Bio-inspired Adaptive Control of Robotic Manipulators for Space Debris Removal and On-Orbit Servicing

Collins Ogundipe[1] and Alex Ellery[1]

[1]Department of Mechanical & Aerospace Engineering, Carleton University, 1125 Colonel by Drive, Ottawa, K1S 5B6, Canada
collinsogundipe@cmail.carleton.ca

Abstract: Adaptive and compliant manipulation has long been a major constraint for grappling in robotic manipulators, especially in space. Currently, space manipulators are typically teleoperated in space by astronauts or by ground operators and are operated very slowly. This acts as a severe restriction on productivity rates, largely because the space manipulators rely on the traditional feedback control to work. With feedback control, errors must exist to invoke the corrective behaviour. However, this is not the case for feedforward control which does not require errors to work. To adopt robotic manipulators in space for the prospect of capturing space debris and transforming them into salvageable assets for re-use, robust adaptive manipulation would be key. We believe that a bio-inspired feedforward approach could provide human-like tactility required for robustness and adaptability in robotic manipulation. We present a novel predictive feedforward control via a forward model, followed by a complete overview of our learning algorithms. Given the similarity in form and dynamics between earth-based and space-based robotic manipulators, we first explored the transfer learning of neural network controllers as an avenue to address the challenges of limited computation resources onboard spacecraft. We introduced a pretrained and learned feedforward neural network for modeling the control error a priori. While the results were encouraging, there are major limitations of neural networks' capability to ensuring the transfer learning of similar earth-based dynamics to space-based dynamics, given that the parameters of contrast are fairly straightforward. With the results not as plausible as expected, an alternative adaptive controller has been learned to demonstrate a viable solution. The controller was trained entirely in simulation via rapid motor adaptation of the robot's controller to the object's properties and environmental dynamics using only proprioception history. As a notable step, we have shown that appropriate models can be learned in this manner by training the control policy via reinforcement learning, which provides avenue for transferring a learned model from earth to space environments. We have shown the viability of this approach for adaptive and compliant space manipulator controller transferable from earth-learned model in simulation of space environment, relying largely on proprioception history.

Keywords: Keywords: Transfer Learning, Neural Network, Forward Model, Space Manipulator, Rapid Motor Adaptation, Proprioception History.

1 INTRODUCTION

In human level manipulation, various sections of the brain extend into the motor area M1 to supply feedback signals. The parietal cortex, for instance, deals with visual control of hand motions, and it calculates the error between the current cartesian position and the desired cartesian position [1]. To do this, an efference copy of the motor commands is required to produce a feedforward compensation. The efference copy of the motor commands is typically transmitted to an emulator which models the input-output response of the musculoskeletal system. From a biomimetic perspective, it is believed that a hierarchical neural network system in any control architecture can imitate this function of the motor cortex [2]. During human manipulation, the error between the actual motor outputs (joint position (θ) and joint velocity ($\dot{\theta}$) evaluated by the proprioceptors) and the commanded motor input (torque τ, from the motor cortex) is fed back as $[\theta^{desired} - \theta]$ having a time delay of 40-60 ms [3]. However, a "forward dynamics model of the musculoskeletal system exists within the spinocerebellum-magnocellular red nucleus system" [3]. This forward model accepts feedback (θ and $\dot{\theta}$) from the proprioceptors and an afferent copy of the motor command (τ) from the motor cortex. Consequently, the forward model receives motor command τ as its input and outputs an estimated predictive trajectory θ^* [3], processing this input-output comparison between the pair (τ and θ^*) to generate a predicted error $[\theta^{desired} - \theta^*]$ in a much faster manner to minimize the error. The forward model does this prediction/comparison in 10-20 ms, transmitting this to the motor cortex in the process [3]. The sensory effects of the motor command are predicted by this forward model. This type of top-down prediction model is centered on the statistical reproducible model of the causative nature of the world learned via input-output pairs. This can be directly explored with predictive neural networks as forward model by adopting input-output models of deep learning architecture or multivariate regression. In human level interaction, these forward models of the musculoskeletal system have been learned through the initial motor babbling that started from infancy [3]. And the learned models are transferred to adapt to changes in stimuli or environments, given the underlying dynamics remain the same.

This leads to the practical problem we have detailed in this paper, which is the transfer learning from earth-based manipulators to space-based manipulators. In space robotics, there are simply two fundamental changes from earth to space which are accounted for through: (i) the absence of gravity in space, and (ii) the direct substitutions of certain derived parameters which are quantified in numbers and readily available as a modification of the earth-based equivalents. So, essentially, the dynamics of the robotic system remain the same, and necessary environmental variations are readily accounted for. All other space-based environmental factors are known to be negligible as they pertain to the dynamics of space robot's interaction. The environmental disturbance torques (gravity gradient, aerodynamics and magnetic torques) imposed on the robot's spacecraft are very small – within 10e-6 Nm [4]. The primary differentiating characteristics of space robotics from terrestrial robotics is that the robot operates in a microgravity environment. Transfer learning of neural network controller trained as a forward model in a biomimetic approach similar to how human manipulation is carried out should be able to exhibit efficient generalization as typically shown for new data input in most deep learning domain/applications. However, the practical limitation of transfer learning of neural network controllers is the exhibition of lack of general intelligence, as detailed in this paper.

Considerable effort has been put into developing machine learning methods that can learn and improve inverse dynamics model of robotic manipulators [5-8]. Online learning has been the focus in these settings because when considering motions with object interactions, learning one global model becomes very challenging, if not impossible, since the model must be a function of contact and payload signals. To approach the issue of global/dynamic model, learning task-specific (error) models has been proposed in the past [9-12], such that the overall global problem is simplified into two subproblems – (1) finding a task-specific inverse dynamics model and (2) detecting which task model to use. This permits to iterate the collection of data specific to a task, learn an error model, and then apply the learned model during the required task execution. However, a key difficulty that has been encountered is the computationally efficient learning of models that are data-efficient as possible, such that only few iterations are required while achieving consistent convergence in the error model learning. We seek to address this using predictive feedforward approach, in a pre-learned fashion, by ensuring the transfer learning of earth-based model to space environment. Our take on this is that pre-learned input-output models are computationally efficient compared with analytical models – the latter require exact knowledge of parameters (commonest sources of errors which include payload variation) and require computation time. Learned models reduce computation by storing model in memory, which also ensure a more compliant and reactive robot.

For feedback control to work, errors must exist to invoke the corrective behaviour. This is not the case for feedforward control which does not require errors to work. Forward model implemented in conjunction with feedback control reduces the potential error excursions [13]. Currently, space manipulators are typically teleoperated in space by astronauts or by ground operators and are operated very slowly. This acts as a severe restriction on productivity rates. The incorporation of feedforward controllers, therefore, offers the advantage to robustify and speed up operations as they do not require error excursion to function. In the following, we first described in Section 2, the background to the derived parameters relating space-based manipulator's kinematics and dynamics to earth-based environment. In Section 3, we detailed a novel predictive feedforward control via a forward model; followed by a complete overview of the bio-inspired motor adaptation approach in Section 4. Finally, we evaluated the results of the proposed scheme in Section 5 and outlined the conclusions in Section 6.

2 SPACE-BASED KINEMATICS AND DYNAMICS

We must first consider the kinematics and dynamics of a freeflyer-mounted manipulator. The main differentiating characteristics of space robots from terrestrial robots is that terrestrial robots are mounted onto a firm ground; in space, we have no such force or torque reaction cancellation to the movement of manipulator arms. Additionally, the robot operates in a microgravity environment; hence, the kinematics and dynamics of free-flying robotic manipulator deployed in space will take a different approach. In the consideration of a free-flying robotic manipulator mounted on a spacecraft bus having dedicated attitude control, the position kinematics (p^*) of the manipulator in connection with inertial space is given by [14, 15]:

$$p^* \quad = r_{c0} + R_0 s_0 + \sum_{i=1}^{n} R_i \, l_i \qquad (1)$$

where r_{c0} is the position of the spacecraft centre of mass with respect to the inertial coordinates; R_0 is the attitude of the spacecraft with respect to the inertial coordinates; s_0 is the position vector of the manipulator base with respect to the spacecraft body centre of mass; R_i is the 3-by-3 direction cosine matrix of each link with respect to the base coordinates; n is the number of serial rigid body links; i represents the link number from 0 to n; while l_i is the vectoral length of link i from $(x_{i-1}, y_{i-1}, z_{i-1})$ to (x_i, y_i, z_i).

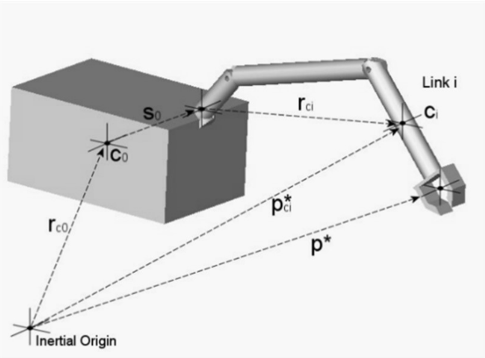


Figure 1: Spacecraft-Manipulator Geometry (C_0 represents spacecraft's center of mass; r_{ci} is the distance between the centres of mass of adjacent links with respect to the base coordinates; C_i is the center of mass of link i; p^*_{ci} is the position of the link i centre of mass with respect to the inertial coordinates)

For spacecraft bus with dedicated attitude control, $R_0 = I_3$ (identity matrix). The center of mass of the whole system (the robotic manipulator, satellite bus mount, and the payload) is represented by [14, 15]:

$$p^*_{cm} = \frac{\sum_{i=0}^{n+1} m_i \, p^*_{ci}}{\sum_{i=0}^{n+1} m_i} \qquad (2)$$

where p^*_{cm} is the location of the centre of mass of the complete manipulator/spacecraft system with regards to the inertial coordinates; m_i is the mass of each component rigid body links; n is the number of rigid body links; $n = 0$ represents the spacecraft body link; p^*_{ci} is the position of link i centre of mass in reference to the inertial coordinates. Similarly to terrestrial manipulator algorithms in the form of $p_i = R_i l_i$, the equation of the space manipulator for the location of the center of mass of the complete manipulator/spacecraft system with regards to the inertial coordinates (p^*_{cm}) has been derived to be [16-18]:

$$\begin{aligned} p^*_{cm} &= r_{c0} + \left(1 - \frac{m_0}{m_T}\right) s_0 + \frac{1}{m_T} \sum_{i=1}^{n} R_i \left(\sum_{j=i+1}^{n+1} m_j \, l_i + m_i r_i \right) .. \\ &+ \frac{m_{n+1}}{m_T} R_{n+1} r_{n+1} \qquad (3) \end{aligned}$$

where m_0 is the mass of the spacecraft bus; m_T is the total mass of the system; m_i is the mass of each component rigid body i comprising the system; r_i is the vectorial distance from the origin of link i to the centre of mass of link i; $n + 1$ represents the corresponding notations for the payload link. Equation (3) was separated into three parts: parts related to body 0 (the spacecraft), bodies 1 to n (the manipulator links) and body $n + 1$ (for the payload). This then reduces to [18]:

$$\begin{aligned} p^*_{cm} &= r_{c0} + (1 - \frac{m_0}{m_T}) s_0 + \sum_{i=1}^{n} R_i \, L_i + (\frac{m_{n+1}}{m_T}) r_{n+1} \\ &\text{where } L_i = \frac{1}{m_T} \left(\sum_{j=i+1}^{n+1} m_j \, l_i + m_i r_i \right) \qquad (4) \end{aligned}$$

This concludes the location of center of mass of the system with respect to inertial space. It is assumed arbitrarily that the local inertial reference frame initially coincides with the spacecraft bus center of mass, that is, $r_{c0} = 0$, since any point fixed in the interceptor body could be regarded as inertially fixed prior to any robotic maneuver [18]. Having defined p^*_{cm}, the term r_{c0} is then substituted into Equation (1), which gives

$$p^* = p^*_{cm} + s_0 + \sum_{i=1}^{n} R_i \, l_i - \frac{1}{m_T} \sum_{i=1}^{n+1} \sum_{j=i}^{n+1} m_j \, r_{ci} \qquad (5)$$

This is further simplified into:

$$\begin{aligned} p^* &= p^*_{cm} + s_0 + \sum_{i=1}^{n} R_i\, l_i \; - \; \dots \\ &\frac{1}{m_T}\sum_{i=1}^{n+1}\sum_{j=i}^{n+1} m_j \left(R_i r_i + R_{i-1} s_{i-1}\right) \qquad (6) \end{aligned}$$

where $r_{ci} = R_i r_i + R_{i-1} s_{i-1}$ [18]. Similarly, we separate out the three parts associated to the spacecraft mount (body 0), bodies $1\ to\ n$ for the manipulator links and body n+1 for the payload [18]. This gives

$$\begin{aligned} p^* &= p^*_{cm} + \frac{m_0}{m_T} s_0 + \sum_{i=1}^{n} R_i\, \lambda_i - \frac{m_{n+1}}{m_T} R_{n+1} r_{n+1} \\ &\text{where } \lambda_i = \frac{1}{m_T}\sum_{j=0}^{i} \left(m_j l_j - m_i r_i \right) \qquad (7) \end{aligned}$$

Accordingly, λ_i is referred to as the lumped kinematic parameter for each manipulator link. The equation (7) of p^* is an equivalent form to that of the terrestrial-based manipulator of the form $p = \sum_{i=1}^{n} R_i\, l_i$ with added constants; (p^*_{cm} is constant, and λ_i is constant as the lumped kinematic/dynamic parameter, replacing the l_i in terrestrial-based manipulator).

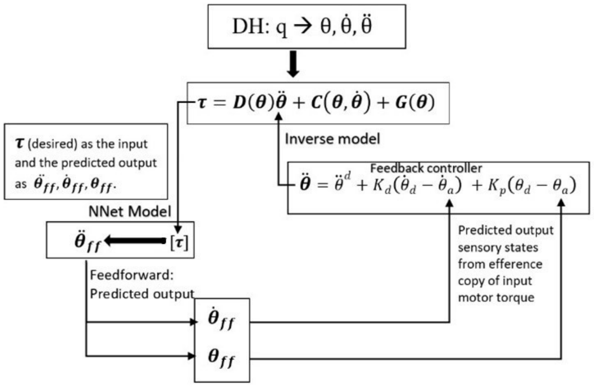
Therefore, the inverse kinematics solution to the space manipulator geometry can be found with little modifications to the terrestrial algorithms.

3 PREDICTIVE FEEDFORWARD CONTROL

Our bio-inspired error-learning approach addresses the need for reactive and adaptive behavior to diverse range of tasks under dynamic environmental conditions. If we could successfully demonstrate this for a terrestrial manipulator, the idea is to incorporate the approach in a free-flyer concept for the removal of space debris of varying sizes; with the aim to offer a solution transferrable from earth to different orbital bands. In effect, we propose here a control scheme that is centered on biomimetic models for predictive forward control in conjunction with traditional feedback control. We believe that bio-inspired forward models could provide solution for adaptive and robust control, which could position robotic manipulators for the complex task of salvaging space debris if the learned model can be successfully transferred to space environment. Adaptivity will be implemented through learning of new forward models to adapt to new situations; robustness is implemented in the form of forward models that provide rapid behavior without relying on error excursions unlike traditional feedback controllers. The superiority of feedforward-feedback control over feedback control only has been clearly demonstrated [13]. Pure feedback control is implausible for reactive manipulation due to substantial delays in sensors' feedback signals. This is like the case biologically where human reaction time is limited to a maximum of about 400-500 ms [19]. Therefore, a predictive feedforward strategy is proposed as added measure to correct the robot's trajectory along with the feedback control. In this current study, we have not yet implemented feedback delays into the forward model yet – the work presented here is the first step in building a more comprehensive and sophisticated manipulator control system. The bio-inspired control system should comprise a paired feedforward-feedback system with a learning system that adapts forward models for different scenarios such as time delays and/or payload variations. Hence, the core of this approach is the forward model presented. A two-layer approach towards grasping has been presented: (i) position control through feedback, which is the traditional approach - but delays in the feedback cycle can generate instabilities; (ii) the addition of a feedforward predictive capability to partially circumvent this problem of instabilities by adopting pre-trained set of neural networks which in a way emulates the function of the cerebellum as seen in humans.

The predictive feedforward approach involves pre-learned models trained offline, which then provide a computationally efficient control model for low controller gains necessary for reactive and adaptive control. We have introduced task-specific models that are able to learn from their errors (make error predictions) under different and varying dynamics. The proposed approach is more practical for space-based manipulators because there would be no major hindrances such as high computational complexity; and secondly, the trained forward models do not require high computational resources to implement which is usually a constraint onboard spacecraft. This is where transfer learning comes in as a practical solution for transferring pre-trained earth-based model to space environment. Most automatic control algorithms have not been demonstrated in space as most manipulator control systems are teleoperated from earth.

Here, we present a forward model that is learned (or trained) as a neural network approximator using some trajectory datasets relating the output torque τ to the kinematic state of the joints $(\theta, \dot{\theta}, \ddot{\theta})^T$ in an experimental teaching mode. The trained forward model will hence be able to take the analytically calculated torque (efference copy of input motor commands) as its input, while the output of the neural network will be the



DH: q → θ, θ̇, θ̈
τ = D(θ)θ̈ + C(θ, θ̇) + G(θ)
Inverse model
τ (desired) as the input and the predicted output as θ̈ff, θ̇ff, θff.
NNet Model
θ̈ff ← [τ]
Feedback controller
θ̈ = θ̈^d + Kd(θ̇d − θ̇a) + Kp(θd − θa)
Predicted output sensory states from efference copy of input motor torque
Feedforward: Predicted output
θ̇ff
θff

Figure 2: The predictive forward model scheme. The neural network ("NNet") model is trained using data from experimental teaching mode. DH stands for Denavit–Hartenberg; q represents DH parameters for forward kinematics. D, C and G represent inertia matrix, coriolis and gravity components respectively; K_d and K_p are derivative and proportional controller gains.

predicted trajectory output $(\theta_{ff},\ \dot{\theta}_{ff},\ \ddot{\theta}_{ff})^T$. The system then incorporates an inverse model with a feedforward adaptive part; that is, it includes a feedback loop and feedforward component. The feedforward controller is trained using the output of the feedback controller which serve as error signals. The trained feedforward component models the inverse dynamics of the system. The feedback controller is effectively a computed torque controller while the feedforward controller employs a gradient descent to minimize the error.

The forward dynamic model of a robotic manipulator is given by (for the sensory joint acceleration rate):

$$\ddot{\theta} = D^{-1}(\theta)[\tau - C(\theta, \dot{\theta}) - G(\theta)]$$

Joint acceleration $\ddot{\theta}$ could be integrated to get joint rate $\dot{\theta}$ and joint rotation θ as the predicted sensory state outputs from torque input τ. The body's muscular nature which produces a predicted trajectory output from efference input motor commands can be imitated by the predictive forward model [20]. To compensate for time delays, the feedforward control consequently predicts its response to system disturbances using a model of the plant process [21]. This predicted trajectory output would be supplied as input to the feedback component to compensate for delays (and this process could continue iteratively). It is believed that forward models can adjust 7.5 times more speedily than when using only inverse models [22]. The forward model, in this case, is executed as a neural network function estimator to the forward dynamics.

4 BIO-INSPIRED MOTOR ADAPTATION

Results of the predictive feedforward model for a WAM Barrett space manipulator have been detailed in a recent study [23], where the results showed how the developed neural network and regression models were capable of predicting (to a high degree of accuracy) forward trajectory variables $(\theta_{ff},\ \dot{\theta}_{ff},\ \ddot{\theta}_{ff})$ from an efference copy of the torque. Models were poised to cancel the sensory effects of the arm movement, providing anticipated sensory consequences from the motor command. However, there was a notable drawback to the trained feedforward model as it needed to re-learn the new dataset as adopted for the space manipulator [23]. It was discovered that the possibility of transfer learning could not be exploited after the optimized initial training, even though the deduced terrestrial-to-space manipulator dynamics were incorporated analytically in the model learning process.

With the results not as plausible as expected for trained neural network transfer from earth-based controller to space environment, an alternative adaptive controller has been learned to demonstrate a viable solution. The controller was trained entirely in simulation via rapid online adaptation of the robot's controller to the object properties and environmental state using only proprioception history. Successful real-world deployment of robotic manipulators in space would require them to adapt in real-time to unseen states like changing inertial origin, changing payloads, contact dynamics, and so on. We adopt here the Rapid Motor Adaptation (RMA) algorithms [24] to solve this problem of real-time online adaptation for robotic manipulator. RMA consists of two components: a base policy and an adaptation module. The combination of these components enables the manipulator to adapt to novel and different situations in a very compliant manner. RMA is trained completely in simulation without using any domain knowledge like reference trajectories or predefined hand trajectory generators and can be deployed in real-world robotic manipulator without fine-tuning, although the scope of this work does not cover real-world deployment beyond the simulation demonstrated. As a notable step, we have shown that appropriate models can be learned in this manner by training the control policy via reinforcement learning (RL), which provides avenue for transferring the learned model from earth to space environments. The prevailing approach is to train an RL-based controller in a physics simulation environment and thereafter transfer to the real world using several sim-to-real techniques [25, 26]. This transfer has proven quite difficult, because the sim-to-real gap itself is the result of multiple factors [24]

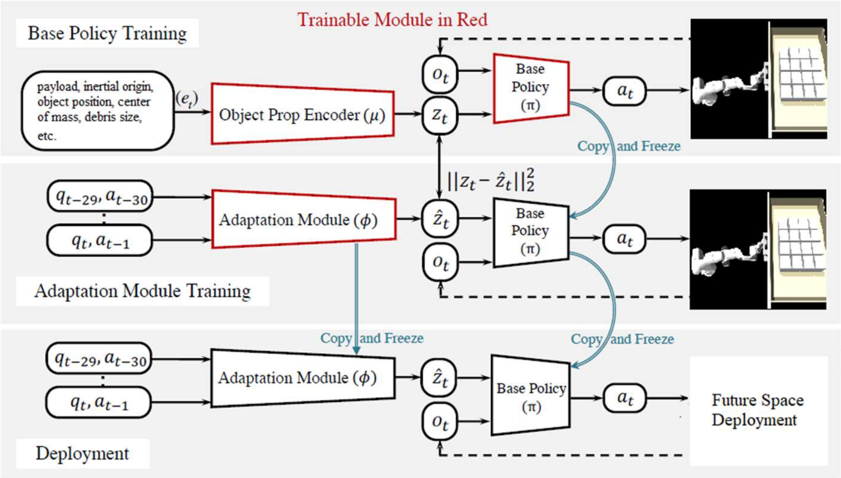


Figure 3: An overview of the RMA approach at different training and deployment stages [27]. In Base Policy Training, the base policy (π) and the environmental/object property encoder (μ) are jointly trained using PPO [28]. The observation o_t contains three past joint positions (the current and two previous $q_{t-2:t}$) and the past commanded actions (the three previous actions $a_{t-3:t-1}$). In Adaptation Module (*f*) Learning, the policy π is frozen and supervised learning is used to train *f* which uses proprioception and action history to estimate the extrinsics vector z_t.

Proposed RMA [24] demonstrates a simplified solution to this challenge for quadruped locomotion, using as an experimental platform the relatively cheap A1 robot from Unitree. It was postulated that RMA has to occur online, at a time scale of fractions of a second. This implies that there is no time to carry out multiple experiments in the physical world, rolling out multiple trajectories and optimizing to estimate various system parameters. As the simplified approach of RMA was demonstrated to be successful, we have hence adopted and extended the method to the problem of training the predictive feedforward model for the adaptive space manipulator approach highlighted in Section 3. This is following the limitations of the feedforward neural network controller highlighted in Section 5.1. This relates to the transfer learning of the pretrained model from earth to space environment without the need (and there is no such time allowed) for online retraining to guarantee adaptive and compliant manipulation. Hence, the reason it was postulated and designed such that RMA has to occur online at a time scale of fractions of a second if we would declare it as viable solution. This is quite crucial for space application of robotic manipulators for probable debris removal because with no prior experience, the robot/manipulator policy would fail often, causing grave damage to the manipulator in a failed attempt to deorbit the debris. Collecting even 2-3 minutes of manipulation data in order to adapt the manipulation policy may be practically infeasible. Our strategy therefore entails that not just the basic manipulation policy, but also RMA must be trained in simulation, and directly deployed in the real world. Figure 3 shows the overview of the RMA approach at different training and deployment stages. It consists of two subsystems: the base policy and the adaptation module, and they work in conjunction to enable online real-time adaptation on a diverse set of simulated environmental configurations. The base policy is trained via reinforcement learning in simulation using privileged information about the environment configuration (e_t) such as payload, inertial origin, object position, center of mass, etc. This stated privileged information can be deduced or perceived by the robotic manipulator and can be compressed into a compact feature representation which can be referred to as the extrinsics – represented with z_t in Figure 3. The vector e_t is encoded beforehand into the feature space z_t using an encoder network (μ) as shown in Figure 3. The vector z_t is then fed into the base policy (π) along with the observation o_t which contains three past joint positions (the current and two previous $q_{t-2:t}$) and the commanded actions (the three previous actions $a_{t-3:t-1}$). The base policy thereafter predicts the desired (next) position commands of the robotic manipulator (a_t), which are converted to torque using a PD controller. The base policy (π) and the object/environmental factor encoder (μ) are jointly trained via RL in simulation. However, this policy cannot be directly deployed because we do not have access to the vector e_t in the real world. Hence, we need to estimate the extrinsics at run time, which is where the Adaptation Module (ϕ) comes in. In adaptation module learning, the policy π is frozen and supervised learning is used to train ϕ which uses proprioception and action history to estimate the extrinsics vector z_t. When it comes to the stage of Deployment as shown in Figure 3, the base policy π uses the extrinsics $\hat{z}_t$ estimated and is updated online by ϕ. It is proven that the policy would infer a low-dimensional embedding of environmental or object's properties such as payload and inertial origin from proprioception and action history, which is then used by the base policy to manipulate the object [27].

5 RESULTS

In this present work, we have adopted and developed a robotic manipulation environment involving a 7 DOF Franka Emika Panda manipulator in simulation to grasp and pull (mimicking deorbiting) a cabinet which is acting as dummy debris with varying payloads. The simulated cabinet has a total of 4 DOF. A gripper with 2 DOF was affixed to the end of the robotic arm, making it a total of 9 DOF for the robotic manipulator. These 9 joints are controlled using position control at 20 Hz. The target position commands are converted to torque using a PD Controller (K p = 3.0, K d = 0.1) at 300 Hz. We use the IsaacGym [29] simulator in the setup. For the base policy training, parallel environments of 8,192 were used to collect the samples for training the agent. Each environment contains a simulated Franka Emika manipulator and a cabinet with different payloads. The simulation frequency is 120 Hz and the control frequency is 20 Hz. Each episode lasts for about 500 control steps. For the learning of the base policy (π) and environmental factor encoder network (μ), training took approximately 5-6 hours on an ordinary desktop machine with 24GB GPU memory, simulating the 1.2 billion total steps.

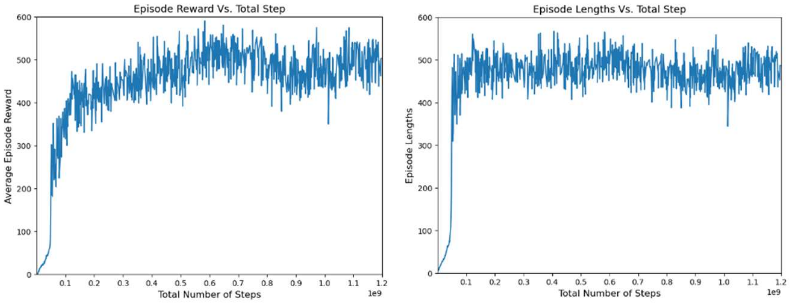


Figure 4: Results from Base Policy (π) Training on our environment with the implemented RMA algorithms. We plot the average episode reward during the total training of 1.2 billion steps. The vertical axis represented by the 'Average Episode Reward' could be termed the 'success rate' in a layman context. The sustained maximized reward shows that policy successfully learned and converged to the expected outcome, consistently and continuously over 1.2 billion steps.

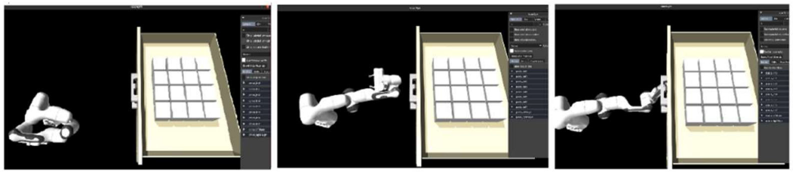


Figure 5: The expected stages of the learning process showing the expected outcome of the Franka manipulator grappling the space object. The base policy agent's reward system is aggregated for the grippers reaching the cartesian position of the handle, making successful contact with the handle, and grappling for the actual passivation to pull the drawer.

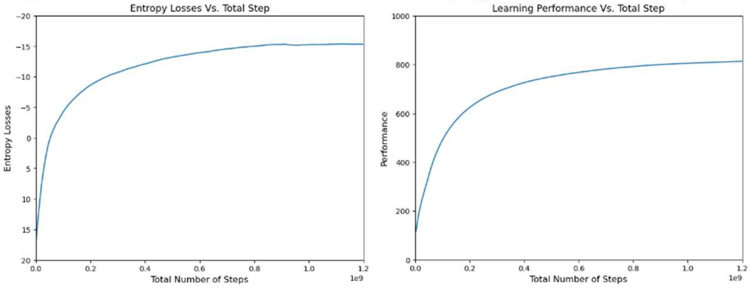


Figure 6: Results from Base Policy (π) Training on the environment with the implemented RMA algorithms, showing entropy losses and learning performance during the total training of 1.2 billion steps.

For the optimization during the base policy training, the joint optimization of the policy π and the environmental/object encoder network μ in each PPO iteration involves the collection of samples from the 8,192 environments with 8 agent steps each. We train 5 epochs with a batch size of 16,384, with 20 gradient updates on the simulated data. The learning rate was 5e−3, and we optimized for 50,000 gradient updates.

For the adaptation module (ϕ) training, we used supervised learning with on-policy data. Adam optimizer [30] was used to minimize the MSE loss. We run the optimization process over a total of 1 billion steps which took about 2 hours of training on the 24GB GPU memory desktop machine, with a learning rate of 5e−4. The major advantage of this approach is that we can use the enormous amount of simulation samples to learn a complex behavior such as this, which will not be possible in the real-world.

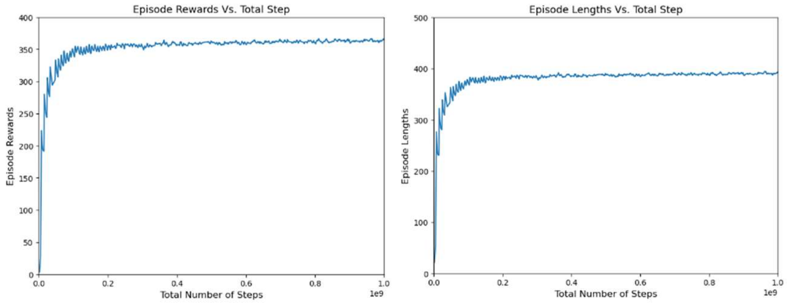


Figure 7: Results from Adaptation Module (ϕ) Training on the environment with the implemented RMA algorithms, showing the average episode reward during the total training of 1 billion steps, along with the episode lengths plot.

For the network architecture, the base policy (π) is a multi-layer perceptron (MLP) which takes in the state (o_t) and the extrinsics vector (z_t), and outputs the action vector (a_t). Every other network architecture and details were adopted and developed as proposed by [27] for the RMA. Compared to other baselines, it has been shown that the RMA method with online continuously adaptation achieves the best performance, as extensively investigated by [24, 27], closely matching the performance of the Expert that has the privileged information as the input.

6 CONCLUSION

Having highlighted a major drawback of input-output mapping of a neural network in retaining the kinematic and dynamic structure of the space manipulator, there is a practical limitation of regression and neural network architecture as a model of general intelligence. An alternate approach has been detailed in this paper exploring bio-inspired rapid motor adaptation as a solution. It has been demonstrated that an adaptive feedforward model can be pre-trained on earth in simulation prior to deployment and subsequently adapts to changes in the dynamics and other environmental parameters of the system in space. We have shown the viability of this approach for adaptive and compliant space manipulator controller transferable from learned model. Continuous online training as typically obtained in the context of traditional reinforcement learning deployment is not feasible in the application of space robotics, for reasons stated in Section 3 – constraint of high computational resources onboard spacecraft. The major advantage of this RMA approach is that we can use the enormous amount of simulation samples to learn a complex behavior such as this, which will be improbable in the real-world context. A future scope to the present body of work would be the experimental deployment of the learned model in real-world.

References

1. Bullock, D., Grossberg, S.: Cortical networks for control of voluntary arm movements under variable force conditions. Cerebral Cortex, vol. 8 (1), pp. 48-62 (1998).

2. Kawato, M., Furukawa, K., Suzuki, R.: Hierarchical neural network model for control and learning of voluntary movement. Biological Cybernetics, 57, pp. 169-185 (1987).
3. Ellery, A.: Tutorial review of bio-inspired approaches to robotic manipulation for space debris salvage. Biomimetics Journal, 12 (5), E19 (2020).
4. Shrivastava, S., Modi, V.: Satellite attitude dynamics and control in the presence of environmental torques—a brief survey. Journal of Guidance, Control, and Dynamics, vol. 6, pp. 461-471 (1983).
5. Vijayakumar, S., Schaal, S.: Locally weighted projection regression: Incremental real time learning in high dimensional space. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 1079–1086 (2000).
6. Nguyen-Tuong, D., Peters, J. R., Seeger, M.: Local Gaussian process regression for real time online model learning. In: Advances in Neural Information Processing Systems (NIPS), pp. 1193–1200 (2008).
7. Gijsberts, A., Metta, G.: Real-time model learning using incremental sparse spectrum Gaussian process regression. Neural Networks, vol. 41, pp. 59–69 (2013).
8. Meier, F., Hennig, P., Schaal, S.: Incremental Local Gaussian Regression. In: Advances in Neural Information Processing Systems, pp. 972–980 (2014).
9. Jamone, L., Damas, B., Santos-Victor, J.: Incremental learning of context-dependent dynamic internal models for robot control. In: Proceedings of the IEEE International Symposium on Intelligent Control (ISIC), pp. 1336-1341, doi: 10.1109/ISIC.2014.6967617 (2014).
10. Toussaint, M., Vijayakumar, S.: Learning discontinuities with products-of-sigmoids for switching between local models. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 904-911 (2005).
11. Petkos, G., Toussaint, M., Vijayakumar, S.: Learning multiple models of non-linear dynamics for control under varying contexts. In: International Conference on Artificial Neural Networks, pp. 898–907, Springer (2006).
12. Wolpert, D. M., Kawato, M.: Multiple paired forward and inverse models for motor control. Neural Networks, pp. 1317-1329, vol. 11 (7-8) (1998).
13. Ross, J., Ellery, A.: Panoramic camera tracking on planetary rovers using feedforward control. Int. Journal of Advanced Robotic Systems, pp. 1-9, May/Jun, (2017).
14. Lindberg, R., Longman, R., Zedd, M.: Kinematics and reaction moment compensation for the spaceborne elbow manipulator. In: 24th Aerospace Sciences Meeting, AIAA-86-0250, Nevada (1986).
15. Longman, R., Lindberg, R., Zedd, M.: Satellite-mounted robot manipulators—new kinematics and reaction compensation. Int. J. Robotics Res., vol. 6(3), pp. 87–103 (1987).
16. Vafa, Z., Dubowsky, S.: On dynamics of manipulators in space using the virtual manipulator approach. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 579–585 (1987).
17. Vafa, Z., Dubowsky, S.: Kinematics and dynamics of space manipulators: the virtual manipulator approach. Int. J. Robotics Res., vol. 9(4), 852–872 (1990).
18. Ellery, A.: An Introduction to Space Robotics. Praxis–Springer Series on Astronomy and Space Sciences, Praxis Publishers (2000).
19. Tovée, M. J.: Neuronal Processing: How fast is the speed of thought? Journal Current Biology vol. 4(12), pp. 1125-1127 (1994).
20. Morasso, P., Baratto, L., Capra, R., Spada, G.: Internal models in the control of posture. Neural Networks, 12, pp. 1173-1180 (1999).
21. Basso, D., Belardinelli, O.: Role of the feedforward paradigm in cognitive psychology. Cognitive Processes, vol. 7, pp. 73-88 (2006)
22. Flanagan, J., Vetter, P., Johansson, R., Wolpert, D.: Prediction precedes control in motor learning. Current Biology, vol. 13, pp. 146-150 (2003).
23. Ogundipe, C., Ellery, A.: Practical Limits to Transfer Learning of Neural Network Controllers from Earth to Space Environments. Proceedings at AI-2022 42nd SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence, Cambridge, England, December 13–15 (2022).
24. Kumar, A., Fu, Z., Pathak, D., Malik, J.: RMA: Rapid Motor Adaptation for Legged Robots. In: The Robotics: Science and Systems, https://doi.org/10.48550/arXiv.2107.04034 (2021).
25. 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. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE (2017).
26. Bin Peng, X., Andrychowicz, M., Zaremba, W., Abbeel, P.: Sim-to-real transfer of robotic control with dynamics randomization. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE (2018).
27. Qi, H., Kumar, A., Calandra, R., Ma, Yi., Malik, J.: In-Hand Object Rotation via Rapid Motor Adaptation. In: 6th Conference on Robot Learning (CoRL), Auckland, New Zealand (2022).
28. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal Policy Optimization Algorithms. arXiv, doi:10.48550/arXiv.1707.06347, (2017).
29. Makoviychuk, V., Wawrzyniak, L., Guo, Y., Lu, M., Storey, K., Macklin, M., Hoeller, D., Rudin, N., Allshire, A., Handa, A., State, G.: Isaac Gym: High Performance GPU-Based Physics Simulation for Robot Learning. arXiv, doi:10.48550/arXiv.2108.10470, (2021).
30. Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations (2015).