Greedy Algorithms for Sparse Sensor Placement via Deep Learning

Louis Ly and Yen-Hsi Richard Tsai
louisly@utexas.edu, ytsai@math.utexas.edu

Oden Institute for Computational Engineering and Sciences
The University of Texas at Austin
Abstract

We consider the exploration problem: an agent equipped with a depth sensor must map out a previously unknown environment using as few sensor measurements as possible. We propose an approach based on supervised learning of a greedy algorithm. We provide a bound on the optimality of the greedy algorithm using submodularity theory. Using a level set representation, we train a convolutional neural network to determine vantage points that maximize visibility. We show that this method drastically reduces the on-line computational cost and determines a small set of vantage points that solve the problem. This enables us to efficiently produce highly-resolved and topologically accurate maps of complex 3D environments. Unlike traditional next-best-view and frontier-based strategies, the proposed method accounts for geometric priors while evaluating potential vantage points. While existing deep learning approaches focus on obstacle avoidance and local navigation, our method aims at finding near-optimal solutions to the more global exploration problem. We present realistic simulations on 2D and 3D urban environments.

1 Introduction

We consider the problem of generating a minimal sequence of observing locations to achieve complete line-of-sight visibility coverage of an environment. In particular, we are interested in the case when environment is initially...
unknown. This is particularly useful for autonomous agents to map out unknown, or otherwise unreachable environments, such as undersea caverns. Military personnel may avoid dangerous situations by sending autonomous agents to scout new territory. We first assume the environment is known in order to gain insights.

Consider a domain \( \Omega \subseteq \mathbb{R}^d \). Partition the domain \( \Omega = \Omega_{\text{free}} \cup \Omega_{\text{obs}} \) into an open set \( \Omega_{\text{free}} \) representing the free space, and a closed set \( \Omega_{\text{obs}} \) of finite obstacles without holes. We will refer to the \( \Omega_{\text{obs}} \) as the environment, since it is characterized by the obstacles. Let \( x_i \in \Omega_{\text{free}} \) be a vantage point, from which a range sensor, such as LiDAR, takes omnidirectional measurements \( P_{x_i} : S^{d-1} \rightarrow \mathbb{R} \). That is, \( P_{x_i} \) outputs the distance to closest obstacle for each direction in the unit sphere. One can map the range measurements to the visibility set \( \mathcal{V}_{x_i} \); points in \( \mathcal{V}_{x_i} \) are visible from \( x_i \):

\[
x \in \mathcal{V}_{x_i} \text{ if } \|x - x_i\|_2 < P_{x_i}\left(\frac{x - x_i}{\|x - x_i\|_2}\right)
\]

As more range measurements are acquired, \( \Omega_{\text{free}} \) can be approximated by the *cumulatively visible set* \( \Omega_k \):

\[
\Omega_k = \bigcup_{i=0}^{k} \mathcal{V}_{x_i}
\]

By construction, \( \Omega_k \) admits partial ordering: \( \Omega_{i-1} \subset \Omega_{i} \). For suitable choices of \( x_i \), it is possible that \( \Omega_n \rightarrow \Omega_{\text{free}} \) (say, in the Hausdorff distance).

We aim at determining a *minimal set of vantage points* \( O \) from which
every $x \in \Omega_{\text{free}}$ can be seen. One may formulate a constrained optimization problem and look for sparse solutions. When the environment is known, we have the *surveillance* problem:

$$\min_{O \subseteq \Omega_{\text{free}}} |O| \quad \text{subject to } \Omega_{\text{free}} = \bigcup_{x \in O} V_x. \quad (3)$$

When the environment is not known apriori, the agent must be careful to avoid collision with obstacles. New vantage points must be a point that is currently visible. That is, $x_{k+1} \in \Omega_k$. Define the set of admissible sequences:

$$A(\Omega_{\text{free}}) := \{ (x_0, \ldots, x_{n-1}) \mid n \in \mathbb{N}, \ x_0 \in \Omega_{\text{free}}, \ x_{k+1} \in \Omega_k \}. \quad (4)$$

For the unknown environment, we have the *exploration* problem:

$$\min_{O \in A(\Omega_{\text{free}})} |O| \quad \text{subject to } \Omega_{\text{free}} = \bigcup_{x \in O} V_x. \quad (5)$$

The problem is feasible as long as obstacles do not have holes.

### 1.1 Related works

The surveillance problem is related to the art gallery problem in computational geometry, where the task is to determine the minimum set of guards who can together observe a polygonal gallery. Vertex guards must be stationed at the vertices of the polygon, while point guards can be anywhere in the interior. For simply-connected polygonal scenes, Chvátal showed that
\lfloor n/3 \rfloor$ vertex guards, where $n$ is the number of vertices, are sometimes necessary and always sufficient [6]. For polygonal scenes with $h$ holes, $\lfloor (n + h)/3 \rfloor$ point guards are sufficient [5, 12]. However, determining the optimal set of observers is NP-complete [36, 24, 19].

Goroshin et al. propose an alternating minimization scheme for optimizing the visibility of $N$ observers [10]. Kang et al. use a system of differential equations to optimize the location and orientation of $N$ sensors to maximize surveillance [13]. Both works assume the number of sensors is given.

For the exploration problem, the “wall-following” strategy may be used to map out simple environments [39]. LaValle and Tovar et al. [32, 18, 33] combine wall-following with a gap navigation tree to keep track of gaps, critical events which hide a connected region of the environment that is occluded from a vantage point. Exploration is complete when all gaps have been eliminated. This approach does not produce any geometric representation of the
environment upon completion, due to limited information from gap sensors.

A class of approaches pick new vantage points along shadow boundaries (aka frontiers), the boundary between free and occluded regions [38]. Ghosh et al. propose a frontier-based approach for 2D polygonal environments which requires \( r + 1 \) views, where \( r \) is the number of reflex angles [8]. For general 2D environments, Landa et al. [17, 15, 16] use high order ENO interpolation to estimate curvature, which is then used to determine how far past the horizon to step. However, it is not necessarily optimal to pick only points along the shadow boundary, e.g. when the map is a star-shaped polygon [8].

Next-best-view algorithms try to find vantage points that maximize a utility function, consisting of some notion of information gain and another criteria such as path length. The vantage point does not have to lie along the shadow boundary. A common measure of information gain is the volume of entire unexplored region within sensor range that is not occluded by obstacles [9, 3, 4, 11]. Surmann et al. count the number of intersections of rays into the occlusion [29], while Valente et al. [37] use the surface area of the shadow boundary, weighted by the viewing angle from the vantage points, to define potential information gain. The issue with these heuristics is that they are independent of the underlying geometry. In addition, computing the information gain at each potential vantage point is costly and another heuristic is used to determine which points to sample.

There has been some attempts to incorporate deep learning into the exploration problem, but they focus on navigation rather than exploration.
The approach of Bai et al. \cite{1} terminates when there is no occlusion within view of the agent, even if the global map is still incomplete. Tai and Liu \cite{30, 31, 20} train agents to learn obstacle avoidance.

Our work uses a gain function to steer a greedy approach, similar to the next-best-view algorithms. However, our measure of information gain takes the geometry of the environment into account. By taking advantage of precomputation via convolutional neural networks, our model learns shape priors for a large class of obstacles and is efficient at runtime. We use a volumetric representation which can handle arbitrary geometries in 2D and 3D. Also, we assume that the sensor range is larger than the domain, which makes the problem more global and challenging.

2 Greedy algorithm

We propose a greedy approach which sequentially determines a new vantage point, $x_{k+1}$, based on the information gathered from all previous vantage points, $x_0, x_1, \cdots, x_k$. The strategy is greedy because $x_{k+1}$ would be a location that maximizes the information gain.

For the surveillance problem, the environment is known. We define the gain function:

$$g(x; \Omega_k) := |V_x \cup \Omega_k| - |\Omega_k|,$$

i.e. the volume of the region that is visible from $x$ but not from $x_0, x_1, \cdots, x_k$. Note that $g$ depends on $\Omega_{obs}$, which we omit for clarity of notation. The next
vantage point should be chosen to maximize the newly-surveyed volume. We define the greedy surveillance algorithm as:

$$x_{k+1} = \arg \max_{x \in \Omega_{\text{free}}} g(x; \Omega_k).$$

(7)

The problem of exploration is even more challenging since, by definition, the environment is not known. Subsequent vantage points must lie within the current visible set $\Omega_k$. The corresponding greedy exploration algorithm is

$$x_{k+1} = \arg \max_{x \in \Omega_k} g(x; \Omega_k).$$

(8)

However, we remark that in practice, one is typically interested only in a subset $S$ of all possible environments $S := \{\Omega_{\text{obs}} | \Omega_{\text{obs}} \subseteq \mathbb{R}^d\}$.

For example, cities generally follow a grid-like pattern. Knowing these priors can help guide our estimate of $g$ for certain types of $\Omega_{\text{obs}}$, even when $\Omega_{\text{obs}}$ is unknown initially.

We propose to encode these priors formally into the parameters, $\theta$, of a learned function:

$$g_\theta(x; \Omega_k, B_k) \text{ for } \Omega_{\text{obs}} \in S,$$

(9)

where $B_k$ is the part of $\partial \Omega_k$ that may actually lie in the free space $\Omega_{\text{free}}$:

$$B_k = \partial \Omega_k \setminus \Omega_{\text{obs}}.$$

(10)

See Figure 2 for an example gain function. We shall demonstrate that
while training for $g_\theta$, incorporating the shadow boundaries helps, in some sense, localize the learning of $g$, and is essential in creating usable $g_\theta$.

![Figure 2: Left: the map of a scene consisting of two disks. Right: the intensity of the corresponding gain function. The current vantage point is shown as the red dot. The location which maximizes the gain function is shown as the red x.](image)

2.1 A bound for the known environment

We present a bound on the optimality of the greedy algorithm, based on submodularity [14], a useful property of set functions. We start with standard definitions. Let $V$ be a finite set and $f : 2^V \rightarrow \mathbb{R}$ be a set function which assigns a value to each subset $S \subseteq V$.

**Definition 2.1.** (Monotonicity) A set function $f$ is monotone if for every $A \subseteq B \subseteq V$,

$$f(A) \leq f(B).$$
Definition 2.2. (Discrete derivative) The discrete derivative of $f$ at $S$ with respect to $v \in V$ is

$$\Delta_f(v|S) := f(S \cup \{v\}) - f(S).$$

Definition 2.3. (Submodularity) A set function $f$ is submodular if for every $A \subseteq B \subseteq V$ and $v \in V \setminus B$,

$$\Delta_f(v|A) \geq \Delta_f(v|B).$$

In other words, set functions are submodular if they have diminishing returns. More details and extensions of submodularity can be found in [14].

Now, suppose the environment $\Omega_{\text{obs}}$ is known. Let $O$ be the set of vantage points, and let $f(O)$ be the volume of the region visible from $O$:

$$\mathcal{V}(O) := \bigcup_{x \in O} \mathcal{V}_x$$

$$f(O) := \left| \mathcal{V}(O) \right|$$

(11)

Lemma 2.1. The function $f$ is monotone.

Proof. Consider $A \subseteq B \subseteq \Omega_{\text{free}}$. Since $f$ is the cardinality of unions of sets,
we have
\[ f(B) = \left| \bigcup_{x \in B} \mathcal{V}_x \right| = \left| \bigcup_{x \in A \cup (B \setminus A)} \mathcal{V}_x \right| \geq \left| \bigcup_{x \in A} \mathcal{V}_x \right| = f(A). \]

\[ \square \]

**Lemma 2.2.** The function \( f \) is submodular.

**Proof.** Suppose \( A \subseteq B \) and \( \{v\} \in \Omega_{\text{free}} \setminus B \). By properties of unions and intersections, we have
\[
 f(A \cup \{v\}) + f(B) = \left| \bigcup_{x \in (A \cup \{v\})} \mathcal{V}_x \right| + \left| \bigcup_{x \in B} \mathcal{V}_x \right| \\
\geq \left| \bigcup_{x \in A \cup \{v\} \cup B} \mathcal{V}_x \right| + \left| \bigcup_{x \in (A \cup \{v\}) \cap B} \mathcal{V}_x \right| \\
= \left| \bigcup_{x \in B \cup \{v\}} \mathcal{V}_x \right| + \left| \bigcup_{x \in A} \mathcal{V}_x \right| \\
= f(B \cup \{v\}) + f(A)
\]

Rearranging, we have
\[
f(A \cup \{v\}) + f(B) \geq f(B \cup \{v\}) + f(A)
f(A \cup \{v\}) - f(A) \geq f(B \cup \{v\}) - f(B)
\Delta f(v | A) \geq \Delta f(v | B).
\]
Submodularity and monotonicity enable a bound which compares the relative performance of the greedy algorithm to the optimal solution.

**Theorem 2.3.** Let $O^*_k$ be the optimal set of $k$ sensors. Let $O_n = \{x_i\}_{i=1}^n$ be the set of $n$ sensors placed using the greedy surveillance algorithm \[7\]. Then,

$$f(O_n) \geq (1 - e^{-n/k}) f(O^*_k).$$

**Proof.** For $l < n$ we have

$$f(O^*_k) \leq f(O^*_k \cup O_l) \quad (12)$$

$$= f(O_l) + \Delta f(O^*_k|O_l) \quad (13)$$

$$= f(O_l) + \sum_{i=1}^{k} \Delta f(x^*_i|O_l \cup \{x^*_1, \ldots, x^*_{i-1}\}) \quad (14)$$

$$\leq f(O_l) + \sum_{i=1}^{k} \Delta f(x^*_i|O_l) \quad (15)$$

$$\leq f(O_l) + \sum_{i=1}^{k} f(O_{l+1}) - f(O_l) \quad (16)$$

$$= f(O_l) + k[f(O_{l+1}) - f(O_l)]. \quad (17)$$

Line (12) follows from monotonicity, (15) follows from submodularity of $f$, and (16) from definition of the greedy algorithm. Define $\delta_l := f(O^*_k) - f(O_l)$,
with \( \delta_0 := f(O_k^*) \). Then

\[
\begin{align*}
  f(O_k^*) - f(O_t) &\leq k[f(O_{t+1}) - f(O_t)] \\
  \delta_t &\leq k[\delta_t - \delta_{t+1}] \\
  \delta_t(1 - k) &\leq -k\delta_{t+1} \\
  \delta_t(1 - \frac{1}{k}) &\geq \delta_{t+1}
\end{align*}
\]

Expanding the recurrence relation with \( \delta_n \), we have

\[
\begin{align*}
  \delta_n &\leq \left(1 - \frac{1}{k}\right)\delta_{n-1} \\
  &\leq \left(1 - \frac{1}{k}\right)^n \delta_0 \\
  &= \left(1 - \frac{1}{k}\right)^n f(O_k^*)
\end{align*}
\]

Finally, substituting back the definition for \( \delta_n \), we have the desired result:

\[
\begin{align*}
  \delta_n &\leq \left(1 - \frac{1}{k}\right)^n f(O_k^*) \\
  f(O_k^*) - f(O_n) &\leq \left(1 - \frac{1}{k}\right)^n f(O_k^*) \\
  f(O_k^*)\left(1 - (1 - 1/k)^n\right) &\leq f(O_n) \\
  f(O_k^*)\left(1 - e^{-n/k}\right) &\leq f(O_n)
\end{align*}
\]

where (18) follows from the inequality \( 1 - x \leq e^{-x} \).

In particular, if \( n = k \), then \( (1 - e^{-1}) \approx 0.63 \). This means that \( k \) steps of the greedy algorithm is guaranteed to cover at least 63% of the total volume,
if the optimal solution can also be obtained with $k$ steps. When $n = 3k$, the greedy algorithm covers at least 95% of the total volume. In [22], it was shown that no polynomial time algorithm can achieve a better bound.

### 2.2 A bound for the unknown environment

When the environment is not known, subsequent vantage points must lie within the current visible set to avoid collision with obstacles:

$$x_{k+1} \in \mathcal{V}(O_k)$$  \hspace{1cm} (19)

Thus, the performance of the exploration algorithm has a strong dependence on the environment $\Omega_{obs}$ and the initial vantage point $x_1$. We characterize this dependence using the notion of the *exploration ratio*.

Given an environment $\Omega_{obs}$ and $A \subseteq \Omega_{free}$, consider the ratio of the marginal value of the greedy exploration algorithm, to that of the greedy surveillance algorithm:

$$\rho(A) := \frac{\sup_{x \in \mathcal{V}(A)} \Delta_f(x|A)}{\sup_{x \in \Omega_{free}} \Delta_f(x|A)}.$$  \hspace{1cm} (20)

That is, $\rho(A)$ characterizes the relative gap (for lack of a better word) caused by the collision-avoidance constraint $x \in \mathcal{V}(A)$. Let $A_x = \{ A \subseteq \Omega_{free} | x \in A \}$
be the set of vantage points which contain $x$. Define the exploration ratio as

$$\rho_x := \inf_{A \in A_x} \rho(A). \quad (21)$$

The exploration ratio is the worst-case gap between the two greedy algorithms, conditioned on $x$. It helps to provide a bound for the difference between the optimal solution set of size $k$, and the one prescribed by $n$ steps the greedy exploration algorithm.

**Theorem 2.4.** Let $O_k^* = \{x_i^*\}_{i=1}^k$ be the optimal sequence of $k$ sensors which includes $x_1^* = x_1$. Let $O_n = \{x_i\}_{i=1}^n$ be the sequence of $n$ sensors placed using the greedy exploration algorithm (8). Then, for $k, n > 1$:

$$f(O_n) \geq \left[1 - \exp\left(\frac{-(n-1)\rho_{x_1}}{k-1}\right)\left(1 - \frac{f(x_1)}{f(O_k^*)}\right)\right] f(O_k^*) f(O_n^*).$$

This is reminiscent of Theorem 2.3 with two subtle differences. The $[1 - \frac{f(x_1)}{f(O_k^*)}]$ term accounts for the shared vantage point $x_1$. If $f(x_1)$ is large, then the exponential term has little effect, since $f(x_1)$ is already close to $f(O_k^*)$. On the other hand, if it is small, then the exploration ratio $\rho_{x_1}$ plays a factor. The idea of the proof is similar, with some subtle differences in algebra to account for the shared vantage point $x_1$, and the exploration ratio $\rho_{x_1}$. 


Proof. We have, for \( l < n \):

\[
f(O_k^*) \leq f(O_k^* \cup O_l)
\]

\[
= f(O_l) + \Delta f(O_k^*|O_l)
\]

\[
= f(O_l) + \sum_{i=1}^{k} \Delta f(x_i^*|O_l \cup \{x_1^*, \ldots, x_{i-1}^*\})
\]  \hspace{1cm} (22)

\[
\leq f(O_l) + \sum_{i=1}^{k} \Delta f(x_i^*|O_l)
\]  \hspace{1cm} (23)

\[
= f(O_l) + \Delta f(x_1^*|O_l) + \sum_{i=2}^{k} \Delta f(x_i^*|O_l)
\]

\[
= f(O_l) + \sum_{i=2}^{k} \Delta f(x_i^*|O_l)
\]  \hspace{1cm} (24)

\[
\leq f(O_l) + \sum_{i=2}^{k} \max_{x \in \Omega_{tree}} \Delta f(x|O_l)
\]

\[
\leq f(O_l) + \frac{1}{\rho_{x_1}} \sum_{i=2}^{k} \max_{x \in \mathcal{V}(O_l)} \Delta f(x|O_l)
\]  \hspace{1cm} (25)

\[
\leq f(O_l) + \frac{1}{\rho_{x_1}} \sum_{i=2}^{k} f(O_{l+1}) - f(O_l)
\]  \hspace{1cm} (26)

\[
= f(O_l) + \frac{k-1}{\rho_{x_1}} [f(O_{l+1}) - f(O_l)].
\]

Line (22) is a telescoping sum, (23) follows from submodularity of \( f \), (24) uses the fact that \( x_1^* \in O_l \), (25) follows from the definition of \( \rho_{x_1} \) and (26) stems from the definition of the greedy exploration algorithm (8).

As before, define \( \delta_l := f(O_k^*) - f(O_l) \). However, this time, note that
\[ \delta_1 := f(O_k^*) - f(O_1) = f(O_k^*) - f(x_1). \] Then

\[ f(O_k^*) - f(O_l) \leq \frac{k-1}{\rho x_1} [f(O_{l+1}) - f(O_l)] \]

\[ \delta_l \leq \frac{k-1}{\rho x_1} [\delta_{l-1}] \]

\[ \delta_l \left(1 - \frac{k-1}{\rho x_1}\right) \leq -\frac{k-1}{\rho x_1} \delta_{l-1} \]

\[ \delta_l \left(1 - \frac{\rho x_1}{k-1}\right) \geq \delta_{l+1} \]

Expanding the recurrence relation with \( \delta_n \), we have

\[ \delta_n \leq \left(1 - \frac{\rho x_1}{k-1}\right)^{\delta_{n-1}} \]

\[ \leq \left(1 - \frac{\rho x_1}{k-1}\right)^{n-1} \delta_1 \]

\[ = \left(1 - \frac{\rho x_1}{k-1}\right)^{n-1} [f(O_k^*) - f(x_1)] \]

Now, substituting back the definition for \( \delta_n \), we arrive at

\[ \delta_n \leq \left(1 - \frac{\rho x_1}{k-1}\right)^{n-1} [f(O_k^*) - f(x_1)] \]

\[ f(O_k^*) - f(O_n) \leq \left(1 - \frac{\rho x_1}{k-1}\right)^{n-1} [f(O_k^*) - f(x_1)] \]

\[ f(O_k^*) - f(x_1) - [f(O_n) - f(x_1)] \leq \left(1 - \frac{\rho x_1}{k-1}\right)^{n-1} [f(O_k^*) - f(x_1)] \]

\[ [f(O_k^*) - f(x_1)] \left(1 - \left[1 - \frac{\rho x_1}{k-1}\right]^{n-1}\right) \leq [f(O_n) - f(x_1)] \]

\[ [f(O_k^*) - f(x_1)] \left(1 - e^{-\frac{(n-1)\rho x_1}{k-1}}\right) \leq [f(O_n) - f(x_1)]. \]
Finally, with some more algebra

\[
[f(O_n) - f(x_1)] \geq \left(1 - e^{-\frac{(n-1)\rho_x}{k-1}}\right)\left[f(O_k^*) - f(x_1)\right]
\]

\[
f(O_n) \geq f(x_1) + \left(1 - e^{-\frac{(n-1)\rho_x}{k-1}}\right)\left[f(O_k^*) - f(x_1)\right]
\]

\[
f(O_n) \geq f(x_1) + \left(1 - e^{-\frac{(n-1)\rho_x}{k-1}}\right)f(O_k^*) + f(x_1)e^{-\frac{(n-1)\rho_x}{k-1}}
\]

\[
f(O_n) \geq \left(1 - e^{-\frac{(n-1)\rho_x}{k-1}}\right)f(O_k^*) + f(x_1)e^{-\frac{(n-1)\rho_x}{k-1}}
\]

\[
f(O_n) \geq \left(1 - e^{-\frac{(n-1)\rho_x}{k-1}}\right)\left[1 - \frac{f(x_1)}{f(O_n^*)}\right]f(O_k^*).
\]

\[\square\]

**Exploration ratio example**

We demonstrate an example where \(\rho_x\) can be an arbitrarily small factor that is determined by the geometry of \(\Omega_{\text{free}}\). Figure 3 depicts an illustration of the setup for the narrow alley environment.

Consider a domain \(\Omega = [0, 1] \times [0, 1]\) with a thin vertical wall of width \(\varepsilon \ll 1\), whose center stretches from \((\frac{3}{2}\varepsilon, 0)\) to \((\frac{3}{2}\varepsilon, 1)\). A narrow opening of size \(\varepsilon^2 \times \varepsilon\) is centered at \((\frac{3}{2}\varepsilon, \frac{1}{2})\). Suppose \(x_1 = x_1^* = A\) so that

\[f(\{x_1\}) = \varepsilon + O(\varepsilon^2),\]

where the \(\varepsilon^2\) factor is due to the small sliver of the narrow alley visible from \(A\). By observation, the optimal solution contains two vantage points. One such solution places \(x_2^* = C\). The greedy exploration algorithm can only
place $x_2 \in \mathcal{V}(x_1) = [0, \varepsilon] \times [0, 1]$. One possible location is $x_2 = B$. Then, after 2 steps of the greedy algorithm, we have

$$f(O_2) = \varepsilon + \mathcal{O}(\varepsilon^2).$$

Meanwhile, the total visible area is

$$f(O^*_2) = 1 - \mathcal{O}(\varepsilon)$$

and the ratio of greedy to optimal area coverage is

$$\frac{f(O_2)}{f(O^*_2)} = \frac{\varepsilon + \mathcal{O}(\varepsilon^2)}{1 - \mathcal{O}(\varepsilon)} = \mathcal{O}(\varepsilon) \quad (27)$$
The exploration ratio is $\rho_{x_1} = \mathcal{O}(\varepsilon^2)$, since

$$\max_{x \in \mathcal{N}(\{x_1\})} \Delta_f(x|\{x_1\}) = \mathcal{O}(\varepsilon^2)$$

(28)

$$\max_{x \in \Omega_{\text{free}}} \Delta_f(x|\{x_1\}) = 1 - \mathcal{O}(\varepsilon)$$

According to the bound, with $k = n = 2$, we should have

$$\frac{f(O_2)}{f(O_2^*)} \geq \left(1 - e^{-\frac{(n-1)\rho_{x_1}}{k+1}} \left[1 - \frac{f(x_1)}{f(O_2^*)}\right]\right)$$

$$= \left(1 - e^{-\mathcal{O}(\varepsilon^2)} \left[1 - \mathcal{O}(\varepsilon)\right]\right)$$

(29)

$$= \Omega(\varepsilon)$$

which reflects what we see in (27).

On the other hand, if $O_2 = \{C, B\}$ and $O_2^* = \{C, B\}$, we would have

$$f(\{x_1\}) = 1 - \mathcal{O}(\varepsilon)$$

and $\rho_{x_1} = 1$, since both the greedy exploration and surveillance step coincide.

According to the bound, with $k = n = 2$, we should have

$$\frac{f(O_2)}{f(O_2^*)} \geq \left(1 - e^{-\frac{(n-1)\rho_{x_1}}{k+1}} \left[1 - \frac{f(x_1)}{f(O_2^*)}\right]\right)$$

(30)

$$\geq 1 - \mathcal{O}(\varepsilon)$$

which is the case, since $f(O_2) = f(O_2^*)$.

By considering the first vantage point $x_1$ as part of the bound, we ac-
count for some of the unavoidable uncertainties associated with unknown environments during exploration.

2.3 Numerical comparison

We compare both greedy algorithms on random arrangements of up to 6 circular obstacles. Each algorithm starts from the same initial position and runs until all free area is covered. We record the number of vantage points required over 200 runs for each number of obstacles.

Surprisingly, the exploration algorithm sometimes requires fewer vantage points than the surveillance algorithm. Perhaps the latter is too aggressive, or perhaps the collision-avoidance constraint acts as a regularizer. For example, when there is a single circle, the greedy surveillance algorithm places the second vantage point $x_2$ on the opposite side of this obstacle. This may lead to two slivers of occlusion forming of either side of the circle, which will require 2 additional vantage points to cover. With the greedy exploration algorithm, we do not have this problem, due to the collision-avoidance constraint. Figure 4 shows a select example with 1 and 5 obstacles. Figure 5 shows the histogram of the number of steps needed for each algorithm. On average, both algorithms require a similar number of steps, but the exploration algorithm has a slight advantage.
Figure 4: Comparing the greedy algorithm for the known (left) and unknown (right) environment on circular obstacles. Spikes on each vantage point indicate the ordering, e.g. the initial point has no spike. Gray areas are shadows from each vantage point. Lighter regions are visible from more vantage points.
Figure 5: Histogram of number of vantage points needed for the surveillance (blue) and exploration (orange) greedy algorithms to completely cover environments consisting up to 6 circles.
3 Learning the gain function

In this section, we discuss the method for approximating the gain function when the map is not known. Given the set of previously-visited vantage points, we compute the cumulative visibility and shadow boundaries. We approximate the gain function by applying the trained neural network on this pair of inputs, and pick the next point according to (7). This procedure repeats until there are no shadow boundaries or occlusions.

The data needed for the training and evaluation of $g_\theta$ are computed using level sets [26, 28, 25]. Occupancy grids may be applicable, but we choose level sets since they have proven to be accurate and robust. In particular, level sets are necessary for subpixel resolution of shadow boundaries and they allow for efficient visibility computation, which is crucial when generating the library of training examples.

The training geometry is embedded by a signed distance function, denoted by $\phi$. For each vantage point $x_i$, the visibility set is represented by the level set function $\psi(\cdot, x_i)$, which is computed efficiently using the algorithm described in [34].

In the calculus of level set functions, unions and intersections of sets are translated, respectively, into taking maximum and minimum of the corresponding characteristic functions. The cumulatively visible sets $\Omega_k$ are
represented by the level set function \( \Psi_k(x) \), which is defined recursively by

\[
\Psi_0(x) = \psi(x, x_0),
\]

\[
\Psi_k(x) = \max \{ \Psi_{k-1}(x), \psi(x, x_k) \}, \quad k = 1, 2, \ldots
\]

where the max is taken point-wise. Thus we have

\[
\Omega_{\text{free}} = \{ x | \phi(x) > 0 \},
\]

\[
\mathcal{V}_{x_i} = \{ x | \psi(x, x_i) > 0 \},
\]

\[
\Omega_k = \{ x | \Psi_k(x) > 0 \}.
\]

The shadow boundaries \( B_k \) are approximated by the ”smeared out” function:

\[
b_k(x) := \delta_\epsilon(\Psi_k) \cdot [1 - H(G_k(x))],
\]

where \( H(x) \) is the Heaviside function and

\[
\delta_\epsilon(x) = \frac{2}{\varepsilon} \cos^2 \left( \frac{\pi x}{\varepsilon} \right) \cdot 1_{[-\frac{\varepsilon}{2}, \frac{\varepsilon}{2}]}(x),
\]

\[
\gamma(x, x_0) = (x_0 - x)^T \cdot \nabla \phi(x),
\]

\[
G_0 = \gamma(x, x_0),
\]

\[
G_k(x) = \max \{ G_{k-1}(x), \gamma(x, x_k) \}, \quad k = 1, 2, \ldots
\]

Recall, the shadow boundaries are the portion of the \( \partial \Omega_k \) that lie in free
space; the role of $1 - H(G_k)$ is to mask out the portion of obstacles that are currently visible from $\{x_i\}_{i=1}^k$. See Figure ?? for an example of $\gamma$. In our implementation, we take $\varepsilon = 3\Delta x$ where $\Delta x$ is the grid node spacing. We refer the readers to [35] for a short review of relevant details.

When the environment $\Omega_{\text{obs}}$ is known, we can compute the gain function exactly

$$g(x; \Omega_k) = \int H\left(H(\psi(\xi, x)) - H(\Psi_k(\xi))\right) d\xi.$$  \hfill (41)

We remark that the integrand will be 1 where the new vantage point uncovers something not previously seen. Computing $g$ for all $x$ is costly; each visibility and volume computation requires $O(m^d)$ operations, and repeating this for all points in the domain results in $O(m^2d)$ total flops. We approximate it with a function $\tilde{g}_\theta$ parameterized by $\theta$:

$$\tilde{g}_\theta(x; \Psi_k, b_k) \approx g(x; \Omega_k).$$ \hfill (42)

If the environment is unknown, we directly approximate the gain function by learning the parameters $\theta$ of a function

$$g_\theta(x; \Psi_k, b_k) \approx g(x; \Omega_k)H(\Psi_k)$$ \hfill (43)

using only the observations as input. Note the $H(\Psi_k)$ factor is needed for collision avoidance during exploration because it is not known \textit{a priori} whether an occluded location $y$ is part of an obstacle or free space. Thus $g_\theta(y)$ must
be zero.

3.1 Training procedure

We sample the environments uniformly from a library. For each $\Omega_{obs}$, a sequence of data pairs is generated and included into the training set $\mathcal{T}$:

\[
\{(\{\Psi_k, b_k\}, g(x; \Omega_k)H(\Psi_k)), \quad k = 0, 1, 2, \ldots \}
\]

For a given environment $\Omega_{obs}$, define a path $O = \{x_i\}_{i=0}^k$ as admissible if $\phi(x_0) > 0$ and $\Psi_i(x_{i+1}) > 0$ for $i = 0, \ldots, k - 1$. That is, it should only contain points in free space and in the case of exploration, subsequent points must be visible from at least one of the previous vantage points. Let $\mathcal{A}$ be the set of admissible paths. Then training set should ideally include all paths in $\mathcal{A}$. However this is too costly, since there are $O(m^{kd})$ paths consisting of $k$ steps. Instead, to generate causally relevant data, we use an $\varepsilon$-greedy approach: we uniformly sample initial positions. With probability $\varepsilon$, the next vantage point is chosen randomly from admissible set. With probability $1 - \varepsilon$, the next vantage point is chosen according to (7). Figure 6 shows an illustration of the generation of causal data along the subspace of relevant shapes.

The function $g_\theta$ is learned by minimizing the empirical loss across all data
Figure 6: Causal data generation along the subspace of relevant shapes. Each dot is a data sample corresponding to a sequence of vantage points.

pairs for each $\Omega_{\text{obs}}$ in the training set $\mathcal{T}$:

$$\argmin_{\theta} \frac{1}{N} \sum_{\Omega_{\text{obs}} \in T} \sum_{k} L \left( g_{\theta}(x; \Psi_k, b_k), g(x; \Omega_k) H(\Psi_k) \right),$$

(45)

where $N$ is the total number of data pairs. We use the cross entropy loss function:

$$L(p, q) = \int p(x) \log q(x) + (1 - p(x)) \log(1 - q(x)) \, dx.$$  

(46)
Figure 7: A training data pair consists of the cumulative visibility and shadow boundaries as input, and the gain function as the output. Each sequence of vantage points generates a data sample which depends strongly the shapes of the obstacles and shadows. a) The underlying map with current vantage points shown in red. b) The cumulative visibility of the current vantage points. c) The corresponding shadow boundaries. d) The corresponding gain function.
Network architecture

We use convolutional neural networks (CNNs) to approximate the gain function, which depends on the shape of $\Omega_{\text{obs}}$ and the location $x$. CNNs have been used to approximate functions of shapes effectively in many applications. Their feedforward evaluations are efficient if the off-line training cost is ignored. The gain function $g(x)$ does not depend directly on $x$, but rather, $x$’s visibility of $\Omega_{\text{free}}$, with a domain of dependence bounded by the sensor range. We employ a fully convolutional approach for learning $g$, which makes the network applicable to domains of different sizes. The generalization to 3D is also straightforward.

We base the architecture of the CNN on U-Net [27], which has had great success in dense inference problems, such as image segmentation. It aggregates information from various layers in order to have wide receptive fields while maintaining pixel precision. The main design choice is to make sure that the receptive field of our model is sufficient. That is, we want to make sure that the value predicted at each voxel depends on a sufficiently large neighborhood. For efficiency, we use convolution kernels of size 3 in each dimension. By stacking multiple layers, we can achieve large receptive fields. Thus the complexity for feedforward computations is linear in the total number of grid points.

Define a conv block as the following layers: convolution, batch norm, leaky relu, stride 2 convolution, batch norm, and leaky relu. Each conv block reduces the image size by a factor of 2. The latter half of the network increases
the image size using *deconv blocks*: bilinear 2x upsampling, convolution, batch norm, and leaky *relu*.

Our 2D network uses 6 *conv blocks* followed by 6 *deconv blocks*, while our 3D network uses 5 of each block. We choose the number of blocks to ensure that the receptive field is at least the size of the training images: 128 × 128 and 64 × 64 × 64. The first *conv block* outputs 4 channels. The number of channels doubles with each *conv block*, and halves with each *deconv block*.

The network ends with a single channel, kernel of size 1 convolution layer followed by the sigmoid activation. This ensures that the network aggregates all information into a prediction of the correct size and range.

4 Numerical results

We present some experiments to demonstrate the efficacy of our approach. Also, we demonstrate its limitations. First, we train on 128 × 128 aerial city blocks cropped from INRIA Aerial Image Labeling Dataset [21]. It contains binary images with building labels from several urban areas, including Austin, Chicago, Vienna, and Tyrol. We train on all the areas except Austin, which we hold out for evaluation. We call this model City-CNN. We train a similar model NoSB-CNN on the same training data, but omit the shadow boundary from the input. Third, we train another model Radial-CNN, on synthetically-generated radial maps, such as the one in Figure 13.

Given a map, we randomly select an initial location. In order to generate
the sequence of vantage points, we apply (7), using $g_\theta$ in place of $g$. Ties are broken by choosing the closest point to $x_k$. We repeat this process until there are no shadow boundaries, the gain function is smaller than $\epsilon$, or the residual is less than $\delta$, where the residual is defined as:

$$r = \frac{|\Omega_{\text{free}} \setminus \Omega_k|}{|\Omega_{\text{free}}|}. \quad (47)$$

We compare these against the algorithm which uses the exact gain function, which we call Exact. We also compare against Random, a random walker, which chooses subsequent vantage points uniformly from the visible region, and Random-SB which samples points uniformly in a small neighborhood of the shadow boundaries. We analyze the number of steps required to cover the scene and the residual as a function of the number of steps.

Figure 8: Comparison of predicted (left) and exact (right) gain function for an Austin map. Although the functions are not identical, the predicted gain function peaks in similar locations to the exact gain function, leading to similar steps.

Lastly, we present simulation for exploring 3D environments. Due to
the limited availability of datasets, the model, 3D-CNN, is trained using synthetic $64 \times 64 \times 64$ voxel images consisting of tetrahedrons, cylinders, ellipsoids, and cuboids of random positions, sizes, and orientations. In the site\cite{visibility.page.link/demo} the interested reader may inspect the performance of the 3D-CNN in some other challenging 3D environments.

For our experiments using trained networks, we make use of a CPU-only machine containing four Intel Core i5-7600 CPU @ 3.50GHz and 8 GB of RAM. Additionally, we use an Nvidia Tesla K40 GPU with 12 GB of memory for training and predicting the gain function in 3D scenes.

Figure 9: Distribution of the residual and number of steps generated across multiple runs over an Austin map. The proposed method is robust against varying initial conditions. The algorithm reduces the residual to roughly 0.1 % within 39 steps by using a threshold on the predicted gain function as a termination condition.

\footnote{http://visibility.page.link/demo}
Figure 10: An example of 36 vantage points (red disks) using City-CNN model. White regions are free space while gray regions are occluded. Black borders indicate edges of obstacles.
Figure 11: Graph showing the decrease in residual over 50 steps among various algorithms starting from the same initial position for an Austin map. Without using shadow boundary information, NoSB-CNN can at times be worse than Random. Our City-CNN model is significantly faster than Exact while remaining comparable in terms of residual.

2D city

The City-CNN model works well on 2D Austin maps. First, we compare the predicted gain function to the exact gain function on a $128 \times 128$ map, as in Figure 8. Without knowing the underlying map, it is difficult to accurately determine the gain function. Still, the predicted gain function peaks in locations similar to those in the exact gain function. This results in similar sequences of vantage points.

The algorithm is robust to the initial positions. Figure 9 show the distribution of the number of steps and residual across over 800 runs from varying initial positions over a $512 \times 512$ Austin map. In practice, using the shadow
boundaries as a stopping criteria can be unreliable. Due to numerical precision and discretization effects, the shadow boundaries may never completely disappear. Instead, the algorithm terminates when the maximum predicted gain falls below a certain threshold $\epsilon$. In this example, we used $\epsilon = 0.1$. Empirically, this strategy is robust. On average, the algorithm required 33 vantage points to reduce the occluded region to within 0.1% of the explorable area.

Figure 10 shows an example sequence consisting of 36 vantage points. Each subsequent step is generated in under 1 sec using the CPU and instantaneously with a GPU.

Even when the maximizer of the predicted gain function is different from that of the exact gain function, the difference in gain is negligible. This is evident when we see the residuals for City-CNN decrease at similar rates to Exact. Figure 11 demonstrates an example of the residual as a function of the number of steps for one such sequence generated by these algorithms on a $1024 \times 1024$ map of Austin. We see that City-CNN performs comparably to Exact approach in terms of residual. However, City-CNN takes 140 secs to generate 50 steps on the CPU while Exact, an $\mathcal{O}(m^4)$ algorithm, takes more than 16 hours to produce 50 steps.

**Effect of shadow boundaries**

*The inclusion of the shadow boundaries as input to the CNN is critical for the algorithm to work.* Without the shadow boundaries, the algorithm cannot
Figure 12: A sequence of 50 vantage points generated from NoSB-CNN. The points cluster near flat edges due to ambiguity and the algorithm becomes stuck. Gray regions without black borders have not been fully explored.
distinguish between obstacles and occluded regions. If an edge corresponds to an occluded region, then choosing a nearby vantage point will reduce the residual. However, choosing a vantage point near a flat obstacle will result in no change to the cumulative visibility. At the next iteration, the input is same as the previous iteration, and the result will be the same; the algorithm becomes stuck in a cycle. To avoid this, we prevent vantage points from repeating by zeroing out the gain function at that point and recomputing the argmax. Still, the vantage points tend to cluster near flat edges, as in Figure 12. This clustering behavior causes the NoSB-CNN model to be, at times, worse than Random. See Figure 11 to see how the clustering inhibits the reduction in the residual.

**Effect of shape**

The shape of the obstacles, i.e. $\Omega^r$, used in training affects the gain function predictions. Figure 13 compares the gain functions produced by City-CNN and Radial-CNN.

**Frequency map**

Here we present one of our studies concerning the exclusivity of vantage point placements in $\Omega$. We generated sequences of vantage points starting from over 800 different initial conditions using City-CNN model on a $512 \times 512$ Austin map. Then, we model each vantage point as a Gaussian with fixed
Figure 13: Comparison of gain functions produced with various models on a radial scene. Naturally, the CNN model trained on radial obstacles best approximates the true gain function. a) The underlying radial map with vantage points shown in red. b) The exact gain function. c) City-CNN predicted gain function. d) Radial-CNN predicted gain function.
Figure 14: Distribution of vantage points generated by \textbf{City-CNN} method from various initial positions. Hot spots are brighter and are visited more frequently since they are essential for completing coverage.

width, and overlay the resulting distribution on the Austin map in Figure 14. This gives us a frequency map of the most recurring vantage points. These hot spots reveal regions that are more secluded and therefore, the visibility of those regions is more sensitive to vantage point selection. The efficiency of the CNN method allows us to address many surveillance related questions for a large collection of relevant geometries.
Art gallery

Our proposed approach outperforms the computational geometry solution \cite{23} to the art gallery problem, even though we do not assume the environment is known. The key issue with computational geometry approaches is that they are heavily dependent on the triangulation. In an extreme example, consider an art gallery that is a simple convex \textit{n-gon}. Even though it is sufficient to place a single vantage point anywhere in the interior of the room, the triangulation-based approach produces a solution with $\lfloor n/3 \rfloor$ vertex guards.

Figure 15 shows an example gallery consisting of 58 vertices. The computational geometry approach requires $\lfloor n/3 \rfloor = 19$ vantage points to completely cover the scene, even if point guards are used \cite{5,12}. The gallery contains $r = 19$ reflex angles, so the work of \cite{8} requires $r + 1 = 20$ vantage points. On average, City-CNN requires only 8 vantage points.

3D environment

We present a 3D simulation of a 250m×250m environment based on Castle Square Parks in Boston. Figure 16 for snapshots of the algorithm in action. The map is discretized as a level set function on a 768×768×64 voxel grid. At this resolution, small pillars are accurately reconstructed by our exploration algorithm. Each step can be generated in 3 seconds using the GPU or 300 seconds using the CPU. Parallelization of the distance function computation will further reduce the computation time significantly. A map
Figure 15: Comparison of the computational geometry approach and the City-CNN approach to the art gallery problem. The red circles are the vantage points computed by the methods. Left: A result computed by the computational geometry approach, given the environment. Right: An example sequence of 7 vantage points generated by the City-CNN model.

of this size was previously unfeasible. Lastly, Figure 17 shows snapshots from the exploration of a more challenging, cluttered 3D scene with many nooks.

5 Conclusion

From the perspective of inverse problems, we proposed a greedy algorithm for autonomous surveillance and exploration. We show that this formulation can be well-approximated using convolutional neural networks, which learns geometric priors for a large class of obstacles. The inclusion of shadow boundaries, computed using the level set method, is crucial for the success of the algorithm. One of the advantages of using the gain function (6), an
integral quantity, is its stability with respect to noise in positioning and sensor measurements. In practice, we envision that it can be used in conjunction with SLAM algorithms [7, 2] for a wide range of real-world applications.

One may also consider $n$-step greedy algorithms, where $n$ vantage points are chosen simultaneously. However, being more greedy is not necessarily better. If the performance metric is the cardinality of the solution set, then it is not clear that multi-step greedy algorithms lead to smaller solutions. We saw in section 2 that, even for the single circular obstacle, the greedy surveillance algorithm may sometimes require more steps than the exploration algorithm to attain complete coverage.

If the performance metric is based on the rate in which the objective function increases, then a multi-step greedy approach would be appropriate. However, on a grid with $m$ nodes in $d$ dimensions, there are $O(m^{nd})$ possible combinations. For each combination, computing the visibility and gain function requires $O(nm^d)$ cost. In total, the complexity is $O(nm^{d(n+1)})$, which is very expensive, even when used for offline training of a neural network. In such cases, it is necessary to selectively sample only the relevant combinations. One such way to do that, is through a tree search algorithm.
Figure 16: Snapshots demonstrating the exploration of an initially unknown 3D urban environment using sparse sensor measurements. The red spheres indicate the vantage point. The gray surface is the reconstruction of the environment based on line of sight measurements taken from the sequence of vantage points. New vantage points are computed in virtually real-time using 3D-CNN.
Figure 17: Snapshots of 3D-CNN applied to exploration of a cluttered scene.
Acknowledgment

This work was partially supported by NSF grant DMS-1913209.

References

[1] Shi Bai, Fanfei Chen, and Brendan Englot. Toward autonomous mapping and exploration for mobile robots through deep supervised learning. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 2379–2384. IEEE, 2017.

[2] Tim Bailey and Hugh Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. IEEE Robotics & Automation Magazine, 13(3):108–117, 2006.

[3] Andreas Bircher, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. Receding horizon” next-best-view” planner for 3d exploration. In 2016 IEEE international conference on robotics and automation (ICRA), pages 1462–1468. IEEE, 2016.

[4] Andreas Bircher, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. Receding horizon path planning for 3d exploration and surface inspection. Autonomous Robots, 42(2):291–306, 2018.
[5] Iliana Bjorling-Sachs and Diane L. Souvaine. An efficient algorithm for guard placement in polygons with holes. *Discrete & Computational Geometry*, 13(1):77–109, 1995.

[6] Václav Chvátal. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory*, B(18):39–41, 1975.

[7] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

[8] Subir K. Ghosh, Joel W. Burdick, Amitava Bhattacharya, and Sudeep Sarkar. Online algorithms with discrete visibility - exploring unknown polygonal environments. *IEEE Robotics Automation Magazine*, 15(2):67–76, June 2008.

[9] Héctor H González-Banos and Jean-Claude Latombe. Navigation strategies for exploring indoor environments. *The International Journal of Robotics Research*, 21(10-11):829–848, 2002.

[10] Rostislav Goroshin, Quyen Huynh, and Hao-Min Zhou. Approximate solutions to several visibility optimization problems. *Communications in Mathematical Sciences*, 9(2):535–550, 2011.

[11] Lionel Heng, Alkis Gotovos, Andreas Krause, and Marc Pollefeys. Efficient visual exploration and coverage with a micro aerial vehicle in
unknown environments. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1071–1078. IEEE, 2015.

[12] Frank Hoffmann, Michael Kaufmann, and Klaus Kriegel. The art gallery theorem for polygons with holes. In *1991 Proceedings 32nd Annual Symposium of Foundations of Computer Science*, pages 39–48. IEEE, 1991.

[13] Sung Ha Kang, Seong Jun Kim, and Haomin Zhou. Optimal sensor positioning; a probability perspective study. *SIAM Journal on Scientific Computing*, 39(5):B759–B777, 2017.

[14] Andreas Krause and Daniel Golovin. Submodular function maximization. In *Tractability: Practical Approaches to Hard Problems*, pages 71–104. Cambridge University Press, 2014.

[15] Yanina Landa, David Galkowski, Yuan R Huang, Abhijeet Joshi, Christine Lee, Kevin K Leung, Gitendra Malla, Jennifer Treanor, Vlad Voroninski, Andrea L Bertozzi, Yen-Hsi Richard Tsai, et al. Robotic path planning and visibility with limited sensor data. In *American Control Conference, 2007. ACC’07*, pages 5425–5430. IEEE, 2007.

[16] Yanina Landa and Yen-Hsi Richard Tsai. Visibility of point clouds and exploratory path planning in unknown environments. *Communications in Mathematical Sciences*, 6(4):881–913, 2008.
[17] Yanina Landa, Yen-Hsi Richard Tsai, and Li-Tien Cheng. Visibility of point clouds and mapping of unknown environments. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 1014–1025. Springer, 2006.

[18] Steven LaValle. *Planning algorithms*. Cambridge University Press, 2006.

[19] Der-Tsai Lee and Arthur Lin. Computational complexity of art gallery problems. *IEEE Transactions on Information Theory*, 32(2):276–282, 1986.

[20] Tai Lei and Liu Ming. A robot exploration strategy based on q-learning network. In *2016 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pages 57–62. IEEE, 2016.

[21] Emmanuel Maggiori, Yuliya Tarabalka, Guillaume Charpiat, and Pierre Alliez. Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2017.

[22] George L Nemhauser and Laurence A Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of operations research*, 3(3):177–188, 1978.

[23] Joseph O’Rourke. *Art gallery theorems and algorithms*, volume 57. Oxford University Press Oxford, 1987.
[24] Joseph O’Rourke and Kenneth Supowit. Some np-hard polygon decomposition problems. *IEEE Transactions on Information Theory*, 29(2):181–190, 1983.

[25] Stanley Osher and Ronald Fedkiw. *Level set methods and dynamic implicit surfaces*, volume 153. Springer Science & Business Media, 2006.

[26] Stanley Osher and James Albert Sethian. Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations. *Journal of computational physics*, 79(1):12–49, 1988.

[27] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[28] James Albert Sethian. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, volume 3. Cambridge university press, 1999.

[29] Hartmut Surmann, Andreas Nüchter, and Joachim Hertzberg. An autonomous mobile robot with a 3d laser range finder for 3d exploration and digitalization of indoor environments. *Robotics and Autonomous Systems*, 45(3-4):181–198, 2003.
[30] Lei Tai and Ming Liu. Mobile robots exploration through cnn-based reinforcement learning. *Robotics and biomimetics*, 3(1):24, 2016.

[31] Lei Tai, Giuseppe Paolo, and Ming Liu. Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 31–36. IEEE, 2017.

[32] Benjamin Tovar, Luis Guilamo, and Steven LaValle. Gap navigation trees: Minimal representation for visibility-based tasks. In *Algorithmic Foundations of Robotics VI*, pages 425–440. Springer, 2004.

[33] Benjamin Tovar, Rafael Murrieta-Cid, and Steven LaValle. Distance-optimal navigation in an unknown environment without sensing distances. *IEEE Transactions on Robotics*, 23(3):506–518, 2007.

[34] Yen-Hsi Richard Tsai, Li-Tien Cheng, Stanley Osher, Paul Burchard, and Guillermo Sapiro. Visibility and its dynamics in a pde based implicit framework. *Journal of Computational Physics*, 199(1):260–290, 2004.

[35] Yen-Hsi Richard Tsai and Stanley Osher. Total variation and level set methods in image science. *Acta Numerica*, 14:509–573, 2005.

[36] Jorge Urrutia. Art gallery and illumination problems. In *Handbook of Computational Geometry*, pages 973–1027. Elsevier, 2000.
[37] Luca Valente, Yen-Hsi Richard Tsai, and Stefano Soatto. Information-seeking control under visibility-based uncertainty. *Journal of Mathematical Imaging and Vision*, 48(2):339–358, 2014.

[38] Brian Yamauchi. A frontier-based approach for autonomous exploration. In *Computational Intelligence in Robotics and Automation, 1997. CIRA ’97., Proceedings., 1997 IEEE International Symposium on*, pages 146–151. IEEE, 1997.

[39] Fumin Zhang, Alan O’Connor, Derek Luebke, and PS Krishnaprasad. Experimental study of curvature-based control laws for obstacle avoidance. In *Robotics and Automation, 2004. Proceedings. ICRA ’04. 2004 IEEE International Conference on*, volume 4, pages 3849–3854. IEEE, 2004.