First-Person Activity Forecasting from Video with Online Inverse Reinforcement Learning

Nicholas Rhinehart, *Student Member, IEEE,* Kris M. Kitani, *Member, IEEE,*

Abstract—We address the problem of incrementally modeling and forecasting long-term goals of a first-person camera wearer: what the user will do, where they will go, and what goal they seek. In contrast to prior work in trajectory forecasting, our algorithm, DARKO, goes further to reason about semantic states (will I pick up an object?), and future goal states that are far in terms of both space and time. DARKO learns and forecasts from first-person visual observations of the user's daily behaviors via an Online Inverse Reinforcement Learning (IRL) approach. Classical IRL discovers only the rewards in a batch setting, whereas DARKO discovers the transitions, rewards, and goals of a user from streaming data. Among other results, we show DARKO forecasts goals better than competing methods in both noisy and ideal settings, and our approach is theoretically and empirically no-regret.

Index Terms—First-Person Vision, Activity Forecasting, Inverse Reinforcement Learning, Online Learning.

1 INTRODUCTION

OUR long-term aim is to develop an AI system that can learn about a person's intent and goals by continuously observing their behavior. Towards this goal, we propose an online Inverse Reinforcement Learning (IRL) technique to learn a decision-theoretic human activity model from video captured by a wearable camera.

The use of a wearable camera is critical to our task, as human activities must be observed up close and across large environments. Imagine a person's daily activities—perhaps they are at home today, moving about, completing tasks. Perhaps they are a scientist that conducts a long series of experiments across various stations in a laboratory, or they work in an office building where they walk about their floor, get coffee, *etc*. As people tend to be very mobile, a wearable camera is ideal for observing a person's behavior.

Since our task is to continuously learn human behavior models from observed behavior captured with a wearable camera, our task is best described as an online IRL problem. The problem is an *inverse* Reinforcment Learning problem because the underlying reward or cost function of the person is unknown. We must infer it along with the policy from the demonstrated behaviors. Our task is also an *online learning problem*, because our algorithm must continuously learn as a part of a life-long process. From this perspective, an online learning approach is required to learn effectively over time.

We present an algorithm that *incrementally learns spatial and semantic intentions* (where you will go and what you will do) of a first-person camera wearer. By tracking the goals a person achieves, the algorithm builds a set of possible futures. At any time, the user's future is predicted among this set of goals. We term our algorithm "Discovering Agent Rewards for K-futures Online" (DARKO), as it learns to associate rewards with semantic states and actions from demonstrations to predict among K possible goals.

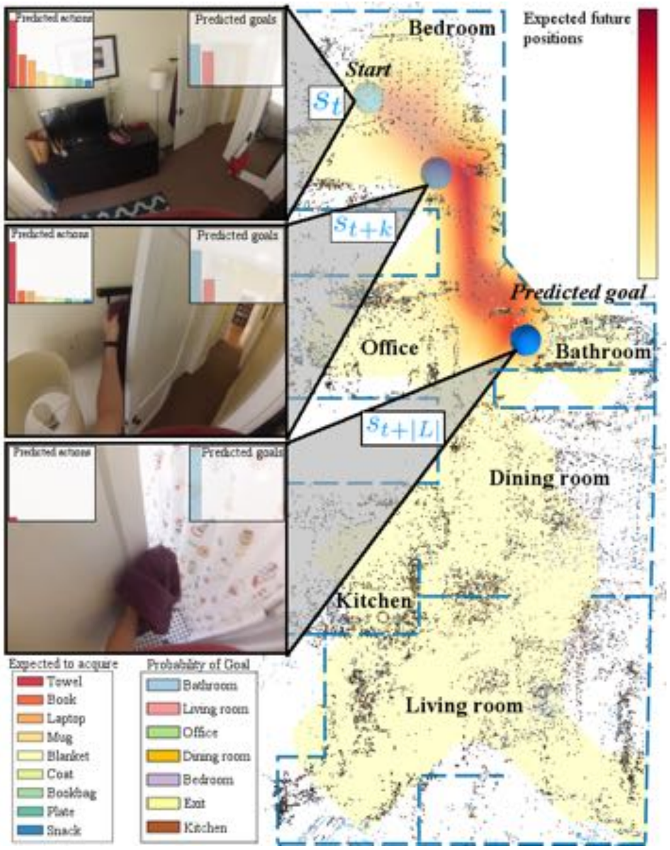
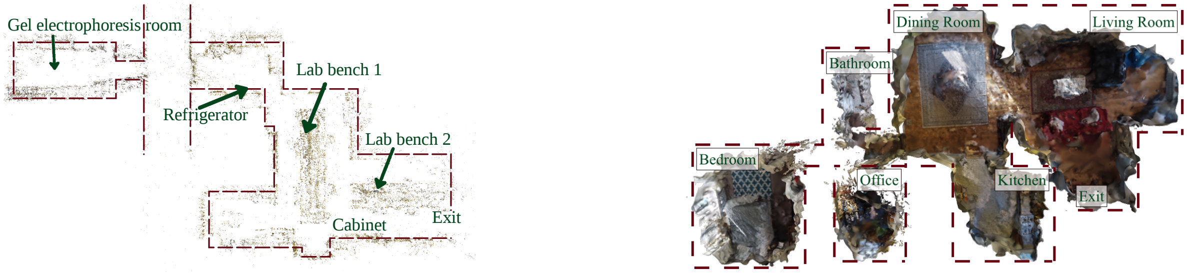


Fig. 1. **Forecasting future behavior from first-person video.** Overhead map shows likely future goal states. s_i is user *state* at time i. Histogram insets display predictions of user's long-term semantic goal (inner right) and acquired objects (inner left).

To the best of our knowledge, we present the first application of ideas from online learning theory and inverse reinforcement learning to the task of continuously learning

- *N. Rhinehart and K. Kitani are with the Robotics Institute within the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 15213. E-mail: {nrhineha, kkitani}@cs.cmu.edu*

Manuscript received -; revised -.



(a) Lab environment

(b) Home 1 environment

Fig. 2. Sparse SLAM points (2a) and offline dense reconstruction (2b) using [8] for two of our dataset environments.

human behavior models with a wearable camera. Our proposed algorithm is distinct from traditional IRL problems as we jointly discover transitions, goals, and the reward function of the underlying Markov Decision Process model. Our proposed human behavior model also goes beyond first-person trajectory forecasting by predicting future human activities that can happen outside the immediate field of view and far into the future.

This paper is an extension of our previous work [29]. Our extensions include (i) a feature ablation experiment and relevant discussion; (ii) a detection noise experiment and relevant discussion; (iii) formalization of the regret bound with proof; (iv) derivation of other inference tasks made possible by our approach; (v) additional visualizations and descriptions throughout the paper.

2 Related Work

We extend a portfolio of visual sensing algorithms (SLAM, stop detection, scene classification, action and object recognition) in concert width a decision-theoretic approach to model and forecast behavior online. Several topics of related work cover components of our approach. We will summarize each in the following and relate them to our approach.

2.1 First-person vision (FPV)

Wearable cameras have been used for various human behavior understanding tasks [7], [14], [15], [17], [23], [32], [37], [50] because they give direct access to detailed visual information about a person's actions. Leveraging this feature of FPV, recent work has shown that it is possible to predict where people will look during actions [17] and how people will use the environment [28]. Previous first-person vision approaches are narrower in their scope of modeling relative to our approach: they are batch models (learned once), and generally do not attempt to model predictions with respect to information outside of the camera's frame of view.

2.2 Decision-Theoretic Modeling

Given agent demonstrations, the task of *inverse reinforcement learning* (IRL) is to recover a *reward function* of an underlying Markov Decision Process (MDP) [2]. IRL has been used to model taxi driver behavior [53] and forecast pedestrian trajectories [11], [54]. In contrast to these approaches, we go beyond physical trajectory forecasting by reasoning over future object interactions and both uncovering and forecasting future goals in terms of scene types.

2.3 Online Learning Theory

The theory of learning to making optimal predictions from streaming data is well-studied [33], but its fruits are seldom applied to computer vision (*e.g.* [5], [30]), compared to the more prevalent application of supervised learning. However, we believe online learning theory and practice will gain traction to confront the challenges of ever-increasing visual data. In the context of IRL, online learning theory was used in [26], an imitation learning framework similar to IRL in its recovery of costs, to analyze performance of a mobile robot path planner.

2.4 Forecasting

Our task fits in the broad category of *forecasting* with visual information. There are three primary research thrusts in the forecasting category: *trajectory pixel*, and *behavior* and forecasting. The former attempts to predict an agent's future behavior in 2D or 3D coordinates. The latter attempts to predict the behavior of agents in terms of categories of low-level activities. Image forecasting attempts to directly predict the pixels of future images from a temporal stream. However, the majority of the work along these veins does not model or capitalize upon two key aspects of the problem: *uncertainty* in the future behaviors (which requires predicting either a distribution or multiple samples), and the *goal-driven nature* of agents. Agents generally take low-level motions and actions in order to achieve goals. Our work explicitly forecasts and models goals with uncertainty.

2.4.1 Trajectory Forecasting

Physical trajectory forecasting has received much attention from the vision community. The task is to predict the future spatial coordinates of an agent. Multiple human trajectory forecasting from a surveillance camera was investigated by [19]. Other trajectory forecasting approaches use demonstrations observed from a bird's-eye view; [49] infers latent goal locations and [3] employ LSTMs to jointly reason about trajectories of multiple humans. In [35], the model forecasted short-term future trajectories of a first-person camera wearer by retrieving the nearest neighbors from a dataset of first-person trajectories under an obstacle-avoidance cost function, with each trajectory representing predictions of where the user will move in view of the frame; in [36], a similar model with learned cost function is extended to multiple users.

A drawback of predicting future spatial coordinates of an agent is an interpetability gap. If a person is tasked with predicting and communicating some agent's future, they will not produce a list of high-fidelity spatial coordinates — instead they will frame their prediction in the *semantics* of activity. Methods that forecast interpretable futures grounded in categories of behavior are known as activity forecasting.

2.4.2 Activity Forecasting

Activity forecasting methods typically treat the problem of predicting future behaviors as a classification task. In [9], [31], the tasks are to recognize an unfinished event or activity. In [9], the model predicts the onset for a single facial action primitive, *e.g.* the completion of a smile, which may take less than a second. Similarly, [31] predicts the completion of short human to human interactions. In [13], a hierarchical structured SVM is employed to forecast actions about a second in the future, and [41] demonstrates a semi-supervised approach for forecasting human actions a second into the future. Other works predict actions several seconds into the future [6], [12], [16]. In contrast, we focus on high-level transitions over a sequence of future actions that may occur outside the frame of view, and take a longer time to complete (in our dataset, the mean time to completion is 21.4 seconds).

2.4.3 Pixel Forecasting

Pixel forecasting methods generate full image or video representations of predictions, which can help a human interpret what the model has learned [4], [10], [21], [40], [42], [43], [44], [45], [46]. In [42], [43], [44], future images are generated by unsupervised models. In [45], surveillance image predictions of vehicles are formed by smoothing a patch across the image. [46] and [40] first represent the future in terms of a visible human pose, and then predict image frames from that pose. In [4], image boundaries are predicted. Drawbacks to these methods include the difficulty in measuring the model's quality, as well as the inability to directly represent the joint future of the agent and the environment. In contrast to pixel forecasting approaches, our approach is to predict the high-level semantic future of agents in the image.

3 Online IRL with DARKO

Our goal is to forecast the future behaviors of a person from a continuous stream of video captured by a wearable camera. Given a continuous stream of FPV video, our approach extracts a sequence of state variables $\{s_1, s_2, \dots\}$ using a portfolio of visual sensing algorithms (*e.g.*., SLAM, stop detection, scene classification, action and object recognition). In an online fashion, we segment this state sequence into episodes (short trajectories) by discovering terminal goal states (*e.g.*., when a person stops). Using the most recent episode, we adaptively solve the inverse reinforcement learning problem using online updates. Solving the IRL problem in an online fashion means that we incrementally learn the underlying decision process model.

3.1 First-Person Behavior Model

A Markov Decision Process (MDP) is commonly used to model the sequential decision process of a rational agent. In our case, we use it to describe the activity of a person with a wearable camera. In a typical reinforcement learning problem, all elements of the MDP are assumed to be known and the task is to estimate an optimal policy $\pi(a|s)$, that maps a state s to an action a, by observing rewards. In our novel online inverse formulation, the transition function, reward function, policy, and goal states are unknown and must be inferred as new video data arrives. Formally, our MDP is defined as:

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, R_\theta).$$

3.2 States

$\mathcal{S}$ is the state space: the set of states an agent can visit. In our online formulation, $\mathcal{S}$ is initially empty, and must be expanded as new states are discovered. We define a state s as a vector that includes the location of the person (3D position), the last place the person stopped (a previous goal state), and information about any object that the person might be holding. Formally, a state $s \in \mathcal{S}$ is denoted as:

$$s = [x, y, z, o_1 \dots, o_{|\mathcal{O}|}, h_1, \dots h_{|\mathcal{K}|}].$$

The triplet $[x, y, z]$ is a discrete 3D position. To obtain the position, we use a monocular visual SLAM algorithm [20] to localize the agent in a continuously built map.

The vector $o_1 \dots, o_{|\mathcal{O}|}$ encodes any objects that the person is currently holding. We include this information in the state vector because the objects a user acquires are strongly correlated to the intended activity [7]. $o_j = 1$ if the user has object j in their possession and zero otherwise. $\mathcal{O}$ is a set of pre-defined objects available to the user. $\mathcal{K}$ is a set of pre-defined scene types available to the user, which can be larger than the true number of scene types. The vector $h_1, \dots h_K$ encodes the last scene type the person stopped. Example scene types are `kitchen` and `office`. $h_i = 1$ if the user last arrived at scene type i and is zero otherwise.

3.3 Goals

We also define a special type of state called a *goal state* $s \subset \mathcal{S}_g$, to denote states where the person has achieved a goal. One of our methods assumes that when a person stops for a certain period of time, their location in the environment is a goal. This method detect goal states by using a velocity-based stop detector. Whenever a goal state is encountered, the sequence of states since the last goal state to the current goal state is considered a completed episode ξ. The set of goals states $\mathcal{S}_g \subset \mathcal{S}$ expands with each detection. We explain later how $\mathcal{S}_g$ is used to perform goal forecasting.

3.4 Actions

$\mathcal{A}$ is the set of actions. $\mathcal{A}$ can be decomposed into two parts: $\mathcal{A} = \mathcal{A}_m \cup \mathcal{A}_c$. The act of moving from one location in the environment to another location is denoted as $a_m \in \mathcal{A}_m$. Like $\mathcal{S}$, $\mathcal{A}_m$ must be built incrementally. The set $\mathcal{A}_c$ is the set of possible `acquire` and `release` actions of each object: $\mathcal{A}_c = \{\texttt{acquire}, \texttt{release}\} \times \mathcal{O}$. The act of releasing or picking up an object is denoted as $a_c \in \mathcal{A}_c$. Each action a_c must be detected. We do so with an image-based first-person action classifier. More complex approaches could improve performance [18].

3.5 Transition Function

The transition function $T : (s, a) \mapsto s'$ represents how actions move a person from one state to the next state. T is constructed incrementally as new states are observed and new actions are performed. In our work, T is built by keeping a table of observed (s, a, s') triplets, which describes the connectivity graph over the state space. More advanced methods could also be used to infer more complex transition dynamics [38], [39], [47].

3.6 Reward Function

$R(s, a; \theta)$ is an instantaneous reward function of action a at state s. The standard and simplest parametric model of R is the inner product between a vector of features $f(s, a)$ and a vector of weights θ. We adopt this standard model, however, different representations could be employed [48]. The reward function is essential in value-based reinforcement learning methods (in contrast to policy search methods) as it is used to compute the policy $\pi(a|s)$. In the maximum entropy setting, the policy is given by $\pi(a|s) \propto e^{Q(s,a)-V(s)}$, where the value functions $V(s)$ and $Q(s, a)$ are computed from the reward function by solving the Bellman equations [53]. In our context, we learn the reward function online.

Intuitively, we would like the features f of the reward function to incorporate information such as the position in an environment or objects in possession, since it is reasonable to believe that the goal of many activities is to reach a certain room or to retrieve a certain object. To this end, we define the features of the reward to mirror the information already contained in the state s_t: the position, previous scene type, and objects held. To be concrete, the feature vector $f(s, a)$ is the concatenation of the 3-d position coordinates $[x, y, z]$, a K-dimensional indicator vector over previous goal state type and a $|\mathcal{O}|$-dimensional indicator vector over held objects. We also concatenate a $|\mathcal{A}_c|$-dimensional indicator vector over actions $a_c \in \mathcal{A}_c$.

3.7 The DARKO Algorithm

We now describe our proposed algorithm for incrementally learning all MDP parameters, most importantly the reward function, given a continuous stream of first-person video (see DARKO in Algorithm 3). The procedure begins by initializing s, reward parameters θ, empty state space $\mathcal{S}$, goal space $\mathcal{S}_g$, transition function T, and current episode ξ.

3.8 State Space Update

Image frames are obtained from a first-person camera (the NEWFRAME function), and SLAM is used to track the user's location (lines 4 and 5). An image-based action detection algorithm, ACTDET, detects hand-object interactions a_c and decides movements a_m as a function of current and previous position. While we provide an effective method for ACTDET, our focus is to integrate (rather than optimize) its outputs. Lines 7 and 8 show how the trajectory is updated and MDP parameters of state space and transition function are expanded. Line 9 represents a collection of generalized forecasting tasks (see Section 4.4), such as the computation of future goal posterior and trajectory forecasting. An example of our primary forecasting task, goal forecasting, is illustrated in Figure 5. Our method uses the current environment and policy models to forecast a distribution over the person's goals, described in more detail later. In the example, the distribution is initially uncertain about the person's goal, and then becomes confident the person will go to the goal in the kitchen, given the evidence that they are in the office and recently acquired a mug.

```
1:  procedure DARKO(SLAM, ACTDET, GOALDET)
2:      $s \leftarrow \mathbf{0}, \theta = \mathbf{0}, \mathcal{S} = \{\}, \mathcal{S}_g = \{\}$, T.INIT(), $\xi = []$
3:      while True do
4:          frame ← NEWFRAME()
5:          $[x, y, z]$ ← SLAM.TRACK(frame)
6:          $a$ ← ACTDET($[x, y, z]$, frame)
7:          $\xi \leftarrow \xi \oplus (s, a), \mathcal{S} \leftarrow \mathcal{S} \cup \{s\}$
8:          T.EXPAND($s$, $a$), $s \leftarrow T(s, a)$
9:          ▶ Goal forecasting, trajectory forecasting, . . .
10:         is_goal ← GOALDET($s$, frame, $\mathcal{S}_g$)
11:         if is_goal then
12:             $\mathcal{S}_g \leftarrow \mathcal{S}_g \cup \{s\}$
13:             $\pi, \theta$ ← ONLINEIRL($\theta$, $\mathcal{S}$, $T$, $\xi$, $\mathcal{S}_g$)
14:             $s \leftarrow T(s, a = \text{at\_goal}), \xi = []$
15:         end if
16:     end while
17: end procedure
```

Fig. 3. DARKO: Discovering Agent Rewards for K-futures Online

```
function ONLINEIRL($\theta$, $\mathcal{S}$, $T$, $\xi$, $\mathcal{S}_g$; $\lambda$, $B$)
    $\overline{f}_i = \sum_{(s,a)\in\xi} f(s, a)$
    ▶ Compute $R(s, a; \theta)\ \forall s \in \mathcal{S}, a \in \mathcal{A}$
    $\pi$ ← SOFTVALUEITERATION($R, \mathcal{S}, \mathcal{S}_g, T$)
    $\hat{f}_i \leftarrow E_\pi [f(s, a)]$
    $\theta \leftarrow \text{proj}_{\|\theta\|_2 \le B}(\theta - \lambda(\overline{f}_i - \hat{f}_i))$
    return $\pi, \theta$
end function
```

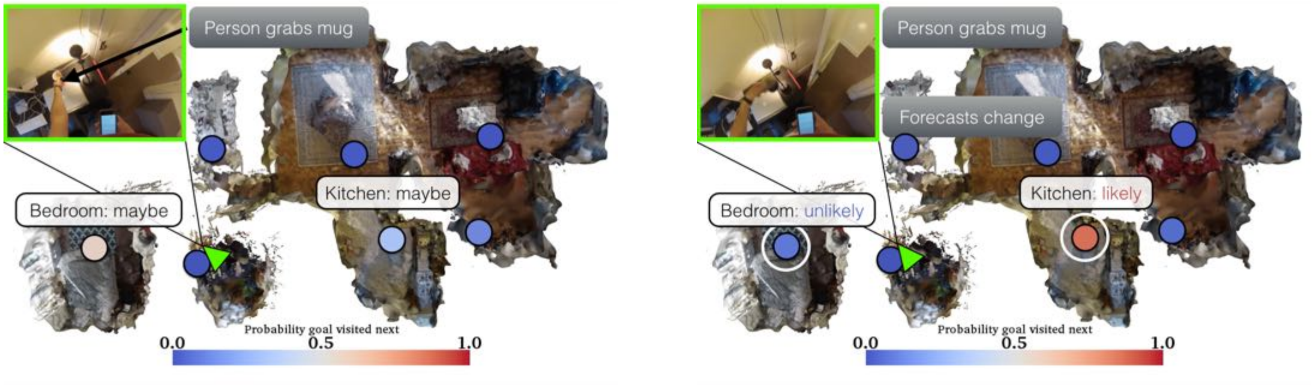
Fig. 4. Online Inverse Reinforcement Learning

3.9 Goal Detection

The GOALDET procedure denotes detecting new goals or recognizing previous goals. One of our GOALDET implementations is a stop-detection algorithm. This uses the camera velocity computed from SLAM (Line 10). If a goal state has been detected, that terminal state is added to the set of goal states $\mathcal{S}_g$. The detection of a terminal state also marks the end of an episode ξ. The previous goal state type is also updated for the next episode. Again, while we provide two effective method for GOALDET, our focus is to integrate (rather than optimize) its outputs.

3.10 Online Inverse Reinforcement Learning

With the termination of each episode ξ, the reward function R and corresponding policy π are updated via the



(a) Goal Posterior Before Mug Acquired

(b) Goal Posterior After Mug Acquired

Fig. 5. **Goal Posterior Change Visualization:** Goal posteriors for two frames are visualized in the Home 1 environment. The person's location is in green, images from the camera are inset at top left, and goal posteriors are colored according to the above colormaps. Before grabbing the mug (Figure 5a), DARKO forecasts roughly equivalent probability to bedroom and kitchen. After the user grabs the mug (Figure 5b), DARKO correctly predicts the user is likeliest to go to the kitchen.

reward parameters θ (Line 13). The parameter update uses a sequence of demonstrated behavior via the episode ξ, and the current parameters of the MDP. More specifically, ONLINEIRL (Algorithm 4) performs online gradient descent on the likelihood under the maximum entropy distribution by updating current parameters of the reward function. The gradient of the loss can be shown to be the difference between the feature counts of the expert, $\bar{f}$, and feature counts of the policy $\hat{f}$. Computing the gradient requires solving the soft value iteration algorithm of [52]. We include a projection step to ensure $\|\theta\|_2 \leq B$.

To the best of our knowledge, this is the first work to propose an online algorithm for maximum entropy IRL in the streaming data setting. Following the standard procedure for ensuring good performance of an online algorithm, we analyze our algorithm in terms of the regret bound. The regret $\mathcal{R}_t$ of any online algorithm is defined as:

$$\mathcal{R}_t(\{\theta_i\}_{i=0}^{t}) = \sum_{i=0}^{t} l_i(\theta_i) - \min_{\theta^*} \sum_{i=0}^{t} l(\theta^*).$$

The regret is the cumulative difference between the performance of the online model using current parameter θ versus the best hindsight model using the best parameters θ^*. The loss l_t is a function of the t'th demonstrated trajectory, and measures how well the model explains the trajectory. In our setup, the loss function is defined as $l_i(\theta_i; \xi_i) = -\frac{1}{|\xi_i|} \sum_{j=0}^{|\xi_i|} \log \pi_\theta(a_j|s_j)$.

Theorem 1 (ONLINEIRL is no-regret)**.** *Let $\hat{f}, \bar{f} \in [0,1]^d$, $||\theta||_2 \leq B$. The regret of Algorithm 4 satisfies $\mathcal{R}_t \leq 2B\sqrt{2td}$.*

Proof. By Equation 2.5 of [33], the regret of online gradient descent on our sequence of convex losses is bounded:

$$\mathcal{R}_t \leq \frac{1}{2\lambda}\|\theta\|_2^2 + \lambda \sum_{i=1}^{t} \|\nabla_{\theta_t}\|_2^2, \tag{1}$$

where λ is the learning rate. We will employ bounds on $\|\theta\|_2^2, \|\nabla_{\theta_t}\|_2^2$, and a minimizing choice of λ to prove the result. Writing the general gradient in terms of the expected features (and omitting the subscript t):

$$\|\nabla_\theta\|_2^2 = \|\bar{f} - \hat{f}\|_2^2 = \bar{f}^T\bar{f} + \hat{f}^T\hat{f} - 2\bar{f}^T\hat{f} \tag{2}$$

Using:

$$\begin{aligned}
0 \leq \|x - y\|_2^2 &= x^Tx + y^Ty - 2x^Ty \\
2x^Ty &\leq x^Tx + y^Ty \\
2(-x)^Ty &\leq (-x)^T(-x) + y^Ty \\
-2x^Ty &\leq x^Tx + y^Ty, \\
\therefore -2\bar{f}^T\hat{f} &\leq \bar{f}^T\bar{f} + \hat{f}^T\hat{f}, \text{ (Setting } x = \bar{f}, y = \hat{f})
\end{aligned}$$

then Equation 2 becomes:

$$\begin{aligned}
\|\nabla_\theta\|_2^2 &\leq \bar{f}^T\bar{f} + \hat{f}^T\hat{f} + \bar{f}^T\bar{f} + \hat{f}^T\hat{f} = 2\bar{f}^T\bar{f} + 2\hat{f}^T\hat{f} \\
&\leq 4d. \text{ (Since } \bar{f}, \hat{f} \in [0,1]^d)
\end{aligned} \tag{3}$$

Thus, using Equation 3 in Equation 1, and that the projection step of θ (constraining the set of θ to be the convex ball with radius B) ensures $\|\theta\|_2 \leq B$:

$$\mathcal{R}_t \leq \frac{B^2}{2\lambda} + \lambda \sum_{i=1}^{t} 4d = \frac{B^2}{2\lambda} + 4\lambda td.$$

With the minimizing choice of $\lambda = \frac{B}{2\sqrt{2td}}$,

$$\mathcal{R}_t \leq B\sqrt{2td} + \frac{2Btd}{\sqrt{2td}} = 2B\sqrt{2td}$$

□

Therefore, our algorithm is no-regret ($\lim_{t\to\infty} \frac{\mathcal{R}_t}{t} = 0$), which guarantees the quality of our continuous forecasting model with respect to one learned in batch. Our experiments also confirm this property.

4 GENERALIZED ACTIVITY FORECASTING

Without Line 9, Algorithm 3 only describes our online IRL process to infer the reward function. In order to make incremental predictions about the person's future behaviors online, we can leverage the current MDP and reward function. An important function which lays the basis for predicting future behaviors is the state visitation function, denoted D. We now show how D can be modified to perform generalized queries about future behavior.

4.1 State Visitation Function D

Using the current estimate of the MDP and the reward function, we can compute the policy of the agent. Using the policy, we can forward simulate a distribution of all possible futures. This distribution is called the state visitation distribution [52]. More formally, the posterior expected count of future visitation to a state s_x can be defined as

$$D_{s_x|\xi_{0\to t}} \triangleq \mathbb{E}_{P(\xi_{t+1\to T}|\xi_{0\to t})}\left[\sum_{\tau=t+1}^{T} I(s_\tau = s_x)\right], \quad (4)$$

where T is the maximum horizon of a single episode. This quantity represents the agent's expectation to visit each state in the future given the partial trajectory. $\xi_{0\to t}$ indicates a partial trajectory starting at time 0 and ending at time t. The expectation is taken under the maximum causal entropy distribution, $P(\xi_{t+1\to T}|\xi_{0\to t})$, which gives the probability of a future trajectory given the current trajectory. I is the indicator function, which counts agent visits to s_x. Equation 4 is computed in terms of the value function via Equation 9.4 of [52] (in which T is not explicitly) during computation, and is also employed in the ONLINEIRL Algorithm shown in Figure 4, line 5 to compute $\hat{f}$.

4.2 Activity Forecasting with State Subsets

In this work, we extend the idea of state visitations to a single state s_x to a more general *subset of states* $\mathcal{S}_p$. While a generalized prediction task was not particularly meaningful in the context of trajectory prediction [11], [53], predictions over a subset of states now represents semantically meaningful concepts in our proposed MDP. By using the state space representation of our first-person behavior model, we can construct subsets of the state space that have interesting semantic meaning, such as "having an object o_i" or "all states closest to goal k with $\mathcal{O}_j$ set of objects."

Formally, we define the expected count of visitation to a subset of states $\mathcal{S}_p$ satisfying some property p:

$$D_{\mathcal{S}_p|\xi_{0\to t}} \triangleq \mathbb{E}_{P(\xi_{t+1\to T}|\xi_{0\to t})}\left[\sum_{\tau=t+1}^{T} I(s_\tau \in \mathcal{S}_p)\right] \quad (5)$$

$$= \sum_{s_x\in\mathcal{S}_p} \mathbb{E}_{P(\xi_{t+1\to T}|\xi_{0\to t})}\left[\sum_{\tau=t+1}^{T} I(s_\tau = s_x)\right]$$

$$= \sum_{s_x\in\mathcal{S}_p} D_{s_x|\xi_{0\to t}}. \quad (6)$$

Equation 6 is essentially marginalizing over the state subspace of Equation 4.

4.3 Forecasting Trajectory Length

Leveraging Equation 6, we present a method to predict the length of the future trajectory. Formally, we can denote the expected trajectory length:

$$\hat{\tau}_{\xi_{t+1\to T}|\xi_{0\to t}} \triangleq \mathbb{E}_{P(\xi_{t+1\to T}|\xi_{0\to t})}\left|\xi_{t+1\to T}\right| \quad (7)$$

Consider evaluating $D_{\mathcal{S}_p|\xi_{0\to t}}$ from Equation 6 by setting $\mathcal{S}_p = \mathcal{S}$, that is, by considering the expected future visitation count to the entire state space. Then,

$$\begin{aligned} D_{\mathcal{S}|\xi_{0\to t}} &= \mathbb{E}_{P(\xi_{t+1\to T}|\xi_{0\to t})}\left[\sum_{\tau=t+1}^{T} I(s_\tau \in \mathcal{S})\right] \\ &= \mathbb{E}_{P(\xi_{t+1\to T}|\xi_{0\to t})}\left[\sum_{\tau=t+1}^{T} 1\right] \\ &= \mathbb{E}\left|\xi_{t+1\to T}\right| = \hat{\tau}_{\xi_{t+1\to T}|\xi_{0\to t}} \end{aligned} \quad (8)$$

where $|\xi|$ indicates the number of states in trajectory ξ.

4.4 Future Goal Forecasting

As previously described, we wish to predict the final goal of a person's action sequence. For example, if I went to the study to pick up a cup, how likely am I to go to the kitchen versus the living room? This problem can be posed as solving for the MAP estimate of $P(g|\xi)\forall g \in \mathcal{S}_g$, the posterior over goals. It describes *what goal the user seeks given their current trajectory*, defined as:

$$P(g|\xi_{0\to t}) \propto P(g)e^{V_{s_t}(g)-V_{s_0}(g)}, \quad (9)$$

where $V_{s_i}(g)$ is the *value* of g with respect to a partial trajectory that ends in s_i. The value function is computed from the learned reward function, see [52] for details. Notice that the likelihood term is exponentially proportional to the value difference between the start state s_0 and the current state s_t. In this way, the likelihood encodes the progress made towards a goal g in terms of the value function. This progress is a function of the current spatial and activity trajectory, and encodes a representation of how the person tends to behave. We illustrate that our model learns intuitive values in Figure 5: the model becomes more certain in the person's future goal after receiving additional evidence (a detection that they acquired a mug).

4.5 Derivation of Other Forecasting Tasks

Our use of the Maximum Entropy Inverse Reinforcement Learning framework enables us to construct additional meaningful inference tasks. While we do not evaluate these tasks, we provide them to illustrate how our approach can be efficiently extended.

4.5.1 Action-Subspace Visitation

To derive the action-subspace visitation, we first use the posterior expected visitation count of performing an action a_y immediately after arriving at a state s_x is given in Equation 10, from [52].

$$D_{a_y,s_x|\xi_{0\to t}} \triangleq \mathbb{E}_{P(\xi_{t+1\to T}|\xi_{0\to t})}\left[\sum_{\tau=t+1}^{T} I(s_\tau = s_x, a_\tau = a_y)\right] \quad (10)$$

$$D_{a_y,s_x|\xi_{0\to t}} = \pi(a_y|s_x)D_{s_x|\xi_{0\to t}} \quad (11)$$

Our definition of the posterior expected action subspace visitation count is given in Equation 12. This expresses the future expectation the user will perform an action a_y

while in a subspace S_p, given their current trajectory $\xi_{0\rightarrow t}$. Hereafter, we denote $\mathbb{E} \triangleq \mathbb{E}_{P(\xi_{t+1\rightarrow T}|\xi_{0\rightarrow t})}$ for brevity.

$$\begin{aligned}
D_{a_y,\mathcal{S}_p|\xi_{0\rightarrow t}} &\triangleq \mathbb{E}\left[\sum_{\tau=t+1}^{T} I(s_\tau \in \mathcal{S}_p) I(a_\tau = a_y)\right] \qquad (12)\\
&= \mathbb{E}\left[\sum_{s_x\in\mathcal{S}_p}\sum_{\tau=t+1}^{T} I(s_\tau = s_x, a_\tau = a_y)\right]\\
&= \sum_{s_x\in\mathcal{S}_p} \mathbb{E}\left[\sum_{\tau=t+1}^{T} I(s_\tau = s_x, a_\tau = a_y)\right]\\
&= \sum_{s_x\in\mathcal{S}_p} D_{a_y,s_x|\xi_{0\rightarrow t}} = \sum_{s_x\in\mathcal{S}_p} \pi(a_y|s_x) D_{s_x|\xi_{0\rightarrow t}}.
\end{aligned}$$

Thus, the posterior expected action subspace visitation is straightforward to compute with $D_{s_x|\xi_{0\rightarrow t}}$. Various inference tasks can be constructed by choosing a_y and $\mathcal{S}_p$ appropriately.

4.5.2 Joint Action-State Subspace Visitation

We additionally derive the expected transition count from a subspace of states to a subspace of actions. This expresses the expectation that the user will perform an $a_y \in \mathcal{A}_y$ from a $s_x \in \mathcal{S}_p$. It is defined as:

$$\begin{aligned}
D_{\mathcal{A}_y,\mathcal{S}_p|\xi_{0\rightarrow t}} &\triangleq \mathbb{E}\left[\sum_{\tau=t+1}^{T} I(s_\tau \in \mathcal{S}_p) I(a_\tau \in \mathcal{A}_y)\right] \qquad (13)\\
&= \mathbb{E}\left[\sum_{\substack{a_y\in\mathcal{A}_y\\ s_x\in\mathcal{S}_p}}\sum_{\tau=t+1}^{T} I(s_\tau = s_x, a_\tau = a_y)\right]\\
&= \sum_{\substack{a_y\in\mathcal{A}_y\\ s_x\in\mathcal{S}_p}} \mathbb{E}\left[\sum_{\tau=t+1}^{T} I(s_\tau = s_x, a_\tau = a_y)\right]\\
&= \sum_{\substack{a_y\in\mathcal{A}_y\\ s_x\in\mathcal{S}_p}} D_{a_y,s_x|\xi_{0\rightarrow t}} = \sum_{\substack{a_y\in\mathcal{A}_y\\ s_x\in\mathcal{S}_p}} \pi(a_y|s_x) D_{s_x|\xi_{0\rightarrow t}}.
\end{aligned}$$

Again, computing this quantity is straightforward with $D_{s_x|\xi_{0\rightarrow t}}$. By marginalizing $D_{s_x|\xi_{0\rightarrow}}$ over various action and state subspaces that have semantic meaning, different inference quantities can be expressed and computed. The construction and evaluation of new and richer inference quantities is a promising direction for future work. For example, consider the forecasting task of predicting whether a person will get a drink in the kitchen (as opposed to predicting a posterior over the goal states). The person might not use a mug, they may use a bottle or another container. The set of relevant outcomes corresponds to a joint action-state subspace: one of several actions (*e.g.* acquire bottle, acquire mug) in one of several states (*e.g.* near cupboard A, near cupboard B).

5 Experiments

We first present the dataset we collected. Then, we discuss our methods for goal discovery and action recognition. To reiterate, our focus is not to engineer these methods, but instead to make intelligent use of their outputs in DARKO for the purpose of behavior modeling. We compare DARKO's performance versus several baselines on the task of goal forecasting, and show DARKO's performance is superior. Next, we analyze DARKO's performance under less noisy conditions, to illustrate how it improves when provided with more robust goal discovery and action detection algorithms. Then, we illustrate DARKO 's empirical no-regret performance, which further shows it is an efficient online learning algorithm. Then, we conduct additional analyses including feature ablation and incorporating uncertainty from goal discovery. Finally, we present trajectory length forecasting results, and find that our length forecasts exhibit low median error.

TABLE 1
Scene types available in each environment.

Environment	Scene Type Set
Home 1	{bathroom, bedroom, exit, dining room, kitchen, living room, office}
Home 2	{bathroom, bedroom, exit, dining room, kitchen, living room, office, television stand}
Office 1	{bathroom, exit, kitchen, lounge, office, printer station, water fountain}
Office 2	{bathroom, exit, kitchen, lounge, office, printer room, water fountain}
Lab 1	{cabinet stand, exit, gel room, lab bench 1, lab bench 2, refrigeration room}

TABLE 2
Objects available in each environment.

Environment	Object Set
Home 1	{bookbag, book, blanket, coat, laptop, mug, plate, snack, towel}
Home 2	{bookbag, book, blanket, coat, guitar, laptop, mug, plate, remote, snack, towel}
Office 1	{bookbag, textbook, bottle, coat, laptop, mug, paper, plate, snack}
Office 2	{bookbag, textbook, bottle, coat, laptop, mug, paper, plate, snack}
Lab 1	{beaker, coat, plate, pipette, tube}

TABLE 3
Labels example: A small snippet of ground truth labels for Home 1.

Frame Index	6750	6900	7200
Action/Arrival	Release Coat	Acquire Bookbag	Arrive Office
Frame Index	7400	7630	7700
Action/Arrival	Acquire Mug	Arrive Kitchen	Release Mug

5.1 First-Person Continuous Activity Dataset

We collected a dataset of sequential goal-seeking behavior in several different environments such as home, office and laboratory. The users (all graduate students in their 20s) recorded a series of activities that naturally occur in each scenario. Each user helped design the script they followed, which involved their prior assumptions about what objects they will use and what goal they will seek. An example direction from a script is "obtain a snack and plate in kitchen, eat at dining room table."

Users wore a hat-mounted Go-Pro Hero camera with 94° vertical, 123° horizontal FOVs. After collection, we labelled our dataset by tagging each video with OpenShot Video Editor, and converting the tagged files to sequences of labels in plaintext format. While there was some temporal ambiguity of a few seconds in deciding what exact frame corresponded to each action, this ambiguity was tolerable for our purpose of high-level forecasting.

Our dataset is comprised of 5 user environments, and includes over 250 actions with 19 objects, 17 different scene types, at least 6 activity goals per environment, and about 200 high-level activities (trajectories). In each environment, the user recorded 3–4 long sequences of high-level activities, where each sequence represents a full day of behavior. Our dataset represents over 15 days of recording. The scenes present in each environment are shown in Table 1, and the objects available in each environment are shown in Table 2. While a potential dataset of non-scripted behavior may suffer from data outliers and class imbalance, our scripted dataset did not have significant bias in how much each category was interacted with. In our dataset, each object was interacted with and each scene-type goal was achieved at least 5 times.

For evaluation, all ground truth labels of objects (*e.g.* cup, backpack), actions (*i.e.* acquire, release) and goals (*e.g.* kitchen, bedroom) were first manually annotated. A goal label correspond to *when* a high-level direction was completed, and *in which scene* it was completed, *e.g.* (dining room, time=$65s$). An action label indicates when an activity was performed, *e.g.* (acquire, cup, time=$25s$). A small example of labels is shown in Table 3.

5.2 Goal Discovery and Action Recognition

We describe two goal discovery methods and an action recognition method that we implemented to serve as input to DARKO. With respect to Algorithm 3, these are GOALDET and ACTDET.

5.3 Scene-based Goal Discovery

This model assumes that if a scene classifier is very confident in the scene type for several images frames, the camera wearer must be in a meaningful place in the environment (*i.e.*., kitchen, bedroom, office). We use the output of a scene classifier from [51] (GoogLeNet model) on every frame from the wearable camera. If the mean scene classifier probability for a scene type is above a threshold for 20 consecutive image frames, then we add the current state s_t to the set of goals $\mathcal{S}_g$.

5.4 Stop-based Goal Discovery

This model assumes that when a person stops, they are at an important location. Using SLAM's 3D camera positions, we apply a threshold on velocity to detect stops in motion. When a stop is detected, we add the current state s_t to the set of goals $\mathcal{S}_g$. In Table 4, temporal accuracies are computed by counting detections within 3-second windows of ground truth labels as true positives; for the scene-based method, a true positive also requires the scene type to match the ground truth scene type. Stop-based discovery is more reliable across all environments, thus, we use it as our primary goal discovery method.

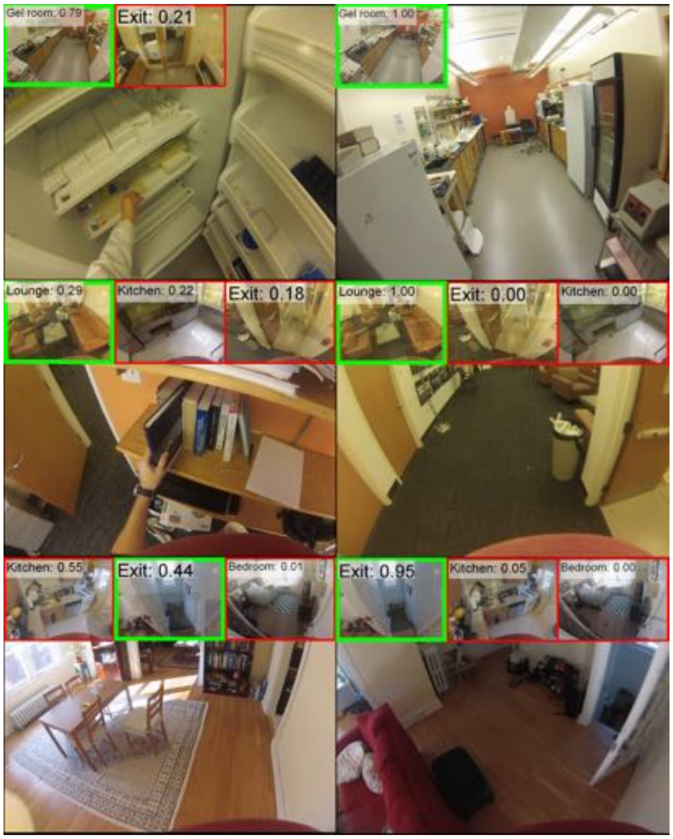


Fig. 6. **Goal forecasting examples:** A temporal sequence of goal forecasting results is shown in each row from left to right, with the forecasted goal icons and sorted goal probabilities inset (green: $P(g^*|\xi)$, red: $P(g_i \neq g^*|\xi)$).
Top: the scientist acquires a sample to boil in the gel electrophoresis room. *Middle*: the user gets a textbook and goes to the lounge. *Bottom*: the user leaves their apartment.

TABLE 4
Goal Discovery and Action Recognition. The per-scene goal discovery and action recognition accuracies are shown for our methods. A 3-second window is used around every goal discovery to compute accuracy.

Method	Home 1	Home 2	Office 1	Office 2	Lab 1
Scene Discovery	0.93	0.24	0.62	0.49	0.32
Stop Discovery	0.62	0.68	0.67	0.69	0.73
Act. Recognition	0.64	0.63	0.66	0.56	0.71

5.5 Image-based Action Recognition

We designed an object recognition approach that classifies the object the user interacts with at every temporally-labeled window. It overwrites the ground-truth object label with its detection. The approach first detects regions of `person` in each frame with [27] to focus on objects near the visible hands, which are cropped with context and fed into an image-classifier trained on ImageNet [34]. The outputs are remapped to our object set, and a final classification is produced by taking the maximum across objects. The per-action classification accuracies in Table 4 demonstrate the method can produce reasonable action classifications across all scenes. To produce the resulting action, a pairing between the relevant objects and relevant actions is manually con-

TABLE 5
Goal Forecasting Results (Visual Detections): Proposed goal posterior (Sec.4.4) achieves best $\overline{P}_{g^*}$ (mean probability of true goal).

Method	Home 1	Home 2	Office 1	Office 2	Lab 1
DARKO	**0.524**	**0.378**	**0.667**	**0.392**	**0.473**
MMED [9]	0.403	0.299	0.600	0.382	0.297
RNN	0.291	0.274	0.397	0.313	0.455
Logistic	0.458	0.297	0.569	0.323	0.348
Uniform	0.181	0.098	0.233	0.111	0.113

structed *a priori*. Passing an object detection into this pairing yields the resulting object-based action. While imperfect, these detections serve as useful input to DARKO.

5.6 Goal Forecasting Performance

At every time step, our method predicts the user's goal or final destination (*e.g.*., bedroom, exit) as described in Section 4.4 and shown in Figure 6.

5.6.1 Baseline goal forecasting models

To understand the goal prediction reliability, we compare our approach to several baseline methods for estimating the goal posterior $P(g|\xi_{0\to t})$, where g is a goal and $\xi_{0\to t}$ is the observed state sequence up to the current time step. Each baseline requires the state tracking and goal discovery components of DARKO.

Uniform Model (Uniform): This model returns a uniform posterior over possible goals $P_n(g) = 1/K_n$ known at the current episode n, defining worst case performance.

Logistic Regression Model (Logistic): A logistic regression model $P_n(g|s_t)$ is fit to map states s_t to goals g. We used the implementation available in scikit-learn [22].

Max-Margin Event Detection (MMED) [9]: A set of max-margin models $P_n(g|\phi(s_{t:t-w}))$ are trained to map features ϕ of a w-step history of state vectors $s_{t:t-w}$ to a goal score. We found the `sumL1norm` features provided with the publicly available code to perform the best, and report those best results.

RNN Classifier (RNN): An RNN is trained to predict $P_n(g|\xi_{0\to t})$. We experimented with a variety of parameters and report the best results. After each goal is detected, the RNN is refit. The settings we experimented with are cell $\in$ {GRU, Basic}, learning rate $\in \{0.1, 0.01, 0.001, 0.0001\}$, hidden dimension $\in \{8, 16, 32, 64\}$, epochs after each goal $\in \{5, 10, 50, 100\}$. We use the implementations available in [1].

Since all methods above are online algorithms, each of the models P_n is updated after every episode n. In order quantify performance with a single score, we use the mean probability assigned to the ground truth goal type g^* over all episodes $\{\xi_n\}_{n=1}^N$:

$$\overline{P}(g^*|\{\xi_n\}_{n=1}^N) = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n} P_n(g^*|\xi_{nt}) \tag{14}$$

We denote Equation 14 as $\overline{P}_{g^*}$ for brevity. The goal forecasting performance results are summarized in Table 5 using $\overline{P}_{g^*}$.

TABLE 6
Goal Forecasting Results (Labelled Detections): Proposed goal posterior achieves best $\overline{P}_{g^*}$ (mean probability of true goal). Methods benefit from better detections.

Method	Home 1	Home 2	Office 1	Office 2	Lab 1
DARKO	**0.851**	**0.683**	**0.700**	**0.666**	**0.880**
MMED [9]	0.648	0.563	0.589	0.624	0.683
RNN	0.441	0.322	0.504	0.454	0.651
Logistic	0.517	0.519	0.650	0.657	0.774
Uniform	0.153	0.128	0.154	0.151	0.167

TABLE 7
Visual goal discovery: Better goal discovery (cf. Table 4) yields better $\overline{P}_{g^*}$. Here, action detection labels are used to isolate performance differences.

Method	Home 1	Home 2	Office 1	Office 2	Lab 1
Scene-based	0.438	0.346	0.560	0.238	0.426
Stop-based	0.614	0.395	0.644	0.625	0.709

5.7 Goal Forecasting with Perfect Visual Detectors

The experimental results up to this point have exclusively used visual detectors as input (*e.g.*., SLAM, scene classification, object recognition). While we have shown that our approach learns meaningful human activity models from real computer vision input, we would also like to understand how our online IRL method performs when decoupled from the noise of the vision-based input. We perform the same experiments described in Section 5.6 but with idealized (ground truth) inputs for goal discovery and action recognition. We still use SLAM for localization.

Table 6 summarizes the mean true goal probability for each of the dataset environments. We observe a mean absolute performance improvement of 0.27 by using idealized inputs. Our proposed model continues to outperform the baselines methods. This performance indicates that as vision-based component technologies improve, we can expect significant improvements in the ability to predict the goals of human activities.

We also measure performance when the action detection is built from ground truth and the goal discovery is built from our described methods. Our expectation is that DARKO with stop-based discovery should outperform DARKO with scene-based based discovery, given the stop-detector's more reliable goal detection performance (Table 4). The results over the dataset are given in Table 7, confirming our expectation.

5.8 Goal Forecasting Performance over Time

In additional to understanding the performance of goal prediction with a single score, we also plot the performance of goal prediction over time. We evaluate the goal forecasting performance as a function of the fraction of time until reaching the goal. In Figure 7, we plot the *mean* probability of the true goal at each fractional time step $\widehat{P}(g^*|\xi_t) = \frac{1}{N}\sum_{n=1}^{N} P_n(g^*|\xi_{nt})$. Using fractional trajectory length allows for a performance comparison across trajectories of different lengths.

As shown in Figure 7, DARKO exhibits the property of maintaining uncertainty early in the trajectory and converg-

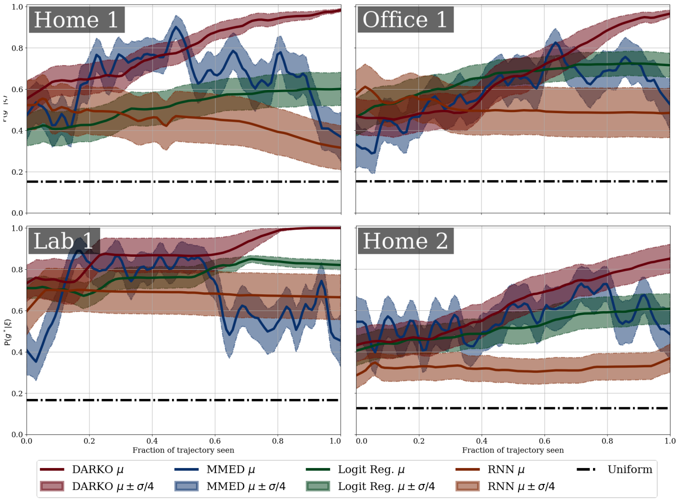


Fig. 7. **Goal posterior forecasting over time:** $\widehat{P}_{g*}$ vs. fraction of trajectory length, across all trajectories. DARKO outperforms other methods and becomes more confident in the correct goal as the trajectories elapse. Best viewed in color.

TABLE 8
Feature Ablation Results: Full state and action features (Sec. 3.1) yield best goal prediction results.

Feature Type	Home 1	Home 2	Office 1	Office 2	Lab 1
State+Action	**0.851**	**0.683**	**0.700**	**0.666**	0.880
State only	0.735	0.574	0.581	0.549	**0.892**
Position only	0.674	0.597	0.605	0.622	0.886

ing to the correct prediction as time elapses in most cases. In contrast, the logistic regression, RNN, and MMED perform worse at most time steps. As it approaches the goal, our method always produces a higher confidence in the correct goal with lower variance. We tried `argmax` and Platt scaling [24] to perform multi-class prediction with MMED; `argmax` yielded higher $\overline{P}_{g^*}$, in addition to making $\widehat{P}_{g^*}$ noisier. While the RNN sees many states, its trajectory-centric hidden-state representation may not have enough data to generalize as well as the state-centric baselines.

5.9 Reward Function Feature Ablation Analysis

In Table 8, we show the mean true goal probability when labels are used as detectors (to isolate performance in the ideal case). While the purely positional representation of state performs well, it is almost always outperformed by the full representation of rewards that include features of both the full state and action. In Lab 1, the simpler representations slightly outperform the full, due to the relative simplicity of the high-level activities in Lab 1. Here, knowledge of the state and previous goal alone is highly predictive of future goal. These results indicate that our method was able to (i) make use of simple representations, and (ii) capitalize on more information present in the richer representations without sacrificing performance where the representation is unnecessary.

5.10 Incorporating Detection Noise

Current paradigms in vision often yield noise in the action and goal detectors necessary for DARKO. We hypothesize that judicious incorporation of these uncertainties can improve our method's performance. We first describe our method for incorporating uncertainty in each goal detection, then conduct a performance analysis with synthetic noise. Then, we analyze the performance with real, noisy goal detection. We find DARKO can still perform well with forms of noisy goal and action detection. We find incorporating goal uncertainty significantly improves performance with synthetic noise, and shows improvements in the real goal detector setting. These results show that DARKO can tolerate the effects of noise, and further support the claim that it can enjoy the benefits of better scene and activity detection algorithms.

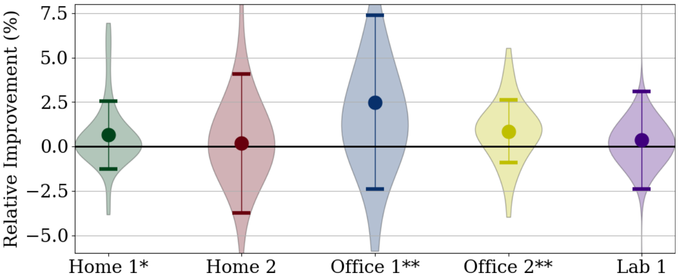


Fig. 8. **Relative improvement from incorporating goal uncertainty.** Per-scene violin plots, means, and standard deviations are shown. Per-scene one-sided paired t-tests are performed, testing the hypotheses that incorporating goal uncertainty improves goal prediction performance. A * indicates $p < 0.05$, and ** indicates $p < 0.005$.

5.10.1 Harnessing goal detection confidence

In many scenarios, probability $\rho_g \in [0, 1]$ may be associated with each goal detection. We designed an effective method for handling real-world uncertainty. For known perfect goals, SOFTVALUEITERATION uses $V(g) = 0, \forall g \in \mathcal{S}_g$. Each goal is a maxima of the value function $V(s) \in (-\infty, 0], \forall s \in \mathcal{S}$ and represents a reward of 1 in log space. *We replace each goal value with its log-probability:* $V(g) = \ln \rho_g$, *which has the effect of biasing the policy towards goals with greater certainty.* This results from the value iteration assigning higher value to states and actions closer to more certain goals, which makes the policy likelier to visit them. For example, if the goal detector yields a false positive of `bathroom` in the same area as a true positive detection of `kitchen`, the goal prediction posteriors for both goals will suffer, unless the false positive has an associated low ρ_g (high uncertainty), in which case the policy is biased towards the correct goal of kitchen.

5.10.2 Noise analysis

To test the efficacy of the goal detection confidence weighting approach, we analyze DARKO under the effect of incorporating adding noise to the GT. Noise is present in both the control and the test. The manipulated variable is $V(g)$: in the control, $V(g) = 0$ (unchanged), in the test, $V(g) = \ln \rho_g$.

We add incorrect goal detections with probabilities $\rho_g \sim \mathcal{N}(0.1, 0.05)$, under various amounts of noise inserted uniformly at random across time: from $10\%, 20\%, \ldots, 90\%$ of the number of original goal detections. For every scene, at each noise amount, we sample noise 5 times, and run

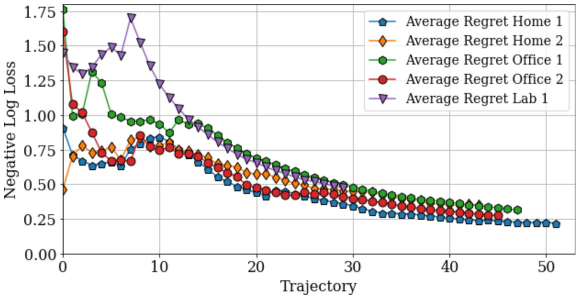


Fig. 9. **Empirical regret.** DARKO exhibits sublinear convergence in average regret. Initial noise is overcome after DARKO adjusts to the user's early behaviors.

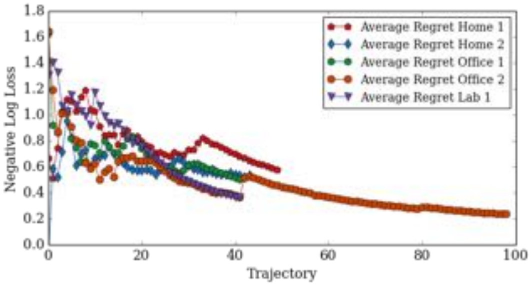


Fig. 10. **Noisy Empirical Regret.** DARKO's online behavior model exhibits sublinear convergence in average regret. Initial noise is overcome after DARKO adjusts to learning about the user's early behaviors.

DARKO with and without goal uncertainty for each corrupted sample, resulting in 225 paired experiments that evaluate the average goal forecasting probability. Per-scene results are shown in Figures 8. *A one-sided Wilcoxon signed-rank test supports the hypothesis that incorporating high goal uncertainty yields better goal posterior prediction performance than not incorporating the uncertainty with* $p < 10^{-7}$.

5.11 Empirical Regret Analysis

We empirically show that our model has no-regret with respect to the best model computed in hindsight under the MaxEntIRL loss function (negative log-loss). In particular, we compute the regret (cumulative loss difference) between our online algorithm and the best hindsight model using the batch MaxEntIOC algorithm [53] at the end of all episodes. We plot the average regret $\frac{\mathcal{R}_t}{t}$ with ground-truth action and goal detection in Figure 9. The average regret of our algorithm approaches zero over time, matching our analysis.

We additionally show the empirical regret when using our goal discovery and action detection methods in Figure 10. We observe somewhat noisier regret behavior than in the original case, as the underlying demonstrations are noisier. The number of trajectories in Office 2 is higher here due to errors in the goal forecasting method, resulting in more goals being detected, which segments the demonstrations into more trajectories.

5.12 Evaluation of Trajectory Length Estimates

Our model can also be used to estimate how long it will take a person to reach a predicted goal state. We predict the expected trajectory length as derived in Section 4.1.

TABLE 9
Trajectory length forecasting results. Error is relative to the true length of each trajectory. Most trajectory forecasts are fairly accurate.

Statistic	Home 1	Home 2	Office 1	Office 2	Lab 1
Med. % Err.	30.0	34.8	17.3	18.4	6.3
Med. % Err. NN	29.0	33.5	42.9	36.0	35.4
Mean $\lvert\xi\rvert$	20.5	31.0	27.1	13.7	23.5

For the n-th episode, we use the normalized trajectory length prediction error defined as $\epsilon_n = \sum_{t=1}^{T_n} \frac{|\tau_{nt}-\hat{\tau}_{nt}|}{\tau_{nt}}$, where τ_{nt} is the true trajectory length and $\hat{\tau}_{nt}$ (Eq. 8) is the predicted trajectory length. Proper evaluation of trajectory length towards a goal is challenging because our approach must learn valid goals in an online fashion. When a person approaches a new goal, our approach cannot accurately predict the goal because it has yet to learn that it is a valid goal state. As a result, our algorithm makes wrong goal predictions during episodes that terminate in new goal states. If we simply evaluate the mean performance, it will be dominated by the errors of the first episode terminating in a new goal state.

We evaluate median ϵ_n over all N episodes. The median is not dominated by the errors of the first episode toward a new goal. We find most trajectory length forecasts are accurate, evidenced by the median of the normalized prediction error in Table 9. We include a partial-trajectory nearest neighbors baseline (NN). In Lab 1, the median trajectory length estimate is within 6.3% of the true trajectory length.

6 VISUALIZATIONS

We provide example 3D visualizations of (i) future state visitation and (ii) the value function. These visualizations were produced in Mayavi [25], and include the SLAM points.

6.1 Future state visitation visualizations

See Figure 11 for example visualizations of the expected future visitation counts and SLAM points. In order to visualize in 3 dimensions, we take the max visitation count across all states at each position. In rows 1 and 2, a single demonstration is shown, which adapts to the agent's trajectory (history). In row 3, the future state distribution drastically changes after each time the agent reaches a new goal.

6.2 Value function visualizations

See Figure 12 for example visualizations of the value function over time and SLAM points. Note 1) the state space size changes, and 2) that the value function changes over time, as the component of state that indicates the previous goal affects the value function.

7 CONCLUSION

We proposed the first method for continuously modeling and forecasting a first-person camera wearer's future semantic behaviors at far-reaching spatial and temporal horizons. Our method goes beyond predicting the physical

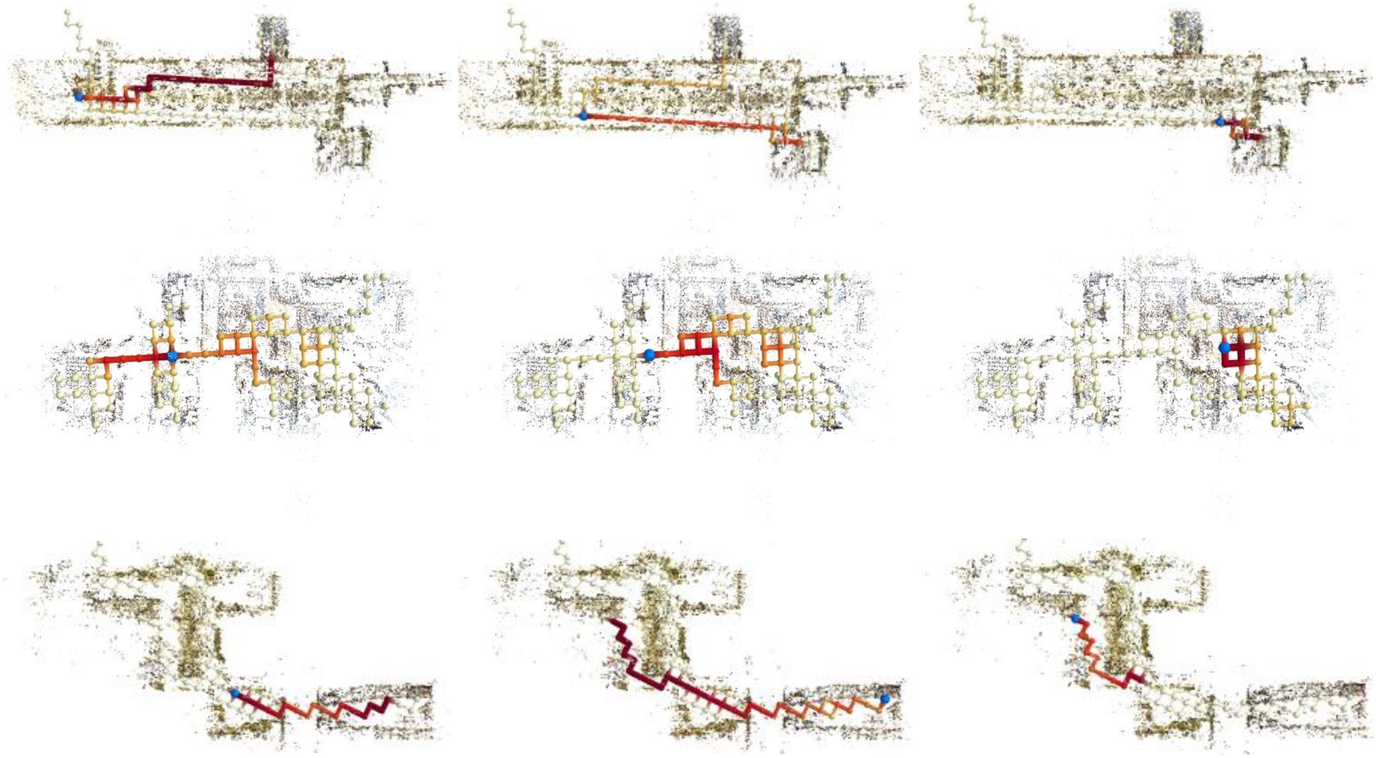


Fig. 11. Future state visitation predictions and sparse SLAM keypoints changing as the agent (blue sphere) follows their trajectory. The state visitations are projected to 3D by taking the max over all states at each location. The visualizations are, by row: Office 1, Home 1, Lab 1. Best viewed in color.

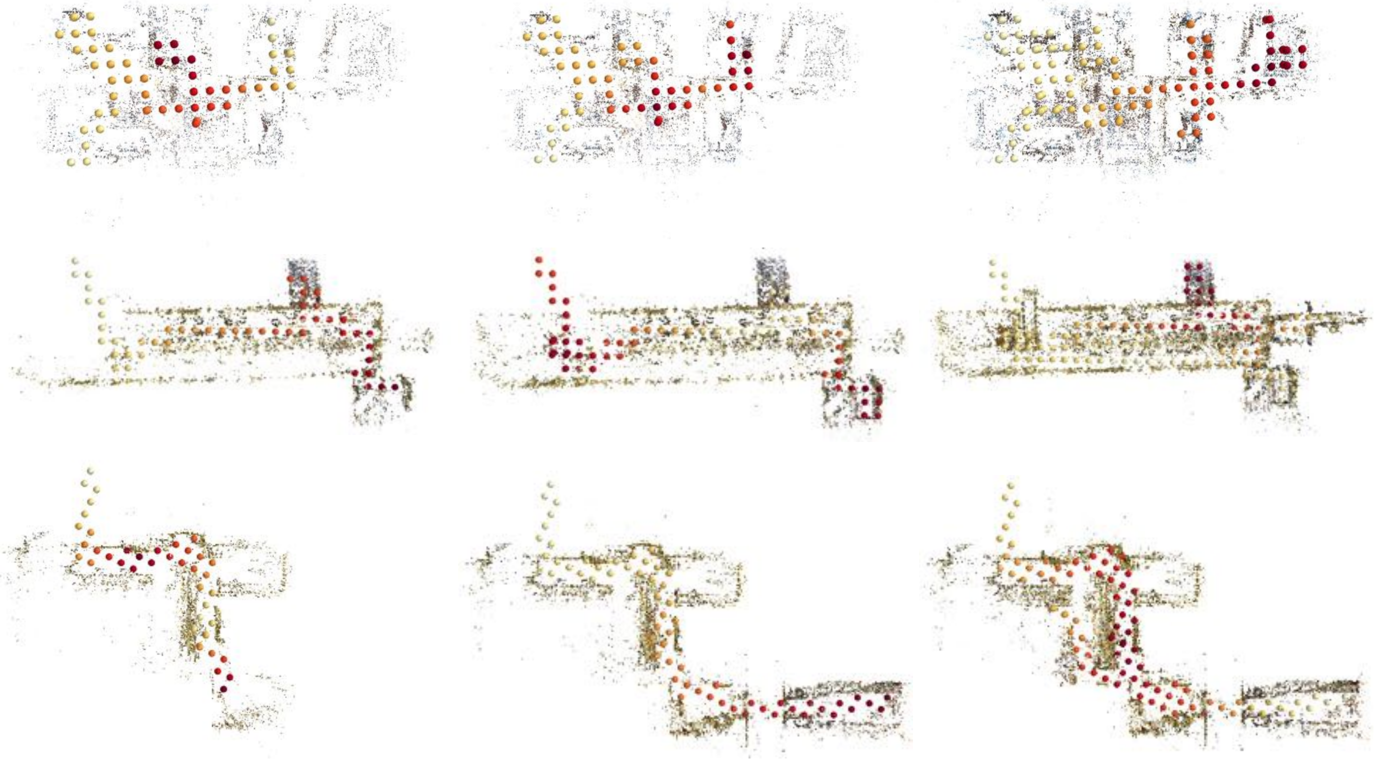


Fig. 12. Projections of the value function ($V(s)$) and sparse SLAM keypoints for environments as time elapses (left to right). The state space expands as the user visits more locations. For each position, the maximum value (across all states at that position) is displayed: $\max_{s \in \mathcal{S}_x} V(s)$. From top to bottom, the environments are Home 1, Office 1, Lab 1. Best viewed in color.

trajectory of the user to predict their future semantic goals, and models the user's relationship to objects and their environment. We have proposed several efficient and extensible methods for forecasting other semantic quantities of interest. Exciting avenues for future work include building upon the semantic state representation to model more aspects of the environment (which enables forecasting of more detailed futures), validation against human forecasting performance, and further generalizing the notion of a "goal" and how goals are discovered.

Acknowledgments

We acknowledge the support of NSF NRI Grant 1227495 and JST CREST Grant JPMJCR14E1. The authors thank J. Andrew Bagnell for technical discussion, and Katherine Lagree and Gunnar A. Sigurdsson for data collection assistance.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1. ACM, 2004.

[3] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[4] A. Bhattacharyya, M. Malinowski, B. Schiele, and M. Fritz. Long-term image boundary prediction. In *Thirty-Second AAAI Conference on Artificial Intelligence*. AAAI, 2017.

[5] F. Cakir and S. Sclaroff. Adaptive hashing for fast similarity search. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[6] Y. Cao, D. Barrett, A. Barbu, S. Narayanaswamy, H. Yu, A. Michaux, Y. Lin, S. Dickinson, J. Mark Siskind, and S. Wang. Recognize human activities from partially observed videos. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[7] A. Fathi, A. Farhadi, and J. M. Rehg. Understanding egocentric activities. In *2011 International Conference on Computer Vision*, pages 407–414. IEEE, 2011.

[8] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Towards internet-scale multi-view stereo. In *CVPR*, 2010.

[9] M. Hoai and F. De la Torre. Max-margin early event detectors. *International Journal of Computer Vision*, 107(2):191–202, 2014.

[10] X. Jia, B. De Brabandere, T. Tuytelaars, and L. V. Gool. Dynamic filter networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 667–675. Curran Associates, Inc., 2016.

[11] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert. Activity forecasting. In *European Conference on Computer Vision*, pages 201–214. Springer, 2012.

[12] H. S. Koppula and A. Saxena. Anticipating human activities using object affordances for reactive robotic response. *IEEE transactions on pattern analysis and machine intelligence*, 38(1):14–29, 2016.

[13] T. Lan, T.-C. Chen, and S. Savarese. A hierarchical representation for future action prediction. In *European Conference on Computer Vision*, pages 689–704. Springer, 2014.

[14] Y. J. Lee, J. Ghosh, and K. Grauman. Discovering important people and objects for egocentric video summarization. In *CVPR*, volume 2, page 7, 2012.

[15] Y. J. Lee and K. Grauman. Predicting important objects for egocentric video summarization. *International Journal of Computer Vision*, 114(1):38–55, 2015.

[16] K. Li and Y. Fu. Prediction of human activity by discovering temporal sequence patterns. *IEEE transactions on pattern analysis and machine intelligence*, 36(8):1644–1657, 2014.

[17] Y. Li, Z. Ye, and J. M. Rehg. Delving into egocentric actions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[18] M. Ma, H. Fan, and K. M. Kitani. Going deeper into first-person activity recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1894–1903, 2016.

[19] W.-C. Ma, D.-A. Huang, N. Lee, and K. M. Kitani. A game-theoretic approach to multi-pedestrian activity forecasting. *arXiv preprint arXiv:1604.01431*, 2016.

[20] R. Mur-Artal, J. Montiel, and J. D. Tardós. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[21] N. Neverova, P. Luc, C. Couprie, J. Verbeek, and Y. LeCun. Predicting deeper into the future of semantic segmentation. *arXiv preprint arXiv:1703.07684*, 2017.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

[23] H. Pirsiavash and D. Ramanan. Detecting activities of daily living in first-person camera views. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2847–2854. IEEE, 2012.

[24] J. Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[25] P. Ramachandran and G. Varoquaux. Mayavi: 3d visualization of scientific data. *Computing in Science & Engineering*, 13(2):40–51, 2011.

[26] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich. Maximum margin planning. In *Proceedings of the 23rd international conference on Machine learning*, pages 729–736. ACM, 2006.

[27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016.

[28] N. Rhinehart and K. M. Kitani. Learning action maps of large environments via first-person vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–588, 2016.

[29] N. Rhinehart and K. M. Kitani. First-person activity forecasting with online inverse reinforcement learning. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[30] N. Rhinehart, J. Zhou, M. Hebert, and J. A. Bagnell. Visual chunking: A list prediction framework for region-based object detection. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 5448–5454. IEEE, 2015.

[31] M. S. Ryoo. Human activity prediction: Early recognition of ongoing activities from streaming videos. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011.

[32] M. S. Ryoo and L. Matthies. First-person activity recognition: What are they doing to me? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2730–2737, 2013.

[33] S. Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

[34] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[35] H. Soo Park, J.-J. Hwang, Y. Niu, and J. Shi. Egocentric future localization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[36] S. Su, J. P. Hong, J. Shi, and H. S. Park. Social behavior prediction from first person videos. *arXiv preprint arXiv:1611.09464*, 2016.

[37] Y.-C. Su and K. Grauman. Detecting engagement in egocentric video. In *European Conference on Computer Vision*, pages 454–471. Springer, 2016.

[38] W. Sun, A. Venkatraman, B. Boots, and J. A. Bagnell. Learning to filter with predictive state inference machines. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1197–1205, 2016.

[39] A. Venkatraman, N. Rhinehart, W. Sun, L. Pinto, M. Hebert, B. Boots, K. Kitani, and J. Bagnell. Predictive-state decoders:

Encoding the future into recurrent networks. In *Advances in Neural Information Processing Systems*, pages 1172–1183, 2017.

[40] R. Villegas, J. Yang, Y. Zou, S. Sohn, X. Lin, and H. Lee. Learning to generate long-term future via hierarchical prediction. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 3560–3569, 2017.

[41] C. Vondrick, H. Pirsiavash, and A. Torralba. Anticipating visual representations from unlabeled video. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[42] C. Vondrick, H. Pirsiavash, and A. Torralba. Anticipating visual representations from unlabeled video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 98–106, 2016.

[43] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, pages 613–621, 2016.

[44] C. Vondrick and A. Torralba. Generating the future with adversarial transformers. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2992–3000, 2017.

[45] J. Walker, A. Gupta, and M. Hebert. Patch to the future: Unsupervised visual prediction. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3302–3309. IEEE, 2014.

[46] J. Walker, K. Marino, A. Gupta, and M. Hebert. The pose knows: Video forecasting by generating pose futures. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3352–3361. IEEE, 2017.

[47] W. W.-S. Wei. *Time series analysis*. Addison-Wesley publ Reading, 1994.

[48] M. Wulfmeier, P. Ondruska, and I. Posner. Maximum entropy deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*, 2015.

[49] D. Xie, S. Todorovic, and S.-C. Zhu. Inferring "dark matter" and "dark energy" from videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2224–2231, 2013.

[50] Y. Yan, E. Ricci, G. Liu, and N. Sebe. Egocentric daily activity recognition via multitask clustering. *IEEE Transactions on Image Processing*, 24(10):2984–2995, 2015.

[51] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014.

[52] B. D. Ziebart. *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. PhD thesis, Carnegie Mellon University, 2010.

[53] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *AAAI Conference on Artificial Intelligence*, pages 1433–1438, 2008.

[54] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa. Planning-based prediction for pedestrians. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3931–3936. IEEE, 2009.

Nicholas Rhinehart is a PhD student in the Robotics Institute at Carnegie Mellon University. He received a BS in Engineering and BA in Computer Science from Swarthmore College, and an MS in Robotics from Carnegie Mellon University. His research currently focuses on Reinforcement Learning and Inverse Reinforcement Learning methods at the interface of Computer Vision and Machine Learning. He is specifically interested in building decision-theoretic models that leverage rich perception sources to drive visual forecasting, functional understanding, general prediction, and general control tasks. His work was awarded the Marr Prize honorable mention at ICCV 2017.

Kris M. Kitani is an assistant research professor in the Robotics Institute at Carnegie Mellon University. He received his BS at the University of Southern California and his MS and PhD at the University of Tokyo. His research projects span the areas of computer vision, machine learning and human computer interaction. In particular, his research interests lie at the intersection of first-person vision, human activity modeling and inverse reinforcement learning. His work has been awarded the Marr Prize honorable mention at ICCV 2017, best paper honorable mention at CHI 2017, best technical paper at W4A 2017, best application paper ACCV 2014 and best paper honorable mention ECCV 2012.