Learning to Act for Perceiving in Partially Unknown Environments

Leonardo Lamanna[1], **Mohamadreza Faridghasemnia**[2], **Alfonso Gerevini**[3], **Alessandro Saetti**[3], **Alessandro Saffiotti**[2], **Luciano Serafini**[1] and **Paolo Traverso**[1]
[1]Fondazione Bruno Kessler, Trento, Italy
[2]Center for Applied Autonomous Sensor Systems, University of Örebro, Sweden
[3]Department of Information Engineering, University of Brescia, Italy
{llamanna, serafini, traverso}@fbk.eu, {mohamadreza.farid, alessandro.saffiotti}@oru.se,
{alfonso.gerevini, alessandro.saetti}@unibs.it

Abstract

Autonomous agents embedded in a physical environment need the ability to correctly perceive the state of the environment from sensory data. In partially observable environments, certain properties can be perceived only in specific situations and from certain viewpoints that can be reached by the agent by planning and executing actions. For instance, to understand whether a cup is full of coffee, an agent, equipped with a camera, needs to turn on the light and look at the cup from the top. When the proper situations to perceive the desired properties are unknown, an agent needs to learn them and plan to get in such situations. In this paper, we devise a general method to solve this problem by evaluating the confidence of a neural network online and by using symbolic planning. We experimentally evaluate the proposed approach on several synthetic datasets, and show the feasibility of our approach in a real-world scenario that involves noisy perceptions and noisy actions on a real robot.

1 Introduction

In embodied AI agents, an amount of knowledge may emerge from the interaction with the environment. In this manner, the knowledge provided to the agent at the beginning, produced by some off-line process, may be extended and adjusted by interacting with the environment. Symbolic planning is an effective technique for generating sequences of actions (i.e. plans) to achieve goals. However, in order to use symbolic planning in embodied AI, agents need to map their high throughput low-level perceptions of the environment (e.g., images) into properties of symbolic states. Moreover, embodied agents should be able to dynamically and autonomously evaluate and improve their perceptual abilities by actively exploring their environment. This is also a well-known challenge in the field of active perception [Bohg *et al.*, 2017]. In partially unknown environments, certain properties can be perceived only in specific situations. For instance, to check if a cup is full of coffee, an agent, equipped with a camera, needs to turn on the light and look at the cup from the top; it is useless to observe the cup from the side. An open challenge is whether an agent can automatically recognize the situations from which a useful perception can be made, and how such situations can be reached by acting in the real world.

In this paper, we devise a general method to solve this problem. We model the agent's perception with a set of (deep) neural networks that map perceptions (e.g., an image of a cup), into properties of symbolic states (e.g., the cup is empty). We propose a method to identify, via clustering, the situations from which the predictions of the neural networks are correct with a confidence higher than a given threshold. Each cluster is associated with an action that can be executed in the identified situations and reaches a proposition considered as the goal of a symbolic planning problem. In this way, the agent can use a symbolic planner to synthesise plans reaching situations where observations are more informative.

We define a framework that allows us to formalize the problem of learning the set of situations in which a perception model of an agent reaches a desired level of confidence. In such a framework, we suppose that the environment behaves like a probabilistic transition system unknown by the agent. The agent knowledge about the environment is represented in terms of a symbolic planning domain. The agent has access to the current state of the environment by a so-called *perception function* that maps observations of the environment into a set of (belief) states of the symbolic model. We develop an algorithm that returns a set of belief states of the symbolic model of the agent, where the prediction of the perception is more reliable. We transform the epistemic planning problem of reaching a belief state into a classical planning problem, similarly to [Belle *et al.*, 2023; Petrick and Bacchus, 2002; Bonet and Geffner, 2011]. We then use a classical planner to find a plan that leads the agent to a useful situation.

We experimentally evaluate the proposed approach by showing that observing from the useful situations found by our method results in a higher accuracy w.r.t. making the prediction directly from the perception of the current environment state. In our experiments, we consider several synthetic datasets, and a real-world scenario that involves noisy perceptions and noisy actions on a real robot.

2 Related Work

A wide variety of approaches and applications have been proposed for enhancing robotic agents with active learn-

ing techniques [Kulick *et al.*, 2013; Cakmak *et al.*, 2010; Chao *et al.*, 2010; Ribes *et al.*, 2015; Hayes and Scassellati, 2014; Huang *et al.*, 2010; Ashari and Ghasemzadeh, 2019; Taylor *et al.*, 2021]. In these works, robotic agent improve their skills or learn new concepts by collecting and labeling data in an online way. All these methods label data either by means of human supervision or by relying on the prediction of a pre-trained model. In [Ugur and Piater, 2015; Lamanna *et al.*, 2023], perceptions are generated by executing actions and labeled with the effect specified in a symbolic planning domain. None of these methods consider the problem of estimating the quality of the obtained perception model under different conditions to discriminate when perception is trustable, which is the main objective of this paper.

Concerning the problem of learning planning domains from continuous observations, the approach by Migimatsu and Bohg (2022) learns to map images into the truth values of predicates of planning states. Differently from us, their approach is offline and requires a sequence of images labeled with actions, while our approach plans for generating this sequence online. We share the idea of learning state representations through interaction with the work by Pinto *et al.* (2016), where they learn visual representations by manipulating objects on a table. They learn these representations in an unsupervised way, through a CNN trained on a dataset generated by interacting with objects. However, the learned representations lack interpretation and they are not suitable for applying symbolic planning. Other works have addressed the problem of learning planning domains from perceptions in the form of high-dimensional raw data (such as images), see, e.g., [Asai and Fukunaga, 2018; Asai, 2019; Janner *et al.*, 2019; Dengler *et al.*, 2021; Konidaris *et al.*, 2018; Occhipinti *et al.*, 2022]. In all the above-mentioned works, the abstract planning domain is obtained by offline pre-training, and the mapping between perceptions to the abstract model is fixed, while we learn and adapt this mapping online.

Our work shares some similarity with planning under partial observability (e.g., [Bertoli *et al.*, 2006; Bonet and Geffner, 2014]): some state variables might not be always observable. However in planning under partial observability the model describes which variables are/are not observable in which state, while in our approach, this information is learned online. Planning with perceptions is also considered in epistemic planning [Belle *et al.*, 2023; Petrick and Bacchus, 2002]. In epistemic planning, the domain state encodes the current belief of the agent by means of epistemic literals; e.g., Kp (resp. $\neg Kp$) for some propositional variable p states that the agent knows (resp. does not know) that p is true. Epistemic literals can appear in the action preconditions and effects, which allow for a uniform treatment of sensing actions and "physical" actions. Epistemic planning does not consider the problems of perceiving from continuous features, and of learning and exploiting perception function. However, we exploit in our methodology a common practice in epistemic planning that transforms planning under partial observability into epistemic planning under full observability.

Finally, active perception is a strongly studied area in robotics [Bajcsy *et al.*, 2018]. Most work in this area deals with reaching the best view points for a vision sensor [Grotz, 2021; Zeng *et al.*, 2020], although some works consider more general observation conditions like lightning or blurring, [Tarabanis *et al.*, 1995], as we do here. The view planning problem can be summarized by: sampling candidate viewpoints, evaluating the quality of the information obtained from them, and choosing the best one. While we share the motivation and the main procedure, we map the low-level data obtained from viewpoints into abstract states of a symbolic planning domain, evaluate the quality of this mapping, and we use symbolic planning to choose the best next viewpoint. More importantly, most approaches to view planning assume that the relation between viewpoints and observability is given: in our work, this relation is learned.

3 Problem Formulation

We model the partially observable environment where the agent operates as $\text{Env} = (S_{\text{Env}}, A_{\text{Env}}, \gamma_{\text{Env}}, O_{\text{Env}}, \text{obs})$, where $(S_{\text{Env}}, A_{\text{Env}}, \gamma_{\text{Env}})$ is a nondeterministic automaton composed by the set S_{Env} of environment states, the set A_{Env} of agent "low-level" actions executable in the environment, and the transition relation $\gamma_{\text{Env}} \subseteq S_{\text{Env}} \times A_{\text{Env}} \times S_{\text{Env}}$; O_{Env} is the set of all possible observations from the environment; and $\text{obs} : S_{\text{Env}} \to O_{\text{Env}}$ is a deterministic observation function.

Example 1. *Consider an agent in a room that wants to see the picture on the wall in front of it, but its view is occluded by a circular object. The agent has the possibility to change its viewpoint so that the object only partially occlude the picture. Furthermore, the light in the room can be turned on or off by the agent. The set O_{Env} of observations of the agent is a set of images of size $w \times h$; some examples of observations are shown in Figure 1.*

(a) (b) (c) (d) (e)

Figure 1: Examples of observations of the environment states where a handwritten digit three is shown on the wall. In (a) the light is off, while in (b)–(e) it is on; in (b) the occluding object is in the center, in (c) is on the edge, and in (d) and (e) is in a corner.

The agent Ag is modeled by $\text{Ag} = (\mathcal{M}_{\text{Ag}}, B_{\text{Ag}}, \text{ex}_{\text{Ag}}, f_{\text{Ag}})$, where $\mathcal{M}_{\text{Ag}} = (S_{\text{Ag}}, A_{\text{Ag}}, \gamma_{\text{Ag}})$ is a finite automaton, S_{Ag} is a set of agent states, A_{Ag} are agent "high-level" actions, and γ_{Ag} is a transition relation contained in $S_{\text{Ag}} \times A_{\text{Ag}} \times S_{\text{Ag}}$. The agent state is specified in terms of a set of state variables $\boldsymbol{x} = (x_1, \ldots, x_n)$. The automaton $\mathcal{M}_{\text{Ag}}$ constitutes the internal abstract representation of the environment adopted by the agent. We distinguish between S_{Ag} and S_{Env} because the internal agent state may abstract away parts of the environment state that are irrelevant for the agent. For instance, the agent state in Example 1 does not consider the room temperature.

Example 2 (Agent's transition system)**.** *The set of states S_{Ag} of the agent can be described by three states variables:* digit *that takes values in* $\{0, \ldots, 9\}$*,* occl *that takes values in* $\{(i, j) \mid 0 \leq i \leq w,$ *and* $0 \leq j \leq h\}$*, and* light *that takes*

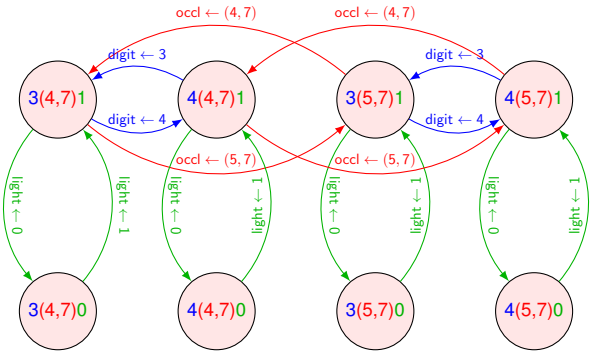

Figure 2: A portion of the transition system of the agent.

boolean values. For instance, the state 3(4,7)0 is a state in S_{Ag} that represents the situation where the digit 3 is on the wall, the occluding object is centered in (4,7), and the light is off. The set of high-level actions in A_{Ag} allows the agent to switch the light on and off, to change the picture on the wall with another digit, and to move the occluding object in different positions. More precisely, for every state variable $x \in \{\mathsf{digit}, \mathsf{occl}, \mathsf{light}\}$, the action effect $x \leftarrow v$, where v is a value in the domain of x, sets the value of x to v. We suppose that all the actions but switching the light on/off, can be performed only if the light is on. A portion of the transition system of the agent, which considers only digits 3 and 4 and the occluding object in (4,7) or (5,7), is shown in Figure 2.

In addition to the internal representation of the environment, the agent is equipped with a subset of states $B_{\text{Ag}} \subseteq S_{\text{Ag}}$ that represent the belief state of the environment. Furthermore, the agent can carry out its high-level actions in A_{Ag} by executing the corresponding low-level actions in A_{Env}. This capacity is represented by the function ex_{Ag}, which associates to each pair $(s, a) \in S_{\text{Ag}} \times A_{\text{Ag}}$ a program $\text{ex}_{\text{Ag}}(s, a)$ of actions in A_{Env} executable in Env, such as, e.g., in hierarchical task networks [Erol *et al.*, 1994]. Since the environment is nondeterministic, the state of the environment resulting from the execution of $\text{ex}_{\text{Ag}}(s, a)$ is also nondeterministic. Finally, the agent is associated with a perception function $f_{\text{Ag}} : O_{\text{Env}} \to Pr(S_{\text{Ag}})$ that returns a probability distribution $f_{\text{Ag}}(s \mid o)$ on a state s in S_{Ag} given an observation o in O_{Env}.

Example 3 (Agent perception function)**.** *Continuing with our example, we can represent the perception function of the agent with three separate perception functions, one for each state variable, i.e., $f_{Ag} = (f_{\mathsf{digit}}, f_{\mathsf{occl}}, f_{\mathsf{light}})$, defining*

$$f_{Ag}(s \mid o) = (f_{\mathsf{digit}}(s_{\mathsf{digit}} \mid o), f_{\mathsf{occl}}(s_{\mathsf{occl}} \mid o), f_{\mathsf{light}}(s_{\mathsf{light}} \mid o))$$

where s_x for every $x \in \{\mathsf{digit}, \mathsf{occl}, \mathsf{light}\}$ denotes the value of x in the state s.

The agent state variables are grouped into control variables and observable variables. The control variables are such that their values can be known in every state, while the values of the observable variables can be known only in some states. For example, variable light is a control variable because its value can be always known; while variable digit is observable because when the light is off its value is unknown. Therefore, in general, not all environment observations can provide to the agent sufficient information to discriminate, by means of f_{Ag}, the values of the observable variables. Indeed, as for our example, if the environment is observed when the light is off, the agent cannot discriminate the value of the observable state variable digit by means of f_{digit}. Therefore, to rely on the prediction of the perception function f_{Ag}, an agent should know (believe) that the environment is in a state where such a prediction is sufficiently confident. This leads us to the definition of confidence of a perception function in a belief state $B_{\text{Ag}} \subseteq S_{\text{Ag}}$.

Definition 1 (Confidence in a state)**.** *The confidence of the agent perception function f_{Ag} in a state $s \in S_{Ag}$, denoted by conf(f_{Ag}, s), is defined as*

$$\frac{1}{|\gamma_{Ag}^{-1}(s)|} \sum_{(s',a) \in \gamma_{Ag}^{-1}(s)} \mathbb{E}_{o \sim ex_{Ag}(s',a)} f_{Ag}(s \mid o) \qquad (1)$$

where $\gamma_{Ag}^{-1}(s) = \{(s', a) \mid (s', a, s) \in \gamma_{Ag}\}$.

Definition 2 (Confidence in a belief state)**.** *The confidence of the agent perception function f_{Ag} in a belief state $B_{Ag} \subseteq S_{Ag}$ is the average of the confidence of f_{Ag} in each state $s \in B_{Ag}$.*

The intuitive reading of (1) is that, for every way to reach s, i.e., for every $(s', a, s) \in \gamma_{\text{Ag}}$, $\mathbb{E}_{o \sim \text{ex}_{\text{Ag}}(s',a)} f_{\text{Ag}}(s \mid o)$ is the expectation of the agent believing to be in s from the observation o obtained after executing a in s'. Since the execution of an action is non-deterministic, the observation o obtained after an action execution is a random variable. The higher the expectation value, the higher the agreement between the abstract model, the perception function and the concrete execution of the high-level action a. This expectation is averaged on all the possible ways of reaching s.

The agent perception function can be modeled as $f_{\text{Ag}} = (f_{x_1}, \ldots, f_{x_n})$. We denote by $\boldsymbol{x}_{-i} = \boldsymbol{v}_{-i}$ the assignments to all the state variables but x_i. The confidence of f_{x_i} conditioned to $\boldsymbol{x}_{-i} = \boldsymbol{v}_{-i}$ is defined as the confidence of f_{Ag} in $S_{\boldsymbol{x}_{-i}=\boldsymbol{v}_{-i}} = \{s \in S_{\text{Ag}} \mid s_{x_j} = v_j \ \forall j \neq i\}$.

Definition 3 (Viewpoint)**.** *A* viewpoint *with confidence at least $t \in [0, 1]$ for a state variable x is a belief state $B_{Ag} \subseteq S_{Ag}$ such that conf$(f_x, B_{Ag}) \geq t$.*

Example 4. *Suppose that f_{digit}, f_{light} and f_{occl} are neural networks trained with labelled observations like those shown in Figure 1. For instance, the network f_{light} is trained with a set of observations similar to the first shown in Figure 1 and labeled with 0, and a set of observations similar to the other four (possibly with different digits and different positions of the occluding object) and labeled with 1. Similarly, the network f_{occl} is trained with observations like (a) and (b) labeled with $(\frac{w}{2}, \frac{h}{2})$ (the occluding object is in the center, though in (a) it is not visible), with observations similar to (c) with label $(\frac{w}{2}, h)$, and with observations similar to (d) and (e) with label equal to (w, h) and $(w - 4, 4)$, respectively. One can reasonably expect that the confidence of f_{light} is high for every belief state B since the visual difference of the observations, independently from the digit and the occluding object, is evident. Differently, the confidence of f_{occl} will be high only in the belief states where the light is on (i.e, the state variable* light *has*

value 1*). Finally,* f_{digit} *will be highly confident when the light is on and the occluding object is in a corner, less confident when the object is on the edge, and not confident when it is in the middle. Globally the confidence of* f_{Ag} *will be the best when the light is on, and the occluding object is on a corner, i.e., in the belief set* $B_{\mathsf{corner}} = \{s \in S_{Ag} \mid s_{\mathsf{light}} = 1$ *and* $s_{\mathsf{occl}} \in \{(0,0),(w,0),(0,h),(w,h)\}\}$.

Knowing the belief states where the perception function is sufficiently confident is a crucial aspect for the agent to correctly perceive the current state of the environment. Indeed, if f_{Ag} has a high confidence on B_{Ag}, then the agent can rely on f_{Ag} to derive its current state from the environment observation, e.g., by selecting $s^* = \text{argmax}_s(f_{\text{Ag}}(s \mid o))$, where o is the current observation of the environment state, and obtaining the least ambiguous (total) belief state equal to $\{s^*\}$. A more cautious selection would be to select the first k most probable states obtaining the belief state $\{s_1^*, \ldots, s_k^*\}$, where s_i^* denotes the i-th best prediction of $f(s \mid o)$. When, instead, f_{Ag} is not confident in B_{Ag}, then making a measure in B_{Ag} does not provide reliable information about the environment. In this situation, the agent might find a plan π that leads to a new belief state $\gamma(\pi, B_{\text{Ag}})$ where f_{Ag} is confident. However, notice that π should not modify the observable state variable that the agent wants to perceive in B_{Ag}, but only change the state control variables that allow for a better perception.

Example 5. *Suppose that the agent is placed in the environment described in Example 1 (a), where the light is off and it wants to know which digit is on the wall. Its initial belief set is the entire* S_{Ag} *(it is completely ignorant, i.e., no beliefs). The current perception is the black image shown in Figure 1. Since* f_{light} *is confident for any belief state, then the agent can rely on the prediction of* f_{light}*. Therefore, the perception of* f_{light} *will lead to a belief set* $S_{\mathsf{light}=0} = \{s \in S \mid s_{\mathsf{light}=0}\}$*. Since in this belief state* f_{digit} *is not confident, the agent needs to find a plan that leads to a belief state where* f_{light} *is more reliable. Such a belief state is* B_{corner} *as defined at the end of Example 4. To reach this belief state it can execute (in sequence) two actions with effects* $\mathsf{light} \leftarrow 1$*, and* $\mathsf{occl} \leftarrow c$ *for some* $c = (0,0),(0,h),(w,0),(w,h)$*. This plan will not modify the value of the state variable* digit*. Therefore, after executing the plan the agent will see an observation similar to the two rightmost pictures of Figure 1, and by the perception function* f_{digit} *it can obtain that* $\mathsf{digit} = 3$ *with a high confidence.*

The aim of our work is addressing the following task.

Problem. *Given an agent* $Ag = (\mathcal{M}_{Ag}, B_{Ag}, f_{Ag}, ex_{Ag})$ *and a state variable* x*, find a set of viewpoints* $B_1, \ldots, B_k$ *for* x *with confidence greater than* t*, and compute a plan* π *that does not change the value of* x *and such that* $\gamma(\pi, B_{Ag}) = B_i$ *for some* $1 \leq i \leq k$.

4 Method

We describe our approach assuming that the perception function is in the form $f_{\text{Ag}} = (f_{x_1}, \ldots, f_{x_n})$. For the sake of presentation, we assume that $x_1, \ldots, x_{n-1}$ are control variables, and therefore the perception function f_{x_i} is perfect in any belief state, i.e., it always returns the ground-truth value, and only variable x_n is observable. However, the method can be applied for any partition of the state variables in control and observable variables. The agent has to find the belief states where x_n can be observed with high confidence by f_{x_n}. For instance, in Example 3 the agent is provided with the perception function for all state variables; we assume that the perception function of f_{occl}, and f_{light} are perfect, while the agent has to find the belief states where the confidence of f_{digit} is above a given threshold.

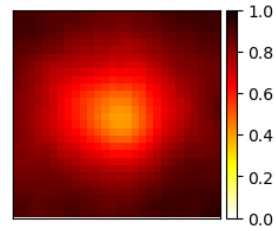

Figure 3: Heatmap of the confidence of the perception function f_{digit} in our running example.

4.1 Estimating the Perception Confidence

In order to estimate the confidence of f_{x_n} in an agent state $s \in S_{\text{Ag}}$, the agent can approximate Equation (1). To this purpose, the agent needs a set of observations $O_s \subseteq O_{\text{Env}}$ labeled with the value of x_n in s. The agent can construct the set of observations O_s by executing a plan that leads to s and collecting observations from the reached environment state. In Example 1, if the agent is in state $4(4,7)1$, it can construct the observation set $O_{3(4,7)1}$ by executing the action with effect $\mathsf{digit} \leftarrow 3$ and adding the observation taken from the resulting state to $O_{3(4,7)1}$.

We focus on the problem of finding a set of belief states $\mathcal{B} = \{B_1, \ldots, B_k\}$ where the state variable x_n is observable. For this purpose, we have to select a set $\mathcal{B} \subseteq 2^{S_{\text{Ag}}}$. One possibility could be to consider all the belief states obtained by assigning the control variables $\boldsymbol{x}_{-n}$ to any possible value $\boldsymbol{v}_{-n}$. In our running example, the agent considers the belief states where the values of the control variables light and occl are fixed to some values; an example of belief state is $S_{\mathsf{light}=1,\mathsf{occl}=(0,0)} = \{0(0,0)1, 1(0,0)1, \ldots, 9(0,0)1\}$.

To estimate the confidence $\text{conf}(f_{x_n}, B_i)$ in each belief state B_i, the agent computes :

$$\frac{1}{|B_i|} \sum_{s \in B_i} \frac{1}{|O_s|} \sum_{o \in O_s} f_{x_n}(s \mid o).$$

In Figure 3, we report the perception function confidence of the neural network predicting the state variable digit considered in our running example. The control state variables are light, occl. For instance, the heatmap in Figure 3 shows the confidence of the network evaluated on images collected in states where $\mathsf{light} = 1$ and the values of occl vary among the coordinates of all image pixels. Specifically, each pixel (i,j) of the heatmap reports the confidence of the network averaged over all images where the occluding circle is centered in position (i,j). As expected, the confidence is lower when the occluding circle is centered in the middle area of the image.

4.2 Find Viewpoints via Clustering

The straightforward methodology described before is not feasible when the number of possible combinations of values of the control variables is large. In this case, the number of states in S_{Ag} is huge, and hence it is not feasible to collect a set of observations for each state s. Furthermore, the set of belief states with the control variable fixed to some values is exponentially large with respect to the number of control variable values. To deal with the problem of collecting a set of observations for each state, we collect observations from a subset of states, which is a good representative sample for the domain of the control variables. Afterward, we cluster the collected observations into a set of belief states, one for each cluster, and we select the clusters where the perception function confidence is sufficiently high. To apply clustering, we assume that the values of the control variables are numerical (e.g., light on/off is encoded by light equal to $1/0$), and we denote the distance measure between a pair of values $\boldsymbol{v}$, $\boldsymbol{v}'$ of the control variables as $\mathit{dist}(\boldsymbol{v}, \boldsymbol{v}')$. Such a procedure is detailed in Algorithm 1.

In the first part of the algorithm (Lines 2–11), for every possible value v_n of the observable variable x_n, the agent collects the confidence y of observing v_n from m states where $x_n = v_n$. Each y is associated with the value $\boldsymbol{v}_{-n}$ of the control variables in the state where x_n has been observed. To this aim, the agent iteratively samples m assignments $\boldsymbol{v}_{-n}$ to the control variables $\boldsymbol{x}_{-n}$ (Line 4); it computes a plan for reaching the state $\boldsymbol{x} = (\boldsymbol{v}_{-n}, v_n)$ from its current belief state B_{Ag} (Line 5). Then, it executes the plan, observes the reached environment state (Line 6), and, on Line 7, computes the confidence $y = f_{x_n}(v_n \mid o)$ of observing v_n. Finally, on Line 9, the agent updates its belief state B_{Ag}. Notice that f_{x_n} is learned from the samples that are collected online. Indeed, for every f_{x_n}, there is an action in the planning domain corresponding to the training of a neural network (Line 6).

In the second part of the algorithm (Lines 13–17), for every element $\boldsymbol{v}_{-n}$ appearing in F, the algorithm computes the average confidence y^* of the prediction done from the neighbor states $N(\boldsymbol{v}_{-n})$, i.e., from states where $\mathit{dist}(\boldsymbol{v}_{-n}, \boldsymbol{v}'_{-n}) \leq r$ for a given distance threshold $r \in \mathbb{R}^+$.

In the last part of the algorithm (Lines 18–25), the agent builds a set of belief states $\mathcal{B}$ starting from F^*. The elements of F^* are clustered in $C_1, \ldots, C_k$ sets (in our experiments we applied the K-mean algorithm). The agent builds the set $\mathcal{C}$ of clusters by selecting the clusters with an average confidence higher than the given threshold t. For each cluster in $\mathcal{C}$, the agent computes the cluster centroid $\boldsymbol{c}_i$ by averaging the values $\boldsymbol{v}_{-n}$ of the control variables (Line 22). Afterward, the agent builds the belief state B_i associated with the cluster C_i by selecting the states in S_{Ag} whose control variables distance from $\boldsymbol{c}_i$ is less than threshold r (Line 23). The set of belief states $\mathcal{B}$ is finally extended with the computed belief state B_i.

4.3 Planning to Reach a Belief State

For perceiving the observable variable x_n, the agent needs to construct and execute a plan π from its current belief state to reach a state of a belief state in the set $\mathcal{B}$ returned by Algorithm 1. To this aim, we adopt symbolic (PDDL) planning.

Algorithm 1 FIND BELIEF STATES

Require: $\text{Ag} = (\mathcal{M}_{\text{Ag}}, B_{\text{Ag}}, \text{ex}_{\text{Ag}}, f_{\text{Ag}})$
Require: $t \in [0, 1]$: confidence threshold
Require: $r \in \mathbb{R}^+$: belief state radius
Require: m: number of sampled values for x_n
1: $F \leftarrow \emptyset$
2: **for** $v_n \in Dom(x_n)$ **do**
3: **for** $i \in \{1, \ldots, m\}$ **do**
4: $\boldsymbol{v}_{-n} \leftarrow$ uniformly sample values for $\boldsymbol{x}_{-n}$
5: $\pi \leftarrow$ PLAN$(\mathcal{M}_{\text{Ag}}, B_{\text{Ag}}, S_{\boldsymbol{x}=(\boldsymbol{v}_{-n}, v_n)})$
6: $o \leftarrow \text{ex}_{\text{Ag}}(\pi)$
7: $y \leftarrow f_{x_n}(v_n \mid o)$
8: $F \leftarrow$ APPEND$(F, \langle \boldsymbol{v}_{-n}, y \rangle)$
9: $B_{\text{Ag}} \leftarrow \gamma_{\text{Ag}}(\pi, B_{\text{Ag}})$
10: **end for**
11: **end for**
12: $F^* \leftarrow \emptyset$
13: **for** $(\boldsymbol{v}_{-n}, y) \in F$ **do**
14: $N(\boldsymbol{v}_{-n}) \leftarrow \{(\boldsymbol{v}'_{-n}, y') \in F \mid \mathit{dist}(\boldsymbol{v}'_{-n}, \boldsymbol{v}_{-n}) \leq r\}$
15: $y^* \leftarrow \frac{1}{|N(\boldsymbol{v}_{-n})|} \sum_{(\boldsymbol{v}'_{-n}, y') \in N(\boldsymbol{v}_{-n})} y'$
16: $F^* \leftarrow$ APPEND$(F^*, \langle \boldsymbol{v}_{-n}, y^* \rangle)$
17: **end for**
18: $C_1, \ldots C_k \leftarrow$ CLUSTERING(F^*)
19: $\mathcal{C} \leftarrow \{C_i \mid \frac{1}{|C_i|} \sum_{\langle \boldsymbol{v}_{-n}, y^* \rangle \in C_i} y^* > t\}$
20: $\mathcal{B} \leftarrow \emptyset$
21: **for** $C_i \in \mathcal{C}$ **do**
22: $\boldsymbol{c}_i \leftarrow \frac{1}{|C_i|} \sum_{\langle \boldsymbol{v}_{-n}, y^* \rangle \in C_i} \boldsymbol{v}_{-n}$
23: $B_i \leftarrow \{s \in S_{\text{Ag}} \mid \mathit{dist}(s_{\boldsymbol{x}_{-n}}, \boldsymbol{c}_i) \leq r\}$
24: $\mathcal{B} \leftarrow \mathcal{B} \cup \{B_i\}$
25: **end for**
26: **return** $\mathcal{B}$

In particular, the agent's automaton $\mathcal{M}_{\text{Ag}}$ is specified as a (PDDL) planning domain. The specification of the planning domain contains a set of operators for modifying the values of the state variables $x_1, \ldots, x_n$. For instance, in our running example, the agent can change the value of the control variable light by executing the action with effect $\text{light} \leftarrow 1$.

For variable x_n, we extend the planning domain with a predicate known_xn, which indicates that the agent knows the value v_n of x_n. For each learned belief state $B_i \in \mathcal{B}$, and values $\boldsymbol{v}_{-n} \in s$ for all $s \in B_i$, we add to the planning domain an action observe_xn$(\boldsymbol{v}_{-n})$ with preconditions $x_i = v_i$ for all $1 \leq i < n$, and effect known_xn. Finally, we define a planning problem as follows. In the initial state, $x_i = v_i$ with $1 \leq i < n$ and $v_i \in s$ for all $s \in B_{\text{Ag}}$; the goal of the planning problem is known_xn.

Example 6. *Suppose* $x_n = \text{digit}$, $B_{Ag} = S_{\text{light}=0,\text{occl}=(\frac{w}{2},\frac{h}{2})}$, *and* $\mathcal{B} = \{S_{\text{light}=1,\text{occl}=(0,0)}, S_{\text{light}=1,\text{occl}=(w,h)}\}$. *Then, the planning domain is extended with two actions* observe_digit$(1, (0, 0))$ *and* observe_digit$(1, (w, h))$. *Their preconditions are respectively* $\{\text{light} = 1, \text{occl} = (0, 0)\}$, *and* $\{\text{light} = 1, \text{occl} = (w, h)\}$*; their (positive) effect is* known_digit(). *In the initial state of the planning problem* $\text{light} = 0, \text{occl} = (\frac{w}{2}, \frac{h}{2})$*; the problem goal is* known_digit().

5 Experimental Analysis

We investigate the effectiveness of our approach for determining the belief states where an agent can perceive observ-

Dataset	#Train	#Test	#Object types	#Images per type
Cifar10	50000	10000	10	6000
Cifar100	50000	10000	100	600
EuroSAT	21600	5400	10	[2000, 3000]
FER	28709	3589	7	[400, 7000]
MNIST	60000	10000	10	7000
OxfordPet	3731	3669	37	200

Table 1: Datasets of object images used for object classification. For each dataset, we report the number of images in the training and test set (2nd, 3rd columns), the number of object types (4th column), and the number of images per object type (5th column).

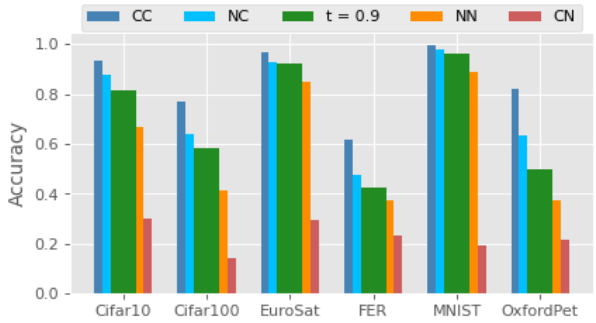

Figure 4: Accuracy of the object type predictions for each dataset.

able state variables. In our experiments, we consider variables representing object types and object properties. Moreover, we evaluate the agent's capabilities of planning and acting to reach a belief state form which it can perceive the values of the observable variables. We experiment our approach on several datasets of images for object classification, and perform a real-world demonstration using a robot with noisy sensors and noisy actions. For evaluating our approach, we adopted standard machine learning metrics, i.e., accuracy, precision, and recall of the perception function w.r.t. the ground truth variable values.

5.1 Learning to Perceive Object Types

For learning to perceive and classify objects, we considered 6 datasets with both synthetic and real-world RGB images of objects that are labeled with their types. The considered datasets are reported in Table 1. For each dataset, we modified perceptual aspects of the images by changing their brightness and blur, and by adding an occluding circle. These perceptual aspects correspond to the control variables of the agent state, and can be controlled through agent's actions. For example, an agent equipped with a camera can change the brightness of its camera image by turning on/off a light; similarly, the blur can be changed by calibrating the camera; and the occlusion of an object can be changed by moving the agent to a different point of view. All images have been modified by: *(i)* decreasing the brightness by 90% with a 0.5 probability; *(ii)* blurring the image by 50% with a 0.5 probability; and *(iii)* adding an occluding circle centered in a position uniformly sampled from the image size and with a diameter equal to 70% of the image size. We refer to the original and modified datasets as *noisy* and *clean* datasets, respectively.

As a perception function for classifying object types, we adopted a neural network architecture composed by a ResNet [He *et al.*, 2016] followed by a linear layer with a number of output perceptrons equal to the number of object types, and the SoftMax activation function. For symbolic (PDDL) planning, we used planner FastDownward [Helmert, 2006].

We consider each image in the *noisy* training set as an observation. For each of these images, the agent computes the values of the control variables $x_1, \ldots, x_{n-1}$ corresponding to the image brightness, blur, and occluding circle position. Then, the agent evaluates its perception function $f_{x_n}(x_n|o)$. In our experiments, x_n is the type of the object in the image. Finally, the agent computes the set of belief states $\mathcal{B}$ where it can observe the object types by clustering the collected observations as described in Algorithm 1. The agent does the same for the *noisy* test set, but in addition it checks if its current belief state is in $\mathcal{B}$; if this is not the case, it plans to reach a state in $\mathcal{B}$; finally, the agent predicts the object type.

We compare our approach with the following baselines:

- Perception function trained on the *Noisy* training set and evaluated on the *Noisy* test set (**NN**): the agent assumes that all the state variables can be always observed, and hence it does not plan to reach a state where the confidence of its perception function is sufficiently high. This baseline provides a lower bound of the performance achievable with our approach.
- Perception function trained on the *Clean* training set and evaluated on the *Noisy* test set (**CN**): as in **NN**, the agent assumes all the state variables can be observed. However, w.r.t. **NN**, the agent is provided with a perception function trained in fully observable environments.
- Perception function trained on the *Noisy* training set and evaluated on the *Clean* test set (**NC**): the agent can reach states where the object type is perfectly observable (i.e., the image has its original brightness, there is no blur, and no occluding circle). This version provides an upper bound of the performance achievable with our approach.
- Perception function trained on the *Clean* training set and evaluated on the *Clean* test set (**CC**): as in **NC**, the agent can reach states where the object type is perfectly observable. This version provides a measure of the complexity of the classification task.

The average accuracy of the object classifier predictions achieved by our approach (with a confidence threshold $t = 0.9$) w.r.t. the baselines is shown in Figure 4. Our approach achieves good accuracy in all domains when compared to the **NC** version, i.e., the agent computes and executes plans that effectively lead to belief states where the perception function for the object types is more reliable. The overall performance of our approach decreases in domain *FER* because the classification task is more complex for this domain, as **CC** provides the worst performance for *FER*. Our approach improves the performance of the **NN** baseline significantly: it is at least 10% better in all domains but *FER*. Such improvement empirically shows the importance of planning and acting to reach states where the agent can better observe a state variable. Finally, the **CN** baseline provides the worst accuracy, which is significantly lower than the **NN** accuracy. This is because the

Object type	Property	Precision	Recall
cup	filled	1	0.66
mug	filled	1	0.94
laptop	on	0.61	0.98
big bowl	full	1	0.18
small bowl	full	1	0.31
chair	free	1	0.28
Average	-	0.94	0.56

Table 2: Precision and recall of the viewpoints w.r.t. the observability of a number of properties of different objects.

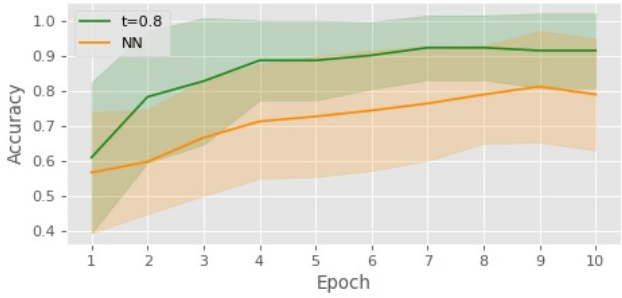

Figure 5: Accuracy of the predictions of the object properties averaged over all properties and objects reported in Table 2. The width of the colored areas measures the standard deviation of the accuracy.

perception function for **CN** is trained in fully observable environments and tested in partially observable environments.

5.2 Real World Experiments

To experimentally show the applicability of our approach in a real-world environment, we performed an experiment with a Softbank Robotic's Pepper humanoid robot with noisy sensors (e.g., RGB and depth camera) and actuators. Pepper is placed in a living room where there is a table with some objects on top of it (e.g., a mug, a laptop). The task is perceiving an object property that is observable. For example, the property filled for objects of type mug can be perceived only when looking at the mug from the top. Firstly, Pepper trains a neural network for recognizing the object property, by collecting online a number of observations (set to 200 in our experiments) according to the implementation proposed by Lamanna *et al.* (2023). Pepper labels the observed images by means of human supervision. For instance, Pepper asks a human to fill the mug, takes a number of pictures of the resulting situation, and finally labels these images with the action effect specified in a PDDL domain. The control variables associated with each observation are the distance, yaw angle, and pitch angle between the Pepper camera and the object. The control variable values are computed by the Pepper's noisy depth image, and noisy odometry position. Pepper evaluates the confidence of the neural network for the collected observations, and clusters the observations according to the associated values of the control variables and the neural network confidence. For clustering, we adopted the K-means algorithm with $K = 8$. The viewpoints where the property is observable are obtained by selecting the clusters with an average confidence higher than a threshold $t = 0.8$.

In Table 2, we evaluate the determined viewpoints by computing their precision and recall w.r.t. the observability of a number of properties of different objects. To do this, we manually annotated the observations collected by Pepper as observable and not observable. Therefore, true positives consist of observations in the viewpoints determined by our approach where the property can be observed; false positives are observations in the determined viewpoints from which the property cannot be observed. Similarly, true negatives are observations in states that are not determined viewpoints from which the property is labeled as non observable; whereas false negatives are observations in states that are not determined viewpoints, but from which the property is observable. The results in Table 2 provide empirical evidence that our approach effectively finds the viewpoints where the properties are observable. For all object types but laptop, the found viewpoints contain observations where the property is always observable (i.e., the precision equals 1). However, for object types big/small bowl and chair, the recall is low, i.e., the agent does not find all the viewpoints where the properties are observable.

We also evaluate the capability of Pepper to plan and act for reaching a viewpoint and observing the object properties. For this purpose, Pepper is placed in a random position, and asked to predict the object properties. Firstly, Pepper observes the object and checks if its current state belongs to the set of previously learned viewpoints. If this is not the case, Pepper plans and acts to reach a viewpoint. Finally, Pepper predicts the truth value of the property. The above procedure is repeated 20 times for each object property. In this experiment, we compare our approach with the **NN** baseline. Essentially, such a baseline assumes that the property can be always observed and hence does not plan to reach a viewpoint before predicting the truth value of the property. The accuracy of the predictions achieved by our approach w.r.t. the **NN** baseline is shown in Figure 5. In particular, we report the performance of property predictors trained for different epochs. The results show that planning and acting for reaching the learned viewpoints enables Pepper to improve its capability to correctly predict the truth value of the object properties.

6 Conclusions

We proposed a method for enabling an agent in a partially unknown environment to learn the situations where a state variable is observable, and reach such situations by planning and acting. We formalized the problem of learning the situations where the perception model of the agent is more confident. Afterward, we developed an algorithm for finding the agent states where the perception model is confident enough, and plan to reach such states by means of symbolic planning.

We experimentally evaluated the effectiveness of our approach for recognizing object types in a number of synthetic datasets, and object properties in a real-world environment involving noisy perceptions and noisy actions on a real robot.

In this work, we assume that the agent can always change the truth value of the observable variables, eventually asking for the help of a human. In future work, we will investigate how the agent can change such value without knowing it.

Acknowledgements

We acknowledge the support of the PNRR project FAIR - Future AI Research (PE00000013), under the NRRP MUR program funded by the NextGenerationEU. This work has also been partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, and AIPlan4EU funded by the EU Horizon 2020 research and innovation program under GA n. 101016442.

References

[Asai and Fukunaga, 2018] Masataro Asai and Alex Fukunaga. Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary. In *Proceedings of the 32nd AAAI conference on artificial intelligence*, volume 32, 2018.

[Asai, 2019] Masataro Asai. Unsupervised grounding of plannable first-order logic representation from images. In *Proceedings of the 29th International Conference on Automated Planning and Scheduling*, volume 29, pages 583–591, 2019.

[Ashari and Ghasemzadeh, 2019] Zhila Esna Ashari and Hassan Ghasemzadeh. Mindful active learning. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 2265–2271, 2019.

[Bajcsy *et al.*, 2018] Ruzena Bajcsy, Yiannis Aloimonos, and John K Tsotsos. Revisiting active perception. *Autonomous Robots*, 42(2):177–196, 2018.

[Belle *et al.*, 2023] Vaishak Belle, Thomas Bolander, Andreas Herzig, and Bernhard Nebel. Epistemic planning: Perspectives on the special issue. *Artificial Intelligence*, 316:103842, 2023.

[Bertoli *et al.*, 2006] Piergiorgio Bertoli, Alessandro Cimatti, Marco Roveri, and Paolo Traverso. Strong planning under partial observability. *Artificial Intelligence*, 170(4–5):337–384, 2006.

[Bohg *et al.*, 2017] Jeannette Bohg, Karol Hausman, Bharath Sankaran, Oliver Brock, Danica Kragic, Stefan Schaal, and Gaurav Sukhatme. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33:1273–1291, 2017.

[Bonet and Geffner, 2011] Blai Bonet and Hector Geffner. Planning under partial observability by classical replanning: Theory and experiments. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 1936–1941, 2011.

[Bonet and Geffner, 2014] Blai Bonet and Hector Geffner. Belief tracking for planning with sensing: Width, complexity and approximations. *Journal of Artificial Intelligence Research*, 50:923–970, 2014.

[Cakmak *et al.*, 2010] Maya Cakmak, Crystal Chao, and Andrea L Thomaz. Designing interactions for robot active learners. *IEEE Transactions on Autonomous Mental Development*, 2(2):108–118, 2010.

[Chao *et al.*, 2010] Crystal Chao, Maya Cakmak, and Andrea L Thomaz. Transparent active learning for robots. In *2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 317–324. IEEE, 2010.

[Dengler *et al.*, 2021] Nils Dengler, Tobias Zaenker, Francesco Verdoja, and Maren Bennewitz. Online object-oriented semantic mapping and map updating. In *Proceedings of the 10th European Conference on Mobile Robots*, pages 1–7. IEEE, 2021.

[Erol *et al.*, 1994] Kutluhan Erol, James A Hendler, and Dana S Nau. Umcp: A sound and complete procedure for hierarchical task-network planning. In *Proceedings of the 2nd International Conference on Artificial Intelligence Planning Systems*, pages 249–254, 1994.

[Grotz, 2021] Markus Grotz. *Active Vision for Scene Understanding*. KIT Scientific Publishing, 2021.

[Hayes and Scassellati, 2014] Bradley Hayes and Brian Scassellati. Discovering task constraints through observation and active learning. In *Proceedings of the 27th International Conference on Intelligent Robots and Systems*, pages 4442–4449. IEEE, 2014.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Helmert, 2006] Malte Helmert. The fast downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.

[Huang *et al.*, 2010] Sheng-Jun Huang, Rong Jin, and Zhi-Hua Zhou. Active learning by querying informative and representative examples. *Proceedings of the 23rd Conference on Neural Information Processing Systems*, 23, 2010.

[Janner *et al.*, 2019] Michael Janner, Sergey Levine, William T. Freeman, Joshua B. Tenenbaum, Chelsea Finn, and Jiajun Wu. Reasoning about physical interactions with object-centric models. In *Proceedings of the 7th International Conference on Learning Representations*, 2019.

[Konidaris *et al.*, 2018] George Konidaris, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. From skills to symbols: Learning symbolic representations for abstract high-level planning. *Journal of Artificial Intelligence Research*, 61:215–289, 2018.

[Kulick *et al.*, 2013] Johannes Kulick, Tobias Lang, Marc Toussaint, and Manuel Lopes. Active learning for teaching a robot grounded relational symbols. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 2013.

[Lamanna *et al.*, 2023] Leonardo Lamanna, Luciano Serafini, Mohamadreza Faridghasemnia, Alessandro Saffiotti, Alessandro Saetti, Alfonso Gerevini, and Paolo Traverso. Planning for learning object properties. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[Migimatsu and Bohg, 2022] Toki Migimatsu and Jeannette Bohg. Grounding predicates through actions. In *Proceedings of the International Conference on Robotics and Automation*, pages 3498—-3504. IEEE Press, 2022.

[Occhipinti *et al.*, 2022] Andrés Occhipinti, Blai Bonet, and Hector Geffner. Learning first-order symbolic planning representations that are grounded. In *Proceedings of ICAPS Workshop on Bridging the Gap Between AI Planning and Reinforcement Learning (PRL)*, 2022.

[Petrick and Bacchus, 2002] Ronald P. A. Petrick and Fahiem Bacchus. A knowledge-based approach to planning with incomplete information and sensing. In *Proceedings of the 6th International Conference on Artificial Intelligence Planning Systems*, pages 212–222. AAAI, 2002.

[Pinto *et al.*, 2016] Lerrel Pinto, Dhiraj Gandhi, Yuanfeng Han, Yong-Lae Park, and Abhinav Gupta. The curious robot: Learning visual representations via physical interactions. In *Proceedings of the 14th European Conference on Computer Vision*, pages 3–18, 2016.

[Ribes *et al.*, 2015] Arturo Ribes, Jesus Cerquides, Yiannis Demiris, and Ramón López de Mántaras. Active learning of object and body models with time constraints on a humanoid robot. *IEEE Transactions on Cognitive and Developmental Systems*, 8(1):26–41, 2015.

[Tarabanis *et al.*, 1995] Konstantinos A Tarabanis, Peter K Allen, and Roger Y Tsai. A survey of sensor planning in computer vision. *IEEE transactions on Robotics and Automation*, 11(1):86–104, 1995.

[Taylor *et al.*, 2021] Annalisa T Taylor, Thomas A Berrueta, and Todd D Murphey. Active learning in robotics: A review of control principles. *Mechatronics*, 77:102576, 2021.

[Ugur and Piater, 2015] Emre Ugur and Justus Piater. Bottom-up learning of object categories, action effects and logical rules: From continuous manipulative exploration to symbolic planning. In *Proceedings of the International Conference on Robotics and Automation*, pages 2627–2633, 2015.

[Zeng *et al.*, 2020] Rui Zeng, Yuhui Wen, Wang Zhao, and Yong-Jin Liu. View planning in robot active vision: A survey of systems, algorithms, and applications. *Proceedings of the 6th International Conference on Computational Visual Media*, 6:225–245, 2020.