Acting Thoughts: Towards a Mobile Robotic Service Assistant for Users with Limited Communication Skills

F. Burget* L.D.J. Fiederer* D. Kuhner* M. Völker* J. Aldinger* R.T. Schirrmeister C. Do
J. Boedecker B. Nebel T. Ball W. Burgard

***Abstract*— As autonomous service robots become more affordable and thus available also for the general public, there is a growing need for user friendly interfaces to control the robotic system. Currently available control modalities typically expect users to be able to express their desire through either touch, speech or gesture commands. While this requirement is fulfilled for the majority of users, paralyzed users may not be able to use such systems. In this paper, we present a novel framework, that allows these users to interact with a robotic service assistant in a closed-loop fashion, using only thoughts. The brain-computer interface (BCI) system is composed of several interacting components, i.e., non-invasive neuronal signal recording and decoding, high-level task planning, motion and manipulation planning as well as environment perception. In various experiments, we demonstrate its applicability and robustness in real world scenarios, considering fetch-and-carry tasks and tasks involving human-robot interaction. As our results demonstrate, our system is capable of adapting to frequent changes in the environment and reliably completing given tasks within a reasonable amount of time. Combined with high-level planning and autonomous robotic systems, interesting new perspectives open up for non-invasive BCI-based human-robot interactions.**

I. Introduction

For patients with heavily impaired communication capabilities, such as severly paralyzed patients, their condition forces them to constantly rely on the help of human caretakers. Robotic service assistants can re-establish some autonomy for these patients, if they offer adequate interfaces and possess a sufficient level of intelligence. Generally, an intelligent system requires adaptive task and motion planning modules to determine appropriate task plans and motion trajectories for the robot, that implement a task in the real world. Moreover, it requires a perception component, e.g., to detect objects of interest or to avoid accidental collisions with obstacles. Typically used interfaces, such as haptic (buttons), audio (speech) or visual (gesture) interfaces, to command the robotic system are intuitive and easy options for healthy users, but difficult to impossible to use for paralyzed individuals.

In this paper, we present a novel framework, schematically depicted in Fig. 1, that allows closed-loop interaction between users with minimal communication capabilities and a robotic service assistant. To do so, we record neuronal activity elicited in the human brain, the common origin of all types of communication, with an electroencephalography (EEG) system. Furthermore, we adopt a convolutional neural network approach for online decoding of neuronal activity, in order to allow users to navigate through a graphical user interface (GUI) provided by a high-level task planner. The set of feasible actions displayed in the GUI, depends in turn on the current state of the world, which is stored in a central knowledge base and continuously updated with information provided by the robot and a camera perception system. Once a task has been selected, it is decomposed into a sequence of atomic actions by the high-level planner. Subsequently, each action is resolved to a motion for the mobile manipulator using low-level motion and manipulation planning techniques. In the following, the individual components shown in Fig. 1 will be described in detail, before presenting a quantitative evaluation of the overall system regarding its performance.

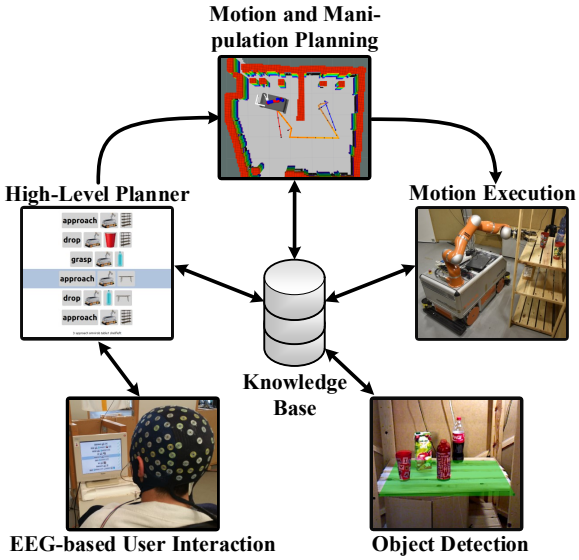

Fig. 1. Framework unifying decoding of neuronal signals, high-level task planning, low-level motion and manipulation planning, scene perception with a centralized knowledge base at its core. Intuitive goal selection is provided through an adaptive graphical user interface.

II. Related Work

Multiple previous studies have focused on robotic systems assisting people with disabilities. For example, Park *et al.* [1] implemented a system for the autonomous feeding of yogurt to a person. Chung *et al.* [2] focused on autonomous drinking which involved locating the drink, picking it up and bringing

* These authors contributed equally to the work. Authors are with the Department of Computer Science and Faculty of Medicine, University of Freiburg, Germany. {burgetf, kuhnerd, do, aldinger, jboedeck, nebel, burgard}@informatik.uni-freiburg.de and {lukas.fiederer, martin.voelker, robin.schirrmeister, tonio.ball}@uniklinik-freiburg.de. This research was supported by the German Research Foundation (DFG, grant number EXC 1086) and grant BMI-Bot by the Baden-Württemberg Stiftung.

it to the person's mouth. Using a hybrid BCI and head movement control, Achic *et al.* [3] studied a setup with a moving wheelchair and an attached robotic arm. None of these systems used pure BCI control. In contrast, Wang *et al.* [4] used a motor imagery BCI with three classes to achieve low-level control of a robotic arm. More relevant, Schröer *et al.* [5] developed a robotic system which receives a BCI command from a user and autonomously assists the user in drinking from a cup. However, this approach only considers a single object and a fixed-base manipulator. More recently, Muelling *et al.* [6] presented a shared-control approach to assistive robotics, albeit focused on invasive BCIs. Nonetheless, their approach could be combined with the high-level planning approach presented in our work.

In these applications, robust decoding of brain signals is required. Inspired by the successes of deep convolutional neural networks (ConvNets) in computer vision [7], [8] and speech recognition [9], [10], deep ConvNets have recently been applied more frequently to EEG brain-signal decoding. Deep ConvNets were already applied to decoding tasks useful for building brain-computer interfaces. Lawhern *et al.* [11] used a deep ConvNet to decode P300 oddball signals, feedback error-related negativity and two movement-related tasks. When evaluated cross-subject, i.e., trained on some subjects and evaluated on others, their ConvNet yields competitive accuracies compared with widely-used traditional brain-signal decoding algorithms. Tabar and Halici [12] used a ConvNet combined with a convolutional stacked autoencoder to decode motor imagery within-subject, yielding better accuracies than several non-ConvNet decoding algorithms. Schirrmeister *et al.* [13] used a shallow and a deep ConvNet to decode both motor imagery and motor execution within-subject, reaching or slightly surpassing the accuracies of the widely used EEG motor-decoding algorithm *filter bank common spatial patterns* [14]. Bashivan *et al.* [15] used a ConvNet trained on fourier-transformed inputs to estimate mental workload. In addition to the work on evaluating ConvNet decoding accuracies, ConvNet visualization methods allow us to get a sense of what brain-signal features the network is using [13], [15], [16]. Taken together, these advances make deep ConvNets a viable alternative for brain-signal decoding in brain-computer interfaces. Still, to our knowledge, online control with deep ConvNets has not yet been reported for an EEG-based brain-computer interface.

III. Online Decoding of Neuronal Signals

The system at hand is developed to control more complex scenarios than the ones considered in previous work. Particularly, we consider scenarios involving manipulation of objects as well as human-robot interaction. Feasible goals are determined by our GUI which is controlled by directional commands. As reliable classification of brain signals into navigation directions cannot yet be achieved directly with non-invasive BCIs, we used a deep ConvNet approach for decoding of multiple mental tasks from EEG (Schirrmeister *et al.* [13]). This approach introduces a hybrid network, combining a deep ConvNet with a shallower ConvNet architecture. The deep part consists of 4 convolution-pooling blocks using exponential linear units (ELU) [17] and max pooling, whereas the shallow part uses a single convolution-pooling block with squaring non-linearities and mean pooling. Both parts use a final convolution with ELU to produce output features. These features are then concatenated and fed to a final classification layer. We trained the ConvNet to decode five mental tasks: right hand finger and both feet toe movements, object rotation, word generation and rest. These mental tasks evoke discernible brain patterns and are used as surrogate signals to control the GUI. *Offline* training was done with a cropped training strategy using shifted time windows within the trials as input data [13].

From our experience it is important to train the BCI decoder and subjects in an environment that is as close as possible to the real application environment to avoid pronounced performance drops. Therefore, we designed a gradual training paradigm within the high-level planner GUI where the displayed environment, timing and actions are identical to those of the real control task. The training paradigm proceeds as follows: We first train each subject *offline* using simulated feedback. Subjects are aware of not being in control of the GUI. The mental tasks are cued using grayscale images presented for 0.5 s in the center of the display. At all times a fixation circle is displayed at the center of the GUI and the subject is instructed to fixate on it to minimize eye movements. After a random time interval of 1-7 s the fixation circle is switched to a disk for 0.2 s, which indicates the end of the mental task. At the same time the GUI action (*go up, go down, select, go back, do nothing*) corresponding to the cued mental task is performed to update the GUI. To keep training realistic we include a 20 % error rate, i.e., on average every fifth action is erroneous. We instruct the subjects to count the error occurrences to assert their vigilancy. This offline data is used to train the individual deep ConvNets. Then, the subjects do *online* training by performing the decoded mental tasks in the GUI. Finally, we stop cueing the mental tasks. To evaluate the performance of the BCI control, we let the subjects create instructed high-level plans in the GUI. These tasks are then executed by a simulated robot or the real mobile manipulator, when available. To provide more control over the mobile manipulator and enhance the feeling of agency, subjects have to confirm the execution of every planned action and can interrupt the chain of actions at any moment during their execution. BCI decoding accuracies for the label-less instructed tasks are assessed by manually rating each decoding based on the instructed task steps. Statistical significance of the decoding accuracies were tested using a conventional permutation test with 100 k random permutations of the labels (i.e., p-value is the fraction of label permutations that would have led to better or equal accuracies than the accuracy for the original labels).

IV. High-Level Goal Formulation Planning

We use domain independent planning to derive the required steps for reaching a desired high-level goal in a

complex task. The user can formulate a high-level goal without knowledge of the internal representation of objects in the planning system and the exact capabilities of the robot. This is achieved by an intuitive graphical user interface, where the object parameters of the goal are specified by incrementally refining the objects by referring to their *type*, e.g., "cup" or *attributes*, e.g., "content = apple-juice".

Domain independent planning identifies a sequence of actions that transforms the current world state into a state satisfying a goal condition. A planning task consists of: (i) a planning domain describing static components such as the object type hierarchy and the available actions and (ii) a problem instance describing the objects present in the world and their current state, as well as a goal description. While the current state of the objects can be extracted from the knowledge base, the goal has to be chosen in the GUI.

A restricted vocabulary is shared between the user and the planning system. Objects or sets of objects are identified by creating *referring expressions* to them composed of *shared references* built on this vocabluary [18]. We briefly describe the relevant aspects of our previous work in this area [19]. In general, a *referring expression* ϕ is a logical formula with a single free variable. ϕ *refers* to an object o if $\phi(o)$ is valid. E.g., the reference $\phi(x) \equiv cup(x) \wedge contains(x, water)$ refers to all cups containing water. We restrict ourselves to references that are simple conjunctions of facts, which is not only preferable for computational reasons, but also allows us to incrementally refine references by adding constraints. For example, adding $contains(x, water)$ to $cup(x)$, restricts the set of all cups to the set of cups containing water.

We distinguish between three types of fundamental object references: *individual references*, *typename references* and *relational references*. *Individual references* are identified by name, such as the "omniRob" robot. *Typename references* can be identified by the name of their type. While we cannot refer to the cups in our scenario directly, we can refer to an unspecific cup. *Relational references* are encountered when objects can be referred to via predicates in which they occur as an argument. The relations in our scenario are object attributes. For example, the *content* of the cup is used to clarify which cup is meant. These object references are used to create references to goals. We start defining goals with the action that achieves it as we found that this is most natural to the user, e.g., $put(x, y) \wedge cup(x) \wedge shelf(y)$. After the initial selection of a goal type (e.g., *drop*) it is necessary to determine the objects for all parameters of the goal predicate or action. These parameters are refined by constraining the previous choice until the argument is either determined uniquely (i.e., it is impossible to constrain the argument further) or the user declares that any remaining option is acceptable. We exclude unreachable goals, but we allow for goals that can only be achieved after a sequence of preceding actions (e.g., *drinking water* could require to fetch a cup, bring it to the patient, fetch a bottle and pour the water into the cup in order to be executed). The goal that is determined by the selection process of the GUI is then passed to a custom domain independent planner.

V. Robot Motion Generation

For generating paths for the mobile base, we apply the sampling-based planning framework *BI²RRT** [20]. Given a pair of terminal configurations, it performs a bidirectional search using uniform sampling in the configuration space until an initial sub-optimal solution path is found. This path is subsequently refined for the remaining planning time, adopting an informed sampling strategy, which yields a higher rate of convergence towards the optimal solution. Execution of paths is implemented via a closed-loop joint trajectory tracking algorithm using robot localization feedback.

To realize pick, place, pour and drink motions efficiently, we adopt a probabilistic roadmap planner approach [21]. The planner uses a graph of randomly sampled task poses (end-effector poses), which are connected by edges. To find a plan between two poses, the planner connects both poses with the roadmap graph and uses the A* algorithm to find an optimal path between the start and goal pose. The execution of the plan maps the task space velocity commands to joint velocity commands by employing a task space motion controller. We sample random poses around the object to determine grasp motions. For dropping objects we extract horizontal planes from the camera's point cloud and sample poses above those planes to find a suitable drop location.

VI. Implementation Details

Implementation of our framework in the real world requires several components, such as neuronal signal decoding, scene perception, knowledge base operations as well as symbolic and motion planning, to run in parallel. Therefore, we distributed the computation across a network of 7 computers, communicating among each other via ROS. The decoding of neuronal signals has four components. EEG measurements are performed using *Waveguard EEG* caps with 64 electrodes and a *NeurOne* amplifier in AC mode. Additionally, vertical and horizontal EOGs, EMGs of the four extremities and ECG's are recorded. For recording and online-preprocessing, we used BCI2000 and Matlab. We then transferred the data to a GPU server where our deep ConvNet classified the data into 5 classes. The high-level planner GUI consists of a back- and front-end. The back-end of the GUI uses the *Fast Downward* planner [22] to iteratively build goal references and to find symbolic plans for the selected goal. As the planning time is not crucial for the performance of our system, we used *Fast Downward* with a basic configuration in our experiments. The central knowledge base is implemented as a ROS node, which is able to store objects with arbitrary attribute information. All changes in the knowledge base automatically trigger updates of the front-end, unexpected ones interrupt the current motion trajectory execution. Finally, we used *SimTrack* [23] for object pose detection and tracking.

VII. Experiments

To evaluate our framework, we consider the environment schematically depicted in Fig. 2, containing two shelves and a table as potential locations for manipulation actions. The

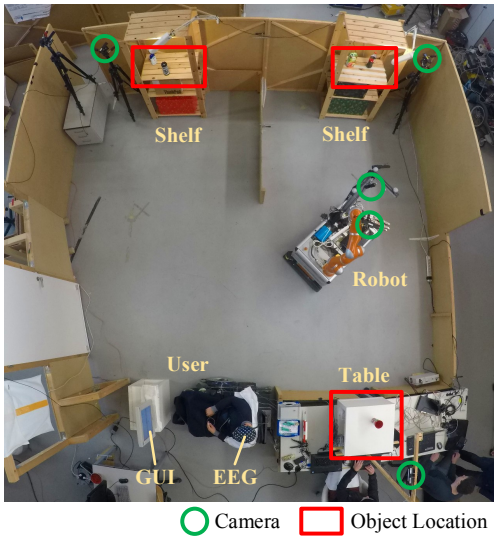

Fig. 2. Experimental environment: Two shelves and a table can be considered by the robot for performing manipulation actions. Five RGBD sensors observe the environment. A human operator selects a goal using EEG control and the high-level planner GUI.

user sits in a wheelchair in front of a screen, displaying the graphical interface of the high-level planner. The robot used in the experiments is the *omniRob* omni-directional mobile manipulator platform by KUKA Robotics, which is composed of 10 degrees of freedom (DOF), i.e., 3 DOF for the mobile base and 7 DOF for the manipulator. Additionally, the *Dexterous Hand 2.0* by Schunk is attached to the manipulator's flange and used to perform grasping and manipulation actions. The tasks we considered in our experiments required the robotic system to autonomously perform the following actions: *drive from one location to another*, *pick up an object*, *drop an object (on a shelf or table)*, *pour liquid from a bottle into a cup*, *supply a user with a drink*. Moreover, we use a perception system composed of five RGBD cameras. Three of them are statically mounted at the shelves and the table, in order to observe the scene and to report captured information to the knowledge base. The other two cameras are carried by the robot on-board. The first one is located at the mobile base and used to perform collision checks in manipulation planning. The second camera is mounted at the robot's end-effector and used for tasks involving physical human-robot interaction. A demonstration of our framework can be found in the accompanying video: http://www.informatik.uni-freiburg.de/~burgetf/ecmr17/.

A. Online Decoding of Neuronal Signals

We evaluated the BCI control setup with four healthy subjects (S1-4, all right-handed, three females, aged 26.75±5.9). At the time of writing the validation, S4 was still in progress and no validation with the mobile manipulator was performed. In total, 52 runs have been recorded (20 with the real robot) where the subjects executed various instructed high-level plans. For 32 runs, we used simulated feedback from the GUI in order to generate a significant amount of data for the evaluation. The performance of the BCI decoding during these runs was assessed using video

TABLE I

AGGREGATED MEAN±STD RESULTS FOR 52 BCI CONTROL RUNS (EXP. VII-A), * P-VALUE $< 10^{-6}$

	Runs #	Accuracy* [%]	Time [s]	Steps #	Path Optimality [%]	Time/Step [s]
S1	18	84.1±6.1	125±84	13.0±7.8	70.1±22.3	9±2
S2	14	76.8±14.1	150±32	10.1±2.8	91.3±12.0	9±3
S3	17	82.0±7.4	200±159	17.6±11.4	65.7±28.9	11±4
S4	3	63.8±15.6	176±102	26.3±11.2	34.5±1.2	6±2
	52	76.7±9.1	148±50	16.7±7.1	65.4±23.4	9±2

recordings of interactions with the GUI. We rated GUI actions as correct if they correspond to the instructed path and incorrect otherwise. Actions which are necessary to remediate a previous error are interpreted as correct if the correction is intentionally clear. Finally, we rated *rest* actions as correct during the (simulated) robot executions, incorrect if the next robotic action had to be initialized and ignored them during high-level plan creation. For evaluation, five metrics have been extracted from the video recordings: (i) the accuracy of the control, (ii) the time it took the subjects to execute a high-level plan, (iii) the number of steps used to execute a high-level plan, (iv) the path optimality, i.e., the ratio of the steps used to the minimally possible number of steps, and (v) the average time per step. We summarized the results in Table I. In total, a 76.67 % correct BCI control was achieved, which required 9 s per step. Selecting a plan using the GUI took on average 148 s and required the user to perform on average 16.74 steps in the GUI of the high-level planner. The path formed by these steps is on average 34.6 % away from the optimal path. The decoding accuracy of every subject is significantly above chance ($p < 10^{-6}$).

The subject-averaged EEG data used to train the hybrid ConvNets and the decoding results of the train/test transfer are visualized in Fig. 3. In Fig. 3(a) we show the signal-to-noise ratio (SNR) of all 5 classes $\mathcal{C}$ of the labeled datasets. We define the SNR for a given frequency f, time t and electrode e as

$$\mathrm{SNR}_{f,t,e} = \frac{\mathrm{IQR}\left(\{\mathrm{median}\left(\mathcal{M}_i\right)\}\right)}{\mathrm{median}\left(\{\mathrm{IQR}\left(\mathcal{M}_i\right)\}\right)} \quad i \in \mathcal{C},$$

where $\mathcal{M}_i$ corresponds to the set of values at position (f, t, e) of the i-th task, with $|\mathcal{M}_i|$ being the number of repetitions. median$(\cdot)$ and IQR$(\cdot)$ is the median and interquartile range (IQR), respectively. The upper part describes the variance of the class medians, i.e., a large variance means more distinguishable class clusters and a higher SNR. The denominator describes the variance of values in each class, i.e., a lower variance of values results in a higher SNR. The low SNR in EMG channels shows that the subjects did not move during the tasks.

The decoding accuracies achieved on the test data after initial training of the ConvNets are visualized in Fig. 3(b). To further support the neural origin of the BCI control signals, Fig. 3(c) shows physiologically plausible input-perturbation network-prediction correlation results (see [13] for methods).

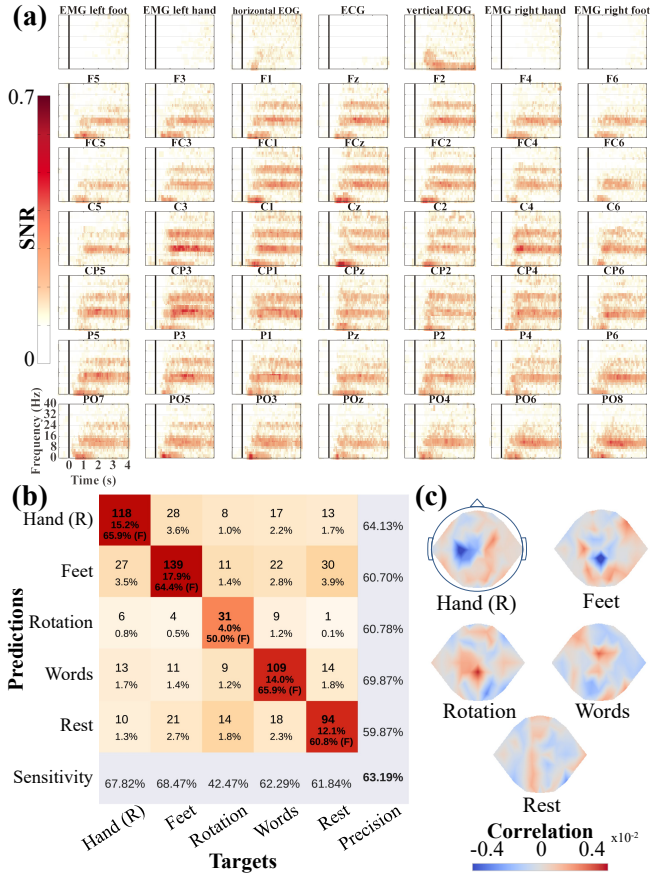

Fig. 3. EEG data and decoding results. **(a)** SNR of the first 4 s of data used to train the hybrid ConvNet. Highest SNR can be observed in the alpha (7-14 Hz) and lower beta (16-26 Hz) bands. These frequency bands are robust markers of task related mental tasks. Note that the non-EEG channels (top row) were withheld from the ConvNets at any time and are displayed as negative control. Not all channels are displayed because of space constraints. **(b)** Confusion matrix of decoding accuracies for the train/test transfer. Accuracies are well above the theoretical chance level of 20 %. **(c)** Topographically plausible input-perturbation network-prediction correlation maps in the alpha (7-13 Hz) frequency band. For details on the visualization technique we refer the reader to [13].

B. Fetch and Carry Task

The first experiment, considering the use of the real robot, evaluates the complete system in fetch-and-carry tasks. The goal was to transfer an object from one location to another, e.g., from a shelf to the table, using the robot. To fulfill such tasks the robot typically needs to execute four subtasks: *approach* object location, *grasp* object, *approach* other location, *drop* object. The user was instructed to select a predefined goal using the EEG-controlled high-level planner. Moreover, we selected a random initial placement for the objects in each run, in order to cover different environment states. The experiment was repeated ten times by the user. Table II shows the averaged results for the experiment. The second column indicates the overall number of desired action calls, as scheduled by the high-level planner, as well as the number of calls actually performed. The third to fifth columns represent the success rate, mean and standard deviation for the runtime of actions, respectively. Note, that the number of scheduled and actually executed actions

TABLE II

AGGREGATED RESULTS FOR 10 RUNS (EXP. VII-B)

Actions	# Executions (# Scheduled)	Success Executions [%]	Runtime [s] Mean	Std
Grasp	10 (10)	90.0	37.56	4.62
Drop	9 (10)	89.0	34.13	5.75
Approach	19 (20)	100.00	33.05	18.48
Total	38 (40)	94.74	34.42	14.02

TABLE III

AGGREGATED RESULTS FOR 10 RUNS (EXP. VII-C)

Actions	# Executions (# Scheduled)	Success Executions [%]	Runtime [s] Mean	Std
Grasp	34 (30)	91.0	40.42	10.31
Drop	30 (30)	97.0	37.59	4.83
Approach	80 (80)	100.0	20.91	7.68
Pour	10 (10)	100.0	62.90	7.19
Drink	13 (10)	77.0	57.10	8.20
Total	167 (160)	95.86	32.46	15.51

might differ for two reasons. A number of executed calls, lower than the scheduled ones, indicates that a previous action step has failed to succeed and plan recovery was not possible. On the other hand, a higher number of executed calls indicates that the user was able to achieve plan recovery by commanding a repetition of the failed action. Moreover, we recorded the largest standard deviation for the *approach* action, which can be attributed to the diverse complexity of the planning problem for the mobile base and the distance to travel between the selected grasp and drop location. In total, our system achieved a success rate of 80% for the entire task. Planning and execution required on average 140.63±36.7 s. Errors were mainly caused by object detection issues, i.e., the system was not able to detect the object or the detection was not precise enough to be able to successfully grasp or drop an object.

C. Drinking Task

The last experiment evaluates the direct interaction between user and robot. Therefore, we implemented an autonomous robotic drinking assistant. Our approach enables the robot to fill a cup with a liquid, move the robot to the user and finally provide the drink to the user by execution of the corresponding drinking motion in front of the user's mouth. In addition to the techniques described above, successful pouring and drinking using a robot requires the detection of the liquid level in the cup and a reliable detection and localization of the user's mouth.

To detect the liquid level while pouring, we follow a vision-based approach introduced by Do *et al.* [24]. Given the camera's viewing angle and the liquid's index of refraction, the liquid height is determined from the depth measurement using a relationship based on Snell's law (see [25] for more details). Using this knowledge, we first detect the cup, extract the depth values for the liquid and finally estimate the real liquid height. The type of liquid and hence the index of refraction is assumed to be given beforehand. The viewing

angle can be determined from the depth data. A Kalman filter is then used to track the liquid level and compensate for noise. Once it is detected that the liquid level has exceeded a user defined value, a stop signal is sent to terminate the pouring motion.

For detection and localizing of the user's mouth, we adopt a two step approach. In the first step, we segment the image based on the output of a face detection algorithm in order to extract the image region containing the user's mouth and eyes. Afterwards, we detect the position of the mouth of the user, considering only the obtained image patch. Regarding the mouth orientation, we additionally consider the position of the eyes in order to obtain a more robust estimation of the face orientation, hence compensating for slightly changing angles of the head. The face, mouth and eye detectors are implemented in OpenCV by applying an algorithm that uses Haar cascades [26], [27].

Table III shows the averaged results for the experiment. Here, only 3.75% of the 160 scheduled actions had to be repeated in order to complete the task successfully. In one run, plan recovery was not possible leading to abortion of the task. Thus, our system achieved in total a success rate of 90% for the drinking task. Planning and execution required on average 545.56 ± 67.38 s. For the evaluation of the liquid level detection approach, we specified a desired fill level and executed 10 runs of the pour action. The resulting mean error and standard deviation is 6.9 ± 8.9 mm. In some instances the bottle obstructed the camera view, resulting in poor liquid level detection and a higher error.

VIII. Conclusions

In this paper, we presented a thought-controlled mobile robotic service assistant, capable of successfully performing complex tasks, including close range interaction with the user, in a continuously changing environment to increase the independence of severely paralyzed patients. Through the use of a high-level planner as an intermediate layer between user and autonomous mobile robotic service assistant, we overcome the curse of dimensionality typically encountered in non-invasive BCI control schemes, thus opening up new perspectives for human-robot interaction scenarios.

References

[1] D. Park, Y. K. Kim, Z. M. Erickson, and C. C. Kemp, "Towards assistive feeding with a general-purpose mobile manipulator," *arXiv:1605.07996*, 2016.

[2] C.-S. Chung, H. Wang, and R. A. Cooper, "Autonomous function of wheelchair-mounted robotic manipulators to perform daily activities," in *Rehabilitation Robotics (ICORR), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1–6.

[3] F. Achic, J. Montero, C. Penaloza, and F. Cuellar, "Hybrid bci system to operate an electric wheelchair and a robotic arm for navigation and manipulation tasks," in *Advanced Robotics and its Social Impacts (ARSO), 2016 IEEE Workshop on*. IEEE, 2016, pp. 249–254.

[4] C. Wang, B. Xia, J. Li, W. Yang, D. Xiao, A. C. Velez, and H. Yang, "Motor imagery bci-based robot arm system," in *Natural Computation (ICNC), 2011 Seventh International Conference on*, vol. 1. IEEE, 2011, pp. 181–184.

[5] S. Schröer, I. Killmann, B. Frank, M. Völker, L. D. J. Fiederer, T. Ball, and W. Burgard, "An autonomous robotic assistant for drinking," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 6482–6487.

[6] K. Muelling, A. Venkatraman, J.-S. Valois, J. E. Downey, J. Weiss, S. Javdani, M. Hebert, A. B. Schwartz, J. L. Collinger, and J. A. Bagnell, "Autonomy infused teleoperation with application to brain computer interface controlled manipulation," *Autonomous Robots*, pp. 1–22, 2017.

[7] 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.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *arXiv:1512.03385*, 2015.

[9] T. N. Sainath, B. Kingsbury, G. Saon, H. Soltau, A.-r. Mohamed, G. Dahl, and B. Ramabhadran, "Deep Convolutional Neural Networks for Large-scale Speech Tasks," *Neural Networks*, vol. 64, pp. 39–48, 2015.

[10] T. Sercu, C. Puhrsch, B. Kingsbury, and Y. LeCun, "Very deep multilingual convolutional neural networks for LVCSR," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 4955–4959.

[11] V. J. Lawhern, A. J. Solon, N. R. Waytowich, S. M. Gordon, C. P. Hung, and B. J. Lance, "Eegnet: A compact convolutional network for eeg-based brain-computer interfaces," *CoRR*, vol. abs/1611.08024, 2016.

[12] Y. R. Tabar and U. Halici, "A novel deep learning approach for classification of EEG motor imagery signals," *Journal of Neural Engineering*, vol. 14, no. 1, p. 016003, 2017.

[13] R. T. Schirrmeister, J. T. Springenberg, L. D. J. Fiederer, M. Glasstetter, K. Eggensperger, M. Tangermann, F. Hutter, W. Burgard, and T. Ball, "Deep learning with convolutional neural networks for brain mapping and decoding of movement-related information from the human eeg," *arXiv:1703.05051*, 2017.

[14] K. K. Ang, Z. Y. Chin, H. Zhang, and C. Guan, "Filter Bank Common Spatial Pattern (FBCSP) in Brain-Computer Interface," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2008, pp. 2390–2397.

[15] P. Bashivan, I. Rish, M. Yeasin, and N. Codella, "Learning Representations from EEG with Deep Recurrent-Convolutional Neural Networks," in *arXiv: 1511.06448*, 2016.

[16] S. Stober, "Learning Discriminative Features from Electroencephalography Recordings by Encoding Similarity Constraints," in *Bernstein Conference 2016*, 2016.

[17] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)," in *ArXiv e-prints*, vol. 1511, 2016, p. arXiv:1511.07289.

[18] R. Dale and E. Reiter, "Computational interpretations of the gricean maxims in the generation of referring expressions," *Cognitive science*, vol. 19, no. 2, pp. 233–263, 1995.

[19] M. Göbelbecker, "Assisting with Goal Formulation for Domain Independent Planning," in *KI 2015: Advances in Artificial Intelligence*. Springer, 2015, pp. 87–99.

[20] F. Burget, M. Bennewitz, and W. Burgard, "BI^{2}RRT*: An efficient sampling-based path planning framework for task-constrained mobile manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, Korea, 2016.

[21] L. E. Kavraki and J.-C. Latombe, *Probabilistic Roadmaps for Robot Path Planning*. John Wiley, 1998, pp. 33–53.

[22] M. Helmert, "The Fast Downward Planning System," *Journal of Artificial Intelligence Research 26 (JAIR 2006)*, pp. 191–246, 2006.

[23] K. Pauwels and D. Kragic, "Simtrack: A simulation-based framework for scalable real-time object pose detection and tracking," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 1300–1307.

[24] C. Do, T. Schubert, and W. Burgard, "A probabilistic approach to liquid level detection in cups using an RGB-D camera," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, Korea, 2016.

[25] Y. Hara, F. Honda, T. Tsubouchi, and A. Ohya, "Detection of Liquids in Cups Based on the Refraction of Light with a Depth Camera Using Triangulation," in *IROS*, 2014.

[26] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001, pp. I–I.

[27] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid object detection," in *Image Processing. 2002. Proceedings. 2002 International Conference on*, vol. 1. IEEE, 2002, pp. I–I.