2015 IEEE International Conference on Robotics and Automation (ICRA)
Washington State Convention Center
Seattle, Washington, May 26-30, 2015

Information-Theoretic Mapping Using Cauchy-Schwarz Quadratic Mutual Information

Benjamin Charrow, Sikang Liu, Vijay Kumar, Nathan Michael

Abstract— **We develop a computationally efficient control policy for active perception that incorporates explicit models of sensing and mobility to build 3D maps with ground and aerial robots. Like previous work, our policy maximizes an information-theoretic objective function between the discrete occupancy belief distribution (e.g., voxel grid) and future measurements that can be made by mobile sensors. However, our work is unique in three ways. First, we show that by using Cauchy-Schwarz Quadratic Mutual Information (CSQMI), we get significant gains in efficiency. Second, while most previous methods adopt a myopic, gradient-following approach that yields poor convergence properties, our algorithm searches over a set of paths and is less susceptible to local minima. In doing so, we explicitly incorporate models of sensors, and model the dependence (and independence) of measurements over multiple time steps in a path. Third, because we consider models of sensing and mobility, our method naturally applies to both ground and aerial vehicles. The paper describes the basic models, the problem formulation and the algorithm, and demonstrates applications via simulation and experimentation.**

I. Introduction

We develop computationally efficient active perception policies that encode platform mobility and sensing capabilities to minimize uncertainty in the probabilistic map representation of an uncertain environment. Noise, limited angle of view, and limited sensing range are common considerations for many robotics sensors. However, in applications such as exploration and mapping, these systemic limitations yield degraded algorithm performance as the robot also contends with limited mobility, precluding it from positioning its sensors arbitrarily throughout the environment. In this work, we seek a method that accounts for these limitations, enabling a robot to build and refine a map through continued control actions and sensor observations.

Active perception formulations of the map building problem seek to optimize an information-theoretic objective function such as Shannon's entropy or mutual information [1], [8]–[10], [20]. These objectives are appealing as they naturally combine exploration and refinement by estimating how positioning and orienting a sensor at a particular pose will reduce map uncertainty. This differs from frontier methods which rely on the maximum likelihood estimate of the map and are more geometric in nature [24]. Control policies which optimize information-theoretic objectives appear in several robotics applications. Most related to this work is the recent approach by Julian et al. [9] where a beam model is employed to calculate the mutual information between a laser range finder and a 2D occupancy grid. They show that a mutual information based control law will eventually cause a robot to drive towards unknown space. Stachniss et al. [20] maximize the entropy reduction of a Rao-Blackwellized particle filter to develop a 2D active SLAM system. Kretzschmar and Stachniss [11] use mutual information as a criterion for storing a minimal number of laser scans toward map reconstruction. Kollar and Roy [10] plan minimum entropy tours to explore a 2D environment, but use the maximum likelihood estimate of the map. Singh et al. [18] maximize mutual information for environmental monitoring. Ryan and Hedrick [15] enable a plane to track a moving target by minimizing the entropy of the target estimate over a finite horizon. Charrow et al. [2] approximate mutual information to address the computational burden incurred by a finite horizon policy formulation. Maddern et al. [12] propose Renyi's quadratic entropy (RQE) to calibrate laser range finders on an autonomous truck.

Most information-theoretic approaches are derived from Shannon's entropy, which provides a measure of the map's uncertainty. However, they are computationally expensive because they require a search over an infinite-dimensional space of control actions, each of which yields a continuous set of noisy observations. Consequently, most approaches employ a myopic greedy optimization using simple platform and sensor models, considering only limited statistics (e.g., expected values) with a maximum likelihood estimate of the map. In this work, we use the Cauchy-Schwarz quadratic mutual information (CSQMI) [13], which is theoretically well motivated since it is related to RQE, but can also be calculated efficiently in order to build and refine a 3D map in real time. Second, and equally importantly, we explicitly model the dependence of separate measurements over multiple time steps to develop online algorithms that search over multiple, multi-step actions enabling us to go beyond one step maximizations. By selecting informative paths, rather than greedily selecting informative destinations, robots choose better control actions that improve the quality of the map with greater efficacy. Finally, because we consider where robots can move, our method naturally applies to both ground and aerial vehicles. We evaluate the approach through a series of simulations and experiments in which mobile robots must build a 3D map using an RGB-D sensor. Our results show that maximizing CSQMI results in a robot exploring the entire environment, while simultaneously ensuring that the map is sufficiently certain.

B. Charrow, S. Liu and V. Kumar are with the GRASP Lab at the University of Pennsylvania, Philadelphia, PA 19104, USA. Email: {bcharrow,sikang,kumar}@seas.upenn.edu

N. Michael is with the Robotics Institute at Carnegie Mellon University Email: nmichael@cmu.edu

II. Occupancy Grid Mapping

We represent the environment as a 3D occupancy grid. Occupancy grids are advantageous in this setting because they are a volumetric and probabilistic representation of 3D space. The volumetric representation enables the robot to determine where it can move next, and knowing the probabilities that various regions are occupied enables it to reason about how its noisy sensors will change the representation.

The typical occupancy grid mapping algorithm discretizes 3D space into a set of regular cubes that we refer to as cells or voxels (volumetric pixels). The map, m, is made of cells $\{c_1, \ldots, c_{|m|}\}$ each of which have a probability of being occupied. Occupancy grid mapping makes three standard assumptions: 1) cells are independent of one another, so the probability of a particular map is $p(m) = \prod_i p(c_i)$, 2) the robot's position is known and 3) unobserved cells have a uniform prior of being occupied [22]. The probability that an individual cell c_i is occupied given sensor measurements from time 1 to t (i.e., $z_{1:t}$) is:

$$p(c_i \mid z_{1:t}) = \left(1 + \frac{1 - p(c_i \mid z_t)}{p(c_i \mid z_t)} \frac{1 - p(c_i \mid z_{1:t-1})}{p(c_i \mid z_{1:t-1})}\right)^{-1} \tag{1}$$

This expresses the probability that a cell is occupied in terms of the probability of the cell being occupied given just the latest measurement, $p(c_i \mid z_t)$, and all other measurements, $p(c_i \mid z_{1:t-1})$. $p(c_i \mid z_{1:t})$ can be efficiently calculated by expressing (1) in log-odds notation:

$$L(c_i \mid z_{1:t}) = L(c_i \mid z_{1:t-1}) + L(c_i \mid z_t) \tag{2}$$

Where $L(c_i \mid z_t)$ is an inverse measurement model [22].

As described, occupancy grid mapping does not estimate the pose of the robot. Consequently, we assume that the robot is capable of estimating its own pose as well as the pose of its sensor in its local frame. Although they are not the focus of this work, modern solutions to the SLAM problem, which model uncertainty in the robot's position as well as the map, can be used to estimate the robot's entire trajectory, enabling efficient construction of occupancy grids [21].

III. Cauchy-Schwarz Quadratic Mutual Information

We are interested in a control policy for the robot that maximizes the Cauchy-Schwarz quadratic mutual information (CSQMI) between the map and measurements the robot will receive over the next several time steps. This formulation gives a unified objective that drives map refinement and exploration, while accounting for limitations of the sensor (e.g., noise, field of view, maximum range) and the robot's own limited mobility.

A. The control policy

We define an action over a time interval $\boldsymbol{\tau} \triangleq t+1 : t+T$ as a discrete sequence of poses, $x_{\boldsymbol{\tau}} = [x_{t+1}, \ldots, x_{t+T}]$. For a given action, the robot will receive a random vector of measurements $z_{\boldsymbol{\tau}} = [z_{t+1}, \ldots, z_{t+T}]$ that each depend on the pose of the robot at that time. Using the distribution over the current state of the environment, $p(m)$, and its expected future locations, the robot can estimate the distribution over measurements, $p(z_{\boldsymbol{\tau}} \mid x_{\boldsymbol{\tau}})$. At each planning step, the robot generates a set of actions $\mathcal{X}$ (Sect. V), and selects the one that maximizes the rate of information gain:

$$x^*_{\boldsymbol{\tau}} = \arg\max_{x_{\boldsymbol{\tau}} \in \mathcal{X}} \frac{\mathrm{I_{CS}}[m; z_{\boldsymbol{\tau}} \mid x_{\boldsymbol{\tau}}]}{D(x_{\boldsymbol{\tau}})} \tag{3}$$

Where $\mathrm{I_{CS}}[m; z_{\boldsymbol{\tau}} \mid x_{\boldsymbol{\tau}}]$ is the CSQMI between the map and measurements the robot will receive if it follows action $x_{\boldsymbol{\tau}}$, and $D(x_{\boldsymbol{\tau}})$ is the time to execute the action. CSQMI can be expressed as:

$$\mathrm{I_{CS}}[m; z_{\boldsymbol{\tau}} \mid x_{\boldsymbol{\tau}}] = \tag{4}$$
$$-\log \frac{\left(\sum \int p(m, z_{\boldsymbol{\tau}} \mid x_{\boldsymbol{\tau}}) p(m) p(z_{\boldsymbol{\tau}} \mid x_{\boldsymbol{\tau}})\, dz_{\boldsymbol{\tau}}\right)^2}{\sum \int p^2(m, z_{\boldsymbol{\tau}} \mid x_{\boldsymbol{\tau}})\, dz_{\boldsymbol{\tau}} \sum \int p^2(m) p^2(z_{\boldsymbol{\tau}} \mid x_{\boldsymbol{\tau}})\, dz_{\boldsymbol{\tau}}}$$

where the sums are over all possible maps, and the integrals are over all possible measurements the robot can receive. In Sect. IV we give an algorithm that can approximately evaluate this quantity with a time complexity that is *linear* in the number of cells that potential measurements could hit.

CSQMI is an appropriate objective because it is a way of quantifying whether knowing the outcome of one random variable (e.g., a measurement at a location) will change the outcome of another random variable (e.g., the map at that location). Similar to other measures of information, the CSQMI between two random variables is a non-negative number and 0 if and only if the variables are independent.

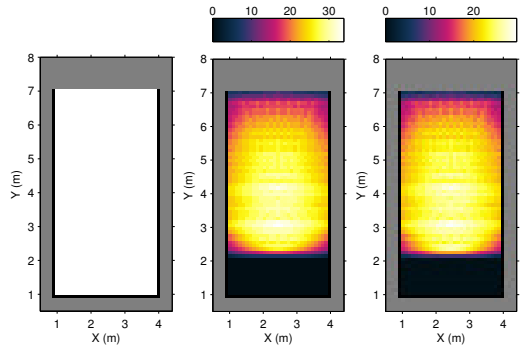
Incorporating the time to execute an action, $D(x_{\boldsymbol{\tau}})$ in (3), is useful as it enables actions with substantially different lengths to be meaningfully compared. Researchers have examined a variety of utility functions for combining information (reward) and time (cost) [20], [23]. We choose this objective as it can be intuitively understood as the rate of information gain.

B. Relationship to mutual information

A natural question is how CSQMI compares to mutual information (MI), a well understood and widely used objective function for active perception [9], [15]. We explain this relationship using a simple 2D environment when a robot is equipped with an omnidirectional 2D laser. The MI between the map and the robot's future measurements is:

$$\mathrm{I_{MI}}[m; z] = \mathrm{H}[z] - \mathrm{H}[z \mid m] \tag{5}$$

where $\mathrm{H}[z]$ is Shannon's entropy, $\mathrm{H}[z \mid m]$ is Shannon's conditional entropy [5], and both quantities are implicitly conditioned on the robot's pose. Fig. 1 shows the reward surfaces for MI and CSQMI. In Fig. 1a, white cells have a low probability of occupancy, black cells have a high probability of occupancy, and gray cells are uncertain. Color in Fig. 1b and Fig. 1c show the value of each objective if the robot places a laser with a minimum range of 0.5 m and a maximum range of 5 m at that position. Julian et al. [9, Fig. 1] use this example to illustrate why choosing MI is beneficial: its maximization would clearly result in the robot moving to



(a) Occupancy grid (b) MI (c) CSQMI

Fig. 1. For an omnidirectional 2D laser in (a), mutual information (MI) and CSQMI are similar and result in the robot moving to make observations that reduce the map's uncertainty.

regions where it could observe high uncertainty regions of the map. In Fig. 1c we see that maximizing CSQMI leads to similar behavior. While frontier-based methods would also perform well in this scenario, directly extending them to 3D environments is difficult as noted by Shen et al. [17] and demonstrated in our experiments (Sect. VI).

The similarity between Fig. 1b and Fig. 1c can be understood from a theoretical perspective. Both MI and CSQMI can be expressed in terms of a divergence measure between the joint distribution, $p(m, z)$ and the product of the marginals $p(m)p(z)$:

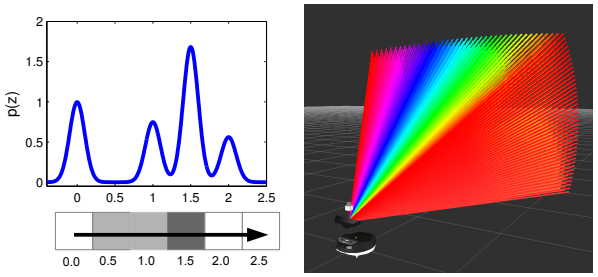
$$\begin{aligned} \mathrm{I}_{\mathrm{MI}}[m; z] &= \mathrm{D}_{\mathrm{KL}}[p(m, z) \mid\mid p(m)p(z)] \\ \mathrm{I}_{\mathrm{CS}}[m; z] &= \mathrm{D}_{\mathrm{CS}}[p(m, z) \mid\mid p(m)p(z)] \end{aligned}$$

where D_{KL} is the Kullback-Leibler divergence and D_{CS} is the Cauchy-Schwarz divergence [13], both of which measure the difference between distributions. Another similarity is that MI can be expressed in terms of cross-entropy and Shannon's entropy, whereas CSQMI can be expressed in terms of Renyi's quadratic cross-entropy and Renyi's quadratic entropy [14]. Both Shannon's and Renyi's definition of entropy share many properties, further explaining the similarity between the two types of information [13].

Importantly, all of the integrals to calculate CSQMI can be evaluated analytically in this setting, making it fast to evaluate, whereas calculating MI requires numerical integration [9]. Therefore, we choose CSQMI to derive our control policies.

IV. Calculating CSQMI

This section describes how to efficiently calculate the Cauchy-Schwarz quadratic mutual information (CSQMI) between the map and the robot's future measurements (i.e., $\mathrm{I}_{\mathrm{CS}}[m; z_{\boldsymbol{\tau}} \mid x_{\boldsymbol{\tau}}]$ in (3)). We first describe the measurement model, and then use that to calculate the CSQMI for a single beam at a single time step. We then extend this to CSQMI with multiple beams over several time steps.



(a) (b)

Fig. 2. Beam based measurement model. (a) one beam, the cells it intersects at various distances, and the resulting distribution over measurements when $z_{\min} = 0.5$ and $z_{\max} = 2$. (b) Raycasts for multiple beams.

A. Measurement Model

As discussed in Sect. III, at time k, the robot receives a random vector of measurements z_k. We model this vector as a collection of B beams so that $z_k = [z_k^1, \ldots, z_k^B]$, where z_k^b is the random variable of the distance that beam b travels. We model multiple beams using spherical coordinates by uniformly discretizing the horizontal and vertical field of view of the sensors and generating a beam for each combination of angles (e.g., Fig. 2b).

We assume the distribution over distances for a single beam is determined by the true distance, d, to the first obstacle that the beam intersects:

$$p(z_k^b = z \mid d) = \begin{cases} \mathcal{N}(z - 0, \sigma^2) & d \leq z_{\min} \\ \mathcal{N}(z - z_{\max}, \sigma^2) & d \geq z_{\max} \\ \mathcal{N}(z - d, \sigma^2) & \text{otherwise} \end{cases} \tag{6}$$

Where $z_{\min}$ and $z_{\max}$ are the minimum and maximum range of the sensor, and $\mathcal{N}(z - \mu, \sigma^2)$ is a Gaussian with mean μ and variance σ^2. This noise model is somewhat simplistic, as measurements from real sensors have non-axial distance dependent noise as well as radial distortion [19]. While (6) only accounts for axial noise and assumes a constant variance, Sect. VI shows that this is sufficient for good experimental results.

Given a map, m, we can calculate the distribution over measurements, $p(z_k^b)$, via marginalization. First, let $c \subseteq m$ be the list of C cells within the maximum range of the sensor that a beam intersects (Fig. 2a). Given the measurement model, (6), cells not in c have no effect on z_k^b, and can be ignored. The distance to the first occupied cell completely determines the distribution for each beam, so cells in c behind an occupied cell can be ignored. This means:

$$p(z_k^b) = \sum_{i=0}^{C} p(c = e_i) p(z_k^b \mid c = e_i) \tag{7}$$

Where e_i, $i > 0$, means that the i^{th} voxel in c is the first occupied cell the beam encounters and e_0 means that all cells in c are unoccupied, so the maximum range will be returned. See Julian et al. [9] for a similar derivation. Fig. 2a shows a typical distribution over measurements using (7).

B. CSQMI for a Single Beam

In this section we give the expression for the CSQMI between beam b at time k and the map. This is equivalent to calculating the CSQMI between the beam and cells the beam intersects because the beam cannot reduce the uncertainty of cells that it does not intersect (i.e., $\mathrm{I_{CS}}[m; z_k^b \mid x_k] = \mathrm{I_{CS}}[c; z_k^b \mid x_k]$). As we show in an extended version of this paper [3], $\mathrm{I_{CS}}[c; z_k^b \mid x_k]$ can be calculated by plugging the measurement distribution (7) into (4):

$$\begin{aligned}\mathrm{I_{CS}}[c; z_k^b \mid x_k] &= \log \sum_{\ell=0}^{C} w_\ell \mathcal{N}(0, 2\sigma^2) \\ &+ \log \prod_{i=1}^{C} (o_i^2 + (1-o_i)^2) \sum_{j=0}^{C} \sum_{\ell=0}^{C} p(e_j) p(e_\ell) \mathcal{N}(\mu_\ell - \mu_j, 2\sigma^2) \\ &- 2 \log \sum_{j=0}^{C} \sum_{\ell=0}^{C} p(e_j) w_\ell \mathcal{N}(\mu_\ell - \mu_j, 2\sigma^2) \end{aligned} \tag{8}$$

where $C = |c|$, $o_i = p(c_i = 1)$ (i.e., the i^{th} cell in c is occupied), and $p(e_j)$ is the probability that c^j is the first occupied cell. μ_j is determined by (6) and e_j. The w's are:

$$w_\ell = p^2(e_\ell) \prod_{j=\ell+1}^{C} (o_j^2 + (1-o_j)^2) \tag{9}$$

for $0 < \ell < C$ while $w_0 = p^2(e_0)$ and $w_C = p^2(e_C)$.

Because they can be calculated iteratively, computing $p(e_\ell)$ and w_ℓ for all ℓ takes $O(C)$ time. The double sums each have $C+1$ Gaussian components, so calculating them exactly takes $O(C^2)$ time.

The primary advantage of (8) is that all of the integration is performed analytically. Another nice property is that the double sums can be approximated in $O(C)$ time with practically no error. While general techniques like the Fast Gauss Transform [7] could also be used, this problem lends itself to a simpler approximation.

The approximation relies on the fact that Gaussians fall off quickly, and $\mathcal{N}(\mu_\ell - \mu_j, 2\sigma^2) \approx 0$ when $|\mu_\ell - \mu_j| > \beta\sqrt{2}\sigma$ (i.e., the means are more than a few standard deviations apart) and β specifies the degree of approximation. Determining which means are close to each other is normally difficult, but in this case we know the means monotonically increase with the index, because they are determined by their distance along the beam. Occupancy grid mapping typically uses cell lengths that are slightly larger than the variance of an individual beam, because finer resolutions will not result in significantly different maps. Thus, means separated by more than a few grid cells must be substantially different, so the double sums can be approximated as:

$$\sum_{j=0}^{C} \sum_{\ell=j-\Delta}^{j+\Delta} \alpha_{j,\ell} \mathcal{N}(\mu_\ell - \mu_j, 2\sigma^2) \tag{10}$$

where Δ is the number of cells beyond which the contribution is effectively 0 and $\alpha_{j,\ell}$ is a weight. As the $\alpha_{j,\ell}$ in (8) are small, Δ is typically 3 or 4 for $\beta = 3$, making the complexity of (10) $O(C)$, meaning CSQMI's complexity is linear in the number of cells the beam intersects.

C. CSQMI for Multiple Beams and Time Steps

We now discuss evaluating the CSQMI between all beams across all time steps: $\mathrm{I_{CS}}[m; z_{\boldsymbol{\tau}} \mid x_{\boldsymbol{\tau}}]$. Exact calculation is computationally intractable because different beams can observe the same cells. As we show, this problem arises form the measurement model and affects all information-theoretic policies, and not just those based on CSQMI. Researchers typically address this issue by assuming individual measurements are independent [9], [11]. We adopt a different approach by calculating CSQMI with a subset of the measurements that *are* nearly independent. This enables efficient computation, often yields better performance than the independence assumption (Sect. VI), and can be applied to other information based metrics like mutual information.

To simplify the discussion, consider two beams z^i and z^j which originate from different robot poses. For brevity, in the remainder of this subsection, we will often not explicitly condition measurements on the robot's position. Marginalizing over all cells in the map, their joint density is:

$$p(z^i, z^j) = \sum_o p(o) \sum_{c^i} p(c^i) p(z^i \mid c^i, o) \sum_{c^j} p(c^j) p(z^j \mid c^j, o) \tag{11}$$

where c^i is the set of cells that only beam i intersects, c^j is the set of cells that only beam j intersects, and o is the set of overlapping cells both beams intersect. When z^i and z^j have no cells in common they are independent because o is the empty set. CSQMI simplifies in this case: $\mathrm{I_{CS}}[m; z^i, z^j] = \mathrm{I_{CS}}[c^i; z^i] + \mathrm{I_{CS}}[c^j; z^j]$ (see the extended paper for the derivation [3]). Unfortunately, calculating CSQMI – or any other information metric – when the beams overlap requires summing over each instantiation of o in (11). There are $2^{|o|}$ such instantiations, and while the integrals can be done analytically, exact calculation is computationally infeasible.

However, even when the beams overlap, the probability that they both hit the same cell may be small. For example, at each time step all beams intersect cells close to the robot, but those cells usually have a low probability of occupancy, so none of the beams has a high probability of hitting them. Fig. 3 illustrates this point. In these cases, the joint density can be approximated as:

$$\begin{aligned} p(z^i, z^j) &\approx \sum_{c^i, o^i} p(c^i, o^i) p(z^i \mid c^i, o^i) \sum_{c^j, o^j} p(c^j, o^j) p(z^j \mid c^j, o^j) \\ &= \hat{p}(z^i) \hat{p}(z^j) \end{aligned} \tag{12}$$

which means z^i and z^j are approximately independent. Here $o^i \subseteq o$ is the subset of overlapping cells that only z^i is likely to hit and $o^j \subseteq o$ is the subset of overlapping cells that only z^j is likely to hit.

To establish (12), consider the case when the two beams are unlikely to collide (i.e., don't hit the same cell). Let $h_{z^i}^{o_\ell}$ be the event that z^i hits o_ℓ, the ℓ^{th} cell in o. If z^i is likely to hit o_ℓ (i.e., $p(h_{z^i}^{o_\ell}) \neq 0$), then beam z_j is not (i.e., $p(h_{z^j}^{o_\ell}) \approx$

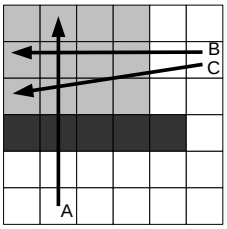


Fig. 3. Nearly independent beams. Beam A intersects beams B and C, but it is nearly independent of them because the probability it reaches the same cells is close to 0. Beams B and C are not nearly independent, because they will likely hit the same cell.

0). For beam z^j, this means $p(z^j \mid c^j, o_\ell) \approx p(z^j \mid c^j)$. Consequently, $p(o_\ell)$ can be factored out of the outermost sum in (11) and included in the sum over c^i. This same argument works when $p(h^{o_\ell}_{z^j})$ is non-negligible. If neither beam is likely to reach o_ℓ, it is effectively marginalized out by the outermost sum. Repeating this process for each cell in o results in (12), which means $\mathrm{I_{CS}}[m; z^i, z^j] \approx \mathrm{I_{CS}}[c^i, o^i; z^j] + \mathrm{I_{CS}}[c^j, o^j; z^j]$

This insight provides a way of approximating CSQMI using a subset of measurements, $\mathcal{Z}$, that are nearly independent:

$$\mathrm{I_{CS}}[m; z_{\boldsymbol{\tau}} \mid x_{\boldsymbol{\tau}}] \approx \sum_{z^b_k \in \mathcal{Z}} \mathrm{I_{CS}}[m; z^b_k \mid x_k] \tag{13}$$

While the sum is an approximation, it tends to lower bound the true CSQMI as $\mathcal{Z}$ does not contain all measurements.

Finding $\mathcal{Z} \subseteq z_{\boldsymbol{\tau}}$ that maximizes (13) is an instance of the maximum-weight independent set problem, meaning it is NP-hard and no polynomial time constant factor approximation algorithm exists [4]. Despite this, we have obtained good results by iterating over all measurements and greedily building $\mathcal{Z}$ (Alg. 1). To test if a new beam z^b_k is independent of those in $\mathcal{Z}$, we check if beams in $\mathcal{Z}$ do not hit any cell that z^b_k hits (Lines 8-11). To check if a collision is unlikely at cell c_i we use the following test:

$$p\left(h^{c_i}_{z^b_k}, \bigcup_{z \in \mathcal{Z}} h^{c_i}_z\right) \leq \min\left\{p(h^{c_i}_{z^b_k}), \sum_{z \in \mathcal{Z}} p(h^{c_i}_z)\right\} \leq \gamma \tag{14}$$

where γ is a user specified cutoff. If this is true for each cell that z^b_k, intersects, we treat it as independent and add it to $\mathcal{Z}$ (Line 13). The inequality in (14) uses the monotonicity of probability and the union bound to upper bound the probability of collision in a way that does not require determining interactions between beams (Lines 14-15).

The computational complexity of Alg. 1 is $O(TBC)$, where T is the number of poses in the action and B is the number of beams the sensor has. The outer loops iterate over all TB beams. For each beam, a raycast is performed and the weights are built, which as described in Sect. IV-B is linear, $O(C)$, in the number of cells the beam intersects. Both the independence check and updating AnyHit iterate over each cell, performing constant time operations at each one.

V. Action Generation

The primary goal of action generation is to create paths whose rate of information gain is large. To do this, we plan shortest paths to places where frontiers can be observed.

First, we identify frontier voxels (i.e., voxels that are unobserved but neighbor observed voxels) [24]. In 3D environments there are too many voxels to generate a path for each one, so we group them into "clusters" with a greedy algorithm. Each cluster is created by sampling a frontier voxel and grouping it with other voxels within a specified distance (0.4 m works well in practice). This process repeats until no frontier voxels remain.

The robot generates one action per cluster by planning a path to where the cluster is visible. A cluster is "visible" from a pose if it is in the robot's sensor footprint and the probability that a ray from the sensor reaches the cluster is high (i.e., the ray is unobstructed). Planning these paths is difficult for two reasons. First, in 3D environments clusters may exist in places that a robot can observe, but cannot go to (e.g., space above obstacles). Second, a robot must consider its orientation when its sensor has a limited angle of view.

A straightforward approach to generate a path to each cluster is to use Dijkstra's algorithm to plan shortest paths from the robot's current pose to every destination in the environment and check to see which clusters are visible from each destination. However, this is computationally expensive because it requires performing a visibility check to all clusters at every destination. To speed the process up, we precompute which clusters are visible from different destinations using Alg 2. samplePosesWhereVisible(cluster) (Line 2) returns a set of poses from where a robot can view a cluster; it can be implemented by sampling poses near the

Algorithm 1. Calculate $\mathrm{I_{CS}}[m; z_{\boldsymbol{\tau}} \mid x_{\boldsymbol{\tau}}]$

```
1:  CSQMI = 0 // CSQMI from measurements in Z
2:  AnyHit = 0 // Mapping from each cell to probability
    that any measurement in Z hits it; initially all 0
3:  for k = t + 1 to t + T do
4:    for b = 1 to B do
5:      c = Raycast to get C cells that z^b_k intersects
6:      Calculate [p(e_0), ..., p(e_C)] // i^th element is the
        probability that z^b_k hits cell c_i
7:      // See if z^b_k is independent of measurements in Z
8:      independent = true
9:      for i = 1 to C do
10:       if min{p(e_i), AnyHit[c_i]} > γ then
11:         independent = false
12:     if independent then
13:       CSQMI += I_CS[m; z^b_k | x_k] // Add z^b_k's info
14:       for i = 1 to C do
15:         AnyHit[c_i] += p(e_i)
```

Algorithm 2. Calculate VisibilityLookupTable, a mapping from robot poses to the clusters it can view at that pose

```
1: for each cluster in clusters do
2:   poses = samplePosesWhereVisible(cluster)
3:   for each pose in poses do
4:     VisibilityLookupTable.at(pose).update(cluster)
```

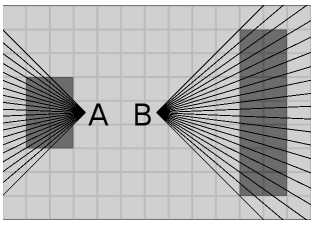


Fig. 4. Independence leads to overconfidence. Positioning a sensor at B results in a larger CSQMI than at A. The approximation from Sect. IV-C picks pose B, but assuming measurements are independent picks A.

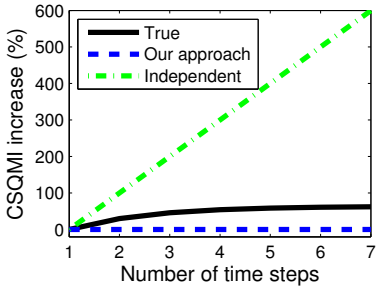


Fig. 5. Percentage increase in CSQMI for a single beam observing the same cells over multiple time steps. The approximation from Sect. IV-C is more conservative, but also more accurate.

cluster and rejecting those that cannot view it [6]. Using this table during the Dijkstra search is substantially faster than performing the visibility checks directly.

VI. Results

A. Approximating CSQMI With and Without Independence

We examine the result of our approximate evaluation of the objective (13), and compare it to the values we get when we assume measurements are independent. Consider a simple example where a robot with a planar 2D laser must evaluate the CSQMI at two different positions as shown in Fig. 4. The laser has 30 beams uniformly spaced over its 90° field of view. In this case, calculating CSQMI by assuming measurements are independent results in an overestimate of the true CSQMI, and chooses pose A. Our approach lower bounds the true value and picks pose B, which maximizes the true CSQMI.

Because we represent the map as an occupancy grid, the independence assumption may appear more reasonable as a sufficiently fine discretization of space will result in beams that are independent at a particular point in time. However, extending this to multiple time steps is problematic. In particular, with the independence assumption the optimal policy will be to position the sensor at the place that maximizes CSQMI at a single time step, and stays there for all remaining steps. Fig. 5 shows this effect in more detail by calculating the CSQMI of a single beam that intersects the same 10 cells multiple times. Each cell is occupied with probability 0.5. Assuming independence results in a linear growth in CSQMI, while the true CSQMI asymptotically reaches a 63% increase. Calculating CSQMI using Alg. 1 only uses the beam at the first time step, as all subsequent beams

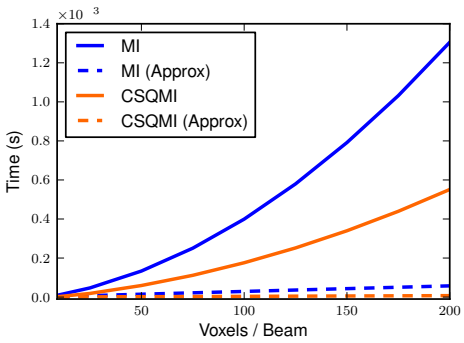


Fig. 7. Time to evaluate the information of a single beam. CSQMI is faster to compute than MI and the approximation from (10) makes computing information linear in the number of voxels.

are dependent on it, which lower bounds the true value. In general, this approach is preferable as maximizing a function by maximizing a lower bound results in better performance than maximizing an upper bound.

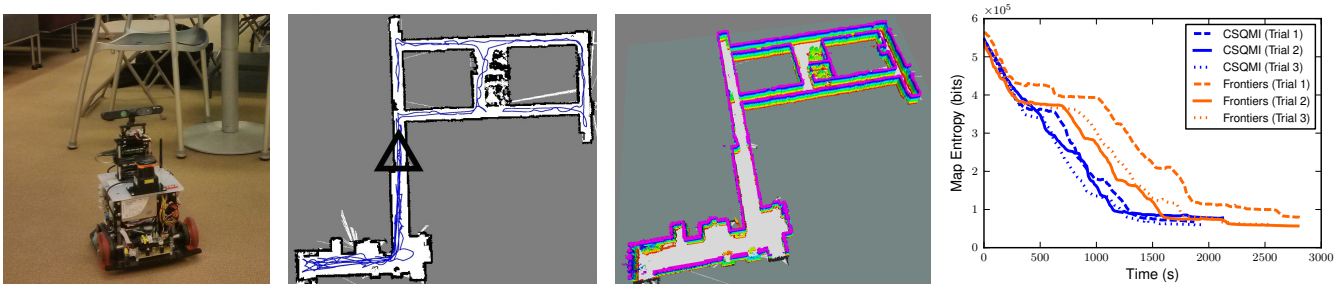
B. Computational Performance

In this section, we benchmark the computational performance of CSQMI by comparing it to mutual information (MI). To do this, we examine the time it takes to evaluate the information of a single beam and a group of voxels over a large number of random trials. Voxels are spaced 0.1 m apart, their probability of occupancy is randomly generated, the beam's noise is $\sigma = 0.03$ m and the distance to the last voxel is treated as the maximum range of the sensor. MI must be computed numerically using (5). We also compare calculations that exploit the rapid decay of a Gaussian, as we did in (10), by ignoring cross-terms whose means are separated by more than 4 standard deviations. Fig. 7 shows the time to compute information with varying numbers of voxels (mean time using 10^5 samples). Using the approximation makes the asymptotic complexity of calculating MI and CSQMI linear instead of quadratic. However, CSQMI is faster in both cases and approximately 7 times faster when using the approximation. This difference is substantial, as a robot often needs to evaluate tens to hundreds of actions during exploration, each of which consists of multiple poses which each have hundreds of beams. Finally, note that this approximation introduced little error; the relative error was below 10^{-3} for all trials.

C. Experimental Results with a Ground Robot

To test our approach experimentally, we use a ground robot to map a 40 m $\times 35$ m $\times 2$ m portion of an office environment in Levine Hall at the University of Pennsylvania. A video of experimental results is available at http://youtu.be/R3CW7VmkjFk.

The ground robot (Fig. 6a) is equipped with a UTM-30LX laser range finder and an Asus Xtion Pro as its RGB-D sensor. The UTM-30LX is mounted parallel to the ground, which the robot uses to estimate odometry and to construct a 2D map using a pose graph based SLAM system [22].



(a) Ground robot (b) 2D map with followed path (c) Final 3D map (d) Rate of entropy decrease

Fig. 6. Ground robot results using CSQMI. (b) The black triangle shows the robot's starting location and the blue line shows the path it followed for one trial. (c) The final 3D map from one trial (ceiling omitted) with occupied cells colored by height. (d) Shannon's entropy of 3D map over time for multiple trials and approaches. Using CSQMI to evaluate the utility of actions helps the robot reduce the map's uncertainty faster.

The 3D occupancy grid is constructed with a resolution of 0.1 m using data from the Xtion and pose information from the 2D SLAM system. The grid is updated incrementally as the robot moves, and regenerated whenever previous pose estimates of the robot change significantly (e.g., due to loop closures). The robot's maximum linear velocity is 0.5 m/ s.

Evaluating CSQMI requires specifying several parameters. To predict future measurements we discretize the $58°$ horizontal field of view and $48°$ vertical field of view of the RGB-D sensor into 20 separate values each, resulting in 400 separate beams. With the 0.1 m resolution of the map, finer angular discretizations typically result in dependent beams that do not substantially change the objective (Sect. IV-C). The sensor's minimum range is $z_{\text{min}} = 0.5$ m, its maximum range is $z_{\text{max}} = 4.5$ m, and its noise is $\sigma = 0.03$ m. To classify measurements as independent, we use a cutoff of $\gamma = 0.1$. When evaluating the CSQMI of an action, we include a pose every 3 m along the path or when the robot's heading changes by more than $50°$. All software is written in C++ and executes on the robot which has an Intel Core i5 processor with 8 GB RAM.

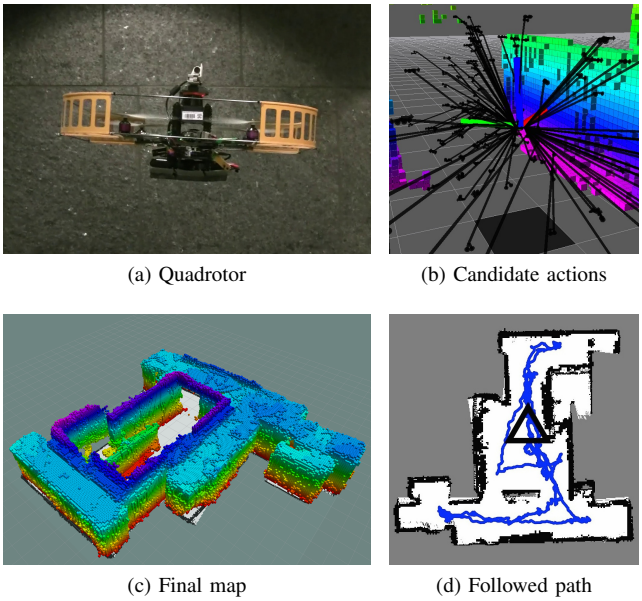
Fig. 6 shows typical results from maximizing CSQMI. Fig. 6b shows a top down view of the path the robot followed overlaid on its 2D laser based map and Fig. 6c shows the maximum likelihood estimate of the 3D map at the end of the trial. The final map had low uncertainty for every area that the robot could observe. Overall, the robot's followed path was efficient, with relatively little time spent revisiting areas that were already well observed. While the robot did revisit sections of the environment, particularly in the lower left corner, these actions occur near the end of the trial, after the robot has already observed most of the environment. The accompanying video submission illustrates the robot's overall behavior more clearly.

To determine whether maximizing CSQMI yields improvements in performance over other methods, we compare it to the common approach of driving to the nearest frontier. To implement frontier-based exploration, we use the same action generation approach, but select the action that takes the smallest amount of time to execute without evaluating CSQMI. The CSQMI approach stops executing actions once the highest information gain is below 30 (a low value for the sensor setup). The frontier-based approach terminates once no frontiers are detected. When comparing these strategies, we are interested in how quickly the robot obtains a low uncertainty estimate of the entire map.

To measure how the map's uncertainty changes, we look at the Shannon entropy of cells the robot could possibly observe (i.e., $\mathrm{H}[m] = -\sum_i o_i \log_2 o_i + (1-o_i)\log_2(1-o_i)$ where o_i is the probability that the i^{th} cell is occupied). We sum over cells in the 3D map whose x and y coordinate are within 0.1 m of an occupied or unoccupied cell in the final 2D map. This excludes most cells that are impossible to observe (e.g., in between hallways) and – because the laser based 2D map is complete – includes cells that the robot may have failed to observe with its RGB-D sensor. Fig. 6d shows the entropy over time for 3 trials for both approaches. For reference, there were approximately 6×10^5 observable cells and the maximum entropy of a single cell is 1 bit (i.e., an occupancy probability of 0.5). Overall, CSQMI significantly outperformed the nearest frontier approach. While the final maps had similar entropies, the rate of decrease using CSQMI was faster. The mean time to completion for CSQMI was 1945 s while for nearest frontier it was 2720 s. The primary reason for this difference is that in 3D environments there are often a large number of frontiers. However, these can be spurious in that the surrounding environment may already be well observed. Evaluating the CSQMI of actions enables the robot to determine whether traveling to a frontier will improve the map, while the frontier-based approach spends a lot of time traveling to areas that have already been well explored. The long flat tail of the entropy curve for the CSQMI trials indicates that the information termination criterion was appropriately conservative.

D. Experimental Results with an Aerial Robot

To examine how our approach works with robots that have greater mobility, we conducted experiments with a quadrotor in a 18 m $\times$ 17 m $\times$ 3 m section of Skirkanich Hall at the University of Pennsylvania. Unlike a ground robot, a quadrotor can change its height, enabling it to travel to more sections of the environment.



(a) Quadrotor (b) Candidate actions

(c) Final map (d) Followed path

Fig. 8. Quadrotor results. By evaluating CSQMI, the quadrotor is able to build a complete map with low uncertainty.

We use an AscTec Pelican (Fig. 8a) equipped with a UTM-30LX and Asus Xtion Pro. It performs laser-based localization onboard using the system by Shen et al. [16]. The state estimates are fed into the same SLAM and CSQMI framework used by the ground robot, which run on a wirelessly connected laptop.

Fig. 8 shows results from a single trial which took 11 min to complete. Due to its limited battery life, approximately 5 min, we manually landed the quadrotor twice to replace its battery. Obstacles (e.g., tables) were present throughout the environment, which the quadrotor accounted for when generating actions (Fig. 8b). Despite these challenges, Figs. 8c and 8d shows that it built a complete map, demonstrating that CSQMI generalizes to robots with more flexible motions.

VII. Conclusion

We presented a general approach and algorithms to plan multi-step actions for active perception that lend themselves to robots equipped with high data rate sensors like RGB-D cameras. The use of the CSQMI and reasoning about the independence and dependence of observations over successive time steps allowed us to develop algorithms that are both accurate and efficient. The paper included several experimental and simulation results that illustrate the ability of robots to explore and map three-dimensional environments, even with mobility constraints that may constrain them to a plane. Our current work addresses cooperative control of aerial and ground robots.

Acknowledgments

This work was supported in part by AFOSR grant FA9550-10-1-0567, ARL grant W911NF-08-2-0004, ONR grants N00014-07-1-0829 and N00014-09-1-1051, NSF grant IIS-1426840, and TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA.

References

[1] F. Bourgault, A. Makarenko, S. Williams, B. Grocholsky, and H. Durrant-Whyte. Information based adaptive robotic exploration. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, 2002.

[2] B. Charrow, V. Kumar, and N. Michael. Approximate representations for multi-robot control policies that maximize mutual information. In *Proc. of Robot.: Sci. and Syst.*, Berlin, Germany, June 2013.

[3] B. Charrow, S. Liu, V. Kumar, and N. Michael. Information-theoretic mapping using cauchy-schwarz quadratic mutual information. Technical Report MS-CIS-15-02, University of Pennsylvania, 2015.

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 3rd edition, 2009.

[5] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley Online Library, 2004.

[6] B. Englot and F. Hover. Three-dimensional coverage planning for an underwater inspection robot. *Int. J. Robot. Research*, 32(9-10):1048–1073, 2013.

[7] L. Greengard and J. Strain. The fast gauss transform. *SIAM Journal on Scientific and Statistical Computing*, 12(1):79–94, 1991.

[8] G. Hoffmann and C. Tomlin. Mobile sensor network control using mutual information methods and particle filters. *IEEE Trans. Autom. Control*, 55(1):32–47, 2010.

[9] B. J. Julian, S. Karaman, and D. Rus. On mutual information-based control of range sensing robots for mapping applications. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pages 5156–5163, Tokyo, Japan, November 2013.

[10] T. Kollar and N. Roy. Efficient optimization of information-theoretic exploration in SLAM. In *AAAI*, volume 8, pages 1369–1375, 2008.

[11] H. Kretzschmar and C. Stachniss. Information-theoretic compression of pose graphs for laser-based SLAM. *Int. J. Robot. Research*, 31(11):1219–1230, August 2012.

[12] W. Maddern, A. Harrison, and P. Newman. Lost in translation (and rotation): Rapid extrinsic calibration for 2d and 3d LIDARs. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 3096–3102, May 2012.

[13] J. C. Principe. *Information Theoretic Learning: Renyi's Entropy and Kernel Perspectives*. Springer, New York, August 2010.

[14] S. Rao. *Unsupervised Learning: An Information Theoretic Framework*. PhD thesis, University of Florida, 2008.

[15] A. Ryan and J. K. Hedrick. Particle filter based information-theoretic active sensing. *Robot. Auton. Syst.*, 58(5):574–584, May 2010.

[16] S. Shen, N. Michael, and V. Kumar. Autonomous multi-floor indoor navigation with a computationally constrained mav. In *Robotics and automation (ICRA), 2011 IEEE international conference on*, pages 20–25. IEEE, 2011.

[17] S. Shen, N. Michael, and V. Kumar. Stochastic differential equation-based exploration algorithm for autonomous indoor 3d exploration with a micro-aerial vehicle. *Int. J. Robot. Research*, 31(12):1431–1444, November 2012.

[18] A. Singh, F. Ramos, H. Durrant-Whyte, and W. Kaiser. Modeling and decision making in spatio-temporal processes for environmental surveillance. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 5490–5497, 2010.

[19] J. Smisek, M. Jancosek, and T. Pajdla. 3d with kinect. In *IEEE International Conference on Computer Vision Workshops*, pages 1154–1160, November 2011.

[20] C. Stachniss, G. Grisetti, and W. Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *Proc. of Robotics: Science and Systems (RSS)*, Cambridge, MA, USA, 2005.

[21] Johannes Strom and Edwin Olson. Occupancy grid rasterization in large environments for teams of robots. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pages 4271–4276, 2011.

[22] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, 2008.

[23] A. Visser and B. Slamet. Balancing the information gain against the movement cost for multi-robot frontier exploration. In *European Robotics Symposium*, pages 43–52, 2008.

[24] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 146–151, July 1997.