# Review : Magnetic control of tokamak plasmas through deep reinforcement learning

**J.S. Kim**

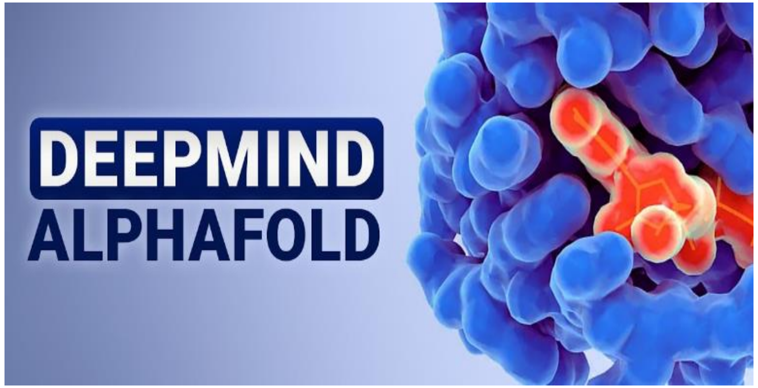
*Department of Nuclear Engineering, Seoul National University, Korea*

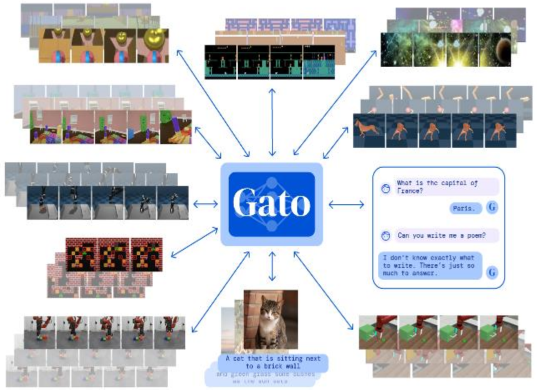
## Contents

- Introduction
- Grad-Shafranov equation
- MPO : Maximum a posteriori policy optimisation
- Experiment
- Discussion
- Appendix

## Introduction

DeepMind





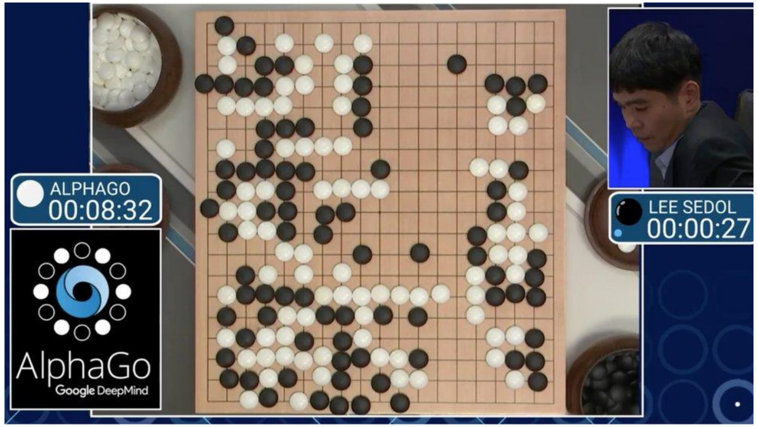
What is the capital of France?

Paris.

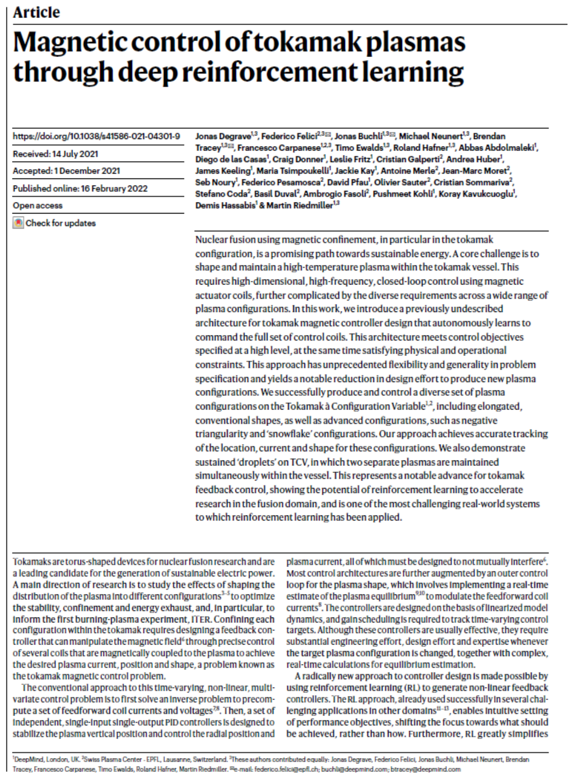
Can you write me a poem?

I don't know exactly what to write. There's just so much to answer.

A cat that is sitting next to a brick wall



# Introduction



Article

Magnetic control of tokamak plasmas through deep reinforcement learning

https://doi.org/10.1038/s41586-021-04301-9

Received: 14 July 2021

Accepted: 1 December 2021

Published online: 16 February 2022

Open access

Check for updates

Jonas Degrave[1,3], Federico Felici[2,3✉], Jonas Buchli[1,3✉], Michael Neunert[1,3], Brendan Tracey[1,3✉], Francesco Carpanese[1,2,3], Timo Ewalds[1,3], Roland Hafner[1,3], Abbas Abdolmaleki[1], Diego de las Casas[1], Craig Donner[1], Leslie Fritz[1], Cristian Galperti[2], Andrea Huber[1], James Keeling[1], Maria Tsimpoukelli[1], Jackie Kay[1], Antoine Merle[2], Jean-Marc Moret[2], Seb Noury[1], Federico Pesamosca[2], David Pfau[1], Olivier Sauter[2], Cristian Sommariva[2], Stefano Coda[2], Basil Duval[2], Ambrogio Fasoli[2], Pushmeet Kohli[1], Koray Kavukcuoglu[1], Demis Hassabis[1] & Martin Riedmiller[1,3]

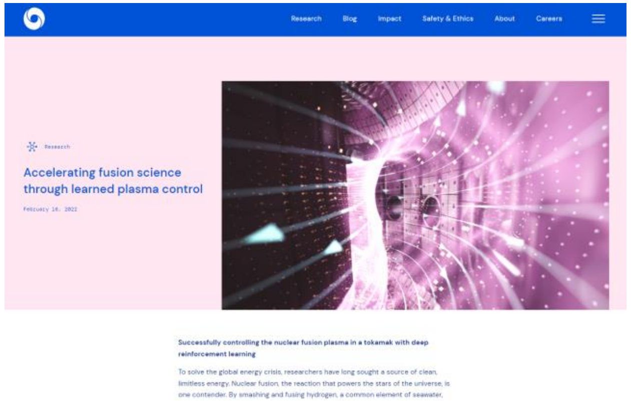
Nuclear fusion using magnetic confinement, in particular in the tokamak configuration, is a promising path towards sustainable energy. A core challenge is to shape and maintain a high-temperature plasma within the tokamak vessel. This requires high-dimensional, high-frequency, closed-loop control using magnetic actuator coils, further complicated by the diverse requirements across a wide range of plasma configurations. In this work, we introduce a previously undescribed architecture for tokamak magnetic controller design that autonomously learns to command the full set of control coils. This architecture meets control objectives specified at a high level, at the same time satisfying physical and operational constraints. This approach has unprecedented flexibility and generality in problem specification and yields a notable reduction in design effort to produce new plasma configurations. We successfully produce and control a diverse set of plasma configurations on the Tokamak à Configuration Variable[1,2], including elongated, conventional shapes, as well as advanced configurations, such as negative triangularity and 'snowflake' configurations. Our approach achieves accurate tracking of the location, current and shape for these configurations. We also demonstrate sustained 'droplets' on TCV, in which two separate plasmas are maintained simultaneously within the vessel. This represents a notable advance for tokamak feedback control, showing the potential of reinforcement learning to accelerate research in the fusion domain, and is one of the most challenging real-world systems to which reinforcement learning has been applied.

Tokamaks are torus-shaped devices for nuclear fusion research and are a leading candidate for the generation of sustainable electric power. A main direction of research is to study the effects of shaping the distribution of the plasma into different configurations[3–5] to optimize the stability, confinement and energy exhaust, and, in particular, to inform the first burning-plasma experiment, ITER. Confining each configuration within the tokamak requires designing a feedback controller that can manipulate the magnetic field[6] through precise control of several coils that are magnetically coupled to the plasma to achieve the desired plasma current, position and shape, a problem known as the tokamak magnetic control problem.

The conventional approach to this time-varying, non-linear, multivariate control problem is to first solve an inverse problem to precompute a set of feedforward coil currents and voltages[7,8]. Then, a set of independent, single-input single-output PID controllers is designed to stabilize the plasma vertical position and control the radial position and plasma current, all of which must be designed to not mutually interfere[6]. Most control architectures are further augmented by an outer control loop for the plasma shape, which involves implementing a real-time estimate of the plasma equilibrium[9,10] to modulate the feedforward coil currents[8]. The controllers are designed on the basis of linearized model dynamics, and gain scheduling is required to track time-varying control targets. Although these controllers are usually effective, they require substantial engineering effort, design effort and expertise whenever the target plasma configuration is changed, together with complex, real-time calculations for equilibrium estimation.

A radically new approach to controller design is made possible by using reinforcement learning (RL) to generate non-linear feedback controllers. The RL approach, already used successfully in several challenging applications in other domains[11–13], enables intuitive setting of performance objectives, shifting the focus towards what should be achieved, rather than how. Furthermore, RL greatly simplifies

[1]DeepMind, London, UK. [2]Swiss Plasma Center - EPFL, Lausanne, Switzerland. [3]These authors contributed equally: Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Martin Riedmiller. ✉e-mail: federico.felici@epfl.ch; buchli@deepmind.com; btracey@deepmind.com

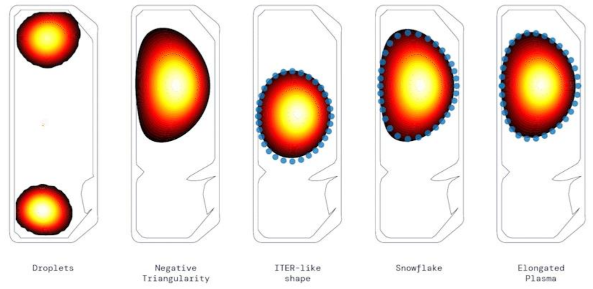


Accelerating fusion science through learned plasma control

February 16, 2022

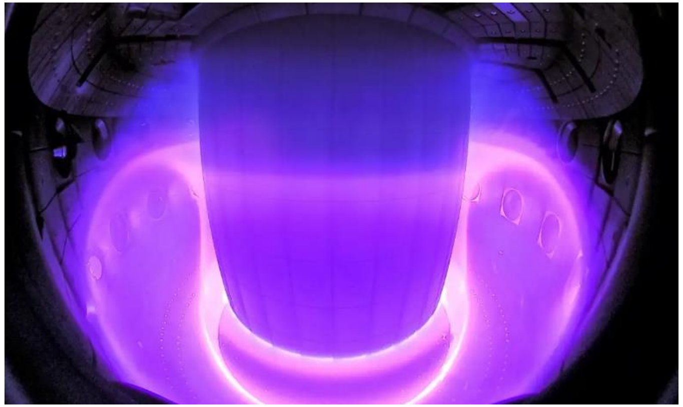
**Successfully controlling the nuclear fusion plasma in a tokamak with deep reinforcement learning**

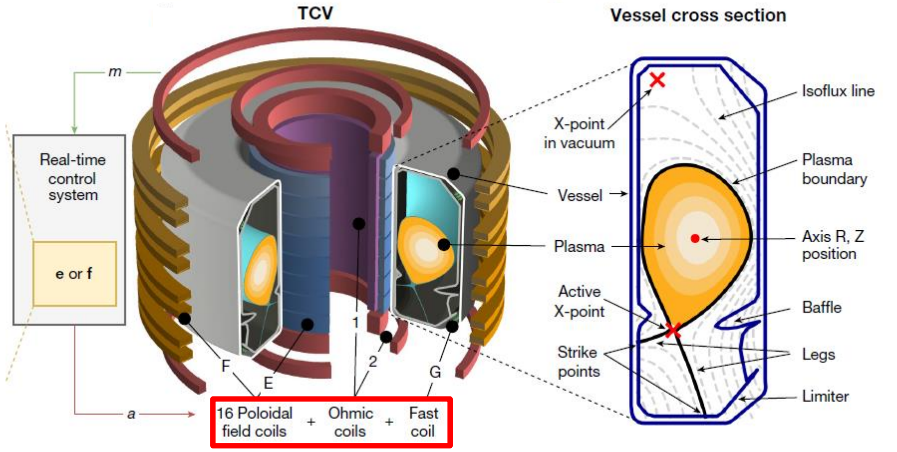
To solve the global energy crisis, researchers have long sought a source of clean, limitless energy. Nuclear fusion, the reaction that powers the stars of the universe, is one contender. By smashing and fusing hydrogen, a common element of seawater,



# Introduction

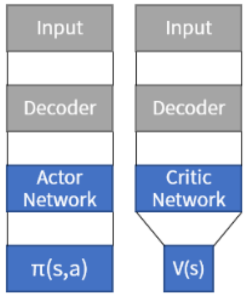
- Setup
  - Control value : TCV coils current(19 coils : Poloidal coils 16 + Ohmic coils 2 + Fast Coil)
  - What to do : Plasma shape control(RL model) + control stability(from reward)
  - Algorithm : Maximize a posteriori policy optimization(MPO, Actor + Critic)
  - Reward : Elongation, LCFS Distance, OH current drift, Plasma current, Radius, Triangularity, x-point
  - Metric : Shape loss + scalar quantities(i.e. plasma current, q, beta, error between reference and the respective estimation from the equilibrium reconstruction)

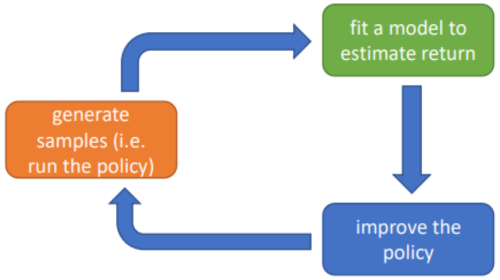


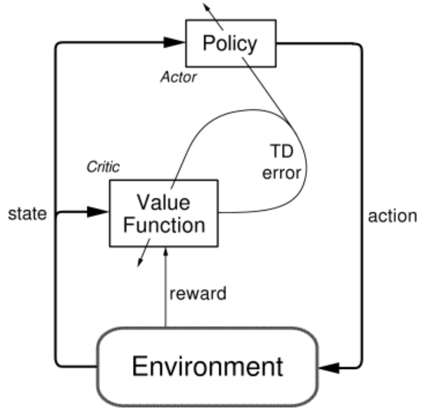


# Introduction

- Setup
  - Actor-Critic : temporal difference version of policy gradient
  - Component
    - Actor network: the actor decides which action should be taken given the state s
    - Critic network : the critic informs the actor how good was the action and how it should adjust

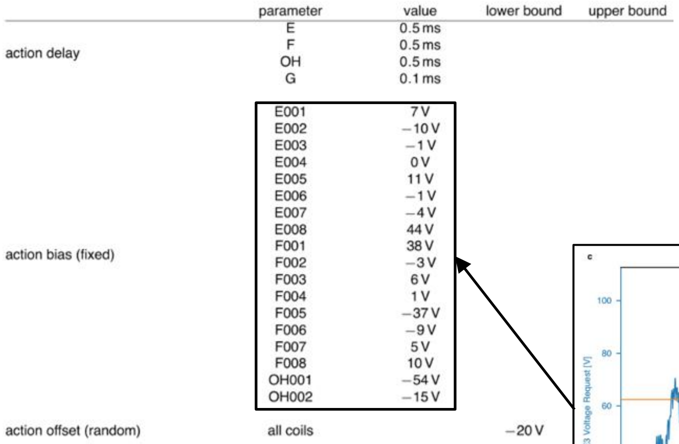



fit a model to estimate return
improve the policy
generate samples (i.e. run the policy)

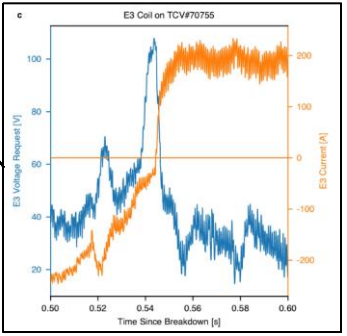


# Introduction

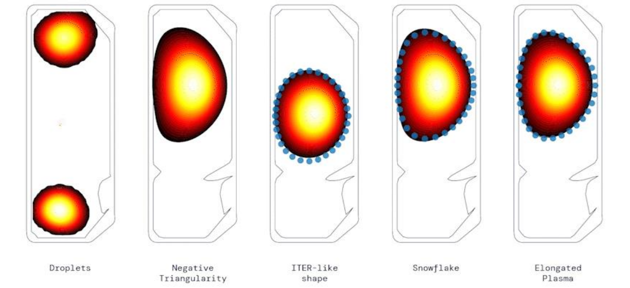
Input Data : TCV coils current(19)



|  | parameter | value | lower bound | upper bound |
|---|---|---|---|---|
| action delay | E | 0.5 ms |  |  |
|  | F | 0.5 ms |  |  |
|  | OH | 0.5 ms |  |  |
|  | G | 0.1 ms |  |  |
| action bias (fixed) | E001 | 7 V |  |  |
|  | E002 | −10 V |  |  |
|  | E003 | −1 V |  |  |
|  | E004 | 0 V |  |  |
|  | E005 | 11 V |  |  |
|  | E006 | −1 V |  |  |
|  | E007 | −4 V |  |  |
|  | E008 | 44 V |  |  |
|  | F001 | 38 V |  |  |
|  | F002 | −3 V |  |  |
|  | F003 | 6 V |  |  |
|  | F004 | 1 V |  |  |
|  | F005 | −37 V |  |  |
|  | F006 | −9 V |  |  |
|  | F007 | 5 V |  |  |
|  | F008 | 10 V |  |  |
|  | OH001 | −54 V |  |  |
|  | OH002 | −15 V |  |  |
| action offset (random) | all coils |  | −20 V |  |



Output : Action distribution for each coil with Gaussian Distribution



32 equiangular points around the LCFS and canonicalize with splines to 128 equidistant points

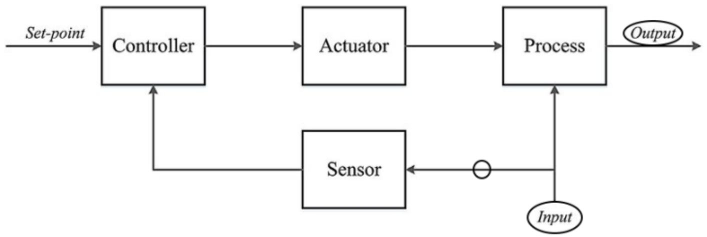
## Introduction

- Conventional approach
  - **Feedforward control** : a system which passes the signal to some external load
  - Process
    - Solve an inverse problem to precompute a set of feedforward coil currents and voltages
    - PID controllers(linearized model dynamic) : stabilize the plasma vertical position and control the radial position and current
    - PID controllers do not mutually interfere

- New approach
  - **Feedback control** : a system where the output depends on the generated feedback signal
  - **Reinforcement Learning** : generate non-linear feedback controllers + policy-based method
    - Single computationally inexpensive controller(Neural Network) replaces the nested control architecture
    - Reduction of the controller development cycle
    - Environment : GS solver(simulator) or surrogate model using classical ML

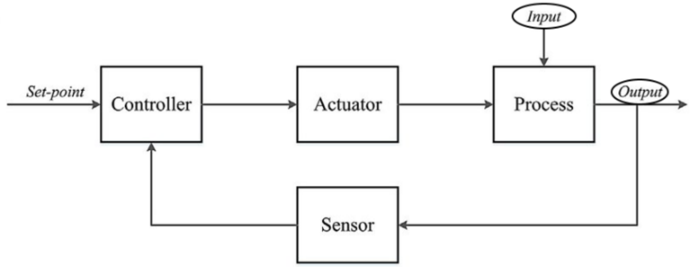
# Introduction

- **Feed-forward**



  - Model-based prediction of input
  - Ideally consists of exact inverse model of the process
  - Effects of disturbance or command input must be predictable
  - May not generalize to other conditions
  - Will not be accurate if the system changes
  - Feedforward component provides rapid response

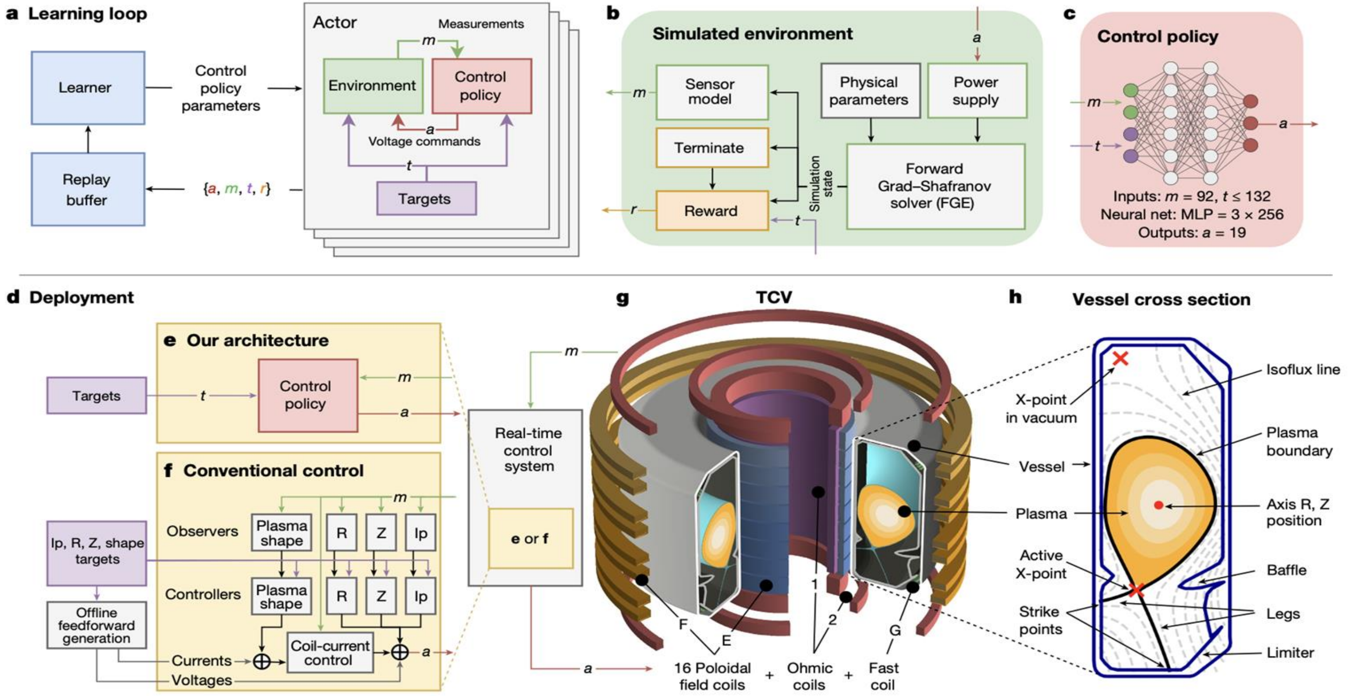
- **Feed-back**



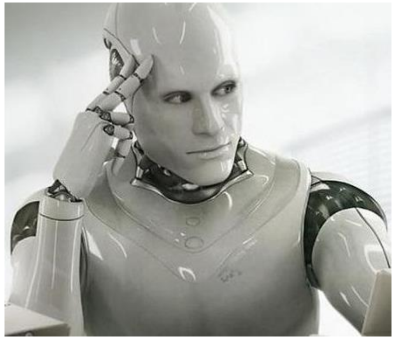
  - Automatically compensates for disturbances (=controller acts on error)
  - Automatically follows change in desired state
  - Reactive / Error - driven
  - Can be very simple
  - Can improve undesirable properties of system
  - Feedback component provides response accurately via compensating for errors in the model

## Introduction

- **RL based controller design architecture**

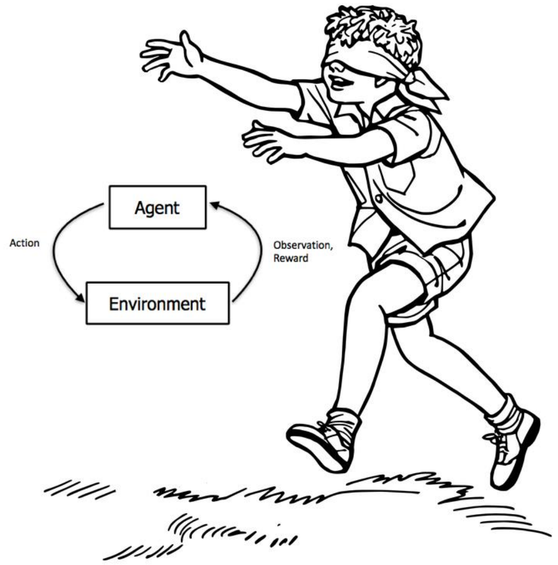


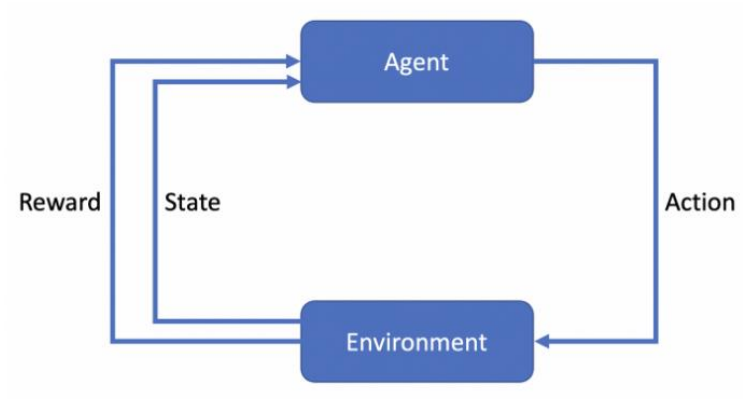
## Introduction



What is Deep Reinforcement Learning?

# Introduction : Reinforcement Learning

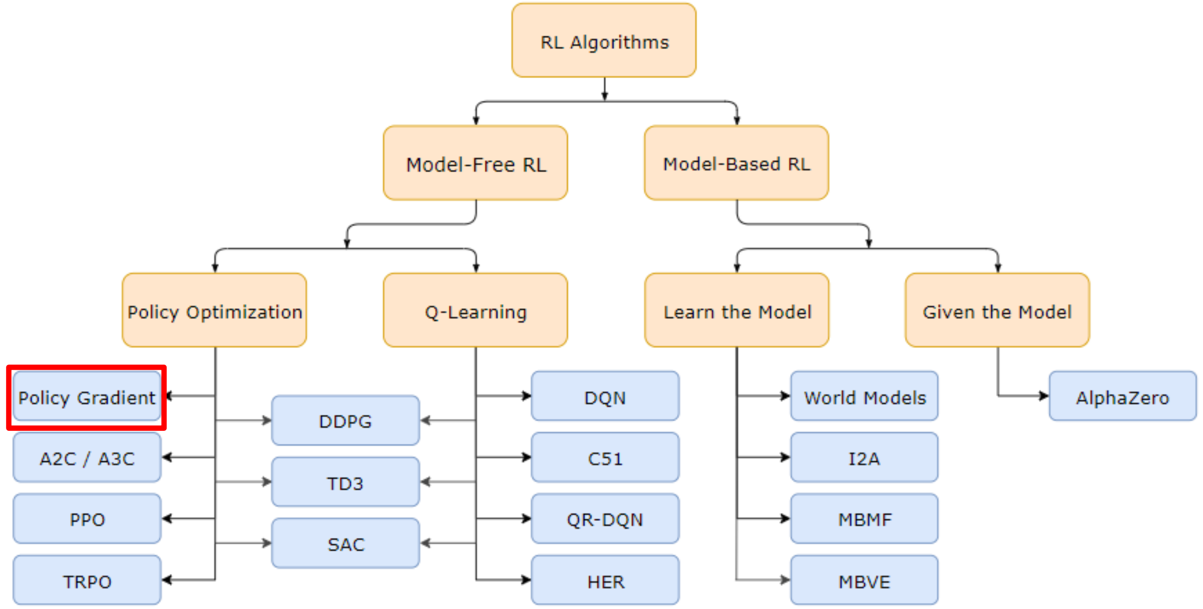




## Introduction : Reinforcement Learning

- Fundamental of Reinforcement Learning
    - Definition : ML technique that enables an agent to learn in an interactive environment by trial and error using feedback from its own actions and experience → **Learn the optimal policy which maximizes reward**
    - Element
        - Agent : Object that takes decisions based on the rewards and punishment
        - Environment : Physical world in which the agent interacts ( GS solver)
        - State : Current situation of the agent
        - Reward : Feedback from the environment
        - Action : Mechanism by which the agent transitions between states of the environment
        - Value : Future reward that an agent would receive by taking an action in a particular state

# Introduction : Reinforcement Learning

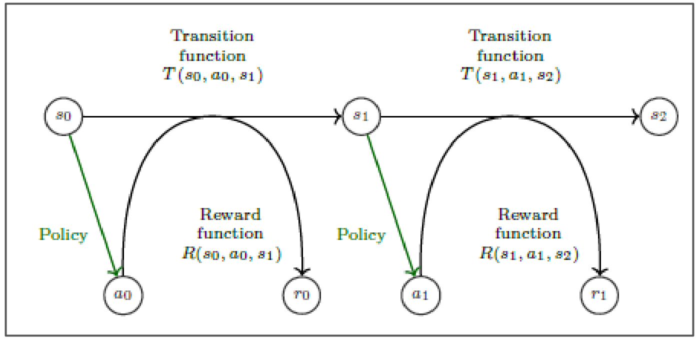


## Introduction : Reinforcement Learning

- Markov Process

**“The future is independent of the past given the present”**

$$P[S_{t+1} \mid S_t] = P[S_{t+1} \mid S_1, \ldots\ldots, S_t]$$



Transition function $T(s_0, a_0, s_1)$
Transition function $T(s_1, a_1, s_2)$
$s_0$
$s_1$
$s_2$
Policy
Reward function $R(s_0, a_0, s_1)$
Policy
Reward function $R(s_1, a_1, s_2)$
$a_0$
$r_0$
$a_1$
$r_1$

**Definition 3.1.** A discrete time stochastic control process is Markovian (i.e., it has the Markov property) if

- $\mathbb{P}(\omega_{t+1} \mid \omega_t, a_t) = \mathbb{P}(\omega_{t+1} \mid \omega_t, a_t, \ldots, \omega_0, a_0)$, and
- $\mathbb{P}(r_t \mid \omega_t, a_t) = \mathbb{P}(r_t \mid \omega_t, a_t, \ldots, \omega_0, a_0)$.

**Definition 3.2.** An MDP is a 5-tuple $(\mathcal{S}, \mathcal{A}, T, R, \gamma)$ where:

- $\mathcal{S}$ is the state space,
- $\mathcal{A}$ is the action space,
- $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition function (set of conditional transition probabilities between states),
- $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathcal{R}$ is the reward function, where $\mathcal{R}$ is a continuous set of possible rewards in a range $R_{\max} \in \mathbb{R}^+$ (e.g., $[0, R_{\max}]$),
- $\gamma \in [0, 1)$ is the discount factor.

## Introduction : Reinforcement Learning

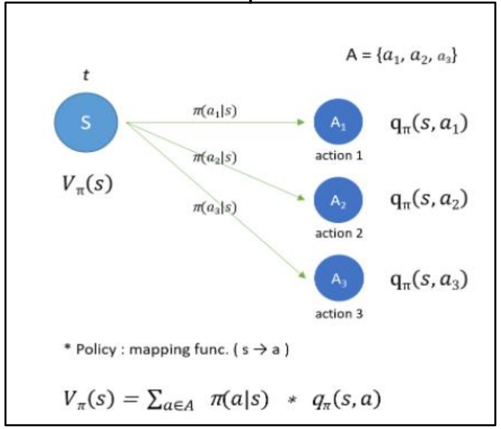
- Policy gradient methods
  - Policy : **Probability** of **choosing action a** given the **state s**
  - Objective : to **find the optimal policy** which achieves **optimal reward**
  - Reward function

$$J(\theta) = \sum_{s\in S} d^\pi(s)V^\pi(s) = \sum_{s\in S} d^\pi(s)\sum_{a\in A}\pi_\theta(a|s)Q^\pi(s,a)$$

  - How to optimize

$$\boxed{\begin{aligned}\nabla_\theta J(\theta) &= \nabla_\theta \sum_{s\in S} d^\pi(s)\sum_{a\in A} Q^\pi(s,a)\pi_\theta(a|s)\\ &\propto \sum_{s\in S} d^\pi(s)\sum_{a\in A} Q^\pi(s,a)\nabla_\theta\pi_\theta(a|s)\end{aligned}} \longrightarrow \boxed{\nabla_\theta J(\theta) = \mathbb{E}_\pi[Q^\pi(s,a)\nabla_\theta \ln \pi_\theta(a|s)]}$$

$$s \xrightarrow{a\sim\pi_\theta(.|s)} s' \xrightarrow{a\sim\pi_\theta(.|s')} s'' \xrightarrow{a\sim\pi_\theta(.|s'')} \dots$$

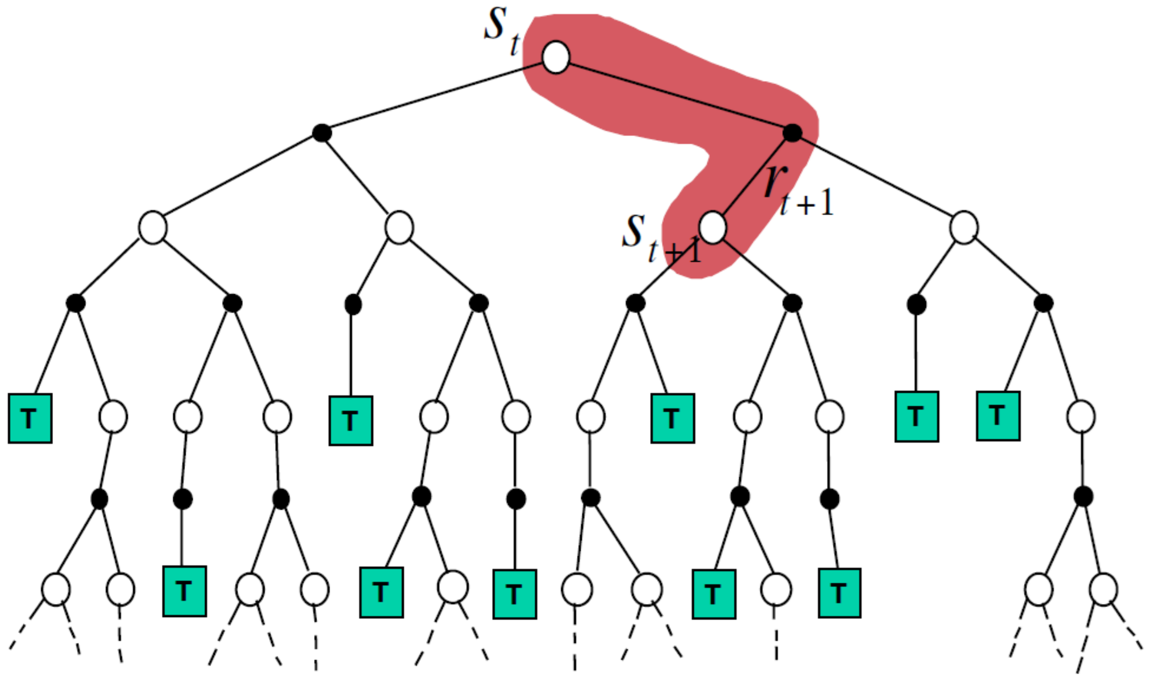


* Policy : mapping func. ( s → a )

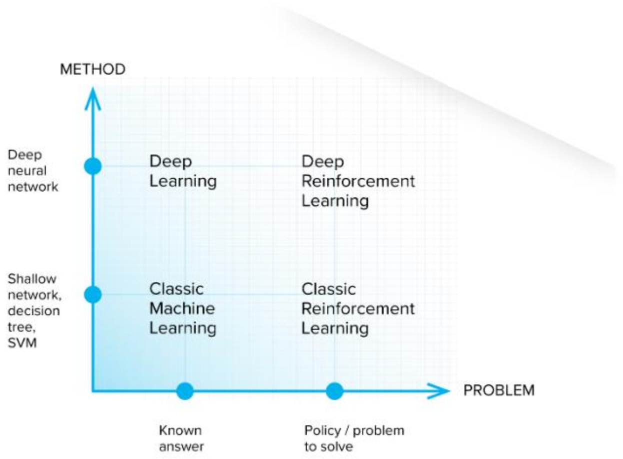
$$V_\pi(s) = \sum_{a\in A} \pi(a|s) * q_\pi(s,a)$$

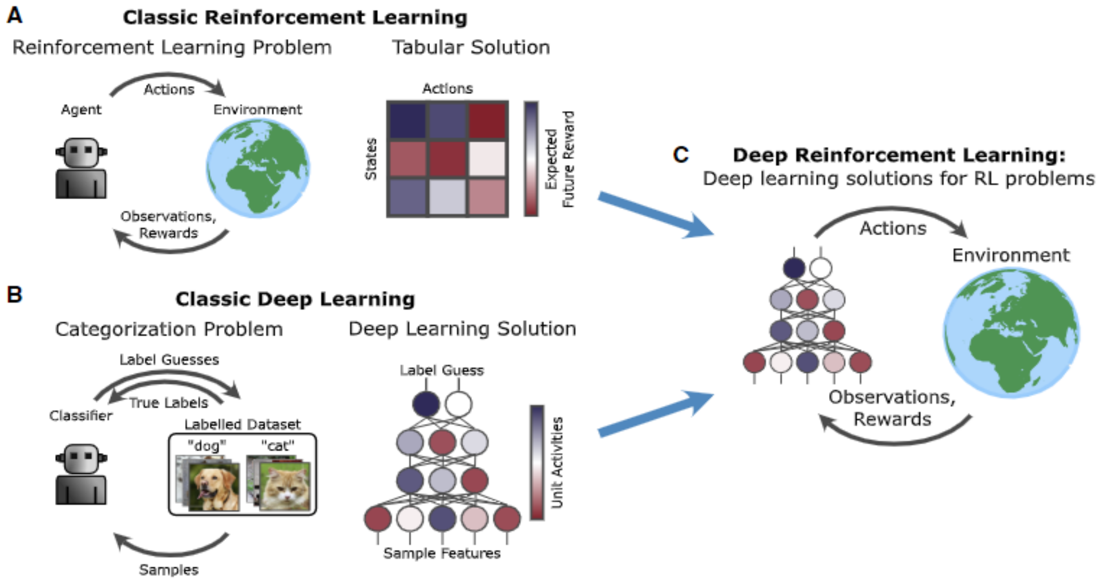
## Introduction : Reinforcement Learning

$$V(S_t) \leftarrow V(S_t) + \alpha\left(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right)$$



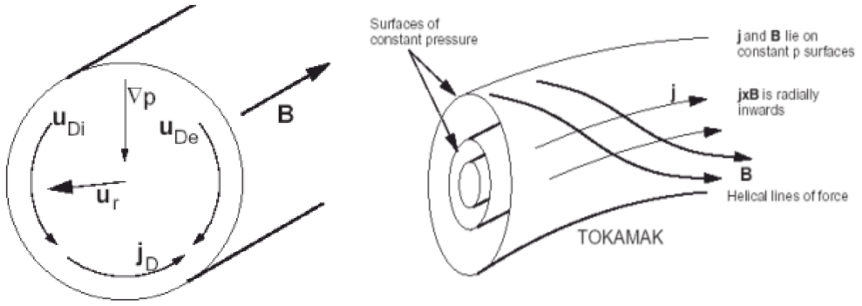
## Introduction : Reinforcement Learning


METHOD
Deep neural network
Shallow network, decision tree, SVM
Deep Learning
Deep Reinforcement Learning
Classic Machine Learning
Classic Reinforcement Learning
PROBLEM
Known answer
Policy / problem to solve



## Grad-Shafranov equation

- MHD equilibrium in Tokamak



$$\vec{J} \times \vec{B} = \nabla P$$

$$\nabla \times \vec{B} = \mu_o \vec{J}$$

$$\nabla \cdot \vec{B} = 0$$

- Grad-Shafranov equation

$$F(\psi) = RB_\phi$$

$$J_\phi(R,Z) = R\frac{dP}{d\psi} + \frac{F(\psi)}{\mu_o R}\frac{dF(\psi)}{d\psi}$$

$$\Delta^*\psi = R\frac{\partial}{\partial R}\left(\frac{1}{R}\frac{\partial\psi}{\partial R}\right) + \frac{\partial^2\psi}{\partial R^2}$$

$$\Downarrow$$

$$\Delta^*\psi = -\mu_o R^2\frac{dP}{d\psi} - F\frac{dF}{d\psi}$$

## Grad-Shafranov equation

- General vacuum solution
  - Condition : zero pressure + zero current outside of the plasma(p=F=0)

$$\Delta^* \psi = 0 \longrightarrow \psi_{vac} = \sum_n c_n R \cos(k_n Z)\left(J_1(k_n R) + d_n Y_1(k_n R)\right)$$

- Vacuum solution of a circular current loop
  - Green's function of the Grad-shafranov operator is given by

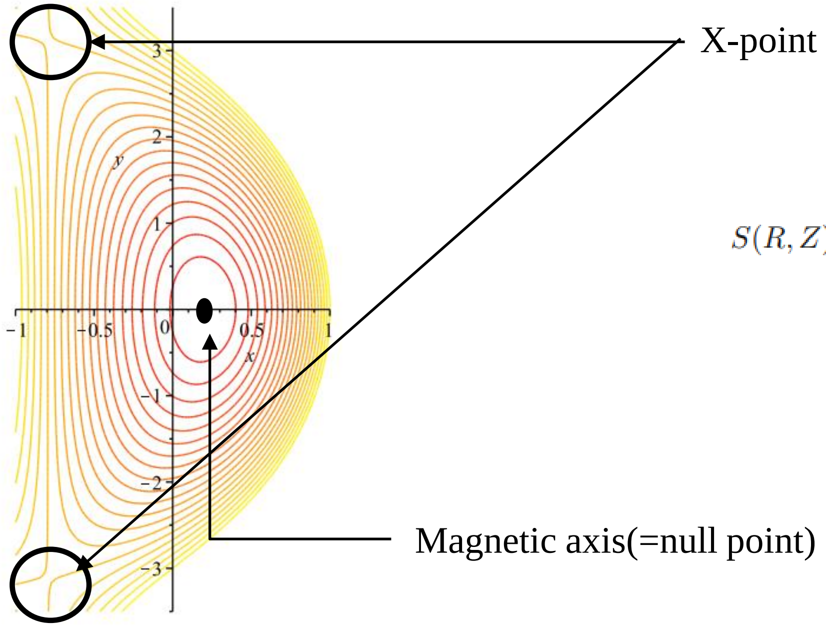
$$G(R,Z;R_0,Z_0) = -\frac{1}{2\pi}\sqrt{RR_0}\frac{1}{k}\left[(2-k^2)K(k) - 2E(k)\right]$$

Elliptic integrals of the 1st and 2nd kind

  - The magnetic flux due to the circular current loop is given by

$$\frac{\psi(R,\theta)}{R} = SR\frac{4(2-k(R,\theta)^2)K(k(R,\theta)^2) - 2E(k(R,\theta)^2)}{\sqrt{a^2+R^2+2aR\sin(\theta)}} \qquad k(R,\theta) = \frac{4aR\sin(\theta)}{a^2+R^2+2aR\sin(\theta)}$$

## Grad-Shafranov equation



$$S(R,Z) \equiv \left(\frac{\partial^2\psi}{\partial R^2}\right)\left(\frac{\partial^2\psi}{\partial Z^2}\right) - \left(\frac{\partial^2\psi}{\partial R \partial Z}\right)^2 \quad \begin{cases} S(R,Z) > 0 : \text{magnetic axis} \\ S(R,Z) < 0 : \text{x-point} \end{cases}$$

$$|\nabla\psi|^2 = 0$$

## Grad-Shafranov equation

- Boundary conditions
  - Fixed boundary : the plasma-vacuum boundary is replaced by the surface of a perfect conductor and only the plasma region is calculated, using as a plasma boundary condition J = 0 such that at this boundary flux = 0
  - Free boundary : the value of magnetic flux is specified on a closed curve, usually in the vacuum region.
  - Constrained boundary : the equilibrium is solved for a given external magnetic field and some constraint such as a contact point with the domain boundary

## Grad-Shafranov equation

- Free boundary Plasma Equilibrium solution

$$\boxed{\begin{aligned} \triangle^*\psi(R,Z) &= -\mu_0 R J_{\phi,pl}(R,Z), \\ J_{\phi,pl}(R,Z) &= Rp'(\psi) + \frac{F(\psi)F(\psi)'}{\mu_0 R} \end{aligned}} \longrightarrow \boxed{\begin{aligned} \triangle^*\psi(R,Z) &= -\mu_0 R J_{\phi}(R,Z), \\ J_{\phi}(R,Z) &= J_{\phi,pl}(R,Z) + J_{\phi,cond}(R,Z) \end{aligned}}$$

Add boundary condition from current density of conductor and plasma

Toroidal current density for conductor and plasma

$$\left\{\begin{aligned} J_{\phi,cond}(R,Z) &= \sum_{k=1}^{N_{cond}} J_{cond,k}(R,Z), \\ J_{cond,k}(R,Z) &= \begin{cases} I_{cond,k}/S_k & \text{if } (R,Z)\in\Omega_{cond,k} \\ 0 & \text{otherwise ,} \end{cases} \\ J_{\phi,pl}(R,Z) &= \begin{cases} \lambda\left[\beta_0 \dfrac{R}{R_{geo}}\tilde{j}_{\mathrm{P}}(\psi,\psi_a,\psi_b) + (1-\beta_0)\dfrac{R_{geo}}{R}\tilde{j}_{\mathrm{F}}(\psi,\psi_a,\psi_b)\right] \\ 0 \end{cases} \end{aligned}\right. \longrightarrow$$

$$\boxed{\begin{aligned} &\psi^{(n)}_{bndry}(R_b,Z_b) \\ &= \int_{\Omega'_{pl}} G(R_b,Z_b;R',Z')J^{(n)}_{\phi,pl}(R',Z')d\Omega'_{pl} \\ &+ \int_{\Omega'_{cond}} G(R_b,Z_b;R',Z')J^{(n)}_{\phi,cond}(R',Z')d\Omega'_{cond} \end{aligned}}$$

## Maximum a posteriori policy optimisation

- Conventional formulations
  - Aim to find a trajectory that maximizes expected reward

- Inference formulations
  - starts from a prior distribution over trajectories
  - condition a desired outcome (ex : achieving a goal state)
  - estimate the posteriori distribution over trajectories consistent with this outcome

## Maximum a posteriori policy optimisation

- Why use MPO?
  - Using MPO can overcome the paucity of data by optimizing the policy due to EM algorithm
  - MPO supports data collection across distributed parallel streams and learns in a data-efficiency way

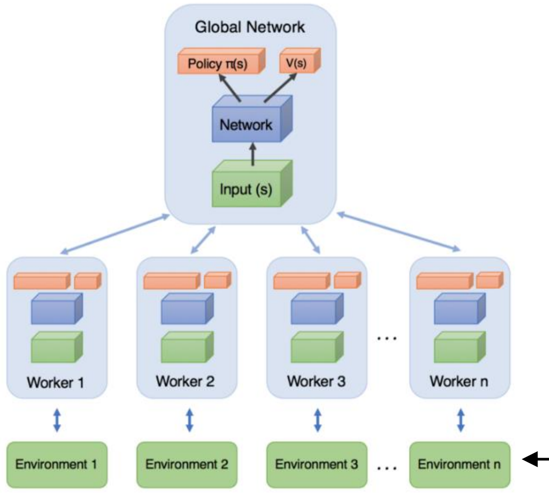
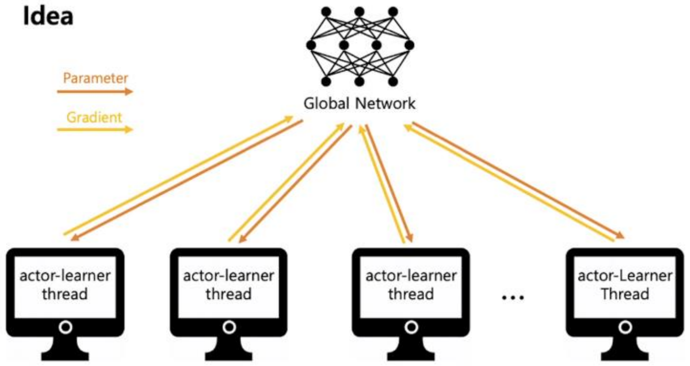


Fig. Training process of A3C models using distributed parallel streams



$$\boxed{\Delta^*\psi = -\mu_o R^2 \frac{dP}{d\psi} - F\frac{dF}{d\psi}}$$

## Maximum a posteriori policy optimisation

- Likelihood and Posterior
  - Likelihood : Probability of data D given the parameter $\theta$
  - Prior : Distribution over parameter $\theta$
  - Posterior : Probability of the parameter $\theta$ given the data D

$$\theta_{MLE} = \arg\max_{\theta} \prod_{i=1}^{n} f(X_i \mid \theta)$$

$$P(\theta \mid D) = \frac{P(D \mid \theta)\, P(\theta)}{P(D)} \longleftarrow P(D) = \int P(D \mid \theta) P(\theta)\, d\theta$$

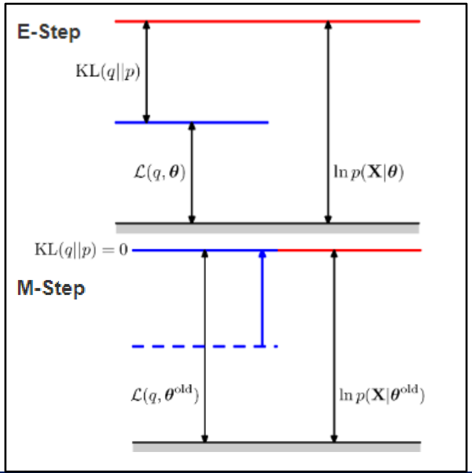
- Maximum a posteriori
  - The purpose of machine learning or deep learning : Maximum likelihood estimator
  - Maximizing likelihood * prior = Maximizing posterior

$$\hat{\theta}_{MAP} = argmax_{\theta} p(\theta|x) = argmax_{\theta} \frac{p(x|\theta)p(\theta)}{p(x)} = argmax_{\theta} p(x|\theta)p(\theta)$$

# Maximum a posteriori policy optimisation

$$\log p_\pi(O=1) = \log \int p_\pi(\tau) p(O=1|\tau) d\tau \geq \int q(\tau) \big[ \log p(O=1|\tau) + \log \frac{p_\pi(\tau)}{q(\tau)} \big] d\tau$$
$$= \mathbb{E}_q \Big[ \sum_t r_t / \alpha \Big] - \mathrm{KL}\Big( q(\tau) || p_\pi(\tau) \Big) = \mathcal{J}(q, \pi),$$

Optimization : EM algorithm


E-Step
$\mathrm{KL}(q||p)$
$\mathcal{L}(q, \boldsymbol{\theta})$
$\ln p(\mathbf{X}|\boldsymbol{\theta})$
$\mathrm{KL}(q||p) = 0$
M-Step
$\mathcal{L}(q, \boldsymbol{\theta}^{\text{old}})$
$\ln p(\mathbf{X}|\boldsymbol{\theta}^{\text{old}})$

RL : Policy Gradient Method

$$\nabla_\theta J(\theta) = \nabla_\theta \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} Q^\pi(s,a) \pi_\theta(a|s)$$
$$\propto \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} Q^\pi(s,a) \nabla_\theta \pi_\theta(a|s)$$

Approach : Maximum a posteriori

Estimated parameter

Log prior

$$\theta_{\text{MAP}} = \underset{\theta}{\operatorname{argmax}} \left( \log(g(\theta)) + \sum_{i=1}^{n} \log(f(X_i|\theta)) \right)$$

Chose the value of theta that maximizes:

Sum of log likelihood

## Variational EM algorithm

- Variational EM algorithm
  - Variational Inference : Approximation of the probabilistic distribution q(z) which is most similar to p(z)
  - Unlike sampling-based methods, Variational inference will almost never find the globally optimal solution
  - If they have converged, we can obtain bounds on their accuracy
  - Metric for the measuring difference between two probabilistic distribution : KL Divergence

- Kullback-Leibler Divergence(KL Divergence)
  - Metric for the measuring difference between two probabilistic distribution

$$D_{KL}(P||Q) = E_{X\sim P}\left[\log\frac{P(x)}{Q(x)}\right] = E_{X\sim P}\left[-\log\frac{Q(x)}{P(x)}\right]$$

## Variational EM algorithm

- Properties of KL Divergence

$$
\begin{aligned}
D_{KL}(P||Q) &= -\sum_x P(x)\log\left(\frac{Q(x)}{P(x)}\right) \\
&= -\sum_x P(x)\{\log Q(x) - \log P(x)\} \\
&= -\sum_x \{P(x)\log Q(x) - P(x)\log P(x)\} \\
&= -\sum_x P(x)\log Q(x) + \sum_x P(x)\log P(x) \\
&= H(P,Q) - H(P)
\end{aligned}
$$

$$D_{KL}(q||p) > 0 \; for \; all \; q \; and \; p.$$

$$D_{KL}(q||p) = 0 \; if \; and \; only \; if \; q = p.$$

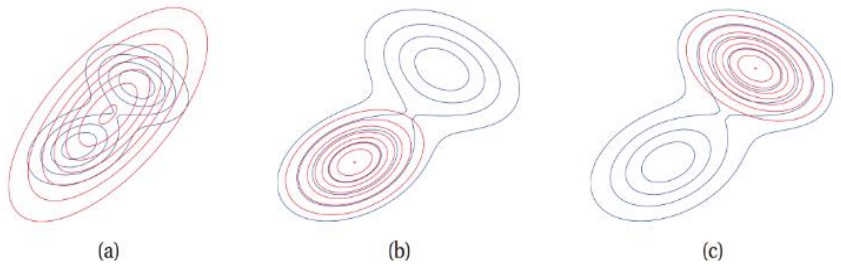
- Else
  - Convex functional to q(z)
  - Jensen’s inequality satisfied

$$E[f(x)] \geq f(E[x])$$

# Variational EM algorithm

- Idea of EM algorithm
  - We want to find out the specific probabilistic distribution q(z) which is mostly similar to the posterior probabilistic distribution p(z|x)
  - To find out q(z), use KL divergence and optimize the parameters of q(z) and p(z|x) to minimize the KL divergence

- Process
  - E-step : minimize KL divergence of q(z) and p(z|x)
  - M-step : maximize lower bound of log p(x)



Fitting a unimodal approximating distribution q(red) to a multimodal p(blue)
(a). use KL(p||q), (b) and (c). use KL(q||p)

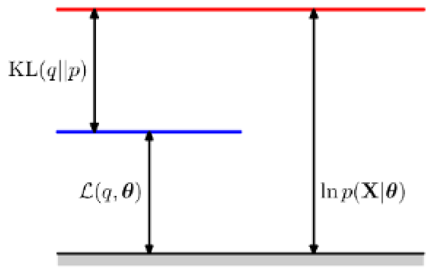
## Variational EM algorithm

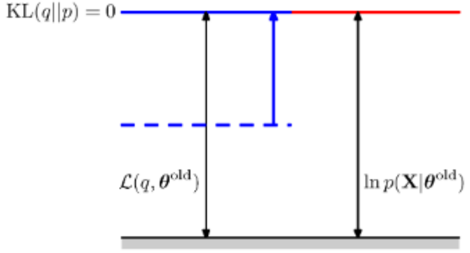
- Process
  - E-step : minimize KL divergence of q(z) and p(z|x)

$$\boxed{\ln \sum_{\mathbf{Z}} p(\mathbf{X},\mathbf{Z}|\theta) = \ln \sum_{\mathbf{Z}} q(\mathbf{Z}) \frac{p(\mathbf{X},\mathbf{Z}|\theta)}{q(\mathbf{Z})} \geq \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln\left\{\frac{p(\mathbf{X},\mathbf{Z}|\theta)}{q(\mathbf{Z})}\right\} = L(q,\theta)}$$

$$\downarrow$$

$$\boxed{\begin{aligned}
\ln p(\mathbf{X}|\theta) - L(q,\theta) &= \ln p(\mathbf{X}|\theta) - \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln\left\{\frac{p(\mathbf{X},\mathbf{Z}|\theta)}{q(\mathbf{Z})}\right\} \\
&= \ln p(\mathbf{X}|\theta) - \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln\left\{\frac{p(\mathbf{Z}|\mathbf{X},\theta)p(\mathbf{X}|\theta)}{q(\mathbf{Z})}\right\} \\
&= \ln p(\mathbf{X}|\theta) - \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln\left\{\frac{p(\mathbf{Z}|\mathbf{X},\theta)}{q(\mathbf{Z})}\right\} - \ln p(\mathbf{X}|\theta) \sum_{\mathbf{Z}} q(\mathbf{Z}) \\
&= -\sum_{\mathbf{Z}} q(\mathbf{Z}) \ln\left\{\frac{p(\mathbf{Z}|\mathbf{X},\theta)}{q(\mathbf{Z})}\right\} = KL[q(\mathbf{Z})||p(\mathbf{Z}|\mathbf{X},\theta)] = KL[q||p]
\end{aligned}}$$


$\mathrm{KL}(q||p)$
$\mathcal{L}(q,\boldsymbol{\theta})$
$\ln p(\mathbf{X}|\boldsymbol{\theta})$


$\mathrm{KL}(q||p) = 0$
$\mathcal{L}(q,\boldsymbol{\theta}^{\mathrm{old}})$
$\ln p(\mathbf{X}|\boldsymbol{\theta}^{\mathrm{old}})$

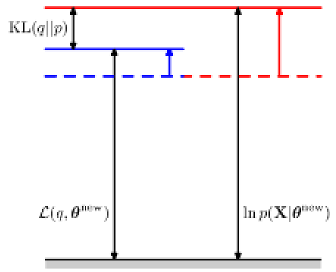
  - Minimization of KL divergence = Maximization of lower bound L

## Variational EM algorithm

- Process
  - M-step : maximize lower bound of log p(x)

$$\boxed{\begin{gathered} q(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X}, \theta^{old}) \\ L(q,\theta) = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X},\theta^{old}) \ln p(\mathbf{X},\mathbf{Z}|\theta) - \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X},\theta^{old}) \ln p(\mathbf{Z}|\mathbf{X},\theta^{old}) = Q(\theta,\theta^{old}) + const \end{gathered}}$$

  - Fixed Z, update new parameter $\theta$ to maximize functional L



$\mathrm{KL}(q||p)$
$\mathcal{L}(q,\boldsymbol{\theta}^{\mathrm{new}})$
$\ln p(\mathbf{X}|\boldsymbol{\theta}^{\mathrm{new}})$

## Maximum a posteriori policy optimisation

- Process
  - E-step : variational policy q(a|s) is a parametric variational distribution

$$\begin{aligned}\max_q \bar{\mathcal{J}}_s(q,\theta_i) &= \max_q T^{\pi,q} Q_{\boldsymbol{\theta}_i}(s,a) \\ &= \max_q \mathbb{E}_{\mu(s)}\Big[\mathbb{E}_{q(\cdot|s)}[Q_{\boldsymbol{\theta}_i}(s,a)] - \alpha \mathrm{KL}(q\|\pi_i)\Big]\end{aligned}$$

since $V_{\boldsymbol{\theta}_i}(s) = \mathbb{E}_{q(a|s)}[Q_{\boldsymbol{\theta}_i}(s,a)]$ and thus $Q_{\boldsymbol{\theta}_i}(s,a) = r(s,a) + \gamma V_{\boldsymbol{\theta}_i}(s)$

Constraint E step

$$\max_q \mathbb{E}_{\mu(s)}\Big[\mathbb{E}_{q(a|s)}\Big[Q_{\theta_i}(s,a)\Big]\Big]$$
$$s.t. \mathbb{E}_{\mu(s)}\Big[\mathrm{KL}(q(a|s), \pi(a|s,\boldsymbol{\theta}_i))\Big] < \epsilon.$$

  - Non-parametric variational distribution

Convex dual function of $\eta$

$$g(\eta) = \eta\epsilon + \eta \int \mu(s) \log \int \pi(a|s,\theta_i) \exp\left(\frac{Q_{\theta_i}(s,a)}{\eta}\right) da ds$$

$$q_i(a|s) \propto \pi(a|s,\theta_i) \exp\left(\frac{Q_{\theta_i}(s,a)}{\eta^*}\right)$$

## Maximum a posteriori policy optimisation

- Process
  - M-step : Given q from the E-step, optimize the lower bound J with respect to $\theta$ to obtain an updated policy

$$\boxed{\max_{\boldsymbol{\theta}} \mathcal{J}(q_i, \theta) = \max_{\boldsymbol{\theta}} \mathbb{E}_{\mu_q(s)}\Big[\mathbb{E}_{q(a|s)}\Big[\log \pi(a|s, \boldsymbol{\theta})\Big]\Big] + \log p(\boldsymbol{\theta})}$$

  - Generalized M-step

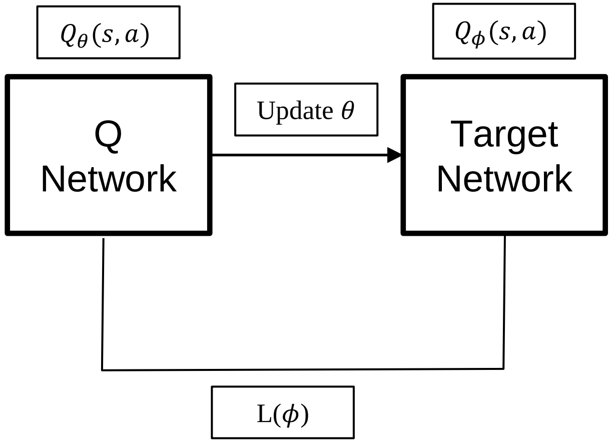
$$\boxed{\max_{\pi} \mathbb{E}_{\mu_q(s)}\Big[\mathbb{E}_{q(a|s)}\Big[\log \pi(a|s, \boldsymbol{\theta})\Big] - \lambda \mathrm{KL}\Big(\pi(a|s, \boldsymbol{\theta}_i), \pi(a|s, \boldsymbol{\theta})\Big)\Big]}$$

  - Constrained M-step

$$\boxed{\begin{aligned} &\max_{\pi} \mathbb{E}_{\mu_q(s)}\Big[\mathbb{E}_{q(a|s)}\Big[\log \pi(a|s, \boldsymbol{\theta})\Big]\Big] \\ s.t.\ &\mathbb{E}_{\mu_q(s)}\Big[\mathrm{KL}(\pi(a|s, \boldsymbol{\theta}_i), \pi(a|s, \boldsymbol{\theta}))\Big] < \epsilon. \end{aligned}}$$

# Maximum a posteriori policy optimisation

- Process
  - Policy Evaluation : fitting value function to policy
  - Use an experience replay buffer and arbitrary behaviour policy given by the action probabilities stored in the buffer
  - Off-policy + Multi-step returns : Retrace algorithm(Munos et al, 2016)



$$\min_\phi L(\phi) = \min_\phi \mathbb{E}_{\mu_b(s),b(a|s)} \Big[ \big(Q_{\theta_i}(s_t, a_t, \phi) - Q_t^{\text{ret}}\big)^2 \Big], \text{with}$$

$$Q_t^{\text{ret}} = Q_{\phi'}(s_t, a_t) + \sum_{j=t}^{\infty} \gamma^{j-t} \Big( \prod_{k=t+1}^{j} c_k \Big) \Big[ r(s_j, a_j) + \mathbb{E}_{\pi(a|s_{j+1})}[Q_{\phi'}(s_{j+1}, a)] - Q_{\phi'}(s_j, a_j) \Big]$$

$$c_k = \min\left(1, \frac{\pi(a_k|s_k)}{b(a_k|s_k)}\right),$$

# Maximum a posteriori policy optimisation

- Process
  - E-step

$$\begin{aligned}\max_q \bar{\mathcal{J}}_s(q,\theta_i) &= \max_q T^{\pi,q} Q_{\boldsymbol{\theta}_i}(s,a)\\ &= \max_q \mathbb{E}_{\mu(s)}\left[\mathbb{E}_{q(\cdot|s)}[Q_{\boldsymbol{\theta}_i}(s,a)] - \alpha \mathrm{KL}(q\|\pi_i)\right]\end{aligned}$$

since $V_{\boldsymbol{\theta}_i}(s) = \mathbb{E}_{q(a|s)}[Q_{\boldsymbol{\theta}_i}(s,a)]$ and thus $Q_{\boldsymbol{\theta}_i}(s,a) = r(s,a) + \gamma V_{\boldsymbol{\theta}_i}(s)$

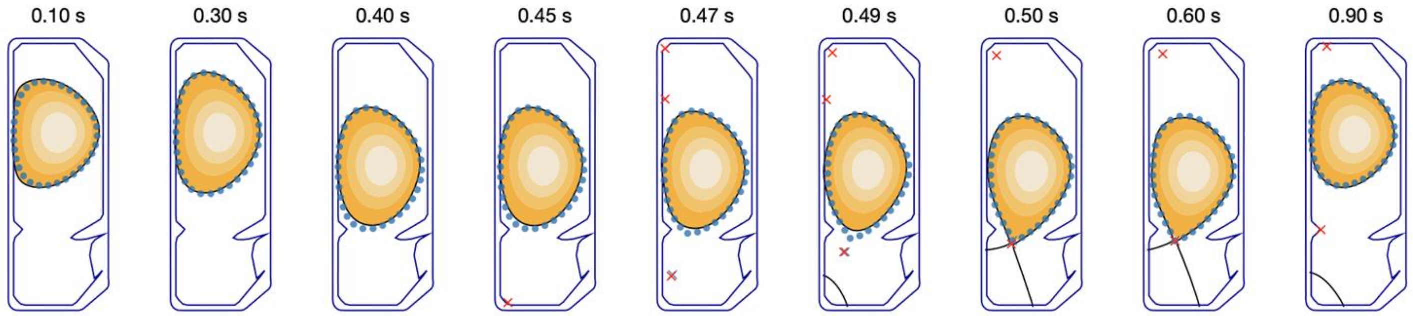
  - M-step

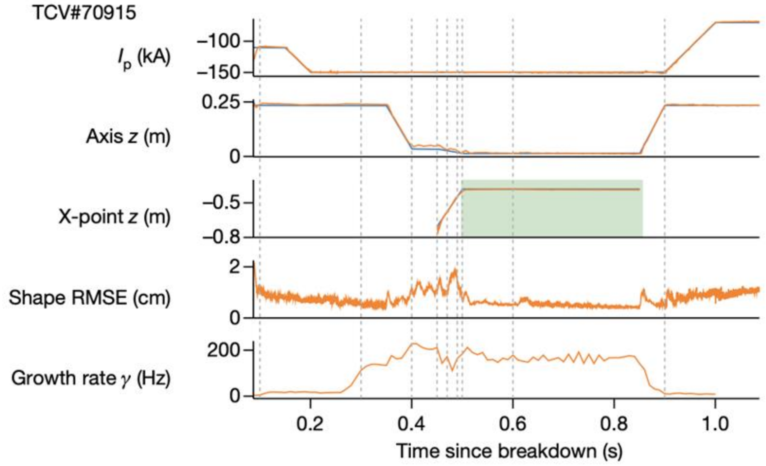
$$\boxed{\max_{\boldsymbol{\theta}} \mathcal{J}(q_i,\theta) = \max_{\boldsymbol{\theta}} \mathbb{E}_{\mu_q(s)}\left[\mathbb{E}_{q(a|s)}\left[\log \pi(a|s,\boldsymbol{\theta})\right]\right] + \log p(\boldsymbol{\theta})}$$

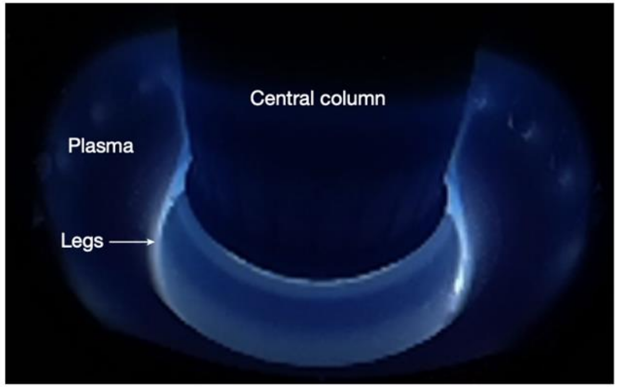
  - Policy Evaluation

$$\boxed{\begin{aligned}&\min_\phi L(\phi) = \min_\phi \mathbb{E}_{\mu_b(s),b(a|s)}\left[\left(Q_{\theta_i}(s_t,a_t,\phi) - Q_t^{\mathrm{ret}}\right)^2\right], \text{with}\\ &Q_t^{\mathrm{ret}} = Q_{\phi'}(s_t,a_t) + \sum_{j=t}^{\infty}\gamma^{j-t}\Big(\prod_{k=t+1}^{j} c_k\Big)\Big[r(s_j,a_j) + \mathbb{E}_{\pi(a|s_{j+1})}[Q_{\phi'}(s_{j+1},a)] - Q_{\phi'}(s_j,a_j)\Big]\\ &c_k = \min\left(1, \frac{\pi(a_k|s_k)}{b(a_k|s_k)}\right),\end{aligned}}$$

## Experiment : Capability Demonstration

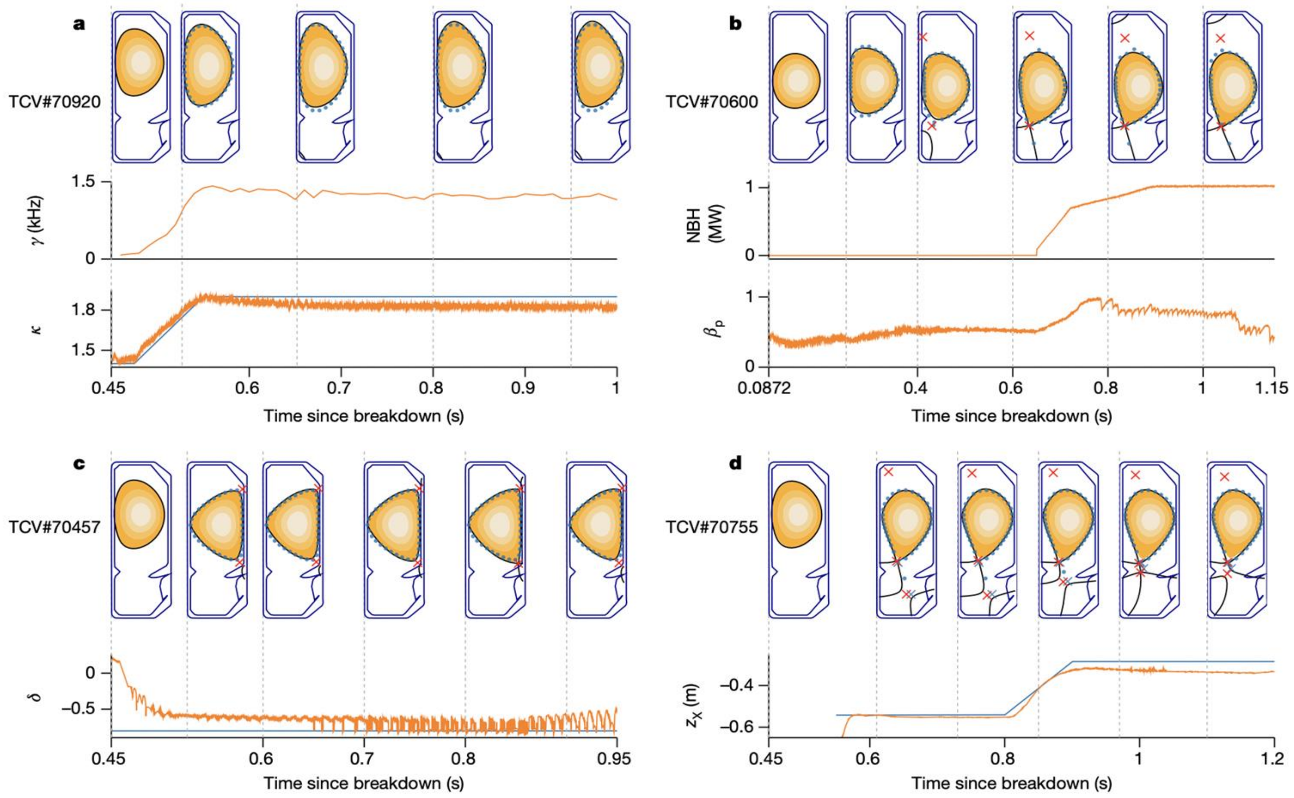




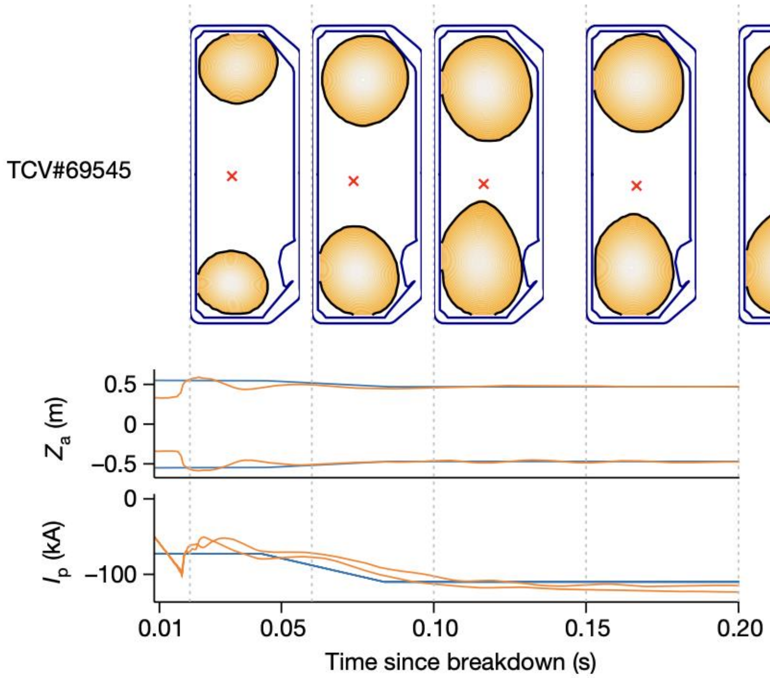


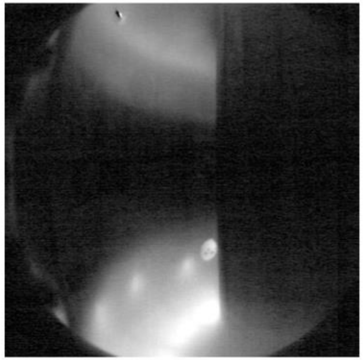
Inside view at 0.6 s

# Experiment : Control Demonstration



## Experiment : Droplets



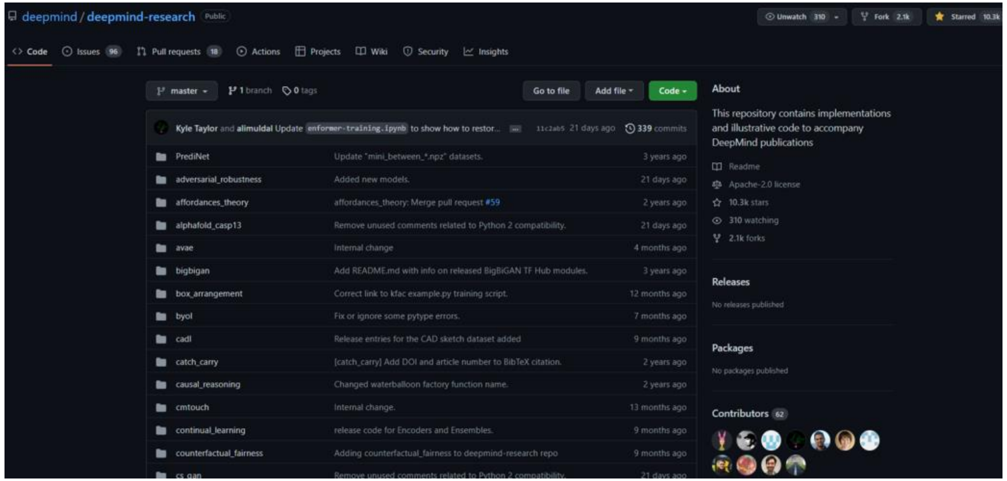


## Conclusion

- Highlight
  - High performance + robustness to uncertain operating conditions, intuitive target specification and unprecedented versatility
  - An accurate, numerically robust simulator(SFC) : trade-off between simulation accuracy and computational complexity / Highly data-efficient RL algorithm(MPO) that scales to high-dimensional problems
  - Free-boundary equilibrium evolution model has sufficient fidelity to develop transferable controllers, offering a justification for using this approach to test control of future devices

- Enhancement
  - Analysis of the non-linear dynamics
  - Reduce training time through increased reuse of data and multi-fidelity learning
  - The model architecture can be coupled to a more capable simulator(i.e. plasma pressure, current density evolution physics)

# Conclusion



Kyle Taylor and alimuldal Update `enformer-training.ipynb` to show how to restor... 11c2ab5 21 days ago 339 commits

| | | |
|---|---|---|
| PrediNet | Update "mini_between_*.npz" datasets. | 3 years ago |
| adversarial_robustness | Added new models. | 21 days ago |
| affordances_theory | affordances_theory: Merge pull request #59 | 2 years ago |
| alphafold_casp13 | Remove unused comments related to Python 2 compatibility. | 21 days ago |
| avae | Internal change | 4 months ago |
| bigbigan | Add README.md with info on released BigBiGAN TF Hub modules. | 3 years ago |
| box_arrangement | Correct link to kfac example.py training script. | 12 months ago |
| byol | Fix or ignore some pytype errors. | 7 months ago |
| cadl | Release entries for the CAD sketch dataset added | 9 months ago |
| catch_carry | [catch_carry] Add DOI and article number to BibTeX citation. | 2 years ago |
| causal_reasoning | Changed waterballoon factory function name. | 2 years ago |
| cmtouch | Internal change. | 13 months ago |
| continual_learning | release code for Encoders and Ensembles. | 9 months ago |
| counterfactual_fairness | Adding counterfactual_fairness to deepmind-research repo | 9 months ago |
| cs_gan | Remove unused comments related to Python 2 compatibility. | 21 days ago |

About

This repository contains implementations and illustrative code to accompany DeepMind publications

Readme
Apache-2.0 license
10.3k stars
310 watching
2.1k forks

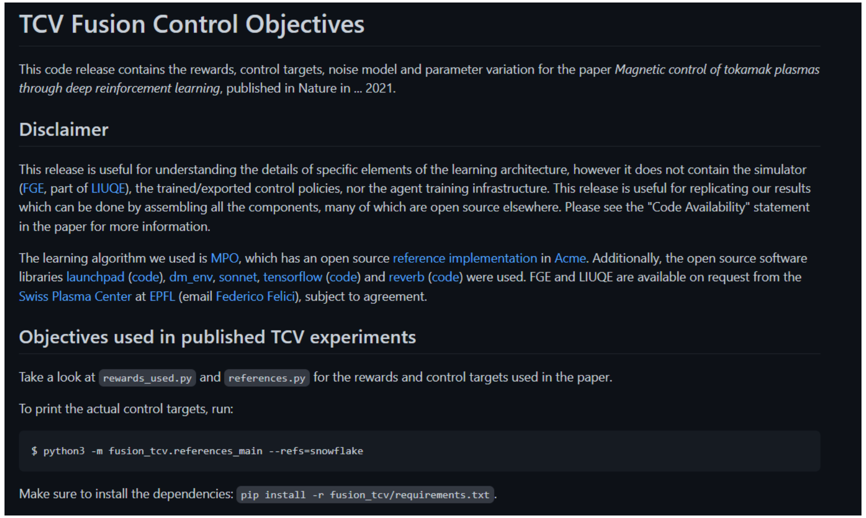
Releases

No releases published

Packages

No packages published

Contributors 62



## TCV Fusion Control Objectives

This code release contains the rewards, control targets, noise model and parameter variation for the paper *Magnetic control of tokamak plasmas through deep reinforcement learning*, published in Nature in ... 2021.

## Disclaimer

This release is useful for understanding the details of specific elements of the learning architecture, however it does not contain the simulator (FGE, part of LIUQE), the trained/exported control policies, nor the agent training infrastructure. This release is useful for replicating our results which can be done by assembling all the components, many of which are open source elsewhere. Please see the "Code Availability" statement in the paper for more information.

The learning algorithm we used is MPO, which has an open source reference implementation in Acme. Additionally, the open source software libraries launchpad (code), dm_env, sonnet, tensorflow (code) and reverb (code) were used. FGE and LIUQE are available on request from the Swiss Plasma Center at EPFL (email Federico Felici), subject to agreement.

## Objectives used in published TCV experiments

Take a look at `rewards_used.py` and `references.py` for the rewards and control targets used in the paper.

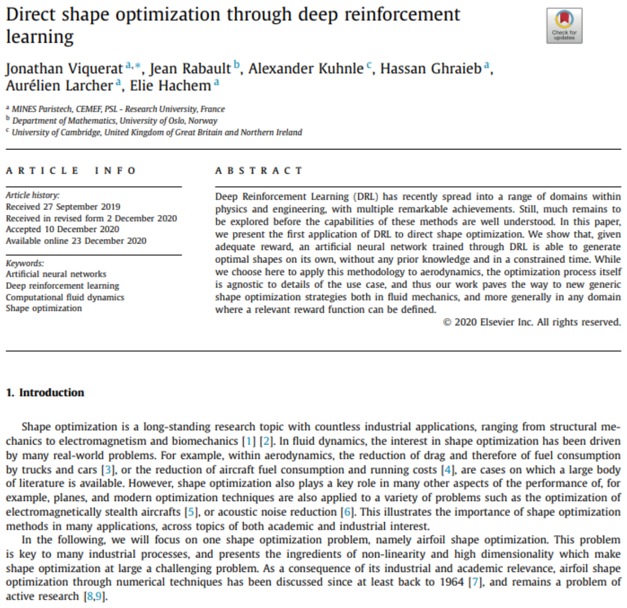
To print the actual control targets, run:

```
$ python3 -m fusion_tcv.references_main --refs=snowflake
```

Make sure to install the dependencies: `pip install -r fusion_tcv/requirements.txt`.

Source code : https://github.com/deepmind/deepmind-research/tree/master/fusion_tcv

# Conclusion



Direct shape optimization through deep reinforcement learning

Jonathan Viquerat [a,*], Jean Rabault [b], Alexander Kuhnle [c], Hassan Ghraieb [a], Aurélien Larcher [a], Elie Hachem [a]

[a] *MINES Paristech, CEMEF, PSL - Research University, France*
[b] *Department of Mathematics, University of Oslo, Norway*
[c] *University of Cambridge, United Kingdom of Great Britain and Northern Ireland*

ARTICLE INFO

*Article history:*
Received 27 September 2019
Received in revised form 2 December 2020
Accepted 10 December 2020
Available online 23 December 2020

*Keywords:*
Artificial neural networks
Deep reinforcement learning
Computational fluid dynamics
Shape optimization

ABSTRACT

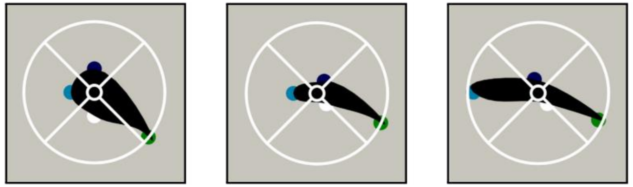
Deep Reinforcement Learning (DRL) has recently spread into a range of domains within physics and engineering, with multiple remarkable achievements. Still, much remains to be explored before the capabilities of these methods are well understood. In this paper, we present the first application of DRL to direct shape optimization. We show that, given adequate reward, an artificial neural network trained through DRL is able to generate optimal shapes on its own, without any prior knowledge and in a constrained time. While we choose here to apply this methodology to aerodynamics, the optimization process itself is agnostic to details of the use case, and thus our work paves the way to new generic shape optimization strategies both in fluid mechanics, and more generally in any domain where a relevant reward function can be defined.

**1. Introduction**

Shape optimization is a long-standing research topic with countless industrial applications, ranging from structural mechanics to electromagnetism and biomechanics [1] [2]. In fluid dynamics, the interest in shape optimization has been driven by many real-world problems. For example, within aerodynamics, the reduction of drag and therefore of fuel consumption by trucks and cars [3], or the reduction of aircraft fuel consumption and running costs [4], are cases on which a large body of literature is available. However, shape optimization also plays a key role in many other aspects of the performance of, for example, planes, and modern optimization techniques are also applied to a variety of problems such as the optimization of electromagnetically stealth aircrafts [5], or acoustic noise reduction [6]. This illustrates the importance of shape optimization methods in many applications, across topics of both academic and industrial interest.

In the following, we will focus on one shape optimization problem, namely airfoil shape optimization. This problem is key to many industrial processes, and presents the ingredients of non-linearity and high dimensionality which make shape optimization at large a challenging problem. As a consequence of its industrial and academic relevance, airfoil shape optimization through numerical techniques has been discussed since at least back to 1964 [7], and remains a problem of active research [8,9].

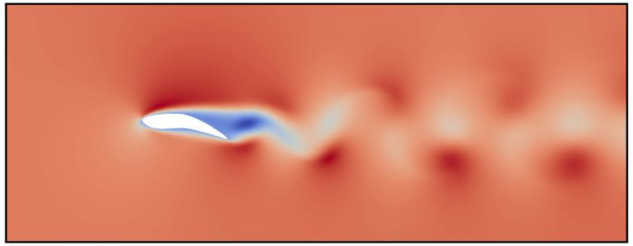
Jonathan Viquerat et al, 2021, Journel of computational physics

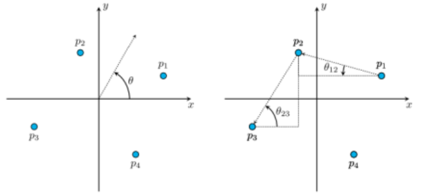


(a) Best shape with 4 points, 1 free point (3 d.o.f.s)

(b) Best shape with 4 points, 3 free points (9 d.o.f.s)

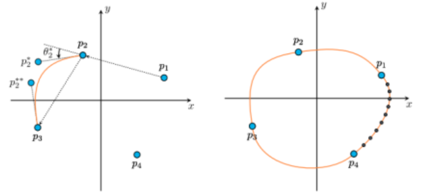
(c) Best shape with 4 points, 4 free points (12 d.o.f.s)





(a) Sort the provided points by ascending trigonometric angle

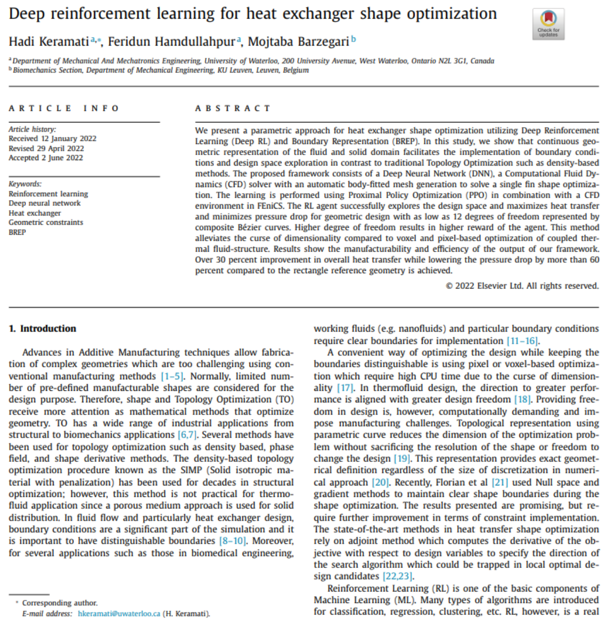
(b) Compute angles between points, and compute an average angle around each point $\theta_i^*$



(c) Compute control points coordinates from averaged angles and generate cubic Bézier

(d) Sample all Bézier lines and export for mesh immersion

# Conclusion



Deep reinforcement learning for heat exchanger shape optimization

Hadi Keramati[a,*], Feridun Hamdullahpur[a], Mojtaba Barzegari[b]

[a] Department of Mechanical And Mechatronics Engineering, University of Waterloo, 200 University Avenue, West Waterloo, Ontario N2L 3G1, Canada
[b] Biomechanics Section, Department of Mechanical Engineering, KU Leuven, Leuven, Belgium

A R T I C L E  I N F O

*Article history:*
Received 12 January 2022
Revised 29 April 2022
Accepted 2 June 2022

*Keywords:*
Reinforcement learning
Deep neural network
Heat exchanger
Geometric constraints
BREP

A B S T R A C T

We present a parametric approach for heat exchanger shape optimization utilizing Deep Reinforcement Learning (Deep RL) and Boundary Representation (BREP). In this study, we show that continuous geometric representation of the fluid and solid domain facilitates the implementation of boundary conditions and design space exploration in contrast to traditional Topology Optimization such as density-based methods. The proposed framework consists of a Deep Neural Network (DNN), a Computational Fluid Dynamics (CFD) solver with an automatic body-fitted mesh generation to solve a single fin shape optimization. The learning is performed using Proximal Policy Optimization (PPO) in combination with a CFD environment in FEniCS. The RL agent successfully explores the design space and maximizes heat transfer and minimizes pressure drop for geometric design with as low as 12 degrees of freedom represented by composite Bézier curves. Higher degree of freedom results in higher reward of the agent. This method alleviates the curse of dimensionality compared to voxel and pixel-based optimization of coupled thermal fluid-structure. Results show the manufacturability and efficiency of the output of our framework. Over 30 percent improvement in overall heat transfer while lowering the pressure drop by more than 60 percent compared to the rectangle reference geometry is achieved.

**1. Introduction**

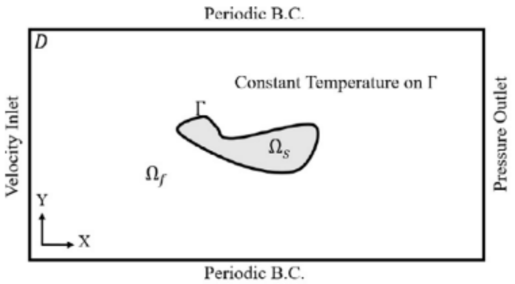
Advances in Additive Manufacturing techniques allow fabrication of complex geometries which are too challenging using conventional manufacturing methods [1–5]. Normally, limited number of pre-defined manufacturable shapes are considered for the design purpose. Therefore, shape and Topology Optimization (TO) receive more attention as mathematical methods that optimize geometry. TO has a wide range of industrial applications from structural to biomechanics applications [6,7]. Several methods have been used for topology optimization such as density based, phase field, and shape derivative methods. The density-based topology optimization procedure known as the SIMP (Solid isotropic material with penalization) has been used for decades in structural optimization; however, this method is not practical for thermo-fluid application since a porous medium approach is used for solid distribution. In fluid flow and particularly heat exchanger design, boundary conditions are a significant part of the simulation and it is important to have distinguishable boundaries [8–10]. Moreover, for several applications such as those in biomedical engineering, working fluids (e.g. nanofluids) and particular boundary conditions require clear boundaries for implementation [11–16].

A convenient way of optimizing the design while keeping the boundaries distinguishable is using pixel or voxel-based optimization which require high CPU time due to the curse of dimensionality [17]. In thermofluid design, the direction to greater performance is aligned with greater design freedom [18]. Providing freedom in design is, however, computationally demanding and impose manufacturing challenges. Topological representation using parametric curve reduces the dimension of the optimization problem without sacrificing the resolution of the shape or freedom to change the design [19]. This representation provides exact geometrical definition regardless of the size of discretization in numerical approach [20]. Recently, Florian et al [21] used Null space and gradient methods to maintain clear shape boundaries during the shape optimization. The results presented are promising, but require further improvement in terms of constraint implementation. The state-of-the-art methods in heat transfer shape optimization rely on adjoint method which computes the derivative of the objective with respect to design variables to specify the direction of the search algorithm which could be trapped in local optimal design candidates [22,23].

Reinforcement Learning (RL) is one of the basic components of Machine Learning (ML). Many types of algorithms are introduced for classification, regression, clustering, etc. RL, however, is a real

* Corresponding author.
*E-mail address:* hkeramati@uwaterloo.ca (H. Keramati).

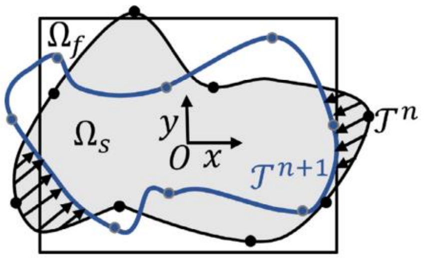
Hadi Keramati al, 2022, International Journal of heat and mass transfer

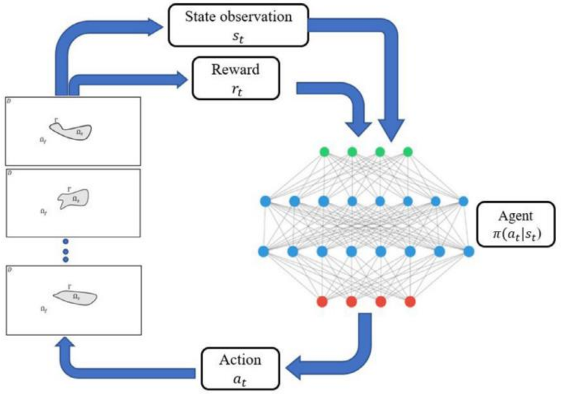


$$\sigma_f(\mathbf{u},p)=2\nu e(\mathbf{u})-pI,\; e(\mathbf{u})=\frac{1}{2}(\nabla\mathbf{u}+\nabla\mathbf{u}^T)$$

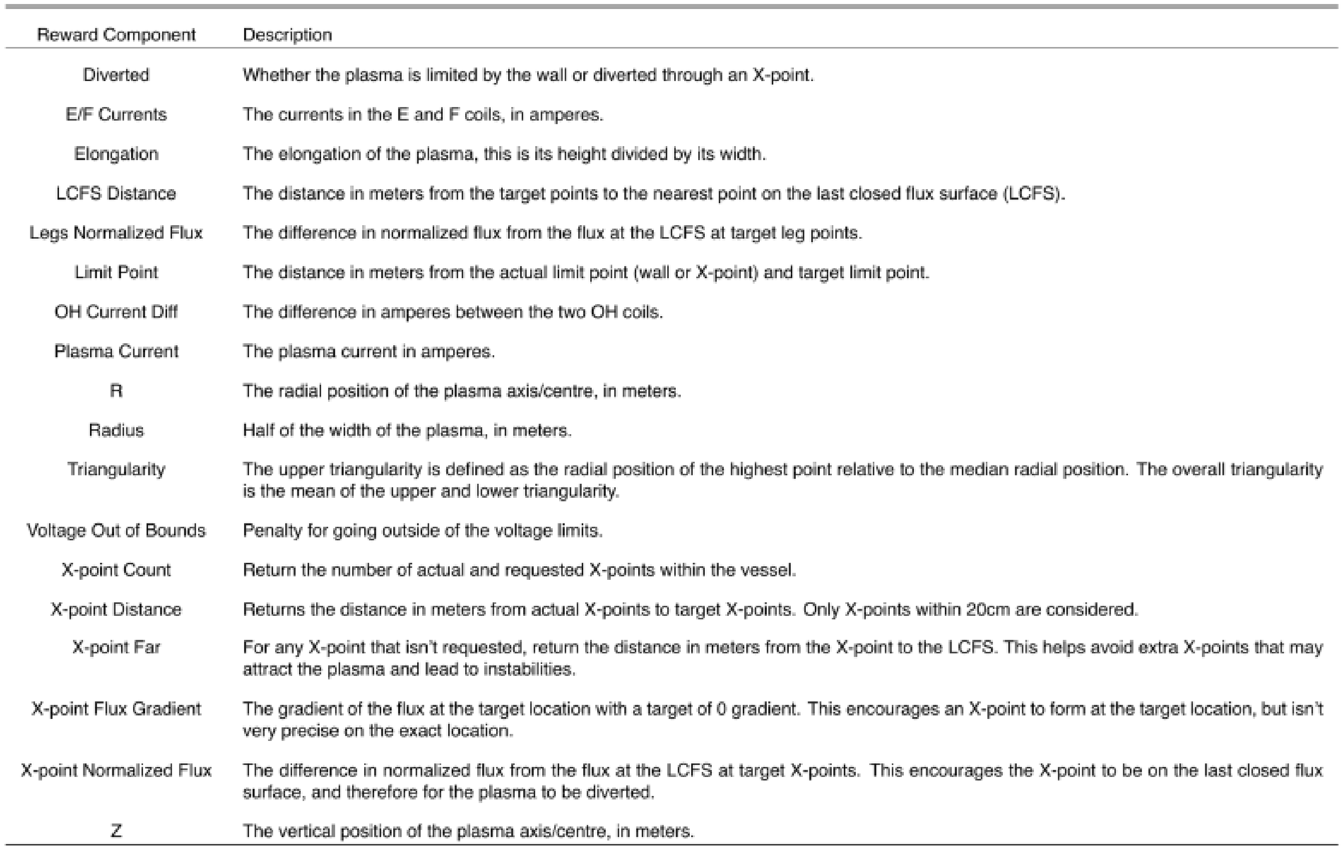
$$\begin{cases}\rho\frac{\partial\mathbf{u}}{\partial t}+\rho((\mathbf{u}\cdot\nabla)\mathbf{u})-\nabla(\sigma_f(\mathbf{u},p))=0 & \text{in} \quad \Omega_f\\ \nabla\cdot\mathbf{u}=0 & \text{in} \quad \Omega_f\\ \mathbf{u}=\mathbf{u}_0 & \text{on} \quad \Omega_{f,in}\\ \sigma_f(\mathbf{u},p)\cdot\mathbf{n}=0 & \text{on} \quad \Omega_{f,out}\\ \mathbf{u}=0 & \text{on} \quad \Gamma\end{cases}$$

$$\begin{cases}\rho c_p\frac{\partial T}{\partial t}+\rho c_p(\mathbf{u}\cdot\nabla T)-\nabla\cdot(k_f\nabla T)=0 & \text{in} \quad \Omega_f\\ T=T_{in} & \text{on} \quad \Omega_{f,in}\\ T=T_0 & \text{on} \quad \Omega_{f,t=0}\\ T_f=T_s & \text{on} \quad \Gamma\\ -k_f\frac{\partial T_f}{\partial n}=-k_s\frac{\partial T_s}{\partial n} & \text{on} \quad \Gamma\end{cases}$$



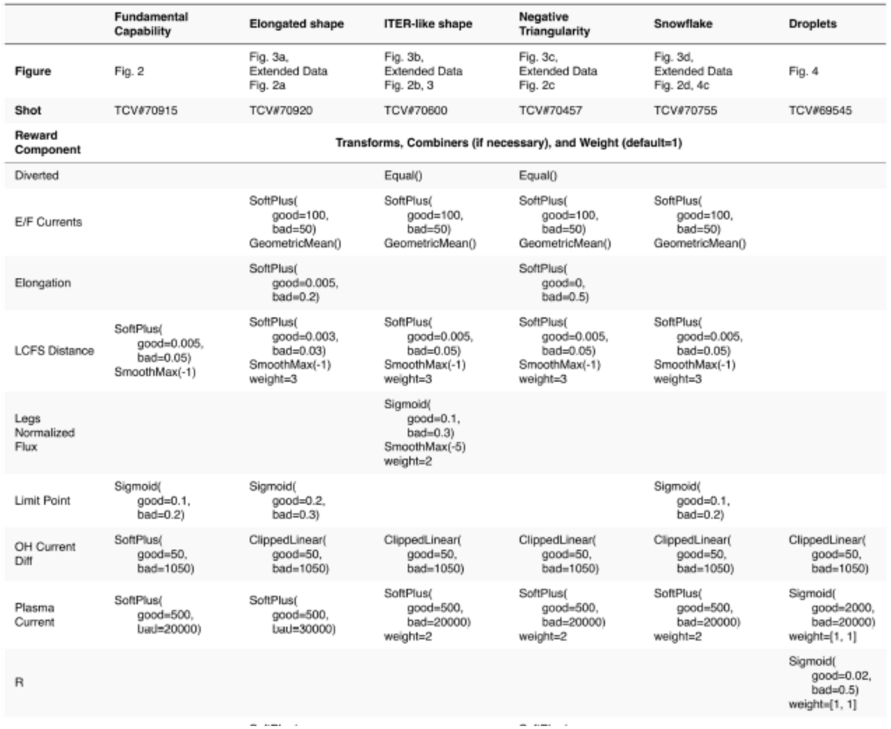


# Appendix : Reward components

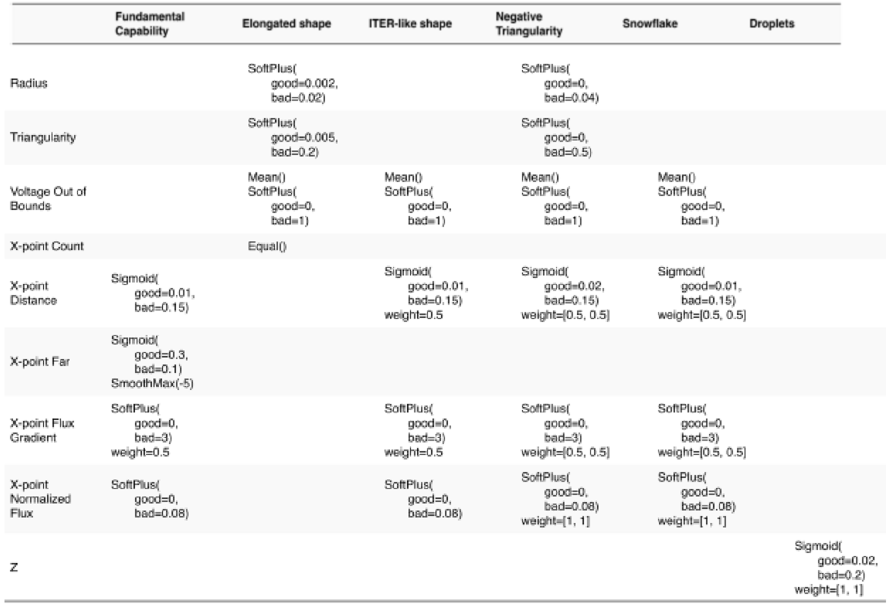


| Reward Component | Description |
|---|---|
| Diverted | Whether the plasma is limited by the wall or diverted through an X-point. |
| E/F Currents | The currents in the E and F coils, in amperes. |
| Elongation | The elongation of the plasma, this is its height divided by its width. |
| LCFS Distance | The distance in meters from the target points to the nearest point on the last closed flux surface (LCFS). |
| Legs Normalized Flux | The difference in normalized flux from the flux at the LCFS at target leg points. |
| Limit Point | The distance in meters from the actual limit point (wall or X-point) and target limit point. |
| OH Current Diff | The difference in amperes between the two OH coils. |
| Plasma Current | The plasma current in amperes. |
| R | The radial position of the plasma axis/centre, in meters. |
| Radius | Half of the width of the plasma, in meters. |
| Triangularity | The upper triangularity is defined as the radial position of the highest point relative to the median radial position. The overall triangularity is the mean of the upper and lower triangularity. |
| Voltage Out of Bounds | Penalty for going outside of the voltage limits. |
| X-point Count | Return the number of actual and requested X-points within the vessel. |
| X-point Distance | Returns the distance in meters from actual X-points to target X-points. Only X-points within 20cm are considered. |
| X-point Far | For any X-point that isn't requested, return the distance in meters from the X-point to the LCFS. This helps avoid extra X-points that may attract the plasma and lead to instabilities. |
| X-point Flux Gradient | The gradient of the flux at the target location with a target of 0 gradient. This encourages an X-point to form at the target location, but isn't very precise on the exact location. |
| X-point Normalized Flux | The difference in normalized flux from the flux at the LCFS at target X-points. This encourages the X-point to be on the last closed flux surface, and therefore for the plasma to be diverted. |
| Z | The vertical position of the plasma axis/centre, in meters. |

# Appendix : Reward used in the experiment

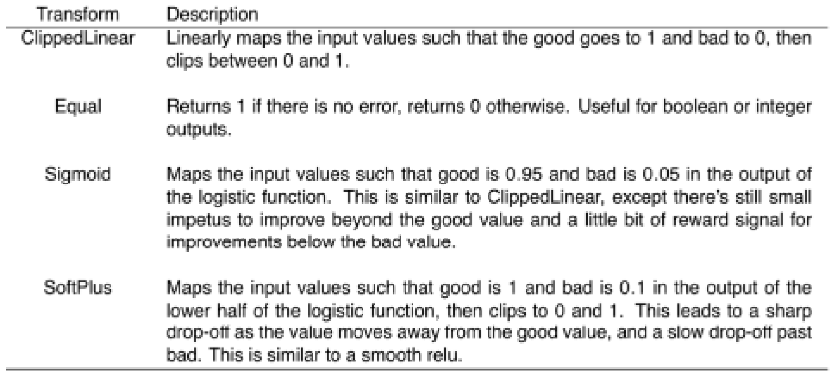


| | Fundamental Capability | Elongated shape | ITER-like shape | Negative Triangularity | Snowflake | Droplets |
|---|---|---|---|---|---|---|
| **Figure** | Fig. 2 | Fig. 3a,<br>Extended Data<br>Fig. 2a | Fig. 3b,<br>Extended Data<br>Fig. 2b, 3 | Fig. 3c,<br>Extended Data<br>Fig. 2c | Fig. 3d,<br>Extended Data<br>Fig. 2d, 4c | Fig. 4 |
| **Shot** | TCV#70915 | TCV#70920 | TCV#70600 | TCV#70457 | TCV#70755 | TCV#69545 |
| **Reward Component** | **Transforms, Combiners (if necessary), and Weight (default=1)** | | | | | |
| Diverted | | | Equal() | Equal() | | |
| E/F Currents | | SoftPlus(<br>good=100,<br>bad=50)<br>GeometricMean() | SoftPlus(<br>good=100,<br>bad=50)<br>GeometricMean() | SoftPlus(<br>good=100,<br>bad=50)<br>GeometricMean() | SoftPlus(<br>good=100,<br>bad=50)<br>GeometricMean() | |
| Elongation | | SoftPlus(<br>good=0.005,<br>bad=0.2) | | SoftPlus(<br>good=0,<br>bad=0.5) | | |
| LCFS Distance | SoftPlus(<br>good=0.005,<br>bad=0.05)<br>SmoothMax(-1) | SoftPlus(<br>good=0.003,<br>bad=0.03)<br>SmoothMax(-1)<br>weight=3 | SoftPlus(<br>good=0.005,<br>bad=0.05)<br>SmoothMax(-1)<br>weight=3 | SoftPlus(<br>good=0.005,<br>bad=0.05)<br>SmoothMax(-1)<br>weight=3 | SoftPlus(<br>good=0.005,<br>bad=0.05)<br>SmoothMax(-1)<br>weight=3 | |
| Legs Normalized Flux | | | Sigmoid(<br>good=0.1,<br>bad=0.3)<br>SmoothMax(-5)<br>weight=2 | | | |
| Limit Point | Sigmoid(<br>good=0.1,<br>bad=0.2) | Sigmoid(<br>good=0.2,<br>bad=0.3) | | | Sigmoid(<br>good=0.1,<br>bad=0.2) | |
| OH Current Diff | SoftPlus(<br>good=50,<br>bad=1050) | ClippedLinear(<br>good=50,<br>bad=1050) | ClippedLinear(<br>good=50,<br>bad=1050) | ClippedLinear(<br>good=50,<br>bad=1050) | ClippedLinear(<br>good=50,<br>bad=1050) | ClippedLinear(<br>good=50,<br>bad=1050) |
| Plasma Current | SoftPlus(<br>good=500,<br>bad=20000) | SoftPlus(<br>good=500,<br>bad=30000) | SoftPlus(<br>good=500,<br>bad=20000)<br>weight=2 | SoftPlus(<br>good=500,<br>bad=20000)<br>weight=2 | SoftPlus(<br>good=500,<br>bad=20000)<br>weight=2 | Sigmoid(<br>good=2000,<br>bad=20000)<br>weight=[1, 1] |
| R | | | | | | Sigmoid(<br>good=0.02,<br>bad=0.5)<br>weight=[1, 1] |

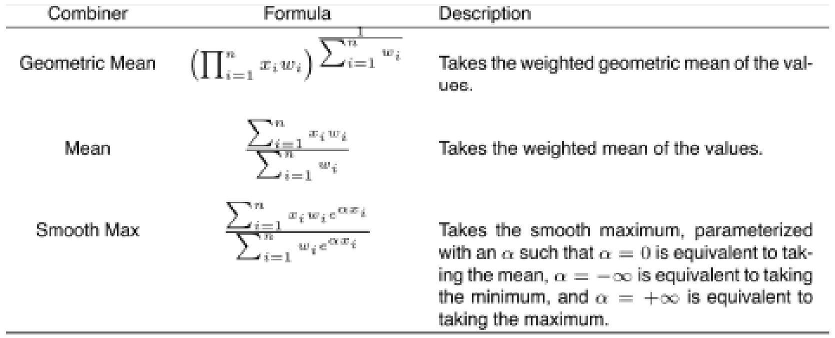


| | Fundamental Capability | Elongated shape | ITER-like shape | Negative Triangularity | Snowflake | Droplets |
|---|---|---|---|---|---|---|
| Radius | | SoftPlus(<br>good=0.002,<br>bad=0.02) | | SoftPlus(<br>good=0,<br>bad=0.04) | | |
| Triangularity | | SoftPlus(<br>good=0.005,<br>bad=0.2) | | SoftPlus(<br>good=0,<br>bad=0.5) | | |
| Voltage Out of Bounds | | Mean()<br>SoftPlus(<br>good=0,<br>bad=1) | Mean()<br>SoftPlus(<br>good=0,<br>bad=1) | Mean()<br>SoftPlus(<br>good=0,<br>bad=1) | Mean()<br>SoftPlus(<br>good=0,<br>bad=1) | |
| X-point Count | | Equal() | | | | |
| X-point Distance | Sigmoid(<br>good=0.01,<br>bad=0.15) | | Sigmoid(<br>good=0.01,<br>bad=0.15)<br>weight=0.5 | Sigmoid(<br>good=0.02,<br>bad=0.15)<br>weight=[0.5, 0.5] | Sigmoid(<br>good=0.01,<br>bad=0.15)<br>weight=[0.5, 0.5] | |
| X-point Far | Sigmoid(<br>good=0.3,<br>bad=0.1)<br>SmoothMax(-5) | | | | | |
| X-point Flux Gradient | SoftPlus(<br>good=0,<br>bad=3)<br>weight=0.5 | | SoftPlus(<br>good=0,<br>bad=3)<br>weight=0.5 | SoftPlus(<br>good=0,<br>bad=3)<br>weight=[0.5, 0.5] | SoftPlus(<br>good=0,<br>bad=3)<br>weight=[0.5, 0.5] | |
| X-point Normalized Flux | SoftPlus(<br>good=0,<br>bad=0.08) | | SoftPlus(<br>good=0,<br>bad=0.08) | SoftPlus(<br>good=0,<br>bad=0.08)<br>weight=[1, 1] | SoftPlus(<br>good=0,<br>bad=0.08)<br>weight=[1, 1] | |
| Z | | | | | | Sigmoid(<br>good=0.02,<br>bad=0.2)<br>weight=[1, 1] |

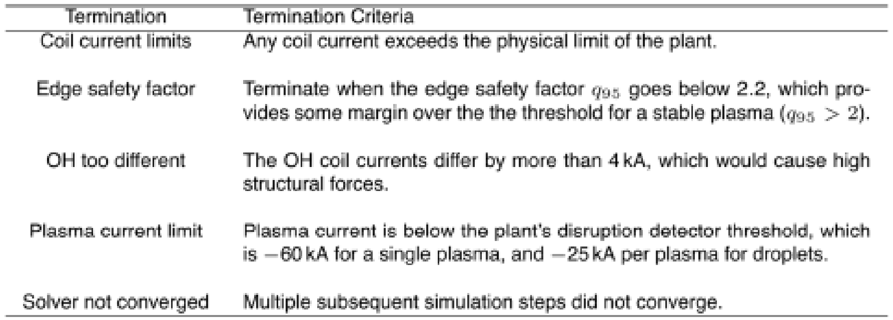
# Appendix : Reward elements



| Transform | Description |
|---|---|
| ClippedLinear | Linearly maps the input values such that the good goes to 1 and bad to 0, then clips between 0 and 1. |
| Equal | Returns 1 if there is no error, returns 0 otherwise. Useful for boolean or integer outputs. |
| Sigmoid | Maps the input values such that good is 0.95 and bad is 0.05 in the output of the logistic function. This is similar to ClippedLinear, except there's still small impetus to improve beyond the good value and a little bit of reward signal for improvements below the bad value. |
| SoftPlus | Maps the input values such that good is 1 and bad is 0.1 in the output of the lower half of the logistic function, then clips to 0 and 1. This leads to a sharp drop-off as the value moves away from the good value, and a slow drop-off past bad. This is similar to a smooth relu. |



| Combiner | Formula | Description |
|---|---|---|
| Geometric Mean | $\left(\prod_{i=1}^{n} x_i w_i\right)^{\frac{1}{\sum_{i=1}^{n} w_i}}$ | Takes the weighted geometric mean of the values. |
| Mean | $\frac{\sum_{i=1}^{n} x_i w_i}{\sum_{i=1}^{n} w_i}$ | Takes the weighted mean of the values. |
| Smooth Max | $\frac{\sum_{i=1}^{n} x_i w_i e^{\alpha x_i}}{\sum_{i=1}^{n} w_i e^{\alpha x_i}}$ | Takes the smooth maximum, parameterized with an $\alpha$ such that $\alpha = 0$ is equivalent to taking the mean, $\alpha = -\infty$ is equivalent to taking the minimum, and $\alpha = +\infty$ is equivalent to taking the maximum. |



| Termination | Termination Criteria |
|---|---|
| Coil current limits | Any coil current exceeds the physical limit of the plant. |
| Edge safety factor | Terminate when the edge safety factor $q_{95}$ goes below 2.2, which provides some margin over the the threshold for a stable plasma ($q_{95} > 2$). |
| OH too different | The OH coil currents differ by more than 4 kA, which would cause high structural forces. |
| Plasma current limit | Plasma current is below the plant's disruption detector threshold, which is $-60$ kA for a single plasma, and $-25$ kA per plasma for droplets. |
| Solver not converged | Multiple subsequent simulation steps did not converge. |

## Reference

[1] Magnetic control of tokamak plasmas through deep reinforcement learning, Jonas Degrave et al, Nature, 2022

[2] Maximum a posteriori policy optimization, Abbas Abdolmaleki et al, ICLR, 2018

[3] Development of free boundary equilibrium and transport solvers for simulation and real-time interpretation of tokamak experiments, Francesco Carpanese, EPFL

[4] Policy Gradient Methods for Reinforcement Learning with Function Approximation, Richard S.Sutton et al, 1999

[5] An Introduction to Reinforcement Learning, Vincent Francois-Lavet et al, 2018

[6] On Multi-objective Policy Optimization as a Tool for Reinforcement Learning, Abbas Abdolmaleki et al, 2021