MIRROR: Mirroring Teammate Knowledge for Multi-Robot Exploration under Communication Constraints

Federico Baldan

A thesis

submitted in partial fulfillment of the

requirements for the degree of

Master of Science

University of Washington

2026

Committee:

Mehran Mesbahi

Jonathan P. How

Program Authorized to Offer Degree:

Aeronautics and Astronautics

University of Washington

Abstract

MIRROR: Mirroring Teammate Knowledge for Multi-Robot Exploration under Communication Constraints

Federico Baldan

Chair of the Supervisory Committee:

Mehran Mesbahi

William E. Boeing Department of Aeronautics and Astronautics

Teams of robots explore an unknown environment faster than a single robot only if they tell each other what they have seen. Most exploration systems assume that two robots in contact merge their maps completely, and on real platforms that assumption is not always feasible, since a link often carries far less than a full map. This thesis therefore takes a communication budget as its central assumption. A contact then delivers a small part of what a robot knows, so transmitting becomes selecting, and in deployed systems that selection falls to the order in which the map sits in memory.

The thesis formalizes this problem and reduces it to one choice, the order in which the

unsent cells travel. It then presents MIRROR, a policy that makes that choice deliberately. The sender maintains an estimate of what the receiver already knows, built from the cells it has delivered and the positions at which it has observed that teammate, and it sends first the cells beyond the edge of that estimate. Existing answers to a thin channel shrink the payload or negotiate its content over the channel itself, so the choice is either left undesigned or paid for in the bandwidth it was meant to save. MIRROR needs no extra message and no reply, so the estimate costs the channel nothing, and it fits any system that already exchanges positions.

MIRROR is evaluated in simulation on six real building floor plans, one of them held out until a single blind run, with teams of two to seven robots, sensor ranges from 5 to 40 m, and budgets spanning two orders of magnitude, against policies from silence to unlimited exchange and an ordering that reads the receiver’s true map. Aiming the messages makes a team explore faster and waste less. With five percent of a full exchange per contact, MIRROR recovers between 93 and 98 percent of what communication can give on every floor tested, returns between a quarter and a half of the time a silent team spends, and removes nearly all of the duplicated sensing that consumes between 44 and 64 percent of a silent team’s effort.

Acknowledgments

I would like to start by thanking Professor Jonathan How of MIT for giving me the opportunity to be part of the ACL lab and for guiding me through this research journey. It has been a privilege to receive his advice and to have him dedicate his time to meetings, ideas, and projects.

I would also like to thank Professor Mehran Mesbahi of the University of Washington, who likewise gave me the freedom and the opportunity to carry out my research thesis in collaboration with his laboratory. His advice and his help in the difficult moments were invaluable to me.

I would like to thank all my labmates at the Aerospace Controls Lab at MIT for welcoming me in the best possible way and for making me feel at home from the very beginning.

I would like to thank my family: my mom Luana, my dad Roberto, and my sister Francesca, for their support throughout my university years and for always encouraging me to follow my passions. My grandparents: Egidio, Sandra, Rossana, with whom I grew up. My American mother, Laura, for everything she has done for me. And all the Mercer Island families: the Medveds, the Weinsteins, Lloyd, and Craig, for always welcoming me.

Finally, I would like to thank all my lifelong friends, from Ravenna all the way to Boston. In particular, Massi, Gugli, Fadda, Paolino (Grim), Lori, Brusio, Dodo, Niko, Lacca, Ettore, Avishai, and the whole crew from Ravenna and Torino. A special thanks to Kosher Suits and Menachem Altein. Thanks to Nathan, Harold, and the Traimans.

Cambridge, MA, August 19, 2026 Federico Baldan

Contents

List of Figures

List of Tables

1 Introduction

Autonomous exploration is the problem of mapping an environment for which no prior map is available. The robot must decide where to go next using only the partial map constructed from its past observations. There are many applications that motivate autonomous exploration, especially those in which sending humans first is not an option: collapsed buildings after an earthquake, mines and tunnels after an accident, caves under Earth, etc. [1, 2]. Applications also exist outside hostile settings. Fleets of robots in warehouses and logistics facilities must build and maintain a model of a space that keeps changing around them. Different missions impose different constraints. A mission may limit the time available, the hardware the robots can carry, the communication between them, or the terrain they can move through. Robots should be able to explore efficiently in the presence of these constraints, because the sooner they have mapped what the mission requires, the sooner the mission is accomplished. We will return to these distinctions throughout the thesis. Deploying a team of robots enables greater efficiency, as multiple parts of the environment can be explored simultaneously. In principle, n robots can split the environment and finish close to n times faster than one. Whether they actually do depends on coordination, and coordination depends on communication. Robots that have not received what their teammates have already seen will then re-explore the same ground, and the potential speedup is lost. Coordination, in turn, does not come for free. More robots mean more information to exchange and more decisions to keep consistent across the team. Throughout this thesis, communication is the binding constraint: robots can exchange data only when they are near each other, and only a limited amount at a time.

The framework underlying most exploration systems has remained largely consistent since the work of Yamauchi [3]. Each robot maintains an occupancy grid, a discretization of the environment into cells labeled unknown, free, or occupied, and updates it with every sensor sweep. On this grid it computes its *frontier*, the known free cells that border unknown ones, which is exactly where its map can still grow. The exploration loop is then simple. The robot chooses a frontier location, typically the nearest one or the best under a utility function [4, 5]. It travels there, senses, updates the map, and repeats until no frontier remains. Exploration can rely on different sensing modalities, most commonly cameras and lidar. This work takes lidar as its reference, the usual choice for mapping tasks. It measures

range directly, carving free and occupied cells along each beam, and it works regardless of lighting. Exploration algorithms differ widely in what they do with this representation, in how frontiers are scored, in which one a robot commits to, and in how they are divided among teammates. However, the frontier itself is the common baseline on which all of them decide. The multi-robot version adds one more ingredient: when robots come within communication range, they exchange information. The exchange can take different forms: entire maps [6], individual map cells [7], positions [8], and more. Taking full map exchange as the reference, a received map benefits the receiver in two ways. The first is in revealing new ground. The regions the teammate explored enter the receiver's map, so the receiver learns about them without traveling there, and the new frontier becomes available to explore. The second benefit is avoiding travel. Some of the receiver's frontier cells lead into regions the teammate has already explored. Now, those cells are no longer frontier, so the receiver stops planning trips toward ground the team already knows.

What the literature almost always assumes is that robots in contact merge their maps instantly and without loss, at every step of the encounter, as the survey of Amigoni et al. [9] documents. On real platforms the assumption often fails. This is for two reasons. Robots may be unable to communicate at all, because the link depends on distance and on what stands between them. When a link does hold, the channel may carry far less than a map exchange requires. Both limits vary with the mission and the environment. Underground, radio signals decay so quickly that robots separated by a few walls or a bend in a tunnel effectively lose contact, as Saboia et al. [10] report. In an open field, the same team may stay connected for the whole mission. The platforms matter as much as the terrain. A ground vehicle can usually afford a high-bandwidth radio, because it has the mass, the power, and the antenna room to carry one. Aerial robots carry low-power links whose throughput is orders of magnitude lower, down to kilobits per second for the long-range radios used in the recent deployments of Bayer and Faigl [8]. The amount of data a robot needs to share varies just as much, because it depends on what the robot senses: a lidar produces dense point clouds, a camera produces imagery, and even the occupancy grid distilled from them must be encoded as bits to be shared. Communicating information has a concrete cost. The more a robot has explored, the more bits its knowledge translates into. For maps of realistic extent, the translation reaches megabits. A designer therefore cannot count on the communication link to deliver everything a robot has gathered. A realistic assumption for certain types of exploration is that what a robot needs to share is far more than what one encounter can deliver. A map keeps growing with everything the robot sees, while the link carries only what the radio and the environment allow. How wide the gap is depends on where the mission takes place and on what the robots carry.

This mismatch creates a problem that is easy to overlook. Consider two robots that meet after exploring separately. Each holds thousands of map cells that the other has never seen, and the encounter can carry only a small fraction of them before the robots move on. Something must therefore decide which part of what a robot knows is transmitted. The rest has to wait for a future meeting that may never happen.

Existing work has concentrated on shrinking the exchange. Some systems transmit only the difference accumulated since the last meeting rather than the whole map [11]. Others compress the map into compact representations, such as the continuous mixture models of Corah et al. [12], the sorted and compressed grids of Shi et al. [13], or the topological graphs of Zhang et al. [14]. Such techniques reduce how much data must travel. None removes the underlying decision. Whenever the budget is smaller than the payload, a selection among the candidate cells is made, whether someone designed it or not.

In systems deployed today, nobody designs this selection. The message is typically filled with whichever cells come first in the order that the map is stored in memory, row by row from one corner of the grid, so the content a teammate receives is usually decided by memory layout, and memory layout reflects nothing about what the receiver needs. That problem is the subject this thesis will address. Once the candidate cells and the budget are fixed, the algorithm contains exactly one free choice, namely which few of the many unsent cells fill the message. Current practice never makes that choice deliberately. Making it well is hard because of a circularity. A transmitted cell is useful only if it lands where the receiver's own knowledge ends, on the boundary where the receiver is exploring. Otherwise, it either repeats what the receiver already knows or describes ground the receiver cannot yet reach. Aiming at that boundary would seem to require the receiver's map, and the receiver's map is exactly what the limited channel keeps from the sender. The problem is therefore not only which cells to choose but how to choose among options without seeing the target.

The main objective of this thesis is to break this circularity and make the one free choice deliberately. The starting observation is that the sender is not as blind as it seems. Two records it already owns say a great deal about what the teammate knows. The first is the list of cells it has itself delivered to that teammate. The second is the set of positions at which it has observed the teammate, each of which implies the ground that teammate has sensed for itself. Neither record costs a single bit to obtain, and neither requires the receiver to answer. This thesis introduces **MIRROR**, a sharing policy built on exactly that. The sender maintains a mirror image of the receiver's map from those two records, and fills each message with the cells that push the boundary of that estimate outward. The estimate is not a map the two robots agree on, and it is not obtained by asking. It is the sender's own inference, and every message it sends updates it, so the aim sharpens as the exchange proceeds without any coordination. The policy is also robust against interrupted contacts. No matter when the link drops, the cells already sent are the ones judged most valuable to the receiver. The rest of the thesis develops this idea formally and evaluates it in simulation. The benchmark sweeps the communication budget from links that barely carry anything to the unlimited exchange of the standard setting. It asks one question: When a contact cannot carry everything, how much of the benefit of full communication can a sender preserve by wisely choosing its few transmitted cells?

1.1 Contribution

In short, the primary contributions of this thesis are as follows:

- We formalize the what-to-send problem for decentralized exploration under a limited communication channel, and we reduce it to a single design choice, the order in which a robot's unshared cells are transmitted. The reduction gives algorithm design an approachable target, since every policy the protocol admits is some ordering of the same pool, and two policies can therefore be compared at an identical number of bits.

- We present MIRROR, a policy that makes that choice by estimating what the receiver already knows, and it pays nothing for the estimate. The sender infers the receiver's map from two records it already owns, namely the cells it has delivered and the positions where it has observed the teammate, and it then sends first the cells that extend the boundary of that inference. No extra message is needed, and no reply, and no change to the protocol, so the policy fits any system that already exchanges positions. Moreover, whenever a contact breaks, the bits already sent are the ones the estimate judged most useful.

- We build a simulator and a benchmark for the question, with a channel whose bandwidth and connectivity are explicit rather than ideal, and with a ladder of policies that runs from silence to unlimited exchange and includes an oracle reading the receiver's true map. Because every policy is charged the same bits, the ladder isolates the ordering itself, and it gives a reference for how much of the channel's value a policy captures.

- We measure what the choice is worth. Across every floor plan, team size, and sensor range tested, MIRROR returns between roughly a quarter and a half of the time a silent team loses and comes within a few percent of unlimited exchange, while each contact carries only a small fraction of what a robot could send.

- We also measure what the wrong choice costs. A policy that sends its own frontier first, the content a sender would judge most valuable, returns none of that time, and on one floor it finishes later than a team that shares nothing at all. The fragments it delivers read to the receiver as new places to go, so they pull it toward ground a teammate has already covered. Removing the budget cap does not rescue the policy, and therefore importance judged at the sender does not transfer to the receiver.

1.2 Thesis Outline

The remainder of this thesis is organized as follows.

Chapter 2 provides the background needed to follow the thesis: how a single robot explores an unknown environment with an occupancy grid and its frontier, how a team of robots explores together, and how communication among robots is modeled.

Chapter 3 reviews the literature most relevant to this work, from coordination strategies for teams to map prediction and communication-constrained exploration. The chapter positions our contribution in relationship to these past lines of research.

Chapter 4 formalizes the decentralized exploration problem under a limited communication channel, derives where the value of communication concentrates, and presents the MIRROR algorithm.

Chapter 5 describes the experimental design, including the baselines, the oracle, the connectivity model, and the metrics, and reports the results of the simulation, from the comparison at a fixed budget to the sweep of the budget itself.

Chapter 6 summarizes the main contributions of the thesis, discusses its limitations, and outlines directions for future research, extending from transmitting observed cells to transmitting predicted maps.

2 Background

This chapter provides the background needed to follow the rest of the thesis. It describes how a single robot explores an unknown environment using an occupancy grid and its frontier (Section 2.1), how a team turns that machinery into a coordinated effort by exchanging maps (Section 2.2), and how communication between robots is modeled, both in the literature and on real platforms (Section 2.3). Together, these sections define every concept on which the following chapters will rely. The literature surrounding each topic is reviewed separately in Chapter 3.

Notation. Cells of the grid are denoted by c, sets of cells by capital letters such as K, F, and V, and the full lattice by $\mathcal{C}$. The subscript t denotes discrete time; when several robots are involved, the robot index comes first, so $m_{i,t}$ is robot i's map at time t. $N(c)$ denotes the four grid neighbors of cell c, $\|\cdot\|$ the Euclidean norm, and $|\cdot|$ the number of elements of a set. The bit is the unit of information.

2.1 Frontier-Based Exploration

Consider a single robot placed in an environment it has never before seen. It possesses no floor plan, no prior knowledge, only its own sensor. Its goal is to explore the surrounding environment. The sensor reveals the robot's limited neighborhood: it has finite range, and it cannot see through walls, so at any moment most of the environment is invisible. Exploration is the process of turning this stream of partial views into a complete map, by repeatedly deciding where to move next on the basis of the incomplete map built so far. The decision is the really challenging part. The robot must choose where to go using a map that is incomplete, and each choice it makes determines how that same map will grow. A poor choice reveals little new ground, and a map that grows slowly leaves less information to guide the next choice.

Figure 2.1 represents the problem before any abstraction. The view from the onboard camera of one of the robots, recorded during an exploration run in the ACL laboratory at MIT, captures the decision the robot faces at every step. The robot is advancing along a corridor that forks, either straight ahead or around the partition. It must commit to one of the branches. The adjacent map shows everything

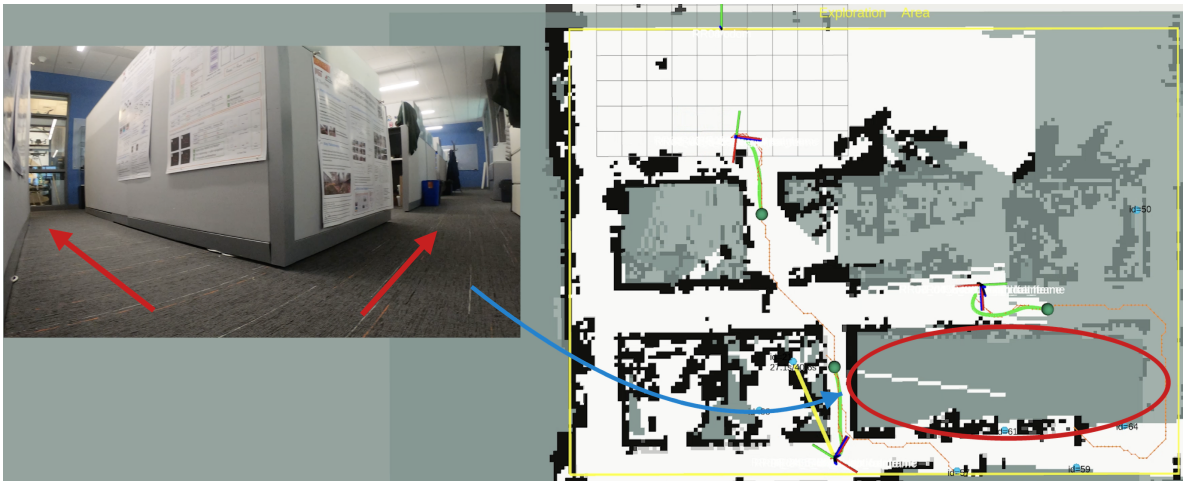

Figure 2.1: *A real exploration run.* A team of ground robots explores an office environment; the occupancy grid is built online, with mapped free space in white, sensed obstacles in black, unobserved space in gray, robot trajectories in green, planned paths in orange, and the assigned exploration area in yellow. The left panel shows one robot's onboard camera view at a given instant. The corridor ahead forks, and the two red arrows mark the branches the robot must choose between, using only the partial map on the right. The blue connector links the right-hand branch to the region it leads into, circled on the map, which the team has not yet observed. Real grids carry what this chapter's abstraction removes: measurement noise, ragged boundaries, localization artifacts. The experiments associated with this thesis were conducted in simulation.

the robot has at its disposal on which to base its decision: white cells already mapped, black cells sensed as obstacles, and a gray field not yet observed; the yellow rectangle bounds the assigned mission area. No information beyond the mapped region is available. Where to go next, given only this partial picture, is the entire problem of exploration. This choice is consequential: the two branches are not equivalent. One may open into unexplored rooms while the other closes after a few meters. Each commitment shapes what the robot will be able to see next. Different sequences of choices reveal different amounts of the environment for the same travel, so the quality of these decisions, compounded over a mission, decides the quality of the exploration. The rest of this chapter builds the machinery behind these decisions.

The standard representation is the *occupancy grid*, introduced by Moravec and Elfes [15] and formalized by Elfes [16]. The environment is discretized into a lattice of square cells $\mathcal{C} \subset \mathbb{Z}^2$ at a resolution chosen so that a cell is either traversable or not. In its original formulation each cell carries a probability of being occupied, updated recursively from noisy range readings; exploration studies commonly adopt the thresholded, deterministic abstraction, and this thesis does the same. The robot's map at time t is then a function

$$m_t : \mathcal{C} \to \{\text{free}, \text{ occ}, \text{ unk}\},$$

where free cells can be stood upon, occupied cells are walls and obstacles, and unknown cells have never been observed. At the start of a mission, $m_0(c) = \text{unk}$ everywhere. The *known set* collects the cells whose state the robot has determined, $K_t = \{\, c \in \mathcal{C} : m_t(c) \neq \text{unk} \,\}$. The map is also a measurable object: with three states, a cell costs two bits, and a grid of realistic extent weighs megabits. Every map exchange in this thesis must move data of this size.

What the sensor contributes at pose x_t is its *visibility region*,

$$V(x_t) = \{\, c \in \mathcal{C} \;:\; \|c - x_t\| \leq R_s \;\wedge\; \text{LoS}(x_t, c) \,\},$$

where R_s is the sensor range and the line-of-sight condition $\text{LoS}(x_t, c)$ holds when the straight segment from x_t to c crosses no occupied cell before c itself: a ray is stopped by the first obstacle it meets, which is itself observed, while everything beyond it stays hidden. Sensing is treated as exact, the standard idealization in exploration studies. It sets every cell of $V(x_t)$ to its true state and touches nothing else, so knowledge only accumulates: $K_t = K_{t-1} \cup V(x_t)$. In reality, range measurements are noisy and cells can be misclassified. Assuming exact sensing removes this error source and lets this thesis isolate the communication problem. The same idealization applies to localization: the robot is assumed to know its own pose, so the map is built in a fixed frame. When several robots are involved, that frame is shared by the whole team, so maps built by different robots can be laid on top of one another directly, and any pose a robot communicates is meaningful to whoever receives it. In practice both the pose and the common frame are the province of simultaneous localization and mapping, a problem with mature solutions surveyed by Cadena et al. [17]. Exploration studies commonly treat it as solved, so that the exploration question can be studied on its own, and this thesis follows the same practice. Figure 2.2 shows the two sides of this construction: the physical scene with the region the sensor has swept, and the grid the robot holds in memory.

On this grid, one structure matters above all others: the *frontier*, introduced by Yamauchi [3],

$$F_t = \{\, c \in \mathcal{C} \;:\; m_t(c) = \text{free} \;\wedge\; \exists\, c' \in N(c) : m_t(c') = \text{unk} \,\},$$

the known free cells with at least one unknown neighbor. The frontier is exactly where the map can still grow. Any cell the robot will ever learn must first be visible from free space, and free space borders the unobserved region only along F_t. The interior of the known region has nothing left to reveal, and the unknown interior cannot be observed without first crossing its rim. When the robot senses from a frontier, the fog retreats and the frontier moves outward; where a wall closes the view, the frontier dissolves for good. In practice the frontier cells form contiguous segments, one per opening of the known region, and the robot targets a representative point of a segment rather than an isolated cell.

The exploration loop follows directly: sense and update the grid; extract F_t; commit to a frontier location

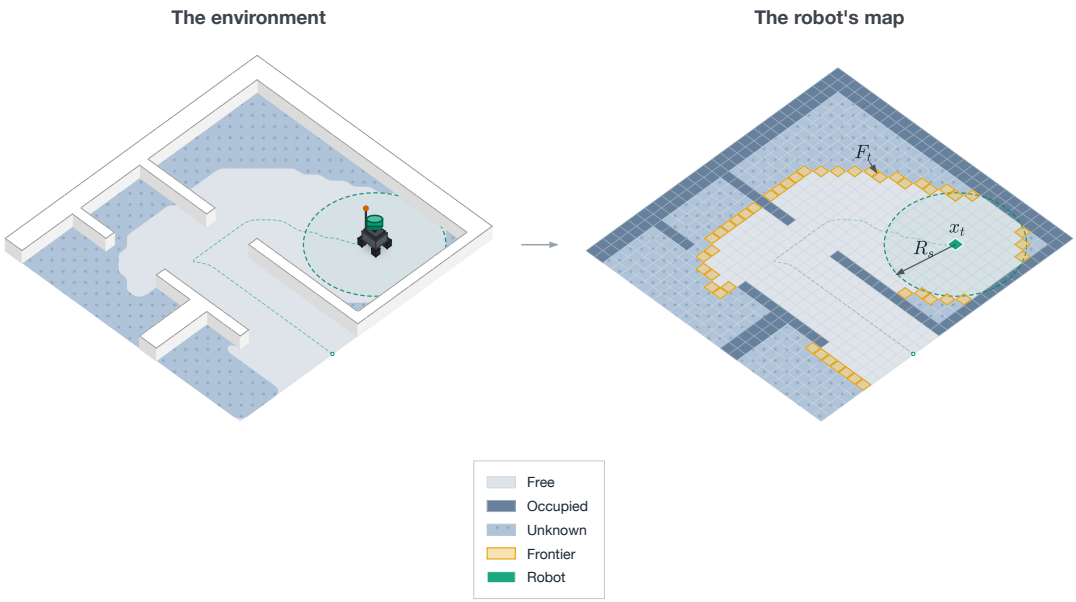

Figure 2.2: *From the environment to the robot's map.* Left: a robot explores an unknown environment; its sensor has finite range and is blocked by walls, so only the swept region is known. Right: the same scene on the occupancy grid, where the floor the sensor has swept is marked free, the floor that has never been observed is unknown, and the amber cells are the frontier F_t, the known free cells bordering unknown space, the only place where the map can still grow. Walls are drawn in full in both panels to keep the geometry readable; what the robot has actually observed is the swept region alone. The environment is deliberately small, for didactic clarity.

as the goal; plan a path to it over known free cells; move, and repeat. Time is discrete throughout the thesis: one time step corresponds to one iteration of this loop. Motion is abstracted accordingly: the robot advances along paths of known free cells, and the kinematics and control that realize the path are outside the scope of this thesis. For the reader it is enough to know that motion is reliable: the robot follows its planned path at bounded speed, and reaching the goal is a matter of time, not chance. Everything above that level, namely where to aim and which route to take, is the layer this thesis works in. The loop terminates when $F_t = \varnothing$, and the termination is meaningful: no frontier left means every reachable, observable cell is known.

The only discretionary step is the choice of goal. The canonical rule, again due to Yamauchi [3], is the nearest frontier,

$$g_t = \arg\min_{f \in F_t} d(x_t, f),$$

where d is the length of the shortest path over known free cells, in practice computed with a grid planner such as the A* algorithm of Hart et al. [18]. The nearest rule spends the least travel to reach new information. Richer selectors weigh travel cost against how much unknown space a goal promises

to reveal. We postpone them to Section 3.1, because selecting frontiers only becomes difficult when several robots select at once. Indeed, for a single robot, any sensible rule yields the same qualitative behavior: the robot always drives toward the border of what it knows, the border moves outward as it arrives, and the map keeps growing until nothing reachable is left.

2.2 Multi-Robot Exploration

The previous section considered a single robot. Moving to a team of n robots leaves the loop of the individual robot almost unchanged: each robot i runs exactly the loop of Section 2.1, with its own pose $x_{i,t}$, its own occupancy grid $m_{i,t}$, and its own known set $K_{i,t}$; the frontier machinery needs no modification to operate alongside teammates, an observation already present in the multi-robot extension of Yamauchi [6]. Most exploration methods, for one robot or for a team, still start from the frontier today. For this reason, Yamauchi's work appears repeatedly in this chapter. What the team changes is how quickly the environment can be covered. In principle, n robots split the environment and finish up to n times faster than one. In practice the speedup is limited by several factors: travel geometry, congestion, and the shape of the environment. Above all, it is limited by the factor the coordination literature was built to remove [4]: redundancy, with two robots spending travel to determine the same cells, each unaware that the other has already explored them. Everything in this section describes the problem class rather than any particular system: the concepts apply unchanged whether the robots are ground vehicles in a building, legged platforms in a mine, or agents in a simulation, and the specific system used in our experiments enters only in Chapter 5. Algorithm 2.1 summarizes the resulting per-robot loop. It is identical to the loop of Section 2.1 with one added step, the exchange with teammates. The loop is built around the frontier. The frontier defines the candidate goals, and the goal-selection rule σ decides among them. Different algorithms can be plugged in as σ, from the nearest-frontier rule to the coordinated rules reviewed in Section 3.1, while the rest of the loop stays unchanged. It is worth stating this loop explicitly, because everything discussed from here on (coordination, communication, and the contribution of this thesis itself) modifies one step of this loop and leaves the others untouched. All robots run the loop concurrently: within a time step each robot senses, exchanges, and moves once, and none has priority over another. How a robot should weigh its teammates' positions and current goals when selecting its own is deliberately left open here; it is the subject of the strategies reviewed in Section 3.1.

Figure 2.3 shows a team of three robots partway through a mission. Each robot has swept a territory of its own, the dotted fog is ground no robot has visited yet, and each has committed to a goal on its own frontier: the team is doing exactly what the promise requires, dividing the unknown. The picture deserves a second look, however, because it is drawn from a viewpoint that does not exist in the system. It is an external view, the union of everything the team knows. No robot holds it. It is nonetheless useful to give that view a name: the *team known set* $E_t = \bigcup_{i=1}^{n} K_{i,t}$, the union drawn in Figure 2.3.

Algorithm 2.1 Decentralized frontier-based exploration: robot i, one time step

write the observed state of every cell of $V(x_{i,t})$ into $m_{i,t}$ ▷ exact sensing (Section 2.1): the observation equals the true state; $K_{i,t} = K_{i,t-1} \cup V(x_{i,t})$
for all teammates j currently able to communicate with i **do**
    exchange map content and fuse: for each received cell c with $m_{i,t}(c) = \text{unk}$, set $m_{i,t}(c) \leftarrow m_{j,t}(c)$
end for
recompute the frontier $F_{i,t}$ on the fused map
if $F_{i,t} = \varnothing$ **then**
    stop ▷ every reachable, observable cell is known
end if
$g_{i,t} \leftarrow \sigma(F_{i,t})$ ▷ goal selection
advance one step along the planned path to $g_{i,t}$

Seen from outside, the mission is complete when E_t covers everything reachable; seen from inside, each robot can only chase the frontier of its own $K_{i,t}$. The figure also restates the lesson of Figure 2.1 at team scale: three simultaneous commitments, one per robot. The quality of the mission hangs on how well those choices divide the unknown.

What each robot actually possesses is its own grid: the floor its own sensor has swept, its own frontier, and fog everywhere else, including over territories a teammate has fully mapped. Two robots' grids agree only where their sweeps happened to overlap. This is the defining condition of decentralized exploration: knowledge is distributed by construction, there is no shared blackboard, and the union of the team's maps exists nowhere. The only way to close the gap between these private pictures is to communicate.

When two robots are able to communicate, they exchange map content and combine, or fuse, what they receive. *Fusion* is a union of knowledge: the receiver adopts the sender's value for every cell it still regards as unknown,

$$m_{j,t}(c) \leftarrow m_{i,t}(c) \qquad \text{for } c \in K_{i,t} \text{ with } m_{j,t}(c) = \text{unk},$$

and no conflict can arise, because under exact sensing two robots that both know a cell agree on its value. This union is the standard rule under the deterministic abstraction adopted here. In fact, systems with probabilistic grids fuse by combining the per-cell evidence instead, and the union is the counterpart of that operation. What a received map does to the receiver is equally precise. A received free cell at the edge of the receiver's known region pushes its frontier outward; a received occupied cell deletes the frontier there for good; in both cases the receiver's map grows without the receiver moving. Put plainly, a robot has two ways of learning the contents of a cell: it can drive until its sensor sees it, or it can receive it from a teammate that already performed that travel. Both produce the same knowledge;

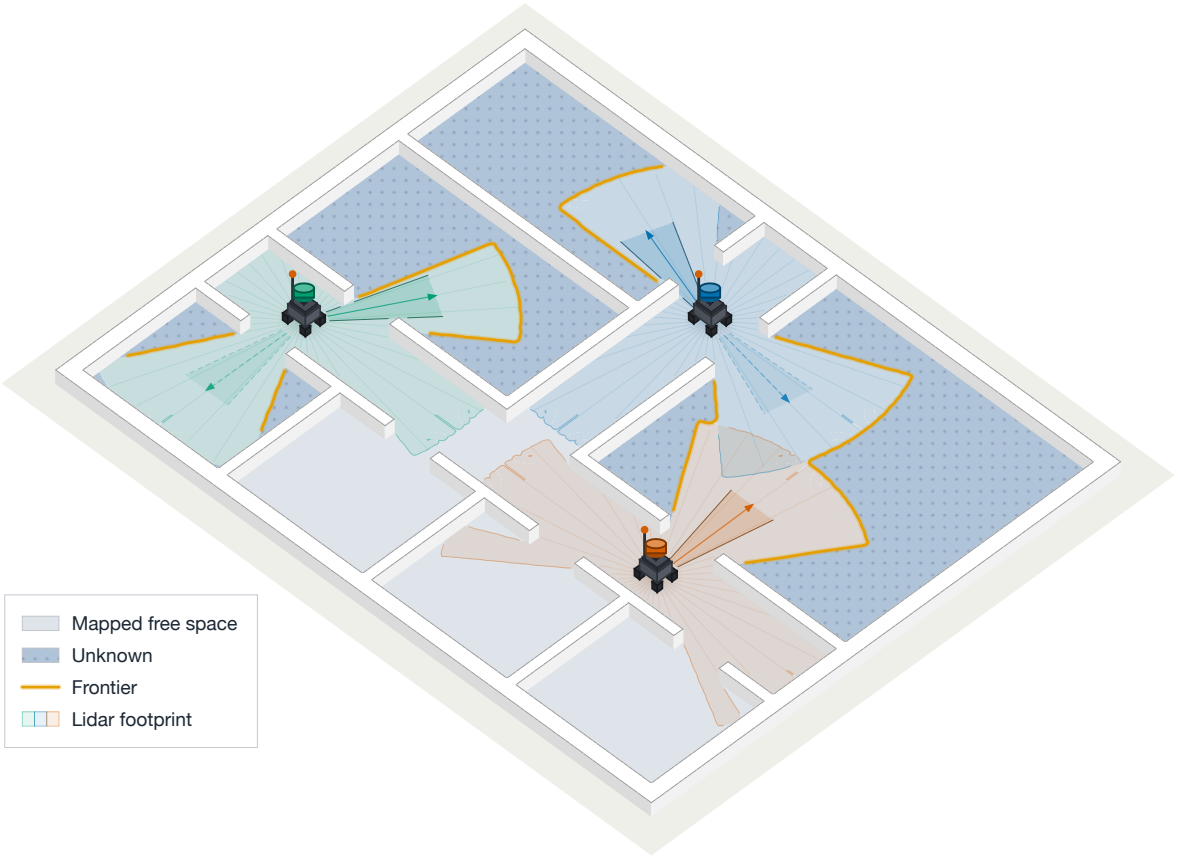

Figure 2.3: *Multi-robot exploration.* Three robots explore the same environment: the light floor is the space the team has mapped, the dotted region is still unknown, the orange arcs are the frontiers, and the tinted wedges are each robot's lidar footprint, with arrows marking the frontier it is heading for. The view is an external one, the sum of everything the team knows, but which no single robot possesses.

the second costs bits instead of meters. This is fundamental to a robot team: communication replaces travel. The moment a robot learns that a teammate has covered some ground, no frontier remains there for it, so its goal selection stops proposing targets in that direction, and the robot turns to what is unknown to everyone. Fusion is also robust to how and when the exchanges happen. Because it is a union of knowledge, the order of exchanges does not matter; receiving the same content twice changes nothing, and any sequence of pairwise exchanges within a connected group ends with the same maps as a single group-wide merge; in algebraic terms, the union is commutative, associative, and idempotent. Correctness therefore requires no synchronization among robots; meeting is enough. Information can also cross the team indirectly: what a robot receives at one meeting, it can pass on at the next, so a cell can reach robots that never met the robot that discovered it.

Fusion alone coordinates exploration. Once maps are exchanged, each robot's frontier excludes, by

definition, everything the team has covered. Thus, robots do not waste resources re-exploring each other's territory and can then explore their own paths without any negotiation. The result is an implicit division of labor. On top of this shared picture, explicit strategies decide who goes where, for instance by discounting goals a teammate is already approaching, as Burgard et al. [4] propose, or by assigning frontiers through ranks and auctions; we review this literature in Section 3.1. Map content is not the only thing a contact can carry: robots also exchange lightweight metadata, their current pose and goal, whose few bytes are negligible next to the map payload. The explicit strategies above rely on exactly this information.

The setting throughout this thesis is decentralized: there is no base station and no robot with a privileged role. In practice this means that every robot runs the same loop and makes its own decisions: nothing passes through a leader, no single failure stops the mission, and the team keeps exploring even when it splits into groups that cannot reach each other. Which pairs can communicate is a property of geometry and transmission radios, and it changes continuously as the team moves; robots come into contact, part, and meet again. Isolation is not a failure but the normal situation. When robots are isolated from each other, their private grids drift apart: each robot keeps discovering ground the others have never seen. Everything in this section has assumed that when robots do meet, the exchange simply happens: complete, instantaneous, free. Whether a real channel supports that assumption is the subject of the next section. What happens when that assumption fails is the subject of this thesis.

2.3 Communication Models for Robot Teams

Section 2.2 let robots exchange maps whenever they were able to communicate, and left undetermined which information was most important for them to exchange. A communication model for a robot team must answer several questions. Among those, two matter most for this thesis, and they are independent of each other. The first is *when* robots can communicate: which pairs are connected, and at which moments. The second is *how much* they can exchange while connected: what volume of data a connection actually carries. The two dimensions have different physical causes and need different models. The first will be captured by a link model, the second by a budget.

For the first question the literature offers a small number of standard families, catalogued in the survey of Amigoni et al. [9]. The simplest is full connectivity: every robot can reach every other at all times, which reduces the team to a single distributed sensor. The workhorse is the range model, in which two robots are connected exactly when they are close enough,

$$\mathrm{Link}_t(i,j) \iff \|x_{i,t} - x_{j,t}\| \le R_c,$$

with R_c the communication range; a stricter variant additionally requires line of sight, so that the

same walls that block the sensor also block the radio. At the far end sit event-based models, in which communication happens only at deliberate meetings: robots plan rendezvous, travel to them, and exchange there. We call a period during which a pair stays connected a *contact*: robots meet, hold a link for some interval, and part. Under any of these models the mission alternates, for each pair, between contact and isolation, and the isolation intervals are where the private maps of the previous section diverge.

Whether two real radios can talk at a given moment depends on the distance between them, on the number and material of the walls in between, on reflections and interference, and on the antennas involved. The same pair that holds a link across an open hall may lose it behind a single concrete corner. No simple rule predicts the moment of contact, which is why real systems treat communication as opportunistic: whenever a link happens to be available, robots exchange whatever the protocol prescribes, map content and the lightweight metadata of Section 2.2, without assuming they will have the chance again. Noise adds a final subtlety: corrupted packets are detected and discarded rather than believed, so an unreliable link does not poison the maps; it simply delivers less. The field experiments of the Kimera-Multi team offer a concrete example: across campus-scale multi-robot deployments, connectivity between robots was intermittent and hard to anticipate. The system was engineered so that mapping continued no matter when the exchanges occurred [19]. The range and line-of-sight models above are therefore best read as tractable stand-ins where true behavior resists modeling, and the budget introduced below plays the same role in the second question.

The second question has received far less attention because its answer is contained in the standard assumption: while a link holds, the exchange is complete, instantaneous, and free. Almost the entire coordination literature, including the fusion mechanism of Section 2.2, is built upon this idealization. Real channels violate this assumption, and it is worth being concrete about how the exchange happens on actual platforms. Robot teams communicate over wireless links whose capacity spans orders of magnitude across technologies and environments: short-range radios offer high throughput but degrade rapidly with distance and obstructions, while long-range, low-power radios hold a link much farther at rates of kilobits per second and below. The environment widens this range further. In underground environments, the signal decays so quickly that teams in the DARPA Subterranean Challenge dropped relay nodes along their paths to stretch connectivity, and links remained intermittent enough that coordination had to be designed around disruption, as Saboia et al. [10] and Chung et al. [1] document. On top of the physical layer sit protocol realities: robots must first discover each other, links drop in mid-transfer, and lost packets must be retransmitted, so the volume a contact usefully delivers sits below the nominal rate of the radio. Deployed multi-robot mapping systems are engineered around exactly this scarcity: they exchange compact intermediate products (submaps, keyframes, or topological summaries) rather than raw grids or sensor streams, as in the Kimera-Multi system of Tian et al. [20], and when even that is too much they sparsify what must travel, as Tian and How [21] do for collaborative estimation.

In recent field deployments on long-range radios, decentralized map exchange proceeded at rates on the order of hundreds of bits per second, as Bayer and Faigl [8] report. The limitation can be written in two lines. Transmitting a full grid costs $|K_t|\, b$ bits, with b the bits per cell, amounting to megabits at realistic sizes (Section 2.1); a contact of duration τ on a link of rate r delivers at most $r\,\tau$ bits. On real links the two sides are separated by orders of magnitude: a ten-second contact at one kilobit per second delivers ten kilobits, half a percent of a two-megabit grid. In practice both dimensions bind. Connectivity decides how often robots may speak, and bandwidth decides how little they can say; everything that follows in this thesis lives inside the gap between $r\,\tau$ and $|K_t|\, b$.

This thesis therefore models both dimensions of the channel explicitly. Connectivity is governed by a link model of the family above, producing contacts; volume is governed by a *budget*: during a contact, each robot may transmit at most B bits per time step to its partner. The budget makes the second dimension a first-class quantity, and it nests the standard settings as limits. As $B \to \infty$ the exchange of Section 2.2 is recovered exactly, complete and free; at $B = 0$ communication carries nothing and the robots explore in permanent isolation. Every intermediate value describes a channel that exists somewhere between a tethered pair and a broken radio, which is why this thesis does not single out one value of B. To generalize better and benchmark fairly, it reports results across budget tiers, from channels that barely carry anything to the unlimited ceiling. The model abstracts the remaining imperfections of a real link: latency and packet loss act by reducing what a contact delivers, and are absorbed into the value of B rather than modeled separately. It is deliberately independent of any particular radio technology, and its formal version, including how bits are priced against map content, is given in Chapter 4.

A finite budget also changes the nature of the exchange. When communication is limited, the complete merge of Section 2.2 stops being an available action. A contact carries only a sliver of what a robot knows, and the rest stays behind. Thus, transmission becomes selection, whether anyone designs the selection or not. How existing systems respond to a thin channel, by meeting more cleverly, compressing the map, or choosing content, is reviewed in Section 3.3. The selection itself is the subject of this thesis. Its treatment starts in Chapter 4.

3 Related Work

This chapter reviews the lines of research closest to this thesis and closes by positioning our contribution with respect to them. Section 3.1 surveys coordination and goal-assignment strategies for exploring teams, one of which our experiments adopt as the fixed selector. Section 3.2 reviews learned map prediction for exploration, an approach this thesis does not use directly but on which its future directions build. Section 3.3 reviews exploration under communication constraints, organized around the three questions a designer must answer: when and where robots should communicate, in what form the map should travel, and which part of it deserves the channel first, the least studied of the three and the one this thesis addresses. Section 3.4 states the resulting gap precisely. The review is tailored to the needs of this thesis rather than comprehensive; where a topic is larger than our use of it, we point to the relevant surveys.

3.1 Coordination Strategies for Multi-Robot Exploration

Chapter 2 described how a team explores an unknown environment: each robot senses, fuses what its teammates send, and commits to a goal on the frontier of its map, around the loop of Algorithm 2.1. Every step of that loop is mechanical except one, the choice of the goal, and this section reviews how the literature has designed that choice, first for a single robot, then for a team. For a single robot the canonical answer is the nearest-frontier rule, and it invites refinement, because the closest frontier is not necessarily the most informative one. González-Baños and Latombe [22] formulated the choice as a next-best-view problem, scoring a candidate position q by the utility

$$g(q) = A(q)\, e^{-\lambda L(q)},$$

where $A(q)$ is the unexplored area the sensor could reveal from q, $L(q)$ the length of the path to reach it, and $\lambda > 0$ a weighting constant. In other words, the robot favors goals that promise much new area for little travel, and distance devalues a goal exponentially. Stachniss et al. [5] carried the same reasoning into the probabilistic setting, selecting the action that most reduces the joint uncertainty of map and pose. Both compute the same kind of quantity: a utility weighing what a goal promises against what it

costs, online, from the current map.

With a team the question changes: when n robots face the same set of frontiers, how should they divide them so that no two robots pay for the same discovery? Figure 2.3 already showed the scenario: three robots in one picture, kept aligned by fusion, and three simultaneous commitments that interact. The multi-robot extension of Yamauchi [6] is the baseline of the field precisely because it does not arbitrate at all: robots share what they sense and select independently, each heading for its nearest frontier. It is worth pausing on why this design was remarkable. Nothing is negotiated, no roles are assigned, and no protocol can fail. The team still divides the environment. Once the maps are fused, no frontier remains on the ground one robot has covered, so the others stop selecting goals there. Coordination emerges from shared knowledge alone. In other words, sharing the maps is enough to make the robots coordinate. No further policy is needed, and this holds for a team of any size. It follows that the failure mode is also a concrete possibility. This is because fusion says where the team has been, but not where each robot is going: nothing prevents two nearby robots from committing to the same opening and paying the same travel twice. The strategies this section reviews exist to remove that redundancy. They differ in what each robot must know about its teammates: positions, goals, or bids. A broader survey is given by Quattrini Li [23].

Burgard et al. [4] answer this question through explicit assignment with discounts. Goals are assigned iteratively: a robot's utility for a frontier target starts from the unknown area in view from it. Each time a target is assigned, the utility of nearby targets is discounted for the robots still waiting, since a teammate arriving there will strip those targets of most of their novelty. Teams coordinated this way spread by construction. Assignment can also be posed as an explicit optimization over structure, which means deciding region by region rather than frontier by frontier, matching robots to the spaces of the environment. For instance, Wurm et al. [24] partition the partial map into segments corresponding to spaces such as rooms and assign robots to segments with the Hungarian method, a computation performed by a central planner. They report faster exploration in structured indoor environments than frontier-level coordination alone.

A second family coordinates through explicit negotiation. In the market of Zlot et al. [25], robots continuously trade candidate goals. A bid reflects what a goal is expected to reveal and the travel it demands. Each goal ends up with the robot that values it most, and no central assignment is needed. The design tolerates communication interruptions by construction: auctions run peer-to-peer among whichever robots can currently hear each other, so a robot out of range simply continues with the goals it already owns and trades again at the next contact. The machinery descends from the auction-based task allocation that Gerkey and Matarić [26] introduced for general multi-robot systems, and Dias et al. [27] survey the family. However, the price is the negotiation itself, because every allocation requires a round of bids and awards, and each of those is a message that must find its counterpart before any robot can

commit to a goal.

One work in this literature matters most to this thesis. MinPos, proposed by Bautin et al. [28], achieves coordinated dispersion with the least possible machinery: no bids, no negotiation, no central step, nothing beyond the map content and the teammate positions the team already exchanges as data (Section 2.2). This is the selector running under every method and baseline in our experiments. The rule is simple to state: for each frontier, a robot counts how many teammates are closer to that frontier than itself, and it commits to the frontier where this count, its rank, is minimal, breaking ties by choosing the nearest. Path distances are computed over the known map. Each robot evaluates the rule locally, from its own fused map and the teammate positions it last received. The property that matters here, because everything in our experiments stands on it, is the dispersion this rule produces. A robot's rank at a frontier is low exactly where it is the best-placed teammate, so each robot is pushed toward the openings it is best placed to claim: the team fans out with no agreement beyond knowing roughly where the others are. The implications run in both directions. As long as positions flow, redundant travel falls without a single negotiation message; and because the rule asks for so little, it keeps producing sensible decisions from whatever position information the channel has managed to deliver. This degrades as that information goes stale. With no teammate information at all, every rank is zero and the rule reduces to nearest-frontier selection.

More recent works replace the geometric utility with an information-theoretic one. Julian et al. [29] proved, for a single mapping robot, that any controller maximizing the mutual information between the map and future range measurements is eventually attracted to unexplored space. Multi-robot planners optimize such objectives directly: Atanasov et al. [30] pose active information acquisition as decentralized sequential planning with suboptimality guarantees, demonstrated on multi-robot SLAM. Corah and Michael [31] exploit the submodularity of information objectives to run distributed greedy assignment with a constant-factor bound in multi-robot exploration. These planners carry the heaviest machinery of the family. They inherit the same dependency as every method above: the objective is evaluated on the team's shared picture. One difference from our setting deserves a note. These planners treat the robots' own pose uncertainty as part of the objective. In this thesis localization is treated as solved (Chapter 2), so that side of the objective stays outside our scope.

Seen together, these strategies answer one question: where each robot should go. They answer it under one shared assumption: that the information the decision is computed on, whether fused maps, teammate positions, goals, or bids, is available wherever the decision is made. The arbitration differs (discounts, segments, prices, or ranks), but none of these methods charges the channel that keeps the picture aligned. It is worth making concrete what that assumption hides, because it is exactly the problem this thesis is about. Every quantity in this section is computed on data received from teammates: the discounts of Burgard et al. [4] need the teammates' goals, MinPos needs their positions, and every method needs

the fused map underneath. Recall the orders of magnitude from Section 2.3: a realistic grid weighs megabits, and a single contact may carry only a small fraction of it. When that is all a contact carries, the robots keep running the same selectors, but on pictures that no longer agree: the assignment divides a map that does not exist. This is the problem the rest of the thesis works on. The coordination layer itself is therefore not where we intervene: we keep it fixed, identical in every arm, so that only the sharing policy varies. All experiments in Chapter 5 fix the goal-selection rule σ of Algorithm 2.1 to MinPos. MinPos is the natural choice for the role: among coordinated strategies it demands from the channel only the positions the team already exchanges, and it involves no negotiation protocol that intermittent links could break mid-round. In this sense the thesis builds on top of this baseline rather than beside it: the selector stays untouched, and our contribution operates upstream of it, in the choice of which map content reaches it. There is some benchmark evidence that supports the choice: in the comparison of Faigl and Kulich [32] across five assignment strategies, the best performance came from the most computationally demanding one, and MinPos performed on par with the discounting scheme of Burgard et al. [4]. The next two sections turn from where robots go to what their maps can know.

3.2 Map Prediction for Exploration

From the exploration loop of Chapter 2 one can read how the unknown is treated: as opaque. Beyond the frontier, nothing can be said until a sensor arrives. Indoor environments are more forgiving than that, because built space repeats itself, corridors continue, rooms close into rectangles, and a partial map often betrays the shape of what it still hides. A line of research therefore learns to complete the map before the robot goes there. The knowledge these methods add is a prior. The model is trained offline on maps of other buildings, and it learns how indoor space tends to continue. During the mission, the robot asks it to guess the part of the map it has not seen yet. Shrestha et al. [33] attached a generative network to a classical information-theoretic explorer. This means that the network predicts the unknown regions of the partially explored map; the prediction sharpens the choice of where to sense next, and the behavior, the authors note, stays interpretable through the predicted map. In the same year, Katyal et al. [34] made the reliability of the prediction itself the guide, using the variance of the generated hypotheses as the exploration heuristic. The embodied-vision community reached the same conclusion from the perception side: Ramakrishnan et al. [35] showed that an agent that anticipates occupancy beyond its sensed field builds spatial awareness faster and explores better. This makes the same point from a second direction: what lies beyond the sensed boundary is regular enough to be predicted, and predicting it buys exploration time.

The works that followed no longer bolt a predictor onto an existing explorer; they design the whole goal selection around the prediction. MapEx, by Ho et al. [36], maintains an ensemble of inpainting networks trained on real campus floor plans and scores viewpoints by a probabilistic information gain: visibility is estimated by raycasting on the mean predicted map, and the gain sums the ensemble

variance inside the visible region. On real building data this outperforms nearest-frontier selection by twenty-five percent on average. Georgakis et al. [37] plan whole paths on the uncertainty field of the learned prior. PIPE, by Baek et al. [38], an evolution of MapEx by largely the same group, integrates the expected observation mask along entire candidate trajectories rather than at isolated viewpoints, with the prediction taming the overestimation that pathwise integration invites. P^2 Explore, by Song et al. [39], trusts the prediction only at the level of rooms and their connectivity, applying the predicted visiting order as a soft constraint, a deliberate hedge against a prior that is right about structure and wrong in detail.

For clarity, it is worth expressing the difference from the online selectors of Section 3.1 in deployment terms: what one must actually build and carry in order to run each approach. To field a prediction-driven explorer one assembles a corpus of floor plans, trains the prior offline, and carries the inference onboard, paying for it at every replanning cycle. The costs of these steps are not trivial. For instance, 4CNet, the multi-robot system discussed below, reports five and a half seconds per prediction end to end on a desktop RTX 4090 GPU, and its field robots ran the network on a dedicated onboard GPU; P^2 Explore states plainly that its full system was not deployed on the real robot "primarily due to insufficient hardware computing power". A frontier selector consults geometry instead: no corpus, no training phase, no dedicated hardware, and no notion of being out of distribution, because the current map is not a guess. The prior, by contrast, inherits the boundaries of its corpus: MapEx trains on 149 campus blueprints from the KTH dataset, P^2 Explore on 140 after deduplication, and 4CNet entirely on procedurally generated terrain. Prediction, it is worth noting, does not even require learning: Luperto et al. [40] complete partially observed rooms from the geometric regularities of indoor layouts and speed up exploration with no trained prior at all.

Two structural facts about these works matter here. First, they are almost entirely single-robot: MapEx, PIPE, and P^2 Explore each plan for one robot, and MapEx lists the extension to teams as future work. The exceptions are recent and few: 4CNet, by Tan et al. [41], brings diffusion-based map prediction to decentralized multi-robot exploration of uneven terrain, and Spinos et al. [42] rank tasks for a small team by the generative entropy of a diffusion inpainter. Second, and decisive for this thesis, prediction informs where to go, never what to transmit. 4CNet makes the point, precisely because it does reason about the channel: robots exchange only their trajectories, a handful of coordinates standing in for whole maps, and each robot's learned prior fills in the rest. The predicted map itself is also never transmitted, and the prediction alters neither the content nor the timing of communication. It answers a thin channel by not sending maps at all. The nearest these works come to a communication decision is PRoID, by Kim et al. [43], where predicted information gain decides when each robot should stop exploring and relay to a base station, accounting for what teammates are already relaying: prediction times the report, but does not choose the content.

This thesis differs from these works in two specific respects and connects to them in a third. The first difference is that nothing in our method is learned: the sender's estimate of the receiver's map is deterministic bookkeeping, replayed from the history the two robots share, trained on nothing and requiring no inference. In practical terms the contrast is clear: our method needs no training corpus and no offline phase. Everything it computes happens online, during the mission, and it amounts to the same cheap map bookkeeping the selectors of Section 3.1 already perform. The second difference is the question itself: prediction enriches what a robot can guess about the world it has not seen, while we decide which part of what a robot has seen deserves the channel first. The third point is a connection rather than a difference: a learned prior of exactly the kind reviewed here could sharpen the sender's estimate of what the receiver is about to discover on its own, and the future-work discussion of Chapter 6 builds on these methods. There is also a practical debt: the campus floor plans on which our experiments run are those distributed with MapEx. A word on the maps themselves is due here: our evaluation deliberately spans different environments, from the KTH campus blueprints to a laboratory floorplan, because no single building is representative and every method's behavior depends on the structure it explores. The environments and the reasons for their variety are detailed in Chapter 5.

3.3 Communication-Constrained Exploration

Chapter 2 ended on the observation that under a finite budget, transmission becomes selection whether anyone designs it or not. This section reviews how exploring systems actually cope with a constrained channel, organized around the three questions a designer must answer: when and where robots communicate, in what form the map travels, and which content deserves the channel first when the budget does not fit everything. The field itself acknowledges the asymmetry among the three. Ma et al. [44], writing recently, name exactly two established strategies against limited bandwidth: communicate less often and make messages smaller; the question of what to send is not in their taxonomy. The two works closest to that question, thirteen years apart, open with the same words: what to communicate is a problem that has "received little attention", wrote Roth et al. [45] in 2006, and Marcotte et al. [7] again in 2019, citing them. The first two questions matured in the interval; the third did not.

The first question is when and where robots should communicate, and it is the most developed of the three, because connectivity can be planned like any other constraint on motion. Banfi et al. [46] plan exploration so that the team periodically reconnects to a base station and new observations always find their way back; Matignon et al. [47] plan policies that keep working when communication breaks, deployed on real robots; and the ACHORD system of Saboia et al. [10] was engineered for the DARPA Subterranean Challenge missions of Section 2.3, where the ground itself cuts the links. Best et al. [48] decide when a link is even worth requesting, and Bramblett et al. [49] model teammates' beliefs to keep coordinating through silence and to plan the next reunion. What we take from these works is one fact.

All of them decide when a link will exist or how to survive without one, and none of them touches what should cross the link once it is up.

The second question is what form the map should take when it travels. In most of the literature, the answer is a smaller one. One family stops sending grids and sends structure instead: the MR-TopoMap of Zhang et al. [14] exchanges a topological skeleton, rooms and their connections, precisely because the full occupancy grid is "a large amount of data"; the field-tested system of Bayer and Faigl [8] goes further still and ends by sharing only the robots' positions. Collaborative SLAM systems send compact intermediate products in place of raw maps [12, 20, 21]. Shi et al. [13] keep the grid but compress it, and one detail of their method matters here: before compressing, they sort the map's voxels along a space-filling curve, because sorted content compresses better. The transmission order of map content does appear in this literature, then, but it is chosen for the compressor, never for the receiver. At the extreme sits the WiSER-X system of Jadhav et al. [50]: no explicit information exchange at all, only relative positions inferred from radio pings, and still fifty-eight percent less overlap than a team sharing nothing. The lesson for this thesis is sharp: coordination signals are nearly free, and it is the map content that must fight for the channel.

For a concrete example of where all of this leaves a real system, consider MARBLE, the Subterranean Challenge finalist of Biggie et al. [11]. MARBLE does prioritize its traffic, but only between kinds of messages: telemetry before video, commands before maps; within the map data itself, pieces cross the link in the order they were produced. Even in a system engineered end to end for constrained communication, which part of the map goes first is decided by chronology, not by anyone's design. That blind spot is exactly where this thesis works.

The third question is ours: when the budget does not fit everything, which content should go first? It has been posed before, but each time in a problem different from ours. Roth et al. [45] posed it first, selecting, under a budget of a few observations per step, the observations that most improve the team's expected reward; the setting consists of benchmark problems a handful of states in size, and the authors themselves note that such approaches "have so far been validated on very small test problems". Marcotte et al. [7] brought the question to robot teams: each robot broadcasts the few cell observations its bandwidth slot allows, chosen by simulating how much each would improve the team's plans; their largest experiments, ten agents on a hundred states, remain far below the scale of a single occupancy grid of Chapter 2. Kepler and Stilwell [51] select at encounters, as we do: meeting agents broadcast only their most informative recent measurements, though for estimating a continuous field rather than for exploring. Tian et al. [52] give the budget its cleanest form, a hard cap on transmitted data with near-optimality guarantees, but the data are SLAM loop closures and the selection is negotiated between the robots at each exchange. Ma et al. [44] let a network learn what to pack into 256-byte messages, cutting communication by 99.2 percent. The price is that nobody can say what the messages contain.

Contemporaneously with this thesis, Niu et al. [53] do order occupancy content by expected map improvement, but centrally: robots stream to a server, and the server tells them what it wants next. The idea our own method rests on, that a sender keeps a copy of what the receiver knows and sends only what that copy lacks, is itself not new: in event-based estimation, Trimpe and D'Andrea [54] had every sensor run a replica of the central filter and transmit only when the replica's uncertainty grew too large. What has never been done is to bring that idea to the map exchange of exploring teams: to order the cells of a peer-to-peer map transfer by their value to the specific robot receiving them, with a receiver model that costs the channel nothing. No work above does this, and stating the gap precisely is the business of the next section.

3.4 Thesis Positioning

This chapter has reviewed three lines of research, each built around one question. Section 3.1 asked where each robot should go. Section 3.2 asked where to go when the unseen part of the map can be predicted. Section 3.3 asked how exploration copes with a finite channel, and split the problem into three further questions: when robots should communicate, in what form the map should travel, and which content deserves the channel first. Before stating where this thesis stands, it is useful to see the whole picture at once. Table 3.1 collects the works reviewed in this chapter, organized by the question each answers.

The Channel column of Table 3.1 shows the assumption each work makes about the channel. When the exchange is assumed free, robots merge their entire maps at every contact. There is nothing to decide, because everything a robot knows reaches its teammates anyway. When the channel has a finite budget, the full merge stops being possible: a contact carries only part of what a robot knows, so some cells are sent and the rest are not, whether anyone designs this selection or not (Section 2.3). Planning when robots meet and shrinking what must travel both help, but under a finite budget the selection remains. The works in the last block of the table are the ones that face it directly.

Why does no row of that block land on our problem? A likely reason is the circularity of Chapter 1: content should be aimed where the receiver's knowledge ends, and that is exactly what a thin channel hides from the sender. Each of these works gets around the problem in a different way, and each way has a drawback that limits exploration. Some works coordinate the selection through the channel itself [52, 53]. Some keep sampled guesses of their teammates [7], or exact ones in worlds a few states wide [45]. Others gate by novelty or let a network choose [44, 51].

For two exploring robots, however, the aim can come for free. The sender knows what it has already sent, and it knows where the two robots have met, because positions travel anyway (Section 2.2). Together, the two records bound what the receiver must know by now, at zero cost on the channel. To our knowledge,

Table 3.1: The literature reviewed here is organized by the question each answers. The Channel column gives each work's assumption about the communication channel: assumed free (complete map exchange at every contact), not modeled (communication is not part of the problem), intermittent (links come and go, but carry everything while up), limited (the volume a link carries is explicitly small), budgeted (a hard cap on the transmitted data), event-triggered (transmission only when a condition fires).

Work	Contribution	Channel	What travels	Receiver model
Where should each robot go?				
Yamauchi [6]	share what you sense, head for the nearest frontier	assumed free	full maps	–
Burgard et al. [4]	frontiers assigned one at a time, discounting targets near assigned ones	assumed free	maps, goals	–
Zlot et al. [25]	goals auctioned to the highest bidder	assumed free	maps, bids	–
MinPos [28]	each robot goes where the fewest teammates are closer than itself	assumed free	maps, positions	–
Corah and Michael [31]	distributed greedy assignment of information utilities, with guarantees	assumed free	maps	–
Where to go, when the unseen can be predicted?				
Shrestha et al. [33]	a generative network completes the partial map and guides where to sense	not modeled	–	–
MapEx [36]	viewpoints scored by information gain on an ensemble of predicted maps	not modeled	–	–
4CNet [41]	terrain predicted from each robot's own map and teammates' trajectories	limited by design	trajectories	–
When and where should robots communicate?				
Matignon et al. [47]	plans that survive communication breaks	intermittent	full maps	–
Banfi et al. [46]	motion planned to reconnect to the base	intermittent	observations	–
Bramblett et al. [49]	teammate beliefs bridge the silence	intermittent	nothing	beliefs (motion)
In what form should the map travel?				
MR-TopoMap [14]	a skeleton graph replaces the grid	limited	graph skeleton	–
Shi et al. [13]	voxels sorted for better compression	limited	compressed grids	–
Bayer and Faigl [8]	field-tested exploration sharing only positions	limited	positions only	–
WiSER-X [50]	positions inferred from radio pings	limited	signal pings	–
PRoID [43]	prediction times the relay to the base: the report, not the content	intermittent	maps, to base	what peers relay
Which content deserves the channel first?				
Roth et al. [45]	send what most improves team reward, in worlds a few states wide	budgeted	few observations	common knowledge
Marcotte et al. [7]	pack each slot by simulated benefit; ten agents on a hundred states	budgeted	few observations	sampled models
Kepler and Stilwell [51]	novelty-gated subsets at encounters	limited	measurements	none stated
Tian et al. [52]	loop closures picked near-optimally	budgeted	loop closures	negotiated
Ma et al. [44]	a network packs beliefs in 256 bytes	budgeted	learned messages	learned, implicit
Niu et al. [53]	occupancy streamed to a central server in layers	limited	occupancy layers	feedback signals
Trimpe and D'Andrea [54]	a filter replica decides when to transmit	event-triggered	one measurement	replica, zero cost

(the principle this thesis imports, from state estimation into the map exchange)

no prior work uses this to order the content of a peer-to-peer map exchange in decentralized exploration by its value to the specific receiving robot, under a budget that caps each contact step. Whether the free estimate is sharp enough to aim with is a question for the experiments.

In light of this picture, the contributions of this thesis connect to the questions above as follows.

- **The problem.** We take the least studied of the three questions as our object. Under a hard bit budget per contact step, the protocol keeps exactly one free choice: the order in which the unsent map cells travel. Chapter 4 formalizes this choice.

- **The policy.** We answer it with MIRROR. The sender estimates what the receiver knows without asking it, inferring the receiver's map from two records it already owns, the cells it has delivered and the positions where it has observed the teammate, and it sends first the cells that extend that inference. The estimate therefore costs the channel nothing, and the order stays valid however early the contact ends (Chapter 4).

- **The benchmark.** We measure the policy against a ladder of arms spanning the Channel column of Table 3.1, from no communication to the unlimited exchange assumed by the coordination literature, and we do so along several axes rather than at a single operating point. The environments are seven real building floor plans across three size tiers, six of which carry the campaigns, and one of those six was held out of all development until it was released for a single blind run. The team size runs from two to seven robots. The sensing range is swept from 5 to 40 m, so that the value of communication can be read against the quantity it competes with. The budget itself is swept as well, over more than two orders of magnitude, and the ladder includes an oracle that reads the receiver's true map, so the policy is measured not only against naive sharing but against the best any ordering could do with the same bits. Every arm pays the same price per transmitted cell and is cut by the same rule at each contact, and the arms differ only in the payload they draw from and the order they impose on it (Chapter 5).

4 Methodology

Chapters 2 and 3 established the machinery of frontier exploration and showed that no existing system designs what crosses a limited channel. This chapter builds that design. Sections 4.1 and 4.2 formalize the exploration model and the budgeted channel, and reduce the protocol to a single free choice, the order in which a robot sends the cells it has not yet delivered. Section 4.3 derives where a delivered cell can act at all, and what an order blind to that answer costs. Section 4.4 answers with MIRROR, which computes that order from two records the sender already keeps and pays nothing on the channel. Section 4.5 assembles the pieces into one program.

4.1 Problem Formulation

Chapter 2 introduced the concepts of exploration in words and pictures. This section defines the same objects formally, in the notation that the analysis of Section 4.3 will use. Communication appears here only as a set of received cells, without a model of when a link exists or of how much one exchange can carry. Those two elements are the subject of Section 4.2. Throughout the chapter, the text states for every construction whether it is inherited from the field or introduced by this thesis.[1]

The environment is a rectangular lattice of square cells,

$$\mathcal{C} \subset \mathbb{Z}^2, \qquad |\mathcal{C}| = H \times W,$$

where H and W are the grid height and width in cells, with a static ground truth

$$m^\star : \mathcal{C} \to \{\text{free},\ \text{occ}\},$$

in which free cells are traversable and occupied cells are walls and obstacles. Time is discrete, and one time step is one iteration of the exploration loop, as fixed in Section 2.1. The team consists of n identical

[1] The numbered statements follow the same rule. All of them are formulated by us. The assumptions collect idealizations that are standard in the field, while the problem and the lemma are our own and rest on elementary, checkable mathematics. The one principle the method imports, the replica idea of Trimpe and D'Andrea [54], is credited where it is reviewed (Section 3.3).

robots, and robot i occupies a free cell $x_{i,t} \in \mathcal{C}$ at time t. All of this is the standard occupancy-grid abstraction of Chapter 2. The first assumption states that the environment does not change during the mission.

Assumption 4.1 (static environment)**.** *The ground truth $m^\star$ is constant for the duration of the mission.*

In other words, nothing moves but the robots. The assumption fits the missions of Chapter 1, a collapsed structure or a mine surveyed after the event; environments that change while being mapped are outside the scope of this thesis.

Each robot holds its own estimate of the map,

$$m_{i,t} : \mathcal{C} \to \{\text{free},\ \text{occ},\ \text{unk}\}, \qquad m_{i,0} \equiv \text{unk},$$

and its known set collects the cells it has determined,

$$K_{i,t} = \{\, c \in \mathcal{C} : m_{i,t}(c) \neq \text{unk} \,\}.$$

Everything a robot ever learns obeys a second assumption.

Assumption 4.2 (error-free mapping)**.** *For every robot i and time t, $m_{i,t}(c) = m^\star(c)$ for all $c \in K_{i,t}$.*

In other words, a robot can be "ignorant" about a cell, never wrong about it. This is not a choice specific to this thesis. Simulation studies of exploration commonly treat the sensor as error-free and localization as solved, the standard idealizations discussed in Section 2.1, so that the results are not confounded by measurement noise. Assumption 4.2 collects the two in a single statement. Every observation lands at its true cell, in a frame the whole team shares. The assumption matters most when maps are exchanged. Two robots that both know a cell agree on its value, so fused maps can never disagree (Section 4.2).

What a robot sees from a pose x is its visibility region, recalled from Section 2.1,

$$V(x) = \{\, c \in \mathcal{C} \ :\ \|c - x\| \le R_s \ \wedge\ \text{LoS}(x, c) \,\},$$

where R_s is the sensor range measured in grid cells, and $\text{LoS}(x, c)$ is the line-of-sight condition, which holds when no wall stands between x and c. Formally, let $\text{seg}(x, c)$ be the set of cells crossed by the straight segment drawn from the center of x to the center of c. Then

$$\text{LoS}(x, c) \iff m^\star(c') = \text{free} \quad \text{for every intermediate cell } c' \in \text{seg}(x, c) \setminus \{x, c\}.$$

The endpoints are excluded so that the robot's own cell does not block the ray and so that an occupied cell can itself be observed. The first wall a ray meets is visible, and everything behind it is not. By

Assumption 4.2, a sweep writes true values: after sensing, every cell of $V(x_{i,t})$ carries its ground-truth label in $m_{i,t}$. The simulator realizes V with a finite fan of rays; the discretization is declared in Chapter 5.

Knowledge accumulates from two sources,

$$K_{i,t} = K_{i,t-1} \cup V(x_{i,t}) \cup \mathrm{recv}_{i,t},$$

where $\mathrm{recv}_{i,t}$ is the set of cells robot i receives from teammates at step t. Its content is the subject of Section 4.2; here it is enough that it exists, because promoting the received cells to a first-class object is precisely where this thesis will intervene. The update restates the fact of Section 2.2: a robot has two ways of learning a cell, driving until its own sensor sees it, or receiving it from a teammate that has already traveled there. Knowledge is monotone, $K_{i,t-1} \subseteq K_{i,t}$. Once determined, a cell stays in the robot's stored map for the rest of the mission. Storing known cells costs little in practice, since the full grid of Section 2.1 occupies a few hundred kilobytes of onboard memory. The exchange mechanism defined later in Section 4.2 depends on this property, because a sender that never re-transmits a delivered cell is correct only if the receiver keeps what it was given.

The metrics of this section must distinguish the two sources. For this purpose we define the *first-hand set*

$$S_{i,t} = \bigcup_{\tau \le t} \big(V(x_{i,\tau}) \setminus K_{i,\tau-1} \big),$$

the cells whose first entry into robot i's map came from its own sensor. A cell the robot already held, whether from past sensing or from a teammate, is never attributed again. The definition exists because the metrics below must know who paid travel for a cell, not who holds it. Consider a robot that receives a fully mapped room from a teammate and later drives through it. Its sensor re-observes every cell, its map does not change, and its first-hand set gains nothing. The drive wasted mission time, and the loss appears in the time metric defined below. Note also that first-hand sets are not disjoint across robots. Two robots can each determine the same cell first-hand, each before receiving it from the other, and the team then pays for that cell twice. To count this double payment, we use the Overlap metric defined below.

The mission-level object is the *team known set*,

$$E_t = \bigcup_{i=1}^{n} S_{i,t},$$

the region some robot has determined first-hand, the union drawn in Figure 2.3. No robot stores this set, because each robot holds only its own map. E_t belongs to the analysis, not to any robot's memory.

Since receiving never creates knowledge but only moves it between robots, every known cell entered the team through some robot's sensor, and trivially

$$E_t = \bigcup_{i=1}^{n} K_{i,t}.$$

The identity is used in both directions later. Progress is measured on the known sets, while the cost of obtaining it is attributed through the first-hand sets. Not every cell of $\mathcal{C}$ can enter E_t, because a sensor only ever observes what its rays can reach. A ray terminates on the first wall cell it meets, so the interior of a wall stays invisible to every sensor. The set a mission can ever reveal is therefore the free cells together with the wall faces bordering them, and we normalize coverage by exactly that set,

$$X = \{\, c : m^\star(c) = \text{free} \,\} \;\cup\; \{\, c : m^\star(c) = \text{occ} \;\wedge\; \exists\, c' \in N(c) : m^\star(c') = \text{free} \,\}.$$

The denominator is chosen so that coverage can reach one. A complete map contains exactly the cells of X. Dividing by $|\mathcal{C}|$ instead would count wall interior that no sensor can ever observe, and coverage would stay below one even for a complete map.

Coverage and the objective follow,

$$\text{Cov}_t = \frac{|E_t \cap X|}{|X|} \in [0,1], \qquad T_\theta = \min\{\, t : \text{Cov}_t \geq \theta \,\},$$

and the team's objective is to minimize T_θ, the time to determine a fraction θ of everything determinable. As Chapter 1 explained, the simplest benchmark of exploration is time, since the sooner the map is complete, the sooner the mission it serves can proceed. A mission can impose further criteria beyond time, already discussed there; time is the one this thesis optimizes. Results are therefore reported at several thresholds θ rather than at a single one, because policies differ most in the late mission. Early on, every selection rule finds abundant frontier close by. The last fractions of coverage lie in the few pockets that remain, and the travel spent reaching them is where the policies separate.

The second metric measures duplicated discovery,

$$\text{Overlap}_t = \frac{\sum_{i=1}^{n} |S_{i,t} \cap X|}{|E_t \cap X|} - 1 \;\geq\; 0,$$

non-negative because a union never exceeds the sum of its parts. At zero, no cell was determined twice. At one, the team paid the sensing cost of the entire explored region a second time. One case is deliberately not counted here. A robot that re-walks ground it already received from a teammate duplicates travel, not discovery; that loss appears in T_θ. The two metrics therefore play different roles. T_θ measures how fast the mission finishes, and Overlap_t measures the redundancy that the coordination

strategies of Section 3.1 were built to remove.

The objects so far describe what a robot knows about the world. The complete state a robot carries is larger, and making it explicit fixes what every later mechanism is allowed to read. Robot i maintains

$$\Sigma_{i,t} = \big(x_{i,t},\ \ m_{i,t},\ \ \{\, (D_{i\to j,t},\ T_{i\to j,t}) : j \neq i \,\} \big),$$

its own pose, its own map, and, for every teammate j, two memories of their past communication. The *delivered log* $D_{i\to j,t}$ holds the cells i has already transmitted to j. The *contact trail* $T_{i\to j,t}$ holds the poses of j that i observed while the two were in contact. Both memories record events i witnessed itself, so neither costs the channel a single bit. The memories themselves, and the use the following sections make of them, are constructions of this thesis. Their dynamics belong to the exchange protocol of Section 4.2; their purpose is anticipated here. From these two memories a sender can estimate what its receiver already knows, the estimate $\hat{K}$ at the center of Section 4.4. The state also fixes what decentralization means in this thesis. Every quantity a robot computes is a function of its own $\Sigma_{i,t}$, and everything it knows of its teammates arrived over a link. The single exception, an oracle used as an upper bound, is declared in Chapter 5.

So far, the assumptions have fixed the environment and the sensing. One aspect remains open, namely how the robot moves once it has chosen a goal. The third and final assumption of this section settles it.

Assumption 4.3 (reliable motion)**.** *A robot advances along its planned path over known free cells at one cell per time step, and a step never fails.*

Thanks to this assumption, reaching a committed goal is a matter of time, not chance. Kinematics and low-level control sit below the layer this thesis works in, as argued in Section 2.1.

The state and the assumptions assemble into the per-robot program, Algorithm 4.1, which is Algorithm 2.1 made precise in the notation of this section. Line by line it reads only $\Sigma_{i,t}$. The frontier it recomputes is the standard one of Section 2.1,

$$F_{i,t} = \{\, c : m_{i,t}(c) = \text{free} \ \wedge\ \exists\, c' \in N(c) : m_{i,t}(c') = \text{unk} \,\},$$

and the selection rule σ is fixed to MinPos in every experiment and in every arm, for the reasons argued in Section 3.1. Among coordinated selectors it demands the least from the channel, and it degrades to nearest-frontier selection when no peer information arrives. Chapter 5 states the rule exactly as the simulator implements it. All robots run the program concurrently and none has priority. The loop is the machinery the field agrees on, it stays identical in every experiment of this thesis, and our contribution is built on top of it. The budgeted selection of line 4 is different in kind. When the channel carries fewer cells than a robot could send, the protocol must decide which cells go first. That decision is the

Algorithm 4.1 One time step of robot i (all robots run it concurrently)

1: write the true state of every cell of $V(x_{i,t})$ into $m_{i,t}$
2: **for all** teammates j linked to i (Section 4.2) **do**
3: append the observed pose $x_{j,t}$ to the trail $T_{i\to j,t}$
4: send a budgeted selection of $K_{i,t} \setminus D_{i\to j,t}$; add the sent cells to $D_{i\to j,t}$
5: for each received cell c with $m_{i,t}(c) = \text{unk}$, set $m_{i,t}(c) \leftarrow m_{j,t}(c)$
6: **end for**
7: recompute the frontier $F_{i,t}$
8: **if** the current goal is reached or unattainable **then** $g_{i,t} \leftarrow \sigma(F_{i,t})$ ▷ σ = MinPos
9: advance one step along the A* path to $g_{i,t}$

subject of this thesis, and it is what varies between the experimental arms.

A word is needed on how the robots run this program together, since no robot ever waits for another. Time advances in shared rounds. Within one step every robot senses, then every linked pair exchanges, then every robot updates its plan, then every robot moves; the phases are common, the decisions inside them are not. A robot linked to several teammates converses with all of them in the same step. Each link carries its own budget in each direction and its own pair of memories, so simultaneous conversations never compete for a shared allowance, and no robot has precedence over another. No arbitration is needed either. Fusion is a union, so received content can only add knowledge, and the delta discipline never re-sends; which exact cells travel may depend on the order in which the pairs are served within a step, but correctness never does. Goals, finally, are commitments. A robot follows the path to its current goal and selects a new one when that goal is reached or becomes unattainable; the information received meanwhile acts at that moment, through the fused map the next selection reads.

Figure 4.1 draws one time step of Algorithm 4.1. The four stages along the bottom row are the machinery every frontier-based system shares. The communicate stage above them is the one this thesis designs. Section 4.2 develops its model, and Figure 4.2 in that section opens the stage in the same visual language.

The formulation is two-dimensional in the sense that robots move and sense on a planar grid, whereas a real environment extends into the third dimension. The definitions above are per-cell and carry over to a voxel grid unchanged. Chapter 6 discusses the transfer. Assumptions 4.1 to 4.3 remove, in order, the confounds of a changing world, of noise, and of execution failures.

4.2 The Communication Channel

Two elements of Algorithm 4.1 are still undefined. Line 2 tests whether a teammate is linked to robot i, and line 4 sends a budgeted selection of the unsent cells. This section defines the link, then the budget,

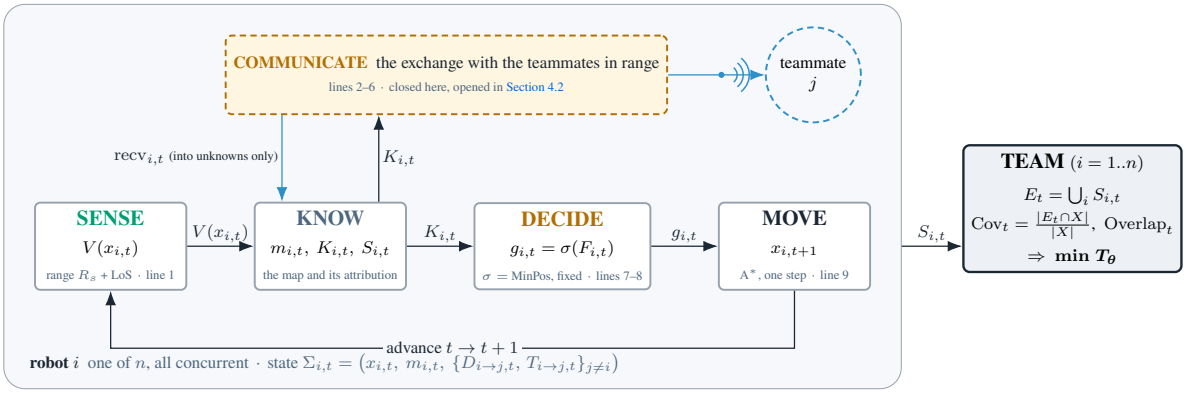

Figure 4.1: *The per-robot loop.* One time step of robot i (Algorithm 4.1): sense, know, decide, move, with the team outcome collected outside the robot. At the top of the figure sits the communicate stage, shown closed on purpose. While a link holds, robot i reads its map up into the exchange and fuses what arrives into its unknowns; the budgeted selection inside it is the object of this thesis. Every other stage is standard machinery. The stage is opened, with its channel, budget, and symbols, in Section 4.2 (Figure 4.2).

and then examines how much freedom the two leave to the exchange. The answer is that they force every step of it except one, the order in which the cells travel.

When two robots can talk is a property of the link model. The basic model is the range disc of Section 2.3,

$$\mathrm{Link}_t(i,j) \iff \|x_{i,t} - x_{j,t}\| \le R_c,$$

with R_c the communication range, measured in the same grid units as robot positions. A stricter variant lets the same walls that block the sensor block the radio. The variant used in this work tests the segment between the robots against the walls the two have observed. Let

$$\mathrm{Occ}_t = \{\, c \in \mathcal{C} : m_{i,t}(c) = \mathrm{occ} \ \vee \ m_{j,t}(c) = \mathrm{occ} \,\}$$

collect the walls either robot has observed. Then

$$\mathrm{Link}_t(i,j) \iff \|x_{i,t} - x_{j,t}\| \le R_c \ \wedge \ \mathrm{seg}(x_{i,t}, x_{j,t}) \cap \mathrm{Occ}_t = \varnothing.$$

In other words, the pair is linked when the robots are close enough and no observed wall crosses the segment between them. Cells still unknown do not cut the link there, so a wall nobody has mapped never breaks a connection. The campaigns of this thesis use neither of these two models. They run the third one below, in which walls play no part and connectivity is a function of distance and of an opportunistic session. That choice errs toward connectivity on purpose, since the model then grants at least as many contact opportunities as a real radio would, and the limits on communication studied in

this thesis therefore come from the budget rather than from the link model. Whichever link model a campaign uses, it is identical in every arm of that campaign, and no comparison in this thesis depends on the choice.

Both gates are deterministic, while Section 2.3 argued that no simple rule predicts when a real link is up. Reproducing the true behavior of a radio inside the simulation would complicate it beyond its scope. Attenuation depends on the walls between the antennas, on their material, and on reflections, and none of these exists in a two-dimensional occupancy grid. One behavior of modern radios is dependable, however. In ordinary indoor environments, laboratories, warehouses, and office buildings, two robots within a short radius reach each other with near certainty, regardless of what stands between them. The third model, introduced by this thesis, builds on exactly this split. Inside a core radius, communication is certain; beyond it, contact becomes a matter of probability. Each unordered pair of robots carries a session switch $W_{ij,t} \in \{\mathrm{ON}, \mathrm{OFF}\}$, updated once per step with

$$\Pr[\mathrm{OFF} \to \mathrm{ON}] = p(d) = p_0 \exp\!\Big(-\tfrac{\max(0,\, d - R_c)}{\lambda}\Big), \qquad \Pr[\mathrm{ON} \to \mathrm{OFF}] = 1/T_{\mathrm{on}},$$

where $d = \|x_{i,t} - x_{j,t}\|$, and the pair is linked when it sits inside the core disc or its session is open,

$$\mathrm{Link}_t(i,j) \iff d \le R_c \ \vee \ W_{ij,t} = \mathrm{ON}.$$

Within the core radius, two robots communicate with certainty, the short-range behavior of a modern radio. A distant pair opens a session with a probability that is largest at the core boundary and decays on the length scale λ, and an open session lasts T_{on} steps on average, persisting even while the robots separate. Here p_0 is the opening probability at the core boundary, λ is measured in grid cells, and T_{on} is measured in time steps. The standard settings return as limits. At $p_0 \to 0$ or $\lambda \to 0$ no session ever opens and the pure disc remains; with R_c at the map diagonal every pair is linked at all times, the full connectivity of Section 2.3.

The switch is a two-state Markov chain, so two of its properties can be computed in closed form: how long a pair waits for a session on average, and what fraction of the time it spends linked. A pair held at distance d waits $1/p(d)$ steps on average for a session, and in the stationary regime it is linked a fraction

$$\phi(d) = \frac{p(d)\, T_{\mathrm{on}}}{1 + p(d)\, T_{\mathrm{on}}}$$

of the time. In practice, a pair exploring at that separation alternates silence and contact. It spends about $1/p(d)$ steps out of touch, then holds a link for about T_{on} steps, exchanges what the budget allows, and falls silent again, so over a long stretch the fraction of steps with a live link approaches $\phi(d)$. Because the model predicts these statistics, the implementation can be tested against them, and Chapter 5 reports exactly this check. The model simplifies deliberately in three ways: session duration does not depend

on distance, an open session persists while the robots separate, and the exponential is the simplest decay law with a single scale. The randomness of the switch is drawn from a stream keyed by the seed, the pair, and the step, so every policy run at the same seed faces the identical schedule of links. A difference between policies can therefore never be explained by luck in the channel.

A time step at which $\mathrm{Link}_t(i,j)$ holds is a *contact step* for the pair, and contact steps are the only moments at which communication happens. Map content is not the only thing a contact must carry. The selector of Section 4.1 needs the poses of the linked teammates, so pose messages compete with map cells for the same channel in principle. In practice the competition is negligible. A pose is one cell address, about $\lceil \log_2 |\mathcal{C}| \rceil$ bits, while the map payload is thousands of cells, and the field systems reviewed in Section 3.3 prioritize exactly this lightweight telemetry ahead of map data. We therefore treat positions as free and charge the budget to map content alone.

Assumption 4.4 (positions are free)**.** *Robots in contact exchange their current poses at no cost against the budget; the budget constrains map content only.*

Since only the poses observed during a contact are available to the observer, a robot never knows where an unlinked teammate currently is.

Contact steps drive the two memories of $\Sigma_{i,t}$. The contact trail appends every pose observed while linked,

$$T_{i\to j,t} = T_{i\to j,t-1} \cup \{x_{j,t}\} \qquad \text{whenever } \mathrm{Link}_t(i,j),$$

and the delivered log grows by exactly what was transmitted,

$$D_{i\to j,t} = D_{i\to j,t-1} \cup R_{i\to j,t-1},$$

where $R_{i\to j,t}$ is the message of step t, defined below. The candidate pool of a contact step is what the sender knows and has not delivered,

$$\Delta_{i\to j,t} = K_{i,t} \setminus D_{i\to j,t},$$

and the message is a subset of it, $R_{i\to j,t} \subseteq \Delta_{i\to j,t}$. Restricting the message to this pool is correct because knowledge is monotone (Section 4.1). A cell delivered once stays with the receiver, so sending it again cannot add information. What this thesis designs is the order in which this pool is transmitted. The receiver folds the message into its own unknowns,

$$m_{j,t}(c) \leftarrow m_{i,t}(c) \qquad \text{for } c \in R_{i\to j,t} \text{ with } m_{j,t}(c) = \mathrm{unk},$$

and by Assumption 4.2 no conflict can arise. If a message is lost in transmission, the delivered log

does not advance, and its cells simply remain in the pool for a later contact, so loss delays the exchange without ever corrupting it.

How much a contact step may carry is the budget. Its formal treatment below, the price per cell and the per-step cap, is introduced by this thesis. A transmitted cell must name its position and its value, so its price is

$$b = \lceil \log_2 |\mathcal{C}| \rceil + 2 \text{ bits},$$

an address plus the two bits of the tristate value.[2] A budget of B bits per contact step and per direction then admits

$$k = \lfloor B/b \rfloor$$

cells each way, and a contact lasting m contact steps carries at most mk cells in each direction. The experiments enforce the cap in three forms. The plain form applies k at every contact step. The banked form lets a credit grow by B bits at every step, linked or not, and spends it at contacts, so it caps the average throughput instead of the instantaneous rate. The fractional form ties the cap to the demand, admitting $\lceil \text{frac} \cdot |\Delta_{i \to j,t}| \rceil$ cells, with a minimum of one; the fraction is dimensionless and therefore comparable across maps and team sizes, and at $\text{frac} = 1$ every contact delivers its whole pool, which reproduces the unlimited exchange exactly. Results are reported by sweeping the budget as an axis, with tiers corresponding to the radio classes of Section 2.3.

With the pool, the price, and the cap fixed, one degree of freedom remains. Whenever $|\Delta_{i \to j,t}| > k$, some cells go and the rest wait, so transmitting is selecting. Every selection of k cells is the prefix of some ordering of the pool, so the protocol can be written in a single line,

$$R_{i \to j,t} = \text{the first } k \text{ cells of } \Delta_{i \to j,t} \text{ under an ordering } \pi,$$

and π is the only entry a designer can change. In existing deployed systems the pool goes out in the order the map array stores it, row by row from one corner of the grid. As a result, a teammate receives the cells in an order that has nothing to do with how useful they are to it. Even the systems designed for limited channels, reviewed in Section 3.3, do not choose this order deliberately. Figure 4.2 assembles the pieces of this section. The link condition opens the stage, the pool flows through π and the cut at k, the sent prefix advances the delivered log, and every pose observed while linked feeds the trail. Every symbol of the figure is now defined, and the question of this thesis can be stated exactly.

Problem 4.1 (what to send)**.** *Under the channel of this section, choose the ordering π, computed by each sender from its own state $\Sigma_{i,t}$ alone, so that the resulting missions minimize T_θ.*

[2] A cell of the pool is never unknown, so one value bit would suffice. The two-bit format is kept because the price is identical across policies, and every comparison in this thesis is invariant to it.

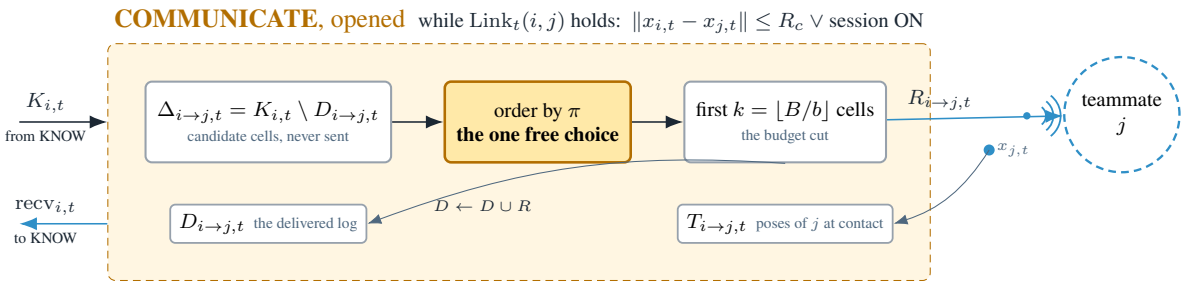

Figure 4.2: *The exchange, opened.* The stage shown closed in Figure 4.1: while a link holds, the never-sent cells $\Delta_{i\to j,t}$ are ordered by π and cut at the budget, $k = \lfloor B/b \rfloor$ cells per contact step and direction; the sent prefix $R_{i\to j,t}$ advances the delivered log, and every pose observed while linked feeds the contact trail. The ordering π is the one free choice of the protocol. This section defines every symbol in this picture.

The statement is short but complete. Any sharing policy whatsoever amounts to some ordering of the pool, so the space of answers to Problem 4.1 contains every policy the protocol admits, from the accidental raster order to the best ordering possible. The clause on $\Sigma_{i,t}$ fixes what a sender is allowed to know when it chooses. It may use what it has seen, what it has sent, and what it has been told, and nothing else. The declared oracle of Chapter 5 breaks exactly this restriction, on purpose, to mark the ceiling that no realizable policy can exceed. The objective ties the choice to the mission. An ordering is good exactly when it makes the team finish sooner, not when its cells look reasonable in isolation. Table 4.1 collects the objects of Sections 4.1 and 4.2 in a single view, each stage of Algorithm 4.1 beside the formula that defines it. Answering the problem requires knowing where a transmitted cell has value at all, and that is the subject of Section 4.3.

4.3 Where the Value of Communication Concentrates

Problem 4.1 asks for an ordering of the pool, and an ordering is only as good as the cells it puts first. The natural starting point is to establish where a delivered cell can change anything on the receiver's side. Each claim that follows states its basis: grid logic, arithmetic, or a hypothesis that the experiments of Chapter 5 test.

The analysis needs one more object of the receiver's map. The *knowledge boundary* of robot j is the unknown side of its frontier,

$$\partial_{j,t} = \{\, c \in \mathcal{C} : m_{j,t}(c) = \text{unk} \ \wedge\ \exists\, c' \in N(c) : m_{j,t}(c') = \text{free} \,\},$$

the unknown cells that touch known free space. The frontier $F_{j,t}$ and the boundary $\partial_{j,t}$ are the two sides of the same rim. The frontier is where j can stand; the boundary is what j can determine next.

What a received cell does to the receiver is settled by the definitions alone.

Lemma 4.2 (value is confined to the boundary)**.** *Let c be a cell delivered to robot j at a contact step of Algorithm 4.1. The delivery can change j's goal selection only if $c \in \partial_{j,t}$, or if c connects to $K_{j,t}$ through other cells delivered at the same contact step.*

Four cases cover every delivered cell. If j already knows c, the message changes nothing, because the two robots agree on its value (Assumption 4.2) and fusion writes only into unknowns. If c lies on $\partial_{j,t}$ and is a wall, the boundary is sealed there; a frontier cell dies as soon as its last unknown neighbor is determined, and by monotonicity it never returns. If c lies on $\partial_{j,t}$ and is free, the boundary advances; c borders unknown space, becomes frontier itself, and the rim moves one layer outward. If c lies deep in j's unknown, it enters the map but no path over known free cells reaches it, so the goal selection, which considers reachable candidates only, is unchanged; the exception is exactly a chain of delivered cells connecting c to $K_{j,t}$, which reduces to the boundary cases. Every case is grid logic, and nothing is estimated.

A single free cell therefore never changes a robot's goal by itself; redirection is a cumulative effect. During a contact that lasts m steps, the sender delivers up to mk cells, layer after layer, and the receiver's boundary reacts in two ways. Where the delivered ground is closed by walls, the frontier cells along that stretch lose their unknown neighbors and disappear for good. Where it is open, the boundary recedes from the receiver, and the goal ranking of MinPos, re-evaluated on the fused map at each new selection, eventually flips to openings that are genuinely unexplored. The value of such a flip becomes concrete once the two ways of learning ground are put side by side. In one time step a robot advances at most

Table 4.1: One view of the system defined in Sections 4.1 and 4.2. The five stages follow Algorithm 4.1; the lower rows are the team quantities every arm is judged on. The communicate stage carries the one free choice.

Stage	Role	Defining object
SENSE	perceive	$V(x_{i,t})$ written into $m_{i,t}$
COMMUNICATE	**choose what to send**	$R_{i\to j,t}$ = first k of $\Delta_{i\to j,t}$ under π
KNOW	accumulate	$K_{i,t} = K_{i,t-1} \cup V(x_{i,t}) \cup \mathrm{recv}_{i,t}$
DECIDE	choose the goal	$g_{i,t} = \sigma(F_{i,t})$ with σ = MinPos
MOVE	act	one step of the A* path to $g_{i,t}$
OUTCOME	team progress	E_t, Cov_t
WASTE	duplicated discovery	$\mathrm{Overlap}_t$
OBJECTIVE	minimize	$T_\theta = \min\{\, t : \mathrm{Cov}_t \geq \theta \,\}$

one cell, and its sensed disc sweeps at most $2\sqrt{2}\,R_s$ new cells, the area gained by a disc of radius R_s displaced along a diagonal step, so the robot's own discovery is capped at about $2\sqrt{2}\,R_s$ cells per travel step. A useful received cell costs b bits and no travel at all. In practical terms, fresh ground is learned fastest by the robot's own sensor; what the sensor cannot reveal is that the ground ahead has already been covered by a teammate. The cells that carry exactly this information are the ones worth bits, and their value is the entire excursions they prevent, as the redirect estimate below makes precise.

What share of the pool lies on the receiver's boundary? To measure it, we define the *useful fraction* of a sender–receiver pair,

$$\varepsilon = \frac{|\partial_{j,t} \cap \Delta_{i\to j,t}|}{|\Delta_{i\to j,t}|},$$

the share of the sender's pool that lies on the receiver's boundary. This share is small for a geometric reason. The boundary is a rim one cell thick, while the pool is the interior of everything the sender has explored, so the numerator grows like a perimeter, the denominator like an area, and ε shrinks as the maps grow. The scaling is heuristic rather than proven, and what Chapter 5 measures is its consequence, the mission time a policy needs, rather than the share itself.

The price of the blind order follows from one hypothesis. The raster rank of a cell, its position in the memory layout, carries no information about boundary membership, so the first k cells of the pool are, with respect to $\partial_{j,t}$, a sample of size k. Under this hypothesis the expected number of useful cells per message is $k\varepsilon$, and whenever $\varepsilon < 1/k$ it drops below one. The budget is then fully spent, and the boundary, in expectation, does not move. The consequence of this hypothesis is what Chapter 5 examines, by varying the order of transmission at an identical budget and reading the mission time that follows.

Different orderings spend the same k cells with very different effect, and comparing them fairly requires a measure of how much of each message lands where it can act. For this purpose we introduce the *channel efficiency* of an ordering, a definition of this thesis, as the useful share of its messages,

$$\eta(\pi) = \frac{\mathbb{E}\,|R_{i\to j,t} \cap \partial_{j,t}|}{k}, \qquad B_{\text{eff}} = \eta\, B,$$

so that B_{eff} is the effective bandwidth spent on cells that can act. Under the hypothesis above the raster order sits at $\eta = \varepsilon$. An ordering that knew $\partial_{j,t}$ exactly could hold $\eta = 1$ for as long as the boundary offers k unsent cells,

$$\eta(\text{raster}) = \varepsilon, \qquad \eta(\text{oracle}) = 1,$$

and the distance between the two is a factor of $1/\varepsilon$ at identical bits. How much of that distance an ordering computed from $\Sigma_{i,t}$ alone can cross is the question that Section 4.4 answers, and Chapter 5 judges the answer by the time the missions take.

Efficiency also decides whether a redirect happens at all. Suppose that flipping the receiver's goal requires advancing its boundary G cells deep along a front w cells wide, about Gw delivered cells in place. A contact delivers $k\eta$ useful cells per step, so the redirect needs about

$$T_{\text{flip}} \approx \frac{G\,w}{k\,\eta}$$

contact steps. At $\eta \approx 1$ this is a handful of steps and fits inside a single contact. At $\eta \approx \varepsilon$ it is inflated by $1/\varepsilon$ and outlasts any contact the connectivity models of Section 4.2 produce. The blind order does not redirect the receiver more slowly. Within a real contact, it fails to redirect it at all, and the entire trip into already-covered ground is paid in full.

Turning this analysis into a method runs into one obstacle. Aiming at $\partial_{j,t}$ requires knowing where the receiver's boundary runs, and the boundary is a function of $m_{j,t}$, the very map the channel cannot carry. The escapes attempted in the literature were reviewed in Section 3.3: coordinating the selection through the channel, keeping sampled models of teammates, or learning the messages end to end. Problem 4.1 rules them out here, since an ordering computed from $\Sigma_{i,t}$ alone admits no negotiation round and no server. The sender's own state, however, already contains two records that bound the receiver's knowledge without any message at all, the delivered log $D_{i\to j,t}$ and the contact trail $T_{i\to j,t}$. Turning them into an estimate of $\partial_{j,t}$, and the estimate into an ordering, is the method of the next section.

4.4 The MIRROR Policy

Section 4.3 ended on the two records of $\Sigma_{i,t}$ that constrain the receiver's knowledge without any message. This section turns them into an estimate of the receiver's map, and the estimate into the ordering that Problem 4.1 asks for. The policy is called MIRROR, and the name describes the mechanism. The sender holds, on its own side, a reflection of the receiver's map, assembled from those two records and updated only by events the sender itself witnessed, the way a mirror shows a scene without touching it. Every message is aimed at the place where that reflection ends, which is where the sender would aim if it could see the receiver's true boundary.

The estimate is the union of the delivered log with sensing discs placed at the trail poses,

$$\hat{K}_{j,t} \;=\; D_{i\to j,t} \;\cup \bigcup_{p\,\in\, T_{i\to j,t}} \operatorname{disc}_{R_s}(p), \qquad \operatorname{disc}_{R_s}(p) = \{\, c \in \mathcal{C} : \|c - p\| \le R_s \,\},$$

the sender's estimate of the receiver's known set. Its two parts carry different kinds of certainty. Every cell of the delivered log is known to the receiver exactly, because the protocol of Section 4.2 delivered it and the receiver never discards delivered cells. Every disc, by contrast, is deduced rather than communicated. A pose observed at a contact implies that the receiver was there with its sensor running,

so the sender can credit it with the surroundings of that pose, up to the common sensor range R_s, without a single cell being transmitted. The estimate reads only $D_{i\to j,t}$ and $T_{i\to j,t}$, both components of $\Sigma_{i,t}$, so it costs the channel nothing. This zero-cost replica of the receiver's map is a contribution of this thesis.

The estimate is not exact, and its two error sources are stated here, where it is defined. The discs ignore occlusion, so around an observed pose the estimate overestimates the receiver's knowledge; walls the receiver could not see through are counted as seen. Between contacts the estimate also goes stale, because everything the receiver senses away from the sender, or receives from other robots, is invisible to the sender. Neither error can corrupt the exchange. Indeed, a cell wrongly believed known loses priority in the ordering but stays in the pool, and a cell wrongly believed unknown wastes at most its slot in a message, since fusion at the receiver ignores what it already has. For this reason, the errors degrade the efficiency of the ordering, never the correctness of the protocol.

Lemma 4.2 states where a message can act, and the estimate makes that target computable. The estimated boundary of the receiver is the edge of the estimate, and the ordering sends first the cells that move it. For each cell c of the pool, define the edge flag

$$e(c) = \mathbf{1}\big[\,\exists\, c' \in N(c) : c' \notin \hat{K}_{j,t}\,\big],$$

which asks whether c borders ground the receiver, by the sender's estimate, does not know. The pool is sorted ascending by the key

$$\big(\,1 - e(c),\;\; \|c - p_{\text{last}}\|^2\,\big),$$

where p_{last} is the receiver's last observed pose, and the message is the first k cells under this order. Cells that move the estimated boundary go first, and among them the ones closest to where the receiver was last seen acting; everything else follows behind. The key implements the lemma using the estimate instead of the truth.

Before assembling the algorithm, one alternative deserves to be examined, because it is the first that comes to mind in this setting. Why not simply send the frontier? The proposal fits inside Problem 4.1 as a selection rule of its own, one that ranks the sender's rim first and sends nothing else, so the comparison is well-posed at identical bits. The analysis of Section 4.3 says what to expect of it. The sender's frontier marks where the sender's knowledge ends, and Lemma 4.2 confines value to where the receiver's knowledge ends; the two sets coincide only when the two maps agree, which is exactly when communication has nothing left to do. The payload also carries the wrong kind of cell. A frontier is made of free cells, so it can never deliver a wall, and a wall on the receiver's boundary is the permanently valuable delivery, the one that seals a frontier for good. A thin rim of free cells landing deep in the receiver's unknown is the no-effect case of the lemma, an unreachable island; where the rim

Algorithm 4.2 MIRROR: the choice of what to send, run by sender i toward receiver j at one contact step (the selection that Algorithm 4.1 leaves open)

Require: map $K_{i,t}$, delivered log $D_{i\to j,t}$, contact trail $T_{i\to j,t}$, budget k

$\hat{K}_j \leftarrow D_{i\to j,t} \cup \bigcup_{p\in T_{i\to j,t}} \mathrm{disc}_{R_s}(p)$ ▷ the belief; zero bits

$\Delta_{i\to j,t} \leftarrow K_{i,t} \setminus D_{i\to j,t}$

for all $c \in \Delta_{i\to j,t}$ **do**

  $e(c) \leftarrow \mathbf{1}\big[\,\exists\, c' \in N(c) : c' \notin \hat{K}_j\,\big]$ ▷ does c push j's knowledge outward?

end for

sort $\Delta_{i\to j,t}$ ascending by $\big(1 - e(c),\ \|c - p_{\text{last}}\|^2\big)$ ▷ boundary movers first, nearest first

$R_{i\to j,t} \leftarrow$ the first k cells; transmit $\{(c,\, m_{i,t}(c)) : c \in R_{i\to j,t}\}$ ▷ the budget cut

$D_{i\to j,t+1} \leftarrow D_{i\to j,t} \cup R_{i\to j,t}$ ▷ the aim advances

is reachable, it advertises the very openings the sender is about to explore itself, and can pull the two robots together rather than apart. Whether this happens is a measurable question. There is, moreover, a difference in how long the two kinds of content stay true. Consider a robot that reaches the doorway of an unexplored room. At that moment the doorway cells are its frontier. Three steps later the robot has driven through, the room is mapped, and the same cells are ordinary interior; a teammate that received them as the frontier now holds a snapshot of a rim that no longer exists. The walls and the floor of that room, by contrast, are facts forever under Assumption 4.1. A protocol that never re-sends should therefore spend its bits on durable map content, which is exactly what MIRROR transmits. The frontier payload, and the variant that adds the rim walls, run as arms of Chapter 5 at identical bits per contact; the question is settled there by measurement.

Algorithm 4.2 assembles the pieces into the complete procedure a sender runs at one contact step. Its place in the system is simple. Whenever two robots hold a link, Algorithm 4.1 requires each of them to choose which of its unsent cells to transmit, and up to now that choice was left abstract, an ordering π followed by the budget cut. Algorithm 4.2 is that choice, computed by the sender alone; sensing, fusion, goal selection, and motion continue exactly as before. MIRROR therefore integrates where the problem itself lives. It takes over the decision that Problem 4.1 isolated, the choice of what crosses the channel, while the message format, the fusion rule, the goal selector, and the planner remain exactly as they were.

Figure 4.3 follows one exchange through the policy, on the floor plan the thesis figures share. In panel 1 two robots have explored separately, each holding its own map, with no link between them. In panel 2 a contact opens; the poses cross the link and enter the trails, and Algorithm 4.2 has its inputs. In panel 3 the sender builds its mirror. The imagined disc surrounds the receiver's observed pose, the boundary the estimate predicts is drawn in red, and the queue head lights up, marking the first cells of the pool under the key. In panel 4 the delivery lands, and the sent prefix enters the receiver's map exactly where the estimated boundary ran. In panel 5 the mission resumes; both robots re-select goals on their fused maps and the team splits, each toward ground that the other has not covered. The direction shown is the

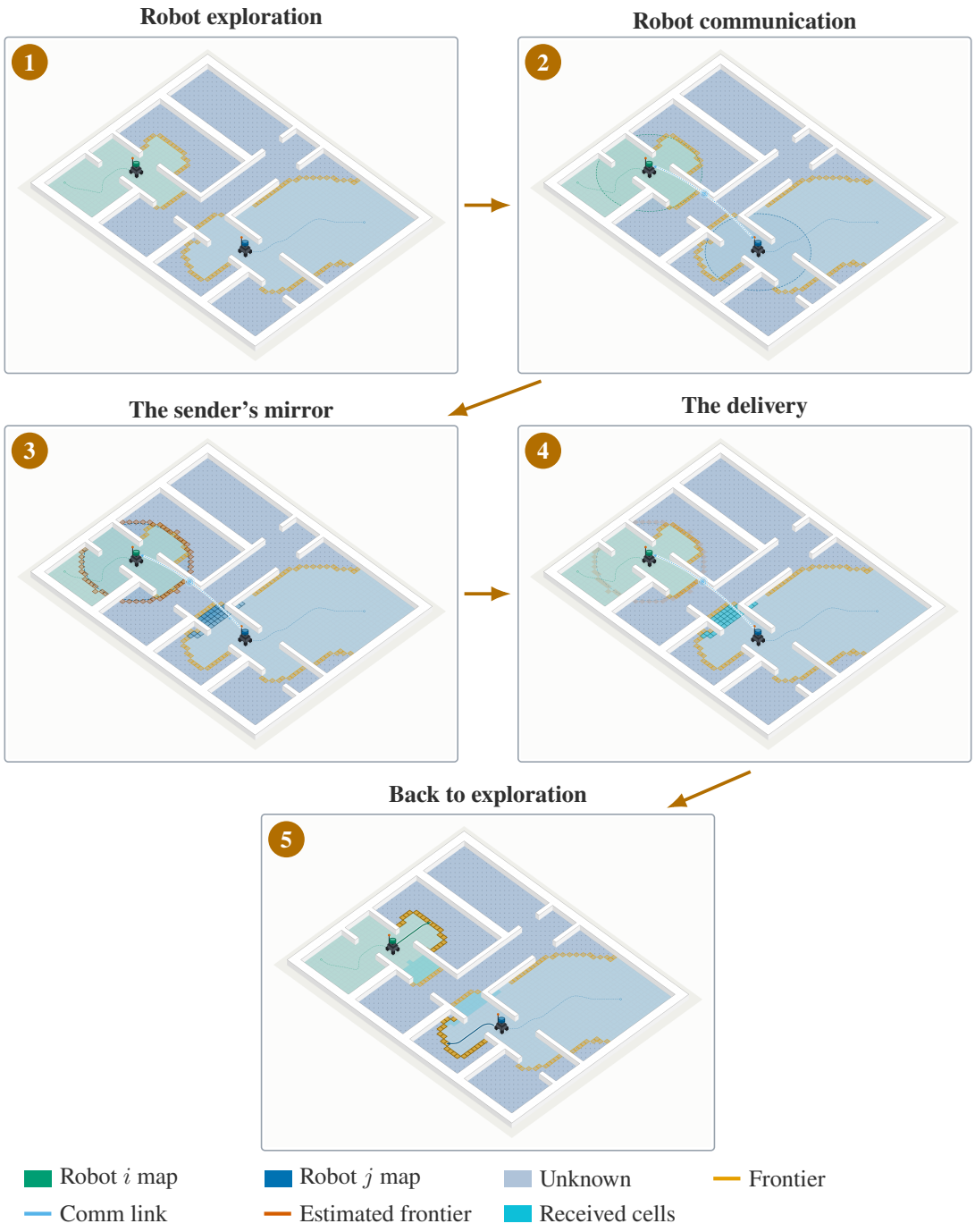

Figure 4.3: *One exchange under MIRROR.* (1) Two robots explore apart, each holding its own map. (2) A contact opens and the observed poses enter the trails. (3) The sender builds its mirror: the imagined disc around the receiver's observed pose, the estimated boundary in red, and the queue head lit, the first cells under the key of Algorithm 4.2. (4) The delivery lands where the estimated boundary ran. The transmitted cells enter the receiver's map as known ground, the delivered region stops being unknown to the receiver, and the receiver's frontier moves onto the far edge of that region. (5) The mission resumes on the fused maps and the team splits. The legend below the panels applies to all five. Panels are drawn by the simulator's own rules on the floor plan of Figure 2.3.

information-rich one, and the protocol runs symmetrically; the closing panel also shows the reverse delivery.

What the algorithm does over repeated contacts follows from its last line alone. The delivered log grows by exactly what was sent, so the estimate of the next contact step contains it, its edge lies one layer farther out, and the next message peels the next layer of the receiver's estimated boundary. In fact, the progression asks nothing of the receiver. No message ever travels back to steer the aim, and the receiver never describes its own map; the sender's log alone records what was delivered, and the log is what pushes the estimated boundary outward from one contact to the next. The order of transmission has, moreover, a second consequence. Because the message is built best-first under the policy's own ranking, a link that drops at any moment leaves the receiver with the cells the policy valued most for exactly that many bits, and for this reason no assumption on contact duration is ever needed. A third property concerns cost. MIRROR reorders the same pool every other policy draws from, so at equal budget it transmits the same number of cells at the same price b. This matters because it makes attribution fair. Any difference in outcome between two policies at the same budget must come from which cells were chosen, not from how many bits were spent. Together with the fact that the estimate reads $\Sigma_{i,t}$ alone, this makes MIRROR an admissible answer to Problem 4.1.

One contact step amounts to a mask union for the estimate, a linear pass for the edge flags, and a sort of the pool, $O(|\Delta| \log |\Delta|)$ in the pool size. The ordering runs only when the pool exceeds the cap, so above the budget at which the cut stops binding every policy delivers its whole pool and all curves meet at the unlimited ceiling. What separates MIRROR from that ceiling is the quality of its estimate. The discs are generous, the trails go stale between contacts, and both gaps push its efficiency η below the oracle's. Whether the estimate recovers enough of that distance to matter, across budgets, team sizes, and maps, is the subject of Chapter 5, which reads it in mission time.

4.5 Summary

The method of this chapter was built step by step for clarity, each piece introduced together with its motivation. Seen assembled, it is one short program. Algorithm 4.3 lists it in full, from deployment to termination. The robot starts with nothing, an unknown map and empty communication memories (line 1). Every cycle then runs the four stages of Figure 4.1. The robot senses (line 3). At every live link it records the teammate's pose, rebuilds its estimate of that teammate's map, orders its pool so that the estimated boundary movers come first, transmits what the budget admits, advances the delivered log, and fuses whatever arrives (lines 5 to 14). On the fused map it recomputes the frontier, commits to a goal under MinPos, and takes one step (lines 16 to 21). The mission ends when no frontier remains, that is, when every reachable cell is known (line 18). The assembled view also makes precise what this thesis brings that did not exist before. First, the formal ground is new. The what-to-send problem in this

Algorithm 4.3 The complete method, from deployment to termination: robot i under MIRROR (all robots run it concurrently)

$t \leftarrow 0$; $m_{i,0} \equiv \text{unk}$; $D_{i\to j,0} \leftarrow \emptyset$ and $T_{i\to j,0} \leftarrow \emptyset$ for every teammate j ▷ deployment
loop
  write the true state of every cell of $V(x_{i,t})$ into $m_{i,t}$ ▷ sense
  for all teammates j with $\text{Link}_t(i,j)$ **do**
    append the observed pose $x_{j,t}$ to the trail $T_{i\to j,t}$
    $\hat{K}_j \leftarrow D_{i\to j,t} \cup \bigcup_{p\in T_{i\to j,t}} \text{disc}_{R_s}(p)$ ▷ the estimate
    $\Delta_{i\to j,t} \leftarrow K_{i,t} \setminus D_{i\to j,t}$
    for all $c \in \Delta_{i\to j,t}$ **do**
      $e(c) \leftarrow \mathbf{1}\big[\,\exists\, c' \in N(c) : c' \notin \hat{K}_j\,\big]$
    end for
    sort $\Delta_{i\to j,t}$ ascending by $\big(1 - e(c),\ \|c - x_{j,t}\|^2\big)$ ▷ the ordering
    $R_{i\to j,t} \leftarrow$ the first k cells; transmit $\{(c,\ m_{i,t}(c)) : c \in R_{i\to j,t}\}$
    $D_{i\to j,t+1} \leftarrow D_{i\to j,t} \cup R_{i\to j,t}$
    for each received cell c with $m_{i,t}(c) = \text{unk}$, set $m_{i,t}(c) \leftarrow m_{j,t}(c)$ ▷ fuse
  end for
  recompute the frontier $F_{i,t}$ on the fused map
  if $F_{i,t} = \emptyset$ **then**
    stop ▷ every reachable cell is known
  end if
  if the current goal is reached or unattainable **then** $g_{i,t} \leftarrow \sigma(F_{i,t})$ with $\sigma = \text{MinPos}$ ▷ decide
  advance one step along the A* path to $g_{i,t}$; $t \leftarrow t + 1$ ▷ move
end loop

form, the reduction of the exchange to an ordering, and the confinement of a message's value to the receiver's boundary are constructions of this thesis (Problem 4.1, Lemma 4.2). Second, the receiver estimate is new. For every teammate, and across the whole mission, the sender maintains a picture of what that teammate knows, built from two records every deployed system already possesses and none uses, at no cost on the channel. Third, the aimed exchange is new. Every message is ordered by its value to the specific robot receiving it, with guarantees that hold by construction, the self-advancing aim, the anytime prefix, and a cost identical to every other policy's. None of the systems reviewed in Section 3.3 does any of the three. Moreover, every line of the program reads the robot's own state $\Sigma_{i,t}$ and nothing else; the method is decentralized to the letter.

Behind the machinery stands one practical necessity, and the close of the chapter is the place to state it plainly. On the platforms this thesis targets, a contact carries a small fraction of what a robot has learned since the last meeting; Section 2.3 put the gap at orders of magnitude. Faced with that gap, existing systems meet more cleverly, compress the map, or send less, and then let the memory layout of an array decide which cells actually cross (Section 3.3). The practical consequence is teams that re-explore ground already paid for, because the one fact that would have prevented the trip was never

delivered where it mattered. MIRROR exists to remove exactly this failure. It makes every scarce contact deliver the cells most likely to change the receiver's next decision, it adds not a single bit of overhead to the channel, and it asks nothing of the rest of the system.

The chapter condenses into four results. Sections 4.1 and 4.2 formalized decentralized exploration under a budgeted channel and showed that the protocol keeps exactly one free choice, the ordering of the unsent cells (Problem 4.1). Lemma 4.2 confined the value of a transmitted cell to the receiver's knowledge boundary, and the efficiency η measured how far from that boundary a blind order spends its budget. MIRROR answers the problem with an estimate of the boundary built from the delivered log and the contact trail, two records the sender already owns, at no cost on the channel. Three guarantees follow directly from this design. The aim advances with every message, the transmitted prefix is the policy's best use of however many bits a contact allowed, and the cost equals every other policy's at the same budget.

What the analysis so far cannot settle is quantitative. How sharp is the estimate on real floor plans, where occlusion and stale trails erode it? How much of the distance between the blind order and the oracle does an honest ordering actually recover? How does that answer change with the amount of communication available, and with the size of the team? The next chapter describes the simulator and the benchmark designed to answer these questions, and reports the measurements themselves.

5 Experiments and Results

Chapter 4 showed how a sender can aim its messages using only records it already owns, and left open how much that aim is worth. This chapter measures it, beginning with the simulator that runs the model and the benchmark of seven real floor plans, six of which carry the campaigns. The results then follow in the natural order, from what the choice buys at a fixed budget, through what happens as the team and the sensor change, to the sweep of the budget itself.

5.1 The Simulator

Chapter 4 defined the exploration model, the channel, and the MIRROR policy on paper. Every measurement in this chapter comes from missions executed inside that model, and executing it requires a simulator that realizes it exactly and can repeat it at scale. This section describes the simulator built for this purpose and gives the reason behind each of its choices.

The simulator is a purpose-built, lightweight two-dimensional engine, written in Python on the scientific stack, with the raycasting kernel accelerated by a just-in-time compiler and verified bitwise-identical to its reference implementation. It extends an exploration codebase internal to the laboratory. Its content is exactly the model of Chapter 4. Each robot carries its own occupancy grid, senses along rays that stop at the first wall, recomputes its frontier, commits to goals under MinPos, plans with the A* algorithm, and exchanges map content under the budgeted channel of Section 4.2; the program each robot executes is Algorithm 4.1, line by line. Nothing else is simulated. The experiments must measure the model of Chapter 4, and a world richer than the model would obscure what is being measured.

For the same reason, the simulator deliberately uses no robotics middleware and no physics engine, neither ROS nor Gazebo. The experiments compare sharing policies, and between two compared runs the only difference is which map cells cross the link. Trajectory tracking and low-level control sit outside the scope of this thesis, as declared in Section 2.1. A physics engine would therefore add computational cost and new sources of noise without adding validity to the comparison this chapter runs.

What the comparison does need is scale and determinism, and the lightweight engine provides both. The campaigns of this chapter total thousands of seeded runs, across team sizes from two to seven robots. Each seed draws different random starting positions inside the environment, and every configuration is repeated over thirty such seeds on each map, so no result depends on one fortunate deployment. Every run is also deterministic. Repeating it with the same seed reproduces it bit for bit, a property verified across macOS and Linux machines, and the randomness of the channel is drawn from a stream keyed by the seed, the robot pair, and the time step (Section 4.2). Two policies run at the same seed therefore face the identical map, the identical starting poses, and the identical schedule of contact opportunities. Every comparison in this chapter is a paired comparison, in which two policies are judged on the same mission, seed by seed, under identical conditions. Each competing configuration, one sharing policy run under one fixed set of settings, is called an arm. A difference between two arms can then have exactly one cause, the cells each policy chose to transmit. This property is what makes the comparison conclusive, and it is exactly what a heavier simulator could not provide at this scale.

Chapter 4 left a small number of implementation constants abstract on purpose, with the promise to declare them here. The visibility region V is realized by a finite fan of rays, one every half degree across the sensor's field of view, each stepped at half a cell and stopped by the first wall it meets; the field of view and the range are given in Section 5.2, together with the rest of the benchmark. The goal-selection rule is MinPos exactly as implemented, with two declared deviations from the original formulation of Bautin et al. [28]. In MinPos a robot commits to the frontier that the fewest teammates are closer to than itself. The first deviation concerns how that comparison is made, since the ranking uses straight-line distance rather than path length, and it counts only the teammates the robot can currently reach through the chain of open links. The second concerns ties, which are broken by the length of the shortest path over known free cells. Frontier detection is incremental, with a local pass every step and a full pass every twelve steps; frontier cells are grouped into clusters of at least five, and goals snap to the free cell nearest to the cluster center. Paths are planned with the A* algorithm over known free cells on the eight-connected grid, and the robot advances one waypoint per time step. The contact trail keeps observed poses at a spacing of at least two cells, and a transmitted cell carries its two-bit value on the wire, as priced in Section 4.2. We added two fallbacks to prevent a robot from stalling. A goal that proves unreachable is banned from re-selection, and when the coordinated rule returns no goal at all, the robot falls back to the nearest frontier. The experiments of this chapter compare arms that differ only in the sharing policy. A choice that is not declared in this section does not exist in the simulator, and every arm shares these constants unchanged, so no comparison can be confounded by them.

One component of the simulator contains randomness, the channel, and it is also the one component the model itself can test. The window model of Section 4.2 is a two-state Markov chain, so two of its properties can be computed exactly from the formulas of that section, before running any code: how long a pair waits for a session on average, and what fraction of the time it spends linked. A pair of

robots held at distance d must wait $1/p(d)$ steps on average before a session opens, and over a long horizon it must be linked a fraction $\phi(d)$ of the time, the two quantities derived there for the channel. Both quantities were then measured on the implementation, with robot pairs held at fixed distances, and they match the formulas. The channel the campaigns run on is therefore the channel that Section 4.2 defined, not an approximation of it.

The abstraction level of the simulator is also the abstraction level of the literature it compares against. The benchmarks of the map-prediction literature reviewed in Section 3.2, among them MapEx and the P^2 Explore system of Song et al. [39], evaluate on two-dimensional occupancy worlds of exactly this kind, and frontier-based exploration was developed on occupancy grids from the start (Chapter 2). The ACL map of the next section is the laboratory where the real system runs, so the simulator explores the same building that produced the run of Figure 2.1.

Figures 5.1 and 5.2 show the simulator in action, on the smallest and on the largest map of the benchmark. Their visual conventions, described once here, are shared by every simulation figure of this chapter. The full floor plan sits in the background in a light shade, for orientation. Explored free space is white, observed walls are black, and unexplored ground is shaded. A robot is a filled dot with a notch marking its heading, and the tinted disc around it is its sensor range. The orange dots are the frontier cells, the only places where the map can still grow. The thin fan of lines from a robot connects it to the reachable frontiers it is currently weighing, the thick dashed line is the planned path to the goal it committed to, with an open circle marking the goal, and the solid colored line is the trail it has traveled. The maps themselves, and the settings every mission shares, are the subject of the next section.

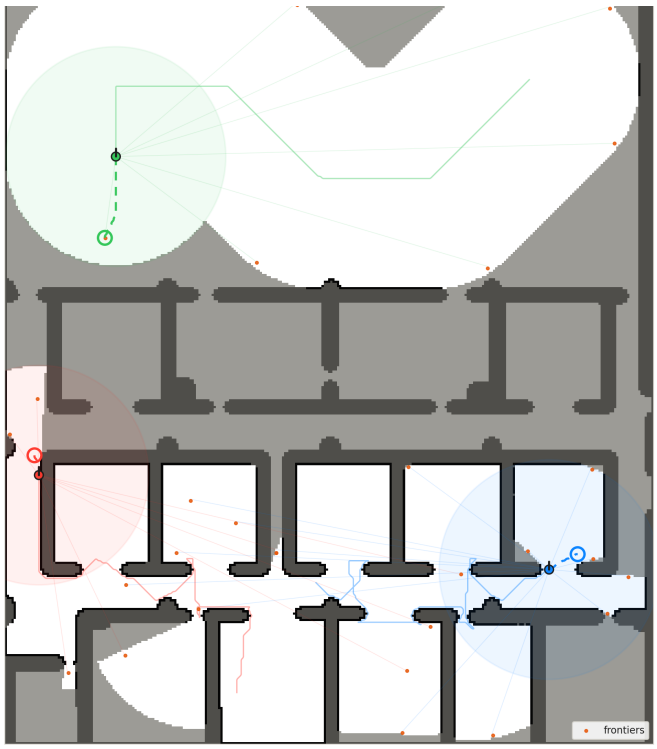

Figure 5.1: *The simulator on the ACL map.* Three robots mid-run (65 % explored) on the office floor of the MIT Aerospace Controls Laboratory, the environment of the real run of Figure 2.1. The picture reads as follows. Each robot is a small filled dot, with a notch marking its heading, and the large tinted circle around it is the reach of its sensor. Everything inside that circle and in line of sight is being observed. The small orange dots are the frontier cells, the known free cells that border unexplored ground, the only places where the map can still grow. The thin lines fanning out of a robot connect it to the reachable frontiers it is currently weighing; the thick dashed line is the path to the frontier it committed to, and the small open circle at its end marks that goal. The solid colored line behind each robot is the path it has traveled so far. The background shows the state of the map. Explored free space is white, observed walls are black, and ground still unknown to the team is dim gray, with the full floor plan in a light shade underneath for orientation. The frame serves only to illustrate the simulator and its visual conventions.

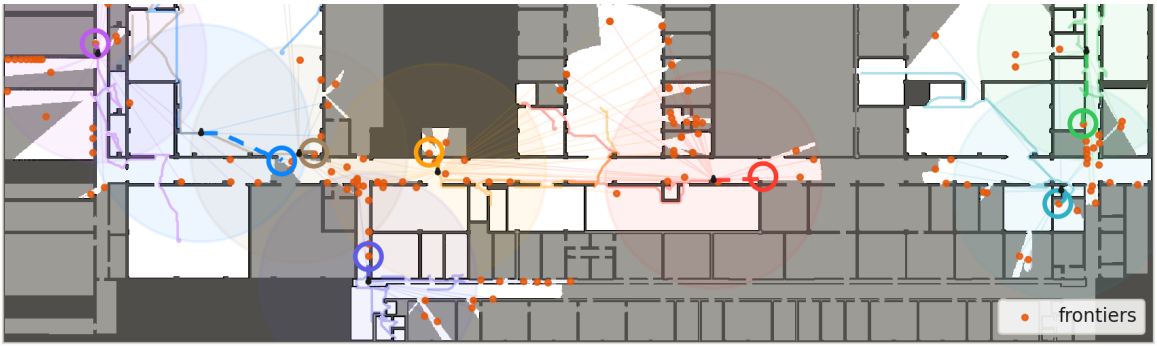

Figure 5.2: *The simulator at benchmark scale.* Eight robots mid-run on KTH-847, the largest benchmark map. The elements read as in Figure 5.1: filled dots are the robots, the tinted circles their sensor reach, the orange dots the frontier cells, the dashed lines the committed paths with an open circle at each goal, and the solid lines the traveled trails. The eight-robot frame is illustrative; the campaigns of this chapter run teams of two to seven robots.

5.2 The Benchmark

Section 5.1 described the simulator and the guarantees it provides. This section presents the seven maps of the benchmark and the settings shared by every campaign. Together with Section 5.1, it gives the reader every element the results depend on. The competing policies are the one remaining piece, and they open Section 5.3.

Six of the seven maps come from the KTH campus floor-plan dataset of Aydemir et al. [55], a collection of real buildings of the KTH campus in Stockholm, in the processed form distributed with the MapEx repository [36] and adopted by its successor PIPE [38]. We adopt this dataset for two reasons. Its maps are real buildings at campaign scale, and they are the benchmarks the map-prediction literature reviewed in Section 3.2 already evaluates on, so the dataset is widely adopted in the existing literature and the results remain comparable with it. The conclusions, moreover, are not confined to the maps the method developed on; they are checked also on maps that played no part in development, through the held-out runs described below. PIPE groups its maps into small, medium, and large tiers by size, and we keep the same buildings and the same tier names. The seventh map is the office of the MIT Aerospace Controls Laboratory, the environment where the real system runs and where the run of Figure 2.1 was recorded. It ties the benchmark to an environment the laboratory can physically deploy in.

Figure 5.3 shows the seven maps, and Table 5.1 collects their measurements. The panels of the figure are drawn at one exact metric scale, identical for every map, so the proportions between the maps are true; a map that looks twice as large as another is twice as large. The explorable cells of Table 5.1 are the coverage denominator of Section 4.1, the free cells plus the wall faces that border them, and coverage is measured against exactly this count.

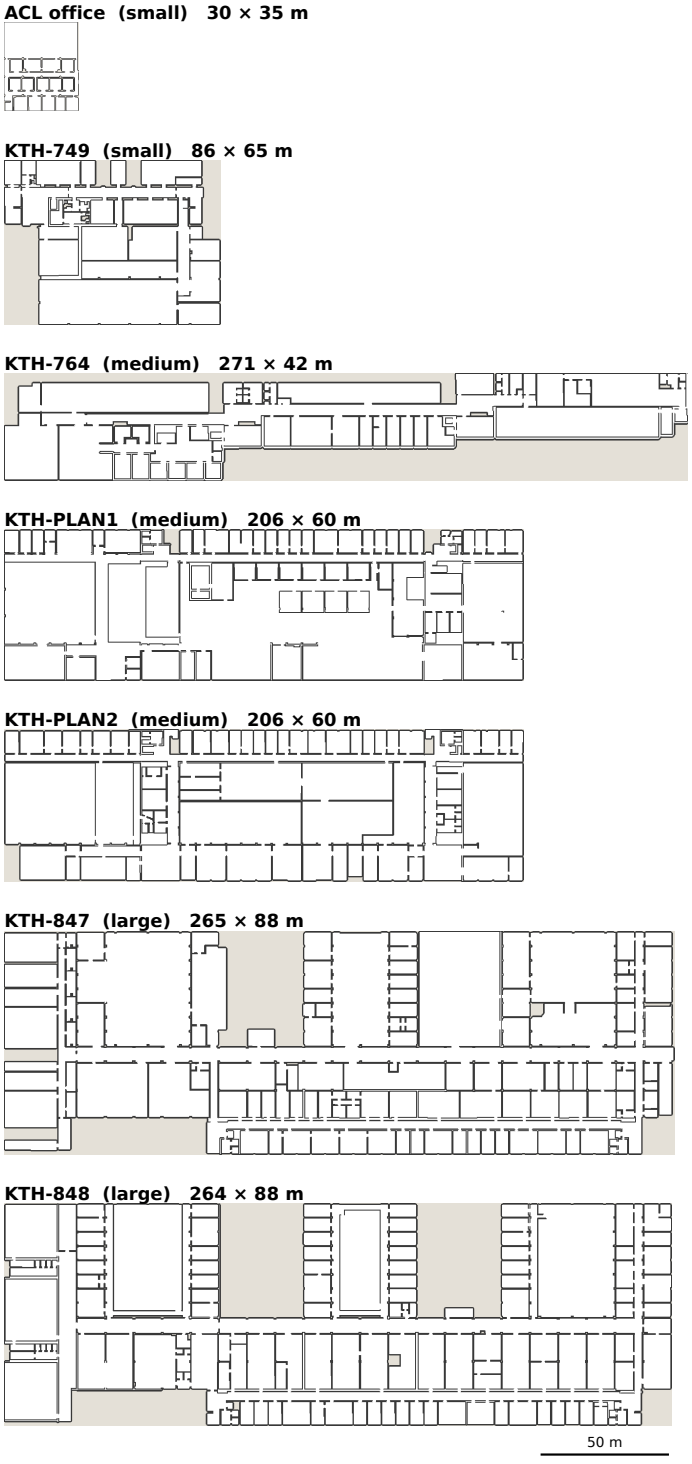

Figure 5.3: *The seven benchmark maps, grouped by size tier.* All panels are drawn at one exact metric scale, identical for every map, so the proportions between the maps are true. Free space is white, structural walls are dark, and the light fill marks blueprint voids and out-of-lot regions, both untraversable in the simulator. The bar at the bottom right measures 50 m.

Table 5.1: The seven maps of the benchmark. Explorable cells are the coverage denominator, the free cells plus the wall faces that border them. In the dataset each building is identified by a number, and the short names used in this thesis abbreviate those numbers: 749 stands for building 50052749, PLAN1 and PLAN2 are plans 1 and 2 of building 50010535, 764 is plan 1 of building 50037764, and 847 and 848 are buildings 50015847 and 50015848.

Map	Tier	Grid, $W \times H$ (cells)	Extent, $W \times H$ (m)	Free (m^2)	Explorable
ACL office	small	295×350	29.5×35.0	862	91,203
KTH-749	small	428×325	85.6×65.0	4,036	109,164
KTH-764	medium	1354×211	270.8×42.2	6,814	186,570
KTH-PLAN1	medium	1029×298	205.8×59.6	10,810	290,496
KTH-PLAN2	medium	1029×301	205.8×60.2	10,652	286,987
KTH-847	large	1327×442	265.4×88.4	18,284	490,597
KTH-848	large	1321×440	264.2×88.0	16,711	453,702

Three reasons drive the choice of these maps. First, every map is a real building rather than a synthetic maze: rooms off corridors, open halls, irregular wings. The behavior of a sharing policy depends on the structure it explores, and results on real buildings carry more weight than results on generated worlds. Second, the suite spans a factor of 21 in free area, from 862 to 18,284 m^2, so the campaigns can test how the value of communication changes with building size. Third, the shapes differ where it matters for communication. The ACL map is a single office suite. KTH-749 is a compact block of rooms. KTH-764 is the most elongated map, 271 m long and only 42 m across, so teammates drift apart along one long axis. KTH-PLAN1 and KTH-PLAN2 are two plans of the same medium building with dense room structure. KTH-847 combines the largest area with three parallel wings connected by long corridors, and KTH-848 is a building of the same family and comparable size.

Two of these maps were held out. A method develops on the maps it is tested on, and it must show that its behavior survives on a map it has never seen, so KTH-PLAN2 and KTH-848 were kept out of every sharing-policy campaign. KTH-848 was later released for a single blind run, under a protocol frozen and declared in advance (Section 5.3).

The working KTH maps are 0.2 m/cell twins of the dataset's 0.1 m/cell originals, whereas the ACL map is used at its native 0.1 m/cell. Working at 0.2 m per cell quarters the number of cells and makes campaign-scale sweeps tractable. The conversion from the original resolution was verified before use. On every map, at least 99.96 percent of the wall cells are identical to the original map, and the few doorways that the coarser grid had closed were re-opened wherever the original map shows a passage. The sensing standard of the campaigns is a range of 25 m, chosen as a conservative stand-in for the effective indoor reach of the lidar carried by the laboratory's real platforms, a Livox MID-360. On the ACL map the range is 5 m, a declared exception. That floor is about 30 m wide, and a 25 m sensor would see through most of it.

Table 5.2 collects the mission settings shared by every arm of every campaign. They realize the assumptions of Chapter 4 with the dials held constant, so that the only axes the campaigns vary are the map, the team size, and the sharing policy.

Table 5.2: The mission settings shared by every arm of every campaign. Lengths in meters apply to all maps; the channel scale λ is set in cells.

Setting	Value
Team size n	2 to 7 robots, fixed per campaign
Starting poses	dispersed at random, minimum separation 2 m; identical across arms at the same seed
Sensor	cone with a 100° field of view; range 25 m; error-free (Assumption 4.2)
Motion	one cell per time step, $\sqrt{2}$ cells on a diagonal (Assumption 4.3)
Robot bodies	robots do not obstruct one another; they are kept apart by the goal assignment and the shared map, not by a physical constraint
Channel	window model of Section 4.2; core radius 10 m; $p_0 = 0.15$, $\lambda = 80$ cells, $T_{\text{on}} = 12$ steps, identical in every campaign
Positions	exchanged at every contact, in every arm, at no cost against the budget (Assumption 4.4)
Goal selection	MinPos, as declared in Section 5.1
Seeds	30 paired seeds per configuration

Performance is mission time, the objective T_θ of Section 4.1. Time is the criterion because it is what efficiency means in exploration. Of two teams exploring the same environment, the better one is the one that finishes sooner (Chapter 1). The mission target sits in the completion band, between 95 and 100 percent of the explorable cells. The last fractions of coverage lie in a few residual pockets and cost disproportionate travel, so a threshold inside this band marks effective completion; the campaigns read the target at 95 percent, with intermediate milestones along the way. The unit is the simulation step, one iteration of the exploration loop. Its physical duration is the cell size divided by the robot speed. At 0.2 m/cell and a nominal 1 m/s, one step is 0.2 s, so a mission of 5,000 steps is about 17 minutes. The choice of speed rescales steps to seconds identically for every arm and changes no comparison, because every result in this chapter is an ordering or a ratio of step counts. Each configuration runs on 30 paired seeds, and a table entry is the median over those runs; a run that never reaches the target within its step cap counts as infinite, and the number of such censored runs is declared next to every affected entry. Thirty seeds are enough in practice. The medians converge at that count, and adding more runs does not move them. Where a mechanism needs it, duplicated discovery, the Overlap of Section 4.1, is reported as a second metric. None of the campaigns is preregistered; they are exploratory benchmarks, and every number is recomputed from the raw run logs before it enters a table. With the model, the simulator, the maps, and the settings fixed, what remains is the competition itself, and the next section introduces the compared policies and the head-to-head results.

5.3 Sharing under a Limited Channel

The model of Chapter 4 fixed what a mission is, Section 5.1 built the simulator that runs it, and Section 5.2 fixed the maps and the shared settings. This section asks the central question of the thesis. When a contact carries only a small fraction of what a robot knows, how much does the choice of that fraction matter?

The campaigns of this section fix the channel at a single budget, written in the fractional form recommended in Section 4.2. At a contact step, and in each direction, an arm holds a set of cells it is ready to transmit, its payload $P_{i\to j,t}$, and the cap admits only

$$k_{i\to j,t} \;=\; \lceil\, 0.05 \cdot |P_{i\to j,t}| \,\rceil$$

of them, each priced at the b bits per cell of Section 4.2. In plain terms, a link delivers one cell in twenty of what it would deliver if it emptied its payload at that step, and the other nineteen wait for a later contact. The cap is written as a fraction rather than as a fixed number of bits because a fraction is dimensionless, so it means the same thing on a small map and on a large one, and at the value one it reproduces the unlimited exchange exactly. Every limited arm pays the same price per transmitted cell under the same cap, so at an equal budget a difference in outcome can come only from which cells crossed the link. Five percent is one representative point of the budget axis, tight enough that the choice of what to send matters, and Section 5.6 varies it.

Four sharing policies are compared, each of them one arm in the sense of Section 5.1 and run under the settings fixed there and in Section 5.2. Two of them are references rather than candidates. The first is no communication, in which the robots exchange their positions and nothing else, and it shows what coordination achieves when no map content flows at all. The second is unlimited, which merges the two maps completely at every contact. Unlimited is the standard assumption of the coordination literature (Chapter 2) and no sharing policy can do better than it, so it serves as the upper bound of the comparison. The two candidates sit between the references, and both of them have to choose. Frontier-only transmits the cells of the sender's own current frontier, the first idea that comes to mind when a contact can carry only a little, and it asks whether a robot that shares only where its own map ends helps its teammates enough. We recall that MIRROR, the method of this thesis (Section 4.4), transmits the unsent cells in the order of what the receiver is estimated not to know yet, and that it builds that estimate from the cells it has already delivered and the positions where it has observed the teammate, so the aim costs nothing on the channel. It asks how much of the upper bound such an aimed selection recovers.

The four arms differ in three places only, and Table 5.3 states them. The payload is the set of cells a policy is willing to send, the order is the sequence in which it sends them, and the cut is the part of that sequence one contact step delivers. Every other stage of a mission, from frontier detection to goal

selection, path planning, sensing, and map fusion, is the shared setting of Section 5.1 and Table 5.2, unchanged in every arm.

Table 5.3: What each arm puts on the channel. Positions are exchanged in every arm at no cost against the budget (Assumption 4.4), and a transmitted map cell costs the same in every arm.

Policy	Payload	Order of transmission	Cut
No communication	none, positions only	none	none
Frontier-only	its own frontier cells not yet delivered, the known free cells that border unknown ground	memory order	5 % of its payload
MIRROR	the whole unsent pool $\Delta_{i\to j}$	by the key $\big(1-e(c),\ \|c-p_{\text{last}}\|^2\big)$ of Section 4.4, the receiver's estimated knowledge edge first and the nearest to its last observed pose within that, with the estimate built from records the sender already owns and therefore at no cost on the channel	5 % of its payload
Unlimited	the whole unsent pool $\Delta_{i\to j}$	any, since all of it is delivered	none

The results are read as follows. Every configuration runs on thirty paired seeds (Section 5.2), curves are medians with interquartile bands, and a run that never reaches the target within its step cap counts as infinite, with the number of such runs declared where it occurs. Progress is judged at the 95 percent target and at the intermediate milestones along the way.

Figure 5.4 shows how the missions unfold, one representative map per tier. The pattern is the same in all three panels, and it is the headline of this chapter. No communication and frontier-only advance together for the whole mission. The MIRROR curve leaves both of them behind and then stays with unlimited until the target, on five percent of the channel. Table 5.4 extends the picture to all six benchmarked maps. From the smallest map to the largest, the time MIRROR gives back grows from 365 to 6,115 steps, almost seventeen times, and the relative reduction nearly doubles, from 27 to 50 percent. As already explained in Chapter 1, the larger the map, the longer the robots explore apart, the further their private maps diverge, and the more a well-aimed channel can give back. The distance to the upper bound, meanwhile, stays small on every map. The median per-seed gap between MIRROR and unlimited lies between 0.4 and 4 percent across the six maps.

To summarize, MIRROR finishes between 27 and 50 percent sooner than a silent team. On the two large floors that is 4,975 and 6,115 steps, roughly seventeen and twenty minutes of exploration. It stays within 0.4 to 4 percent of what an unlimited link achieves, and every contact it uses carries one twentieth of a full exchange. The four arms differ in nothing but the cells they put on the wire, so the time comes from that choice alone. The larger the building, the more the choice is worth throughout the exploration.

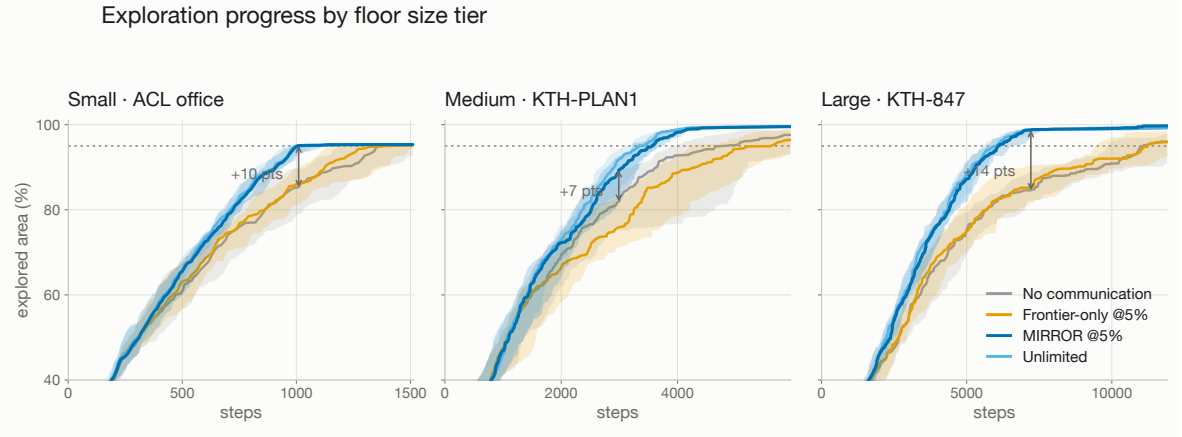

Figure 5.4: *Exploration progress by map size tier.* Coverage over time at $n = 3$, one representative map per tier. Curves are medians over 30 paired seeds with interquartile bands, a run that has reached the target carries its final value forward, and the limited arms run at the five percent budget. The dotted line is the 95 percent target, the arrow marks the widest gap between the MIRROR and the no-communication medians in coverage points, and the horizontal axis of each panel stops just past the arrival of the slowest policy.

Table 5.4 reports the finish line, and a finish line says nothing about how the advantage forms. For this reason the same runs are read again at intermediate coverage levels in Table 5.5. None of the saved time is earned early. At half coverage the four arms are nearly level, and on KTH-PLAN1 the limited arms even trail slightly. The gap then grows with the mission. Every contact gives a sharing policy one more chance to send a robot somewhere better, so the longer the team explores, the further ahead MIRROR gets, and the distance is largest at the target. This is exactly where Section 4.3 located the value of selection. Early on, every policy finds abundant frontier close by, and communication has little to redirect. Late in the mission, the remaining ground hides in a few pockets, and knowing which pockets a teammate has already paid for is worth entire trips.

One result of this section is worth pausing on, because it says something about communication that none of the other comparisons does. The ladder is built on the assumption that moving up it helps, and everywhere else in this chapter it does. Frontier-only is where that assumption breaks. In the medium panel of Figure 5.4 its curve runs below the no-communication curve for most of the mission, and Table 5.5 confirms it milestone by milestone on KTH-PLAN1, where it trails silence at 50, 70, 80, 90, and 95 percent coverage alike. In other words, a team that spent its budget sharing map content arrived later than a team that said nothing at all, which is the first indication that communication is not useful in itself.

Replaying those runs shows the cells of the sender's frontier reaching the receiver as thin slivers of free space with unknown ground all around them, and since each of those slivers borders the unknown, the

Table 5.4: MIRROR at the five percent budget across the six benchmarked maps at $n = 3$. Steps saved and the reduction are measured against no communication, on censoring-aware medians, in which a run that misses the target within its cap counts as infinite, and such runs occur only for no communication and frontier-only (at most six of thirty, on KTH-749). The gap to unlimited is the median across seeds of the per-seed relative difference; in single seeds MIRROR can finish ahead of unlimited, and the paired median never favors it.

Tier	Map	Free area (m^2)	Steps saved	Reduction	Gap to unlimited
small	ACL	862	365	$-26.6\,\%$	$+0.4\,\%$
small	KTH-749	4,036	1,190	$-39.1\,\%$	$+2.4\,\%$
medium	KTH-764	6,814	1,405	$-27.9\,\%$	$+1.9\,\%$
medium	KTH-PLAN1	10,810	1,400	$-28.2\,\%$	$+1.0\,\%$
large	KTH-847	18,284	4,975	$-45.0\,\%$	$+4.0\,\%$
large	KTH-848 (held-out)	16,711	6,115	$-50.2\,\%$	$+2.1\,\%$

receiver reads every one of them as a new frontier. Its goal selection has no way of telling them apart from the frontiers its own sensor produced, so robots are sent off to distant slivers that a teammate has usually finished with by the time they arrive, while ground that is still unexplored a few meters away keeps waiting. The message carried something that pointed the receiver in the wrong direction, which is the second sense in which communication can fail to be useful, not by being too small but by being misleading.

Across the six maps the paired per-seed comparison against no communication always contains zero, and the same holds at every sensor range of Section 5.5, so we do not claim that sharing the frontier is reliably slower than saying nothing. What is certain is that at the budget where MIRROR gives back between 27 and 50 percent of the mission, the frontier gives back nothing at all, and wherever it does cost time it costs it through the mechanism just described.

It would be natural to blame the budget, so we removed it and let the arm send its whole frontier at every contact. Almost nothing changed. The mission went from 5,620 steps at five percent to 5,060 with no cap at all, which the seeds do not separate from zero, while the traffic rose only from 1.4 to 3.1 Mbit against the 34 an unlimited exchange moves here. Sending more of the wrong cells does not help, because importance decided at the sender does not transfer to the receiver.

Figure 5.5 says where the saved time comes from. The quantity it plots is the duplicated discovery of Section 4.1, read as a share of the sensing effort. The simulator records, for every robot, the set of cells that robot determined with its own sensor during the run. Summing those counts over the team counts a cell once for every robot that sensed it, while the number of distinct cells the team sensed counts it once, so the ratio of the two is $1 + \mathrm{Overlap}$. The share of first-hand sensing that landed on ground a teammate had already covered is therefore $\mathrm{Overlap}/(1 + \mathrm{Overlap})$. In other words, it is the fraction of the team's sensing effort that bought the team nothing. The value is read at the end of each run.

Table 5.5: Median steps to intermediate coverage milestones at $n = 3$ on the three tier representatives (30 paired seeds; censored counts in parentheses, counted as infinite in the median). The last row of each block gives the share of the no-communication time that MIRROR saves at each milestone. The arms differ little at half coverage; the gap opens in the second half of the mission and is widest at the target. No communication and unlimited are the floor and the ceiling of the comparison rather than competitors, so no entry is marked as a winner.

	Coverage milestone				
Policy	50 %	70 %	80 %	90 %	95 %
ACL (small)					
No communication	320	640	865	1,175	1,370
Frontier-only	310	635	865	1,130	1,375
MIRROR	320	570	730	915	1,005
Unlimited	315	560	715	910	1,000
time saved vs no comm.	*0 %*	*11 %*	*16 %*	*22 %*	*27 %*
KTH-PLAN1 (medium)					
No communication	1,070	2,125	2,800	3,580	4,965 (1)
Frontier-only	1,100	2,460	3,290	4,315	5,620 (1)
MIRROR	1,110	1,865	2,510	3,040	3,565
Unlimited	1,090	1,830	2,375	2,895	3,390
time saved vs no comm.	*−4 %*	*12 %*	*10 %*	*15 %*	*28 %*
KTH-847 (large)					
No communication	2,655	4,360	5,665	9,625	11,065
Frontier-only	2,650	4,180	5,560	8,790	11,275
MIRROR	2,390	3,525	4,400	5,280	6,090
Unlimited	2,230	3,365	4,290	5,180	5,975
time saved vs no comm.	*10 %*	*19 %*	*22 %*	*45 %*	*45 %*

Without map exchange, between 44 and 64 percent of the team's sensing is duplicated. Frontier-only barely reduces the share, while MIRROR removes nearly all of it and matches unlimited, on every map.

The same effect can be followed directly on the map. Figure 5.6 shows one mission on KTH-PLAN1, started from identical robot poses and then run under three sharing policies, with one row per policy and one column per instant of the mission. At the first instant the three teams have covered almost the same ground, which is what the milestones already told us. In the following instants the region explored only by the MIRROR team, drawn in blue, keeps growing, and by the last instant that team is finishing rooms that the other two teams have not reached yet.

All the results above were measured on maps the method was developed on, which is why KTH-848 was kept aside (Section 5.2). It was released only once the protocol had been frozen, and it was run blind under the settings declared for its twin KTH-847. The outcome is the last row of Table 5.4. MIRROR returned half of the mission time and finished two percent from the upper bound, which is what KTH-847 had already given. The conclusions of this section therefore hold on a map the method

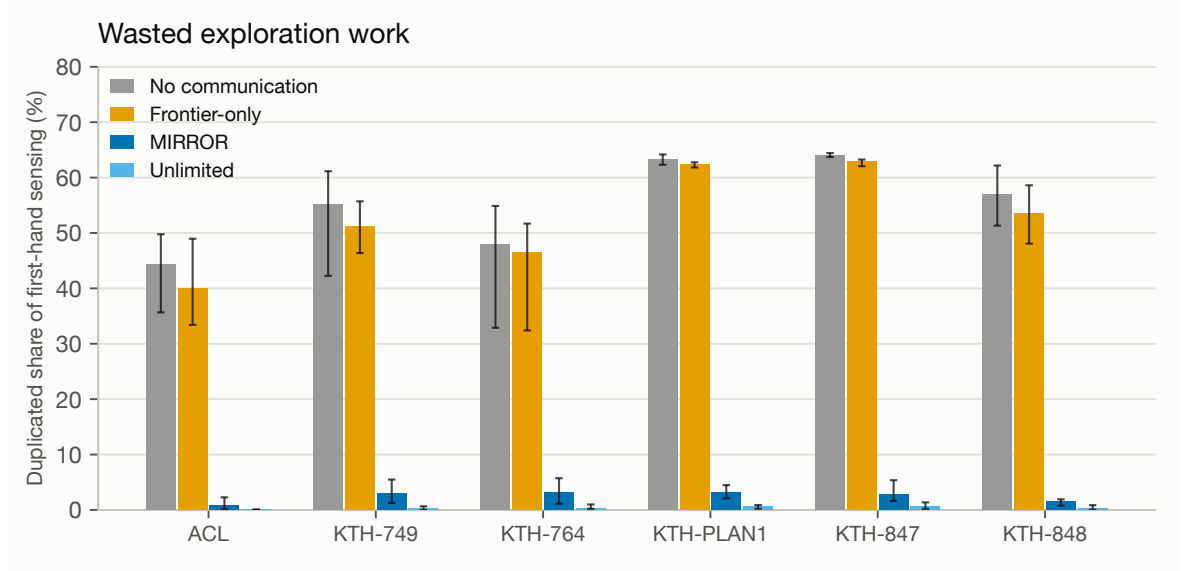

Figure 5.5: *Wasted exploration work.* The share of the team's first-hand sensing that re-sensed ground another robot had already covered, per map at $n = 3$. Bars are medians over 30 paired seeds, and the error bars span the interquartile range. Without map exchange, between 44 and 64 percent of the sensing effort is duplicated, and frontier-only barely reduces the share, while MIRROR removes nearly all of it and matches unlimited.

had never seen. They rest, however, on a single team size, and the next section varies it.

5.4 Scaling with Team Size

The results so far were measured with three robots. A reader is entitled to ask whether they survive a change of team size, and there is a reason to doubt it. Adding robots is itself a way of exploring faster. A larger team also meets more often, so more map content crosses the link even under the same budget, and one may expect the choice of what to send to matter less and less until it stops mattering at all. This section puts that expectation to the test.

The sweep runs on the two carrying maps of the benchmark, from two to seven robots, always with the four arms of Section 5.3, the same five percent budget, and thirty paired seeds per configuration. Everything else is held at the settings of Table 5.2. Mission time is counted in simulation steps, the unit fixed earlier in this thesis. One step is one iteration of the exploration loop, and a robot advances one cell in it, so the step is what converts simulated progress into real time. At the 0.2 m cells of the KTH maps and a nominal speed of one meter per second, a step lasts 0.2 s, and a mission of 5,000 steps is about seventeen minutes of real exploration.

One property of these campaigns should be stated before the numbers are read. The number of steps a

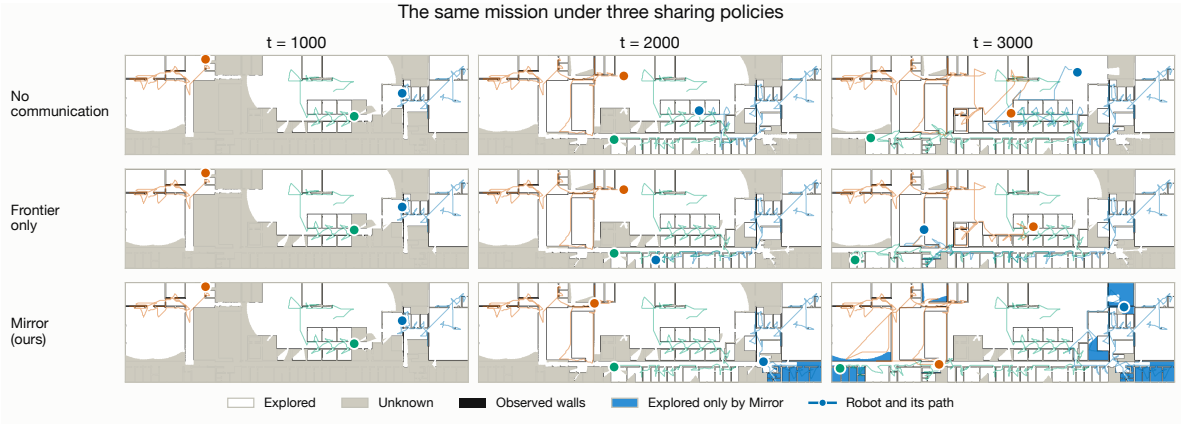

Figure 5.6: *The same mission under three sharing policies.* One mission on KTH-PLAN1 at $n = 3$, identical starting poses, shown at three instants. White is ground the team has explored, the light shade is still unknown, black is the observed walls, and the blue fill marks ground that only the MIRROR team has explored at that instant. The advantage is invisible at $t = 1000$ and grows through the mission.

run was allowed to last was set separately for each team size, since larger teams finish sooner. With the shorter allowance, the slowest arms sometimes failed to reach the target before the run was stopped. A run of this kind is treated as never finishing, and since it happens to fewer than half of the runs, the median reported for that arm is still a time that was actually measured.

Figure 5.7 and Table 5.6 report the simulation results, and Figure 5.8 follows the missions themselves, one panel per team size. Every arm becomes faster as robots are added, and the gain per robot shrinks. On KTH-PLAN1 the silent team goes from 7,575 steps at two robots to 2,445 at seven, which is 3.1 times faster with 3.5 times the robots, and MIRROR improves in the same sublinear way. Adding robots is therefore useful, but it does not replace the choice of what to send, because the ordering of the four arms never changes. At every team size on both maps, MIRROR is faster than frontier-only and than no communication, and it sits on the unlimited bound. The saving MIRROR returns does not fade as the team grows either. It stays between 28 and 41 percent on KTH-PLAN1 and between 30 and 54 percent on KTH-847, with no trend in either direction that the data can support.

The same numbers answer a more practical question when they are read horizontally rather than vertically. Instead of asking how fast a team of a given size finishes, we ask how large a silent team has to be in order to match a smaller team that shares. On KTH-PLAN1, three robots at the five percent budget reach the target in 3,565 steps, and five robots that never exchange map content need 3,645. On KTH-847 the channel is worth more. Three sharing robots finish in 6,090 steps against 6,155 for six silent ones. Choosing well what to put on a narrow channel therefore buys the same thing that robots buy, and on the large map it buys about three of them. This is the answer to the objection that opened the section. A team does not outgrow the need to choose what to transmit, and the choice is worth

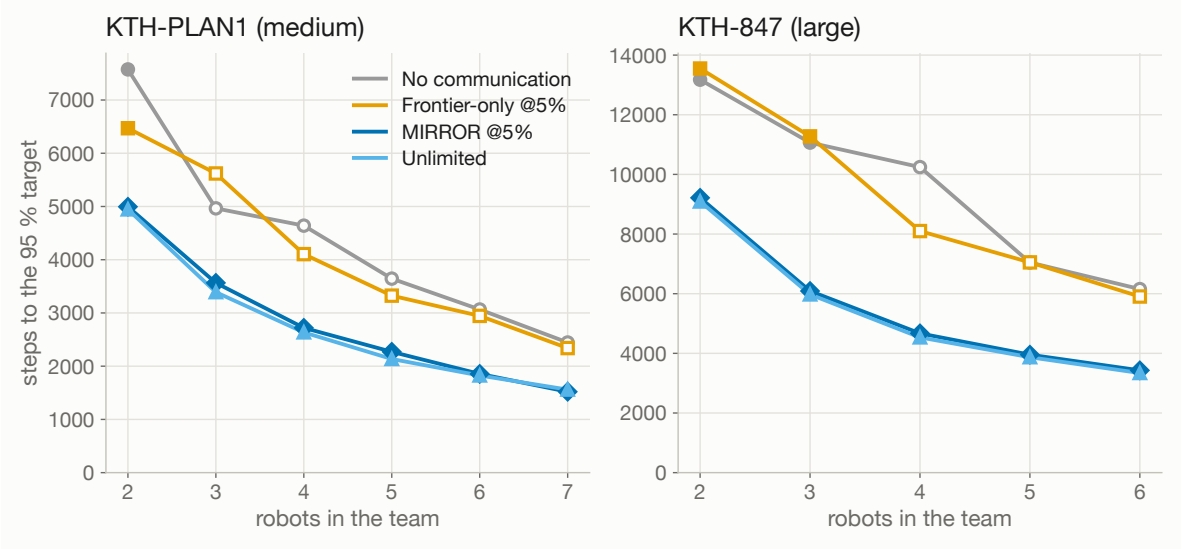

Figure 5.7: *Mission time against team size.* Median steps to the 95 percent target over 30 paired seeds, per team size, on the two carrying maps. A run that misses the target within its step cap counts as infinite in the ranking, and a marker is drawn hollow when the campaign behind it contains such runs, which happens only for the two slowest arms at the larger team sizes. The KTH-847 panel stops at six robots, because this figure is drawn from the raw run logs and the log of the seven-robot campaign was not available.

roughly as much as doubling the fleet.

The mechanism behind that number is the one Figure 5.5 already measured at three robots, and it grows with the team. On KTH-PLAN1 the share of first-hand sensing that a silent team spends on ground a teammate had already covered rises from 43 percent at two robots to 67 percent at six. This is what one should expect. In fact, the more robots explore the same building without exchanging maps, the more often two of them pay for the same ground. The waste that a sharing policy removes therefore becomes larger, not smaller, as robots are added, which is precisely why its advantage does not thin out.

One question this sweep cannot settle is whether some team size benefits from communication more than the others. The measurements do point somewhere. On KTH-847 the time MIRROR returns rises from 30 percent at two robots to 54 percent at four, and on KTH-PLAN1 the largest value, 41 percent, falls at four robots as well. Read on its own, this suggests that for a given map and a given sharing policy there may be a team size at which the exchange is worth the most, a size at which the robots are numerous enough to keep meeting and still far enough apart to carry knowledge the others lack.

However, the same comparison read before the end of the mission does not reproduce that maximum. Taken at 90 percent coverage, seed by seed, the largest value on KTH-PLAN1 falls at six robots rather than four, and the confidence intervals of the neighboring team sizes overlap; on KTH-847 the value

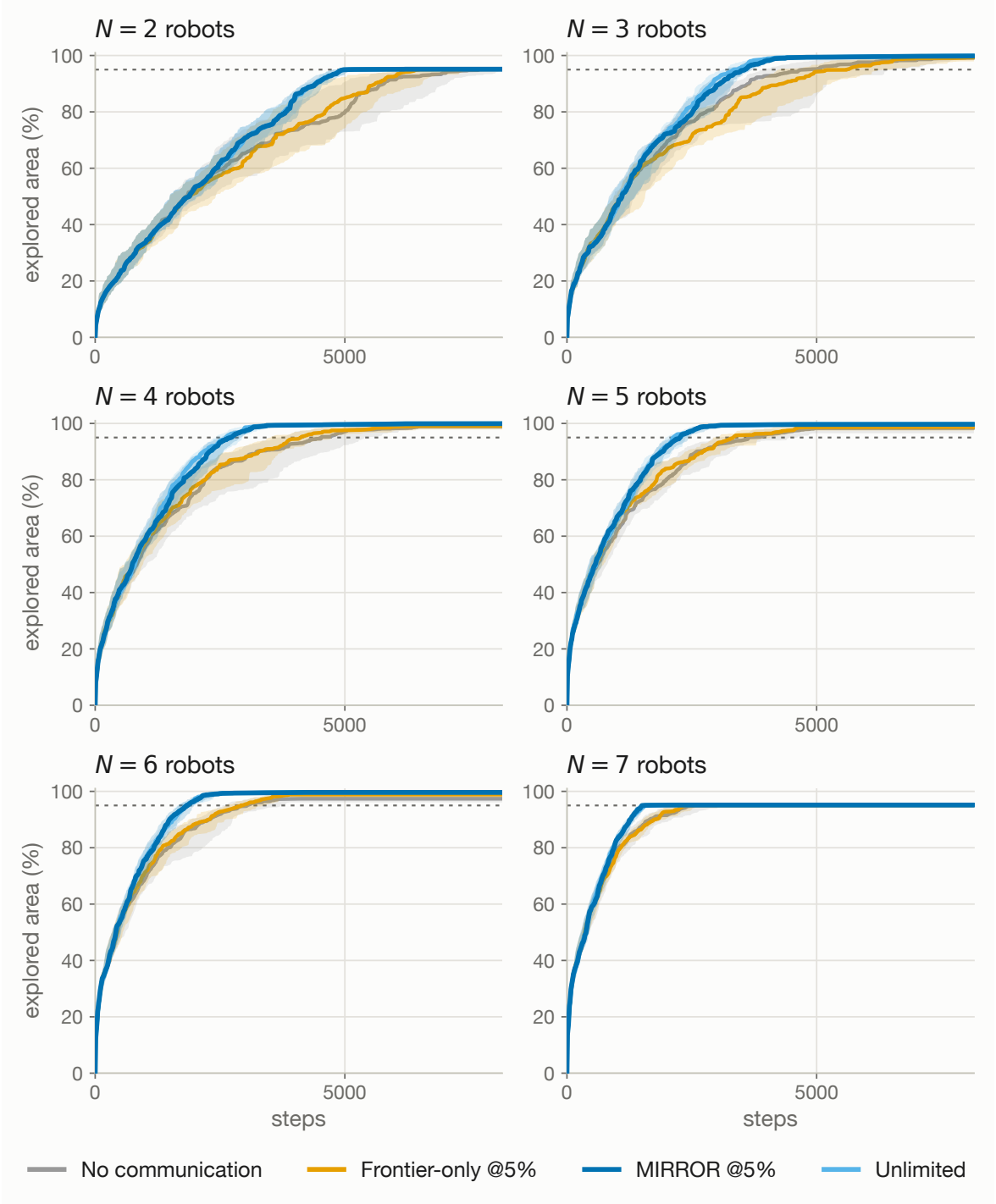

Figure 5.8: *Exploration progress at every team size.* Coverage over time on KTH-PLAN1, one panel per team size, medians over 30 paired seeds with a band covering the middle half of the runs, and the same conventions as Figure 5.4. Adding robots compresses every mission towards the left, and the separation between MIRROR and the two lower arms opens in the second half of the mission at every team size.

Table 5.6: Median steps to the 95 percent target per team size, over 30 paired seeds. The last row of each block is the share of the no-communication time that MIRROR returns. Every entry was recomputed from the raw run logs, with one exception. The KTH-847 column at seven robots is quoted from the campaign record, because the raw log of that campaign was not available for re-derivation.

	Team size					
Policy	$n = 2$	$n = 3$	$n = 4$	$n = 5$	$n = 6$	$n = 7$
KTH-PLAN1 (medium)						
No communication	7,575	4,965	4,640	3,645	3,065	2,445
Frontier-only	6,470	5,620	4,105	3,325	2,945	2,340
MIRROR	4,995	3,565	2,720	2,270	1,855	1,520
Unlimited	4,945	3,390	2,635	2,135	1,825	1,560
time returned by MIRROR	*34 %*	*28 %*	*41 %*	*38 %*	*39 %*	*38 %*
KTH-847 (large)						
No communication	13,175	11,065	10,245	7,045	6,155	5,490
Frontier-only	13,550	11,275	8,100	7,050	5,905	5,545
MIRROR	9,215	6,090	4,670	3,955	3,430	2,910
Unlimited	9,090	5,975	4,535	3,870	3,340	2,865
time returned by MIRROR	*30 %*	*45 %*	*54 %*	*44 %*	*44 %*	*47 %*

decreases steadily as robots are added. What appears at the target may therefore belong to the last phase of a mission rather than to the team size. Adding robots keeps improving the mission on both maps. Whether a true optimum exists, and what would fix its position, is left as a question for future work (Chapter 6).

Team size, in short, changes how long a mission takes, and it leaves the answer to what a robot should send unchanged. At every size from two to seven robots, and on both maps, MIRROR returns between 28 and 54 percent of the silent mission and is never more than six percent behind unlimited communication, so the worth of choosing well does not thin out as the fleet grows, and a narrow channel used properly keeps buying what three more robots would buy. Two settings have been held fixed throughout, the sensor the robots carry and the width of the channel, and the next sections vary them.

5.5 Further Observations

The results so far were measured with one sensor and reported one quantity, the time a mission takes. This section adds the two measurements that complete them. The first varies the sensor range and asks how much of the advantage depends on it. The second measures what the sender pays in computation to decide the order in which its cells travel. Both are taken at the five percent budget used so far, a value that Section 5.6 then examines in its own right.

Section 5.3 measured the four policies at one budget, and Section 5.4 varied the size of the team. Both

campaigns held the rest of the mission fixed, including the sensor the robots carry. How much of the advantage reported so far belongs to that particular choice?

The sensor range is the first setting to vary, because it is the quantity the channel competes against. A robot determines a cell in one of two ways, by traveling until its own sensor covers it, or by receiving it from a teammate that already paid that travel (Section 2.2), and the sensor range fixes the price of the first way. At a long range one sweep determines a wide area, so a team that never communicates still advances quickly and loses little by staying silent. At a short range the same area costs many more sweeps and much more travel, and every cell a robot cannot reach with its own sensor must therefore arrive over the channel or be paid for in meters. The scarcer the sensing, the more a sharing policy should be worth.

The campaign sweeps the sensor range from 5 to 40 m on KTH-PLAN1 at three robots, with the four arms of Section 5.3 and the settings of Table 5.2 held at their standard values, so that the range is the only quantity that differs between two points of the sweep. The points from 5 to 25 m run 60 paired seeds, and those from 30 to 40 m run 30. Confidence intervals on paired differences are obtained by resampling the seeds. The sweep varies one map at one team size, so it establishes what the sensing assumption is worth on this building rather than a law that holds across the benchmark.

Figure 5.9 reports the result, and the quantity to read is the vertical distance between MIRROR and no communication, which is the mission time the policy returns to the team. That distance grows as the sensor range shrinks. At the standard 25 m range MIRROR reaches the target 1,520 steps before the silent team, with a 95 percent confidence interval of $[990, 2{,}280]$ on the paired per-seed difference. At 5 m the same difference is 5,060 steps, $[4{,}070, 5{,}610]$. The advantage therefore more than triples, and since the two intervals do not overlap, the growth cannot be attributed to the choice of seeds. Measured as a share of the silent mission, it rises from 30 to 44 percent. The shaded band reports the same effect for communication in general, because it is the distance between silence and the unlimited arm, and MIRROR covers nearly all of that distance at every range in the sweep.

We understand from this that duplicated work, the Overlap of Section 4.1, is the mechanism behind the effect. At a 5 m range the silent team senses 1.21 extra cells for every cell it adds to the team map, so it pays the sensing cost of everything it discovers a second time over, whereas under MIRROR the same quantity is 0.00. In other words, a robot with a long sensor range determines the ground ahead of it before any message could have warned it, while a robot that senses only its immediate surroundings has no way of learning that a teammate has already covered that ground unless it is told.

The same sweep also speaks to the size of the budget. In Figure 5.9 the five percent arm follows the unlimited one at every range tested, from a sensor that sees 5 m to one that sees 40, and the widest separation anywhere is 145 steps. What a team needs on the link does not seem to scale with how much

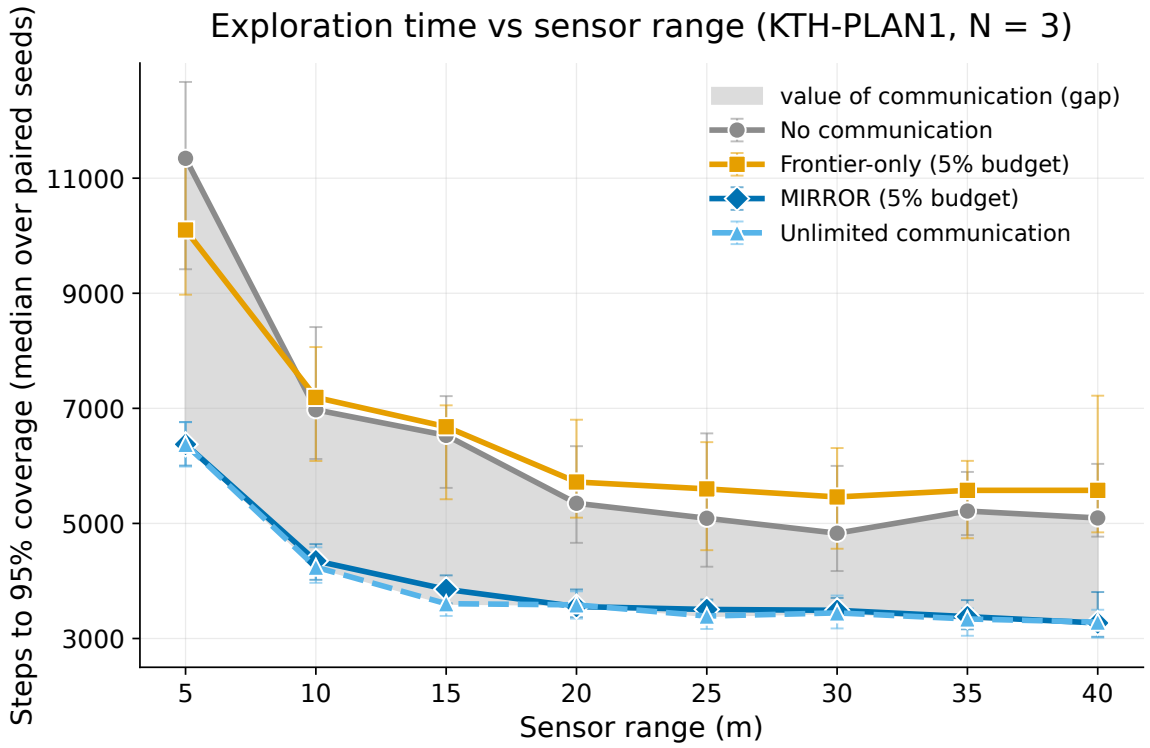

Figure 5.9: *Mission time against sensor range.* Steps to the 95 percent target on KTH-PLAN1 at $n = 3$, medians over paired seeds, with error bars covering the middle half of the runs. The shaded band is the distance between no communication and unlimited communication, the value of communication itself. The band widens as the sensor shrinks, while MIRROR at the five percent budget matches the unlimited arm at every sensor range. Points from 5 to 25 m are medians over 60 paired seeds, and those from 30 to 40 m over 30.

Table 5.7: Team travel to the 95 percent target on KTH-PLAN1 at $n = 3$, in meters at 0.2 m per cell, medians over the paired seeds of the sensor sweep. The last column is what MIRROR saves against no communication. The rows from 5 to 20 m are medians over 60 seeds and the rest over 30. The 25 m row is taken from the runs that stop at the target, since the campaign that ran to a fixed horizon keeps accumulating travel after the target is reached and its distances are not comparable with the others.

Sensor	No comm.	Frontier-only	**MIRROR**	Unlimited	Saving
5 m	7,724 m	6,856 m	4,317 m	4,306 m	−44.1 %
10 m	4,732 m	4,885 m	2,924 m	2,856 m	−38.2 %
15 m	4,396 m	4,493 m	2,594 m	2,418 m	−41.0 %
20 m	3,593 m	3,847 m	2,386 m	2,403 m	−33.6 %
25 m	3,413 m	3,690 m	2,323 m	2,258 m	−31.9 %
30 m	3,244 m	3,641 m	2,327 m	2,310 m	−28.2 %
35 m	3,530 m	3,743 m	2,264 m	2,223 m	−35.9 %
40 m	3,425 m	3,743 m	2,183 m	2,199 m	−36.3 %

its robots see for themselves. It seems to scale with how often a robot would otherwise walk into ground a teammate has already covered, and on this floor preventing that appears to cost only the few cells that change where a robot goes next.

Mission time converts directly into distance driven, and Table 5.7 reads the same sweep in meters. The saving is the same size in both units, which is what it should be, since a robot advances about one fifth of a meter per step whatever policy it runs. What the table adds is the physical scale. At the standard 25 m sensor MIRROR saves the team about 1.1 km of driving, and at 5 m about 3.4 km, on a building 206 m long. For a platform whose endurance is set by its battery rather than by its clock, that distance is the quantity that matters.

At every range in the sweep, MIRROR at the five percent budget and the unlimited arm finish within a few percent of one another. The two are indistinguishable at the hardest point, where the paired difference at 5 m is 30 steps in MIRROR's favor and both arms return the same 44 percent of the silent mission. The widest separation anywhere in the sweep is 145 steps at 15 m, about 4 percent. Five percent of the channel therefore buys nearly all of the value of communication across the whole range of sensing tested, and not only at the range the earlier campaigns assumed.

At every range from 10 to 40 m the median mission of frontier-only is longer than that of a silent team, as it is in Section 5.3, whereas at 5 m it is 11 percent shorter. The paired comparison separates the two arms at no range in the sweep, so what follows explains the direction rather than a measured advantage. The phantom-frontier failure requires the transmitted ring to be a thin sliver of the sender's map landing far inside the receiver's unknown region. A 5 m range keeps the sender's known region small, so its rim is a large fraction of that region, and the ring therefore arrives adjacent to what the receiver already holds instead of far beyond it.

The right end of the sweep is not settled. From 30 m outward the advantage of MIRROR stops falling and rises again, from 1,240 steps at 30 m to 1,790 at 40 m, and the duplicated work of the silent team rises with it, from 0.78 to 1.05. A 40 m sensor reaches most of the way across a building 60 m wide, so two silent robots sweep much of the same ground, and communication recovers value for a second reason. The reading is coherent, but the three right-hand points carry half the seeds of the rest of the sweep, and we do not claim the effect until they are measured at the same count.

The second measurement of this section is the price of the decision itself. MIRROR puts the same number of bits on the link as any other policy at the same budget (Section 4.4). What it does spend is computation, because the sender has to decide the order before it can send anything. The remainder of this section measures what that decision costs.

The measurement is a direct one. The four policies ran the same missions on KTH-PLAN1 with three robots, three missions each, under the configuration of Table 5.2, and a timer recorded how long every stage of the exploration loop took. Since the four policies differ inside a single stage, the exchange, the timer separates what the sharing decision costs from what the rest of the mission costs. The runs were executed one at a time, so that no other run could disturb the timings; the durations recorded during the large parallel campaigns are not usable for this purpose, and Table 5.8 replaces them.

Table 5.8: Computational cost per simulation step on KTH-PLAN1 at $n = 3$, medians over three full missions per policy, run one process at a time. The exchange is the only stage in which the policies differ. The per-exchange column divides that stage by the number of directional exchanges it served, one sender preparing one message for one teammate at one contact step. No communication is the reference row, since it enters the exchange stage at every contact and leaves it without preparing anything.

Policy	Step total	Exchange stage	Per exchange	Overhead
No communication	35.7 ms	0.00 ms	0.00 ms	reference
Frontier-only	36.7 ms	0.90 ms	1.41 ms	+2.6 %
MIRROR	42.1 ms	5.38 ms	9.87 ms	+17.8 %
Unlimited	38.1 ms	0.44 ms	0.83 ms	+6.5 %

Most of a simulation step is spent sensing. Of the 37.6 ms an average step costs, the sensor model takes 31.7, the frontier update 2.0, and goal selection 2.0, while path planning and the channel together account for about 0.3. A sharing policy therefore works inside the remaining sixth of the step, and inside that sixth the three policies separate clearly. Preparing one message costs unlimited communication 0.83 ms, because it only copies the cells it has not yet delivered. Frontier-only costs 1.41 ms, since it derives its frontier as well. MIRROR costs 9.87 ms, seven times more than frontier-only and twelve times more than unlimited.

The overhead column needs one word of care, because it is not the exchange alone. Unlimited communication spends only 0.44 ms per step inside the exchange, yet its steps are 2.3 ms longer than

those of a silent team. The reason is that a robot which receives map content ends up holding a larger map, and both sensing and goal selection grow with the size of the map a robot holds. Part of the cost of communicating is therefore paid by any policy that communicates at all, whatever it sends. Measured against unlimited communication instead of against silence, the price of aiming the messages is 4.0 ms per step, about 11 percent.

The difference arises because MIRROR is the only policy that reasons about the receiver, and the version measured here rebuilds its estimate of the receiver's map from scratch at every contact. It copies the delivered log, paints a sensor disc at every teammate pose it has ever observed, finds the edge of the result, and sorts the pool against that edge. Every one of those operations passes over the whole grid, so the cost grows with the size of the map, and the list of observed poses gets longer as the mission goes on, so the cost grows with time as well. The three missions show the second effect directly, since preparing one message rises from 8.7 to 13.1 ms as the share of time the robots spend in contact rises from 0.24 to 0.43.

Two qualifications push in the same direction. The estimate does not need to be rebuilt at every contact, because the delivered log grows only by what was just sent and a disc is added only when the teammate is observed at a new pose, so an incremental implementation would amortize nearly all of this cost. The number in the table is therefore an upper bound on what the method requires rather than a property of the method. The engine is also unoptimized Python running on a laptop, so the absolute milliseconds carry no meaning outside this table. What carries meaning is the comparison between the four policies, measured under identical conditions on the same missions the rest of this chapter reports. Both measurements above were taken at the five percent budget, and that budget is the last setting of the chapter still to be examined.

5.6 The Budget Axis

Section 5.3 compared the policies at a single budget, five percent of a full exchange, and Section 5.5 varied the sensor while holding that budget fixed. This section varies the budget itself, from one cell in five thousand to one in twenty. Two questions follow. How much of what communication can give does a given budget actually buy? And is five percent a property of the method, or only of the floor it was chosen on?

One further arm is needed to answer them, and it is not a policy but a reference. The oracle sends first the cells that lie on the receiver's true boundary, which it can only do by reading the receiver's map, so no robot could ever run it. That is precisely why it is useful. It marks the $\eta = 1$ ordering of Section 4.3, the best that any policy could do with the same bits, and it turns the comparison into a measure of how much room MIRROR still leaves on the table.

One point has to be clear before the curves are read. The budget caps what a single contact may carry, not what a mission spends, since a cell that does not fit in one message waits for the next contact. A tight cap therefore delays traffic rather than removing it, and on KTH-PLAN1 the cap grows by a factor of two hundred and fifty across the sweep while the data actually delivered grows only by a factor of thirteen. For this reason performance is read against the data delivered rather than against the cap in Figure 5.10, where each marker is one budget and moving to the right means paying more. That data is counted inside the run in our simulation. Every cell that crosses the link is charged the fixed price of Section 4.2, and the total is that price times the cells actually delivered, so the axis carries the traffic the mission generated rather than an estimate of it.

The shape of the MIRROR curve is the first answer. Half a percent already returns most of what communication can give on this floor, and what comes after is paid at a much worse rate, since going from half a percent to five costs sixty-four percent more data and returns seven percent less time. That last gain is small, yet it is real, because the paired comparison supports it (median -370 steps, sign test $p = 0.043$). In the other direction the budget stops paying entirely, and a tenth of a percent and below cannot be separated from silence.

The oracle answers the harder question, whether MIRROR is merely better than naive sharing or close to the best possible use of a constrained channel. It is not yet close. At two tenths of a percent the oracle already finishes sooner than MIRROR at half a percent, on 21 of the 30 seeds, and it does so on a little over half the data, on 27 of them. From half a percent upward it recovers the performance of unlimited map exchange in full, matching it in mission time while spending less data on every one of the thirty seeds. An ordering that knew the receiver exactly would therefore buy the entire ceiling for a little over half the traffic, and the distance between the two curves is the part of that value which MIRROR's free estimate does not yet capture.

This is where the question behind the whole axis can be answered. Why should a twentieth of what a robot holds be enough? The reason is not that maps are small, since they are not, and it is not that the channel is generous. It is that most of a map cannot act. Lemma 4.2 settles it by the definitions alone. A delivered cell changes what the receiver does only if it lands on the edge of what that receiver already knows, and that edge is a rim one cell thick around a region that grows like an area. Whatever the budget, the larger part of any payload arrives inert. A message aimed at that rim would then carry most of what the whole pool would have carried, and this is the reading the campaigns of this chapter support.

The oracle curve of Figure 5.10 is the same statement in measured form, since it sends boundary cells first by construction. From half a percent of a payload it already matches the unlimited link, which points to the value of a full exchange fitting in a small part of one, provided the cells are the right ones. Figure 5.11 adds that the same saturation appears on every floor tested, and the milestones of Table 5.5

suggest why the demand stays modest, since the decisions worth changing are few and arrive late in a mission. The frontier-only control of Section 5.3 carries the same lesson in the opposite direction, since removing its cap entirely does not make it useful.

The same campaigns also ran the unaimed order, which carries the identical payload in memory order and therefore differs from MIRROR in nothing but the sequence. At half a percent the two separate clearly on the large floors, by 2,020 steps on KTH-847 and 1,310 on KTH-848, and on nearly every seed, while at five percent no floor separates them. The order therefore decides the mission where the cut binds, and stops mattering once it does not, which is the claim of Chapter 1 in its measured form.

Little, in this section, means little per contact. Across a mission the cut defers rather than cancels, so a team held at five percent still moves nearly all the data an unlimited link moves, which is the distance between the cap and the delivered data noted at the start of this section. What is small is not the traffic. It is the part of it that has to arrive at the right moment, and that part is what an ordering decides.

The same campaigns also tested a simpler version of MIRROR, one that aims at the teammate's last observed position and keeps no replica of its map. It performs as well as MIRROR on every floor, with paired differences that never separate the two ($p = 0.10$ to 1.00), and on KTH-749 it is the faster of the two at the five percent budget ($p = 0.019$). Most of what MIRROR earns on this benchmark can therefore be earned by aiming at where the teammate was last seen, and the replica, which is what distinguishes MIRROR from that simpler rule, is not what carries the result here.

There remains the choice that every result of this chapter rests on. All the comparisons so far were run at a budget of five percent, fixed on one floor, and a value chosen on one floor justifies nothing by itself. The ladder was therefore repeated on five further environments as a check on that choice. Each of them ran in a single campaign that also measured its own silence and its own unlimited reference, so no comparison crosses a campaign boundary, and the six floors together span an order of magnitude in difficulty, from 1,370 steps in silence on the ACL office to 12,170 on KTH-848.

Since the floors differ so much in absolute time, each one is measured against itself in Figure 5.11, where zero is its own mission time without communication and one hundred is its own mission time with an unlimited link. A curve at seventy-five therefore means the budget recovered three quarters of the distance between the two. Each ratio is formed inside a seed before the median is taken, which matters here, since the marginal form of the same quantity puts one floor above a ceiling it cannot exceed.

The floors disagree when the channel is starved and agree once it is not. At a tenth of a percent the captured gain ranges from eleven to fifty-six percent, and the ranking has nothing to do with the size of the building, since the lowest value belongs to a medium floor and the two highest to the largest ones.

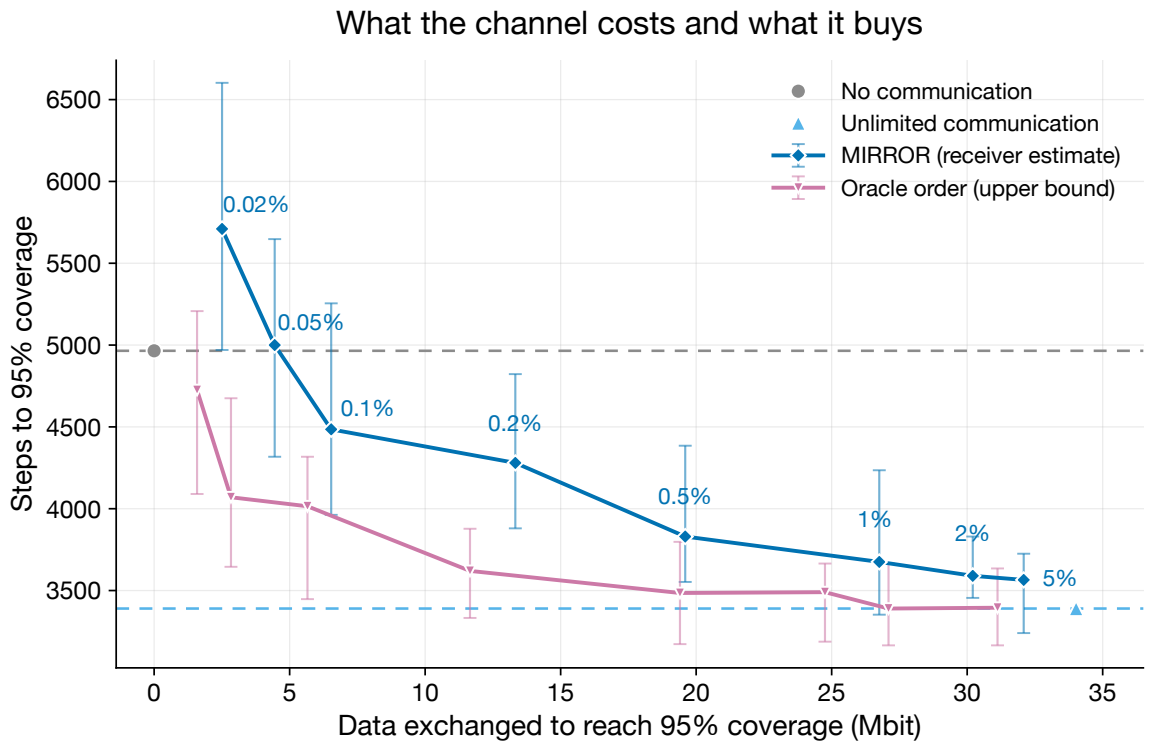

Figure 5.10: *What the channel costs and what it buys.* KTH-PLAN1 at $n = 3$, thirty paired seeds per point. Each marker is one budget, from 0.02 to five percent, so the budget is not an axis here. It is the dial that moves a policy along its own curve. The horizontal position is the map data the team delivered by the time it reached the target, so a point further to the right paid more for the time it bought. Whiskers cover the middle half of the runs, and a run that never reaches the target counts as infinite in the median. The dashed lines are the same team with no map exchange and with an unlimited link.

By half a percent the range has narrowed to sixty-eight to eighty-nine percent. At five percent every floor sits between ninety-three and ninety-eight percent of its own ceiling. This is what justifies the five percent budget used in every result of this chapter. It is the point at which the channel stops being the binding constraint, so that on all six environments MIRROR recovers nearly all the performance of unlimited map exchange, and the comparisons between policies made at that budget measure the policies rather than the width of the link. The sensor sweep of Section 5.5 was run at the same value, so its conclusions rest on this justification as well. Half a percent remains the setting to prefer when bits matter more than time, since it already returns most of the gain for a fraction of the data.

Read together, the two figures answer the question Chapter 1 closes with. Under a limited communication budget, carefully selecting what to transmit can preserve nearly all of the benefits of full communication. At a five-percent budget, MIRROR leaves at most seven percent of the achievable gain unclaimed across all environments tested. The oracle shows that the same performance can be reached with even less

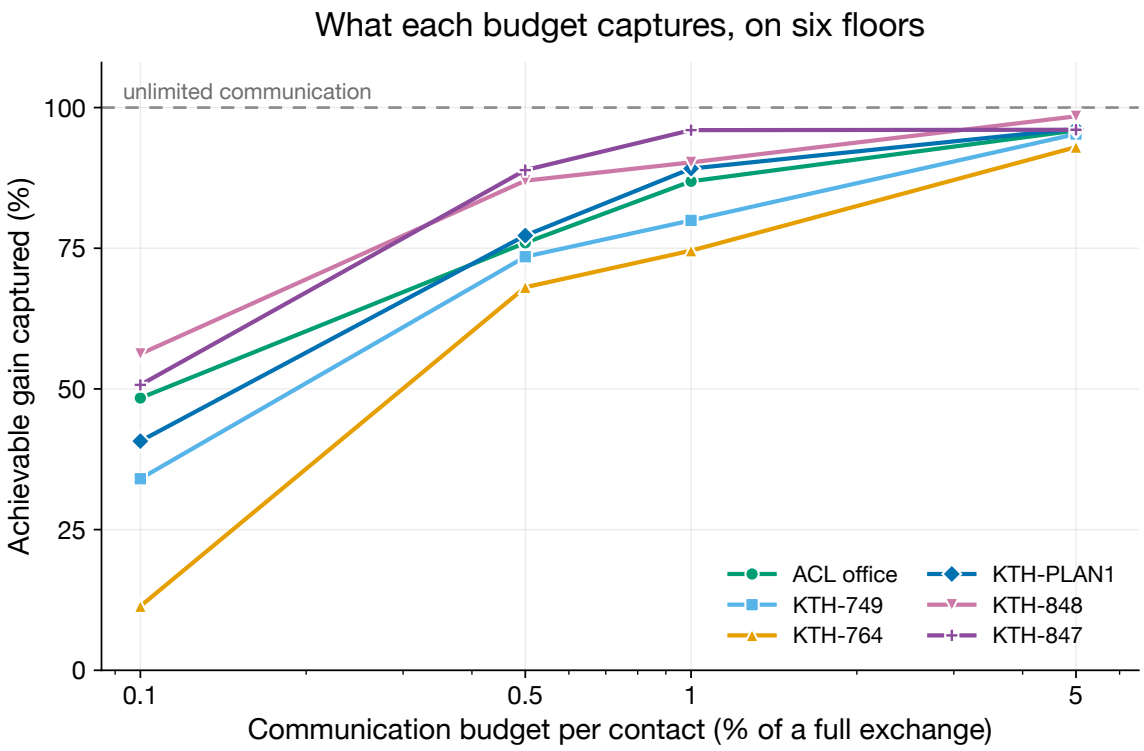

Figure 5.11: *What each budget captures, on six floors.* MIRROR at $n = 3$, thirty paired seeds per point. The vertical axis is the share of the achievable gain of each floor, where zero is that floor with no map exchange and one hundred is the same floor with an unlimited link. The share is paired, formed inside each seed and then taken as the median over seeds. Seeds whose own distance between silence and unlimited is shorter than fifty steps carry no information about a share and are dropped, which leaves between twenty-six and thirty seeds per point, and a run that never reaches the target counts as infinite. The six floors span an order of magnitude in difficulty, from 1,370 steps in silence on the ACL office to 12,170 on KTH-848.

communication when the sender knows the receiver's true map. The remaining question is therefore how closely MIRROR's estimate of the receiver can approach this oracle efficiency.

6 Conclusions and Future Work

6.1 Conclusions

This thesis built MIRROR, a way for a robot to decide which part of its map to transmit when a contact cannot carry all of it. The sender keeps an estimate of what its teammate already knows, assembled from the cells it has itself delivered and from the positions where it has seen that teammate, and it sends first the cells that fall beyond the edge of that estimate. Building that estimate costs nothing, since both records come from exchanges that have already happened. There is no reply to wait for and no extra message, and the protocol does not change, so a system that already exchanges positions can adopt MIRROR as it stands, and if a contact breaks early, what has gone through is what mattered most.

We built MIRROR because the choice of which cells to send first cannot be avoided. A map of a real floor amounts to megabits, a contact between two robots lasts seconds, and the radios that survive underground or on a small flying platform carry kilobits per second. Work on coordination has mostly assumed that maps merge whenever robots meet, and work on communication has concentrated on making the payload smaller, through compression, topological summaries, or the difference since the last meeting. Under a real budget something still has to choose which cells go first, and in the systems deployed today that choice falls to the order in which the map happens to sit in memory. We preferred to make it deliberately.

The method was measured in simulation, on six real building floor plans that span a factor of twenty in area, with teams of two to seven robots, sensors from 5 to 40 m, and budgets spanning more than two orders of magnitude. Every configuration ran on thirty paired seeds, so two policies always faced the same building, the same starting poses, and the same chances to talk. The comparison ran from a team that shares nothing to one that merges its maps completely, and it included an ordering that reads the receiver's true map, which no robot could run and which therefore says how much room any policy still has.

What the measurements show is that choosing which cells to send buys time. Across those floors,

MIRROR gives back between a quarter and a half of the mission a silent team spends, about seventeen and twenty minutes of exploration on the two large buildings, and it finishes within a few percent of a team whose link is unlimited. The gain comes from what the robots stop doing rather than from any new speed. A team that shares nothing spends between 44 and 64 percent of its sensing on ground a teammate has already covered, and once messages are aimed that duplication almost disappears.

The interesting part is how little the link has to carry for this. Each contact delivers one twentieth of what a full exchange would deliver at that moment, and at that setting the team already captures between 93 and 98 percent of everything communication can give, on every floor we tested. What is saved there is time rather than bits, since a cap on one contact only postpones the rest, and a mission run this way still moves nearly all the data an unlimited link would move. A team that has to spare the channel itself can go lower. At half a percent per contact it spends between 61 and 72 percent of the traffic and still returns between 68 and 89 percent of what communication can give, which is where to sit when bits matter more than minutes.

The advantage also survives the two settings that could most easily explain it away. It holds at every sensor range we tried, and it grows as sensing becomes scarce, roughly tripling when the sensor shrinks from 25 to 5 m, which on that floor is the difference between saving about one kilometer of driving and saving three. It holds as the team grows too, staying between 28 and 54 percent from two to seven robots, and on the large building three robots that share finish in the time six silent ones need. Choosing well what to transmit is worth about as much as doubling the fleet.

Two things we found say more about communication than about our policy. Sending the wrong content is not merely wasteful, it can cost time, since a robot that shares its own frontier hands its teammate isolated fragments that look like new places to go, and the teammate drives to them. Taking that policy’s budget away entirely does not rescue it, so the trouble lies in what it selects rather than in how much it may send. The order in which cells travel, on the other hand, decides the mission exactly where the link is tight. At half a percent the same cells sent in memory order cost 2,020 steps on the largest floor, while at five percent no floor tells the two apart, because by then almost everything gets through anyway.

Some limits belong with all of this. The numbers come from a two-dimensional simulator with exact sensing and solved localization, one channel model and one goal selector, so they rank policies rather than predict times for a real deployment. The estimate is also not proven necessary in the form we built it, since a simpler variant that only remembers where the teammate was last seen does as well on this benchmark. And the ceiling is still above us, because an ordering that knows the receiver exactly reaches an unlimited link on little more than half the traffic MIRROR spends. What this thesis does establish is that aiming a message with what the sender already has recovers most of what an unconstrained channel would give, and that what remains to be gained lies in the sharpness of that aim rather than in

the capacity of the link.

6.2 Future Work

Future work may be directed first at the setting in which the policy is evaluated. The definitions of this thesis extend to three dimensions unchanged, whereas its quantities do not. In a plane, the boundary between what a robot knows and what it does not is a thin rim around the mapped area, and only the cells on that rim can change what the receiver does next. In a volume that boundary becomes a surface enclosing the mapped space, so the share of a payload able to act on the receiver is different, and it must be measured again before any budget is claimed. A real deployment raises the same requirement, since sensing is noisy, localization drifts, and the estimate of the receiver's map degrades in ways a simulator does not reproduce. A deployment on the laboratory platforms, in the office floor already used as one of the benchmarks of this thesis, would be the natural first step.

A second direction concerns the estimate itself, which the results identify as the remaining bottleneck. A sender currently relies on the cells it has delivered and on the positions at which it has observed its teammate, whereas buildings are strongly structured, and a learned model of that structure would allow it to anticipate what the teammate is about to discover rather than only what it has been told. The difficulty lies in the generality of such a model, since a predictor trained on one set of floors tends to perform well on those floors and to mislead elsewhere, and that bias would propagate into every message it orders. This line therefore requires a learned estimate that does not depend on the buildings used for training, evaluated on floors excluded from it, and characterized by how it degrades when the environment differs from anything seen during training.

Bibliography

[1] Timothy H. Chung, Viktor Orekhov, and Angela Maio. Into the robotic depths: analysis and insights from the DARPA Subterranean Challenge. *Annual Review of Control, Robotics, and Autonomous Systems*, 6:477–502, 2023. doi: 10.1146/annurev-control-062722-100728.

[2] Timothy N. Titus, J. Judson Wynne, Michael J. Malaska, Ali-akbar Agha-Mohammadi, Peter B. Buhler, E. Calvin Alexander, et al. A roadmap for planetary caves science and exploration. *Nature Astronomy*, 5(6):524–525, 2021. doi: 10.1038/s41550-021-01385-1.

[3] Brian Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings of the 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pages 146–151, Monterey, CA, USA, 1997. IEEE. doi: 10.1109/CIRA.1997.613851.

[4] Wolfram Burgard, Mark Moors, Cyrill Stachniss, and Frank E. Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21(3):376–386, 2005. doi: 10.1109/TRO.2004.839232.

[5] Cyrill Stachniss, Giorgio Grisetti, and Wolfram Burgard. Information gain-based exploration using Rao-Blackwellized particle filters. In *Robotics: Science and Systems I*, Cambridge, MA, USA, 2005. Robotics: Science and Systems Foundation. doi: 10.15607/RSS.2005.I.009.

[6] Brian Yamauchi. Frontier-based exploration using multiple robots. In *Proceedings of the Second International Conference on Autonomous Agents (AGENTS)*, pages 47–53. ACM, 1998. doi: 10.1145/280765.280773.

[7] Ryan J. Marcotte, Xipeng Wang, Dhanvin Mehta, and Edwin Olson. Optimizing multi-robot communication under bandwidth constraints. *Autonomous Robots*, 44(1):43–55, 2020. doi: 10.1007/s10514-019-09849-0. First published online 17 April 2019.

[8] Jan Bayer and Jan Faigl. Decentralized multi-robot exploration under low-bandwidth communications. *Autonomous Robots*, 50(1), 2026. doi: 10.1007/s10514-025-10234-3. First published online 29 December 2025.

[9] Francesco Amigoni, Jacopo Banfi, and Nicola Basilico. Multirobot exploration of communication-restricted environments: a survey. *IEEE Intelligent Systems*, 32(6):48–57, 2017. doi: 10.1109/MIS.2017.4531226.

[10] Maira Saboia, Lillian Clark, Vivek Thangavelu, Jeffrey A. Edlund, Kyohei Otsu, Gustavo J. Correa, et al. ACHORD: communication-aware multi-robot coordination with intermittent connectivity. *IEEE Robotics and Automation Letters*, 7(4):10184–10191, 2022. doi: 10.1109/LRA.2022.3193240.

[11] Harel Biggie, Eugene R. Rush, Danny G. Riley, Shakeeb Ahmad, Michael T. Ohradzansky, Kyle Harlow, Michael J. Miles, Daniel Torres, Steve McGuire, Eric W. Frew, Christoffer Heckman, and J. Sean Humbert. Flexible supervised autonomy for exploration in subterranean environments. *Field Robotics*, 3:125–189, 2023. doi: 10.55417/fr.2023004.

[12] Micah Corah, Cormac O'Meadhra, Kshitij Goel, and Nathan Michael. Communication-efficient planning and mapping for multi-robot exploration in large environments. *IEEE Robotics and Automation Letters*, 4(2):1715–1721, 2019. doi: 10.1109/LRA.2019.2897368.

[13] Liuyu Shi, Longji Yin, Fanze Kong, Yunfan Ren, Fangcheng Zhu, Benxu Tang, and Fu Zhang. Real-time bandwidth-efficient occupancy grid map synchronization for multi-robot systems. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8489–8496. IEEE, 2024. doi: 10.1109/IROS58592.2024.10802281.

[14] Zhaoliang Zhang, Jincheng Yu, Jiahao Tang, Yuanfan Xu, and Yu Wang. MR-TopoMap: multi-robot exploration based on topological map in communication restricted environment. *IEEE Robotics and Automation Letters*, 7(4):10794–10801, 2022. doi: 10.1109/LRA.2022.3192765.

[15] Hans Moravec and Alberto Elfes. High resolution maps from wide angle sonar. In *Proceedings of the 1985 IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 116–121. IEEE, 1985. doi: 10.1109/ROBOT.1985.1087316.

[16] Alberto Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22 (6):46–57, 1989. doi: 10.1109/2.30720.

[17] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, Jose Neira, Ian Reid, and John J. Leonard. Past, present, and future of simultaneous localization and mapping: toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016. doi: 10.1109/TRO.2016.2624754.

[18] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. doi: 10.1109/TSSC.1968.300136.

[19] Yulun Tian, Yun Chang, Long Quang, Arthur Schang, Carlos Nieto-Granda, Jonathan P. How, and Luca Carlone. Resilient and distributed multi-robot visual SLAM: datasets, experiments, and lessons learned. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11027–11034. IEEE, 2023. doi: 10.1109/IROS55552.2023.10342377.

[20] Yulun Tian, Yun Chang, Fernando Herrera Arias, Carlos Nieto-Granda, Jonathan P. How, and Luca Carlone. Kimera-multi: robust, distributed, dense metric-semantic SLAM for multi-robot systems. *IEEE Transactions on Robotics*, 38(4):2022–2038, 2022. doi: 10.1109/TRO.2021.3137751.

[21] Yulun Tian and Jonathan P. How. Spectral sparsification for communication-efficient collaborative rotation and translation estimation. *IEEE Transactions on Robotics*, 40:257–276, 2024. doi: 10.1109/TRO.2023.3327635.

[22] Héctor H. González-Baños and Jean-Claude Latombe. Navigation strategies for exploring indoor environments. *The International Journal of Robotics Research*, 21(10-11):829–848, 2002. doi: 10.1177/027836490202101083 4.

[23] Alberto Quattrini Li. Exploration and mapping with groups of robots: recent trends. *Current Robotics Reports*, 1(4):227–237, 2020. doi: 10.1007/s43154-020-00030-5.

[24] Kai M. Wurm, Cyrill Stachniss, and Wolfram Burgard. Coordinated multi-robot exploration using a segmentation of the environment. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1160–1165. IEEE, 2008. doi: 10.1109/IROS.2008.4650734.

[25] Robert Zlot, Anthony Stentz, M. Bernardine Dias, and Scott Thayer. Multi-robot exploration controlled by a market economy. In *Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 3016–3023. IEEE, 2002. doi: 10.1109/ROBOT.2002.1013690.

[26] Brian P. Gerkey and Maja J. Matarić. Sold!: auction methods for multirobot coordination. *IEEE Transactions on Robotics and Automation*, 18(5):758–768, 2002. doi: 10.1109/TRA.2002.803462.

[27] M. Bernardine Dias, Robert Zlot, Nidhi Kalra, and Anthony Stentz. Market-based multirobot coordination: a survey and analysis. *Proceedings of the IEEE*, 94(7):1257–1270, 2006. doi: 10.1109/JPROC.2006.876939.

[28] Antoine Bautin, Olivier Simonin, and François Charpillet. MinPos: a novel frontier allocation algorithm for multi-robot exploration. In *Intelligent Robotics and Applications (ICIRA 2012)*, volume 7507 of *Lecture Notes in Computer Science*, pages 496–508. Springer, 2012. doi: 10.1007/978-3-642-33515-0_49.

[29] Brian J. Julian, Sertac Karaman, and Daniela Rus. On mutual information-based control of range sensing robots for mapping applications. *The International Journal of Robotics Research*, 33(10): 1375–1392, 2014. doi: 10.1177/0278364914526288.

[30] Nikolay Atanasov, Jerome Le Ny, Kostas Daniilidis, and George J. Pappas. Decentralized active information acquisition: theory and application to multi-robot SLAM. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4775–4782. IEEE, 2015. doi: 10.1109/ICRA.2015.7139863.

[31] Micah Corah and Nathan Michael. Distributed matroid-constrained submodular maximization for multi-robot exploration: theory and practice. *Autonomous Robots*, 43(2):485–501, 2019. doi: 10.1007/s10514-018-9778-6. First published online 19 July 2018.

[32] Jan Faigl and Miroslav Kulich. On benchmarking of frontier-based multi-robot exploration strategies. In *2015 European Conference on Mobile Robots (ECMR)*, pages 1–8. IEEE, 2015. doi: 10.1109/ECMR.2015.7324183.

[33] Rakesh Shrestha, Fei-Peng Tian, Wei Feng, Ping Tan, and Richard Vaughan. Learned map prediction for enhanced mobile robot exploration. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1197–1204. IEEE, 2019. doi: 10.1109/ICRA.2019.8793769.

[34] Kapil D. Katyal, Katie Popek, Chris Paxton, Philippe Burlina, and Gregory D. Hager. Uncertainty-aware occupancy map prediction using generative networks for robot navigation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5453–5459. IEEE, 2019. doi: 10.1109/ICRA.2019.8793500.

[35] Santhosh K. Ramakrishnan, Ziad Al-Halah, and Kristen Grauman. Occupancy anticipation for efficient exploration and navigation. In *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pages 400–418. Springer, 2020. doi: 10.1007/978-3-030-58558-7_24.

[36] Cherie Ho, Seungchan Kim, Brady Moon, Aditya Parandekar, Narek Harutyunyan, Chen Wang, Katia Sycara, Graeme Best, and Sebastian Scherer. MapEx: indoor structure exploration with probabilistic information gain from global map predictions. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13074–13080. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11128862.

[37] Georgios Georgakis, Bernadette Bucher, Anton Arapin, Karl Schmeckpeper, Nikolai Matni, and Kostas Daniilidis. Uncertainty-driven planner for exploration and navigation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 11295–11302. IEEE, 2022. doi: 10.1109/ICRA46639.2022.9812423.

[38] Seungjae Baek, Brady Moon, Seungchan Kim, Muqing Cao, Cherie Ho, Sebastian Scherer, and Jeong hwan Jeon. PIPE planner: pathwise information gain with map predictions for indoor robot exploration. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7684–7691. IEEE, 2025. doi: 10.1109/IROS60139.2025.11246190.

[39] Kun Song, Gaoming Chen, Masayoshi Tomizuka, Wei Zhan, Zhenhua Xiong, and Mingyu Ding. P^2 Explore: efficient exploration in unknown cluttered environment with floor plan prediction. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13090–13096. IEEE, 2025. doi: 10.1109/IROS60139.2025.11247205.

[40] Matteo Luperto, Luca Fochetta, and Francesco Amigoni. Exploration of indoor environments through predicting the layout of partially observed rooms. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 836–843. IFAAMAS, 2021. doi: 10.65109/xald3176.

[41] Aaron Hao Tan, Siddarth Narasimhan, and Goldie Nejat. 4CNet: a diffusion approach to map prediction for decentralized multirobot exploration. *IEEE Transactions on Robotics*, 42:1371–1388, 2026. doi: 10.1109/TRO.2026.3666133.

[42] Alexander Spinos, Bradley Woosley, Justin Rokisky, Christopher Korpela, John G. Rogers, III, and Brian A. Bittner. Map prediction and generative entropy for multi-agent exploration, 2025. arXiv preprint arXiv:2501.13189.

[43] Seungchan Kim, Seungjae Baek, Micah Corah, Graeme Best, Brady Moon, and Sebastian Scherer. PRoID: predicted rate of information delivery in multi-robot exploration and relaying, 2026. arXiv preprint arXiv:2604.10433.

[44] Yixiao Ma, Jingsong Liang, Yuhong Cao, Derek Ming Siang Tan, and Guillaume Sartoretti. Privileged reinforcement and communication learning for distributed, bandwidth-limited multi-robot exploration. In *Distributed Autonomous Robotic Systems (DARS 2024)*, Springer Proceedings in Advanced Robotics, pages 337–350. Springer, 2025. doi: 10.1007/978-3-032-04584-3_23.

[45] Maayan Roth, Reid Simmons, and Manuela Veloso. What to communicate? Execution-time decision in multi-agent POMDPs. In *Distributed Autonomous Robotic Systems 7*, pages 177–186. Springer, 2006. doi: 10.1007/4-431-35881-1_18.

[46] Jacopo Banfi, Alberto Quattrini Li, Ioannis Rekleitis, Francesco Amigoni, and Nicola Basilico. Strategies for coordinated multirobot exploration with recurrent connectivity constraints. *Autonomous Robots*, 42(4):875–894, 2018. doi: 10.1007/s10514-017-9652-y. First published online 25 July 2017.

[47] Laetitia Matignon, Laurent Jeanpierre, and Abdel-Illah Mouaddib. Coordinated multi-robot exploration under communication constraints using decentralized Markov decision processes. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI)*, pages 2017–2023. AAAI Press, 2012. doi: 10.1609/aaai.v26i1.8380.

[48] Graeme Best, Michael Forrai, Ramgopal R. Mettu, and Robert Fitch. Planning-aware communication for decentralised multi-robot coordination. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1050–1057. IEEE, 2018. doi: 10.1109/ICRA.2018.8460617.

[49] Lauren Bramblett, Shijie Gao, and Nicola Bezzo. Epistemic prediction and planning with implicit coordination for multi-robot teams in communication restricted environments. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5744–5750. IEEE, 2023. doi: 10.1109/ICRA48891.2023.10161553.

[50] Ninad Jadhav, Meghna Behari, Robert J. Wood, and Stephanie Gil. WiSER-X: wireless signals-based efficient decentralized multi-robot exploration without explicit information exchange, 2024. arXiv preprint arXiv:2412.19876.

[51] Michael E. Kepler and Daniel J. Stilwell. An approach to reduce communication for multi-agent mapping applications. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4814–4820. IEEE, 2020. doi: 10.1109/IROS45743.2020.9341117.

[52] Yulun Tian, Kasra Khosoussi, Matthew Giamou, Jonathan P. How, and Jonathan Kelly. Near-optimal budgeted data exchange for distributed loop closure detection. In *Robotics: Science and Systems XIV*. Robotics: Science and Systems Foundation, 2018. doi: 10.15607/RSS.2018.XIV.071.

[53] Guanchong Niu, Shuwake Nuerdaolieti, Jinglei Li, Zewei Jing, Lanxiang Hou, and Jiayi Sun. Feedback-guided hierarchical transmission for multi-robot mapping under bandwidth constraints. *IEEE Robotics and Automation Letters*, 11(7):8576–8583, 2026. doi: 10.1109/LRA.2026.3699241.

[54] Sebastian Trimpe and Raffaello D'Andrea. Event-based state estimation with variance-based triggering. *IEEE Transactions on Automatic Control*, 59(12):3266–3281, 2014. doi: 10.1109/TAC.2014.2351951.

[55] Alper Aydemir, Patric Jensfelt, and John Folkesson. What can we learn from 38,000 rooms? Reasoning about unexplored space in indoor environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4675–4682, 2012.

Disclaimer on LLM Usage

The passages, main ideas, theoretical developments, and experimental results presented in this thesis are the original work of the author. In the interest of transparency in academic work, Large Language Models (LLMs), such as ChatGPT and Claude, were used to assist with preliminary brainstorming of possible research directions, with the typesetting and formatting of the manuscript, and with proofreading and polishing the final text. The author has verified the correctness of all content and takes full responsibility and ownership of all content in the thesis.