Reinforcement Learning Motion Planning for an EOG-centered Robot Assisted Navigation in a Virtual Environment

Luís Garrote[1], João Perdiz[1], Gabriel Pires[1,2] and Urbano J. Nunes[1]

Abstract— **This paper presents a new collaborative approach for robot motion planning of an assistive robotic platform that takes into account the intentions of the user provided through Electrooculographic (EOG) signals, as well as obstacles surrounding the robotic platform. In order to increase human confidence in the operation of robotic platforms with some degree of navigational autonomy, the intent of the user must be included in the decision process. In our system, the human-robot interface works through ocular movements (saccades and blinks), which are acquired as EOG signals and classified using a Convolutional Neural Network. In our proposed approach, a model-free Reinforcement Learning (RL) layer is used to provide commands to a virtual robotic platform. The RL layer is constantly being updated with the inputs from the user's intent, environment perception and previous machine-based decisions. In order to prevent collisions, machine-based perception using the proposed RL motion planning approach will assist the user by selecting suitable actions while learning from prior driving behaviors. The approach was validated by a set of tests that consisted of driving a robotic platform in an in-house 3D virtual model of our Research Center (ISR-UC). The experimental results show a better performance of the proposed approach with RL when compared to the version without the RL-based motion planning component. Results show that the approach is a promising step in the concept put forward for collaborative Human-Robotic Interaction (HRI), and opens a path for future research.**

I. Introduction

In the last decade, there have been significant advancements in the development of robotic interfaces intended for the still incipient field of assistive robotics. From a utility and usability point-of-view, Human-Machine Interfaces (HMIs) must be capable of instilling in the user a sense of confidence when being used. The robot's reliable performance is the most important element when building a user's trust in the machine [1]. This is a particularly important aspect when developing navigational aids for motor-impaired users. In this work an RL-based navigation paradigm is proposed, with user EOG-based commands (no movement, leftward movement, rightward movement and single or double blink), which can be the basis for the implementation of different collaborative human-robot navigation approaches featuring learning capacity. Having in view the need to mitigate the apprehension with which the mobility aids – both motorized and non-motorized – are often seen by their users [2], it becomes necessary to equip the devices with human-oriented modes supporting reliable, safe and smooth navigation. To accomplish this goal, in this paper we propose an RL-based approach, aided by an EOG-based HMI in which the user's ocular movements contribute to navigating the platform around an indoor environment. The RL approach was implemented in a virtual environment so that its premises could be safely tested, validated, and compared to the results that an equivalent platform would obtain without the RL component.

We research whether it is possible to build a robotic navigation paradigm that benefits from model-free training, by taking into account the user intent, to overcome difficult navigation situations and penalize collisions and dead-ends, and compare it to an EOG-only navigation paradigm. Our approach to robotic navigation has the potential to increase trust of human users on robotic platforms, a particularly relevant topic in assistive robotics as it enhances adherence of potential users to robotic interfaces whose purpose is to assist aged or impaired people.

The remainder of the paper is organized as follows. Section II refers to related work, and the structure and modules of the proposed approach are detailed in Section III. In Section IV we present the virtual environment, the experimental results, and respective discussion. Final conclusions are drawn in Section V.

II. Related Work

A. EOG for HMIs

Electrooculography (EOG) signals have been mostly used to perform activity recognition of users [3], [4], but over the last few years interest has grown on the use of EOG in Human-Machine Interfaces (HMIs), in particular for physically impaired people. EOG has been used as a single input in wheelchair navigation [5], [6], [7], or in multimodal navigational paradigms employing two or more biosignals sources to control the wheelchair, either in a hybrid setting [8] or as one of a multiplicity of alternatives [9]. Similar to wheelchair guidance, EOG has also been used to aid human navigation. For example, in [10] it was used to provide six low-level navigational commands that were tested for navigational purposes in a virtual environment. To some extent, EOG-based implementations intended for navigational purposes can also be applied to games which, albeit simple, can be made intuitive enough to be successfully played by users without any experience in the use of EOG-based HMIs [11].

For the purpose of this work we employed a Convolutional Neural Network (CNN) for decision upon transformed EOG inputs. CNNs have seen exponential growth in their usage since 2012, but have remained mostly confined to image

[1] Authors are with the Institute of Systems and Robotics, Department of Electrical and Computer Engineering, University of Coimbra, Portugal.
[2] Engineering Department, Polytechnic Institute of Tomar, Portugal.
Emails: {garrote, joao.perdiz, gpires, urbano}@isr.uc.pt.

recognition [12], [13], with some recent applications in facial recognition [14]. Their use in biosignal-based Human-Computer Interfaces (HCIs) is still limited, although some examples combining CNNs with EOG signals can already be found [15], [16].

B. Reinforcement Learning in HMIs

The use of RL in the context of HMIs has become steadily more prevalent over the last few years. This has happened because RL is a fitting method for dealing with the unpredictability that comes when introducing a human factor into the context of machine learning. This factor, in turn, is compounded by the rise of collaborative robotics in which Human-Robot Interaction (HRI) is becoming more and more mainstream [17]. An example of a human-robot collaborative RL-based algorithm is proposed in [18]. The learning algorithm empowers the robot to adaptably switch its collaboration mode from autonomous to semi-autonomous.

Another example of the use of RL in an HRI can be seen in [19], where an RL-based robot-assisted navigation is implemented for a robotic walker taking into account both environmental data and user intent in the decision-making process.

III. Proposed Approach

In this Section we provide an overview of the systems that were used in this work, and describe our novel collaborative approach for robotic platform operation. In this context, we also describe the subsidiary systems whose development was necessary to arrive at a proof-of-concept situation.

A. System Overview

The system is composed of three major modules: the EOG acquisition and classification module, the virtual environment on which the platform moves, and the Reinforcement Learning robot motion planning method used to fuse input data from three different sources in order to output a valid motion command. An overview of the proposed system pipeline is presented in Fig. 1.

B. EOG classification

1) Ocular event definition: Since EOG-based classification decisions are one of the input types to the motion planning method, and the sole source of human interaction with the robotic platform, we divided ocular movements into four distinct classes: a "Null" event comprised of either no ocular movement or small leftward and rightward movements ($E = 0$), a leftward movement ($E = 1$), a rightward movement ($E = 2$), a single or a double blink ($E = 3$). Respectively, they will signal the user's intention for the platform to keep moving forward, to turn left or right, and stop (or resume moving if the platform was stopped). A stronger behavior of the platform towards user intent can be attained when the same event is repeated successively several times (through repeated EOG commands). The algorithm to obtain the user's intent from the EOG events is presented in **Algorithm** 1.

2) Neural network architecture, training and validation: For the EOG-based image classification that takes place in this work we implemented a CNN with three convolutional layers and four fully connected layers, with a Rectified Linear Unit (ReLU) following each layer. The architecture of this CNN is shown in Fig. 2. We used a cross-entropy loss function to train it and the Adam optimizer [20]. The training dataset consists of 3200 EOG images, collected from the calibration phases of several independent experimental trials conducted in our laboratory employing EOG as one of the biosignal inputs [11]. Images produced for the dataset were expert-labelled into one of the four classes. We used a workstation equipped with an Intel® Core™ i7-5930K processor, 64GB of memory and a NVIDIA's GeForce GTX 1080 graphics card.

C. Reinforcement Learning Robot Motion Planning

The platform's simulated 3D point cloud is its only source of information from the surrounding environment. To produce a viable command for the robotic platform, this information is combined both with decisions based on the ocular movements (or lack thereof), acquired via EOG signals, and with the robot's estimated pose in the virtual environment.

The proposed approach, using RL, aims to learn patterns (represented by states) between the local environment representation (M) and the user's intended motion. The proposed RL approach is divided in two stages: an offline learning stage and an online decision stage. The RL model has three inputs: a 3D point cloud, the pose of the robotic platform and the user's intention. The proposed RL model follows a divide and conquer approach commonly used in optimization tasks, in which a global problem is subdivided in smaller tasks that later will contribute to attain the final solution.

The consequences of both EOG-based commands and obstacle detection-based actions will impact the Reinforcement Learning method. Our method is a model-free Q-learning method employing state-based penalties and rewards. At each moment, states are determined based on the proximity and position of obstacles relative to the platform. State update rules are defined by both the local representation and the user's intention, and are subject to the consequences of previous actions (e.g., actions taken with or without assisted navigation).

After each command is issued to the platform, an evaluation of its effects on the platform's final state is executed, and a reward is calculated. The complete information pipeline, from EOG-based decisions to final evaluation in the context of Q-learning, is shown in Fig. 1. The proposed RL model is composed by multiple submodels (kernels) that compute local RL models from patches of the Obstacle cost map (**RLsubmodels**). Those models are evaluated in order to obtain an action that the platform will perform, where the action is given by the consensus of all the submodels.

The learning stage occurs offline (in the sense that it does not require a user to provide commands or move the platform) and creates an initial RL model that will provide

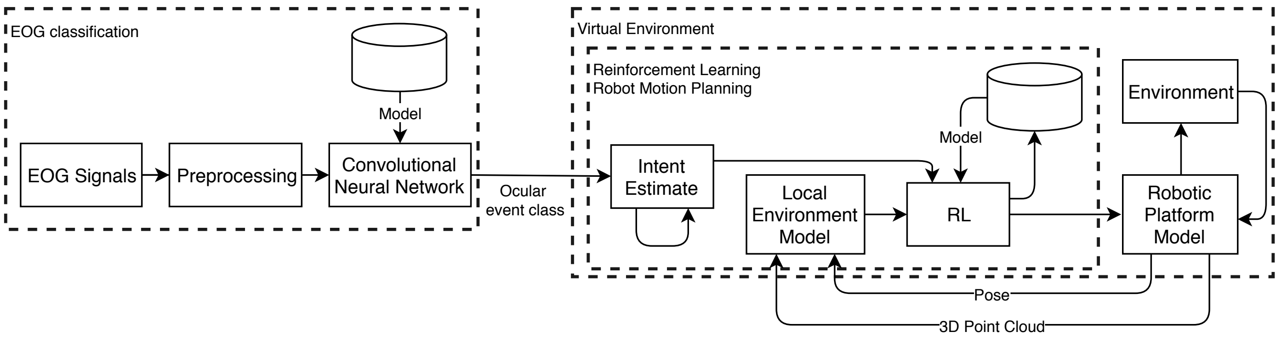

Fig. 1. Data processing of the navigation system approach encompassing both human and machine-based inputs. Note that effects of each command issued to the platform can only be evaluated in the following time step, and that most of the EOG inputs occurring will belong to the "Null" class of ocular events, such that human input will only produce an effect when the user deems it actually necessary.

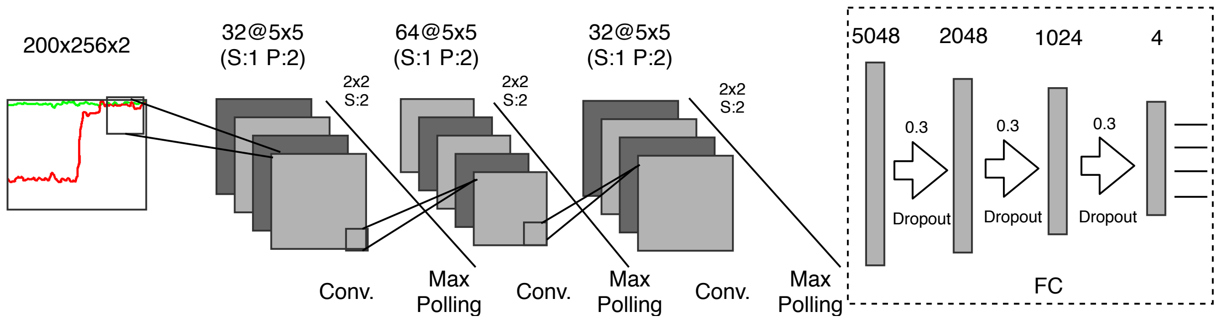

Fig. 2. Architecture of the CNN used in this work. It is composed of two convolutional layers and three fully connected layers.

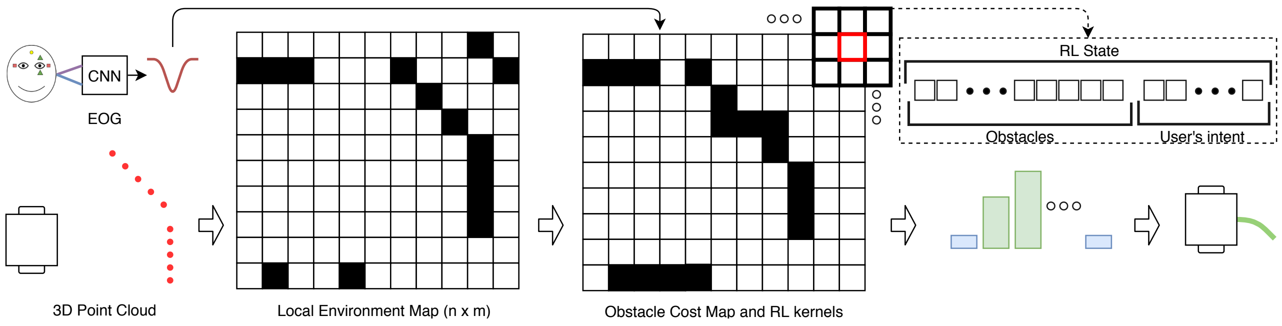
CNN
EOG
3D Point Cloud
Local Environment Map (n x m)
Obstacle Cost Map and RL kernels
RL State
Obstacles
User's intent

Fig. 3. Pipeline of the decision process showing the three different inputs of the RL model: user intent, point cloud, and robot pose. The obstacle cost map is merged with the local map and each of the RL submodels (kernels) compute a preferred action taking into account all three inputs; the most voted action is chosen to actuate the robotic platform. The RL state, for each kernel, is represented by a frame with 49 bits allocated to obstacle data from the 7×7 submodel and 8 bits dedicated to user intent information.

initial actions for the robotic platform to follow. The first step in the learning stage is to update the current local environment representation (**LocalMapUpdate**) with the 3D point cloud. In this work we consider a 2D occupancy grid representation, and for that reason, the ground points from the 3D point cloud are removed and the occupancy cells are updated by the remaining points, projected in the XY plane. The following step provides a memory-like model (M_{RL}) by merging the local map with a modified obstacle map (**RLMapUpdate**). A decay is applied to each cell in M_{RL} and the occupied cells in M are projected to the new grid with an inflation radius. A reward is computed (**GetReward**) considering the nearest obstacle, the user's requested intent and previous action command. A new intent ($EV0$ through $EV3$) is computed from the last EOG event (E) and the previous estimated intent (**UpdateIntent**). The EOG event 3 ($EV3$) updates the linear speed while the other events update the angular speed. Since events are computed at 4Hz, a buffer window of 1s is used to aid the update of user intent. A left or right EOG event is in most cases provided sequentially (e.g., users tend to move the eyes to the left side of the screen and then return to the middle) and need to be carefully processed in order to provide an accurate representation of an user's intent. The proposed method is presented in **Algorithm** 1.

The last action is obtained by analyzing the data from a user driving a robotic platform and by exploring the possible actions in each scenario. For each submodel in the set K_{RL} the state is computed and the internal Q-Matrix is updated.

Algorithm 1: Buffer-based user intent update (**UpdateIntent**).

```
Input: Event window ($B_E$)
       EOG event ($E$)
       linear and angular velocities ($v, w$)
       angular update ($\Delta w$)
Initialization:
$B_E \leftarrow 0_{1xN}$;

Update Rules:
if $EV3 \notin B_E$ then
    if $E = 3$ then
        $w \leftarrow 0$;
        $v \leftarrow \begin{cases} 0 & ,\text{If } v = v_{max} \\ v_{max} & ,\text{If } v = 0 \end{cases}$;
if $EV1 \in B_E$ then
    if $E = 1$ then
        $B_E \leftarrow B_E \bigcup E$;
    else
        $B_E \leftarrow B_E \bigcup \begin{cases} EV3 & ,\text{If } E = 3 \\ EV0 & ,\text{otherwise} \end{cases}$
else if $EV2 \in B_E$ then
    if $E = 2$ then
        $B_E \leftarrow B_E \bigcup E$;
    else
        $B_E \leftarrow B_E \bigcup \begin{cases} EV3 & ,\text{If } E = 3 \\ EV0 & ,\text{otherwise} \end{cases}$;
else
    if $E = 1$ then
        $w \leftarrow w + \Delta w$;
    else if $E = 2$ then
        $w \leftarrow w - \Delta w$;
    else
        $w \leftarrow 0$;
    $B_E \leftarrow B_E \bigcup E$;
$B_E \leftarrow B_E \setminus B_E(1)$;
Output: $u_{user} \leftarrow (u, w)$
```

The online decision stage initially follows a similar approach as in the learning stage. Instead of relying on the user's actions, this stage relies on the previous action provided by the RL model. After the model's update, an histogram is computed with all the decisions of the submodels. For a state not observed in the training stage or for a maximum Q-value action that has a negative Q-value a special empty action is accumulated in the histogram (-1). The output action is given by the most voted action by all the submodels; a threshold can be added in this step to guarantee a certain degree of confidence in the final decision. If the selected action corresponds to the empty action, a DWA-like procedure [19] with a modified cost function is applied to compute an acceptable command that complies with the user's request and avoids the nearest obstacles. The DWA-like cost function is defined as follows:

$$c_{DWA}(M, u_{user}, u)) = N_t |u_{user} - u| + K_O \sum_{i=1}^{N_t} \frac{1}{d_i(M)} \qquad (1)$$

where N_t is the number of lookahead steps in the DWA, u the control command associated with a given action, d_i the distance at step i to the nearest obstacle and K_O a weight.

Algorithm 2: RL-RMP - Q-Learning inspired local robot motion planning.

```
Input: Local Environment Model ($M$)
       Pose ($\hat{\mathbf{x}}_\mathbf{R}$)
       3D Point cloud ($P$)
       EOG event ($E$)
Initialization:
$K_{RL} \leftarrow$ RLsubmodels(size($M$)),$M_{RL} \leftarrow \emptyset$,$u_{user} \leftarrow (0,0)$,$u_{cmd} \leftarrow (0,0)$;
$S_A \leftarrow$ ActionCommands($w_b$)
Learning Loop:
$M \leftarrow$ LocalMapUpdate($M$,$P$,$\hat{\mathbf{x}}_\mathbf{R}$);
$M_{RL} \leftarrow$ RLMapUpdate($M$,$M_{RL}$);
$R \leftarrow$ GetReward($M$,$u_{user}$,$u_{cmd}$);
$u_{user} \leftarrow$ UpdateIntent($E$,$u_{user}$);
$a \leftarrow$ getLastAction();
foreach $k \in K_{RL}$ do
    $s \leftarrow$ GetState($M_{RL}$,$u_{user}$);
    $k_Q(k_s,a) \leftarrow k_Q(k_s,a) + \alpha(R + \max_{A'}(k_Q(s,:)) - k_Q(k_s,a))$;
    $k_s \leftarrow s$;
$u_{cmd} \leftarrow u_{user}$

Decision (with update):
$M \leftarrow$ LocalMapUpdate($M$,$P$,$\hat{\mathbf{x}}_\mathbf{R}$);
$M_{RL} \leftarrow$ RLMapUpdate($M$,$M_{RL}$);
$R \leftarrow$ GetReward($M$,$u_{user}$,$u_{cmd}$);
$u_{user} \leftarrow$ UpdateIntent($E$,$u_{user}$);
foreach $k \in K_{RL}$ do
    $s \leftarrow$ GetState($M_{RL}$,$u_{user}$);
    $k_Q(k_s,k_a) \leftarrow k_Q(k_s,k_a) + \alpha(R + \max_{A'}(k_Q(s,:)) - k_Q(k_s,k_a))$;
    $k_s \leftarrow s$;
$D \leftarrow 0_{(|A| \times |1|)}$;
foreach $k \in K_{RL}$ do
    $a \leftarrow \underset{a_i \in A}{\text{argmax}}\ k_Q(k_s,a_i)$;
    $D(a) \leftarrow D(a) + 1$;
a $\leftarrow \underset{a_i \in A}{\text{argmax}}$ (Histogram($D(a_i)$));
if $a = -1$ then
    $a$,$u_{cmd} \leftarrow$ DWA($M$,$u_{user}$);
else
    $u_{cmd} \leftarrow S_A(a)$
foreach $k \in K_{RL}$ do
    $k_a \leftarrow a$;
Output: $a$,$u_{cmd}$
```

The proposed RL approach is shown in **Algorithm** 2.

1) States: The RL state encodes the environment representation as well as the user's intended action. The definition of the RL state must be made carefully, as in this particular application, the use of a large kernel window may result in an overwhelmingly large number of possible states; this would make it prohibitive in terms of memory, for example, if all possible states have a representation. For a 7×7 RL kernel, the number of possible obstacles' combinations is 5.63×10^{14}. This would be a concern if all combinations where equally possible; however, some combinations will never occur in online applications. The use of the obstacles in the RL state would only allow the generation of an almost random avoidance model where the actions would be selected in order to avoid collisions. In order to generate a model that is able to select an action that is according to the perceived user's action a representation of that intention must be present in the RL state definition. In order to incorporate the user's EOG-based request, the user's intent is discretized into a total of 256 possible orientations. Each RL state for each submodel has the format shown in Fig. 3.

2) Actions: The actions are defined based on the limits imposed on the angular speed of the robotic platform. In the specific application of this study, the linear speed of the robotic platform will be small ($\pm$ 0.1m/s) and only two linear motion states are allowed (considering the CNN output, the double blinking event – $EV3$ – will pause or unpause the platform). The linear and angular speeds are computed from the selected action. The linear component is taken from the double blinking event while the angular speed is selected from a discretized set of previously defined angular speed values (**ActionCommands**) that respect the designed angular speed limits of the platform (w_b). The number of elements in the set of angular speed values is the same as the number of actions.

3) Rewards: To assess the reward for a performed motion, two metrics are evaluated: distance to the nearest obstacles and the existence of a major discrepancy between the computed command (u_{cmd}) and the user's intended motion (u_{user}). Depending on the scenario, the two metrics may be conflicting (e.g., the user may try to go towards an obstacle) and in order to avoid ambiguous rewards, the final reward (R) is given by the minimum reward for the two metrics,

$$R = \min\left(\underbrace{K_o - K_l e^{-\left(\frac{(x_o-x_c)^2}{2\sigma_x^2}+\frac{(y_o-y_c)^2}{2\sigma_y^2}\right)}}_{\text{Nearest Obstacle}}, \underbrace{K_u + e^{-\left(\frac{|u_{cmd}-u_{user}|}{\sigma_u}\right)^2}}_{\text{Error to Request}}\right) \tag{2}$$

where (x_o,y_o) are the coordinates of the nearest obstacle, (x_c,y_c) are the coordinates of the robotic platform in the local map and σ_x, σ_y and σ_u are metric-specific parameters. K_l, K_o and K_u are bias constants employed to modify the reward.

IV. Results and Discussion

A. Virtual Environment

Validation tests were conducted in a virtual environment. The virtual model "driven" by the user simulates a differential drive robotic platform, which is inspired in a real platform in continuous development at ISR-UC [21]. Its environmental perception module is informed by a virtual LIDAR-like 3D point cloud that is generated through ray-tracing. This point cloud can have up to 4000 points.

Figure 4 shows a view, as seen by a user of the robotic platform navigating in the virtual environment, which represents a full-scale model of the first floor of ISR-UC.

B. System architecture and physical setup

We used 67% of the dataset described in Section III-B.2 for training and 33% for validation. The dataset was divided into batches of 100 events and optimized over 200 epochs, with a final accuracy of 93.73% in cross-validation. The confusion matrix is shown in Table I.

TABLE I
Cross-validation results of CNN training for the classification of EOG events.

		Target class				
		0	1	2	3	Precision
Output class	0	416	13	9	10	92.9%
	1	12	222	0	0	94.9%
	2	10	0	244	0	96.1%
	3	11	0	1	106	89.8%

TABLE II
Validation parameters.

Parameter	Value	Parameter	Value	Parameter	Value
K_o	0.5	K_O	0.05	v_{max}	0.1
K_l	-1	(x_c, y_c)	(0,0)	α	0.5
σ_x	0.4	σ_y	0.3	N_t	20
K_u	-0.5	σ_u	0.3	Δw	0.1
w_b	[-0.5, 0.5]	–	-	–	-

We acquire EOG channels using a g.tec g.USBamp biosignal amplifier connected to a Matlab/Simulink-based PC interface. Horizontal and vertical EOG signals acquired from bipolar facial electrodes are converted into a single intensity map constructed as an RGB matrix, which is encoded into an image. One new image is sent to the CNN every 0.25s using a TCP/IP connection. Each decision of the CNN model serves as input to **Algorithm** 1, where it is converted into a user intent command. The user intent command is applied on the virtual robotic platform, so that this method can be generalized for human collaboration on any type of mobile robotic platform. Table II presents the parameters used during the validation process with the proposed approach.

C. Test scenario and metrics

The virtual environment that was constructed for the validation tests is shown in Fig. 5; the built-up area is a model of the first floor of the ISR-UC building. The virtual test area consists of a rendering of part of the ISR-UC HCMR lab, where the work was developed, the corridor around it, and the corridor leading to the entrance to a second room, where the endpoint of the test path lies. Two test modalities were conducted:

- Scenario 1 – Direct control of the robotic platform using EOG events (user's intent).
- Scenario 2 – Human-robot collaboration, with EOG input and RL model.

Both scenarios were performed by four volunteers, and the geometric paths of each volunteer's run with and without the RL robot motion planning are presented in Fig. 5. For each test performed by each volunteer, we recorded the path followed by the platform, as well as the user intent and the control commands sent to the platform. We also present in Fig. 6 the occurrences in which human and machine commands were conditioned by the Q-learning method, and how long did it take for the platform to complete the assigned course. An initial training of the RL model was performed prior to the tests, with a virtual joystick, in order to initialize the model, with good and bad examples of motion behaviours performed during this training.

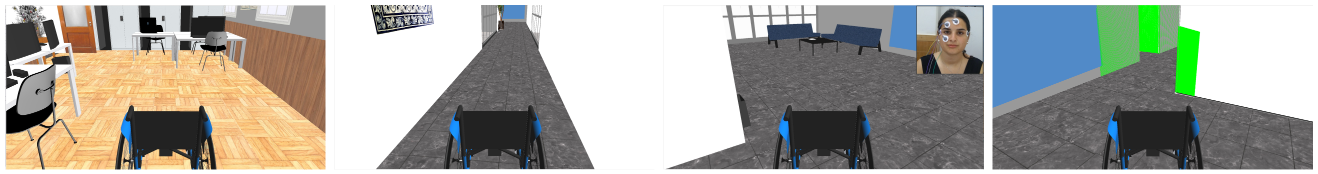

Fig. 4. Snapshots from the virtual environment from the user's point of view. From left to right: HCMR lab; corridor; ISR-UC hall entrance with overlapping picture of a volunteer with the EOG acquisition physical setup, and corridor with overlapped 3D point cloud.

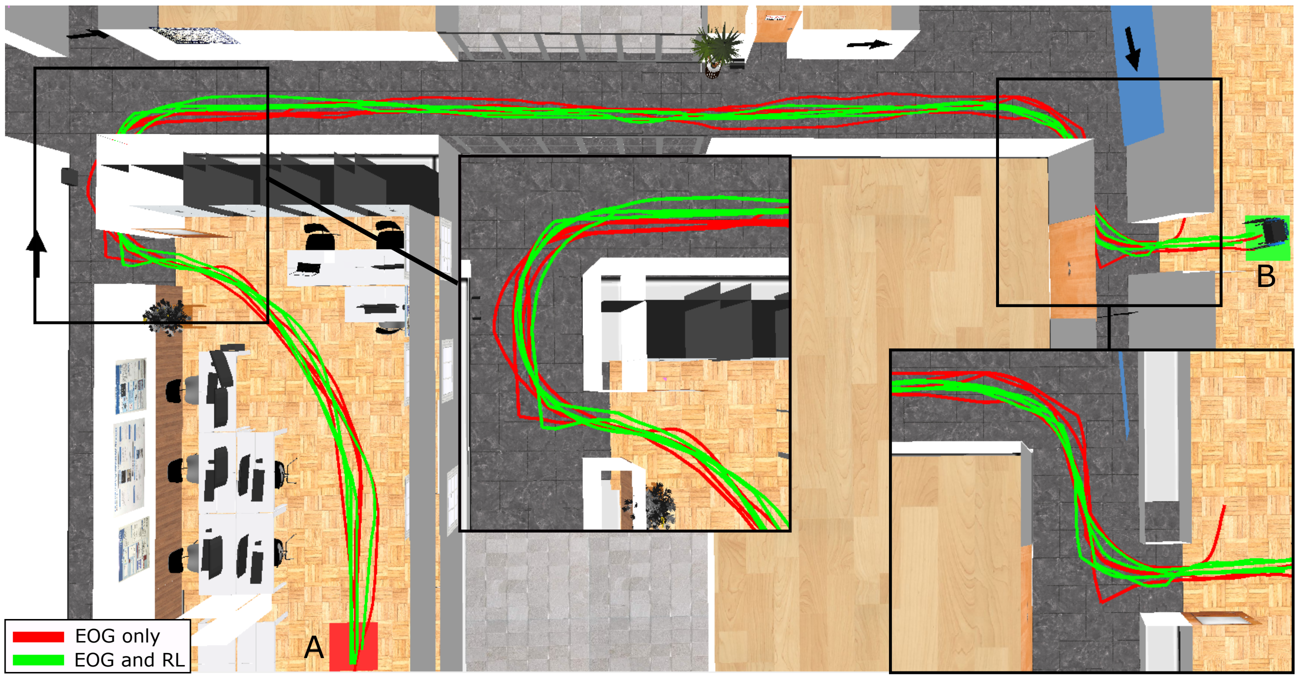

Fig. 5. Top view of the virtual environment used in the tests, showing an at scale representation of the ground floor of ISR–Coimbra and the obtained results for both scenarios performed by four volunteers. **A** and **B** squares mark the start and end points of the test path (in red and green respectively). In red the EOG-only geometric paths (Scenario 1) and in green the geometric paths where the RL was enabled (Scenario 2).

The results presented in Fig. 6 show that it was possible for the volunteers to execute the requested task in the two scenarios. However, for the proposed method the obtained geometric paths present a smoother shape and were more quickly completed (see Fig. 6). The volunteers struggled with the door entrances, where aligning the platform in order to avoid collisions was hard and led to multiple in-place adjustments. Although the behaviour of the CNN model is not directly evaluated in these scenarios, the proposed pipeline successfully provided events that were accurate classifications of user intent. The challenges observed by the volunteers were heavily related to the environment structure. For volunteers, the dependability of the robotic platform depends mainly on the smoothness of the navigation it provides, which means its effectiveness will increase with the proposed robotic navigation paradigm.

One of the volunteers was asked to perform two runs with the proposed RL approach. For the second run, the RL model that was updated during the previous run was used. The three runs for the two scenarios (with linear and angular speeds) are shown in Fig. 6. The obtained results show that using the proposed RL approach, the time needed to complete each run decreased substantially. It is important to note that the volunteers before the runs, performed tests inside the environment in order to familiarize with the system and the environment. The results for the first RL run show that in most cases the pre-trained RL model was not enough to drive the model, and the DWA-like approach was required to aid the user and update the RL model. However, for the second RL run and using the previous model learned under the DWA-like approach updates, the RL model was successfully refined and did not require in most cases the help from the motion planner.

V. Conclusion

In this paper we provide exploratory results that merging user intent and RL-based motion planning can be a factor in improving the performance of users of mobility assistance platforms when compared to a system lacking either the former or the latter input modes. Results obtained with a virtual platform are promising, and the RL motion planning algorithm successfully controlled the platform, aided by user intention anchored on EOG signals, after a single round of training for the RL states. Overall, volunteers improved their driving performance and the time needed to perform the

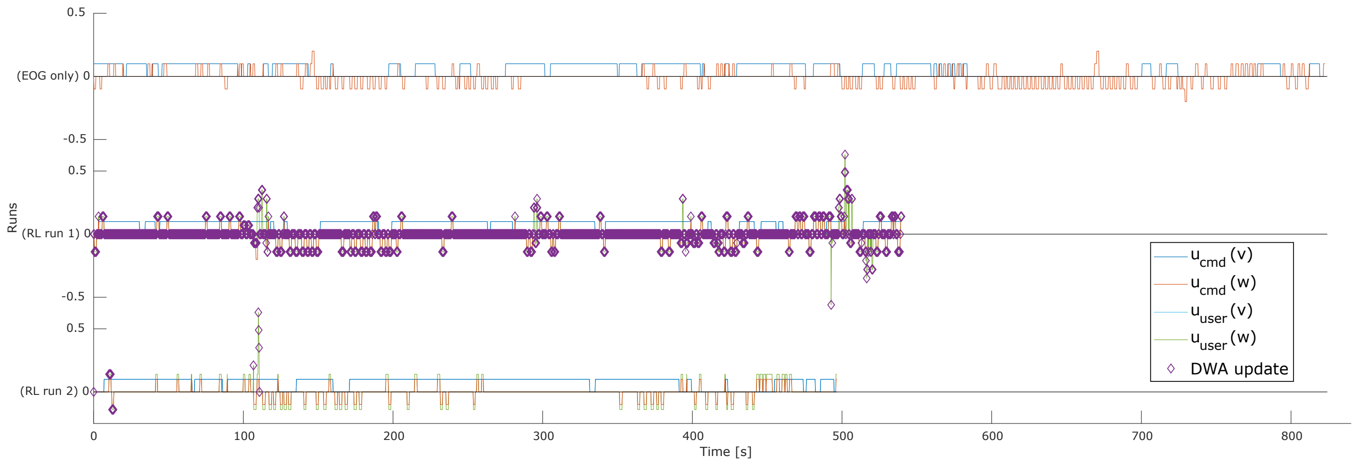

Fig. 6. Plot of the three runs of one of the volunteers, showing where in time user commands ($u_{user}(v)$ for linear speed intent and $u_{user}(w)$ for angular speed intent) and machine commands ($u_{cmd}(v)$ for linear speed and $u_{cmd}(w)$ for angular speed) were issued, and the moments when DWA was requested for lack of a state describing the fusion of orientation and user intent. The first run (top plot) was conducted only on user intent (EOG-based guidance), while the latter two allowed for the intervention of the proposed RL motion planning method. Note **RL run 1** had a significant number of motions based on DWA, while **RL run 2** did not. This can be attributed to the update that the RL models suffered during **RL run 1**, in which they incorporated the most common actions by this user into the learning model; when similar actions were executed again in **RL run 2** the RL motion planning algorithm already had states describing them, discarding the need for DWA-based motion planning to be used and making possible to complete the path even faster.

required scenarios was reduced. However, these preliminary results were achieved with a small number of volunteers, being in our plans to carry out a significant set of tests in order to obtain a consistent empirical validation.

Acknowledgments

This work was supported by the Portuguese Foundation for Science and Technology (FCT) under the PhD grants with reference SFRH/BD/88459/2012 and SFRH/BD/104985/2014, with funds from QREN – POPH and the European Social Fund from the European Union. It was also partially supported by the projects funded by FCT with reference FCT/COMPETE Centro2020/P2020 through grants UID/EEA/00048/2013, MATIS - CENTRO-01-0145-FEDER-000014 and B-RELIABLE - PTDC/EEI-AUT/30935/2017, Portugal.

References

[1] P. A. Hancock, D. R. Billings, K. E. Schaefer, J. Y. Chen, E. J. De Visser, and R. Parasuraman, "A meta-analysis of factors affecting trust in human-robot interaction," *Human factors*, vol. 53, no. 5, pp. 517–527, 2011.

[2] H. Bateni and B. E. Maki, "Assistive devices for balance and mobility: benefits, demands, and adverse consequences," *Archives of physical medicine and rehabilitation*, vol. 86, no. 1, pp. 134–145, 2005.

[3] A. Bulling, J. A. Ward, H. Gellersen, and G. Troster, "Eye Movement Analysis for Activity Recognition Using Electrooculography," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 4, pp. 741–753, 2011.

[4] S. Mala and K. Latha, "Feature selection in classification of eye movements using electrooculography for activity recognition," *Computational and mathematical methods in medicine*, 2014.

[5] R. Barea, L. Boquete, M. Mazo, and E. López, "Wheelchair Guidance Strategies Using EOG," *Journal of intelligent and robotic systems*, vol. 34, no. 3, pp. 279–299, 2002.

[6] A. Marjaninejad and S. Daneshvar, "A low-cost real-time wheelchair navigation system using electrooculography," in *2014 22nd Iranian Conference on Electrical Engineering (ICEE)*. IEEE, 2014.

[7] Q. Huang, S. He, Q. Wang, Z. Gu, N. Peng, K. Li, Y. Zhang, M. Shao, and Y. Li, "An EOG-based humanmachine interface for wheelchair control," *IEEE Transactions on Biomedical Engineering*, vol. 65, no. 9, pp. 2023–2032, Sep. 2018.

[8] N. Kim-Tien and N. Truong-Thinh, "Using electrooculogram and electromyogram for powered wheelchair," in *2011 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2011.

[9] M. Mazo, "An integral system for assisted mobility [automated wheelchair]," *IEEE Robotics & Automation Magazine*, vol. 8, no. 1, pp. 46–56, 2001.

[10] C.-C. Postelnicu, F. Girbacia, and D. Talaba, "EOG-based visual navigation interface development," *Expert Systems with Applications*, vol. 39, no. 12, pp. 10 857–10 866, 2012.

[11] J. Perdiz, L. Garrote, G. Pires, and U. J. Nunes, "Measuring the impact of reinforcement learning on an electrooculography-only computer game," in *2018 IEEE 6th International Conference on Serious Games and Applications for Health (SeGAH)*, 2018.

[12] O. Russakovsky, J. Deng, and H. e. a. Su, "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, Dec 2015.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[14] M. Peng, C. Wang, T. Chen, and G. Liu, "NIRFaceNet: A convolutional neural network for near-infrared face identification," *Information*, vol. 7, no. 4, 2016.

[15] X. Zhu, W.-L. Zheng, B.-L. Lu, X. Chen, S. Chen, and C. Wang, "EOG-based drowsiness detection using convolutional neural networks." in *IJCNN*, 2014.

[16] S. Hoppe and A. Bulling, "End-to-end eye movement detection using convolutional neural networks," *arXiv:1609.02452*, 2016.

[17] B. Chandrasekaran and J. M. Conrad, "Human-robot collaboration: A survey," in *SoutheastCon 2015*. IEEE, 2015.

[18] U. Kartoun, H. Stern, and Y. Edan, "A human-robot collaborative reinforcement learning algorithm," *Journal of Intelligent & Robotic Systems*, vol. 60, no. 2, pp. 217–239, 2010.

[19] L. Garrote, J. Paulo, J. Perdiz, P. Peixoto, and U. J. Nunes, "Robot-assisted navigation for a robotic walker with aided user intent," in *2018 27th IEEE Int. Symp. on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2018.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd Int. Conf. for Learning Representations (ICLR)*, 2015.

[21] R. Cruz, L. Garrote, A. Lopes, and U. J. Nunes, "Modular software architecture for human-robot interaction applied to the interbot mobile robot," in *2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, April 2018.