Human Activity Recognition with Deep Reinforcement Learning using the Camera of a Mobile Robot

Teerawat Kumrai, Joseph Korpela, Takuya Maekawa
Department of Multimedia Engineering
Graduate School of Information Science and Technology, Osaka University
Osaka, Japan
teerawat.kumrai@ist.osaka-u.ac.jp, korpela1@gmail.com, maekawa@ist.osaka-u.ac.jp

Yen Yu, Ryota Kanai
Araya Inc.
Tokyo, Japan
{yen.yu, kanair}@araya.org

***Abstract*—This paper presents a new human activity recognition method that uses a camera mounted on a mobile robot. We assume that the robot's camera captures images of a person and recognizes his/her activities based on skeletal and visual features extracted from the images. A key issue encountered with this method for activity recognition is that it requires the robot to position itself so that it has an adequate field of view of the activities being conducted. For example, if the robot is directly behind a person while observing that person making tea, it will be difficult for the robot to distinguish that activity from other similar activities such as preparing a meal or washing dishes. Our method employs deep reinforcement learning to control the movements of the mobile robot that is observing the activities in order to maximize its recognition accuracy while minimizing its energy consumption related to its movement. We propose effective action- and state-space designs that can achieve early training convergence and highly accurate activity recognition by: (i) incorporating the confidence of the activity recognition output when evaluating the quality of the current state (position), (ii) incorporating the costs of subsequent actions when estimating values for those actions, and (iii) designing an effective action space that accelerates reinforcement learning by restricting the movement space of the robot to the circumference of a circle with a predefined radius centered on the person.**

***Index Terms*—Activity recognition and understanding, Reinforcement learning, Robotics**

I. Introduction

Human activity recognition (HAR) using sensor systems has been actively studied in the pervasive community, since HAR has many real-world applications, such as healthcare, elderly care, and lifelogging. Included in the sensor systems used for HAR are wearable sensors (e.g., smartwatches) and environmental sensors (e.g., RFID tags and surveillance cameras). In particular, body-worn inertial sensors have been used in several studies on recognizing simple activities [1]–[4]. Wearable cameras have also been used in the wearable and computer vision research communities to recognize complex activities [5]–[9]. However, constantly wearing a camera to support HAR in our daily life is not practical, because of the physical burdens imposed (especially for elderly persons) and rapid battery consumption.

Due to recent advances in robotics technologies, home robots are currently coming on to the market, with simple house-cleaning robots already in widespread use and more complex humanoid robots gaining popularity. For example, iRobot's Roomba[1] achieved cumulative sales of 20 million units as of September 2017 and SoftBank's Pepper[2], which is a humanoid mobile robot equipped with a variety of sensors such as a camera and depth sensor, achieved cumulative sales of 20 thousand units in Japan in early 2017. With robots likely becoming pervasive in our homes in the near future, using the sensors mounted on these robots for HAR is becoming feasible [10]. They provide a mobile platform from which to observe daily activities when performing HAR, without the physical burdens that come with wearable sensors. Furthermore, these robots will need to recognize the daily activities of their residents in order to implement context-aware services and improve human-robot interaction.

This study focuses on performing HAR using the camera found on a mobile household robot. In order to accurately conduct HAR while a person moves about the house, the robot should be controlled so that the activity is captured by its camera from an appropriate position and angle. For example, assume that a person is making tea in the kitchen. When the robot is observing the activity from directly behind the person, it will be difficult for the robot to distinguish it from other similar activities such as preparing a meal or washing dishes. In this situation, the robot should move itself to a position with an adequate field of view in order to improve recognition accuracy. In contrast, when the person performs a simple activity such as sleeping, the robot can easily recognize the activity without changing its position, reducing its energy consumption. Therefore, we propose a method for HAR that uses reinforcement learning (RL) to train the movements of the mobile robot observing the activities.

Our method uses the RL technique known as deep reinforcement learning to train a neural network to automatically control the actions of a robot in order to maximize its activity recognition accuracy. Specifically, we employ a deep Q-network (DQN) [11] to do this by moving the robot to positions that maximize the recognition accuracy of another neural network that is performing HAR on images captured by the robot's camera. In order to facilitate movement control by the RL network, we extract the confidence scores from the

[1]https://www.irobot.com/for-the-home/vacuuming/roomba
[2]https://www.softbankrobotics.com/emea/en/pepper

978-1-7281-4657-7/20/$31.00 ©2020 IEEE

HAR neural network for use by the RL network. The HAR network's expected confidence score is then used by the RL network to estimate values for the actions that the robot can perform.

This study proposes effective action- and state-space designs, along with reward engineering that can achieve early training convergence and highly accurate HAR, while considering various issues encountered by mobile robots in the home environment. We first propose an efficient action space to reduce the time needed for the deep Q-network to converge in comparison with when it is provided with an action space that allows for the robot to move freely throughout the house while performing HAR. In addition, the robot control method that we base on this action space permits us to control the movements of the robot in a way that ensures it does not hamper the daily activities of the person. For example, when a person is in the kitchen preparing a meal, the robot positioning itself too close to the person could interfere with his/her ability to do so. Therefore, the robot's movement is controlled so as to keep its distance from the person and to not enter/stay in predefined prohibited areas.

In order to estimate values for subsequent actions, we have designed a state space that contains the following information: (i) The confidence of the activity recognition results from the HAR neural network. Incorporating this confidence value into the state allows the RL network to determine when a position change is necessary. (ii) Obstacle information for the robot's immediate surroundings. This allows the network to learn how to avoid household objects and occluding objects when performing HAR. In addition, we introduce a simple format for encoding the obstacle information for the deep Q-network. (iii) Costs for the different actions that the robot can take. For example, when there is an obstacle on the robot's right, the robot will need to go around that obstacle when moving to the right, which results in a higher cost for that action.

To the best of our knowledge, this is the first work on HAR by a mobile robot using deep RL. In the rest of this paper, we first review activity recognition studies using embedded sensors and cameras. We then describe our proposed method for activity recognition by a mobile robot and evaluate our method in a virtual environment.

II. Related Work

In the pervasive computing community, previous studies have recognized human activities using sensors embedded in indoor environments, such as RFID tags and switch sensors installed in the environments [12]–[14]. Furthermore, in the pervasive and wearable computing communities, wearable sensors such as body-worn accelerometers and microphones have been used to recognize human activities by capturing postures, sounds, and repetitive motions that are characteristic of the activities [1], [15]–[18].

Due to recent advances in deep learning technologies, activity recognition studies using first-person images captured by wearable cameras have been actively studied in the ubicomp, wearable, and computer vision communities [7]–[9]. Moreover, several studies have focused on HAR for robot-assisted living. Vieira et al. [10] implemented a real-time application for human daily activity recognition by a robot that uses a Dynamic Bayesian Mixture Model (DBMM) for activity recognition. Their robot is able to localize, navigate, detect obstacles, follow a person, and recognize human activities in an indoor environment. Piyathilaka et al. [19] proposed a HAR method that uses 3D skeleton features generated from a robot's depth camera with a joint weight model.

Similar to our study, several studies have also attempted to optimize a camera's position when monitoring human activities. Bodor et al. [20] analytically optimized the position of a mobile surveillance camera for monitoring walking persons by maximizing the visibility of the walking path. Schroeter et al. [21] optimized the observation pose for a mobile camera used in indoor living environments by focusing on light sources as well as obstacles. They minized the issues caused by glare and shadows from light sources by introducing the light's representation. Kessler et al. [22] tried to observe humans unobtrusively by positioning the robot using a method based on particle swarm optimization. Ishara et al. [23] controlled the position of a mobile camera in order to capture the entire skeleton (i.e., positions of body parts) of the person being observed. In contrast, our study controls a mobile robot in order to directly maximize its HAR accuracy through the use of RL.

III. Activity Recognition Method

A. Preliminaries

In this study, we evaluate the proposed method using the HoME Platform virtual environment [24], because a controllable environment is needed to evaluate our RL-based method. Such virtual environments are able to realistically animate the movements of humanoid characters using a skeletal animation model in various household settings and can produce a stream of realistic images captured from a camera position and angle of the user's choosing. We can use this platform to simulate the daily-life activities of a person by having our humanoid character perform a variety of activities in the environment with the character's position set to the appropriate location for each activity (e.g., sitting on a sofa in the living room or preparing a meal in the kitchen), with the humanoid character walking between locations when transitioning from one activity to the next. We assume that the skeletal animation models are captured from actual persons using a mocap system in advance.

The mobile robot is then simulated by controlling the camera position and angle based on the size and movement specifications of the robot we are modeling. The mobile robot used for our virtual environment is designed as follows based on a commercially available humanoid robot (Softbank Pepper) in order to facilitate real-world implementation. (i) The camera is mounted on the robot's head at a height of 1 m facing to the front. (ii) The movement speed and rotation speed of the robot are 0.83 m/s and 34.26 deg/s, respectively. (iii) The resolution of the camera is 512 by 512 pixels and the

frame rate is 24 fps. Fig. 1 shows example images captured by the robot of a person in the virtual environment.

For simplicity, we simulate the task of indoor positioning of the robot and human within the virtual environment by obtaining their positions using HoME platform's API, since the focus of this study is on controlling the mobile robot's movement during HAR using RL. For methods for acquiring robot and human positions in real environments, refer to [10], [25], [26]. We also assume that a floor map including information about the locations of obstacles is given.

Fig. 1: Example images captured by a robot in a virtual environment

B. Overview

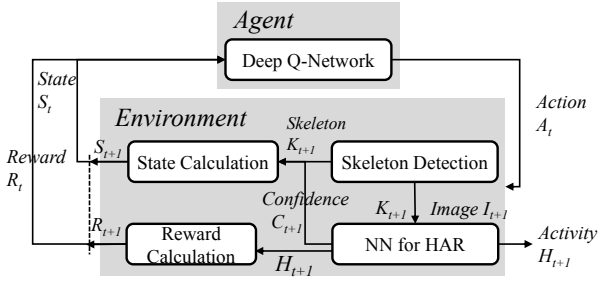


Fig. 2: Overview of the proposed method

Figure 2 shows an overview of the proposed method. In RL, an agent learns in an environment by trial and error (exploration and exploitation) using feedback based on its own actions and experiences. As shown in Fig. 2, when starting at the state S_t at time t, the agent (robot) determines its action (movement) A_t using its deep Q-network. As a result of the action, the position of the robot's camera changes, and images are captured by the camera at that new position. The human skeleton K_{t+1} is then detected in those images [27], which is used to generate a new state S_{t+1} and is used by the HAR neural network (NN) to estimate a new human activity class H_{t+1}. Along with the skeleton information K_{t+1}, the portions of the images that correspond to the locations of the user's hands are also fed into the HAR NN, enabling us to capture information about the objects used in the activities. The HAR NN also outputs the confidence of its estimate (C_{t+1}), which is used to estimate a value for the agent's current state.

C. Reinforcement Learning for HAR

The goal of the agent in general RL is to find a policy that maximizes its expected future rewards. In this study, we train the agent to find a policy that maximizes its HAR accuracy. To do this, we compute the confidence of the output from the HAR network and incorporate it into the current state. The deep Q-network [11] in the agent learns a Q function of policy π that takes as its input an agent's state and action, and maps them to probable future rewards as follows: $Q_\pi(s;a) = E[R_{t+1}|S_t = s; A_t = a]$, where R_{t+1} shows the reward at time $t+1$. The agent then uses this Q function to select an action that maximizes the expected discounted sum of future rewards[3]. In our case, an action corresponds to a movement by the robot with the agent determining its next action once every 3 seconds in our implementation.

When training the deep Q-network, the network is updated to correct the difference between its expected reward and the observed reward to adjust its weights as follows:

$$Q(S_t, A_t) \leftarrow (1-\alpha) \cdot \underbrace{Q(S_t, A_t)}_{old\ value} + \alpha \cdot \underbrace{(R_{t+1} + \gamma \cdot \max_a Q(S_{t+1}, a))}_{learned\ value}, \quad (1)$$

where α shows the learning rate and $\gamma \in [0, 1]$ is a discount factor. Therefore, the network is trained by using stochastic gradient descent to minimize the following loss:

$$(R_{t+1} + \gamma_{t+1} \max_{a'} Q_{\overline{\theta}}(S_{t+1}, a') - Q_\theta(S_t, A_t))^2,$$

where γ_{t+1} is the discount at $t+1$, θ are the parameters of an online Q-network used for selecting an action, and $\overline{\theta}$ are the parameters of a target network, which is a periodic copy of the online network which is not directly optimized.

Next, we explain the NN used for HAR; give details about our designs for the action space, state space, and reward; and describe the deep Q-network used for determining actions.

D. Neural Network Used for HAR

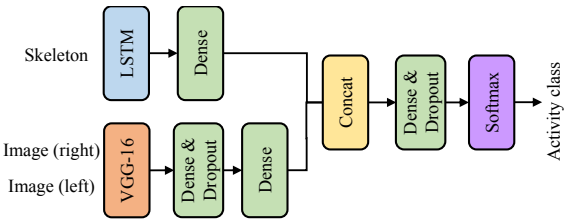


Fig. 3: Architecture of the neural network for HAR

The input of the HAR NN is comprised of skeleton information extracted from the raw camera images (K_t) along with a cropped image from each of the hand positions detected using the skeleton information (I_t). First, we extract skeleton information from several consecutive camera images using the OpenPose library [27]. This skeleton information is a set of coordinates for each image that describes the position of 25 of a person's body parts (e.g., positions of the head, right hand, left hand, etc.), with several consecutive images used to capture the motion of those body parts over time. In our method, the x- and y-coordinates are both normalized to the range [-1,1] and the coordinates of occluded body parts are regarded as missing values. Images from the hand positions detected in that skeleton information are then cropped so that each image is centered on one of the hands, with a null image used when a hand is not detected. Note that the cropped images are extracted only from the last of the consecutive images in order to reduce the computational cost.

[3]A future reward is discounted by using a discount factor $[0, 1]$.

The structure of the HAR network used in this study is shown in Fig. 3, which is based on a long-short term memory (LSTM) network, a type of recurrent NN used for time-series analysis [28], and a pre-trained convolutional neural network (CNN) for object recognition (VGG-16 [29] without the output layer). VGG-16 was chosen over other state-of-the-art networks due to its simplicity and low loss rate, combined with the fact that it still performs almost as well as the other networks on public datasets. The input for this network is the skeleton information from each 2-second window of images (512x512) along with the cropped images (64x64) from each detected hand position described above, with the skeleton information and cropped images fed into the LSTM and CNN layers, respectively. The output of the LSTM and CNN layers are then concatenated and processed in densely connected layers. Finally, an output layer with H nodes outputs the predicted probabilities for each of the H activity classes, with each node outputting a class probability for one class. We employ the Rectified Linear Units (ReLU) function as the activation function for the nodes in the LSTM layers, the ReLU function as the activation function for the nodes in the densely connected layers, and the softmax function as the activation function of nodes in the output layer. The NN also outputs the confidence of its estimates, which is used to estimate values for subsequent actions, as follows: $C_t = \max_i P(H_i|K_t, K_{t-1}, K_{t-2}, ..., I_t)$, where H_i is the i-th activity class.

We trained the network to minimize the cross-entropy between the distribution of the ground truth and the distribution estimated by the softmax output layer, employing backpropagation using Adam [30], which enables us to automatically adjust the learning rate, with the pretrained VGG-16 layers frozen during training. It was trained for 25 epochs using a batch size of 64 and an initial learning rate of 0.001. Fig. 4 shows the performance of this HAR NN as we change the viewing angle of the camera (agent) (see the Evaluation section for details on how this network was trained). As shown in these results, the recognition accuracy (macro-averaged F-measure) degrades when the activity is observed from behind. Note that this experiment was performed in an empty environment (i.e., no obstacles).

E. Design of the Action Space

As was mentioned earlier, an overly complicated action space can hinder network convergence. To simplify the action space, we propose having the robot follow the person at a given distance with its front camera facing the person so that the person is captured at the center of the camera's images. Therefore, as shown in Fig 5, the movement space available to the deep Q-network (robot) is restricted to the circumference of a circle centered on the person with a radius of d. Consequently, the actions that the robot can take are restricted to three general actions; a_{stay} (stay), a_{left} (go left), and a_{right} (go right). When the a_{left} or a_{right} action is selected, the robot moves along the circumference of the circle for 0.35 meters (10 degrees). After the robot arrives at the next position, the deep Q-network determines the next action. Note that when the person is walking to change his/her location (the movement speed is faster than s_w), the robot suspends the RL and HAR processes and simply follows the person.

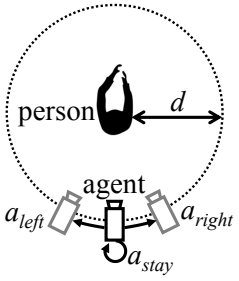
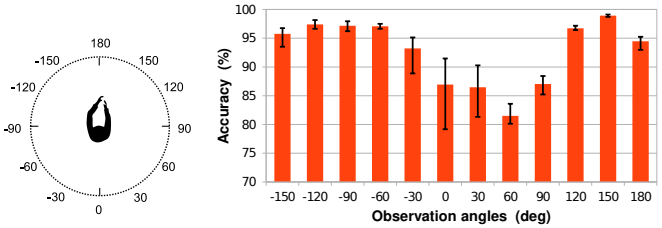


Fig. 5: Action space in the proposed method

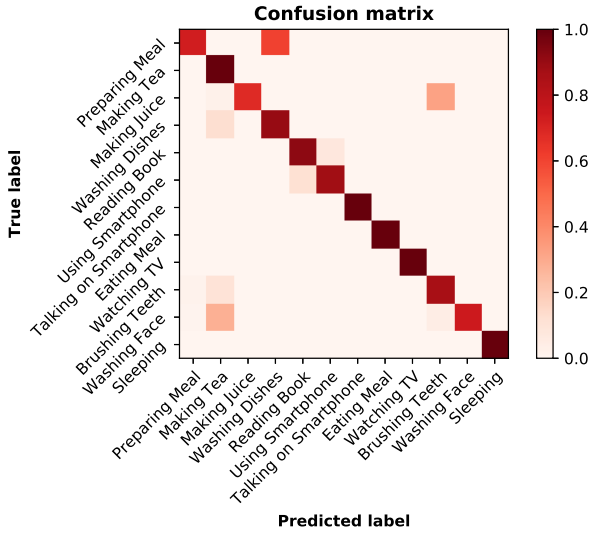
F. Design of the State Space

The agent's state, S_t, captures information about the current status of the agent and environment, such as skeleton information observed from its current position, and is used by the agent when determining which action to take next. At each time t, S_t is constructed by concatenating the skeleton information of the person K_t, the confidence of the HAR network C_t, and the obstacle information from the local environment O_t. The skeleton information, K_t, is composed of the x- and y-coordinates normalized to the range [-1,1] of each body part. The confidence C_t helps the deep Q-network to estimate a value for its current state, since C_t directly represents the current ambiguity of the HAR network's output. The obstacle information O_t helps the deep Q-network



(a) Observation angles

(b) Macro-averaged F-measures for different angles



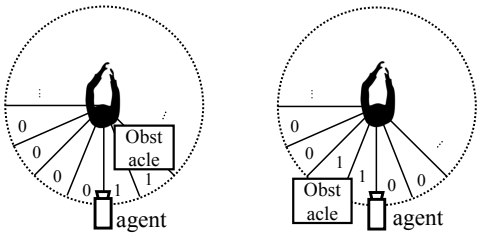
(c) Confusion matrix for 0 degrees

Fig. 4: The performance of the HAR NN. The distance between the person and camera is set at two meters. The HAR model was trained on five sessions of training data and tested on five sessions of test data for each participant with the results shown averaged over all participants. See the Evaluation section for more details.

estimate how the local environment will affect its ability to perform HAR, since obstacles between the robot and the person that are taller than the height of the camera can interfere with HAR.

In this paper, we assume that the robot has a floor map that contains information about obstacles in the local environment, with the information encoded as shown in Fig. 6 (a). The circle centered on the person is equally divided into N_O regions and the existence of an obstacle within an area is binary encoded (i.e., 0 or 1). Using this information, the agent can learn a movement policy that takes into account possible camera occlusion by obstacles. In addition to obstacles that can possibly occlude the camera, the obstacle information O_t also encodes information about obstacles that lie directly on the circumference of the circle, since such obstacles will block the movement of the robot. These obstacles are also binary encoded as shown in Fig. 6 (b). Note that while the movement space available to the deep Q-network (robot) is restricted to the circumference of the circle, we can program the robot to temporarily deviate from this area to bypass obstacles lying in its path when the deep Q-network has decided to move toward an obstacle.

Moreover, in order to facilitate the selection of an appropriate following action, O_t contains information about the costs of the possible following actions (a_{left} and a_{right}). For example, in the case of Fig. 6 (b), an obstacle exists to the robot's left and so the robot must go around the obstacle if it selects a_{left}, which increases the cost of this action. In order to represent such variations in the costs of actions, we include the estimated costs of the robot movements (i.e., movement distances) for a_{left} and a_{right} into O_t. The algorithm used to calculate the route from the current position to a destination is available in the supplementary materials[4].



(a) Encoding obstacles within the circle

(b) Encoding obstacles on the circle

Fig. 6: Examples of obstacle encodings

G. Design of the Reward

The reward R_t is computed from the HAR result (H_t); the movement distance of the previous action (d_m), which represents the energy consumed for the previous action; and the travel distance between the positions of the robot and person (d_{rp}) as follows: $R_t = A(H_t) - (e_p \cdot d_m + (d_{rp}/d))$, where d_m is the movement distance of the previous action, e_p is a hyperparameter related to the energy consumption of movement, and $A(H_t)$ indicates whether the HAR output is correct and is computed from the HAR result H_t at time t formulated as follows:

$$A(H_t) = \begin{cases} 1, & \text{if the agent can predict the activity correctly} \\ 0, & \text{otherwise} \end{cases}$$

Furthermore, we introduce d_{rp}, which is computed using Dijkstra's algorithm (see the supplementary materials[4] for details), in order to penalize situations where there is an obstacle between the robot and person or where the robot is situated in a different room from the person (i.e., non-line-of-sight). In such situations, the travel distance between the positions of the robot and person becomes longer than d because of the obstacle or walls. Otherwise, d_{rp} is close to d.

H. Deep Q-network

The deep Q-network used in this study consists of three densely connected layers with eight nodes and an output layer with three nodes. We employ the ReLU function as the activation function for the nodes in the densely connected layers. The number of nodes in the output layer corresponds to the number of actions, with each node in the output layer outputting the class probability of its corresponding action. The optimizer used for training is RMSProp [31]. To efficiently train the network, we used the following state-of-the-art RL techniques based on [32].

1) Categorical DQN: The original DQN algorithm employs the Q function to represent the expected reward. Instead of directly estimating the expected reward, Bellemare et al. [33] learned the distribution of rewards, permitting us to consider the variance of rewards and the multimodality of the reward distribution. In particular, the reward distribution is represented as a histogram in their implementation.

2) Multi-step RL: The original DQN algorithm employs a 1-step reward to generate training data. One alternative is to use forward-viewing n-step rewards to accelerate the training [32], [34] as follows: $R_t^{(n)} \equiv \sum_{k=0}^{n-1} \gamma_t^{(k)} R_{t+k+1}$, where $\gamma_t^{(k)}$ is the discount for a reward k steps in the future, defined as $\gamma_t^{(k)} = \prod_{i=1}^{k} \gamma_{t+i}$.

3) Double DQN: In the original Q-learning algorithm, the same Q function is used for selecting and evaluating actions (online and target networks; Eq. 1), resulting in overestimations. Double DQN copes with this issue by decoupling the selection from the evaluation [35].

4) Prioritized Experience Replay: The original DQN algorithm stores an agent's experiences $(S_t, A_t, R_{t+1}, S_{t+1})$, i.e., transitions, into a replay buffer, which is used to update the deep Q-network on a batch of experiences randomly sampled from the buffer. This sampling can be prioritized to accelerate learning. More specifically, a transition with a high TD error[5] is prioritized, which means that transitions that deviate from our estimates are prioritized.

5) Dueling Networks: To directly represent the values of states independently from actions ($V(s)$), Wang et al. [36] proposed an architecture that separately learns state values and action advantages. Here, the advantage is defined as

[4]http://www-mmde.ist.osaka-u.ac.jp/~maekawa/paper/supple/supplementary_information_for_percom2020.pdf

[5]A Temporal Difference (TD) error shows the difference between the estimated Q value and the actual Q value.

$A(s,a) = Q(s,a) - V(s)$. This network architecture is composed of a stream used to evaluate state values and a stream used to evaluate action advantages, which are merged by a special aggregator to output $Q(s,a)$.

6) Noisy Nets: The original DQN algorithm employs the ϵ-greedy strategy, which has a probability ϵ that the next action is selected randomly in order to introduce a form of exploration. Fortunato et al. [37] expanded on this by introducing a noisy linear layer into the NN to achieve exploration by the network.

I. Avoiding Obstacles

Here we explain several heuristics introduced in our method to avoid obstacles. Assume that there is an obstacle on the circumference of the circle on the robot's right, and the deep Q-network outputs a_{right}. However, because of the obstacle, the robot cannot move to the right. In this case, the agent temporarily pauses control by the RL process, detours around the obstacle to a position beyond it on the circumference of the circle, and then restarts control by the RL process. When there are no unobstructed points beyond the obstacle on the circumference of the circle, the robot simply ignores the action. The navigation method of the robot is further described in the supplementary materials[4].

IV. Evaluation

A. Data Set and Environments

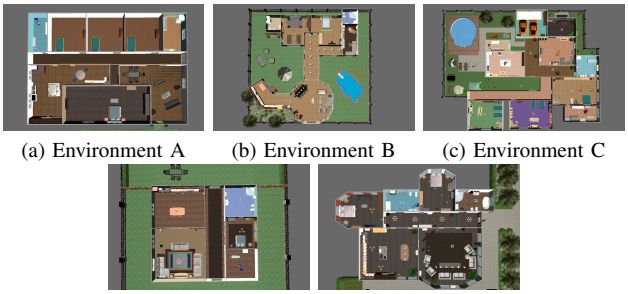
TABLE I: Activities performed in our experiment and objects used in each activity

Activity	Objects
Preparing Meal	Pan, spatula, knife and vegetable
Making Tea	Can of tea, teacup and kettle
Making Juice	Blender
Washing Dishes	Dish and sponge
Reading Book	Book
Using Smartphone	Smartphone
Talking on Smartphone	Smartphone
Eating Meal	Knife and fork
Watching TV	Remote control
Brushing Teeth	Toothbrush
Washing Face	N/A
Sleeping	N/A
Walking	N/A

TABLE II: Participants in our experiment

Participant	A	B	C	D	E
Height [cm]	168	173	162	155	173
Age	30s	20s	30s	50s	20s
Sex	M	M	F	F	M

We evaluated our method using the virtual home environment described in Section III-A. The movements of the humanoid characters used in the virtual environment were generated based on human activity data collected using the Perceptron Neuron mocap system[6], recording the coordinates of 25 body parts of each participant (e.g., the right hand, left hand, head, etc.). We collected the mocap data with each participant conducting ten sessions of activities. They performed the activities listed in Table I in an arbitrary order, with each session containing instances of each activity. Each of these sessions was conducted in either an actual home environment or in our laboratory. The humanoid characters used in the virtual environment were created using the MakeHuman[7] toolkit. A separate humanoid character was created for each participant according to their physical characteristics (listed in Table II).



(a) Environment A (b) Environment B (c) Environment C

(d) Environment D (e) Environment E

Fig. 7: Virtual environments used in this study

Furthermore, the virtual objects used by their characters in the virtual environment were created to each have their own unique appearance. These humanoid characters were then animated using the mocap data in the virtual houses shown in Fig. 7, with each participant performing activities in their own corresponding environment (e.g., Participant A performed activities in Environment A). Each activity was performed at an appropriate location in the house (e.g., preparing meal was performed in the kitchen), with the walking activity used to move the humanoid characters to different locations throughout the house. The duration of each virtual session was about 15 minutes while the average duration of each instance of activity was about 45 seconds.

B. Evaluation Methodology

We first trained a neural network for HAR. The HAR network that was used for each test participant was trained using the test user's five sessions of training data augmented with each of the other participants' ten sessions of data, since network training requires a substantial amount of training data. The training data used for the HAR networks was generated by animating the humanoid characters in an empty virtual environment (i.e., no obstacles) and recording their actions from twelve different angles using the camera on our virtual robot.

We trained the deep Q-network for each environment using its user's five sessions of training data. The deep Q-network was trained for 100 episodes. Note that the order of activities performed was randomized for each training iteration.

To evaluate the effectiveness of the proposed method, we prepared the following methods.

- Proposed: The proposed method.
- Naive: This method does not employ RL. In place of movement control by RL, the robot simply tracks the person so that the person is captured at the center of the camera image while keeping a distance d.
- NaiveAct: This method employs RL with a more complex action space than that used by the proposed method. In

[6]https://neuronmocap.com/

[7]http://www.makehumancommunity.org/

this method, the robot is allowed to move freely (forward, backward, left, and right in increments of 0.35 meters; stay; and rotate left/right in increments of 10 degrees) while maintaining a minimum distance d.

- W/o confidence: This method employs RL but does not use the confidence values from HAR in the state information.
- W/o cost: This method employs RL but does not incorporate the costs of the following actions (i.e., the expected movement distances) into the state information.
- DQN: This method employs RL but does not use the six state-of-the-art RL techniques described in Section III-H.

We evaluated the above methods using the macro-averaged F-measure calculated for the per frame estimates from the HAR networks when run in each environment using that environment's user's five sessions of test data. The hyperparameters used in this study are listed in Table III.

TABLE III: Hyperparameters

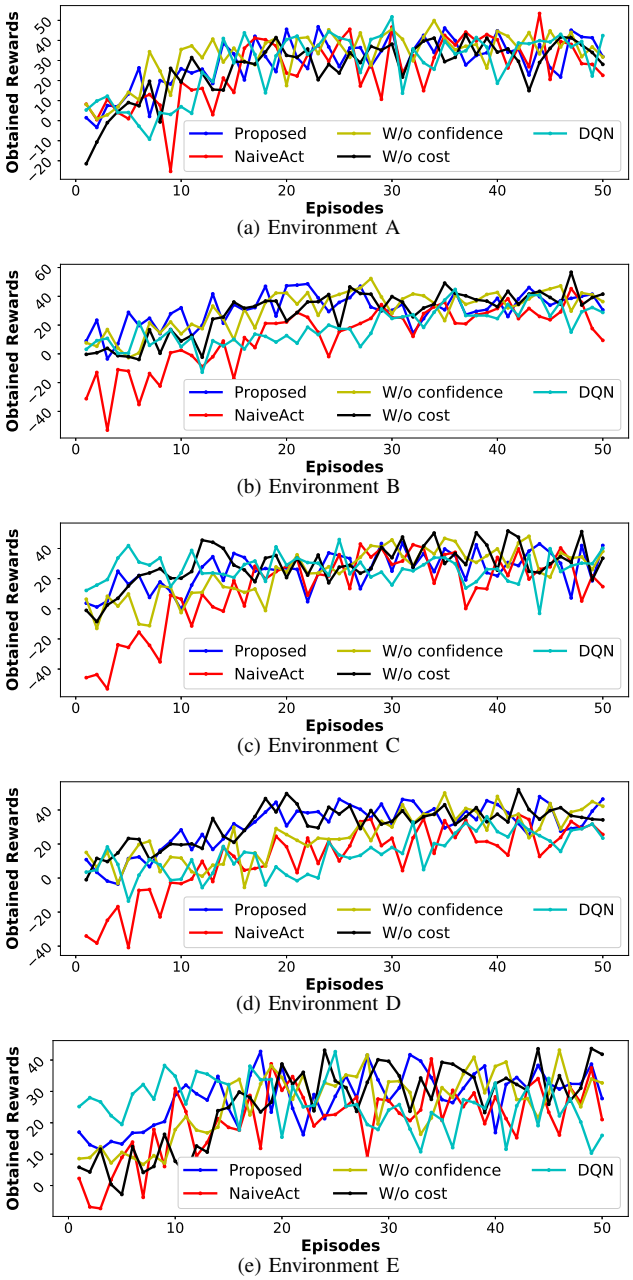
Hyper-parameter	Value	Description
d	2 m	distance between the person and robot
N_O	10	number of regions into which the circle centered on a person is divided
e_p	10^{-1}	energy consumption per 1 meter of movement
s_w	1 m/s	threshold speed used to define walking

C. Results

1) Recognition Accuracy and Reward: Figure 8 shows the reward curves for the RL based methods. As shown in these results, in many cases the reward increases much earlier for Proposed than it does for NaiveAct and DQN, showing the effectiveness of the proposed method. While this evaluation was conducted in virtual environments, an important future application of a HAR system such as this is in real-world environments. In such cases, the convergence speed is a very important aspect of training, since it can be costly for the agent to perform exploration and exploitation in a real-world environment.

Meanwhile, Fig. 9 shows the evolution of the movement distances of the robot (which are related to energy use) when performing HAR. Note that these distances do not include the robot's movement when it is following the person as he or she walks between the activity locations. In many cases the movement distances for Proposed converge earlier than the distances for W/o confidence, NaiveAct, and DQN. As was mentioned above, convergence speed is an important aspect of training in our task.

Figure 10 shows the average distance moved during instances of each activity class during testing. Because the circumference of the circle centered on the person is about 12.5 meters, these distances translate into an average traversal of about 100 degrees of that circle for each activity. As shown in the figure, the movement distances for "reading book," "using smartphone," "talking on smartphone," and "watching TV" are relatively long. All of these activities are performed while sitting on a sofa, so the robot cannot recognize these activities from behind the person. In contrast, the movement distances

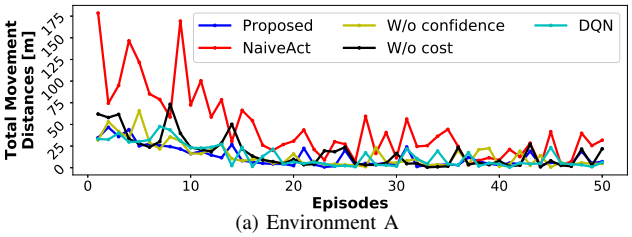


(a) Environment A
(b) Environment B
(c) Environment C
(d) Environment D
(e) Environment E

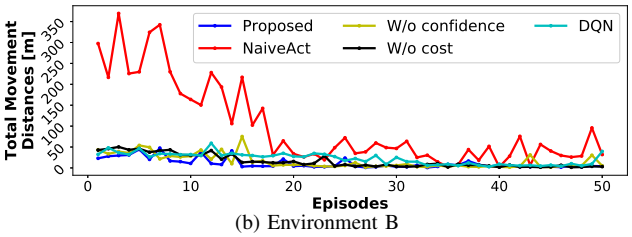
Fig. 8: Comparisons between proposed and the other methods in terms of reward curves plotted for 50 episodes of training

for "washing face" and "brushing teeth" are relatively short, because the robot can observe the characteristic body movements and postures for these activities from many angles. Also note that the average movement distance for "sleeping" was inflated by the results from Environment D, where the robot moved an average of 6.5 meters during instances of "sleeping." This was due to the bedroom in Environment D being smaller than the bedrooms in other environments, which led to the robot mistakenly entering neighboring rooms as it attempted to traverse the circle. Excluding Environment D, the average distance moved during "sleeping" was only 2.5 meters.

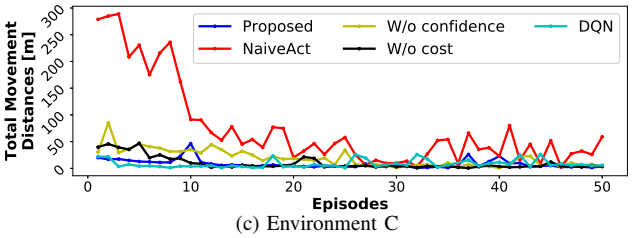
Figure 11 shows the macro-averaged F-measures for HAR for each of the methods in each of the environments during



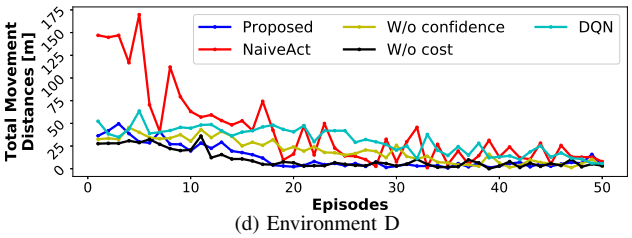
(a) Environment A



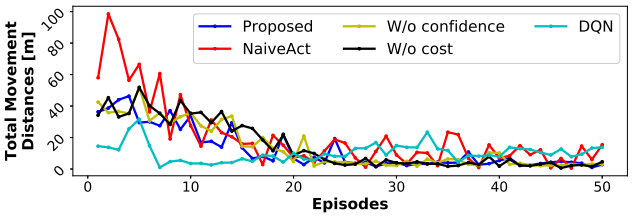
(b) Environment B



(c) Environment C



(d) Environment D



(e) Environment E

Fig. 9: Comparisons between proposed and the other methods in terms of the evolution of total movement distances [m] over 50 episodes of training

testing. As shown in these results, Proposed achieved the highest overall HAR accuracy. In particular, Proposed outperformed Naive by about 5-10%, showing the importance of adjusting the observation angle during HAR. Note that the accuracies shown here are somewhat poorer than those shown in Fig. 4, because the evaluation for Fig. 4 was done in an environment with no obstacles.

2) Effectiveness of Reinforcement Learning: As is illustrated in Fig. 11, Naive, which does not use RL, performs more poorly at HAR because it does not control the robot's movement in order to improve the robot's view of the activities being conducted. For each of the methods being evaluated,

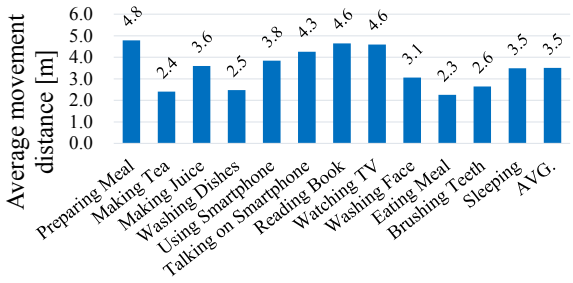


Fig. 10: Average movement distance by Proposed for each activity class during testing

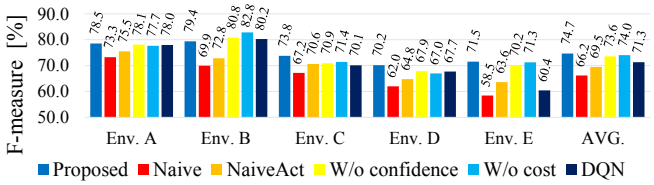
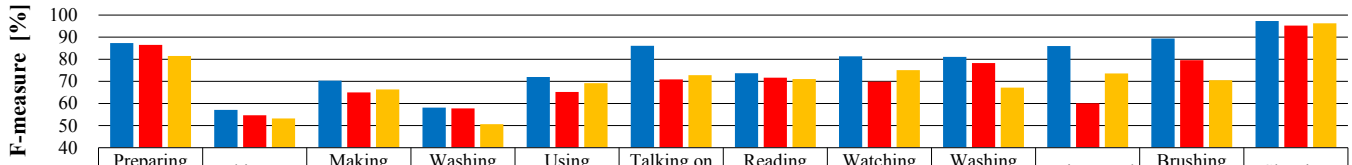


Fig. 11: Macro-averaged F-measures for HAR during testing

the robot arrives at the locations where it performs HAR by following the person being observed, meaning that HAR typically starts with the robot behind the person. Fig. 12 breaks the results down to F-measures per activity for Proposed, Naive, and NaiveAct. Here we can see that even activities like "washing face" and "brushing teeth" that have characteristic body movements are recognized more poorly by Naive, since there is still an advantage to improving the viewing angle during recognition for these activities. Moreover, the F-measures for activities such as "watching TV" and "talking on smartphone" that are performed on the couch are even more impacted by Naive's inability to adjust the observation angle, resulting in much poorer performance when RL is not used. As for the activities "making tea" and "washing dishes," their F-measures are poor because these activities were confused with each other. This may be because the robot could not capture distinguishing actions or objects due to the kitchen counter.

Figure 11 shows that the F-measures for Proposed, which employs state-of-the-art RL techniques, are higher than those for DQN in Environments D and E. In contrast, Fig. 13 which shows the average movement distance of each method for each environment, shows that the movement distances for DQN are generally much shorter than those for the other methods in Environments D and E. Based on these results, it appears that DQN attempted to get high rewards by saving the energy related to movements in these environments.

3) Effectiveness of the Action Space: As shown in Fig. 11, NaiveAct, which allows the robot to move freely, performs more poorly at HAR than Proposed despite its higher degree of freedom. The unrestricted movement in NaiveAct results in many choices of actions for the RL network, which then requires more episodes to train and converges more slowly than Proposed (see Fig. 8). In addition, Fig. 13 shows the average movement distance of NaiveAct for each environment, showing significantly longer movement distances for NaiveAct



	Preparing Meal	Making Tea	Making Juice	Washing Dishes	Using Smartphone	Talking on Smartphone	Reading Book	Watching TV	Washing Face	Eating Meal	Brushing Teeth	Sleeping
Proposed	87.3	57.1	70.3	58.2	72.0	86.1	73.7	81.3	81.1	86.0	89.4	97.3
Naive	86.5	54.7	65.0	57.8	65.2	70.9	71.6	69.8	78.3	60.1	79.6	95.3
NaiveAct	81.5	53.2	66.3	50.6	69.2	72.8	71.1	75.1	67.2	73.6	70.5	96.3

Fig. 12: Macro-averaged F-measure for each activity class during HAR

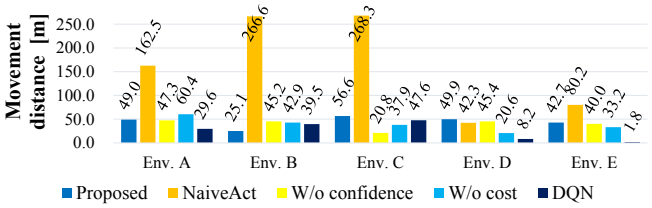


Fig. 13: Average movement distances for each environment

than for Proposed. Based on these results, we believe that our action space is suitable for efficient activity recognition.

4) Effectiveness of Activity Recognition Confidence: As shown in Fig. 9, Proposed converges much earlier than W/o confidence in Environments C and D. Because Proposed leverages information directly from the HAR network (i.e., HAR confidence), it can easily determine its policy using that information, accelerating the convergence of the RL network. On the other hand, W/o confidence must learn the concept corresponding to HAR confidence indirectly from the skeletal information, which requires many more iterations of training. Overall, these results show the usefulness of using the HAR confidence as an input.

5) Effectiveness of Costs of Next Actions: As shown in Fig. 9, Proposed converges earlier than W/o cost only in Environment B. This is likely because the cost information of each action is somewhat easy to predict from other input (i.e., obstacle information), meaning that W/o cost could learn a concept similar to the cost information within a few iterations of training. Comparing these results to those in the previous subsection, it is apparent that providing the confidence information from HAR to the RL network is more beneficial than providing the cost information for actions. This may be because the confidence information is useful for estimating a value for the current state that is difficult to accurately estimate using the other input.

V. Discussion

A. Limitations

One primary limitation of this study is that we assume that the robot only has simple movement functions. Therefore, our experiment considered only single-story houses and assumed that all the doors in the houses were always kept open. As a part of our future work, we plan to consider a robot with more advanced movement functions that include opening and closing doors and climbing up and down stairs.

TABLE IV: Macro-averaged F-measure [%] for HAR when deep Q-networks are trained in another environment

Train \ Test	Env. A	Env. B	Env. C	Env. D	Env. E
Env. A	–	72.18	64.43	72.43	69.32
Env. B	73.90	–	69.17	69.05	70.34
Env. C	78.38	80.48	–	65.88	68.63
Env. D	78.56	81.28	70.57	–	71.73
Env. E	76.68	79.48	67.47	65.93	–

B. Deep Q-network Trained in Another Environment

In our experiment, the deep Q-networks used were environment dependent (trained using data collected in the environment of interest). Here we investigate the performance of environment-independent deep Q-networks that avoid the need to train a network in the environment of interest. Table IV shows the F-measures for HAR when we use deep Q-networks trained in each environment to conduct HAR in each of the other environments. Surprisingly, in many cases the F-measures for the networks trained in different environments were almost as good as those of the environment-dependent models (Proposed in Fig. 11) and likewise still outperformed Naive in many cases. While some environmental conditions such as the size of the rooms and the positions of obstacles deteriorate the performance of the models in different environments, the agent seems to be able to learn general movement strategies that can be used in any environment (e.g., changing positions when the confidence of HAR is low), because of the simplicity of our proposed action space.

VI. Conclusion

This study proposed a new activity recognition method based on camera images captured by a mobile robot in the home. To maximize the activity recognition accuracy, our method employs deep reinforcement learning to control the movement of the robot while reducing the energy consumption related to that movement. We evaluated our method in virtual environments and confirmed its effectiveness. As a part of our future work, we plan to extend our method to maximize the activity recognition accuracy for multiple residents in a single house. In addition, we plan to deal with privacy issues related to our method. For example, we can restrict the robot from entering particular rooms and/or change its distance from the observed person depending on the estimated activity class.

VII. Acknowledgments

This work is partially supported by JST CREST JPMJCR15E2, JSPS KAKENHI Grant Number JP16H06539 and JP17H04679.

References

[1] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *Pervasive 2004*, 2004, pp. 1–17.

[2] R. Chavarriaga, H. Sagha, A. Calatroni, S. T. Digumarti, G. Tröster, J. d. R. Millán, and D. Roggen, "The Opportunity challenge: A benchmark database for on-body sensor-based activity recognition," *Pattern Recognition Letters*, vol. 34, no. 15, pp. 2033–2042, 2013.

[3] J. Korpela, K. Takase, T. Hirashima, T. Maekawa, J. Eberle, D. Chakraborty, and K. Aberer, "An energy-aware method for the joint recognition of activities and gestures using wearable sensors," in *International Symposium on Wearable Computers (ISWC 2015)*, 2015, pp. 101–108.

[4] T. Maekawa and S. Watanabe, "Unsupervised activity recognition with user's physical characteristics data," in *International Symposium on Wearable Computers (ISWC 2011)*, 2011, pp. 89–96.

[5] B. Clarkson, A. Pentland, and K. Mase, "Recognizing user context via wearable sensors," in *Int'l Symp. on Wearable Computers (ISWC 2000)*, 2000, pp. 69–75.

[6] T. Maekawa, Y. Yanagisawa, Y. Kishino, K. Ishiguro, K. Kamei, Y. Sakurai, and T. Okadome, "Object-based activity recognition with heterogeneous sensors on wrist," in *Pervasive 2010*, 2010, pp. 246–264.

[7] H. Pirsiavash and D. Ramanan, "Detecting activities of daily living in first-person camera views," in *CVPR 2012*, 2012, pp. 2847–2854.

[8] M. Ma, H. Fan, and K. M. Kitani, "Going deeper into first-person activity recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1894–1903.

[9] S. Singh, C. Arora, and C. Jawahar, "First person action recognition using deep learned descriptors," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2620–2628.

[10] M. Vieira, D. R. Faria, and U. Nunes, "Real-time application for monitoring human daily activity and risk situations in robot-assisted living," in *Robot 2015: Second Iberian Robotics Conference*. Springer, 2016, pp. 449–461.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[12] M. Philipose, K. P. Fishkin, M. Perkowitz, D. J. Patterson, D. Fox, H. Kautz, and D. Hähnel, "Inferring activities from interactions with objects," *IEEE Pervasive Computing*, vol. 3, no. 4, pp. 50–57, 2004.

[13] E. M. Tapia, S. S. Intille, and K. Larson, "Portable wireless sensors for object usage sensing in the home: Challenges and practicalities," in *European Conference on Ambient Intelligence*. Springer, 2007, pp. 19–37.

[14] T. Van Kasteren, A. Noulas, G. Englebienne, and B. Kröse, "Accurate activity recognition in a home setting," in *the 10th International Conference on Ubiquitous Computing (Ubicomp 2008)*, 2008, pp. 1–9.

[15] P. Lukowicz, H. Junker, M. Stäger, T. von Bueren, and G. Tröster, "Wearnet: A distributed multi-sensor system for context aware wearables," in *International Conference on Ubiquitous Computing*. Springer, 2002, pp. 361–370.

[16] P. Lukowicz, J. A. Ward, H. Junker, M. Stäger, G. Tröster, A. Atrash, and T. Starner, "Recognizing workshop activity using body worn microphones and accelerometers," in *Pervasive 2004*, 2004, pp. 18–32.

[17] M. Blum, A. S. Pentland, and G. Tröster, "Insense: Interest-based life logging," *IEEE Multimedia*, vol. 13, no. 4, pp. 40–48, 2006.

[18] J. Lester, T. Choudhury, and G. Borriello, "A practical approach to recognizing physical activities," in *Pervasive 2006*, 2006, pp. 1–16.

[19] L. Piyathilaka and S. Kodagoda, "Human activity recognition for domestic robots," in *Field and Service Robotics*. Springer, 2015, pp. 395–408.

[20] R. Bodor, A. Drenner, M. Janssen, P. Schrater, and N. Papanikolopoulos, "Mobile camera positioning to optimize the observability of human activity recognition tasks," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005.(IROS 2005)*. IEEE, 2005, pp. 1564–1569.

[21] C. Schroeter, M. Hoechemer, S. Mueller, and H.-M. Gross, "Autonomous robot cameraman-observation pose optimization for a mobile service robot in indoor living space," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 424–429.

[22] J. Kessler, M. Schmidt, S. Helsper, and H.-M. Gross, "I'm still watching you: Update on observing a person in a home environment," in *2013 European Conference on Mobile Robots*. IEEE, 2013, pp. 300–306.

[23] K. Ishara, I. Lee, and R. Brinkworth, "Mobile robotic active view planning for physiotherapy and physical exercise guidance," in *2015 IEEE 7th International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*. IEEE, 2015, pp. 130–136.

[24] S. Brodeur, E. Perez, A. Anand, F. Golemo, L. Celotti, F. Strub, J. Rouat, H. Larochelle, and A. Courville, "HoME: A household multimodal environment," *arXiv preprint arXiv:1711.11017*, 2017.

[25] A. Yassin, Y. Nasser, M. Awad, A. Al-Dubai, R. Liu, C. Yuen, R. Raulefs, and E. Aboutanios, "Recent advances in indoor localization: A survey on theoretical approaches and applications," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 1327–1346, 2016.

[26] J. Xiao, Z. Zhou, Y. Yi, and L. M. Ni, "A survey on wireless indoor localization from the device perspective," *ACM Computing Surveys (CSUR)*, vol. 49, no. 2, p. 25, 2016.

[27] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime multi-person 2D pose estimation using part affinity fields," *arXiv preprint arXiv:1812.08008*, 2018.

[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[30] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[31] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[32] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018, pp. 3215–3222.

[33] M. G. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 449–458.

[34] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine learning*, vol. 3, no. 1, pp. 9–44, 1988.

[35] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *AAAI*, vol. 2, 2016, pp. 2094–2100.

[36] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas, "Dueling network architectures for deep reinforcement learning," *arXiv preprint arXiv:1511.06581*, 2015.

[37] M. Fortunato, M. G. Azar, B. Piot, J. Menick, I. Osband, A. Graves, V. Mnih, R. Munos, D. Hassabis, O. Pietquin *et al.*, "Noisy networks for exploration," *arXiv preprint arXiv:1706.10295*, 2017.