Learning Visuomotor Policies for Aerial Navigation Using Cross-Modal Representations

Rogerio Bonatti*[1], Ratnesh Madaan[2], Vibhav Vineet[2], Sebastian Scherer[1], and Ashish Kapoor[2]

***Abstract*— Machines are a long way from robustly solving open-world perception-control tasks, such as first-person view (FPV) aerial navigation. While recent advances in end-to-end Machine Learning, especially Imitation and Reinforcement Learning appear promising, they are constrained by the need of large amounts of difficult-to-collect labeled real-world data. Simulated data, on the other hand, is easy to generate, but generally does not render safe behaviors in diverse real-life scenarios. In this work we propose a novel method for learning robust visuomotor policies for real-world deployment which can be trained purely with simulated data. We develop rich state representations that combine supervised and unsupervised environment data. Our approach takes a cross-modal perspective, where separate modalities correspond to the raw camera data and the system states relevant to the task, such as the relative pose of gates to the drone in the case of drone racing. We feed both data modalities into a novel factored architecture, which learns a joint low-dimensional embedding via Variational Auto Encoders. This compact representation is then fed into a control policy, which we trained using imitation learning with expert trajectories in a simulator. We analyze the rich latent spaces learned with our proposed representations, and show that the use of our cross-modal architecture significantly improves control policy performance as compared to end-to-end learning or purely unsupervised feature extractors. We also present real-world results for drone navigation through gates in different track configurations and environmental conditions. Our proposed method, which runs fully onboard, can successfully generalize the learned representations and policies across simulation and reality, significantly outperforming baseline approaches.**

Supplementary video: https://youtu.be/VKc3A5HlUU8
Open-sourced code available at: https://github.com/microsoft/AirSim-Drone-Racing-VAE-Imitation

I. Introduction

Aerial navigation of drones using first-person view (FPV) images is an exemplary feat of the human mind. Expert pilots are able to plan and control a quadrotor with high agility using a potentially noisy monocular camera feed, without comprising safety. We are interested in exploring the question of what would it take to build autonomous systems that achieve similar performance levels.

One of the biggest challenges in the navigation task is the high dimensional nature and drastic variability of the input image data. Successfully solving the task requires a representation that is invariant to visual appearance and robust to the differences between simulation and reality. Collecting labeled real-world data to minimize the sim-to-real gap, albeit possible, requires intensive effort and specialized equipment for gathering ground-truth labels [1], [2]. Attempting to solve the task solely with real-world data is also challenging due to poor sample efficiency of end-to-end methods, and often leads to policies that are unable to deal with large perceptual variability [3]–[5]. Additionally, end-to-end training in the real world is expensive and dangerous, especially in early phases of training when policies are highly prone to errors and collisions.

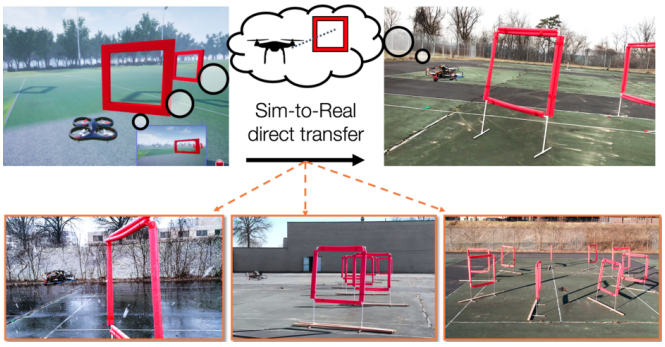

Fig. 1. The proposed framework uses simulations to learn a rich low-dimensional state representation using multiple data modalities. This latent vector is used to learn a control policy which directly transfers to real-world environments. We successfully deploy the system under various track shapes and weather conditions, ranging from sunny days to strong snow and wind.

Sim-to-real transfer learning methods aim to partially alleviate these challenges by training policies in a synthetic environment and then deploying them in the real-world [6]–[8]. Domain randomization uses a large collection of object appearances and shapes, assuming that any real-life features encountered will be represented within a subset of the database. Simulations can be used to generate large amounts of synthetic data under a wide variety of conditions [9], [10].

In this work, we introduce cross-modal learning for generating representations which are robust to the simulation-reality gap, and do not overfit to specificities of the simulated data. In particular, the need to explain multiple modes of information during the training phase aids in implicit regularization of the representation, leading to an effective transfer of models trained in simulation into the real world. We use high-fidelity simulators [11] to generate both labeled and unlabeled modalities of simulated data. Furthermore, our proposed framework can also use unlabeled real-world data in the training phase, thereby allowing us to incorporate real-life unlabeled traces into the state representation. Fig. 1 depicts the overall concept, showing a single perception module shared for simulated and real autonomous navigation. Our unmanned aerial vehicle (UAV) uses FPV images to extract

[1]The Robotics Institute, Carnegie Mellon University, Pittsburgh PA {rbonatti, basti}@cs.cmu.edu
[2]Microsoft Corporation, Redmond, WA {ratnesh.madaan, vibhav.vineet, akapoor}@microsoft.com
* Work done while interning at Microsoft Corporation, Redmond

a low-dimensional representation, which is then used as the input to a control policy network to determine the next control actions. We successfully test the system operating in challenging conditions, many of which were previously unseen by the simulator, such as snow and strong winds.

Our proposed framework entails learning a cross-modal representation for state encoding. The first data modality considers the raw unlabeled sensor input (FPV images), while the second directly characterizes state information directly relevant for the desired application. In the case of drone racing, our labels correspond to the relative pose of the next gate defined in the drone's frame. We learn a low-dimensional latent representation by extending the Cross-Modal Variational Auto Encoder (CM-VAE) framework from [12], which uses an encoder-decoder pair for each data modality, while constricting all inputs and outputs to and from a single latent space. Consequently, we can naturally incorporate both labeled and unlabeled data modalities into the training process of the latent variable. We then use imitation learning to train a control policy that maps latent variables into velocity commands for the UAV. The representation uses only raw images during deployment, without access to the ground-truth gate poses. While closest to our work is the recent line of research on autonomous drone racing [1], [10], we would like to note that our objective here is not to engineer a system that necessarily finishes the race the fastest. Unlike the prior work, we do not assume prior knowledge during deployment in terms of an accurate dynamics model coupled with optimal trajectory generation methods. Instead, our goal is to learn visuomotor policies operating on learned representations that can be transferred from simulation to reality. Thus, the methods presented in this paper are not directly comparable to [1], [10].

While in this paper we specifically focus on the problem of aerial navigation in a drone racing setting, the proposed techniques are general and can be applied to other perception-control tasks in robotics. Our key contributions are:

- We present a cross-modal framework for learning latent state representations for navigation policies that use unsupervised and supervised data, and interpret the bi-modal properties of the latent space encoding;
- We provide simulation experiments comparing variants of the cross-modal framework with baseline feature extractors such as variational auto-encoders (VAEs), gate pose regression, and end-to-end learning;
- We provide multiple real-world navigation experiments and performance evaluations of our control policies. We show that our proposed representation allows for sim-to-real deployment of models learned purely in simulation, achieving over one kilometer of cumulative autonomous flight through obstacles.

II. Related Work

a) Navigation policies: Classically, navigation policies rely on state estimation modules that use either visual-inertial based odometry [13] or simultaneous localization and mapping [14]. These techniques can present high drift and noise in typical field conditions, impacting the quality of both the robot localization and the map representation used for planning. Therefore, trajectory optimization based algorithms [15]–[17] can result in crashes and unsafe robot behaviors. Against these effects, [18] learn a collision avoidance policy in dense forests using only monocular cameras, and [19] learn a steering function for aerial vehicles in unstructured urban environments using driving datasets for supervision.

Recently, [3], [20]–[22] explore learning separate networks for the environment representation and controls, instead of the end-to-end paradigm. The goal of an intermediate representations is to extract a low-dimensional space which summarizes the key geometrical properties of the environment, while being invariant to textures, shapes, visual artifacts. Such intermediate representations mean that behavior cloning or reinforcement learning methods have a smaller search space [21] and more sample efficiency.

b) Learning representations for vision: Variational Autoencoder (VAE) based approaches have been shown to be effective in extracting low-dimensional representation from image data [23]–[26]. Recently, VAEs have been to leveraged to extract representations from multiple modalities [12], [27]–[29]. Relevant to our work, [12] propose a cross-modal VAE to learn a latent space that jointly encodes different data modalities (images and 3D kyepoints associated with hand joints) for a image to hand pose estimation problem.

c) Drone Racing: We find different problem definitions in the context of autonomous drone racing. [1], [10] focus on scenarios with dynamic gates by decoupling perception and control. They learn to regress to a desired position using monocular images with a CNN, and plan and track a minimum jerk trajectory using classical methods [15], [30]. [10] utilize domain randomization for effective simulation to reality transfer of learned policies.

Gate poses are assumed as *a priori* unknown in [31]–[33]. [31] use depth information and a guidance control law for navigation. [32], [33] use a neural network for gate detection on the image. A limitation of the guidance law approach is that the gate must be in view at all times, and it does not take into account gate relative angles during the approach.

[2] formulate drone racing as flight though a predefined ordered set of gates. They initialize gate locations with a strong prior via manual demonstration flights. The belief over each gate location is then updated online by using a Kalman Filter over the gate pose predictions from a neural network.

In our work we take inspirations from the fields of policy and representation learning to present a method that combines unsupervised and supervised simulated data to train a single

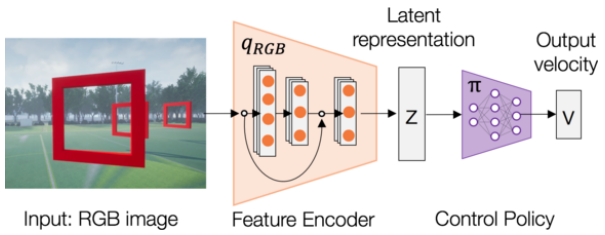

Fig. 2. Control system architecture. The input image is encoded into a latent representation of the environment. A control policy acts on the lower-dimensional embedding to output the desired robot velocity commands.

latent space on which a control policy acts. The bi-modal nature of the latent space implicitly regularizes the representation model, allowing for policy generalization across the simulation-reality gap.

III. Approach

This work addresses the problem of robust autonomous navigation through a set of gates with unknown locations. Our approach is composed of two steps: first, learning a latent state representation, and second, learning a control policy operating on this latent representation (Fig. 2). The first component receives monocular camera images as input and encodes the relative pose of the next visible gate along with background features into a low-dimensional latent representation. This latent representation is then fed into a control network, which outputs a velocity command, later translated into actuator commands by the UAV's flight controller.

A. Definitions and Notations

Let W define the world frame, B the body frame, and G_i the frame of the target gate. Let E define the full environment geometry and object categories. Assuming that all gates are upright, let $y_i = [r, \theta, \phi, \psi]$ define the relative spherical coordinates and yaw gate frame G_i, in the B frame.

We define $q_{RGB}(I_t) \rightarrow \mathbb{R}^N$ to be an encoder function that maps the current image I_t to a latent compressed vector z_t of size N. Let $\pi(z_t) \rightarrow \mathbb{R}^4$ be a control policy that maps the current encoded state to a body velocity command $v_B = [v_x, v_y, v_z, v_\psi]$, corresponding to linear and yaw velocities.

Let π^* be an expert control policy. Our objective is to find the optimal model parameters Θ^* and Φ^* that minimize the expectation of distance D between our control policy and the expert, taken over observed states s. Note that the expert policy operates with full knowledge of the environment E, while our policy only has access to the observation $q_{RGB}(I_t)$:

$$\Theta^*, \Phi^* = \arg\min_{\Theta,\Phi} \quad \mathbb{E}_s \left[D\Big(\pi^*\big(E\big),\ \pi^{\Phi}\big(q^{\Theta}_{RGB}(I)\big)\Big)\right] \tag{1}$$

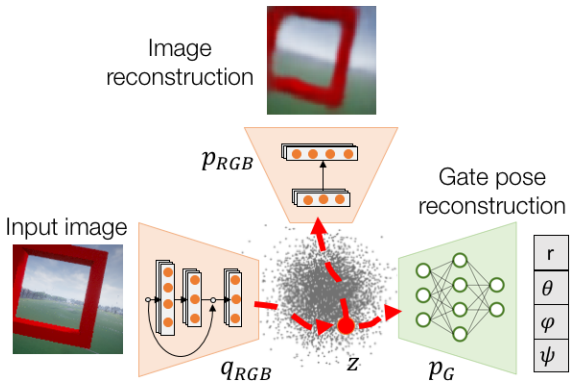

Fig. 3. Cross-modal VAE architecture. Each data sample is encoded into a single latent space that can be decoded back into images, or transformed into another data modality such as the poses of gates relative to the UAV.

B. Learning Cross-Modal Representations for Perception

The goal of the perception module is to extract all pertinent information for the current task from the current state of the environment E and UAV. Several approaches exist for feature extraction, from fully supervised to fully unsupervised methods, as mentioned in Section II.

An effective dimensionality reduction technique should be smooth, continuous and consistent [12], and in our application, also be robust to differences in visual information across both simulated and real images. To achieve these objectives we build on the architecture developed by [12], who use a cross-modal variant of variational auto-encoders (CM-VAE) to train a single latent space using multiple sources of data representation. The core hypothesis behind our choice of encoding architecture is that, by combining different data modalities into one latent vector, we can induce a regularization effect to prevents overfitting to one particular data distribution. As shown in Section IV-C, this becomes an important feature when we transfer a model learned with simulated data to the real world, where appearances, textures and shapes can vary significantly.

CM-VAE derivation and architecture: The cross-modal architecture works by processing a data sample x, which can come from different modalities, into the same latent space location (Fig. 3). In robotics, common data modalities found are RGB or depth images, LiDAR or stereo pointclouds, or 3D poses of objects in the environment. In the context of drone racing, we define data modalities as RGB images and the relative pose of the next gate to the current aircraft frame, *i.e*, $x_{RGB} = I_t$ and $x_G = y_i = [r, \theta, \phi, \psi]$. The input RGB data is processed by encoder q_{RGB} into a normal distribution $\mathcal{N}(\mu_t, \sigma_t^2)$ from which z_t is sampled. Either data modality can be recovered from the latent space using decoders p_{RGB} and p_G.

In the standard definition of VAEs, the objective is to optimize the variational lower bound on the log-likelihood of the data [23], [34]. In [12], this loss is re-derived to account for probabilities across data modalities x_i and x_j, resulting in

the new lower bound shown in Eq. 2:

$$\mathbb{E}_{z\sim q(z|x_i)}\left[\log p(x_j|z)\right] - D_{KL}(q(z|x_i)||p(z)) \qquad (2)$$

We use the Dronet [19] architecture for encoder p_{RGB}, which is equivalent to an 8-layer Resnet [35]. We choose a small network, with about 300K parameters, for its low onboard inference time. For the image decoder q_{RGB} we use six transpose convolutional layers, and for the gate decoder p_G we use two dense layers.

Training procedure: We follow the training procedure outlined in Algorithm 1 of [12], considering three losses: (i) MSE loss between actual and reconstructed images $(I_t, \hat{I}_t)$, (ii) MSE loss for gate pose reconstruction $(y_i, \hat{y_i})$, and (iii) Kullback-Leibler (KL) divergence loss for each sample. During training, for each unsupervised data sample we update networks q_{RGB}, p_{RGB}, and for each supervised sample we update both image encoder q_{RGB} and gate pose decoder p_G with the gradients.

Imposing constraints on the latent space: Following recent work in distantangled representations [24], [36], we compare two architectures for the latent space structure. Our goal is to improve performance of the control policy and interpretability of results. In first architecture, z_{unc} stands for the unconstrained version of the latent space, where: $\hat{y_i} = p_G(z_{unc})$ and $\hat{I}_t = p_{RGB}(z_{unc})$. For second architecture, instead of a single gate pose decoder p_G, we employ 4 independent decoders for each gate pose component, using the first 4 elements of z_{con}. As human designers, we know that these features are independent (*e.g*, the distance between gate and drone should have no effect on the gate's orientation). Therefore, we apply the following constraints to z_{con}: $\hat{r} = p_r(z_{con}^{[0]})$, $\hat{\theta} = p_\theta(z_{con}^{[1]})$, $\hat{\psi} = p_\psi(z_{con}^{[2]})$, $\hat{\phi} = p_\phi(z_{con}^{[3]})$. The image reconstruction step still relies on the full latent variable: $\hat{I}_t = p_{RGB}(z_{con})$.

C. Imitation learning for control policy

Expert trajectory planner and tracker: To generate expert data $(\pi^*(E))$ we use a minimum jerk trajectory planner following the work of [15], [30], [37] considering a horizon of one gate into the future, and track it using a pure-pursuit path tracking controller. We generate a dataset of monocular RGB images with their corresponding controller velocity commands.

Imitation learning algorithm: We use behavior cloning (BC), a variant of supervised learning [38], to train the control policy $\pi(q(I))$ when minimizing Equation 1. We freeze all encoder weights when training the control policy. In total we train 5 policies for the simulation experiments: BC_{con} and BC_{unc}, which operate on z_{con} and z_{unc} respectively as features, BC_{img}, which uses a pure unsupervised image reconstruction VAE for features, BC_{reg}, which uses a purely supervised regressor from image to gate pose as features, and finally BC_{full}, which uses a full end-to-end mapping from images to velocities, without an explicit latent feature vector. We train an additional policy BC_{real} for the physical experiments, using unsupervised real-life images along the CM-VAE architecture, as further detailed in Section IV-C.

To provide a fair comparison between policy classes, we design all architectures to have the same size and structure. BC policies learned on top of representations are quite small, with only 3 dense layers and roughly 6K neurons. The end-to-end BC_{full} layout is equivalent to the combination of the Dronet encoder plus BC policy from the other cases, but initially all parameters are untrained.

IV. Results

A. Learning Representations

Our first set of experiments aims to valuate the latent representations from three different architectures: (i) q_{reg}, for direct regression from $I_t \rightarrow z_{reg} = [r, \theta, \phi, \psi] \in \mathbb{R}^4$, (ii) q_{unc}, for the CM-VAE using RGB and pose modalities without constraints: $I_t \rightarrow z_{unc} \in \mathbb{R}^{10}$, and (iii) q_{con}, for the CM-VAE using RGB and pose modalities with constraints: $I_t \rightarrow z_{con} \in \mathbb{R}^{10}$.

We generated 300K pairs of 64×64 images along with their corresponding ground-truth relative gate poses using the AirSim simulator [11]. We randomly sample the distance to gate, aircraft orientation, and gate yaw. 80% of the data was used to train the network, and 20% was used for validation.

Fig. 4 displays images decoded over various regions of the latent spaces z_{con} and z_{unc}. Each row corresponds to variations in z values in one of the 10 latent dimensions. We verify that the latent variables can encode relevant information about the gate poses and background information. In addition, the constrained architecture indeed learned to associate the first four dimensions of z_{con} to affect the size, the horizontal offset, the vertical offset and the yaw of the visible gate.

Fig. 5 depicts examples of input images (left) and their corresponding reconstructions (right). We verify that the reconstruction captures the essence of the scene, preserving both the background and gate pose.

The smoothness of the latent space manifold with respect to the gate poses and image outputs is a desirable property (*i.e.*, similar latent vectors correspond to similar gate poses). Intuitively, our single cross-modal latent space should lead to such smooth latent space representation, and our next analysis confirms that such properties emerge automatically. In Fig. 6 we show the decoded outputs of a latent space interpolation between the encoded vectors from two very different simulated images. Both images and their decoded poses are smoothly reconstructed along this manifold.

Additionally, we quantitatively evaluate the predictions of the three architectures that can recover the gate poses from the images, as shown in Table I. When trained for the same number of epochs, q_{reg}, q_{unc}, and q_{con} achieve roughly the same error in prediction. The cross-modal latent space can encode gate pose information slightly better than direct

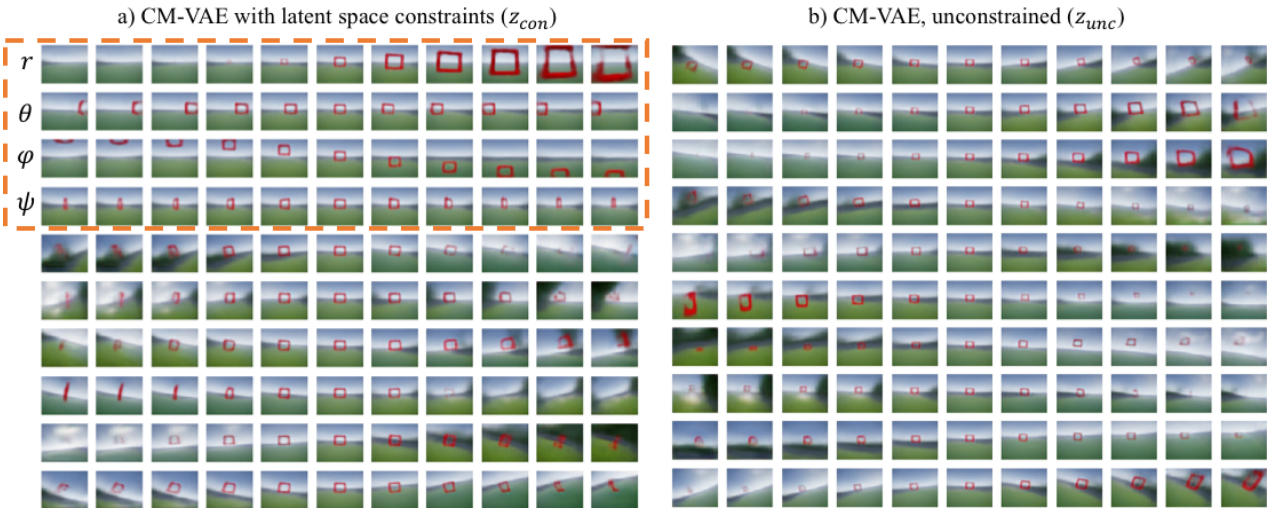

Fig. 4. Visualization of latent space from a) constrained and b) unconstrained cross-modal representations. The constraints on latent space force the disantanglement of the first four variables of z_{con} to encode the relative gate pose, condition that is also observed in the image modality.

TABLE I

AVERAGE AND STANDARD ERRORS FOR ENCODING GATE POSES

q	**Radius** r **[m]**	**Azimuth** θ [°]	**Polar** ϕ [°]	**Yaw** ψ [°]
q_{reg}	0.41 ± 0.013	2.4 ± 0.14	2.5 ± 0.14	11 ± 0.67
q_{unc}	0.42 ± 0.024	$\mathbf{2.3 \pm 0.23}$	$\mathbf{2.1 \pm 0.23}$	$\mathbf{9.7 \pm 0.75}$
q_{con}	$\mathbf{0.39 \pm 0.023}$	2.6 ± 0.23	2.3 ± 0.25	10 ± 0.75

regression, likely due to the additional unsupervised gradient information. Small variations can also be attributed to the training regime using stochastic gradient descent.

B. Simulated navigation results

Our next set of experiments evaluates control policies learned over five different types of feature extractors. As described in Section III-C, we train behavior cloning policies on top of the CM-VAE latent spaces (BC_{con}, BC_{unc}), a direct gate pose regressor (BC_{reg}), vanilla VAE image reconstruction features (BC_{img}), and finally full end-to-end training (BC_{full}).

For data collection we generated a nominal circular track with 50m of length, over which we placed 8 gates with randomized position offsets in XYZ changing at every drone traversal. We collected 17.5K images with their corresponding expert velocity actions while varying the position offset level from 0-3m. 80%, or 14K datapoints, were used to train the behavior cloning policies, and the remainder were used for validation.

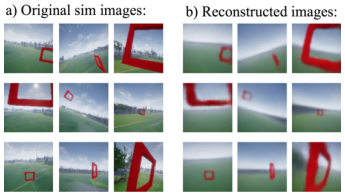

Fig. 5. Comparison between original simulated images with their respective CM-VAE reconstructions. Reconstructed images are blurrier than the original, but overall gate and background features can be well represented.

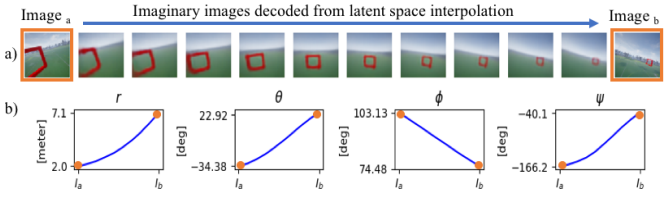

Fig. 6. Visualization of latent space interpolation between two simulated images. Smooth interpolation can be perceived in both image and gate pose data modalities. Even background features such as the ground's tilt are smoothly captured.

We evaluate our proposed framework under controlled simulation conditions analogous to data collection. Similarly to previous literature [1], [2], [10], we define a success metric of 100% as the UAV traversing all gates in 3 consecutive laps. For each data point we average results over 10 trials in different randomized tracks. Figure 7 shows the performance of different control policies which were trained using different latent representations, under increasing random position offset amplitudes. At a condition of zero noise added to the gates, most methods, except for the latent representation that uses pure image compression, can perfectly traverse all gates. As the track difficulty increases, end-to-end behavior cloning performance drops significantly, while methods that use latent representations degrade slower. At a noise level of 3 m over a track with 8 m of nominal radius the proposed cross-modal representation BC_{con} can still achieve approximately 40% success rate, 5X more than end-to-end learning. We invite the reader to watch the supplementary video for more details.

The three architectures that implicitly or explicitly encode gate positions (BC_{con}, BC_{unc}, BC_{reg}) perform significantly better than the baselines. This behavior likely spans from the small pixel-wise footprint of gates on the total image, which makes it harder for the vanilla VAE architecture or end-to-end learning to effectively capture the relative object poses. However, even though the regression features have

a relatively good performance in simulation, policy success degrades when exposed to real-world images, as detailed in Subsection IV-C.

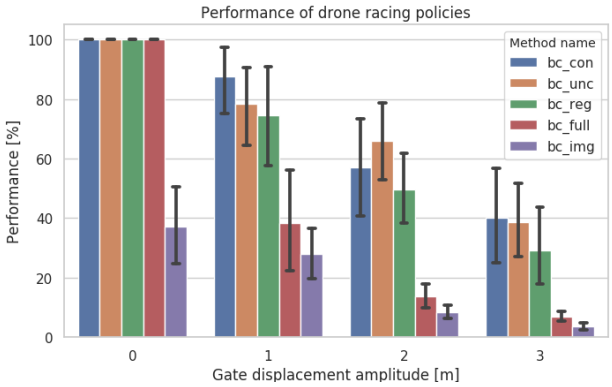

Fig. 7. Performance of different navigation policies on simulated track.

C. Real-World Results

We also validate the ability of the different visuomotor policies to transfer from simulation to real-world deployment. Our platform is a modified kit[1], as shown in Figure 8. All processing is done fully onboard with a Nvidia TX2 computer, with 6 CPU cores and an integrated GPU. An off-the-shelf Intel T265 Tracking Camera provides odometry, and image processing uses the Tensorflow framework. The image sensor is a USB camera with 83° horizontal FOV, and we downsize the original images to dimension 128×72.

Fig. 8. UAV platform.

First we evaluate how the CM-VAE module, which was learned only with simulated data, performs with real-world images as inputs. We only focus on z_{con} given that it presented the best overall performance in simulation experiments. Fig. 9 shows that the latent space encoding remains smooth and consistent. We train this model with a new simulation dataset composed of 100K images with size 128×72 and FOV equal to our hardware USB camera.

To show the capabilities of our approach on a physical platform, we test the system on a S-shaped track with 8 gates and 45m of length, and on a circular track with 8 gates and 40m of length, as shown in Fig 10. We compare three policies: BC_{con}, BC_{reg}, and BC_{real}. To train this last control policy, BC_{real}, we use a CM-VAE trained using not only the 100K images from simulation, but also additional 20K unsupervised real-world images. Our goal with this policy is to compare if the use of unsupervised real data can help in the extraction of better features for navigation.

We display results from both tracks on Table II. The navigation policy using CM-VAE features trained purely in simulation significantly outperforms the baseline policies in both tracks, achieving over $3\times$ the performance of BC_{reg} in the S-track. The performance gap is even larger on the circuit track, with BC_{con} achieving a maximum of 26 gates traversed before any collision occurs. It is important to note that some of the circuit trials occurred among wind gusts of up to 20km/h, a fact that further shows that our policies learned in simulation can operate in physical environments.

[1]https://www.getfpv.com/student-competition-5-bundle.html

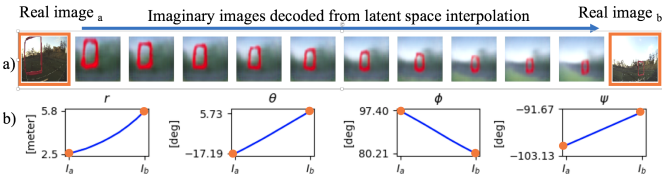

Fig. 9. Visualization of smooth latent space interpolation between two real-world images. The ground-truth and predicted distances between camera and gate for images A and B were (2.0, 6.0) and (2.5, 5.8) meters respectively.

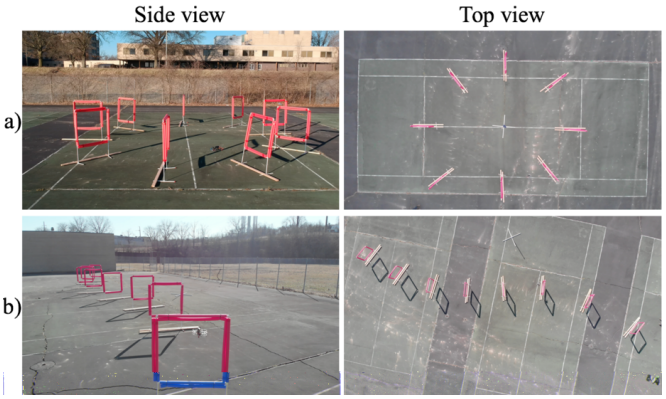

Fig. 10. Side and top view of: a) Circuit track, and b) S-shape track.

We also investigate the cause of the performance drop in BC_{reg} when transferred to real-world data. Table III shows the ground-truth and predicted gate distances for different input images. The CM-VAE, despite being trained purely on simulation, can still decode reasonable values for the gate distances. Direct regression, however, presents larger errors. In addition, Figure 11 displays the accumulated gate poses as decoded from both representations during 3s of a real flight test. The regression poses are noticeably noisier and farther from the gate's true location.

In the experiments thus far we deployed the learned policies on physical environments that roughly resemble the visual appearances of the simulation dataset. There, all images were generated in a grass field with blue skies, and trees in the background. To verify policy and representation robustness to extreme visual changes, we perform additional tests in more challenging scenarios. Fig. 12 shows examples of successful test cases: Fig. 12a) indoors where the floor is blue with red stripes, and Fig. 12b-c) among heavy snow. We invite the reader to visualize these experiments in the video attachment (https://youtu.be/VKc3A5HlUU8).

TABLE II

POLICY PERFORMANCE IN NUMBER OF GATES TRAVERSED

	S-Track [12 trials]		Circuit [6 trials]	
	Mean	Max	Mean	Max
BC_{con}	**7.8**	**8**	**14.3**	**26**
BC_{real}	5.0	7	3.1	5
BC_{reg}	2.3	5	1.2	2

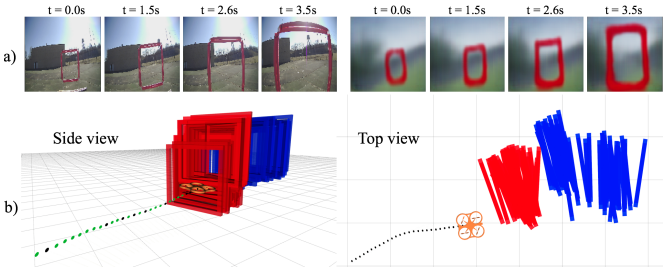

Fig. 11. Analysis of a 3-second flight segment. a) Input images and their corresponding images decoded by the CM-VAE; b) Time history of gate center poses decoded from the CM-VAE (red) and regression (blue). The regression representation has significantly higher offset and noise from the true gate pose, which explains its poor flight performance.

TABLE III

EXAMPLES OF DISTANCE TO GATES DECODED FROM REAL IMAGES

Image				
Ground-truth $[m]$	2	4	6	8
CM-VAE $[m]$	**2.16**	**3.78**	**6.10**	**8.77**
Regression $[m]$	4.67	5.50	6.68	9.13

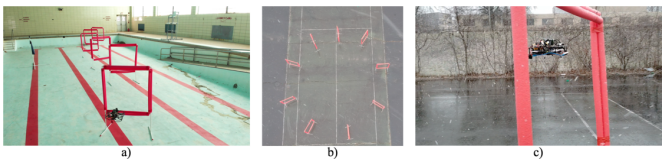

Fig. 12. Examples of challenging test environments: a) Indoors, with blue floor with red stripes, and b-c) among heavy snowfall.

V. CONCLUSION AND DISCUSSION

In this work we present a framework to solve perception-control tasks that uses simulation-only data to learn rich representations of the environment, which can then be employed by visuomotor policies in real-world aerial navigation tasks. At the heart of our approach is a cross-modal Variational Auto-Encoder framework that jointly encodes raw sensor data and useful task-specific state information into a latent representation to be leveraged by a control policy. We provide detailed simulation and real-world experiments that highlight the effectiveness of our framework on the task of FPV drone navigation. Our results show that the use of cross-modal representations significantly improves the real-world performance of control policies in comparison with several baselines such as gate pose regression, unsupervised representations, and end-to-end approaches.

The main finding we can infer from our experiments is that by introducing multiple data modalities into the feature extraction step, we can avoid overfitting to specific characteristics of the incoming data. For example, even though the sizes of the square gates were the same in simulation and physical experiments, their width, color, and even intrinsic camera parameters are not an exact match. The multiple streams of information that the CM-VAE encounters regularize its model, leading to better generalization among appearance changes.

From our experiments in Fig. 7 we can also infer that features trained for unsupervised image reconstruction can serve as important cues describing the UAV's current state for the control policy, on top of the explicit human-defined supervised parameters. For example, by using background features such as the line of horizon, a pilot may infer the UAV's current roll angle, which influences the commanded velocities. This remark can serve as an additional motivator for the use of a semi-supervised features extraction module, since it is difficult to hand-define all relevant features for a particular control task. Another advantage of the CM-VAE architecture is that it can allow the robot operator to gain insights onto the decisions made by the networks. For instance, a human can interpret the decoded gate pose and decoded images in real time and stop the vehicle if perception seems to be abnormal.

Interestingly, BC_{real} did not outperform BC_{con} in our real-world experiments, as we originally expected. However, it was still better than BC_{reg}. We suspect that the drop in performance happens because the dataset used for imitation learning only contained images from simulation, and there is distribution shift in comparison with images used for training the representation. As future work we envision using adversarial techniques such as [39] for lowering the distance in latent space between similar scenes encountered in sim and real examples.

Additional future work includes extensions of the cross-modal semi-supervised feature extraction framework to other robotics tasks, considering the use of multiple unsupervised data modalities that span beyond images. We believe that applications such as autonomous driving and robotic manipulation present perception and control scenarios analogous to the aerial navigation task, where multiple modalities of simulated data can be cheaply obtained.

ACKNOWLEDGMENTS

We thank Matthew Brown, Nicholas Gyde, Christoph Endner, Lorenz Stangier, and Nicolas Iskos for the help with experiments, and Debadeepta Dey for insightful discussions.

REFERENCES

[1] E. Kaufmann, A. Loquercio, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Deep drone racing: Learning agile flight in dynamic environments," *arXiv preprint arXiv:1806.08548*, 2018.

[2] E. Kaufmann, M. Gehrig, P. Foehn, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Beauty and the beast: Optimal methods meet learning for drone racing," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 690–696.

[3] K. Kang, S. Belkhale, G. Kahn, P. Abbeel, and S. Levine, "Generalization through simulation: Integrating simulated and real data into deep reinforcement learning for vision-based autonomous flight," *arXiv preprint arXiv:1902.03701*, 2019.

[4] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 31–36.

[5] J. Zhu, Z. Zhang, C. Zhang, J. Wu, A. Torralba, J. B. Tenenbaum, and W. T. Freeman, "Visual object networks: Image generation with disentangled 3d representation," *CoRR*, vol. abs/1812.02725, 2018.

[6] F. Sadeghi and S. Levine, "(CAD)^{2}RL: Real single-image flight without a single real image," *arXiv preprint arXiv:1611.04201*, 2016.

[7] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[8] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 23–30.

[9] Y. Wu, Y. Wu, G. Gkioxari, and Y. Tian, "Building generalizable agents with a realistic and rich 3d environment," *arXiv preprint arXiv:1801.02209*, 2018.

[10] A. Loquercio, E. Kaufmann, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Deep drone racing: From simulation to reality with domain randomization," *arXiv preprint arXiv:1905.09727*, 2019.

[11] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," *Field and Service Robotics*, 2017. [Online]. Available: https://arxiv.org/abs/1705.05065

[12] A. Spurr, J. Song, S. Park, and O. Hilliges, "Cross-modal deep variational hand pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 89–98.

[13] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2502–2509.

[14] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 573–580.

[15] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2520–2525.

[16] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran, "Continuous-time trajectory optimization for online uav replanning," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 5332–5339.

[17] R. Bonatti, C. Ho, W. Wang, S. Choudhury, and S. Scherer, "Towards a robust aerial cinematography platform: Localizing and tracking moving targets in unstructured environments," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[18] S. Ross, N. Melik-Barkhudarov, K. S. Shankar, A. Wendel, D. Dey, J. A. D. Bagnell, and M. Hebert , "Learning monocular reactive uav control in cluttered natural environments," in *IEEE International Conference on Robotics and Automation*. IEEE, March 2013.

[19] A. Loquercio, A. I. Maqueda, C. R. Del-Blanco, and D. Scaramuzza, "Dronet: Learning to fly by driving," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1088–1095, 2018.

[20] D. Ha and J. Schmidhuber, "World models," *arXiv preprint arXiv:1803.10122*, 2018.

[21] T. Lesort, N. Díaz-Rodríguez, J.-F. Goudou, and D. Filliat, "State representation learning for control: An overview," *Neural Networks*, vol. 108, pp. 379–392, 2018.

[22] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," *arXiv preprint arXiv:1811.04551*, 2018.

[23] M. W. Diederik Kingma, "Autoencoding variational bayes," in *ICLR*, 2014.

[24] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France*, 2017.

[25] H. Kim and A. Mnih, "Disentangling by factorising," *arXiv:1802.05983*, 2018.

[26] C. P. Burgess, L. Matthey, N. Watters, R. Kabra, I. Higgins, M. Botvinick, and A. Lerchner, "Monet: Unsupervised scene decomposition and representation," *CoRR*, 2019.

[27] Y. Aytar, L. Castrejon, C. Vondrick, H. Pirsiavash, and A. Torralba, "Cross-modal scene networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 10, pp. 2303–2314, 2017.

[28] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 689–696.

[29] V. E. Liong, J. Lu, Y.-P. Tan, and J. Zhou, "Cross-modal deep variational hashing," in *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 4097–4105.

[30] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics Research*. Springer, 2016, pp. 649–666.

[31] S. Jung, S. Cho, D. Lee, H. Lee, and D. H. Shim, "A direct visual servoing-based framework for the 2016 iros autonomous drone racing challenge," *Journal of Field Robotics*, vol. 35, no. 1, pp. 146–166, 2018.

[32] S. Jung, S. Hwang, H. Shin, and D. H. Shim, "Perception, guidance, and navigation for indoor autonomous drone racing using deep learning," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2539–2544, 2018.

[33] S. Jung, H. Lee, S. Hwang, and D. H. Shim, "Real time embedded system framework for autonomous drone racing using deep learning techniques," in *2018 AIAA Information Systems-AIAA Infotech Aerospace*, 2018, p. 2138.

[34] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," *arXiv preprint arXiv:1401.4082*, 2014.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[36] H. Kim and A. Mnih, "Disentangling by factorising," *arXiv preprint arXiv:1802.05983*, 2018.

[37] M. Burri, H. Oleynikova, , M. W. Achtelik, and R. Siegwart, "Real-time visual-inertial mapping, re-localization and planning onboard mavs in unknown environments," in *Intelligent Robots and Systems (IROS 2015), 2015 IEEE/RSJ International Conference on*, Sept 2015.

[38] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, J. Peters *et al.*, "An algorithmic perspective on imitation learning," *Foundations and Trends® in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

[39] F. Zhang, J. Leitner, Z. Ge, M. Milford, and P. Corke, "Adversarial discriminative sim-to-real transfer of visuo-motor policies," *The International Journal of Robotics Research*, vol. 38, no. 10-11, pp. 1229–1245, 2019.