Learning search behaviour from humans

Guillaume de Chambrier and Aude Billard

***Abstract*— A frequent method for taking into account the partially observable nature of an environment in which robots interact lies in formulating the problem domain as a Partially Observable Markov Decision Process (POMDP). By having humans demonstrate how to act in this partially observable context we can leverage their prior knowledge, experience and intuition, which is difficult to encode directly in a controller, to solve a task formulated as a POMDP. In this work we learn search behaviours from human demonstrators and transfer this knowledge to a robot in a context where no visual information is available. The task consists of finding a block on a table. This is a non-trivial problem since no visual information is available and as a result, the belief of the demonstrator's state (position in the environment) has to be inferred. We show that by representing the belief of the human's position in the environment by a particle filter (PF) and learning a mapping from this belief to their end-effector velocities with a Gaussian Mixture Model (GMM), we model the human's search process. We compare the different types of search behaviour demonstrated by the humans to that of our learned model, to validate that the search process has been successfully modelled. We then contrast the performance of this human-inspired search model to a greedy controller and show that (similarly to humans) the learned controller minimises uncertainty, hence demonstrating more robustness in the face of false belief.**

I. INTRODUCTION

Constructing controllers or policies to act within a context where the state space is partially observable is of high relevance to all real robotic applications. Because of inaccurate perception information, only an approximation of the environment is available at any given time. If this inherent uncertainty is not taken into account during planning or control there is a non-negligible risk of missing goals, getting lost and wasting valuable resources. This work takes a *Programming by demonstration* (PbD) approach to learn a control policy in a partially observable environment where no visual information is available. In this context an expert (human or robot) demonstrates how to accomplish a given task.

Partially Observable Markov Decision Processes (POMDP) are an extensive area of research in the operational research, planning and decision theory community [1][2]. The emphasis is to be able to act optimally when the state information is only partially available. Most large scale state-space POMDP planning problems are resolved via approximate methods such as *point-based value iteration* (PBVI) [3], in which the policy is optimized at a set of sampled points drawn from the belief space (a simplex in which a point is a probability distribution over the state space). These methods rely on exploratory/search heuristics to discover a sufficient set of probability densities to able to discover an optimal policy. Taking human demonstrations to estimate the parameters of a policy acting in a POMDP is advantageous over classical PBVI approaches, as it avoids performing the time consuming exploration step and is applicable to both continuous actions and state spaces. The demonstrations immediately provide a set of examples of the (assumed) optimal decisions. Humans perform an informed search contrary to stochastic sampling methods since they utilise past experience and are able to evaluate the cost of their actions in the future. This foresight and experience are implicitly encoded in the parameters of the learned policy. Planning and Reinforcement Learning (RL) methods reason with respect to a Markovian process where all the information required to make a decision is encapsulated in the current state and no other information is used. The discovery of the optimal path, embedding the implicit information, is difficult to retrieve in this Markovian setting.

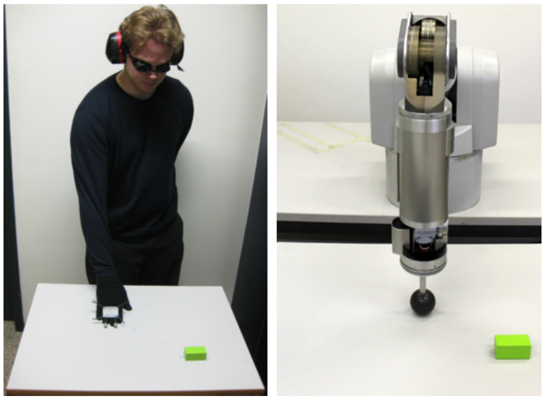

Fig. 1: *Left:* Human demonstrator searching for the green wooden block on the table given that both vision and hearing senses are impeded. *Right:* WAM Robot 7 DOF reproduces the search strategies demonstrated by humans to find the object.

In this work we consider a task in which both a robot and a human must search for an object on a table whilst deprived of vision. The environmental setup is prior knowledge to the robot and the human making this a specific search problem with no required mapping of the environment. In figure 1, a human has his sense of vision and hearing impeded, making the perception of the environment partially observable and only leaving the sense of touch available for solving the task. Before each demonstration the human volunteer is disoriented. His transitional position is varied with respect to the table and his heading remains the same (facing the table)

G. de Chambrier and A. Billard are with the Learning Algorithms and Systems Laboratory (LASA), School of Engineering, École Polytéchnique Fédérale de Lausanne (EPFL), Switzerland

leaving the uncertainty component out of the orientation. The reason for the disorientation step is to ensure that the human's believed location is uniform. At the first time step, the human's state of mind can be considered observable. All proceeding beliefs can then be recursively estimated from the initial belief. The hearing sense was impeded since it can facilitate localisation when no visual information is available and the robot has no equivalent giving an unfair advantage to the human. By impeding hearing we reduce the perception correspondence between the human and robot.

A crucial aspect of our work is that the robot should be capable of inferring the belief of the human doing the search. Work on modelling human being beliefs and intentions [4][5] has been undertaken in cognitive science. This work was able to show that humans perform inference in a similar fashion to Bayesian models. Our work takes this further by combining the modelling of both belief and action. The performance of the model is evaluated in three separate ways: 1) whether the search output of the model is comparable to that of humans 2) how well the model performs against a greedy approach when solving the search task and 3) how robust the model is to false beliefs.

II. RELATED WORK

The domain of our work lies at the intersection of three fields namely programming by demonstration, cognitive science and acting under uncertainty. We review the latest developments in each field, highlighting the relevance to our work.

In many PbD research studies, single one-shot successful demonstrations (such as *Pick & Place*) in fully observable environments have been encoded through either statistical methods such as GMM with Stable Estimator of State Dynamics (SEDS) [6], in a latent space with Gaussian Process (GP) [7] or trajectory encoding methods such as Dynamic Motion Primitives (DMP) [8] and splines. For an in-depth review of PbD the reader may refer to [9]. The benefits of these approaches is the dramatic reduction in the search space of the optimal policy through leveraging human knowledge, but no work has be undertaken to make them compatible in a context where the state space in not fully observable.

One aspect of our work employs a probabilistic representation of a human's belief over his state in the environment. Human mind attributes, such as beliefs, desires and intentions, are not directly observable. They have to be inferred from actions. In [10], the authors present a Bayesian framework for modeling the way humans reason about and predict actions of an intentional agent. The comparison between the model and humans' predictions when asked to infer the intentions of an agent in a 2D world yielded similar inference capabilities. This provided evidence supporting the hypothesis that human beings integrate information using Bayes rule. Further, in [4], a similar experiment was performed in which the inference capabilities of humans, with regards to both belief and desire of an agent, were comparable to that of their Bayesian model.

Many robotic applications have to handle the partially observable nature of the environment they act in. A widely used approach to model the dynamics of the problem is to formulate it as a POMDP. Value Iteration (VI) [11] (employed for discrete state and action space in RL) is a popular approach to learn a policy in a POMDP. However an exact solution only exists in a discrete encoding of the state-action space [12, p.513]. This is due to the fact that the value function is defined over the space of state belief. As the agent can occupy a large number of possible states, the computational costs grow exponentially. As a result much effort has been put into evaluating an approximation of the value function at a set of representative beliefs rather than over the full belief space. Such methods fall under the category of PBVI [3] in which most research has focused on determining the best set of beliefs, [13] to be evaluated by the value function, see [14] for a review. Other approaches compress the belief to sufficient statistics (mean and entropy) as in [15] and perform standard VI. The draw back with these methods is that they aren't able to deal with both continuous state and action space. The noticeable exception is Monte Carlo POMDP [16] which represents the belief of the position of a robot by a particle filter. However the value function is difficult to compute and requires storing belief instantiations for evaluating new unseen beliefs. The major draw back of all these approaches lies with the exploration problem which becomes infeasible as the number of states and actions increase.

Decision-theoretic based approaches have also been applied. Notable examples are [17] and [18] where a decision tree graph is constructed with nodes representing beliefs and edges actions. An time horizon planner is used which makes a trade of between reducing uncertainty and achieving the wanted goal. The shortcomings of these methods lie with the computational cost of constructing the search tree with PF for the belief nodes. It also effects the responsiveness of the system which takes time to perform the planning

Our work differs from the above approaches in that we use human experts to provide training data on how to solve a specific task in a POMDP setting. The benefits of our approach are that we can use both continuous actions and state spaces and largely reduce the exploration problem since we can leverage the prior knowledge by means of the human demonstrations. These demonstrations restrict the solution's search space and hence free us from having to explore all the branches of the belief-state-action tree.

III. PROBLEM STATEMENT

The search task being considered is to find a wooden block on the table given that both vision and hearing senses have been impeded. The first consideration is that the human or robot localises himself in the environment. He/It then navigates towards the goal using a range of strategies ranging from risk averse strategies, where the path taken remains close to salient features so as to not to get lost, to risk taking strategies where the person follows the shortest path to the goal's location. It is non-trivial to have a robot learn the

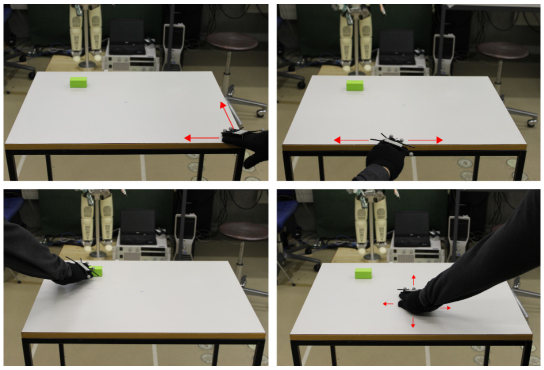

Fig. 2: A participant is trying to locate the green wooden block on the table given that both vision and hearing senses have been inhibited. A black glove is worn which has had its fingers sewn together in order to limit the variability of motion. These two measures were taken in order to equate the human's level of perception to that of the robot, and hence reduce the correspondence problem. The top of the glove harbours a small platform with three reflective markers which are used to track the hand with the OptiTrack® system.

behaviour exhibited by humans performing this task. As we cannot encapsulate the true complexity of human thinking, we take a simplistic approach and model the human's state through two variables. The first variable is the human's uncertainty about his current location. The second variable is the human's belief of his position. The various strategies adopted by human's are modelled by building a mapping from the state variables to actions, consisting of the motion of the human arm. Aside from the problem of correctly approximating the belief and its evolution over time, the model need to take into consideration that people act very differently given the same situation. As a result it is not just a single strategy that will be transferred but rather a mixture of strategies. While this will provide the robot with a rich portfolio of search strategies, appropriate methods must be developed to encode these, at times, contradictory strategies.

A. Experimental setup

In the experimental setup, a group of 15 human volunteers were asked to search for a wooden green block located at a fixed position on a bare table, see figure 2. Each participant repeated the experiment 10 times from each of 4 mean starting points with an associated small variance. These starting positions were: in front, to the left, to the right, and being on the table itself. Before each trial the participant was told that he would always be facing the same direction with respect to the table (so always facing the goal, like in the case of a door) but his transitional starting position would vary. For instance, the table might not be always directly in front of him and his distance to the edge or corner could be varied.

B. Formulation

In the standard PbD formulation of this problem, a parametrised function is learned, mapping from state x, which denotes the current position of the demonstrator's hand, to $\dot{x}$, which denotes the displacement of the hand at the next time step. In our case since the environment is partially observable we have a belief or probability density function $p(x_t|z_{0:t})$, which is conditioned on all sensing information z up to time t, over the state space at any given point in time. We seek to learn this mapping from demonstrations:

$$f : p(x_t|z_{0:t}) \mapsto \dot{x} \tag{1}$$

During each demonstration we record a set of variables consisting of the following:

1) $\dot{x} \in \mathbb{R}^3$, velocity of the hand in Cartesian space, which is normalised.
2) $\hat{x} = \arg\max_x p(x_t|z_{0:t})$, the most likely position of the end-effector, or believed position.
3) $U \in \mathbb{R}$, the level of uncertainty which is evaluated through the entropy of $p(x_t|z_{0:t})$.

A statistical controller was learned from a data set of triples $\{(x, \hat{x}, U)\}$ and a desired direction (normalised velocity), was obtained from conditioning on the belief and uncertainty.

Having described the experiment and the type of data, we proceed to give an in-depth description of the mathematical representation of the belief and that of the dynamics.

IV. MODEL OF BELIEF

A human's belief of his location in an environment can be multimodal or unimodal, gaussian or non-gaussian and may change from one distribution to another. To be able to represent such a wide range of probability distributions we chose a particle filter. From previous literature [4] it has been shown that there is a similarity between Bayes update rule and the way humans integrate information over time. Under this assumption we hypotheses that if the initial belief of the human is known then the successive update steps of the particle filter should correspond to a good approximation of the next beliefs.

A particle filter is a Bayesian probabilistic methods which recursively integrates dynamics and sensing to estimate a posterior from a prior probability density. The particle filter has two elements. The first estimates a distribution over the possible next state given dynamics and the second corrects it through integrating sensing. Given a *motion model* $p(x_t|x_{t-1}, \dot{x}_t)$, and a *sensing model* $p(z_t|x_t)$, we recursively apply a prediction phase, where we incorporate motion to update the state time index, and an update phase, where the sensing data is used to compute the state's posterior distribution. The two steps are depicted below.

prediction:

$$p(x_t|z_{0:t-1}) = \int p(x_t|x_{t-1}, \dot{x}_t)\, p(x_{t-1}|z_{0:t-1})\, dx_{t-1} \tag{2}$$

update:

$$p(x_t|z_{0:t}) = \frac{p(z_t|x_t)p(x_t|z_{0:t-1})}{p(z_t|z_{0:t-1})} \tag{3}$$

The probability distribution over the state $p(x_t|z_{0:t})$ is represented by a set of weighted particles $\{w_i, x_i\}^{i=1\ldots N}$ which represent hypothetical locations of the end-effector and their density which is proportional to the likelihood.The particular particle filter used was the *Regularised Sequential Importance Sampling* [19, p.182]. We proceed to describe the two components needed for filtering namely the sensing and the motion models.

A. Sensing model

The sensing model represents the likelihood, $p(z|x)$, of a particular sensation z given a position x. In a human's case, the sensation of a curvature indicates the likelihood of being near an edge or a corner. However the likelihood cannot be modelled through using the human's sensing information. Direct access to pressure, temperature and such salient information is not available. Real sensory information needs to be matched against virtual sensation at each hypothetical location x of a particle. Additionally, for the transfer of behaviour from human to robot to be successful, the robot should be able to perceive the same information as the human, given the same situation. An approximation of what a human or robot senses can be inferred, based on the end-effector's distance to particular features in the environment. In our case four main features are present, namely corners, edges, surfaces and an additional dummy feature defining no contact, air. The choice of these features is prior knowledge given to our system and not extracted through statistical analysis of recorded trajectories. We represent the sensing model as a Multinomial distribution, M, evaluated from the normalised histogram of the euclidean distance to the closest features in the environment. The likelihood is evaluated by taking the Jensen-Shannon divergence (JSD) of the Multinomial distribution of the actual real inferred sensation $M_{\mathbf{r}}$ and that of the hypothetical virtual $M_{\mathbf{v}}$ sensation.

$$p(z|x) = 1 - JSD(M_{\mathbf{r}}||M_{\mathbf{v}}) \tag{4}$$

B. Motion model

The motion model is straight forward compared with the sensing model. In the robot's case the Jacobian gives the next Cartesian position given current joint angles and angular velocity of the robot's joints. From this the motion model is given by:

$$\dot{x} = \mathrm{J}(q)\dot{q} + \epsilon \tag{5}$$

where q is the angular position of the robot's joints, $J(q)$ is the Jacobian and $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ is white noise. The robot's motion is very precise and it's noise variance is very low. For humans, the motion model is the velocity of the hand movement provided by the tracking system.

V. STATISTICAL MODEL OF SEARCH

A detailed description is given next on, A) the computation of the uncertainty and belief, B) the statistical encoding of the strategies demonstrated by the human volunteer and C) the combination of the two in a control loop.

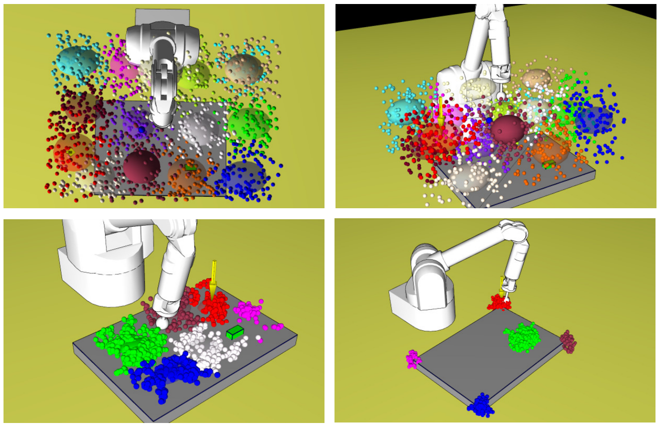

Fig. 3: Representation of the estimated density function. *Top Left and Right:* Initial starting point, all Gaussian functions are uniformly distributed with uniform priors. The red cluster always has the highest likelihood (indicated by the yellow arrow) is taken to be the believed location of the robots/humans end-effector. *Bottom Left:* Contact with the table has been established, the robot location differers with his belief. *Bottom Right:* Contact has been made with a corner, the clusters reflect that the robot could be at any corner (note that weights are not depicted, only cluster assignment).

A. Uncertainty & Belief

A natural framework to represent uncertainty in the context of probability distributions is entropy. It is the expectation of a random variable's total amount of unpredictability. The higher the entropy the more uncertainty, and the lower the less uncertainty. In our context we don't have at our disposition the true probability density function of the belief, $p(x|z)$, but instead a set of weighted samples, $\{w_i, x_i\}^{i=1\dots N}$, drawn from it. A reconstruction of the underlying probability density is achieved by fitting a set weighted Gaussian functions to the particles. The main difficulty of this step is determining the number of parameters of the density function in a computationally efficient manner. We approach this problem by finding all the modes in the particle set via mean-shift hill climbing and set these as the means of the Gaussian functions. Their covariances are determined by maximizing the likelihood of the density function via Expectation-Maximization (EM).

Given the estimated density we can compute the upper bound of the differential entropy [20], H, which is the uncertainty U.

$$H(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \left(-\log(\pi_k) + \frac{1}{2}\log((2\pi e)^D |\Sigma_k|) \right) \tag{6}$$

Where e is the base of the natural logarithm and D the dimension (being 3 in our case). The reason we use the upper bound is because the exact differential entropy of a Mixture of Gaussian functions has no analytical solution. We computed both the upper and lower bound and found that the difference between the two were insignificant, making any bound a good approximation of the true entropy. The choice of the believed location of the robot/human end-effector is taken to be the mean of the Gaussian function with the highest weight π. Figure 3 depicts different configurations of the modes (clusters) and believed position of the end-effector (yellow arrow).

B. Model of human search

From the trajectories recorded during the experiments, different actions are present for the same belief and uncertainty making the data multimodal (for a particular position and uncertainty different velocities are present). The Gaussian Mixture Model (GMM) was chosen as the statistical method to model the normalised velocity, belief and uncertainty. It is assumed that a mixture of strategies are present with the data gathered from the demonstrations. That is multiple actions are possible given a specific point in space or belief. This results in a one-to-many mapping which is not a valid function, eliminating any regression technique which directly learns a non-linear function.

The velocity was normalised, in order to reduce the amount of information to be learned and to take into consideration that velocity is more specific to embodiment capabilities: the robot might not be able to reproduce safely some of the velocity profiles demonstrated.

The training data set comprised a total of 20'000 triples $(\dot{x}, \hat{x}, U)$, from the 150 trajectories gathered from the demonstrators. A generative GMM $\mathcal{P}(\dot{x}, \hat{x}, U)$ was fitted, which had a total of 7 dimensions, 3 for direction, 3 for position and 1 scalar for uncertainty. The definition of the GMM is presented below in equation 7.

$$\mathcal{P}(\dot{x}, \hat{x}, U|\theta) \quad = \sum_{k=1}^{K} \pi_k \, \mathcal{N}(\dot{x}, \hat{x}, U|\mu_k, \Sigma_k) \qquad (7)$$

$$\mu_k = \begin{bmatrix} \mu_{\dot{x}} \\ \mu_{\hat{x}} \\ \mu_U \end{bmatrix} \Sigma_k = \begin{bmatrix} \Sigma_{\dot{x}\dot{x}} & \Sigma_{\dot{x}\hat{x}} & \Sigma_{\dot{x}U} \\ \Sigma_{\hat{x}\dot{x}} & \Sigma_{\hat{x}\hat{x}} & \Sigma_{\hat{x}U} \\ \Sigma_{U\dot{x}} & \Sigma_{U\hat{x}} & \Sigma_{UU} \end{bmatrix}$$

Where K is the number of Gaussian components, the scalar π_k represents the weight associated to mixture component k (indicating the component's overall contribution to the distribution) and $\sum_{k=1}^{K} \pi_k = 1$. The parameters μ_k and Σ_k are the mean and covariance of the normal distribution k. The total set of parameters of the GMM is $\theta = \{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$.

The following section details the model selection, akin to finding the number of mixture components K, and parameter fitting, finding the values of θ.

1) Model selection & Parameter learning: The trajectories were segmented based on whether they are either on or off the table and then on their direction. This step was necessary since the optimisation employs EM which only guarantees local maximisation of the likelihood function. It is difficult to find the global optimum when starting the learning process from the whole data set in one go. For each segmented data set (one for trajectories off the table, and 4 for trajectories on the table), the Bayesian Information Criterion (BIC) was used to find the optimal number of mixture components and five sets of parameters were learned.

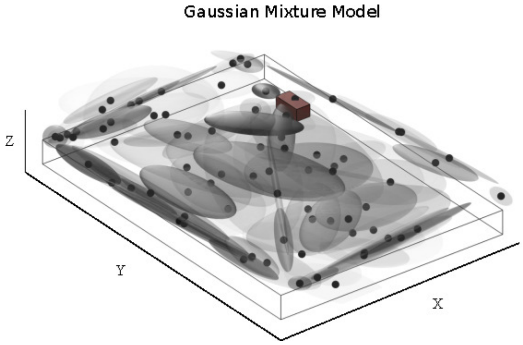

Fig. 4: The resulting GMM for the table, a total of 67 Gaussian mixture components are present. We note the many overlapping Gaussians: this results from the level of uncertainty over the different choices taken. For example, humans follow along the edge of the table in different directions and might leave the edge once they are confident with respect to their location.

The parameters from each set (mean and covariance) were combined and served as an initialisation when retraining over the whole data set which resulted in the final model. A total of 83 Gaussian functions were used in the final model, 67 for trajectories on the table and 15 for those in the air. In figure 4 we illustrate the model learned from human demonstrations where we plot the 3 dimensional slice (the position) of the 7 dimensional GMM to give a sense of the size of the model.

C. Control

To get a control output from a GMM we condition on the most likely position and uncertainty and the result is a new distribution over direction. The output is the expected value of the conditional (see equation 8 below).

$$\dot{x} = \mathbb{E}\{\mathcal{P}(\dot{x}|\hat{x}, U)\} = \sum_{k=1}^{K} \pi^k_{\dot{x}|\hat{x},U} \cdot \mu^k_{\dot{x}|\hat{x},U} \qquad (8)$$

The problem with this expectation approach, also know as Gaussian Mixture Regression (GMR), is that it averages out opposing directions or strategies and may leave a net velocity of zero. One possibility would be to sample from the conditional, however this can lead to non-smooth behaviour and flipping back and forth between modes resulting in no displacement. To maintain consistency between the choices and avoid random switching we perform a weighted expectation on the means so that directions (modes) similar to the current direction of the end-effector receive a higher weight than opposing directions. For every mixture component k, a weight α_k is computed based on the distance between the current direction and itself. If the current direction agrees with the mode then the weight remains unchanged but if it is in disagreement a lower weight is calculated according to the equation below.

$$\alpha_k(\dot{x}) = \pi^k_{\dot{x}|\hat{x},U} \cdot \exp(-\cos^{-1}(< \dot{x}, \mu^k_{\dot{x}|\hat{x},U} >)) \qquad (9)$$

GMR is then performed with the normalised weights $\boldsymbol{\alpha}$

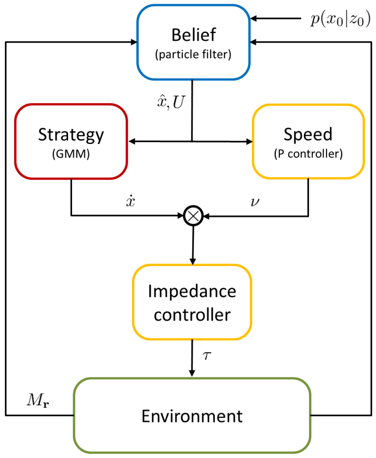

Fig. 5: Overview of the decision loop. At the top given an initial belief $p(x_0|z_0)$ of the location of the end-effector a strategy is chosen (initially through sampling the conditional) and based on the believed distance to the goal a speed is applied to the given direction. This velocity is passed onwards to a low level impedance controller which sends out the required torques. The resulting sensation, encoded through the Multinomial distribution over the environment features, and actual displacement are sent back to update the belief.

instead of $\boldsymbol{\pi}$, the initial weight obtained when conditioning.

$$\dot{x} = \mathbb{E}_{\alpha}\{\mathcal{P}(\dot{x}|\hat{x}, U)\} = \sum_{k=1}^{K} \alpha_k(\dot{x})\, \mu^k_{\dot{x}|\hat{x},u} \qquad (10)$$

The final output of equation 10 gives the desired direction ($\dot{x}$ is re-normalised). In the case when the mode suddenly disappears (because of sudden change of the level of uncertainty caused by the appearance or disappearance of a feature) another present mode is selected at random For instance, when the robot has reached a corner, the level of uncertainty for this feature drops to zero. A new mode, and hence new direction of motion, will then be computed. However this is not enough to be able to safely control the robot. One needs to control the amplitude of the velocity and ensure compliant control of the end-effector when in contact with the table. This behaviour is not learned here, as this is specific to the embodiment of the robot and unrelated to the search strategy. The amplitude of the velocity is computed by a proportional controller based on the believed distance to the goal.

$$\nu = \max(\min(\beta_1, K_p(x_{\mathrm{g}} - \hat{x}), \beta_2) \qquad (11)$$

where the β's are lower and upper amplitude limits, x_g is the position of the goal, and K_p the proportional gain which was tuned through trials.

As mentioned previously, the other important aspect when having the robot duplicate the search strategies is compliance. As a result of the uncertainty, collisions with the environment occur. To avoid risks of breaking the table or the robot sensors we have at the lowest level an impedance controller which outputs appropriate joint torques τ. The overall control loop is depicted in figure 5.

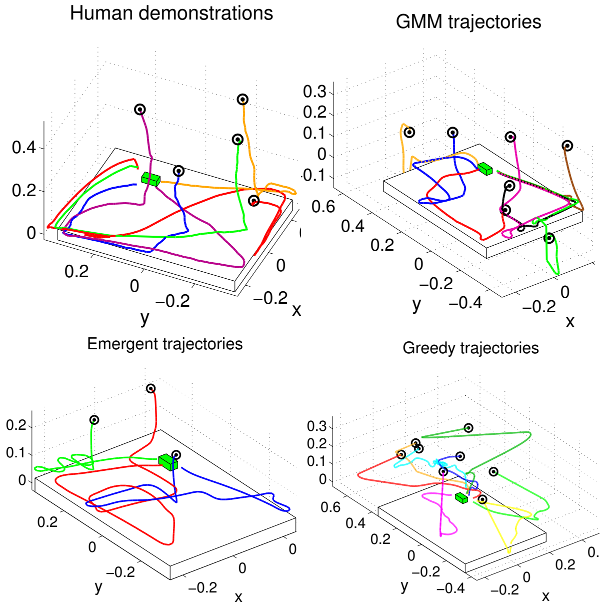

Fig. 6: Illustration of trajectories. *Top left:* 5 sample trajectories from the human volunteers. *Top right:* 6 sample trajectories generated from the learned model and controller. The red and orange trajectories are risk-prone since they are either not fully localised (orange) or take a long straight shot toward the goal through featureless space (red). On the other hand the pink and green trajectories stay close to features until as close as possible to the goal. *Lower left:* 3 emergent strategies not witnessed in training data due to the combination of multiple strategies. The blue trajectory is similar to the inverse of the purple trajectory in the top left figure, however it goes in opposite direction. *Lower right:* 6 trajectories from the greedy controller, non-smooth and abrupt. The scale is in meters.

VI. EXPERIMENTAL RESULTS

We evaluate our system by firstly comparing search roll outs against those of the human demonstrators. We make a qualitative analysis of the modes present in the GMM. We contrast the performance, with respect to the distance taken to reach the goal and how the uncertainty decreases over time for three controllers (greey, GMM and hybrid). Finally, we test the robustness of the system with respect to false beliefs.

A. Human & GMM search trajectories

We visually compare the trajectories gathered from the human volunteers with those of the learned controller. We notice that humans like to play safely, meaning that they remain as close as possible to informative features such as the edges. Once close to the goal they go straight towards it. Figure 6 contrasts the trajectories of the human's hand (top left) with those generated by our GMM controller (top right). Starting points were drawn from a uniform distribution over the table and the colour coding is to better differentiate the different trajectories in each sub-figure. The generated trajectories from the GMM model are similar to the training data provided by the human demonstrators, as one would expect.

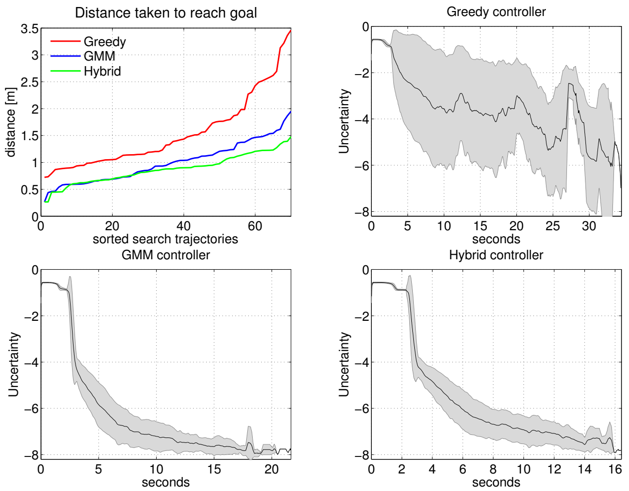

Fig. 7: *Top Left:* Plot of distance taken to reach to goal for all three controllers. The x-axis values correspond to a specific roll out, whilst the y-axsi values, are the distances taken to reach the goal. The trajectories are sorted in ascending order. The greedy controller by far takes the most time to reach the goal as oppose to both the GMM and hybrid. Using the variance (uncertainty) at the beginning plays a vital part in the performance of the controllers. The hybrid goes even faster than the GMM since once localized it goes straight to the goal. *Other three plots:* Level of uncertainty with variance (gray shaded area) decreasing over time for greedy, GMM and hybrid controllers. The decrease in uncertainty of the GMM and hybrid controllers is much more rapid than the greedy one. This reflects the fact that as humans we tend to play safe and avoid taking risks as opposed to the greedy controller. For the three controllers a total of a 70 trials were gathered for this analysis.

For both the human and GMM trajectories they all start by going downwards until a contact with the table is made. Then proceed to an edge and follow it until as close as possible to the goal (risk-averse). Other trajectories (orange in top left sub-figure) once localised go straight to the goal through a featureless space where no edges or corners are present (risk-prone). However, this does not hold true for all generated trajectories (lower left sub-figure). This is due to the way we perform the control. A trajectory is generated from a mixture of strategies which can lead to the emergence of previously unseen behaviour and the zig-zagging behaviour of the green trajectory is due to unstable attractors. We also note, through observing resulting generated trajectories from the GMM model, that not all strategies demonstrated are encoded in the GMM. For example, there is an instance when a demonstrator cuts across the table (see red trajectory in the top left plot of figure 6). There were not many examples of such behaviour, making it statistically insignificant with respect to the GMM which in the EM learning stage did not attribute a Gaussian function to represent it. However since the search strategy of the robot is composed from a mixture of strategies it is possible that new trajectories emerge which are similar to these one-off demonstrations.

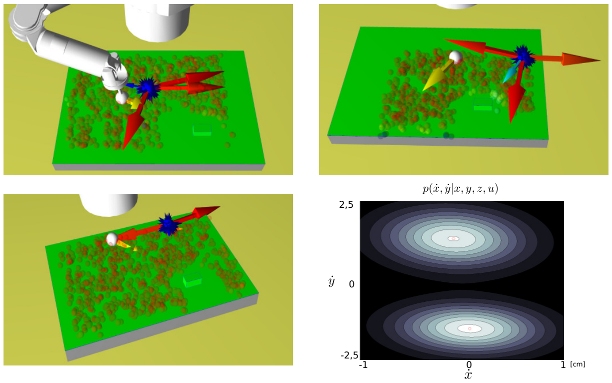

Fig. 8: Illustration of three different types of modes present during the execution of the task where the robot is being controlled by the learned model. The white ball represents the actual position of the robot's end-effector. The blue ball represents the believed position of the robot's end-effector and the robot is acting according to it. Arrows of the blue ball represent modes, colours encode the modes weights given by the priors π_k after conditioning (but not re-weighted as previously described). The spectrum ranges from red (high weight) to blue (low weight). *Top left:* Three modes are present, but two agree with each other. *Top right:* Three modes are again present indicating appropriate ways to reduce the uncertainty. *Lower left:* Two modes in opposing directions, no flipping behaviour between modes occurs since preference is given to the modes pointing in the same direction as the robot's current trajectory. *Lower right:* GMM modes when conditioned on the state represented in the lower left figure. The two modes represent the possible directions (un-normalised).

B. *Qualitative analysis of modes*

We next illustrate some of the modes (action choices) present during simulation and evaluated their plausibility. Figure 8 shows that multiple decision points have been correctly embedded in the GMM model. All directions (red arrows) indicate directions that reduce the level of uncertainty.

C. *Greedy vs GMM vs Hybrid controller*

We evaluated the performance of a greedy controller, which takes the most likely position $\hat{x}$ and goes straight towards the goal, as opposed to a controller solely learned from human demonstrations and a hybrid controller which uses the GMM controller until a minimum uncertainty threshold is reached before switching to the greedy controller. We performed 70 runs in each case and evaluated the uncertainty and distance taken to reach the goal. The results are illustrated in figure 7 and six trajectories of the greedy controller are depicted in the lower right of figure 6. The results confirm that the GMM controller decreases uncertainty quadratically as opposed to the greedy method where the uncertainty does not seem to decrease in a consistent fashion. The trajectories of the greedy controller are also non-smooth, abrupt and unnatural. The Hybrid controller takes even less time/distance to reach the goal since it does not seek to stay close to informative features once localized and goes straight

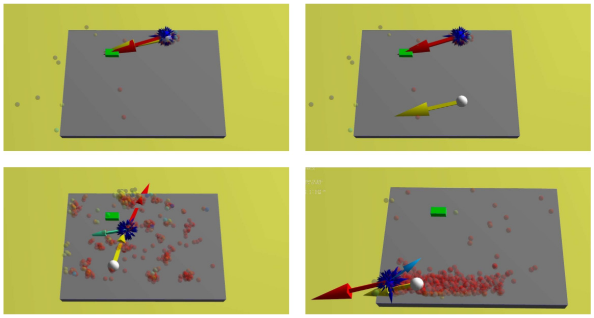

Fig. 9: Depiction of the robustness with respect to false beliefs. *Top left:* both the believed and actual position of the end-effector coincide with each other and most of the probability mass $p(x|z)$ lies on top of them. *Top right:* the actual end-effector's position, white ball, has been teleported to another position making the believed position, blue ball, inaccurate. *Bottom left:* all the particles which were at the end-effector's believed position were resampled to feasible areas which yield similar sensing to the actual position. *Bottom right:* the overall search process continues until the goal is reached.

towards the goal. The GMM on the contrary reflects the risk-averse behavior of humans. When we don't have any visual feedback our behavior is very different and this is not taken into account by the Hybrid controller (in the final stage of the search) which has no such concept of prudence.

D. Robustness

We now turn to the evaluation of robustness of the learned model with respect to false beliefs. False belief, in our experiment, corresponds to situations where the ”believed” location of the end-effector is far from the true position. To simulate this situation, once the robot had localised itself (that is, the uncertainty level is close to zero) and was heading towards the goal, it was teleported to one of four possible locations (middle of the table in the air, near top right, bottom right and bottom left corners of the table). The system recovered well from such failure, as the end-effector moved towards the goal but failed to reach it as it had expected, the probability density $p(x|z)$ redistributes itself across the feasible locations in the environment, see figure 9. This is made possible since we keep 1% of the particles distributed at random across the environment at all times. A total of 50 runs were performed with the teleportation mentioned above and in the runs the goal was found.

VII. CONCLUSION

In this work we have shown a novel approach in teaching a robot to act in a partially observable environment. Through having human volunteers demonstrate the task of finding an object on a table, we recorded both the inferred believed position of their hand and associated action (normalised velocity). A generative model mapping the believed end-effector position to actions was learned, encapsulating this relationship. As speculated and observed, multiple strategies are present given a specific belief reflecting the fact that humans act differently given the same situation. Some trajectories generated by the model were similar to those of the human demonstrations while others emerged through the combination of multiple strategies. When compared to a greedy controller humans prefer to first reduce uncertainty and then minimise risk. The model is able to handle false beliefs and environmental perturbations. Future research will focus on adding another probability density function to represent the believed location of the goal. In this way the goal no longer has to be fixed and this situation makes for a more interesting problem, where interacting probability density functions need to be addressed.

ACKNOWLEDGMENT

This research was supported by the European project, Flexible Skill Acquisition and Intuitive Robot Tasking for Mobile Manipulation in the Real World (First-MM), in Robotic Research.

REFERENCES

[1] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, “Planning and acting in partially observable stochastic domains,” *Artificial intelligence*, vol. 101, pp. 99–134, 1998.

[2] T. Smith, “Probabilistic planning for robotic exploration,” Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 2007.

[3] J. Pineau, G. Gordon, and S. Thrun, “Point-based value iteration: an anytime algorithm for pomdps,” in *IJCAI*, 2003, pp. 1025–1030.

[4] C. Bake, J. Tenenbaum, and R. Saxe, “Bayesian theory of mind: Modeling joint belief-desire attribution,” *Journal of Cognitive Science*, 2011.

[5] H. Richardson, C. Bake, J. Tenenbaum, and R. Saxe, “The development of joint belief-desire inferences,” *Cognitive Science Socitety*, 2012.

[6] S. M. Khansari-Zadeh and A. Billard, “Learning stable non-linear dynamical systems with gaussian mixture models,” *IEEE Transaction on Robotics*, 2011.

[7] A. P. Shon, K. Grochow, and R. P. N. Rao, “Robotic imitation from human motion capture using gaussian processes,” in *Humanoids*, 2005.

[8] H. et. al, “Learning and generalization of motor skills by learning from demonstration,” *ICRA*, pp. 763–768, 2009.

[9] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, “Robot programming by demonstration,” in *Springer Handbook of Robotics*, 2008, pp. 1371–1394.

[10] C. Bake, J. Tenenbaum, and R. Saxe, “Bayesian models of human action understanding,” *NIPS*, 2006.

[11] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998, a Bradford Book.

[12] W. B. Sebastian Thrun and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[13] H. Kurniawati, D. Hsu, and W. S. Lee, “Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces,” in *In Proc. Robotics: Science and Systems*, 2008.

[14] G. Shani, J. Pineau, and R. Kaplow, “A survey of point-based pomdp solvers.” *Autonomous Agents and Multi-Agent Systems*, vol. 27, no. 1, pp. 1–51, 2013.

[15] N. Roy, J. Pineau, and S. Thrun, “Spoken dialogue management using probabilistic reasoning,” *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, 2000.

[16] S. Thrun, “Monte carlo pomdps,” Carnegie Mellon University, Tech. Rep., 1999.

[17] K. Hsiao, L. Kaelbling, and T. Lozano-Perez, “Task-driven tactile exploration,” in *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, June 2010.

[18] P. Hebert, “Action inference: The next best touch,” *RSS*, 2012.

[19] M. S. A. et. al, “A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking,” *IEEE Transactions on Signal Processing*, vol. 50, pp. 174–188, 2002.

[20] M. Huber, T. Bailey, H. Durrant-Whyte, and U. Hanebeck, “On entropy approximation for gaussian mixture random vectors,” in *Multisensor Fusion and Integration*, 2008, pp. 181–188.