Collaborative Visual Navigation

Haiyang Wang[1,2], Wenguan Wang[3*], Xizhou Zhu[2], Jifeng Dai[2], Liwei Wang[1]
[1] Key Laboratory of Machine Perception, MOE, Peking University [2] SenseTime Research [3] Computer Vision Lab, ETH Zurich
wanghaiyang@stu.pku.edu.cn, wenguanwang.ai@gmail.com, {daijifeng, zhuwalter}@sensetime.com, wanglw@cis.pku.edu.cn
https://github.com/Haiyang-W/MAVN

Abstract

As a fundamental problem for Artificial Intelligence, multi-agent system (MAS) is making rapid progress, mainly driven by multi-agent reinforcement learning (MARL) techniques. However, previous MARL methods largely focused on grid-world like or game environments; MAS in visually-rich environments has remained less explored. To narrow this gap and emphasize the crucial role of perception in MAS, we propose a large-scale 3D dataset, CollaVN, for multi-agent visual navigation (MAVN). In CollaVN, multiple agents are entailed to cooperatively navigate across photo-realistic environments to reach target locations. Diverse MAVN variants are explored to make our problem more general. Moreover, a memory-augmented communication framework is proposed. Each agent is equipped with a private, external memory to persistently store communication information. This allows agents to make better use of their past communication information, enabling more efficient collaboration and robust long-term planning. In our experiments, several baselines and evaluation metrics are designed. We also empirically verify the efficacy of our proposed MARL approach across different MAVN task settings.

1. Introduction

Human intelligence would collectively solve the problem that otherwise is unthinkable by a single person. For instance, Condorect's Jury Theorem [48] indicates that, under certain assumption, adding more voters in a group would increase the probability that the majority chooses the right answer. Thus it is believed that a next grand challenge of AI is to answer how multiple intelligent agents could learn human-level collaborations [28, 64]. With the recent rapid advances of machine learning and robotic technologies, the collaboration of AI agents gathers significantly increasing attention from researchers, and finds great potential in a wide range of real-world applications, such as search-and rescue [52], area coverage [78, 42], harbour protection [57], and so on. In short, such multi-agent system (MAS) [79] is of central importance to AI and still in its infancy.

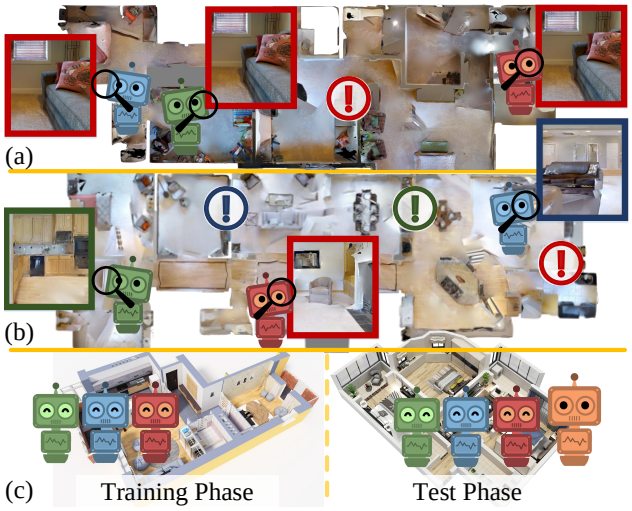

Figure 1: Three collaborative visual navigation task settings in CollaVN: (a) CommonGoal, (b) SpecificGoal, and (c) Ad-hoCoop.

Multi-agent reinforcement learning (MARL) [11] and embodied systems [65] are the main driving power behind MAS. MARL field is rapidly expanding, and some of approaches even show super-human level performance in various games [37], such as Magent [96], StarCraft [70], and Dota2 [7]. However, most of existing algorithms [55, 69, 24, 49] operate on low-dimensional observations, *e.g.*, grid-world like or game environments. As for embodied systems, the availability of large-scale, visually-rich 3D datasets [4, 13, 88, 77] and community interest in active perception led to a recent surge of simulation platforms [75, 13, 45, 71, 90]. Based on these high-performance 3D simulators, various active vision tasks are investigated, such as embodied navigation [16, 22], instruction following [3], embodied question-answering [20] and perception [53], *etc*. However, most of them either do not support multi-agent setting [16, 22, 3, 20], or little consider planning [20, 53].

Thus there is still a huge gap between MAS and realistic visual environments. To close this gap, we introduce

*Corresponding author: Wenguan Wang.

a collaborative visual navigation (MAVN) task for image-goal based, multi-agent navigation, under visually realistic environments, and develop a corresponding dataset, called CollaVN. Autonomous navigation forms a core building block in embodied systems, and long received research interest across many fields, such as computer vision [16], robotics [44], linguistics[80], and cognition[47]. In MAVN, agents cooperate to reach a target position (determined by a corresponding image), with their individual perception. Three task settings are explored to cover more challenges in MAS/MARL: i) **CommonGoal** (Fig. 1(a)), where agents are assigned *a same goal picture*; ii) **SpecificGoal** (Fig. 1(b)), where agents are assigned *different goal pictures;* and iii) **Ad-hoCoop** (Fig. 1(c)), where the *agent numbers* in training and testing are different. *CommonGoal* serves as a basic setting where agents learn how to cooperate to reach a goal faster. *SpecificGoal* captures more complex scenarios where agents need to collaborate with individual targets. *Ad-hoCoop* is built upon *CommonGoal* but agents are required to adapt to different team sizes during deployment, simulating a more open MAS situation [6, 12].

Our CollaVN dataset is built upon iGibsonV1 simulator [89]. Besides inheriting the advantages of iGibsonV1 in high-quality rendering, rich diversity and good accessibility, CollaVN has the following unique characteristics:

- **Multi-agent operation:** CollaVN supports MAVN with arbitrary group sizes (2-4 agents used in current setting). Agents can be safely initialized with collision detection.
- **Multi-task setup:** All the three task variants are involved and companied with standard evaluation tools.
- **Flexible target generation:** CollaVN allows to generate different target pictures flexibly, well supporting the three MAVN settings and future dataset extension.
- **Panoramic observation:** A panoramic camera is equipped for rendering agent egocentric panorama in real-time.

We further propose a novel memory-augmented communication approach for MAVN. Communication is viewed as one of the most desired and efficient ways of multi-agent interaction/coordination [43, 49]. Communication is especially crucial when a set of cooperative agents are placed in a partially observable environment, such as our MAVN setting – the agents need to exchange information (*e.g.*, past experience, current perception, future plan, *etc.*), coordinate their actions, behave as a group and achieve the target goal better [27, 34]. Thus learning communication has become a center topic in MARL field. From earlier natural-language-based [49] to recent hidden-state-based [64] communication forms and pre-defined communication protocols [83] to learnable strategies [24, 64, 81], communication based MARL made great advance. Some most recent ones are concerned with practical communication-bandwidth-limited situations [76, 21, 53, 54] and conduct studies mainly around the theme of learning *what*, *when* and *who* to communicate. However, in previous methods, agents simply discard all the information after each communication round. This is problematic, as some communication information, even not useful in present, may be valuable in the future. This raises the risk of losing essential information and easily leads to suboptimal behaviors driven by short-termism. To address these limitations, in our memory-augmented communication approach, each agent is equipped with a private, compartmentalized memory, for safely storing and accurately recalling its past communication information. During each communication round, each agent can send a more reasonable request to other agents, by considering its current states and stored information. Each of other agents will check its current state and past communication information to give a better response. The agents can make full use of the rich information stored in the memory during navigation. Overall, our approach helps agents conduct more efficient communication and enhances their long-term planning ability, eventually leading to better collaboration and navigation performance.

In a nutshell, our contributions are three-fold:

- To narrow the gap between MAS and visual perception, we develop a large-scale and publicly-accessible dataset, CollaVN, for collaborative visual navigation (MAVN), in photo-realistic, multi-agent environments.
- Diverse MAVN task settings are explored to cover many core challenges in MAS/MARL, with packaged evaluation tools and baselines.
- A novel memory-augmented communication approach is proposed to address more efficient multi-agent collaborations and robust long-term planning.

Several baselines are evaluation tools are developed for comprehensive benchmarking. Experimental results show that our memory-augmented communication framework achieves promising performance over the three MAVN settings on CollaVN dataset. This work is expected to provide insights into the critical role of MAS in embodied perception tasks, and foster research on the open issues raised.

2. Related Work

Vision-based Navigation. As a crucial step towards building intelligent robots, autonomous navigation has been long studied in robotics community. Recently vision-based navigation [16] attained growing attention in computer vision community and was explored in many other forms, such as point-goal based or object-oriented (*i.e.*, reaching a specific position or finding an object) [97, 16, 14, 22], natural-language-guided [3, 86], audio-assisted navigation [17], and from indoor environments [3] to street scenes [18]. Prominent early methods rely on a pre-given/-computed global map [8, 44] for path planing, while later ones refer to simultaneous localization and mapping (SLAM) techniques [84, 31] that reconstruct map on the fly. Some more

recent methods learn planning and mapping jointly [30, 63, 92, 15, 50, 85].

Although great advance has been achieved, visual navigation in multi-agent setting has remained largely unexamined. Most existing studies [3, 17, 3, 18] focused on single-agent navigation, excepting [38, 39] considering two-agent collaboration in finding and lifting bulky items. In this work, we develop a large-scale dataset for indoor, collaborative, multi-agent visual navigation. The agents are required to work together to perform same or different tasks, or even without any assumption on the number of involved agents. Thus our task setting is more novel and general.

Embodied AI Environments. The recent surge of research in embodied perception is greatly driven by new 3D environments [90, 4, 13, 10, 77] and simulation platforms, such as iGibsonV1 [89], AI2-THOR [45], and Habitat [71]. Compared with grid-like or game environments [96, 70, 7, 37], these open accessible, photo-realistic platforms bring perception and planning in a close loop and make the training and testing of embodied agents reproducible [17]. Based on these platforms, numerous embodied vision datasets were proposed [22, 3, 17, 22], while most of them are designed for single-agent tasks. We build our dataset upon iGibsonV1 [89] which is featured by large scale, rich visual diversity and strong extensibility. Although iGibsonV2 [75] also involves multiple agents, it addresses obstacle avoidance in social scenes. It does not investigate collaboration among agents, while which is the core nature and research focus in MAS/MARL. Moreover, our dataset supports diverse essential MARL task settings, making it unique in the filed of visual navigation.

Multi-Agent Reinforcement Learning (MARL). MARL tackles the sequential decision-making problem of multiple agents in a common environment, each of which aims to achieve its own long-term goal by interacting with the environment and other agents [11, 34]. Basically, MARL algorithms can be placed into four categories [34]. i) *Analysis of emergent behaviors*. Some early studies [67, 5] analyze single-agent RL algorithms in multi-agent environments. ii) *Learning communication*. Methods in this category [59, 49, 24] address collaboration through explicit communication, which attracts increasing attention recently. iii) *Learning cooperation*. Many other efforts [62, 56, 61] indirectly arrive at cooperation via, for example, policy parameter sharing [29] or experience replay buffer [26]. iv) *Agents modeling agents.* These methods [68, 35] build models to reason the behaviors of other agents for better decision-making.

These algorithms mostly focus on grid-world like scenes, despite the role of perception to some degree. Our CollaVN dataset provides a visually-rich platform for fostering studies about MARL in computer vision. We consider MAVN is of a cooperative nature and agents are situated in a partially observable environment, and concern with learning efficient communication for long-term and collaborative planning.

Learn to Communicate. Communication is a fundamental aspect of intelligence, enabling agents to behave as a group, rather than a collection of individuals. Along with the direction of communication based collaboration, MARL researchers first used predefined communication protocols [83], and then learnable strategies [24, 64, 41, 81] for information exchanging. However, sharing information among all agents is problematic, as communication bandwidth is limited the real world. To reduce bandwidth wasting, some recent methods adopt pre-defined communication groups [41, 40] or set "what, when and/or who to communicate" as a part of policy learning [24, 76, 21, 53, 54].

As previous methods only have an "immediate memory" during communication, they suffer from a bias towards short-sighted behaviors. We instead equip each agent with a private, external memory that persistently stores the information in all past communication rounds and enables the reuse of these information during future navigation. This enhances the long-term planning ability of our agents. This idea is also distinctively different from [64, 66], where all the agents need to transmit messages to a central memory, incurring a major communication bottleneck.

Ad-Hoc Cooperation. The problem of ad-hoc team play in multi-agent cooperative games was raised in the early 2000s [9] and is mostly studied in the robotic soccer domain [32]. In many close MARL environments agents learn a fixed and team-dependent policy [95]. While in ad-hoc setting agents are desired to assess and adapt to the capabilities of others to behave optimally in an open environment. See [1] for a survey. Existing work in ad-hoc MARL mainly focuses on game settings, such as hanabi (card game) [12]. Our work makes a further step towards exploring ad-hoc setting in visually-rich MARL; the navigation agents are required to adapt to different team sizes during deployment.

3. Multi-Agent Visual Navigation Task

3.1. Our CollaVN Dataset

CollaVN is a visual MARL dataset, created for the development of intelligent, cooperative navigation agents.

iGibsionV1 [89]. CollaVN is built upon iGibsonV1, a recently released open-source 3D simulator. iGibsonV1 supports fast visual rendering in 400 fps and allows researchers to easily train and evaluate embodied navigation agents.

New Modules. To support multi-agent navigation task, three modules are developed: i) An agent initialization module is responsible for "safely" placing multiple agents at the beginning of each navigation episode, *i.e.*, randomly placing agents but being aware of constraints of scene and physics, such as avoiding collision. ii) As the panoramic visual observation space is widely adopted in current embodied vision tasks [16, 23], a panoramic camera is equipped to

Dataset	Pub.	Year	MA	CM	LS	CA	PA
AI2THOR [45]	arXiv	2017	✓	✓			
Habitat [71]	ICCV	2019			✓		✓
iGibsonV1 [89]	RAL	2020			✓	✓	
iGibsonV2 [75]	arXiv	2020	✓		✓	✓	
CollaVN	-	2021	✓	✓	✓	✓	✓

Table 1: Point-to-point comparison to closely relevant 3D platforms/datasets. MA: Multi-agent. CM: Multi-agent communication. LS: Large scale. CA: Continuous action. PA: Panorama.

render agents 360° egocentric views in real-time. iii) A target generation module is built for generating a target picture as the goal of navigation. This module adopts a front-view camera, taking pictures around the agent height (*i.e.*, 0.9 m) under the constraints of scene and physics.

Scenes. All the 572 full buildings in iGibsonV1 are involved in CollaVN, covering a total area of 211K m^2. The area of a physical navigation space considered in CollaVN is from 10 m × 10 m to 30 m × 30 m.

Agent. In CollaVN, we use Locobot, a widely used simulation agent [60]. Its body width is 0.36 m, controlled by two wheels. The maximum translational velocity is 70 cm/s and rotational velocity is 180 deg/s.

Data Generation. Following Gibson-tiny split [89], we use 25/5/5 scenes for creating `train`/`val`/`test` data. We build three sub-dasets for different MAVN tasks, *i.e.*, *CommonGoal*, *SpecificGoal*, *Ad-hoCoop*. A total of 1M/60K/120K episodes/samples are generated for `train`/`val`/`test`. According to initial agent-target distance at the beginning of each episode, fine-grained annotations for `val`/ `test` data are given: {*easy* (1.5-3 m), *medium* (3-5 m), *hard* (5-10 m)}. We provide more details in §3.2.

Comparison to Previous Datasets. As shown in Table 1, Habitat [71] only supports the single-agent setting. AI2THOR [45] just supported multi-agent setting recently. However, its main focus is task planning of sequential actions to change object states and its environment space is small (only 9 m × 4 m), making it not suitable for studying challenging, long-range navigation. Note that [38, 39] recently extended AI2THOR to involve a collaborative task, *i.e.*, finding and lifting furniture. They are limited to highly-correlated actions between two agents and cannot explore complex multi-agent collaborative behaviors under more general situations (*e.g.*, larger team sizes, different goals). iGibsonV2 [75], also built upon iGibsonV1, mainly focuses on obstacle avoidance in social scenes, *i.e.*, only one agent is required to execute point-goal navigation while other agents are moving around (acting as pedestrians). Thus iGibsonV2 does not address collaboration among multiple navigators.

3.2. Task Setup

Basic Setting. MAVN assumes there are N autonomous agents situated in a CollaVN environment. Each agent is required to navigate the environment to reach a target location (indicated by a goal image), under a cooperative setting. Formally, at the beginning of each navigation episode, each agent n will be assigned a target goal image g_n, *i.e.*, a $3\times128\times128$ RGB image. Each agent does not know other agents' positions. At each time step t, each agent n receives its visual observation O_n^t, *i.e.*, a first-person panoramic-view $3\times512\times128$ RGB image. As in many communication based MARL settings [41, 24, 43], agents operate in a partially observable environment and perceive the environment from their own view. They can exchange information via a low bandwidth communication channel. Sensing and control errors and communication delay are not considered.

Action Space. The agents can control its wheels, and the maximum allowed translational and rotational velocities are 70 cm/s and 180 deg/s, respectively. At each time step t, each agent n takes a continuous action $a_n^t \in [-1, 1]^2$, corresponding to the normalized velocities. For example, $a_n^t = (1, 0.5)$ means the agent will move at the maximum translational velocity (*i.e.*, 70 cm/s) and half of the maximum rotational velocity (*i.e.*, 90 deg/s) during $[t, t{+}1]$. The interval between t and $t{+}1$ is 1 s.

MAVN Task Settings. To better cover challenges in MARL and MAS, three different MAVN task settings are designed:

- **CommonGoal:** N robot agents are assigned a common goal, *i.e.*, $g_1 = \cdots = g_n = \cdots = g_N$. Agents need to collaboratively navigate to the target goal within a maximum time step length T. This is the most basic MAVN task setting with the purpose of investing how several agents can learn, from visual environment, to communicate so as to effectively and collaboratively solve a same given task.
- **SpecificGoal:** The difference from *CommonGoal* is that, *SpecificGoal* assigns N agents with N different goal images, *i.e.*, $g_1 \neq \cdots \neq g_n \neq \cdots \neq g_N$. In this setting, we are particularly interested in how agents learn communication and collaboration with different long-term targets.
- **Ad-hoCoop:** In *CommonGoal* and *SpecificGoal*, the team size remains fixed during both training and deployment phases. The learned agents may not generalize to new configurations of teams. *Ad-hoCoop* is a more open world setting, in which agents must adapt to different team sizes. We test the ad-hoc team play by adding or removing agents at test time. Note that in *Ad-hoCoop* agents are assigned a same goal in each episode.

Task-Specific Sub-Dataset. Our CollaVN has three sub-datasets, correspond to the three MAVN tasks:

- **CommonGoal-CollaVN**: Three subsets are created for different agent numbers (*i.e.*, $N = \{2, 3, 4\}$) and agents in each episode are assigned a same target image. For each subset, we generate 5K episodes per training scene. For each val/test scene, we generate 0.5K/1K episodes per configuration, *i.e.*, {*easy* (1.5-3 m), *medium* (3-5 m), *hard*

(5-10 m)}. Finally, each subset has 125K/7.5K/15K samples for `train`/`val`/`test`.

- **SpecificGoal-CollaVN**: Three subsets are created for different agent numbers (*i.e.*, $N = \{2, 3, 4\}$) and agents in each episode are assigned different target images. Similarly, each subset has 125K/7.5K/15K samples for `train`/`val`/ `test`.
- **Ad-Hoc-CollaVN**: Two subsets are created for different ad-hoc team size changing situations (*i.e.*, $N\colon 2 \rightarrow 3$ and $N\colon 3 \rightarrow 2$), and each has 125K/7.5K/15K samples for `train`/`val`/`test`. Agents in each episode are assigned a same target image. For $N\colon 2 \rightarrow 3$, its `train` set is the one in *CommonGoal*-CollaVN with $N = 2$, but its `val` and `test` sets are new generated. Each training sample in $N = 2$ contains 2 agents while each val/test sample has 3 agents. The subset of $N\colon 2 \rightarrow 3$ is built in a similar way.

3.3. Evaluation Measures

For performance evaluation, we use four metrics, where the former three are multi-agent augmented versions of SR (Success Rate), DTS (Distance to Success) and SPL (Success weighted by Path Length [2]), and the last one, named SSR (Success weighted by Step Ratio), is new proposed.

SR. Following [89], we consider an episode is successful for an agent if it reaches the target location within 1 m radius. Let $\tau_{k,n} \in \{0, 1\}$ be a binary indicator of success in episode k for agent n. SR is given as: $\frac{1}{KN}\sum_k\sum_n \tau_{k,n}$.

DTS. Let $d^g_{k,n}$ denote the geodesic distance between agent n and its goal location at the end of episode k, and d^s the success threshold (*i.e.*, 1 m). DTS $= \frac{1}{KN}\sum_k\sum_n d^g_{k,n} - d^s$.

SPL. Let $l^g_{k,n}$ be the geodesic distance from starting position to the goal of agent n in episode k, and $l_{k,n}$ the length of the path actually taken by agent n in episode k. We have: SPL $= \frac{1}{KN}\sum_k\sum_n \tau_{k,n} l^g_{k,n} / \max(l^g_{k,n},\ l_{k,n})$.

SSR. Let T denote the allowed maximum time step and $T_{k,n}$ the number of navigation steps used by agent n in episode k. SSR $= \frac{1}{KN}\sum_k\sum_n \tau_{k,n} T / \min(T, T_{k,n})$. As MAVN is a very difficult task, in our experiments (§5.3), we found agents usually need to take many steps to reach the targets, *i.e.*, $l^g_{k,n} \ll l_{k,n}$, making SPL very small and less discriminative. So we design SSR as a complementary.

4. Methodology

4.1. Problem Setup

MARL Background. We consider a system of N agents operating in a common environment as a N-agent extension of partially observable Markov decision processes [51]. It assumes a set [66], $\mathcal{S}$, containing all the states characterising the environment; action spaces $\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_N$, where $\mathcal{A}_n$ is a set of possible actions for the n^{th} agent; and observation spaces $\mathcal{O}_1, \mathcal{O}_2, \cdots, \mathcal{O}_N$ where $\mathcal{O}_n$ contains the observations available to the n^{th} agent. Each agent n receives a private observation $O_n \in \mathcal{O}_n$ (*i.e.*, a partial characterisation of the current state $\mathcal{S}$), and chooses an action according to a stochastic policy $\boldsymbol{\pi}_{\theta_n}\colon \mathcal{O}_n \mapsto \mathcal{A}_n$, parametrized by θ_n. The next state is produced according to a transition function $\mathcal{T}\colon \mathcal{S} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_N \mapsto \mathcal{S}$. Each agent n obtains reward as a function of the state and its action $r_n : \mathcal{S} \times \mathcal{A}_n \mapsto \mathbb{R}$ and aims to maximize its expected return $R_n = \sum_{t=1}^{T} \gamma^t r^t_n(s^t, a^t_n)$, where γ is a discount factor and T is the time horizon.

Task Statement. With our three MAVN tasks, we explore how agents learn from visually-rich environments to achieve collaborative navigation. In *CommonGoal* (*SpecificGoal*), agents learn to cooperatively navigate to same (different) locations, determined by target images $\{g_n \in \mathcal{G}_n\}_{n=1}^N$. In *Ad-hoCoop*, agents are even expected to learn more robust policies $\boldsymbol{\pi}$ that are adaptive to the team size N.

4.2. Memory-Augmented Communication for MAVN

Overview. Our agents are connected by a low-bandwidth communication network without any central controller, through which they can coordinate their actions and behave as a group. As shown in Fig. 2(a), each agent mainly has two components: i) *Map Building Module* (§4.2.1) and ii) *Memory-Augmented Communication Module* (§4.2.2). For i), map building is a crucial step in navigation; an environment representation $\hat{\mathcal{S}}_n$ is estimated for path planning. For ii), each agent n learns to generate useful information for communication and gather messages $w_n \in \mathcal{W}_n$ from other agents simultaneously. It is equipped with a private memory for accurately storing and recalling past communication information. For each agent n, based on its private observation, estimated local map, communication information and target goal, its policy becomes $\boldsymbol{\pi}_{\theta_n}\colon \mathcal{O}_n \times \hat{\mathcal{S}}_n \times \mathcal{W}_n \times \mathcal{G}_n \mapsto \mathcal{A}_n$. See §3.2 for the detailed definitions of $\mathcal{G}$, $\mathcal{A}$, and $\mathcal{O}$. Our whole system is built as trainable end-to-end; neural networks are used as approximations for policies and modules. Corresponding descriptions are presented below.

4.2.1 Map Building Module

Learning environment layouts is crucial for autonomous navigation [63, 92]. Inspired by [14, 15], we equip each agent with a map building module $\mathcal{F}^{\text{map}}$, which online estimates environment structures from agent's past percepts. For each agent n, we denote its pose as $p_n = (x_n, y_n, o_n)$, where (x_n, y_n) indicates its xy co-ordinate and o_n represents its orientation in radians [14]. We set $p^0_n = (0, 0, 0)$ and let each agent build map on its ego-centric coordinate system.

For each agent n at time step t, $\mathcal{F}^{\text{map}}$ takes its local observation O^t_n, current and last pose $p^{t-1:t}_n$, prior predicted map S^{t-1}_n as input, and outputs an updated map:

$$S^t_n = \mathcal{F}^{\text{map}}(O^t_n, p^{t-1:t}_n, S^{t-1}_n). \tag{1}$$

S_n is a $(256{+}4){\times}L{\times}L$ matrix where $L{\times}L$ denotes the map size. The first 256 channels store visual features over all the

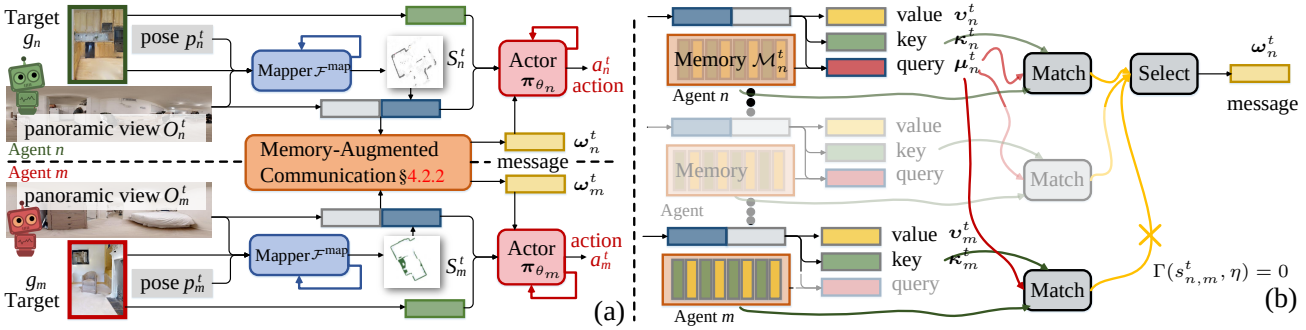

Figure 2: (a) Overview of our method. (b) Detailed illustration of our memory-augmented communication strategy.

explored locations. To save the map size, the grid size is set as 9 m^2(3 m × 3 m) in the physical world. The last four channels include a probability map of obstacle and three binary maps storing explored area, agent past trajectory and current location, with 100 cm^2(10 cm × 10 cm) gride size. At the beginning of an episode, S_n^0 is initialized with all zeros and the agent n is put in the center of S_n^0.

4.2.2 Memory-Augmented Communication Module

In current communication based MARL algorithms [41, 24, 43], all the information generated in t^{th}-round communication are directly abandoned at the beginning of time step t+1 and new messages are generated for $(t+1)^{th}$-round communication. However, some discarded information may be useful in the future. In addition, the communication is mainly about agents' current observations, failing to involving their past experience explicitly. It is difficult if an agent wants to know another agent's status in a previous time step. We instead propose a memory-augmented communication strategy, that allows agent to accurately store and recall their past communication information. It is instantiated as a handshake communication based MARL [21, 54]

Handshake Strategy. Three stages are involved in vanilla handshake communication, *i.e.*, `request`, `match`, and `select`, and each agent acts both *requester* and *supporter*:

- `request`. At the beginning of each time/communication step t, each agent n first compresses its local observation O_n^t to a *query* $\boldsymbol{\mu}_n^t$, *key* $\boldsymbol{\kappa}_n^t$, and *value* $\boldsymbol{v}_n^t$:

$$\boldsymbol{\mu}_n^t = \mathcal{F}^{\text{query}}(O_n^t), \quad \boldsymbol{\kappa}_n^t = \mathcal{F}^{\text{key}}(O_n^t), \quad \boldsymbol{v}_n^t = \mathcal{F}^{\text{value}}(O_n^t), \tag{2}$$

  where $\boldsymbol{\mu}_n^t$ and $\boldsymbol{\kappa}_n^t$ are compact vectors while $\boldsymbol{v}_n^t$ is a high-dimension vector to fully preserve the information of O_n^t. Then the agent n broadcasts $\boldsymbol{\mu}_n^t$ to other agents. As $\boldsymbol{\mu}_n^t$ is small-size, this only causes little bandwidth transmission.
- `match`. Each of other agents derives a matching score $s_{n,m}$ between the received query $\boldsymbol{\mu}_n^t$ and its own key $\boldsymbol{\kappa}_m^t$:

$$s_{n,m}^t = \psi(\boldsymbol{\mu}_n^t, \boldsymbol{\kappa}_m^t), \quad \forall n \neq m, \tag{3}$$

  where ψ is a learnable similarity function. $s_{n,m}$ can be intuitively viewed as the importance of the information provided by the supporter m for the requester n.
- `select`. Some connections can be removed if their matching scores are small. Then the requester n collects information from its temporally connected supporters m^t at time step t:

$$\boldsymbol{\omega}_n^t = \sum\nolimits_{m^t} s_{n,m^t}^t \boldsymbol{v}_{m^t}^t, \quad \forall n \neq m^t, \tag{4}$$

  where $\boldsymbol{v}_{m^t}^t$ is the value vector returned from the supporter m^t, and the integrated message $\boldsymbol{\omega}_n^t$ is used to help the requester n take an action decision a_n^t.

Memory-Augmented Communication. As shown in Fig. 2(b), each agent n maintains a private, external memory $\mathcal{M}_n^t$, which stores all its past generated communication information: $\{\boldsymbol{v}_n^{1:t-1}, \boldsymbol{\kappa}_n^{1:t-1}\}$. Then our memory-augmented handshake communication has the following four stages:

- `request`. At the beginning of each time step t, each agent n generates *query*, *key*, and *value* vectors:

$$\begin{aligned} \boldsymbol{\mu}_n^t &= \mathcal{F}^{\text{query}}\big(O_n^t, S_n^t, \texttt{P}(\boldsymbol{v}_n^{1:t-1})\big), \\ \boldsymbol{\kappa}_n^t &= \mathcal{F}^{\text{key}}(O_n^t, S_n^t), \quad \boldsymbol{v}_n^t = \mathcal{F}^{\text{value}}(O_n^t, S_n^t), \end{aligned} \tag{5}$$

  where $\texttt{P}(\cdot)$ is avg-pooling and both private observation O_n^t and map S_n^t are considered. In addition, past generated communication information, stored in $\mathcal{M}_n^t$, are also involved for generating a more reasonable $\boldsymbol{\mu}_n^t$.
- `match`. Then the agent n broadcasts $\boldsymbol{\mu}_n^t$ to other agents. Each of other agents looks up its current key $\boldsymbol{\kappa}_m^t$, and all the past self-generated keys $\boldsymbol{\kappa}_m^{1:t-1}$ stored in $\mathcal{M}_m^t$:

$$s_{n,m,1:t}^t = \psi(\boldsymbol{\mu}_n^t, \boldsymbol{\kappa}_m^{1:t}), \quad \forall n \neq m, \tag{6}$$

  Then the final matching score and returned message are:

$$s_{n,m}^t = \frac{1}{t}\sum\nolimits_{t'=1}^{t} s_{n,m,t'}^t, \quad \hat{\boldsymbol{v}}_m^t = \frac{1}{t}\sum\nolimits_{t'=1}^{t} s_{n,m,t'}^t \boldsymbol{v}_m^{t'}, \tag{7}$$

  The agent n also computes the correlations between the query $\boldsymbol{\mu}_n^t$ and all its current and past keys $\boldsymbol{\kappa}_n^{1:t}$ stored in $\mathcal{M}_n^t$:

$$s_{n,n,1:t}^t = \psi(\boldsymbol{\mu}_n^t, \boldsymbol{\kappa}_n^{1:t}), \tag{8}$$

Similar to Eq. 7, we have:

$$s_{n,n}^t = \frac{1}{t}\sum_{t'=1}^{t} s_{n,n,t'}^t, \quad \hat{\boldsymbol{\upsilon}}_n^t = \frac{1}{t}\sum_{t'=1}^{t} s_{n,n,t'}^t \boldsymbol{\upsilon}_n^{t'}. \quad (9)$$

- `select`. To filter out some less connections, we apply an activation function $\Gamma(s_{n,m}^t, \eta)$ for each score $s_{n,m}^t$:

$$\Gamma(s_{n,m}^t, \eta) = \begin{cases} 0, & \text{if } s_{n,m}^t \leq \eta \\ s_{n,m}^t, & \text{otherwise} \end{cases} \quad (10)$$

As in [53], only the supporters with non-zero Γ scores are allowed to send messages to the requestor n (*i.e.*, learning *who* to communicate). Moreover, when $\Gamma(s_{n,n}^t, \eta) = 0$, that means the requestor n thinks there is no need to establish any communication at this time step t, (*i.e.*, learning *when* to communicate). We set $\eta = 1/N$ following [53]. If $\Gamma(s_{n,n}^t, \eta) > 0$, the requester n will collect information from itself and other connected supporters m_t (with $\Gamma(s_{n,m^t}^t, \eta) > 0$):

$$\boldsymbol{\omega}_n^t = s_{n,n}^t \hat{\boldsymbol{\upsilon}}_n^t + \sum\nolimits_{m^t \neq n} s_{n,m^t}^t \hat{\boldsymbol{\upsilon}}_{m^t}^t. \quad (11)$$

- `store`. Next, the agent n will store its value and key information $\{\boldsymbol{\upsilon}_n^t, \boldsymbol{\kappa}_n^t\}$ in the memory, getting $\mathcal{M}_n^t$ for the next round communication. In our current implementation, we simply assume the private memory for each agent is large enough to store all its generated communication information over the whole navigation episode. There are several ways to improve the efficiency. One can perform `store` operation in a certain interval of time, and/or build the memory as a stack with fixed capacity, and/or even learn a specific `write` operation to see if it is necessary to store current communication information by comparing its uniqueness with existing memory slots. We put this as a part of our future work.

After t^{th}-round communication, each agent n takes an navigation action through:

$$a_n^t \sim \boldsymbol{\pi}_{\theta_n}(O_n^t, S_n^t, \boldsymbol{\omega}_n^t, g_n). \quad (12)$$

4.3. Fully Decentralized Learning

At each time step t, each agent n executes an individual action $a_n^t \sim \boldsymbol{\pi}_{\theta_n}$, with the joint goal of maximizing the average rewards of all the N agents, *i.e.*, $\frac{1}{N}\sum_{n=1}^{N} R_n$. We adopt an actor-critic model [58] for training and consider it under a fully decentralized MARL setting [93, 82, 94] in which agents are connected via a time-varying communication network. Standard actor-critic methods consist of two models, an actor proposes an action and a critic gives a feedback/evaluation accordingly. In PPO [74], neural networks are used as approximations of both the actor, represented by a policy function $\boldsymbol{\pi}_{\theta_n}$, and its corresponding critic, represented by a value function $V_{\phi_n} : \mathcal{O}_n \mapsto \mathbb{R}$. In order maximize $\mathbb{E}(R_n)$, we have the following policy loss (without clip operation):

$$\begin{aligned} \mathcal{L}(\theta_n) &= \mathbb{E}_{O_n^t, a_n^t}\left[\frac{\boldsymbol{\pi}_{\theta_n}(a_n^t|O_n^t)}{\pi_{\theta_n^{old}}(a_n^t|O_n^t)} A_n^t\right], \\ \mathcal{L}(\phi_n) &= \mathbb{E}_{O_n^t}\left[V_{\phi_n}(O_n^t) - \hat{V}_n^t\right], \end{aligned} \quad (13)$$

where the advantage A_n^t is estimated from V_{ϕ_n} and R_n [73], and $\hat{V}_n^t$ indicates the expected discounted return [87].

In this formulation, as there is no interaction between agents, the policies are learned independently. Previous MARL works [55, 25, 91, 36, 66] mainly address this by training multiple interacting agents in a *Centralized Train and Decentralized Execution* manner. They learn decentralised policies with a centralized critic $V_\phi : \mathcal{O}_1 \times \mathcal{O}_2 \times \cdots \times \mathcal{O}_N \mapsto \mathbb{R}$ which collects all the observations to estimate joint state of all the agents. However, our communication based MARL method is essentially under a *fully decentralized* setting [93, 82, 94], in which agents are connected via a time-varying communication network. As agents can exchange information through communication, we learn an individual value function for each agent, dependent on both local observation and received message:

$$\begin{aligned} \mathcal{L}(\theta_n) &= \mathbb{E}_{O_n^t, a_n^t, \boldsymbol{\omega}_n^t}\left[\frac{\boldsymbol{\pi}_{\theta_n}(a_n^t|O_n^t, \boldsymbol{\omega}_n^t)}{\pi_{\theta_n^{old}}(a_n^t|O_n^t, \boldsymbol{\omega}_n^t)} A_n^t\right], \\ \mathcal{L}(\phi_n) &= \mathbb{E}_{O_n^t, \boldsymbol{\omega}_n^t}\left[V_{\phi_n}(O_n^t, \boldsymbol{\omega}_n^t) - \hat{V}_n^t\right]. \end{aligned} \quad (14)$$

5. Experiments

We conduct experiments on the three MAVN tasks, *i.e.*, *CommonGoal*, *SpecificGoal*, *Ad-hoCoop* on the corresponding subdatasets of CollaVN, *i.e.*, CommonGoal-CollaVN, SpecificGoal-CollaVN, and Ad-Hoc-CollaVN.

5.1. Experimental Setup

Network Details. The embedding of local visual observation O_n is from a pretrained ResNet18 [33] (compressed in to 1024-d). The implementation of map building module $\mathcal{F}^{\text{map}}$ (§4.2.1) mainly follows [14, 15] and the map size is $38.4m \times 38.4m$ in the physical world. Query $\boldsymbol{\mu}$, key $\boldsymbol{\kappa}$, and value $\boldsymbol{\upsilon}$ in §4.2.2 are 256-d, 256-d, 2048-d vectors, respectively. All the functions in §4.2.2 are one-layer MLPs. The policy $\boldsymbol{\pi}$ and value function V are GRUs [19].

Evaluation. For different MALN tasks, we adopt `test` sets of corresponding sub-datasets in CollaVN for benchmarking, measured by (§3.3) SR, DTS, SPL and SSR. The maximum episode length is set as $T = 80$ steps.

Training. Our model was trained using dense reward of geodesic distance reduced to the goal location and constant -0.05 step penalty to collision.

Reproducibility. Our model is implemented in PyTorch and trained on four NVIDIA Tesla V100 GPUs with a 32GB

N	Method	Easy (1.5-3 m)				Medium (3-5 m)				Hard (5-10 m)				Overall			
		SR↑	DTS↓	SSR↑	SPL↑	SR↑	DTS↓	SSR↑	SPL↑	SR↑	DTS↓	SSR↑	SPL↑	SR↑	DTS↓	SSR↑	SPL↑
1	Random	8.31	1.26	0.10	0.02	0.90	3.01	0.01	0.00	0.00	6.49	0.00	0.00	3.07	3.59	0.04	0.01
	IL	37.10	1.22	2.64	0.13	10.10	2.75	0.25	0.05	1.80	5.81	0.07	0.01	16.33	3.26	0.98	0.06
2	IL	37.30	1.22	2.66	0.13	10.00	2.76	0.24	0.05	1.81	5.80	0.07	0.01	16.37	3.26	0.99	0.06
	MARL *w/o* com	39.10	1.19	2.98	0.19	12.10	2.61	0.53	0.06	1.89	5.76	0.07	0.01	17.69	3.18	1.11	0.08
	MARL *w/o* mem	40.21	1.15	3.13	0.19	12.71	2.56	0.32	0.06	1.98	5.71	0.07	0.01	18.29	3.14	1.17	0.09
	Ours	**45.71**	**1.05**	**3.53**	**0.22**	**16.55**	**2.42**	**0.41**	**0.08**	**2.31**	**5.54**	**0.09**	**0.01**	**21.52**	**3.00**	**1.34**	**0.10**
3	MARL *w/o* com	39.00	1.19	2.97	0.19	12.31	2.58	0.31	0.06	1.87	5.75	0.09	0.01	17.72	3.17	1.11	0.08
	MARL *w/o* mem	40.81	1.14	3.21	0.20	12.92	2.54	0.34	0.06	2.01	5.71	0.07	0.01	18.57	3.13	1.20	0.08
	Ours	**47.03**	**1.03**	**3.69**	**0.23**	**17.50**	**2.36**	**0.45**	**0.08**	**2.65**	**5.45**	**0.10**	**0.01**	**22.39**	**2.94**	**1.41**	**0.11**
4	MARL *w/o* com	39.40	1.19	2.96	0.19	11.91	2.60	0.32	0.06	1.87	5.73	0.07	0.01	17.72	3.17	1.12	0.09
	MARL *w/o* mem	41.02	1.14	3.23	0.20	13.10	2.52	0.34	0.06	2.01	5.71	0.07	0.01	18.70	3.12	1.21	0.09
	Ours	**48.01**	**1.01**	**3.75**	**0.24**	**18.40**	**2.31**	**0.48**	**0.09**	**2.91**	**5.40**	**0.11**	**0.01**	**23.10**	**2.90**	**1.45**	**0.11**

Table 2: Performance on *CommonGoal* task with different team sizes ($N = 2, 3, 4$) over our CommonGoal-CollaVN `test`.

N	Method	Easy (1.5-3 m)				Medium (3-5 m)				Hard (5-10 m)				Overall			
		SR↑	DTS↓	SSR↑	SPL↑	SR↑	DTS↓	SSR↑	SPL↑	SR↑	DTS↓	SSR↑	SPL↑	SR↑	DTS↓	SSR↑	SPL↑
2	MARL *w/o* com	39.02	1.19	2.99	0.19	12.00	2.59	0.30	0.06	1.88	5.75	0.07	0.01	17.63	3.18	1.12	0.09
	MARL *w/o* mem	39.86	1.17	3.05	0.19	12.40	2.58	0.31	0.06	1.94	5.72	0.07	0.01	18.06	3.16	1.14	0.09
	Ours	**44.50**	**1.07**	**3.49**	**0.21**	**15.95**	**2.45**	**0.40**	**0.08**	**2.26**	**5.58**	**0.08**	**0.01**	**20.90**	**3.03**	**1.32**	**0.10**
3	MARL *w/o* com	39.08	1.19	2.98	0.19	12.12	2.58	0.30	0.06	1.87	5.74	0.07	0.01	17.69	3.17	1.12	0.09
	MARL *w/o* mem	40.20	1.16	3.08	0.19	12.65	2.56	0.32	0.06	1.96	5.71	0.07	0.01	18.27	3.14	1.16	0.09
	Ours	**45.83**	**1.05**	**3.61**	**0.22**	**17.00**	**2.38**	**0.43**	**0.08**	**2.57**	**5.49**	**0.09**	**0.01**	**21.80**	**2.97**	**1.38**	**0.10**
4	MARL *w/o* com	39.14	1.19	2.95	0.19	12.01	2.59	0.31	0.06	1.88	5.73	0.07	0.01	17.68	3.17	1.11	0.09
	MARL *w/o* mem	40.60	1.15	3.13	0.20	12.85	2.54	0.33	0.06	1.98	5.71	0.07	0.01	18.47	3.13	1.18	0.09
	Ours	**46.79**	**1.03**	**3.71**	**0.23**	**17.82**	**2.33**	**0.46**	**0.09**	**2.80**	**5.42**	**0.10**	**0.01**	**22.47**	**2.93**	**1.42**	**0.11**

Table 3: Performance on *SpecificGoal* task with different team sizes ($N = 2, 3, 4$) over our Specific-CollaVN `test`.

N	Method	Easy (1.5-3 m)				Medium (3-5 m)				Hard (5-10 m)				Overall			
		SR↑	DTS↓	SSR↑	SPL↑	SR↑	DTS↓	SSR↑	SPL↑	SR↑	DTS↓	SSR↑	SPL↑	SR↑	DTS↓	SSR↑	SPL↑
2→3	MARL *w/o* com	39.10	1.19	2.97	0.19	12.3	2.58	0.32	0.06	1.88	5.75	0.07	0.01	17.76	3.17	1.12	0.09
	MARL *w/o* mem	40.50	1.15	3.17	0.20	12.87	2.55	0.34	0.06	1.99	5.72	0.07	0.01	18.45	3.14	1.19	0.09
	Ours	**46.66**	**1.03**	**3.68**	**0.22**	**17.30**	**2.37**	**0.45**	**0.08**	**2.60**	**5.47**	**0.10**	**0.01**	**22.17**	**2.96**	**1.41**	**0.11**
3→2	MARL *w/o* com	39.00	1.19	2.98	0.19	12.20	2.60	0.30	0.06	1.89	5.76	0.07	0.01	17.70	3.18	1.12	0.09
	MARL *w/o* mem	39.90	1.17	3.10	0.19	12.40	2.58	0.31	0.06	1.96	5.71	0.07	0.01	18.08	3.15	1.16	0.09
	Ours	**45.20**	**1.06**	**3.51**	**0.22**	**16.10**	**2.43**	**0.41**	**0.08**	**2.27**	**5.58**	**0.08**	**0.01**	**21.19**	**3.02**	**1.33**	**0.10**

Table 4: Performance on *Ad-hoCoop* task with two ad-hoc team-play settings ($N:2\rightarrow3$ and $N:3\rightarrow2$) over our Ad-Hoc-CollaVN `test`.

memory per-card. To reveal full details of our algorithm, our implementations are released.

5.2. Baseline

Following baselines are used in our experiments (their sub-modules are the same with ours unless specified):

- **Random walk** is the simplest heuristic for navigation: agent randomly selects an action at each step.
- **IL** learns an off policy by imitation learning (IL). One agent is trained and several copies are used for testing.
- **MARL *w/o* com** learns target-driven MARL policy without explicit communication, using IPPO [72].
- **MARL *w/o* mem** can be viewed as a variant of our model without using memory.

5.3. Experimental Results

Performance on *CommonGoal* Task. For the settings with different agent numbers, *i.e.*, $N = 2, 3, 4$ on the CommonGoal-CollaVN sub-dataset, all the benchmarked models are trained on the corresponding `train` sets and tested on the corresponding `test` sets. The results are summarized in Table 2. As seen, our model outperforms over all other baselines across all the metrics. Interestingly, we can also observe IL based baseline, *i.e.*, IL-M, even shows comparable performance with a MARL baseline, *i.e.*, MARL *w/o* com, revealing the difficulty of this task.

Performance on *SpecificGoal* Task. Also, for different agent numbers on the SpecificGoal-CollaVN sub-dataset, we train and evaluate the models on the corresponding `train` and `test` sets. From Table 3 we can find that our method sill gets better performance. However, compared with the results in Table 2, all the methods suffer from performance drop, which indicates that it is harder to learn communication when agents perform different tasks.

Performance on *Ad-hoCoop* Task. For $N: 2 \rightarrow 3$ setting in the Ad-Hoc-CollaVN dataset, each of `train` scenes contains two agents while each of `test` scenes have three agents. Similarly, for $N: 3 \rightarrow 2$ setting, each of `train` scenes contains three agents while each of `test` scenes have two agents. For the two settings, all the methods are

trained and test on the corresponding `train` and `test` sets. The results in Table 4 show again our model performs better on the two ad-hoc team play settings, mainly due to our fully decentralized learning strategy.

Effect on Memory-Based Communication. Compared with MARL *w/o* mem, our model shows significantly better performance over all the three MAVN settings. MARL *w/o* mem only shows limited improvements over MARL *w/o* com. These observations suggest the importance of long-term memory in MAVN.

6. Conclusion

A new dataset was introduced for multi-agent collaborative navigation in complex environments. A memory-based communication model was presented to address the reuse of past communication information and facilitate cooperation. Our experiments showed that there is still huge room for further improvement. This work is expected to inspire more future efforts towards this promising direction.

References

[1] Stefano V Albrecht and Peter Stone. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence*, 258:66–95, 2018. 3

[2] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018. 5, 15

[3] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, 2018. 1, 2, 3

[4] Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *CVPR*, 2016. 1, 3

[5] Trapit Bansal, Jakub Pachocki, Szymon Sidor, Ilya Sutskever, and Igor Mordatch. Emergent complexity via multi-agent competition. *arXiv preprint arXiv:1710.03748*, 2017. 3

[6] Samuel Barrett, Peter Stone, and Sarit Kraus. Empirical evaluation of ad hoc teamwork in the pursuit domain. In *AAMAS*, pages 567–574, 2011. 2

[7] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019. 1, 3

[8] Johann Borenstein and Yoram Koren. The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, 7(3):278–288, 1991. 2

[9] Michael Bowling and Peter McCracken. Coordination and adaptation in impromptu teams. In *AAAI*, 2005. 3

[10] Simon Brodeur, Ethan Perez, Ankesh Anand, Florian Golemo, Luca Celotti, Florian Strub, Jean Rouat, Hugo Larochelle, and Aaron Courville. Home: A household multimodal environment. *arXiv preprint arXiv:1711.11017*, 2017. 3

[11] Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008. 1, 3

[12] Rodrigo Canaan, Julian Togelius, Andy Nealen, and Stefan Menzel. Diverse agents for ad-hoc cooperation in hanabi. In *IEEE Conference on Games*, 2019. 2, 3

[13] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niebner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. In *3DV*, 2018. 1, 3

[14] Devendra Singh Chaplot, Dhiraj Gandhi, Abhinav Gupta, and Ruslan Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *arXiv preprint arXiv:2007.00643*, 2020. 2, 5, 7

[15] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *ICLR*, 2020. 3, 5, 7, 13

[16] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural topological slam for visual navigation. In *CVPR*, 2020. 1, 2, 3

[17] Changan Chen, Unnat Jain, Carl Schissler, Sebastia Vicenc Amengual Gari, Ziad Al-Halah, Vamsi Krishna Ithapu, Philip Robinson, and Kristen Grauman. Soundspaces: Audio-visual navigation in 3d environments. In *ECCV*, 2020. 2, 3

[18] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *CVPR*, 2019. 2, 3

[19] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 7

[20] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *CVPR*, 2018. 1

[21] Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. Tarmac: Targeted multi-agent communication. In *ICML*, 2019. 2, 3, 6

[22] Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, Dustin Schwenk, Eli VanderBilt, Matthew Wallingford, et al. Robothor: An open simulation-to-real embodied ai platform. In *CVPR*, 2020. 1, 2, 3

[23] Zhiwei Deng, Karthik Narasimhan, and Olga Russakovsky. Evolving graphical planner: Contextual global planning for vision-and-language navigation. *arXiv preprint arXiv:2007.05655*, 2020. 3

[24] Jakob Foerster, Ioannis Alexandros Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *NeurIPS*, 2016. 1, 2, 3, 4, 6

[25] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *AAAI*, 2018. 7

[26] Jakob Foerster, Nantas Nardelli, Gregory Farquhar, Triantafyllos Afouras, Philip HS Torr, Pushmeet Kohli, and Shimon Whiteson. Stabilising experience replay for deep multi-agent reinforcement learning. In *ICML*, 2017. 3

[27] Dieter Fox, Wolfram Burgard, Hannes Kruppa, and Sebastian Thrun. A probabilistic approach to collaborative multi-robot localization. *Autonomous robots*, 8(3):325–344, 2000. 2

[28] Ben Goertzel and Cassio Pennachin. *Artificial general intelligence*, volume 2. Springer, 2007. 1

[29] Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *International Conference on Autonomous Agents and Multiagent Systems*, 2017. 3

[30] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *CVPR*, 2017. 3

[31] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2

[32] Matthew Hausknecht, Prannoy Mupparaju, Sandeep Subramanian, Shivaram Kalyanakrishnan, and Peter Stone. Half field offense: An environment for multiagent learning and ad hoc teamwork. In *AAMAS Adaptive Learning Agents (ALA) Workshop*, 2016. 3

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 7

[34] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33(6):750–797, 2019. 2, 3

[35] Zhang-Wei Hong, Shih-Yang Su, Tzu-Yun Shann, Yi-Hsiang Chang, and Chun-Yi Lee. A deep policy inference q-network for multi-agent systems. *arXiv preprint arXiv:1712.07893*, 2017. 3

[36] Shariq Iqbal and Fei Sha. Actor-attention-critic for multi-agent reinforcement learning. In *ICML*, 2019. 7

[37] Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019. 1, 3

[38] Unnat Jain, Luca Weihs, Eric Kolve, Ali Farhadi, Svetlana Lazebnik, Aniruddha Kembhavi, and Alexander Schwing. A cordial sync: Going beyond marginal policies for multi-agent embodied tasks. In *ECCV*, 2020. 3, 4

[39] Unnat Jain, Luca Weihs, Eric Kolve, Mohammad Rastegari, Svetlana Lazebnik, Ali Farhadi, Alexander G Schwing, and Aniruddha Kembhavi. Two body problem: Collaborative visual task completion. In *CVPR*, 2019. 3, 4

[40] Jiechuan Jiang, Chen Dun, Tiejun Huang, and Zongqing Lu. Graph convolutional reinforcement learning. In *ICLR*, 2020. 3

[41] Jiechuan Jiang and Zongqing Lu. Learning attentional communication for multi-agent cooperation. In *NeurIPS*, 2018. 3, 4, 6

[42] Yiannis Kantaros, Michalis Thanou, and Anthony Tzes. Distributed coverage control for concave areas by a heterogeneous robot–swarm with visibility sensing constraints. *Automatica*, 53:195–207, 2015. 1

[43] Daewoo Kim, Sangwoo Moon, David Hostallero, Wan Ju Kang, Taeyoung Lee, Kyunghwan Son, and Yung Yi. Learning to schedule communication in multi-agent reinforcement learning. *arXiv preprint arXiv:1902.01554*, 2019. 2, 4, 6

[44] Dongsung Kim and Ramakant Nevatia. Symbolic navigation with a generic map. *Autonomous Robots*, 6(1):69–88, 1999. 2

[45] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017. 1, 3, 4

[46] Ilya Kostrikov. Pytorch implementations of reinforcement learning algorithms. https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail, 2018. 14

[47] Benjamin Kuipers. Modeling spatial knowledge. *Cognitive Science*, 2(2):129–153, 1978. 2

[48] Helene Emilie Landemore. *Democratic reason: Politics, collective intelligence, and the rule of the many*. PhD thesis, Harvard University, 2008. 1

[49] Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. Multi-agent cooperation and the emergence of (natural) language. In *ICLR*, 2017. 1, 2, 3

[50] Lisa Lee, Emilio Parisotto, Devendra Singh Chaplot, Eric Xing, and Ruslan Salakhutdinov. Gated path planning networks. In *ICML*, 2018. 3

[51] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings*, pages 157–163. 1994. 5

[52] Yugang Liu and Goldie Nejat. Multirobot cooperative learning for semiautonomous control in urban search and rescue applications. *Journal of Field Robotics*, 33(4):512–536, 2016. 1

[53] Yen-Cheng Liu, Junjiao Tian, Nathaniel Glaser, and Zsolt Kira. When2com: multi-agent perception via communication graph grouping. In *CVPR*, 2020. 1, 2, 3, 7

[54] Yen-Cheng Liu, Junjiao Tian, Chih-Yao Ma, Nathan Glaser, Chia-Wen Kuo, and Zsolt Kira. Who2com: Collaborative perception via learnable handshake communication. In *ICRA*, 2020. 2, 3, 6

[55] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv preprint arXiv:1706.02275*, 2017. 1, 7

[56] Laetitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. *Knowledge Engineering Review*, 27(1):1–31, 2012. 3

[57] Suruz Miah, Bao Nguyen, François-Alex Bourque, and Davide Spinello. Nonuniform deployment of autonomous agents in harbor-like environments. *Unmanned Systems*, 2(04):377–389, 2014. 1

[58] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *ICML*, 2016. 7

[59] Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. In *AAAI*, 2018. 3

[60] Adithyavairavan Murali, Tao Chen, Kalyan Vasudev Alwala, Dhiraj Gandhi, Lerrel Pinto, Saurabh Gupta, and Abhinav Gupta. Pyrobot: An open-source robotics framework for research and benchmarking. *arXiv preprint arXiv:1906.08236*, 2019. 4

[61] Gregory Palmer, Karl Tuyls, Daan Bloembergen, and Rahul Savani. Lenient multi-agent deep reinforcement learning. *arXiv preprint arXiv:1707.04402*, 2017. 3

[62] Liviu Panait and Sean Luke. Cooperative multi-agent learning: The state of the art. *Autonomous Agents and Multi-Agent Systems*, 11(3):387–434, 2005. 3

[63] Emilio Parisotto and Ruslan Salakhutdinov. Neural map: Structured memory for deep reinforcement learning. In *ICLR*, 2019. 3, 5

[64] Peng Peng, Ying Wen, Yaodong Yang, Quan Yuan, Zhenkun Tang, Haitao Long, and Jun Wang. Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games. *arXiv preprint arXiv:1703.10069*, 2017. 1, 2, 3

[65] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *ICRA*, 2018. 1

[66] Emanuele Pesce and Giovanni Montana. Improving coordination in small-scale multi-agent deep reinforcement learning through memory-driven communication. *Machine Learning*, pages 1–21, 2020. 3, 5, 7

[67] Maithra Raghu, Alex Irpan, Jacob Andreas, Bobby Kleinberg, Quoc Le, and Jon Kleinberg. Can deep reinforcement learning solve erdos-selfridge-spencer games? In *ICML*, 2018. 3

[68] Roberta Raileanu, Emily Denton, Arthur Szlam, and Rob Fergus. Modeling others using oneself in multi-agent reinforcement learning. In *ICML*, 2018. 3

[69] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *ICML*, 2018. 1

[70] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder De Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043*, 2019. 1, 3

[71] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *ICCV*, 2019. 1, 3, 4

[72] Christian Schroeder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip HS Torr, Mingfei Sun, and Shimon Whiteson. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020. 8

[73] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015. 7

[74] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 7, 14

[75] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martín-Martín, Linxi Fan, Guanzhi Wang, Shyamal Buch, Claudia D'Arpino, Sanjana Srivastava, Lyne P Tchapmi, et al. igibson, a simulation environment for interactive tasks in large realisticscenes. *arXiv preprint arXiv:2012.02924*, 2020. 1, 3, 4

[76] Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. Learning when to communicate at scale in multiagent cooperative and competitive tasks. In *ICLR*, 2020. 2, 3

[77] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. In *CVPR*, 2017. 1, 3

[78] Yiannis Stergiopoulos, Michalis Thanou, and Anthony Tzes. Distributed collaborative coverage-control schemes for non-convex domains. *IEEE Transactions on Automatic Control*, 60(9):2422–2427, 2015. 1

[79] Peter Stone and Manuela Veloso. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8(3):345–383, 2000. 1

[80] Kristina Striegnitz, Alexandre Denis, Andrew Gargett, Konstantina Garoufi, Alexander Koller, and Mariët Theune. Report on the second challenge on generating instructions in virtual environments (give-2.5). In *European Workshop on Natural Language Generation*, 2011. 2

[81] Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. Learning multiagent communication with backpropagation. *arXiv preprint arXiv:1605.07736*, 2016. 2, 3

[82] Wesley Suttle, Zhuoran Yang, Kaiqing Zhang, Zhaoran Wang, Tamer Basar, and Ji Liu. A multi-agent off-policy actor-critic algorithm for distributed reinforcement learning. *arXiv preprint arXiv:1903.06372*, 2019. 7

[83] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *ICML*, 1993. 2, 3

[84] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002. 2

[85] Hanqing Wang, Wenguan Wang, Wei Liang, Caiming Xiong, and Jianbing Shen. Structured scene memory for vision-language navigation. In *CVPR*, 2021. 3

[86] Hanqing Wang, Wenguan Wang, Tianmin Shu, Wei Liang, and Jianbing Shen. Active visual information gathering for vision-language navigation. In *ECCV*, 2020. 2

[87] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 1992. 7

[88] Yi Wu, Yuxin Wu, Georgia Gkioxari, and Yuandong Tian. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*, 2018. 1

[89] Fei Xia, William B Shen, Chengshu Li, Priya Kasimbeg, Micael Edmond Tchapmi, Alexander Toshev, Roberto Martín-Martín, and Silvio Savarese. Interactive gibson benchmark: A benchmark for interactive navigation in cluttered environments. *IEEE Robotics and Automation Letters*, 5(2):713–720, 2020. 2, 3, 4, 5

[90] Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *CVPR*, 2018. 1, 3

[91] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of mappo in cooperative, multi-agent games, 2021. 7

[92] Jingwei Zhang, Lei Tai, Joschka Boedecker, Wolfram Burgard, and Ming Liu. Neural slam: Learning to explore with external memory. *arXiv preprint arXiv:1706.09520*, 2017. 3, 5

[93] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *arXiv preprint arXiv:1911.10635*, 2019. 7

[94] Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Basar. Fully decentralized multi-agent reinforcement learning with networked agents. In *ICML*, 2018. 7

[95] Tianjun Zhang, Huazhe Xu, Xiaolong Wang, Yi Wu, Kurt Keutzer, Joseph E Gonzalez, and Yuandong Tian. Multi-agent collaboration via reward attribution decomposition. *arXiv preprint arXiv:2010.08531*, 2020. 3

[96] Lianmin Zheng, Jiacheng Yang, Han Cai, Ming Zhou, Weinan Zhang, Jun Wang, and Yong Yu. Magent: A many-agent reinforcement learning platform for artificial collective intelligence. In *AAAI*, 2018. 1, 3

[97] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *ICRA*, 2017. 2

N	Difficulty	CommonGoal		
		Train	Val	Test
2	Easy (1.5-3m)	125K	2.5K	5K
	Medium (3-5m)	125K	2.5K	5K
	Hard (5-10m)	125K	2.5K	5K
3	Easy (1.5-3m)	125K	2.5K	5K
	Medium (3-5m)	125K	2.5K	5K
	Hard (5-10m)	125K	2.5K	5K
4	Easy (1.5-3m)	125K	2.5K	5K
	Medium (3-5m)	125K	2.5K	5K
	Hard (5-10m)	125K	2.5K	5K

Table 5: **Dataset Statistics** on CommonGoal-CollaVN.

N	Difficulty	SpecificGoal		
		Train	Val	Test
2	Easy (1.5-3m)	125K	2.5K	5K
	Medium (3-5m)	125K	2.5K	5K
	Hard (5-10m)	125K	2.5K	5K
3	Easy (1.5-3m)	125K	2.5K	5K
	Medium (3-5m)	125K	2.5K	5K
	Hard (5-10m)	125K	2.5K	5K
4	Easy (1.5-3m)	125K	2.5K	5K
	Medium (3-5m)	125K	2.5K	5K
	Hard (5-10m)	125K	2.5K	5K

Table 6: **Dataset Statistics** on SpecificGoal-CollaVN.

N	Difficulty	Ad-HoCoop		
		Train	Val	Test
Add ($N:2\rightarrow3$)	Easy(1.5-3m)	125K	2.5K	5K
	Medium(3-5m)	125K	2.5K	5K
	Hard(5-10m)	125K	2.5K	5K
Remove ($N:3\rightarrow2$)	Easy(1.5-3m)	125K	2.5K	5K
	Medium(3-5m)	125K	2.5K	5K
	Hard(5-10m)	125K	2.5K	5K

Table 7: **Dataset Statistics** on Ad-Hoc-CollaVN.

Indicator	2 (3-5m)	3 (3-5m)	4 (3-5m)
SR	18.58	21.23	24.32
SPL	0.08	0.09	0.10
SSR	0.69	0.75	0.80

Table 8: **SPL *vs* SSR.** Performance of our model on *CommonGoal* task with medium difficulty and different team sizes ($N=2,3,4$).

In the supplementary material, we first elaborate on our dataset construction (§A). Then, more implementation details and discussions on evaluation metric designs are provided in §B and §C. Finally, we present some representative visual results in §D.

A. Details of CollaVN

Our CollaVN dataset has three subdatasets, corresponding to the three MAVN tasks: CommonGoal, SpecificGoal and Ad-HoCoop.

CommonGoal-CollaVN: It contains three subsets, which are created for different agent numbers (*i.e.*, $N=\{2,3,4\}$), and agents in each episode are assigned a same target image. For each subset of different agent number, we randomly sample 5K episodes per training scene. For each scene in val/test split, we generate 0.5K/1K episodes per configuration, *i.e.*, {*easy* (1.5-3 m), *medium* (3-5 m), *hard* (5-10 m)}. The distance range means that the initial start and target position of all agents are within this limit. Finally, each subset has 125K/7.5K/15K samples for `train`/`val`/`test`, *i.e.*, $25\times5\text{K}/3\times5\times0.5\text{K}/3\times5\times1\text{K}=125\text{K}/7.5\text{K}/15\text{K}$. Detailed statistics of CommonGoal-CollaVN subdataset are shown in Table 5.

SpecificGoal-CollaVN: Three subsets are created for different agent numbers (*i.e.*, $N=\{2,3,4\}$), but agents in each episode are assigned different target images. Similarly, each subset has 125K/7.5K/15K samples for `train`/`val`/`test`. Detailed statistics of SpecificGoal-CollaVN subdataset are shown in Table 6.

Ad-Hoc-CollaVN: Two subsets are created for different ad-hoc team size changing situations (*i.e.*, $N:2\rightarrow3$ and $N:3\rightarrow2$), and each has 125K/7.5K/15K samples for `train`/`val`/`test`. Agents in each episode are assigned a same target image. For $N:2\rightarrow3$, its `train` set is the one in *CommonGoal*-CollaVN with $N=2$, but its `val` and `test` sets are new generated. Each training sample in $N=2$ contains 2 agents while each val/test sample has 3 agents. The subset of $N:3\rightarrow2$ is built in a similar way. Detailed statistics of Ad-Hoc-CollaVN subdataset are shown in Table 7.

B. Implementation Details

B.1. Reward Structure

During training, rewards are provided to all agents individually at every step. The reward includes two terms: **i)** $+1.00\times\triangle d$, where d is the geodesic distance between agent's current location and target location, and $\triangle d$ refers to the distance change after adapting a navigation action at this step. $+1.00$ is the weight of goal-driven reward. **ii)** A penalty of -0.05 whenever the agent collides with environment or other agents. The maximum total reward achievable for a single agent is $+D$, where D is the initial distance between agent and goal, *i.e.*, reaching the goal through the shortest path without any collision. Our models are trained to maximize the expected discounted cumulative gain with a discounting factor $\gamma=0.99$.

B.2. Hyperparameter Details

We use PyTorch for implementing and training our model. As for map building, we follow [15], maintain a FIFO memory of size 5000 for training. After one step in each thread, we perform 10 updates to the map building module with a batch size of 64.

During training, we train our agents with 25 parallel

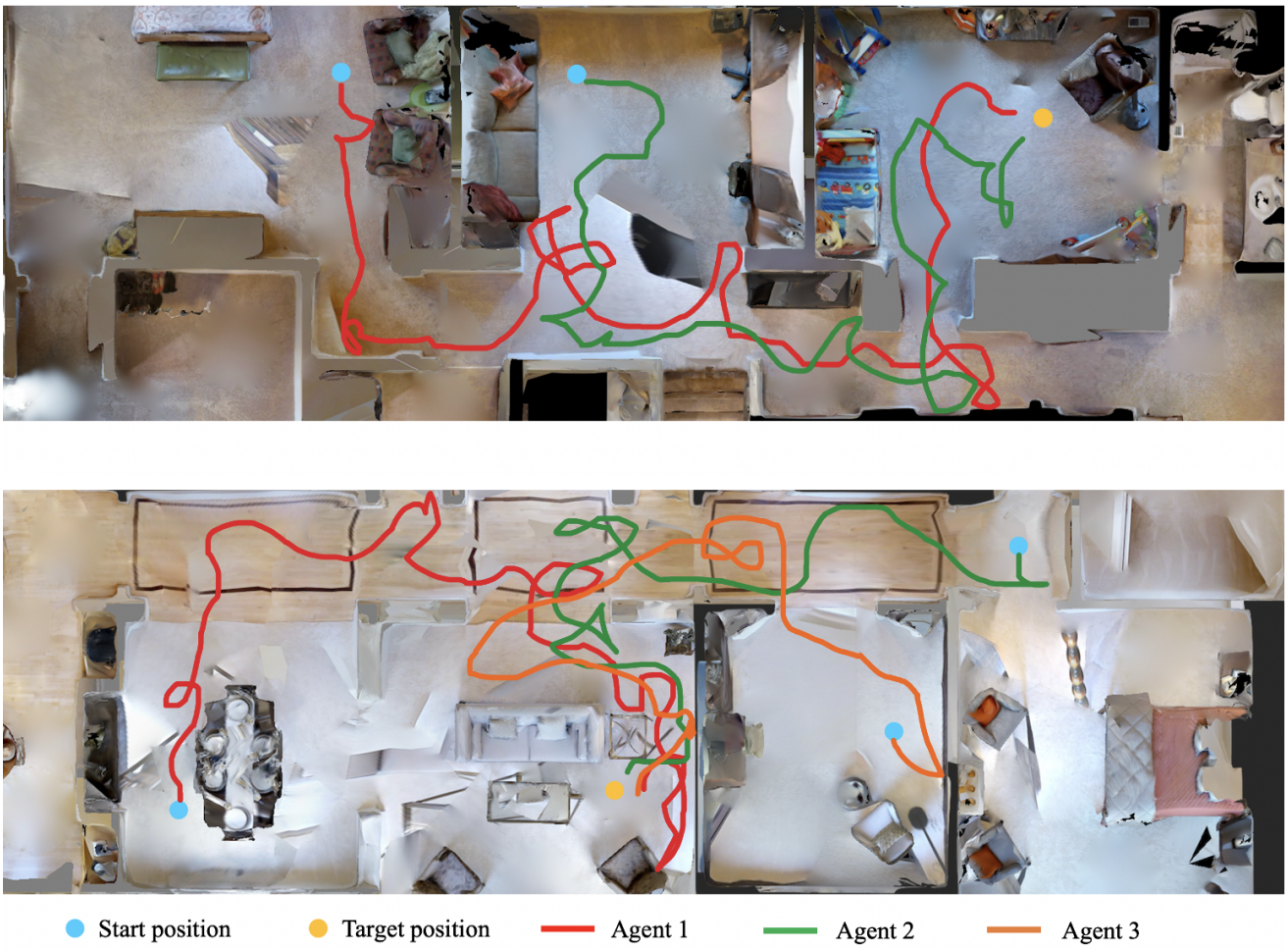

Figure 3: Trajectory of two or three agents on CommonGoal task (**Top**: two agents **Down**: three agents).

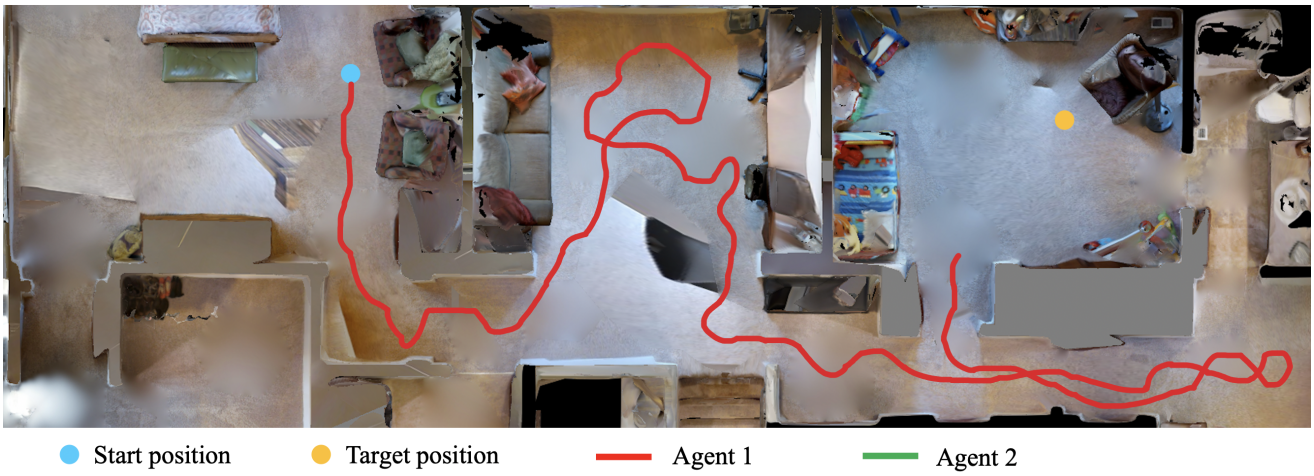

Figure 4: Trajectory of single agent (same initialization with two agents of CommonGoal task).

threads, with each thread using one scene in the training set. We use Proximal Policy Optimization (PPO) [74], with 5 mini-batches, and 8 epochs in each PPO update. Our PPO implementation is based on [46]. We use Adam optimizer

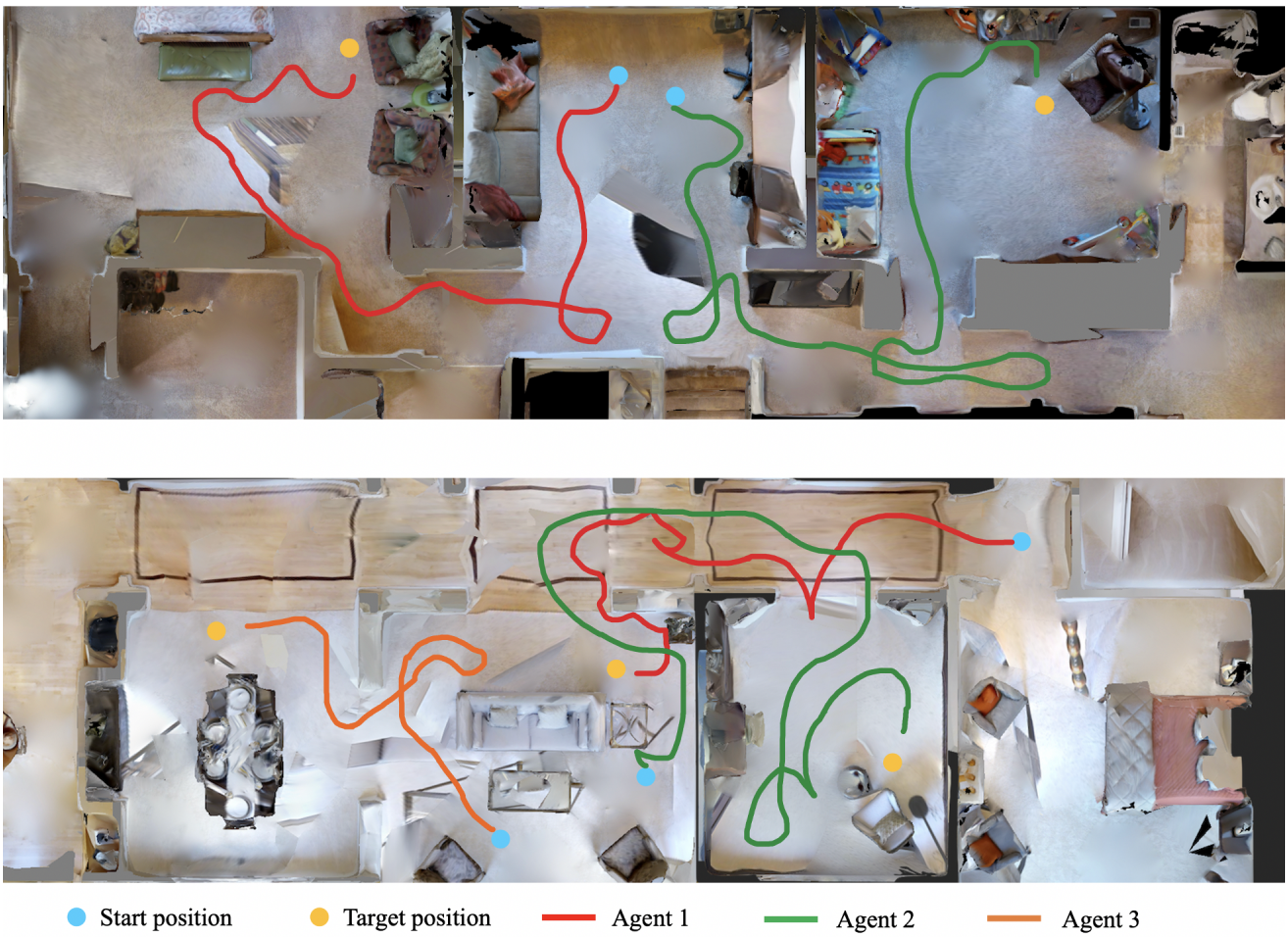

Figure 5: Trajectory of two or three agents on Specific task (**Top**: two agents **Down**: three agents).

with a learning rate of 0.00001, a discount factor of γ = 0.99, an entropy coefficient of 0.001, value loss coefficient of 0.5 for training.

C. Evaluation Details

Why SSR instead of SPL? SPL was introduced in [2]:

$$SPL = \frac{1}{NK}\sum_{k=1}^{K}\sum_{n=1}^{N}\tau_{k,n}\frac{l^g_{k,n}}{\max(l^g_{k,n},\ l_{k,n})}. \tag{15}$$

where N is the agent number, K equals the number of test episodes, $\tau_{k,n} \in \{0, 1\}$ is a binary indicator of success in episode k for agent n, $l^g_{k,n}$ is the geodesic distance from starting position to the goal, and $l_{k,n}$ the length of the path actually taken by agent.

The value of $l^g_{k,n}/\max(l^g_{k,n},\ l_{k,n})$ is always less than one ($<$1). If we perform a long-term or difficult navigation task, the success rate is usually low and the navigation path is very long. In this case, SPL will be very small and lose discriminability. This phenomenon can be observed in Table 8, which shows the performance of our model on CommonGoal task with medium difficulty and different team sizes. SPL basically stays still with the success rate changing.

To resolve this problem, we introduce SSR (Success weighted by Step Ratio):

$$SSR = \frac{1}{NK}\sum_{k=1}^{K}\sum_{n=1}^{N}\tau_{k,n}\frac{T}{\min(T,\ T_{k,n})}. \tag{16}$$

T denotes the allowed maximum time step and $T_{k,n}$ the number of navigation steps used by agent n in episode k. As shown in Table 8, SSR is more discriminative than SPL.

D. Qualitative Results

In this section, we provide more intuitive results of our model.

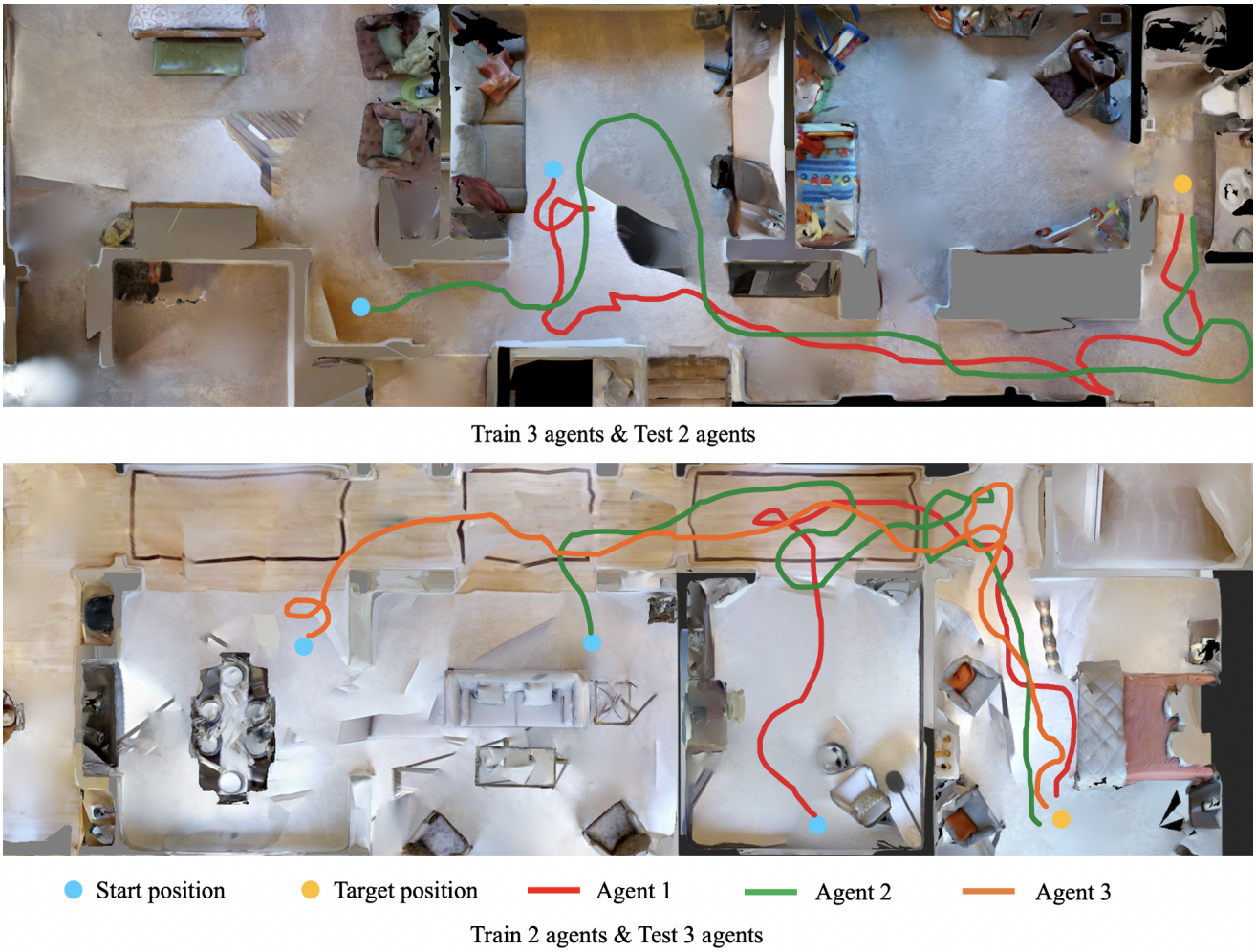

Figure 6: Trajectory of add and remove on Ad-HoCoop task (**Top**: remove ($N : 3 \rightarrow 2$) **Down**: add ($N : 2 \rightarrow 3$)).

D.1. Trajectory Result

The visualization of navigation trajectories of three task settings with different team size ($N = 2, 3$) are shown in Fig. 3, Fig. 5, and Fig. 6. The start and target locations are represented by blue circles and orange circles, respectively. From these figures, we can observe that the agents trained with our method is able to learn a robust policy to collaboratively explore the environment and successfully reaches the target location. From Fig. 6, we find that our method can adapt to different team sizes at test times well.

D.2. Effect of Communication

To show the effectiveness of communication in MVAN task, we demonstrate the trajectory of a single agent system in Fig. 4. We compare the effect by deploying this agents for a particular initialization of an episode, *i.e.*, the scene, agents' start location and target location are same as CommonGoal task with team size of two (Top figure in Fig. 3). As shown in Fig. 4, the single agent failed to reach the target position within maximum time step (it spends too much time on exploring a wrong room). However, in CommonGoal task, we find that implicit communication can allow agents to avoid this wrong exploration as much as possible based on the experience of other agents and achieve the task faster.