Policy Gradient Based Deceptive Reinforcement Learning

Waneya Iqbal 919750, Honglong Zhang 985262
iqbalw@student.unimelb.edu.au, honglongz@student.unimelb.edu.au

Under Supervision of:
Associate Prof. Tim Miller
tmiller@unimelb.edu.au

COMP90055 Research Project, First Semester 2020, Credit Points: 25
The University of Melbourne

Abstract

In this report, we study the problem of deception in Artificial Intelligence using policy-gradient based reinforcement learning. Policy-gradient provides two distinct advantages which are the generation of stochastic policy and the ability to handle continuous action spaces with ease, and we exploited both these advantages to study deception in Artificial Intelligence We first uses a model that exploits stochastic policies to achieve deception in path-planning in discrete action space using *dissimulation* – a deception strategy that *'hides the truth'*. We evaluated this model computationally. Results conclude that the policy is indeed deceptive and hides the true destination of the agent. Moreover, policy gradient scales well to continuous action spaces and offers an advantage to study deception in such spaces. We present a *dissimulation* based deceptive model in continuous action space using the actor-critic method which is an enhanced form of policy gradient. We evaluated our model computationally in the continuous space environment. Results conclude that policy is indeed deceptive and policy gradients can indeed be used in continuous action space for deception.

1 Introduction

Deception in Artificial Intelligence is gaining more and more attention in the research community because of its numerous interesting applications. Consider the problem of escorting a VIP to one of five possible destinations. If the destination is recognised by an observer, the observer plans to assign an assassin to the destination. Deceptive path planning is very important to protect the VIP because it ensures that the true destination of VIP is not revealed for as long as possible. Moreover, if the VIP travels more than once, for example on the next day, between the same source and destination, the policy must produce a different deceptive path. Additionally, in continuous action space, consider an AI that has reached the penalty-kicks round of a football match with five chances to score the goal and win the round by scoring more goals than the adversary. To win the round, it not only needs to be deceptive to trick the goal-keeper and maximise the chance of scoring a goal at a given shot but also it needs to select a different deceptive action during each shot to win the round. Both problems require stochastic deceptive policies.

Policy-gradient based reinforcement learning is a natural candidate in these kinds of deception problems because not only it can generate stochastic policies, but also it handles continuous action space with relative ease. In this report, we present utilisation of policy gradient to generate deceptive stochastic policies in both continuous and discrete action space.

We extend the deceptive *dissimulation* based irrationality model for the path planning problem in a discrete action space presented in [1] to achieve deception in continuous action environment of OpenAI [2]. The model uses a parameter α ($0 \leq \alpha \leq 1$) that control the extent of deception by balancing the choice of the rational (an action that leads closest to the real goal) and irrational (an action that does not lead closer to any goal) actions. We used various values of α in our experiments and results show that the parametric irrationality model successfully deceives in the continuous action space as well. Moreover, increasing α increases the deception and vice versa. However, as the agent ultimately converges to the real goal, the real destination or target is revealed.

Moreover, we also present a new negative irrationality model and use it with policy-gradient to generate deceptive stochastic policies for path planning in discrete action space using the Python Path Planning Project (P4) framework [3]. The negative irrationality model takes action that leads closer to all goals. We ran 360 trials of deceptive path generation using our policy and compared it with the baseline ambiguity model presented in [1]. The results concluded that our model is more deceptive in the initial phase (up to 20% of the full deceptive path) compared to the baseline. After that baseline models perform better in deception. Different goal pruning techniques used in our policy and the baseline explain the result because the baseline uses a more sophisticated pruning technique. However, based on better deception in initial 20% of the path, we conclude that our model has the potential to perform better than baseline after replacing our pruning technique with that described in the baseline ambiguity model in [1].

The rest of the report is organised as follows. We first present the background and related work in section 2, based on which our work has been built. After that section 3 covers two deceptive models. One of them is used in continuous action space and presented in section 3.1 along with the experiments and results. The other one is presented in section 3.2 and it covers deception in discrete path planning along with corresponding experiments and results. Finally, the conclusion is presented in section 4.

2 Background and Related Work

The first and, to our knowledge, the only general objective method to achieve deception is presented by the military strategists Walley and Bell in [4] and [5] who presents deception as a distortion of reality. They presented that deception can only be achieved by showing the false (simulation) or hiding the truth (dissimulation).

Masters and Sardina in [6] apply dissimulation and simulation to study deception in Artificial Intelligence Path Planning in the model-based environment and formalises it as the problem of finding a path for an agent such that the probability of an observer identifying agent's final destination before it has been reached, is minimized. A naïve observer modelled as a probabilistic goal recognition system is used to determine the probability of each goal at any given point along a path. The unlikelihood of an observer belief for the actual goal was used to assess successful deception. They used various strategies to generate deceptive paths and assess them based on the naïve model.

Yang, Miller and others in [1] extended the concept of deceptive path planning to Reinforcement Learning and addressed the problem of preserving the privacy of the reward function of Reinforcement Learning through the dissimulation approach of deception in a model-free MDPs. Two models have been used which involves ambiguity to maximise the entropy of the observer, and an irrationality model which takes semi-rational actions. These models were evaluated using intention recognising algorithm based on a naïve observer (observer does not think that it is being deceived) and non-naïve human participants to estimate the likelihood of different destinations. The results show that the agents are successful in hiding the reward function, but reward functions are revealed as more and more actions are executed. The irrationality model deceives more than the ambiguity model but received the lesser discounted reward on actual reward functions.

3 Policy-Gradient Based Deceptive Reinforcement Learning

In this report, we extend the related work to study deception in Artificial Intelligence using policy-gradient based Reinforcement Learning.

Compared to other reinforcement learning approaches such as q-learning or approximate q-learning etc which does not scale well to continuous action spaces, policy gradient offers an advantage in this area. Thus, we have explored deception in continuous action space using a parametric policy based on policy-gradient. The details of this work are presented in section 3.1.

Moreover, in some cases, stochastic policies are inherently more deceptive then deterministic policies. We have exploited such a stochastic policy to add deception in discrete path planning. The details are presented in section 3.2.

3.1 Deception in Continuous Action Space

Physical world is continuous in terms of action space for most artificial intelligence agent. For example, actions of a robot traversing in the environment or self-driving cars are all continuous. Generating deceptive policies in such environments requires approaches that scale well to continuous action spaces. In this section, we present a policy-gradient based actor-critic neural network that utilises the irrationality model of deception presented in [1] to achieve deception in continuous action space. The actor-critic model reduces the variance in reward updates and, thus, accelerates the learning process [7] of the deceptive model.

3.1.1 Hypotheses

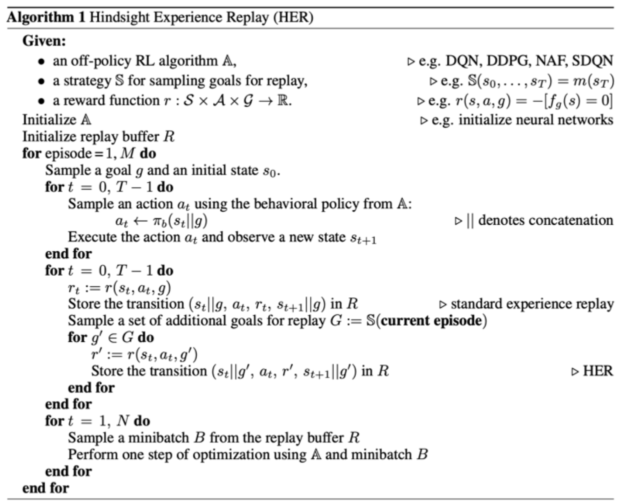
We hypothesise that the actions sampled from continuous action space of OpenAI [2] by the trained actor-critic neural network can be used with irrationality model of deception in presented in [1]. Though the irrationality model in [1] was used for deception in discrete action space for deception in path planning, we believe it can well be used to achieve deception in continuous action space. We believe that the irrationality model will make the agent converge to the real goal in continuous action space of [2], while trying to confuse the observer. The confusion results from irrationality model and it means that the probability of the real goal and fake goal fluctuates at each instant of action selection so that the observer cannot identify the real goal as long as possible. We believe that creating such confusion would make the agent deceptive in continuous action space. The details of the method, model and further explanation regarding action selection are presented in section 3.1.2.

3.1.2 Method

In this section, we present the model of deception that we used for policy gradient based deception in continuous action space, along with the deceptive policy of action selection. We also present Hindsight Experience Replay first, that solves an important problem associated with policy training.

3.1.2.1 Hindsight Experience Replay

Sparse reward poses a challenge while training a deep neural network, especially in robotics. Due to sparse rewards, the robot can get a positive reward only when it reaches the goal, and all the other actions may not receive a reward. Moreover, the episodes are terminated (our training uses Monte-Carlo) after a fixed number of actions and the goal is never reached. This leads to a problem that the rewards received during different episodes have high variance, depending upon the number of actions taken to reach the goal or whether the goal is reached or not. Overall, this high variance problem affects the convergence during the training. Hindsight Experience Replay (HER) solves this problem. The intuition behind HER is very simple, and it involves taking advantage of even those episodes out of all episodes $\{e_0, e_1 ...., e_n\}$ in which the robot does not reach the goal [8] and receives a reward. This information can be harvested by adopting an off-policy reinforcement learning algorithm and replay the episodes but replace g by s_t, where g is the goal and s_t is the last state of previous episode [8], and reaching s_t will provide reward equal to that of reaching goal. HER is used in this paper to accelerate the training process.



```
Algorithm 1 Hindsight Experience Replay (HER)
Given:
  • an off-policy RL algorithm $\mathbb{A}$,                       ▷ e.g. DQN, DDPG, NAF, SDQN
  • a strategy $\mathbb{S}$ for sampling goals for replay,          ▷ e.g. $\mathbb{S}(s_0, \dots, s_T) = m(s_T)$
  • a reward function $r : \mathcal{S} \times \mathcal{A} \times \mathcal{G} \to \mathbb{R}$.   ▷ e.g. $r(s, a, g) = -[f_g(s) = 0]$
Initialize $\mathbb{A}$                                              ▷ e.g. initialize neural networks
Initialize replay buffer $R$
for episode = 1, $M$ do
    Sample a goal $g$ and an initial state $s_0$.
    for $t = 0, T-1$ do
        Sample an action $a_t$ using the behavioral policy from $\mathbb{A}$:
                $a_t \leftarrow \pi_b(s_t||g)$                       ▷ || denotes concatenation
        Execute the action $a_t$ and observe a new state $s_{t+1}$
    end for
    for $t = 0, T-1$ do
        $r_t := r(s_t, a_t, g)$
        Store the transition $(s_t||g, a_t, r_t, s_{t+1}||g)$ in $R$    ▷ standard experience replay
        Sample a set of additional goals for replay $G := \mathbb{S}$(current episode)
        for $g' \in G$ do
            $r' := r(s_t, a_t, g')$
            Store the transition $(s_t||g', a_t, r', s_{t+1}||g')$ in $R$   ▷ HER
        end for
    end for
    for $t = 1, N$ do
        Sample a minibatch $B$ from the replay buffer $R$
        Perform one step of optimization using $\mathbb{A}$ and minibatch $B$
    end for
end for
```

3.1.2.2 Irrationality Model of Deception

Irrationality model is proposed in [1]. This model is based on the cost difference presented in [9] and Irrationality Measure as provided in [1].

If $\vec{o}$ denotes the observation sequence so far (a sequence of (s, a)) and Q is the q-value of a given state-action pair, the cost difference ($\Delta'_{ri}(\vec{o})$) for given goal i can be calculated by equation [a]. Cost difference effectively measures how far the observed sequence so far deviates from the optimal path of each goal.

$$\Delta'_{ri}(\vec{o}) = \sum_{(s,a\in\vec{o})} \frac{Q_{ri}(s,a)}{max_{a'\in A} Q_{ri}(s,a')} \qquad [a]$$

The Irrationality Measure can then be calculated as follows. For an observed sequence of state-action pairs $\vec{o}$, the $irrationality\ measure$ of $\vec{o}$ for the reward function r_i is given by equation [b]:

$$\text{IM}(\vec{o}) = 1 - \max_{r_i \in R} \Delta'_{r_i}(\vec{o}) \qquad [b]$$

Where r_i refers to the reward function of ith goal. IM ($\vec{o}$), effectively is a measure of divergence from all goals – it is irrational that the agent does not move towards at-least one goal [1]. Now, the policy based on irrationality model can be defined by equation [c].

$$\pi^D(\vec{o}, s) = \text{argmax}[\underbrace{(1-\alpha)Q'_r(s,a)}_{\text{Optimal}} + \underbrace{\alpha\, IM(\vec{o} \cdot (s,a))}_{\text{Irrational}}] \qquad [c]$$

The policy outputs the action which has the maximum weighted sum of optimal Q-value and irrational value, weighted by the parameter α which controls the extent of irrationality. When α equals to zero, the agent will only take the action with optimal Q-value (i.e. closest to real goal) into account and selects it. Effectively, the agent is honest. With α equals to 1, the agent will be completely irrational and does not converge to the real goal. With $0 < \alpha < 1$, we achieve deception depending on the value of α. The larger the parameter α is, the more irrational the agent will be. Moreover, the optimal Q-value and IM value is normalized as well. In this way, we value each goal equally and the result is not sensitive to α.

3.1.2.3 Deceptive Policy for Continuous Action Space

To generate a deceptive policy for continuous action space, the Q value of each state-action pair is firstly obtained by actor neural network together with the critic neural network. Given $s_i \in$ S represents the state and, $a_j \in A$ is the action set, "Actor ()" is the actor neural network and "Critic ()" means the critic neural network. Assuming the agent is in state $s_{current}$, then action for each goal is $\{Actor_j(s_{current})\}$, where j is the j_{th} goal (real goal and fake goals), finally Q-value for each goal is obtained by Q = {Critic($s_{current}, action_j$)}. After this process, irrational measure IM for each goal can be calculated by Equation [b] with the above-mentioned setup.

For the final action selection, we have to ensure that Equation [c], which is originally defined for discrete space can be used with continuous action space. We made it possible by sampling actions from infinite actions because it is not possible to evaluate these infinite actions. We decided to sample five action though it can be changed. These actions are chosen randomly from the continues action space of the environment. Besides those five actions, the optimal action is obtained from the Actor neural network. It takes the current state of the agent ($state_{current}$) and the position of real goal ($target_{real}$) as input to generate the optimal action, given by the equation [d].

$$action_{optimal} = \text{Actor}(state_{current}, target_{real}) \quad [d]$$

Now the action space is compatible with Equation [c] which can be used to achieve both deception and convergence as defined in section 3.1.2.2

3.1.3 Experiments

This experiment aims at testing whether the irrationality model proposed in [1] works for continuous action space. It also involves evaluating the impact of irrationality parameter (α) on the performance of the deceptive policy.

Independent variables

- The agent that generates deceptive policies is an independent variable. Difference agents can generate different policies. However, in our work we only used one agent.
- The irrationality parameter α which controls the extent of deception is the main independent variable of our experiment. We experimented with $\alpha \in$ {0, 0.1, 0.2, 0.3}

Measures

Two measures are adopted:

- The path cost of each deceptive policy.
- The probability of each goal being regarded as the real target. After the agent takes action, we will use the critic neural network to get the probability distribution(probability are calculated using softmax function) of each goal in the environment of [2] at each instant of action selection from sampled actions. The procedure is defined in section 3.1.2.2

Experiment Setup

- An actor neural network with three layers.
- A critic neural network with three layers.
- One real goal in the environment.
- Two fake goals in the environment.

Moreover, the actor and critic neural network are pre-trained in the OpenAI environment with a 500*500*500 space size.

Environment

OpenAI is an open-source reinforcement learning environment, which combines various scenarios to test and evaluate artificial intelligence algorithms and approaches. For this project, we selected the robotics environment of "FetchReach-v1" [2]. The environment involves picking the red ball and throwing it as shown in Figure 1. We modified the environment so that it contains more than one ball to pick up as shown in Figure 2. The red ball is the target ball to pick, whereas the black balls are fake. The objective is picking the red ball while being deceptive so that the observer cannot recognise, as long as possible, that which ball is the target for pickup.

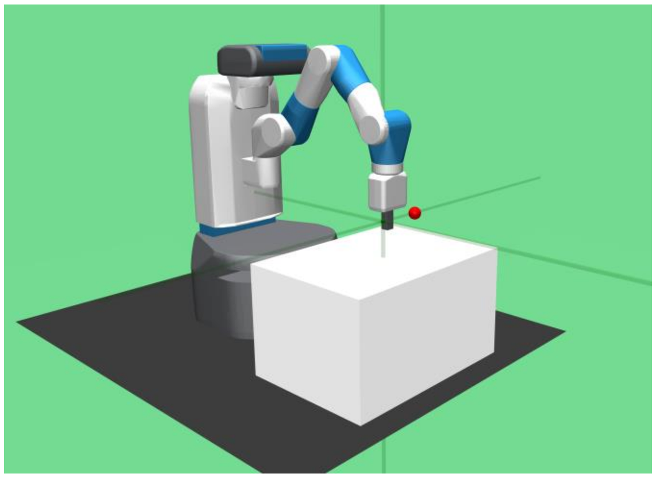


Figure 1 FetchReach-v1

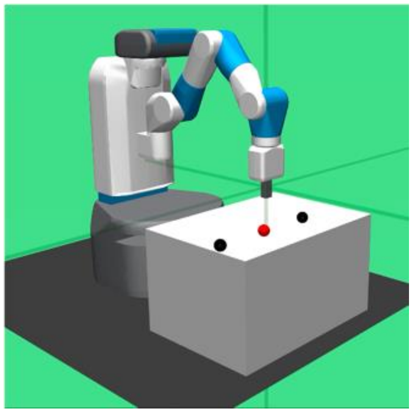


Figure 2: Red ball is the real goal; black balls are fake

3.1.4 Results

The results of our experiments are based on the experimental setup presented in 3.1.3. We first evaluated the honest behaviour of agent (i.e. when α = 0). The results are presented in Figure 3. After that, we make the agent deceptive by setting the value of parameter α = 0.3. The behaviour is described in Figure 4 and Figure 5. Our results further shows that as the value of α increases the agent becomes more deceptive and it is presented in Figure 6. The details of the results are as follows.

Figure 3 illustrates how the probability of real goal and one of the two fake goals change when the agent is honest. Figure 3 shows that when α equals to 0, the robot will always take the optimal action and go directly to the real goal. Resultantly, the probability of "the real goal" increases in a few steps, whereas that of fake goal decreases. This means the observer can recognize the target easily and quickly because the agent is honest.

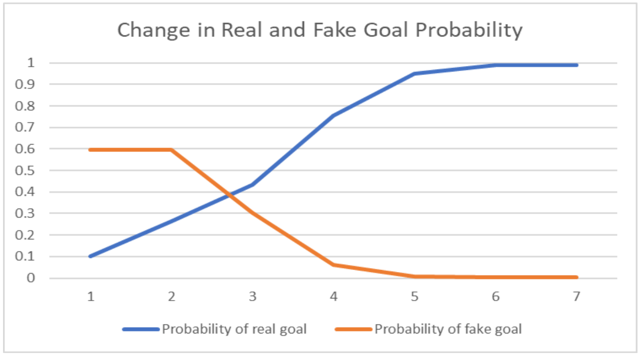


Figure 3: Real Vs Fake Goal Probabilities at α =0 (honest)

In contrast to the honest behaviour depicted in Figure 3, Figure 4 and Figure 5 shows the probabilities of the real goal and one of the fake goals respectively when the agent is deceptive. We take α = 0.3 as an example to depict the behaviour in which the robot will take the deceptive policy and the observer will be confused. For instance, the probability of real goal at step 6 is 0.74, which drops to 0.05 in step 7 as shown in Figure 4. A similar probability distribution can be observed in Figure 5 for one of the Fake Goal. An example is step 8 and 9 where probability changes from 0.25 to 0.90, respectively. This means that each step is deceptive and for the observer, it is difficult to judge which one is the real goal. In this way, the robot achieves deception. However, the trendline in Figure 4 and Figure 5 specifies that as the robot takes more and more actions (steps), the probability trend of "real goal" increases whereas that of fake goal decreases. This trend is inevitable because the final aim of the robot is to reach the real goal. As it approaches the real goal, the robot intention becomes more obvious and the deception decreases.

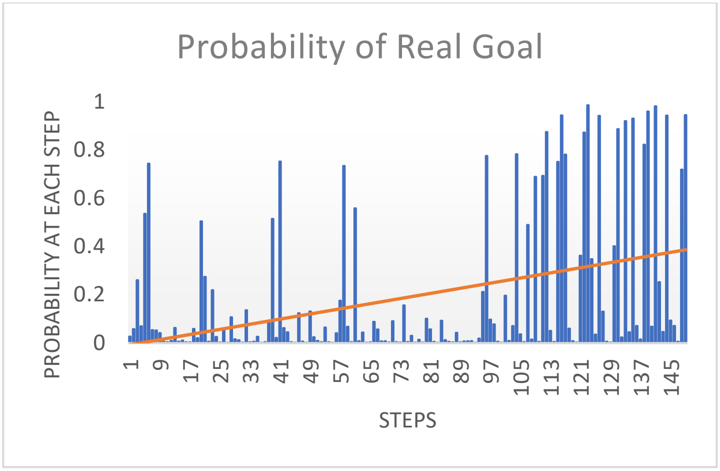


Figure 4: Probability of Real Goal when α = 0.3

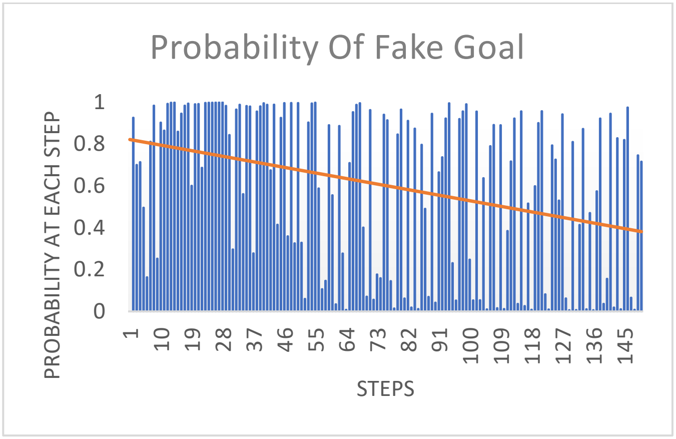


Figure 5: Probability of one Fake Goal when α = 0.3

With other values of α such that $0 < \alpha < 1$, the changes in the probability of the real and fake goals will show a similar trend as that of Figure 4 and Figure 5, and that's why they are not presented.

However, as the value of α increases the agent become more irrational and takes a longer path to reach the real goal because it takes more irrational actions. Resultantly the path cost increases. This is depicted in Figure 6 which compares path cost with various values of α. These results show that the higher the irrationality, the higher are the number of steps (thus higher the cost), and more deceptive will the agent be. Figure 6 shows the result.

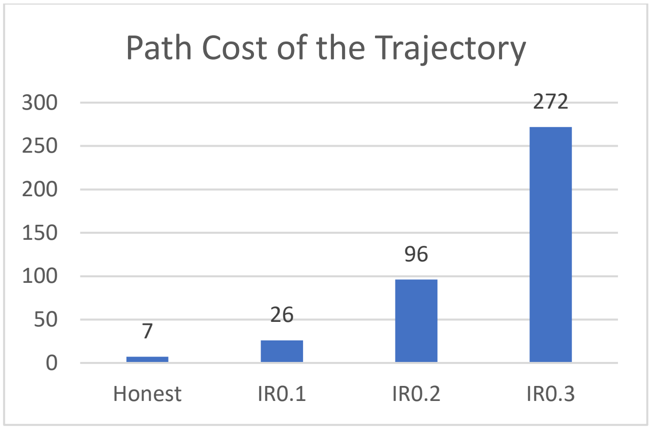


Figure 6: The Path Cost

3.2 Deception in Discrete Action Space

In this section, we present the application of policy gradients for deceptive path planning in a discrete action space.

Stochastic policies are inherently deceptive in nature. To emphasise the importance of stochastic policies in deception, consider the example of Rock, Paper and Scissor game. Playing with a deterministic policy will ultimately result in the adversary to recognise the deterministic policy. After that, all games will be lost. Playing stochastically in such games add deception and is the best policy to win. We extend this concept to develop a deceptive hypothesis for path planning, presented in the following section.

3.2.1 Hypothesis

Policy gradients can generate stochastic policies that can be used for deception. Based on this, we plan to use a stochastic policy generated by policy-gradients for increasing deception in path planning. Clearly, following a complete stochastic policy for action selection in path planning does not make any sense because the agent will keep moving randomly. As a result, we have proposed a negative irrationality model for action selection which leads closer to all goals. We believe such an action does not reveal the true goal. Our hypothesis includes maximising the probability of selecting this action while generating the stochastic policy through policy gradient. If p_{best} is the probability of selecting negative irrational action, we aim to maximise this probability during training. The resultant stochastic policy will select actions proportional to their probabilities and, as a result, negative irrational action is frequently selected with probability p_{best}. However, the stochastic policy will occasionally select other actions with a low probability proportional to (1 - p_{best}). We hypothesise that this occasional random selection of action further improves deception. It also ensures that a different trajectory is generated during each episode. The details of the work are presented in the following sections.

3.2.2 Method

This section describes the negative irrationality model and the Stochastic Policy used to achieve deception. The details are as follows.

3.2.2.1 Negative Irrationality Model

The irrationality measure defined by equation [b] in section 3.1.2.1 favours action that does not lead the robot closer to any goal – it is irrational not to make progress towards at least one goal [1]. Conversely, the policy based on negative irrationality model favours action that leads closer to as many goals as possible – action closer to all goals make the agent deceptive. At each step from start-point to point-b, the agent takes action that leads it closer to all goals.

To calculate the closeness of an action a ∈ A to all goals G, we first calculate the divergence D of an action at state s from all goals using equation [e] :

$$D(s,a) = \sum_{g\in G} \left(distance_g(s,a) - distance_g\ (s,a^*) \right) \quad [e]$$

Where $distance_g(s,a)$ represents the heuristic distance from state s'(state after taking action a from state s) to goal g, and $a^*$is the optimal action. Inversing $D(s,a)$ results in the action closest to all goals. We achieved this inversion during policy-gradient training, and it is described in the next section.

3.2.2.2 Deceptive Stochastic Policy in Discrete Path Planning

Linear Regression is used as the machine learning approach to train the policy gradient such that h(s, a) represents the linear combination of parameter vector $\boldsymbol{\theta}$ and feature vector $\boldsymbol{f(s,a)}$.

$$h(s,a) = \boldsymbol{\theta}^{\boldsymbol{T}} . \boldsymbol{f(s,a)} \quad \textbf{[}f]$$

$D(s,a)$ described in equation [e] is used as the main deceptive feature of Linear Regression, which receives a negative reward during the policy gradient training phase. Thus, the higher the divergence D of an action a, the lower will be the reward it receives. Thus, the least divergent action – which in fact is the action that takes closer to all goals – will become most probable in the stochastic policy returned by the policy gradient. The second feature includes receiving a small negative reward for each step taken. Lastly, reaching the real goal is heavily rewarded positively while reaching a fake goal is heavily rewarded negatively. Soft-max function over h (s, a) is used to get the stochastic policy probability distribution. The resultant policy is denoted by $\pi_{first-stage}$ and the action it returns is denoted by $a_{first-stage}$. During the training, the probability of most divergent action, denoted by p_{best}, will become highest, and it will be selected most of the time. Other actions are selected less frequently with probability proportional to (1 - p_{best}). Selecting these less frequent actions adds further to deception. Figure 7 presents one such deceptive plan based on this policy $\pi_{first-stage}$.

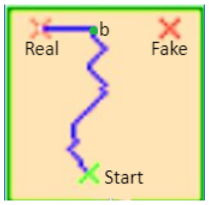


Figure 7: Stochastic Deceptive Path (Start-point to point-b)

The discussion until now does not ensure convergence to the real goal. To ensure convergence we use a simple pruning technique based on a parametric stochastic second stage action selection policy denoted by $\pi_{deceptive}$. In actual, $\pi_{deceptive}$ is used from start-point to real-goal in Figure 7 that returns $a_{deceptive}$. During the path from start-point to point-b, $\pi_{deceptive}$ returns $a_{first-stage}$. As the agent reaches point b at which it is no longer possible to take action that leads closer to both goals, it starts zigzagging left and right at point b (we call this behaviour *confusion* in action selection). At this point $\pi_{deceptive}$ selects action that leads closest to Real Goal (denoted by $a_{real-goal}$) with a probability equals to parameter β (p_β) such that (0.5 < β ≤ 1). Further, $\pi_{deceptive}$ selects $a_{first-stage}$ with probability 1 – β ($p_{1-\beta}$). Mathematically, $\pi_{deceptive}$ returns $a_{deceptive}$ as given by equation [g].

$$a_{deceptive} = \begin{cases} a_{first\ stage} & if\ no\ pruning \\ \left(a_{real-goal\ with\ p_\beta}\ or\ a_{first-stage\ with\ p_{1-\beta}}\right) & if\ pruning \end{cases} \quad [g]$$

Effectively from point-b, $\pi_{deceptive}$ selects $a_{first\ stage}$ with a probability below 0.5, whereas $a_{real-goal}$ is selected with a probability above 0.5 (because 0.5 < β ≤ 1). This ultimately ensures convergence to real-goal. Figure 7 emphasis $\pi_{deceptive}$ with β = 1 from point-b to real-goal. With β = 1, the agent moves deterministically towards real-goal from point-b.

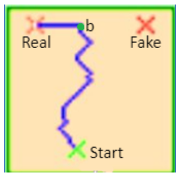
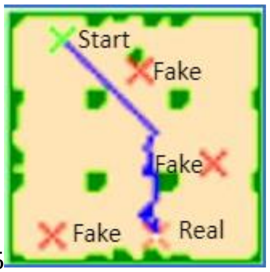


Figure 8: β = 1, the agent moves deterministically from point-b to real-goal

Another deceptive path based on $\pi_{deceptive}$ with β = 0.85 is given in Figure 9.



5

Figure 9: Deceptive Path with β = 0.85

3.2.3 Experiments

P4 is also known as the Python Path Planning Project and is described in [3]. It is a path planning simulator and framework that can be used for evaluating and benchmarking path planning algorithms. It is possible to use various maps in the format specified in [3] and specify the agent's Start position, multiple goals and obstacles for path planning.

In our report, this framework has been used to evaluate deception in the path planning using policy gradient. Set S represents all possible states within the map. A state s ∈ S is represented by a point in the map. Set A represents all possible discrete action and a ∈ A, represents a specific action at state s. At a state s ∈ S, the agent takes an action a ∈ A and end up in the state s' ∈ S.

The goal of the experiment is testing the hypothesis presented in section 3.2.1 i.e. it involves determining whether the policy $\pi_{deceptive}$ shows deception in path planning, and also the extent to which the path returned by the policy is deceptive.

We consider the Boolean variable Deception, denoted by D as the dependent variable for the experiment. This dependent variable represents whether the *path* generated by the policy $\pi_{deceptive}$ is deceptive or not when one of the independent variables is changed while keeping others constant. The independent variables for the experiment are as follows.

Observation density is the first independent variable. As per hypothesis, the agent will ultimately converge to the goal, reveals its true destination and will no longer be deceptive. Thus, it is important to measure how long the agent remains deceptive. Observation density captures this. It is simply the percentage of observations from the full path of the deceptive agent, that is considered by intention recognition algorithm to return the probability distribution of all goals. For instance, the observation density of 10% means first 10% observations from the deceptive path are considered by the intention recognition algorithm. This is explained in Figure 10.

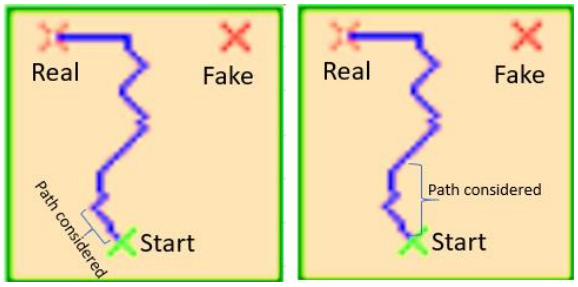


Figure 10 Lower observation density on the left side (almost 10% path considered). Higher observation density on the right side (almost 35% path considered).

Obstacles in the path of the agent affect its ability to deceive because the planning algorithm needs to plan to avoid these obstacles, sometimes at the cost of deception. We used five different maps with increasing level of obstacles, ranging from an empty map with no obstacles to maze type map with many obstacles in which the path to real-goal is reasonably strict and offers little chance of deception. These maps are presented in Figure 11.

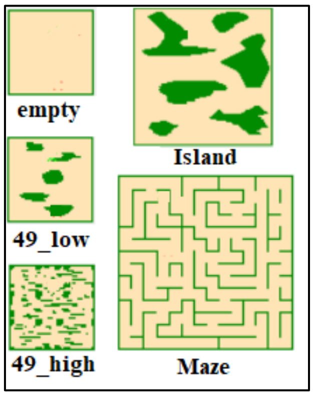


Figure 11: Maps used in experiments

The start position of the agent, the position of the real goal and fake goals, and the number of fake goals are important and all but the first one are used to directly compute Divergence which is defined in equation [e]. Moreover, the start position is important because if it is very close to the real goal, there is little chance of deception. However, each of these variables is not considered separately as an independent variable. Instead, an independent variable *configuration* is used to capture these four items.

The second stage action selection parameter β which is explained in section 3.1.2.3 influences the deceptive path. However, in our experimental trial, β is considered constant with a value of 1.

The naïve cost-based probabilistic intention recognition algorithm which is proposed in [9] is used to measure the deception. The algorithm compares the cost of path returned by $\pi_{deceptive}$ with the optimal path cost to each goal, and generate the probability distribution of the agent's possible destination. We consider the path returned by $\pi_{deceptive}$ to be deceptive if the P(real goal) as returned by the algorithm in [9] is less than or equal to the probability of at least one fake goal.

Lastly, the parametric stochastic policy $\pi_{first-stage}$ is pre-trained on various maps to ensure it maximises the probability of negative irrational action as specified in section 3.2.2.2.

3.2.4 Results

Having defined the experiment's constant, dependent and independent variables and the criteria to measure these in section 3.2.3, we conducted 360 trials of deceptive path generation and used the naïve intention recognition algorithm to detect deception. For each map, we used 8 different *configurations* such that the start position and position of real and fake goals are randomly generated, whereas 4 out of 8 contains 2 fake goals and remaining 4 contains 4 fake goals. For each map-configuration pair, we took observations with densities ranging from 10% to 90% with a granularity of 10%, which is plotted on the x-axis. Thus, for a given map and observation density we have eight configurations, whose deceptive paths are observed by the naïve intention recognition algorithm; and the count of deceptive outcomes (out of 8) are presented as the label on each map's line graph against the observation density in Figure 12.

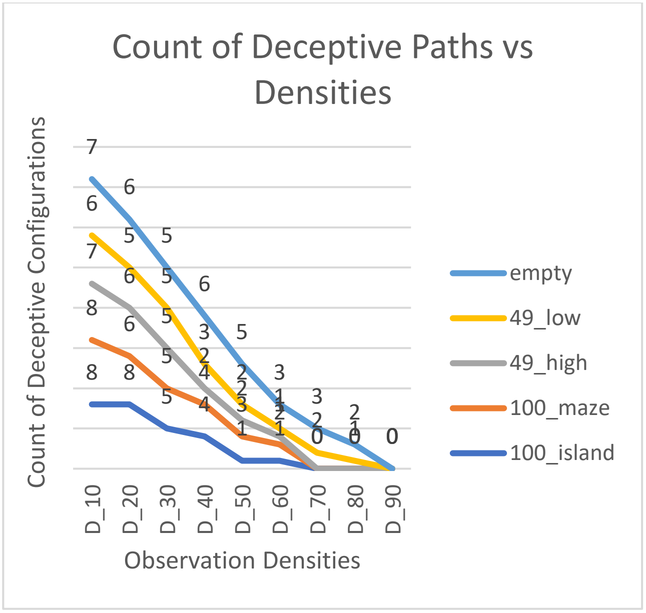


Figure 12: Count of Deceptive paths for each map out of eight configurations for various observation densities

The results show that the policy $\pi_{deceptive}$ is indeed deceptive. At an observation density of 10%, 8 out of 8 configurations (each configuration means different start points, real and fake goal positions and the number of fake goals) are deceptive for 100_island and 100_maze map, 7, 7 and 6 out of 8 configurations are deceptive for empty, 49_high and 49_low maps are deceptive respectively. As the observation densities increases, the possibility of deception decreases and reduced to a count of 0 at 90% density. Clearly, the agent moves deterministically towards the real goal, later in the trajectory which is captured by higher observation densities. However, this is in agreement with the hypothesis that states that the real intention of agent will ultimately be revealed. Overall, 38.9% (140 out of 360) paths generated by $\pi_{deceptive}$ are deceptive.

The ambiguity model proposed in [1] is used as the baseline for comparing our results. The comparison is presented in Figure 13 to Figure 17. We conducted 360 trials of baseline exactly similar to $\pi_{deceptive}$ trials. The comparison provides very interesting results. Almost 39% (140 out of 360) paths generated by $\pi_{deceptive}$ are deceptive, whereas baseline generates almost 45% (161 out of 360) deceptive paths. However, at lower observation densities of up to 20%, our policy returned 83.75% (67 out of 80) deceptive paths whereas baseline returns 80% (64 out of 80) deceptive paths. As the observation densities increase, baseline performs better. From observation densities of 30% to 90%, baseline generates 97 out of 280 (34.6%) deceptive paths, whereas our policy generates 73 out of 280 (26.1%) deceptive paths.

The difference in pruning strategies used in our model and baseline explains these results. While our model uses a simple pruning technique, that makes agent start moving towards the real goal (and hence become deterministic) when the selection of least divergent action becomes *confusing*, the baseline uses a better goal elimination and reconsideration strategy. However, because our model is more deceptive at low observation densities, we believe that implementing a similar pruning strategy as that of the baseline while selecting the least divergent action will improve our model, possibly beyond the baseline. Other than that, experimenting with lower values of β (our experiments used **β** = 1 which directly makes the agent deterministic after confusion) should improve deception as the model becomes more stochastic. These improvements form important future work.

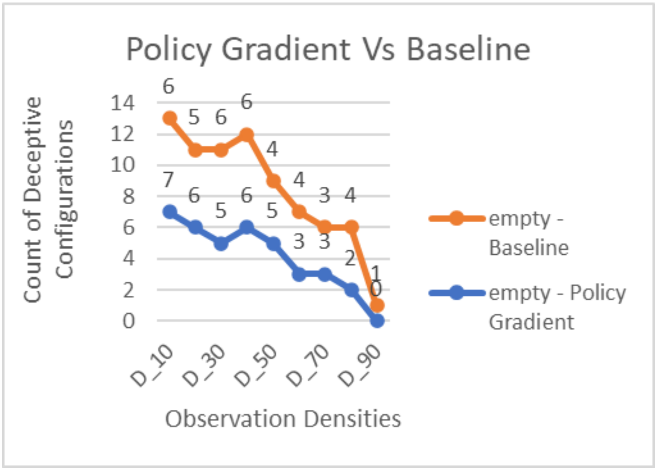


Figure 13: Comparison - empty map

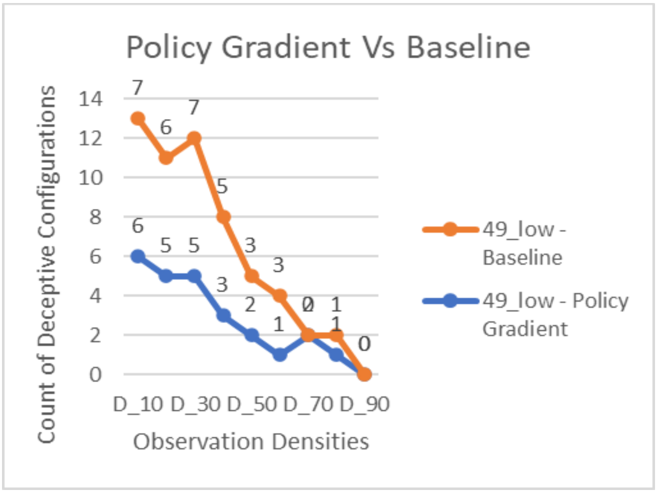


Figure 14: Comparison - 49_low map

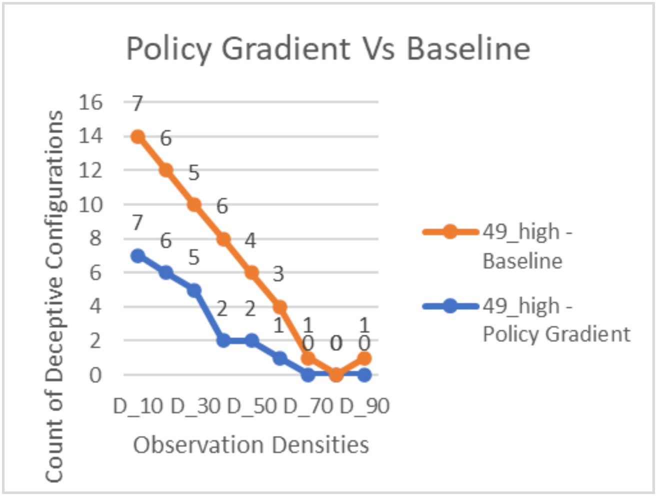


Figure 15: Comparison - 49_high

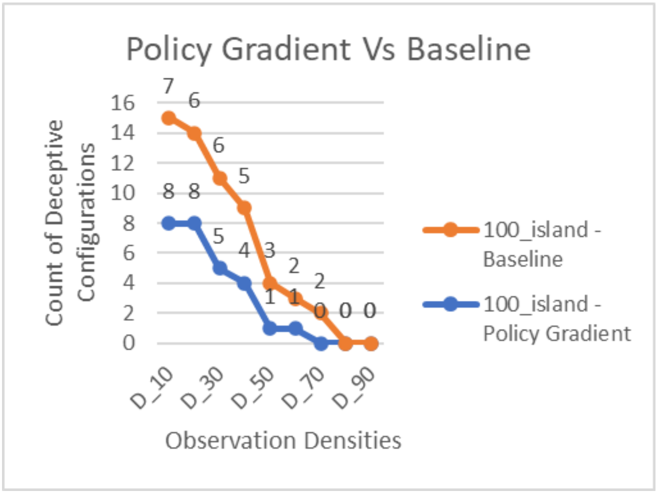


Figure 16: Comparison – 100_island map

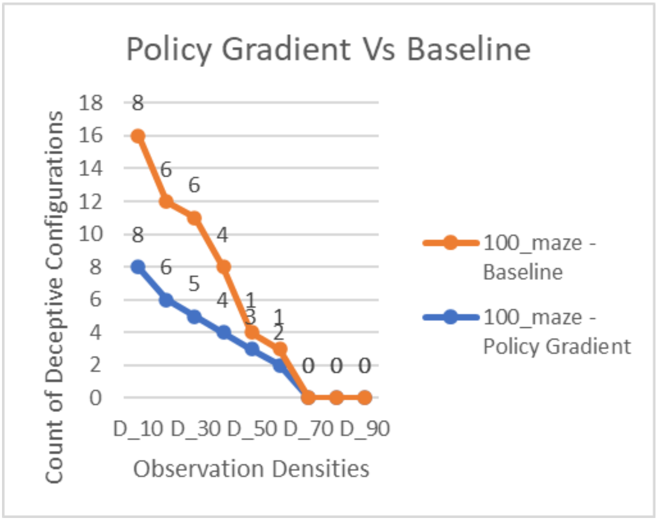


Figure 17: Comparison – 100_maze map

4 Conclusion

In this project, we researched policy gradient based deceptive reinforcement learning in continuous action space based on actor-critic neural network theory, in which the policy gradient generates stochastic policy. To accelerate the training process for the sparse reward, "HER" techniques are adopted as well. Then we use the irrationality model to generate the deceptive action by using parameter α to balance the selection of irrational and rational action. Finally, experiments are done to test the performance of our model. The result shows that irrationality model successfully shows deception in continuous action space and deception can be controlled using α. The work shows that hypothesis proposed are correct and the policy gradient method, together with the irrationality model work well to achieve deception in continues action space.

In the deceptive path planning part, which involves policy-gradient based stochastic policies, we have proposed a negative irrational model that maximises the probability of actions that leads the agent closer to all goals to increase deception. The stochastic policy, at the first stage of action selection, frequently selects this action. It also selects other actions rarely with a much lower probability at the first stage of selection to make the actions random and this adds deception. The second stage action selection ensures convergence. We hypothesised that this policy adds deception in path planning. We further hypothesised that towards the end of the trajectory the agent's real goal will ultimately be revealed. The experiments validated the hypothesis. Overall, 38.9% of the paths generated by our model are deceptive. Our model is especially effective at lower observation densities and returns 83.75% deceptive paths for observation densities up to 20%. Our model can be improved for higher observation densities by incorporating a better pruning strategy that does not consider completely irrational goals while selecting deceptive actions. Moreover, further experimenting with lower values of second stage action selection parameter β may also improve deception.

5 References

[1] Y. Yang, Z. Liu, P. Masters and T. Miller, "Deceptive Reinforcement Learning for Preserving the Privacy of Reward Functions," 2019.

[2] "FetchReach-v1," OpenAi, [Online]. Available: https://gym.openai.com/envs/FetchReach-v1/.

[3] S. Sardina, "Python Path Planning Project," [Online]. Available: https://bitbucket.org/ssardina-research/p4-simulator/src/master/. [Accessed 13 June 2020].

[4] B. Whaley, "Toward a general theory of deception," *The Journal of Strategic Studies,* vol. 5, no. 1, pp. 178-192, 1982.

[5] J. B. Bell, "Towards a Theory of Deception," *International Journal of Intelligence and Counter Intelligence,* vol. 16, no. 2, pp. 244-279, 2003.

[6] S. S. Peta Masters, "Deceptive Path-Planning," *International Joint Conference on Artificial Intelligence (IJCAI-17),* pp. 4368-4375, 2017.

[7] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2018.

[8] F. W. A. R. J. S. R. F. P. W. B. M. J. T. P. A. W. Z. Marcin Andrychowicz, "Hindsight Experience Replay," 2018.

[9] Ramirez and Geffner, "Probabilistic plan recognition using off-the-shelf classical planners," *In Proceedings of AAAI,* 2010.

[10] S. S. Peta Masters, "Cost-Based Goal Recognition for Path-Planning," *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems,* p. Pages 750–758, 2017.