Hyp2Nav: Hyperbolic Planning and Curiosity for Crowd Navigation

Preprint · July 2024

DOI: 10.48550/arXiv.2407.13567

4 authors, including:

Alessandro Flaborea
Sapienza University of Rome
12 PUBLICATIONS **60** CITATIONS
SEE PROFILE

Pascal Mettes
University of Amsterdam
82 PUBLICATIONS **1,499** CITATIONS
SEE PROFILE

Hyp^{2}Nav: Hyperbolic Planning and Curiosity for Crowd Navigation

Guido M. D'Amely di Melendugno*[1] Alessandro Flaborea*[1] Pascal Mettes[2] Fabio Galasso[1]

***Abstract*— Autonomous robots are increasingly becoming a strong fixture in social environments. Effective crowd navigation requires not only safe yet fast planning, but should also enable interpretability and computational efficiency for working in real-time on embedded devices. In this work, we advocate for hyperbolic learning to enable crowd navigation and we introduce Hyp^{2}Nav. Different from conventional reinforcement learning-based crowd navigation methods, Hyp^{2}Nav leverages the intrinsic properties of hyperbolic geometry to better encode the hierarchical nature of decision-making processes in navigation tasks. We propose a hyperbolic policy model and a hyperbolic curiosity module that results in effective social navigation, best success rates, and returns across multiple simulation settings, using up to 6 times fewer parameters than competitor state-of-the-art models. With our approach, it becomes even possible to obtain policies that work in 2-dimensional embedding spaces, opening up new possibilities for low-resource crowd navigation and model interpretability. Insightfully, the internal hyperbolic representation of Hyp^{2}Nav correlates with how much *attention* the robot pays to the surrounding crowds, e.g. due to multiple people occluding its pathway or to a few of them showing colliding plans, rather than to its own planned route.**

I. Introduction

Crowd navigation is paramount to deploying robots in environments shared with people [1]. Most recently, robots are finding applications in hospitals, navigating busy corridors to transport medications, in robot postal services, restaurants and hotels. Across these applications, robots require advanced social navigation capabilities, which are yet to be acquired. Thus, the ubiquity of robots in settings traditionally occupied by humans has spurred a significant body of research [2], [3], [4], [5], [6] focused on enhancing their autonomy and interaction capabilities to ensure a safe human-robot co-existence. Among the compelling challenges stemming from this paradigm, the problem of robot navigation in crowded environments is critical to guarantee safe and seamless integration [7], [8].

For social navigation in crowds, the challenge is to guarantee a minimum travel time while ensuring maximum comfort/safety for human co-inhabitants. This task is further complicated by computational constraints, the unpredictability of human behavior, and the need for an optimal representation of decision-making processes, which currently remain open problems. We observe that the efficiency requirement conflicts with the high-dimensional representations typically employed by (Euclidean) Deep Reinforcement Learning (DRL). Such high-dimensional representations are currently needed for effectively modeling states in crowd navigation but lead to excessive memory usage. Also, the complexity of human behavior may result in abrupt changes in the plans of the crowds and, eventually, collisions, which demands increased interpretability of the robot's decisions. Finally, a DRL path decision process resembles a hierarchical tree of the agent's internal statuses, and its representation tends to suffer from distortion in conventional Euclidean latent spaces [9]. State-of-the-art works employ conventional neural networks, which are de facto operating in Euclidean space [8], [10]. Here, we argue that the global state of the environment and the inherent Markov Decision Process (MDP) are intrinsically graphs, inconvenient for Euclidean spaces [11]. Hyperbolic learning holds significant potential in this respect, which the state-of-the-art [8], [10] does not leverage.

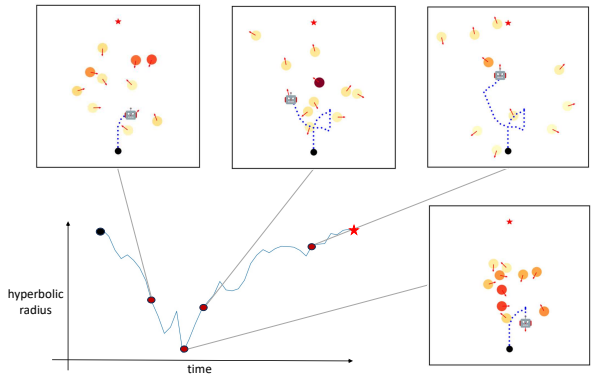

Fig. 1. **Visualizing the hyperbolic radius**, the magnitude of the hyperbolic embeddings. Optimizing policies for crowd navigation in hyperbolic space results in embeddings that correlate with the robot's uncertainty in navigating within the obstacles in the scene. In the bottom left plot, the value of the hyperbolic radius (*y-axis*) is depicted over time (*x-axis*) in a typical rollout. The radius decreases when encountering obstacles directed towards the robot (*top-left box*), indicating reduced confidence in the robot's decisions. Notice that people are color-coded circle, the *red*-er, the larger is the robot attention to them (instead of self). Conversely, as the robot successfully navigates through challenging scenarios (*bottom-right*), the hyperbolic radius increases (*top center*), reflecting improved confidence and more straightforward decision-making toward the final goal ★ (*top right*).

We propose a novel hyperbolic path-planner with hyperbolic curiosity within a deep reinforcement learning framework, which we dub *Hyp^{2}Nav*. Thanks to hierarchically-organized decision processes, endowed by the hyperbolic model, *Hyp^{2}Nav* increases success rates and rewards at a fraction of the parameter count. Our approach is inspired by advances in Hyperbolic Neural Networks (HNNs), which have recently garnered attention in computer vision, graph networks, recommender systems, and more for embedding

*Authors contributed equally.

[1]Sapienza University of Rome, Italy, email: lastname@di.uniroma1.it
[2]University of Amsterdam, Netherlands, email: P.S.M.lastname@uva.nl

hierarchical data due to their inherent property of embedding tree-like structures with minimal distortion. We advocate for adopting hyperbolic latent spaces to represent the states in an MDP, proposing a hyperbolic policy module, *HyperPlanner*, to operate entirely in low-dimensional latent hyperbolic spaces, for the first time exploiting extremely lightweight *2-dimensional* representations. We furthermore present *HyperCuriosity*, to enforce exploration during training in hyperbolic space. Coupling the benefits of the recent *Intrinsic Curiosity Module* (ICM) [12] with the novel hyperbolic states representations, HyperCuriosity reports increased exploration, especially in the early training episodes, speeding up convergence to the optimal policy. Hyp^{2}Nav yields accurate and generalizable policies, as we demonstrate in comparative evaluation.

We also investigate the inner working of Hyp^{2}Nav, identifying interesting properties of the hyperbolic radius, i.e., the magnitude of the hyperbolic embedding. We show in our experiments that the radius of the representations correlates with the current navigation's complexity, the collision's danger, and, thus, the robot's uncertainty. Fig. 1 supplies an example of this, showing that the hyperbolic radius decreases as the robot gets closer to humans (*top-left* and *bottom-right* frames) and increases when the path is clear of obstacles (*top-central* and *top-right* frames).

Following established literature [4], [6], [7], [10], we assess our approah by benchmarking it against state-of-the-art techniques with fair simulations in complex and simple scenarios. Overall our main contributions are as follows:

- We propose *Hyp^{2}Nav*, the first DRL hyperbolic path-planner with hyperbolic curiosity that features a hierarchical path decision process;
- *Hyp^{2}Nav* maintains high success rates and returns with low-dimensional embeddings, as low as 2;
- We find novel interpretable properties of the radius embedding norm, said hyperbolic radius.

II. Related Works

A. Crowd Navigation

Early studies in crowd navigation [5], [13], [14] propose models describing agents' behavior within crowds. These studies introduce fundamental concepts such as *Social Forces* [5] and provide models that guarantee collision avoidance in case of perfect information even for multiple agents simultaneously [14]. The approaches only employ past information about the crowd motion and do not explicitly model the future trajectories of the obstacles. A second line of work [15], [16], [17] focuses on predicting the trajectories of the dynamic obstacles to devise a safe path for the robot. While effective in terms of predictions, these techniques require a lot of computation, especially when dealing with large crowds, causing the decision-making model to be too slow for real-time applications. Chen et al. [18] describe the decision-making problem as an MDP and proposes to adopt DRL to solve the task. Following this approach, Chen et al. [4] extend the agent's attention from human-robot interaction to a more comprehensive crowd-robot interaction, explicitly modeling the human-robot and human-human interactions. SG-D3QN [7] introduces a social attention mechanism to retrieve a graph representation of the crowd-robot state, which can be further improved [10], [6] by relying on intrinsic rewards [12], [19] and spatio-temporal maps for environment representations. Our work takes inspiration from Martinez et al. [10] and introduces hyperbolic latent representations to encode the MDP states, enabling us to achieve high performance with low-dimensional embeddings.

Exploration in Crowd Navigation is paramount for the autonomous agent to learn nuanced decision-making, balancing the need to navigate close to obstacles for efficiency with ensuring safety in populated environments. ICM [12] and RE3 [19] are recent methods that implement this strategy providing the policy network with "bonus rewards" when the agent visits unknown spaces. In ICM, a small network is fed with the current state and the action the agent is performing, and it is tasked to predict the representation of the subsequent state. The more the error (meaning that the state is unknown), the more the intrinsic reward. This work extends ICM, adopting hyperbolic state representations to increase further the reward for exploring novel states.

B. Hyperbolic Neural Networks

Hyperbolic Neural Networks (HNNs) are emerging as a powerful tool for capturing hierarchical representations, yielding compact representations, dealing with uncertainty, and much more [20], [9]. Recent works in computer vision for example demonstrate the potential of HNNs, outperforming traditional Euclidean models in several tasks [21], [22], [23]. Recently, Cetin et al. [24], acknowledge that MDPs can be represented as tree graphs in the states space, thus they advocate for using hyperbolic latent representations in the domain of DRL, as they natively extract hierarchical features from the data. They demonstrate the advantages of coupling hyperbolic learning with DRL methods on Procgen [25] and Atari-100k [26], reporting performance improvements on a wide range of test environments. They define a hybrid network with an Euclidean backbone and hyperbolic projective and classification layers. This work, for the first time applies the insights of Cetin et al. [24] to the crowd navigation problem. We furthermore introduce new value and curiosity modules that operate entirely in hyperbolic space. Moreover, we investigate the efficacy of hyperbolic intrinsic rewards and interpretability by analyzing hyperbolic features.

III. Background

A. Problem formulation

The crowd navigation problem consists of finding the optimal policy that drives an autonomous agent to a goal position in a crowd. For our simulation, we leverage the widely adopted CrowdNav simulator [4], see e.g. [4], [7], [6], [10], [18], [27]. In this environment, the dynamic obstacles are informed of each other agent's state (apart from the robot's state) to avoid reciprocal collisions. Those states encompass their current position ($p \in \mathbb{R}^2$), their velocity ($v \in \mathbb{R}^2$), and

the radius r used as a 2D proxy of their volume. Thus, we describe the i-th obstacle visible state at time t as:

$$w_t^i = [p_x, p_y, v_x, v_y, r] \tag{1}$$

and refer to the state of all obstacles as $w_t^h = [w_t^1, \cdots, w_t^n]$ The robot is designed as a holonomic robot, and its state encompasses the current position ($p \in \mathbb{R}^2$) and velocity ($v \in \mathbb{R}^2$), its radius r, the maximum scalar velocity v_M, the current steering angle θ and the goal position ($g \in \mathbb{R}^2$), such that:

$$w_t^r = [p_x^r, p_y^r, v_x, v_y, r, v_M, \theta] \tag{2}$$

At each timestep, the robot senses the environment's visible state, i.e., $w_t = [w_t^r, w_t^h]$. The autonomous agent moves according to the policy provided by our RL algorithm, which allows the models to steer along 16 equally spaced angles in $[0, 2\pi)$ and move at 5 different velocities or stay still, resulting in 81 possible actions in each state. In the following, we define $\mathcal{A}$ as the set of allowed actions for the robot.

B. Reinforcement Learning for crowd navigation

Following leading approaches [4], [6], [7], [10], we cast the problem of crowd navigation as an MDP, where at each timestep t the robot has to perform the optimal action ($a_t^* \in \mathcal{A}$) according to the current visible state. The problem is set as finding the optimal deterministic policy $\pi^* : w_t \mapsto a_t^*$ that associates the optimal action to the state w_t for every timestep t. For retrieving the optimal policy, we define $Q^*(w, a)$ as the optimal state-action function describing the expected value of taking a particular action a being in a certain state w. $Q^*(w, a)$ satisfies the Bellman equation:

$$Q^*(w, a) = \sum_{w', r} \mathbf{P}(w', r|w, a)[r + \gamma^t \max_{a'} Q^*(w', a')] \tag{3}$$

where w' is the state following w after performing the action a, r is the extrinsic reward provided by the environment, and $\gamma \in (0, 1)$ is the discount factor that adjusts the interset for future rewards. The optimal policy π^* is then defined as:

$$\pi^*(w_t) = \arg\max_a Q^*(w_t, a) \tag{4}$$

From Eq. 3, it follows that accurately defining the reward is a key factor for RL algorithms. The extrinsic reward ($r_t^e = r^e(w_t)$) should be shaped to encourage the agent to reach its goal fast while avoiding collisions and creating discomfort to the human dynamic obstacles. As done in [10], we formalize it as follows:

$$r_t^e = \begin{cases} 0.25 & \text{if goal reached} \\ -0.25 & \text{if collision} \\ -0.2 \ d_t^g + \sum_{i=0}^{N} f(d_t^i) & \text{otherwise} \end{cases} \tag{5}$$

where d_t^g and d_t^i represent the current distance from the goal and from the i-th obstacle, respectively. The summation term promotes a safety constraint, increasing the reward if the robot maintains a minimal distance from each obstacle. Thus, we devise f as:

$$f(d_t^i) = \begin{cases} d_t^i - 0.2 & d_t^i < 0.2 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

C. Hyperbolic Deep Learning

This paper advocates for hyperbolic learning to perform crowd navigation. A hyperbolic metric space is a Riemannian manifold with constant negative curvature $-c$ (in this paper, we consider $c = 1$ unless explicitly stated) [11]. Its definition encompasses several isometric models, among which we select the Poincaré ball $\mathbb{D}^n$, following recent literature [24], [21], [23], [28], [29]. The Poincaré ball is an open ball of radius 1, $\mathbb{D}^n = \{x \in \mathbb{R}^{n+1} : \ ||x|| < 1\}$, endowed with the Riemannian metric:

$$g_x^{\mathbb{D}} = \frac{2I_n}{1 - ||x||^2} \tag{7}$$

with I^n an $n \times n$ Identity matrix. Important in this paper is the ability to map points between hyperbolic space and its tangent space (i.e. Euclidean space), which are given by the exponential and logarithmic mapping functions. Given a point $v \in \mathbb{D}^n$, the exponential map with basepoint v $\exp_v$ projects any point from the tangent space of $\mathbb{D}^n$ in u $u \in T_v(\mathbb{D}^n)$ into $\mathbb{D}^n$:

$$\exp_v(u) = v \oplus (\tanh(\frac{||u||}{1 - ||v||^2})\frac{u}{||u||}) \tag{8}$$

where $\oplus$ represent the Möbius addition [11]. As a common practice, we consider the origin O of the Poincaré ball to be the basepoint for projections. This simplifies Eq. 8 to:

$$\exp_O(u) = \tanh(||u||)\frac{u}{||u||} \tag{9}$$

Inversely, the logarithmic map with basepoint O, $\log_O$, is defined as:

$$\log_O(v) = \tanh^{-1}(||v||)\frac{v}{||v||} \tag{10}$$

The distance between points on the Poincaré ball is:

$$d_{\mathbb{D}}(x, y) = \text{arcosh } (1 + \frac{2||x - y||}{(1 - ||x||^2)(1 - ||y||^2)}) \tag{11}$$

Eq. 11, highlights the growth of the volume in hyperbolic space: when a point approaches the boundary of $\mathbb{D}^n$, its norm grows exponentially. Leveraging the base operations described in Ganea et al. [11], namely Möbius addition and Möbius matrix multiplication, we can extend the main modules commonly used for deep learning. In this work, we adopt hyperbolic multi-layer perceptrons (h-MLPs), which rely on Möbius matrix multiplication and addition.

IV. Methodology

In this work, we introduce Hyp^{2}Nav. We propose a policy network, rooted in hyperbolic deep learning, dubbed HyperPlanner and responsible for taking optimal decisions (in Sec. IV-A), and a novel intrinsic reward module, dubbed HyperCuriosity and responsible for exploration (in Sec. IV-B). We outline both components below. An overview of our approach is shown in Figure 2.

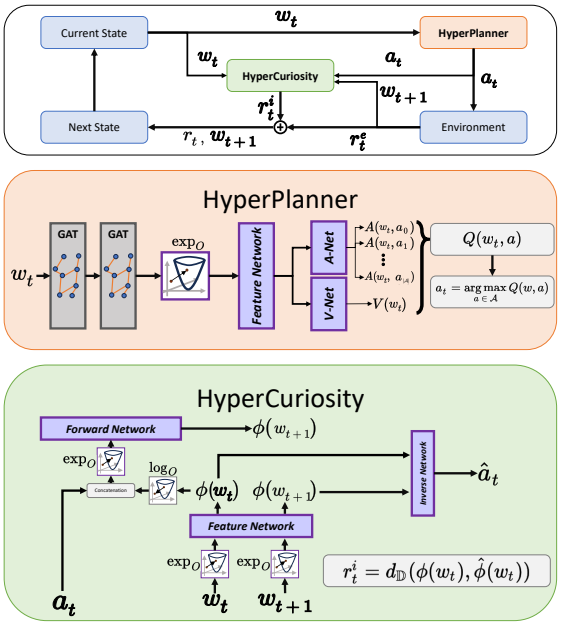

Fig. 2. **Overview of Hyp^{2}Nav.** We propose a hyperbolic policy network and a curiosity module to enable effective crowd navigation using only a few embedding dimensions. Modules in purple denote hyperbolic networks.

A. HyperPlanner

We introduce a model to navigate autonomous agents in dynamic environments. For this, we employ graph encoding for the current state, followed by a value network adapted to fully operate in hyperbolic space that is tailored for graph representations and allows for extreme dimensionality reduction. Inspired by the Dueling-DQN [30], the HyperPlanner focuses on accurately determining the value of potential actions in each state, facilitating informed decision-making by the robot.

Environment graph. We consider the environment's visible state w_t as a graph whose nodes are represented by the agents in the scene, and the features of each node coincide with its observable state. This work relies on two subsequent graph attention (GAT) modules to encode this graph effectively. The first GAT layer receives as input an embedding $\Phi(w_t^*)$ of each agent's current state, obtained via an MLP Φ to have the same dimension for both w_t^r ($\in \mathbb{R}^9$) and w_t^i ($\in \mathbb{R}^5$), while the second layer outputs high-level representations along with softmax-normalized attention coefficient that describe the strength of the interaction of each agent with each other. We employ two GNN layers to account for second-order interactions, as the reciprocal interactions of two obstacles can affect the robot planning.

Hyperbolic Policy Network. Next, we map the weighted representations to hyperbolic space via exponential mapping (cf. Eq. 9). A hyperbolic-MLP (h-MLP) module is in charge of extracting the compact hierarchical features and refining the coefficients given by the GATs. In the remainder of this section and in Fig. 2, with a slight abuse of notation, we omit to represent the action of this module to keep the notations easy to read. Starting from these representations, two separate modules. are tasked to (1) estimate the value of the current state (the *V-Net*, V) and (2) quantify the relative advantage of each possible action on the current state (the *A-Net*, A). The two modules are modeled as h-MLPs with ReLU activations:

$$\begin{aligned} V(w_t) &= \text{h-MLP}(\text{ReLU}(\text{h-MLP}(w_t))), \\ A(w_t) &= [A(w_t, a^0), ..., A(w_t, a^{|\mathcal{A}|})] \\ &= \text{h-MLP}(\text{ReLU}(\text{h-MLP}(w_t))) \end{aligned} \tag{12}$$

Notably, aiming to reduce the computational overhead severely, we constraint these modules to operate in 2-dimensional latent spaces. Indeed, by leveraging the capacity of hyperbolic modules to encode data with a hierarchical structure, we rely on the compact yet expressive power of this reduced dimensional setting, allowing for efficient computation without sacrificing the depth and quality of the agent's environmental understanding. Finally, we aggregate the outputs to retrieve a single value for each action in the current state as:

$$Q(w, a) = V(w) + (A(w, a) - \frac{1}{\mathcal{A}} \sum_{a' \in \mathcal{A}} A(w, a')) \tag{13}$$

and we choose the next action as

$$a_t = \pi(w_t) = \arg\max_{a \in \mathcal{A}} Q(w, a) \tag{14}$$

B. HyperCuriosity

Within our DRL framework, we propose a module, dubbed HyperCuriosity, that refines the exploration-exploitation trade-off by leveraging the distinctive metric properties of hyperbolic space. We build upon the concept of ICM [12], which, akin to human curiosity, encourages exploration by generating intrinsic rewards based on the prediction error of the consequences (next state) of the agent's action. Pathak et al. [12] show that the intrinsic reward mechanism benefits from state representations invariants to factors that do not affect the agent's decisions. HyperCuriosity further extends this study, adopting hyperbolic latent space to compactly encode the hierarchical features of the environment.

HyperCuriosity consists of three interconnected components settled in hyperbolic space: a feature extractor ϕ, a forward module f, and an inverse module g; its input corresponds to the triplet (w_t, a_t, w_{t+1}) representing the interaction between the agent and the environment produced by the current policy $a_t = \pi(w_t)$. The feature extractor ϕ consists of an exponential mapping to embed the states into hyperbolic space and an h-MLP layer:

$$\phi(w) = \text{h-MLP}(\exp_O(w)) \tag{15}$$

We apply ϕ to two subsequent states ($w_t \xrightarrow{a_t} w_{t+1}$) recovering their high-level representations $\phi(w_t), \phi(w_{t+1})$. Next, the forward module f, devised as an h-MLP, uses the current state's representation and the transition action a_t (encoded as a one-hot vector) to predict the next state's representation provided by ϕ. Since the concatenation involves two terms defined in different spaces, we first apply the $\log_O$ map

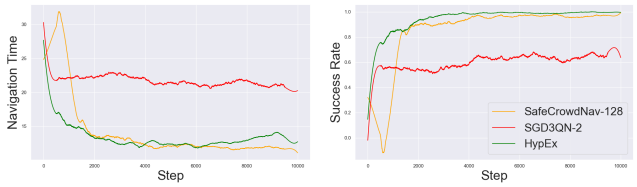

Fig. 3. Learning curves during training of the navigation time and success rate of Hyp^{2}Nav, SGDQ3N [10], and SafeCrowdNav [6]. The smoothing factor is 0.99.

(Eq. 10) to $\phi(w_t)$ to project it in the Euclidean space. We formalize this layer as:

$$\begin{aligned} f(w_t, a_t) &= \text{h-MLP}(\exp_O([\log_O \phi(w_t), a_t])) \\ &= \hat{\phi}(w_{t+1}) \end{aligned} \tag{16}$$

where $[\cdot, \cdot]$ is the concatenation operation. The third component, g, acts as a regularizing module, using the representations from the feature extractor to infer the action that transitions between the two, indirectly modeling the information conveyed by ϕ's representations. Besides using hyperbolic latent spaces to compactly represent the relevant features of the environment, our proposal lies in the intrinsic reward computation, which is derived from the discrepancy between $\phi(w_{t+1})$ and the predictive module's anticipated next-state representation $\hat{\phi}(w_{t+1})$:

$$r_t^i = d_{\mathbb{D}}(\phi(w_{t+1}), \hat{\phi}(w_{t+1})) \tag{17}$$

Fully-Hyperbolic Architecture. The entire model architecture is hyperbolic, and it inherits its metric properties. The advantages of this choice are twofold. First, when representing hidden states of an MDP, the hyperbolic latent space is more appropriate as this space is naturally suited to embed tree-like structures, adhering to the hierarchical nature of consecutive decisions and state space decision-depending evolution [24]. Second, the Poincaré distance sets a penalty that grows exponentially with the hyperbolic radius. This is reflected in an increased contribution of the intrinsic reward, effectively motivating the agent to explore novel states during training. Moreover, this property yields, as shown in Sec. VI-B, the opportunity for the model to increase exponentially the penalty for mistaken decisions when the situation is *simple* and to reduce the penalty when the situation is *complex*. Interestingly, when guided so, the hyperbolic radius of the embeddings is unsupervisedly learned, and its magnitude correlates with the situation's complexity.

All the components of HyperCuriosity are randomly initialized and trained end-to-end with the policy network. This scheme entails that the intrinsic rewards are more influential in the first part of the training and smoothly decrease throughout the training due to the optimization of the predictive network. This represents an opportunity for the model to tune the exploration importance directly from data, which represents an intriguing alternative to the systematic decrease of exploration adopted in classical exploration strategies [10].

TABLE I

COMPARISON ON THE COMPLEX SETTING OF CROWDNAV [4]. HYP^{2}NAV OBTAINS THE BEST PERFORMANCE AT A FRACTION OF THE REQUIRED NUMBER OF LEARNABLE PARAMETERS.

	Params.	Nav. Time↓	Avg. Return↑	Succ. Rate↑
Circle				
ORCA [14]	–	11.01	0.331	76.9
SGD3QN-ICM [10]	361K	11.37	0.668	96.8
SGD3QN-RE3 [10]	149K	11.01	0.682	97.1
SafeCrowdNav [6]	361K	11.18	0.673	97.7
Hyp^{2}Nav-128	144K	**10.94**	0.678	97.9
Hyp^{2}Nav	**55K**	12.04	**0.698**	**99.3**
Square				
ORCA [14]	–	12.86	0.442	84.0
SGD3QN-ICM [10]	361K	10.73	0.687	97.0
SGD3QN-RE3 [10]	149K	10.22	0.675	95.8
SafeCrowdNav [6]	361K	10.54	0.678	97.7
Hyp^{2}Nav-128	144K	**10.15**	0.693	98.6
Hyp^{2}Nav	**55K**	11.08	**0.715**	**99.8**

Implementation details. We train Hyp^{2}Nav for 10k episodes and then select the best checkpoint. We adopt RiemmanianAdam as optimizer with a learning rate of 10^{-3}. Fig. 3 shows the training curves.

V. EXPERIMENTS

In this section, we first describe the benchmark we adopt, detailing the simulation environment, the baseline models, and the metrics (Sec. V-A). Next, we discuss the results of Hyp^{2}Nav in two increasingly complex scenarios (Sec. V-B).

A. Simulation setup and benchmarking

Simulations. To evaluate the proposed Hyp^{2}Nav, we follow Zhou et al. [7] and use two scenarios from the CrowdNav [4] simulation environment: simple and complex. The simple scenario consists of 5 humans in the scene, positioned in a circle, who must reach a predefined goal by crossing the circle's center. The complex scenario involves 10 humans, each with a predetermined objective, where 5 are positioned in a circle or a square, and the other 5 are randomly set. The agent obtains a new random destination goal upon reaching its current one to avoid still obstacles. The ORCA [14] algorithm determines the human agents' paths, allowing them to navigate without collisions. Following literature, we define the interval between two consecutive timesteps as 0.25 seconds. We train Hyp^{2}Nav for 10000 episodes, evaluating its performance every 500 episodes. Finally, we consider three conditions for ending an episode, namely collision(the robot collides with an obstacle), out-of-time (the robot fails to reach its destination within 30 seconds, but no collision occurs), and success (the robot safely reaches the goal within 30 seconds).

Baselines. We benchmark our proposed Hyp^{2}Nav against several state-of-the-art methods which we adopt as comparative baselines. We consider ORCA [14], which employs a reactive policy, setting pedestrian radii to $d_s = 0.2$ for preserving safety distances. Additionally, we evaluate against two versions of Intrinsic-SGD3QN [10], ICM and

TABLE II

COMPARISON ON THE SIMPLE SETTING OF CROWDNAV [4]. AKIN TO THE COMPLEX SETTING, WE ARE ABLE TO OBTAIN THE HIGHEST RETURN AND SUCCESS RATE WITH HIGHER PARAMETER EFFICIENCY.

Method	Nav. Time↓	Avg. Return↑	Succ. Rate↑
ORCA [14]	13.87	0.323	73.6
SGD3QN-ICM [10]	**9.79**	0.696	96.6
AEMCARL [31]	12.86	0.539	92.0
SafeCrowdNav [6]	9.98	0.707	98.6
Hyp^{2}Nav-128	10.56	**0.747**	**100**
Hyp^{2}Nav	10.66	0.707	99.5

RE3, which adapts the exploration modules from Pathak et al. [12] to social navigation tasks. Furthermore, we assess SafeCrowdNav [6], the former state-of-the-art method, which utilizes intrinsic exploration rewards as in Martinez et al. [10] and provides quantitative safety score assessments. We report the results of our solution for both the low (2) and high-dimensional (128) embedding versions.

Metrics. We test each method on 1000 randomly generated episodes and evaluate them using three metrics: "Average Return", the average cumulative return over steps, "Navigation Time", the average time required for the robot to reach the goal, and "Success Rate", the rate of the robot reaching the goal without collisions.

B. Results

We analyze the quantitative performance of Hyp^{2}Nav with state-of-the-art approaches on complex and simple scenarios, reporting the results in Tables I and II, respectively.

Complex Scenario. Table I presents the evaluation on the complex scenario where the robot navigates through 10 humans to reach the final goal. Our proposal demonstrates higher success rates than other methods across the 2- and 128-dimensional versions. Notably, the 2-dimensional variant achieves a success rate of 99.3%, surpassing the 128-dimensional version by 1.4% and outperforming the best baseline by 2.9% in the circle scenario, and by 1.2% and 2.1% in the square one. Moreover, our method achieves higher average returns than state-of-the-art methods, particularly with the low-dimensional version in both cases. Interestingly, although the 128-dimensional version reaches its destination $\sim$1 second faster on average than the 2-dimensional counterpart in both scenarios, Hyp^{2}Nav consistently yields the best success rate, provoking fewer collisions. The advantage of navigation in hyperbolic space is that this performance comes at a fraction of the parameters. Compared to the current state-of-the-art, our approach is 2.7$\times$ to 6.5$\times$ more parameter-efficient.

Simple Scenario. Table II shows the evaluation results for the simple scenario. Similar to the complex case, our method demonstrates superior success rates and average returns for the 2- and 128-dimensional variants, showing improvements of 1.4% and 5.6%, respectively. Our solution achieves a perfect success rate in all 1000 test cases, reporting no collisions. Interestingly, in contrast to the complex scenario, the high-dimensional version outperforms the low-dimensional one in this case. In the simple scenario, the Euclidean models appear more aggressive, as confirmed by the lower navigation time of [8], [10]. However, this also results in decreased safety, as the Euclidean-based models report substantial drops in success rate, more likely to expose potential humans to collisions.

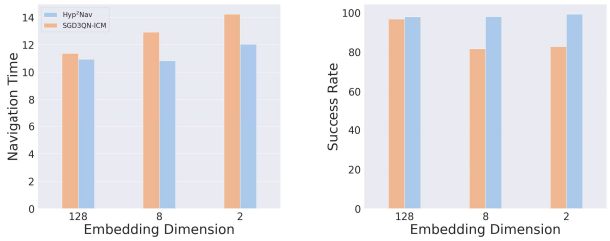

Fig. 4. Comparison of the proposed Hyp^{2}Nav and SGDQ3N [10] with different embedding dimensions. For both navigation time (left) and success rate (right), Hyp^{2}Nav achieves better performance across all the dimensionalities. In the case of 2 and 8, SGDQ3N [10] fails to converge, whereas our proposed model shows the best performance with 2 dimensions.

TABLE III

COMPARISON ON A MORE COMPLEX SETTING OF CROWDNAV [4]. TRAINED ON THE COMPLEX SETTING, HYP^{2}NAV ACHIEVES THE BEST SUCCESS RATE AND AVERAGE RETURN WHEN DOUBLING THE OBSTACLES IN THE SCENE.

Method	Nav. Time ↓	Avg. Return ↑	Success Rate ↑
SGD3QN [10]	**13.81**	0.559	93.2
SafeCrowdNav[6]	13.86	0.478	90.6
Hyp^{2}Nav	14.55	**0.571**	**94.6**

VI. DISCUSSION

A. Embedding Dimensionalities

Fig. 4 illustrates the performance comparison between two methods, *Intrinsic-SGD3QN* [10] and our proposed approach *Hyp^{2}Nav*, across different embedding dimensions (2, 8, and 128). In all the considered dimensions, our hyperbolic learner consistently outperforms its Euclidean counterpart, illustrating two critical advantages: the effectiveness of hyperbolic modules even with minimal embedding sizes and the overall convenience of adopting a hyperbolic approach for state representation in the crowd navigation task. Indeed, the Euclidean competitor fails to converge when employing only 2 dimensions, reporting a clear drop in both the average navigation time and success rate metrics.

In Fig. 3 we compare the training trends of Hyp^{2}Nav and Intrinsic-SGD3QN [10]. The plot clearly shows that the Euclidean model fails to converge when the latent space is constrained to have dimensionality 2 (orange curve), reporting low success rate, and high average time, indicating that most episodes conclude matching an out-of-time condition.

B. Hyperbolic Radius and Uncertainty

While training its policy, Hyp^{2}Nav learns to encode the different states composing an episode in different areas of the

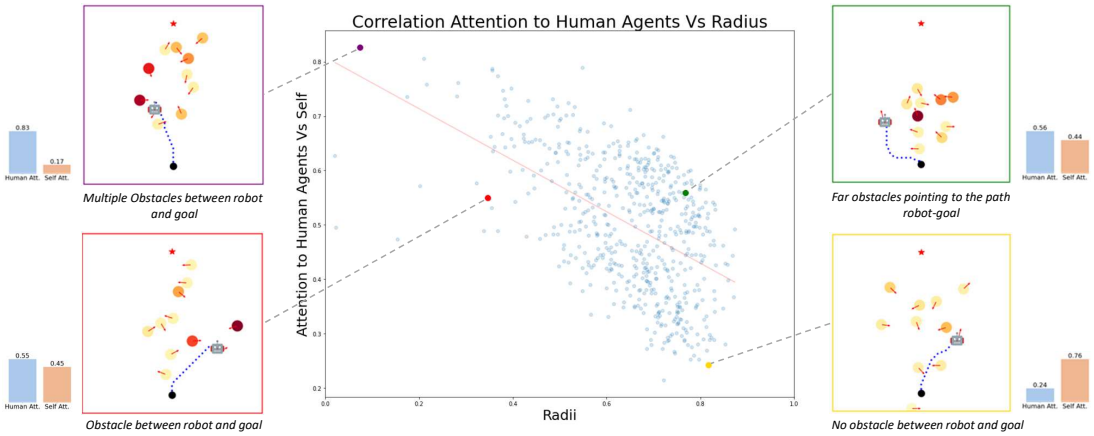

Fig. 5. **Correlation analysis between hyperbolic radius and robot uncertainty.** Our hyperbolic embedding spaces inherently encode various navigation scenarios, ranging from simple to complex. Complex scenarios often require altering a planned trajectory or adjusting steering to avoid potential collisions. When dealing with many close obstacles and possible collisions, our policy obtains embeddings closer to the origin (*purple* and *red* frames in the Figure) . At the same time, easier and more certain cases (*green* and *yellow* frames) yield embeddings near the boundary of hyperbolic space. Hence, our approach comes with a simple way to measure how certain a robot is at any given time.

Poincaré Disk. We reach this conclusion upon investigation of Hyp^{2}Nav's representations by analyzing their hyperbolic radius, which is the distance from the center of the Poincaré Disk to the embeddings. In Fig. 5, we plot the hyperbolic radius of the current state Vs the attention the system is paying to obstacles other than self for a single timestep. The plot reveals a high negative correlation between these measures (0.55). This confirms that more complex situations correspond to a smaller radius, indicating that the agent is more attentive to possible collisions. Consequently, the smaller radius reflects increased uncertainty.

On the sides of Fig. 5, we associate to some points the visualization of the current dynamic running in the simulation. In particular, we select two points with low radius (≤ 0.5, red and purple points) and two with high radius (≥ 0.5, yellow and green points). As can be seen from the Figure, we found that the points with lower radius values correspond to situations in which the obstacles are likely to impede the robot from easily moving forward. In the bottom-left frame, the robot cannot move toward the goal as a person is approaching on its left and cannot devise a safe root moving to the right, given the presence of a second obstacle on its right, demonstrating awareness of the situation's complexity. Similarly, in the purple frame, the robot is surrounded by several obstacles moving toward the line between the agent and its goal. Still, the robot successfully avoids collision, but the safer way it has to take is opposite to the direction of the goal. On the other hand, the other two examples show the robot being sure of the next action, as no obstacle seems to approach the path to the goal. Surprisingly, the robot is near an obstacle in the bottom-right frame, but it correctly predicts its intention to move away. In the top-right one, the robot is mainly focused on obstacles that are far away from it. Still, they point to the optimal direction for the robot and risk colliding in a future step.

C. Generalization

We perform a study on the generalization capabilities of our model to further investigate the quality of Hyp^{2}Nav's internal representations. For this experiment, we increase the scenario's complexity; Table III reports the results of the best performer models from Table I when doubling the number of dynamic obstacles in the scene. In particular, all the baselines represented in Table III have been trained in the complex scenario (10 obstacles) but are assessed with 20 humans. Even in this challenging scenario, Hyp^{2}Nav shows the best performance for success rate and average return, outperforming the most recent model from [6] by 4 p.p. points and surpassing the Intrinsic-SGD3QN (128-embedding dimensions). Although there is a slight increase (≤ 1 sec.) in navigation time compared to the best performer baseline, the trade-off is balanced by the gains in navigational safety and efficiency. These findings highlight the robustness and adaptability of Hyp^{2}Nav, which is key in real-world crowd navigation when the number of entities composing the crowd is unknown and can vary with time.

VII. Limitations

In Sec. V-B, we show that our proposed model reports the best success rate at a fraction of the competitors' parameter count, which results in a decreased memory footprint (0.21 Vs. 1.38 MB for Hyp^{2}Nav and SafeCrowdNav, respectively). However, we acknowledge that ours is not the faster model in terms of runtime (30.6 Vs. 13.4 msec for Hyp^{2}Nav and SafeCrowdNav, respectively). This is due to the intrinsic complexities associated with the hyperbolic space computations. The primary hyperbolic operations (cf. Eqs. 9,10,11) are mathematically complex and less optimized in the current deep learning framework, which primarily caters to operations in Euclidean spaces.

Another limitation stems from employing simulations. Hyp^{2}Nav has been tested in challenging realistic crowded

simulations but the human agents follow a given ORCA [3] policy. Also, the robot does not elicit human reactions by design. Both aspects may be resolved by future more realistic simulations or by real-world deployment tests, having ensured the safety of the human participants.

VIII. Conclusions

In this paper, we have introduced Hyp^{2}Nav, a novel model for crowd navigation that exploits hyperbolic latent spaces to encode the environment into states which are natively hierarchical. Adhering to the hyperbolic learning framework allows Hyp^{2}Nav to achieve state-of-the-art success rates and average return with a significant reduction in parameter count with respect to competitive baselines, ensuring that Hyp^{2}Nav consistently devises safe paths in an efficient manner. We have shown that the hyperbolic framework comes with the additional benefit of greater interpretability, as monitoring the hyperbolic radius throughout a crowd navigation episode reveals insights about the complexity of the current state of the environment from the robot's perspective.

References

[1] P. T. Singamaneni, P. Bachiller-Burgos, L. J. Manso, A. Garrell, A. Sanfeliu, A. Spalanzani, and R. Alami, "A survey on socially aware robot navigation: Taxonomy and future challenges," *The International Journal of Robotics Research*, p. 02783649241230562, 2024.

[2] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The international journal of robotics research*, vol. 17, no. 7, pp. 760–772, 1998.

[3] J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *2008 IEEE international conference on robotics and automation*. Ieee, 2008, pp. 1928–1935.

[4] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *2019 international conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 6015–6022.

[5] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[6] J. Xu, W. Zhang, J. Cai, and H. Liu, "Safecrowdnav: safety evaluation of robot crowd navigation in complex scenes," *Frontiers in neurorobotics*, vol. 17, 2023.

[7] Z. Zhou, P. Zhu, Z. Zeng, J. Xiao, H. Lu, and Z. Zhou, "Robot navigation in a crowd by integrating deep reinforcement learning and online planning," *Applied Intelligence*, vol. 52, no. 13, pp. 15 600–15 616, 2022.

[8] Z. Zhou, J. Ren, Z. Zeng, J. Xiao, X. Zhang, X. Guo, Z. Zhou, and H. Lu, "A safe reinforcement learning approach for autonomous navigation of mobile robots in dynamic environments," *CAAI Transactions on Intelligence Technology*, 2023.

[9] W. Peng, T. Varanka, A. Mostafa, H. Shi, and G. Zhao, "Hyperbolic deep neural networks: A survey," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 44, no. 12, pp. 10 023–10 044, 2021.

[10] D. Martinez-Baselga, L. Riazuelo, and L. Montano, "Improving robot navigation in crowded environments using intrinsic rewards," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 9428–9434.

[11] O. Ganea, G. Bécigneul, and T. Hofmann, "Hyperbolic neural networks," *Advances in neural information processing systems*, vol. 31, 2018.

[12] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International conference on machine learning*. PMLR, 2017, pp. 2778–2787.

[13] G. Ferrer, A. Garrell, and A. Sanfeliu, "Robot companion: A social-force based approach with human awareness-navigation in crowded environments," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1688–1694.

[14] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics Research: The 14th International Symposium ISRR*. Springer, 2011, pp. 3–19.

[15] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 961–971.

[16] K. D. Katyal, G. D. Hager, and C.-M. Huang, "Intent-aware pedestrian prediction for adaptive crowd navigation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3277–3283.

[17] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII*. Springer-Verlag, 2020, p. 683–700.

[18] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 285–292.

[19] Y. Seo, L. Chen, J. Shin, H. Lee, P. Abbeel, and K. Lee, "State entropy maximization with random encoders for efficient exploration," in *International Conference on Machine Learning*. PMLR, 2021, pp. 9443–9454.

[20] P. Mettes, M. G. Atigh, M. Keller-Ressel, J. Gu, and S. Yeung, "Hyperbolic deep learning in computer vision: A survey," *arXiv preprint arXiv:2305.06611*, 2023.

[21] M. G. Atigh, J. Schoep, E. Acar, N. Van Noord, and P. Mettes, "Hyperbolic image segmentation," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 4443–4452.

[22] K. Desai, M. Nickel, T. Rajpurohit, J. Johnson, and R. Vedantam, "Hyperbolic Image-Text Representations," in *Proceedings of the International Conference on Machine Learning*, 2023.

[23] M. van Spengler, E. Berkhout, and P. Mettes, "Poincaré resnet," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE Computer Society, oct 2023, pp. 5396–5405.

[24] E. Cetin, B. P. Chamberlain, M. M. Bronstein, and J. J. Hunt, "Hyperbolic deep reinforcement learning," in *The Eleventh International Conference on Learning Representations*, 2023.

[25] K. Cobbe, C. Hesse, J. Hilton, and J. Schulman, "Leveraging procedural generation to benchmark reinforcement learning," in *International conference on machine learning*. PMLR, 2020, pp. 2048–2056.

[26] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *Journal of Artificial Intelligence Research*, vol. 47, pp. 253–279, 2013.

[27] M. Everett, Y. F. Chen, and J. P. How, "Collision avoidance in pedestrian-rich environments with deep reinforcement learning," *IEEE Access*, vol. 9, pp. 10 357–10 377, 2021.

[28] A. Flaborea, B. Prenkaj, B. Munjal, M. A. Sterpa, D. Aragona, L. Podo, and F. Galasso, "Are we certain it's anomalous?" in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2896–2906.

[29] M. Ghadimi Atigh, M. Keller-Ressel, and P. Mettes, "Hyperbolic busemann learning with ideal prototypes," *Advances in Neural Information Processing Systems*, vol. 34, pp. 103–115, 2021.

[30] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1995–2003.

[31] S. Wang, R. Gao, R. Han, S. Chen, C. Li, and Q. Hao, "Adaptive environment modeling based reinforcement learning for collision avoidance in complex scenes," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 9011–9018.