# ARI humanoid robot imitates human gaze behaviour using reinforcement learning in real-world environments

Khashayar Ghamati[1,*], Abolfazl Zaraki[1] and Farshid Amirabdollahian[1]

***Abstract*— This paper presents a novel approach to enhance the social interaction capabilities of the ARI humanoid robot using reinforcement learning. We focus on enabling ARI to imitate human attention/gaze behaviour by identifying salient points in dynamic environments, employing the Zero-Shot Transfer technique combined with domain randomisation and generalisation. Our methodology uses the Proximal Policy Optimisation algorithm, training the reinforcement learning agent in a simulated environment to maximise robustness in real-world scenarios. We demonstrated the efficacy of our approach by deploying the trained agent on the ARI humanoid and validating its performance in human-robot interaction scenarios. The results indicated that using the developed model, ARI can successfully identify and respond to salient points, exhibiting human-like attention/gaze behaviours, which is an important step towards acceptability and efficiency in human-robot interactions. This research contributes to advancing the capabilities of social robots in dynamic and unpredictable environments, highlighting the potential of combining Zero-Shot Transfer with domain randomisation and generalisation for robust real-world applications.**

Fig. 1: A triadic social HRI between ARI humanoid robot and a human participant, photographed at the Robot House, University of Hertfordshire, UK.

## I. INTRODUCTION

Robotics, unlike other disciplines, is more complex to optimise due to the dynamic environments in which robots operate. This requires continuous updates to models, which is computationally intensive and difficult to maintain [1]. Updating models necessitates collecting data while the robot is running and performing tasks, and learning from that data to adapt the robot to new situations is another challenge. Emerging Reinforcement Learning (RL) techniques offer capabilities that enable an agent to learn on the fly, modelling the environment's behaviour [2]. We anticipate that the combination of robotics and RL will help to overcome these challenges. However, in some scenarios, the agent trains in a simulation, making the transfer from simulation to reality challenging. The discrepancy between the simulated environment and the real world complicates the direct transfer of policies learned in simulation to real-world applications [3]. Unmodelled physical effects and differences in sensor fidelity between simulated and real-world environments contribute to this gap [4].

Training Deep Reinforcement Learning (DRL) agents directly in real-world environments is often impractical due to safety concerns, time constraints, and costs. Simulated environments offer a more controlled, safe, and scalable way to train these agents. Various methods address transferring models from simulation to the real world. One such method is Domain Randomisation [5], which involves training RL agents in environments with randomised properties to improve their generalisation capabilities. This technique aims to make agents robust to environmental variations they might encounter in the real world. Additionally, Replay Across Experiments [6] is a method where the agent's training process incorporates data from multiple past experiments, improving data efficiency and helping the agent learn from a broader set of experiences. Moreover, NeRF Rendering [7] is another approach used to create realistic static scenes. During training, dynamic objects are rendered over these static backgrounds, introducing variability in the visual input. Furthermore, Zero-shot transfer (ZST) [8] refers to the ability of a model or agent to apply knowledge learned in one domain (typically a simulated environment) to a different domain (the real world) without any additional training or fine-tuning. ZST is achieved through a combination of sophisticated simulation techniques, realistic rendering, extensive domain randomisation, and robust training methodologies. These techniques enable the robotic agents to perform well in real-world scenarios despite being trained exclusively in a simulated environment [9].

In this paper, we aim to build an attention model to enable a humanoid robot to identify salient points in the real world. To achieve this, we will employ *ZST* and combine it with *domain randomisation* and *generalisation*, as we believe the combination of these techniques results in agents that can perform robustly in real-world scenarios immediately after

[1] K. Ghamati*, A. Zaraki, and F. Amirabdollahian are with the School of Physics, Engineering and Computer Science (SPECS), and Robotics Research Group of the University of Hertfordshire, Hatfield, AL10 9AB, United Kingdom. [k.ghamati@herts.ac.uk, https://ghamati.com/rlbam]

being trained in a simulated environment. The agents demonstrate agility and effective performance in interacting with the environment, akin to their simulated training, showcasing successful *ZST*. We implemented the developed method on a humanoid robot called ARI and tested it in real-world scenarios, demonstrating that the ARI robot successfully learned and displayed appropriate behaviour towards its interaction partners.

## II. Related Works

RL is extensively utilised in robotics due to its remarkable success in enabling robots to perform effectively in both real-world and simulated environments. This machine-learning technique allows robots to autonomously learn from their interactions with the environment. Recent studies have demonstrated various applications of RL in robotics, such as social intelligence in robots, assistive robots, robot control, and human-robot interaction [10]–[13].

In [14], a framework is proposed to utilise social-human behaviour to enhance HRI by employing interactive behaviours to ensure safety through bi-directional information transfer between humans and robots. This framework uses RL to help robots determine safe and unsafe actions to guarantee safety in shared environments. It states that it is necessary to ensure the robot learns robust behaviours despite potential human errors or adversarial actions and improve data efficiency to make effective use of interactive behaviours without requiring excessive real-world data. Moreover, adaptability will enhance the model to handle a variety of environments, including unstable, uncertain, and unknown scenarios. This study combines RL with human interactive behaviour to build a robust model because most of the introduced methods are compatible with simulation, not the real world, which is a main challenge in the field of RL and robotics.

In the context of assistive robots, [15] focuses on robot-assisted pedestrian regulation through the optimisation of robot motion planning to influence pedestrian flow. The study involves modelling and utilising the interactions between humans and robots to achieve desired outcomes in densely populated areas. It uses DRL to optimise the robot's behaviour in regulating pedestrian movements, thereby enhancing both safety and efficiency in crowd management scenarios. This research notes that accurately modelling the complex dynamics of human motion under the effect of HRI is challenging. Additionally, the robot must adapt its behaviour in response to changing pedestrian inflows and other dynamic environmental factors to optimise pedestrian outflow consistently. Indeed, robustness and adaptability are other challenges, and developing an accurate observation transition model for pedestrians influenced by HRI is difficult due to the variability and unpredictability of human behaviour. In this case, RL allows the robot to solve this challenge, but ensuring that the models and policies developed in simulations perform well in real-world scenarios involves overcoming differences between simulated and actual environments, which is a new challenge that comes from RL.

In the context of robot control, developing reliable walking controllers for bipedal robots is notably challenging. Adapting robots to dynamic environments is crucial and difficult. Also, a robust model to improve the ability of robots to handle unexpected disturbances and uncertainties in their environment ensures stable and reliable movement. Therefore, finding innovative ways to teach robots to walk safely and accurately in unseen environments is essential. Combining RL with domain randomisation has been proposed to address this issue. Still, Sim-to-Real is another RL challenge in robotics. Thus, in [16], by using domain randomisation, it effectively transfers the policies learned in simulation to the real world for overcoming the reality gap. Research [17] developed a method for automatic curriculum learning tailored to the capabilities of reconfigurable robots, particularly in challenging environments like search and rescue missions where the human and robot share the environment. This study introduced challenges like dynamic and adaptive learning, which express that the learning framework must adapt to changing environments and obstacles. The paper overcomes the challenges of using curriculum to improve learning efficiency in DRL by systematically organising learning tasks based on their complexity and the robot's performance.

To control the robot manipulator, [18] employs the Actor-Critic (AC) approach to utilise the benefits of two different RL training methods: direct and indirect learning. This empowers the robot to learn and adapt to the object during task operation, enhancing stability.

Despite the capabilities and successes of RL in robotics, the literature mentioned above highlights several challenges in integrating RL with robotics. These challenges include *robot adaptation*, *developing robust models*, and *transferring trained agents* from simulation to the real world. In fact, in some projects, running the robot or using online interactions to train an RL agent is not feasible. In this case, using simulation and using synthetic or collected data can help to train the model, but providing various situations that the robot will encounter in the real world is not possible. The replay across Experiments [6] is a method that allows the RL agent to reuse data under the off-policy method to build a robust RL agent. This method aims to improve RL performance by using experience data from various experiments. This approach is particularly beneficial in complex tasks where data collection is expensive or time-consuming. Regarding transferring the agent from the simulation to the real world, [19] presents a structured approach to transfer learning in RL. By categorising transfer methods and analysing their benefits and trade-offs, the paper addresses the challenge of improving generalisation and efficiency in RL, promoting more robust and adaptable learning systems. Regarding providing all situations that the agent will face, in [7] authors employ Neural Radiance Fields to generate a photorealistic 3D model of the environment, integrate it with a physics simulator, and train a humanoid robot for navigation and object interaction tasks. The approach demonstrates successful ZST to actual robots, maintaining high fidelity and performance.

In this work, we integrate an attention model with RL

to identify the saliency point as the initial step towards the robot adaptation. Additionally, we train our model using *Proximal Policy Optimisation (PPO)*, which uses *generalisation* to approximate a function through a gradient-based policy method. Additionally, we employ *randomisation* to add complexity during training, empowering the agent to cover a wide range of real-world circumstances. Finally, we transfer the trained agent from the simulation to the real world (sim2real) using *ZST*.

## III. Materials and Methods

In this section, we explain our model and methodology to illustrate how we designed and transferred an RL agent from sim2real. As outlined in previous sections, we opted for *ZST* to transition the model and to encompass the majority of situations the robot might encounter. To achieve this, we utilise *Domain Randomisation*. This method enables us to create a variety of states reflecting different circumstances, preparing the RL agent with the necessary knowledge during training for real-world operation. We employ *PPO*, which is one of the *AC* algorithms, to train the model due to its performance in training and *ZST*, which facilitates the transfer of our trained RL agent from simulation to the real world. Figure 2 depicts an abstract view of the combination of these methods. In the following subsections, we will explain this combination in detail, demonstrating how we use them to enable a humanoid robot to mimic human behaviour using an attention model.

### A. Generalisation and Randomisation

As discussed in the previous section, a key challenge in deploying a trained RL agent in the real world is bridging the gap between the training environment and the dynamic, open-ended nature of the real world. It is crucial to enable the agent to function normally when it encounters unseen states. To achieve this, we must build a robust model during the training phase, ensuring it can handle variations and differences in the new states encountered in reality. This robustness comes from the policy's ability to generalise from training experiences to new, unforeseen situations. This is where Randomisation and Generalisation are crucial, as they apply various predictable scenarios to known states to enhance the agent's behaviour and build a robust policy during training.

Aligned with our goal of developing an attention model using RL, the RL agent must identify saliency points based on environmental factors such as human activities, their distances from the robot, and their numbers. Randomisation involves varying different aspects of the training environment during the training phase, while generalisation involves training an RL agent to apply the knowledge it has gained to new and unseen environments. We employ these two approaches to create more complex states and build a robust model.

We generate new states by permuting each participant's activity and proximity. For instance, with two participants based on their observed activities, if participant 1 performs an activity like hand waving, we add a new state for participant 2 using this activity, thereby creating a set of users' activities to build new states for other users. We also consider scenarios where two participants are within the robot's field of view, performing different activities simultaneously. Another parameter we use to create new states is proximity, as this affects saliency detection according to the attention model.

Figure 2 illustrates how we generate new states from an observed state. Suppose two participants are present, and the agent has observed two activities: speaking and entering (entering the robot's field of view, such as a human coming into the room). Initially, participant 1 enters the robot's field of view, like a human entering a classroom after the course has started, and is in the personal space in terms of proximity. The black line represents this observed state. Based on this state and the activity of entering, we generate new states for the two observed participants by permuting proximity, activities, and participants. The purple line represents one of these states, where for both participants, we create two new states using the observed activity (entering) but in different proximity spaces (intimate space). The pink lines connect these two participants to the intimate space to show that the next states will be generated by previously observed activities (such as speaking) from the activity space. In essence, we create combinations involving permutations of their activities and proximity, with the pink line displaying one of these states. The orange lines represent different states that include up to six people, with one proximity and activity as a sample, demonstrating how we add complexity to the environment. Even though the agent initially observed only two participants, we generated various states by considering six people engaging in different activities simultaneously within the robot's field of view. In other words, we consider up to six people, where four of them do not exist, but states are generated for all of them like the two participants. We combine all these states to include scenarios involving one to six people. The idea of creating these states stems from the generalisation method, adding complexity to build a robust model. In the next subsection, we will discuss the RL agent.

### B. RL and Proximal Policy Optimisation (PPO)

To effectively employ RL, several essential elements are required, with the reward function being among the most critical. The reward function guides the agent in distinguishing between correct actions and mistakes. To achieve our objective, we have integrated a controlling gaze system designed to manage the robot's gaze in order to identify a salient point in the environment [20]. This system enables us to design our reward function to encourage the agent to locate individuals who are the centre of attention, considering factors such as activity and proximity. This system, named Gaze Control System (GCS), introduces a formula to model human behaviour for a humanoid robot. The formula, known as elicited attention (EA), utilises five variables to generate a score for each participant within the robot's field of view, with the participant having the highest score becoming the target as the salient point. Equation (1) demonstrates this formula, where, $F_{s,j}$ represents social features presented by

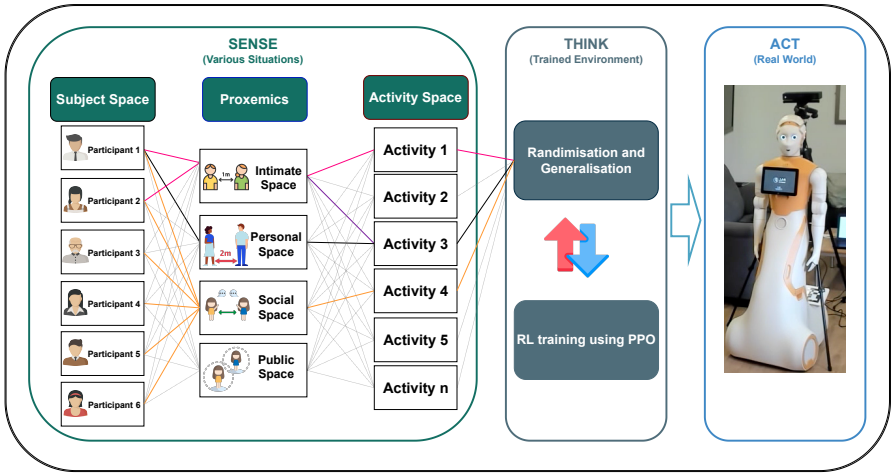


Fig. 2: The workflow for building the attention model using a robust RL agent - The Grey dotted lines indicate relationships between various items. The black line represents an observed state. The purple lines signify randomisation based on observed participants (1-2) and states. The orange lines demonstrate how new states are generated for unobserved participants (3-6) using observed data.

participant $s$ with the participant number $j$, $P(r)$ stands for the Proxemics area, $O(\theta)$ pertains to orientation (angle), $EAM_{s,j}$ denotes the memory component, $r$ is distance/proximity, $\theta$ is orientation angle with respect to the current robot's gaze, and $t$ is time.

$$EA_{s,j}(t) = F_{s,j} + P(r) + O(\theta) + EAM_{s,j} \quad (1)$$

The paper introducing this equation also provides a list of activities prioritised based on human behaviours, asserting that these actions have different values for humans. TABLE I lists these activities and their values. Additionally, it states that the distance between people and the robot affects attention absorption, necessitating different weights for various degrees of proximity to focus more on closer people. Moreover, the equation considers the orientation of participants to the robot as a significant parameter and incorporates a memory component. However, we decided to exclude these two parameters. Based on this system, we designed our reward function. Specifically, a score is produced for each observed user in the environment based on their activity in each state. If the agent decides to look at that person, it can achieve this score. Therefore, since the participant at the salient point has the highest score, the agent will receive the maximum reward by looking at that participant. Over time, the agent will learn to identify the participant performing a higher-priority activity and who is closest to the robot.

The agent should select an action from the action space, which includes:

- *Gaze_At_Environment*
- *Gaze_At_Object*
- *Gaze_At_Participant_1*
- *Gaze_At_Participant_2*
- *Gaze_At_Participant_3*
- *Gaze_At_Participant_4*
- *Gaze_At_Participant_5*
- *Gaze_At_Participant_6*

For training, we chose PPO, a popular RL algorithm known for its performance and stability. Various studies have combined this algorithm with the Randomisation method to train robust RL agents [21]–[23]. Thus, we are confident that it is a suitable choice. In general, two approaches to maximising cumulative rewards in RL are value-function methods and policy gradient methods, both aiming to find an optimal policy. However, they differ in their methodologies. Value-function methods optimise the policy by first estimating value functions and subsequently deriving the policy from these estimates. This process involves two main steps: value function estimation and policy derivation. On the other hand, policy gradient methods directly optimise the policy by adjusting its parameters to maximise the expected cumulative reward. This approach does not explicitly estimate value functions but focuses instead on finding the optimal policy directly. Indirect methods (value functions) are simpler to understand and implement for discrete state and action spaces but struggle with high-dimensional or continuous spaces. Conversely, direct methods (policy gradient) are more complex, suitable for high-dimensional and continuous action spaces, and often require sophisticated optimisation techniques. AC is an approach that combines the benefits of both indirect and direct methods, leading the agent to learn more stable and efficient policies. PPO is derived from the AC framework in which the actor selects actions according to a policy $\pi(a \mid s; \theta)$, where $\theta$ are the policy parameters that define how the policy is implemented, $s$ denotes the state, and $a$ denotes the action. The actor directly maps states to actions and is optimised to maximise the expected cumulative reward. Meanwhile, the critic evaluates the actor's actions by estimating the value function. There are two approaches for computing the value function. The first is the state-

value function $V(s; w)$, where $w$ represents the parameters of the function approximator used to estimate $V(s)$. The second approach is the action-value function $Q(s, a; w)$, which estimates the expected return starting from state $s$ by taking action $a$ and following the current policy thereafter. The critic, parameterised by $w$, provides feedback to the actor by estimating the Temporal Difference (TD) error. This error indicates how much better or worse the action taken was compared to the expected outcome.

The goal of the AC method is to optimise the policy to maximise the expected cumulative reward while using the critic to provide more accurate estimates of the policy's value, thereby reducing variance and improving learning stability. Therefore, the policy gradient is computed using the equation (2):

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\pi_\theta} \left[ \nabla_\theta \log \pi_\theta(a \mid s) \hat{A}(s, a) \right] \tag{2}$$

where $J(\pi_\theta)$ represents the objective function that the policy $\pi_\theta$ aims to maximise, typically the expected cumulative reward over time. $\pi_\theta(a \mid s)$ denotes the parameterised policy, specifying the probability of selecting action $a$ in the state $s$ given parameters $\theta$.

The advantage function $\hat{A}(s, a)$, used to assess the relative advantage of taking action $a$ in state $s$, is often approximated by the TD error $\delta_t$, defined as:

$$\delta_t = r_t + \gamma V(s_{t+1}; w) - V(s_t; w) \tag{3}$$

Here:

- $\delta_t$ is the TD error at time $t$.
- $r_t$ is the reward received at time $t$.
- $\gamma$ is the discount factor.
- $V(s_{t+1}; w)$ is the estimated value of the next state $s_{t+1}$ with parameters $w$.
- $V(s_t; w)$ is the estimated value of the current state $s_t$ with parameters $w$.

The advantage function helps provide insight into the training behaviour of the agent. A negative advantage suggests that the action taken was worse than the average action for that state, indicating exploitation of known actions. A positive advantage indicates that the action taken was better than average, suggesting exploration of potentially better actions. An ideal value of zero for the advantage function implies that the agent has converged and learned the best action for each state.

PPO, derived from the policy gradient framework, aims to maximise the objective function $J(\pi_\theta)$ using the policy gradient $\nabla_\theta J(\pi_\theta)$. However, PPO introduces the Clipped Surrogate Objective to prevent large policy updates that could destabilise learning. This objective is represented by the parameter in Equation (4):

$$\hat{L}(\theta) = \mathbb{E}_t \left[ \min(\rho_t(\theta) A_t, \text{clip}(\rho_t(\theta), 1 - \epsilon, 1 + \epsilon) A_t) \right] \tag{4}$$

where:

- $\rho_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the importance sampling ratio at time step $t_1$.

TABLE I: The priority of human social attention adopted from [20].

| Priority | Social Cue | GCS Score |
|---|---|---|
| 1 (highest priority) | Entering | 100 |
| 2 | Speaking | 100 |
| 3 | Hand motion/body gesture | 65 |
| 4 | Leaving | 55 |
| 5 (lowest priority) | Facial expression | 45 |

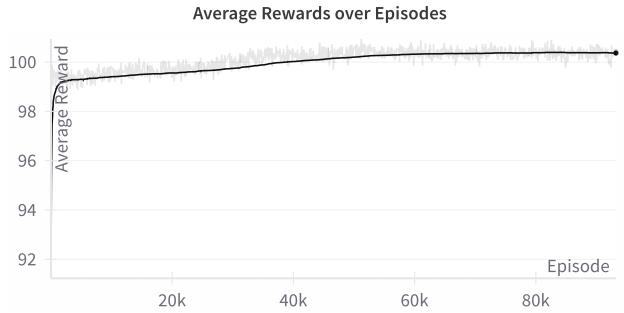


Fig. 3: This graph shows a consistent rise in the mean reward demonstrating the agent's learning and gradual performance.

- $A_t$ is the advantage function at time step $t$.
- $\epsilon$ is a small constant used in clipping to stabilize training.

PPO uses $\hat{L}(\theta)$, an approximation of $\nabla_\theta J(\pi_\theta)$, to guide policy updates, ensuring conservative improvements that maintain stability and incremental learning progress.

### C. Transferring

After training the agent, it is time to transfer it to the real world and integrate it with the robot. We have selected the ARI robot to run our model. However, before proceeding, we must ensure the model's robustness. To achieve this, we employ randomisation by creating scenarios involving two to six people in a scene. Each person is assigned a random activity, performed simultaneously, and based on their distance to the robot and the activity, the agent selects an action. We then evaluate each user's EA score and compare it to the agent's decision to verify its accuracy.

To assess the agent's performance, we monitored its behaviour during training. As depicted in Figure 3, the agent's performance converges after approximately 60000 episodes. Additionally, the average reward becomes more stable over time, with reduced variance. The plot shows a relatively smooth increase in stability, suggesting that the learning process was effective. Furthermore, the average reward remains high and consistent across various episodes, if the agent is robust.

Subsequently, we deployed the agent into the robot and tested it using the ARI robot. As anticipated, the agent performed effectively and was able to identify and focus on the participant with the highest EA score in all states.

### D. The Development of Robot Learning Architecture

Our model's architecture comprises two primary pipelines: simulation training and real-world applications. This section elaborates on these two pipelines in detail.

*1) Simulation:* In the development and training of our model, we establish an environment and train the agent using the Gym[1] and RayRL[2] libraries. We customise this environment to initially include a single human participant. By observing the participant's activities and proximity, we employ randomisation to generate new states that encompass various circumstances. Subsequently, we introduce a second participant, utilising both participants and the principles of randomisation to expand the state space. This approach aims to cover a wide range of scenarios that the agent may encounter in real-world settings, from single-person scenarios up to environments involving six individuals.

For training the agent, we utilise the PPO algorithm and tune its hyperparameters such as learning rate, gamma, number of epochs, and entropy coefficient using the RayRL library. RayRL seamlessly integrates with Gym, facilitating simultaneous tuning and training of the agent.

To design the reward function, we implement this functionality within the environment inherited from the Gym's main environment. This function retrieves information necessary for calculating rewards associated with each participant in every state. These rewards are computed when the agent selects an action and interacts with the environment.

*2) Real World:* Following the agent's training and the establishment of a robust model, we deploy the agent onto the ARI robot. To detect participants and their activities, we utilise the Kinect V2 sensor and its SDK. Each participant is assigned a unique identifier, and their keypoints and additional metadata [24] provided by the SDK are utilised. Using ZeroMQ[3](ZMQ), this data is transmitted to another workstation for processing and activity recognition.

Given our testing environment on the ARI robot, we utilise its SDK to control its gaze. Accordingly, we develop a ROS node to interface the trained agent with the ARI. This node receives information about each participant's activities and proximity, which are then relayed to the RL agent. Based on this input, the RL agent makes decisions to control the ARI's gaze, focusing on participants in salient points.

## IV. Proof of Concept Evaluation

To validate our model, given our use of EA in designing the reward function, we opted to benchmark our results against those reported in [20] by replicating the same experiment involving two participants.

### A. Experiment Scenario

Following the GCS framework [20] incorporating EA for scenario design, we assess how a robot responds to the two human participants engaging in various activities to determine saliency. The scenario is set so that participant 1 enters the robot's field of view, prompting the robot to recognise and process the activities. Using EA as a reward function, RL then controls the robot's gaze in response to the activities observed. Later, participant 2 joins, and both participants engage in different activities. Consequently, the robot directs its attention towards each participant, while identifying the saliency point.

### B. Results

Our testing involved complex scenarios where two human participants engaged in social interactions with the ARI robot and performed various activities such as initiating interaction, speaking, body gestures and hand motion, and terminating social interaction. All these social cues were presented to the robot in different proximities in the Robot House of the University of Hertfordshire. The primary goal was to test the developed RL model if it is able to drive the dynamic attention of the robot and direct it accurately to the right targets at the right time. The following items summarise the result according to the attention allocation and gaze direction developed model, which includes three targets: participant 1, participant 2, and the environment.

- **Attention Allocation:** The RL agent's performance was evaluated based on its ability to direct the robot's gaze across three primary targets: the environment, participant 1, and participant 2.
- **Environment:** The RL agent directed ARI's attention to look at different points in the environment 16 times when no participants were in view. This is expected behaviour, ensuring the robot remains engaged even in the absence of human activity.
- **Participant 1:** ARI focused on participant 1 (target 2) 173 times out of 220 states (78.6%). This high frequency indicates that participant 1 often had higher priority activities.
- **Participant 2:** ARI focused on participant 2 (target 3) 35 times (15.9%). This lower frequency suggests participant 2 had fewer priority activities compared to participant 1.

Figure 6 displays the distribution of ARI's focus across these targets, highlighting the RL agent's decision-making effectiveness. The RL agent's decisions can be considered effective if they align with the participants' activities and priorities. TABLE II summarises key states, illustrating the RL agent's decision-making process in challenging circumstances where two participants were engaged in various activities with different priorities. As displayed in this table, at time zero, with no participants in view, ARI follows low-level saliency cues, mimicking human-like scanning behaviour. At 59 seconds, upon participant 1 entering the view, the RL agent correctly directs ARI's gaze to the participant. At 1 minute and 6 seconds, participant 1 moves closer and begins speaking, prompting the RL agent to adjust ARI's focus accordingly. At 2 minutes and 43 seconds, with two participants in view, participant 2, having a higher EA score due to entering the view and proximity, becomes the focal point.

[1]https://gymnasium.farama.org

[2]https://docs.ray.io/en/latest/rllib/index.html

[3]https://zeromq.org

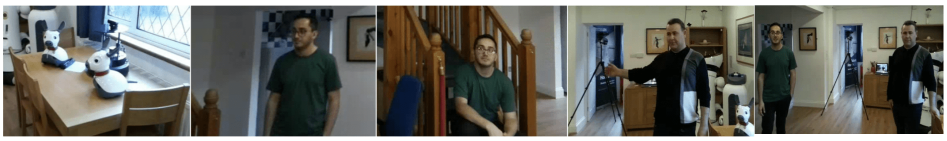


Fig. 4: Some of the targets identified by ARI's attention model include points A through E. The RL agent, which controls ARI's gaze, has determined these targets based on EA, participants' activities and Proxemics of the participants.

TABLE II: Results of the scenario showing five samples of RL-selected actions and ideal actions.

| Time | Ideal Action | Participant_1 EA score | Participant_2 EA score | RL Reward | RL Action | State |
|---|---|---|---|---|---|---|
| 00:00:00 | Gaze at environment | 0 | 0 | 0 | Low level saliency | No one is in the robot's field of view |
| 00:00:59 | Gaze at participant_1 | 75 | 0 | 75 | Gaze at participant_1 | Human participant_1 entered the robot's field of view and is in the personal space |
| 00:01:06 | Gaze at participant_1 | 200 | 0 | 200 | Gaze at participant_1 | Human participant_1 is inside the social space and is speaking |
| 00:02:43 | Gaze at participant_2 | 165 | 200 | 200 | Gaze at participant_2 | Human participant_1 is waving his hand and participant_2 enters the robot's field of view. Both of them are in the social space |
| 00:03:03 | Gaze at participant_1 | 200 | 165 | 200 | Gaze at participant_1 | Human participant_1 is speaking and participant_2 is waving his hand. Both of them are in the social space |

Finally, when participant 1 is speaking and participant 2 is waving, the RL agent prioritises participant 1, recognising speaking as a higher priority activity. These results show that the RL agent's attention distribution aligns well with expected behaviours, prioritising higher EA scores and relevant activities. The fact that ARI looked at participant 1 173 times (78.6%) versus participant 2 only 35 times (15.9%) during the experiment demonstrates the agent's ability to discern and prioritise based on the importance of the activities, indicating that the model works effectively. Figure 4 illustrates various states of the robot's view, showcasing ARI's gaze behaviour in different circumstances. Also, Figure 5 shows the scaled reward versus time, highlighting the points where the RL agent directed attention to different targets during the whole of the scenario.

Comparing the ARI's attention behaviour with the reference behaviour obtained from [20] shows that the developed model is able to identify the correct target points at the right times all the time without any error. The RL agent's performance, as shown through these observations, validates its effectiveness in dynamically directing the robot's attention in a way that mimics human-like prioritisation, making it a valuable approach.

## V. Discussion and Conclusion

In this work, we developed an attention model for social robots to identify saliency points in unknown environments. The proposed model utilises the PPO to train an RL agent in simulation and implements ZST to deploy the trained agent in real-world settings. The randomisation method has been used to cover the diverse situations an agent may encounter in the real world. This hybrid strategy effectively enabled the robot to identify salient points and smoothly direct ARI's gaze towards participants, exhibiting natural behaviour which encouraged the participants to accept the robot as an active participant in the environment. While developing this learning architecture for the robot we encountered various challenges, as discussed below:

- Training the agent requires careful tuning of PPO parameters such as entropy coefficient, value-function clip parameter, gamma, and learning rate to ensure thorough exploration and faster convergence, as convergence time is a significant issue.
- While randomisation helps address the variability of real-world data, it is insufficient to fully overcome sensor noise.

For future work, to address these challenges, we aim to enhance our model to learn dynamically by combining model-free and model-based methods. This approach will improve sensor fidelity and enable the agent to adapt to new scenarios in real-world environments.

## VI. Acknowledgements

The authors would like to acknowledge the support of the Robot House of the University of Hertfordshire in providing the facility to carry out this study.

## References

[1] C. Yang, G. Peng, Y. Li, R. Cui, L. Cheng, and Z. Li, "Neural networks enhanced adaptive admittance control of optimized robot–environment interaction," *IEEE transactions on cybernetics*, vol. 49, no. 7, pp. 2568–2579, 2018.

[2] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[3] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: a survey," in *2020 IEEE symposium series on computational intelligence (SSCI)*. IEEE, 2020, pp. 737–744.

[4] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," 2017.

[5] O. S. Ajani, S. ho Hur, and R. Mallipeddi, "Evaluating domain randomization in deep reinforcement learning locomotion tasks," *Mathematics*, 2023.

[6] D. Tirumala, T. Lampe, J. E. Chen, T. Haarnoja, S. Huang, G. Lever, B. Moran, T. Hertweck, L. Hasenclever, M. Riedmiller, N. Heess, and M. Wulfmeier, "Replay across experiments: A natural extension of off-policy rl," 2023.

[7] A. Byravan, J. Humplik, L. Hasenclever, A. Brussee, F. Nori, T. Haarnoja, B. Moran, S. Bohez, F. Sadeghi, B. Vujatovic, and N. Heess, "Nerf2real: Sim2real transfer of vision-guided bipedal motion skills using neural radiance fields," 2022.

[8] I. Higgins, A. Pal, A. Rusu, L. Matthey, C. Burgess, A. Pritzel, M. Botvinick, C. Blundell, and A. Lerchner, "Darla: Improving zero-shot transfer in reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1480–1490.

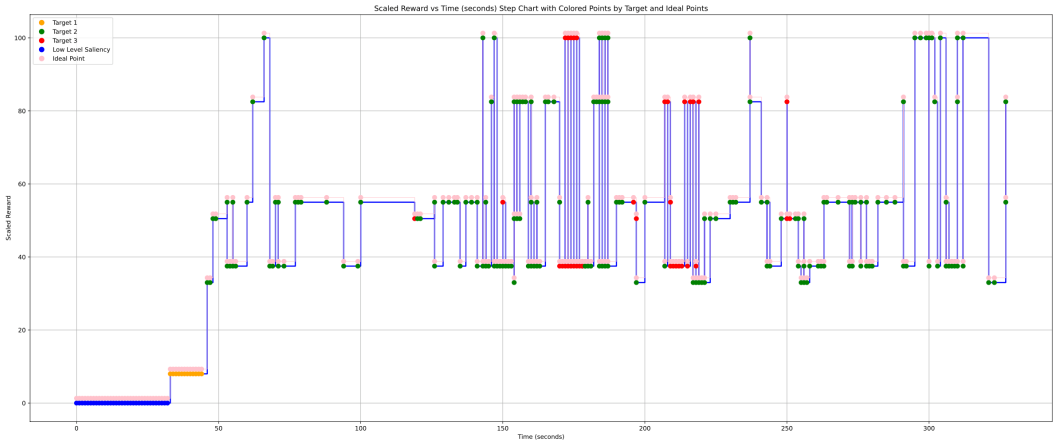


Fig. 5: Scaled Reward vs. Time (seconds). This step chart illustrates the RL agent's decisions over time, highlighting when ARI directed its attention to different targets.

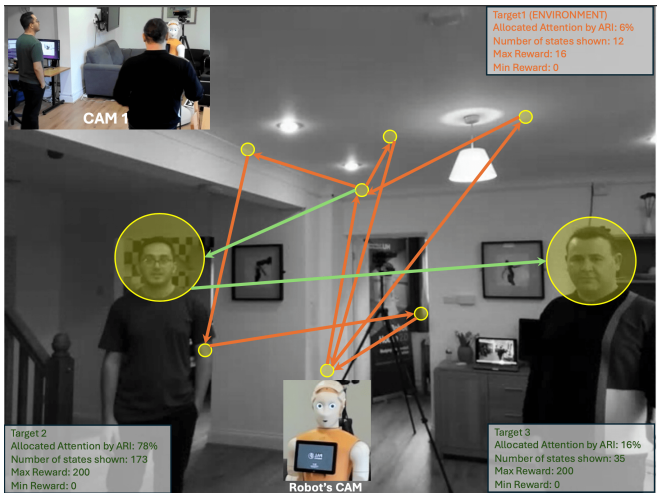


Fig. 6: The two camera views show how the attention of the ARI humanoid is dynamically allocated to the environment, participant 1, and participant 2 in a triadic HRI. The larger image shows ARI's head camera view, where the smaller circles represent points that ARI looked at when no participants were in the environment. Additionally, the larger circles indicate ARI's attention based on the participants' activities when they were in the environment.


[9] D. Tirumala, M. Wulfmeier, B. Moran, S. Huang, J. Humplik, G. Lever, T. Haarnoja, L. Hasenclever, A. Byravan, N. Batchelor, N. Sreendra, K. Patel, M. Gwira, F. Nori, M. Riedmiller, and N. Heess, "Learning robot soccer from egocentric vision with deep reinforcement learning," 2024.

[10] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[11] A. Sunilkumar, F. Bahrpeyma, and D. Reichelt, "An overview of the applications of reinforcement learning to robot programming: discussion on the literature and the potentials," 2024.

[12] A. Zaraki, M. Khamassi, L. J. Wood, G. Lakatos, C. Tzafestas, F. Amirabdollahian, B. Robins, and K. Dautenhahn, "A novel reinforcement-based paradigm for children to teach the humanoid kaspar robot," *International Journal of Social Robotics*, vol. 12, pp. 709–720, 2020.

[13] L. J. Wood, A. Zaraki, B. Robins, and K. Dautenhahn, "Developing kaspar: a humanoid robot for children with autism," *International Journal of Social Robotics*, vol. 13, no. 3, pp. 491–508, 2021.

[14] S. Gu, A. Kshirsagar, Y. Du, G. Chen, J. Peters, and A. Knoll, "A human-centered safe robot reinforcement learning framework with interactive behaviors," *Frontiers in Neurorobotics*, vol. 17, 2023.

[15] Z. Wan, C. Jiang, M. Fahad, Z. Ni, Y. Guo, and H. He, "Robot-assisted pedestrian regulation based on deep reinforcement learning," *IEEE Transactions on Cybernetics*, vol. 50, no. 4, pp. 1669–1682, 2020.

[16] Z. Li, X. Cheng, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for robust parameterized locomotion control of bipedal robots," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 2811–2817.

[17] Z. Karni, O. Simhon, D. Zarrouk, and S. Berman, "Automatic curriculum determination for deep reinforcement learning in reconfigurable robots," *IEEE Access*, vol. 12, pp. 78 342–78 353, 2024.

[18] H. R. Nohooji, A. Zaraki, and H. Voos, "Actor–critic learning based pid control for robotic manipulators," *Applied Soft Computing*, vol. 151, p. 111153, 2024.

[19] M. Wulfmeier, A. Byravan, S. Bechtle, K. Hausman, and N. Heess, "Foundations for transfer in reinforcement learning: A taxonomy of knowledge modalities," 2023.

[20] A. Zaraki, D. Mazzei, M. Giuliani, and D. De Rossi, "Designing and evaluating a social gaze-control system for a humanoid robot," *IEEE Transactions on Human-Machine Systems*, vol. 44, no. 2, pp. 157–168, 2014.

[21] S. N. Gowda, "Synthetic sample selection for generalized zero-shot learning," 2023.

[22] B. Mazoure, A. M. Ahmed, P. MacAlpine, R. D. Hjelm, and A. Kolobov, "Cross-trajectory representation learning for zero-shot generalization in rl," *ArXiv*, 2021.

[23] A. Guo, L. Song, and X. Chen, "Learning similar tasks based on ppo by transferring trajectory," in *2019 ICNSC*, 2019, pp. 126–131.

[24] A. Zaraki, M. Pieroni, D. De Rossi, D. Mazzei, R. Garofalo, L. Cominelli, and M. B. Dehkordi, "Design and evaluation of a unique social perception system for human–robot interaction," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, no. 4, pp. 341–355, 2016.