Conditioning Neural Controllers to Reduce UAV Power Consumption

Leonard Bauersfeld, Davide Scaramuzza

***Abstract*— Recently, learning-based controllers have been shown to push mobile robotic systems to their limits and provide the robustness needed for many real-world applications. However, only classical optimization-based control frameworks offer the inherent flexibility to be dynamically adjusted during execution by, for example, setting target speeds or actuator limits. In the context of energy-efficient operation this flexibility is critical as it enables a control strategy where the robot uses its full performance only in emergency scenarios and, in all other cases, operates with derated actuators to reduce power consumption. We present a framework that adds this functionality to neural controllers by conditioning them on an auxiliary input controlling the maximum thrust available to the controller. This advance is enabled by including a feature-wise linear modulation layer (FiLM). To demonstrate our controller, we use model-free reinforcement-learning to train quadrotor control policies for the task of navigating through a sequence of waypoints in minimum time. Experiments in simulation and in the real-world show that a single control policy can achieve nearly time-optimal flight at maximum thrust as well as in a derated-operations scenario where only 25% thrust is available. At the lowest thrust setting the policy takes 2.25 times longer to complete the waypoint sequence compared to continous operation at maximum thrust, but the average power draw is over 5 times lower compared to the fast run.**

I. Introduction

Recently, learned controllers have become extremely popular in the mobile robotics community due to their success in a variety of complex real-world tasks, such as legged locomotion in challenging environments [1], underground exploration [2] and autonomous drone racing [3]–[5]. In all the aforementioned works, neural-network controllers outperform their classical model-based counterparts both in terms of performance and success rate but they are trained to overfit on a narrowly defined task. Consider, for example, a mobile robot tasked with time-optimal navigation: using model-predictive control (MPC) it would be straightforward to limit the maximum acceleration during deployment by adjusting the actuator constraints inside the model [6]. This allows the controller to maximize the range and endurance of the UAV. However, most neural controllers cannot be regulated and naively adding an additional input to the learned policy may not lead to the desired performance.

The authors are with the Robotics and Perception Group, Department of Informatics, University of Zurich, and Department of Neuroinformatics, University of Zurich and ETH Zurich, Switzerland (`http://rpg.ifi.uzh.ch`). This work was supported by the Swiss National Science Foundation (SNSF) through the National Centre of Competence in Research (NCCR) Robotics, the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No. 871479 (AERIAL-CORE), and the European Research Council (ERC) under grant agreement No. 864042 (AGILEFLIGHT).

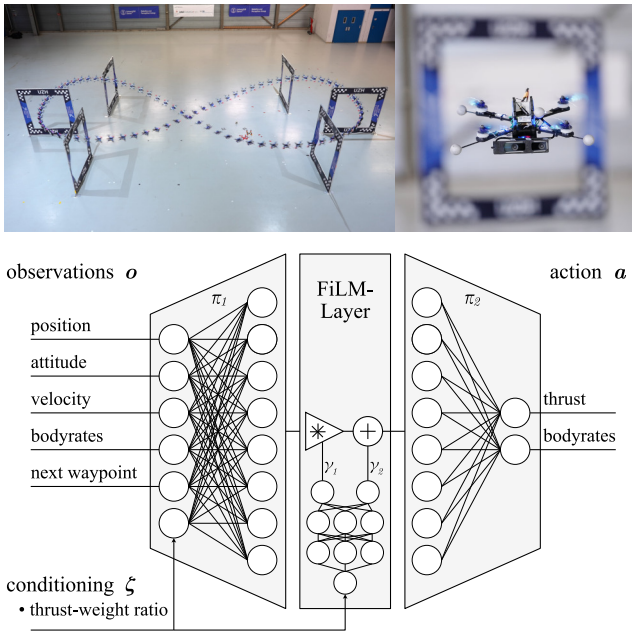


Fig. 1. Conditioning a control policy for agile quadrotor flight on an auxilliary input can be achieved through a FiLM architecture [7]. There, the intermediate activations of a policy that directly maps observations to control commands are linearly transformed based on the conditioning signal supplied by the user. In this work, we study conditioning on the maximum thrust-to-weight ratio available to the controller as this has large impact on the energy efficiency of the UAV.

This paper proposes an approach to alleviate the drawback of rigid task execution of learning-based controllers by conditioning the control policies on an auxiliary input which can influence the neural controller as shown in Fig. 1. This conditioning is crucial to enable energy-efficient operation, as the learned controller enables derating the motor thrust and only utilize the actuators full potential when a high-level controller (human, planner) deems it to be necessary. In our waypoint flight experiments, the conditioned policies decrease the average power draw five times and yield a more than two-fold improvement in energy consumption per pathlength.

Training an embodied agent that can react to user inputs is a difficult endeavor as it requires to learn an entire *distribution* of policies, as opposed to learning a static policy that maps sensory observations to actions. In the context of mobile robotics, to the best of the authors' knowledge, only one prior work [8] exists where conditioning has been applied for three discrete user inputs: a remote-controlled car is trained to either turn left, go straight, or turn right at intersections by using a control network with a shared encoder and three disjoint network heads that are selected based on the operator's input.

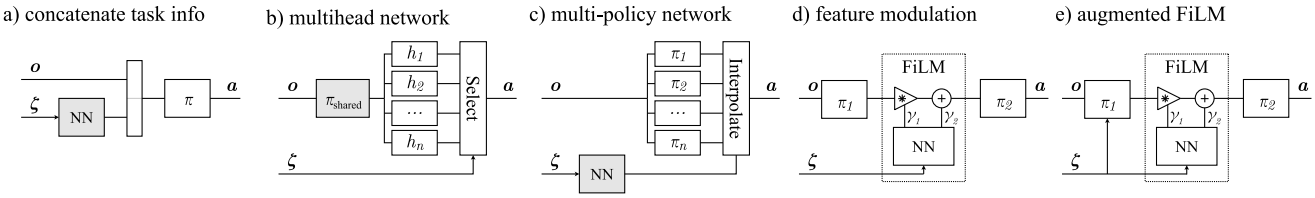


Fig. 2. Overview of the different architectures for conditioning of neural networks commonly found in literature. Boxes in gray represent optional components that can also be replaced with a direct connection. The variable $\boldsymbol{o}$ denotes some vector of observations (e.g. the system state) supplied to the policy. The conditioning signal is denoted by $\boldsymbol{\zeta}$ and the output of the network (e.g. control action) is denoted by $\boldsymbol{a}$.

A. Contribution

We present the first learning-based controller for an autonomous agile quadrotor where the vehicle's maximum thrust-to-weight ratio can be influenced through a continuous conditioning input supplied by a user. This advance is made possible by integrating a modified version of the feature-wise linear modulation layer (FiLM) [7] into the neural network controller, which is trained using model-free reinforcement learning. We demonstrate the applicability of our proposed method in real-world experiments where the continuous user input regulates the desired agility level. We show that policies which are conditioned to fly with variable thrust-to-weight ratios can reduce the power consumption to 1/5 compared to a maximum-thrust policy. When comparing a conditioned policy to a policy that is overfit to fly at maximum thrust, we observe a less than a 2% performance difference between them. Therefore, the conditioning proposed in this work enables energy efficient flight while retaining the ability to push the vehicle to its limit when desired.

II. Related Work

Outside the field of mobile robotics, conditioning neural networks on auxiliary user inputs has been studied in recent years for a variety of applications. However, in the context of this work, it is more informative to compare the works in terms of the architectures they leverage to condition their networks. Figure 2 presents a summary of the common approaches found in literature.

The conceptually simplest approach to conditioning neural networks is shown in Fig. 2 (a) where the conditioning signal $\boldsymbol{\zeta}$ is simply appended to the policy observation $\boldsymbol{o}$ [9].

The multihead (b) and multipolicy (c) architectures shown in Fig. 2 are very similar. In a multihead network all heads operate on the same latent representation produced by a shared encoder π_{shared}. A subsequent multiplexer then selects one of the heads based on the current task signal. The multi-policy approach with a subsequent interpolation layer presented in [10], [11] is very similar. However, no shared encoder is used and the multiplexer is replaced by an interpolation module. The latter enables this approach to handle continuous task signals as the individual control actions by the respective policies are combined smoothly.

A novel approach to conditioning a network that does not require training separate heads nor performs naive input feature concatenation presented in [7]. Their proposed feature-wise linear modulation (FiLM) layer is illustrated in Fig. 2 d). The idea is that a FiLM layer is inserted between two layers of an existing network, effectively splitting the original network into two parts π_1 and π_2. The activations of the first part π_1 are passed through the FiLM layer which applies an affine transform with trainable parameters γ_1 and γ_2. The transformed activations are then used as input to π_2 which generates the final control action. This architecture was originally devised for transforming feature maps of convolutional neural networks [7] but has been applied to robotic manipulation tasks [12], optimal information encoding, and style transfer tasks [13]. As an extension, we also propose an augmented FiLM architecture Fig. 2 e) which also feeds the conditioning signal into the control policy directly.

III. Methodology

In this work we will compare and evaluate the different architectures shown in Fig. 2 for the task of conditioning a quadrotor control on user input. Focusing on the challenging task of agile quadrotor flight, policies are trained using model-free reinforcement learning and directly map a set of observations $\boldsymbol{o}_t$ to low-level control actions $\boldsymbol{a}_t$ [14]. This section first presents the neural controller and then introducing the demonstrators evaluated in this work. The quadrotor dynamics used for simulation and training of the RL agent are described in [15] and left out here for conciseness.

A. Neural Controller

In this work, the task of fast and agile quadrotor flight is defined as follows: Navigate through a sequence of predefined waypoints g_i in minimum time and pass each waypoint within an l_∞ distance less then the dimension of a gate. To accomplish this, the control policy directly maps an observation $\boldsymbol{o}_t$ and a conditioning input ζ_t to an action (control command) $\boldsymbol{a}_t$. The control policies are trained using model-free reinforcement learning (PPO [16]) purely in simulation.

1) Observation and Action Space: At each timestep t the policy has access to an observation $\boldsymbol{o}_t$ from the environment which contains (i) the current robot state, (ii) the relative position to the next waypoint to be passed, and (iii) the current conditioning signal. Specifically, the state consists of the vehicle position $\boldsymbol{p}_{\mathcal{WB}}$, its velocity in body-frame $\boldsymbol{v}_{\mathcal{B}}$ and its attitude. The value network used during training time has access to the same input features as the policy network. In contrast to the policy network, the value network architecture does not contain any FiLM layers.

The control command $\boldsymbol{a}_t$ consists of a desired collective mass-normalized thrust c and a bodyrate setpoint $\boldsymbol{\omega}_{\mathcal{B},\text{ref}}$. Those commands are then tracked by a low-level controller,

which finally controls the motors. In contrast to more abstract control modalities such as linear velocity references, operating on collective thrust and bodyrates has been shown to be well suited for agile learned quadrotor control [14].

2) Conditioning: We compare and evaluate different network architectures (illustrated in Fig. 2) to condition a neural controller for agile quadrotor flight. Specifically, the following architectures are considered:

- *Naive-c* a naive baseline (see Fig. 2 a)) where continuous scalar conditioning signal is concatenated with the observation,
- *Multihead* an architecture (see Fig. 2 b)) with a discrete conditioning signal similar to [8],
- *FiLM-c* a standard FiLM architecture (see Fig. 2 d)) with a continuous scalar conditioning input,
- *FiLM*-c* our augmented FiLM architecture (see Fig. 2 e)) with a continuous scalar conditioning input.

3) Reward Function: We use a dense shaped reward to encode the task of high-speed flight through a set of pre-defined waypoints. The reward r_t at time step t is given by

$$r_t = r_t^{\text{prog}} + r_t^{\text{perc}}(\zeta) - r_t^{\text{twr}}(\zeta) - r_t^{\text{crash}} \,, \tag{1}$$

where r^{prog} rewards progress towards the next waypoint (gate) to be passed [5], $r^{\text{perc}}(\zeta)$ encodes perception awareness by adjusting the vehicle's attitude such that the optical axis of its camera points towards the next waypoint, $r^{\text{twr}}(\zeta)$ is a penalty for violating the user-specified maximum thrust-to-weight ratio, and r^{crash} is a binary penalty that is only active when colliding with a gate or when the platform leaves a pre-defined bounding box, which also ends the episode.

Progress, perception, thrust-to-weight, and collision reward components are formulated as follows:

$$\begin{aligned} r_t^{\text{prog}} &= \lambda_1 \left(d_{\text{Gate}}(t-1) - d_{\text{Gate}}(t) \right) \\ r_t^{\text{perc}}(\zeta) &= \lambda_2 \exp\left(\lambda_3 \cdot \delta_{\text{cam}}(\zeta)^4 \right) \\ r_t^{\text{twr}}(\zeta) &= \max(\lambda_4 \exp\left(\lambda_5 (c_{\text{cmd}} - c_{\text{twr}}(\zeta)) \,/\, c_{\text{max}} \right) - 1, 0) \\ r_t^{\text{crash}} &= -5.0, \quad \text{if } p_z < 0 \text{ or in collision with gate.} \,, \end{aligned} \tag{2}$$

where $d_{\text{Gate}}(t)$ denotes the distance from the quadrotor's center of mass to the center of the next gate, $\delta_{\text{cam}}(\zeta)$ is the angle between the optical axis of the camera and center of the next gate. The parameters c_{cmd}, $c_{\text{twr}}(\zeta)$ and c_{max} are the commanded mass normalized thrust, the current user-specified maximum allowable mass normalized thrust and the maximum mass normalized thrust physically available for the quadrotor, respectively. The hyperparameters $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5$ trade-off objectives regarding perception awareness and thrust-to-weight ratio constraints against progress objectives.

IV. Experiments

Using the training methodology described in the previous section, our experiments aim to answer the following research questions: (i) Which of the architectures (Naive, Multihead, FiLM, our augmented FiLM) presented in the previous section (III-A.2) is best suited for conditioning mobile robot control policies? (ii) Do the results transfer to a real-world quadrotor platform?

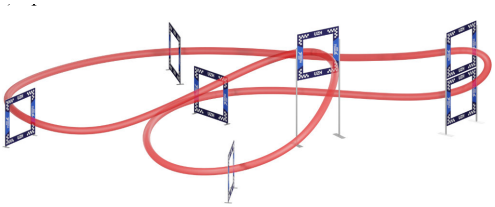


Fig. 3. We evaluate neural policy conditioning on the task of fast waypoint flight. Different approaches for policy conditioning are evaluated on a challenging track, called *Split-S* track. The track spans about 10 m by 16 m.

A. Experimental Setup

The evaluated control policies are all trained for the task of autonomous drone racing. In contrast to prior work tackling autonomous racing, our experiments focus on the racing performance and energy-efficiency when conditioned on a maximum thrust-to-weight ratio.

Our study is performed on the the layout shown in Fig. 3: a complex three-dimensional track layout [4] called *Split-S* track, due to the maneuver required to pass the double gate on the far right.

B. Choice of Network Architecture

In a first set of simulation experiments we aim at identifying the best network architecture for efficient neural policy conditioning in autonomous waypoint flight. Specifically, the policies for all architectures are conditioned on the available thrust-to-weight ratio (TWR), ranging from 1.6 TWR to 4.5 TWR. To have an estimate of the lower bound of the laptime, we also train so-called *fixed-TWR* policies. These policies are trained for a single TWR setting, allowing them to overfit for a specific agility level, which typically results in faster training progress and superior performance.

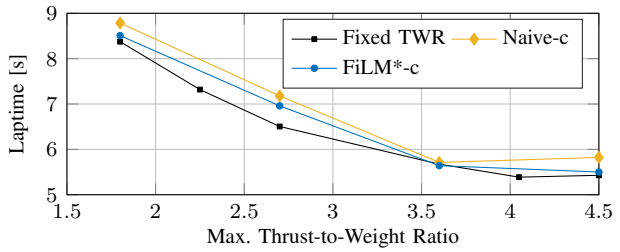
Figure I shows the results of this experiment. The fixed-speed reference is trained for and evaluated at 14 evenly spaced points throughout the TWR interval $[1.6,\ 4.5]$. Each of the conditioned policies are then evaluated at these thrust-to-weight ratio setpoints.

All the architectures we evaluated result in control policies that are able to race at a wide range of thrust-to-weight ratios. However, upon a closer look one can see that the *FiLM*-c* policy leveraging our augmented FiLM architecture outperforms the other approaches in terms of laptime. More importantly, the *FiLM*-c* is less than 0.6 % slower on average than a set of specifically trained fixed-TWR policies. We therefore gain the flexibility to regulate the neural controller during deployment while paying almost no penalty in terms of the optimality (i.e. laptime) of the control policy.

TABLE I

All architectures are able to condition a quadrotor control policy for agile flight on the maximum thrust-to-weight ratio. Our augmented *FiLM*-c* architecture even manages to be within 0.6 % of a fixed-TWR baseline, indicating that one does not have to trade-off control performance for the added flexibility to regulate the controller during deployment.

Architecture	Avg. Rel. Laptime [%]	Max. Rel. Laptime [%]
Naive-c	2.63	3.52
Multihead-d	3.23	4.23
FiLM-c	2.80	3.64
FiLM*-c	**0.54**	**1.62**



Architecture	Avg. Rel. Laptime [%]	Max. Rel. Laptime [%]
Naive-c	4.52	10.43
FiLM*-c	**1.91**	**6.98**

Fig. 4. The plot and table compare the laptimes achieved by the *Naive-c* and *FiLM*-c* approach in real-world experiments on the *Split-S* track. Similar to the simulation results, the *FiLM*-c* architecture outperforms the naive baseline.

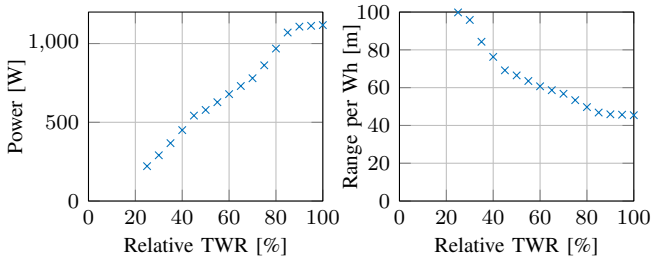


Fig. 5. A single conditioned policy is able to fly the race-track with varying levels of aggressiveness. The left plot shows the average power consumption of the drone during the waypoint flight. The right plot shows the range per Wh, a measure that captures how far a UAV can fly with a single battery.

C. Energy Efficiency & Real-World Experiments

The ablation study conducted to select the optimal architecture was conducted in simulation. In this section we present the transferability of our approach to real-world experiments conducted on an agile quadrotor platform. The platform used for these experiments is shown in 1 and it matches our simulated quadrotor in specifications.

In a first set of experiments we repeat a subset of the experiments presented in IV-B and compare the *FiLM*-c* architecture with both fixed thrust-to-weight ratio policies and the *Naive-c* network (see Fig. 4). Similar to what we observed in simulation, the conditioning with the *FiLM*-c* works well and it outperforms the naive baseline in terms of laptime while being within 2 % of the fixed-TWR reference.

We also evaluate the energy efficiency shown in Fig. 5. At low thrust-to-weight ratio settings the policy is able to fly very efficiently achieving a range of nearly 100 m/Wh. In a setting where energy efficiency is of less importance and performance is key, the same policy can fly much faster at the expense of the efficiency which drops to below 50 m/Wh. At peak performance, the UAV draws up to 1100 W from the battery.

V. Conclusion

This work presented a method to condition learning-based control policies for agile quadrotor flight on an auxiliary input. We evaluated different network architectures that process such user input through simple concatenation, multiple action heads, or by leveraging FiLM layers on the intermediate activations. In an ablation study, in simulation we compared the individual approaches by conditioning control policies on the maximally available thrust-to-weight ratio. Our augmented FiLM architecture achieved the best performance and is less than 0.6 % (in simulation) or 2 % (in the real world) slower than a set of policies trained specifically for one thrust-to-weight ratio. At the same time our conditioned policy can reduce the average power consumption by a factor of 5 compared to a fixed-TWR policy that always utilizes the maximum thrust available. This improves the energy consumption per distance travelled more than two-fold while retaining the ability to use the full actuator potential when required. These findings implicate that we gain the additional flexibility to regulate a neural network controller and do not have to trade-off control performance. Therefore, we believe that this work is an important step in making neural controllers more accessible to deploy for aerial vehicles.

References

[1] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, 2020.

[2] M. Tranzatto, T. Miki, M. Dharmadhikari, L. Bernreiter, M. Kulkarni, F. Mascarich, O. Andersson, S. Khattak, M. Hutter, R. Siegwart, and K. Alexis, "Cerberus in the darpa subterranean challenge," *Science Robotics*, vol. 7, no. 66, p. eabp9742, 2022.

[3] P. Foehn, D. Brescianini, E. Kaufmann, T. Cieslewski, M. Gehrig, M. Muglikar, and D. Scaramuzza, "Alphapilot: Autonomous drone racing," *Auton. Robots*, vol. 46, no. 1, p. 307–320, 2022.

[4] E. Ackerman, "Autonomous Drones Challenge Human Champions in First "Fair" Race," *IEEE Spectrum*.

[5] Y. Song, M. Steinweg, E. Kaufmann, and D. Scaramuzza, "Autonomous drone racing with deep reinforcement learning," *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2021.

[6] A. Romero, S. Sun, P. Foehn, and D. Scaramuzza, "Model predictive contouring control for time-optimal quadrotor flight," *IEEE Transactions on Robotics*, pp. 1–17, 2022.

[7] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[8] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 4693–4700, IEEE, 2018.

[9] M. P. Deisenroth, P. Englert, J. Peters, and D. Fox, "Multi-task policy search for robotics," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3876–3881, 2014.

[10] B. C. Da Silva, G. Konidaris, and A. G. Barto, "Learning parameterized skills," in *Proceedings of the 29th International Conference on International Conference on Machine Learning*, ICML'12, (Madison, WI, USA), p. 1443–1450, Omnipress, 2012.

[11] T. Schaul, D. Horgan, K. Gregor, and D. Silver, "Universal value function approximators," in *Proceedings of the 32nd International Conference on Machine Learning* (F. Bach and D. Blei, eds.), vol. 37 of *Proceedings of Machine Learning Research*, (Lille, France), pp. 1312–1320, PMLR, 2015.

[12] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn, "BC-z: Zero-shot task generalization with robotic imitation learning," in *5th Annual Conference on Robot Learning*, 2021.

[13] A. Dosovitskiy and J. Djolonga, "You only train once: Loss-conditional training of deep networks," in *International Conference on Learning Representations*, 2020.

[14] E. Kaufmann, L. Bauersfeld, and D. Scaramuzza, "A benchmark comparison of learned control policies for agile quadrotor flight," in *2022 International Conference on Robotics and Automation (ICRA)*, IEEE, 2022.

[15] L. Bauersfeld and D. Scaramuzza, "Range, endurance, and optimal speed estimates for multicopters," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2953–2960, 2022.

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv e-prints*, 2017.