other space applications.

In order to render our spacecraft position estimation application more meaningful, we considered arbitrary belief state distributions over a reasonably-sized grid of discrete states and possible observations, in addition to space-varying observation models that should model sensor output degradation. Not only were these conditions complex to model, but they also posed additional challenges to our implementations of exact inference algorithms, which we learned from our practical experience do not scale well with the problem size. During our search for alternatives, we learned about sampling-based approximation methods that could greatly speed up the computations in our application while retaining, or at least bounding, the performance guarantees of the algorithms. Another major takeaway from our implementations of submodular function optimizers was the simplicity of coding greedy strategies to approximate hard optimization problems. Not only they are easy to code, but their simplicity lends to predictability, making them suitable for risky autonomous operations where it is of paramount importance to be able to anticipate a robot's actions before any tragic event takes place.

# Bibliography

[1] Mark Fiala. ARTag fiducial marker system applied to vision based spacecraft docking. In *Proc. Intl. Conf. Intelligent Robots and Systems (IROS) 2005 Workshop on Robot Vision for Space Applications*, pages 35–40, 2005.

[2] NK Philip and MR Ananthasayanam. Relative position and attitude estimation and control schemes for the final phase of an autonomous docking mission of spacecraft. *Acta Astronautica*, 52(7):511–522, 2003.

[3] Eric Lambrecht and Subbarao Kambhampati. Planning for information gathering: A tutorial survey. *ASU CSE Techincal Report*, pages 96–017, 1997.

[4] Johan De Kleer and Brian C Williams. Diagnosing multiple faults. *Artificial intelligence*, 32(1):97–130, 1987.

[5] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14(1):265–294, 1978.

[6] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[7] Andreas Krause and Carlos Guestrin. Near-optimal nonmyopic value of information in graphical models. *CoRR*, abs/1207.1394, 2012.

[8] Michel Minoux. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*, pages 234–243. Springer, 1978.

[9] Satoru Fujishige. Polymatroidal dependence structure of a set of random variables. *Information and Control*, 39(1):55–72, 1978.