Offline Model-Based Reinforcement Learning for Tokamak Control

Ian Char[1], **Joseph Abbate**[2], **László Bardóczi**[3], **Mark D. Boyer**[2], **Youngseog Chung**[1], **Rory Conlin**[4], **Keith Erickson**[2], **Viraj Mehta**[5], **Nathan Richner**[6], **Egemen Kolemen**[2,4], and **Jeff Schneider**[1,5]

[1]Machine Learning Department, Carnegie Mellon University
[2]Princeton Plasma Physics Laboratory
[3]General Atomics
[4]Department of Mechanical and Aerospace Engineering, Princeton University
[5]Robotics Institute, Carnegie Mellon University
[6]Oak Ridge Associated Universities

1 Introduction

Unlocking the potential of nuclear fusion as an energy source would have profound impacts on the world. Nuclear fusion is an attractive energy source since the fuel is abundant, there is no risk of meltdown, and there are no high-level radioactive byproducts [Walker et al., 2020]. Perhaps the most promising technology for harnessing nuclear fusion as a power source is the tokamak: a device that relies on magnetic fields to confine a torus shaped plasma. While strides are being made to prove that net energy output is possible with tokamaks [Meade, 2009], there are still crucial control challenges that exist with these devices [Humphreys et al., 2015].

In this work, we focus on learning controls via offline model-based reinforcement learning for DIII-D, a device operated by General Atomics in San Diego, California. This device has been in operation since 1986, during which there have been over one hundred thousand "shots" (runs of the device). We use approximately 15k shots to learn a dynamics model that can predict the evolution of the plasma subject to different actuator settings. This dynamics model can then be used as a simulator that generates experience for the reinforcement learning algorithm to train on. We apply this method to train a controller that uses DIII-D's eight neutral beams to achieve desired β_N (the normalized ratio between plasma pressure and magnetic pressure) and differential rotation targets. This controller was then evaluated on the DIII-D device. This work marks one of the first efforts for doing feedback control on a tokamak via a reinforcement learning agent that was trained on historical data alone.

2 Related Work

Offline Reinforcement Learning. Unlike standard reinforcement learning (i.e. *online* reinforcement learning) settings, in which agents gather experience through interactions with the environment, offline reinforcement learning attempts to learn a policy through logged, historical interactions from possibly many different policies. This is an attractive setting since many real-world problems will have logged interactions to leverage; however, the restriction of no additional observations usually causes deep reinforcement learning algorithms designed for the online setting to fail because they pick actions that are out of distribution for the dataset.

To combat this problem, offline reinforcement learning algorithms add in extra penalizations to ensure that the optimization procedure chooses actions within the support of the dataset [Kumar et al., 2020, Wu et al., 2019, An et al., 2021]. There have also been a number of successful model-based offline reinforcement algorithms. These algorithms rely on the uncertainty in the dynamics models for penalization [Yu et al., 2020, Kidambi et al., 2020]. In these papers the dynamics models are only accurate for a handful of timesteps (see Appendix G of Yu et al. [2020]). We believe our setting is unique from the environments tested in these papers since we are able to learn a dynamics model that is accurate for entire shots.

Machine Learning and the Physical Sciences workshop, NeurIPS 2022.

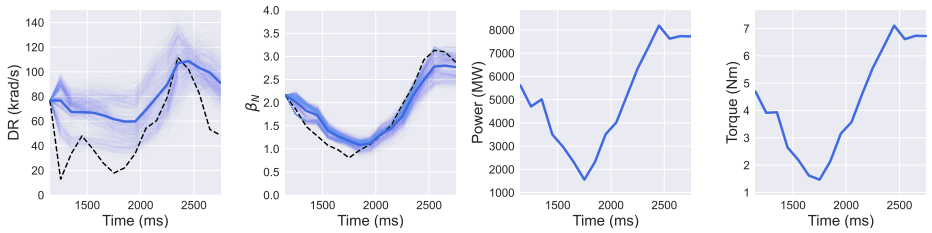

Figure 1: **Replay of Shot 187076.** Here, the model receives the first observations of β_N and DR, but then autoregressively predicts these values into the future. The faded lines are different samples of the neural network parameterization, and the solid lines are the average over the different predictions. Note that there are no faded lines for the power and torque plots since these were the inputs to the model. The black dashed lines are the real observations.

Learning Controls for Tokamaks There has recently been a surge of interest in applying machine learning for controls of tokamaks. Many of these works focus on predicting disruptions for avoidance or safe shutdowns [Fu et al., 2020, Parsons, 2017, Rea et al., 2019, Boyer et al., 2021]; however, in this work, we focus on control during stable operation scenarios. Under this scenario, Char et al. [2019] uses contextual bayesian optimization to find controls that balance increasing β_N and keeping the plasma stable. In terms of modeling dynamics, Abbate et al. [2021] used a convolutional neural network to model the evolution of the plasma's profiles, and this was later used for model predictive control [Abbate et al., 2023].

There have also been some interest in applying reinforcement learning to fusion control. Seo et al. [2021, 2022] learned a dynamics model for the KSTAR tokamak, and used reinforcement learning for tracking several scalar values including β_N; however, they used this policy to generate feedforward controls only. Wakatsuki et al. [2021] trained a policy to do ion temperature gradient control. This policy was both trained and tested in the TOPICS simulator for the JT-60 tokamak. In Degrave et al. [2022], the authors leverage a simulator to learn a controller for the plasma's shape on Tokamak a Configuration Variable [Coda et al., 2019]. Our work differs not only in the goal and actuators used, but also from the fact that we leverage historical data exclusively. The plasma shape can be modeled much more precisely and cheaply than other aspects of the plasma, and for this reason, we focus on learning a simulator from logged data.

3 Method

3.1 The Dynamics Model

States and Actuators. In our work, we assume that the state of the plasma can be characterized by three scalar values and five so-called "profiles". The scalar states consist of the line-averaged electron density, the internal inductance, and β_N, which is the normalized ratio between plasma pressure and magnetic pressure. β_N is an important quantity as it can be used as a rough economic indicator of efficiency. Radial profiles of the plsama rotation, pressure, temperature, electron density, and the safety factor, known as "q", are also used. The q profile is the number of toroidal transits per poloidal transit of a magnetic fieldline. Following Boyer and Chadwick [2021], we reduced the dimensionality of these profile states by using principal component analysis (PCA). We find that we can explain 99% of the variance in the data by using two principal components for the q and pressure profiles, and by using four components for the rest of the profiles. In total, the state space is 19-dimensional (3 dimensions for the scalar states and 16 dimensions for the profile states).

DIII-D has eight neutral beams that inject particles into the core of the plasma. Because two of these beams can be oriented in the opposite toroidal direction as the other six beams, the total power and torque injected, summed across all beams, can be decoupled. In addition to the total power and torque, the model takes as input the current target and density target which are tracked using the loop coil and gas valves respectively. Lastly, we assume that the control of the plasma's shape is good enough that we can treat the shape itself as an actuator. We characterize the plasma shape by four scalar values: the elongation of the plasma, the top triangularity, the bottom triangularity, and the minor radius. In total the actuator space is 8-dimensional.

Dynamics Model and Training. The dynamics model is a simple fully connected neural network that takes in the state of the plasma at time t, the difference in states at time t and $t - \Delta t$, the actuator settings at time t, and the actuator settings at time $t + \Delta t$. The model then predicts the change in

states from time t to time $t + \Delta t$. Here we set $\Delta t = 100$ ms. The dataset consists of 15,534 shots, resulting in 268,702 time steps. As shown by previous works [Chua et al., 2018], it is important to have uncertainty estimates as part of the dynamics model. We incorporate uncertainty by learning a subspace of good network parameters instead of a single good parameterization [Wortsman et al., 2021, Benton et al., 2021]. In particular, we use an ensemble of five networks to learn a simplex of good network parameters following the procedure described in Wortsman et al. [2021]. We repeat this training procedure five times to learn five different simplices. By making a uniform draw from this collection of network parameters, we can sample a new possibility for the dynamics.

To do hyperparameter tuning and evaluation, we take the most recent 10% of shots as our test set. Our tuning procedure targets high explained variance (EV) for one-step predictions. After performing grid search, we settled on a model with 4 hidden layers of 512 units, and a learning rate of 3e-4. When learning the simplex, we encourage diversity by adding a cosine similarity penalizer to the loss function (see Wortsman et al. [2021]), and we find that a coefficient of 5 to this penalty gets the best results. Averaged across five seeds and one hundred samples from the simplex for each seed, these mean predictions achieve an EV score of 0.46 for β_N, 0.43 for the first rotation PCA component, and 0.33 averaged across all output signals. We use the Uncertainty Toolbox library Chung et al. [2021] to evaluate our ensemble's predictive uncertainty. We find that our model tends to be overconfident and achieves a miscalibration area of 0.26 for β_N, 0.25 for the first rotation PCA component, and 0.297 averaged across all predictions. We believe that part of the reason these scores are poor is that the future shots in the test set are meaningfully different. Qualitatively, the model often captures the trends of the state quite well. Figure 1 shows the replay for shot 187076, which was not seen during training. This shot is significantly unique from other shots in the dataset in that there is a drastic drop then increase of both power and torque.

3.2 Learning a Controller

In this work, we learn a controller that sets the total power and torque of the neutral beams in order to achieve a specified target β_N and *differential rotation* (DR). Specifically in this work, DR refers to the difference in the rotation profile at the locations where $q = 1$ and $q = 2$. This is an important quantity of interest as it is hypothesized that higher DR results in a more stable plasma [Bardoczi et al., 2021, Tobias et al., 2016]. However, note that, unlike β_N, this quantity is harder to predict since it relies on accurate predictions of both the rotation and q profiles.

The learned dynamics model is used to form a simulator that can be used to collect experience. Because not all quantities are able to be observed in real time, we form a partially observable Markov decision process (POMDP) where the agent only observes the current and previous values of β_N, DR, the total power injected, and the total torque injected. The controller then specifies the next setting of the total power and torque. The reward function is simply the negative sum of the absolute difference between the current observations and the target values in normalized space. We use states from the historical shots for start states, and we replay the actuator settings for that shot since the reinforcement learning agent only has control over the beams. For each episode in the POMDP, new β_N and DR targets are sampled, and a new parameterization for the dynamics model is drawn.

We use PPO [Schulman et al., 2017] with a policy and value network with 2 hidden layers consisting of 500 units each. Policy evaluation in offline reinforcement learning continues to be a challenging problem. In order to tune hyperparameters and ultimately select the best policy, we decided on a procedure in which we have two dynamics models: one used for training and one used for evaluation. The only difference between the two models is that the model used for evaluation was trained using both the training and testing set. Additionally, for the start state selection and actuator replays, we reserve some historical shots that are used only during the evaluation period.

4 Experiment

We implemented our trained policy into DIII-D's plasma control system (PCS) [Margo et al., 2020]. We use the Keras2C library [Conlin et al., 2021] in order to transfer our policy network to a version of C that does not rely on external libraries. Because the policy only decides the total power and torque, we use the algorithm presented in Boyer et al. [2019], which decides the duty cycles of each of the eight beams to hit the requested power and torque targets. For inputs to the policy, we rely on the profile fitting algorithm [Shousha et al., 2023] and the charge exchange recombination diagnostic system [Gohil et al., 1991]. The policy sends requests for updates to the beams roughly every 10 ms. During our testing session, we were able to test the β_N and DR tracking separately. We used shot

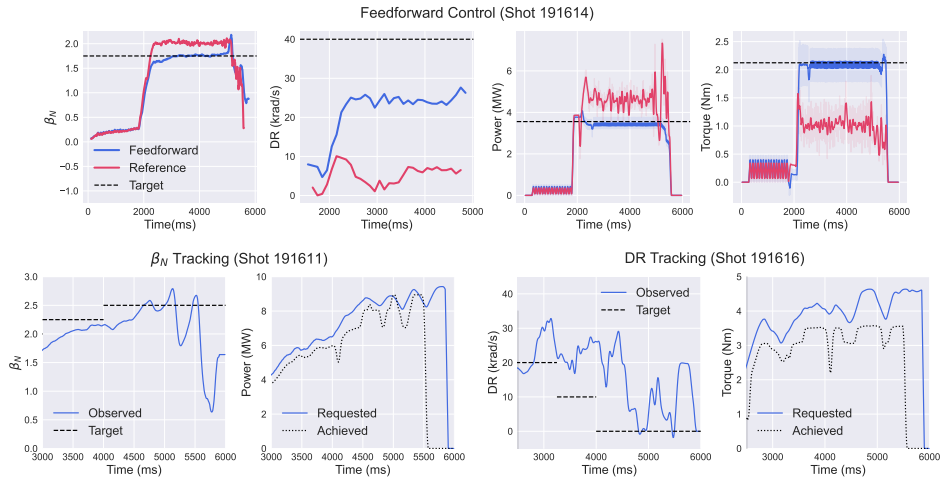

Figure 2: **Experiment Shots.** The top four plots show the β_N, DR, total power, and total torque during both shot 191614 (feedforward control) and the reference shot 164987 (red). These values are smoothed when needed and the original, unsmoothed values are shown as faded lines. The DR values are taken after doing preprocessing and dimensionality reduction via PCA. For the bottom plots, the left pair of plots shows the experiment controlling the power to hit β_N targets, while the right pair of plots shows controlling the torque injected to hit DR targets. In each pair, we show the requested amount of power or torque requested by our controller vs the actual value achieved.

164987 (shots are labeled) as a reference shot, and the actuators besides the beams were mostly used from this shot.

Feedforward Control To disentangle the predictive power of the dynamics model from the policy learning procedure, we used the dynamics model only to prepare feedforward control for the neutral beams. In this feedforward control of the neutral beams, power and torque are ramped up to constant values that are maintained throughout the remainder of the shot. To find these constant power and torque values, a two-dimensional grid search was performed to find the configuration with the best cumulative rewards. For the targets of $\beta_N = 1.75$ and $DR = 40$ krad/s, our optimization procedure found that setting the total power to 3.6 MW and the total torque to 2.1 Nm was best. As shown in Figure 2, the choice of these actuators resulted in hitting the β_N target remarkably well. Although the DR achieved was lower than the target, one can see that it moved closer to the target compared to the reference shot. For reference, DR has a standard deviation of 35.2 and an interquartile range of 47.2 amongst all shots in our dataset, so the error between the target and the value achieved is not as bad as it may appear.

Feedback Control Next, we tested the learned policy's ability to do feedback control. We started by having the controller track increasing β_N targets. Because β_N primarily relies on the power injected, we use the policy to control the injected power only and set the total torque to be 2 Nm throughout the shot. For shot 191611 in Figure 2, one can see that the controller increases the power in order to hit the target values. The last target is overshot slightly and some oscillatory behavior occurs. Upon further inspection, it seems that this was likely due to a problem where the magnitude of change in power was greater than that requested by the controller (4500 ms onward). Near the end of the shot, the plasma begins to disrupt and all control is lost. While there are pre-existing controllers on DIII-D that can hit β_N targets more reliably than this, we still believe that this is a step in the right direction for showing model-based reinforcement learning's value in learning controls.

Unlike β_N tracking, there is no other controller in the DIII-D PCS that specifically tracks the difference in rotation between the $q = 1$ and $q = 2$ surfaces. To test our controller's ability to do so, we set a series of decreasing DR targets for the controller to achieve using only total torque (power is set at a constant value of 5 MW). The controller is unable to track the DR targets nearly as well as the β_N targets. While there are some instances of the policy doing the right thing (e.g. torque is decreased at time 4000 ms time to drop DR to the target), the policy shortly after observes DR dropping too quickly and raises torque back up again (shot 191616 in Figure 2). We believe that this is due to the fact that our dynamics model usually does not predict such severe changes in DR.

5 Discussion

In this work, we presented the first offline model-based reinforcement learning controller for a tokamak. These results show the first steps towards being able to learn tokamak controllers with reinforcement learning using logged data alone. Furthermore, through our feedforward controls, we have demonstrated the predictive ability of our dynamics models for control. Going forward, we believe that there is an opportunity to leverage these models to discover new, performant scenarios for the plasma. We also found that our setting was unlike those reasoned in most model-based offline reinforcement learning papers. The key diference was that a reasonable dynamics model was able to be learned from the data and used to generate experience. We believe there is an opportunity for more research in this setting, i.e. when there is a large, undirected dataset in which one can learn a reasonable dynamics model.

Broader Impact This work further investigates what impacts reinforcement learning can have on tokamak control. While reinforcement learning itself may have both positive and negative impacts to society depending on the application, making nuclear fusion energy a reality would be overwhelmingly positive. Having nuclear fusion as an energy source would help the world cut back on fossil fuels. At the same time, it does not produce the same hazardous biproducts as current nuclear fission reactors.

Acknowledgement

We would like to thank Nikolas Logan, Jayson Barr, and everyone at the DIII-D National Fusion Facility that helped with the preparation and running of this experiment.

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Fusion Energy Sciences, using the DIII-D National Fusion Facility, a DOE Office of Science user facility, under Awards DE-AC02-09CH1146 and DE-FC02-04ER54698. This work was also supported by DE-SC0021414 and DE- SC0021275 (Machine Learning for Real-time Fusion Plasma Behavior Prediction and Manipulation). Additionally, this work is supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE1745016 and DGE2140739. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section ???.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...

(a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] We show the results of our experiment in Section 4.

(b) Did you describe the limitations of your work? [Yes] We show where the method was successful and unsuccessful in Section 4.

(c) Did you discuss any potential negative societal impacts of your work? [No] There are no potential negative impacts we can think of.

(d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...

(a) Did you state the full set of assumptions of all theoretical results? [N/A]

(b) Did you include complete proofs of all theoretical results? [N/A]

3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] The data is not publically available.

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [N/A]

(b) Did you mention the license of the assets? [N/A]

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

References

J. Abbate, R. Conlin, and E. Kolemen. Data-driven profile prediction for diii-d. *Nuclear Fusion*, 61 (4):046027, 2021.

J. Abbate, R. Conlin, K. Erickson, and K. Egemen. Data-driven control in tokamaks. *Journal of Plasma Physics (In Submission.)*, 2023.

G. An, S. Moon, J.-H. Kim, and H. O. Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in neural information processing systems*, 34:7436–7447, 2021.

L. Bardoczi, N. Logan, and E. Strait. Neoclassical tearing mode seeding by nonlinear three-wave interactions in tokamaks. *Physical Review Letters*, 127(5):055002, 2021.

G. Benton, W. Maddox, S. Lotfi, and A. G. G. Wilson. Loss surface simplexes for mode connecting volumes and fast ensembling. In *International Conference on Machine Learning*, pages 769–779. PMLR, 2021.

M. Boyer, K. Erickson, B. Grierson, D. Pace, J. Scoville, J. Rauch, B. Crowley, J. Ferron, S. Haskey, D. Humphreys, et al. Feedback control of stored energy and rotation with variable beam energy and perveance on diii-d. *Nuclear Fusion*, 59(7):076004, 2019.

M. Boyer, C. Rea, and M. Clement. Toward active disruption avoidance via real-time estimation of the safe operating region and disruption proximity in tokamaks. *Nuclear Fusion*, 62(2):026005, 2021.

M. D. Boyer and J. Chadwick. Prediction of electron density and pressure profile shapes on nstx-u using neural networks. *Nuclear Fusion*, 61(4):046024, 2021.

I. Char, Y. Chung, W. Neiswanger, K. Kandasamy, A. O. Nelson, M. Boyer, E. Kolemen, and J. Schneider. Offline contextual bayesian optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

Y. Chung, I. Char, H. Guo, J. Schneider, and W. Neiswanger. Uncertainty toolbox: an open-source library for assessing, visualizing, and improving uncertainty quantification. *arXiv preprint arXiv:2109.10254*, 2021.

S. Coda, M. Agostini, R. Albanese, S. Alberti, E. Alessi, S. Allan, J. Allcock, R. Ambrosino, H. Anand, Y. Andrèbe, et al. Physics research on the tcv tokamak facility: from conventional to alternative scenarios and beyond. *Nuclear Fusion*, 59(11):112023, 2019.

R. Conlin, K. Erickson, J. Abbate, and E. Kolemen. Keras2c: A library for converting keras neural networks to real-time compatible c. *Engineering Applications of Artificial Intelligence*, 100: 104182, 2021.

J. Degrave, F. Felici, J. Buchli, M. Neunert, B. Tracey, F. Carpanese, T. Ewalds, R. Hafner, A. Abdolmaleki, D. de Las Casas, et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022.

Y. Fu, D. Eldon, K. Erickson, K. Kleijwegt, L. Lupin-Jimenez, M. D. Boyer, N. Eidietis, N. Barbour, O. Izacard, and E. Kolemen. Machine learning control for disruption and tearing mode avoidance. *Physics of Plasmas*, 27(2):022501, 2020.

P. Gohil, K. Burrell, R. Groebner, J. Kim, W. Martin, E. McKee, and R. Seraydarian. The charge exchange recombination diagnostic system on the diii-d tokamak. Technical report, General Atomics, 1991.

D. Humphreys, G. Ambrosino, P. de Vries, F. Felici, S. H. Kim, G. Jackson, A. Kallenbach, E. Kolemen, J. Lister, D. Moreau, et al. Novel aspects of plasma control in iter. *Physics of Plasmas*, 22(2): 021806, 2015.

R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33:21810–21823, 2020.

A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

M. Margo, B. Penaflor, H. Shen, J. Ferron, D. Piglowski, P. Nguyen, J. Rauch, M. Clement, A. Battey, and C. Rea. Current state of diii-d plasma control system. *Fusion Engineering and Design*, 150: 111368, 2020.

D. Meade. 50 years of fusion research. *Nuclear Fusion*, 50(1):014004, 2009.

M. S. Parsons. Interpretation of machine-learning-based disruption models for plasma control. *Plasma Physics and Controlled Fusion*, 59(8):085001, 2017.

C. Rea, K. Montes, K. Erickson, R. Granetz, and R. Tinguely. A real-time machine learning-based disruption predictor in diii-d. *Nuclear Fusion*, 59(9):096016, 2019.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

J. Seo, Y.-S. Na, B. Kim, C. Lee, M. Park, S. Park, and Y. Lee. Feedforward beta control in the kstar tokamak by deep reinforcement learning. *Nuclear Fusion*, 61(10):106010, 2021.

J. Seo, Y.-S. Na, B. Kim, C. Lee, M. Park, S. Park, and Y. Lee. Development of an operation trajectory design algorithm for control of multiple 0d parameters using deep reinforcement learning in kstar. *Nuclear Fusion*, 62(8):086049, 2022.

R. Shousha, K. Erickson, and E. Kolemen. Development and experimental demonstration of real-time kinetic profile fitting algorithm for improved equilibrium reconstruction (rtefit) and control on diii-d. *Journal of Plasma Physics (In Submission.)*, 2023.

B. Tobias, M. Chen, I. Classen, C. Domier, R. Fitzpatrick, B. Grierson, N. Luhmann Jr, C. Muscatello, M. Okabayashi, K. Olofsson, et al. Rotation profile flattening and toroidal flow shear reversal due to the coupling of magnetic islands in tokamaks. *Physics of Plasmas*, 23(5):056107, 2016.

T. Wakatsuki, T. Suzuki, N. Oyama, and N. Hayashi. Ion temperature gradient control using reinforcement learning technique. *Nuclear Fusion*, 61(4):046036, 2021.

M. L. Walker, P. De Vries, F. Felici, and E. Schuster. Introduction to tokamak plasma control. In *2020 American Control Conference (ACC)*, pages 2901–2918. IEEE, 2020.

M. Wortsman, M. C. Horton, C. Guestrin, A. Farhadi, and M. Rastegari. Learning neural network subspaces. In *International Conference on Machine Learning*, pages 11217–11227. PMLR, 2021.

Y. Wu, G. Tucker, and O. Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

T. Yu, G. Thomas, L. Yu, S. Ermon, J. Y. Zou, S. Levine, C. Finn, and T. Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.