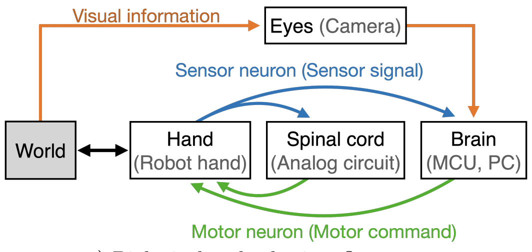
Bio-inspired Reflex System for Learning Visual Information for Resilient Robotic Manipulation

Kai Junge*[1], Kevin Qiu*[1], Josie Hughes[1]

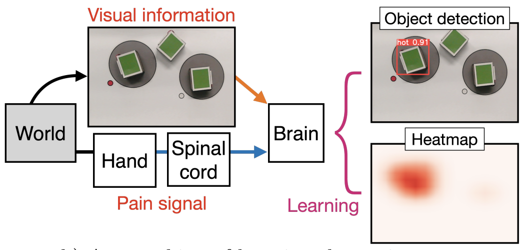
***Abstract*— Humans have an incredible sense of self-preservation that is both instilled, and also learned through experience. One system which contributes to this is the pain and reflex system which both minimizes damage through involuntary reflex actions and also serves as a means of 'negative reinforcement' to allow learning of poor actions or decision. Equipping robots with a reflex system and parallel learning architecture could help to prolong their useful life and allow for continued learning of safe actions. Focusing on a specific mock-up scenario of cubes on a 'stove' like setup, we investigate the hardware and learning approaches for a robotic manipulator to learn the presence of 'hot' objects and its contextual relationship to the environment. By creating a reflex arc using analog electronics that bypasses the 'brain' of the system we show an increase in the speed of release by at least two-fold. In parallel we have a learning procedure which combines visual information of the scene with this 'pain signal' to learn and predict when an object may be hot, utilizing an object detection neural network. Finally, we are able to extract the learned contextual information of the environment by introducing a method inspired by 'thought experiments' to generate heatmaps that indicate the probability of the environment being hot.**

I. INTRODUCTION

For robots to be ubiquitous and operate in complex and potentially dangerous human environments, one challenge that must be addressed is their life-span and resilience. Unlike humans who can live for prolonged periods of time, it is challenging to develop robots that can 'survive' for even a fraction of that amount [1]. This may be due to hardware degradation, but also resulting from poor actions or decisions made by a robot that lead to damage [2]. To address this concern, we must find both hardware and learning based solutions by which robots can choose to make actions or decisions which are not self-harming throughout their lifespan [3]. One biological example of self-preservation is pain detection and the corresponding reflex arc. Animals detect pain through specific receptors (nocireceptors) which initiate a reflex action, an involuntary withdrawal reflex. This decision bypasses the brain and is performed by the spinal cord allowing for a rapid and unconscious withdrawal. In addition to reducing damage, this pain signal is also motivational for both short term and long term harm-avoiding behaviors [4]. The concept of pain and the way by which the brain uses this information to inform future behavior is still being investigated by neuroscientists [5], however it is clear that this signal contributions to learning. A number of architectures have been proposed to explain the learning



a) Biological and robotic reflex system



b) Approaching of learning the environment

Fig. 1. a) Biological interaction to an external signal from the world and their robotic counterparts (in brackets). b) Overview of the learning approach where visual information about hot objects is learned to allow for prediction of hot objects and generation of heatmaps.

process, including reinforcement learning [6]. In addition to equipping robots with improved self-preservation, developing robots with nocireceptors, reflexes and learning could allow different hypothesis to be validated.

The implementation of bio-inspired reflex actions has been demonstrated previously in a number of robotic systems. In the 1980s, an artificial reflex arc was implemented for a knee structure [7]. More recent approaches have shown a decentralized spinal reflex which is achieved using purely mechanical devices for a brainless walking robot [8], and utilizing of reflexes for the control of a quadruped [9]. Leveraging exciting advances in sensor and material technologies, a decentralized robotic sensory stem has been demonstrated that uses self heal-able neuromorphic memtransistor elements to both detect and heal after damage [10]. Reflex-like systems have also been shown for a number of applications including collision avoidance [11], human assistive systems [12], and safe human-robot interaction [13]. These existing technological advances and applications demonstrate the potential for a physical reflex response. The next open challenge is how we bring the brain into the loop. How can we have this reflex system running in parallel to a 'brain' like system, which learns from visual information to make decision or actions which minimize the likelihood of pain? Many learning processes inherently rely on 'negative reward' signals, and are thus well formulated for integration with

*These authors contributed equally to this work [1]CREATE Lab, EPFL, Lausanne, Switzerland. Contact emails: kai.junge, longlai.qiu, josie.hughes, @epfl.ch.

'pain-like' negative reinforcement signals [14]. However, from this feedback the robot must learn to understand how the environment led to the negative reinforcement signal to minimize future 'painful' actions. Thus, the goal of this work is to develop a combined body-brain approach to enable a robotic manipulator to minimize damage when interacting with hot objects. This is achieved by using both an involuntary reflex action and also the continued learning of objects' and the environment's thermal properties from visual information.

Starting from biological inspiration we implement a reflex-inspired system using analog electronics such that the response is entirely involuntary and overrides any decisions from the 'brain' or microcontroller. This detected 'pain' is also used to train a neural network to learn a mapping from visual information from the object and surrounding area, to obtain the probability of objects to be hot. We focus on the specific scenario of a stove top with hot plates, where the goal of the robot is to clean or remove objects from it safely. A learning pipeline (Fig. 1b) has been developed that seeks to identify the probability of an object being hot given contextual environmental information (e.g. hot plate, and on/off indicator) by iteratively training a neural network classifier. Using this trained neural network we can then augment scene images with 'virtual' objects and estimate the probability of the virtual object being hot. By running many of these 'thought-experiments' across the space of the image, we can build up a heatmap that reflects the probability of a hot environment across the entire scene.

In the remainder of this paper we introduce the analog electronics and hardware used to achieve this reflex action and demonstrate the potential for a rapid action which bypasses any computation. Using this detection of pain, we show the results for an exploration and learning approach which allows prediction of hot objects and a resulting choice of object. We conclude with a discussion of this approach and further steps to generalize the approach.

II. PROBLEM STATEMENT

The specific scenario we focus on is that of a cooking stove. We create a stove setup that have 'visual clues' that represent a typical stove structure - round heating elements and also neighboring lights (stove indicators) that indicate if these specific heating elements are hot. On the stove we place objects: blocks with green markers for easy visual identification. When these are on top of the hotplates, they are set to be physically hot (Arbitrarily set to $> 30°C$ which is consistently hotter than the ambient temperature). The overall goal is create a robot that removes as many objects as possible, 'cleaning' the stove of objects, whilst minimizing damage to the manipulator by picking as few hot objects as possible. Furthermore, we want to be able to generalize our understanding of the environment such that we learn which areas of the scene are hot and cold such that for an unseen object type we can determine which of these objects the robot should not touch.

III. REFLEX INSPIRED HARDWARE & CONTROL

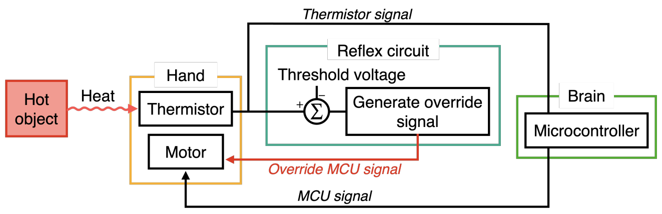
We propose a hardware and control architecture where the temperature sensor signal used as the nocireceptor output is passed to the brain (microcontroller) but can also trigger an analog electronics reflex system in parallel. On detecting high temperatures the output from this analog reflex can override the manipulator motor commands from the microcontroller. This reflex override triggers a high 'over-drive' voltage for a split second to quickly release objects. This occurs involuntarily without any decision from the 'brain'. In this section we detail the analog electronics and then the mechanical design of the manipulator.

A. Analog Reflex System

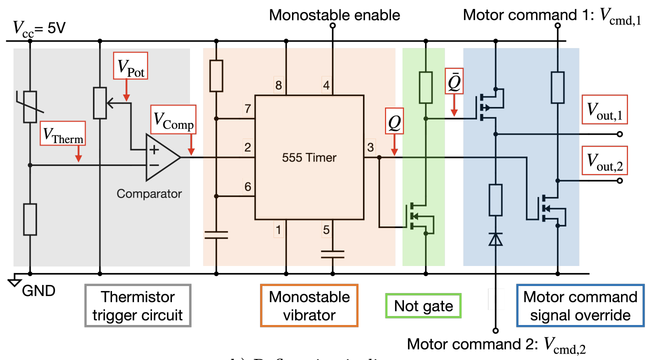
The analog reflex and microcontroller (Arduino Nano) are both connected to the thermistor which is used as the sensory receptor, and to the motor which actuates the gripper. When a temperature threshold is exceeded this triggers the analog reflex action which must override the motor control signal from the microcontroller to enable a higher voltage to be applied to the DC motors allowing for quick release. This requires analog electronics which can rapidly trigger the generation of a signal to switch transistor drive circuits in turn drive the motor.

The full implementation of this circuit is summarized in Fig. 2. The output from the thermistor potential divider is connected to a comparator with a reference voltage set by a potentiometer. When the thermistor voltage exceeds this reference voltage the comparator is triggered and generates a falling edge. This falling edge is connected to a monostable vibrator circuit formed from a 555 timer which latches the output signal Q and enables the motor control to be overridden until the latch is reset. The inverse of the output from the monostable ($\bar{Q}$) is produced using an FET-based not gate. These two signals ($Q, \bar{Q}$) are used to override the motor command signals initiating from the microcontroller. The DC motor controlling the gripper is driven by an H-bridge circuit through two voltage signals $V_{\text{out},1}$ and $V_{\text{out},2}$. In normal operation, the H-bridge is controlled via digital and pwm signals from the microcontroller, $V_{\text{out},1} = V_{\text{cmd},1}$ and $V_{\text{out},2} = V_{\text{cmd},2}$. When the reflex circuit is triggered, the two signals ($Q, \bar{Q}$) will override the microcontroller signals such that $V_{\text{out},1} = V_{\text{cc}}$ and $V_{\text{out},2} = \text{GND}$ to enable the maximum voltage from the power supply to be applied to the DC motor. Once the reflex action has released the object, the microcontroller then resets the monostable to regain normal operation.

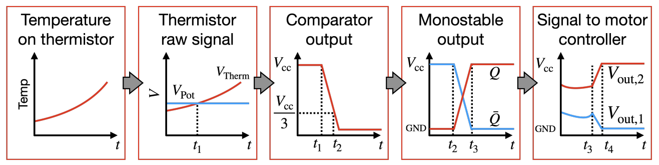
The circuit has a number of limitations in its performance. For example, with a faster microcontroller, the reaction speed can be increased to match that of the analog circuit, achiving a higher reaction time purely by the 'brain'. Furthermore, with the current circuit, the reaction speed is limited by the thermistor response, and the threshold temperature is fixed. While these limitations can be improved, we emphasize that this circuit demonstrates the fundamental characteristics of a robotic counterpart for the human reflex action: a sensory-motor control which bypasses and temporarily blocks signals



a) Reflex system high level diagram



b) Reflex circuit diagram



c) Reflex circuit functional flows

Fig. 2. a) The high level diagram of how the reflex circuit interacts with the 'body' (robot hand) and the 'brain' (microcontroller). b) The circuit diagram of the reflex circuit with sections of different functionality highlighted. c) Representation of how the signal propagates from measured temperature increase to the motor command.

for the brain enabling a resultant motion which has higher energy than normal operation.

B. Manipulator Design & Capabilities

A two finger parallel gripper has been designed for the grasping experiments (Fig. 3). The gripper uses a rack and pinion mechanism to convert a rotary motion to parallel motion. The motor (typically rated for 12V) can be overdriven at 30V for a short period of time which causes a much faster motion which imitates the rapid reflex response motion. The thermistor is mounted one on the surface of one of the gripper fingers. The gripper has the electronics mounted above (reflex circuit, driver electronics, microcontroller) and can be mounted on a robot arm (UR3 arm).

To illustrate the speed of response of this 'reflex action' in comparison to the action which is made when passing through the 'brain', we show the comparison of the release action for these two approaches in Fig. 4. When triggered and utilizing the robots 'reflex arc' the speed of response is approximately twice that of the computational method.

IV. Learning Methods

In this section we outline the methods used to learn potential causes of pain at both the object and environment level. To achieve this, we use an object detection network to identify objects that are hot. This problem is challenging for two

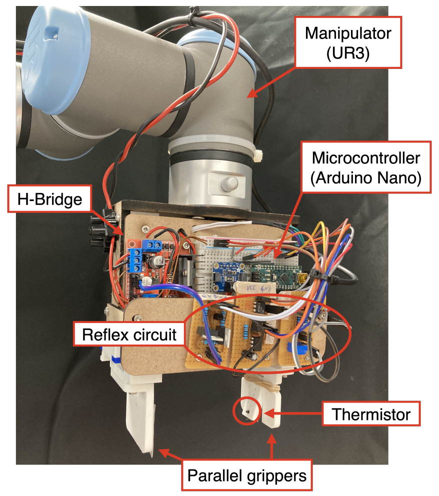


Fig. 3. The parallel finger gripper with thermistor on the finger with the electronics mounted about the mechanism.

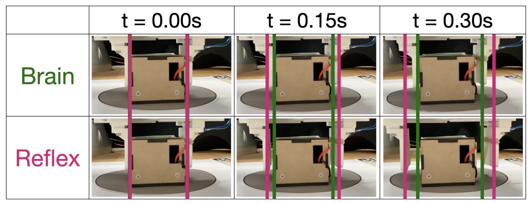


Fig. 4. Speed comparison showing the release of a block when activated via the brain or via the reflex action. The green and maroon lines indicate the position of the gripper fingers respectively.

reasons. Firstly, the generation of training data comes from the robot exploring the environment. As a result, ground truth labels are only obtained when the robot encounters a painful stimulus such as that when in contact with a hot object. Additionally, such information can only be gained for a small part of the scene. Secondly and resulting from this, the robot must rapidly learn from a limited dataset to minimize the number of painful experiences.

To learn both an object and environmental level understanding of hot areas we introduce two methods: an iterative learning method inspired by reinforcement learning, and a 'thought-experiment' like method to extract the robot's belief of the environment status using the trained neural network.

A. Iterative Learning Process

We present an iterative learning process using Ultralytics' `YOLOv5` [15] architecture to identify *hot objects* in the scene and probability of them being hot. We first present the overview iterative process before detailing the specific training of `YOLOv5`.

The iterative learning process, described through Algorithm 1, is inspired by Q-learning [16]. A action-reward matrix $\mathbf{Q} = \begin{bmatrix} q_{\text{pick,hot}} & q_{\text{pick,cold}} \\ q_{\bar{\text{pick}},\text{hot}} & q_{\bar{\text{pick}},\text{cold}} \end{bmatrix}$ is first initialized with null values which store the immediate reward and penalty values based on the stimulus that the robot comes into contact with (N.B.: $q_{\bar{\text{pick}},\text{hot}}, q_{\bar{\text{pick}},\text{cold}}$ will never be updated since the robot needs to come in contact with the objects to obtain rewards). The values of the true rewards are set to $r_{\text{pick,hot}} = -2$ and

$r_{\text{pick,cold}} = 1$. The action-reward matrix is applied *per object* when calculating optimal action.

When presented with a new stove configuration, we first capture an image I and detect the necessary objects and its grasping points. This is done by segmenting the HSV color space of the image and detecting the green box contours of the objects. Note at this point, the objects are detected, but the robot has not yet inferred the belief of each objects' temperature state.

The probability of the object being hot $p = P(\text{hot})$ is then obtained by applying the `YOLOv5` neural network to image I. The output is a list of bounding boxes and its classification confidence. If an object is not classified with a bounding box, we let $p = 0$. Initially before the neural network is trained, every object is assumed to have $p = 0$.

By matching the location of the bounding boxes and the object location detected using the green rectangles, we obtain a list of all objects and their $P(\text{hot})$ values.

The action space is defined per object and is $a = 0$ or $a = 1$ corresponding to the robot not picking or picking the box respectively. To compute the optimal action, we compute the predicted reward for the actions given for this stove configuration by (1).

$$R_{\text{pred}}(\mathbf{a}, \mathbf{p}, \mathbf{Q}) = \sum_i r(a_i, p_i, \mathbf{Q})$$
$$r(a, p, \mathbf{Q}) = \begin{cases} p q_{\text{pick,hot}} + (1-p) q_{\text{pick,cold}}, & a = 1 \\ 0, & a = 0 \end{cases} \tag{1}$$

The optimal action for each object detected is found by maximizing the reward as in (2).

$$\mathbf{a}^* = \operatorname*{argmax}_{\mathbf{a}} R_{\text{pred}}(\mathbf{a}, \mathbf{p}, \mathbf{Q}) \tag{2}$$

The true action applied to object i is determined through the epsilon-greedy algorithm where the action a_i is chosen to be either the optimal action a_i^* or a random action.

$$a_i = \begin{cases} a_i^*, & \text{Probability: } 1 - \epsilon \\ \text{Random action}, & \text{Probability: } \epsilon \end{cases} \tag{3}$$

The value of ϵ is calculated by (4), where n is the number of training iterations and γ is a parameter set to 0.3.

$$\epsilon = \exp(-\gamma n) \tag{4}$$

When an action for each object is chosen, the robot will execute the action while obtaining ground truth labels of each object that it interacted with. After the robot executes all its actions for a stove configuration, and if it has encountered a hot object, it will train `YOLOv5` upon its existing network weights before moving on to a new stove configuration.

To train the neural network, we begin using the `YOLOv5s` pre-trained model with the default hyperparameters and initialize the batch size and epochs to be 10 and 100 respectively. After each training iteration, the highest performing weights are stored to be used both for testing and as the initial set of weights for the subsequent training iteration.

TABLE I
DATA AUGMENTATION PARAMETERS

Transformation	Amount	Probability
Rotate	7°	0.7
RandomScale	0.05	0.5
Flip	-	0.9

We use an expanding dataset, where the full dataset is used for training the network at every iteration. For every stove configuration, if one or more hot object was encountered, we use the image of that stove and the corresponding labels and bounding boxes for training. We enlarge the bounding boxes on the detected green rectangle using a constant scaling factor of 1.4 in order to gather some amount of contextual information from the box's background. To increase the amount of training data, we apply data augmentation to generate 14 images per each observed scene. Specifically, we apply a series of transformations as listed in Table I. The probability associated with each transformation is the likelihood of the transformation occurring on said image. For each augmented set, we randomly split between the train and validation images under a 10:4 ratio.

B. Environmental Generalization: 'Thought Experiments'

We introduce a 'thought experiment' as part of the robot's ability to accumulate knowledge and apply its learning to different environmental settings. The neural network is trained to detect box-like objects that fall on hot areas. Although the direct output from this network will only provide temperature information in the presence of blocks, the network has learned some information regarding visual features of hot areas of the environment around the object. An additional step is required to extract this information from the network to enable a 'heatmap' of the robot's belief of the probability of an area being hot over a given image of the scene to be generated.

For a particular image, provided the learning was successful, the neural network will predict whether the objects used for training is hot or not based on where it is placed. Since the test objects are all practically identical, we can conclude the network has learned some visual relationship between object's temperature and the environment it has been placed on. Equally, the test objects can be considered to be points in space where the robot samples whether the environment is hot or not. To leverage this fact, we can place synthetic objects to the captured image in various locations, to conduct a 'thought experiment': "if we place an object in a particular location, then would the object be hot?". As shown in Fig. 5, a series of images with synthetic objects placed in various locations can be generated. For each image, the neural network returns bounding boxes and confidence (or the lack of). This information can be gathered into a matrix collating p at each point in space, resulting in a heatmap shown on the right hand side of Fig. 5.

To implement this method we extract a single image cutout of the object from one scene which can be used to augment

Algorithm 1 Iterative learning process

```
1:  Initialize $n \leftarrow 0$                              ▷ $n$: iteration count
2:  Initialize $\mathbf{Q} \leftarrow$ empty                 ▷ $\mathbf{Q}$: immediate reward table
3:  Initialize $\mathcal{D} \leftarrow$ empty                ▷ $\mathcal{D}$: dataset for training
4:  Initialize $\mathbf{w} \leftarrow$ YOLOv5s               ▷ $\mathbf{w}$: NN weights
5:  Initialize $\gamma$                                      ▷ $\gamma$: decay rate
6:  $I \leftarrow$ camera image
7:  Initialize $l_{\text{hot}} \leftarrow$ empty             ▷ $l_{\text{hot}}$: hot labels
8:  Detect objects and their grasping points in $I$
9:  Obtain $p_i$ for every object $i$. $\mathbf{p} = [p_0, p_1, ..]$ using
    YOLOv5
10: Determine $\mathbf{a}^*$                                 ▷ See (2)
11: $\epsilon = \exp(-\gamma n)$
12: for object in $I$ do
13:     Determine $a_i$ using equation 3
14:     if $a_i$ = pick then
15:         Robot picks up block
16:         $l_i \leftarrow$ ground truth label obtained from robot
17:         if $l_i$ = hot then
18:             Robot releases the block due to reflex action
19:             $q_{\text{pick,hot}} \leftarrow r_{\text{pick,hot}}$
20:             Append $l_i$ to $l_{\text{hot}}$
21:         else
22:             Remove block from workspace
23:             $q_{\text{pick,cold}} \leftarrow r_{\text{pick,cold}}$
24:         end if
25:     end if
26: end for
27: if $l_{\text{hot}}$ is not empty then
28:     Apply data augmentation on $I$ using $l_{\text{hot}}$
29:     Append augmented data to $\mathcal{D}$
30:     Train NN on $\mathcal{D}$ using $\mathbf{w}$
31:     $\mathbf{w} \leftarrow \mathbf{w}$ from 30               ▷ update weights from training
32:     $n \leftarrow n + 1$                                 ▷ increment iteration count
33: end if
34: Move on to new stove and object configuration
35: go to 6                            ▷ restart process by taking a new image
```

other scenes. This cutout can be overlaid at any x,y co-ordinate and a new image generated. We create a series of 225 augmented images per scene to generate a heat map which are spaced in a grid like manner (15x15) over the image. From this a heat map can be generated. The higher the number of augmented images the greater the resolution of the generated heatmap, but the longer the computational time. After generating the augmented images and running the trained neural network across each one, we can average the probability for each grid across the scene to generate the heat map. Finally Gaussian smoothing is then used to obtain a smoother more continuous representation of the heatmap.

V. Experimental Setup

The specific scenario we focus on is that of a cooking stove. We create a stove that has 'visual clues' that represent a typical stove structure - round heating elements and also neighboring lights (stove indicators) that indicate if these

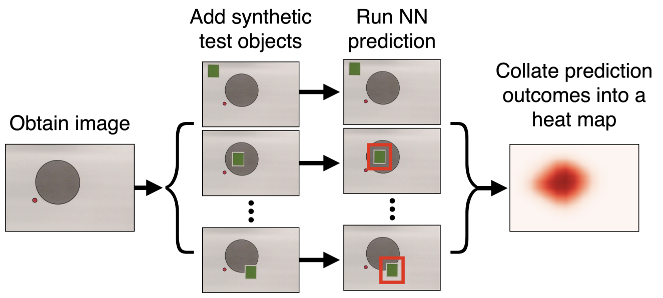

Obtain image
Add synthetic test objects
Run NN prediction
Collate prediction outcomes into a heat map

Fig. 5. Illustration of the 'thought-experiment' concept, showing how an evaluation of synthetic test points across a scene is used to build up a heatmap.

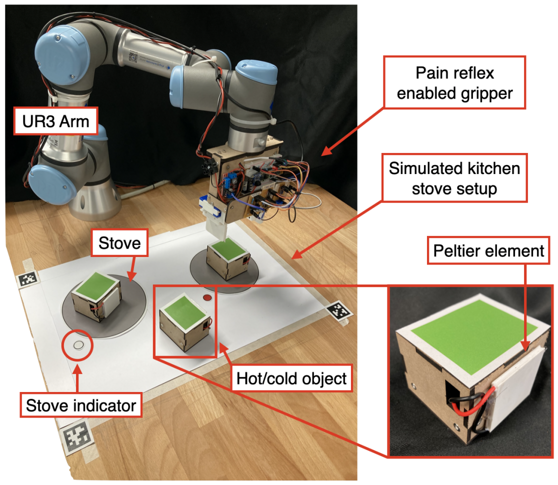


Fig. 6. Experiment setup showing the custom manipulator, UR3 robot arm and the stove setup. A red stove indicator means the stove is ON, and thus any object on it is hot.

specific heating elements are turned on (a red indicator means the stove is on). We create this representative setup within the workspace of a robot UR3 arm on which the manipulator equipped with the reflex circuit is mounted. The objects we consider are blocks with square green markers on top to facilitate object detection. To simulate these blocks conducting the heat from the stove, each have Peltier elements on the side which are connected to batteries within. This setup allows us to rapidly create different stove configurations by placing the round hot plate areas, the indicators, and objects in any possible configuration.

Above the simulated stove, a webcam is placed to capture the scene. The recorded image of the stove is used for training/inference using the neural network, and to obtain grasping points of the blocks automatically. A simple calibration system utilizing April Tags at the corners of the workspace of the robot has been used to allow robot co-ordinates to be converted to real world co-ordinates.

The manipulator is controlled via serial communication from an external computer which runs the learning and action selection algorithms.

VI. Results

A. Iterative Learning

To test the iterative learning process we perform an experiment where the exploration and learning algorithm is

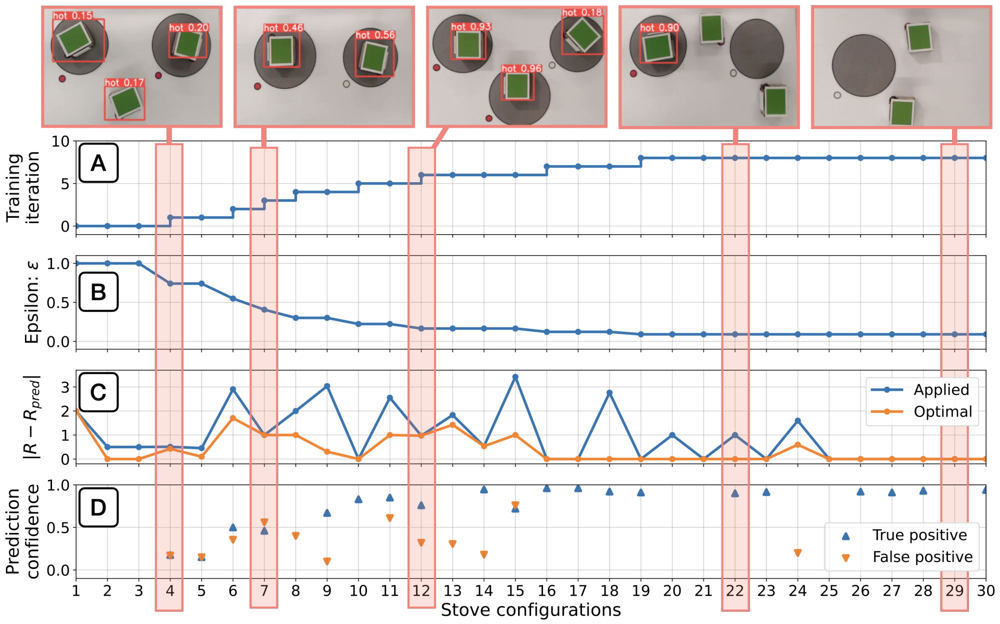


Fig. 7. Result of the iterative learning process, summarized by a time series evolution of the training iteration, exploration parameter ϵ, error in predicted reward, and confidence of detected objects. A sample of 5 prediction images returned from the neural network and their corresponding stove configurations are shown.

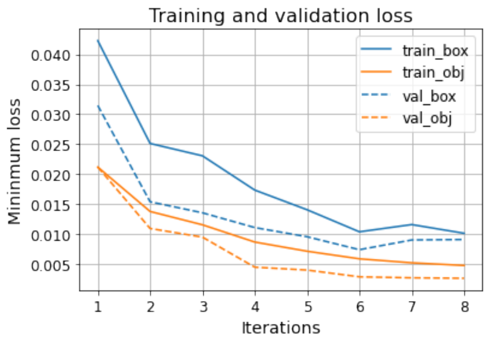


Fig. 8. Minimum training and validation loss during each iteration for bounding box localization and object classification.

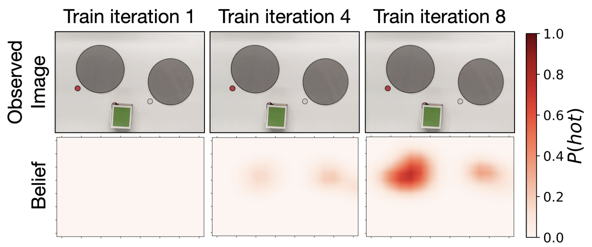
applied to a growing number of stove configurations. Fig. 7 summarizes the results for this iterative learning process which involved 30 different stove setups. Time series a) shows when the network is retrained, and time series b) shows the evolution of ϵ and the shift from exploration to exploitation. To assess the performance of the decisions made by the robot, time series c) plots the magnitude of the error of the predicted reward and the maximum reward obtainable in a particular stove configuration given by R. The orange curve shows the error based on the action that produces the optimal predicted reward $|R - R_{pred}(\mathbf{a}^*, \mathbf{p}, \mathbf{Q})|$ (i.e. without the addition of any exploration caused by ϵ). Whereas, the blue curve is the error based on the applied action $|R - R_{pred}(\mathbf{a}, \mathbf{p}, \mathbf{Q})|$, which may divert from the optimal due to random exploration. This metric shows both the convergence of the neural network's prediction quality and the exploration performed to divert from the optimal predicted action. For both curves, we see the error reaches zero towards the end of the stove configurations. In fact, for the latter half of the iteration experiments, 14 out of 15 stove configurations have been perfectly assessed.

Another analysis of the performance of the iterative training is the confidence of predictions, shown by time series d) for both true and false positives. Here, we define true positives as the network correctly labelling a hot object and the false positives as the network labelling a cold object as hot. The true positive average prediction confidence starts low (less than 50%), but gradually increases with each training iteration until the confidence excelled 90%. For the false positive average prediction confidence there are very few cases after 6 training iterations and when they do occur, the probability is considerably lower.

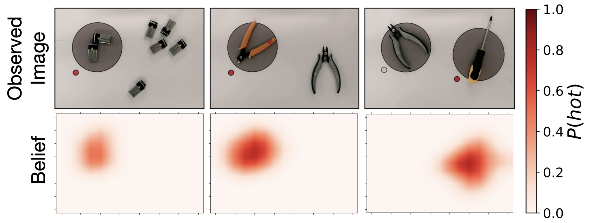
The minimum training and validation loss during training iteration for the bounding box localization and object classification are presented in Fig. 8. For both networks, the loss decreases until iteration 6. This is also approximately the iteration at which the prediction confidence and action selection improves (see Fig. 7). Despite the limited amount of training data and limited number of training cycles, the plateuing of the loss and the experimental results suggests the robot can begin to learn the necessary visual features to identify hot objects.

B. Heatmap Generalization

To demonstrate the generalization of the learned information captured by the neural network to understand the heat profile of the environment we apply our 'thought experiment' method. A number of examples of these heatmaps and their respective scenes are given in Fig. 9. Fig. 9a shows the heatmaps generated from unseen images using the neural network for various training iterations (corresponding to Fig. 7). The heatmap generated from the first iteration trained network results shows no understanding of the heat profile of the surface. As the number of iterations increases, the heatmap belief improves which a clear prediction of the 'on' hotplate are having a higher probability of being hot. The



a) Robot's belief of the environment for different training iterations



b) Robot's belief of the environment with unobserved images using training iteration 8

Fig. 9. Heatmaps generated using the 'thought experiment' method for different images and trained networks.

second 'off' hot plate does have some probability of being hot, but much lower. To evaluate this method more generally we create three additional scenes which have 'unseen' objects (Fig. 9b). In all cases, the heatmap correctly generates an approximately round area to indicate the 'on' hotplate, with no 'false-positive' observed for the 'off' hotplate. The heatmap generation seems largely unaffected by the presence of additional objects in the scene suggesting that this methods seems promising for generalizing object understanding to environmental understanding. In this demonstration, the grasping location and orientation of each 'unseen' object is pre-defined and the heatmap used only to decided whether to execute the grasp or not.

VII. CONCLUSIONS & DISCUSSION

In this work we present the implementation of a robotic manipulation system with an analog reflex override and a visual learning system. The learning system leverages the pain signal from the reflex to learn the visual context that signals an object is hot. We show how this allows the presence of objects being on a 'hot plate' and the corresponding indicator light be learned with minimal training data to allow the robot to make 'safe' decisions. This learning information can be generalized from object detection to detection of hot areas across a hot by utilizing the concept of a 'thought-experiment'. This mirrors the concept of an internal simulation in the brain.

However, some challenges remain. For example, due to small dataset size the performance of the network was sensitive to the labels and information on the training dataset. Likewise, the exploration parameter ϵ heavily affects the amount and type of data obtained, which must be tuned. This could be addressed for example if the training loss data can be used to guide exploration. Further exploration and optimization of these parameters would improve the robustness of this method.

Although the specific case study is far from a realistic scenario and the robotic manipulator is not seriously damaged by hot temperatures, these approaches offer an insight into potential approaches that could be deployed. By extending this approach to incorporate additional nociceptors, and exploring advances in self-healing materials and damage detection [17] would allow exploration in a more complex and realistic scenario.

REFERENCES

[1] T. Zhang, W. Zhang, and M. M. Gupta, "Resilient robots: concept, review, and future directions," *Robotics*, vol. 6, no. 4, p. 22, 2017.

[2] R. A. Bilodeau and R. K. Kramer, "Self-healing and damage resilience for soft robotics: A review," *Frontiers in Robotics and AI*, vol. 4, p. 48, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2211601X11001404

[3] S. Thrun and T. M. Mitchell, "Lifelong robot learning," *Robotics and autonomous systems*, vol. 15, no. 1-2, pp. 25–46, 1995.

[4] S. Cao, D. Fu, X. Yang, P. Barros, S. Wermter, X. Liu, and H. Wu, "How can ai recognize pain and express empathy," *arXiv preprint arXiv:2110.04249*, 2021.

[5] D. Talmi, P. Dayan, S. J. Kiebel, C. D. Frith, and R. J. Dolan, "How humans integrate the prospects of pain and reward during choice," *Journal of Neuroscience*, vol. 29, no. 46, pp. 14 617–14 626, 2009.

[6] B. Seymour, "Pain: A precision signal for reinforcement learning and control," *Neuron*, vol. 101, no. 6, pp. 1029–1041, 2019.

[7] G. Bekey and R. Tomovic, "Robot control by reflex actions," in *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, vol. 3. IEEE, 1986, pp. 240–247.

[8] Y. Masuda, K. Miyashita, K. Yamagishi, M. Ishikawa, and K. Hosoda, "Brainless running: a quasi-quadruped robot with decentralized spinal reflexes by solely mechanical devices," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4020–4025.

[9] A. A. Saputra, J. Botzheim, A. J. Ijspeert, and N. Kubota, "Combining reflexes and external sensory information in a neuromusculoskeletal model to control a quadruped robot," *IEEE Transactions on Cybernetics*, 2021.

[10] R. A. John, N. Tiwari, M. I. B. Patdillah, M. R. Kulkarni, N. Tiwari, J. Basu, S. K. Bose, C. J. Yu, A. Nirmal, S. K. Vishwanath, *et al.*, "Self healable neuromorphic memtransistor elements for decentralized sensory signal processing in robotics," *Nature communications*, vol. 11, no. 1, pp. 1–12, 2020.

[11] L. Cellier, P. Dauchez, R. Zapata, and M. Uchiyama, "Collision avoidance for a two-arm robot by reflex actions: Simulations and experimentations," *Journal of Intelligent and Robotic Systems*, vol. 14, no. 2, pp. 219–238, 1995.

[12] A. Kara, K. Kawamura, S. Bagchi, and M. El-Gamal, "Reflex control of a robotic aid system to assist the physically disabled," *IEEE Control Systems Magazine*, vol. 12, no. 3, pp. 71–77, 1992.

[13] S. Yigit, C. Burghart, and H. Woern, "Applying reflexes to enhance safe human-robot-co-operation with a humanlike robot arm," in *Proc., 35th International Symposium on Robotics*. Citeseer, 2004.

[14] J. Wang, S. Elfwing, and E. Uchibe, "Modular deep reinforcement learning from reward and punishment for robot navigation," *Neural Networks*, vol. 135, pp. 115–126, 2021.

[15] G. Jocher, A. Stoken, A. Chaurasia, J. Borovec, NanoCode012, TaoXie, Y. Kwon, K. Michael, L. Changyu, J. Fang, A. V, Laughing, tkianai, yxNONG, P. Skalski, A. Hogan, J. Nadar, imyhxy, L. Mammana, AlexWang1900, C. Fati, D. Montes, J. Hajek, L. Diaconu, M. T. Minh, Marc, albinxavi, fatih, oleg, and wanghaoyang0106, "ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support," Oct. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5563715

[16] C. J. Watkins and P. Dayan, "Technical note: Q-learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, May 1992. [Online]. Available: https://doi.org/10.1023/A:1022676722315

[17] T. George Thuruthel, A. W. Bosman, J. Hughes, and F. Iida, "Soft self-healing fluidic tactile sensors with damage detection and localization abilities," *Sensors*, vol. 21, no. 24, p. 8284, 2021.