Robust Policy Search for an Agile Ground Vehicle Under Perception Uncertainty

Shahriar Sefati[1], Subhransu Mishra[1], Matthew Sheckells[2], Kapil D. Katyal[3], Jin Bai[1], Gregory D. Hager[4], Marin Kobilarov[1]

***Abstract*— Learning robust policies for robotic systems operating in presence of uncertainty is a challenging task. For safe navigation, in addition to the natural stochasticity of the environment and vehicle dynamics, the perception uncertainty associated with dynamic entities, e.g. pedestrians, must be accounted for during motion planning. To this end, we construct an algorithm with built-in robustness to uncertainty by directly minimizing an upper confidence bound on the expected cost of trajectories instead of employing a standard approach based on minimizing the expected cost itself. Perception uncertainty is incorporated into the policy search framework by predicting each pedestrian's intent belief and propagating their state distribution in time using *closed-loop* goal-directed dynamics. We train the policy in simulation and show that it could be transferred to an agile ground vehicle for successful autonomous robot navigation in presence of pedestrians with perception uncertainty. We further show the superior performance of this policy over a policy that does not consider pedestrian intent and perception uncertainty.**

Fig. 1. JHU all-terrain agile ground vehicle.

I. Introduction

Planning safe trajectories is of great importance for autonomous vehicles. The uncertainty introduced by the inherent stochastic nature of the operating environment and the stochastic dynamics of the vehicle and other participant entities makes this task challenging. In addition, the uncertainty associated with sensory observations from perception imposes further challenges to this task. Failure to account for such uncertainty in planning may lead to unsafe behavior.

In robotics tasks, uncertainty could arise from different sources such as robot localization [1], [2], control [3], [4] and perception [5], [6], [7]. In this work, we mainly focus on the perception uncertainty. Previous works have tackled the problem of planning under perception uncertainty from different perspectives. Jha et al. proposed a probabilistic extension of temporal logic that can be used to specify correctness requirements in presence of perception uncertainty [5]. Xu et al. used Gaussian propagation of uncertainty along predicted trajectories for traffic participants to achieve safer trajectories in autonomous driving [6]. Jasour et al. used risk contours map that contain the risk information of different regions in uncertain environments [7]. Sampling-based methods such as rapidly-exploring random belief trees have also been used for motion planning under uncertainty [8], [9].

In this work, we approach this problem through policy search, which is a method for computing the optimal control parameters of a robotic system operating in an unknown, uncertain environment. Consider a ground vehicle (e.g. the one shown in Fig. 1) autonomously performing a navigation task with uncertainty induced in the environment by the presence of pedestrians. For safe navigation, we incorporate the sensory measurement uncertainties associated with the perceived pedestrian states into the policy search problem. This is done by predicting the participant agents' intents and rolling out their trajectories with Gaussian uncertainty propagation in time using *closed-loop* goal-directed dynamics.

A multitude of update strategies exists for the policy search problem: 1) policy gradient methods like PEPG, DDPG [10], and REINFORCE [11], 2) gradient-free methods, such as Reward-weighted Regression (RwR) [12] and CMA-ES, 3) information-theoretic methods like REPS and TRPO, 4) actor-critic methods like A3C and TRPO [13], [14], [15], 5) methods that minimize an upper confidence bound on the expected cost of trajectories, e.g. High Confidence Policy Improvement [16] and Probably-Approximately-Correct Robust Policy Search (PROPS) [17]. The last update strategy is particularly useful because it provides a bound on the expected performance of the policy which it computes, where the bound can be regarded as a certificate for guaranteed future performance.

To leverage the guaranteed future performance of the

[1] Laboratory for Computational Sensing and Robotics, Johns Hopkins University, Baltimore, MD, USA. sefati|smishra9|baijin|marin@jhu.edu

[2]Space Exploration Technologies Corp. (SpaceX), Hawthorne, CA, USA. msheckells@gmail.com

[3]Johns Hopkins University, Applied Physics Lab, Laurel, MD USA. Kapil.Katyal@jhuapl.edu

[4]Malone Center for Engineering in Healthcare, Johns Hopkins University, Baltimore, MD USA. hager@cs.jhu.edu

policy, we employ Actor-Critic PROPS (AC-PROPS) [18] that minimizes an upper confidence probably approximately correct (PAC) bound on the negative advantage of a control policy at each policy update iteration. The algorithm estimates these unknown advantages using Generalized Advantage Estimation (GAE) [19]. This approach results in an advantage estimator that has a tunable bias-variance trade-off. This algorithm is categorized as actor-critic since the advantage estimation makes use of a learned value function in order to update the policy.

Problem statement: Given the perceived state of the world x, our objective is to learn the parameters of a control policy π for the vehicle which minimizes the upper confidence bound on the expected value of a user-defined cost function J, that encodes desired performance metrics such as mission progress, obstacle avoidance, and safety. We assume the state of the vehicle and other entities are perceived through the on-board sensors and an initial belief over the future intent of dynamic entities is available which gets updated as new measurements z become available from the perception module. We train an AC-PROPS policy [18] in simulation using a high quality stochastic dynamics model [20] of a $1/5$-scale agile ground vehicle (Fig. 1) and show that the policy could be transferred to the vehicle for successful autonomous robot navigation around a track in presence of pedestrians with perception uncertainty.

II. Methods

A. Stochastic Policy Search

Consider a finite horizon Markov Decision Process (MDP) defined by the combined state $x = (x^0, x^1, \ldots, x^{n_p})$, combined control inputs $u = (u^0, u^1, \ldots, u^{n_p})$. Each agent with index i has initial state probability density $p_0(x^i)$ and transition density $p(x^i_{k+1}|x^i_k, u^i_k)$, where superscript index $i = 0$ corresponds to the vehicle and $i = 1, \ldots, n_p$ denote the pedestrians (*non-player* characters) and subscript k is the time step. We assume that the vehicle control policy $u^0 = \pi(x; \xi)$ is parameterized by a vector ξ (described in section II-D). Each pedestrian is modeled using a goal-driven controller $u^i = \phi^i(x^i, x^i_g)$, where x^i_g is an estimated goal-state (described in section II-B). The robot state-control trajectory over N time-segments is denoted by $\tau \triangleq (x^0_0, u^0_0, ..., u^0_{N-1}, x^0_N)$, the i-th pedestrian trajectory is $\eta^i \triangleq (x^i_0, ..., x^i_N)$ with all pedestrian trajectories denoted by $\eta \triangleq (\eta^1, ..., \eta^{n_p})$, and have densities

$$p(\tau|\xi) = p_0(x^0_0) \prod_{k=0}^{N-1} p(x^0_{k+1}|x^0_k, u^0_k)\pi(u_k|x_k; \xi),$$

$$p(\eta^i) = p_0(x^i_0) \prod_{k=0}^{N-1} p(x^i_{k+1}|x^i_k, \phi^i(x^i, x^i_g)),$$

for $i = 1, \ldots, n_p$. The objective is to find the optimal set of vehicle policy parameters ξ^* such that:

$$\xi^* = \arg\min_{\xi} \mathbb{E}_{\tau\sim p(\cdot|\xi), \eta^i\sim p(\cdot)}[J(\tau, \eta)], \tag{1}$$

where $J(\tau, \eta) = -\sum_{t=0}^{N} r(x_t, u_t)$ is a cost function encoding the desired performance metric. Note that the pedestrian dynamics is assumed to be independent of the robot, while the robot control policy π depends on the pedestrian states. Instead of directly searching for the optimal ξ to solve (1) a common strategy is to iteratively construct a surrogate stochastic model $\pi(\xi|\nu)$ with hyper-parameters $\nu \in \mathcal{V}$. The model, thus, induces a joint density $p(\tau, \xi|\nu) = p(\tau|\xi)\pi(\xi|\nu)$ that encodes natural stochasticity $p(\tau|\xi)$ and artificial control-exploration stochasticity $\pi(\xi|\nu)$. In previous work [17], a robust policy search methodology was developed that was based on the PAC bounds on the performance of a stochastic policy. This algorithm directly minimizes an upper confidence bound on the expected cost of trajectories instead of employing a standard approach based on the expected cost itself. Consequently, it has built-in robustness to uncertainty, as the bound can be regarded as a certificate for guaranteed future performance.

Given a prior distribution $\pi(\cdot|\nu_0)$ on control parameters and M executions based on the prior, the expected cost of the new policy $\pi(\cdot|\nu)$ based on episode-based policy sampling is given by:

$$\mathcal{J}(\nu) \triangleq \mathbb{E}_{\tau,\xi\sim p(\cdot|\nu)}[J(\tau, \eta)] = \mathbb{E}_{\tau,\xi\sim p(\cdot|\nu_0)}\left[J(\tau, \eta)\frac{\pi(\xi|\nu)}{\pi(\xi|\nu_0)}\right].$$

For step-based policy sampling, this learning objective can alternatively be written in terms of step-wise advantages [18]:

$$\mathcal{J}(\nu) \triangleq \mathbb{E}_{x,\xi\sim p(x|\xi)\rho(\xi|\nu)}\left[-A^{\nu_0}(x, u)\frac{\rho(\xi|\nu)}{\rho(\xi|\nu_0)}\right], \tag{2}$$

where $A^{\nu}(x_t, u_t) = Q^{\nu}(x_t, u_t) - V^{\nu}(x_t)$ is the advantage function, with $Q(\cdot)$ and $V(\cdot)$ defining the state-action and state value functions, respectively.

$\mathcal{J}(\nu)$ can be approximated empirically using samples $\xi_{t,j} \sim \rho(\xi|\nu)$ and $x_{t,j} \sim p(x|\xi_{t,j})$. The change of measure likelihood ratio $\frac{\rho(\xi_{t,j}|\nu)}{\rho(\xi_{t,j}|\nu_0)}$, however, can be unbounded [21]. A robust estimation technique [22] could instead be employed to deal with the unboundedness of the policy adaptation. In addition, to obtain sharp bounds it is useful to employ samples over multiple iterations of the iterative stochastic policy optimization algorithm, i.e. from policies $\nu_0, \nu_1, \ldots, \nu_{L-1}$ computed in previous iterations. The cost (2) of executing ν can then be equivalently expressed as:

$$\mathcal{J}(\nu) \equiv \frac{1}{L}\sum_{i=0}^{L-1} \mathbb{E}_{z\sim p(\cdot|\nu_i)}\ell_i(z, \nu),$$

where $z = (\tau, \eta, \xi)$ and $\ell_i(z, \nu) \triangleq J(\tau, \eta)\frac{\pi(\xi|\nu)}{\pi(\xi|\nu_i)}$. This can be approximated by the empirical mean $\mathcal{J}(\nu) \approx \frac{1}{ML}\sum_{i=0}^{L-1}\sum_{j=1}^{M}[\ell_i(z_{ij,\nu})]$. A more robust estimate [22] is given by:

$$\widehat{\mathcal{J}}_{\alpha}(\nu) \triangleq \frac{1}{\alpha LM}\sum_{i=0}^{L-1}\sum_{j=1}^{M}\psi\left(\alpha\ell_i(z_{ij}, \nu)\right), \tag{3}$$

where $\alpha > 0$ and $\psi(x) = \log(1 + x + \frac{1}{2}x^2)$. As outlined in [22], [17], with probability $1 - \delta$ the expected cost of

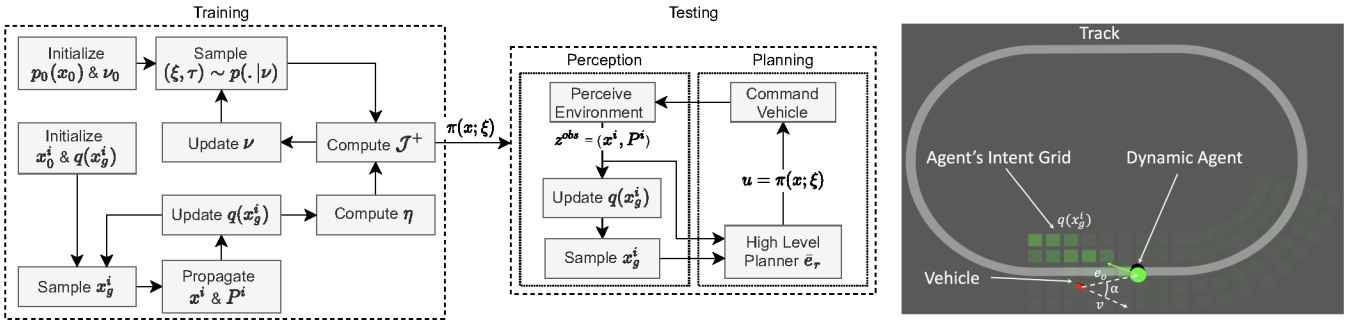

Fig. 2. Block diagram of steps during policy training (left) policy testing (middle). The track navigation task in simulation (right).

executing a stochastic policy with parameters $\xi \sim \pi(\cdot|\nu)$ is bounded according to:

$$\mathcal{J}(\nu) \leq \inf_{\alpha>0} \mathcal{J}_\alpha^+(\nu), \tag{4}$$

$$\mathcal{J}_\alpha^+(\nu) = \widehat{\mathcal{J}}_\alpha(\nu) + \frac{\alpha}{2L}\sum_{i=0}^{L-1} b_i^2 e^{D_2(\pi(\cdot|\nu)||\pi(\cdot|\nu_i))} + \frac{\alpha^{-1}}{LM}\log\frac{1}{\delta}$$

computed after L iterations with M samples $z_{i1}, ..., z_{iM} \sim p(\cdot|\nu_i)$, where $D_\beta(p||q)$ denotes the Renyii divergence between p and q. The constants b_i are such that $0 \leq J(\tau, \eta) \leq b_i$ at each iteration. Consequently, a new policy ν_{i+1} is computed using observed costs from previous iterations by optimizing the bound (4) jointly over α and the policy ν:

$$\nu^* = \arg\min_{\nu} \min_{\alpha>0} \mathcal{J}_\alpha^+(\nu). \tag{5}$$

The block diagram of the steps taken during training and testing of the policy is shown in Fig. 2. As observed in the left figure, during training, the policy hyper-parameter ν_0, as well as the robot and pedestrians states are randomly initialized. Given the observations and predicted trajectory of pedestrians (described in Sections II-B and II-C), ν is iteratively updated until the cost function falls below a desired threshold. During testing, given the perception observations and the pedestrian intent predictions, robot controls will be generated from the policy $\pi(x;\xi)$.

B. Closed-loop Pedestrian Uncertainty Propagation

We assume that each pedestrian's state has a Gaussian distribution $x_k^i \sim \mathcal{N}(\cdot|\hat{x}_k^i, P_k^i)$ for $i = 1, \ldots, n_p$ which, once a pedestrian goal x_g^i is selected, can be propagated in time using the pedestrian known *closed-loop* goal-directed dynamics for N time segments to form $\eta^i = (x_0^i, ..., x_{N-1}^i)$. Let the pedestrian dynamics and controller be:

$$\dot{x}^i = f^i(x^i, u^i), \qquad u^i = \phi^i(x^i, x_g^i), \tag{6}$$

where x_g^i is the i^{th} pedestrian's goal state. The closed-loop dynamics $\bar{f}^i$ could then be written as:

$$\dot{x}^i = f^i(x^i, \phi^i(x^i, x_g^i)) \triangleq \bar{f}^i(x^i, x_g^i). \tag{7}$$

More specifically, we employ a double-integrator model with state $x^i = (p^i, v^i)$ consisting of position $p^i \in \mathbb{R}^2$ and velocity $v^i \in \mathbb{R}^2$ with dynamics:

$$\dot{p}^i = v^i,$$
$$\dot{v}^i = u^i + w^i,$$

where $u = \phi^i(x, x_g) = K(x - x_g)$ with $K = [-k_p I_2, -k_v I_2]$ for some chosen gains $k_p, k_v > 0$, and $w^i(t)$ being zero-mean with variance $E[w^i(t)w^i(\tau)^T] = Q_c\delta(t-\tau)$. The closed-loop dynamics could be written in matrix form as:

$$\dot{x}^i = (F + GK)x^i - GKx_g^i + Lw^i. \tag{8}$$

The state propagation over time segment $[t_{k-1}, t_k]$ is then:

$$\hat{x}_k^i = \Phi_{k-1}\hat{x}_{k-1}^i - \Gamma_{k-1}Kx_g^i, \tag{9}$$
$$P_k^i = \Phi_{k-1}P_{k-1}^i\Phi_{k-1}^T + Q_{k-1}, \tag{10}$$

where Γ_{k-1}, Q_{k-1} are defined as:

$$\Gamma_{k-1} = \int_{t_{k-1}}^{t_k} \Phi(t_k, \tau)G d\tau,$$
$$Q_{k-1} = \int_{t_{k-1}}^{t_k} \Phi(t_k, \tau)LQ_cL^T\Phi^T(t_k, \tau)d\tau,$$

where the state-transition matrix $\Phi_{k-1} = e^{\Delta t(F+GK)}$ for simplicity could be approximated by:

$$\Phi_{k-1} \approx I + \Delta t(F + GK). \tag{11}$$

C. Intent-aware Pedestrian Prediction

To propagate the pedestrian state uncertainty using *closed-loop* dynamics, a goal state is required. While various approaches exist for human motion trajectory prediction [23], we adapt the Bayesian method proposed in [24] and [25], that estimates the probability of a desired goal, x_g^i, on a grid-like representation (Fig. 2) for the i^{th} pedestrian based on history of past observations, z_i^{obs}:

$$q(x_g^i|z_i^{obs}) = \frac{q(x_g^i)q(z_i^{obs}|x_g^i)}{q(z_i^{obs})}, \tag{12}$$

where $q(x_g^i|z_i^{obs})$ is the posterior probability of pedestrian goal, given an observation history z_i^{obs}. $q(x_g^i)$ is the prior probability of i^{th} pedestrian's goal. $q(z_i^{obs}|x_g^i)$ is the likelihood probability of observing z_i^{obs} given x_g^i and modelled as a Gibbs measure:

$$q(z_i^{obs}|x_g^i) = \frac{1}{\Psi(\beta)}\exp(-\beta E(z_i^{obs}|x_g^i)), \tag{13}$$

where $E(z_i^{obs}|x_g^i)$ is an energy function that is set equal to distance between the observed trajectory and the shortest trajectory to the goal. β adjusts the landscape of the resulting probability distribution and $\Psi(\beta)$ is a normalizing constant.

In practice, for the i^{th} pedestrian, a probability distribution $q^i(x_g^i)$ on the set of possible goals in the grid representation is provided to the planner by the perception module. Consequently, the goal state for the i^{th} pedestrian could be generated by sampling $x_g^i \sim q^i(.)$.

D. Vehicle Control Policy

We express the state of the autonomous vehicle with respect to a curvilinear coordinate system, with the reference curvature of the coordinate system following the road or path centerline. The state of the vehicle is defined as $x = (s, e_r, e_\theta, v, a, \delta_s) \in \mathbb{R}^6$, where s is the arc length along the reference path, e_r is the lateral offset from the path, e_θ is the angular offset from the path tangent at s, and v is the forward body-velocity, a is the forward body-acceleration, and δ_s is the steering angle. The control inputs to the system consist of the jerk $u_1 \in \mathbb{R}$ and steering angle rate $u_2 \in \mathbb{R}$. Given $\kappa(s)$, the curvature of the path at s, and L, the vehicle's wheelbase, typical bicycle dynamics expressed using path coordinates are derived as:

$$\begin{bmatrix} \dot{s} \\ \dot{e_r} \\ \dot{e_\theta} \\ \dot{v} \\ \dot{a} \\ \dot{\delta_s} \end{bmatrix} = \begin{bmatrix} \frac{vcos(e_\theta)}{1-\kappa(s)e_r} \\ vsin(e_\theta) \\ v\frac{tan\delta_s}{L} - \kappa(s)\dot{s} \\ a \\ u_1 \\ u_2 \end{bmatrix}. \tag{14}$$

We use a Lyapunov stable controller that achieves a desired track offset and longitudinal velocity in a decoupled manner, with relatively few learnable parameters [20]. Taking into account the uncertainty associated with perception detections, a higher level planner commands a track offset to the lateral controller to avoid detected dynamic entities. The controller parameters are: lateral control gains k_{r_p}, kr_d, k_{z_θ}, velocity control parameters k_{v_p}, k_{v_d}, a_{latmax}, and k_{det} and k_{shift} for detecting pedestrians and shifting the lateral offset to navigate around them.

For more intelligent obstacle avoidance, we introduce additional learnable parameters to the controller that incorporate perception uncertainty and pedestrian intent distribution in the planning task using intuitive geometrical interpretations. In previous work [20], if a static obstacle was detected within some radius of the vehicle, denoted k_{det}, then the desired track offset generated by the high level planner was shifted by a fixed value, k_{shift}. This methodology, however, does not take into account the uncertainty associated with the detections. To do so, we define the lateral shift as $d_{shift} = k_{shift} \cdot S_{det}$ that incorporates the uncertainty from perception, where S_{det} is the largest principal axis of the covariance ellipse P in (10) for the pedestrian closest to the vehicle. On the contrary to using only a fixed offset (k_{shift}), such a formulation will adapt the overall lateral shift according to the observed uncertainty from perception.

The high level planner would naturally generate track offsets in the direction that the vehicle is pointing relative to the pedestrians. For instance, if the robot is pointing to the left of a pedestrian, then the desired track offset is shifted to the left. However, taking into account the pedestrian intent, the high level planner could more intelligently reason about the direction to swerve around the pedestrian that would avoid potential future interference of the vehicle with the pedestrian's intended path.

Given the state and dynamics limitations of the vehicle, a sudden change of direction may be physically impossible or require maneuvers with very sharp turns at high velocities. We, therefore, allow the policy to decide on taking sharp turns or not, based on a learnable parameter, k_{feas} that geometrically encodes the feasibility, and cost of executing such maneuvers, if necessary. More specifically, if the angle between the vehicle's heading and the line that connects the vehicle to the closest obstacle, α (Fig. 2), is larger than a learned angle threshold, k_{feas}, a change of direction would either be infeasible, too costly, or unnecessary. Consequently, the high level planner generates the desired lateral controller track offset $\bar{e}_r$ as:

$$\begin{aligned} \bar{e}_r &= e_{r_t} - e_{r_d}, \\ e_{r_d} &= e_{r_o} + dir\,(k_{shift} \cdot S_{det}), \end{aligned} \tag{15}$$

where e_{r_t} and e_{r_o} (computed using x^i from perception) are the vehicle's and the closest obstacle's lateral offset from the center of the track, respectively. dir is determined by the high level planner depending on the closest obstacle's sampled intent lateral offset and whether $\alpha \leq k_{feas}$, where $\alpha = cos^{-1}(\frac{e_o}{\|e_o\|} \cdot \frac{v}{\|v\|})$, e_o is the vector connecting the vehicle to the closest obstacle, and v is the vehicle heading (Fig. 2).

We define the policy search cost function as:

$$\begin{aligned} J(\tau) = \sum_{t=0}^{t_{f/dt}} [Ra_t^2 + Q_r e_{r_t}^2 + Q_v({}^{v_t}/_{v_{goal}} - 1)^2 \\ + |v_t| O(d_t) + Q_i q_t^c], \end{aligned} \tag{16}$$

where $(\cdot)_t$ indicates the state at a discrete time index t, a is the vehicle acceleration, v_{goal} is the goal velocity, R, Q_r, Q_v, $Q_i > 0$ are tuning weights, q_t^c is the pedestrian intent probability for the grid cell, c, that is the closest to the vehicle at time index t, d_t is the time step, and $O(d)$ is a cost that encourages obstacle avoidance [20]. Of note, the computed $J(\tau)$ in (16) is used to compute $\widehat{\mathcal{J}}_\alpha(\nu)$ in (3).

III. System Setup

A. Vehicle Hardware

The all-terrain $^1/_5$-scale ground vehicle used in our experiments is shown in Fig. 1. A heavily modified Redcat Racing Rampage XB-E serves as the base vehicle. The drive motor and steering servo have upgraded motor controller to enable control and retrieval of rotational position and velocity information over serial link. An ATmega2560 board is used as the low-level controller board to interface with the motor controllers and radio receiver. In addition, it also

receives control commands from on-board computer and relays wheel odometry information back. The computer subsystem consists of an Intel NUC CPU, Nvidia Jetson AGX Xavier, an ethernet switch and a wifi router. For sensing, in addition to the wheel odometry feedback, we have an Intel Realsense D455 RGB-D camera, a LORD Microstrain 3DM-GX4-25 IMU and a u-blox ZED-F9P RTK-GPS unit for obtaining global position and absolute heading of the vehicle.

B. Perception

The perception algorithms used can be primarily divided into two groups: vehicle localization and pedestrian detection.

Vehicle localization is performed by fusing wheel odometry, IMU and GPS position and heading by employing an extended Kalman filter (EKF). We use an open-source ROS implementation of the EKF named *robot_localization* [26].

The objective of the pedestrian detection algorithm is to detect and localize pedestrians in the environment as well as to compute the uncertainty of the associated detections. In particular, the perception module receives RGB and depth images from the camera system and produces a pose and covariance estimate of all pedestrians with respect to the a global coordinate frame. The pedestrian detection algorithm is based on the Gaussian version of the Yolov3 architecture [27], [28] which uses the Darknet-53 network as the underlying feature extractor. This network architecture was selected primarily due to its ability to produce accurate bounding box locations of detected objects with real time performance. The Gaussian version of the Yolov3 architecture outputs two parameters for each bounding box coordinate, a mean and standard deviation (Fig. 3). To determine the pose, we compute the center pixel of the bounding box for each detected pedestrian, i represented as u_i, v_i. This is followed by using the intrinsic camera calibration matrix to convert the center pixel to real world coordinates, x_i, y_i. We then use the coregistered depth image to provide the z_i coordinate corresponding to the depth of the pedestrian. To compute the covariance matrix associated with the pose, we add the standard deviation to the pixel coordinates, u_i, v_i, reproject this point to real world coordinates and subtract from the mean pose, x_i, y_i. This produces a standard deviation in real world coordinates that is subsequently used to populate the covariance matrix. Our final step is to perform a series of transformations to convert the pose and covariance of the pedestrian with respect to the global coordinate frame which is subsequently used by the planning algorithm.

IV. Experiments

Using collected robot motion trajectories, a high quality stochastic dynamics model was previously developed for the vehicle [20]. We leverage this dynamics model to train policies in simulation and transfer them to the physical system for navigating around an oval track with pedestrians present in the environment. The block diagram of the steps taken during training and testing of policies is shown in Fig. 2.

The simulation environment is shown in Fig. 2. The vehicle attempts to follow a 22 m $\times$ 14 m oval track in presence of simulated moving pedestrians with the associating grid representation of their intent belief. During training, in the beginning of each episode, a pedestrian is spawned 8 m in front of the vehicle with randomly sampled track offset, goal location, and initial simulated state covariance. Throughout the episode, the pedestrian moves with the controller outlined by (6) and its state (x^i) and uncertainty (P^i) gets propagated according to (9) and (10). An episode terminates if the vehicle collides with the pedestrian or if t_f seconds elapses. The vehicle state at the end of an episode is maintained as its initial state at the beginning of the next episode, so that the vehicle always remains in motion.

We train two control policies using step-based sampling in simulation: *policy 1* and *policy 2*. Both policies share the learnable parameters k_{v_p}, k_{v_d}, k_{r_p}, kr_d, k_{z_θ}, k_{det}, k_{shift}, a_{latmax}. *Policy 1* shifts the desired lateral offset by the fixed value of k_{shift} without any notion of the pedestrian intent and uncertainty [18]. *Policy 2* takes into account the pedestrian intent and uses k_{feas} and k_{shift} to shift the desired lateral offset by incorporating perception uncertainty according to (15). For *policy 2*, the belief representation of the intent ($q(g)$) for the pedestrians get updated by maintaining a history of the past ten observations (z^{obs}) of each pedestrian's state. A new goal is sampled from the belief distribution $x_g \sim q(g)$ whenever the distance between the updated belief and the previous belief in terms of KL divergence reaches an empirically determined threshold, γ.

We train each policy for 200 iterations, with 50 episodes (i.e. trajectory roll-outs) in each iteration, using a sliding window of 20 batches for each policy update. For the cost function, we set $v_{goal} = 3.5$ m/s, $t_f = 7$ s, $dt = 0.02$ s, $R = 10^{-3}$, $Q_r = 0.25$, $Q_v = 4$, $Q_i = 10$, $C_{low} = 800$, $C_{high} = 80$, $o_{low} = 0.5$ m, and $o_{high} = 1.0$ m. For the pedestrian controller and intent, we set $k_p = 0.2$, $k_v = 0.6$, $\beta = 0.3$, and $\gamma = 0.5$. The two trained policies are then tested in simulation during 100 episodes with pedestrians randomly spawned in the scene.

Finally, we transfer the policies trained in simulation to the JHU all-terrain $^1/_5$-scale agile ground vehicle for testing in an off-road environment. In this case, the vehicle navigates

Fig. 3. Pedestrian detection (bold lines) and uncertainty (narrow lines) bounding boxes determined by the Gaussian Yolov3 detector (left). Top-down view of the track (right).

around a similar but smaller oval track of size 18 m $\times$ 10 m. Pedestrians are detected from the perception module along with their uncertainties as outlined in section III-B.

V. Results and Discussion

Fig. 4 demonstrates the convergence of the parameters over time for the two learned policies. *Policy 1* (blue) learned larger lateral controller gains (k_{r_p}, k_{r_d}, k_{z_θ}) to track the reference more aggressively compared to *policy 2*. *Policy 2* learned larger longitudinal controller gains (k_{v_p} and k_{v_d}), but smaller allowed maximum lateral acceleration (a_{latmax}) to track the reference velocity faster whenever the vehicle is not taking sharp turns, i.e. $v_d \leq v_{max}(\delta_s) = \sqrt{a_{latmax} L / |tan\delta_s|}$, where δ_s and L are the steering angle and wheelbase of the vehicle. Both policies learned a similar k_{det} for obstacle detection range. *Policy 2* learned a larger k_{shift} for a fixed shift in the lateral offset, whereas *policy 1* learned a smaller k_{shift} that is combined with the perception uncertainty to determine a varying lateral shift dependent on the level of uncertainty.

Fig. 5 shows the performance of each policy during 100 test episodes in terms of how often the vehicle traveled over high probability regions of pedestrian intent, as well as how close it got to the pedestrian state covariance. *Policy 1* and *policy 2* travelled over pedestrian intent regions with probabilities above 10% in 60 and 20 of the episodes, with average intent probability of 18% and 7%, respectively. In addition, *policy 1* maintained a mean distance of 0.71 m to the closest pedestrian's state covariance ellipse and intruded it in 5 episodes, as it shifted the desired lateral offset by a constant amount of k_{shift} regardless of the uncertainty. *Policy 2*, on the other hand, used the uncertainty associated with the perception detections to determine the required lateral shift by (15), resulting in a mean distance of 2.01 m and no intrusion in the pedestrian state covariance ellipse. Of note, a negative distance in the figures denotes vehicle intrusion into the pedestrian state covariance ellipse.

Fig. 6 shows the performance of each policy in a single lap around the track with perfect simulated detections (no uncertainty) and relatively high uncertainty (covaraince disk of 1 m radius). With perfect sensing (top row), *policy 1* deviated farther from the track (max 1.61 m) by taking a more conservative deviation from the pedestrians (min 0.93 m). *Policy 2*, however, remained closer to the track (max 1.21 m) while still staying outside the uncertainty region and maintaining a safe distance to the pedestrians (min 0.65 m). For the uncertain sensing scenario (bottom row), *policy 2* always maintained a safe distance to the pedestrians (min 0.91 m) by regarding the uncertain state estimations, whereas *policy 1* occasionally intruded the uncertain region around the pedestrian state (min -0.21 m), thus increasing the chance of collision. For this scenario, *policy 1* and *policy 2* maximum deviation from track were 2.62 m and 3.71 m, respectively.

Finally, we deployed each policy on the real ground vehicle and tested it during navigation around the track in presence of pedestrians. Both policies were able to perform 10 episodes of successful navigation around the track. The top-down view of the track, as well as an example pedestrian detection bounding box along with its standard deviation from Gaussian Yolov3 are shown in Fig. 3.

The mean (std) [max] for the variances obtained from Gaussian Yolov3 in the x and y directions were 0.0032 (0.0037) [0.0255] m and 0.0006 (0.0017) [0.0191] m, respectively. With such small detection uncertainties, the perception module is outputting near perfect detections. In practice, however, perception detections could be more uncertain due to e.g. occlusion or limited visibility. To simulate this behavior and test the behavior of the trained policies, we performed experiments with synthesized detection covariances, by sampling the covariance ellipse axes in the range of [1 m, 2 m].

To avoid hitting pedestrians, *policy 1* uses a fixed shift in the desired lateral offset, whereas *policy 2* uses the detection

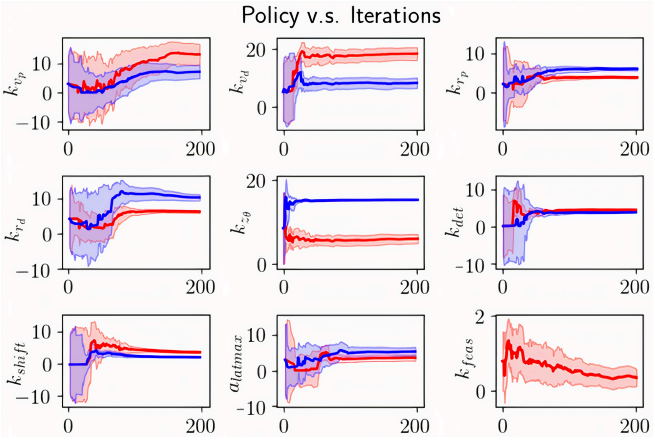

Fig. 4. Comparison of learned parameters for Policy 1 (blue) and policy 2 (red). The horizontal axis represents the number of iterations.

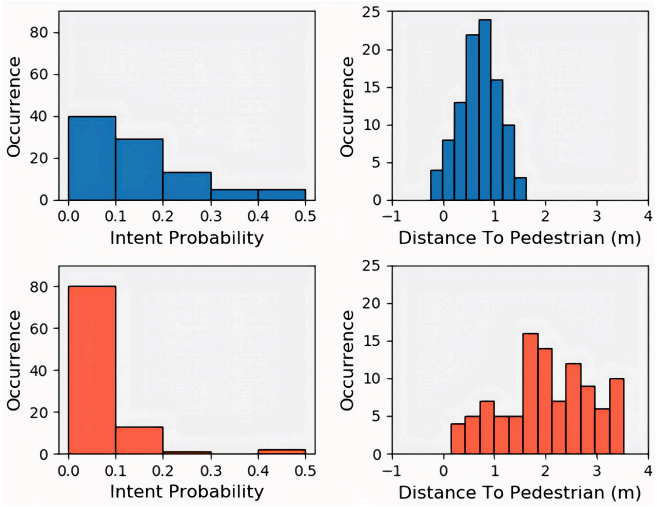

Fig. 5. Comparison of traveling over high intent probability and distance to pedestrian in 100 episodes of test for *policy 1* (blue) and *policy 2* (red). A negative distance denotes vehicle intrusion into the pedestrian state covariance ellipse.

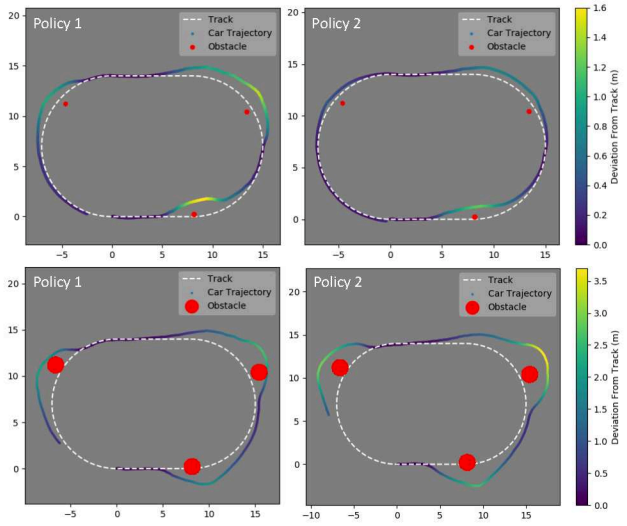

Fig. 6. Vehicle deviation from the track during obstacle avoidance with no uncertainty (top), and 1 m covariance disk radius (bottom).

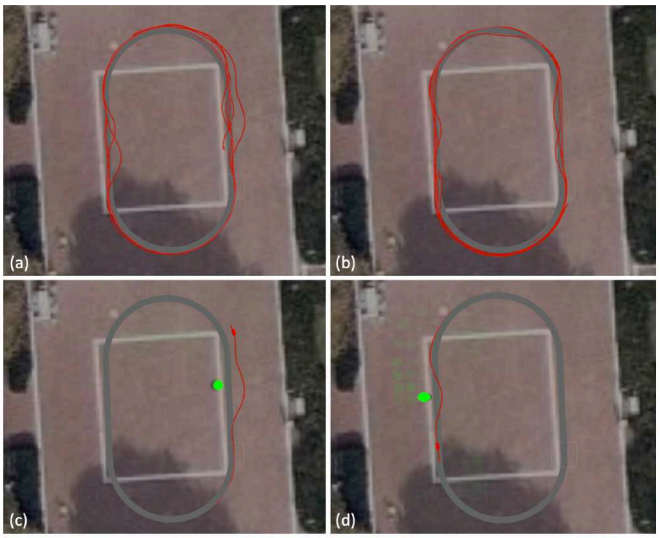

Fig. 7. Vehicle trajectory during ten episodes of navigation around the track with pedestrians using (a) *policy 1*, and (b) *policy 2*. Pedestrian avoidance cases with maximum deviation from the track using (c) *policy 1*, and (d) *policy 2*.

TABLE I
VEHICLE LATERAL DEVIATION FROM TRACK

	Mean (m)	Std (m)	Max (m)
Policy 1	0.35	0.20	1.26
Policy 2	0.24	0.12	0.61

uncertainties to determine the desired lateral offset. Fig. 7 (a) and (b) demonstrate the overall performance of *policy 1* and *policy 2*, respectively, in 10 episodes of navigation around the track in presence of pedestrians. *Policy 2* resulted in a smoother tracking performance and less track deviation compared to *policy 1* due to incorporation of uncertainty in determining more intelligent desired lateral offsets during pedestrian avoidance. The pedestrian avoidance cases with maximum deviation from the track for each policy are shown in Fig. 7(c) and (d), where *policy 1* has taken a sharper deviation from the pedestrian and the track compared to *policy 2*, even though it had a smaller pedestrian detection uncertainty. Table I summarizes the statistics of the vehicle lateral deviation from the track for each policy. The maximum deviation from track for *policy 1* and *policy 2* were 1.26 m and 0.61 m, respectively, indicating a smoother trajectory for *policy 2* while successfully navigating around pedestrians.

Finally, it should be noted that the underlying assumptions in this work were that the pedestrians do not interact with one another and they do not necessarily react to the robot's or other pedestrians' presence.

VI. CONCLUSION

We presented a robust policy search framework for a $^1/_5$-scale agile ground vehicle with built-in robustness to uncertainty by minimizing an upper confidence bound on the expected cost of trajectories. We showed in simulation and through real vehicle testing how perception uncertainty and pedestrian intent could be incorporated into this framework to enable more intelligent navigation in presence of pedestrians. Future work will combine deep neural networks with the existing algorithm to enable more versatile control frameworks for autonomous navigation in presence of dynamic entities with uncertainty.

REFERENCES

[1] D. Hennes, D. Claes, W. Meeussen, and K. Tuyls, "Multi-robot collision avoidance with localization uncertainty." in *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2012, pp. 147–154.

[2] A. Artunedo, J. Villagra, J. Godoy, and M. D. del Castillo, "Motion planning approach considering localization uncertainty," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5983–5994, 2020.

[3] J.-C. Latombe, A. Lazanas, and S. Shekhar, "Robot motion planning with uncertainty in control and sensing," *Artificial Intelligence*, vol. 52, no. 1, pp. 1–47, 1991.

[4] S. M. LaValle and S. A. Hutchinson, "An objective-based framework for motion planning under sensing and control uncertainties," *The International Journal of Robotics Research*, vol. 17, no. 1, pp. 19–42, 1998.

[5] S. Jha, V. Raman, D. Sadigh, and S. A. Seshia, "Safe autonomy under perception uncertainty using chance-constrained temporal logic," *Journal of Automated Reasoning*, vol. 60, no. 1, pp. 43–62, 2018.

[6] W. Xu, J. Pan, J. Wei, and J. M. Dolan, "Motion planning under uncertainty for on-road autonomous driving," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 2507–2512.

[7] A. M. Jasour and B. Williams, "Risk contours map for risk bounded motion planning under perception uncertainties." in *Robotics: Science and Systems*, 2019.

[8] B. Burns and O. Brock, "Sampling-based motion planning with sensing uncertainty," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 3313–3318.

[9] A. Bry and N. Roy, "Rapidly-exploring random belief trees for motion planning under uncertainty," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 723–730.

[10] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[11] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[12] J. Peters and S. Schaal, "Reinforcement learning by reward-weighted regression for operational space control," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 745–750.

[13] P. Wawrzyński and A. K. Tanwani, "Autonomous reinforcement learning with experience replay," *Neural Networks*, vol. 41, pp. 156–167, 2013.

[14] N. Heess, G. Wayne, D. Silver, T. Lillicrap, Y. Tassa, and T. Erez, "Learning continuous control policies by stochastic value gradients," *arXiv preprint arXiv:1510.09142*, 2015.

[15] J. Peters and S. Schaal, "Natural actor-critic," *Neurocomputing*, vol. 71, no. 7-9, pp. 1180–1190, 2008.

[16] P. Thomas, G. Theocharous, and M. Ghavamzadeh, "High confidence policy improvement," in *International Conference on Machine Learning*. PMLR, 2015, pp. 2380–2388.

[17] M. Sheckells, G. Garimella, and M. Kobilarov, "Robust policy search with applications to safe vehicle navigation," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2343–2349.

[18] M. Sheckells, G. Garimella, S. Mishra, and M. Kobilarov, "Actor-critic pac robust policy search," *2019 International Conference on Robotics and Automation (ICRA)*, 2019.

[19] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[20] M. Sheckells, G. Garimella, S. Mishra, and M. Kobilarov, "Using data-driven domain randomization to transfer robust control policies to mobile robots," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3224–3230.

[21] C. Cortes, Y. Mansour, and M. Mohri, "Learning bounds for importance weighting." in *Neural Information Processing Systems (NIPS)*, vol. 10. Citeseer, 2010, pp. 442–450.

[22] M. Kobilarov, "Sample complexity bounds for iterative stochastic policy optimization." in *Neural Information Processing Systems (NIPS)*, 2015, pp. 3114–3122.

[23] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human motion trajectory prediction: A survey," *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 895–935, 2020.

[24] S. Qi and S.-C. Zhu, "Intent-aware multi-agent reinforcement learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7533–7540.

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

[26] T. Moore and D. Stouch, "A generalized extended kalman filter implementation for the robot operating system," in *Proceedings of the 13th International Conference on Intelligent Autonomous Systems (IAS-13)*. Springer, July 2014.

[27] J. Choi, D. Chun, H. Kim, and H.-J. Lee, "Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[28] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018. [Online]. Available: http://arxiv.org/abs/1804.02767