



Probability propagation for path planning in unknown environments[☆]

Giovanni Di Gennaro ^{a,*,} Amedeo Buonanno ^{b, ID}, Giovanni Fioretti ^{a, ID}, Francesco Verolla ^{a, ID},
Francesco A.N. Palmieri ^{a, c, ID}, Krishna R. Pattipati ^c

^a Università degli Studi della Campania “Luigi Vanvitelli”, Dipartimento di Ingegneria, Aversa (CE), Italy

^b ENEA, Department of Energy Technologies and Renewable Sources, Portici (NA), Italy

^c University of Connecticut, Department of Electrical and Computer Engineering, Storrs, CT, USA

ARTICLE INFO

Keywords:

Bayesian networks
Factor graphs
Path planning
Unknown environment
Unknown goal position

ABSTRACT

We propose a probability propagation framework for path planning on discrete grids where an agent can navigate in an unknown environment to discover new areas and goals. We introduce a technique in which the probabilistic backward flow provides guidance towards discovering multiple distributed goals and hidden regions. This is achieved using a maximum likelihood path estimation framework in which the hidden areas become constrained goals that “attract” the agent. Simulations on various grids are included in the paper. The results show how this idea, applied to a completely unknown environment and goal position, may provide a unifying and powerful method for distributed dynamic path planning.

1. Introduction

Efficient algorithms for path planning are important in robotic applications for determining the best behavior of an agent, trajectory prediction, scene modeling, or predicting human behavior taking into account bounded rationality and cognitive biases aspects of human decision-making. The problem has been approached in many different ways that range from clever heuristics to probabilistic inference. In this paper, we demonstrate that by working within a probability framework based on message propagation, we are able to capture the essence of most widely-used methods for path planning. The technique presented here is quite flexible and can be extended to applications involving dynamic decision-making under uncertainty.

In some of our previous works on modeling human motion in urban areas and maritime tracks, we have explored several techniques including: traditional Kalman filter-based approaches (Coscia & Bracaet al., 2018; Coscia & Palmieriet al., 2018), models with social forces (Coscia, Ballan, Palmieri, Alahi, & Savarese, 2020), motion models driven by polar histograms (Coscia et al., 2018; Coscia, Castaldo, Palmieri, Ballan, Alahi, & Savarese, 2016), and Markov chains (Castaldo, Palmieri, & Regazzoni, 2014).

The probabilistic framework adopted in this work is based on Factor Graphs (FG) (Loeliger, 2004) that represent a unified way of translating directed and undirected probabilistic graphical models in an easy-to-manipulate forward and backward message propagation (signal flow). Our work on FG in reduced normal form (FG_{rn}) (Palmieri, 2016) has further simplified the FG framework reducing the network to interconnections of single-input single-output (SISO) blocks and diverters. Probability densities can be easily propagated for inference and learning. Seminal work on FG can be found in Loeliger (2004), Loeliger, Dauwels, Hu, Korl, Li, and Kschischang (2007); some details on the optimized implementations of FG_{rn} can be found in Di Gennaro, Buonanno, and Palmieri (2021). Extensions to multilayer configurations can be found in Buonanno and Palmieri (2015a).

The original idea of using probability propagation on a Markov chain for path planning goes back to a paper by Attias (2003), which proposed how starting points and goals can be directly imposed as initial and final constraints on a graph. Intermediate positions and actions can then be computed as inferences. Connections and equivalences between dynamic programming (DP)-based optimization (Bertsekas, 2019) and probabilistic approaches have been addressed in Toussaint

[☆] Work partially supported by POR CAMPANIA FESR 2014/2020, ITS for Logistics, awarded to CNIT (Consorzio Nazionale Interuniversitario per le Telecomunicazioni). Research of Pattipati was supported in part by the U.S. Office of Naval Research and US Naval Research Laboratory under Grants #N00014-18-1-1238, #N00173-16-1-G905 and #HPCM034125HQ, and by a Space Technology Research Institutes grant (number 80NSSC19K1076) from NASA's Space Technology Research Grants Program, USA.

* Corresponding author.

E-mail addresses: giovanni.digennaro@unicampania.it (G. Di Gennaro), amedeo.buonanno@enea.it (A. Buonanno), giovanni.fioretti@studenti.unicampania.it (G. Fioretti), francesco.verolla1@studenti.unicampania.it (F. Verolla), francesco.palmieri@unicampania.it (F.A.N. Palmieri), krishna.pattipati@uconn.edu (K.R. Pattipati).

URL: <https://www.mlsptlab-unicampania.it/> (F.A.N. Palmieri).

<https://doi.org/10.1016/j.iswa.2025.200527>

Received 21 October 2021; Received in revised form 9 January 2025; Accepted 24 April 2025

Available online 10 May 2025

2667-3053/© 2025 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

(2009), Toussaint and Storkey (2006), where the objective function, typically the (log) likelihood, is combined with the expected rewards. Indeed, the optimal value function in dynamic programming is the Helmholtz free energy (Kappen, Gomez, & Oppen, 2013; Todorov, 2009).

Since Helmholtz energy features prominently in graphical model inference, this opens up a number of ways of quantitatively addressing the computation of near-optimal effort, including bounded rationality exhibited by human decision makers (Ortega & Braun, 2013). There is also a recent review by Levine (Levine, 2018) that provides interesting insights on the topic and connections to free-energy based stochastic models (Baltieri & Buckley, 2017; Buckley, Kim, McGregor, & Seth, 2017; Parr & Friston, 2018a) and to KL-learning (Alexandre, Oleg, & Giovanni, 2018). These ideas seem to be very promising in providing a framework for intelligent behavior modeling (Kaplan & Friston, 2018; Parr & Friston, 2018b) and causal reasoning (Nair, Zhu, Savarese, & Li, 2019).

In our recent work (Palmieri, Pattipati, Di Gennaro, Fioretti, Verolla, & Buonanno, 2022), we have provided a unified view on the topic and shown how, by changing some of the propagation rules in the Bayesian graph, we can include and generalize a large number of previously proposed algorithms for sequential decision-making under uncertainty. These issues are beyond the scope of this paper whose intent is to focus on grid exploration and exploitation. However, some brief comments on the equivalence of the probability propagation approach and Markov Decision Process (MDP) will be included in this paper.

We have already explored the probability framework for path planning in one of our previous works (Palmieri, Pattipati, Fioretti, Di Gennaro, & Buonanno, 2021) using tensor flows. The obstacles in the grid are treated as reflective (Palmieri et al., 2021), differently from the Attias' model (Attias, 2003), and multiple goals and multiple agents are addressed as well. However, differently from some of these methods, map and goals are assumed here to be completely unknown at the beginning and get revealed as the agent traverses the grid.

1.1. Path planning in known environment

In a classic path planning problem, we typically identify the space of all possible agent configurations (C-space), which includes the space where the agent can move (C-free), and the space where obstacles are located (C-obj) (Gasparetto, Boscariol, Lanzutti, & Vidoni, 2015). Without loss of generality, we assume that all of them are in a discrete $N \times M$ rectangular grid.

Numerous techniques, briefly summarized in the following, have been proposed in the literature to solve this well-known computational problem, some of which bear intriguing similarities to our probability-based approach, although they have been proposed in completely different application domains:

- *Flood fill algorithms* find the area connected to a given cell by flooding the map. It is a simple algorithm, also used to program a micromouse to autonomously find the center of a maze, moving through obstacles (Law, 2013; Tjiharjadi, Wijaya, & Setiawan, 2017);
- *Roadmap techniques* map the free space connectivity into a system of one-dimensional curves (using, for example, the visibility graph Lozano-Perez, 1983; Lozano-Pérez & Wesley, 1979, or the Voronoi diagrams Canny & Donald, 1987; Takahashi & Schilling, 1989);
- *Cell decomposition algorithms* subdivide the C-free into several regions, called cells, in such a way that the path is easy to calculate (Abbadì & Prenosil, 2015a, 2015b);
- *Artificial potential methods (APM)* are based on the idea that the agent is moving in a potential field, where goals produce an attractive potential and the obstacles generate a repulsive potential (Coscia et al., 2020; Hwang & Ahuja, 1992; Nadira, Omar, & Hailma, 2016). Some other works use the APM to train a neural network that solves a POMDP, as in Yao et al. (2020);

- *Probabilistic Roadmap Planners (PRM)* take random samples from the C-free and connect them, until the roadmap is dense enough. Then, the Dijkstra's shortest path algorithm is invoked to find the best path through the roadmap (Amato & Wu, 1996; Nissoux, Simeon, & Laumond, 1999);
- *Rapidly-exploring Random Trees (RRT)* build space-filling trees from samples drawn randomly from the C-free (Abbadì & Prenosil, 2015a; Laval, 1998).

As previously mentioned, there are some intriguing connections between some of the above techniques and the backward probability tensor flow used in this paper. More specifically, our backward probability flow bears strong resemblance to the *flooding algorithm*. The directionality provided by the action-state probabilities resembles a vector field (Palmieri et al., 2021) for which the agent would be driven by a *potential field*. Furthermore, if we consider the planning problem as a possible path through a very large trellis, a connection can be found to the Viterbi algorithm on a Factor Graph (Palmieri et al., 2021). However, our approach is completely re-derived in a probabilistic goal-seeking framework, as it will be detailed in the following.

1.2. Path planning in completely unknown environment

Partial knowledge of the environment is ubiquitous in numerous planning and inference applications. As an example, a robot may not be aware of the whole map and it may need to rely solely on what it "sees" at a particular moment. This applies to both obstacles and goals.

Another example is troubleshooting a complex system with imperfect test outcomes.

To fix ideas, we can imagine a car that enters a parking garage and is looking for a free spot. The driver does not necessarily know the garage's layout (obstacles), and may not know where empty spots may be (goals). Therefore, the agent may have to go through an exploration phase around the obstacles until at least one of its goals becomes visible. It is only at this point that the agent can begin to steer towards the goal following a path planning strategy.

For this reason path planning in an unknown environment is a problem noteworthy importance for automated cars, robots and unmanned vehicles: an autonomous car that seeks to find an empty parking spot; a robot that looks for a free spot; a misplaced object in an automated warehouse; a robot involved in search and rescue task (looking for some objects or people, reach and rescue them, and return to the initial position), etc.

Several approaches have been proposed for path planning in partially unknown environments. The classical search strategies, such as A* (Hart, Nilsson, & Raphael, 1968), D* (Stentz, 1993), D* Lite (Koenig & Likhachev, 2005) and their variants can be applied to path planning in known and unknown environments but differ from the proposed approach, because they require information on the goal position, or an estimation of the distance of the agent to the goal (termed the heuristic evaluation function or the cost-to-go). Also, the algorithms that belong to the Bug family (Choset et al., 2005) can be applied to planning and navigation in an unknown environment, but they also require information on the goal position.

The problem of planning towards an unknown goal in a completely unknown environment can be seen as a problem of search of an object in an unknown and cluttered environment (Everett, Miller, & How, 2019), or as a part of search and rescue task (Niroui, Sprenger, & Nejat, 2017). One part of the problem can be linked to the exploration of the unknown environment for identifying the goal position, and another one is related to classical path planning algorithms towards the identified goal position. Even though the exploration phase can be approached using the classical frontier exploration, where the agent moves towards the frontier points in order to expand the observed map (Yamauchi, 1997), it has to be adapted because the main objective of the exploration is different from the search objective: the agent

spends more time to explore areas that will not contain the goal. The focus should rather be on exploration only in the areas that can potentially contain the goal. Contextual information related to the observed scene (in terms of geometric or semantic relationship among observed objects) has been used recently in a biologically-inspired navigational model (Naghizadeh, Berenjian, Margolis, & Metaxas, 2020). The problem is also related to the Exploratory Path Planning approach (Santos, Otero, Johnson, Osorio, & Toledo, 2020; Zhu, Hu, Cai, & Henschen, 2011).

Different from the listed solutions, our work does not assume any knowledge about the environment and any knowledge/assumptions/estimation of the goal position. Nevertheless, any information available to the agent can be simply injected and leveraged by the proposed approach to reach the goal. This is possible because the probability propagation technique (the backward flow generated by unknown areas, and then by discovered goals) is a very general framework that allows the application of this solution in very different real world situations (e.g., search and rescue tasks, exploration tasks, autonomous vehicle context, etc.) with different levels of available knowledge.

1.3. Paper contribution

In this paper, we extend the work reported in Palmieri et al. (2021) by demonstrating how the probability flow can provide an effective and general technique to guide the agent's progressive discovery of initially hidden areas and goals. The unknown regions are treated as distributed goals and attract the agent through the backward flow originated by them. The main contribution of the work is hence methodological; it shows that the movement of the agent arises spontaneously through the mere propagation of probability without the use of any assumptions or heuristics. This highlights the power of the formalism and makes it usable directly in different contexts such as: planning in a completely known environment, planning in a completely unknown environment and goal position, planning with targets that appear in the line of sight of the agent, planning with different reward areas, etc. This feature, to our knowledge, has never been included in MDP path planning algorithms. Some preliminary results of this work have been reported in a recent conference abstract (Palmieri, Pattipati, Fioretti, Verolla, Di Gennaro, & Buonanno, 2020).

1.4. Paper outline

In Section 2, we define the reference grid and in Section 3 the dynamic model. In Section 4, the probability propagation framework, via a properly defined backward flow, is proposed as a means to discover unknown areas and goals. In Section 5, we show how the probability propagation progresses in a pure diffusion mode and in Section 6 we introduce the control policy using the backward flow. In Section 7 several simulations are presented. The vision algorithm is detailed in Appendix A and a simulation with different viewing apertures is reported in Appendix B. Section 8 contains conclusions and future research directions.

2. The discrete grid

Consider an $N \times M$ rectangular grid, where the agent's states are the positions on the grid. We confine our attention to a finite grid with motion that, by avoiding obstacles, can only progress one step at a time because our objective here is to focus on the fundamentals of the stochastic model as a very flexible framework for guiding the agent's behavior. Extensions to more complex motion kernels can be easily included as well as motion in a continuous space, but they are beyond the scope of this paper and they will be reported elsewhere.

The state space is comprised of the $n_S = N \times M$ locations on the grid. The scene is delimited by borders (immediately outside the grid), with each state corresponding to a pair of indices at discrete times t

$$s_t = (i_t, j_t), \quad i_t = 1, \dots, N; \quad j_t = 1, \dots, M \quad (1)$$

Assuming n_A possible actions, all the messages in the system can be described by $N \times M \times n_A$ state-action tensors and can be combined using sums and products. As mentioned above, we limit the agent to $n_A = 9$ possible actions:

$$a_t \in \{ \text{still}, \text{up}, \text{up-right}, \text{right}, \text{down-right}, \text{down}, \text{down-left}, \text{left}, \text{up-left} \} \quad (2)$$

Clearly, the action space can be extended to larger sets that include speed and finer bearing angle quantizations. As n_A grows, the tensor size becomes larger, and the computational issues would have to be accounted for depending on the available computing power.

3. The dynamic model

The Bayesian discrete-time model is depicted in Fig. 1 in the form of a factor graph in the reduced normal form (FGrn) (Palmieri, 2016) with a finite time horizon T . States and Actions $\{(s_t, a_t) \in S \times \mathcal{A}, t = 1, \dots, T\}$ are considered jointly and they belong to the finite sets $S = \{\sigma^1, \dots, \sigma^{n_S}\}$ and $\mathcal{A} = \{\alpha^1, \dots, \alpha^{n_A}\}$. Fig. 1 represents¹ the system in free evolution where no control policy is included yet. The model corresponds to the factorization

$$p[(s_1 a_1), (s_2 a_2), \dots, (s_T a_T)] = \pi(s_1 a_1) c_T(s_T a_T) \prod_{t=2}^T p(s_t a_t | s_{t-1} a_{t-1}) c_{t-1}(s_{t-1} a_{t-1}) \quad (3)$$

where $p(s_t a_t | s_{t-1} a_{t-1})$ is the transition probability function; $\pi(s_1 a_1)$ is the distribution for the initial state-action pair and the factors $c_t(s_t a_t)$ are prior distributions on the states and actions.²

Note that the model is an oriented graph and that a number of junction blocks (equality constraints) are included. The attractiveness of FGrn is its capability to reduce an arbitrary Bayesian graph to a network of single-input/single-output (SISO) blocks and junctions for ease of manipulation and message propagation (Buonanno & Palmieri, 2015b). Information can be injected into the system much like in a pipeline, and inference can be done by combining the messages (the arrows in the picture) using sums and products. *Forward* (f) and *backward* (b) messages are associated with each edge of the graph and are proportional to probability distributions at the connected nodes. In a tree or forest structure that contains no loops, these messages carry the information ("sufficient statistics") accumulated in their side of the graph (Loeliger, 2004; Palmieri, 2016).

Fig. 2 shows the details of a generic junction. The variable $(S_t A_t)$ is replicated for every input/output (in this case, three times — note the superscripts) and each variable is associated with a different forward and backward message. More specifically, the sum-product rules are

$$\begin{aligned} f_{(S_t A_t)^1}(s_t a_t) &= \sum_{s_{t-1} a_{t-1}} p(s_t a_t | s_{t-1} a_{t-1}) f_{(S_{t-1} A_{t-1})^3}(s_{t-1} a_{t-1}); \\ b_{(S_{t-1} A_{t-1})^3}(s_{t-1} a_{t-1}) &\propto \sum_{s_t a_t} p(s_t a_t | s_{t-1} a_{t-1}) b_{(S_t A_t)^1}(s_t a_t); \\ f_{(S_t A_t)^2}(s_t a_t) &= c_t(s_t a_t); \\ b_{(S_t A_t)^2}(s_t a_t) &\propto b_{(S_t A_t)^3}(s_t a_t) f_{(S_t A_t)^1}(s_t a_t); \\ b_{(S_t A_t)^1}(s_t a_t) &\propto f_{(S_t A_t)^2}(s_t a_t) b_{(S_t A_t)^3}(s_t a_t); \\ f_{(S_t A_t)^3}(s_t a_t) &\propto f_{(S_t A_t)^1}(s_t a_t) f_{(S_t A_t)^2}(s_t a_t). \end{aligned} \quad (4)$$

The symbol $\propto$ means "proportional to" because only some messages are normalized. This is not important for message combination because the information is contained in their shape and not in their absolute values (Loeliger, 2004; Palmieri, 2016). In any case, normalization is often advisable for numerical stability. The posterior distributions of all variables are obtained via the product rule:

$$p_{(S_t A_t)^i}(s_t a_t) \propto b_{(S_t A_t)^i}(s_t a_t) f_{(S_t A_t)^i}(s_t a_t), \quad t = 1, \dots, T; \quad i = 1, 2, 3. \quad (5)$$

¹ Representation of state-action pair $(s_t a_t)$ without comma is deliberate.

² In our recent paper (Palmieri et al., 2022), the connections between these prior distributions and rewards in MDP models with Dynamic Programming are formalized.

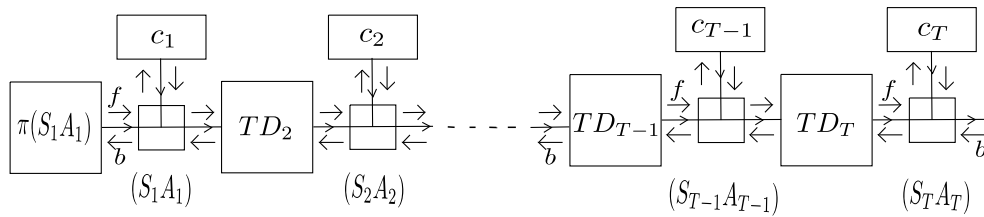


Fig. 1. The Bayesian model as a Factor Graph in reduced normal form. The square between $S_{t-1}A_{t-1}$ and S_tA_t represents $p(S_tA_t|S_{t-1}A_{t-1})$, the transition dynamics at time t (TD_t).

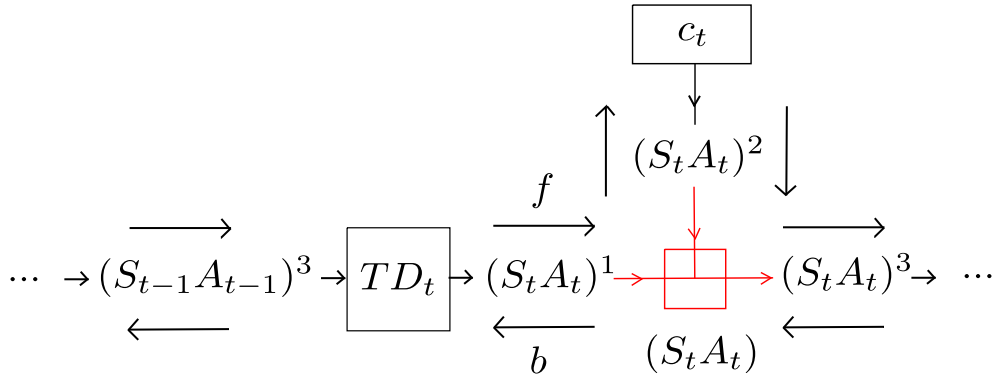


Fig. 2. The junction (in red) and its messages. TD_t represents the transition dynamics at time t : $p(S_t A_t | S_{t-1} A_{t-1})$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Even though the model can handle quite complex state-action constraints, to fix ideas and to show the efficacy of this framework, we experiment on a simplified scenario. More complex situations will be reported elsewhere.

Assumption 1. States and actions are conditionally independent:

$$p(s_t a_t | s_{t-1} a_{t-1}) = p(s_t | s_{t-1} a_{t-1}) p(a_t). \quad (6)$$

In other words, previous state and action condition the current state and the action distribution is generated independently.

We will see in the following, how this free evolution model does not limit the power of the method when a control policy is introduced into the system. Simple extensions can consider the current action dependent on the previous one with $p(a_t|a_{t-1})$ (Markov chain) to model, for example, stiffness in the motion. More general functions, such as $p(a_t|s_{t-1}a_{t-1})$, can be introduced, for example, to forbid certain actions when the state is in certain configurations. We leave them out for now.

Fig. 3 shows, in graphical form, the transition probability function in free space $p(s_t | s_{t-1} a_{t-1})$ that we use in our simulations. Only one step at a time is possible and motion is stochastic with probability concentrated in one of the eight directions, or in the center for *still*. The chosen values are arbitrary and they can be easily replaced with samples of given circular distributions such as, for example, Gaussian for speed and Von Mises for angles as proposed also in some of our previous works (Coscia et al., 2018; Coscia et al., 2016).

The state grid, in its simplest configuration, contains *obstacles*, a *starting point* and a *goal* as shown in the example of Fig. 4.

The obstacle avoidance is modeled in the agent dynamics as shown in the three examples in Fig. 5: when there is an obstacle that “overlaps” the current distribution $p(s_t|s_{t-1}a_{t-1})$, the overlapping pixels are set to zero and the other values are normalized to one. This process can be seen as an undirected reflection.

When the rule is extended to larger diffusion kernels, i.e., to transition functions with more than nine actions, that can account for higher speed and finer bearing angle discretization, it can be easily modified to avoid the unrealistic situations of an agent jumping over the obstacles.

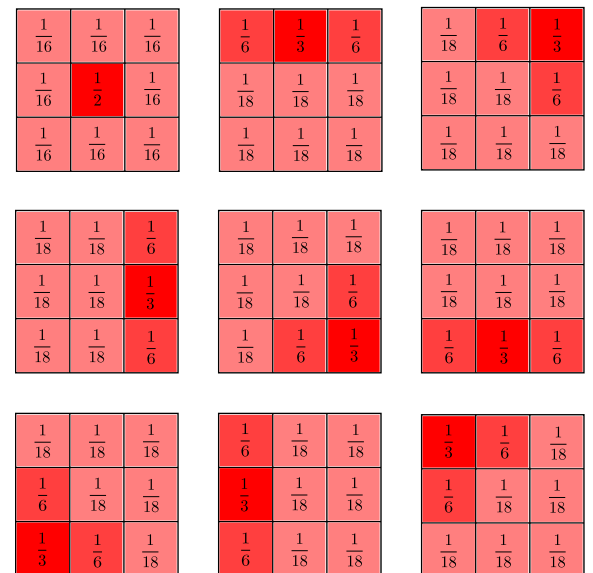


Fig. 3. Pictorial rendering of how the probability distribution $p(s_i | s_{i-1}, a_{i-1})$ is diffused on the grid. The state s_{i-1} is the center pixel and the actions $a_{i-1} \in \{\text{still, up, up-right, right, down-right, down, down-left, left, up-left}\}$, graphically shown from the upper left corner (North-West) to the lower right corner (South-East). In this example, heuristic values are assigned to the probabilities to impose direction on the flow (darker colors correspond to higher probability values).

In this paper, without loss of generality, we prefer to maintain the scenario simple and limit ourselves to one-step motion. The mentioned extensions will be reported elsewhere.

The distributions c_t , included in the model of Fig. 1, represent our prior knowledge about the state-action flow. In the simplest case, they can be assumed to be stationary: $c_t(s_t, a_t) = c(s_t, a_t)$, and they can express



Fig. 4. A grid for the simulation where the green cell indicates the start and the red cell the goal. The blue cells are obstacles.

$\frac{1}{14}$	$\frac{1}{14}$	0	$\frac{3}{14}$	$\frac{3}{7}$	0	0	0	0
$\frac{1}{14}$	$\frac{4}{7}$	0	$\frac{1}{14}$	$\frac{1}{14}$	0	$\frac{1}{5}$	$\frac{1}{5}$	0
$\frac{1}{14}$	$\frac{1}{14}$	$\frac{1}{14}$	$\frac{1}{14}$	$\frac{1}{14}$	$\frac{1}{14}$	$\frac{3}{5}$	0	0

Fig. 5. Examples of censoring of the transition distribution in proximity of obstacles (in black) for (from left to right) *still*, *up*, and *down* actions.

our “semantic” constraints on the map. For example, there may be preferences for the agent to be in certain positions and/or perform a subset of actions.

A pedestrian may prefer to walk on the sidewalks rather than on the street, even if he/she is sometimes forced to cross the road to reach his/her destination. These prior distributions can also be seen as a Bayesian equivalent of the MDP reward functions. We refer the reader to our detailed paper on this topic (Palmieri et al., 2022). In this paper, since we concentrate only on obstacles and path discovery, we assume that $c(s_t a_t)$ is a uniform distribution.

4. Agent with limited visibility

In the limited view situation, the map is divided into unknown areas (black pixels), obstacles (blue pixels) and free areas (white pixels), as depicted in Fig. 6. Assuming that agent has not seen the goal yet, it will start an exploration phase in trying to uncover what is beneath the unknown areas. Therefore, *the unknown areas become a distributed goal and the backward flow from the whole invisible area is computed*. Then the agent moves one step. At this point, the viewing algorithm calculates again which cells are visible from the current agent’s position and if a goal has become visible. If no goal is visible yet, the whole unknown area is the new goal, the backward flow is recomputed and a new step is taken. The procedure continues until a goal becomes visible. In this case, the algorithm continues to update the visibility map, and the dark areas are not set as goals anymore. During the path towards the visible goal, other unknown areas are unveiled and other possible goals can be observed.

A key aspect of our formulation is that the agent’s strategy is based on a dynamically changing grid with cells that may, or may not be visible from the current agent’s position. As our focus is primarily on searching in an unfamiliar environment, all sensor quality and noise considerations are neglected, essentially considering sensor noise below the grid resolution. Fig. 6 shows the geometry of an agent that is looking at a 315° angle direction (we use counterclockwise angles) with a viewing aperture of 90° and a perceptual range that is limited to the borders of the environment.

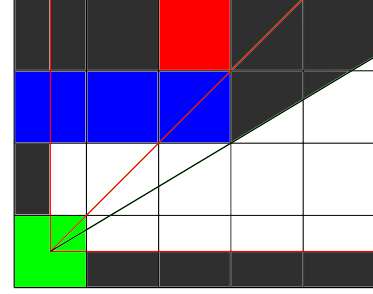


Fig. 6. Agent’s viewing geometry. The agent (green) looks at 315° (counterclockwise angles respect to the frontal direction — vertical axis) with 90° viewing aperture and has a perceptual range that is limited to the borders of the environment. Because of the presence of the obstacles (blue), the goal (red) is not visible. The black areas are the other invisible regions. In the discrete approximation, we assume that a partially-visible cell is visible. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

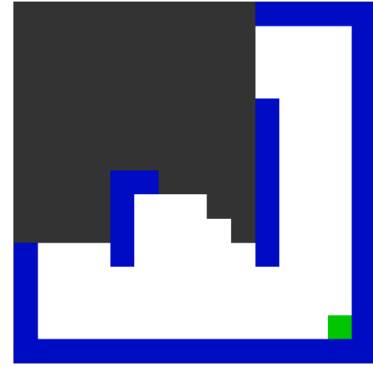


Fig. 7. Limited visibility at $t = 1$. The black zone represents the area of the map that the agent does not know yet. The green cell indicates the start, the blue cells are the visible obstacles and the white cells are the visible free areas. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Fig. 7 shows an agent with limited visibility at time $t = 1$, that can recognize obstacles, free areas and the goal. The goal may be just discovered at current time, or may have appeared in its field of view in previous time steps. The details on the geometry for computing the visibility map are reported in Appendix A. In the following simulations, the viewing angle is kept at 360°. However, in Appendix B there is a comparison with an agent with a narrower viewing angle.

5. Pure diffusion model

In the message propagation framework, the maximum a posteriori solution is found by injecting messages at the beginning and at the end of the chain and letting the sum-product algorithm compute the solution. More specifically, if the initial state (starting point) is $S_1 = \bar{s}_1$ and no preferred initial action is imposed, we have

$$\pi(s_1 a_1) = \delta(s_1 - \bar{s}_1) U(a_1), \quad (7)$$

where δ is the usual Kronecker delta function ($\delta(x) = 1$ if $x = 0$ and zero otherwise) and $U(y)$ is the discrete uniform distribution on the action space. At the end of the chain if the goal is in $S_T = \bar{s}_T$, the backward message is

$$b_{S_T A_T}(s_T a_T) = \delta(s_T - \bar{s}_T) U(a_T), \quad (8)$$

where we have assumed no preference on action at the end. Actually, the action at the end is useless because the motion stops there. We have included it to keep the notation consistent for the entire time horizon $t = 1 : T$.

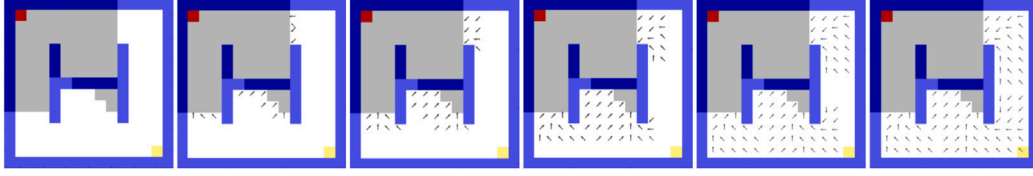


Fig. 8. Backward flow in the limited view situation when the agent is at the initial position. It requires 5 steps for the backward flow to reach the agent. The yellow cell indicates the current agent's position and the gray cells are not visible to the agent in the current position. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Although one of the main apparent weaknesses of Attias' approach (Attias, 2003) is to have a fixed time horizon T (we will address this issue later), we have found in our experience that the model provides great flexibility and important clues on the intended motion and system dynamics through the mere visualization of the probability flow. For this reason, we focus first on the process run in a pure diffusion mode, i.e., with no control module, to understand how the model tends to distribute the probability in the state-action space as a consequence of its dynamics and the imposed initial and final constraints.

Fig. 8 shows the propagation of the *backward flow* in the grid of Fig. 4 when the agent is at the initial position. In this case, 5 steps are needed for the backward flow to reach the agent's position. The goal state is depicted in red, the current position of the agent is in yellow, the gray cells represent the cells that are not visible to the agent in the current position. The unknown area can contain obstacles, goals or free spaces. The arrows in the cells describe, at various time steps, the maximum direction of the two-dimensional backward flow.

The backward message at time T , when the goal is visible, is a Kronecker delta function on the goal and uniform in the action space as described in (8). Conversely, if the goal is not visible, the unknown area is considered as a generalized goal and the backward message at the end of the chain is a uniform probability distributed on all the unknown pixels.

As the flow progresses backward in time, the backward distribution fills in the space with non-trivial distributions in the state-action space. When the messages finally reach the agent, it has all the necessary information to decide on the next move that, in the absence of any visible goal, is attracted by the unknown areas (exploration phase).

The *forward flow* (not shown) can also be computed, but in this application does not provide direction information. The backward messages play instead the crucial role in directing the agent. The backward flow is essentially an "inverse diffusion process" that provides the crucial clues that a path planning algorithm can use to guide the decision maker towards the final goal. In the absence of a visible goal, the backward diffusion attracts the agent to explore the unknown regions. The backward flow could be interpreted, from a physics point of view, as a force field generated by the goal and deflected by the obstacles. Possible relations with a probabilistic field theory are beyond our scope, but they seem to suggest exciting theoretical analogies. Certainly, by visualizing the backward probability flow, the "social forces" used many times in modeling agents' behavior seem to emerge naturally!

Inspection of the backward flow also suggests an easy way to determine the minimum time to reach the goal and to overcome the assumption of a fixed time horizon: *run the backward flow until there is a non null probability on the starting point (Reachability Condition)*. This idea has been suggested in a previous paper, where additional simulations are reported (Palmieri et al., 2021). In the limited view situation, the minimum time horizon is calculated as the minimum time necessary to reach the closest point of the unknown area.

6. Finding the path

The results of the preliminary analysis of the planning system in a pure diffusion mode clearly indicates the crucial role that the backward

flow may play in directing the agent towards its goal. The *posterior flow* (not shown), which is proportional to the product of backward and forward distributions (5), could be computed to estimate, at various time steps, where the agent is most likely to be located, and these probabilities would summarize the best inference provided by the model. Examples are reported in Palmieri et al. (2021). However, the posterior distribution with no constraints may lead to disconnected paths (Palmieri et al., 2021).

Rather, the final inference on maximization of the posteriors, is computed following specific trajectories in a much easier way, using the backward flow through the evolution of motion, with no need to compute the forward flow.

Fig. 9 shows a section of the chain of Fig. 1 with the addition of the control policy block Π . The system starts at time $t-1$ from the current point $(S_{t-1} A_{t-1}) = (\bar{s}_{t-1} \bar{a}_{t-1})$, formally described as a forward message in the form of a Kronecker delta function. The transition model and the prior distribution $c_t(s_t a_t)$ give the temporary forward message

$$f_{(S_t A_t)^3}(s_t a_t) \propto p(a_t) p(s_t | S_{t-1} = \bar{s}_{t-1} A_{t-1} = \bar{a}_{t-1}) c_t(s_t a_t). \quad (9)$$

Information coming from the future is carried by the backward distribution $b_{(S_t A_t)^3}(s_t a_t)$ that combined with the above forward message gives the posterior

$$\bar{p}(s_t a_t) \propto f_{(S_t A_t)^3}(s_t a_t) b_{(S_t A_t)^3}(s_t a_t). \quad (10)$$

Decision on state and action at time t can now be made from $\bar{p}(s_t a_t)$ in a greedy deterministic fashion as

$$(\bar{s}_t \bar{a}_t) = \underset{(s_t a_t)}{\operatorname{argmax}} \bar{p}(s_t a_t), \quad (11)$$

or by sampling from it

$$(\bar{s}_t \bar{a}_t) \sim \bar{p}(s_t a_t). \quad (12)$$

Therefore, the final forward message will be formally

$$f_{(S_t A_t)^3}(s_t a_t) = \delta(s_t - \bar{s}_t a_t - \bar{a}_t), \quad (13)$$

as shown in Fig. 9.

In the case of deterministic transition model, the path is optimum if T is the minimum time and the rule is equivalent to Viterbi algorithm that maximizes the total likelihood (the proof is straightforward and is omitted here for brevity). In the random case, in which we sample from the posterior, a random realization is generated in the neighborhood of the optimum path. Clearly, when the posterior is highly concentrated around its maximum value, the results of the two approaches will be very similar. The random sampling approach may be useful in applications where we need to generate a collection of similar paths. Some examples are reported also in Palmieri et al. (2021). Comparisons of the sum-product framework used here to other approaches, such as Dynamic programming for deterministic and stochastic systems, have been recently discussed in detail in Palmieri et al. (2022).

7. Simulations

In the simulations reported here, we show how the proposed probability framework is an excellent candidate for providing a systematic

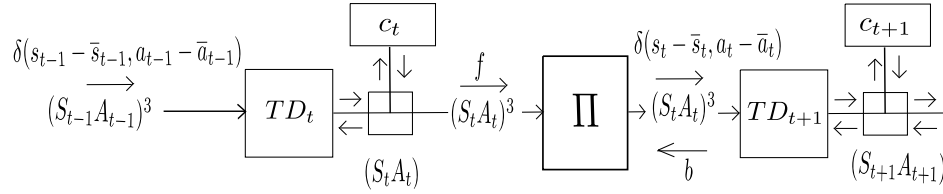


Fig. 9. A section of the factor graph model with the control policy Π included. Π produces an instance of the state-action pair at time t from a chosen state-action pair (delta distributions) at time $t-1$. The controller bases its choice on the forward message $f_{(S_t A_t)^3}$ and on the backward message $b_{(S_t A_t)^3}$. For the message notation around the junctions, see Fig. 2.

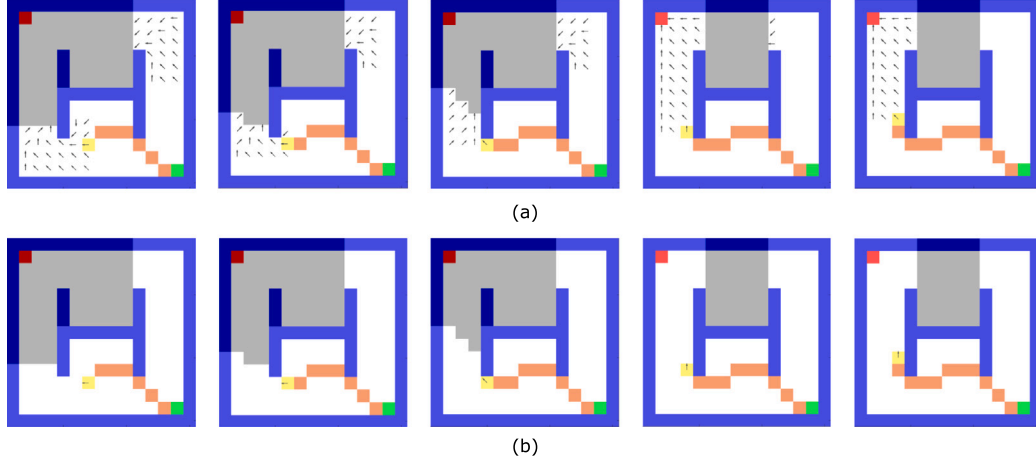


Fig. 10. An example of the algorithm running in exploration mode. The frames show, from the left to right, at time steps $t=8$, $t=9$, $t=10$, $t=11$ and $t=12$, the starting position (green), the current agent's positions (yellow), the hidden area (gray), the goals (red). The top row (a) shows the backward flow that is re-computed at every time step because the agent has acquired knowledge on more portions of the grid. The bottom row (b) shows the next state and action derived from the posterior distribution. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

way to treat the path planning problem in known and unknown environments. Animations of the presented simulations are made available on our website at <https://www.mlsptlab-unicampania.it/research/topics/path-modeling.html>.

7.1. Simple grid

Fig. 10 shows the time steps 8, 9, 10, 11 and 12 of the procedure for the grid of Fig. 7. At each time step, the backward flow is generated from the hidden area (gray) and is propagated until it reaches the current cell (yellow cell). The plots in Fig. 10(a) show the backward flow reaching the agent in its current position. The backward flow, together with the forward flow (not shown), allow the computation of the posterior probability at each step as in Eq. (10). This is depicted in the plots in Fig. 10(b), the arrow on the yellow cell represents the action that, at the current time and position, corresponds to the largest posterior probability that leads the agent to the next best state. Note that, at each time step, the unknown area changes because the agent has seen more space. Therefore, the backward flow and the minimum time T are re-computed at each step. Then, the procedure is repeated until the goal becomes visible (in Fig. 10 at step 11). At that point, the goal becomes the only source of the backward flow and the agent is driven towards it.

Note that, once the goal is discovered, the vision algorithm can be kept active for the agent to become aware also of other areas. This may be useful if multiple goals are placed in the scene, as in the parking example that is reported in the following.

7.2. Complex maze

In this simulation, we have challenged our algorithm to drive the agent in the rather complex maze of Fig. 11. In the pictures, for

simplicity, we show only the backward flows. It is interesting to see how the agent explores first the closest unknown areas (Fig. 11(a)). When the exploration is concluded in the first neighbourhood without discovering the goal, the agent goes to next unknown areas (Figs. 11(b), 11(c)) until it sees the goal. At this point, just as in the other examples, the agent is attracted only by the goal, that now is the only part of the map that generates the backward flow. Clearly now, if the vision sensor is kept active, the agent on its way to the goal can also acquire knowledge on other parts of the map, perhaps discovering a new goal, as in the parking garage example that follows.

7.3. Parking garage

Another simulation scenario is depicted in Fig. 12 where we have imagined that an autonomous car is looking for a parking spot in a garage. Since there may be multiple available spots, this is a case of multiple to-be-discovered goals. The agent progressively explores the various spots to “see” if they are free. When the agent discovers a free spot on the right side of the map, it starts moving towards it, but, after a few steps, the updates on the visibility map make another free parking spot visible. Since the new goal is closer than the first one, the agent is attracted towards it. The algorithm, not having access to the complete map, finds the minimum path to the goal using the information it gradually acquires. The new visible free parking lots are evaluated based on the current position of the agent and not with respect to its initial position. In this sense the algorithm has found an optimal solution to the problem.

7.4. Comparison to the agent with the perfect knowledge of the map

A natural question arises from the previous simulations: how efficiently the agent can move to discover and then reach the goal in

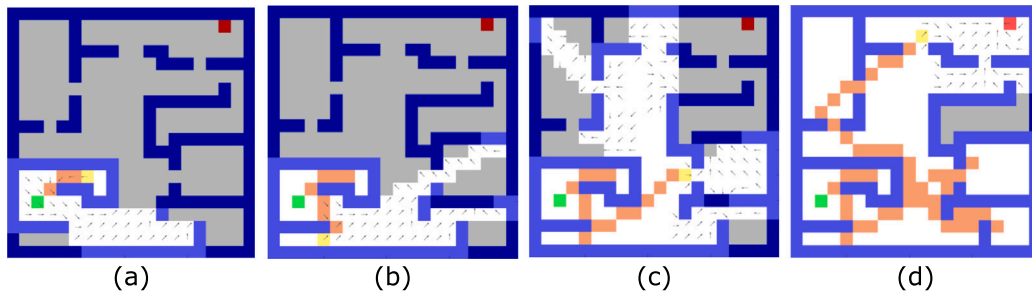


Fig. 11. Finding a goal in a maze. The frames show the agent's positions, the hidden area, the goals, the path and the backward flow at $t = 5$ (a), $t = 12$ (b), $t = 20$ (c) and $t = 55$ (d). The red, green and yellow cells represent, respectively, the goal, the starting position and the current position. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

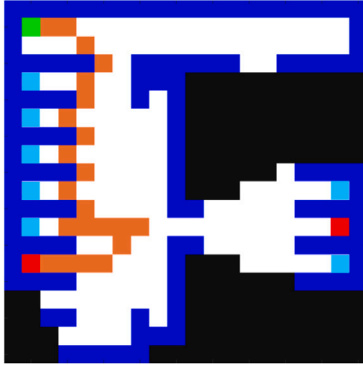


Fig. 12. Parking garage simulation. In light blue the occupied spots and in red the free ones. Starting from the green cell, the agent looks for an available spot. On its way to the first-discovered spot (right on the map), the agent sees another free spot on the left, which, becoming a new source of the backward flow, attracts the agent more as it is closer. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

comparison to an agent that does its path planning knowing the whole map and the goal location? To address this question we have performed a comparison between the two situations: path planning in a completely known and a completely unknown environment. Fig. 13 shows a grid where we have carefully hidden the goal. The agent at time $t = 12$, attracted by the hidden area and equipped with the vision sensor, has discovered some regions, but it has not seen the goal yet.

Considering the same grid of Fig. 13, we have evaluated the behavior of the agent that knows the map and the goal exactly, and we compare the results as depicted in Fig. 14. As expected, the exploration phase has caused the agent to take more time and a longer path to reach the goal (32 steps versus 22 if the environment was completely known). The same comparison has been performed for all simulations presented in this work. Table 1 summarizes the number of steps required by the algorithm to reach the goal when the environment is completely known and when it is not. Moreover, the elapsed time for all simulations is reported as well. The simulations have been run on a MacBook Pro 2018 (CPU: 2,2 GHz Intel Core i7 6 cores, RAM: 16 GB RAM, GPU: Radeon Pro 555X 4 GB, OS: MacOS 13.0) using MATLAB R2021b. A factor of 10 or more speedup is expected when implemented in programming languages, such as C.

8. Conclusions and future work

Probability propagation on a graph that models motion on a grid can be a very powerful and flexible framework for providing solutions to the path planning problem even in completely unknown environments. We have shown how the probability flow plays the crucial role in guiding the agent first through exploration and then to the discovered



Fig. 13. Initial path and visibility map at $t = 12$. The agent has acquired knowledge on some areas, but the goal is still invisible.

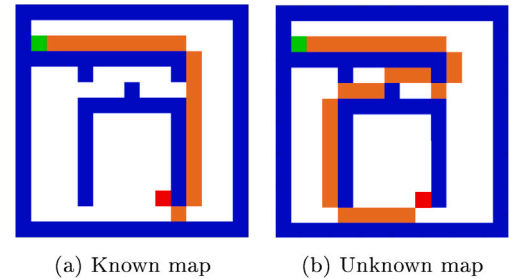


Fig. 14. Paths obtained with fully known map (a) and after exploration (b). The green and red cells represent, respectively, the starting position and the goal. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1

Comparison between steps and seconds necessary to reach the goal in the presented simulations when the agent knows or not the map.

	Known environment	Unknown environment
H maze (Fig. 10)	20 [4.6 s]	20 [64.2 s]
Simple Maze (Fig. 14)	22 [5.1 s]	32 [105.6 s]
Complex Maze (Fig. 11)	24 [9.0 s]	62 [400.5 s]
Parking Garage (Fig. 12)	18 [6.1 s]	24 [150.3 s]

goal(s). Map and goals not known initially are handled via injection of information in a dynamically changing backward flow.

In this paper, we have limited our attention to a single agent, but the extension to multiple agents is straightforward; the system can run multiple probability flows in parallel and dynamically adapt the behavior of each agent to the changing scene. Some examples of multiple agents following their probability flow can be found in our previous papers (Di Gennaro, Buonanno, Fioretti, Verolla, Pattipati, & Palmieri, 2022; Palmieri et al., 2021). One of the crucial issues that characterizes this probability framework is the determination of the

time horizon. We have shown how this can be done dynamically using a reachability criterion with the backward flow.

In summary, we can state that the probability propagation framework may provide a very powerful method for path planning in complex and unknown scenarios. The study of the backward flow, in comparison to other techniques proposed in the literature and briefly reviewed in the introduction, reveals an intriguing, purpose-driven and unifying view.

Even though our preliminary experiments are on artificial grids, the emerging behaviors are realistic and we are pursuing the application of this method to experimental trajectories on real maps with human agents and vehicles.

CRedit authorship contribution statement

Giovanni Di Gennaro: Conceptualization, Methodology, Writing – original draft, Writing – review & editing, Visualization, Software. **Amedeo Buonanno:** Conceptualization, Methodology, Writing – original draft, Writing – review & editing, Visualization. **Giovanni Fioretti:** Visualization, Software, Writing – review & editing. **Francesco Verolla:** Visualization, Software, Writing – review & editing. **Francesco A.N. Palmieri:** Conceptualization, Methodology, Visualization, Supervision, Writing review & editing, Funding acquisition. **Krishna R. Pattipati:** Conceptualization, Methodology, Visualization, Supervision, Writing – review & editing, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Visibility map computation

Suppose that, at a generic time step, we need to establish if a cell (c, d) is visible or not, when the agent is located at coordinates (a, b) and looks along direction $dirA$. The viewing angle is calculated from the center of the agent's pixel $(x, y) = (a + 0.5, b + 0.5)$, which also defines the center of a new reference system, while the destination cell is considered through its four vertices in the new reference system $\{(i, j)_k, k = 1 \dots 4\} = \{(c - x, d - y), (c - x + 1, d - y), (c - x, d - y + 1), (c -$

$x + 1, d - y + 1)\}$. The reference system has to be changed, through a rotation matrix.

$$R_\theta = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

where $\theta = dirA$.

After the transformation, the cells have coordinates $(\hat{i}, \hat{j})_k, k = 1, \dots, 4$. By computing

$$\alpha_k = \arccos \frac{\hat{j}_k}{\sqrt{\hat{i}_k^2 + \hat{j}_k^2}}$$

for $k = 1 \dots 4$, we take the two extreme angles $\alpha_{min} = \min_k \alpha_k$ and $\alpha_{max} = \max_k \alpha_k$, which define as the extremes of the corner sector to be evaluated for the desired cell. Considering a viewing angle α_{vis} , we assume that the agent can see $\hat{\alpha}_{vis} = \frac{\alpha_{vis}}{2}$ both to his left and to his right.

If $\alpha_{min} = \alpha_{max}$, the cell to be checked will be on the ordinate axis of the new reference system, imposing appropriate considerations, but, in general, if one of the relations $\hat{\alpha}_{min} < \hat{\alpha}_{vis} < \hat{\alpha}_{max}$ (partially visible cell) or $\hat{\alpha}_{vis} < \hat{\alpha}_{min}$ (fully visible cell), is satisfied, the cell with index (c, d) in the visibility Map gets value 1, which implies that it is visible and “known”. This operation is done at each time t for each cell of the grid.

The directions allowed to look around are multiples of 45° with respect to the vertical axis. The algorithm can work in several ways just by changing the values of the viewing angle and the viewing distance.

Appendix B. Simulation with modified viewing aperture

Fig. 15 shows a comparison between two agents with a different viewing aperture: 360° for Fig. 15(a) and 120° for Fig. 15(b). In this simulation the computed path is the same but the discovered areas are different due to narrower viewing angle of the second agent. In fact, in the final steps, the higher part of the map continues to be unknown in Fig. 15(b) while it is completely observed in Fig. 15(a). The importance of this simulation is linked to the fact that another hypothetical goal in the gray area of the map could not be discovered by the second agent.

Data availability

No data was used for the research described in the article.

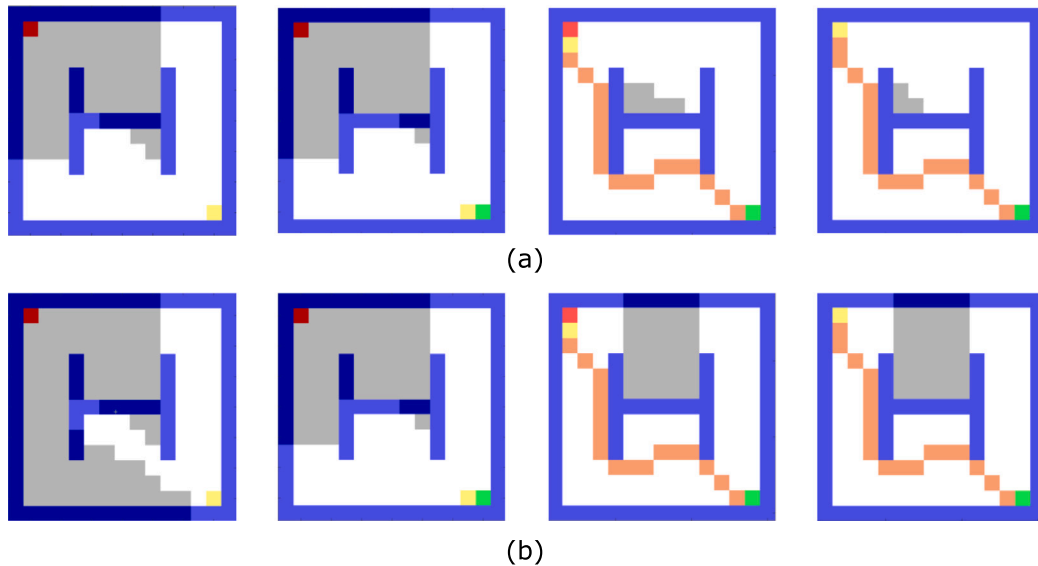


Fig. 15. Comparison between two agents with a different viewing aperture: 360° (a) and 120° (b). From the left to right, time steps $t = 1$, $t = 2$, $t = 19$ and $t = 20$.

References

- Abbadì, A., & Prenosil, V. (2015a). Collided path replanning in dynamic environments using RRT and cell decomposition algorithms. *Modelling and Simulation for Autonomous Systems*, 9055, 131–143.
- Abbadì, A., & Prenosil, V. (2015b). Safe path planning using cell decomposition approximation. *International Conference on Distance Learning, Simulation and Communication*, 8–14.
- Alexandre, Z., Oleg, S., & Giovanni, P. (2018). An information-theoretic perspective on the costs of cognition. *Neuropsychologia*, 123, 5–18.
- Amato, N., & Wu, Y. (1996). A randomized roadmap method for path and manipulation planning. *Proceedings of IEEE International Conference on Robotics and Automation*, 1(1), 113–120.
- Attias, H. (2003). Planning by probabilistic inference. *Proc. of the 9th Int. Workshop on Artificial Intelligence and Statistics*.
- Baltieri, M., & Buckley, C. (2017). An active inference implementation of phototaxis: arXiv:1707.01806v1 [q-bio.NC] 6 2017. ARXIV.
- Bertsekas, D. (2019). *Reinforcement learning and optimal control*. Athena Scientific.
- Buckley, C., Kim, C., McGregor, S., & Seth, A. (2017). The free energy principle for action and perception: a mathematica review. *Journal of Mathematica Psychology*, 81, 55–79.
- Buonanno, A., & Palmieri, F. (2015a). Two-dimensional multi-layer factor graphs in reduced normal form. In *Proceedings of the international joint conference on neural networks, IJCNN2015* (pp. 12–17). Killarney, Ireland.
- Buonanno, A., & Palmieri, F. (2015b). Simulink implementation of belief propagation in normal factor graphs. *Smart Innovation, Systems and Technologies*, 37, 11–20.
- Canny, J., & Donald, B. (1987). *Simplified voronoi diagrams*. Massachusetts Institute of Technology.
- Castaldo, F., Palmieri, F., & Regazzoni, C. (2014). Application of Bayesian techniques to behavior analysis in maritime environments. *Advances in Neural Networks: Computational and Theoretical Issues: Smart Innovation, Systems and Technologies*, 37, 175–183.
- Choset, H., Lynch, K., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L., et al. (2005). *vol. 5, Principles of robot motion: theory, algorithms, and implementations*. MIT Press.
- Coscia, P., Ballan, L., Palmieri, F., Alahi, A., & Savarese, S. (2020). Linear artificial forces for human dynamics in complex contexts: Smart innovation. *Systems and Technologies. Neural Approaches To Dynamics of Signal Exchanges*, 151, 21–33.
- Coscia, P., Braca, P., Millefiori, L., Palmieri, F., & Willett, P. (2018). Multiple ornstein-uhlenbeck processes for maritime traffic graph representation. *IEEE Transactions on Aerospace and Electronic Systems*, 54, 2158–2170.
- Coscia, P., Castaldo, F., Palmieri, F., Alahi, A., Savarese, S., & Ballan, L. (2018). Long-term path prediction in urban scenarios using circular distributions. *Image and Vision Computing*, 69, 81–91. (Editors Choice Award 2021).
- Coscia, P., Castaldo, F., Palmieri, F., Ballan, L., Alahi, A., & Savarese, S. (2016). Point-based path prediction from polar histograms. *Proceedings of the 19th International Conference on Information Fusion (FUSION 2016)*, 196, 1–1967.
- Coscia, P., N. Palmieri, F., Braca, P., Millefiori, L., & Willett, P. (2018). Unsupervised maritime traffic graph learning with mean-reverting stochastic processes. *2018 21st International Conference on Information Fusion (FUSION)*, 1822–1828.
- Di Gennaro, G., Buonanno, A., Fioretti, G., Verolla, F., Pattipati, K., & Palmieri, F. (2022). Probabilistic inference and dynamic programming: A unified approach to multi-agent autonomous coordination in complex and uncertain environments. *Frontiers in Physics*, 10.
- Di Gennaro, G., Buonanno, A., & Palmieri, F. (2021). Optimized realization of Bayesian networks in reduced normal form using latent variable model. *Soft Computing*, 25, 7029–7040.
- Everett, M., Miller, J., & How, J. (2019). Planning beyond the sensing horizon using a learned context. *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1064–1071.
- Gasparetto, A., Boscaroli, P., Lanzutti, A., & Vidoni, R. (2015). Path planning and trajectory planning algorithms: A general overview. *Mechanisms and Machine Science*, 29, 3–27.
- Hart, P., Nilsson, N., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4, 100–107.
- Hwang, Y., & Ahuja, N. (1992). A potential field approach to path planning. *IEEE Transactions on Robotics and Automation*, 8, 23–32.
- Kaplan, R., & Friston, K. (2018). Planning and navigation as active inference. *Biological Cybernetics*, 112, 323–347.
- Kappen, H., Gomez, V., & Opper, M. (2013). Optimal control as a graphical model inference problem. *Proceedings of the Twenty-Third International Conference on Automated Planning and Scheduling*.
- Koenig, S., & Likhachev, M. (2005). Fast replanning for navigation in unknown terrain. *IEEE Transactions on Robotics*, 21, 354–363. <http://dx.doi.org/10.1109/TRO.2004.838026>.
- Lavalle, S. (1998). Rapidly-exploring random trees: A new tool for path planning. *The Annual Research Report*.
- Law, G. (2013). Quantitative comparison of flood fill and modified flood fill algorithms. *International Journal of Computer Theory and Engineering*, 503–508.
- Levine, S. (2018). Reinforcement learning and control as probabilistic inference: Tutorial and review. 20, (2018), ArXiv:1805.00909v3 [cs.LG].
- Loeliger, H. (2004). An introduction to factor graphs. *IEEE Signal Processing Magazine*, 21, 28–41.
- Loeliger, H., Dauwels, J., Hu, J., Korl, S., Li, P., & Kschischang, F. (2007). The factor graph approach to model-based signal processing. *Proceedings of the IEEE*, 95, 1295–1322.
- Lozano-Perez, T. (1983). Spatial planning: A configuration space approach. *Institute of Electrical and Electronics Engineers. Transactions on Computers*, C-32, 108–120.
- Lozano-Pérez, T., & Wesley, M. (1979). An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22, 560–570.
- Nadira, S., Omar, R., & Hailma, C. (2016). Potential field methods and their inherent approaches for path planning.
- Naghizadeh, A., Berenjian, S., Margolis, D., & Metaxas, D. (2020). GNM: GridCell navigational model. *Expert Systems with Applications*, 148, Article 113217.
- Nair, S., Zhu, Y., Savarese, S., & Li, F. (2019). Causal induction from visual observations for goal directed tasks. 3, (2019), ArXiv:1910.01751v1 [cs.LG].
- Niroui, F., Sprenger, B., & Nejat, G. (2017). Robot exploration in unknown cluttered environments when dealing with uncertainty. *2017 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*, 224–229.
- Nissoux, C., Simeon, T., & Laumond, J. (1999). Visibility based probabilistic roadmaps. *Proceedings 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human and Environment Friendly Robots with High Intelligence and Emotional Quotients (Cat. No. 99CH36289)*, 3, 1316–1321.
- Ortega, P., & Braun, D. (2013). Thermodynamics as a theory of decision-making with information-processing costs. *Proceedings of the Royal Society A*.
- Palmieri, F. (2016). A comparison of algorithms for learning hidden variables in Bayesian factor graphs in reduced normal form. *IEEE Transactions on Neural Networks and Learning Systems*, 27, 2242–2255.
- Palmieri, F., Pattipati, K., Di Gennaro, G., Fioretti, G., Verolla, F., & Buonanno, A. (2022). A unifying view of estimation and control using belief propagation with application to path planning. *IEEE Access*, 10, 15193–15216.
- Palmieri, F., Pattipati, K., Fioretti, G., Di Gennaro, G., & Buonanno, A. (2021). Path planning using probability tensor flows. *IEEE Aerospace and Electronic Systems Magazine*, 36.
- Palmieri, F., Pattipati, K., Fioretti, G., Verolla, F., Di Gennaro, G., & Buonanno, A. (2020). Exploration/exploitation in path planning using probability propagation. *Proceeding of the IEEE International Conference on Robotics and Automation (ICRA 2020), 2nd Workshop on Long-Term Human Motion Prediction (LHMP)*.
- Parr, T., & Friston, K. (2018a). Generalised free energy and active inference: can the future cause the past?. *BioARXIV*.
- Parr, T., & Friston, K. (2018b). The anatomy of inference: Generative models and brain structure. *Frontiers in Computational Neuroscience*, 12.
- Santos, V., Otero, F., Johnson, C., Osorio, F., & Toledo, C. (2020). Exploratory path planning for mobile robots in dynamic environments with ant colony optimization. *Genetic and Evolutionary Computation Conference (GECCO '20)*.
- Stentz, A. (1993). Optimal and efficient path planning for unknown and dynamic environments. *International Journal of Robotics and Automation*, 10, 89–100.
- Takahashi, O., & Schilling, R. (1989). Motion planning in a plane using generalized voronoi diagrams. *IEEE Transactions on Robotics and Automation*, 5, 143–150.
- Tijharjadi, S., Wijaya, M., & Setiawan, E. (2017). Optimization maze robot using a* and flood fill algorithm. *International Journal of Mechanical Engineering and Robotics Research*, 6, 366–372.
- Todorov, E. (2009). Efficient computation of optimal actions. *Proceedings of the National Academy of Sciences*, 106, 11478–11483.
- Toussaint, M. (2009). Probabilistic inference as a model of planned behavior. *Kunstliche Intelligenz*, 3.
- Toussaint, M., & Storkey, A. (2006). Probabilistic inference for solving discrete and continuous state Markov decision processes. *ICML '06 Proceedings of the 23rd International Conference on Machine Learning*, 2006, 945–952.
- Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. *Proceedings of the 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 146.
- Yao, Q., Zheng, Z., Qi, L., Yuan, H., Guo, X., Zhao, M., et al. (2020). Path planning method with improved artificial potential field—A reinforcement learning perspective. *IEEE Access*, 8, 135513–135523.
- Zhu, Q., Hu, J., Cai, W., & Henschen, L. (2011). A new robot navigation algorithm for dynamic unknown environments based on dynamic path re-computation and an improved scout ant algorithm. *Applied Soft Computing*, 11, 4667–4676.