Review

Deep Reinforcement Learning for Soft Robotic Applications: Brief Overview with Impending Challenges

Sarthak Bhagat [1,2,†],**Hritwick Banerjee** [1,3†] and **Hongliang Ren** [1,3,4,*]

[1] Department of Biomedical Engineering, Faculty of Engineering, 4 Engineering Drive 3, National University of Singapore, Singapore 117583, Singapore; biehb@nus.edu.sg (H.B.)
[2] Department of Electronics and Communications Engineering, Indraprastha Institute of Information Technology, Delhi, New Delhi, Delhi 110020, India; sarthak16189@iiitd.ac.in (S.B.)
[3] Singapore Institute for Neurotechnology (SINAPSE), Centre for Life Sciences, National University of Singapore, Singapore 117456.
[4] NUS (Suzhou) Research Institute (NUSRI), Wuzhong Dist., Suzhou City, Jiangsu Province, China.
† These authors equally contributed towards this manuscript.
* Correspondence: ren@nus.edu.sg (H.R.); Tel.: +65-6601-2802

Abstract: The increasing trend of studying the innate softness of robotic structures and amalgamating it with the benefits of the extensive developments in the field of embodied intelligence has led to sprouting of a relatively new yet extremely rewarding sphere of technology. The fusion of current deep reinforcement algorithms with physical advantages of a soft bio-inspired structure certainly directs us to a fruitful prospect of designing completely self-sufficient agents that are capable of learning from observations collected from their environment to achieve a task they have been assigned. For soft robotics structure possessing countless degrees of freedom, it is often not easy (something not even possible) to formulate mathematical constraints necessary for training a deep reinforcement learning (DRL) agent for the task in hand, hence, we resolve to imitation learning techniques due to ease of manually performing such tasks like manipulation that could be comfortably mimicked by our agent. Deploying current imitation learning algorithms on soft robotic systems have been observed to provide satisfactory results but there are still challenges in doing so. This review article thus posits an overview of various such algorithms along with instances of them being applied to real world scenarios and yielding state-of-the-art results followed by brief descriptions on various pristine branches of DRL research that may be centers of future research in this field of interest.

Keywords: deep reinforcement learning; imitation learning; soft robotics

1. Introduction

1.1. Soft robotics: a new surge in robotics

The past decade has seen engineering and biology coming together [1–5] leading to cropping up of a relatively newer field of research- Soft Robotics (SoRo). SoRo has come to our aid by enhancing physical potentialities of robotic structures amplifying the flexibility, rigidity and the strength and hence, accelerating their performance. SoRo is capable of creating three dimensional bio-inspired structures[6] that are capable of self-regulated homeostasis, resulting in robotics actuators that have the potential to mimic biomimetic motions with simple, inexpensive actuation [7,8] and able to achieve bending, twisting, extension, flexion with non-rigid materials [5,9]. These developments in creating such robotic hardware presents before us an opportunity to couple them with imitation learning techniques by exploiting the special properties of these materials to clich much higher levels of precision and accuracy.

1.2. Deep learning: an overview

There has been a certain incline towards utilization of deep learning techniques for creating autonomous systems that are capable to replace humans in varied domains. Deep learning [10] approaches have shown tremendous amount of success when combined with reinforcement learning (RL) tasks in the past decade and are known to produce state-of-art results in various diverse fields [11]. There have been several pioneer algorithms in this domain that have shown ground-breaking results in tasks difficult to handle with former methods. The need for creating completely autonomous intelligent robotic systems has led to the heavy dependence on the use of Deep RL to solve a set of complex real world problems without any prior information about the environment. These continuously evolving systems aid the agent to learn through a sequence of multiple time steps, gradually moving towards an optimal solution.

1.3. Robotics: perception & control

Essentially, all robotics tasks can be broken down into 2 different fragments, namely perception [12,13] and control. The task of perception can be viewed as a relatively simpler problem wherein agents are provided with necessary information about the environment *via* sensory inputs, from which they may extract desired target quantities or properties. But in the case of learning a control policy, the agent actively interacts with environment trying to achieve an optimal behaviour based on the rewards received.

The problem of control goes one step further than the former due to the following factors:

- Data distribution : In case of Deep RL for perception, the observations are independent and identically distributed while in case of controls, they are accumulated in an online manner due to their continuous nature where each one is correlated to the previous ones[14].
- Supervision Signal : Complete supervision is often provided in case of perception in form of ground truth labels while in controls there are only sparse rewards available.
- Data Collection : Dataset collection can be done offline in perception but requires online collection in case of controls. Hence, this greatly affects the data we can collect due the fact that the agent needs to execute actions in the real world which is not a primitive task in most scenarios.

1.4. Deep learning in SoRo

The implementation of Deep Learning Techniques for solving compound problems in the task of controls in soft robotics has been one of the hottest topics of research. Hence, there has been development of various algorithms that have certainly surpassed the accuracy and precision of earlier approaches. There are innumerable control problems that exist but by far the most popular and in-demands control task involves manipulation. The last decade has seen a huge dependence on soft robotics (and/or bio-robotics) for solving the control related tasks, and applying such DRL techniques on these soft robotics systems has become a focus point of heavy ongoing research. Hence, the amalgamation of these budding fields of interests presents before us a challenge as well as potential of building much smarter control systems that can handle objects of varying shapes [15], adapt to continuously diverging environments and perform substantial tasks of manipulation combining the hardware capabilities of a soft robotic system alongside the learning procedures of a deep learning agent. Hence, in this paper we focus applying DRL and imitation learning techniques to soft robots to perform the task of control of robotic systems.

1.5. Forthcoming challenges

The next big step towards learning policy controls for robotic applications is imitation learning. In such approaches, the agents learns to perform a task by mimicking the actions of an expert, gradually improving its performance with time just as a Deep Reinforcement Learning agent. Even though, it is really hard to design the reward/loss function in these cases due to the huge dimension of the action space pertaining to the wide variety of motions possible in soft robots, these approaches are extremely useful for humanoid robots or manipulators with high degrees of freedom where it is easy to demonstrate desired behaviours as a result of the magnified flexibility and tensile strength. Manipulation tasks especially the ones that involve the use of soft robotics are effectively integrated with such imitation learning algorithms giving rise to agents that are able to imitate expert's actions much more accurately than normal robotic agent that do not make use of bio-inspired structures[3,16, 17].

1.6. Overview of the present study

This review article comprises of various sections broadly divided into two sub-categories - deep reinforcement learning4 and imitation based learning10. The former gives a basic overview about the algorithms in RL3 and Deep RL4 followed by descriptive explanation about the application of Deep RL in Navigation6 and Manipulation7 mainly on SoRo environments. The succeeding section10 talks about behavioural cloning followed by inverse RL and generative adversarial imitation learning alongside some famous examples where they have been applied to solve real world problems. The paper also incorporates separate sections on problems faced while transferring learnt policies from simulation to real world and possible solutions to avoid observing such a *reality gap*8 which gives way to a section9 that talks about various simulation softwares available including some especially designed for soft robots. We also include a section11 at the end on challenges of such technologies and budding areas of global interest that may be future centers of DRL research for soft robotic systems.

2. Brief Overview of Reinforcement Learning

Each robotics task that we wish to perform can be seen as a Markov Decision Process (MDP), consisting of a 5-tuple such as :(i) S : set of all states; (ii) A : set of all actions; (iii) P : transition dynamics; (iv) R : set of all rewards; and γ : discount factor. In most situations, we consider Episodic MDPs, where there exists a terminal state which once obtained ends the learning procedure. Episodic MDPs with time horizon T, ends after T time steps regardless of the fact that it has reached its goal or not. In the problem of controls for robots, the information about the environment is gathered through the sensors which might not be enough to collect all information that it requires to make a decision about the action it must perform in the future time steps, such MDPs are called Partially Observable MDPs. These are countered by either stacking all observations upto that time step before processing them or by using a recurrent neural network. In any RL task, we intend to maximize the expected discount return that is the weighted sum of rewards received by the agent [18]. For this purpose, we have two type of policies namely stochastic (π(a | s)) where actions are drawn from a probability distribution and deterministic (μ(s)) where they are selected specifically for every state. Then, we have Value functions (V$_\pi$(s)) that depict the expected outcome starting from state s and following policy π.

3. Reinforcement Learning Algorithms

This section provides an overview of major RL algorithms that have been extended by using deep learning frameworks.

- Value-based Methods: These methods estimates the probability of being in a given state, using which the control policy is determined. The sequential state estimation is done by making use of Bellman's Equations (Bellman's Expectation Equation and Bellman's Optimality Equation). Most popular Value-based RL algorithms include State-Action-Reward-State-Action (SARSA) and Q-Learning, which differ in their td-targets that is the target value to which Q-values are recursively updated by a step size at each time step. SARSA is an on-policy method where the value estimations are updated towards a policy while Q-Learning being an off-policy method updates the value estimations towards a target optimal policy. Detailed explanation can be found in recent works like Dayan[19], Kulkarni *et al.*[20], Barreto *et al.*[21], and Zhang *et al.*[22].

- Policy-based Methods: In contrast to the Value-based methods, Policy-based methods directly update the policy without looking at the value estimations. They are quite a few ways better than value-based methods in the terms of convergence, solving problems with continuous high dimensional data, and effectiveness in solving deterministic policies. They perform in two broad ways - gradient-based and gradient-free[23,24] methods of parameter estimation. We focus on gradient-based methods where gradient descent seems to be the popular choice of optimization algorithm. Here, we try to optimize the objective function as:

$$J(\pi_\theta) = E_{\pi_\theta}[f_{\pi_\theta}(.)] \tag{1}$$

and the score function[25], given by $f_{\pi_\theta}(.)$. In previous equation, decides on how good the current policy performs on the task in hand. A popular RL algorithm is the REINFORCE algorithm[26], that simply plugs in the sample return equal to the score function given by:

$$f_{\pi_\theta}(.) = G_t. \tag{2}$$

A baseline term b(s) is often subtracted from the sample return to reduce the variance of estimation which updated the equation in the following manner:

$$f_{\pi_\theta}(.) = G_t - b_t(s_t). \tag{3}$$

While using Q-value function as our score function we can make use of either stochastic policy gradient4 or deterministic policy gradient5[27] given by:

$$\nabla_\theta(\pi_\theta) = E_{s,a}[\nabla_\theta \log \pi_\theta(a|s).Q^\pi(s,a)] \tag{4}$$

and

$$\nabla_\theta(\mu_\theta) = E_s[\nabla_\theta \mu_\theta(s).Q^\mu(s,\mu_\theta(s))]. \tag{5}$$

- Actor Critic Method: These are algorithms that keep a clear representation of the policy and state estimations. The score function for this is obtained by replacing the return G_t from equation 3 of policy based methods with $Q^{\pi_\theta}(s_t,a_t)$ and baseline b(s) with $V^{\pi_\theta}(s_t)$ that results in the following equation:

$$f_{\pi_\theta}(.) = Q_{\pi_\theta}(s_t,a_t) - V_{\pi_\theta}(s_t). \tag{6}$$

The advantage function A(s,a) is given by:

$$A(s,a) = Q(s,a) - V(s). \tag{7}$$

- Integrating Planning and Learning: All algorithms discussed uptil now learn a control policy by maximizing rewards obtained from actual experiences. There also exists methods wherein the agent learns from experiences itself but also can collect imaginary roll-outs[28]. Such methods have also been upgraded by using them alongside DRL methods[29,30].

4. Deep Reinforcement Learning Algorithms

Neural networks have been real asset in approximating various optimal value functions in Reinforcement Learning Algorithms and hence, have been extensively applied to predict the most favourable control policy in robotics. The following are the common DRL algorithms that have been put in practise and have shown fine potential in solving such control problems:

- Deep Q-Network (DQN) [31]: In this approach the optimal value of Q-function is obtained using a deep neural network (generally a CNN), just like we do in all other value-based algorithms.

Table 1. Key differences between Value Based and Policy Based (along with Actor Critic Methods) on various different factors of variation.

Algorithm	Value Based Methods	
Examples	Q-Learning, SARSA, Value Iteration	Rl
Steps Involved	Finding optimal value function and find the policy based on that (policy extraction)	
Iteration	The two processes (listed in above cell) are not repeated after once completed	The above two p
Convergence	Relatively Slower	
Type of Problem Solved	Relatively Harder control problems	
Method of Learning	Explicit Exploration	
Advantages	Simpler to train off-policy	
Process Basis	Based on Optimality Bellman Operator- is non-linear operator	

We denote the weights of this Q-network by Q*(s,a) and the td-error and td-target are given by equations:

$$\delta_{\mathrm{t}}{}^{\mathrm{DQN}} = y_{\mathrm{t}}{}^{\mathrm{DQN}} - Q(s_{\mathrm{t}}, a_{\mathrm{t}}; \theta_{\mathrm{t}}{}^{\mathrm{Q}}) \tag{8}$$

and

$$y_{\mathrm{t}}{}^{\mathrm{DQN}} = R_{\mathrm{t+1}} + \gamma. \tag{9}$$

The weights are recursively updated by the equation:

$$\theta_{\mathrm{t+1}} \leftarrow \theta_{\mathrm{t}} - \alpha(\partial(\delta_{\mathrm{t}}{}^{\mathrm{DQN}}(\theta_{\mathrm{t}}{}^{\mathrm{Q}}))^2/\partial\theta_{\mathrm{t}}{}^{\mathrm{Q}}). \tag{10}$$

moving in the direction of decreasing gradient with a rate equal to the learning rate. These are capable of solving problems involving high dimensional state spaces but restricted to discrete low dimension when it comes to action space. These methods have been extremely useful when applied to various problems of manipulation using soft robotics that have structure inspired from biological creatures, combining which gives systems are able to interact with the environment alongside additional flexibility and adaptation to changing situations.

Two main methods employed in DQN for learning are:

- Target Network: Target network Q- has same architecture as the Q-network but while learning the weights of only the Q-network are updated, while repeatedly being copied to weights of θ^- network. In this procedure, td-target is computed from the output of θ^- function[32].
- Experience Replay: The collected data in form of state-action pairs with their rewards are not directly utilized but are stored in a replay memory. While actual training, samples are picked up from the memory to serve as mini-batches for the learning. The further learning task follows the usual steps of using gradient descent to reduce loss between learned Q-network and target Q-network.

These two methods are used to stabilize the learning in DQN by reducing the correlation between estimated and target Q-values, and between consecutive observations respectively. Advanced techniques for stabilizing and creating more efficient models include Double DQN[33] and Dueling DQN[34].

- Deep Deterministic Policy Gradients (DDPG) [32]: This is a modification of the DQN combining techniques from actor-critic methods allowing us to model problems with continuous high dimensional action spaces. The training procedure of a DDPG agent is depicted in figure 1. The equations for stochastic11 and deterministic12 policies are given by equations:

$$Q^{\pi}(s_t, a_t) = E_{R_{t+1}, s_{t+1} \sim E}[R_{t+1} + \gamma E_{a_{t+1} \sim \pi}[Q^{\pi}(s_{t+1}, a_{t+1})]] \tag{11}$$

and

$$Q^{\mu}(s_t, a_t) = E_{R_{t+1}, s_{t+1} \sim E}[R_{t+1} + \gamma Q^{\mu}(s_{t+1}, \mu(s_{t+1}))]. \tag{12}$$

The difference between this and DQN lies in the dependence of Q-value on action where it is represented by giving one value from each action in DQN and by taking action as input to theta Q in case of DDPG. This method remains to be one of the premiere algorithms in the field of DRL that have been successfully applied to systems have soft robotic structure.

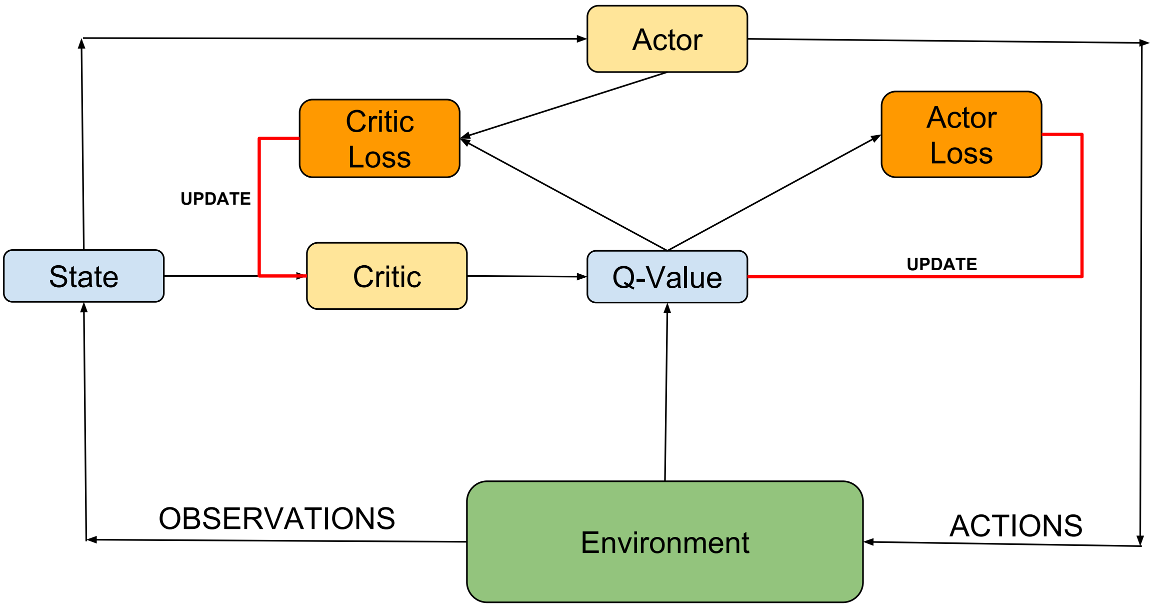

Figure 1. The figure depicts the training architecture of DDPG agent.

- Normalised Advantage Function (NAF) [35] : This functions in a similar way as DDPG in the sense that it also helps us to enable Q-learning in continuous high dimensional action spaces by employing the use of deep learning. In NAF, Q-function Q(s,a) is represented so as to ensure that its maximum value can easily be determined during the learning procedure. The difference in NAF and DQN lies in the network output, wherein it outputs θ^V, θ^μ and θ^L in its last linear layer of the neural network. θ^μ and θ^L help us predict the advantage necessary for the learning technique. Similar to a DQN, it also makes use of Target Network and Experience Replays to ensure there is least correlation in observations collected over time. The advantage term in NAF is given by:

$$A(s, a; \theta^{\mu}, \theta^{L}) = -(1/2)(a - \mu(s, \theta^{\mu}))^T P(s; \theta^L)(a - \mu(s; \theta^{\mu})) \tag{13}$$

wherein,

$$P(s; \theta^L) = L(s; \theta^L) L(s; \theta^L)^T. \tag{14}$$

This is a much simpler form of solving problems that can't be solved using common DQN techniques. Asynchronous NAF approach has also been introduced in the work by Gu *et al.*[36].
- Asynchronous Advantage Actor Critic (A3C) [37]: In several asynchronous DRL approaches, various actors-learners are utilized to collect as many observations as possible, each storing gradient for their respective observations that used to update the weights of the network. A3C is the most commonly used algorithm of this type. This method always maintains a policy representation π(a | s;θ^π) and a value estimation V(s;θ^V) making use of score function in form of an advantage function that is obtained by observations that are provided by all the action-learners. Each actor-learner collects roll-outs of observations of its local environment upto

T steps, accumulating gradients from samples in the roll-outs. The approximation of advantage function used in this approach is given by equation:

$$A(s_t, a_t; \theta^\pi, \theta^V) = [\sum_{k=t}^{T-1}[\gamma^{k-t}] + \gamma^{T-t}V(s_T; \theta^V) - V(s_t; \theta^V); \theta^\pi]. \tag{15}$$

The network parameters θ^V and θ^π are updated repeatedly according to the equations given by:

$$d\theta^\pi \leftarrow d\theta^\pi + \nabla_{\theta^\pi} \log \pi(a_t|s_t; \theta^\pi) A(s_t, a_t; \theta^\pi, \theta^V) \tag{16}$$

and

$$d\theta^t V \leftarrow d\theta^V + \partial A(s_t, a_t; \theta^\pi, \theta^V)^2 / \partial\theta^V. \tag{17}$$

This approach doesn't require learning stabilization techniques like memory replay as the parameters are updated simultaneously rather than sequentially hence, eliminating the correlation factor between them. Also, there are several action-learners involved in this method that tend to explore a much wider view of the environment and helping learning a more optimal policy. A3C has proven out to be the stepping stone for DRL research and a popular algorithm that has been extremely efficient in providing state-of-art results alongside greatly reduced time and space complexity and its range of problem solving capabilities.

- Advantage Actor Critic (A2C) [38,39]: In some scenarios, it is not necessary that asynchronous methods lead to better performance. It has been shown in some papers, that synchronous version of the previous algorithm also provides fine results wherein each actor-learner finishes collecting observation after which they are averaged and an update is made.
- Guided Policy Search (GPS) [40]: This approach involves collecting samples making use of current policy, generating a training trajectory at each iteration that are utilized to update the current policy according to supervised learning. The change is bounded by adding it like a regularization term in the cost function, to prevent sudden changes in policies leading to instabilities.
- Trust Region Policy Optimization (TRPO) [41]: In Schulman *et al.*[41], an algorithm was proposed for optimization of large nonlinear policies which gave some improvement in the accuracy. Discount cost function for an infinite horizon MDP is given by replacing reward function with cost function giving the equation:

$$\eta(\pi) = E_\pi[\sum_{t=0}^{\infty} \gamma^t c(s_t)|s_0 \sim \rho_0]. \tag{18}$$

Similarly, the same replacement made to state-value functions give us the following two equations: (38) and (39) Hence, resulting in advantage function given by:

$$A^\pi = Q^{pi}(s, a) - V^\pi(s). \tag{19}$$

Optimizing equation 19 would result in giving us an updation rule for the policy as follows:

$$\eta(\pi) = \eta(\pi_{old}) + E[\sum_{t=0}^{\infty} \gamma^t A^{\pi_{old}}(s_t, a_t)|s_0 \sim \rho_0]. \tag{20}$$

This has been mathematically proved via advanced literatures by Kakade and Langford [42] that this method greatly improves the performance by quite a bit, and hence its popularity. This algorithm requires advanced optimization problem solving techniques using conjugate gradient and then using line search[41].

- Proximal Policy Optimization (PPO) [43]: This methods tries to solve soft constraint optimization problem making use of standard Stochastic Gradient Descent problem, in which C is tuned

doi:10.20944/preprints201811.0510.v1

according the KL-divergence term. This is one of the most popular DRL algorithms due to its simplicity and effectiveness in solving control problems. It has been widely applied to various fields of policy estimation and is also the default algorithm in OpenAI.

- Actor Critic Kronecker-Factored Trust Region (ACKTR) [39]: This uses a form of trust region gradient descent algorithm for a actor-critic with curvature estimated using Kronecker-Factored approximation. It is effectively one of the most efficient DRL algorithms being computationally better than TRPO. It makes use of natural gradient descent being much more sample efficient than other methods using gradient descent.

These methods have shown great prospect when combined with the innumerable physical capabilities of the soft structure of bio-inspired robots[3,16], and are topic of great interest. Various works showcase that such neural network oriented approaches have been hugely successful when dealing with controls tasks especially manipulation with soft robots.

5. Deep Reinforcement Learning Mechanisms

Many mechanisms have been proposed that can greatly affect the learning procedure while solving control problems involving soft robots (especially for manipulation) by the aid of DRL algorithms. These act as catalysts to the task in hand acting orthogonally to the actual algorithm. The task of solving DRL problems to obtain nearly optimal action with respect to each state for soft robotics has achieved great success as well as attention of robotics researcher around the world, hence, increasing the demand to not only formulate computationally less expensive DRL algorithms but also introduce mechanisms that enhance the productivity and efficiency if these algorithms. Some of them that have been common applied to diverse set of control tasks include :

- Auxiliary Tasks[44–48]: Usage of several other supervised and unsupervised machine learning methods like regressing depth images from colour images, detecting loop closures etc. besides the main algorithm to receive some information from sparse supervision signals in the environment.
- Prioritized Action Replay[49]: Prioritizing memory replay according to td-error
- Hindsight Action Replay[11]: Relabeling the rewards for the collected observations by more effective use of failure trajectories along with using binary/sparse labels that speed the off-policy methods.
- Curriculum Learning[50–52]: Exposing the agent to more sophisticated setting of the environment helping it to learn to solve complex tasks.
- Curiosity-Driven Exploration[53]: Incorporating internal rewards besides external ones collected from observations.
- Asymmetric Action Replay for Exploration[54]: The interplay between two forms of the same learner generates a curricula, hence, driving exploration
- Noise in Parameter Space for Exploration[55,56]: Inserts additional noise so as to aid exploration more in the learning procedure.

6. Deep Reinforcement for Soft Robotics Navigation

Autonomous driving tasks with completely automated navigation in which the goal of the agent is to reach a given goal point avoiding static or moving obstacles in the path while following a trajectory that is planned minimizing the cost/effort exerted remains to be one of the pioneer tasks in robotic controls even with soft robotic systems. Deep RL approaches have turned out be an aid for such tasks helping to generate such trajectories by learning from observations taken from the environment in form of state-action pairs. Like all other DRL problems, the navigation problem too is considered as a MDP that inputs sensor (LIDAR scans and depth images from on-board camera) readings and in return outputting a trajectory (policy in form of actions to be taken a particular states), that will help the agent to complete the task of reaching the goal in the given span of time.

Several experiments have been carried out in this growing field of research and some of them are stated below:

- Zhu *et al.*[57] gave the A3C system the first-person view alongside the target image to conceive the goal of reaching the destination point, by the aid of universal function approximators. The network used for learning is a ResNet[58] that is trained using a simulator[59] that creates a realistic environment consisting of various rooms each as a different scene for scene-specific layers. The model gave 70% accuracy for predicting the policy for targets away by one step from the trained targets and 42% for the ones two steps away. After being provided with images of a real scene of an office the robot could navigate effectively and in a collision-free manner. An A3C agent was trained on 100 million frames with an optimal solution of 17.6 steps over an average trajectory length of 210.7[57].
- Zhang *et al.*[22] implemented a deep successor representation formulation for predicting Q-value functions that learn representations interchangeable between navigation tasks that may be related in some way. Successor feature representation[20,21] breaks down the learning into two fragments - learning task-specific reward functions and task specific features alongside their evolution for getting the task in hand done. This method takes motivation from other DRL algorithms that make use of optimal function approximators to relate and utilize the information gained from previous tasks for solving the tasks we currently intend to perform[22]. This method has not only been observed to work effectively in transferring current policies to different goal positions in the same environment and varied/scaled reward functions but also to newer complex situations like new unseen environments. The models (pre-trained or transferred) attained high accuracy (nearly best possible) in solving control problem given a 3D maze with given start and goal point and RGB images for observation.

  Both these methods intend to solve the problem of navigation for autonomous robots that have inputs in form of RGB images of the environment by either getting the target image[57] or by transferring information that is gained through previous processes[22,60]. In contrast, Tai *et al.*[60] proposes an approach in which it tries to create a trajectory for the mobile robot with the help of relative position of the robot with respect to the ultimate destination position that could be obtained with the help of Wi-fi or visible light localization. Such models are trained via asynchronous DDPG for varied set of real and simulations of real environments.
- Mirowski *et al.*[44] made use of a popular DRL mechanism of using a additional supervision signals available (especially loop closures and depth prediction losses) in the environment allowing the robot to freely move between varying start and end point with increased accuracy and efficiency.
- Chen *et al.*[61] proposed a solution for highly compound problems involving dynamic environment (essentially obstacles) like navigating on a path with pedestrians as probable obstacles. It utilizes a simple set of hardware to demonstrate the proposed algorithm in which LIDAR sensor readings are used to predict the different properties associated with pedestrians (like speed, position, radius) that contribute to forming the reward/loss function. Long *et al.*[62] also makes use of PPO to conduct multi-agent obstacle avoidance task.
- Simultaneous Localisation and Mapping (SLAM)[63] have been at the heart of recent robotic advancements with a loads of papers been written regularly annually in this field in the last decade. SLAM may make use of DRL methodologies partially or completely and have shown to produce one of the best results in such tasks of localisation and navigation.
- A popular imitation learning problem that will be dealt with later in more detail trains a Cognitive Mapping and Planning (CMP)[64] model with DAGGer that inputs 3-channeled images and with the aid of Value Iteration Networks (VIN) creates a map of the environment. The work of Gupta *et al.*[65] has also introduced a new form of amalgamation of spatial reasoning and path planning.
- Zhang *et al.*[51] also introduced a new form of SLAM called Neural SLAM that took inspiration from works of Parisotto and Salakhutdinov[66] that allowed our agent to interact with Neural Turing Machine(NTM). Graph-based SLAM[63][67] led way for Neural Graph Optimiser by Parisotto *et al.*[68] which inserted this global pose optimiser in the network.

Further advanced readings on Navigation using DRL techniques include Zhu *et al.*[57], Schaul *et al.*[69], He *et al.*[58], Kolve *et al.*[59], Zhu *et al.*[70], Long *et al.*[62], Gupta *et al.*[64], Khan *et al.*[71], Bruce *et al.*[72], Chaplot *et al.*[73] and Savinov *et al.*[74].

7. Deep Reinforcement Learning for Soft Robotics Manipulation

The extensive use of soft robots in all domains especially the industrial department replacing human labour and beating them in all possible fields in terms of accuracy and efficiency has lead to sprouting of this new field of interest for robotic researchers - The application of DRL techniques for Manipulation tasks like picking, dropping, reaching etc.[32,37,41,75]. The enhanced rigidity, flexibility, strength and adaptation capability of soft robots over hard robots[76,77] has lead to its extensive applications in the manipulation field and combining it with DRL has been observed to give highly precise and satisfactory results. The coming of the soft robotics technologies and its extremely effective blend with such deep learning technologies has often contributed to robots becoming a crucial part of almost all manipulation tasks.

After great developments in the domain of soft robotics, we require much better learning algorithms that can solve much complex manipulation tasks alongside taking care that we follow all constraints that are enforced as a result of the physical structure of such bio-inspired robots[3,16]. Coming of DRL technologies have hugely influenced and more importantly enhanced the performance of such agents.

In the past few years, we have witnessed a drastic increase in research focus towards DRL techniques while dealing with soft robots. Some of the milestone papers of DRL used for manipulation tasks are listed below:

- Gu *et al.*[36]: Gave a modified form of NAF that works in a asynchronous fashion in the task of door opening taking state inputs like joint angles, end effector position, and position of target. It gave a whopping 100% accuracy in this task and learning it in mere 2.5 hours of time.
- Levine *et al.*[47]: Proposed a visuomotor policy based model that is an extended deep version of GPS algorithm studied earlier. The architecture of the network consists of convolutional layers along with softmax activation function taking in as input images of the environment and concatenating the necessary information gained along with the robot's state information. The required torques are predicted by passing this concatenated input to linear layers at the end of the network. These experiments were carried out with a PR2 robot for various tasks like screwing a bottle, inserting block into a shape sorting cube etc. Despite its highly desirable results, it is not widely used in real world applications are it requires complete observability in state space that is sometime difficult to obtain in real life.
- Finn *et al.*[78] and Tzeng *et al.*[79]: Made use of DRL techniques for predicting optimal control policies by studying state representations.
- Fu *et al.*[80]: Introduced the use of a dynamic network for creating a one-shot model for manipulation problems. There have been advancements in the area of manipulation using multi-fingered bio-inspired hands that may be model-based[81,82] or model-free[83].
- Riedmiller *et al.*[45]: Gave a new algorithm that enhanced the learning procedure from the time complexity as well as accuracy point of view. It said that sparse rewards for the model help in attaining more optimal policy faster than providing binary rewards that may lead to policy that not the desirable trajectories for the end effector. For this, another policy (referred to as intentions) was learned for auxiliary tasks whose inputs are easily attainable via basic sensors [84]. Alongside this, a scheduling policy is also learned for scheduling the intention policies. Such a system have much better results than a normal DRL algorithm for the task of lifting that took about 10 hours to learn from scratch.

Refer to figure 2 for images of some these ground-breaking works in field of DRL utilized for robotic manipulation tasks.

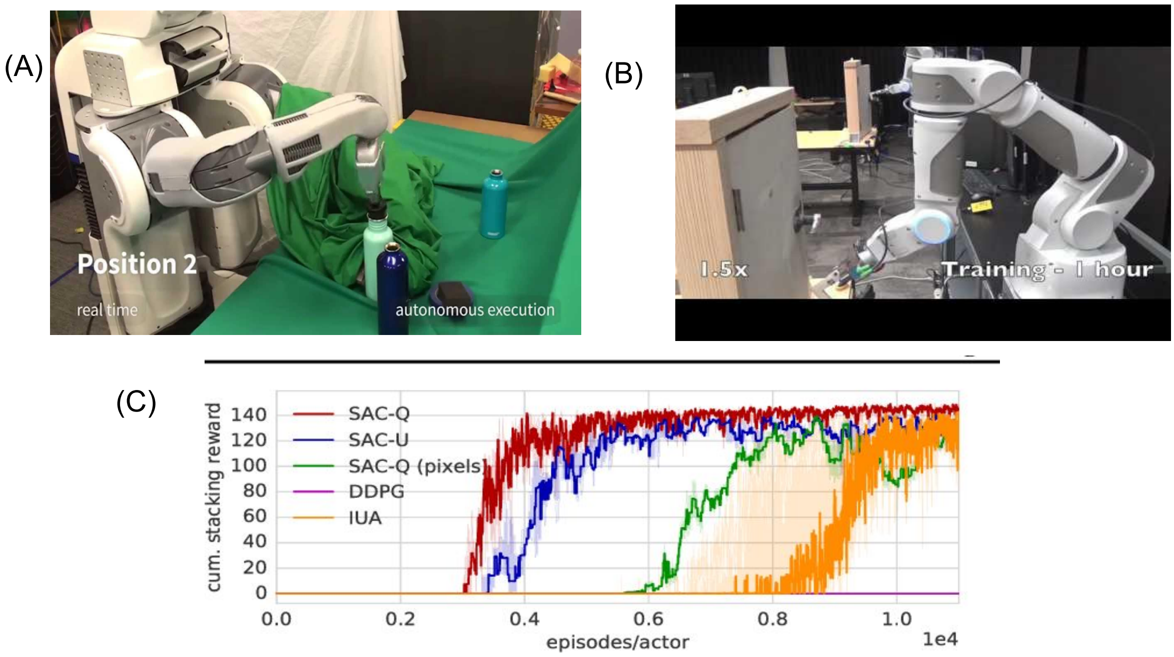

Figure 2. (A) Manipulation robot trained using techniques proposed by Gu *et al.*[36] showcasing autonomous execution of a task of opening caps of coloured bottles. (B) Manipulation robot trained using techniques proposed in Levine *et al.*[47] for a total training time of 1 hour. (C) Graphical Plots for Various Methods trained for the same manipulation task depicting the superiority in performance of the model in terms of reward achieved when provided with sparse rewards proposed in Riedmiller *et al.*[45].

Soft Robots have taken over major departments of the industrial sector due such heavy developments in the fields of manipulation controls via DRL methods for their unquestionable superiority over other alternative methods. The accuracy, precise and efficiency displayed by these autonomous bio-inspired industrial systems are huge in comparison to the human or hard robotic counterparts[85]. Involving more of such soft robots in place of humans has also lead to a drastic dip in chances of industrial disasters due to human error (as a result of their environment adaptation property) and they have proven to be extremely useful for working environments that are unsuitable for our bodies.

8. Difference between Simulation and Real World

Collecting training samples is not an easy task while solving the problem of controls in soft robotics as it is in perception for similar systems. Collection of real world dataset (state-action pair) is a costly operation due to the high dimensionality of the control spaces and the lack of availability of a central source of control data for every environment. This causes what we call as *reality gap* which refers to the difference in various factors in simulations and real world. This inflicts upon us various challenges while bringing models that have been trained in simulation to real world scenarios. Even though, we have various simulation software for soft robotics especially designed for manipulation tasks like picking, dropping, placing etc., there are still challenges that act as hindrances in this problem of solving control tasks making use of flexible

Soft robots that have bio-inspired designs and materials making use of DRL techniques listed in the previous sections. Such a gap is most prevalently viewed in disparities in visual data[22] or laser readings[60] and some papers have aimed at reducing the discrepancies by proposing certain approaches. Following section provides an overview of some of them:

- Domain Adaptation: This basically just translates images from source domain to the destination domain. Domain confusion loss that was first proposed in the paper Tzeng *et al.*[86] learns a representation that is undeviated and steady towards changes in domain. But the limitation to this approach lies in the fact that it requires the source and destination domain information before the training step, which is not possible for most models. Visual data coming from multiple sources is represented by X (simulator) and Y (on-board sensors). The problem arises when we train the model on X and test it on Y, wherein we observe a considerable amount of performance difference between the two.
- This problem can be solve if we have some kind of a mapping function that can map data from one domain to the other one. This can be done by employing a deep generative model called Generative Adversarial Network or commonly known as GANs[87–89]. GANs are deep models that have two basic components - a discriminator and a generator. The job of the generator or popular known as the decoder is produce images samples from the source domain to the destination domain, while that of the discriminator is to differentiate between true and false (generated) samples.
  - CycleGAN[70]: First proposed in Zhu *et al.*[70], works on the principle that it is possible and feasible to predict a mapping that maps from input domain to output domain simply by adding a cycle consistent loss term as a regulariser for the original loss for making sure the mapping is reversible. It is a combination of two normal GANs and hence, two separate encoders, decoders and discriminators are trained according to equations:

$$L_{GAN_Y}(G_Y, D_Y; X, Y) = E_y[\log D_Y(y)] + E_x[\log(1 - D_Y(D_Y(x)))] \tag{21}$$

    and

$$L_{GAN_X}(G_X, D_X; Y, X) = E_x[\log D_X(x)] + E_y[\log(1 - D_X(D_X(y)))]. \tag{22}$$

    The loss term for the complete weight updation (after incorporating the cycle consistent loss terms for each GAN) step now turns out to be:

$$\begin{aligned} L(G_Y, G_X, D_Y, D_X; X, Y) = L_{GAN_Y}(G_Y, D_Y; X, Y) + L_{GAN_X}(G_X, D_X; Y, X) \\ +\lambda_{cyc} L_{cyc_Y}(G_X, G_Y; Y) + \lambda_{cyc} L_{cyc_X}(G_Y, G_X; X). \end{aligned} \tag{23}$$

    Hence, the final optimization problem turns out to be equation 24.

$$G_Y^*, G_X^* = arg \min_{G_Y, G_X} \max_{D_Y, D_X} L(G_Y, G_X, D_Y, D_X) \tag{24}$$

    This is known to produce desirable results for scenes where it is relatively simpler to draw comparisons/relations between both domains but sometimes fails on complex domains environments.
  - CyCADA[90]: The problems that CycleGAN, that was first introduced in Hoffmann *et al.*[90], faced were resolved by making use of the semantic consistency loss that could be used to map complex environments. It trains a model to move from the source domain containing semantic labels, helping us to map the domain images from X to that in Y. The equations that is used for mapping using the decoder are given by:

$$L_{sem_Y}(G_Y; X, f_X) = CE(f_X(X), f_X(G_Y(X))) \tag{25}$$

    and

$$L_{sem_X}(G_Y; Y, f_X) = CE(f_X(Y), f_X(G_X(Y))). \tag{26}$$

Here, CE(S_X,f_X(X)) represents the cross-entropy loss between data-points predicted by pre-trained model and the true labels S_X.

The training architecture of a CycleGAN and a CyCADA is described in figure 3.

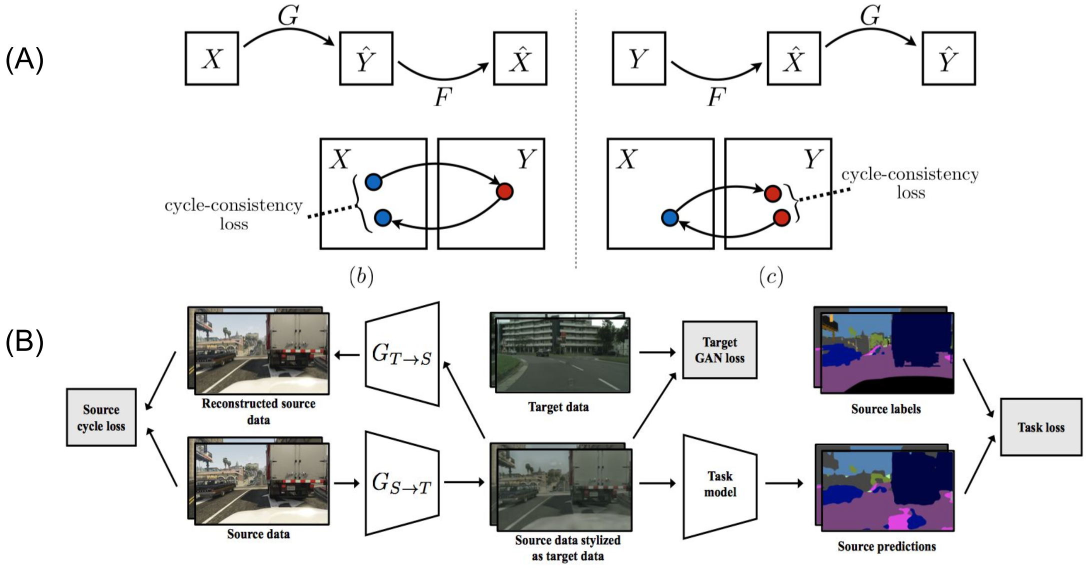

Figure 3. (A) Training Architecture of CycleGAN. (B) Training Architecture of a CyCADA.

Not just these two applications (Domain Adaptation and GANs) but there are many more algorithms/techniques in which the usage of modern day deep learning frameworks like GANs[87–89], VAEs[91], disentangled representations[92,93] and many that have greatly aided the process of controls of soft robots. These developing frameworks have certainly widened the perspective of DRL for robotic controls during the current era of technological advancements, have proven that there is a huge scope for research in these fields and that they have ever growing applications in robotics. The combination of two such extremely tender fields of technology - soft robotics and modern deep learning frameworks (especially generative models) may be something that might act as stepping stones to major technological advancement in the next decade, and an essential part of all domains making use of robotics for controls (more generally manipulation).

- Domain Adaptation for Visual DRL Policies: In such adaptation techniques, we try to transform the policy from source domain to destination domain.

  Bousmalis *et al.*[94] proposed a new way to solve problems of reality gap in policies trained on simulations and applying them in real life scenarios.

There have been recent developments with the aim of developing newer training techniques to avoid such a gap in efficiency while testing on simulation and real world scenarios besides advancements in the simulation environments possible to create virtually. Tobin *et al.*[95] randomised the lightning conditions, viewing angles and texture of objects to ensure the agent is manifested to all disparities in the factors of variation. The works by Peng *et al.*[96] and Rusu *et al.*[97] also focus on more such training methods. The recent advancements in VR Goggles[98] have separated policy learning and its adaptation in order to minimize the transfer steps required for moving from simulation to real world. A new optimization objective comprising of an additional shift loss regularisation term was also deployed on such a model that borrows motivation from artistic style transfer proposed by

Ruder *et al.* [99]. Works in the domain of scene adaptation include indoor scene adaptation where the semantic loss is not added (A VR Goggles[98] model tested on Gazebo [100] simulator using a Turtlebot3 Waffle) and outdoor scene adaptation where we do add such a semantic loss term to the original loss function. Outdoor scene adaptation involves collection of real world data through a dataset like RobotCar [101] which is tested on a simulator(like CARLA [102]). The network is trained using DeepLab Model [103] after adding the semantic loss term. Such a model may turn out to be extremely useful for situations where the simulation fails to accurately represent the real agent [104].

The problem of making effective software that not just works on the simulation but is also is effective in real world still remains to be the toughest challenges in the path of roboticists and researchers that aim to make completely autonomous systems using designs that have biologically similar structures and are composed of edible degradable materials. With sharp rise in number of simulations softwares (and simulation methods [105]) available publically and the growth in the hardware industry for soft robots - upcoming of 3D printed bio-robots[106–109] that are much effective than normal ones for specific tasks for which they have been designed for alongside special flexible electronics for such systems[110], there is still some scope for improvement in development of real world soft robots with practical applications, making way for a hot topic for future research in the upcoming years.

9. Simulation Platforms

There are several platforms that are available for simulation purposes of DRL agents before testing them on real world applications. Some of them are given below in table **??**.

Table 2. The following tables lists down various simulation softwares available for simulating real-looking environments for training of DRL agents alongside the modularities available with them and their special domain of use.

Simulator	Modalities	Special Purpose of Use
Gazebo (Koenig *et al.*[100])	Sensor Plugins	General Purpose
Vrep (Rohmer *et al.*[111])	Sensor Plugins	General Purpose
Airsim (Shah *et al.*[112])	Depth/Color/Semantics	Autonomous Driving
Carla (Dosovitskiy *et al.*[102])	Depth/Color/Semantics	Autonomous Driving
Torcs (Pan *et al.*[113])	Color/Semantics	Autonomous Drivings
AI-2 (Kolve *et al.*[59])	Color	Indoor Navigation
Minos (Savva *et al.*[114])	Depth/Color/Semantics	Indoor Navigation
House3D (Wu *et al.* [115])	Depth/Color/Semantics	Indoor Navigation

The readily available softwares have lead to ease in robotics software development, and hence contributing heavily to the upcoming research in the field of controls. The fact that soft robotics hardware is relatively expensive and not easy to use and that a normal DRL agent requires lots of training even before testing it in the real environments, makes the presence of special purpose simulation tools more and more important. Hence, these upcoming simulation softwares have come to the aid of robotics researchers who wish to contribute in this field of robotic research.

Another problem that arises when we utilize soft robots for solving control tasks in place of the hard ones is that are there are much fewer simulation softwares available for soft robotics applications[116] as compared to the hard ones. The fact that soft robotics is a relatively new field results is the reason that there are scarcely any simulation softwares for manipulation tasks using a soft robot. The number of such softwares are expected to rise at a much brisk rate due to the heavy demand of soft robotics and the fact that DRL techniques are extremely expensive to train in real world environments. One of the most famous soft robotics simulator is SOFA that allows the user to model, simulate and control a soft robot. Soft-robotics toolkit[117] is a plugin that aids us to simulate soft robots using SOFA framework[118]. Others that are also capable of modeling and simulating soft robotics agents are V-REP simulator[119], Action simulation by Energid, and MATLAB (Simscape modeling)[120]. Some of these simulation softwares are shown in figure 4.

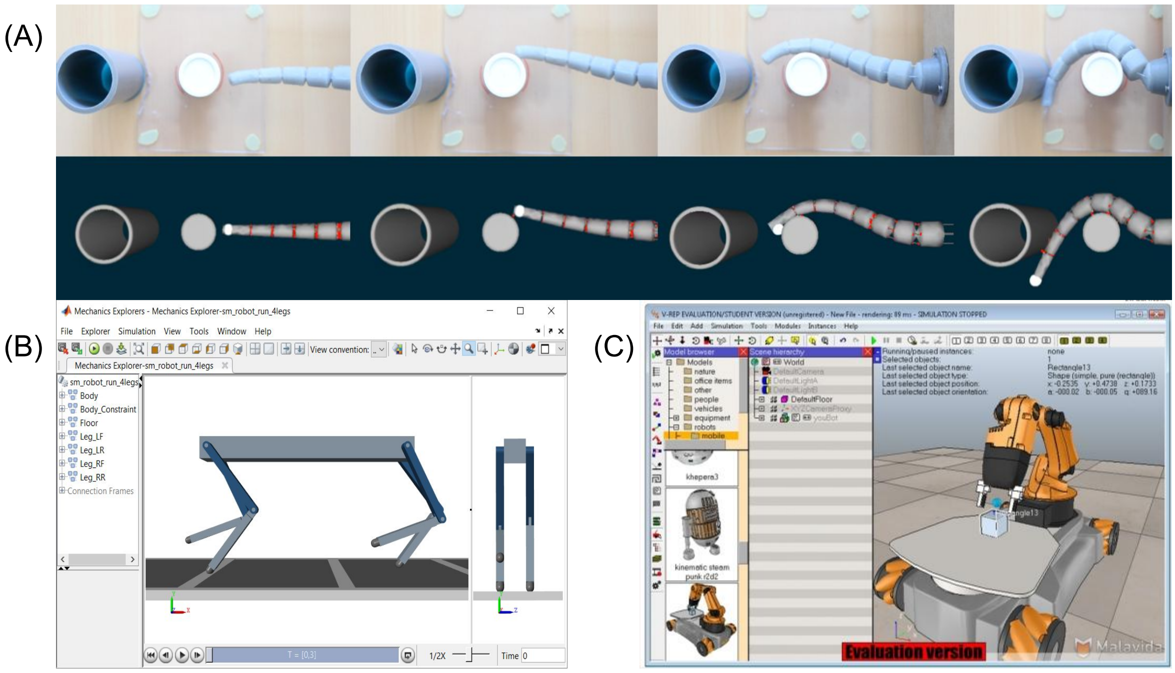

Figure 4. (A) Soft Robot Simulation on SOFA using Soft-robotics toolkit. (B) Walking Robot Simulation on V-REP. (C) Manipulation Robot Simulation on MATLAB using Simscape Modeling

10. Imitation Learning for Soft Robotic Actuators

There are several drawbacks of training a DRL agent to perform control tasks : relatively high training time and data that requires robot to interact with the real world which is computationally expensive, a well formulated reward function which might not be practical to obtain in some scenarios. Imitation Learning is a unique approach to perform a control task especially the ones involving use of bio-inspired robots without the requirement of a reward function, but requires an expert whose actions are mimicked by the agent[14]. In situations where we have such an expert present with high degrees of movement/action space and it is difficult to give a reward function to describe the problem, Imitation Learning is extremely useful. An overview of the training procedure for an imitation learning agent is shown in figure 5.

The use of imitation learning for solving problems of manipulation like picking, dropping, etc.[121,122] where we can exploit several benefits of soft robotics over hard ones have become extremely essential. Controls tasks in such situations generally have tough to compute cost functions due to high dimension of action space caused by the flexibility introduced in the motion of the actuator/end-effector because of the soft structure of the robot leading to increased difficulty of applying DRL techniques. Under such situations the study of imitation learning becomes a topic of utmost significance due to the fact that it doesn't require a cost function, all that it requires is an expert agent which in most cases of manipulation using soft robotics is a person who performs the same tasks that robot is required to copy. Therefore, manipulation with soft robotics and imitation learning algorithms for performing the control task in hand really go hand in hand and compliment each other, and hence, combining these two and finding new and better ways to do so may be a topic that gathers much attention in the coming years.

The most primitive imitation learning algorithm is supervised learning problem. But simply applying the normal steps of supervised learning to tasks involving formulation of control policy doesn't work. There are some minor and major changes/variations that must be made due to difference

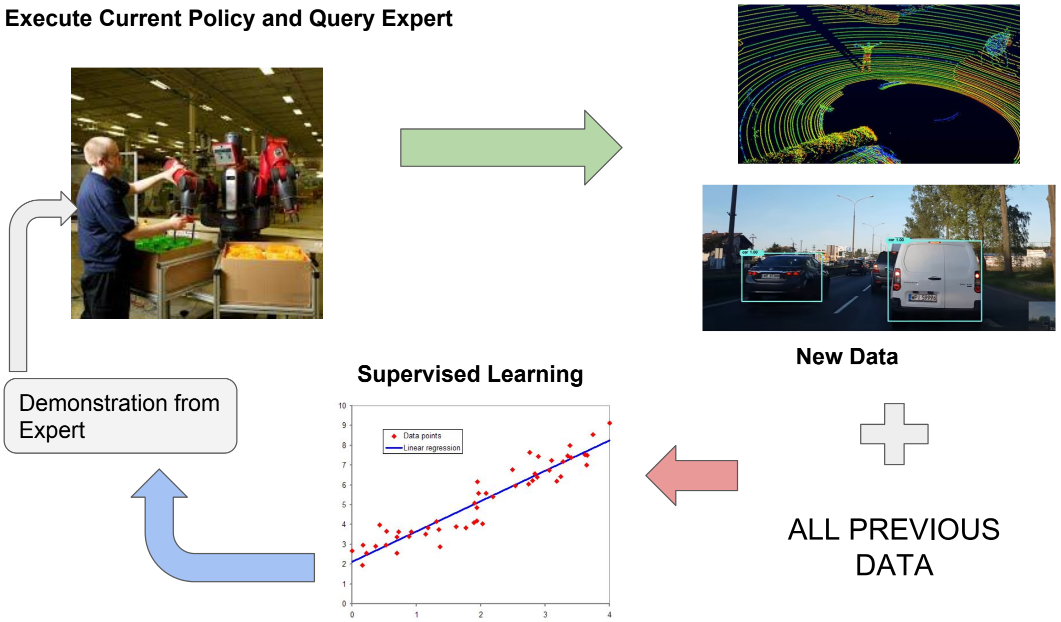

Figure 5. The figure depicts the training procedure of an Imitation Learning Agent

in common supervised learning problems and control problems. Following section provides overview of some of the variations:

- Errors - Independent or Compound: The basic assumption of common supervised learning task that assumes that the actions of the soft robotics agent do not affect the environment in any way is violated in the case of imitation learning control tasks. The presupposition that all data samples (observations) collected are independent and identically distributed is not valid for imitation learning tasks, hence, causing error pr opagation making the system highly unstable due to minor errors too.
- Time-step Decisions - Single or Multiple: Supervised learning models generally ignore the dependence between consecutive decisions which might be different from what primitive robotic approaches. The goal of imitation learning might be sometimes different from simply mimicking the expert's actions. At times, there is a hidden objective that is often missed by the agent while simply copying the actions of the expert. These hidden objectives are somethings like avoiding to collide with obstacles, increasing the chances to complete a specific task, or minimizing the effort by the mobile robot.

In the next section, we describe three of the main imitation learning algorithms that have been successfully applied to real life scenarios effectively.

- Behaviour Cloning: This is one of most basic imitation learning approaches in which we train a supervised learning agent based on actions of the expert agent from input states to output states via performed actions. Despite it also facing problems mentioned in the last section, it gives reliable results provided we have good amount of training data available. DAGGer(Data AGGregation)[123] is one of the algorithms described earlier that solves the problem of propagation of errors in a sequential system. This simple yet useful algorithm is quite similar to common supervised learning problems in which at each iteration the updated (current) policy is applied and observations hence recorded are labeled by expert policy. The data collected is concatenated to the data already available and the training procedure is applied to it. This technique has been readily utilized in diverse domain applications due it simplicity and effectiveness. Even though this algorithm has now been observed to give satisfactory results in various fields of controls but this is still not believed to be effective with soft robots, and so in those case we generally avoid using this due to lack of labelled data. Bojarski *et al.*[124] trained a navigation control model that collected data from 72 hours of actual driving by the expert agent and tried to mimic the state (images pixels) to actions (steering commands) with the help of a mapping function. Similarly, Tai *et al.*[125] and Giusti *et al.*[126] also came up with imitation learning applications for real life robotic control. Advanced readings also include Codevilla *et al.*[127] and Dosovitskiy *et al.*[102].

Imitation learning is also effective in problems involving manipulation. Some of them are listed below:

- Duan *et al.*[128] improved the one-shot imitation learning to formulate the low-dimensional state to action mapping, using behavioural cloning that tries to reduce the differences in agent's and the expert actions. He used this method in order to make a robotic arm stack various blocks in the way the expert does it, observing the relative position of the block at each time step. The performance level achieved after incorporating various other additional features like temporal dropouts and convolutions was similar to that of a DAGGer.
- Finn *et al.*[129] and Yu *et al.*[46] modified the already existing Model Agnostic Meta-Learning (MAML)[130], which is a highly diverse algorithm that train a model on several varied tasks and making it capable to solve a new unseen task when given it. The updation of weights is quite similar to the common gradient algorithm and given by equation:

  $$\theta_i' = \theta - \alpha \nabla_\theta L_{T_i}(f_\theta). \tag{27}$$

  The learning is done to achieve the objective function given by:

  $$\sum_{T_i \sim p(T)} L_{T_i}(f_{\theta'}) = \sum_{T_i \sim p(T)} L_{T_i}(f_{\theta - \alpha} \nabla_{\theta L_{T_i}(f_\theta)}) \tag{28}$$

  which leads us to the gradient descent step given by:

  $$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{T_i \sim p(T)} L_{T_i}(f_{\theta_i'}) \tag{29}$$

  wherein β represents the meta step size.
- While Duan *et al.*[128] and Finn *et al.*[129] proposes a way to train a model that works on newer set of samples as well the earlier described Yu *et al.*[46] is an effective algorithm in case of domain shift problems. Eitel *et al.*[131] came up with a new approach wherein he gave a new model that takes in over segmented RGB-D images as inputs and gives actions as outputs for segregation of objects in an environment.

- Inverse Reinforcement Learning: This method aims to formulate the utility function that makes the desired behaviour nearly optimal. Once, we have the utility function, our tasks is simplified as we only have to apply reinforcement learning algorithms on it to find the correct policy. A popular IRL algorithm called as Maximum Entropy IRL[132] uses an objective function as given by equation:

$$arg \max_{c \in C}(\min_{\pi \in \prod} -H(\pi) + E_\pi[c(s,a)]) - E_{\pi^E}[c(s,a)]. \tag{30}$$

It is a highly appropriate algorithm for robotic applications where the robot is supposed to follow a set a constraints[133–135] (like for a soft robot trying to pick and drop certain coloured boxes the constraints could to assemble them according to their colour) as in such situations it is not viable to formulate an accurate reward function but these actions are easy to demonstrate. Soft robotic systems generally have constraint issues due to the composition and configuration of different materials of the actuators resulting in elimination of a certain part of the action space (or sometimes state space also) hence, forcing us to involve IRL techniques under such situations. This is an extremely fruitful algorithm for soft robotic systems unlike the one mentioned before this, due to the ease of performing the task that we want to solve by the human expert due to the flexibility and adaptability of the soft robot. Maximum Entropy IRL[136] has been successfully used alongside deep convolutional networks to learn the multiplex representations in problems involving a soft robot to copy the actions of a human expert for simple control tasks.

- Generative Adversarial Imitation Learning: Even though, IRL algorithms are highly effective but they require large sets of data and huge training time. Hence, a more efficient alternative was proposed by Ho and Ermon[137] who gave Generative Adversarial Imitation Learning (GAIL) that comprises of an Generative Adversarial Network (GAN)[87]. Such generative models are extremely essential when working with soft robotic systems as they require huge sets of training data because of the wide variety of actions-state pairs possible in such cases and the fact that GANs are much complex deep networks that are able to learn complex representations in the latent space.

  Like all other GANs, GAIL also consists of two separately and independently trained fragments - generator (or the decoder) that tries generate state-action pairs close to that of the expert and the discriminator that learns to distinguish between samples created by the generator and real samples. The objective function of such a model is given by equation:

$$E_{\pi_\theta}[\log(D(s,a))] + E_{\pi^E}[\log(1 - D(s,a))] - \lambda H(\pi_\theta). \tag{31}$$

  Some extensions of GAIL have been proposed in recent works including Baram *et al.*[138] and Wang *et al.*[139]. GAIL has been extremely successful in solving imitation learning problems in navigation (Li *et al.*[140] applied GAIL to autonomous navigation problems and Tai *et al.*[141] applied it for the purpose of finding socially complaint policies) as well as manipulation (Stadie *et al.*[142] used GAIL for mimicking an expert's actions through domain agnostic representation). This presents before us an opportunity to be applied to systems involving soft actuators for its composite structure and unique learning technique.

Imitation learning for soft robotics being a relatively newer field of research, hasn't yet been explored to its fullest and has extreme potential for future research lying ahead of it. It is highly effective in the domains of industrial applications, capable of replacing its human counterpart due to its precision, accuracy, reliability, and efficiency. It is the future of robotic developments in all areas where we have an expert agent (in most situations a person) whose actions can be mimicked to perfection by an autonomous soft robotics agent. These are the go-to algorithms in fields that involve tough and sometimes almost impossible formulation of an appropriate cost function due to high action space dimensionality of soft robots. These techniques present before us a huge scope of combining with common DRL approaches and could form a really effective amalgam that could copy the expert as well as learn on it own via exploration depending on the situation in hand, and hence, must be a center for future deep learning developments in soft robots.

11. Future Scope

Even though, deep learning has been successfully applied to innumerable real life problems and has proven out to be the best alternative to solve control problems like manipulation in soft robots, there are still various challenges that need to be conquered and hence, open new avenues for further growth and research in this field. Further, we list some of the steps stones in the path of using deep reinforcement approaches to solve such tasks that will certainly be hot topics of research in the near future:

- Sample Efficiency: It takes great amount of effort and resources in collecting observations for training by making our agent interact with the environments especially for soft robotic systems due to the huge number of actions possible at each state pertaining to the biomimetic motions possible [9,143] because of the flexible bio-inspired actuators, hence, making way for further research in creating efficient systems who can collect experiences without much expense.
- Strong Real-time Requirements: Training huge networks with millions of neurons and tons of tuneable parameters that requires special hardware and loads of time. The current policies need to made compact in its representation to prevent wastage of time and hardware resources in training. The dimensionality of the actions as well as the state space for soft robotic actuated systems is much sizeable as compared to its hard counterpart leading to a rise in the number of neurons in the deep network.
- Safety Concerns: The control policies need designed need to extremely precise in its decision making process as robots like factories producing food items using soft robots are required to operate in environments where even a small error could cause loss of life and property.
- Stability, Robustness and Interpretability[144]: Slight changes in configurations of parameters or robotic hardware or chances in concentration or composition of material of soft robot over time might affect the performance of the agent in a great way, hence making it unstable. Especially in soft robotic systems a constant or static configuration of the agent is extremely hard to maintain and the fact that our model is unable to operate for such changes after completion of training is a challenges when involving these technologies on such systems. Some kind of a learned representation that can detect adversarial scenarios could be of great use and a topic of interest for researchers aiming to improve performance of DRL agents on soft robotic systems.
- Lifelong Learning: The appearance of the environment differs drastically when observed at different moments, alongside the composition and configuration of soft robotics systems also varying with time could result in a certain dip in performance of the learned policies. Hence, this problem provokes us to create adapting technologies that are always evolving and learning from changes in the environmental conditions besides keeping the policies already learnt intact.
- Generalization between tasks: A completely trained model is able to perform well in the tasks it has been trained on but performs poorly in new tasks and situations. For soft robotics systems that are required to perform varied set of tasks that are correlated, it is necessary to come up with methods that can transfer the learning from one training procedure when being tested on some other (yet correlated in some way) task. So, there is a requirement of creating completely autonomous systems that take up least resources for training and still are diverse in application.

Despite these challenges in control problems for soft robots, there are some topics that are gaining attention of DRL researchers around due to future scope of development in these areas of research. Two of them are:

- Unifying Reinforcement Learning and Imitation Learning: There have been quite a few developments[145–148] with the aim to combine the two algorithms and reap the benefits of both wherein the agent can learn from the actions of the expert agent alongside interacting and collecting experiences from the environment itself. The learning from expert's actions (generally a person for soft robotic manipulation problems) can sometimes lead to less-optimal solutions while using deep neural networks to train reinforcement learning agent can turn out to be an expensive task. Current research in this domain focuses on creating a model where soft robotic agent is able to learn from expert's demonstrations and then as the time progresses it moves to a more DRL based exploration technique wherein it interacts with the continuously evolving environment to collect observations. In the near future, it may be quite possible to see complete self-determining soft robotics systems that have the best of both world - can learn from the expert in the beginning and equipped with capabilities to learn on its own when necessary hence, resulting in giving full justice to the benefits of the amalgamated mechanical structure by exploiting all its benefits. Such advancements could really boost their mechanical capabilities and help them outperform not only humans but also robots trained only using either imitation learning or DRL techniques.

- Meta-Learning: Methods proposed in Finn *et al.*[130] and by Nichol and Schulman[149] have found out a way to find parameters that help agents to learn from relatively less data samples and produce much better results on newer tasks that they have not been trained on. These development can be prospective stepping stones to further developments leading to creation of completely robust and universal policy solutions. This could be a milestone research item when it comes to combining deep learning technologies with soft robotics, as generally it is hard to retrieve a large dataset for soft robotic systems due to the heavy expenses in allowing it to interact with its environment. Soft robotic systems are generally harder to deal with compared to the harder ones and therefore, such learning procedures could aid our soft systems to perform satisfactorily well even with a small set of training data.

Control of soft robots that have enhanced flexibility and strength due to their structure and material that is inspired by living beings, has become one of the premier domains of recent research and has caught the attention of robotics researchers around the globe. There have been numerous DRL and imitation learning algorithms proposed for such systems but there is still some room for enhancement due the challenges stated above. Some recent works have shown massive span for further development including some that could branch out as separate areas of soft robotics research themselves. These challenges have opened new doors for more such artificially intelligent algorithms that will be a trending topic of discussion and debate for the coming decades. Combining deep learning frameworks with soft robotic systems and extracting the benefits of both is seen a potential area of future developments.

12. Conclusion

This paper gives an overview of popular deep reinforcement learning and imitation learning algorithms that have successfully been applied to problems involving control of soft robots and have been observed to give state-of-art results in their domains of applications especially manipulation where soft robots are extensively utilized. We have majorly described learning paradigms of various such learning techniques, followed by the instances of them being applied to solve real life robotic control problems. Despite the massive growth in research in this field of universal interest in the last decade, there are still challenges in controls of soft robots (for it being a relatively new field of research in robotics) that need more concentrated attention. Soft Robotics is a constantly growing academic field that focuses exploiting the mechanical structure by integration of materials, structures and software, and when combined with the boons of imitation learning and other DRL mechanisms can create systems capable of replacing human at each discipline possible. We also list the stepping stones to the development such soft robots that are completely autonomous and self-adapting yet physically strong systems, besides mentioning some future areas of research in this domain that has so much to offer.

In the nutshell, the subject that gathers the attention of one and all remains to be - how the incorporation of DRL and imitation learning approaches can help accelerate the ever so satisfactory performances of soft robotic systems and unveil before us plethora of possibilities of creating altogether self-sufficient systems in the near future.

Acknowledgments: This work was supported by the Singapore Academic Research Fund under Grant R-397-000-227-112, and the NUSRI China Jiangsu Provincial Grants BK20150386 & BE2016077 awarded to H.R.

Author Contributions: H.R. provided the outline for the draft and critical revision; S.B. and H.B. conducted the literature search and S.B., H.B.drafted the manuscript; H.B. and S.B. also collected data for tables and figures.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Trimmer, B. A Confluence of Technology: Putting Biology into Robotics, 2014.

2. Banerjee, H.; Tse, Z.T.H.; Ren, H. SOFT ROBOTICS WITH COMPLIANCE AND ADAPTATION FOR BIOMEDICAL APPLICATIONS AND FORTHCOMING CHALLENGES. *International Journal of Robotics and Automation* **2018**, *33*.
3. Trivedi, D.; Rahn, C.D.; Kier, W.M.; Walker, I.D. Soft robotics: Biological inspiration, state of the art, and future research. *Applied bionics and biomechanics* **2008**, *5*, 99–117.
4. Banerjee, H.; Ren, H. Electromagnetically responsive soft-flexible robots and sensors for biomedical applications and impending challenges. In *Electromagnetic Actuation and Sensing in Medical Robotics*; Springer, 2018; pp. 43–72.
5. Banerjee, H.; Aaron, O.Y.W.; Yeow, B.S.; Ren, H. Fabrication and Initial Cadaveric Trials of Bi-directional Soft Hydrogel Robotic Benders Aiming for Biocompatible Robot-Tissue Interactions.
6. Kim, S.; Laschi, C.; Trimmer, B. Soft robotics: a bioinspired evolution in robotics. *Trends in biotechnology* **2013**, *31*, 287–294.
7. Ren, H.; Banerjee, H. A Preface in Electromagnetic Robotic Actuation and Sensing in Medicine. In *Electromagnetic Actuation and Sensing in Medical Robotics*; Springer, 2018; pp. 1–10.
8. Banerjee, H.; Shen, S.; Ren, H. Magnetically Actuated Minimally Invasive Microbots for Biomedical Applications. In *Electromagnetic Actuation and Sensing in Medical Robotics*; Springer, 2018; pp. 11–41.
9. Banerjee, H.; Suhail, M.; Ren, H. Hydrogel Actuators and Sensors for Biomedical Soft Robots: Brief Overview with Impending Challenges. *Biomimetics* **2018**, *3*, 15.
10. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural networks* **2015**, *61*, 85–117.
11. Andrychowicz, M.; Wolski, F.; Ray, A.; Schneider, J.; Fong, R.; Welinder, P.; McGrew, B.; Tobin, J.; Abbeel, O.P.; Zaremba, W. Hindsight experience replay. Advances in Neural Information Processing Systems, 2017, pp. 5048–5058.
12. Deng, L. A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing* **2014**, *3*.
13. Guo, Y.; Liu, Y.; Oerlemans, A.; Lao, S.; Wu, S.; Lew, M.S. Deep learning for visual understanding: A review. *Neurocomputing* **2016**, *187*, 27–48.
14. Bagnell, J.A. An invitation to imitation. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA ROBOTICS INST, 2015.
15. Ho, S.; Banerjee, H.; Foo, Y.Y.; Godaba, H.; Aye, W.M.M.; Zhu, J.; Yap, C.H. Experimental characterization of a dielectric elastomer fluid pump and optimizing performance via composite materials. *Journal of Intelligent Material Systems and Structures* **2017**, *28*, 3054–3065.
16. Shepherd, R.F.; Ilievski, F.; Choi, W.; Morin, S.A.; Stokes, A.A.; Mazzeo, A.D.; Chen, X.; Wang, M.; Whitesides, G.M. Multigait soft robot. *Proceedings of the national academy of sciences* **2011**, *108*, 20400–20403.
17. Banerjee, H.; Pusalkar, N.; Ren, H. Single-Motor Controlled Tendon-Driven Peristaltic Soft Origami Robot. *Journal of Mechanisms and Robotics* **2018**, *10*, 064501.
18. Sutton, R.S.; Barto, A.G.; others. *Reinforcement learning: An introduction*; MIT press, 1998.
19. Dayan, P. Improving generalization for temporal difference learning: The successor representation. *Neural Computation* **1993**, *5*, 613–624.
20. Kulkarni, T.D.; Saeedi, A.; Gautam, S.; Gershman, S.J. Deep successor reinforcement learning. *arXiv preprint arXiv:1606.02396* **2016**.
21. Barreto, A.; Dabney, W.; Munos, R.; Hunt, J.J.; Schaul, T.; van Hasselt, H.P.; Silver, D. Successor features for transfer in reinforcement learning. Advances in neural information processing systems, 2017, pp. 4055–4065.
22. Zhang, J.; Springenberg, J.T.; Boedecker, J.; Burgard, W. Deep reinforcement learning with successor features for navigation across similar environments. Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on. IEEE, 2017, pp. 2371–2378.
23. Fu, M.C.; Glover, F.W.; April, J. Simulation optimization: a review, new developments, and applications. Proceedings of the 37th conference on Winter simulation. Winter Simulation Conference, 2005, pp. 83–95.
24. Szita, I.; Lörincz, A. Learning Tetris using the noisy cross-entropy method. *Neural computation* **2006**, *18*, 2936–2941.
25. Schulman, J.; Moritz, P.; Levine, S.; Jordan, M.; Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438* **2015**.

26. Williams, R.J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* **1992**, *8*, 229–256.
27. Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; Riedmiller, M. Deterministic policy gradient algorithms. ICML, 2014.
28. Sutton, R.S. Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin* **1991**, *2*, 160–163.
29. Weber, T.; Racanière, S.; Reichert, D.P.; Buesing, L.; Guez, A.; Rezende, D.J.; Badia, A.P.; Vinyals, O.; Heess, N.; Li, Y.; others. Imagination-augmented agents for deep reinforcement learning. *arXiv preprint arXiv:1707.06203* **2017**.
30. Kalweit, G.; Boedecker, J. Uncertainty-driven imagination for continuous deep reinforcement learning. Conference on Robot Learning, 2017, pp. 195–206.
31. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; others. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529.
32. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* **2015**.
33. Van Hasselt, H.; Guez, A.; Silver, D. Deep Reinforcement Learning with Double Q-Learning. AAAI. Phoenix, AZ, 2016, Vol. 2, p. 5.
34. Wang, Z.; Schaul, T.; Hessel, M.; Van Hasselt, H.; Lanctot, M.; De Freitas, N. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581* **2015**.
35. Gu, S.; Lillicrap, T.; Sutskever, I.; Levine, S. Continuous deep q-learning with model-based acceleration. International Conference on Machine Learning, 2016, pp. 2829–2838.
36. Gu, S.; Holly, E.; Lillicrap, T.; Levine, S. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. Robotics and Automation (ICRA), 2017 IEEE International Conference on. IEEE, 2017, pp. 3389–3396.
37. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. International conference on machine learning, 2016, pp. 1928–1937.
38. Wang, J.X.; Kurth-Nelson, Z.; Tirumala, D.; Soyer, H.; Leibo, J.Z.; Munos, R.; Blundell, C.; Kumaran, D.; Botvinick, M. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763* **2016**.
39. Wu, Y.; Mansimov, E.; Grosse, R.B.; Liao, S.; Ba, J. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. Advances in neural information processing systems, 2017, pp. 5279–5288.
40. Levine, S.; Koltun, V. Guided policy search. International Conference on Machine Learning, 2013, pp. 1–9.
41. Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; Moritz, P. Trust region policy optimization. International Conference on Machine Learning, 2015, pp. 1889–1897.
42. Kakade, S.; Langford, J. Approximately optimal approximate reinforcement learning. ICML, 2002, Vol. 2, pp. 267–274.
43. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* **2017**.
44. Mirowski, P.; Pascanu, R.; Viola, F.; Soyer, H.; Ballard, A.J.; Banino, A.; Denil, M.; Goroshin, R.; Sifre, L.; Kavukcuoglu, K.; others. Learning to navigate in complex environments. *arXiv preprint arXiv:1611.03673* **2016**.
45. Riedmiller, M.; Hafner, R.; Lampe, T.; Neunert, M.; Degrave, J.; Van de Wiele, T.; Mnih, V.; Heess, N.; Springenberg, J.T. Learning by Playing-Solving Sparse Reward Tasks from Scratch. *arXiv preprint arXiv:1802.10567* **2018**.
46. Yu, T.; Finn, C.; Xie, A.; Dasari, S.; Zhang, T.; Abbeel, P.; Levine, S. One-Shot Imitation from Observing Humans via Domain-Adaptive Meta-Learning. *arXiv preprint arXiv:1802.01557* **2018**.
47. Levine, S.; Finn, C.; Darrell, T.; Abbeel, P. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research* **2016**, *17*, 1334–1373.
48. Jaderberg, M.; Mnih, V.; Czarnecki, W.M.; Schaul, T.; Leibo, J.Z.; Silver, D.; Kavukcuoglu, K. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397* **2016**.

49. Schaul, T.; Quan, J.; Antonoglou, I.; Silver, D. Prioritized experience replay. *arXiv preprint arXiv:1511.05952* **2015**.
50. Bengio, Y.; Louradour, J.; Collobert, R.; Weston, J. Curriculum learning. Proceedings of the 26th annual international conference on machine learning. ACM, 2009, pp. 41–48.
51. Zhang, J.; Tai, L.; Boedecker, J.; Burgard, W.; Liu, M. Neural SLAM. *arXiv preprint arXiv:1706.09520* **2017**.
52. Florensa, C.; Held, D.; Wulfmeier, M.; Zhang, M.; Abbeel, P. Reverse curriculum generation for reinforcement learning. *arXiv preprint arXiv:1707.05300* **2017**.
53. Pathak, D.; Agrawal, P.; Efros, A.A.; Darrell, T. Curiosity-driven exploration by self-supervised prediction. International Conference on Machine Learning (ICML), 2017, Vol. 2017.
54. Sukhbaatar, S.; Lin, Z.; Kostrikov, I.; Synnaeve, G.; Szlam, A.; Fergus, R. Intrinsic motivation and automatic curricula via asymmetric self-play. *arXiv preprint arXiv:1703.05407* **2017**.
55. Fortunato, M.; Azar, M.G.; Piot, B.; Menick, J.; Osband, I.; Graves, A.; Mnih, V.; Munos, R.; Hassabis, D.; Pietquin, O.; others. Noisy networks for exploration. *arXiv preprint arXiv:1706.10295* **2017**.
56. Plappert, M.; Houthooft, R.; Dhariwal, P.; Sidor, S.; Chen, R.Y.; Chen, X.; Asfour, T.; Abbeel, P.; Andrychowicz, M. Parameter space noise for exploration. *arXiv preprint arXiv:1706.01905* **2017**.
57. Zhu, Y.; Mottaghi, R.; Kolve, E.; Lim, J.J.; Gupta, A.; Fei-Fei, L.; Farhadi, A. Target-driven visual navigation in indoor scenes using deep reinforcement learning. Robotics and Automation (ICRA), 2017 IEEE International Conference on. IEEE, 2017, pp. 3357–3364.
58. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
59. Kolve, E.; Mottaghi, R.; Gordon, D.; Zhu, Y.; Gupta, A.; Farhadi, A. AI2-THOR: An interactive 3d environment for visual AI. *arXiv preprint arXiv:1712.05474* **2017**.
60. Tai, L.; Paolo, G.; Liu, M. Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation. Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on. IEEE, 2017, pp. 31–36.
61. Chen, Y.F.; Everett, M.; Liu, M.; How, J.P. Socially aware motion planning with deep reinforcement learning. Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on. IEEE, 2017, pp. 1343–1350.
62. Long, P.; Fan, T.; Liao, X.; Liu, W.; Zhang, H.; Pan, J. Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning. *arXiv preprint arXiv:1709.10082* **2017**.
63. Thrun, S.; Burgard, W.; Fox, D. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*; 2001.
64. Gupta, S.; Davidson, J.; Levine, S.; Sukthankar, R.; Malik, J. Cognitive mapping and planning for visual navigation. *arXiv preprint arXiv:1702.03920* **2017**, *3*.
65. Gupta, S.; Fouhey, D.; Levine, S.; Malik, J. Unifying map and landmark based representations for visual navigation. *arXiv preprint arXiv:1712.08125* **2017**.
66. Parisotto, E.; Salakhutdinov, R. Neural map: Structured memory for deep reinforcement learning. *arXiv preprint arXiv:1702.08360* **2017**.
67. Kümmerle, R.; Grisetti, G.; Strasdat, H.; Konolige, K.; Burgard, W. g 2 o: A general framework for graph optimization. Robotics and Automation (ICRA), 2011 IEEE International Conference on. IEEE, 2011, pp. 3607–3613.
68. Parisotto, E.; Chaplot, D.S.; Zhang, J.; Salakhutdinov, R. Global pose estimation with an attention-based recurrent network. *arXiv preprint arXiv:1802.06857* **2018**.
69. Schaul, T.; Horgan, D.; Gregor, K.; Silver, D. Universal value function approximators. International Conference on Machine Learning, 2015, pp. 1312–1320.
70. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint* **2017**.
71. Khan, A.; Zhang, C.; Atanasov, N.; Karydis, K.; Kumar, V.; Lee, D.D. Memory augmented control networks. *arXiv preprint arXiv:1709.05706* **2017**.
72. Bruce, J.; Sünderhauf, N.; Mirowski, P.; Hadsell, R.; Milford, M. One-shot reinforcement learning for robot navigation with interactive replay. *arXiv preprint arXiv:1711.10137* **2017**.
73. Chaplot, D.S.; Parisotto, E.; Salakhutdinov, R. Active neural localization. *arXiv preprint arXiv:1801.08214* **2018**.

74. Savinov, N.; Dosovitskiy, A.; Koltun, V. Semi-parametric topological memory for navigation. *arXiv preprint arXiv:1803.00653* **2018**.
75. Heess, N.; Sriram, S.; Lemmon, J.; Merel, J.; Wayne, G.; Tassa, Y.; Erez, T.; Wang, Z.; Eslami, A.; Riedmiller, M.; others. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286* **2017**.
76. Calisti, M.; Giorelli, M.; Levy, G.; Mazzolai, B.; Hochner, B.; Laschi, C.; Dario, P. An octopus-bioinspired solution to movement and manipulation for soft robots. *Bioinspiration biomimetics* **2011**, *6 3*, 036002.
77. Martinez, R.V.; Branch, J.L.; Fish, C.R.; Jin, L.; Shepherd, R.F.; Nunes, R.M.D.; Suo, Z.; Whitesides, G.M. Robotic tentacles with three-dimensional mobility based on flexible elastomers. *Advanced materials* **2013**, *25 2*, 205–12.
78. Finn, C.; Tan, X.Y.; Duan, Y.; Darrell, T.; Levine, S.; Abbeel, P. Deep spatial autoencoders for visuomotor learning. *arXiv preprint arXiv:1509.06113* **2015**.
79. Tzeng, E.; Devin, C.; Hoffman, J.; Finn, C.; Peng, X.; Levine, S.; Saenko, K.; Darrell, T. Towards adapting deep visuomotor representations from simulated to real environments. *CoRR, abs/1511.07111* **2015**.
80. Fu, J.; Levine, S.; Abbeel, P. One-shot learning of manipulation skills with online dynamics adaptation and neural network priors. Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. IEEE, 2016, pp. 4019–4026.
81. Kumar, V.; Todorov, E.; Levine, S. Optimal control with learned local models: Application to dexterous manipulation. Robotics and Automation (ICRA), 2016 IEEE International Conference on. IEEE, 2016, pp. 378–383.
82. Gupta, A.; Eppner, C.; Levine, S.; Abbeel, P. Learning dexterous manipulation for a soft robotic hand from human demonstrations. Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. IEEE, 2016, pp. 3786–3793.
83. Popov, I.; Heess, N.; Lillicrap, T.; Hafner, R.; Barth-Maron, G.; Vecerik, M.; Lampe, T.; Tassa, Y.; Erez, T.; Riedmiller, M. Data-efficient deep reinforcement learning for dexterous manipulation. *arXiv preprint arXiv:1704.03073* **2017**.
84. Prituja, A.; Banerjee, H.; Ren, H. Electromagnetically Enhanced Soft and Flexible Bend Sensor: A Quantitative Analysis With Different Cores. *IEEE Sensors Journal* **2018**, *18*, 3580–3589.
85. Sun, J.Y.; Zhao, X.; Illeperuma, W.R.; Chaudhuri, O.; Oh, K.H.; Mooney, D.J.; Vlassak, J.J.; Suo, Z. Highly stretchable and tough hydrogels. *Nature* **2012**, *489*, 133–136.
86. Tzeng, E.; Hoffman, J.; Zhang, N.; Saenko, K.; Darrell, T. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474* **2014**.
87. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. Advances in neural information processing systems, 2014, pp. 2672–2680.
88. Radford, A.; Metz, L.; Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* **2015**.
89. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein gan. *arXiv preprint arXiv:1701.07875* **2017**.
90. Hoffman, J.; Tzeng, E.; Park, T.; Zhu, J.Y.; Isola, P.; Saenko, K.; Efros, A.A.; Darrell, T. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213* **2017**.
91. Doersch, C. Tutorial on Variational Autoencoders. *CoRR* **2016**, *abs/1606.05908*.
92. Szabó, A.; Hu, Q.; Portenier, T.; Zwicker, M.; Favaro, P. Challenges in Disentangling Independent Factors of Variation. *CoRR* **2017**, *abs/1711.02245*.
93. Mathieu, M.; Zhao, J.J.; Sprechmann, P.; Ramesh, A.; LeCun, Y. Disentangling factors of variation in deep representations using adversarial training. NIPS, 2016.
94. Bousmalis, K.; Irpan, A.; Wohlhart, P.; Bai, Y.; Kelcey, M.; Kalakrishnan, M.; Downs, L.; Ibarz, J.; Pastor, P.; Konolige, K.; others. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. *arXiv preprint arXiv:1709.07857* **2017**.
95. Tobin, J.; Fong, R.; Ray, A.; Schneider, J.; Zaremba, W.; Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on. IEEE, 2017, pp. 23–30.
96. Peng, X.B.; Andrychowicz, M.; Zaremba, W.; Abbeel, P. Sim-to-real transfer of robotic control with dynamics randomization. *arXiv preprint arXiv:1710.06537* **2017**.

doi:10.20944/preprints201811.0510.v1

97. Rusu, A.A.; Vecerik, M.; Rothörl, T.; Heess, N.; Pascanu, R.; Hadsell, R. Sim-to-real robot learning from pixels with progressive nets. *arXiv preprint arXiv:1610.04286* **2016**.
98. Zhang, J.; Tai, L.; Xiong, Y.; Liu, M.; Boedecker, J.; Burgard, W. Vr goggles for robots: Real-to-sim domain adaptation for visual control. *arXiv preprint arXiv:1802.00265* **2018**.
99. Ruder, M.; Dosovitskiy, A.; Brox, T. Artistic style transfer for videos and spherical images. *International Journal of Computer Vision* **2018**, pp. 1–21.
100. Koenig, N.P.; Howard, A. Design and use paradigms for Gazebo, an open-source multi-robot simulator. IROS. Citeseer, 2004, Vol. 4, pp. 2149–2154.
101. Maddern, W.; Pascoe, G.; Linegar, C.; Newman, P. 1 year, 1000 km: The Oxford RobotCar dataset. *The International Journal of Robotics Research* **2017**, *36*, 3–15.
102. Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; Koltun, V. CARLA: An open urban driving simulator. *arXiv preprint arXiv:1711.03938* **2017**.
103. Chen, L.C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062* **2014**.
104. Yang, L.; Liang, X.; Xing, E. Unsupervised Real-to-Virtual Domain Unification for End-to-End Highway Driving. *arXiv preprint arXiv:1801.03458* **2018**.
105. Uesugi, K.; Shimizu, K.; Akiyama, Y.; Hoshino, T.; Iwabuchi, K.; Morishima, K. Contractile performance and controllability of insect muscle-powered bioactuator with different stimulation strategies for soft robotics. *Soft Robotics* **2016**, *3*, 13–22.
106. Niiyama, R.; Sun, X.; Sung, C.; An, B.; Rus, D.; Kim, S. Pouch Motors : Printable Soft Actuators Integrated with Computational Design. 2015.
107. Gul, J.Z.; Sajid, M.; Rehman, M.M.; Siddiqui, G.U.; Shah, I.; Kim, K.C.; Lee, J.W.; Choi, K.H. 3D printing for soft robotics – a review. Science and technology of advanced materials, 2018.
108. Umedachi, T.; Vikas, V.; Trimmer, B. Highly deformable 3-D printed soft robot generating inching and crawling locomotions with variable friction legs. *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems* **2013**, pp. 4590–4595.
109. Mutlu, R.; Tawk, C.; Alici, G.; Sariyildiz, E. A 3D printed monolithic soft gripper with adjustable stiffness. *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society* **2017**, pp. 6235–6240.
110. Lu, N.; hyeong Kim, D. Flexible and Stretchable Electronics Paving the Way for Soft Robotics. 2013.
111. Rohmer, E.; Singh, S.P.; Freese, M. V-REP: A versatile and scalable robot simulation framework. Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on. IEEE, 2013, pp. 1321–1326.
112. Shah, S.; Dey, D.; Lovett, C.; Kapoor, A. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. Field and service robotics. Springer, 2018, pp. 621–635.
113. Pan, X.; You, Y.; Wang, Z.; Lu, C. Virtual to real reinforcement learning for autonomous driving. *arXiv preprint arXiv:1704.03952* **2017**.
114. Savva, M.; Chang, A.X.; Dosovitskiy, A.; Funkhouser, T.; Koltun, V. MINOS: Multimodal indoor simulator for navigation in complex environments. *arXiv preprint arXiv:1712.03931* **2017**.
115. Wu, Y.; Wu, Y.; Gkioxari, G.; Tian, Y. Building generalizable agents with a realistic and rich 3D environment. *arXiv preprint arXiv:1801.02209* **2018**.
116. Iida, F.; Laschi, C. Soft robotics: challenges and perspectives. *Procedia Computer Science* **2011**, *7*, 99–102.
117. Coevoet, E.; Bieze, T.M.; Largilliere, F.; Zhang, Z.; Thieffry, M.; Sanz-Lopez, M.; Carrez, B.; Marchal, D.; Goury, O.; Dequidt, J.; Duriez, C. Software toolkit for modeling, simulation, and control of soft robots. *Advanced Robotics* **2017**, *31*, 1208–1224.
118. Duriez, C.; Coevoet, E.; Largilliere, F.; Bieze, T.M.; Zhang, Z.; Sanz-Lopez, M.; Carrez, B.; Marchal, D.; Goury, O.; Dequidt, J. Framework for online simulation of soft robots with optimization-based inverse model. *2016 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)* **2016**, pp. 111–118.
119. Rohmer, E.; Singh, S.P.N.; Freese, M. V-REP: A versatile and scalable robot simulation framework. *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems* **2013**, pp. 1321–1326.
120. Olaya, J.; Pintor, N.; Avilés, O.F.; Chaparro, J. Analysis of 3 RPS Robotic Platform Motion in SimScape and MATLAB GUI Environment. *International Journal of Applied Engineering Research* **2017**, *12*, 1460–1468.
121. Ratliff, N.D.; Bagnell, J.A.; Srinivasa, S.S. Imitation learning for locomotion and manipulation. *2007 7th IEEE-RAS International Conference on Humanoid Robots* **2007**, pp. 392–397.

122. Langsfeld, J.D.; Kaipa, K.N.; Gentili, R.J.; Reggia, J.A.; Gupta, S.K. Towards Imitation Learning of Dynamic Manipulation Tasks : A Framework to Learn from Failures. 2014.

123. Ross, S.; Gordon, G.; Bagnell, D. A reduction of imitation learning and structured prediction to no-regret online learning. Proceedings of the fourteenth international conference on artificial intelligence and statistics, 2011, pp. 627–635.

124. Bojarski, M.; Del Testa, D.; Dworakowski, D.; Firner, B.; Flepp, B.; Goyal, P.; Jackel, L.D.; Monfort, M.; Muller, U.; Zhang, J.; others. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316* **2016**.

125. Tai, L.; Li, S.; Liu, M. A deep-network solution towards model-less obstacle avoidance. Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. IEEE, 2016, pp. 2759–2764.

126. Giusti, A.; Guzzi, J.; Ciresan, D.C.; He, F.L.; Rodríguez, J.P.; Fontana, F.; Faessler, M.; Forster, C.; Schmidhuber, J.; Di Caro, G.; others. A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots. *IEEE Robotics and Automation Letters* **2016**, *1*, 661–667.

127. Codevilla, F.; Müller, M.; Dosovitskiy, A.; López, A.; Koltun, V. End-to-end driving via conditional imitation learning. *arXiv preprint arXiv:1710.02410* **2017**.

128. Duan, Y.; Andrychowicz, M.; Stadie, B.C.; Ho, J.; Schneider, J.; Sutskever, I.; Abbeel, P.; Zaremba, W. One-Shot Imitation Learning. NIPS, 2017.

129. Finn, C.; Yu, T.; Zhang, T.; Abbeel, P.; Levine, S. One-shot visual imitation learning via meta-learning. *arXiv preprint arXiv:1709.04905* **2017**.

130. Finn, C.; Abbeel, P.; Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400* **2017**.

131. Eitel, A.; Hauff, N.; Burgard, W. Learning to singulate objects using a push proposal network. *arXiv preprint arXiv:1707.08101* **2017**.

132. Ziebart, B.D.; Maas, A.L.; Bagnell, J.A.; Dey, A.K. Maximum Entropy Inverse Reinforcement Learning. AAAI. Chicago, IL, USA, 2008, Vol. 8, pp. 1433–1438.

133. Okal, B.; Arras, K.O. Learning socially normative robot navigation behaviors with bayesian inverse reinforcement learning. Robotics and Automation (ICRA), 2016 IEEE International Conference on. IEEE, 2016, pp. 2889–2895.

134. Pfeiffer, M.; Schwesinger, U.; Sommer, H.; Galceran, E.; Siegwart, R. Predicting actions to act predictably: Cooperative partial motion planning with maximum entropy models. Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. IEEE, 2016, pp. 2096–2101.

135. Kretzschmar, H.; Spies, M.; Sprunk, C.; Burgard, W. Socially compliant mobile robot navigation via inverse reinforcement learning. *The International Journal of Robotics Research* **2016**, *35*, 1289–1307.

136. Wulfmeier, M.; Ondruska, P.; Posner, I. Maximum entropy deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888* **2015**.

137. Ho, J.; Ermon, S. Generative adversarial imitation learning. Advances in Neural Information Processing Systems, 2016, pp. 4565–4573.

138. Baram, N.; Anschel, O.; Mannor, S. Model-based adversarial imitation learning. *arXiv preprint arXiv:1612.02179* **2016**.

139. Wang, Z.; Merel, J.S.; Reed, S.E.; de Freitas, N.; Wayne, G.; Heess, N. Robust imitation of diverse behaviors. Advances in Neural Information Processing Systems, 2017, pp. 5320–5329.

140. Li, Y.; Song, J.; Ermon, S. Inferring the latent structure of human decision-making from raw visual inputs. *arXiv preprint* **2017**.

141. Tai, L.; Zhang, J.; Liu, M.; Burgard, W. Socially-compliant navigation through raw depth inputs with generative adversarial imitation learning. *arXiv preprint arXiv:1710.02543* **2017**.

142. Stadie, B.C.; Abbeel, P.; Sutskever, I. Third-person imitation learning. *arXiv preprint arXiv:1703.01703* **2017**.

143. Banerjee, H.; Ren, H. Optimizing double-network hydrogel for biomedical soft robots. *Soft robotics* **2017**, *4*, 191–201.

144. Henderson, P.; Islam, R.; Bachman, P.; Pineau, J.; Precup, D.; Meger, D. Deep reinforcement learning that matters. *arXiv preprint arXiv:1709.06560* **2017**.

145. Vecerík, M.; Hester, T.; Scholz, J.; Wang, F.; Pietquin, O.; Piot, B.; Heess, N.; Rothörl, T.; Lampe, T.; Riedmiller, M.A. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *CoRR, abs/1707.08817* **2017**.

146. Nair, A.; McGrew, B.; Andrychowicz, M.; Zaremba, W.; Abbeel, P. Overcoming exploration in reinforcement learning with demonstrations. *arXiv preprint arXiv:1709.10089* **2017**.

147. Gao, Y.; Lin, J.; Yu, F.; Levine, S.; Darrell, T.; others. Reinforcement learning from imperfect demonstrations. *arXiv preprint arXiv:1802.05313* **2018**.

148. Zhu, Y.; Wang, Z.; Merel, J.; Rusu, A.; Erez, T.; Cabi, S.; Tunyasuvunakool, S.; Kramár, J.; Hadsell, R.; de Freitas, N.; others. Reinforcement and imitation learning for diverse visuomotor skills. *arXiv preprint arXiv:1802.09564* **2018**.

149. Nichol, A.; Schulman, J. Reptile: a Scalable Metalearning Algorithm. *arXiv preprint arXiv:1803.02999* **2018**.