0 Contents

1 Results and Discussion

1.1 Training

In this section how the training of the player and adversary will be investigated, using the theory discussed thus far. It is expected that results will be variable due to the minimal amount of data that can be generated with due to computational limitations, however, performance is expected to improve overall for both the adversary and the player. These results are then interpreted with a discussion of their merits and possible explanations. The results are then checked against a base-line neural network that takes the true state, $\boldsymbol{x}$, as input. It is expected that these will have comparable performance. For the training runs shown, the parameters used are given in table 1.1.

Unopposed Episodes/iter	Opposed Episodes/iter	nMCTS Simulations	$F_2^{(max)}$
100	40	15	0.05
$\boldsymbol{x}^{(2D)}$ bins	$\boldsymbol{x}^{(2D)}$ discount	Learning Rate	Mini-Batch Size
50	0.5	0.0005	64

Table 1.1: Training parameters used

1.1.1 MCTS

Figure 1.1 shows the structure of a typical tree search when relying solely on the tree search. This is significantly better than with no tree-search, which typically achieves an episode length of 7.

`Do graph of MCTS sims vs episode length with no training?`

Figure 1.2 shows the sequential maximisation that occurs during the tree search. Note that the action-value, $Q(\boldsymbol{x}_t, u_t)$, for the adversary is negated and Q is between -1 and 0 for the player. $N(\boldsymbol{x}, u)$ is correlated with the predicted value, V_{pred}, and U, which suggests that the neural network and tree search are working as expected. At the first branching in fig. 1.2 the *ucb* is 17% of U. Intuitively, this is within the correct range as a larger

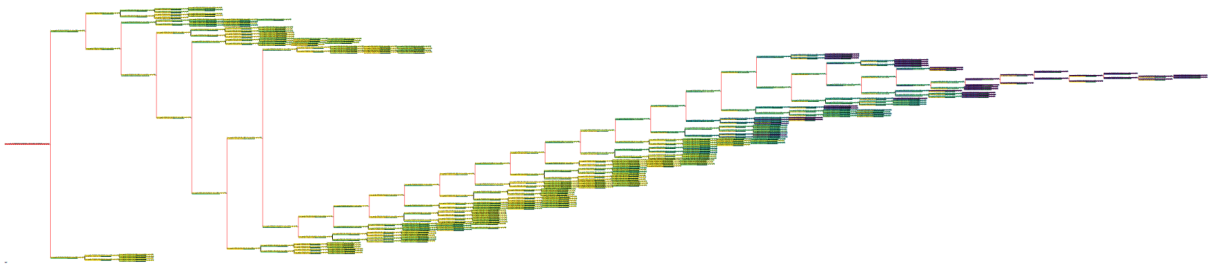

Figure 1.1: An example MCTS tree structure lasting 33 steps with 15 MCTS simulations per step. Both agents are untrained. The adversary has and average power of 5% of the player ($F_2 = 0.05F_1$). Lighter (yellow) colours represent "better" states.

exploration term would cause the MCTS to branch more, diminishing the computational benefit of the structured search, whereas a smaller exploration term inhibits learning. To get this ratio, a value of 1 was chosen for $c_{explore}$. The action-state visit counts are 22 and 17, which are markedly higher than the number of simulations per step (10 in this example), suggesting the MCTS is performing a highly structured search. In this example, the visit counts reflect the similar Q values of each branch.

1.1.2 Neural Network Training

In order to get similar magnitudes of action and value losses ($\mathcal{L}_{action} \approx \mathcal{L}_{value}$) a constant, c_{pareto}, was chosen. The minimum entropy for two possible actions is 1 bit and since the action loss is the negative log likelihood of a categorical distribution, the minimum loss is -1. The value losses are calculated using the MSE, which is positive, and has an average value of ≈ 0.1, therefore c_{pareto} was set to 0.1.

How much to train the player before introducing an adversary and how strong to make the adversary is difficult to balance. Typically, either the adversary is too strong and the episode is over within 20 time steps, or the adversary learns to simply push in one direction only. Figure 1.3 shows this transition occuring over just one episode. Iteration 1 has MCTS predictions grouped as either left or right

Over the first few training iterations, the loss decreases steadily figs. 2.1 to 2.4.

After the second iteration, the action losses do not decrease much, suggesting that they have already been trained (possible over-fitting)

1.1.3 Action and State Values

show the evolution of action training, and how they start out biased due to the training with f2=0 and then gets less and less biased..

talk about general trends and what the optimal control policy is likely to be

Figure 1.2: A zoomed in view of fig. 1.1. The three colours of each node are based on the value of $U = Q(\boldsymbol{x}, u) + ucb$ as in **??**, the number of state-action visits, $N(\boldsymbol{x}, u)$, and the state-cost, $c(\boldsymbol{x}_t)$. The red nodes follow the true trajectory. Ns ($N(\boldsymbol{x})$) is the total number of state visits, this does not equal the action-state visit count, Nsa ($N(\boldsymbol{x}, u)$), these differ along the true trajectory due to the re-running of the MCTS simulation.

how did things being discrete affect things? what might improve these results? Do average/smoothed one?

1.2 Performance

1.3 Performance Without an Adversary

1.4 Performance Against a Random Opponent

Selected Distributions

do against a constant random opponent do against say off for 30 steps then hit with a +- 5 do against a multinomial distribution with expected value of 0.05? do against

1.5 Performance Against a Trained Adversary

Compare when trained n the adversay against the random opponents? and also against no adversary? Is it robust?

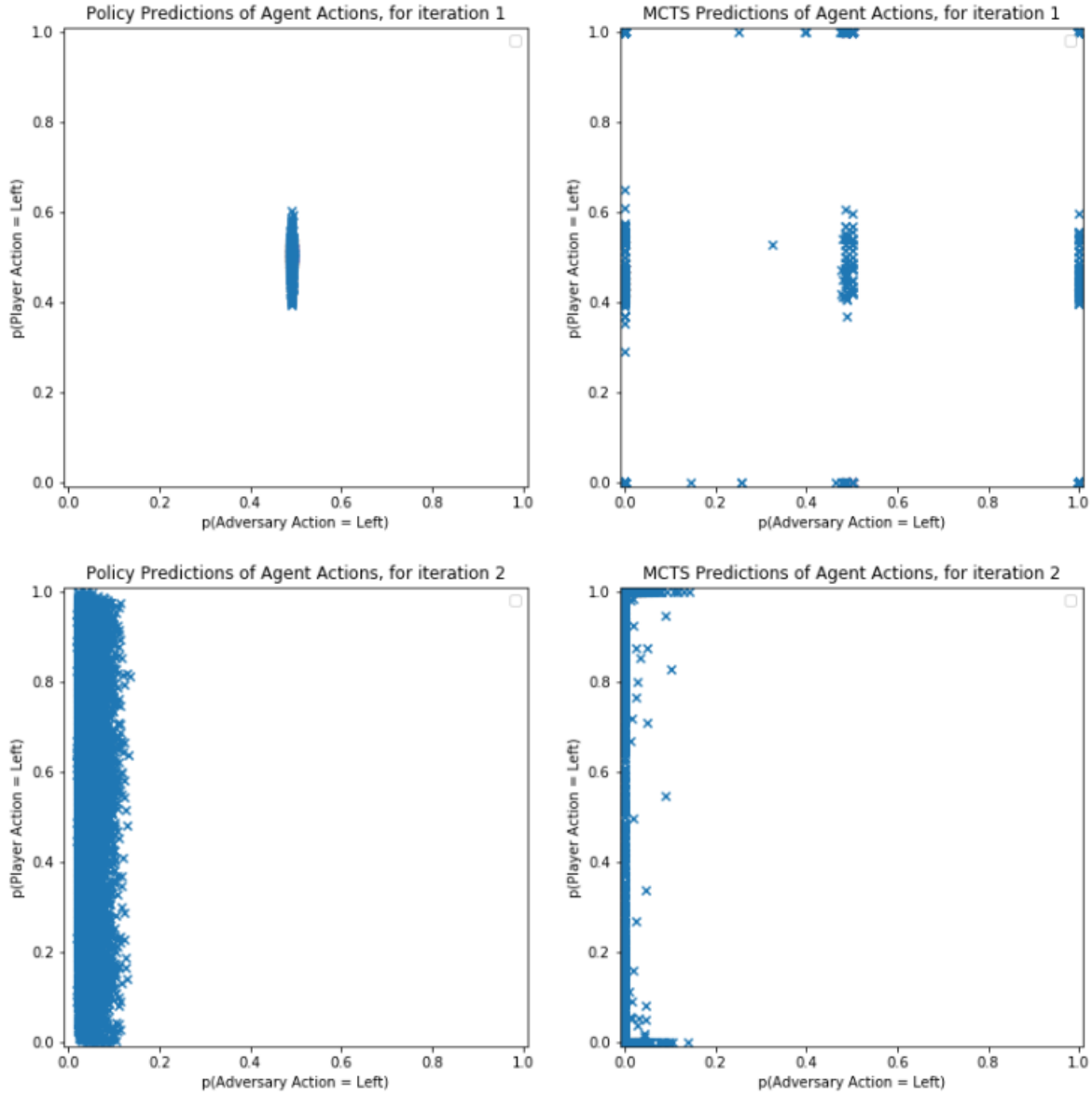

Figure 1.3: The change in MCTS predicted actions (right) and policy predicted actions (left) between iteration 1 (top) and iteration 2 (bottom). The points represent sequential action pairs, (u_{t-1}, u_t), and show how correlated an adversary action is with the previous agent's (player) action.

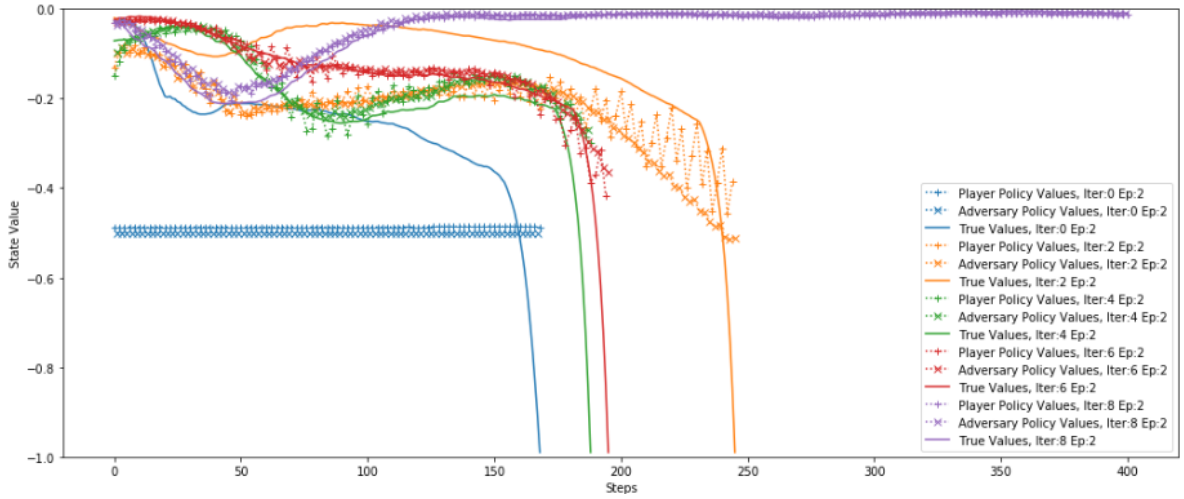

Figure 1.4: Value vs Step.

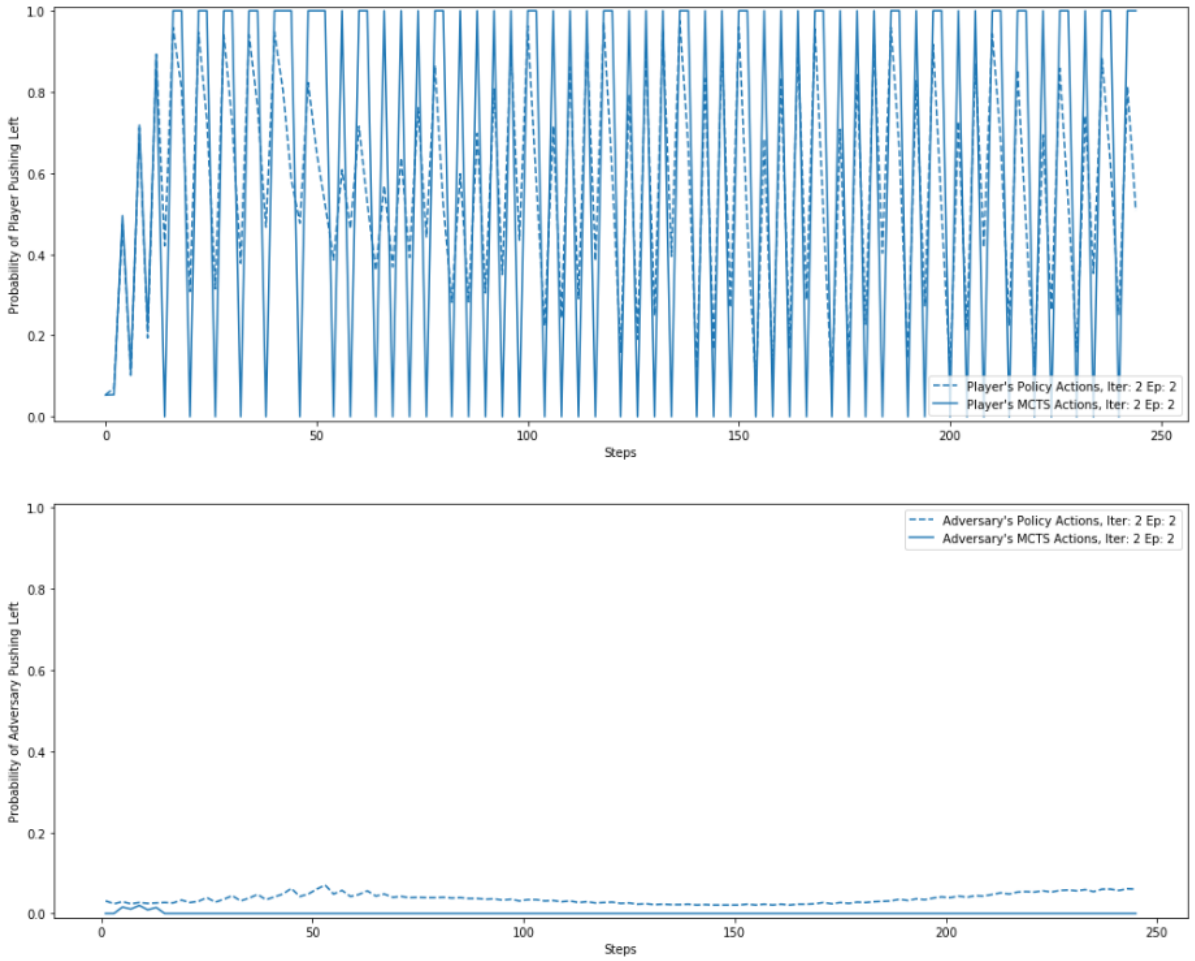

Figure 1.5: Action vs Step

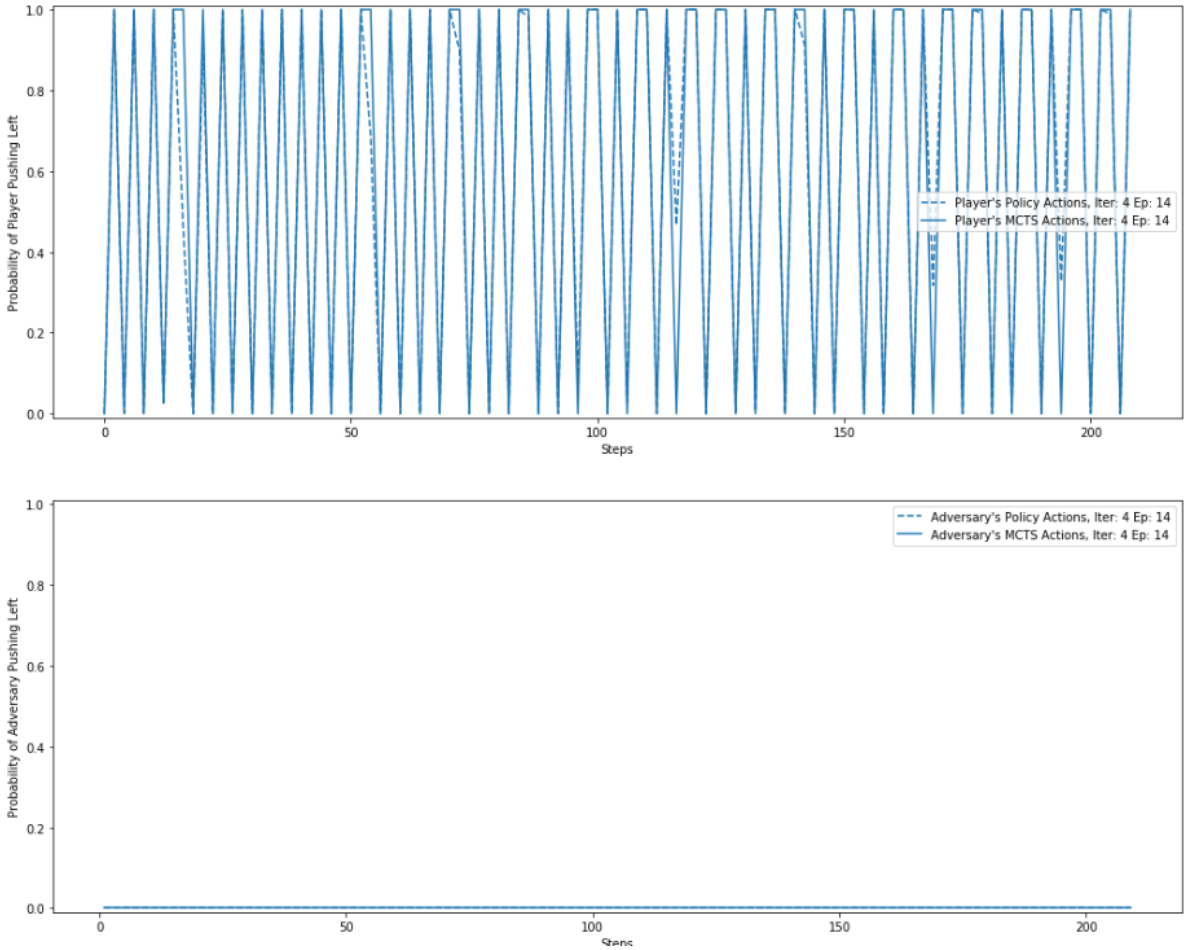

Figure 1.6: Action vs Step

2 Appendices

2.1 Inverted Pendulum Dynamics Derivation

We can find the state space equations for the Inverted Pendulum using d'Alembert forces. Firstly we define the distance and velocity vectors to the important points:

$$\begin{aligned}
\boldsymbol{r}_P &= x\boldsymbol{i} \\
\boldsymbol{r}_{B_1/P} &= Lsin\theta\boldsymbol{i} + Lcos\theta\boldsymbol{j} \\
\boldsymbol{r}_{B_1} &= (x + Lcos\theta)\boldsymbol{i} + L\dot{\theta}sin\theta\boldsymbol{j} \\
\dot{\boldsymbol{r}}_{B_1} &= (\dot{x} + L\dot{\theta}cos\theta)\boldsymbol{i} - L\dot{\theta}sin\theta\boldsymbol{j}
\end{aligned}$$

Linear Momentum, $\boldsymbol{\rho} = \sum_i m_i\dot{\boldsymbol{r}}_{i/o} = m\dot{\boldsymbol{r}}_{B_1} + M\dot{\boldsymbol{r}}_P$:

$$\boldsymbol{\rho} = \begin{bmatrix} (M+m)\dot{x} + mL\dot{\theta}cos\theta \\ -mL\dot{\theta}sin\theta \\ 0 \end{bmatrix}$$

Moment of momentum about P, $\boldsymbol{h}_P = \boldsymbol{r}_{B_1/P} \times m\dot{\boldsymbol{r}}_{B_1}$:

$$\begin{aligned}
\boldsymbol{h}_P &= -mL(L\dot{\theta} + \dot{x}cos\theta)\boldsymbol{k} \\
\therefore \dot{\boldsymbol{h}}_P &= -mL(L\ddot{\theta} + \ddot{x}cos\theta - \dot{x}\dot{\theta}sin\theta)\boldsymbol{k}
\end{aligned}$$

We can balance moments using $\dot{\boldsymbol{h}}_P + \dot{\boldsymbol{r}}_P \times \boldsymbol{\rho} = \boldsymbol{Q}_e$ and $\boldsymbol{Q}_e = \boldsymbol{r}_{B_1/P} \times -mg\boldsymbol{j} + \boldsymbol{r}_{B_2/P} \times F_2\boldsymbol{i}$:

$$\dot{\boldsymbol{h}}_P + \dot{\boldsymbol{r}}_P \times \boldsymbol{\rho} = \begin{bmatrix} 0 \\ 0 \\ -mL(\ddot{x}cos\theta + L\ddot{\theta}) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -L(mgsin\theta + 2F_2cos\theta) \end{bmatrix} = \boldsymbol{Q}_e$$

And also balance linear momentum using $\boldsymbol{F}_e = \dot{\boldsymbol{\rho}}$:

$$\dot{\boldsymbol{\rho}} = \begin{bmatrix} (m+M)\ddot{x} + mL(\ddot{\theta}cos\theta - \dot{\theta}^2sin\theta) \\ -mL(\ddot{\theta}sin\theta + \dot{\theta}^2cos\theta) \\ 0 \end{bmatrix} = \begin{bmatrix} F_1 + F_2 \\ R - (M+m)g \\ 0 \end{bmatrix} = \boldsymbol{F}_e$$

Finally we can write the system dynamics in terms of $\ddot{\theta}$ and $\ddot{x}$:

$$\ddot{\theta}\big(M + msin^2\theta\big)L = \left(\frac{2M+m}{m}F_2 - F_1\right)cos\theta + g(M+m)sin\theta - mL\dot{\theta}^2 sin\theta cos\theta \quad (2.1)$$

$$\ddot{x}(M + msin^2\theta) = F_1 - F_2 cos(2\theta) + msin\theta(L\dot{\theta}^2 - gcos\theta) \quad (2.2)$$

Simplifying this for our problem by substituting in constants, we can write the full state space equation:

$$\begin{bmatrix}\dot{x}\\ \ddot{x}\\ \dot{\theta}\\ \ddot{\theta}\end{bmatrix} = \begin{bmatrix}\dot{x}\\ \frac{\left(\frac{2M+m}{m}F_2 - F_1\right)cos\theta + g(M+m)sin\theta - mL\dot{\theta}^2 sin\theta cos\theta}{(M+msin^2\theta)}\\ \dot{\theta}\\ \frac{F_1 - F_2 cos(2\theta) + msin\theta(L\dot{\theta}^2 - gcos\theta)}{L(M+msin^2\theta)}\end{bmatrix} = \begin{bmatrix}f_1(\boldsymbol{x}, F_1, F_2)\\ f_2(\boldsymbol{x}, F_1, F_2)\\ f_3(\boldsymbol{x}, F_1, F_2)\\ f_4(\boldsymbol{x}, F_1, F_2)\end{bmatrix} \quad (2.3)$$

Using Lyapunov's indirect method, we can write the linearised equations about the equilibrium, $\boldsymbol{x}_e = [x_e, \dot{x}_e, \theta_e, \dot{\theta}_e]^T = [0,0,0,0]^T$, as:

$$\begin{bmatrix}\delta\dot{x}\\ \delta\ddot{x}\\ \delta\dot{\theta}\\ \delta\ddot{\theta}\end{bmatrix} = \begin{bmatrix}\frac{\partial f_1}{\partial x}\big|_{\boldsymbol{x}_e} & \frac{\partial f_1}{\partial \dot{x}}\big|_{\boldsymbol{x}_e} & \frac{\partial f_1}{\partial \theta}\big|_{\boldsymbol{x}_e} & \frac{\partial f_1}{\partial \dot{\theta}}\big|_{\boldsymbol{x}_e}\\ \frac{\partial f_2}{\partial x}\big|_{\boldsymbol{x}_e} & \frac{\partial f_2}{\partial \dot{x}}\big|_{\boldsymbol{x}_e} & \frac{\partial f_2}{\partial \theta}\big|_{\boldsymbol{x}_e} & \frac{\partial f_2}{\partial \dot{\theta}}\big|_{\boldsymbol{x}_e}\\ \frac{\partial f_3}{\partial x}\big|_{\boldsymbol{x}_e} & \frac{\partial f_3}{\partial \dot{x}}\big|_{\boldsymbol{x}_e} & \frac{\partial f_3}{\partial \theta}\big|_{\boldsymbol{x}_e} & \frac{\partial f_3}{\partial \dot{\theta}}\big|_{\boldsymbol{x}_e}\\ \frac{\partial f_4}{\partial x}\big|_{\boldsymbol{x}_e} & \frac{\partial f_4}{\partial \dot{x}}\big|_{\boldsymbol{x}_e} & \frac{\partial f_4}{\partial \theta}\big|_{\boldsymbol{x}_e} & \frac{\partial f_4}{\partial \dot{\theta}}\big|_{\boldsymbol{x}_e}\end{bmatrix}\begin{bmatrix}\delta x\\ \delta\dot{x}\\ \delta\theta\\ \delta\dot{\theta}\end{bmatrix} + \begin{bmatrix}\frac{\partial f_1}{\partial F_1}\big|_{\boldsymbol{x}_e} & \frac{\partial f_1}{\partial F_2}\big|_{\boldsymbol{x}_e}\\ \frac{\partial f_2}{\partial F_1}\big|_{\boldsymbol{x}_e} & \frac{\partial f_2}{\partial F_2}\big|_{\boldsymbol{x}_e}\\ \frac{\partial f_3}{\partial F_1}\big|_{\boldsymbol{x}_e} & \frac{\partial f_3}{\partial F_2}\big|_{\boldsymbol{x}_e}\\ \frac{\partial f_4}{\partial F_1}\big|_{\boldsymbol{x}_e} & \frac{\partial f_4}{\partial F_2}\big|_{\boldsymbol{x}_e}\end{bmatrix}\begin{bmatrix}\delta F_1\\ \delta F_2\end{bmatrix} \quad (2.4)$$

$$\begin{bmatrix}\delta\dot{x}\\ \delta\ddot{x}\\ \delta\dot{\theta}\\ \delta\ddot{\theta}\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0\\ 0 & 0 & \frac{mg}{M} & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & \frac{(m+M)}{ML}g & 0\end{bmatrix}\begin{bmatrix}\delta x\\ \delta\dot{x}\\ \delta\theta\\ \delta\dot{\theta}\end{bmatrix} + \begin{bmatrix}0 & 0\\ \frac{1}{M} & -\frac{1}{M}\\ 0 & 0\\ -\frac{1}{ML} & \frac{2M+m}{mML}\end{bmatrix}\begin{bmatrix}\delta F_1\\ \delta F_2\end{bmatrix} \quad (2.5)$$

The eigenvalues are given by $det(\lambda I - A) = \lambda^2(\lambda^2 - \frac{(m+M)}{ML}g) = 0$. Therefore, the system is unstable about $\boldsymbol{x}_e$ due to the right half plane pole, $\lambda = \sqrt{\frac{(m+M)}{ML}g}$. Additionally, the time constant of this unstable system is $\tau = \sqrt{\frac{ML}{g(m+M)}}$. Note, if $M >> m, \tau \to \sqrt{\frac{L}{g}}$, which is the time constant for a simple pendulum.

It can be proved that the inverted pendulum system is controllable by showing:

$$rank[\boldsymbol{B}\ \boldsymbol{AB}\ \boldsymbol{A^2B}\ \boldsymbol{A^3B}] = 4 \quad (2.6)$$

Therefore for any initial condition we can reach $\boldsymbol{x}_e$ in finite time under these linear assumptions.

2.2 Propagation of Quantisation Error

The state space model for the quantisation of the linearised inverted pendulum can be written as:

$$\boldsymbol{x}_t^{(2D)} = C\boldsymbol{x}_t + \boldsymbol{V}_t \qquad \boldsymbol{V}_t \sim \mathcal{U}\left(\begin{bmatrix}\frac{1}{\delta x}\\ \frac{1}{\delta\theta}\end{bmatrix}\right) \tag{2.7}$$

$$\boldsymbol{x}_t = A\boldsymbol{x}_{t-1} + B\boldsymbol{u}_t \tag{2.8}$$

Where A and B are the linearised system dynamics (valid for small time steps), and C is the linear transformation to a 2D state space, with quantisation noise $\mathbf{V}$.

Assuming the quantisation bin sizes, δx and $\delta\theta$, are small and that x and θ are independent within the bin, the quantisation noise can be modelled as uniform random variables with covariance, $cov(\boldsymbol{V}, \boldsymbol{V}) = \mathbb{E}[\boldsymbol{V}\boldsymbol{V}^T]$:

$$= \mathbb{E}\begin{bmatrix} x^2 & x\theta \\ \theta x & \theta^2 \end{bmatrix} = \begin{bmatrix} \int_{-\delta x/2}^{\delta x/2} x^2 \cdot \frac{1}{\delta x} dx & 0 \\ 0 & \int_{-\delta\theta/2}^{\delta\theta/2} \theta^2 \cdot \frac{1}{\delta\theta} d\theta \end{bmatrix} = \begin{bmatrix} \frac{\delta x^2}{12} & 0 \\ 0 & \frac{\delta\theta^2}{12} \end{bmatrix} \tag{2.9}$$

For simplicity, let $\delta x = \delta\theta$, and therefore, $cov(\boldsymbol{V}, \boldsymbol{V}) = \sigma_v^2 I$.

Kalman filtering can be used to find an optimal estimate $\hat{\boldsymbol{x}}_n = K[\boldsymbol{x}_n | \boldsymbol{y}_{n-k:n}]$ using the algorithm (derived in [3]).

Algorithm 1 Multivariate Kalman Filtering

1: **Given:** $\hat{\boldsymbol{x}}_n = K[\boldsymbol{x}_n|\boldsymbol{y}_{1:n}]$ and $\Sigma_n = \mathbb{E}[(\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n)(\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n)^T]$

Prediction:

2: $\bar{\boldsymbol{x}}_{n+1} = K[\boldsymbol{x}_n|\boldsymbol{y}_{1:n}] = A\hat{\boldsymbol{x}}_n + B\boldsymbol{u}_{n+1}$

3: $\bar{\Sigma}_{n+1} = \mathbb{E}[(\boldsymbol{x}_{n+1} - \bar{\boldsymbol{x}}_{n+1})(\boldsymbol{x}_{n+1} - \bar{\boldsymbol{x}}_{n+1})^T] = A\Sigma_n A^T + \Sigma_w$ ▷ $\Sigma_w = \mathbf{0}$

Update:

4: $\tilde{\boldsymbol{y}}_{n+1} = \boldsymbol{y}_{n+1} - C\bar{\boldsymbol{x}}_{n+1}$ ▷ Calculate Innovation residual

5: $S_{n+1} = \Sigma_v + C\bar{\Sigma}_{n+1}C^T$ ▷ Calculate Innovation Covariance

6: $\hat{\boldsymbol{x}}_{n+1} = \bar{\boldsymbol{x}}_{n+1} + \bar{\Sigma}_{n+1}C^T S_{n+1}^{-1}\tilde{\boldsymbol{y}}_{n+1}$ ▷ $\Sigma_v = \sigma_v^2 I$

7: $\Sigma_{n+1} = (I - \bar{\Sigma}_{n+1}C^T S_{n+1}^{-1}C)\bar{\Sigma}_{n+1}$

Thus we can find the optimal linear estimate of the covariance of Equation (2.7) from algorithm 1 line 7:

$$\Sigma_{n+1} = (I - \bar{\Sigma}_{n+1}C^T S_{n+1}^{-1}C)\bar{\Sigma}_{n+1} \tag{2.10}$$

$$= (I - \bar{\Sigma}_{n+1}C^T(\sigma_v^2 I + C\bar{\Sigma}_{n+1}C^T)^{-1}C)\bar{\Sigma}_{n+1}, \qquad where\ \ \bar{\Sigma}_{n+1} = A\Sigma_n A^T \tag{2.11}$$

For the univariate case this reduces to:

$$\sigma_{n+1}^2 = \bar{\sigma}_{n+1}^2(1 - \frac{C^2\bar{\sigma}_{n+1}^2}{\sigma_v^2 + C^2\bar{\sigma}_{n+1}^2}) \tag{2.12}$$

$$= A^2\sigma_n^2(1 - \frac{C^2A^2\sigma_n^2}{\sigma_v^2 + C^2A^2\sigma_n^2}) \tag{2.13}$$

It is obvious from this that as $\sigma_v^2 \to 0$, $\sigma_{n+1}^2 \to 0$. Furthermore, if the spectral radius, $\rho\left(A^2(1 - \frac{C^2A^2\sigma_n^2}{\sigma_v^2+C^2A^2\sigma_n^2})\right) < 1$, then the mean squared error from quantisation will reduce to zero. Since $0 < ||\frac{C^2A^2\sigma_n^2}{\sigma_v^2+C^2A^2\sigma_n^2}||_2^2 < 1$ if $\sigma_v^2 > 0$, a sufficient condition is that $A^2 \leq 1 \implies \rho(A) \leq 1$.

For the multivariate case, it can similarly be shown (by Gelfand's Theorem, $\rho(M_1...M_n) \leq \rho(M_1)...\rho(M_n)$) that $\rho\left(\bar{\Sigma}_{n+1}C^T(\sigma_v^2 I + C\bar{\Sigma}_{n+1}C^T)^{-1}C)\right) < 1$, therefore a sufficient condition for the decay of the covariance is $\rho(A\Sigma_n A^T) \leq \rho(\Sigma_n) \implies \rho(AA^T) \leq 1$. For the linearised dynamics derived in eq. (2.5) the eigenvalues, λ are given by $\lambda^2(1-\lambda)^2 = 0 \implies \rho(AA^T) \leq 1$. Therefore, the covariance decays to zero in the linearised multivariate case and the 2D state becomes a lossless estimate of the state.

The rate of decay for the univariate case can be determined with a first order approximation about $\sigma_v^2 = 0$:

$$\sigma_{n+1}^2(\sigma_v^2) = A^2\sigma_n^2(1 - \frac{C^2A^2\sigma_n^2}{\sigma_v^2 + C^2A^2\sigma_n^2}) \tag{2.14}$$

$$\approx \sigma_{n+1}^2(0) + \delta\sigma_v^2\frac{\partial\sigma_{n+1}^2(\sigma_v^2)}{\partial\sigma_v^2}|_{\sigma_v^2=0} \tag{2.15}$$

$$= \frac{\delta\sigma_v^2}{(AC\sigma_n)^2} \tag{2.16}$$

As shown above, the mean square error decays with the square of the bin-size. Similarly, linearising about $\sigma_n^2 \approx 0$ gives $\sigma_{n+1}^2 = A^2$ for the linear case, which is constant.... Need to go to second order...

2.3 Neural Network Losses

2.4 Network Biases

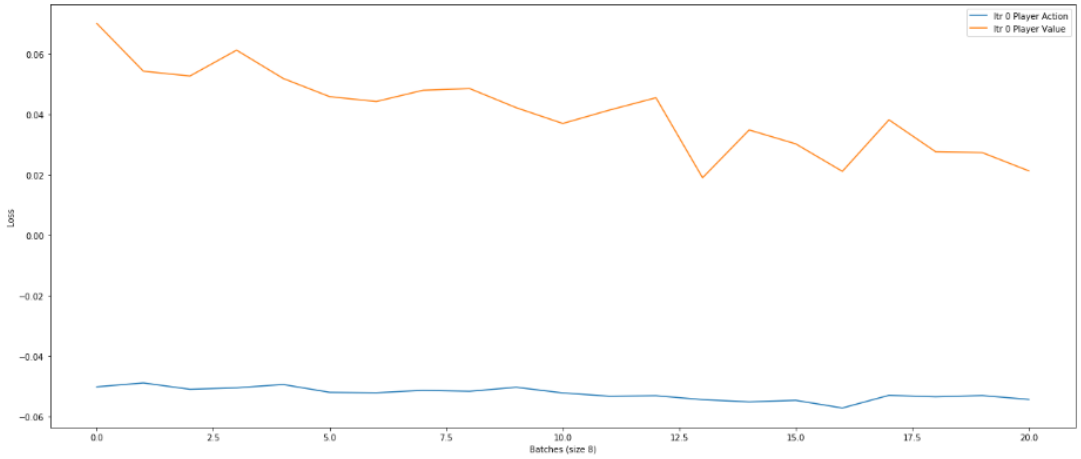

Figure 2.1: The losses for both the player and adversary neural networks after one set of training examples.

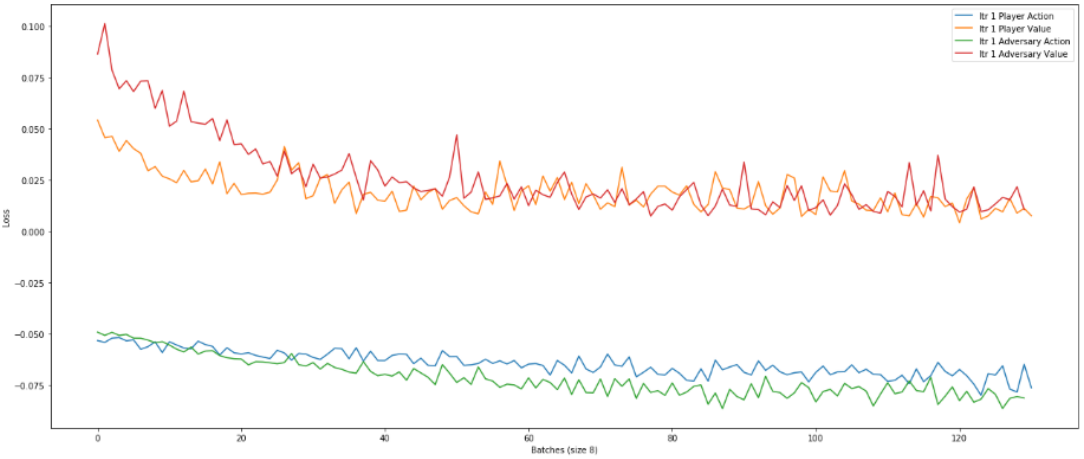

Figure 2.2: The losses for both the player and adversary neural networks after two sets of training examples.

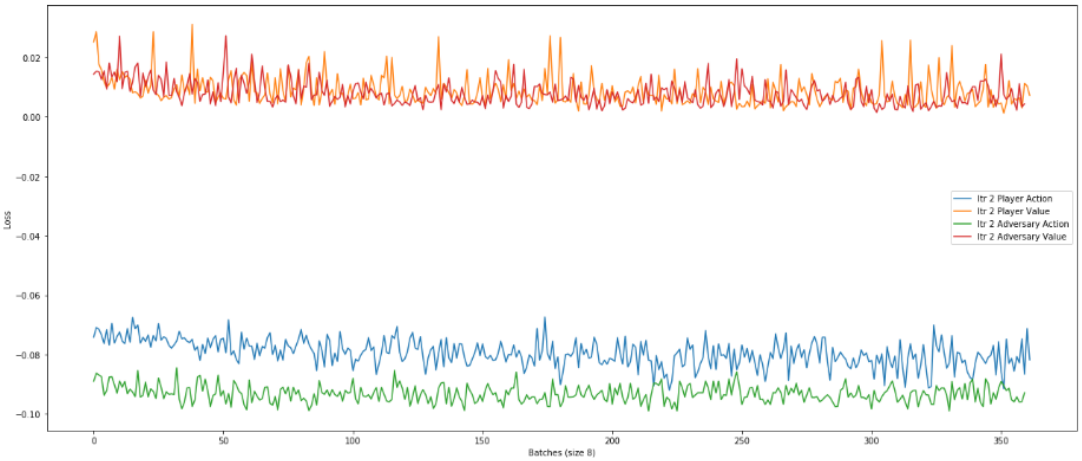

Figure 2.3: The losses for both the player and adversary neural networks after three sets of training examples.

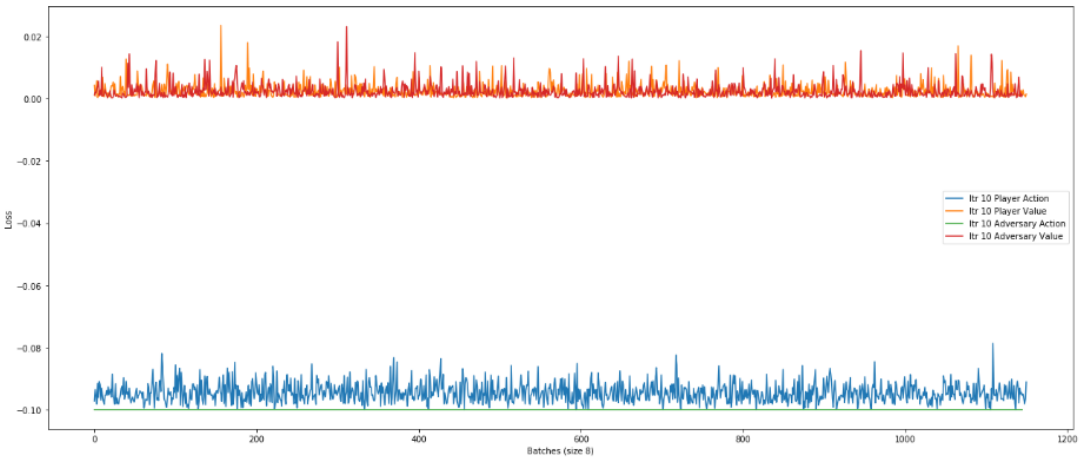

Figure 2.4: The losses for both the player and adversary neural networks after ten sets of training examples.

3 References

[1] M.C. Smith, I Lestas, *4F2: Robust and Non-Linear Control* Cambridge University Engineering Department, 2019

[2] G. Vinnicombe, K. Glover, F. Forni, *4F3: Optimal and Predictive Control* Cambridge University Engineering Department, 2019

[3] S. Singh, *4F7: Statistical Signal Analysis* Cambridge University Engineering Department, 2019

[4] Arthur E. Bryson Jr, *Optimal Control - 1950 to 1985*. IEEE Control Systems, 0272-1708/95 pg.26-33, 1996.

[5] I. Michael Ross, Ronald J. Proulx, and Mark Karpenko, *Unscented Optimal Control for Space Flight*. ISSFD S12-5, 2014.

[6] Zheng Jie Wang, Shijun Guo, Wei Li, *Modeling,Simulation and Optimal Control for an Aircraft of Aileron-less Folding Wing* WSEAS TRANSACTIONS on SYSTEMS and CONTROL, ISSN: 1991-8763, 10:3, 2008

[7] Giovanni Binet, Rainer Krenn and Alberto Bemporad, *Model Predictive Control Applications for Planetary Rovers*. imtlucca, 2012.

[8] Raković, Saša, *"Robust Model-Predictive Control*. Encyclopedia of Systems and Control, pg.1-11 ,2013.

[9] Russ Tedrake, *Underactuated Robotics*. MIT OpenCourseWare, Ch.3, Spring 2009.

[10] Richard S. Sutton and Andrew G. Barto, *Reinforcement Learning: An Introduction (2nd Edition)*. The MIT Press, Cambridge, Massachusetts, London, England. 2018.

[11] I. Carlucho, M. De Paula, S. Villar, G. Acosta . *Incremental Q-learning strategy for adaptive PID control of mobile robots*. Expert Systems with Applications. 80. 10.1016, 2017

[12] Yuxi Li, *Deep Reinforcement Learning: An Overview*. CoRR, abs/1810.06339, 2018.

[13] Sandy H. Huang, Martina Zambelli, Jackie Kay, Murilo F. Martins, Yuval Tassa, Patrick M. Pilarski, Raia Hadsell, *Learning Gentle Object Manipulation with Curiosity-Driven Deep Reinforcement Learning*. arXiv 2019.

[14] David Silver, Julian Schrittwieser, Karen Simonyan et al, *Mastering the game of Go without human knowledge*. Nature, vol. 550, pg.354–359, 2017.

[15] David Silver, Thomas Hubert, Julian Schrittwieser et al, *A general reinforcement learning algorithm that masters chess, shogi and Go through self-pla*. Science 362:6419, pg.1140-1144, 2018.

[16] S. Thakoor, S. Nair and M. Jhunjhunwala, *Learning to Play Othello Without Human Knowledge* Stanford University Press, 2018 `https://github.com/suragnair/alpha-zero-general`

[17] HowlingPixel.com, *Elo Rating System*. `https://howlingpixel.com/i-en/Elo_rating_system` acc: 11/03/2019. published 2019