Exploration and Prediction: Beyond-the-Frontier Autonomous Exploration in Indoor Environments

LUDVIG ERICSON

Doctoral Thesis
Stockholm, Sweden, 2025

Division of Robotics, Perception and Learning
School of Electrical Engineering and Computer Science
KTH Royal Institute of Technology
SE-100 44 Stockholm, Sweden

TRITA-EECS-AVL-2025:47
ISBN 978-91-8106-270-0

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig granskning för avläggande av Teknologie doktorexamen i datavetenskap 27 maj 2025 klockan 09:00 i Kollegiesalen, Brinellvägen 6, Kungliga Tekniska Högskolan, Stockholm.

Tryck: Universitetsservice US AB

The struggle itself towards the heights is enough to fill a man's heart. One must imagine Sisyphus happy.

—A. Camus, "The Myth of Sisyphus"

Abstract

Autonomous exploration is a fundamental problem in robotics, where a robot must make decisions about how to navigate and map an unknown environment. While humans rely on prior experience and structural expectations to act under uncertainty, robotic systems typically operate without such priors, exploring reactively based only on what has been observed. The idea of incorporating predictions into exploration has been proposed previously, but the tools required to learn general, high-capacity models have only recently become available through advances in deep learning. This thesis addresses two tightly connected challenges: learning predictive models of indoor environments, and constructing exploration strategies that are able to benefit from such predictions. A core obstacle in this research area is a cyclic dependency: there is little value in developing better predictive models unless exploration methods can make effective use of them, and little value in designing such exploration methods unless reliable models exist. This dependency has historically limited progress. By breaking it, this thesis enables the study and development of both components in tandem. The thesis introduces deep generative models that capture structural regularities in indoor environments using autoregressive sequence modeling. These models outperform traditional approaches in predicting unseen regions beyond the robot's current observations. However, standard exploration methods are shown to perform worse, not better, when informed by accurate predictions. To resolve this, new planning heuristics are proposed, including the distance advantage strategy, which prioritizes exploring regions that are likely to be more difficult to reach in the future. These methods allow predictive models to be used effectively, reducing path length by avoiding situations where the robot must backtrack to previously visited locations. Together, these contributions provide a foundation for autonomous exploration that is informed by learned expectations, and establish a framework where map-predictive modeling and decision-making can be studied and improved jointly.

Sammanfattning

Autonom utforskning är ett grundläggande problem inom robotik, där en robot måste fatta beslut om hur den ska navigera och kartlägga en okänd miljö. Medan människor förlitar sig på tidigare erfarenheter och strukturella förväntningar för att agera under osäkerhet, arbetar robotsystem vanligtvis utan sådana s.k. priors och utforskar reaktivt, enbart baserat på vad som har observerats. Idén att införliva prediktioner i utforskning har föreslagits tidigare, men verktygen som krävs för att lära sig generella, modeller med hög kapacitet har först nyligen blivit tillgängliga genom framsteg inom djupinlärning. Denna avhandling behandlar två nära sammanlänkade utmaningar: att lära sig prediktiva modeller av inomhusmiljöer, och att konstruera utforskningsstrategier som kan dra nytta av sådana prediktioner. Ett centralt hinder inom detta forskningsområde är ett cykliskt beroende: det finns litet värde i att utveckla bättre prediktiva modeller om inte utforskningsmetoder effektivt kan utnyttja dem, och vice versa finns det litet värde i att utforma sådana utforskningsmetoder om inte tillförlitliga modeller finns. Detta beroende har historiskt sett begränsat framsteg. Genom att bryta detta beroende möjliggör denna avhandling parallell utveckling och analys av båda komponenterna. Avhandlingen introducerar djupa generativa modeller som fångar strukturella regelbundenheter i inomhusmiljöer med hjälp av autoregressiv sekvensmodellering. Dessa modeller överträffar traditionella metoder i att förutsäga osedda områden bortom robotens nuvarande observationer. Det visar sig dock att standardmetoder för utforskning presterar sämre, inte bättre, när de informeras av exakta prediktioner. För att lösa detta föreslås nya planeringsheuristiker, inklusive distance advantage-strategin, som prioriterar att utforska områden som sannolikt kommer vara svårare att nå i framtiden. Dessa metoder möjliggör ett effektivt utnyttjande av prediktiva modeller, vilket minskar färdvägens längd genom att undvika situationer där robeten behöver backa tillbaka till tidigare besökta platser, s.k. backtracking. Tillsammans utgör dessa bidrag en grund för autonom utforskning som är informerad av inlärda förväntningar, och etablerar ett ramverk där kartprediktion och beslutsfattande kan studeras och förbättras i samspel.

Preface

The journey to a doctoral degree is a miniature version of the journey through life itself, the hero's journey: to become something greater, we must let go of our old selves. For the doctoral student, every paper is itself a little death. The ideas we believe in and explore exist inside us; they are the thoughts we live and breathe. For one set of ideas to metamorphose into the next, the ego must die and be reborn, making way for something new. It is a painful process, both for grieving the death of what was and for the existential crisis that follows as we reinvent ourselves. But this transformation is not only our own.

No doctoral student works in isolation, we are shaped by those who came before us through our supervisors. Their intellectual legacies, their ambitions, their unrealized projects—these form the invisible scaffolding of our own work. The pressures we feel are not just our own; they are, in part, the weight of what remains undone in the minds of those who guide us. As Jung wrote, *"The greatest burden a child must bear is the unlived life of the parents."* Likewise, the greatest burden a doctoral student must bear is the unlived intellectual life of their supervisor—the lingering research questions of previous students, their unfinished theories, the paths they could not take, now inherited by us. This transfer happens even if, as in my case, the supervisor tries to prevent it. The student, in their crisis of identity, searches for something to cling to, to build their identity from.

A wise man once said that the result of the doctoral program is the person, not the thesis. I always thought this was only superficially true, that the real value is the science done, and that lives in the papers and the results, objective and undeniable. Now that I have written my own thesis I think I see the point. Words on pages are merely evidence of the thoughts and ideas of the people who wrote those words. Sadly, the ideas of the student therefore often wither and die when the student graduates. Be that as it may, I am incredibly grateful for having been given this opportunity to study a subject so deeply. It is an amazing privilege.

This document is a *compilation thesis*, consisting of two parts: the *overview* and the *included papers*. The overview is self-sufficient, a standalone work that comprehensively summarizes the included papers and relates them and their results to the larger field in which they belong. Nevertheless, the papers themselves are part of the thesis, and to understand the thesis, one must understand the papers too.

Acknowledgements

I would like to thank my supervisor Patric for giving me the opportunity to pursue this degree project, for his confidence in me over the years, and for being a mentor to me in all of life's assorted challenges. Your humility, patience, and dedication is truly inspiring. To the wonderful people of my research group, you have been instrumental in keeping the fire burning and coping with the absurdity of it all, I feel blessed to have had your support. 618 truly is not a place, but a state of mind.

To the rock climbers and pebble wrestlers in my life, who knew you could deal with existential crises by climbing to the top of somewhat high rock faces. To the Gotland crew, what an unforgettable time we had, I'll always remember it. Thanks to all the talented, curious, wonderful people at RPL, and to RPL itself. I am honored to have shared in your lives, and I will miss you all incredibly. Thank you to all the students that I had the privilege of teaching throughout the years, nothing is as exciting to me as the vicarious rush of seeing big ideas click into place for the first time.

Tack till alla de som stöttat mig genom åren på KTH, jag kommer aldrig glömma tiden i korridorsboendet på Lappkärrsberget, en livstid utspelade sig där på *Big Brother: Lappis*. Tack till mina språkpartners genom åren, tyvärr blev jag ingen kinesisk poet, men det var värt ett försök.

Ett särskilt tack till David, Isac, Johan, och Jonas, ni har lärt mig så mycket genom de fantastiska personer ni är. Ni besitter den ovanliga men ovärderliga förmågan att värdera och stå upp för er själva och andra, det är sann styrka, och jag inspireras av er varje dag.

Till mina bröder, Ossian och Alexander, jag minns när vi spelade tv-spel tillsammans i vardagsrummet, den slitna röda lädersoffan med insydda knappar, och hur vi fick turas om på familjedatorn som stod på ett alldeles för litet bord, ni har stöpt mig till den jag är idag. Tack mamma, för att du alltid trott på mig och funnits där när det behövts, jag kommer alltid minnas när du höll Felicia i famnen för första gången, då cirkeln slöts och son blev far.

我要感謝斯斯的父母，沒有你們的幫助，我就不能寫這份文件。Jag önskar att jag en dag kan ta lika väl hand om mina barnbarn som ni gör Felicia. Ert stöd har varit ovärderligt.

Framförallt vill jag tacka min familj, Sisi och Felicia. Jag älskar er mer än ord kan beskriva, och ni har gjort mig rikare än pengar någonsin kan. Ingenting gör mig lyckligare än er, och jag är så glad att få dela mitt liv med er.

Ludvig Ericson
March 2025
Stockholm

Contents

Part I

Overview

Chapter 1

Introduction

Humans regularly make decisions despite incomplete knowledge. For example, when Alice is planning to visit Bob tomorrow for dinner, she might not know every detail—what to wear, what route to take, or where to park—but past experience and reasonable assumptions help her navigate these uncertainties successfully. This everyday ability to handle unknowns contrasts sharply with how robots plan their actions, which often requires complete, detailed knowledge of the environment. The first step in practically all robotics tasks is to build a map of the environment which then serves as the single source of truth for the computer program that solves the actual task. This map is constructed from a blank slate, without knowledge of how maps or environments usually look, and without the ability to interpret cues in the environment.

This shortcoming is particularly apparent in the robotics task of *autonomous exploration*, where the goal is for a robot to explore an environment in service of some higher-level goal, e.g., to map it out for downstream tasks, such as when surveying construction sites [1], or to find something, as in search-and-rescue missions in disaster zones [2, 3]. At each moment, the robot decides where next to move by assessing the merit, or *value*, of moving to each of the candidate *frontiers*. Frontiers are places on the boundary of what is known to the robot, and moving towards a frontier pushes the boundary away, increasing the coverage of the environment. The value of a frontier can, in principle, only be determined by computing the optimal tour through the environment, but doing so requires already knowing the environment. This presents a paradox:

> To decide where to explore, the robot must already know what it will find.

In other words, the robot must have *some* expectations about the unknown to make a decision at all, much like Alice. A smarter robot, one that has more accurate expectations, should therefore be better able to evaluate the frontiers, and

consequently better execute its task; just as Alice would have made a better choice if Bob had told her that there is no parking near his house.

Indoor environments are highly structured and *quasi-regular*, and a number of basic assumptions can be made—for example: walls are flat and meet at right angles; rooms have at least one human-size door; there can be no rooms outside of the building; and so on. None of these propositions are strictly true, yet they are rarely false either. This quasi-regularity is also found in human language, and it has been famously difficult to construct logic-based computer programs that understand and produce language. Likewise, to date, no programs have shown an understanding of architecture or interior design. Instead, a model must be learned from examples and experience rather than programmed explicitly, and it is only recently that the tools necessary to create such implicit models have become available with the advent of deep learning and large language models. This is the topic of Chapter 3 of this thesis: to model indoor environments using the tools of language models (Paper [A]), and to predict *beyond the frontier* for autonomous exploration (Paper [B]).

More accurate information only improves performance if the robot can use that information effectively, which is not a foregone conclusion. Even if the robot had access to an oracle that told it exactly how the environment will look, computing the optimal exploration tour is computationally unfeasible even in trivial cases due to the extremely large search space, equivalent to the well-known computer science problem the *traveling salesman problem*. Instead, the value of a frontier is approximated as the reward of going there, e.g., how much new information does the robot expect to gain, minus the cost of going there, e.g., how far does it need to travel.

This approximation does not yield better results with better predictions; instead, it has the opposite effect, causing the robot to make worse decisions. To understand why, consider the *corridor-closet gedankenexperiment*, illustrated in Fig. 1.1: the robot is moving along a corridor, and as it comes across a supply closet, it decides between exploring the supply closet, or continuing further down the corridor. The supply closet can only be accessed from this one place and should be explored first; otherwise the robot will eventually have to go back to this place again. If the robot assumes unknown space is simply empty, it will overestimate the reward of the closet and may correctly decide to explore it first. An oracle, on the other hand, would reveal that the closet contains less new information, making the robot erroneously choose the corridor. This is the topic of Chapter 4: how should an algorithm for autonomous exploration be constructed so that it can make use of predictions. Paper [C] first identifies the issue and shows that it is possible to do better with predictions, and Paper [D] presents a practical algorithm that not only is able to use predictions to inform its choices, but performs better than existing exploration methods *without* predictions.

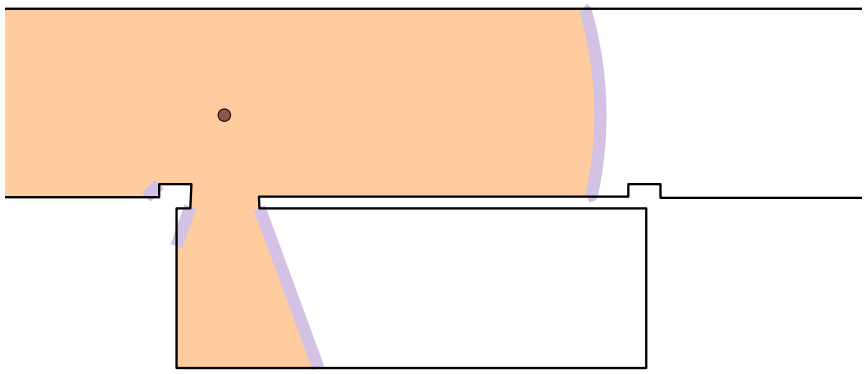


Figure 1.1: The corridor-closet problem is a demonstration of where unnecessary backtracking can occur, i.e., going back over already-explored regions to reach an unexplored region. The robot (brown circle) is facing a choice of either exploring the supply closet below it, or continuing down the corridor to the right. If the robot does not explore the supply closet below as it passes it by, the robot will be forced to go back to this region later. The already-explored region is shaded orange, and the frontier is indicated by a purple outline.

1.1 Scope & Limitations

This thesis aims to study the effects of prior knowledge in autonomous exploration and so seeks to control for other sources of noise and errors. This sets the overall scope of the thesis: the robot must know its location with respect to the environment to build its map, and it must know the map in order to know its location, a classic robotics problem known as *simultaneous localization and mapping* (SLAM). Throughout this thesis, the hypothetical SLAM system is assumed to have no errors and to produce noise-free maps.

In order to reduce the degrees of freedom of the robot, a *holonomic* robot is assumed. A holonomic robot is one that can move in any direction, and this affects what states are easy or difficult to reach: cars are non-holonomic and are difficult to move sideways or rotate on the spot; the gantry of a claw machine, however, is holonomic and can move to any position in any order of movements. A symmetric sensor geometry is also assumed, meaning that the robot has no orientation, only position. The world is considered two-dimensional, except in Paper [C] where it is three-dimensional.

1.2 Contributions & Included Papers

This thesis addresses the following two tightly connected questions:

Q1. How can robots learn to predict unknown regions of the environment to aid autonomous exploration? (Papers [A, B])

Q2. How can better plans be made when prior knowledge is available for autonomous exploration? (Papers [C, D])

Perhaps the most important contribution of this thesis is to break the cyclic dependency between these two questions: there is little reward in solving Q1 until Q2 is solved; likewise, there is little reward in solving Q2 until Q1 is solved. The included papers follow. All papers are available online at the listed DOI or arXiv links.

[A] L. Ericson and P. Jensfelt, "FloorGenT: Generative Vector Graphic Model of Floor Plans for Robotics", in *International Conference on Intelligent Robots and Systems*, IEEE, 2022. DOI: 10.1109/IROS47612.2022.9982144.

[B] L. Ericson and P. Jensfelt, "Beyond the Frontier: Predicting Unseen Walls from Occupancy Grids by Learning from Floor Plans", *IEEE Robotics and Automation Letters*, 2024. DOI: 10.1109/LRA.2024.3410164.

[C] L. Ericson, D. Duberg, and P. Jensfelt, "Understanding Greediness in Map-Predictive Exploration Planning", in *European Conference on Mobile Robots*, IEEE, 2021. DOI: 10.1109/ECMR50962.2021.9568793.

[D] L. Ericson, J. Pedro, and P. Jensfelt, "Information Gain Is Not All You Need", Under review, 2025. arXiv: 2504.01980.

A more comprehensive summary of the included papers is available at the end of this first half of the thesis in Chapter 5. The following is a list of contributed papers that were not included.

[X1] M. C. Welle, L. Ericson, R. Ambruș, and P. Jensfelt, "On the Use of Unmanned Aerial Vehicles for Autonomous Object Modeling", in *European Conference on Mobile Robots*, IEEE, 2017. DOI: 10.1109/ECMR.2017.8098656.

[X2] J. Tang, L. Ericson, J. Folkesson, and P. Jensfelt, "GCNv2: Efficient Correspondence Prediction for Real-Time SLAM", *IEEE Robotics and Automation Letters*, 2019. DOI: 10.1109/LRA.2019.2927954.

[X3] L. Wild, L. Ericson, R. Valencia, and P. Jensfelt, "ExelMap: Explainable Element-based HD-Map Change Detection and Update", in *ECCV Workshop on Vision Centric Autonomous Driving*, Springer, 2024. arXiv: 2409.10178.

Chapter 2

Autonomous Exploration

The beginnings of autonomous exploration lie in *active perception* as proposed by Bajcsy [4] who first identified the connection between perception and exploration when she wrote, *"Perceptual activity is exploratory, probing, searching; percepts do not simply fall onto sensors as rain falls onto ground. We do not just see, we look."* She stresses that perception should not be treated as a passive process but rather one in which the agent deliberately and dynamically controls its sensing parameters. By actively choosing where to look, how to move, and how to focus sensors, the robot reduces its uncertainty of the environment with intent rather than by accident.

The term *autonomous exploration* was first coined by Whaite and Ferrie [5] in their paper "Autonomous Exploration: Driven by Uncertainty". They proposed a framework in which the robot iteratively selects new vantage points to reduce uncertainty in its current model of the environment. By quantifying and actively responding to this uncertainty, the robot determines where and how to move next by maximizing *information gain*. The approach of driving perception by an explicit measure of the agent's knowledge gaps ties closely to the ideas of Bajcsy [4].

Though autonomous exploration did not originally refer to exploration using a mobile platform, that is what the term has come to mean in recent years, and it is how the term will be used in this thesis.

2.1 Problem Definition

Autonomous exploration can be described as the problem of designing an exploration policy $\pi : S \rightarrow A$, a mapping from the robot state S to actions A. The robot state is some positional state $x \in X = \mathbb{R}^d$ and the reconstructed map $M \in \mathcal{M} : E \rightarrow \{0, 1\}$. The map is an indicator function of occupied points O defined over the domain of *explored* points $E \subseteq X$. The map is constructed from N observations $Z = \{z_1, z_2, \ldots, z_N\} \subseteq \mathcal{Z}$, obtained by the observation function $h : X \rightarrow \mathcal{Z}$. The observations Z are intentionally left undefined, noting that they

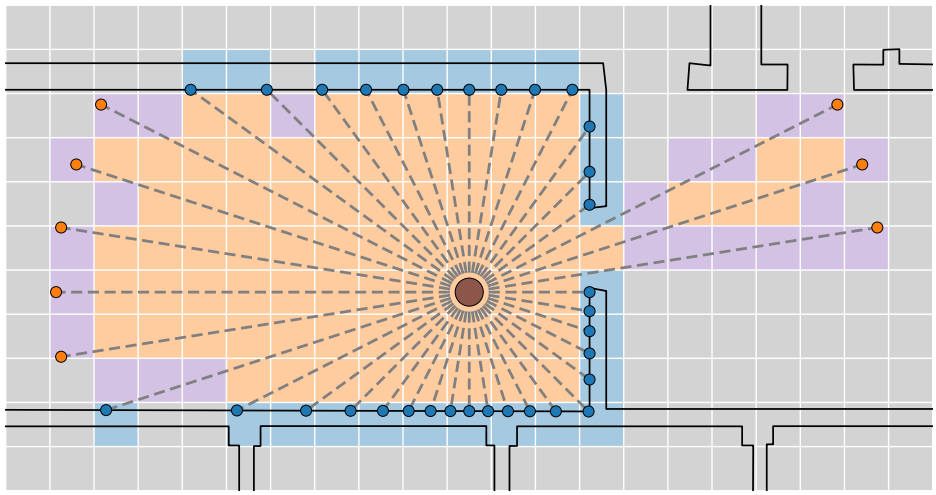


Figure 2.1: Illustration of exploration in the setting of a 2D occupancy grid. The robot (brown circle) is equipped with an omnidirectional range sensor that emits laser pulses, *rays*, in all directions (dashed lines) which measures the time it takes for each ray to reflect back, the *time of flight*, which is proportional to the distance a given ray traveled. Because light decoheres as it propagates, there is an upper range limit; if the ray terminates before then, it is because it is obstructed by the environment, and it is considered a *hit* (blue circles). The occupancy grid cell of each hit is marked occupied (blue); otherwise, where the ray (dashed lines) intersected cells, cells are marked free (orange). Free cells next to unknown cells (gray) are frontier cells (purple). Note that this example sensor only emits a small number of rays whereas real-world range sensors emit hundreds of rays, and the cell sizes have been made larger for the purpose of illustration.

are only relevant to the exploration problem in that they reconstruct the map M. The actions are the possible targets to navigate towards, and is selected among one of the *frontiers* $f \in F$ where the frontier set $F = \partial E \setminus O$ is the unoccupied boundary of explored space. An illustration of the autonomous exploration task is shown in Fig. 2.1. The general approach in exploration is to navigate towards whichever frontier has the maximum score $s : X \to \mathbb{R}$, i.e.,

$$\pi(x, M) = \operatorname*{argmax}_{f \in F} s(f). \tag{2.1}$$

One of the earliest and most influential works in autonomous exploration in the mobile robotics sense is *nearest frontier exploration* by Yamauchi [6], who first introduced the idea of frontiers. As the name suggests, the robot moves towards the nearest frontier, with

$$s(f) = -d_M(x, f) \tag{2.2}$$

and $d_M(x, f)$ is the travel distance from x to f in map M.

The idea of using information gain in autonomous exploration was first proposed by [7] in *Information Based Adaptive Robotic Exploration*, where the key idea is that the exploration policy should consider uncertainty in state estimation since the map is dependent on accurate localization and vice versa, i.e., the SLAM problem. Bourgault et al. [7] thus proposes two terms of information gain; one with respect to state estimation of x, and one with respect to the map M. Subsequent works in autonomous exploration only include the latter term, and the two-term variant has since become *active SLAM*.

The central idea in gain-based exploration policies is to prioritize regions of higher gain first, subject to some penalty for distance. The scoring function is therefore distance d as before, and information gain $I(Z, f)$ at the frontier f,

$$s(f) = \lambda I(Z, h(f)) - d_M(x, f) \tag{2.3}$$

where $\lambda \in \mathbb{R}$ is the *information gain affinity*. A larger λ means the policy will travel longer distances for a given information gain, while $\lambda = 0$ reduces to nearest frontier exploration. Though the definition of I varies between works, the perhaps most ubiquitous class are information-theoretic definitions such as [7], [8], [9], where information gain is the *relative entropy* of the map *belief* distribution $\mathrm{p}(M \mid Z)$ given observations Z and post-observation belief distribution $\mathrm{p}(M \mid Z, z_f)$ after an additional observation $z_f = h(f)$:

$$I(Z, z_f) = \sum_{M \in \mathcal{M}} \mathrm{p}(M \mid Z, z_f) \left(\log \mathrm{p}(M \mid Z, z_f) - \log \mathrm{p}(M \mid Z) \right). \tag{2.4}$$

Intuitively, $I(Z, z_f)$ measures the reduction in entropy of the map belief distribution by an observation z_f. Note that z_f is a random variable for planning purposes, so Eq. (2.4) must be computed in expectation.

So far, the mathematical framework has been kept general to cover all forms of autonomous exploration. The uncountably infinite state space X is however algorithmically challenging, and in practice, it is discretized by either quantization or sampling. In occupancy and voxel grids maps, X is defined as a finite set of *cells* by quantizing with $rx \mapsto \lfloor x \rfloor$, where the resolution r determines the number of cells per unit of length, and $\lfloor \cdot \rfloor$ is the floor operator. The map belief distribution $\mathrm{p}(M \mid Z)$ is then defined as the joint distribution of independent Bernoulli random variables $M_x \sim \mathrm{Bern}(p_x)$ representing the occupancy belief of each cell,

$$\mathrm{p}(M \mid Z) = \prod_{x \in X} \mathrm{p}(M_x = M(x) \mid Z). \tag{2.5}$$

Occupancy belief is initialized to its maximum entropy setting, i.e., $\mathrm{p}(M_x) = \frac{1}{2}$, and is represented in log-odds form with the post-observation belief

$$\log \frac{\mathrm{p}(M_x = 1 \mid Z, z_f)}{\mathrm{p}(M_x = 0 \mid Z, z_f)} = \log \frac{\mathrm{p}(z_f \mid Z, M_x = 1)}{\mathrm{p}(z_f \mid Z, M_x = 0)} + \log \frac{\mathrm{p}(M_x = 1 \mid Z)}{\mathrm{p}(M_x = 0 \mid Z)} \tag{2.6}$$

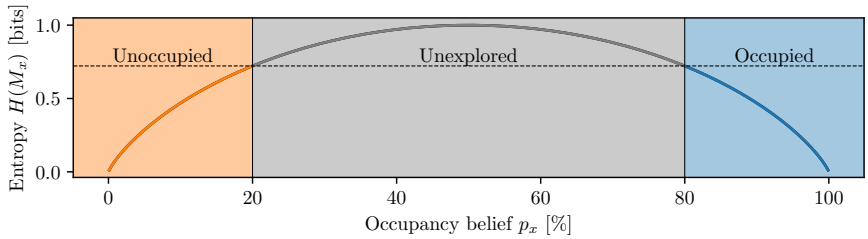


Figure 2.2: The relationship between occupancy belief parameter p_x and the entropy $H(M_x)$ for a single occupancy grid cell. The maximum likelihood estimate $p_x \mapsto [\frac{1}{2} < p_x]$ is only meaningful when the entropy, or uncertainty, is below a predefined threshold (dotted line). Otherwise, the cell is considered unexplored.

Maximizing Eq. (2.4) therefore becomes equivalent to taking actions that maximally push the per-cell occupancy beliefs towards either occupied or unoccupied, illustrated in Fig. 2.2.

It is non-trivial to compute the expectation over all possible observations in Eq. (2.4), and information gain is often approximated by the *belief-of-fact assumption* (BOFA), namely that the maximum likelihood estimate of the map belief is true in explored space, both before and after observation, and that unexplored space has maximum entropy (i.e., one bit). Equation (2.4) then reduces to a single term for the true map M^* after observation,

$$\begin{aligned} I(Z, z_f) &= \log \mathrm{p}(M^* \mid Z, z_f) - \log \mathrm{p}(M^* \mid Z) \\ &= \sum_{x \in X} \left(\log \mathrm{p}(M_x = M_x^* \mid Z, z_f) - \log \mathrm{p}(M_x = M_x^* \mid Z) \right). \end{aligned} \tag{2.7}$$

Since $\mathrm{p}(M_x = M_x^* \mid Z, z_f) = 1$ for cells $x \in X$ explored after observing z_f, but $\mathrm{p}(M_x) = \frac{1}{2}$ for the *novel* cells $x \in F \setminus E$, the sum reduces to

$$I(Z, f) = |F \setminus E| \tag{2.8}$$

measured in bits. In other words, BOFA entails that information gain is equivalent to the number of novel cells observed by visiting a given frontier, which is referred to as *cell-counting gain* in this thesis.

If BOFA holds at least in part, for example because of low-noise sensors, then the map distribution *in unexplored space* determines Eq. (2.4). This is what this thesis will now turn its attention to.

Chapter 3

Modeling Unexplored Space

In principle, the map is not stochastic, and there really is no distribution over maps; the map is the map. The robot, however, can only have *belief* in a map, and its belief is stochastic. In other words, $\mathrm{p}(M \mid Z)$ represents the belief in map M given observations Z. As we saw in the previous chapter, map belief in unexplored space determines information gain under some simple assumptions. This chapter is therefore concerned with modeling map belief in unexplored space, i.e., extrapolating from observations beyond the frontier.

An information-seeking exploration policy will tend to move towards open frontiers regardless of map representation, as unexplored space has maximum entropy by definition. Put differently, it is rarely more informative to remake an observation compared to making a novel observation, except in somewhat contrived circumstances[1]. One of the simplest ways to improve beyond-the-frontier map belief in the occupancy grid representation is to empirically calibrate the unconditional occupancy belief probability $\mathrm{p}(M_x)$; after all, most space is empty. This will cause the policy to be less novelty-seeking, as it now expects to gain less information from novel cells. This is in fact exactly equivalent to attenuating the gain affinity λ in Eq. (2.8), with the relation

$$\lambda \log p_x = \lambda' \log p'_x. \tag{3.1}$$

The first group of works that aim to extrapolate beyond the frontier rely on databases. Ström et al. [10] were first in proposing *predictive autonomous exploration* by matching against a database of previous maps. Oßwald et al. [11] assume a map has been provided, and solve the *offline* autonomous exploration problem where π can be computed ahead of time. In a similar vein, Luperto et al. [12] explore the idea of prior knowledge as given by the user in the form of sketches, bounding boxes, floor plan drawings, or actual maps from previous runs.

[1] If there is a small unexplored hole in the map, it is possible that the information gained from that hole is less than the information gained by observing already-explored space, or more likely, some other open frontier.

Pimentel et al. [13] are perhaps first to predict unexplored space from observations alone, as they propose to use the *Hough line transform* [14] of the occupancy grid to extrapolate the walls into unexplored space on a per-frontier basis, by clustering the frontier into "prediction zones" using a connectivity heuristic, and cell-counting gain is computed for each zone. Luperto et al. [15] predict unexplored space by detecting partially-explored rooms through fitting quadrilaterals to the partially explored map.

Note that in every work reviewed so far, information-seeking exploration policies have been shown to produce substantially *longer* exploration paths than nearest frontier exploration [8], [10], [11], [12], [13]. This counterintuitive result will be put aside for the time being, and instead discussed at length in Chapter 4.

3.1 Learning-based Models

A natural development from explicit approaches are learning-based approaches, which seek to remedy the main drawbacks of explicit approaches: they either require prior knowledge, i.e., there is already a database or map, or the methods have been hand-crafted by experts in an ad-hoc fashion, i.e., line extrapolation and quadrilateral fitting. Learning-based instead model the environment by inductively, learning from experience. Bai et al. [16] use *Gaussian processes* [17], [18] to directly model the information gain $I(Z, f)$. They are perhaps first to point out that there exists an exploration-exploitation tradeoff *within* exploration itself; i.e., should the robot explore so that its model is less uncertain (model exploration), or should it explore what its model believes will improve task performance (model exploitation). In their work, this tradeoff takes the concrete form of exploring frontiers with higher score mean versus those with higher score variance.

Shrestha et al. [19] first proposed to use a *variational auto-encoder* (VAE) as in [20, 21] to learn occupancy beliefs $\mathrm{p}(M \mid Z)$. An auto-encoder is a pair of learned encoder and decoder functions, the encoder $E : \mathcal{M} \to \mathbb{R}^d$ transforms M to a low-dimensional latent code ℓ, the decoder is its inverse $D : \mathbb{R}^d \to \mathcal{M}$, reconstructs M from ℓ. In [19], the two functions are *convolutional neural networks* (CNN) [22, 23] learned by minimizing the reconstruction loss $|\hat{M} - M|$. In a variational auto-encoder, the latent code is a random variable $\ell \sim \mathcal{N}(\mu, \sigma^2)$, and the encoder $E(M) = (\mu, \sigma^2)$ parameterizes the posterior $\mathrm{p}(\ell \mid M)$. The decoder $D(\ell)$ similarly parameterizes the likelihood function $\mathrm{p}(M \mid \ell)$.

Despite multiple attempts by various authors to replicate the results of Shrestha et al. [19], the generated maps have consistently exhibited artifacts such as smudges, holes, and bulges along walls to a degree where the results are unusable. The original network weights were reportedly lost due to a hard drive crash, complicating further replication efforts. However, it may well be that these artifacts stem from inherent limitations of the approach itself. Shrestha et al. [19] deviate from the usual VAE regime by reconstructing the true occupancy grids M^* from the partial occupancy grids M, i.e., not a true auto-encoding formulation. Zangeneh et al.

[24] show that such a formulation is prone to *mode collapse* when learning multi-modal distributions since there may not be any distinctive features in the partial map that allow the encoder to deduce which latent code should be produced. The distribution $\mathrm{p}(M^* \mid M)$ is indeed such a multi-modal distribution, since a given partial occupancy M grid can have many valid true maps M^*. Instead, a *conditional* VAE should be used, encoding and decoding the true map M^* in a traditional auto-encoding formulation; the conditioning M is then provided only to the decoder as auxiliary information.

The occupancy grid factorization in Eq. (2.5) assumes independence of individual occupancy beliefs, and drawing samples from $\mathrm{p}(M)$ therefore amounts to flipping a biased coin in each cell to decide whether it is occupied, with no regard for neighboring cells. Consider a partially explored room whose depth is unknown, the shape (flat) and orientation (perpendicular) of its back wall are nevertheless nearly certain. Once a single sensor ray establishes the true depth, the uncertainty along that entire wall should ideally vanish. Cell-wise independence precludes such correlations, and the probability mass must be instead be spread among all possible wall positions, leading to the smudging effect in which blurred or averaged walls fail to represent the true structure of the environment, the so-called *mean regression* phenomenon.

Humans excel at this kind of spatial reasoning, yet do not reason in terms of dense, cell-based occupancy grids; instead, we form higher-level abstractions and reason about those. Motivated by this insight, Paper [A] and Paper [B] represent the environment as the line segments of floor plans. Representing walls and other architectural boundaries this way enables capturing the essential geometry, and makes it easier both to generalize across different environments and to reason about structural constraints, ultimately allowing for more robust and accurate models.

3.2 Autoregressive Sequence Modeling

Drawing inspiration from Nash et al. [25], the idea in Paper [A] and Paper [B] is to use the tools of natural language models such as *BERT* [26] and *GPT* [27], [28] to model the geometric relationships in floor plans as a sequence of line segment endpoints. In one-shot methods, it is as if the model is painter that must produce the entire painting in one "stroke", whereas in the autoregressive regime, that painter now draws individual strokes one at a time, stopping in between each stroke to consider the partially-finished drawing to decide where to start the next stroke.

The key breakthrough in attention-based networks is the *attention* mechanism proposed by Vaswani et al. [29] and its ability to model long-range dependencies in the sequence. Previous deep learning approaches to sequence learning like [30], [31], [32] must propagate information through each step of the sequence, making long-range dependencies difficult. The attention mechanism solves this issue by allowing every token to *attend* to every other token, thus no propagation is necessary. In

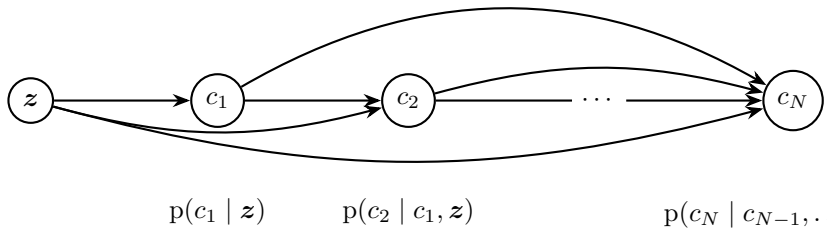


Figure 3.1: Factorization of the joint probability $\mathrm{p}(\boldsymbol{c}, \boldsymbol{z})$ as a probabilistic graphical model. $\boldsymbol{c} = (c_1, c_2, \ldots, c_N)$ is the sequence of tokens, $\boldsymbol{z}$ is conditioning, e.g., sensor data.

generative models, i.e., those that can be sampled, *causal masking* is used so that past tokens cannot attend to future tokens.

The attention mechanism is best understood as a differentiable, associative memory; given a query vector $\boldsymbol{q}$, and the *memory* of key-value pairs $(\boldsymbol{k}_i, \boldsymbol{v}_i)$, the compatibility of each key-query pair is computed by $a_i = \exp(\boldsymbol{q} \cdot \boldsymbol{k}_i)$. The output vector $\boldsymbol{y}$ is the average of the value vectors $\boldsymbol{v}_i$ weighted by a_i. Intuitively, the keys and queries can be thought of as the names of attributes: color, shape, size, etc; the values are the corresponding answers: red, round, large. In *multi-head attention*, each layer consists of many heads, each head able to query different attributes; the next layer then further processes the answers to each head's query.

In autoregressive sequence models, the next-token distribution $p_\theta(c' \mid \boldsymbol{c}, \boldsymbol{z})$ is learned through a deep network $\boldsymbol{y} = f_\theta(\boldsymbol{c}, \boldsymbol{z})$ where $\boldsymbol{y}$ is a parameterization of the probability function, $\boldsymbol{c}$ are the previous tokens, and $\boldsymbol{z}$ is any potential conditioning; a graphical model is provided in Fig. 3.1. In Paper [A] and Paper [B], a categorical distribution is used with $\boldsymbol{y}$ being the *logits* of the next token, transformed to probabilities by the softmax function

$$p_\theta(c' \mid \boldsymbol{c}, \boldsymbol{z}) = \frac{\exp y_{c'}}{\sum_k \exp y_k}. \tag{3.2}$$

The weights θ are found by minimizing the *cross-entropy* between p_θ and the one-hot distribution $q(c') = [c' = \hat{c}']$ of the ground truth next token $\hat{c}'$, which is the same as maximizing the log-likelihood of that token:

$$\begin{aligned} \operatorname*{argmin}_\theta H(p_\theta, q) &= \operatorname*{argmax}_\theta \log p_\theta(\hat{c}' \mid \hat{\boldsymbol{c}}, \boldsymbol{z}) \\ &= \operatorname*{argmax}_\theta \left(y_{\hat{c}'} - \text{log-sum-exp}\, \boldsymbol{y} \right). \end{aligned} \tag{3.3}$$

In deep learning, Eq. (3.3) is optimized numerically by some form of *stochastic gradient descent* [33], [34], [35], [36], wherein batches of random samples are drawn uniformly from the set of training pairs

$$\mathcal{D} = \{(c'_1, \boldsymbol{c}_1, \boldsymbol{z}_1), (c'_2, \boldsymbol{c}_2, \boldsymbol{z}_2), \ldots\} \tag{3.4}$$

to approximate the gradient of f_θ. The initial weights are randomly sampled, and a step of length η is taken at each sampling point to obtain the next parameters θ':

$$\theta' = \theta + \eta \, \mathbb{E} \, \nabla_\theta \log p_\theta(\hat{c}' \mid \hat{\boldsymbol{c}}, \boldsymbol{z}). \tag{3.5}$$

The process is then repeated, drawing a new sample of training pairs. Crucially, the attention mechanism is able to process entire sequences at once rather than individual next-token transitions, enabling vastly more efficient training than traditional multi-layer perceptrons without the locality bias of convolutional neural networks.

3.3 Modeling Floor Plans

There is no one-size-fits-all map representation, and modeling maps as they are represented in a given robot embodiment is consequently limited to that embodiment. In Paper [A], we sought to avoid this issue by making a general model over floor plans, allowing users to perform whatever sensor simulation is appropriate for their given embodiment.

A floor plan is a set of line segments L, each line segment is a pair of vertices $\{(a, b), (c, d)\}$. This definition is invariant to the ordering of the segments, as well as the direction of each segment, and an ordering of both line segments and vertices must be established to transform them into canonical sequences. The solution used in both Paper [A] and Paper [B] is a *proximity heuristic*, where line segments are ordered by the distance from the robot to the nearest point on each line segment. The vertices of each segment are then ordered lexicographically. In this way, the set L is unambiguously transformed into an $|L| \times 2 \times 2$ tensor. In Paper [A], this tensor is flattened and the token sequence are the coordinates

$$\boldsymbol{c}(L) = (a_1, b_1, c_1, d_1, \tag{3.6}$$

$$a_2, b_2, c_2, d_2, \ldots) \tag{3.7}$$

as in [25]. However, this means that sample generation is also coordinate-by-coordinate, which is not necessary. Indeed, Paper [A] shows that the uncertainty is lowest for a_i tokens, i.e., the first coordinate of the first vertex, suggesting that the decision is often made already then. For this reason, Paper [B] does not flatten the last dimension of the tensor, the token sequence is thus the vertices themselves

$$\boldsymbol{c}(L) = ((a_1, b_1), (c_1, d_1), \tag{3.8}$$

$$(a_2, b_2), (c_2, d_2), \ldots) \tag{3.9}$$

Finally, the network input $\boldsymbol{x}$ is obtained from $\boldsymbol{c}$ by *discrete embedding*, where each token (i.e., a coordinate or a vertex) is quantized and a look-up table is used to determine the learned embedding vector.

The key difference between Paper [A] and Paper [B] lies in the context $\boldsymbol{z}$. The goal in Paper [A] was to show that language modeling is a viable tool for modeling

floor plans for robotics applications, and the context was primarily empty except in the evaluation of rasterization-based generation where the 25 initial line segments were provided as a binary image. The method was evaluated for general geometric indoor understanding, demonstrated by showing that the model could better predict the shortest travel distance to points in the vicinity of the robot from its current location.

The aim in Paper [B], on the other hand, was to extend Paper [A] to a realistic robotics scenario by predicting unseen walls from partial occupancy grids, i.e., precisely the problem faced in autonomous exploration. This is considerably more difficult, as the occupancy grid is the result of past actions, i.e., it is path dependent. For example, a given trajectory can be translated slightly, in whole or in part, to yield a similar but slightly different occupancy grid, and it should produce largely the same output. Paper [B] evaluates the model on predicted versus actual information gain, and it was shown that the line segment-based model outperforms all cell-based approaches evaluated. Crucially, those cell-based approaches all converge to the same error plateau where training error can no longer be reduced, suggesting that the task itself (predicting per-cell occupancy beliefs) is the limiting factor, rather than network architecture or parameter capacity.

3.4 Cluttered Environments

Many map modeling works, e.g., [11], [13], [15], Paper [A], Paper [B], assume that the environment is devoid of *clutter*. Clutter refers to objects in the environment that are typically non-static and small compared to the environment itself, such as furniture, objects on tables, clothes hung on coat racks, etc. Such clutter is not part of the environment *ideal*, but rather temporary guests, and are difficult to model for this reason. This uncluttered assumption is justified for two reasons: first, convenience, as datasets of floor plans such as [37] describe ideals; second, the hypothesis in modeling the map is that the overall environment structure is what matters to exploration, not the clutter in it. For example, it is unrealistic to model what objects will lie on a predicted table in a predicted room. The boundary between clutter and ideal is necessarily ill-defined, but it must be drawn somewhere; floor plans offer such a boundary. It is important to note that exploration can serve many purposes, e.g., in [38], the task is to find an object in an unknown environment; in that case, such objects cannot be considered clutter.

This clutter-free assumption presents a problem for deploying such models in real-world environments which are often cluttered. By an auspicious coincidence, this problem has been at least partially addressed by the technology giants' recent race to provide indoor turn-by-turn navigation. Turn-by-turn navigation faces exactly the same problem of wanting to extract ideal representations from cluttered realities. Apple Inc. [39], for example, provide a toolset for inferring the floor plan directly from a stream of sensor data on their handheld mobile devices. Paper [B] was therefore able to address this limitation by showing that, without adaptation,

the model is able to predict unseen parts of the environment from the floor plan recovered by such a toolset.

3.5 Future Work

The problem of map modeling is not limited to autonomous exploration, and a separate strand of similar research has started in the field of autonomous driving. Predicting the road network is in many ways similar to predicting floor plans; they are both planar idealizations of a cluttered 3D world. Liao, Chen, et al. [40] propose to solve this problem in a similar way to Paper [B], by predicting line segments of the road and its lanes. This sparked work on Paper [X3]. Since there is considerably more research effort put into autonomous driving, translating those developments to map prediction is a promising future direction. One could also imagine a combination of the two: autonomous exploration for road networks, the road network models then really are the map models.

More concretely, latent diffusion models [41] are also a promising direction for future work, as these lift one of the main limitations of generative autoregressive sequence models: namely, once a choice is made, i.e., a token is sampled, it is never resampled. Diffusion models instead start from noise, and iteratively refine the entire generated sequence, allowing later positions to influence earlier positions. In fact, a diffusion model can be defined over sets instead of sequences, which is a more natural choice for floor plans as they are ultimately sets of line segments.

A separate but related field that uses floor plans models is found in architecture design tools, or "inverse CAD", i.e., recovering a *computer-aided design* document (CAD) from an image of a floor plan, or sensor data [39], [42], [43], [44], [45]. These works often focus on semantic understanding, which could be interesting as a means of providing robotic intuition in the form of "a fork can be found in the kitchen", which could be part of the pipeline of holistic planning work such as [38]. Recent work using *vision language models* such as [46] and other foundation models have had similar aims, as a means of connecting the semantic to the physical world, such as [47].

As noted previously, improving exploration performance is not as simple as improving the map model, and neither Paper [A] nor Paper [B] were for this reason evaluated as the basis of an exploration policy. This will be the focus of Chapter 4, improving exploration performance with prior map knowledge.

Chapter 4

Predictions and Exploration

As noted in Chapter 1, maximizing information gain does not necessarily improve exploration performance. This has been known since at least 2003, when Stachniss and Burgard [8] showed that *any* gain affinity λ in Eq. (2.3) produces longer paths, shown in Fig. 4.1. They also show the opposite effect for the number of observations, and since their observations are from a sonar, they take a relatively long time to make, and are noisy; λ controls the tradeoff between number of observations and path length. However, as shown in the figure, the path length grows an order of magnitude more than the number of observations shrinks as λ is increased, therefore the potential improvement due to prioritizing information gain is limited.

Indeed, Ström et al. [10] report a factor 1.7 times longer paths for their map-predictive method compared to nearest frontier, arguing that *"The advantages of the prediction-based approach come at a cost—the cost of traversing exploration paths that are longer than the ones generated by the frontier-based approach."* Oßwald et al. [11] find the optimal exploration tour offline by solving a *traveling salesman problem* (TSP), and find that maximizing information gain produces worse results than nearest frontier. Pimentel et al. [13] show that nearest frontier exploration outperforms both information gain maximization and their proposed method. Interestingly, they also report that time to completion follows the same pattern, disproving the proposition that information gain should be maximized even for the case of slow sensors.

Paper [D] investigates this effect, and we show that *negative* values of λ actually perform best, suggesting, absurdly, that information gain should be *minimized*. The reason is that information gain is a proxy for frontier *depth*: the closet has lower information gain precisely because it is a shallower frontier, i.e., it is easily finished.

That is not to say that there is no place for information gain maximization. In Paper [D], we argue that if the goal is to minimize distance traveled, in what we call *quality-constrained exploration*, then information gain should not be used to score frontiers. Rather, information gain decides which parts of the environment are considered unexplored, i.e., where the frontiers are. If the goal however is to

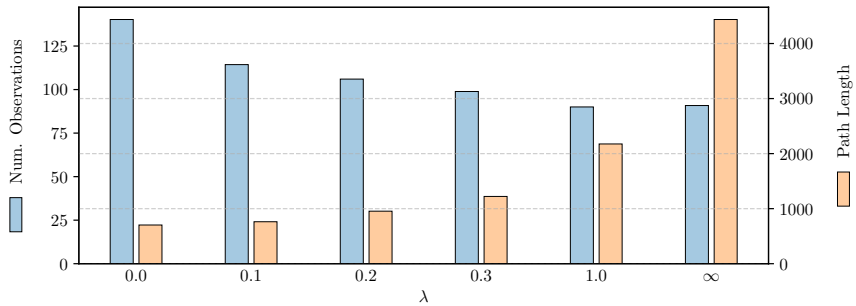


Figure 4.1: Effect of gain affinity λ, adapted from Stachniss and Burgard [8]. Nearest frontier, $\lambda = 0$, produces the least travel distance (path length), but also the largest number of observations. Pure information gain maximization, $\lambda = \infty$, has the opposite effect. The number of observations is reduced by a factor 0.7, while the path length increases by a factor 6.3.

make as few observations as possible, what we call *budget-constrained exploration*—either because of a slow sensor, or a limited resource such as a finite film roll—then it is appropriate to maximize information gain. It is therefore illogical to evaluate a method's travel distance, a quality-constrained behavior, if it is maximizing information gain, a budget-constrained behavior. Instead, a budget-constrained method should be evaluated on the amount of information it collects under its budget. This budget-vs-quality conflation is common in literature, as information gain maximization has become the de-facto standard approach to autonomous exploration. This is the main point of Paper [D]: to challenge the status quo of information gain-driven exploration.

The effect is particularly pronounced when map predictions are used because the information gain can be better maximized and so its consequences become more apparent. It is no surprise then that in evaluating map-predictive methods, authors often choose to terminate exploration before full completion, instead reporting travel distance at some percentage of total *coverage* [15], [19]. Definitions of coverage vary, but generally refers to the volume explored $V(E)$ as a proportion of the volume of all space $V(X)$. Stopping at a percentage of total coverage therefore presupposes $V(X)$ is known *a priori*, and such a *stopping criterion* is consequently impossible to implement in practice. Luperto et al. [48] propose to solve this issue by predicting overall completion via deep learning. We argue instead that it is the conflation of quality-vs-budget constrained exploration that makes it necessary to stop early in the first place. In our framework, quality-constrained exploration stops once there is no more information to gain while minimizing the budget spent; conversely, budget-constrained exploration stops once there is no more budget to spend while maximizing information gained. The choice of paradigm is ultimately application dependent, but one should not evaluate a budget-constrained method with a quality-like metric, and vice versa.

4.1 Beyond Information Gain

It is possible to maximize information gain with respect to the model, and therefore minimizing uncertainty in the distribution of maps. However, the learned distribution of $\mathrm{p}(M^* \mid M)$ is not the same as the empirically-derived $\mathrm{p}(M \mid Z)$, and one must at this point reassert the purpose of autonomous exploration. Even if the model is a perfect oracle, it is not sufficient to produce a predicted map; the environment must be observed directly. A rescue robot that believes it knows the map with low uncertainty cannot terminate its mission early. With that said, there is an exploration-exploitation tradeoff, similar to [16], where the robot can either reduce its model uncertainty so that it can make better plans (explore), or it can execute those better plans to observe the environment (exploit).

Ho et al. [49] recently proposed such a method where the information gain with respect to the map model is maximized, i.e., no exploitation. A stated benefit is that some parts of the map then do not need to be explored since it is certain what they will contain. We argue that this only true if predicting the map is itself the the end goal, i.e., if exploration is done in order to predict a map. We take the opposite standpoint, that the map is predicted in order to do autonomous exploration.

While information-theoretic approaches are no doubt central to the exploration problem, there is something missing in the formulation of Eqs. (2.3) and (2.4). Information gain is computed with respect to a given frontier f, in other words, it only measures the information gain of a single point in space, disregarding the likelihood of obtaining that information elsewhere. This is ultimately why the mistake in the corridor-closet example in Chapter 1 is made, the information gain in the closet is in fact higher if it was computed in expectation over visited future states. This is however intractable to compute, as it is an expectation over future states, observations, and the map belief.

Nearest frontier exploration stands out as one of few exploration paradigms that does not maximize information gain. Still, Yamauchi [6] proposed that nearest frontier exploration is tantamount to doing so: "To gain the most new information about the world, move to the boundary between open space and uncharted territory." No argument is presented to support this proposition, and in fact, the reason that nearest frontier exploration is still used to this day is precisely that it does *not* maximize information gain.

Li et al. [50] showed in 2012 that the margin for improvement compared to the state-of-the-art was small by solving the optimal exploration problem offline, using the tree search method A* [51]. However, the exploration state includes the partially-explored map M, and searching the exploration tree becomes intractable for even trivial map sizes. This is also the main limitation of [50], the map sizes and types are somewhat trivial, and mistakes made by the exploration policy may not be evident in the results because of this. For example, Li et al. [50] report travel distances around 50 m to 100 m, while we in Paper [B] observe distances around 1000 m to 2000 m, roughly a factor of 20 times longer paths. Oßwald et al. [11] further corroborate this analysis by showing that their TSP-based approach in fact

significantly reduces travel distance.

One group of works solve TSP online in the partially-explored map [52], [53], seemingly inspired by works performing global path optimization such as [11]. Such a TSP solution answers a different question, namely how to optimally visit all frontiers, however, when the robot approaches a frontier and pushes it back, the TSP solution becomes invalid. The TSP must therefore encode the cost of finishing the frontier, e.g., is it leading into a supply closet, or a corridor. These methods are therefore particularly suited to the map-predictive paradigm, as the TSP can then accurately reflect the cost of finishing each frontier, as opposed to just reaching it.

4.2 Inverse Covisibility Scoring

The main goal of Paper [A] was to determine what kind of planner and predictor are necessary to improve autonomous exploration in the map-predictive paradigm. It is also shown empirically that information gain maximization leads to increased *fragmentation* of unexplored space as small regions of unexplored space are left behind, as quantified by measuring *isoperimetric ratio* (IPR), the ratio of the surface area to the volume.

The proposed approach is called *inverse covisibility scoring* (ICVS), with the goal of prioritizing the exploration of points in space that are the least covisible with other unexplored points. The frontier score is formulated as the integral of score density $\rho(x) \in \mathbb{R}$ over the visible points $h(x)$,

$$s(x, f) = \sum_{x \in h(f) \setminus E} \rho(x) \quad \text{with} \quad \rho(x) = \exp(-\alpha |W(x) \setminus E|). \tag{4.1}$$

The covisible points $W(x) \subseteq X$ is the set of all points y visible from any state s that observes x, i.e., the visible points $h(x)$ of the inverse visibilities $h^{-1}(p)$,

$$W(x) = \{y \in h(s) : s \in h^{-1}(x)\}. \tag{4.2}$$

Note that since $\rho(x)$ tends to one exponentially as the covisible volume $|W(x) \setminus E|$ tends to zero linearly, the score is near-zero in most places except those that observe non-covisible points. The robot is driven into corners and similar, since only unexplored space is considered; the closer the robot gets to the corner, the fewer covisible points there are, and the larger the score grows. A central question in Paper [A] is to assess, once an exploration policy improves with prior map knowledge, quantifying how much prior knowledge is necessary to reach saturation of performance improvement. It is found that already at 1 m to 2 m, performance improves drastically, and saturates at 8 m. The exact numbers are of course contingent on sensor radius, environment size, etc; the point is rather to show that extrapolating from the frontier is a viable approach, rather than having to assume nearly complete prior knowledge as in Oßwald et al. [11] and Li et al. [50].

Paper [A] also evaluates the use of multi-step planning, i.e., not just maximizing the score of a target frontier and planning a shortest path, but instead planning

the highest-scoring path directly. The hypothesis was that to solve the corridor-closet issue, the planning depth must be sufficiently large to plan into and then back out of the closet and into the corridor. However, it is shown that such an approach actually confers relatively little benefit over simply choosing the highest-scoring shortest path. This is because multi-step planning is a form of tree search, and even at relatively modest tree depths becomes so large that it is unfeasible to search exhaustively. The fact that the robot should explore the closet before the corridor requires that the robot finds the optimal tour over the entire map, similar to Oßwald et al. [11]. Computing the optimal tour is not feasible in many scenarios for two reasons: first, it requires prior map knowledge to be nearly complete, so that an optimal tour can be planned; and second, it is expensive, and solving the problem offline implies difficulties in replanning if the prior map knowledge turns out to be wrong. ICVS solves both issues, but has at least one critical flaw: it requires sensor simulation when evaluating $h(s)$, and the sensor inverse $h^{-1}(x)$ for essentially every point in E, which is impractical outside of simulated experiments.

4.3 Distance Advantage

The key idea in Paper [D] is that some places are more inaccessible than others, and that the exploration policy should visit generally inaccessible frontiers that are easily accessible from its current location. This is quantified by *distance advantage* which measures the distance to a given frontier $f \in F$ from the current location x, compared to the average distance. More formally, distance advantage scoring is

$$s(x, f) = \mathop{\mathbb{E}}_{y} d_M(y, f) - d_M(x, f). \tag{4.3}$$

The random variable $y \in X$ represents future locations the robot may be at, and plays an important role in determining distance advantage as its distribution function $\mathrm{p}(y)$ is the only design choice. Somewhat problematically, this violates causality as the actual distribution of future states depends on the modeled distribution. One way to escape from this infinite regress is to instead assume that the robot will be moving from some random *source* location, moving by shortest path to some random *target* location. If the distribution of source and target are known, the probability of being at any one state can be computed by *betweenness centrality*, a graph centrality measuring how many shortest paths between pass through each point in the map. $\mathrm{p}(y)$ can then be set to be proportional to how frequently a shortest path passes through y. However, we found that a much simpler approach suffices, letting $\mathrm{p}(y)$ be a uniform distribution over the *local window* around the current location x. The local window is the set of states within some maximum straight-line distance of the robot.

Distance advantage is positive whenever the robot is closer to f than it is expected to be otherwise, and it is zero when the current distance is the average distance; if it is negative, the distance is expected to be lower elsewhere. For example, in the corridor-closet scenario, the robot is closer to the closet than it will be

anywhere else, there are no other ways to enter the closet. An example is provided in Fig. 4.2. By contrast, gain-based methods need a conversion factor (i.e., the gain affinity λ) between relative entropy (bits) and travel cost (meters). There is no reason to believe that λ should be a constant, or that the conversion rate should be linear, log-linear, or some other function of gain and cost. Distance advantage compares like-for-like units: length versus length. No proportionality constant is necessary.

Distance advantage is evaluated on its travel distance against nearest frontier and information gain maximization baselines in three environments: a large office space, a cave, and a maze, and we find that nearest frontier produces a factor 1.2 more travel than distance advantage in the office environment, and for information gain maximization, a factor 1.5.

Paper [D] also examines the behavior of distance advantage in the presence of prediction errors, both with false negatives, i.e., failing to predict some parts of the environment, as well as false positives, i.e., predicting non-existent parts of the environment. These errors take the form of clutter, represented by randomly sampling triangles and inserting them into the environment. We found that distance advantage outperforms its baselines even in these cases. Curiously, with extreme cluttering, non-predictive and predictive distance advantage perform almost exactly the same. We reason that this is because the expected distance in Eq. (4.3) is the same in explored and unexplored space, i.e., the path lengths are similarly distributed. In other words, because the distribution is the same with and without predictions, the expected distance is the same, too. Extreme clutter, as opposed to just clutter, is when the environment is substantially different to its original form, blocking off entire regions. Extreme clutter also seems to bring gain-based exploration and nearest frontier exploration to perform the same. This, we reason, happens by a similar mechanism: expected information gain is distributed the same over all frontiers.

Since distance advantage does not involve the sensor model $h(x)$ at all and performs one breadth-first search (BFS) per evaluated frontier f, it solves one of the main drawbacks of Paper [C], namely, its impracticality. Considering a graph of V vertices and E edges, the worst-case time complexity of distance advantage is V instances of BFS, which is $\mathcal{O}(V(V+E))$, and in a four-connected occupancy grid, $E < 4V$. The overall worst-case time complexity is thus $\mathcal{O}(V^2)$.

The main limitation of distance advantage is that it is only applicable in scenarios where path cost is worth optimizing. There are many cases where this may not true, e.g., non-mobile autonomous exploration as proposed by Whaite and Ferrie [5]. The sensor can be moved almost at low cost, so optimizing its path is less important than figuring out and obtaining high-quality views. It is also unclear whether distance advantage is worthwhile for aerial and subnautical scenarios where space is mostly traversible by straight lines.

4.4 Conclusion & Future Work

Paper [D] finally brings to a close the quest for a realistic exploration policy that improves given prior map knowledge. By reasoning from the perspective of prior map knowledge, a novel exploration policy was developed that improves performance even *without* prior map knowledge, representing the first new baseline since 1997 when Yamauchi [6] proposed nearest frontier exploration.

Distance advantage is not a replacement for information gain maximization, but rather a successor to nearest frontier exploration. Instead, information gain maximization should be thought of as an augmentation to nearest frontier exploration, and the same augmentation can be made to distance advantage exploration. Distance advantage does not try to place the sensor optimally; rather, it tries placing the robot itself optimally to reach inaccessible places. This often means staying near walls, which is often at the cost of to sensor coverage.

There are many potential future directions for distance advantage. For example, distance advantage lends itself to multi-agent scenarios, where $\mathrm{p}(y)$ also includes future locations of other robots. Though distance advantage shows improved performance without prior map knowledge, it also improves even more with; it would therefore be interesting to connect distance advantage to a map model such as those proposed in Chapter 3. Equation (4.3) would then be reformulated as an expectation with respect to both y and M.

Another interesting direction is to apply distance advantage subject to a non-trivial motion model, one where the cost of a path is not simply its length. A single-query sampling-based planner such as [54], [55] would likely be too inefficient, and a multi-query representation in the style of *probabilistic roadmaps* [56] would be preferable.

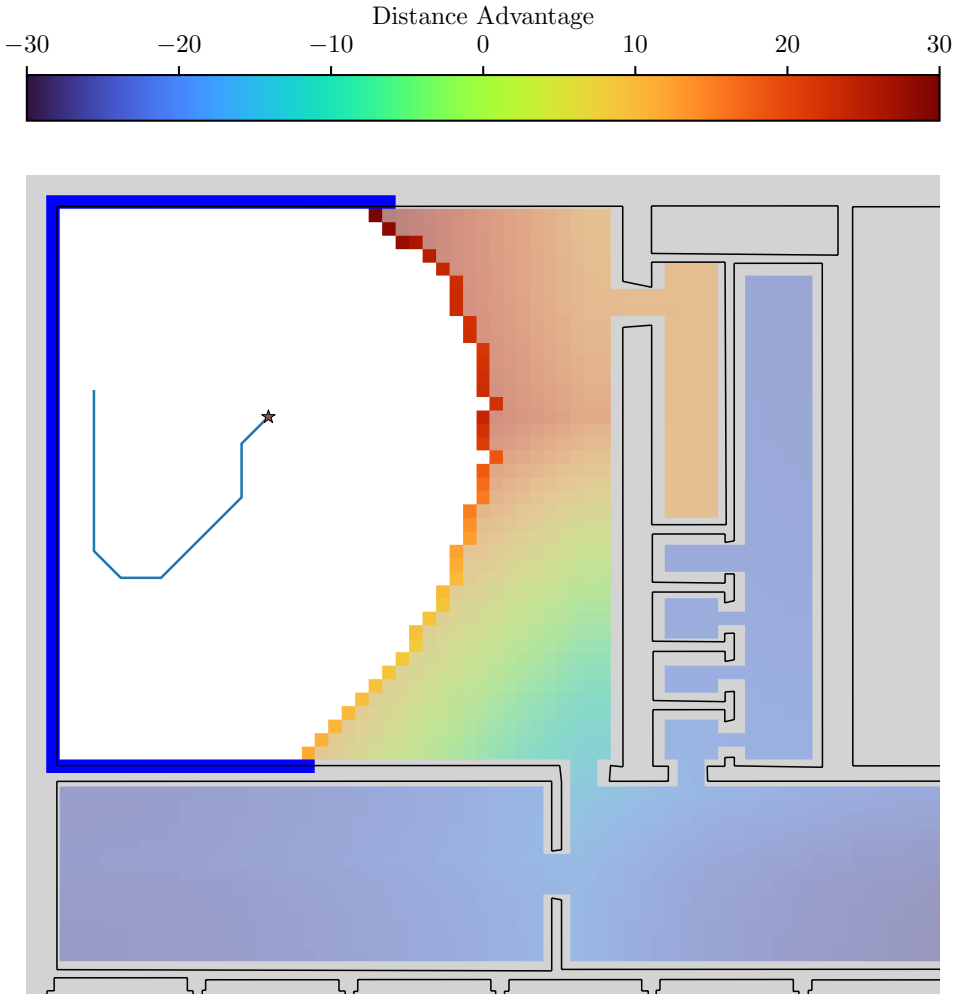


Figure 4.2: Distance advantage at the beginning of exploration with prior map knowledge. Frontier cells F are illustrated as fully-saturated colors; desaturated colors indicate unexplored cells. The scenario is similar to a corridor-closet situation; distance advantage correctly estimates that the closet should be visited first, as it will be further away later on. The reverse is also true, the corridor is further away than it will be in the future. This happens because the corridor connects many rooms, and the distance from most points in each room is lower than this deeper room. The environment is the floor plan of an actual building at a university campus.

Chapter 5

Summary of Included Papers

This chapter contains abstracts of the included papers and contributions by the author of the thesis.

Paper A

FloorGenT: Generative Vector Graphic Model of Floor Plans for Robotics

L. Ericson and P. Jensfelt

In *Proc. Int. Conf. Intelligent Robots and Systems*, 2022

Abstract: Floor plans are the basis of reasoning in and communicating about indoor environments. In this paper, we show that by modelling floor plans as sequences of line segments seen from a particular point of view, recent advances in autoregressive sequence modelling can be leveraged to model and predict floor plans. The line segments are canonicalized and translated to sequence of tokens and an attention-based neural network is used to fit a one-step distribution over next tokens. We fit the network to sequences derived from a set of large-scale floor plans, and demonstrate the capabilities of the model in four scenarios: novel floor plan generation, completion of partially observed floor plans, generation of floor plans from simulated sensor data, and finally, the applicability of a floor plan model in predicting the shortest distance with partial knowledge of the environment.

Contributions by the author: Proposed, designed, and executed ideas and algorithms presented in the paper, and the actual paper.

Paper B

Beyond the Frontier: Predicting Unseen Walls from Occupancy Grids by Learning from Floor Plans

L. Ericson and P. Jensfelt

In *IEEE Robotics and Automation Letters*, 2024

Abstract: In this paper, we tackle the challenge of predicting the unseen walls of a partially observed environment as a set of 2D line segments, conditioned on occupancy grids integrated along the trajectory of a 360° LIDAR sensor. A dataset of such occupancy grids and their corresponding target wall segments is collected by navigating a virtual robot between a set of randomly sampled waypoints in a collection of office-scale floor plans from a university campus. The line segment prediction task is formulated as an autoregressive sequence prediction task, and an attention-based deep network is trained on the dataset. The sequence-based autoregressive formulation is evaluated through predicted information gain, as in frontier-based autonomous exploration, demonstrating significant improvements over both non-predictive estimation and convolution-based image prediction found in the literature. Ablations on key components are evaluated, as well as sensor range and the occupancy grid's metric area. Finally, model generality is validated by predicting walls in a novel floor plan reconstructed on-the-fly in a real-world office environment.

Contributions by the author: Proposed, designed, and executed ideas and algorithms presented in the paper, and wrote the actual paper.

Paper C

Understanding Greediness in Map-Predictive Exploration Planning

L. Ericson, D. Duberg, and P. Jensfelt

In *Euro. Conf. Mobile Robots*, 2021

Abstract: In map-predictive exploration planning, the aim is to exploit a-priori map information to improve planning for exploration in otherwise unknown environments. The use of map predictions in exploration planning leads to exacerbated greediness, as map predictions allow the planner to defer exploring parts of the environment that have low value, e.g., unfinished corners. This behavior is undesirable, as it leaves holes in the explored space by design. To this end, we propose a scoring function based on inverse covisibility that rewards visiting these low-value parts, resulting in a more cohesive exploration process, and preventing excessive greediness in a map-predictive setting. We examine the behavior of a non-greedy map-predictive planner in a bare-bones simulator, and answer two principal questions: a) how far beyond explored space should a map predictor predict to aid exploration, i.e., is more better; and b) does shortest-path search as the basis for planning, a popular choice, cause greediness. Finally, we show that by thresholding covisibility, the user can trade-off greediness for improved early exploration performance.

Contributions by the author: Formulated the problem together with D. Duberg, designed and implemented the solution presented in the paper, and wrote the actual paper.

Paper D

Information Gain Is Not All You Need

L. Ericson, J. Pedro, and P. Jensfelt

Under review

Abstract: Autonomous exploration in mobile robotics is driven by two competing objectives: coverage, to exhaustively observe the environment; and path length, to do so with the shortest path possible. Though it is difficult to evaluate the best course of action without knowing the unknown, the unknown can often be understood through models, maps, or common sense. However, previous work has shown that improving estimates of information gain through such prior knowledge leads to greedy behavior and ultimately causes backtracking, which degrades coverage performance. In fact, any information gain maximization will exhibit this behavior, even without prior knowledge. Information gained at task completion is constant, and cannot be maximized for. It is therefore an unsuitable choice as an optimization objective. Instead, information gain is a decision criterion for determining which candidate states should still be considered for exploration. The task therefore becomes to reach completion with the shortest total path. Since determining the shortest path is typically intractable, it is necessary to rely on a heuristic or estimate to identify candidate states that minimize the total path length. To address this, we propose a heuristic that reduces backtracking by preferring candidate states that are close to the robot, but far away from other candidate states. We evaluate the performance of the proposed heuristic in simulation against an information gain-based approach and frontier exploration, and show that our method significantly decreases total path length, both with and without prior knowledge of the environment.

Contributions by the author: The paper has shared authorship between L. Ericson and J. Pedro. Formulated the problem and co-developed solution together with J. Pedro, implemented and executed method and evaluation, co-authored the paper with J. Pedro, with special focus on producing figures and experimental results.

References

[A] L. Ericson and P. Jensfelt, "FloorGenT: Generative Vector Graphic Model of Floor Plans for Robotics", in *International Conference on Intelligent Robots and Systems*, IEEE, 2022. DOI: 10.1109/IROS47612.2022.9982144.

[B] L. Ericson and P. Jensfelt, "Beyond the Frontier: Predicting Unseen Walls from Occupancy Grids by Learning from Floor Plans", *IEEE Robotics and Automation Letters*, 2024. DOI: 10.1109/LRA.2024.3410164.

[C] L. Ericson, D. Duberg, and P. Jensfelt, "Understanding Greediness in Map-Predictive Exploration Planning", in *European Conference on Mobile Robots*, IEEE, 2021. DOI: 10.1109/ECMR50962.2021.9568793.

[D] L. Ericson, J. Pedro, and P. Jensfelt, "Information Gain Is Not All You Need", Under review, 2025. arXiv: 2504.01980.

[X1] M. C. Welle, L. Ericson, R. Ambruș, and P. Jensfelt, "On the Use of Unmanned Aerial Vehicles for Autonomous Object Modeling", in *European Conference on Mobile Robots*, IEEE, 2017. DOI: 10.1109/ECMR.2017.8098656.

[X2] J. Tang, L. Ericson, J. Folkesson, and P. Jensfelt, "GCNv2: Efficient Correspondence Prediction for Real-Time SLAM", *IEEE Robotics and Automation Letters*, 2019. DOI: 10.1109/LRA.2019.2927954.

[X3] L. Wild, L. Ericson, R. Valencia, and P. Jensfelt, "ExelMap: Explainable Element-based HD-Map Change Detection and Update", in *ECCV Workshop on Vision Centric Autonomous Driving*, Springer, 2024. arXiv: 2409.10178.

[1] S. Omari, P. Gohl, M. Burri, M. Achtelik, and R. Siegwart, "Visual Industrial Inspection using Aerial Robots", in *International Conference on Applied Robotics for the Power Industry*, IEEE, 2014.

[2] D. Calisi, A. Farinelli, L. Iocchi, and D. Nardi, "Autonomous Exploration for Search and Rescue Robots", *Transactions on the Built Environment*, 2007.

[3] F. Colas, S. Mahesh, F. Pomerleau, M. Liu, and R. Siegwart, "3D Path Planning and Execution for Search and Rescue Ground Robots", in *International Conference on Intelligent Robots and Systems*, IEEE, 2013.

[4] R. Bajcsy, "Active Perception", *Proceedings of the IEEE*, vol. 76, no. 8, pp. 966–1005, 1988.

[5] P. Whaite and F. P. Ferrie, "Autonomous Exploration: Driven by Uncertainty", *Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 3, pp. 193–205, 1997.

[6] B. Yamauchi, "A Frontier-based Approach for Autonomous Exploration", in *International Symposium on Computational Intelligence in Robotics and Automation*, IEEE, 1997.

[7] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, "Information Based Adaptive Robotic Exploration", in *International Conference on Intelligent Robots and Systems*, IEEE, 2002.

[8] C. Stachniss and W. Burgard, "Exploring Unknown Environments with Mobile Robots using Coverage Maps", in *International Joint Conference on Artificial Intelligence*, 2003.

[9] M. Selin, M. Tiger, D. Duberg, F. Heintz, and P. Jensfelt, "Efficient Autonomous Exploration Planning of Large-Scale 3D Environments", *Robotics and Automation Letters*, 2019.

[10] D. P. Ström, F. Nenci, and C. Stachniss, "Predictive Exploration Considering Previously Mapped Environments", in *International Conference on Robotics and Automation*, IEEE, 2015.

[11] S. Oßwald, M. Bennewitz, W. Burgard, and C. Stachniss, "Speeding-up Robot Exploration by Exploiting Background Information", *Robotics and Automation Letters*, vol. 1, no. 2, pp. 716–723, 2016.

[12] M. Luperto, M. Antonazzi, F. Amigoni, and N. A. Borghese, "Robot Exploration of Indoor Environments using Incomplete and Inaccurate Prior Knowledge", *Robotics and Autonomous Systems*, 2020.

[13] J. M. Pimentel, M. S. Alvim, M. F. Campos, and D. G. Macharet, "Information-Driven Rapidly-Exploring Random Tree for Efficient Environment Exploration", *Journal of Intelligent & Robotic Systems*, vol. 91, pp. 313–331, 2018.

[14] R. O. Duda and P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures", *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[15] M. Luperto, V. Arcerito, and F. Amigoni, "Predicting the Layout of Partially Observed Rooms from Grid Maps", in *International Conference on Robotics and Automation*, IEEE, 2019.

[16] S. Bai, J. Wang, F. Chen, and B. Englot, "Information-theoretic Exploration with Bayesian Optimization", in *International Conference on Intelligent Robots and Systems*, IEEE, 2016.

[17] C. E. Rasmussen, "Gaussian Processes in Machine Learning", in *Summer School on Machine Learning*, Springer, 2003, pp. 63–71.

[18] S. O'Callaghan, F. Ramos, and H. Durrant-Whyte, "Contextual Occupancy Maps using Gaussian processes", in *International Conference on Robotics and Automation*, IEEE, 2009, pp. 1054–1060.

[19] R. Shrestha, F.-P. Tian, W. Feng, P. Tan, and R. Vaughan, "Learned Map Prediction for Enhanced Mobile Robot Exploration", in *International Conference on Robotics and Automation*, IEEE, 2019.

[20] D. P. Kingma, M. Welling, et al., "Auto-Encoding Variational Bayes", in *International Conference on Learning Representations*, Banff, Canada, 2014.

[21] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic Backpropagation and Approximate Inference in Deep Generative Models", in *International Conference on Machine Learning*, PMLR, 2014.

[22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based Learning Applied to Document Recognition", *Proceedings of the IEEE*, 1998.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", *Advances in Neural Information Processing Systems*, 2012.

[24] F. Zangeneh, L. Bruns, A. Dekel, A. Pieropan, and P. Jensfelt, "A Probabilistic Framework for Visual Localization in Ambiguous Scenes", in *International Conference on Robotics and Automation*, IEEE, 2023.

[25] C. Nash, Y. Ganin, S. A. Eslami, and P. Battaglia, "Polygen: An autoregressive generative model of 3D meshes", in *International Conference on Machine Learning*, PMLR, 2020.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding", 2018. arXiv: `1810.04805`.

[27] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al., "Language models are unsupervised multitask learners", *OpenAI blog*, 2019. [Online]. Available: `https://openai.com/blog/better-language-models/`.

[28] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language Models are Few-Shot Learners", in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2020.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need", in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2017.

[30] J. L. Elman, "Finding Structure in Time", *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[31] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory", *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[32] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech Recognition with Deep Recurrent Neural Networks", in *International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2013, pp. 6645–6649.

[33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Representations by Back-propagating Errors", *nature*, 1986.

[34] L. Bottou, "Online Learning and Stochastic Approximations", *Online Learning in Neural Networks*, vol. 17, no. 9, p. 142, 1998.

[35] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization", in *International Conference on Learning Representations*, 2015.

[36] T. Bachlechner, B. P. Majumder, H. Mao, G. Cottrell, and J. McAuley, "ReZero is All You Need: Fast Convergence at Large Depth", in *Uncertainty in Artificial Intelligence*, PMLR, 2021.

[37] A. Aydemir, P. Jensfelt, and J. Folkesson, "What can we learn from 38 000 rooms? Reasoning about unexplored space in indoor environments", in *Conference on Intelligent Robots and Systems*, IEEE, 2012.

[38] N. Hawes, C. Burbridge, F. Jovan, L. Kunze, B. Lacerda, L. Mudrová, J. Young, J. Wyatt, D. Hebesberger, T. Körtner, R. Ambrus, N. Bore, J. Folkesson, P. Jensfelt, L. Beyer, A. Hermans, B. Leibe, A. Aldoma, T. Fäulhammer, M. Zillich, M. Vincze, E. Chinellato, M. Al-Omari, P. Duckworth, Y. Gatsoulis, D. C. Hogg, A. G. Cohn, C. Dondrup, J. P. Fentanes, T. Krajník, J. M. Santos, T. Duckett, and M. Hanheide, "The STRANDS project: Long-term Autonomy in Everyday Environments", *Robotics & Automation Magazine*, vol. 24, no. 3, pp. 146–156, 2017. DOI: 10.1109/MRA.2016.2636359.

[39] Apple Inc., *3D Parametric Room Representation with RoomPlan*, 2022. [Online]. Available: https://machinelearning.apple.com/research/roomplan.

[40] B. Liao, S. Chen, et al., "MapTR: Structured Modeling and Learning for Online Vectorized HD Map Construction", in *International Conference on Learning Representations*, 2023.

[41] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis with Latent Diffusion Models", in *Computer Vision and Pattern Recognition*, 2022.

[42] Y. Yue, T. Kontogianni, K. Schindler, and F. Engelmann, "Connecting the Dots: Floorplan Reconstruction Using Two-Level Queries", in *Conference on Computer Vision and Pattern Recognition*, 2023.

[43] J.-W. Su, K.-Y. Tung, et al., "SLIBO-Net: Floorplan Reconstruction via Slicing Box Representation with Local Geometry Regularization", in *Conference on Neural Information Processing Systems*, 2023.

[44] J. Chen, Y. Qian, and Y. Furukawa, "HEAT: Holistic Edge Attention Transformer for Structured Reconstruction", in *Conference on Computer Vision and Pattern Recognition*, IEEE/CVF, 2022.

[45] A. Gueze, M. Ospici, D. Rohmer, and M.-P. Cani, "Floor Plan Reconstruction from Sparse Views: Combining Graph Neural Network with Constrained Diffusion", in *International Conference on Computer Vision*, IEEE/CVF, 2023.

[46] A. Radford, J. W. Kim, C. Hallacy, et al., "Learning Transferable Visual Models From Natural Language Supervision", in *Proceedings of the International Conference on Machine Learning*, PMLR, 2021.

[47] F. L. Busch, T. Homberger, J. Ortega-Peimbert, Q. Yang, and O. Andersson, *One Map to Find Them All: Real-time Open-Vocabulary Mapping for Zero-shot Multi-Object Navigation*, 2024. arXiv: 2409.11764.

[48] M. Luperto, M. M. Ferrara, G. Boracchi, and F. Amigoni, "Estimating Map Completeness in Robot Exploration", 2024. arXiv: 2406.13482.

[49] C. Ho, S. Kim, B. Moon, A. Parandekar, N. Harutyunyan, C. Wang, K. Sycara, G. Best, and S. Scherer, "MapEx: Indoor Structure Exploration with Probabilistic Information Gain from Global Map Predictions", 2024. arXiv: 2409.15590.

[50] A. Q. Li, F. Amigoni, and N. Basilico, "Searching for Optimal Off-Line Exploration Paths in Grid Environments for a Robot with Limited Visibility", in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2012.

[51] P. E. Hart, N. J. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", *Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[52] Z. Meng, H. Qin, Z. Chen, X. Chen, H. Sun, F. Lin, and M. H. Ang, "A Two-Stage Optimized Next-View Planning Framework for 3-D Unknown Environment Exploration, and Structural Reconstruction", *Robotics and Automation Letters*, vol. 2, no. 3, pp. 1680–1687, 2017.

[53] B. Zhou, Y. Zhang, X. Chen, and S. Shen, "FUEL: Fast UAV Exploration using Incremental Frontier Structure and Hierarchical Planning", *Robotics and Automation Letters*, 2021.

[54] S. M. LaValle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning", Department of Computer Science, Iowa State University, Technical Report, 1998.

[55] S. Karaman and E. Frazzoli, "Sampling-based Algorithms for Optimal Motion Planning", *International Journal of Robotics Research*, vol. 30, no. 7, 2011.

[56] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces", *Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

Part II

Included Publications

Paper A

FloorGenT: Generative Vector Graphic Model of Floor Plans for Robotics

L. Ericson and P. Jensfelt
KTH Royal Institute of Technology

Abstract

Floor plans are the basis of reasoning in and communicating about indoor environments. In this paper, we show that by modelling floor plans as sequences of line segments seen from a particular point of view, recent advances in autoregressive sequence modelling can be leveraged to model and predict floor plans. The line segments are canonicalized and translated to sequence of tokens and an attention-based neural network is used to fit a one-step distribution over next tokens. We fit the network to sequences derived from a set of large-scale floor plans, and demonstrate the capabilities of the model in four scenarios: novel floor plan generation, completion of partially observed floor plans, generation of floor plans from simulated sensor data, and finally, the applicability of a floor plan model in predicting the shortest distance with partial knowledge of the environment.

 Paper accepted for the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2022). All authors are with the Division of Robotics, Perception, and Learning at KTH Royal Institute of Technology, Stockholm, SE-11428, Sweden. This work was financed by the Swedish Research Council grant XPLORE3D. For e-mail correspondence, contact `ludv@kth.se`.

A.1 Introduction

Reasoning and planning in indoor environments is an important capability in the context of robotics. One approach to this problem is through modelling floor plans. Floor plans are a simplified and minimalist map of an indoor environment, and can be used for both reasoning and communicating about such environments. In this paper, the aim is to predict floor plans based on environmental cues in order to facilitate robotics tasks in unknown environments, such as autonomous exploration and search-and-rescue. To this end, we construct a generative model that encodes the implicit rules of floor plans without explicitly stating those rules, e.g., that walls join at right angles, rooms are symmetrical, and doorways join rooms.

Predicting floor plans from environmental cues is not a novel idea, and it has previously been cast as an image in-painting problem where an image model, typically a convolutional neural network, is used to fill in the missing regions of a rasterized floor plan. However, convolutions imply an inductive bias towards local dependencies in pixel space which is not necessarily a good fit for floor plans, which often exhibit non-local correlations. Furthermore, floor plan rasterizations are unlike natural images consisting exclusively of high-frequency content, making them particularly prone to artifacts such as bleeding and blurring. Care must be taken to prevent such effects as shown in [1].

By modelling floor plans as vector graphics, this issue is avoided. It also lets us leverage recent advances in language models, such as attention-based neural network architectures. In an autoregressive sequence formulation such as [2], [3], the model takes its own previous output into account at each step as it generates an output sequence. This approach has shown remarkable results in modelling and generating various sequences, including but not limited to natural languages. By casting floor plans as sequences, similar performance could be obtained in the floor plan domain, and leveraged in a robotics context.

In this paper we present such an implicit model of indoor environments, and demonstrate its usefulness in a typical robotics tasks. In summary, the contributions of this paper are:

1. A method of representing floor plans as a sequence of ordered line segments seen from a particular point of view within the floor plan, and a procedure to canonicalize the sequences.

2. A attention-based generative model tailored to dealing with such sequences of line segments.

3. An evaluation of the performance in generating diverse novel floor plan sequences, sequence completions from partial sequences, and sequence generation from partial birds-eye view raster images.

4. A demonstration of the model's abilities in solving a typical robotics task, namely path planning in a partially observed environment.

A.2 Related Work

Floor plans as the basis of robotic planning and reasoning has been explored before. In the category of predicting maps, approaches range from explicit to implicit. A recent example of an explicit method is [4], in which an algorithm for predicting the layouts of partially observed rooms in indoor environments is proposed, by constructing an *explicit* set of rules and assumptions about those environments in the algorithm itself. A majority of recent work is based on deep learning, learning the rules from data rather than by design from an expert. The present work belongs to this latter *implicit* category, but differs in a key regard: previous work is largely based on the convolutional neural networks from computer vision, such as [5], [6]. We instead cast the problem as modelling a sequence of line drawing instructions.

Contrastingly, [7] use a *topological* rather than metric representation, e.g., "offices connect to kitchens through corridors", which is then used for symbolic reasoning and planning. In pursuit of this, the *KTH floor plan dataset* was constructed, consisting vector graphics of 38 000 rooms in 184 floor plans from 27 different university buildings. We use the same dataset, though not for its topological information. It is important to note that these floor plans are large, with hundreds of rooms per floor.

Separately from the robotics-aimed line of work is what is becoming known as *inverse CAD*, where the goal is mapping images or pointclouds to architectural blueprints. For example, [8] propose a method to produce a vector graphics floor plan from a set of RGBD sensor scans that fully cover an apartment. In [1], a method is proposed to classify room types (e.g., kitchen, bathroom, bedroom) from rasterized floor plan drawings. Both are based on convolutional neural networks. Inverse CAD is in general aimed at realtors and architects, rather than robotics.

Though not directed at solving robotics tasks, recent work has also been aimed at modelling vector graphics generally. For example, [9], [10] show that vector graphics can be embedded in a latent space of fixed dimensionality. The embeddings then have some spatial relationship to each other, such that similar drawings are nearby in that space. Neither work is aimed at prediction from partial inputs.

In terms of network architecture, we leverage recent developments in sequence modelling for natural languages in a similar style to attention-based models such as [3], [11]. In [12], a method is proposed to model 3D meshes as sets of polygons through the use of attention-based sequence models as proposed in [13]. Our model is built on that same foundation.

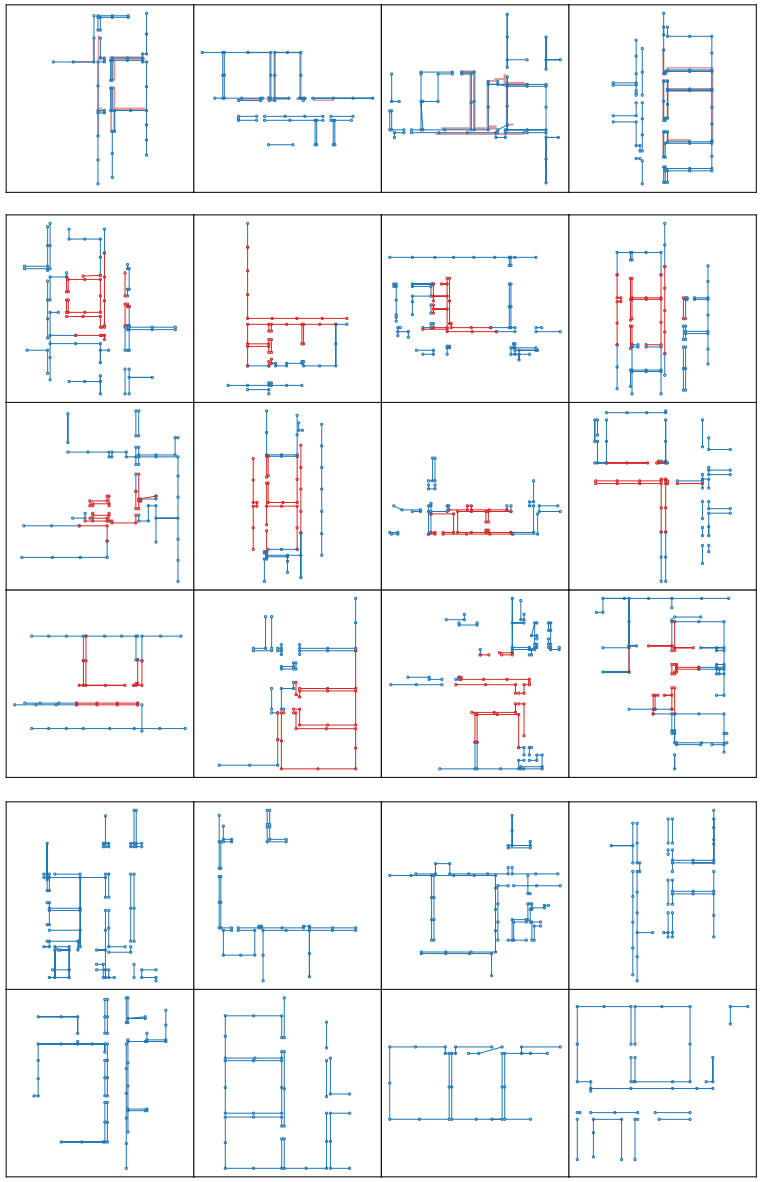


Figure A.1: Randomly picked samples of a FloorGenT network trained on the KTH floor plan dataset generated with nucleus sampling ($p = 90\,\%$). Blue is sampled model output. **Top**: unconditioned novel samples. **Middle**: partial sequence completion samples conditioned on the segments shown in red (first 25 segments of randomly selected test sequences, i.e., novel data to the network). **Bottom**: partial image conditioned samples with the input image shown in red (rasterization of the first 25 segments of randomly selected test sequences).

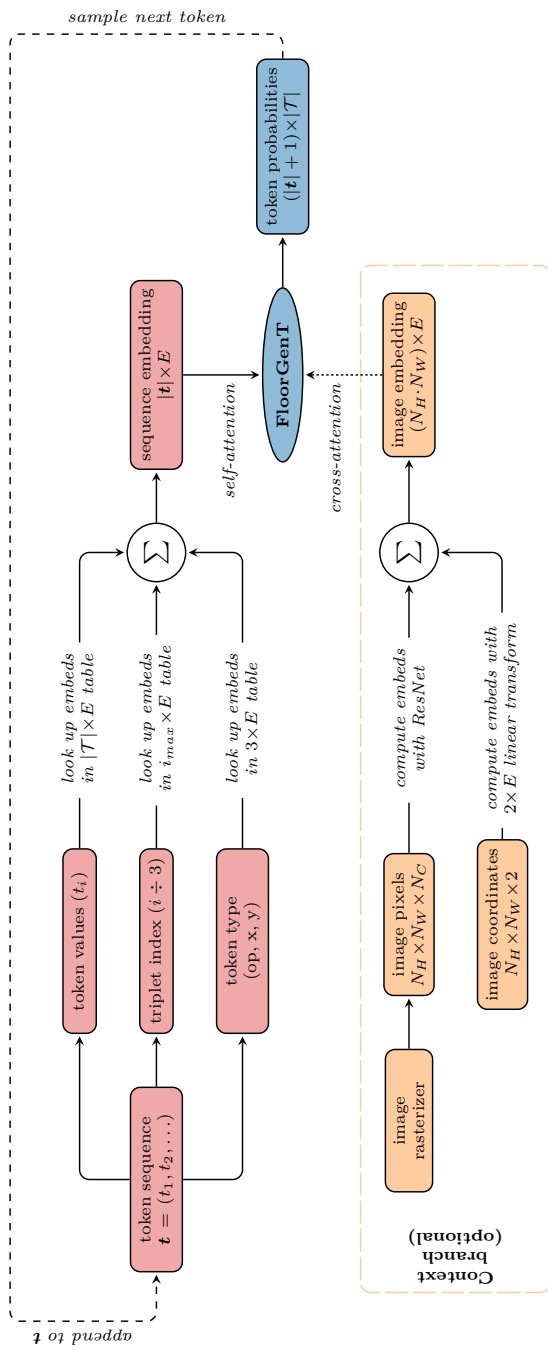


Figure A.2: Overview of data flow in FloorGenT. The input sequence is a possibly empty sequence of tokens $\boldsymbol{t}$. Each token is embedded as a sum of three discrete embedding vectors. The embedded tokens are the inputs of the first self-attention layer, and later layers take the previous layer's output as their input. In image models, the input image pixel values is embedded through a convolutional neural network, and are embedded along with their respective coordinates through a linear transformation. The per-pixel embeddings are then input to the cross-attention layers. When sampling, the next token is repeatedly drawn from the next token distribution, and fed back into the network at the end of the token sequence. E is the number of embedding dimensions.

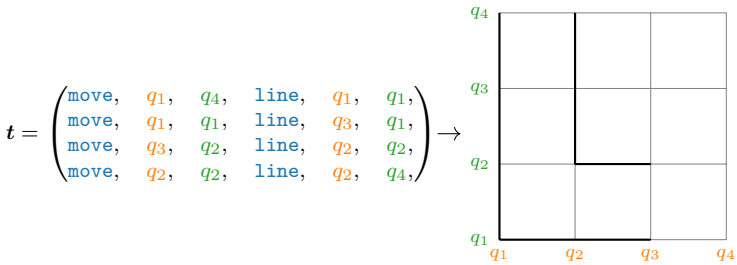


Figure A.3: An example of a token sequence and its corresponding drawing in the shape of an L. Note that in practice, the line segments would be sorted by their distance to some origin coordinate as described in Section A.3.4.

A.3 FloorGenT

We model the distribution over floor plans $\mathcal{F}$ by translating them to sequences of line drawing instructions, encoded as a *token sequence* $\boldsymbol{t}$ and factorizing in autoregressive manner:

$$p(\mathcal{F}) \triangleq p(\boldsymbol{t}) = \prod_{i=1}^{k} p(t_i|\boldsymbol{t}_{<i}) \tag{A.1}$$

where t_i is the ith token in the sequence, and $\boldsymbol{t}_{<i}$ are the tokens before i. The tokens combine into triplets (c, x, y) to form the line drawing instructions, consisting of the opcode $c \in \{\texttt{stop}, \texttt{move}, \texttt{line}\}$ meaning "end of sequence", "move to coordinate", and "draw line to coordinate" respectively, and the operands are the discrete coordinates $x, y \in \{q_1, q_2, \ldots, q_{N_Q}\}$. We denote the set of tokens

$$\mathcal{T} = \{\texttt{stop}, \texttt{move}, \texttt{line}, q_1, q_2, \ldots, q_{N_Q}\} \tag{A.2}$$

Each line segment is encoded as two triplets, a `move` followed by a `line`, illustrated in Fig. A.3. This representation allows manipulating sequences at line segment level, with the drawback that it is redundant when a line segment is joined with the previous segment. This is not typical in our case, as about 12.4 % segment pairs are so joined.

A.3.1 Token Sequence Model

With the sequence representation defined, we turn to modelling the distribution over sequences defined in Eq. (B.6). We use the Transformer decoder architecture with *multi-head scaled dot product attention* as in [13]. These networks were originally proposed to model natural languages, and our token sequences can similarly be thought as a synthetic language with its semantics defined primarily by the resulting vector graphic, making it necessary to "read between the lines".

The architecture is presented in Section A.6. The output of the model is a categorical distribution $p(t_i|\boldsymbol{t}_{<i};\theta)$, where θ denotes the parameters of the model. The loss $\mathcal{L}$ is the negative log likelihood of the ground truth token values $\hat{t}$ under the model's induced distribution,

$$\mathcal{L}(\hat{t},\theta) = \sum_i \log p(t_i = \hat{t}_i \,|\, \boldsymbol{t}_{<i} = \hat{\boldsymbol{t}}_{<i};\theta) \tag{A.3}$$

A.3.2 Embeddings

We use discrete embedding vectors for the token sequence as illustrated in Fig. A.2. Though it is possible to embed each vertex (x, y) with a single token by summing or concatenating each coordinate's embedding, [12] reported that this reduced performance significantly, and we therefore keep them embedded as separate tokens. It is also possible to remove the opcode tokens, however, [14] suggest that separator tokens allow the network to better propagate information between attention heads.

Some variants of our model take images as inputs, and these are embedded by feeding the pixel values through an off-the-shelf convolutional neural network, together with pixel coordinate embeddings from a learned linear transformation of the pixel coordinates. This is illustrated as an optional context branch in Fig. A.2. We evaluate two types of image embedding networks: first, as in [12], a pre-activation style ResNet [15], and second, the more recently proposed MLP Mixer [16]. Neither network is pretrained.

A.3.3 Turning Floor Plans into Token Sequences

In the KTH floor plan dataset, a floor plan $\mathcal{F}$ is defined as a set of *spaces*, with each space having a bounding polygon defined as a sequence of line segments. Each segment has a type, e.g., wall, window, or "portal", i.e., a connection to another space. For our purposes, we treat windows as walls, and portals are ignored.

We model an *in situ* agent partially observing an environment with some sensor, e.g., a LIDAR, at some location in the environment. This lets us target on-line estimation scenarios such as autonomous exploration or search-and-rescue. It also means the dataset is "over-sampled", in the sense that we generate many training samples from a single floor plan.

We sample *valid* locations inside a floor plan by rejection sampling N_P points $\mathcal{P}$ by sampling uniformly over the floor plan's bounding box until each sample are inside a space by some minimum clearance. We then maximize the distance between the *selected* points $\mathcal{S} \subset \mathcal{P}$ by iteratively constructing $\mathcal{S}_i = \mathcal{S}_{i-1} \cup \{\boldsymbol{s}_i\}$ from the selected point $\boldsymbol{s}_i$ at each iteration where

$$\boldsymbol{s}_i = \underset{\boldsymbol{p} \in \mathcal{P} \setminus \mathcal{S}_{i-1}}{\arg\max} \left(\min_{\boldsymbol{p}' \in \mathcal{S}_{i-1}} \|\boldsymbol{p} - \boldsymbol{p}'\|_2 \right) \tag{A.4}$$

until $|\boldsymbol{s}_i - \boldsymbol{s}_{i-1}| < d_{pmin}$. We initialize with $\mathcal{S}_1 = \{\boldsymbol{p}_1\}$ without loss of generality as all $\boldsymbol{p}_i$ are independent and identically distributed.

A.3.4 Canonicalization

To remove extraneous degrees of freedom, we canonicalize the line segments so the set of token sequences that produce identical visual result is as small as possible. First, let the canonical order of endpoints of a line segment from (x_0, y_0) to (x_1, y_1) be such that $x_0 < x_1$, and $y_0 < y_1$ in the case when $x_0 = x_1$. In other words, the segments are always oriented left-to-right, or bottom-to-top for vertical lines.

The line segments are centered and scaled by a global scale factor, determined by the dataset statistics, before being quantized. We extend the segments by considering any two overlapping parallel lines, then extending them to each other's furthest endpoints, resulting in one longer line. This yields a representation where each unbroken line segment is represented by only one actual segment. We then break each line segments at each intersection with another segment, so that no segments cross in their interior. Finally, we subdivide each segment so the longest segment length is bounded.

A.3.5 Partial Sequences

For each sample location s, the corresponding floor plan's line segments are ordered by their distance to s, and only the first (nearest) N_{segs} line segments are kept. The distance is computed as the Euclidean distance between the location and the closest point on the line segment. The segments are translated so that the point-of-view location is at the origin. Finally, only the N_{segs} nearest segments are kept and mapped to a sequence of tokens by inserting the appropriate opcodes.

A.3.6 Image Rasterization

A line drawing algorithm is used to rasterize the first $N_{raster} \ll N_{segs}$ segments into a black-and-white bitmap to approximate a partial birds-eye-view occupancy grid. We do not use antialiasing.

A.4 Experiments

In this section, we present both quantitative metrics and qualitative demonstrations of the model's output in different settings. In all experiments, the quantization level N_Q is 256, the sequence is at most $N_{segs} = 100$ segments, each segment is at most 2.5 m long, the nearest segment is at least 40 cm away, and the furthest segment is at most 7.5 m away.

A.4.1 Network Architecture and Training Details

We use six attention layers, E is 512, and N_{heads} is 8. See Section A.6. The models are trained with the Adam [17] optimizer on the KTH floor plan dataset, divided into a 90–10 split with a learning rate of $3 \cdot 10^{-4}$. The dropout rate was set to

Table A.1: Evaluation and comparison of selected models

Model	NLL (bits)	Top-1	Top-5
Uniform	8.02	0.4 %	1.9 %
Nearest Neighbor	||	59.7 %	63.3 %
Equivalent MLP	1.87	68.2 %	84.3 %
FloorGenT	1.09	81.4 %	91.8 %
- Opcode Tokens	1.12	81.0 %	91.5 %
- Position Embed.	1.34	77.8 %	90.0 %
FloorGenT Images	0.83	84.3 %	95.0 %
- Position Embed.	0.99	81.7 %	93.5 %
+ All Segm., ResNet	0.50	89.5 %	98.2 %
+ All Segm., MLP Mixer	0.40	91.4 %	98.9 %

60 % for all models. Each model was trained over 10^6 batches of size $N_B = 8$, after which the optimization has converged. All layers use ReZero [18], this accelerates convergence and allows deeper network architectures.

Data Augmentation Each floor plans is sampled at high density with multiple overlapping sequences as a form of data augmentation, which also preserves invariants imposed by the canonicalization such as segment order. We then mirror and rotate by a multiple of 90° at random using a *signed permutation*. There are exactly eight such signed permutation matrices, generated by enumerating all combinations of three primitive operations: mirror X axis, mirror Y axis, and swap X and Y axes.

Train-Test Split To avoid test data leaking into the training set, we split training and test before shuffling the set of sampled sequences. This ensures that a given floor plan is only in one of the splits. Since a building's floors can be similar or even identical, care must also be taken to ensure that a given *building* is only in one of the splits. In our case, the floor plans are ordered by the building they belong to, so such leakage is prevented by the same mechanism.

A.4.2 Predictive Performance on Test Set

The KTH floor plan dataset test set contains 2.46 million tokens with an average sequence length of 363 tokens. Likelihood and accuracy metrics of the models and variants described below are presented in Table A.1. Negative log likelihood (NLL) is reported on the test set as mean bits per token. Top-k accuracy is computed as the frequency with which the ground truth token value is in the k most likely tokens predicted by each model, with sequences from the test set. Top-1 is equivalent to the maximum likelihood estimate.

Uniform We report a worst case reference point with uniform probabilities. The NLL is $\log |\mathcal{T}|$, and the top-k accuracy follows a hypergeometric distribution.

Nearest Neighbors Predictions are formed by finding the N_k most similar subsequences in the training set, using the sliding window approach outlined in Section A.6.1 to construct subsequences of N_w tokens. The most common following token is then the prediction. Sequence similarity is measured by Hamming distance, i.e., the number of unequal tokens. We report performance with $N_w = 10$ and $N_k = 32$, tuned by hyperparameter search and cross-validated. NLL is not reported as the model is deterministic.

Equivalent MLP We ablate the attention mechanism entirely, turning the network into a multi-layer perceptron (MLP), refer to Section A.6.1 for architecture details. Notably, this uses nearly four times more network parameters. The MLP had the best performance of the baselines we measured, though not as good as its attention-based counterparts.

FloorGenT This is the standard formulation of the model as described in Section A.3.

Opcode Tokens We evaluate a model that is only given coordinate tokens and the `stop` opcode token in its input. We let $p(t_i = \hat{t}_i) = 1$ for the removed opcode tokens, since the parity of the triplet index is sufficient to determine where `move` and `line` opcodes should be inserted. We find that performance is largely unchanged from the standard formulation, though inference time is shorter since a third of the tokens have been removed.

Position Embeddings Line segments are the same regardless of their position in the token sequence, so their representation should arguably also be position invariant. We follow [19] and remove the position embeddings, so the model is unable to rely on statistics of the sequence position. We find that though this delayed overfitting somewhat, dropout was still necessary. The accuracy and NLL was unsurprisingly worse, as it is not possible to know which token was last (or anywhere) in the sequence without position embeddings, and it is therefore more difficult to predict what the next token will be — though not impossible since the segments are ordered by distance from the origin.

A.4.3 Novel and Partial Sequence Completion

Generative language models are often used to complete sentences with contextually relevant completions. In the same way, we can complete partial floor plan sequences, generating plausible continuations of partially observed floor plans. We present a set of such samples from our model in Fig. B.4.

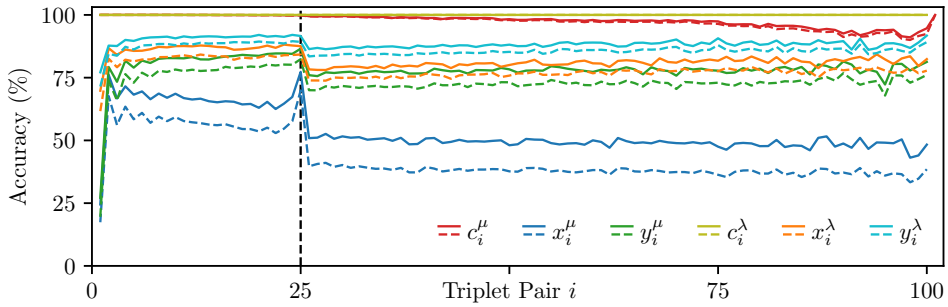


Figure A.4: Predicted probability (dashed lines) versus empirical accuracy (solid lines) per token position on the test set. The six plots correspond to token types when grouped into `move-line` triplet pairs denoted μ and λ. Each point on the horizontal axis thus corresponds to a line segment. These results are from the partial image model, and the dashed vertical line separates segments visible in the input image from those that are not. The plot for c_i^μ extends one index longer since it includes the final `stop` token.

Sampling is performed iteratively in an autoregressive manner. Given a partial or empty sequence, the network returns a distribution over possible next tokens, a sample is drawn from this distribution, and then added to end of the sequence. This iteration continues until the sampled token is the `stop` token, or the maximum iteration count i_{max} is reached. This process is illustrated in Fig. A.2. Nucleus sampling [20] is applied with top-p at 90 %, which works by moving probability mass from the "unreliable tail" of the token distribution to its *nucleus*.

A.4.4 Partial Image Conditioning

We condition the model on binary 128×128 pixel images by rasterizing the first 25 line segments. The image is intentionally made ambiguous: there a quarter as many pixels as there are quantized coordinates, and only a subset of the line segments are drawn into the image. The network must therefore both learn to reproduce line segments from a rendering of the 25 nearest, and generate up to 75 novel line segments that fit the first 25.

To see the effect of the image embedding network, we report the performance of two models where *all* line segments are rasterized, with two different networks, ResNets [15] and MLP Mixer [16]. The MLP Mixer variant is 40 % smaller, yet it has 20 % lower NLL. The results are reported together with the non-image results in Table A.1.

The predicted token probability by the model, and the empirically estimated accuracy grouped by triplet pairs is presented in Fig. A.4, where $c^\mu, x^\mu, y^\mu, c^\lambda, x^\lambda, y^\lambda$ refer to a pair of `move` and `line` triplets (denoted μ and λ respectively). We find the predicted probability to always be slightly less than the accuracy, and the

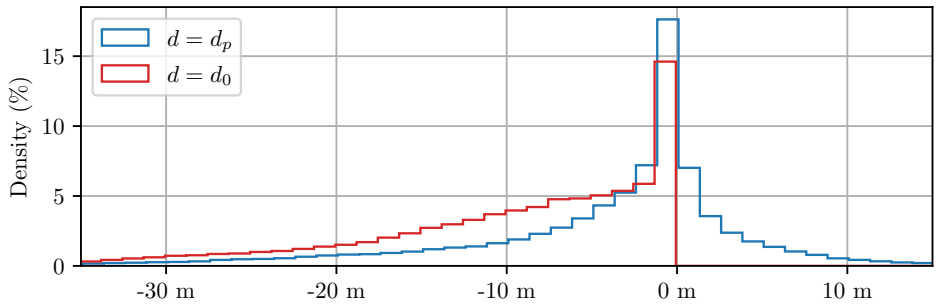


Figure A.5: Histogram of the distance error $\epsilon = d - \hat{d}$ with distances estimated in a completed floor plans from the first 25 line segments of the test data (d_p), compared to distances estimated using just the 25 first line segments themselves (d_0). $\hat{d}$ is the distance using the true floor plan. The histogram bins are eight cell sides wide. The two plots are shifted slightly horizontally to improve legibility.

difference is more pronounced when the probability is lower. Both drop sharply after the 25th triplet pair, which corresponds to the last visible line segment. We find that x^μ more or less determines the other three coordinates, suggesting that polar coordinates (ϕ, r) may be easier to model since the line segments are mostly ordered by r.

A.4.5 Predicting the Shortest Path

Finally, we evaluate the applicability of FloorGenT to a typical robotics task, namely, predicting the distance (and path) to an unknown point in space given environmental cues. The scenario is as follows: a robot observes its immediate surroundings, precisely the 25 nearest line segments at a location in the test set. FloorGenT is used to estimate the expected distance under the model given the observation via a Monte Carlo approximation. To that end, we generate 8 completions from the observation, and the completions are rasterized to create the occupancy grids $M_{p1}, \ldots, M_{p8}$. A shortest-path search from the origin to each cell in each occupancy grid is performed, with obstacles inflated by four cells. The predicted distance d_p for each cell is then defined to be the median distance of the 8 completions. The same process is repeated for the "null hypothesis" occupancy grid M_0, created by rasterizing only the observed 25 line segments, and also repeated for the true floor plan rasterization $\hat{M}$. The occupancy grids are 256×256 cells over an area of $40\,\text{m} \times 40\,\text{m}$. The search is 8-connected and uses Euclidean distance cost. We denote the distance under the null hypothesis d_0, and the distance in the true floor plan $\hat{d}$. A cell is considered trivial if $d_p = d_0 = \hat{d}$ and is left out of the following statistical evaluation. An example calculation is presented in Fig. A.6.

The mean absolute error $|\epsilon|$ is 6.21 m using predictions and 9.65 m (+3.44 m) under the null hypothesis, i.e., assuming that only the observed obstacles exist. A

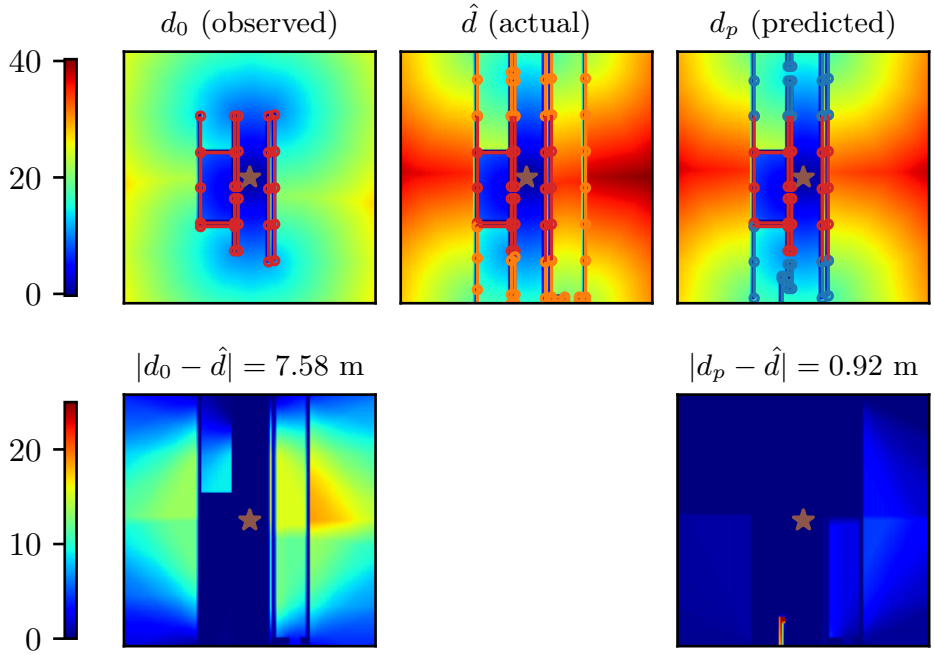


Figure A.6: An example of the three distance grids $d_0, \hat{d}, d_p$, and the absolute error $|\epsilon|$ per cell. All quantities are in meters. The corresponding line segments are drawn on top of each grid. Red line segments are observed, blue predicted, and orange is the true floor plan. The star denotes the origin, i.e., from where the distance is computed.

histogram of the errors is shown in Fig. A.5. Note that the error ϵ is better centered around zero with predictions, while the null hypothesis by definition results in only underestimating, i.e., $\epsilon \leq 0$ (mean ϵ is -4.03 m and -9.65 m respectively).

Though we have only looked at the median distance, the same approach could for example be used to gauge certainty by computing the variance, or to assert the satisfaction of safety parameters, e.g., "cell can be reached without running out of battery".

A.5 Conclusions

We have presented FloorGenT, a generative model for floor plans, and showed its potency in modelling and predicting large-scale floor plans like the KTH floor plan dataset. We demonstrated the model's ability to incorporate other data modalities, in particular, rasterized images similar to occupancy grids which is a common

representation in robotics applications, and showed its ability to reasoning about unknown space in a typical robotics task.

We believe there are many interesting real-world use cases for predictive floor plan modelling, and particularly in conjunction with sensor data. Concretely, we plan to employ map predictions in an autonomous exploration scenario as proposed in our previous work [C]. This could be accomplished with pretrained networks for the context embedding, and potentially combined with semi-supervised learning since there are relatively few datasets of indoor floor plans with sensory data, but abundant data from indoor scenes.

References

[1] Z. Zeng, X. Li, Y. K. Yu, and C.-W. Fu, "Deep floor plan recognition using a multi-task network with room-boundary-guided attention", in *International Conference on Computer Vision*, IEEE, 2019.

[2] A. van den Oord, N. Kalchbrenner, L. Espeholt, k. kavukcuoglu koray, O. Vinyals, and A. Graves, "Conditional Image Generation with PixelCNN Decoders", in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2016.

[3] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al., "Language models are unsupervised multitask learners", *OpenAI blog*, 2019. [Online]. Available: `https://openai.com/blog/better-language-models/`.

[4] M. Luperto, V. Arcerito, and F. Amigoni, "Predicting the Layout of Partially Observed Rooms from Grid Maps", in *International Conference on Robotics and Automation*, IEEE, 2019.

[5] M. Saroya, G. Best, and G. A. Hollinger, "Online exploration of tunnel networks leveraging topological cnn-based world predictions", in *International Conference on Intelligent Robots and Systems*, IEEE, 2020.

[6] L. Wang, H. Ye, Q. Wang, Y. Gao, C. Xu, and F. Gao, "Learning-based 3D Occupancy Prediction for Autonomous Navigation in Occluded Environments", in *International Conference on Intelligent Robots and Systems*, IEEE, 2021.

[7] A. Aydemir, P. Jensfelt, and J. Folkesson, "What can we learn from 38 000 rooms? Reasoning about unexplored space in indoor environments", in *Conference on Intelligent Robots and Systems*, IEEE, 2012.

[8] J. Chen, C. Liu, J. Wu, and Y. Furukawa, "Floor-SP: Inverse CAD for Floorplans by Sequential Room-Wise Shortest Path", in *International Conference on Computer Vision*, IEEE, 2019.

[9] A. Carlier, M. Danelljan, A. Alahi, and R. Timofte, "DeepSVG: A hierarchical generative network for vector graphics animation", *Advances in Neural Information Processing Systems*, 2020.

[10] P. Reddy, M. Gharbi, M. Lukac, and N. J. Mitra, "Im2Vec: Synthesizing vector graphics without vector supervision", in *Conference on Computer Vision and Pattern Recognition*, IEEE, 2021.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding", 2018. arXiv: `1810.04805`.

[12] C. Nash, Y. Ganin, S. A. Eslami, and P. Battaglia, "Polygen: An autoregressive generative model of 3D meshes", in *International Conference on Machine Learning*, PMLR, 2020.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need", in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2017.

[14] G. Weiss, Y. Goldberg, and E. Yahav, "Thinking Like Transformers", in *International Conference on Machine Learning*, PMLR, 2021.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks", in *European Conference on Computer Vision*, Springer, 2016.

[16] I. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. P. Steiner, D. Keysers, J. Uszkoreit, M. Lucic, and A. Dosovitskiy, "MLP-Mixer: An all-MLP Architecture for Vision", in *Advances in Neural Information Processing Systems Pre-proceedings*, Curran Associates, Inc., 2021.

[17] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization", in *International Conference on Learning Representations*, 2015.

[18] T. Bachlechner, B. P. Majumder, H. Mao, G. Cottrell, and J. McAuley, "ReZero is All You Need: Fast Convergence at Large Depth", in *Uncertainty in Artificial Intelligence*, PMLR, 2021.

[19] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh, "Set transformer: A framework for attention-based permutation-invariant neural networks", in *International Conference on Machine Learning*, PMLR, 2019.

[20] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The Curious Case of Neural Text Degeneration", in *International Conference on Learning Representations*, Curran Associates, Inc., 2019.

[C] L. Ericson, D. Duberg, and P. Jensfelt, "Understanding Greediness in Map-Predictive Exploration Planning", in *European Conference on Mobile Robots*, IEEE, 2021. DOI: 10.1109/ECMR50962.2021.9568793.

[21] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization", 2016. arXiv: 1607.06450.

A.6 Appendix: Network Architecture Details

An attention layer is defined as $\boldsymbol{y} = \text{AttnLayer}(\boldsymbol{x}_0)$ by

$$\begin{aligned}\boldsymbol{x}_1 &= \boldsymbol{x}_0 + \alpha_1 \,\mathrm{D}_r(\text{MHA}(\overline{\boldsymbol{x}_0}, \overline{\boldsymbol{x}_0}; N_{heads}, E)) \\ \boldsymbol{x}_2 &= \boldsymbol{x}_1 + \alpha_2 \,\mathrm{D}_r(\text{MHA}(\overline{\boldsymbol{x}_1}, \boldsymbol{v}_{context}; N_{heads}, E)) \\ \boldsymbol{y} &= \boldsymbol{x}_2 + \alpha_3 \,\mathrm{D}_r(\text{Linear}(\boldsymbol{x}_2))\end{aligned}$$

where $\text{Linear}(\boldsymbol{z}_0) = \boldsymbol{z}_2$ with

$$\begin{aligned}\boldsymbol{z}_1 &= \text{Dense}(\overline{\boldsymbol{z}_0}; N_{fc}) \\ \boldsymbol{z}_2 &= \text{Dense}(\text{ReLU}(\boldsymbol{z}_1); E)\end{aligned}$$

The network is largely as proposed by [13]. D_r is dropout at rate r. $\overline{\boldsymbol{x}}$ is layer normalization as in [21]. $\text{MHA}(\boldsymbol{q}, \boldsymbol{m})$ is multiple heads of scaled dot-product attention as in [13], with queries $\boldsymbol{q}$, keys and values $\boldsymbol{m}$. Dense is a fully-connected layer with given number of output units. α_i are the ReZero coefficients. N_{heads} is the number of attention heads. E is the embedding dimension. $\boldsymbol{v}_{context}$ is the context embedding, Cf. image embedding in Fig. A.2. For non-image models, we let $\boldsymbol{x}_2 = \boldsymbol{x}_1$. Each successive layer operates on the output of the previous, so the output $\boldsymbol{y}_{net}$ of an L-layer network is defined

$$\begin{aligned}\boldsymbol{y}_0 &= \textit{sequence embedding (Cf. Fig. A.2)} \\ \boldsymbol{y}_i &= \text{AttnLayer}(\boldsymbol{y}_{i-1}) \\ \boldsymbol{y}_{net} &= \overline{\boldsymbol{y}_L}\end{aligned}$$

Finally, $\boldsymbol{y}_{net}$ is projected to $|\mathcal{T}|$ dimensions to obtain the logits of the predictive distribution $p(t_i|\boldsymbol{t}_{<i}; \theta)$.

A.6.1 Equivalent MLP

In the "Equivalent MLP" model, we take N_w subsequences of each input sequence in a sliding window fashion, e.g., the sequence *abcdef* would under a sliding window with $N_w = 3$ yield four subsequences: *abc, bce, cde, def*. Padding items are prepended so that we obtain as many subsequences as there are tokens in the input, e.g., *ssabcdef* with a start token *s*. The network input $\boldsymbol{y}_0$ and output $\boldsymbol{y}_{net}$ is as before, but with the preprocessing step $\boldsymbol{y}_1$ and a different layer definition $\boldsymbol{y}_i$

$$\begin{aligned}\boldsymbol{y}_1 &= \text{Join}(\text{SlidingWindow}(\boldsymbol{y}_0); N_w{\cdot}E) \\ \boldsymbol{y}_i &= \text{MLP Layer}(\boldsymbol{y}_{i-1})\end{aligned}$$

where SlidingWindow is said sliding window operation and yields a $S{\times}N_w{\times}E$ tensor for input length S. Join flattens the last two dimensions as indicated, i.e.,

a concatenation of the embeddings in each window. A single layer is defined $\boldsymbol{y} = \text{MLP Layer}(\boldsymbol{x}_0)$ with

$$\begin{aligned}
\boldsymbol{x}_1 &= \text{Dense}(\overline{\boldsymbol{x}_0}; N_w \cdot N_{fc}) \\
\boldsymbol{x}_2 &= \text{Split}(\boldsymbol{x}_1; N_w \times N_{fc}) \\
\boldsymbol{x}_3 &= \text{Dense}(\text{ReLU}(\boldsymbol{x}_1); E) \\
\boldsymbol{x}_4 &= \text{Join}(\boldsymbol{x}_3; N_w \cdot E) \\
\boldsymbol{y} &= \boldsymbol{x}_0 + \alpha \, \text{D}_r(\boldsymbol{x}_4)
\end{aligned}$$

Split is the inverse of Join and unflattens the two last dimensions as indicated. The above setup mimics the attention layers: the first dense layer can "attend" to anything in the given window, and the second dense layer projects each position individually to E dimensional embeddings. In our experiments, we have six MLP layers, $E = N_{fc} = 12$, and $r = 5\,\%$.

Paper B

Beyond the Frontier: Predicting Unseen Walls from Occupancy Grids by Learning from Floor Plans

L. Ericson and P. Jensfelt
KTH Royal Institute of Technology

Abstract

In this paper, we tackle the challenge of predicting the unseen walls of a partially observed environment as a set of 2D line segments, conditioned on occupancy grids integrated along the trajectory of a 360° LIDAR sensor. A dataset of such occupancy grids and their corresponding target wall segments is collected by navigating a virtual robot between a set of randomly sampled waypoints in a collection of office-scale floor plans from a university campus. The line segment prediction task is formulated as an autoregressive sequence prediction task, and an attention-based deep network is trained on the dataset. The sequence-based autoregressive formulation is evaluated through predicted information gain, as in frontier-based autonomous exploration, demonstrating significant improvements over both non-predictive estimation and convolution-based image prediction found in the literature. Ablations on key components are evaluated, as well as sensor range and the occupancy grid's metric area. Finally, model generality is validated by predicting walls in a novel floor plan reconstructed on-the-fly in a real-world office environment.

 This work was supported by the Swedish Research Council. All authors are with the Division of Robotics, Perception and Learning at KTH Royal Institute of Technology, Stockholm, SE-10044, Sweden. For e-mail correspondence, contact ludv@kth.se.

B.1 Introduction

Human and robotic problem-solving approaches differ in dealing with the unknown and predicting the near future. Classical robotic approaches seek exactness at the cost of intuition and foresight, and on the contrary, humans do not meticulously maintain metric maps of their worlds. Even schematics and blueprints, documents explicitly intended to specify technical details, leave room for interpretation. Abstraction seems necessary for our ability to move between the specifics of reality and the generality of ideas and ideals. Replicating this ability to reason abstractly with explicit algorithms has historically proven difficult, but recent advances in learning-based approaches have opened up new avenues and great strides have been made across many subfields of robotics.

In this work, floor plans are used as the medium through which abstract reasoning is made possible. Floor plans are the architectural blueprints of our built environment, a distillate of the real world as a tidy set of shapes and symbols encoding the layouts and purposes of rooms, positions of walls, doorways, and windows. Floor plans obey rules, symmetries, and regularities that are impossible to state explicitly, often driven by aesthetic considerations rather than logic. We have previously shown that recent advances in autoregressive language models can be leveraged to produce a generative model over floor plans as sequences of vector graphic instructions [A]. By contrast to single-shot approaches such as [1], [2], an autoregressive approach casts the floor plan generation task as a series of decisions and their consequences, enabling the model to reason in steps, analogous to the chain-of-thought paradigm in large language models [3].

Autonomous exploration planning is an obvious example of a classical robotics problem where such a prediction model should be immediately applicable; however, we have previously shown that traditional non-predictive exploration planners are not well-suited to using predictions, and predictions can actually have a negative impact on exploration performance [C]. In this article, we have limited the scope to evaluating predictions by the primary variable that they affect in the exploration context: *predicted information gain*.

This paper is structured as follows. In Section B.4, an model for predicting floor plans from sensor history, dubbed *Floorist*, is defined. Data modality is the main difference from our previous model [A], which dealt solely with abstract floor plans. Floorist is instead grounded in the real world by taking a partially observed environment as input in the form of 2D occupancy grids from a 360° LIDAR sensor and predicting the unobserved walls of that environment as line segments. In Section B.5, a dataset generation method is outlined wherein a virtual robot navigates between randomly sampled waypoints in a collection of annotated floor plans, generating input occupancy grids and their target wall segments. In Section B.6, cluster-based predicted information gain is defined as in [4], [5], it is the evaluation metric used in this work, suitable for occupancy grid-based prediction models. In Sections B.7 and B.8, three prediction models are evaluated: Floorist, a baseline convolution-based architecture as in [1], [2], [5], [6], [7], and a non-predictive

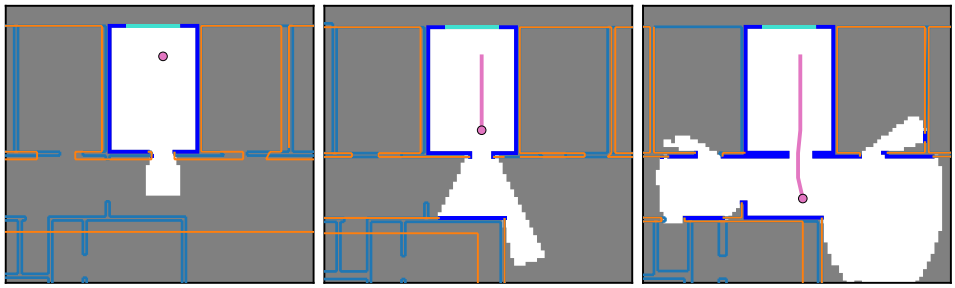


Figure B.1: Three consecutive occupancy grids with Unknown, Free, Occupied, and Window cells; Predicted walls from a Floorist model; Target walls; and Trajectory. Initially (left), few lines match up exactly, apart from the northern exterior wall which is visible, and the predicted rooms do exist though not in the exact locations predicted. In the next step (middle), more information about the corridor is observed, and the predicted segments are also improved, though the adjoining rooms are still misaligned and their doorways misplaced. Finally (right), the adjacent rooms are partially observed, and the predicted doorway alignment is now correct, and both room's widths are correctly adjusted. The images have been cropped for legibility.

approach as in [8]. Floorist performance under ablation of key components is also reported, as well as its sensitivity to sensor range and occupancy grid area. Finally, in Section B.8.4, Floorist is applied to an on-the-fly floor plan reconstruction of a real-world office environment to validate its generality.

The dataset generation software along with training and inference code for Floorist is published in tandem with this work under an open-source license at `lericson.se/floorist`.

B.2 Related Work

The approach of predicting the unknown from occupancy grids is perhaps most common in the autonomous exploration literature. Autonomous exploration is the task of reconstructing a map of an environment, typically with no prior information about that particular environment, e.g., [4], [9], [10], though not always [11]. In *frontier-based* exploration planners, the planner considers *frontiers* by spatial clustering of the boundary between free and unknown space. A score is assigned to each frontier, and the planner navigates to the highest-scoring frontier. The score is typically a function of the travel distance and the predicted information gain, an estimate of how many bits of new information will be obtained by visiting that frontier. A popular predictive approach, e.g., [1], [2], [5], [6], [7], is to derive training data directly from autonomous exploration planning and then train some neural network to predict a 2D occupancy grid, computing the predicted information gain

from the occupancy grid prediction. In sampling-based exploration planners such as [8], [12], [13], path and exploration planning are performed simultaneously with some variant of RRT [14]. Some sampling-based works predict information gain directly by a regression formulation, with a deep network [13] or a Gaussian process [12]. In [6], a reinforcement learning-based approach is proposed paired with a convolutional network for occupancy grid prediction; [7] extends the approach to a real-world exploration system in the form of a micro-aerial vehicle.

Another line of inquiry is the explicit reconstruction of floor plans from sensor data for architectural purposes, typically in an offline setting. In [15], [16], [17] use an attention-based network to approach the task of predicting a cohesive floor plan given a set of point clouds covering the entire environment. [18] solve a similar task, though using sparse multi-views instead of point clouds. Their approach is to model the environment topologically before generating a floor plan suitable to those topological constraints. In [19], the aim is somewhere between offline and online prediction, posing the problem of inferring what is behind closed doors given an occupancy grid of the observable parts of the environment, i.e., after exploration is completed. The authors demonstrate favorable performance in path planning tasks into unknown space, behind closed doors.

Some works concerning autonomous driving take a similar approach to ours in representing and predicting the environment as a set of line segments with a neural network [20], [21]. Emphasis is placed on reconstruction of the visible environment as opposed to reasoning about the unknown, though many of the technical challenges are similar. In the category of learning-based approaches on vector graphics, the approach is typically to embed an entire graphic to enable downstream tasks on that embedding, such as animation and interpolation [22], [23].

In [24], a topological approach to modeling and reasoning about indoor environments is taken. It also introduces the *KTH floor plan dataset*, which is used for synthesizing training data in the present work. It is a collection of floor plans from a university campus, annotated with positions of walls, doors, and windows.

B.3 Notation

$\langle \cdot \rangle$ denotes expectation under some probability distribution. $[\cdot]$ denotes the Iverson bracket, sometimes called the indicator function. It maps the truth value of a proposition to one or zero. $\langle [x = 1] \rangle$ thus denotes the probability that $x = 1$. $\lfloor x \rfloor$ denotes the floor function, i.e., the greatest integer less than or equal to x.

B.4 Probabilistic Model

The problem of predicting floor plans is formulated as an autoregressive prediction task on sequences of line segment vertices. The perspective is an in-situ robot located at the origin having observed some part of its surroundings, and the task is to predict the walls of the floor plan beyond what has already been observed, as

illustrated in Fig. B.1. The floor plan is represented as a matrix $\boldsymbol{S} \in \mathbb{R}^{N\times 4}$ of the N target line segments from (x, y) to (x', y'), i.e.,

$$\boldsymbol{S} = \begin{pmatrix} x_1 & y_1 & x'_1 & y'_1 \\ x_2 & y_2 & x'_2 & y'_2 \\ \vdots & \vdots & \vdots & \vdots \\ x_{\mathrm{N}} & y_{\mathrm{N}} & x'_{\mathrm{N}} & y'_{\mathrm{N}} \end{pmatrix} \tag{B.1}$$

The vector graphic produced by $\boldsymbol{S}$ is invariant to row-wise permutations, and vertex order reversal. However, in an autoregressive regime, order does matter and must be chosen carefully for two reasons. First, any function that models an ordered sequence must by necessity subsume the ordering algorithm. Secondly, autoregression implies that later positions in the sequence are informed by earlier positions. We use a proximity heuristic where the rows are ordered by the distance from the robot to the nearest point on each line segment, as in [A]. The intuition is that segments near the robot are often partially observed, and can be predicted using the occupancy grid as grounding. Later segments that are far away and ungrounded can then be chosen to fit with the nearby grounded segments. For vertex order, a simple strategy suffices. The vertices of each segment are ordered lexicographically, i.e.,

$$(x < x') \vee ((x = x') \wedge (y \leq y')) \tag{B.2}$$

B.4.1 Sequence Tokenization and Factorization

The line segment prediction problem is cast as a sequence prediction problem on the *token sequence* $\boldsymbol{t}$ where the joint distribution is conditioned on some contextual input C, that is

$$\mathrm{p}(\boldsymbol{S} \mid C) = \mathrm{p}(\boldsymbol{t} \mid C) \tag{B.3}$$

The sequence $\boldsymbol{t}$ is initialized and terminated by a 'start' and 'end' token respectively, and otherwise consists of vertex pairs, each vertex is quantized by $q : \mathbb{R}\times\mathbb{R} \to \mathcal{V}$ with the index set $\mathcal{V} = \{1, 2, \ldots, HW\}$ and

$$q(x, y) = \lfloor W(\tfrac{1}{2}H - s_y y) + (\tfrac{1}{2}W + s_x x) \rfloor \tag{B.4}$$

for some $H\times W$ grid at scale s_x, s_y in cells/m. The function from segments $\boldsymbol{S}$ to tokens $\boldsymbol{t}(\boldsymbol{S}) \in \mathcal{T}^{2N+2}$ with the token vocabulary $\mathcal{T} = \{\text{'start'}, \text{'end'}\} \cup \mathcal{V}$ is then defined

$$\begin{aligned} \boldsymbol{t}(\boldsymbol{S}) = (\text{'start'}, & q(x_1, y_1), q(x'_1, y'_1), \\ & q(x_2, y_2), q(x'_2, y'_2), \ldots \\ & q(x_{\mathrm{N}}, y_{\mathrm{N}}), q(x'_{\mathrm{N}}, y'_{\mathrm{N}}), \text{'end'}) \end{aligned} \tag{B.5}$$

The joint probability is factorized autoregressively as in the recursion

$$\begin{aligned} \mathrm{p}(\boldsymbol{t}_{[j\leq i]} \mid C) &= \mathrm{p}(t_i \mid \boldsymbol{t}_{[j<i]}, C)\ \mathrm{p}(\boldsymbol{t}_{[j<i]} \mid C) \\ \mathrm{p}(\boldsymbol{t}_{[j\leq 1]} \mid C) &= [t_1 = \text{'start'}] \end{aligned} \tag{B.6}$$

where t_i denotes the ith token, $\boldsymbol{t}_{[j<i]}$ and $\boldsymbol{t}_{[j\leq i]}$ denote the subsequence of $\boldsymbol{t}$ up to but excluding (or including) i. The next-token distribution is parameterized by logits from a deep network $\boldsymbol{f}_\theta$, i.e.,

$$\mathrm{p}(t_i \mid \boldsymbol{t}_{[j<i]}, C) = \sigma(\boldsymbol{f}_\theta(\boldsymbol{t}_{[j<i]}, C))_i \tag{B.7}$$

where $\sigma(\cdot)_i$ is the ith element of the normalized logistic function. The function $\boldsymbol{f}_\theta$ assigns probability mass to the true token sequence $\hat{\boldsymbol{t}}$ by gradient descent on $\langle L \rangle$, the expected negative log likelihood under the data distribution, with

$$L_\theta(\hat{\boldsymbol{t}}, C) = -\sum_{\hat{t}_i \in \hat{\boldsymbol{t}}} \log \sigma(\boldsymbol{f}_\theta(\hat{\boldsymbol{t}}_{[j<i]}, C))_{\hat{t}_i} \tag{B.8}$$

B.4.2 Contextual Inputs

The contextual input C consists of two parts: a partial occupancy grid $\boldsymbol{M} \in \mathcal{C}^{H\times W}$, and the *visible line segments* $\boldsymbol{S}(\boldsymbol{M}) \in \mathbb{R}^{N\times 4}$ recovered from the occupancy grid. The cell labels are

$$\mathcal{C} = \{\text{‘unknown’}, \text{‘free’}, \text{‘occupied’}, \text{‘window’}\} \tag{B.9}$$

The label ‘window’ indicates that a cell contains an *exterior window*, i.e., a window facing outside the building, and hints the floor plan’s perimeter. Marching squares [25] is used to find the visible line segments $\boldsymbol{S}(\boldsymbol{M})$. In regions where at least one cell is marked ‘unknown’, no line segment is produced.

B.4.3 Network Architecture

Following success in the language modeling domain, the choice of f_θ in Eq. (B.7) is a transformer encoder-decoder with multi-head attention (MHA) as in [27], illustrated in Fig. B.2. The encoder is computed once per sequence to encode the contextual input, while the decoder is iteratively re-evaluated step-by-step during sampling as the generated sequence is constructed. Unlike [27], each transformer layer is a residual layer with a gated residual as in [26]. The two modalities of the contextual input are tokenized separately. The occupancy grid $\boldsymbol{M}$ is encoded using a ViT network [28], and the tokenized visible line segments $\boldsymbol{t}(\boldsymbol{S}(\boldsymbol{M}))$ are encoded with a discrete embedding with absolute position encoding. The context tokens are concatenated along the sequence dimension and cross-attended to.

Chromatization Cell labels are mapped to 3-channel pseudo-colors by $k : \mathcal{C} \to \mathbb{R}^3$ with

$$\begin{aligned} k(\text{‘unknown’}) &= [-1;-1;-1] & k(\text{‘free’}) &= [-1;+1;-1] \\ k(\text{‘occupied’}) &= [-1;-1;+1] & k(\text{‘window’}) &= [+1;-1;+1] \end{aligned}$$

The exact choice of pseudo-colors is inconsequential as long as they are unique.

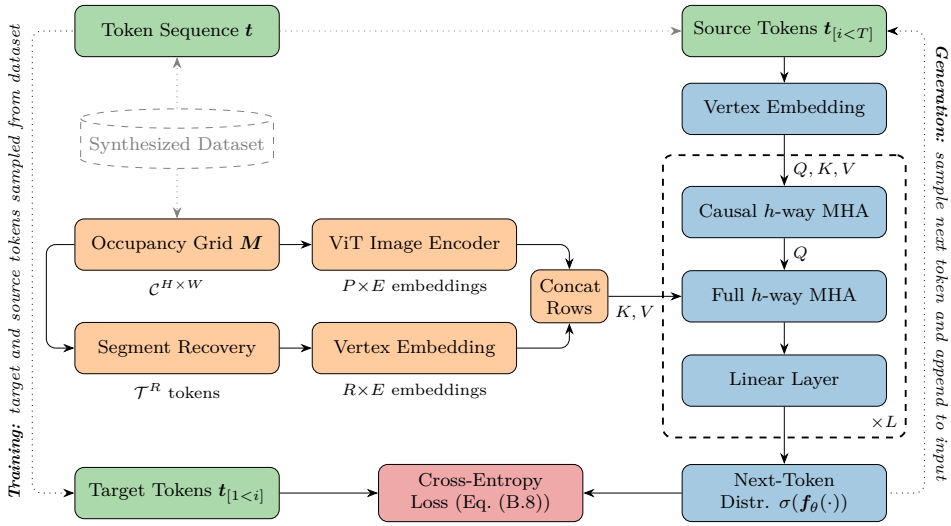


Figure B.2: The Floorist network architecture with training and generation pathways. In training, source and target token sequences are samples from the dataset, where Encoder blocks and Decoder blocks are evaluated in one pass. In generation, the source sequence starts as 'start' and the encoder side is only evaluated once. The decoder is then evaluated to obtain the next-token distribution, and a token is sampled from it and appended to the source sequence before the process repeats again. $R = |\boldsymbol{t}(\boldsymbol{S}(\boldsymbol{M}))|$ is the number of tokens in the line segments visible in $\boldsymbol{M}$, $T = |\hat{\boldsymbol{t}}|$ is the number of target tokens. P is the number of patches from ViT, E is the embedding dimension, Q, K, V are the query, key, and value matrices for a multi-head attention (MHA) layer. $\boldsymbol{t}_{[i<T]}$ denotes removal of the last token (i.e., 'end'), and $\boldsymbol{t}_{[1<i]}$ denotes removal of the first token (i.e., 'start'). This shifts the two sequences so that the target is always the next token. Note that each attention block is a gated residual connection as in [26].

Image Encoding The chromatized image is fed through a ViT network that produces a set of image tokens by considering the image as a set of non-overlapping patches which are linearly projected to the embedding dimension E, and then fed through a stack of self-attention layers. The ViT network is not pretrained.

Sequence Embedding Each token value $t_i \in \mathcal{T}$ is mapped to a learnable embedding vector, along with a learned absolute position embedding for each i. The two embeddings are summed to form the final token embedding.

B.4.4 Image-Based Formulation

Our baseline is an image-based formulation as in [1], [2], where a chromatized occupancy grid $\boldsymbol{k}(\boldsymbol{M})$ is the input and the target is a rasterized image of the target line segments. The outputs are per-cell occupancy logits Y_{ij}. At inference time, the occupancy state O_{ij} is the maximum likelihood estimate

$$O_{ij} = [Y_{ij}(\boldsymbol{k}(\boldsymbol{M})) < 0] \tag{B.10}$$

Marching squares is used to recover the predicted line segments. The loss function is per-pixel binary cross-entropy.

B.5 Data Synthesis

Similar to our previous work, we derive training and test data from the KTH floor plan dataset [24]. Each floor plan is represented as a set of rooms, each room being a polygon of line segments categorized as walls, doors or windows. For our purposes, only two boolean properties of the segment categories are considered: *transparent* and *passable*. Doors are assumed to be open and so are both transparent and passable; windows are transparent but impassable; and, walls are non-transparent and impassable. The segments are collected into two sets, one with impassable segments used to generate paths in the floor plan, and one with non-transparent segments used for simulating sensor occlusions. Note that windows are not only exterior windows, but also glass walls which are common in office environments such as the KTH floor plan dataset.

Waypoints are first sampled by farthest point sampling [29] inside each floor plan. Paths are then generated by Dijkstra's algorithm [30] between every pair of waypoints with a cost based on the truncated distance to the nearest wall, so that some minimum wall clearance is maintained where possible. A virtual sensor is then simulated along each path at a predefined step length, yielding a partial occupancy grid at each step of the path built from the scans up to that step by ray marching. Incident cells are first marked 'free', and the terminal cells are then marked 'occupied' or 'window' if the ray is a *hit*, i.e., terminated before the sensor's maximum range. Each occupancy grid is accompanied by its target line segments from the floor plan. The target segments are filtered by removing subsegments that lie inside 'occupied' cells.

B.5.1 Axis Alignment

The occupancy grid axes are automatically rotated to coincide with the visible line segments, which adds invariance in the world-to-robot rotation up to an integer multiple of 90°. Alignment also improves fidelity, as straight lines in the environment are rotated to coincide with either the row or column axis of the occupancy grid which minimizes discretization error, which is important for recovering the visible line segments. Following [31], a histogram is computed over the angle of

the line passing through each pair of neighboring LIDAR points, modulo 90°. The robot-to-surroundings rotation α is then computed by an average of the histogram's mode and the two adjacent bins, weighted by frequency. Finally, the LIDAR scans and target line segments are rotated by $-\alpha$ to align them to the occupancy grid axes before the occupancy grid is rendered.

B.5.2 Subdividing Segments

Subdividing segments serves to simplify the prediction problem by allowing long segments to be predicted piece by piece. The subdivision algorithm must be reproduced by the model function f_θ, so it should be simple and *predictable*. Furthermore, increasing the sequence by padding tokens improves model performance [32], [33], suggesting that shorter segments should improve performance, ceteris paribus. In practice, it becomes a compromise between model performance versus inference speed and memory usage.

Our solution is to subdivide the segments by superimposing a 21×21 regular grid and cutting the segments where they intersect this grid. The effect is that from the model's perspective, the segments are always subdivided at the same predefined coordinates.

B.6 Predicted Information Gain

Evaluating generative models is notoriously difficult as there is typically no way to quantify the quality of novel generations, and the measure of quality is often subjective. A common approach to this issue is to evaluate the generation quality by measuring feature statistics from a different neural network [34], [35]. As argued by [36], generative models are best evaluated by an intended application, which in our case is to predict information gain in frontier-based autonomous exploration. To define information gain, a probabilistic interpretation must be made of the occupancy grid $\boldsymbol{M}$. For this definition, independence between cells is assumed, and a categorical probability distribution is defined for the label of each cell by

$$\mathrm{p}(M_{ij} = k) = \begin{cases} |\mathcal{C}|^{-1} & \text{if } M_{ij} = \text{`unknown'} \\ [M_{ij} = k] & \text{otherwise} \end{cases} \tag{B.11}$$

If a cell is unknown, its label distribution is uniform, otherwise, the distribution is an indicator function of the label, with $M_{ij} \in \mathcal{C}$ as defined in Eq. (B.9). Let $\boldsymbol{M'}$ denote the occupancy grid $\boldsymbol{M}$ after sensor data integration at some frontier. Information gained in $\boldsymbol{M'}$ given $\boldsymbol{M}$ is then defined as

$$I(\boldsymbol{M'} \mid \boldsymbol{M}) = \sum_{ij,k} \mathrm{p}(M'_{ij} = k) \log \frac{\mathrm{p}(M'_{ij} = k)}{\mathrm{p}(M_{ij} = k)} \tag{B.12}$$

Notice that terms where $M_{ij} = M'_{ij}$, or $M'_{ij} =$ 'unknown', become zero. Assuming known cells in $\boldsymbol{M}$ remain the same label, then the only non-zero terms will be

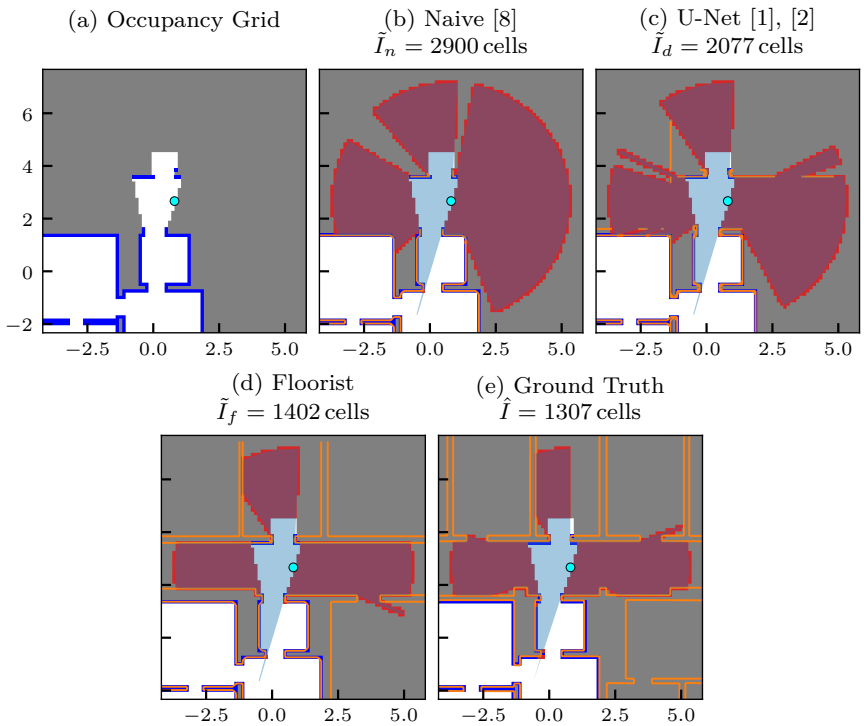


Figure B.3: Illustration of how information gain is computed for a ● Frontier location found along —● Trajectory. The initial occupancy grid $\boldsymbol{M}$ is shown in (a), while (b) to (e) show the occupancy grid $\boldsymbol{M}'$ after a simulated sensor scan in the predicted environment has been integrated into $\boldsymbol{M}$, with ▬ Information gain cells, ▬ Sensor scan, and — Walls. Occupancy grid colors as in Fig. B.1. In (b), walls are extracted from (a), corresponding to the typical way information gain is estimated for a frontier in non-predictive autonomous exploration [8]. In (c), walls are predicted by a model using (a) as input to a U-Net predictor as in [1], [2], and in (d) with the method proposed in this paper. In (e), the ground truth walls are used. The naive $\tilde{I}_n$, convolutional $\tilde{I}_d$, and our approach $\tilde{I}_f$ differ from the ground truth $\hat{I}$ by 1593, 770, and 95 cells (relative difference 122 %, 58.9 %, and 7.26 %) respectively.

those where M_{ij} is unknown and M'_{ij} is known. In other words, Equation (B.12) is proportional to the number of unknown-turned-known cells, which is the definition of information gain used in this and other works, e.g., [5].

Each occupancy grid cell is classified as either a frontier or non-frontier based on its 4-connected neighbors. Specifically, a cell is deemed a frontier cell if it is free, and amongst its four neighboring cells there exists both free and unknown cells.

Following this classification, the DBSCAN algorithm [37] is used to group these frontier cells into frontier clusters. The frontier location is the location of the cell closest to the average cell location.

B.7 Experimental Setup

Since the KTH floor plan dataset represents rooms as closed polygons, there are no connecting line segments where rooms connect, e.g., doorways, making it possible to traverse and see inside the walls at these doorways. A heuristic is used to insert wall segments in such situations: find a bijection from each door segment uv to its nearest door segment $u'v'$ and insert two wall segments uu' and vv' so that a quadrilateral is formed, if $\|uu'\| + \|vv'\| < 2\,\text{m}$. After this, each set of segments is canonicalized by first merging nearly identical vertices (within 1 mm), then joining overlapping line segments, and finally removing vertices that are not corners; i.e., vertices v with exactly two neighbors u, w that form a straight line uw through v.

The generated paths are filtered by length and number of turns, only keeping paths between 5 m to 100 m long and having at least 3 turns. The virtual sensor returns 720 points per scan, and a scan is obtained every 80 cm along the path. A cell is 'window' if a hit inside it is within 10 mm from the nearest exterior window segment. A window segment is classified as exterior if both its vertices are within some threshold distance (100 mm) from the building perimeter.

Frontier clusters with less than 3 cells are excluded from evaluation, as are those whose center lies within 5 cells of the edge of the occupancy grid. Cluster larger than 30 cells are divided into subclusters by k-means [38].

There are on average 75 context line segments and 165 target line segments per sample in 6 392 919 samples from 164 floor plans. Each occupancy grid represents a $15{\times}15\,\text{m}^2$ area in $121{\times}121\,\text{cells}^2$. The vertex quantization function Eq. (B.4) is identical in scale and size to the occupancy grid, i.e., $\frac{H}{s_y} = \frac{W}{s_x} = 15\,\text{m}$ and $H = W = 121$. The maximum sensor range is $r = 4.5\,\text{m}$ unless otherwise stated.

B.7.1 KTH Dataset Considerations

It is important to note that the KTH floor plan dataset contains duplicated floor plans, and that there are strong similarities between floors of a single building. It would therefore be an error to simply shuffle the entire dataset, as this would contaminate the training set with test data. As in [A], we deduplicate the KTH floor plan dataset and split it into training and test *before* shuffling, and adjust the splitting point such that a single building's floor plans are all exclusively training or test data, preventing any cross contamination.

B.7.2 Model Hyperparameters

In all experiments, the embedding dimension $E = 512$. AdamW [39] is used with weight decay 1×10^{-2}, learning rate 1×10^{-4}, and batch size 6. A 10 % dropout is

also applied. Since the dataset is large compared to the model size, the network can be trained indefinitely without overfitting; training was stopped when the validation loss stopped decreasing. The occupancy grids, visible line segments, and target line segments are jittered by one of the eight symmetries of the square, i.e., a random combination of mirroring and rotating. Performance was not improved by relaxed regularization. Top-p sampling [40] is used for all generation, with $p = 80\,\%$. Other choices of p were evaluated but yielded worse results. No temperature scaling or repetition penalty is applied.

B.8 Experimental Results

In Figs. B.4 and B.5, examples of the generated wall segments are shown for six continuous steps of randomly sampled trajectories in the test set. Frontier locations used when evaluating predicted information gain are also shown. Results from a model with larger area but same network size is also shown, showing less coherent output, as the prediction task is harder; e.g., there are segments passing through free space, a mistake that the smaller model does not tend to make. Finally, results from the image-based predictor are also presented, illustrating the difficulty in a pixel-wise approach to occupancy regression.

B.8.1 Predicted Information Gain

Predicted information gain is evaluated using the following different sets of wall segments, illustrated in Fig. B.3:

Naive: Assume that only what has been observed to be occupied is occupied, i.e., no segments are predicted and only the walls visible in the occupancy grid occlude the sensor. This is a common approach in non-predictive exploration planning [8], and is equivalent to assuming that unknown space can be considered sensor transparent.

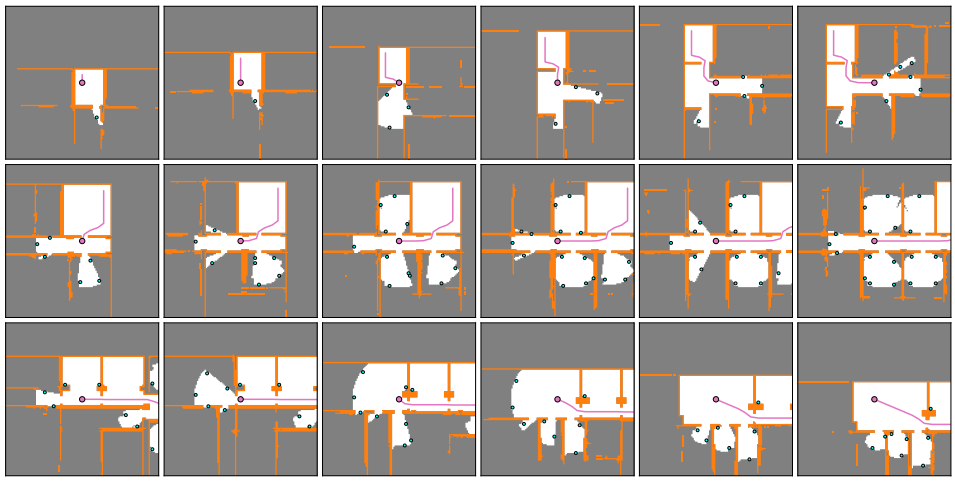
U-Net: U-Net [41] is a fully-convolutional image segmentation network, chosen as a baseline as it has been used in previous work on occupancy grid prediction for autonomous exploration [1], [2]. The network was not pretrained. Several network architectures were evaluated for image-based prediction [6], [28], [41], [42], [43], [44]. All architectures reached convergence at the same loss value, suggesting that convergence is caused by the intrinsic difficulty of image-based prediction.

Floorist: A 121×121 vertex grid with $L = 6$ attention layers, 8-way MHA, 4096 GeLU units, and $E = 512$ embedding dimensions. The image encoder is a three-layer ViT [28] with patch size 6 (i.e., $P = 400$), 8-way MHA, and 4096 hidden units.

Ground Truth: Using the actual walls of building, i.e., true information gain.

Figure B.4: Examples of predictions on random samples from the test set. Each row is a section of a single trajectory, in sequence from left to right. ● Frontier locations used in the information gain evaluation are also shown. Other conventions as in Fig. B.3. Because of the fine detail in the figure, it is advisable to enlargen the vector graphics by using the zoom tool of a digital document viewer.

(a) U-Net [1], [2]



(b) Floorist $30\times30\,\mathrm{m}^2$

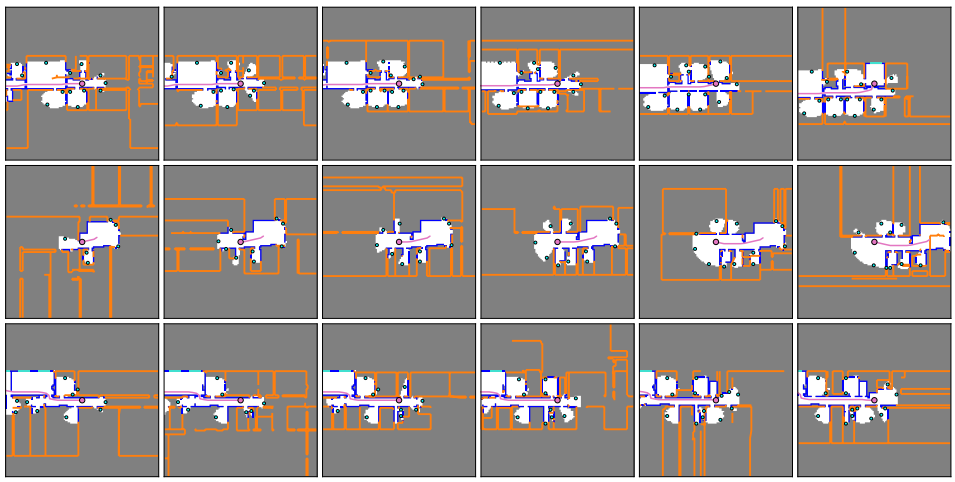


Figure B.5: Further examples of predictions on random samples from the test set. Conventions as in Fig. B.4. Note that in (a), the predicted occupancy grid is shown with Occupied cells. Because of the fine detail in the figure, it is advisable to enlargen the vector graphics by using the zoom tool of a digital document viewer.

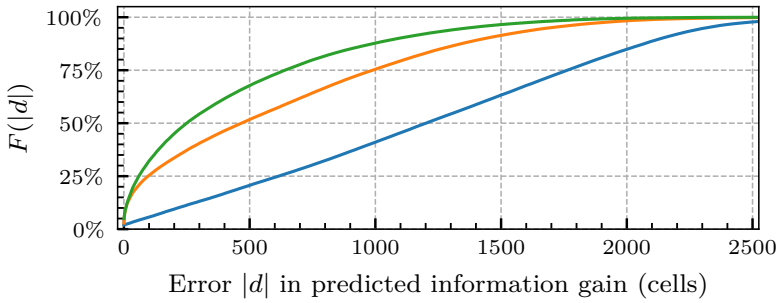


Figure B.6: Cumulative distribution function F of absolute error $|d| = |\tilde{I} - \hat{I}|$ in predicted information gain $\tilde{I}$ from the true information gain $\hat{I}$ using line segments from — Naive, — U-Net, and — Floorist. $N = 1\,464\,140$.

In Fig. B.6, the cumulative distribution function of absolute errors in predicted information gain is presented for the three evaluation targets on test data. Image-based prediction provides substantial improvements over naive non-predictive estimation, and our sequence-based approach in turn provides substantial improvements over image-based prediction. 95 % confidence intervals for the median absolute error (MAE) are 1195 ± 2, 452 ± 1, and 236 ± 1 cells. The intervals are computed by bootstrapping with 1000 trials. A *two-sample Kolmogorov-Smirnoff test* (KS) indicates whether a pair of CDFs differ significantly by measuring the largest vertical gap. Image-based predictions are significantly more accurate than naive estimation (ΔMAE 745 cells, KS 34.5 %), and Floorist predictions are in turn significantly more accurate than image-based prediction (ΔMAE 217 cells, KS 16.2 %).

B.8.2 Scale and Sensor Range

To assess the impact of scale, the information gain evaluation is performed on a model with twice the area, $30 \times 30\,\mathrm{m}^2$. The occupancy grid size is limited by GPU memory, and remains 121×121 cells. The resolution is therefore four times lower ($16.27\,\mathrm{cells/m}^2$ vs $65.07\,\mathrm{cells/m}^2$). The larger area resulted in 75 % longer token sequences, requiring significantly more GPU memory; batch size was therefore reduced to 2. Information gain is evaluated as in Section B.6, with two sensor range settings: $r = 4.5\,\mathrm{m}$ and $r = 9\,\mathrm{m}$. To ensure the results are comparable, information gain is computed at the base resolution. The MAE of each model at each setting is reported in Table B.1, with the frequency of overestimating and underestimating of the information gain. The first three rows are the results reported in Section B.8.1. The performance of the larger-scale model is slightly worse than the base model in the shorter range setting (ΔMAE 40 cells). The large-scale model is significantly better than the base model in the 9-meter case (ΔMAE 679 cells), because the longer sensor range often reaches outside the $15 \times 15\,\mathrm{m}^2$ prediction area of the base model, giving an overestimating effect similar to naive estimation. This is reflected

Table B.1: Evaluation of Scale and Sensor Range

r (m)	Area (m^2)	Median $\lvert\tilde{I} - \hat{I}\rvert$ (cells)	$\langle[\tilde{I} < \hat{I}]\rangle$ (%)	$\langle[\tilde{I} > \hat{I}]\rangle$ (%)
4.5	Naive	1195	0.00	98.0
	U-Net	452	11.3	85.9
	15×15	**236**	28.5	66.5
	30×30	276	34.0	58.7
9.0	15×15	1475	19.0	80.1
	30×30	**796**	31.5	66.0

Table B.2: Evaluation of Model Ablations on Test Set

Ablation	Loss $\langle L\rangle$ (bits)	Accuracy $\langle[\hat{t} = t^*]\rangle$ (%)
Floorist 15×15 m^2	1.13	81.7
No Window Labels	1.17 (+0.043)	81.3 (-0.38)
Shared Vertex Embedding	1.20 (+0.065)	80.8 (-0.87)
No Context Segments	1.23 (+0.103)	80.1 (-1.55)
No Image Encoder	1.40 (+0.267)	78.1 (-3.54)

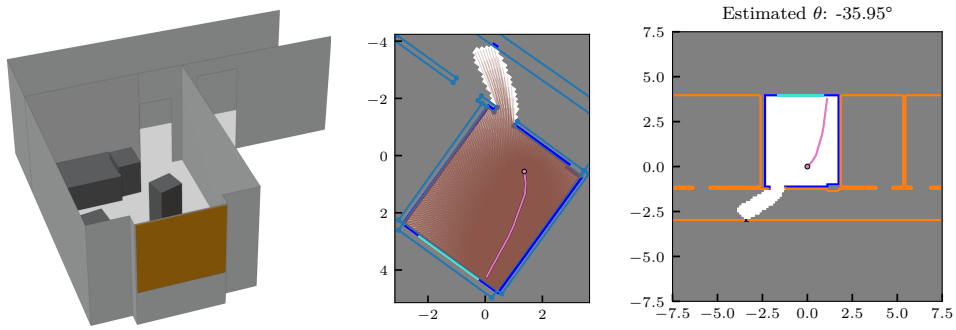
Values in parentheses are relative to the first row. L is defined as in Eq. (B.8), and t^* is the maximum likelihood estimate.

in the sharp increase in frequency of overestimation.

B.8.3 Model Ablations

We ablate some key components of our model and evaluate the impact on its performance on the test set in terms of the loss and the accuracy of the maximum likelihood estimate. The following ablations are reported in Table B.2: 'window' labels replaced by 'occupied' in training and test occupancy grids; embedding weights of visible segments is shared with embedding weights of the target segments; without visible segments as cross-attended tokens, i.e., only using the occupancy grid as context; and, without the image encoder, i.e., only using visible segments as context. We see that each ablation degrades performance, and that the occupancy grid context is most important. This may be because the occupancy grid is the only volumetric representation of the environment, and indicates where free versus unknown space is.

Window labels have the smallest impact on the reported metrics since windows are relatively uncommon in the dataset. Window labels were included as a cue for the building exterior, so the effect of those labels should be evaluated by evaluating if they prevent the model from predicting walls outside the building exterior. We specifically look at instances where at least one occupancy grid cell has the 'win-



(a) (b) (c)

Figure B.7: Illustration of (a) a reconstructed floor plan from [45], with walls, a window, and furniture; (b) simulated —● Trajectory inside the floor plan derived from (a) with sensor scans; and (c) the axis-aligned occupancy grid, and an example of — Predicted walls from Floorist.

dow' label, and compute the length of predicted line segments that fall outside the building perimeter as a proportion of the total predicted segment length. Window labels reduced the length of segments outside the building perimeter by 22.4 %, from 10.93 % to 8.48 %, indicating that the model does respond to the window cues.

B.8.4 Reconstructed Floor Plans

The software library *RoomPlan* [45] enables online reconstruction of floor plans from real-world sensor data in cluttered environments. As a test of the generality of our approach, we show that our method can use such reconstructions. RoomPlan runs on a modern smartphone using an image sensor and a solid-state LIDAR sensor, and produces a parametric 3D model of the environment as a set of objects of some predefined types: walls, doors, windows, and miscellaneous furniture. From this model, 2D line segments are derived by the orthogonal projection of the walls and windows onto the ground plane. A LIDAR sensor is simulated along the trajectory, and the floor plan is predicted from the resulting occupancy grid. The reconstructed 2D floor plan, the trajectory, and a wall prediction is shown Fig. B.7. The model correctly infers that there is a corridor outside, and that there is an adjoining room; however, it predicts the room to be in the middle of the corridor, not the end, as is the actual case.

B.9 Conclusion

In this work, we have presented an attention-based generative model for floor plans conditioned on realistic sensor data. We have shown that our model can cope with the high dimensionality of a time-dependent occupancy grid representation with a

realistic sensor model and make competent predictions as quantified by evaluating the predicted information gain. The approach offers advantages over traditional image-based predictions at the cost of longer inference time, though autoregressive sampling performance is an active area of research. We have also shown that the approach is robust enough to be used in real-world environments, demonstrated by using an off-the-shelf floor plan mapping solution as the source of floor plan data. In the future, we aim to extend our real-world demonstration to create a real-time floor plan prediction system from sensor data. Another interesting direction is adversarial multi-agent contexts where intuiting the surrounding is important, such as pursuit-evasion and other search games.

References

[A] L. Ericson and P. Jensfelt, "FloorGenT: Generative Vector Graphic Model of Floor Plans for Robotics", in *International Conference on Intelligent Robots and Systems*, IEEE, 2022. DOI: 10.1109/IROS47612.2022.9982144.

[1] Y. Tao, Y. Wu, et al., "SEER: Safe Efficient Exploration for Aerial Robots using Learning to Predict Information Gain", in *International Conference on Robotics and Automation*, IEEE, 2023.

[2] K. Katyal, K. Popek, C. Paxton, P. Burlina, and G. D. Hager, "Uncertainty-aware Occupancy Map Prediction using Generative Networks for Robot Navigation", in *International Conference on Robotics and Automation*, IEEE, 2019.

[3] J. Wei, X. Wang, et al., "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models", *Advances in Neural Information Processing Systems*, vol. 35, 2022.

[C] L. Ericson, D. Duberg, and P. Jensfelt, "Understanding Greediness in Map-Predictive Exploration Planning", in *European Conference on Mobile Robots*, IEEE, 2021. DOI: 10.1109/ECMR50962.2021.9568793.

[4] B. Zhou, Y. Zhang, X. Chen, and S. Shen, "FUEL: Fast UAV Exploration using Incremental Frontier Structure and Hierarchical Planning", *Robotics and Automation Letters*, 2021.

[5] R. Shrestha, F.-P. Tian, W. Feng, P. Tan, and R. Vaughan, "Learned Map Prediction for Enhanced Mobile Robot Exploration", in *International Conference on Robotics and Automation*, IEEE, 2019.

[6] E. Zwecher, E. Iceland, S. R. Levy, S. Y. Hayoun, O. Gal, and A. Barel, "Integrating Deep Reinforcement and Supervised Learning to Expedite Indoor Mapping", in *International Conference on Robotics and Automation*, IEEE, 2022.

[7] Y. Tao, E. Iceland, B. Li, E. Zwecher, U. Heinemann, A. Cohen, A. Avni, O. Gal, A. Barel, and V. Kumar, "Learning to Explore Indoor Environments using Autonomous Micro Aerial Vehicles", 2023. arXiv: 2309.06986.

[8] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding Horizon "Next-Best-View" Planner for 3D Exploration", in *International Conference on Robotics and Automation*, IEEE, 2016.

[9] D. Duberg and P. Jensfelt, "UFOExplorer: Fast and Scalable Sampling-based Exploration with a Graph-based Planning Structure", *Robotics and Automation Letters*, 2022.

[10] J. Yu, H. Shen, J. Xu, and T. Zhang, "ECHO: An Efficient Heuristic Viewpoint Determination Method on Frontier-based Autonomous Exploration for Quadrotors", *Robotics and Automation Letters*, 2023.

[11] M. Luperto, M. Antonazzi, F. Amigoni, and N. A. Borghese, "Robot Exploration of Indoor Environments using Incomplete and Inaccurate Prior Knowledge", *Robotics and Autonomous Systems*, 2020.

[12] M. Selin, M. Tiger, D. Duberg, F. Heintz, and P. Jensfelt, "Efficient Autonomous Exploration Planning of Large-Scale 3D Environments", *Robotics and Automation Letters*, 2019.

[13] L. Schmid, C. Ni, Y. Zhong, R. Siegwart, and O. Andersson, "Fast and Compute-efficient Sampling-based Local Exploration Planning via Distribution Learning", *Robotics and Automation Letters*, 2022.

[14] S. Karaman and E. Frazzoli, "Sampling-based Algorithms for Optimal Motion Planning", *International Journal of Robotics Research*, vol. 30, no. 7, 2011.

[15] Y. Yue, T. Kontogianni, K. Schindler, and F. Engelmann, "Connecting the Dots: Floorplan Reconstruction Using Two-Level Queries", in *Conference on Computer Vision and Pattern Recognition*, 2023.

[16] J.-W. Su, K.-Y. Tung, et al., "SLIBO-Net: Floorplan Reconstruction via Slicing Box Representation with Local Geometry Regularization", in *Conference on Neural Information Processing Systems*, 2023.

[17] J. Chen, Y. Qian, and Y. Furukawa, "HEAT: Holistic Edge Attention Transformer for Structured Reconstruction", in *Conference on Computer Vision and Pattern Recognition*, IEEE/CVF, 2022.

[18] A. Gueze, M. Ospici, D. Rohmer, and M.-P. Cani, "Floor Plan Reconstruction from Sparse Views: Combining Graph Neural Network with Constrained Diffusion", in *International Conference on Computer Vision*, IEEE/CVF, 2023.

[19] M. Luperto, F. Amadelli, M. Di Berardino, and F. Amigoni, "Mapping Beyond What You Can See: Predicting the Layout of Rooms Behind Closed Doors", *Robotics and Autonomous Systems*, 2023.

[20] B. Liao, S. Chen, et al., "MapTR: Structured Modeling and Learning for Online Vectorized HD Map Construction", in *International Conference on Learning Representations*, 2023.

[21] Q. Li, Y. Wang, et al., "HDMapNet: An Online HD Map Construction and Evaluation Framework", 2021. arXiv: `2107.06307`.

[22] A. Carlier, M. Danelljan, A. Alahi, and R. Timofte, "DeepSVG: A hierarchical generative network for vector graphics animation", *Advances in Neural Information Processing Systems*, 2020.

[23] P. Reddy, M. Gharbi, M. Lukac, and N. J. Mitra, "Im2Vec: Synthesizing vector graphics without vector supervision", in *Conference on Computer Vision and Pattern Recognition*, IEEE, 2021.

[24] A. Aydemir, P. Jensfelt, and J. Folkesson, "What can we learn from 38 000 rooms? Reasoning about unexplored space in indoor environments", in *Conference on Intelligent Robots and Systems*, IEEE, 2012.

[25] C. Maple, "Geometric design and space planning using the marching squares and marching cube algorithms", in *International Conference on Geometric Modeling and Graphics*, IEEE, 2003.

[26] T. Bachlechner, B. P. Majumder, H. Mao, G. Cottrell, and J. McAuley, "ReZero is All You Need: Fast Convergence at Large Depth", in *Uncertainty in Artificial Intelligence*, PMLR, 2021.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need", in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2017.

[28] A. Dosovitskiy, L. Beyer, et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale", 2020. arXiv: 2010.11929.

[29] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space", *Advances in neural information processing systems*, 2017.

[30] E. Dijkstra, "A note on two problems in connexion with graphs", *Numerische Mathematik*, 1959.

[31] G. Weiß, C. Wetzler, and E. Von Puttkamer, "Keeping Track of Position and Orientation of Moving Indoor Systems By Correlation of Range-finder Scans", in *International Conference on Intelligent Robots and Systems*, IEEE, vol. 1, 1994.

[32] G. Weiss, Y. Goldberg, and E. Yahav, "Thinking Like Transformers", in *International Conference on Machine Learning*, PMLR, 2021.

[33] S. Goyal, Z. Ji, A. S. Rawat, A. K. Menon, S. Kumar, and V. Nagarajan, "Think before you speak: Training Language Models With Pause Tokens", 2023. arXiv: 2310.02226.

[34] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved Techniques for Training GANs", *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[35] K. Pillutla, S. Swayamdipta, R. Zellers, et al., "Mauve: Measuring the gap between neural text and human text using divergence frontiers", *Advances in Neural Information Processing Systems*, 2021.

[36] S. Barratt and R. Sharma, "A note on the inception score", 2018. arXiv: 1801.01973.

[37] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al., "A density-based algorithm for discovering clusters in large spatial databases with noise", in *International Conference on Knowledge Discovery and Data Mining*, vol. 96, AAAI Press, 1996.

[38] J. MacQueen, "Some methods for classification and analysis of multivariate observations", in *Symposium on Mathematical Statistics and Probability*, vol. 1, University of California Press, 1967.

[39] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization", in *International Conference on Learning Representations*, 2015.

[40] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The Curious Case of Neural Text Degeneration", in *International Conference on Learning Representations*, Curran Associates, Inc., 2019.

[41] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation", in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2015.

[42] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A ConvNet For The 2020s", in *Conference on Computer Vision and Pattern Recognition*, IEEE/CVF, 2022.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition", in *Conference on Computer Vision and Pattern Recognition*, IEEE/CVF, 2016.

[44] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking Atrous Convolution for Semantic Image Segmentation", 2017. arXiv: `1706.05587`.

[45] Apple Inc., *3D Parametric Room Representation with RoomPlan*, 2022. [Online]. Available: `https://machinelearning.apple.com/research/roomplan`.

Paper C

Understanding Greediness in Map-Predictive Exploration Planning

L. Ericson, D. Duberg, and P. Jensfelt
KTH Royal Institute of Technology

Abstract

In map-predictive exploration planning, the aim is to exploit a-priori map information to improve planning for exploration in otherwise unknown environments. The use of map predictions in exploration planning leads to exacerbated greediness, as map predictions allow the planner to defer exploring parts of the environment that have low value, e.g., unfinished corners. This behavior is undesirable, as it leaves holes in the explored space by design. To this end, we propose a scoring function based on inverse covisibility that rewards visiting these low-value parts, resulting in a more cohesive exploration process, and preventing excessive greediness in a map-predictive setting. We examine the behavior of a non-greedy map-predictive planner in a bare-bones simulator, and answer two principal questions: a) how far beyond explored space should a map predictor predict to aid exploration, i.e., is more better; and b) does shortest-path search as the basis for planning, a popular choice, cause greediness. Finally, we show that by thresholding covisibility, the user can trade-off greediness for improved early exploration performance.

 Paper accepted for the 10th European Conference on Mobile Robots (ECMR 2021). All authors are with the Division of Robotics, Perception and Learning at KTH Royal Institute of Technology, Stockholm, SE-10044, Sweden. This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP), the SSF project FACT and the VR grant XPLORE3D. For e-mail correspondence, contact `ludv@kth.se`.

C.1 Introduction

Exploration is an essential capability in any truly autonomous robot dealing with an unknown or changing environment. It is particularly useful in radio-denied scenarios where time is of the essence, such as search-and-rescue in disaster zones [1]. The typical metric used is coverage over time or distance traveled [2], [3], though some recent work focuses on reconstruction error [4].

Although exploration implies an unknown environment, it is exceedingly rare that no prior information is available. For example, in man-made indoor environments, there is plenty of structure and common knowledge that a system should be able to exploit [5]. In spite of this, few state-of-the-art autonomous exploration methods make use of this.

Suppose a robot is tasked with exploring an indoor environment and is placed in a small room near a doorway to a long corridor. If the robot decides to explore the corridor first, it will have to return back to the room once it has reached the end of the corridor, thus resulting in unnecessary travel. Ideally, the robot would estimate what the rooms look like before they have been explored, and decide its course of action based on its estimation. This is the aim in *map-predictive exploration planning*.

Map prediction can be understood in multiple ways, from nearest-neighbor search in a patch-based database, to using a deep learning-based generative model to allow some degree of spatial intuition. In this paper, "predictive" is not restricted to the statistical sense of the word, but more generally "ahead of time". Predictions could be made from a longer-range sensor, e.g., using a LiDAR to plan the path of an RGB-D camera. If a partial or outdated map is available, as in multi-agent exploration or search-and-rescue in a collapsed building, prediction could mean using that knowledge.

Recent work on map-predictive exploration [6], [7] use map predictions to improve estimation of a utility function, and demonstrate improved performance. However, we find that because the estimation is more accurate, it has caused the exploration to become more greedy, maximizing short-term utility. In [8], up to 71 % of the total time is spent on covering the last 10 % of space. We find this phenomenon in most state-of-the-art exploration methods.

In light of these findings, we propose that non-greedy exploration is key to truly unlocking map-predictive capabilities for exploration. In this paper, a minimal model of autonomous exploration planning is defined, in order to study the causes of greediness in map-predictive settings. To this end, we substitute prediction with an oracle, giving the exploration planner knowledge of the environment a-priori. The contributions are: 1. We propose a scoring strategy that removes greediness by weighting the utility function by *inverse covisibility*, demonstrated in Fig. C.1. 2. We investigate the use of an exhaustive local search as the basis for exploration planning, as RRT-based planning may cause greediness in map-predictive settings. 3. We investigate the impact of a *prediction horizon* on greediness, which is the distance beyond explored space that is known in advance.

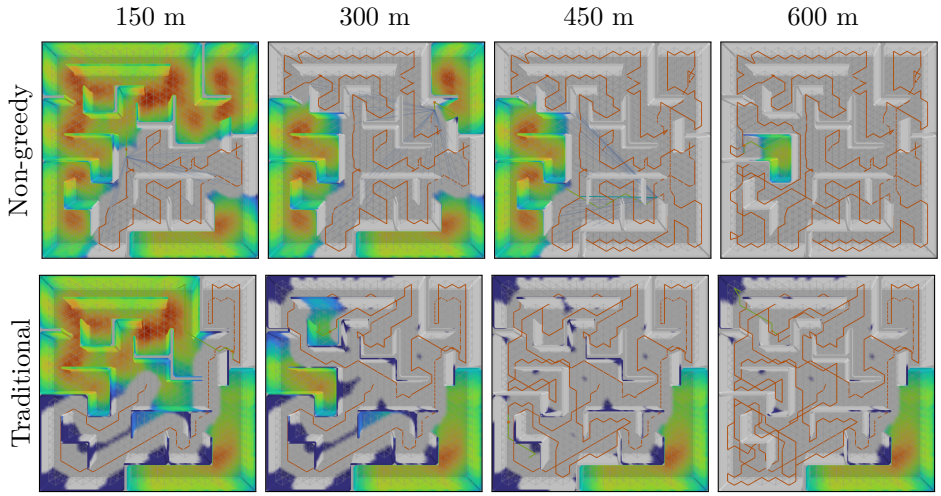


Figure C.1: An illustration of how traditional vs. non-greedy exploration planner performs in a map-predictive scenario. A snapshot is taken every 150 m of travel. Top: using a non-greedy planner, bottom: a variant using a traditional information gain formulation. Color shows the value of a proposed non-greedy scoring function that assigns high value in difficult-to-see regions. Red is lowest, dark blue is highest, and gray is already explored space. For the traditional planner, color illustrates what value the non-greedy scoring function would have assigned if it had been used. The traditional planner skips small regions early on which it must then revisit, allowing the non-greedy planner to finish earlier, after 620 m, compared to 1 067 m.

C.2 Background & Preliminaries

Generally, the goal of *autonomous exploration* is typically to explore some real-world environment in the shortest time possible. In its simplest form, it is a cycle of the following: 1. plan where to go by some heuristic, plan, and 3. integrate sensor readings into a map. The cycle continues until the map is sufficiently complete, or a predefined time limit is reached.

In *frontier-based exploration* such as [6], [7], [9], [10], [11], the planner selects which frontier to visit according to some utility function. A frontier is defined as a boundary between free and unexplored space. Another common formulation is *sampling-based exploration* such as Bircher et al. [2] and Schmid et al. [4]. A planning tree T rooted at the current state is constructed in an RRT-like fashion [12], [13]. Specifically, T is iteratively constructed by sampling a point p, moving it to some maximum distance from the nearest node p^* in T, and adding the edge $p^* \to p$ to T. Once the planning tree is built, the best path is chosen according to some score function over tree nodes.

The score functions can be said to consist of two parts: gain g, and cost c. The gain is formulated as an integral over an uncovered surface or volume V

$$g(V) = \int_V \rho \, dV \tag{C.1}$$

where dV is a surface or volume element of V as appropriate, and ρ is a *value density function*. V is typically discretized, turning the integral into a sum. The cost is normally based on distance or trajectory execution time.

In [2], a sampling-based receding horizon approach is presented where a single step is taken before replanning. The planning tree can fail to reach unexplored space when the environment is difficult to navigate, or the nearest unexplored space is far away. AEP [3] solves this by a falling back to a frontier-based strategy when the information gain for the best sampled path is insufficient. Like Bircher et al. [2], the AEP gain is volumetric. The method is evaluated both in simulation and on a drone with coverage as the metric. In [4] a sampling-based method is presented where the gain function is designed to minimize the reconstruction error. The authors also propose to normalize the score function by computing the sum gain per cost over each path in the planning tree in *global normalization*. Works [2], [3], [4] primarily rely on traditional information gain, i.e., $\rho = 1$.

In terms of map prediction, Aydemir et al. [5] presents a database of CAD models and is used to make predictions for topological maps. Elhafsi et al. [14] present a map-predictive approach where the perspective is one of motion planning in the presence of unknown obstacles. Their contribution is improving maneuvering and safety in unknown environments.

A map-predictive exploration method is presented by Shrestha et al. [7]. Generative image inpainting based on deep-learning is used to predict a 2D grid map, providing better estimates of the gain by inferring V in Eq. (C.1). They use a standard frontier-based method and demonstrate using simulation that map predictions allow the robot to perform efficient coarse exploration.

In [6], a frontier-based method is presented that investigates how different types of prior knowledge and predictions can be used for exploration planning. Predictions of room layouts are made from a partial 2D grid map using line detection heuristics. The authors note that the better their prediction is, the more it leaves fragments of unexplored space behind. This result is consistent with Shrestha et al. [7], where the greediness results in incomplete coverage.

In [15], map-predictive exploration is cast as offline exploration planning with simple simulation in 2D grid maps. Exploration planning is accomplished by a frontier-based A* planner, including the exploration state as part of the search tree, which is not feasible beyond relatively small 2D grid maps. Notably, Li et al. [15] explicitly prevent the planner from visiting "small clusters" in the interest of efficiency, which results in incomplete coverage even for the simple maps presented.

We observe that in each of [2], [3], [4], [6], [7], [15], the system quickly gets to 85–95 % coverage, then spends most of its time on the last 5–15 %, if at all covered. This finding is the key motivation to our study of greediness in map-predictive

exploration. We take a similar approach to Li et al. [15], using a simplified model of exploration as the basis of investigation.

C.3 Model

Recent work in autonomous exploration typically relies on high-fidelity simulators for evaluation. In this work, we study the theoretical aspects of map-predictive exploration. A simple simulator is therefore not only sufficient, but to be preferred, as it allows us to eliminate extraneous error sources, such as sensor noise, complex motion models, and heavy simulation environments.

C.3.1 World Representation

We represent occupied space as a mesh of K triangular faces, denoted $S_\Omega = \{f_1, f_2, \ldots, f_K\}$, and explored space as a subset $S \subseteq S_\Omega$ of those K faces. This lets us define completion exactly as the percentage of area seen, and have 100 % explored if and only if $S = S_\Omega$. A face $f \in S$ is *seen*, and a face $f \notin S$ is *unseen*.

This representation is efficient enough to also enable us to do sensor integration in the planning step, and to thus determine the exploration state (see Eq. (C.2)) along every path inside the planning algorithm.

Motion is simplified by being confined to a precomputed roadmap, represented as an undirected graph $G = (V, E)$ with the states $V \subseteq \mathbb{R}^3$. The roadmap is distributed as a connected lattice, so that each edge is of equal length, and there is a path between every pair of states. The motion model is deterministic. The *visible faces* $S_{\{u,v\}} \subseteq S_\Omega$ along the edge $\{u, v\} \in E$ are the set of unoccluded faces within sensor range between u and v.

The exploration state is thus parameterized by the tuple (S, v) with $S \subseteq S_\Omega$ the set of seen faces, and $v \in V$ the current state. Its successor function is

$$\big(S, v\big) \leftarrow \big(S \cup S_{\{v,v'\}}, v'\big) \tag{C.2}$$

where $v' \in V$ is the successor state.

In summary, our world representation differs from the typical exploration literature in these regards:

1. The state space is essentially finite, defined by V, and we can therefore plan by a graph search on G.

2. Occupied space is represented as a triangular mesh, and explored space is a subset of occupied space, hence eliminating reconstruction errors.

3. While many exploration methods "double count" uncovered space when there is sensor overlap in the planned path, we do not, since the same state successor function is used in planning and execution.

These deviations allow us to isolate the planning part of exploration planning and also to close the gap between the plan and the execution in terms of using sensor information.

C.3.2 Path Planning with Map Predictions

RRT-based planners are not a well-motivated choice for map-predictive exploration planning, since they compute the shortest path which is not generally desirable in unexplored space. To illustrate this point, suppose a perfect maze is to be explored in a map-predictive setting, starting at the maze entrance. A perfect maze is equivalent to a tree where junction points are tree nodes, and the tree leaves are dead ends. The optimal exploration path is then equivalent to a depth-first search of the tree. An RRT-based planner, however, would tend to select "the longest shortest path", i.e., the path to the furthest-away leaf node in the maze tree, since such a path explores the most space. This phenomenon is visible in Fig. C.1, where the traditional planner first explores the bottom left corner, then the top right, then the top left, and so on.

We therefore compare two path planners, which we call *backtracking search* and *shortest-path search*. Shortest-path search models RRT-based path planning and is equivalent to a perfect RRT-based path planner, as it computes the actual shortest path in G.

In backtracking search, an exhaustive local search on G is performed by generating paths via breadth-first search. It revisits the same state multiple times, allowing backtracking paths such as $v_1 - v_2 - v_1 - \cdots$ to form. Since the set of possible paths grows exponentially with path length, the search is limited to the first N paths so generated.

Temporary *jump edges* are inserted into the roadmap G from the current state to each state in the frontier set $F \subseteq V$ at the start of path planning. A state is in F if and only if it has at least one seen and one unseen face along its edges. A jump edge is equivalent to traversing the shortest path over already-explored edges, so all jump edges have $S_{\{u,v\}} \subseteq S$ by definition, and consequently zero gain. Jump edges are approximately equivalent to starting a sub-search from each state in F. Jump edges also guarantee that the planner finds unseen faces, since the distance to at least one unseen face is two edges at most.

C.3.3 Prediction Horizon

Map prediction will realistically only be able to accurately predict at some maximum distance ahead of explored space. We call this distance the *prediction horizon*. Such a horizon is sure to impact exploration performance, and we therefore impose a horizon artificially by cropping the environment at d meters from explored space.

Specifically, we remove states in the roadmap G that are more than d meters plus the sensor range away from the nearest visited state. This means that planning is restricted to explored space plus d meters beyond.

C.3.4 Evaluating Greediness

Greediness can be quantified by the surface-to-volume ratio of unexplored space. It is preferable to have the remaining unexplored space concentrated in one dense area, as opposed to having small disconnected clusters spread out over the entire environment. The latter is a consequence of greedy exploration, so a non-greedy planner should have a lower surface-to-volume ratio of its unexplored space. We therefore evaluate greediness by the *isoperimetric ratio* (IPR). For the unexplored faces $U = S_\Omega \setminus S$,

$$\mathrm{IPR}(U) = L_U^2 A_U^{-1} \tag{C.3}$$

where L_U, A_U is the perimeter and area of U, respectively. IPR is a convenient metric, as it is dimensionless, and invariant to similarity transforms such as rotation or scaling.

C.4 Non-Greedy Exploration

Map predictions lead to ballooning gain in the predicted-but-unexplored space. The consequence is that the planner prefers moving towards these high-gain regions, leaving comparatively low-gain fragments and slivers of unexplored space behind. Eventually, these low-gain fragments are all that remain, and the robot must travel long distances to re-visit areas it has already been in, to visit the remaining fragments and complete the exploration.

Adding or increasing a distance-based cost is not sufficient to prevent this phenomenon. Consider a robot that is approaching a corner of an unexplored room. It can either visit the entire corner, or just come near it and continue away from the corner. The latter plan almost surely has higher gain, regardless of how high the distance cost is set.

The solution is to "finish what you started" by weighting those small regions higher. These regions are visible from places where the sensor would see few other faces simultaneously. Consequently, *sensor covisibility* is low for such faces. By inversely weighting each face by the covisibility, greediness can be avoided.

C.4.1 Sensor Covisibility

We define *sensor covisibility* as a binary relation $C_{ij} \in \{0, 1\}$ between a pair of faces f_i and f_j. The pair is covisible if at some edge in G, they are both visible to the sensor:

$$C_{ij} = 1 \iff \exists \{u, v\} \in E : \{f_i, f_j\} \subseteq S_{\{u,v\}} \tag{C.4}$$

with E and $S_{\{u,v\}}$ as before. Naturally, the relation is symmetric, so $C_{ij} = C_{ji}$.

C_{ij} forms a relation matrix $\boldsymbol{C} \in \{0, 1\}^{K \times K}$ which is used to find the total unseen covisible area c_i for each face with the matrix-vector product

$$\begin{bmatrix} c_1 & c_2 & \dots & c_K \end{bmatrix}^{\mathrm{T}} = \boldsymbol{C} \begin{bmatrix} a_1 & a_2 & \dots & a_K \end{bmatrix}^{\mathrm{T}} \tag{C.5}$$

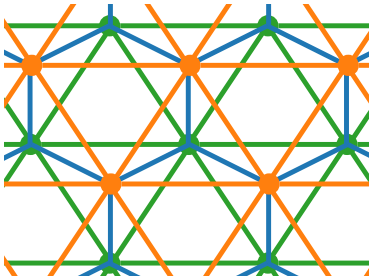


Figure C.2: Illustration of the roadmap lattice for two layers. The orange layer is sitting on top of the green layer, with blue edges connecting the two. The distance between the layers is such that all blue, orange, and green edges are of equal length.

where a_i is the area of the ith face, but zero for seen faces. Since seen faces do not contribute, c_i will decrease as its covisible faces are explored. We weight inversely by c_i, which encourages further exploration in the neighborhood of ith face, and thus drives the planner to finish exploring a region before moving on.

C.4.2 Inverse Covisibility Scoring

The gain function $g(\cdot)$ is formulated as a surface integral over uncovered space similar to Schmid et al. [4] and Yoder and Scherer [16]. In our case, it is a sum over the set of new faces seen when traversing from u to v, i.e., $\Delta S = S_{\{u,v\}} \setminus S$, and

$$g(\Delta S) = \sum_{f \in \Delta S} \rho_f \quad \text{with} \quad \rho_f = \exp\left(-\alpha\sqrt{c_f}\right). \tag{C.6}$$

The parameter α determines the strength of preference for low covisibility faces. Note that traversing an already visited edge has a gain of zero, since $\Delta S = \emptyset$ in that case.

Similar to Selin et al. [3], cost is modelled by exponential decay

$$c(D) = \exp\left(-\lambda D + b\right) \tag{C.7}$$

where D is distance traveled from the current state, and λ is the decay rate. The bias b is introduced to improve numerical stability. The per-edge score function is

$$q(\Delta S, D) = g(\Delta S)c(D) \tag{C.8}$$

and the score of a path is a sum of q over the path's edges.

The prediction horizon d affects covisibility scoring in that we mark faces that are more than d meters away from the nearest seen face as non-covisible with all other faces, i.e., $c_f = 0$ for such faces.

C.5 Implementation

The covisibility matrix C is computed by ray-casting radially outwards at regular intervals of 25 cm along every edge $\{u, v\}$. The sensor geometry is thus a sphere,

Table C.1: Parameter settings in our experiments

Algorithm		**Model**	
α	1.2	Sensor Range [m]	3.0
λ	0.35	Edge Length [m]	1.5
b	7	Branching Factor	~ 6
N	30 000		

Table C.2: Environments used in our experiments

Environment	**Office [3]**	**Labyrinth [4]**
Bounding Box	$34 \times 25 \times 5$	$40 \times 40 \times 3$ m^3
Surface Area of S_Ω	1 725	2 942 m^2
Faces K	24 386	30 819
Face Area	~ 0.07	~ 0.095 m^2
$\lvert V \rvert$	573	1 449
$\lvert E \rvert$	1 540	4 822
Pairwise Path Length	~ 7.6	~ 10.5 m

limited to a radius of 3 m. The sphere is approximated by an icosphere of 2 562 vertices, with one ray cast at each vertex. Since G is constant, ray casting is done once.

The roadmap is constructed by multiple layers of triangular tilings in the XY plane, shown in Fig. C.2. Each layer is offset from the one below so that its vertices end up at the triangle centers of the layer below. Each vertex in the upper layer is connected to the three vertices making up the triangle below it. The layers are stacked Z-wise so that the inter-layer edges are the same length as the sides of the triangles. This means that all edges are equally long, in our case, 1.5 m. We remove edges that intersect the environment mesh. Faces that are not visible from anywhere in G are discarded from S_Ω.

With a budget of $N = 30\,000$ paths and a branching factor around 6, we search all paths of length 3 and below, and about half the paths of length 4. The selected path is most often of length 4 since Eq. (C.8) is non-negative. We take a single step of the best path, thus moving 1.5 m before the planning cycle begins again. All pertinent parameters are presented in Table C.1.

C.6 Experiments & Evaluation

In this section, we first show that non-greedy planning does improve long-term performance for map-predictive exploration. Following that, we answer the two questions posed at the beginning of this paper: what is the effect of the prediction horizon, and does an RRT-based planner exhibit more greediness given map pre-

Figure C.3: Unexplored surface area over distance with the same settings as in Fig. C.1, in the maze environment. Shaded area is $\pm 2\sigma$. "ICVS" is using the inverse covisibility score described in Section C.4 with backtracking search, and traditional uses shortest-path search with uniform information gain. In both cases, the prediction horizon $d = \infty$. The non-greedy variant lags behind until around the 5 % mark, and reaches completion shortly after.

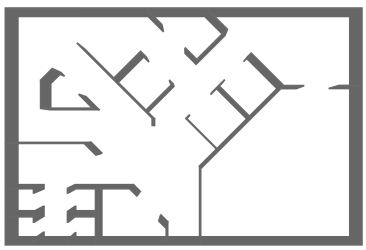


Figure C.4: The office environment from [3] used in the experiments of Section C.6.3. Even though it is considerably smaller, completion distances are comparable to those of the maze environment for both traditional and non-greedy exploration.

dictions. We then show that a hybrid approach can be taken where long-term and short-term benefits are gained from map predictions. We run our experiments in two indoor environments: first, the maze environment from Schmid et al. [4] seen in Fig. C.1, and second, the office environment from Selin et al. [3] depicted in Fig. C.4. Environment statistics are summarized in Table C.2 for comparison. An important difference between the two environments is that the office environment has *outlier faces* that can only be seen from particular edges in the roadmap. The outlier faces are artifacts from reconstructing the mesh from noisy sensor data using [3]. The maze environment, on the other hand, has no outlier faces as it is not a reconstruction. Unless otherwise stated, each experiment is repeated 10 times.

C.6.1 Traditional Vs. Non-Greedy Planning

Our first experiment establishes whether or not a non-greedy scoring function actually outperforms a traditional formulation in the long run, as we have hypothesized. The traditional formulation uses shortest-path search, with uniform information gain, i.e., $\rho = 1$, and both are given knowledge of the full map, $d = \infty$. Figure C.3 shows the aggregate statistics of that comparison.

In Figure C.1, eight snapshots are presented from one such run: four from the non-greedy planner, and four from the traditional planner. The snapshots show that the traditional planner leaves corners and other smaller regions unexplored throughout the map. The covisibility score is encoded in color, showing that these regions would have had high gain if covisibility scoring had been used in the traditional setting.

Figure C.3 shows unexplored space over distance traveled for the same runs. As expected, the traditional formulation is fast to cover ground early on, but as exploration nears completion, the non-greedy planner outpaces it since it has not left

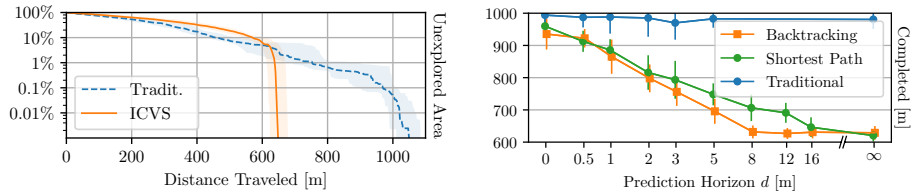


Figure C.5: Distance at completion in meters for some choices of prediction horizon d, comparing the two path planning variants in the office environment, both using inverse covisibility scoring. For reference, a traditional information gain with shortest-path planning is also shown. The horizontal axis is logarithmic. Vertical lines show $\pm 2\sigma$. Backtracking search improves until $d \geq 8$, while shortest-path search improves until $d = \infty$ but slower. The traditional formulation is unaffected by the prediction horizon.

any fragments behind. Meanwhile, the traditional planner must travel an additional 400 m to finish, 63 % longer. This mirrors the findings in Bircher et al. [2], Selin et al. [3], Schmid et al. [4], Luperto et al. [6], and Shrestha et al. [7], and therefore validates the model.

C.6.2 Planning Style and Map Prediction

Figure C.5 shows a comparison of the two planning styles using inverse covisibility scoring along with a traditional formulation using a uniform gain in the office environment for a selection of prediction horizons. Backtracking search performs better than shortest-path search, except when the prediction horizon is either very near, or very far away. Backtracking search reaches its optimum at $d = 8$, which is approximately the planner's reach beyond explored space, and also approaching the environment size. Notably, the optimum for shortest-path search is reached much later, at $d = \infty$, where the two planners perform equally well. Traditional exploration is unaffected by map prediction.

C.6.3 Effects of the Prediction Horizon

Figure C.6a (top) shows completion over distance for a selection of d values, when using inverse covisibility scoring and backtracking search. Lower d values lead to more rapid exploration at first, suggesting more greediness. These are caught up by the high d conditions as more of the environment is explored. The higher d conditions finish abruptly, while the low d conditions do not because of the considerably fragmented unexplored space they have left behind as evidenced by the IPR.

The greediness thus decreases with the prediction horizon d, and higher d conditions finish with a shorter total distance. For $d \geq 8$, the performance is the same as for $d = \infty$. This is the limit of the planner's ability to take advantage of map predictions due to limited search depth.

C.6.4 Leaving Some Things Behind

Figures C.3 and C.6a shows that improved late-stage performance has come at the cost of early-stage performance, which is the intended behavior. Non-greedy exploration finishes exploring every corner and crevice before moving on, therefore delaying exploring high gain regions. However, since the office environment is a reconstruction from noisy sensor data, it contains outlier faces that are only visible from a single edge in the roadmap. The proposed scoring function can be amended to ignore such outliers by setting their score low. Specifically, we substitute the covisible area with a thresholded variant in the gain function of Eq. (C.6),

$$\hat{g} = g[c_f \leftarrow \hat{c}_f] \quad \text{where} \quad \hat{c}_f = \begin{cases} c_f + P & c_f < c_T \\ c_f & \text{otherwise} \end{cases} \tag{C.9}$$

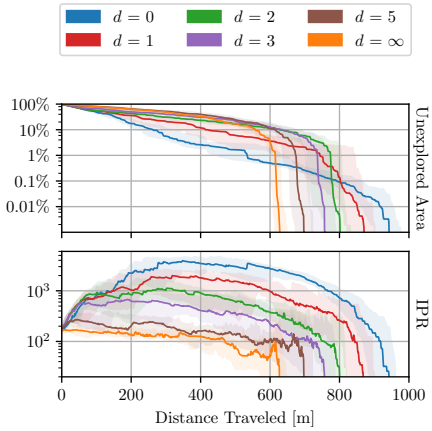
where P is a penalty incurred for faces of below-threshold covisibility, and c_T is the threshold itself. Since c_f is an exponent in Eq. (C.6), the penalty drastically lowers the gain for such faces, allowing the planner to skip them until only such below-threshold faces.

Figure C.6b shows the completion and IPR of this formulation in the office environment with a selection of thresholds c_T, with $d = \infty$ and $P = 200$. Baseline is using traditional information gain. With $c_T \in \{0.05, 0.10, 0.25\}$, only outlier faces are skipped. Consequently, results are similar to $d = \infty$ of Fig. C.6a, but early-stage performance has improved: at 400 meters of travel, about 12 % more space is explored.

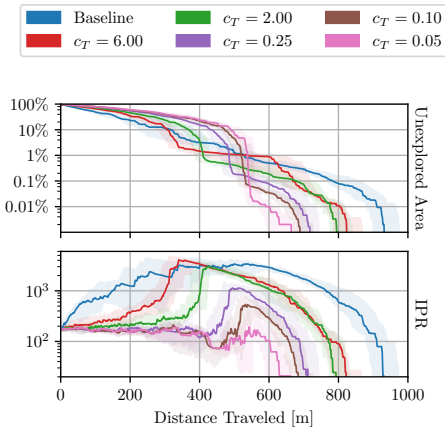
With $c_T = 6.00$, we reintroduce some greediness by allowing to skip larger regions. This condition gets to 10 % unexplored at the same time as the baseline, but manages to finish earlier, showing it has improved late-stage performance without sacrificing early-stage performance. The IPR curve also shows the planner is not simply reverting to traditional exploration, as IPR rises later, and decreases faster than the baseline. This is because even though some unexplored regions are left behind, once only such regions remain, they are explored in a non-greedy manner as before.

C.7 Conclusion

In traditional exploration planning, space is either explored and known, or unexplored and unknown. With map predictions, a third state arises: unexplored but predicted space. This third state of space calls into question the use of shortest-path search as the basis for exploration planning. We have demonstrated that a more flexible path planning can indeed improve performance, but these benefits are modest compared to the use of inverse covisibility scoring. This suggests that similar heuristics should be sufficient to enable map-predictive capabilities in RRT-based exploration planners. We have also shown that using such a heuristic, greediness decreases as the prediction horizon increases, indicating that even a relatively weak



(a) Completion and isoperimetric ratio at different prediction horizons d meters. Solid lines represent median, shaded regions represent a 90 % confidence interval. Low d values approach completion faster but are caught up by higher d values towards the end. The IPR explains why: low d values get ahead by being less thorough, while high d values have left no unexplored space behind when they approach completion.



(b) Using gain Eq. (C.9) we achieve a trade-off between speed and thoroughness. Smaller values of c_T means that smaller areas are left behind. Higher values allow skipping the most difficult-to-see faces, which are among the most costly to get to. An inflection point occurs when only faces with $c_f < c_T$ remain. IPR increases *before* that point, as unexplored space starts to become fragmented.

map predictor may be useful. Finally, we have shown that it is possible to combine early and late stage performance with a thresholded inverse covisibility scoring function, allowing the planner to skip small details without losing its non-greedy behavior.

The next step in this line of work is to design an inverse covisibility score for a state-of-the-art map-predictive method. We stress that it must be map predictive, as we have not demonstrated any advantage to inverse covisibility scoring without map prediction. A second line of inquiry is to predict volumetric data, e.g., [17], [18], as exploration state typically is represented.

References

[1] H. Ardiny, S. Witwicki, and F. Mondada, "Autonomous exploration for radioactive hotspots localization taking account of sensor limitations", *Sensors*, vol. 19, no. 2, p. 292, 2019.

[2] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding Horizon "Next-Best-View" Planner for 3D Exploration", in *International Conference on Robotics and Automation*, IEEE, 2016.

[3] M. Selin, M. Tiger, D. Duberg, F. Heintz, and P. Jensfelt, "Efficient Autonomous Exploration Planning of Large-Scale 3D Environments", *Robotics and Automation Letters*, 2019.

[4] L. Schmid, M. Pantic, R. Khanna, L. Ott, R. Siegwart, and J. Nieto, "An Efficient Sampling-Based Method for Online Informative Path Planning in Unknown Environments", *IEEE Robotics and Automation Letters*, 2020.

[5] A. Aydemir, P. Jensfelt, and J. Folkesson, "What can we learn from 38 000 rooms? Reasoning about unexplored space in indoor environments", in *Conference on Intelligent Robots and Systems*, IEEE, 2012.

[6] M. Luperto, M. Antonazzi, F. Amigoni, and N. A. Borghese, "Robot Exploration of Indoor Environments using Incomplete and Inaccurate Prior Knowledge", *Robotics and Autonomous Systems*, 2020.

[7] R. Shrestha, F.-P. Tian, W. Feng, P. Tan, and R. Vaughan, "Learned Map Prediction for Enhanced Mobile Robot Exploration", in *International Conference on Robotics and Automation*, IEEE, 2019.

[8] M. Luperto, M. Antonazzi, F. Amigoni, and N. A. Borghese. "Exploration of Indoor Environments Predicting the Layout of Partially Observed Rooms". Accessed 2020, Oct. [Online]. Available: `https://youtu.be/OQz-N8xAR6Q?t=121`.

[9] W. Gao, M. Booker, A. Adiwahono, M. Yuan, J. Wang, and Y. W. Yun, "An Improved Frontier-Based Approach for Autonomous Exploration", in *International Conference on Control, Automation, Robotics and Vision*, 2018. DOI: `10.1109/ICARCV.2018.8581245`.

[10] T. Cieslewski, E. Kaufmann, and D. Scaramuzza, "Rapid exploration with multi-rotors: A frontier selection method for high speed flight", in *International Conference on Intelligent Robots and Systems*, IEEE, 2017, pp. 2135–2142.

[11] A. Dai, S. Papatheodorou, N. Funk, D. Tzoumanikas, and S. Leutenegger, "Fast Frontier-based Information-driven Autonomous Exploration with an MAV", 2020. arXiv: `2002.04440`.

[12] S. M. LaValle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning", Department of Computer Science, Iowa State University, Technical Report, 1998.

[13] S. Karaman and E. Frazzoli, "Sampling-based Algorithms for Optimal Motion Planning", *International Journal of Robotics Research*, vol. 30, no. 7, 2011.

[14] A. Elhafsi, B. Ivanovic, L. Janson, and M. Pavone, "Map-Predictive Motion Planning in Unknown Environments", in *International Conference on Robotics and Automation*, IEEE, 2020.

[15] A. Q. Li, F. Amigoni, and N. Basilico, "Searching for Optimal Off-Line Exploration Paths in Grid Environments for a Robot with Limited Visibility", in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2012.

[16] L. Yoder and S. Scherer, "Autonomous Exploration for Infrastructure Modeling with a Micro Aerial Vehicle", in *Field and Service Robotics*, Springer, 2016, pp. 427–440.

[17] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean Signed Distance Fields for On-Board MAV Planning", in *International Conference on Intelligent Robots and Systems*, IEEE, 2017.

[18] D. Duberg and P. Jensfelt, "UFOMap: An Efficient Probabilistic 3D Mapping Framework That Embraces the Unknown", *IEEE Robotics and Automation Letters*, 2020.

Paper D

Information Gain Is Not All You Need

L. Ericson, J. Pedro, and P. Jensfelt
KTH Royal Institute of Technology

Abstract

Autonomous exploration in mobile robotics is driven by two competing objectives: coverage, to exhaustively observe the environment; and path length, to do so with the shortest path possible. Though it is difficult to evaluate the best course of action without knowing the unknown, the unknown can often be understood through models, maps, or common sense. However, previous work has shown that improving estimates of information gain through such prior knowledge leads to greedy behavior and ultimately causes backtracking, which degrades coverage performance. In fact, any information gain maximization will exhibit this behavior, even without prior knowledge. Information gained at task completion is constant, and cannot be maximized for. It is therefore an unsuitable choice as an optimization objective. Instead, information gain is a decision criterion for determining which candidate states should still be considered for exploration. The task therefore becomes to reach completion with the shortest total path. Since determining the shortest path is typically intractable, it is necessary to rely on a heuristic or estimate to identify candidate states that minimize the total path length. To address this, we propose a heuristic that reduces backtracking by preferring candidate states that are close to the robot, but far away from other candidate states. We evaluate the performance of the proposed heuristic in simulation against an information gain-based approach and frontier exploration, and show that our method significantly decreases total path length, both with and without prior knowledge of the environment.

Under review. All authors are with the Division of Robotics, Perception and Learning at KTH Royal Institute of Technology, Stockholm, SE-10044, Sweden. For e-mail correspondence, contact `ludv@kth.se`.

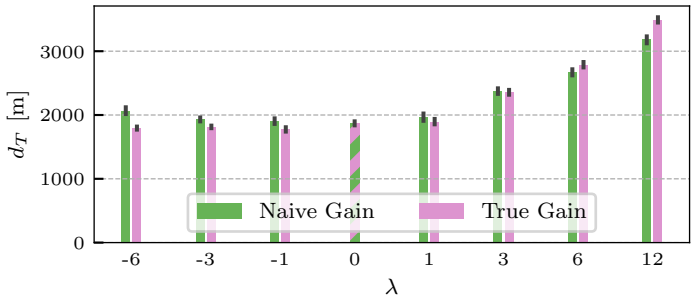


Figure D.1: Distance at completion d_T for a selection of gain affinities λ, where a higher λ means stronger preference for gain and a lesser concern with the length of the path to acquire it. Naive gain refers to the assumption that unknown space is occlusion-free, i.e., yields maximal gain; in true gain, the real would-be sensor scan is used for gain computation. Tellingly, negative affinities, i.e., *minimizing* gain, results in a lower d_T than maximization, and no choice is substantially better than nearest frontier, i.e., $\lambda = 0$.

D.1 Introduction

Autonomous exploration is a fundamental problem in mobile robotics, found in a wide variety of applications ranging from search and rescue [1, 2] to industrial inspection [3]. Its definition varies between applications, and we identify two main reoccurring variations: *budget-constrained exploration*, and *quality-constrained exploration*. In budget-constrained exploration, it is assumed that the robot's budget is limited, e.g., by its battery, a deadline, or a camera roll. The budget is insufficient to fully cover the environment and the optimal set of views must be selected. In quality-constrained exploration, the constraint is flipped and it is assumed that the robot can cover its environment, and instead must collect views such that the whole map is of some minimum level of quality, i.e., the remaining *gain* falls below a given threshold.

Crucially, no part of the environment is left unexplored in quality-constrained exploration, implying that the total gain is essentially constant. The rich body of exploration methods that maximize gain must therefore be said to implicitly assume a budget-constrained scenario, because to maximize gain per sensor scan is to assume that some gain will be left unexplored, and that total gain is not constant.

The two paradigms are sometimes mixed up, such as by using a method suited for budget-constrained exploration but evaluating its quality-constrained properties, e.g., evaluating gain maximization by path length. In many cases, authors report counterintuitive results such as longer total path lengths and greedy behavior with improved gain estimation, owing to unnecessary backtracking as exploration draws to a finish [C, 4, 5]. Figure D.1 shows that path length increases with

stronger affinity to maximize gain, and that gain should in fact be *minimized*, if at all considered. This incongruence ultimately stems from the mix-up of the budget-constrained and quality-constrained paradigms: in budget-constrained exploration, obtaining gain is the priority, so greediness is, in fact, the desired outcome. Relatively little prior work can therefore be said to be targeted at the quality-constrained paradigm directly, which is the aim of this paper.

While some backtracking is inevitable, there can also be unnecessary backtracking. Oßwald et al. [5] show that unnecessary backtracking is incurred if and only if a given region is not yet explored by the *last* time that the optimal plan passes it. In other words, the robot typically has multiple opportunities to explore a given region, and unnecessary backtracking is guaranteed only once it has missed its last chance.

The optimal plan is, of course, not available to the robot, and it cannot know when it has its last opportunity to explore a region. A cue that this might be the case, however, is that the robot is closer to that region now than it is expected to be later; it has a rare opportunity to explore the region with low path cost. We dub this heuristic *distance advantage*.

The phenomenon where gain maximization causes worse performance was first identified in works aiming to improve exploration by leveraging additional knowledge [5, B]; consequently, in this paper, we propose a method for autonomous exploration that does not maximize gain, and instead centers on reducing backtracking by leveraging prior knowledge through our distance advantage heuristic.

The paper is outlined as follows: first, related work is analyzed through the lens of budget contra quality-constrained exploration in Section D.2, the quality-constrained exploration problem is then formally defined in Section D.3. Next, our proposed method is presented in Section D.4, the experimental setup and results in Sections D.5 to D.7, and finally, our conclusions and limitations are presented in Sections D.8 and D.9. To summarize, our main contributions are:

- the relationship between autonomous exploration and gain maximization is disentangled once and for all, explaining and justifying counterintuitive findings reported in existing literature;
- a novel method for autonomous exploration, distance advantage, is proposed, which directly aims to minimize future potential backtracking; and
- its performance is evaluated in simulation, producing significantly shorter paths in the quality-constrained paradigm than nearest frontier [6] or gain maximization [7].

The reader is encouraged to review the supplementary video material for this paper, as it offers insights that are hard to convey with text and images.

D.2 Related Work

Autonomous exploration was initially proposed in the context of active perception by Bajcsy [8], with the term being coined by Whaite and Ferrie [9]. In many tasks, a single measurement is insufficient, either due to the nature of the sensor or due to uncertainty. Therefore, it is necessary to plan where to place the sensor in order to collect measurements that most reduce uncertainty. Since the focus of active perception was on object reconstruction in a small workspace, the cost of the path necessary to move the sensor was not relevant. Therefore, these first works were direct extensions of the next-best-view methods by Connolly [10] from vision to robotics, by maximizing the predicted gain of the next measurement [8, 9, 11, 12].

The subtle conflation of budget- and quality-constrained exploration was already present in the active perception community. Whaite and Ferrie [9] defined the goal of autonomous exploration as that of determining representations of acceptable fidelity, i.e., quality-constrained exploration. However, they proposed to approach autonomous exploration through the lens of gain maximization, implicitly adopting the budget-constrained paradigm.

Carrying over to mobile robotics, where the cost of moving the robot cannot be neglected, the focus became to maximize gain with respect to the distance traveled [6, 7], remaining in the implicit budget-constrained paradigm. Finding the best trade-off between gathering information and moving efficiently has proven to be a challenging problem, leading to extensive research and the very term 'autonomous exploration' being appropriated by the mobile robotics community.

Yamauchi [6] argued that to efficiently observe the environment, the robot should plan to visit states that are predicted to have high gain while minimizing the path cost. The concept of frontiers, the border between known and unknown space, was introduced and it was proposed that exploration be done by navigating to the nearest frontier. Directly extending the next-best-view approaches, González-Baños and Latombe [7] proposed to explicitly optimize for measurement gain, weighted inversely to the distance necessary to collect it. Already when introducing gain maximization for mobile robotics, González-Baños and Latombe [7] observed that prioritizing gain led to fast short-term exploration, but ultimately made completing exploration take longer.

A wide body of literature exists building on Yamauchi [6] and González-Baños and Latombe [7], extending it to more complex scenarios and improving upon its assumptions. RH-NBV [13] evaluates the objective along a *path*, instead of a single step. While searching over paths scales combinatorially, when compared to single decisions, this can be dealt with through sampling-based planning. AEP [14] combines frontiers and RH-NBV to mitigate the effect of greediness due to gain maximization in the global path, while preserving the efficient local coverage performance of RH-NBV. Some works [15, 16], most notably FUEL by Zhou et al. [16], attempt to plan optimal tours for a Traveling Salesman Problem (TSP) that visits every frontier cluster. The gain of each frontier cluster can also be locally optimized in a refinement step [16]. RH-NBV and AEP require extensive gain

estimation, which can take up to 95% of planning time [17], while FUEL computes solutions to a TSP, which can be computationally demanding. Returning to single-step planning, Duberg and Jensfelt [18] show that determining frontiers to be states with a minimum amount of gain, the nearest frontier exploration strategy produces shorter paths than RH-NBV, AEP and FUEL, while being computationally cheaper. Yu et al. [19] combine Duberg and Jensfelt [18] and Zhou et al. [16], choosing the nearest gain-having frontier and then optimizing the viewpoint to improve gain. More works extend these ideas, by improving upon the sampling-based planner and the path cost function [17, 20, 21], or considering improved estimates of gain through learning-based methods [22, 23, 24, 25], among others.

Importantly, most works aim to efficiently complete coverage of unknown environments and therefore evaluate quality-constrained exploration metrics, such as time or path length. However, they are implicitly performing budget-constrained exploration since they maximize gain. Several counterintuitive results have been reported, such as that maximizing gain ultimately leads to longer coverage paths [5, 7], or that improved estimates of gain lead to longer paths [C].

Few works have explicitly addressed the mismatch between quality-constrained exploration goals and the budget-constrained optimization objectives. Li et al. [26] attempted to determine the optimal exploration path offline by using A* with an admissible heuristic, estimated from a complete map of the environment. The determined path is used to estimate the competitive ratio of other exploration strategies, showing that pure gain maximization is far from optimal. When an abstract topological map of the environment is available, a high-level global exploration path can be determined by solving a TSP [5]. The solution to the TSP can be used to determine the optimal order in which to visit frontiers, potentially using information gain to locally prioritize frontiers, similar to Selin et al. [14]. Ultimately, it is shown that using information gain leads to longer paths, and that strictly prioritizing the order in which frontiers are visited based on the TSP solution is better, since exploration is finished only when there are no more frontiers. Since more information about the environment does not lead to shorter paths for information gain, Ericson et al. [C] proposed a heuristic that prioritizes visiting frontiers that observe elements of the environment that cannot be observed together with other elements. This is shown to lead to shorter paths than gain maximization and to improve with access to more information about the environment.

Combining information gain with quality-driven exploration is possible, and, in fact, paramount to tasks like active SLAM. There, exploration must compete with active localization: the goal is to produce a sufficiently high-quality map, but that requires maintaining an accurate state estimate. Works by Stachniss et al. [27] and Zhang et al. [28] address this scenario, where the role of information gain is not to drive exploration, but instead as a form of exploitation for minimizing state uncertainty.

D.3 Problem Statement

A plan π is a sequence of states, i.e., $\pi = (s_0, \dots, s_T)$, with s connected to its successor s' by the shortest path, with length $d(s, s')$. As the robot follows a plan, it builds a map M_π of the environment. The problem of determining a plan for quality-constrained exploration can then be formulated as

$$\min_{\pi} \; d(\pi) \quad s.t. \; \text{DesiredCoverage}(M_\pi), \tag{D.1}$$

where $d(\pi)$ is the total length of the plan. However, it is typically not possible to evaluate the coverage or feasibility of a complete plan, since at best the environment is partially known. Therefore, in most cases an exploration plan cannot be obtained offline by solving Eq. (D.1).

When performing exploration online, the exploration plan at any given time can be decomposed into three components: the plan already followed up to state s, the next state t to be determined, and the unknown optimal plan π^* that follows it. Then, Eq. (D.1) can be reformulated as the sequential decision problem of determining the optimal next state

$$t^* = \underset{t \in C}{\operatorname{argmin}} \Big(d(s, t) + d(\pi^*) \Big), \tag{D.2}$$

where C are the states that will improve the coverage of the map. While evaluating C and $d(\pi^*)$ is still not possible without knowing the environment, we now discuss how to address these problems when only the current map and, possibly, local predictions are available.

Like coverage, it is not possible to determine all states C that can improve coverage without knowing the whole environment. However, it is enough to consider the states F that advance coverage and are nearest to the current state, since any path to C must pass through one of these boundary states. F can be determined by finding the states that collect at least a minimum gain [18] or, more commonly, approximated by frontiers [6].

Since the states that improve coverage depend on the map and, through it, on the already followed plan, the decision problem suffers from the curse of history and is intractable to solve. Therefore, determining a good heuristic estimate of $d(\pi^*)$ is one of the fundamental problems of autonomous exploration planning. Predictive ability of the environment for autonomous exploration should reflect in better heuristic estimates and, consequently, better exploration plans. However, it has been reported in the literature that existing heuristics do *not* improve with better predictions [C], indicating there is room for improvement in the planning heuristic.

D.3.1 Traditional Exploration Heuristics

Having identified the sequential decision problem corresponding to quality-constrained exploration, it is now possible to analyze the implicit assumptions of the traditional

autonomous exploration heuristics: nearest frontier and information gain.

Nearest frontier [6, 18]: The exploration plan is obtained through

$$t^* = \operatorname*{argmin}_{t \in F} d(s,t), \tag{D.3}$$

implicitly assuming every frontier has an equal-length optimal plan it can follow afterwards.

Gain maximization [7, 13]: The gain $G(s,t)$ is the increase in coverage obtained by following the shortest path from s to t. Its estimate is typically an upper bound [7, 13], but can be improved using learning-based approaches [B, 23, 24]. The exploration plan is obtained through

$$t^* = \operatorname*{argmax}_{t \in F} \Big(\lambda \log G(s,t) - d(s,t) \Big), \tag{D.4}$$

where λ determines the affinity of gain maximization relative to path cost. Determining an exploration path by maximizing gain corresponds to assuming that collecting more information on the path to t will lead to a shorter optimal path afterward.

D.4 Distance Advantage

In order to design a quality-constrained exploration planner, a suitable heuristic has to be found. It has already been pointed out by Oßwald et al. [5] that unnecessary backtracking happens when a frontier state is not explored by the *last* time that the optimal plan comes close to it. Therefore, a suitable heuristic for quality-constrained exploration is one that minimizes backtracking by determining which frontiers are unlikely to be revisited.

An indication that a frontier might not be revisited is that it is isolated, i.e., it has high average distance to other reachable states R. Therefore, we propose to determine an exploration plan through

$$t^* = \operatorname*{argmax}_{t \in F} \left(\frac{1}{|R|} \sum_{s' \in R} d(t,s') - d(s,t) \right), \tag{D.5}$$

where $|R|$ is the number of reachable states. This heuristic prefers to visit frontiers that are near the robot, but on average isolated from reachable space, i.e., frontiers that the robot has a *distance advantage* to visit.

The reachable states are determined using the map and, if available, map predictions, which we show lead to an improvement in the estimate. Determining an exploration path through Eq. (D.5) corresponds to assuming that the cost of the optimal path after the frontier is lower if the frontier is more isolated, since leaving it behind would cause backtracking. In Fig. D.2, an example exploration state in an occupancy grid environment is shown, with unexplored cells colored according to their distance advantage and frontier cells highlighted with solid coloring.

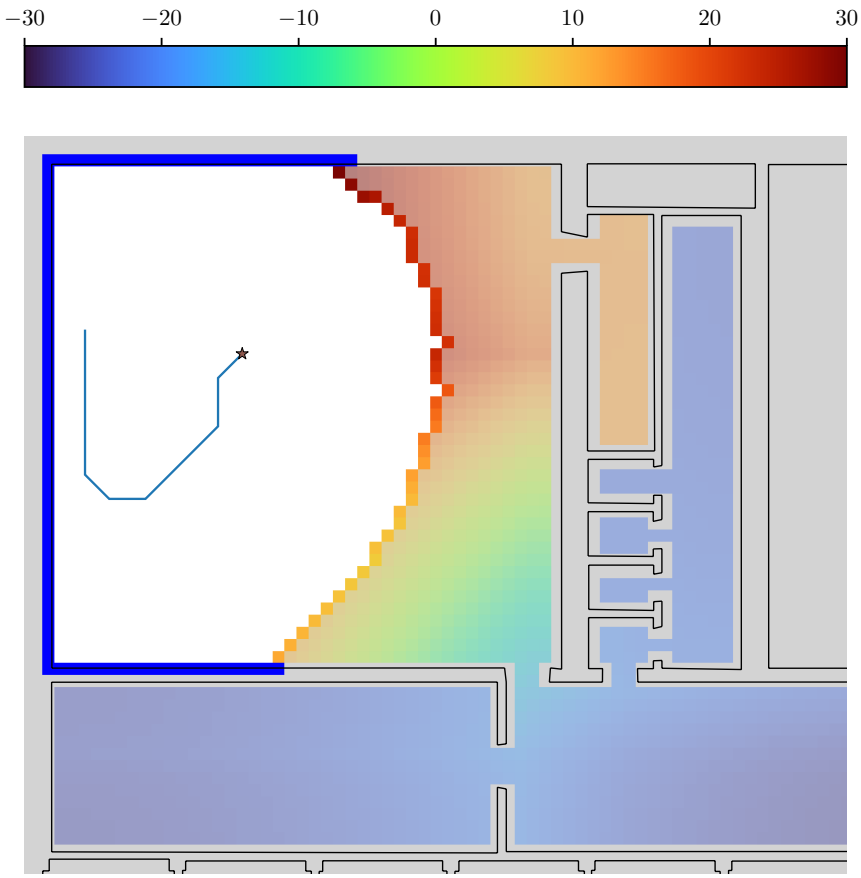
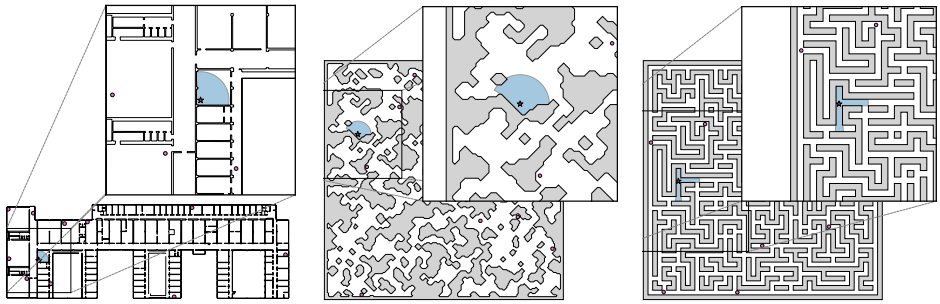


Figure D.2: Illustration of distance advantage in the beginning of exploration. The robot (star) preferentially explores frontiers (solid coloring) with higher distance advantage. It is heading towards a closed off room because it is nearer that region than it would be from most other places. By contrast, its distance to the corridor is higher than it would be elsewhere, repelling it from that region.

Computing the distance advantage requires determining the shortest path distance from every frontier to every reachable state. In a graph-like environment, computing shortest path distances requires as many single-source shortest path problems as there are frontiers, and the computational cost for each scales with the size of the graph, i.e., the number of reachable states. Therefore, in order to keep the computational cost bounded regardless of the environment size, only frontiers and reachable states that are in a local window centered around the current state are considered. If the environment is a weighted graph, $d(t, s')$ is determined using Dijkstra's algorithm, while breadth-first search is used for unweighted graphs.



(a) Office environment, 4755 m^2 interior in a 132×44 m^2 bounding box.

(b) Cave environment, 3651 m^2 interior in a 80×80 m^2 bounding box.

(c) Maze environment, 1300 m^2 interior in a 49×49 m^2 bounding box.

Figure D.3: Data is collected in three diverse environments: a large office from a real-world floor plan with both small cubicles and large lecture halls, a non-rectilinear cave environment with many small pockets, and a labyrinth-like maze with both shallow and deep dead-ends. Pink circles indicate starting locations, the light blue region depicts a sensor scan from the point of view of an example starting location indicated by the brown star polygon. The zoomed region is the same size as a local window for the planner.

Frontiers that are outside the local window are not considered unless the local window has been fully explored, in which case the fallback planning chooses the nearest frontier outside the local window.

D.5 Experimental Setup

In Sections D.6 and D.7, experiments are conducted to evaluate distance advantage, nearest frontier, and information gain with respect to their quality-constrained exploration performance, and their sensitivity to predictions. The experiments are conducted in simulation in order to fairly compare the planning objectives. This section describes the simulation environment and other relevant implementation details.

D.5.1 Sensor & Mapping

The robot is simulated as a point-like sensor, with holonomic motion capabilities. The simulated sensor is a 360° laser range sensor, with 720 evenly spaced rays and a maximum range of 4.5 m. As the robot moves and collects sensor scans, these are accumulated into an occupancy grid map that discretizes the environment into 25×25 cm^2 cells and marks them as either free, occupied, or unknown. The scans

are accumulated conservatively, such that 8-connected paths through free space in the map are guaranteed to be collision-free.

D.5.2 Localization & Path Execution

When the next state is chosen by the planner, a shortest 8-connected path is found through exhaustive search in the map using Dijkstra's algorithm. Since there are often many such shortest paths, the one which keeps the most distance from walls is chosen. As the path is executed, at each step in the map, a new sensor scan is obtained. The path terminates when the goal state is reached or when the sensor scan causes a map cell that was unknown to be marked as occupied or free.

Since the path is executed deterministically, the robot is perfectly localized with respect to the starting state.

D.5.3 Predictions

A map predictor, e.g., [B, 24, 29], is assumed to be available and capable of providing map completions in a local window centered around the current state. The map predictor extends the map with the real environment and the local window size is $30 \times 30\,\mathrm{m}^2$.

D.6 Quality-Constrained Evaluation

For the purposes of autonomous exploration, environments can be characterized by their connectivity. One end of the spectrum is a perfectly tree-like environment, e.g., a maze without loops. In this case, it is practically irrelevant which frontier is chosen, as long as it is explored to completion, since the robot must always return to the branching point to pursue the next frontier. On the other end of the spectrum is a fully connected environment, where the unexplored space beyond every frontier eventually connects. Real environments are somewhere in the middle of this spectrum, with some branches interconnected and some branches dead ends, e.g., an office environment or a cave system.

The three heuristics are evaluated in three environments with different characteristics, presented in Fig. D.3: a large office, with a wide variety of rooms-within-rooms, looping corridors, and wide-open spaces; a cave-like environment with high global connectivity, but low local connectivity; and a maze with a relatively small amount of loops, mostly consisting of dead-end branches of varying depth. The distance at completion for all heuristics in each environment is shown in Table D.1. Since nearest frontier is a special case of information gain (IG), with no preference for information, it is used as a reference.

In the office environment, which has the most complex connectivity of the three, distance advantage obtains an improvement of 16 % over nearest frontier, while information gain is 23 % worse. As the connectivity of the environments simplifies, the margin for improvement or degradation over nearest frontier decreases, since the

Table D.1: Distance at completion for each method in each environment of Fig. D.3. Data collected across 10 runs for each method/environment pair from different starting locations. The difference to nearest frontier is computed per starting location.

	Distance at Completion d_T (m)		
Method	Office	Cave	Maze
Nearest Frontier	1892.5 ± 50.5	1854.3 ± 106.1	1377.8 ± 36.0
Dist. Adv.	1578.1 ± 43.9	1652.1 ± 93.4	1249.5 ± 27.5
ΔNearest Frontier	-303.6 ± 35.0	-194.8 ± 68.8	-128.3 ± 45.3
IG Max.	2330.3 ± 71.8	2313.2 ± 133.0	1514.1 ± 40.9
ΔNearest Frontier	437.8 ± 89.8	407.5 ± 102.1	136.3 ± 60.9

simpler connectivity leads to there being more opportunities for sub-optimal policies to correct mistakes at a low cost. These results show that distance advantage consistently obtains shorter paths, with the difference being more noticeable in environments with more complex connectivity. Information gain is consistently outperformed by nearest frontier and, by extension, distance advantage.

D.6.1 Greediness

Information gain has previously been shown to lead to greedy behaviors, due to sacrificing long-term exploration performance for early short-term gains [C, 5]. In order to evaluate how greediness affects performance, the coverage $c(d)$ and frontier size $f(d)$ as functions of the distance traveled d were considered, as is shown in Fig. D.4.

There are nearly no differences in the coverage rate early on, indicating that the information gain maximization strategy is not successful in doing so. However, there is a large difference in the number of the frontiers, with nearest frontier quickly growing to double the amount of distance advantage, and gain maximization more than doubling nearest frontier. In accordance with [C], these outstanding frontiers represent a kind of debt to be paid, in the form of travel distance at the end of the run. This outstanding frontier debt explains why the coverage rate for information gain, and to a lesser extent nearest frontier, decreases as exploration progresses and ultimately leads to a longer path. By contrast, the frontier size for distance advantage remains approximately constant during most of the exploration run, allowing it to keep an almost constant coverage rate until the end of exploration.

D.6.2 Gain Maximization Affinity

When first introducing gain maximization for autonomous exploration, González-Baños and Latombe [7] already highlighted that lower affinity λ led to a more

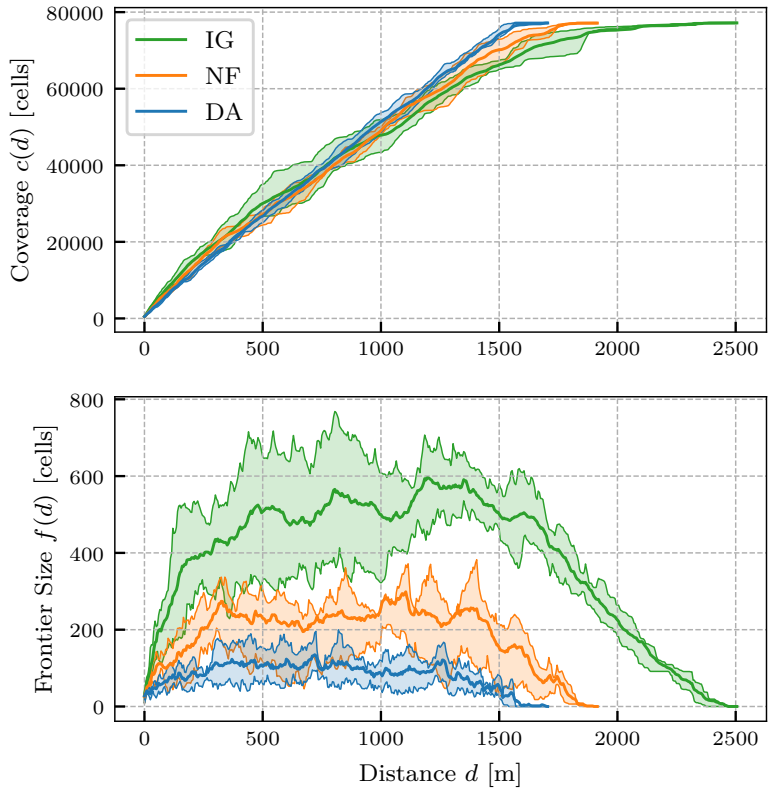


Figure D.4: Comparison of coverage $c(d)$ and total frontier size $f(d)$ as functions of distance traveled d. The shaded areas indicates an 80 % confidence interval, the solid line indicates the mean. Data collected across 10 runs for each method from different starting locations in the office environment.

meticulous covering of the environment. We evaluate the effect of gain maximization affinity on path length with two gain estimators: naive, which assumes unknown space is non-occluding, and true, which has access to the true gain. The resulting completion distances are shown in Figure D.1, clearly showing that higher affinity leads to longer paths and that the effect is worsened with more accurate estimates.

An interesting effect that, to the best of the authors' knowledge, has not been reported is that *negative* affinity leads to shorter paths than nearest frontier. The fact that preference for lower gain improves performance can be understood through the lens of isolation, since a low predicted gain amounts to predicting that a frontier region is soon to terminate. In that way, the preference for low gain is an *ad hoc* heuristic for preferring shallow frontiers, that are unlikely to be revisited by the optimal path since they do not continue deeper into unknown space.

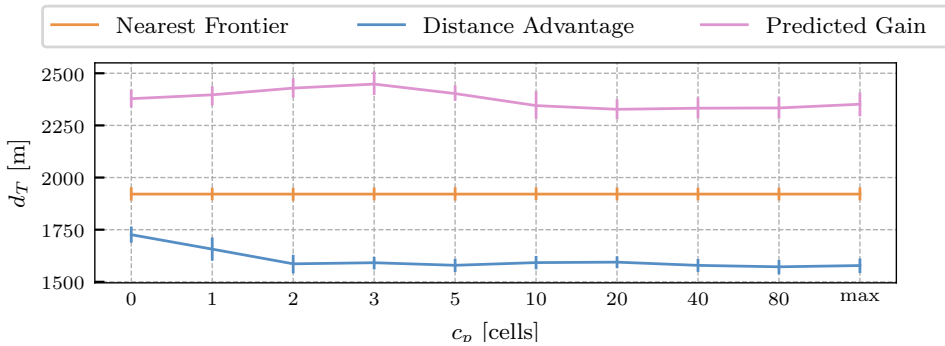


Figure D.5: The effect of prediction range c_p on completion distance d_T in the office environment. Data collected across 10 runs from different starting locations, for each method and prediction range. Error bars represent one standard deviation.

D.7 Sensitivity to Predictions

The previous experiments all consider perfect predictions, given by an oracle that knows the true environment. However, in a real exploration scenario, these predictions would come either from prior knowledge of the environment, e.g., a floor plan, or from learning-based models [B, 24, 29]. In those cases, the predictions will not perfectly reflect the environment, and it is important to assess the sensitivity of each method to the predictions, with respect to both the amount of predicted information and its accuracy. Nearest frontier is completely insensitive to predictions and is presented as a baseline.

D.7.1 Prediction Range

The amount of information made available through the predictor could have a large influence in the exploration plan. While Ericson and Jensfelt [B] showed that it is possible to predict $30{\times}30\,\mathrm{m}^2$ windows from the occupancy map built by the robot in office environments, this might not be possible in more complex environments or too computationally demanding in some situations. Therefore, we examine the effect of the prediction range beyond the frontier, parameterized by the maximum straight line distance c_p in number of cells.

Figure D.5 illustrates how different settings of c_p affect performance for all methods in the office environment. It can be seen that the ranking of the methods does not change in the absence of predictions ($c_p = 0$), and distance advantage continues to outperform nearest frontier. Most of the gain that distance advantage gets from the predictions is already realized at $c_p = 2$, which corresponds to a prediction range of only $\sim 50\,\mathrm{cm}$ beyond the frontier. In agreement with the results reported by Ericson et al. [C], information gain fails to take advantage of predictions.

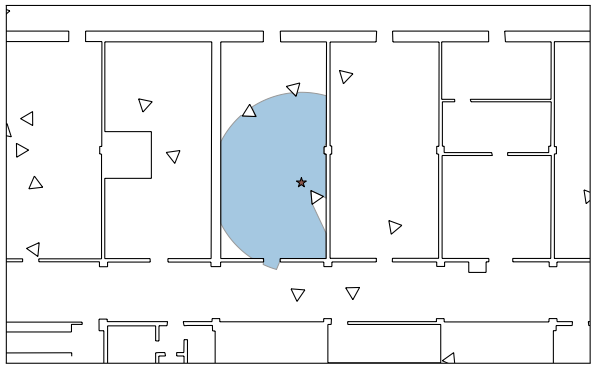


Figure D.6: An example of clutter for the office environment. The clutter consists of polygons, roughly 1 m, randomly placed in the reachable environment. A section of the environment does not get disconnected from the rest due to the clutter closing a passage, like the clutter of actual human environments.

D.7.2 Prediction Accuracy

A common mode of failure for predictions is non-structural elements of the environment such as furniture, human occupants, other robots, etc., since those are likely to move or be moved around the environment. In the office environment, the floor plan is fixed and predictions of it might be accessible, but elements like desks, chairs, etc., can be unpredictable. A different source of prior knowledge could also be an old map, in which case the mismatch could go the other way; the clutter in the map might no longer reflect the true state of the environment. To assess the impact of this kind of mismatch between predictions and the actual environment, clutter was generated for the office environment in Fig. D.3a by randomly sampling triangles, illustrated in Fig. D.6. The same set of triangles is used for every method, but different sets are used between starting locations, predictions and the true environment. The results of this evaluation are presented in Table D.2.

Clutter is a source of occlusion, so when the environment is cluttered it is harder to observe the environment from far away, causing all methods to produce longer paths. While nearest frontier produces $\sim 150\,\text{m}$ longer paths, distance advantage and information gain degrade by $\sim 200\,\text{m}$. This suggests that the 150 m increase can be explained due to the cluttering of the environment making observing it harder, with the inaccuracy in predictions accounting for the remaining 50 m. Importantly, the way in which the predictions are wrong, i.e., whether they do not contain clutter or they contain mismatched clutter, does not have a significant influence on the performance.

Table D.2: Distance at completion when there is mismatch between the predictions and the environment, due to clutter. The environment and prediction clutter are independently sampled. Data collected across 10 runs for each method/environment/prediction tuple from different starting locations.

Method	Environment	Prediction	d_T (m)
Nearest Frontier	Clean	Clean	1892.5 ± 50.5
	Noise	Clean	2041.7 ± 57.4
	Noise	Noise	2054.0 ± 32.1
Dist. Adv.	Clean	Clean	1578.1 ± 43.9
	Noise	Clean	1782.3 ± 52.1
	Noise	Noise	1812.0 ± 14.8
IG Max.	Clean	Clean	2351.6 ± 85.2
	Noise	Clean	2567.8 ± 76.8
	Noise	Noise	2563.0 ± 85.6

D.8 Limitations

The main focus of this work is to highlight how the current state-of-the-art autonomous exploration planning methods ultimately do not optimize the correct objectives. Although the proposed objective is well-motivated both by intuition and by empirical results, a more thorough theoretical analysis has the potential to yield even better formulations. Future work should attempt to characterize the fundamental limits of the proposed objective and improve upon it. Another aspect that warrants additional work is the computational scalability of estimating the optimization objective. Since the proposed objective requires solving a multi-source shortest path problem, the computational time scales with the size of the explored map. This issue was addressed by limiting the computation to a fixed-size local map, but future work should investigate whether it is possible to improve upon this solution.

The experimental results are limited to the minimal case, with no uncertainty and perfect mapping.

D.9 Conclusion

In this work, we highlight the important differences between budget- and quality-constrained exploration, and address the inconsistencies observed in the autonomous exploration community. We investigate why traditional heuristics, such as information gain and nearest frontier, perform poorly in the quality-constrained paradigm, and propose a new heuristic for quality-constrained exploration. Since quality-constrained exploration is defined to be completed when the map is of sufficient quality, total gain is fixed; maximizing information gain of individual frontiers is

therefore ultimately irrelevant as all the gain will be collected eventually. Therefore, the central problem in quality-constrained exploration planning is to determine the correct order in which to explore frontiers, such that the length of unnecessary detours is minimized.

We propose a heuristic, named *distance advantage*, that attempts to identify which frontiers have higher opportunity cost if missed, i.e., those more likely to require a detour later in exploration if they are not explored now, by estimating their average distance to other states. This heuristic is compared to nearest frontier and information gain, and it is shown to consistently explore the environments with shorter paths. Perhaps most importantly, among the evaluated heuristics, distance advantage is the only one to show improvements in performance as access to predictions improves, and is able to handle imperfect predictions. We believe these results clearly show the different nature of budget- and quality-constrained exploration, and indicate that further work should be done in understanding the correct objective for quality-constrained exploration, and exploring the proposed objective.

References

[1] D. Calisi, A. Farinelli, L. Iocchi, and D. Nardi, "Autonomous Exploration for Search and Rescue Robots", *Transactions on the Built Environment*, 2007.

[2] F. Colas, S. Mahesh, F. Pomerleau, M. Liu, and R. Siegwart, "3D Path Planning and Execution for Search and Rescue Ground Robots", in *International Conference on Intelligent Robots and Systems*, IEEE, 2013.

[3] S. Omari, P. Gohl, M. Burri, M. Achtelik, and R. Siegwart, "Visual Industrial Inspection using Aerial Robots", in *International Conference on Applied Robotics for the Power Industry*, IEEE, 2014.

[C] L. Ericson, D. Duberg, and P. Jensfelt, "Understanding Greediness in Map-Predictive Exploration Planning", in *European Conference on Mobile Robots*, IEEE, 2021. DOI: 10.1109/ECMR50962.2021.9568793.

[4] M. Luperto, M. M. Ferrara, G. Boracchi, and F. Amigoni, "Estimating Map Completeness in Robot Exploration", 2024. arXiv: 2406.13482.

[5] S. Oßwald, M. Bennewitz, W. Burgard, and C. Stachniss, "Speeding-up Robot Exploration by Exploiting Background Information", *Robotics and Automation Letters*, vol. 1, no. 2, pp. 716–723, 2016.

[B] L. Ericson and P. Jensfelt, "Beyond the Frontier: Predicting Unseen Walls from Occupancy Grids by Learning from Floor Plans", *IEEE Robotics and Automation Letters*, 2024. DOI: 10.1109/LRA.2024.3410164.

[6] B. Yamauchi, "A Frontier-based Approach for Autonomous Exploration", in *International Symposium on Computational Intelligence in Robotics and Automation*, IEEE, 1997.

[7] H. H. González-Baños and J.-C. Latombe, "Navigation Strategies for Exploring Indoor Environments", *The International Journal of Robotics Research*, 2002.

[8] R. Bajcsy, "Active Perception", *Proceedings of the IEEE*, vol. 76, no. 8, pp. 966–1005, 1988.

[9] P. Whaite and F. P. Ferrie, "Autonomous Exploration: Driven by Uncertainty", *Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 3, pp. 193–205, 1997.

[10] C. Connolly, "The Determination of Next Best Views", in *International Conference on Robotics and Automation*, IEEE, vol. 2, 1985, pp. 432–435.

[11] J. Maver and R. Bajcsy, "Occlusions as a Guide for Planning the Next View", *Transactions on Pattern Analysis and Machine Intelligence*, 1993.

[12] R. Pito, "A Sensor-based Solution to the "Next Best View" Problem", in *International Conference on Pattern Recognition*, IEEE, 1996.

[13] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding Horizon "Next-Best-View" Planner for 3D Exploration", in *International Conference on Robotics and Automation*, IEEE, 2016.

[14] M. Selin, M. Tiger, D. Duberg, F. Heintz, and P. Jensfelt, "Efficient Autonomous Exploration Planning of Large-Scale 3D Environments", *Robotics and Automation Letters*, 2019.

[15] M. Kulich, J. Faigl, and L. Přeučil, "On Distance Utility in the Exploration Task", in *International Conference on Robotics and Automation*, IEEE, 2011.

[16] B. Zhou, Y. Zhang, X. Chen, and S. Shen, "FUEL: Fast UAV Exploration using Incremental Frontier Structure and Hierarchical Planning", *Robotics and Automation Letters*, 2021.

[17] L. Schmid, M. Pantic, R. Khanna, L. Ott, R. Siegwart, and J. Nieto, "An Efficient Sampling-Based Method for Online Informative Path Planning in Unknown Environments", *IEEE Robotics and Automation Letters*, 2020.

[18] D. Duberg and P. Jensfelt, "UFOExplorer: Fast and Scalable Sampling-based Exploration with a Graph-based Planning Structure", *Robotics and Automation Letters*, 2022.

[19] J. Yu, H. Shen, J. Xu, and T. Zhang, "ECHO: An Efficient Heuristic Viewpoint Determination Method on Frontier-based Autonomous Exploration for Quadrotors", *Robotics and Automation Letters*, 2023.

[20] B. Lindqvist, A.-A. Agha-Mohammadi, and G. Nikolakopoulos, "Exploration-RRT: A multi-objective Path Planning and Exploration Framework for Unknown and Unstructured Environments", in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[21] C. Witting, M. Fehr, R. Bähnemann, H. Oleynikova, and R. Siegwart, "History-Aware Autonomous Exploration in Confined Environments using Mavs", in *International Conference on Intelligent Robots and Systems*, IEEE, 2018.

[22] D. Deng, R. Duan, J. Liu, K. Sheng, and K. Shimada, "Robotic Exploration of Unknown 2D Environment using a Frontier-based Automatic-Differentiable Information Gain Measure", in *International Conference on Advanced Intelligent Mechatronics (AIM)*, IEEE, 2020.

[23] L. Schmid, C. Ni, Y. Zhong, R. Siegwart, and O. Andersson, "Fast and Compute-efficient Sampling-based Local Exploration Planning via Distribution Learning", *Robotics and Automation Letters*, 2022.

[24] R. Shrestha, F.-P. Tian, W. Feng, P. Tan, and R. Vaughan, "Learned Map Prediction for Enhanced Mobile Robot Exploration", in *International Conference on Robotics and Automation*, IEEE, 2019.

[25] Y. Tao, Y. Wu, et al., "SEER: Safe Efficient Exploration for Aerial Robots using Learning to Predict Information Gain", in *International Conference on Robotics and Automation*, IEEE, 2023.

[26] A. Q. Li, F. Amigoni, and N. Basilico, "Searching for Optimal Off-Line Exploration Paths in Grid Environments for a Robot with Limited Visibility", in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2012.

[27] C. Stachniss, G. Grisetti, and W. Burgard, "Information Gain-based Exploration Using Rao-Blackwellized Particle Filters", in *Robotics: Science and Systems I*, Robotics: Science and Systems Foundation, 2005.

[28] Y. Zhang, B. Zhou, L. Wang, and S. Shen, "Exploration with Global Consistency Using Real-Time Re-integration and Active Loop Closure", in *International Conference on Robotics and Automation*, IEEE, 2022.

[29] M. Luperto, F. Amadelli, M. Di Berardino, and F. Amigoni, "Mapping Beyond What You Can See: Predicting the Layout of Rooms Behind Closed Doors", *Robotics and Autonomous Systems*, 2023.

So long, and thanks for all the fish.