Goal-recognition-based adaptive brain-computer interface for navigating immersive robotic systems

Mohammad Abu-Alqumsan[1], Felix Ebert[2] and Angelika Peer[3]

[1] Chair of Automatic Control Engineering, Technical University of Munich (TUM), Munich, Germany
[2] Institute of Autonomous Systems Technology, University of the Bundeswehr, Munich, Germany
[3] Bristol Robotics Laboratory, University of the West of England, Bristol, UK

E-mail: moh.marwan@lsr.ei.tum.de

Abstract. *Objective.* This work proposes principled strategies for self-adaptations in EEG-based Brain-computer interfaces (BCIs) as a way out of the bandwidth bottleneck resulting from the considerable mismatch between the low-bandwidth interface and the bandwidth-hungry application, and a way to enable fluent and intuitive interaction in embodiment systems. The main focus is laid upon inferring the hidden target goals of users while navigating in a remote environment as a basis for possible adaptations. *Approach.* To reason about possible user goals, a general user-agnostic Bayesian update rule is devised to be recursively applied upon the arrival of evidences, i.e. user input and user gaze. Experiments were conducted with healthy subjects within robotic embodiment settings to evaluate the proposed method. These experiments varied along three factors: the type of the robot/environment (simulated and physical), the type of the interface (keyboard or BCI), and the way goal recognition (GR) is used to guide a simple shared control (SC) driving scheme. *Main results.* Our results show that the proposed GR algorithm is able to track and infer the hidden user goals with relatively high precision and recall. Further, the realized SC driving scheme benefits from the output of the GR system and is able to reduce the user effort needed to accomplish the assigned tasks. Despite the fact that the BCI requires higher effort compared to the keyboard conditions, most subjects were able to complete the assigned tasks, and the proposed GR system is additionally shown able to handle the uncertainty in user input during SSVEP-based interaction. The SC application of the belief vector indicates that the benefits of the GR module are more pronounced for BCIs, compared to the keyboard interface. *Significance.* Being based on intuitive heuristics that model the behavior of the general population during the execution of navigation tasks, the proposed GR method can be used without prior tuning for the individual users. The proposed methods can be easily integrated in devising more advanced SC schemes and/or strategies for automatic BCI self-adaptations.

Keywords: Adaptive BCI, SSVEP, Intention Recognition, Shared Control, Robotic Embodiment

Submitted to: *J. Neural Eng.*

1. Introduction

Brain-computer interfaces (BCIs) are *direct* communication and control channels between the brain and artificial devices like computers, prosthetic limbs or robots. The *directness* here refers to the fact that BCIs bypass the natural neural (outside the brain) and muscular pathways [1] which are required for all other kinds of human-human and human-machine interaction. Typically in BCIs, the users' brain signals are decoded into machine actions using a mapping that is known to both the users and the devices they control or communicate with.

Recently, there has been growing interest in deploying BCIs in immersive robotic embodiment systems [2–7], whereby the objective is to allow users to exercise physical and social interaction in a remote environment through robotic avatars, which are typically equipped with locomotion and manipulation capabilities in order to allow for enhanced interaction. Hereby and throughout this work, the terms *immersion* and *embodiment* are used similarly to Slater and Sanchez-Vives [8]. That is, *immersion* is used to refer to the objective description of the system technicalities that enable the users' perception of the remote environment based on their natural sensorimotor contingencies (e.g. head tracking). On the other hand, *robotic embodiment* is used to define the process of replacing the user's body by a robotic avatar, where one speaks of the senses of self-location, agency and body ownership as subcomponents of embodiment [9].

Fig. 1 depicts a schematic of a closed loop BCI-based immersive robotic embodiment system. Users continuously receive perceptual feedback from their robot avatar embedded in the remote environment and communicate their intentions by selectively attending to (or gazing at) one of the available interface elements. Other BCI paradigms like motor imagery (MI) are possible [5], but we are mostly concerned in this work with BCIs based on selective attention, e.g. P300 and SSVEP-based BCIs. Ultimately, in order to fully immerse users in the remote environment, the perceptual channels from the local environment should be replaced by the ones that reflect the state of the avatar in the remote environment. In this regard, vision is the most important perceptual modality, but other modalities undoubtedly would enhance the user subjective experience of immersion and embodiment. The robot, on the other side, continuously receives user commands and translates them using available low and high level controllers into robotic actions, that change the state of the environment or of the robot itself.

Generally speaking, at each time instant during BCI-based interaction, only a limited number of interface elements can fit into the interface, whether that be based on P300, SSVEP or MI and consequently, only a few commands will be available to users. This gives rise to the primary challenge in robotic applications of BCIs, that is, the considerable mismatch between the low-bandwidth interface and the bandwidth-hungry application. Hierarchical menu style BCI and adaptive BCIs offer a way to allow for an extended set of robotic actions and thus to overcome the bandwidth mismatch.

Interface self-adaptation is expected to bring savings in time and in the user effort needed to arrive at the commands of interest. Savings can be computed for example as the difference between the cost of selecting a command through the adaptive interface (C_a) and the cost of selecting the same command with a deterministic hierarchical interface (C_h) [10]. C_a can be computed from the time needed to visually scan the predicted interface elements till the element of interest is found plus the required time for selecting that specific element. C_h, on the other hand, is computed from summation of the cognitive time needed to formulate the sequence of interface selections, and the time needed to make these selections. The design of adaptive user interfaces (AUIs) is generally a highly challenging task due to the fact that interface self-adaptations need to (1) be unobtrusive to users (2) keep the users in control of the system (3) allow the user to maintain a coherent mental model of the interface. Adaptive BCIs inherit all these challenges, but given the limited bandwidth of the interface, self-adaptation is a necessity, rather than an extra feature.

In order for interface self-adaptations to be of any benefit, the system should be able to infer the hidden goals, which the user tries to achieve in the domain. This way, initiatives made by the interface can be received positively by the user. The problem of inferring user goals from observations is typically referred to as intention or goal recognition, and is the main concern of this work. By considering the general problem of navigation in BCI-based immersive robotic embodiment systems, a general user-agnostic Bayesian framework for goal recognition is proposed. Hereby, novel intuitive heuristics are used to model

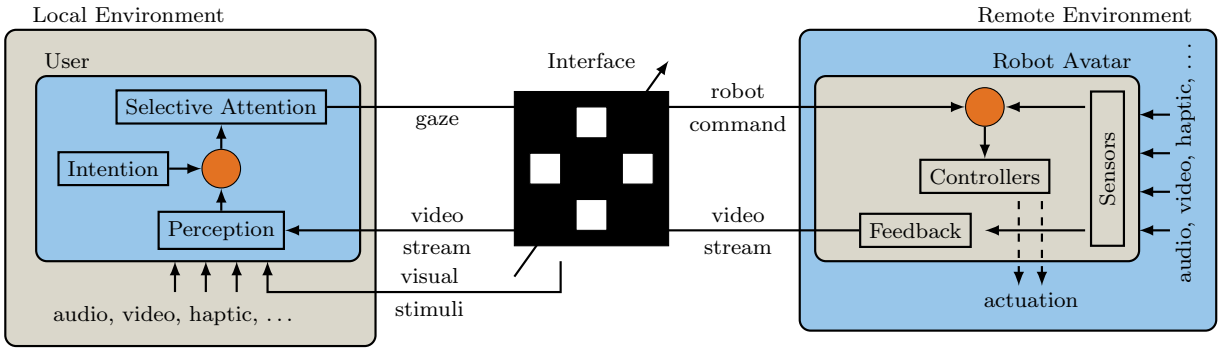


Figure 1: Closed loop BCI-based robotic embodiment system. Users communicate their intentions by selectively attending to one of the interface elements, which are typically shown overlaid on the visual feedback arriving from the remote environment. Self-adaptations in the BCI allow for extended set of actions available to the users. An SSVEP-based interface is shown as an example here.

the user behavior, on the basis of general behavioral patterns that are observed in humans during task execution. The Bayesian framework is designed for the general population allowing to infer user intentions, without any prior training. The output of the Bayesian inference module is a belief vector that sorts all target goals in the environment according to how probable they are given the observed evidence. Moreover, a novel metric that measures the non-uniformness of the belief vector is proposed as to reflect upon the confidence of the inference system in its computed beliefs. To show the usefulness of the computed beliefs and confidence measures, a simple probabilistic shared control scheme is devised so that adaptations are applied to the robot movements, according to the belief/confidence information. In order to evaluate the performance of the designed Bayesian inference module, experiments were conducted with healthy subjects in an immersive robotic embodiment setup and in simulation. Results show that the intention recognition system is able to track the hidden user goals with relatively high accuracy. When the belief of the intention recognition module is used to modulate parameters of the robot movement, less user effort (measured by the number of interaction) is required to accomplish the assigned tasks.

This paper proceeds as follows. Sec. 2 provides background information about adaptive interfaces in the context of robotic navigation, to motivate the work on intention and goal recognition. Sec. 3 provides a short review on state-of-the-art approaches to intention recognition for robotic navigation tasks. Our Bayesian goal inference systems is introduced in Sec. 4. Detailed information about the experiments used to evaluate the proposed methods is described in Sec. 5. Experimental results are reported in Sec. 6 followed by a discussion in Sec. 7. This paper concludes in Sec. 8.

2. Background and Motivation

User commands in BCI-based robotic navigation applications range from commanding the mobile base to move for some distance in one direction, to commanding the mobile base to move to a specific location or to autonomously perform complex maneuvers like “move to the kitchen” or “walk through the doorway”. Obviously, the varying degrees in the goal-directness of incoming user commands require adequate levels of autonomy at the robot side. And conversely, by deciding first on the level of autonomy, adaptive interfaces can suggest conforming sets of commands for interaction. The exact relation between the interface and the level of autonomy, therefore, can be decided upon either by the interface itself as in [11] or by the user as in [12, 13]. Either way, the higher the robot autonomy, the less control is left to the user. In a general sense, robot autonomy is thought of here as a continuum where the fully autonomous and fully manual control modes are its extremes. Often the selection of the optimal point on the autonomy continuum is done in two steps. First, the continuum is discretized into different general levels (or modes) [14, 15] and the appropriate mode is selected. The optimal point of automation is then selected within that mode. In the context of robotic navigation applications, we borrow from [16] and adopt the 6 basic levels or robot operation modes shown in Fig. 2.

In *goal-oriented control* mode, users only communicate the end goal location they have in mind, and the robot autonomously navigates towards it. This requires global knowledge of the environmental map and path planning capabilities at the robot side. In the *help with some tasks* mode, like walking through a door, the robot needs to plan a safe local trajectory, by which it remains e.g. equidistant from the doorjambs. In the

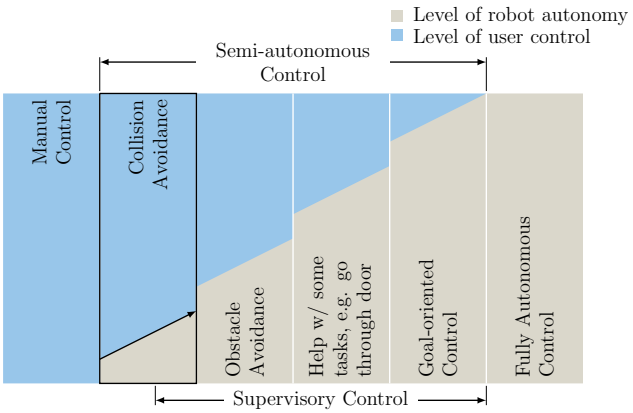


Figure 2: Robot operational modes for navigation tasks. The level of automation can be modulated within each mode of operation as well.

obstacle avoidance mode, it is the robot's job to execute maneuvers that avoid obstacles while the human user is issuing commands that move it through the environment. In the *collision avoidance* mode, a typical behavior of the robot is to halt movement in the face of obstacles while the user is commanding movements.

As hinted previously, the different automation levels define the nature of interface self-adaptations. For instance, in the *goal-oriented* mode, the interface might provide the user with the most probable next goals (or goal locations) in the environment. Such predictions can be performed on the basis of the transition probabilities between the different goals in the environment, which can be learned e.g. from the history of user interaction with the system. In absence of any knowledge about the user preferences in the domain, other interface adaptation strategies should be devised. One possible approach, is to allow users to interact with the robot in a lower autonomy mode, so that some evidence can be gathered throughout interaction and used to predict the target goals of the user.

In this work, we mainly focus on the problem of goal recognition during interaction within the collision avoidance mode, where user input is additionally restricted to incremental commands that translate or rotate the robot in the different directions with predefined steps. The reason for this is threefold. First, it has been noted in [17], that powered wheelchair users want to actively drive the wheelchair rather than being merely its passengers. The same can be said about the users of teleoperation/embodiment robots, and therefore, it is imperative to give users the sense of control they need [18], as much as can be allowed by other existing constraints like the safety of the human, the machine and the environment, the capabilities of the robot and the abilities/disabilities of the human users. The collision avoidance mode leaves most of the control in the hands (or rather the brains) of the users, and at the same time guarantees navigation safety. Second, and most importantly, it allows us to develop the methodologies, by which gathered evidence from user commands in low autonomy levels can guide the interface self-adaptations in the higher autonomy modes. Third, with long term interaction with the robot, transition probabilities can be learned, and used later to guide interface adaptations in higher autonomy modes. The next section will provide a brief review on the problem of goal/intention recognition in general, with some emphasis on intention recognition in navigation tasks.

3. Related Work: Goal/Intention Recognition

The term *plan recognition* has been defined by Schmidt et al. [19] as the process of inferring an agent's goals from observing the actions the agent is performing in the domain and organizing these actions into a plan structure which explicitly describes the goal-subgoal relations among them. Conformingly, recent research on human action understanding within navigation contexts shows that a machine equipped with inverse planning is able to efficiently model this cognitive process [20]. We refer interchangeably to the agent whose actions are observed as the *actor* or the *user*, whereas the observing agent that makes the inference is referred to as the *recognizer*.

The *goals* refer to desired states of the world or states of knowledge about the world. In order to arrive at these states, an actor needs to perform a sequence of actions that defines a *plan*. *Actions* taken might change the state of the world, or the state of the actor's knowledge about the world. Actions are defined by their *preconditions*, i.e. possible states in which they can take place, and *effects* that describe the state transition when the action is performed. In some domains, it is possible to enumerate all possible goal states and plans an actor might have. The set of these plans is referred to as *plan library*.

The problems of plan and goal recognition have many details in common. Systems which try to infer the final target state only are referred to as *goal recognition* (GR) systems, and those which additionally predict the sequence of actions which moves the world from its current state to the desired final state are referred to as *plan recognition* systems. In navigation applications, these refer respectively to inferring the final goal location the user has in mind and the full trajectories, with which the user wants to arrive at these target locations. Depending on the level of abstraction used in recognition, both systems might

be referred to as *intention recognition* systems †.

Intention recognition systems can be classified with respect to the actor's attitude towards the recognizer into three categories. First, *the keyhole recognition* refers, as the term suggests, to the case when the actor is not aware of the existence of the recognizer [21, 22]. In the second category, which is referred to as *intended recognition*, the actor cooperatively chooses actions which reduce the ambiguity about his/her hidden intentions [23]. Finally, *adversarial recognition* concerns itself with the case when the actors actively try to confuse the recognizer by choosing misleading actions [22, 24].

Charniak and Goldman [25] argue that the problem of intention recognition is largely a problem of inference under conditions of uncertainty, rendering the probabilistic approaches of best fit. Such approaches introduce a numerical measure of belief that reflects how likely or probable individual plans are and can thus, explicitly represent the uncertainty associated [26]. The vector that contains the probability of all plans (or goals) is often referred to as the *belief vector*. Bayesian networks, and in particular, dynamic Bayesian networks (DBNs) are probably the mostly used probabilistic method for intention recognition in different domains, e.g. for predicting user plans in a game [21].

In robotic navigation contexts, Perrin et al. [27] use a DBN for goal recognition, in a system, with which the user interacts by either confirming or rejecting its propositions. The proposed Bayesian network additionally includes variables that integrate the time of the day, whether the phone is ringing and the previously visited goals. The conditional probability distributions (CPDs) used in the network are learned from training sequences. Alternatively, goal recognition is realized in [17] on the basis of simple metrics that are computed from the distances and relative orientations the wheelchair has to available goal locations (available doors in this case). On the other hand, several approaches have regarded the goal/plan recognition problem as a planning problem [28–30]. The main idea here is that, instead of enumerating all the plans in the domain, which is not always feasible, a general planner can be used to generate the possible plans to all available target states. Along these concepts, the works in [31, 32] use a recursive Bayesian update for plan recognition based on environmental data, global path planning, and history of interaction. The Bayesian update hereby is defined with

$$
\begin{aligned}
P_k(\mathbf{i}_{k-m:k} \mid u_{k-m:k}) = & && \text{(posterior)}\\
& P(u_k \mid \mathbf{i}_{k-m:k}, u_{k-m:k-1}) && \text{(user model)}\\
& P(\mathbf{i}_k \mid \mathbf{i}_{k-m:k-1}, u_{k-m:k-1}) && \text{(plan process model)}\\
& P_{k-1}(\mathbf{i}_{k-m:k-1} \mid u_{k-m:k-1}) && \text{(prior)}\\
& \eta, && \text{(normalization)}
\end{aligned}
\tag{1}
$$

where $\mathbf{i}_k$ is the user mental plan to move from the current location to the target location in mind, u_k is the user input at time instant k, $u_{k-m:k}$ is the sequence of the user input from time instant $k-m$ to k, and m defines the past time instances that influence the plan and the user input at any time. Hereby, for instance, the user model defines the likelihood of the user input given that the user has the plan evolution $\mathbf{i}_{k-m:k}$ and issued previous commands $u_{k-m:k-1}$. The user and plan process models are defined differently for a variety of systems and inputs. For a BCI-based input and free space area [31], the user model is defined with a simple heuristic function that takes into consideration the relative orientations of the robot to the different goals and the distances to each of them. The plan process model is computed such that when the robot moves from the pose $\mathbf{x}_{k-1}$ to $\mathbf{x}_k$, the straight path between $\mathbf{x}_{k-1}$ and the j^{th} goal is transformed into the straight path between $\mathbf{x}_k$ and the j^{th} goal, and the j^{th} probability is transferred to the new path. More detailed user and plan process models are proposed in [32] for joysticks and deterministic discrete interfaces for different kinds of environments.

The different approaches to intention recognition in the previous short review make full or partial use of available information about the environment and the history of interaction to arrive at good estimates about the user's hidden goals or plans. The Bayesian approach allows to model this information in a sparse representation, and yet achieves reliable estimates. We conjecture that, in navigation tasks with discrete interfaces, the information that can be obtained with goal recognition (rather than the full plan recognition as used in [31, 32]) is sufficient to guide interface adaptations. Our approach to the goal recognition problem, as will become clearer later, uses computed optimal or suboptimal path plans to all target goals as extra evidence regarding the probability of these goals. Similar to different approaches in plan and goal recognition [28–32], these plans are computed with off-the-shelf classical planners. The major difference in this respect between the proposed goal recognition method and similar plan recognition methods [31, 32] is that in the former only a single plan is computed per target goal, whereas in the latter, individual goals might have different global path plans (whose probabilities need to be inferred). Furthermore, in comparison to the approach in [27],

† Other abstraction levels exist. For instance, the user might want to move to the kitchen to prepare a meal, and this very objective of the user can be considered the user intention.

we aim at goal recognition systems that require no training whatsoever, so that users can benefit from the recognition system in their first use. On the other hand, straightforward methods like [17] may not be able to exploit useful available information, e.g. the previous user actions towards the target goal.

4. Methods: Bayesian Framework to Goal Recognition in Navigation Tasks

4.1. Problem Statement

The objective of the goal recognition module is to infer users' target goals throughout interaction, where evidence is gathered from observed users' commands and their directions of gaze. It is assumed here that the users are unaware of the existence of the recognizer, i.e. keyhole intention recognition.

We denote the set of enumerable goals in the environment with $\mathcal{G} = \{{}^1\mathbf{g}^W, {}^2\mathbf{g}^W, \cdots, {}^n\mathbf{g}^W\}$, where each goal is defined by its pose with respect to the global coordinate system of the available map (W), such that ${}^m\mathbf{g}^W = [{}^mx^W, {}^my^W, {}^mz^W, {}^m\theta^W]^T$, where $m \in \{1, 2, \cdots, n\}$ and ${}^m\theta^W \in [-\pi, \pi]$ is the smallest angle between the x-coordinate of the m^{th} goal frame and that of the global frame (W), where the z-coordinate of all goals is parallel to that of the world frame. We assume that the number (n) and the poses of these goals, which might represent the location of salient objects the user frequently uses, are accessible by the goal recognition system. For the sake of simplicity in the notation, the postscript W is dropped when we refer to goals and their poses in the global coordinate system. The unknown target goal, which the user has in mind, is defined here as a discrete random variable G with a sample space $\mathcal{G}$.

While having a goal location ${}^m\mathbf{g}$ in mind at time instant k, the user updates the mental path plan ${}^m\mathbf{i}_k$ of how to arrive there from the robot's current pose $\mathbf{x}_k = [x_k, y_k, \theta_k]^T$. We write this in the form, ${}^m\mathbf{i}_k = \mathbf{x}_k \rightarrow {}^m\mathbf{g}$. Additionally, we define the length of a path plan (in meters) with $l({}^m\mathbf{i}_k) = {}^ml_k$. In order to follow the plan in mind, the user issues a command $u_k \in \mathcal{U}$. In this work, we only consider incremental commands that can be issued with discrete interfaces (e.g. keyboard or SSVEP-BCI), and therefore, the set of all possible commands is enumerable, i.e. $\mathcal{U} =$ {move forward, move backward, move left, move right, turn left, turn right, stop}. User commands are translated into robot actions a_k that change the state of the robot from $\mathbf{x}_k$ to $\mathbf{x}_{k+1}$. The change in the robot pose that is triggered by translational and rotational commands is set respectively to the default values δd_{def} (m/command) and $\delta\theta_{\text{def}}$ (rad/command).

In order to make the math easier to follow, we introduce the dummy user command u_0, which implies $\mathbf{x}_0 = \mathbf{x}_1$ and ${}^m\mathbf{i}_0 = {}^m\mathbf{i}_1$ for all m. The user's gaze direction at time instant k is denoted by $\mathbf{h}_k$, which we assume here to be defined with the rotational angles of the gaze direction relative to an arbitrary reference frame.

We assume that the localization module provides a reliable estimate of $\mathbf{x}_k$, and that the mental path plans to all goals (i.e. $\mathbf{x}_k \rightarrow {}^m\mathbf{g}$) of the user can be estimated reliably with a global path planner on the basis of the 2D cost map of the environment, denoted by $\mathbf{M}_k$. Implicitly, with this assumption we hypothesize that users in the navigation domain act approximately optimally (rationality assumption [20]) and try to follow the path that minimizes some cost function. The global path planner is expected to find paths similar to the ones the user plans for the different goals.

Therefore, we can assume that at time instant k, the GR module has access to the following information

- The observed sequence of user commands up to time instant k, which is denoted by $u_{0:k} = (u_0, \cdots, u_k)$.
- The observed sequence of user gaze up to time k, which is denoted by $\mathbf{h}_{0:k} = (\mathbf{h}_0, \cdots, \mathbf{h}_k)$.
- The sequence of plans computed to all goals up to time instant k, denoted by ${}^{1:n}\mathbf{i}_{0:k} = ({}^1\mathbf{i}_{0:k}, \dots {}^n\mathbf{i}_{0:k})$, where ${}^m\mathbf{i}_{0:k} = (\mathbf{x}_0 \rightarrow {}^m\mathbf{g}, \cdots, \mathbf{x}_k \rightarrow {}^m\mathbf{g})$.

The problem of goal recognition can be then formally defined as estimating the probability

$$ {}^mP_k = P_k(G = {}^m\mathbf{g} \mid u_{0:k}, \mathbf{h}_{0:k}, {}^{1:n}\mathbf{i}_{0:k}), \forall m, \qquad (2) $$

where $\sum_{m=1}^{n} {}^mP_k = 1$ for all k. The probabilities of all goals can be concatenated in the n-dimensional vector $\mathbf{P}_k$ which encodes the system belief about the user's hidden goal at time instant k, where $\mathbf{P}_k = [{}^1P_k, {}^2P_k, \cdots, {}^nP_k]^T$. In this work, $\mathbf{P}_k$ is referred to as the *belief vector*.

We assume that before the user starts navigating in the remote environment, the GR module has no prior knowledge about the next pursued goal, and therefore ${}^mP_0 = 1/n, \forall m$. Additionally, the subscript k is reset to 0 each time the robot arrives at one of the available goals, which automatically resets the belief vector to a near-uniform distribution, wherein the last visited goal is assigned a smaller probability relative to other available goals, i.e. ${}^mP_0 = 0.1/n$ if goal m was the last visited goal and ${}^iP_0 = 1/n, \forall i \neq m$. The belief vector $\mathbf{P}_0$ is then normalized to obey probability axioms.

4.2. Bayesian Update Rule

The objective of this section is to devise a recursive Bayesian update rule for the belief about user goals

based on available information to the GR module. We begin by assuming that the recognizer has access to the true user input u_k, e.g. keyboard-based interface, but later, the update rule is extended to noisy interfaces, e.g. BCIs. At a later point in this section, it is further discussed how to deal with the fact that the user commands and information about the user's gaze typically arrive asynchronously.

The conditional probability of a goal m given previous observations can be computed recursively according to

$$
\begin{aligned}
&P_k({}^m\mathbf{g} \mid u_{0:k}, \mathbf{h}_{0:k}, {}^{1:n}\mathbf{i}_{0:k}) = && \text{(posterior)}\\
&\quad P(u_k \mid {}^m\mathbf{g}, u_{0:k-1}, \mathbf{h}_{0:k}, {}^{1:n}\mathbf{i}_{0:k}) && \text{(user input model)}\\
&\quad P(\mathbf{h}_k \mid {}^m\mathbf{g}, u_{0:k-1}, \mathbf{h}_{0:k-1}, {}^{1:n}\mathbf{i}_{0:k}) && \text{(user gaze model)}\\
&\quad P({}^{1:n}\mathbf{i}_k \mid {}^m\mathbf{g}, u_{0:k-1}, \mathbf{h}_{0:k-1}, {}^{1:n}\mathbf{i}_{0:k-1}) && \text{(plans evolution model)}\\
&\quad P_{k-1}({}^m\mathbf{g} \mid u_{0:k-1}, \mathbf{h}_{0:k-1}, {}^{1:n}\mathbf{i}_{0:k-1}) && \text{(prior)}\\
&\quad \eta, && \text{(normalization)}
\end{aligned}
\tag{3}
$$

where η is a normalization factor which guarantees $\sum_{m=1}^{n} P_k({}^m\mathbf{g} \mid u_{0:k}, \mathbf{h}_{0:k}, {}^{1:n}\mathbf{i}_{0:k}) = 1$ for all k. The major difference between the two rules in (3) and (1), is that ${}^{1:n}\mathbf{i}_{0:k}$ is assumed known for all possible goals in (3), whereas in (1), the computation of their probabilities is the target of the update rule.

In the following, generic and user-agnostic models for the user input, the user gaze and the path plans evolution will be proposed.

4.2.1. User Input Model

The term $P(u_k \mid {}^m\mathbf{g}, u_{0:k-1}, \mathbf{h}_{0:k}, {}^{1:n}\mathbf{i}_{0:k})$ models the likelihood a user issues a command u_k at time k, while having the goal ${}^m\mathbf{g}$ in mind and given the sequence of current and previous path plans to all goals ${}^{1:n}\mathbf{i}_{0:k}$, the sequence of previously issued commands $u_{0:k-1}$ and the sequence of user gaze $\mathbf{h}_{0:k}$ up to time instant k. We assume that the user input u_k is conditionally independent of $u_{0:k-1}, \mathbf{h}_{0:k}$ and ${}^{m'}\mathbf{i}_{0:k}$, for all $m' \neq m$, given the current target goal and the mental path plan at time instant k. This reduces the user input model to $P(u_k \mid {}^m\mathbf{i}_k)$. Intuitively, this means that the user command at time instant k is only influenced by the mental path plan that brings the robot to the target goal location at that time instant. We assume additionally that the issued command is mainly influenced by the local surroundings of the robot, and in particular by a subgoal or a viapoint on the path ${}^m\mathbf{i}_k$, referred to as ${}^m\mathbf{g}_k$. Subgoals are determined on each path to each defined goal, i.e. $\mathbf{x}_k \rightarrow {}^m\mathbf{g}_k \rightarrow {}^m\mathbf{g}$ as the furthest point, to which a straight line can be drawn from the current robot position without touching any obstacle. This definition is similar to the one proposed in [32, 33] for discrete interfaces. Subgoals are searched for within a predefined circle around the robot with a radius d_{subgoal}. The process of finding the subgoals is depicted in Fig. 3. These assumptions

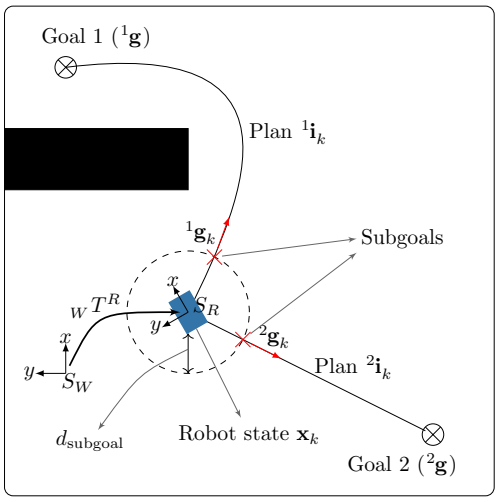


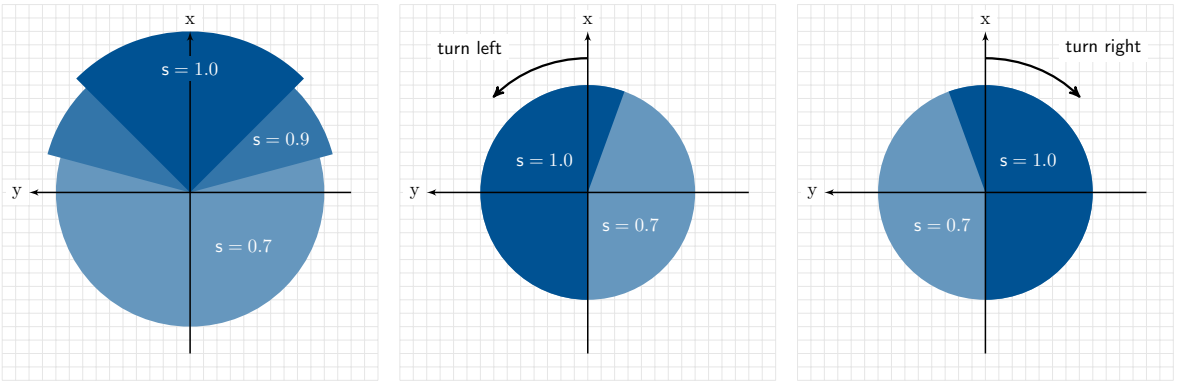
Figure 3: An example of estimating subgoals for goals 1 and 2. Subgoals in the figure are indicated by the red crosses and the black rectangle corresponds to an obstacle.

lead to the new approximation for the user model as $P(u_k \mid {}^m\mathbf{i}_k) \approx P(u_k \mid \mathbf{x}_k, {}^m\mathbf{g}_k)$.

Once the user command u_k arrives, the relative orientations of the robot with respect to the computed subgoals are used to compute the approximated likelihood function $P(u_k \mid \mathbf{x}_k, {}^m\mathbf{g}_k)$ as shown in Fig. 4. For instance, when a turn left command is issued by the user, all computed subgoals which lie in the left semicircle with respect to the robot's heading are assigned a higher score than those which lie in the right semicircle. The user model ignores the z-component (the altitude) of the available goals.

4.2.2. User Gaze Model

The term $P(\mathbf{h}_k \mid {}^m\mathbf{g}, u_{0:k-1}, \mathbf{h}_{0:k-1}, {}^{1:n}\mathbf{i}_{0:k})$ denotes the likelihood of a user's gaze $\mathbf{h}_k$ given the sequence of user commands $u_{0:k-1}$, the previous gaze sequence $\mathbf{h}_{0:k-1}$ and the sequence of all computed plans ${}^{1:n}\mathbf{i}_{0:k}$, where the user has the goal ${}^m\mathbf{g}$ as a target goal. We assume that the user always tries, if feasible, to bring the final goal ${}^m\mathbf{g}$ into sight from the current robot location $\mathbf{x}_k$. This assumption is supported by the work in [34], where it has been concluded that, in natural settings, look-ahead fixations represent a task-dependent strategy. In our application, look-ahead user fixations are assumed necessary to update the navigation plan every time the robot moves or objects in the environment move. Additionally, it has been shown in [35] that subjects with normal vision direct their gaze primarily ahead or at the goal while walking a simple and obstacle-free route. Our assumption obviously ignores the possibility that the user's focus of attention can be shifted towards subgoals, which



(a) Forward (b) Turn left (c) Turn right

Figure 4: The user input model shown for the translational and rotational commands. Subgoals are assigned a score depending on their location relative to the robot heading (corresponds to the x-axis in the plots). The lighter the area, the lower the score, e.g. subgoals, which are located in the negative x direction and have an angle greater than 180° (behind the robot) get the lowest scores assigned if a forward command were issued. The shown scores are exemplary.

he/she is trying to reach on the path towards the final goal. Nonetheless, such subgoals can be accounted for by the user input model. The user gaze model can then be approximated as $P(\mathbf{h}_k \mid {}^m\mathbf{g}, \mathbf{x}_k)$.

In order to compute $P(\mathbf{h}_k \mid {}^m\mathbf{g}, \mathbf{x}_k)$, we note that the field of view of human vision is typically divided into an inner (i.e. foveal vision) and an outer (i.e. peripheral vision) part. The foveal vision corresponds to the sight area which maps on the central part of the retina with the highest receptor density and highest visual resolution. Thus, for sharp vision, one has to align the eyes/head to look directly at the point of interest. The peripheral vision, on the other hand corresponds to the remaining area of the visible field of view with lower resolution.

Inspired by these characteristics of the human vision system, we define two regions for the focus of attention. An inner region that is defined by the two angles ϵ_0 and δ_0, which respectively determine the horizontal and vertical openings around the gaze direction, as depicted in Fig. 5. Similarly, the outer region can be determined by the angles ϵ_1 and δ_1. With these assumptions, target goals are classified into three categories at any time instant k, according to their position with respect to the two regions of attentional focus, i.e. in the inner or outer region or out of sight. By conforming transformation, the goal locations (ignoring their relative orientation, i.e. ${}^m\theta$) can be defined with respect to the moving gaze frame as

$$ {}^m\mathbf{g}^H = {}_W T^H \cdot {}^m\mathbf{g}^W, \tag{4} $$

where ${}_W T^H$ denotes the transformation matrix from the fixed world frame (W) to the moving gaze frame (H).

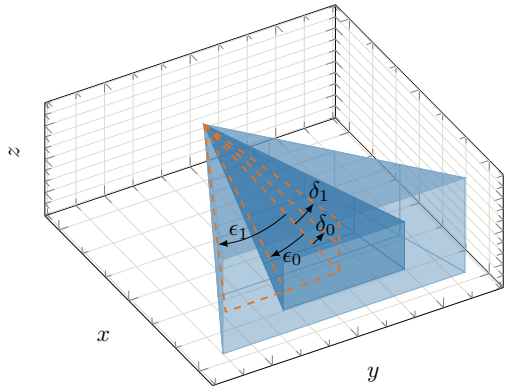


Figure 5: The inner and outer spatial regions of attentional focus are shown with respect to the coordinates system of the moving gaze frame.

Consequently, three conditions need to be fulfilled for a goal m to be assigned to the inner region. These are

$$ {}^m x^H > 0 \tag{5} $$

$$ \operatorname{atan}({}^m y^H, {}^m x^H) < \epsilon_0 \tag{6} $$

$$ \operatorname{atan}({}^m z^H, {}^m x^H) < \delta_0 \tag{7} $$

where ${}^m x^H, {}^m y^H$ and ${}^m z^H$ are the x, y and z components of the goal location in the moving gaze frame (H). Similar conditions can be derived for the outer region using the limiting angles ϵ_1 and δ_1.

Based on our assumption that users occasionally bring their goals into their field of view, we consider

the gaze mode as

$$P(\mathbf{h}_k \mid \mathbf{x}_k, {}^m\mathbf{g}) \propto \begin{cases} a & \text{if } {}^m\mathbf{g} \text{ lies within inner fov} \\ b & \text{if } {}^m\mathbf{g} \text{ lies within outer fov} \\ c & \text{otherwise,} \end{cases} \tag{8}$$

where $a > b > c > 0$ are constants to be chosen at design time.

4.2.3. Plans Evolution Model

The path plans evolution model denoted by $P({}^{1:n}\mathbf{i}_k \mid {}^m\mathbf{g}, u_{0:k-1}, \mathbf{h}_{0:k-1}, {}^{1:n}\mathbf{i}_{0:k-1})$ defines the likelihood that the robot currently has the path plans ${}^{1:n}\mathbf{i}_k$ given that the user has ${}^m\mathbf{g}$ in mind, and the sequences of issued commands and gaze points and all path plans previously computed for all goals up to time instant $k-1$. Assuming that the plans ${}^{1:n}\mathbf{i}_k$ are conditionally independent of the previously issued commands and gaze points given previous path plans and the target goal, the path evolution model reduces to $P({}^{1:n}\mathbf{i}_k \mid {}^m\mathbf{g}, {}^{1:n}\mathbf{i}_{0:k-1})$. Despite this simplification, the computation of this model remains a bit tricky as it involves all plans computed to all end goals up to time instant k. In the following, a series of approximations allow to arrive at a reasonably simple model. Hereby, we consider first the path plan length, i.e. $({}^m l_k)$, as an approximate sufficient statistic of the plan ${}^m\mathbf{i}_k$. This yields $P({}^{1:n}l_k \mid {}^m\mathbf{g}, {}^{1:n}l_{0:k-1})$ as an approximation of the plan evolution model, where ${}^{1:n}l_k$ denotes the path plan lengths to all goals at time instant k. With this approximate statistic, only the lengths of the plans, rather than the complete path plans, need to be stored in memory in order to compute the score of the plans evolution model. With further simplification, the memory requirements are reduced to a single value. Fig. 6 shows a simple example with three different goals, and the evolution of their path plans. At each time instant k, the current and previous path plans are used to reason about the possible target goals.

As a heuristic estimate of the plan evolution model, we use the relative changes in path lengths returned by the path planner at each update to capture special trends towards (or away from) one (or more) end goals. Formally, we define $\Delta^m l_k = \frac{{}^m l_k - {}^m l_{k-1}}{{}^m l_k + C}$ as the relative difference in length at time k for goal m, where division by zero is mitigated by the constant $C \in \mathbb{R}^+$. Dividing by the current path length assures that closer goals are favored over farther ones. To account for path length differences further back in time, a weighted moving average over $\Delta^m l_k$ denoted by $\Delta^m \tilde{l}_k$ is adopted and computed with $\Delta^m \tilde{l}_k = \alpha \Delta^m l_k + (1-\alpha)\Delta^m \tilde{l}_{k-1}$, where $0 < \alpha < 1$ is a forgetting factor, and $\Delta^m l_1 = \Delta^m \tilde{l}_1 = 0$ for all m since $\mathbf{x}_0 = \mathbf{x}_1$.

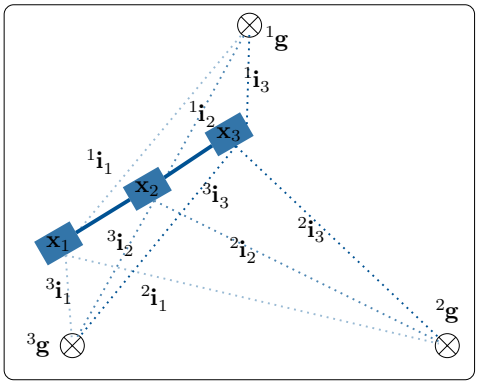


Figure 6: A simple example showing the evolution of trajectories towards three target goals. The plan evolution model assigns more scores to the goals whose record so far shows a trend of approach (e.g. ${}^1\mathbf{g}$ in the figure since ${}^1l_3 < {}^1l_2 < {}^1l_1$).

The values of $\Delta^m \tilde{l}_k$ summarize approach (and depart) trends with respect to the different target goals in the environment, and thereby the plan evolution score can be approximated with

$$P({}^{1:n}l_k \mid {}^m\mathbf{g}, {}^{1:n}l_{0:k-1}) \propto f(\Delta^m \tilde{l}_k), \tag{9}$$

where only the sequence of path plans towards goal m are used to compute $f(\Delta^m \tilde{l}_k)$. This is rather a simplification, but given that $P({}^{1:n}l_k \mid {}^m\mathbf{g}, {}^{1:n}l_{0:k-1})$ needs to be computed for all goals in the environment, all available information from the computed path plans will be made use of. The function $f(\Delta^m \tilde{l}_k)$ is defined as

$$f(\Delta^m \tilde{l}_k) = \beta_1 \cdot \left(1 - \frac{1}{\exp\left(-\beta_2 \cdot \Delta^m \tilde{l}_k\right) + 1}\right) + \beta_3, \tag{10}$$

where the parameters β_1, β_2 and β_3 to be chosen at design time.

Intuitively, the score function in (10) favors goals whose new plans are shorter than the previously computed and among those, favors the closer end goals to the farther ones.

4.2.4. GR in Noisy Interfaces

For noisy interfaces, the recognizer does not have access to the true user command u_k, but rather to a noisy version $\hat{u}_k$ thereof. Using the law of total probability, the update rule can be modified, similar to [31], to account for the partially observable user input in the following manner:

$$
\begin{aligned}
&P_k({}^m\mathbf{g} \mid \hat{u}_{0:k}, \mathbf{h}_{0:k}, {}^{1:n}\mathbf{i}_{0:k}) \\
&= \sum_{u_{1:k} \in \mathcal{U}^k} P_k({}^m\mathbf{g} \mid u_{0:k}, \hat{u}_{0:k}, \mathbf{h}_{0:k}, {}^{1:n}\mathbf{i}_{0:k}) \cdot P(u_{0:k} \mid \hat{u}_{0:k}, \mathbf{h}_{0:k}, {}^{1:n}\mathbf{i}_{0:k}) \\
&\overset{(1)}{=} \sum_{u_{1:k} \in \mathcal{U}^k} P_k({}^m\mathbf{g} \mid u_{0:k}, \mathbf{h}_{0:k}, {}^{1:n}\mathbf{i}_{0:k}) \cdot P(u_{0:k} \mid \hat{u}_{0:k}) \\
&\overset{(2)}{=} \sum_{u_{1:k} \in \mathcal{U}^k} P_k({}^m\mathbf{g} \mid u_{0:k}, \mathbf{h}_{0:k}, {}^{1:n}\mathbf{i}_{0:k}) \cdot \prod_{i=1}^{i=k} P(u_i \mid \hat{u}_i)
\end{aligned}
\tag{11}
$$

The first simplification (1) is based on the assumption that $u_{0:k}$ is conditionally independent of $\mathbf{h}_{0:k}$ and ${}^{1:n}\mathbf{i}_{0:k}$ given $\hat{u}_{0:k}$. On the other hand, (2) is based on the assumption that consequent user commands are independent of each other and only depend on the current noisy measure thereof.

The formulation in (11) means that the recognizer should keep a record of all possible hypotheses about the user input which can be traced back to the first observation received, i.e. $\hat{u}_1$. The number of these hypotheses, however, grows exponentially with k, i.e. $|\mathcal{U}|^k$. Therefore, in this work, we limit tracing these hypotheses to the last issued command only, which implies that the uncertainty about each command is only accounted for once. In case of SSVEP-based interaction, the probabilities $P(u_i \mid \hat{u}_i)$ are obtained from the interface confusion matrix which can be computed from a short training session as shown in [36]. Hereby, (11) also includes the possibility that the BCI issues a command while the user wants to be in the idle state (i.e. NOOP command). However, this is taken into consideration only in updating the posterior of the belief vector. The wrongly classified commands during the idle state propagate to the execution phase.

4.2.5. Asynchronous Posterior Updates

In absence of any additional information, e.g. user navigation preferences, the system starts with $\mathbf{P}_0 = [1/n, \cdots 1/n]$. The update rule in (3) can be triggered either by the arrival of a user command or a new gaze point at the time instant k. Obviously, both triggers arrive in an asynchronous manner, and the probability that both will arrive exactly at the same time can be neglected. It is possible here for example to wait for the availability of the two signals to trigger the update synchronously. However, in this case many useful information, from which the recognizer can benefit will be lost. The other possibility is to trigger the update with every new information observed about the user. Consequently, the user input model contribution to the posterior in (3) will be ignored in case of gaze-based trigger, and the user gaze model contribution will be ignored when a new user command arrives.

In case of updates triggered by new user commands, a multiplication of the user input probability, the plan evolution probability and the prior followed by a normalization step yields the posterior. Since the path evolution model is independent of the user command which triggers the update, we compute its contribution every time the robot state $\mathbf{x}$ changes in order to react as quickly as possible to the arrival of user commands. In the case of SSVEP-based interaction, uncertainty in user commands is accounted for with the formula in (11). Gaze-triggered updates are computed in a similar manner.

5. Experimental Evaluation

In order to empirically evaluate the Bayesian framework from Sec. 4, experiments were conducted with healthy subjects with a real and a simulated robot. This study is part of a larger project, which is approved by the Ethics Committee of the Faculty of Medicine of the Technical University of Munich (TUM).

Hereby, the performance of the GR framework is directly evaluated as the fraction of instances, at which the recognizer is able to correctly estimate the hidden user goals, while users navigate towards different goals in the remote environment. However, the predictions of the GR (i.e. the belief vectors) remain useless unless they are used in a way or another to improve interaction. Therefore, we adopted a shared control (SC) application of the belief vector, whereby when the GR module becomes confident about its belief, online modulation of the translational and rotational steps is performed so that the robot gets closer to high probable goals. The integration of an unobtrusive SC with a reliable GR system is expected to reduce the number of user commands required to accomplish navigation tasks. This way, the shared control (SC) application can provide indirect measure of the GR method reliability. Formally, the step modulation is obtained with

$$
x = \begin{cases} x_{\text{default}} & \text{if } \; s < s_{\text{thresh}} \\ s \cdot x_{\text{opt}} + (1-s) \cdot x_{\text{default}} & \text{if } \; s \geq s_{\text{thresh}}, \end{cases}
\tag{12}
$$

where x_{default} is the default translational (δd_{def}) or rotational steps ($\delta\theta_{\text{def}}$), $s \in [0,1]$ is the modulation factor and x_{opt} is chosen in a way so that the robot gets more attracted to the goals with the highest belief scores. Furthermore, we define s_{thresh} as a threshold, below which online parameters modulation is disabled. The modulation factor is chosen here to be the confidence of the GR module in its belief, for which we propose a new metric, see Appendix A. Further details about the computation of δd_{opt} and $\delta\theta_{\text{opt}}$ can be found in Appendix B and Appendix C.

5.1. Hypotheses

Based on the characteristics of the Bayesian inference system, we have formulated the following hypotheses. First, the integration of SC exploiting command-triggered updates of the GR into the robotic system is expected to result in a fewer number of user commands when compared to the situation when SC is disabled (H_1). Second, the incorporation of gaze information, in addition to user commands, into the GR system should produce more accurate estimates about the hidden goals and consequently should result in a fewer number of user commands when SC is enabled (H_2) compared to the case when the gaze information is not made use of in the GR system. Third, we expect the GR system, though mostly designed with 2D path planning for 2D navigation commands, to generalize to flat floor 3D environments, but perhaps with reduced performance. On this account, we hypothesize that comparable results, in terms of user commands required to accomplish the assigned tasks, are obtained in experiments within a 3D physical environment and a 2D simulated version thereof (H_3). Fourth, BCIs typically require higher number of user commands in different application domains when compared to deterministic interfaces like keyboards. We hypothesize that the magnitude of reduction in user commands that results from the incorporation of GR into SC should differ between the keyboard and the non-deterministic BCIs (H_4).

5.2. Conditions and Experimental Design

The experiments required that subjects drive a robot in a remote physical and a simulated environment and to visit a predefined subset of goal locations within these environments. We have varied the experimental setup within three factors:

(i) The type of the robot/environment used (RBT): simulated 2D (S) or physical 3D (P).

(ii) The type of the interface ($INTFC$): keyboard (K) vs. SSVEP-based BCI (B).

(iii) The type of the GR-SC ($SCTRL$): GR is based on user commands only with SC not activated ($L1$), SC is applied on the basis of command-triggered belief updates only ($L2$) or SC on the basis of command and gaze-triggered belief updates ($L3$).

A fully crossed design was not feasible, since the gaze (estimated with the head orientation as will be explained later) cannot be incorporated within the GR system for the 2D simulated robot/environment. Fig. 7 visualizes the resulting factorial design with the black cells referring to the infeasible conditions. The following subsections provide more details about the implementation of the different conditions.

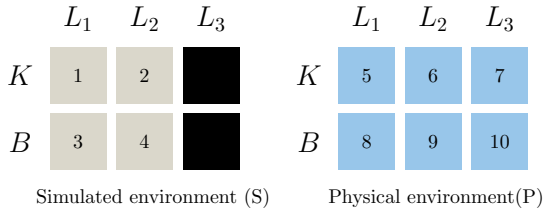


Figure 7: Experimental factorial design consisting of 10 different experimental conditions. Black cells correspond to infeasible conditions.

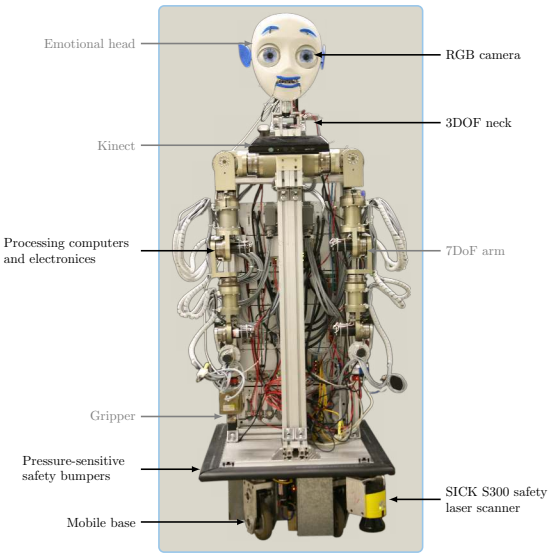


Figure 8: The robot avatar. The labels shown in gray indicate robot parts irrelevant to the scope of this work.

5.3. Experimental Setup

5.3.1. Physical Robot: Fig. 8 shows the robot avatar used in the physical robot conditions (P). The robot has two 7 DoFs arms with a human-like reachable working space for manipulation and a non-holonomic omnidirectional mobile base with rectangular footprint (dimensioned 68 cm × 82 cm) that consists of four wheels. For safety consideration, the mobile base is enclosed with a pressure sensitive bumper, which halts the robot immediately once it makes any contact with any rigid body. Additionally, a pan-tilt-roll unit is mounted onto the torso of the robot and serves as a 3 DoFs neck, where an emotional head is attached. Two RGB cameras (Point Grey, Richmond, Canada) serve as the eyes of the robot. Furthermore, two SICK S300 laser scanners (Waldkirch, Germany) are mounted on two opposite corners of the base to provide a 360° view and used for obstacle detection. However, since the scanners are static, they only can detect

obstacles in the 2D plane parallel to the floor plane (which is assumed to be a flat surface) and having a distance $d = 10$ cm to it. A Kinect (Microsoft, USA) camera is fixed to the robot's chest, which is typically used in manipulation tasks to detect objects in front of the robot. Processing is done with two computers running Ubuntu 12.04 with real-time kernel patch. The Robot Operating System (ROS) [37] is used as the default interprocess communication infrastructure.

During the experiments, the robot received incremental commands from the user that define the direction of translation or rotation only. These commands were translated in turn into linear and angular velocity commands which the low-level controller of the mobile base can understand. The linear velocity is denoted by $\nu = [\nu_x, \nu_y]^T$ and the angular velocity by ω, which were assigned the default values: $\nu_x = \nu_y = \pm 0.25$ m/s and $\omega = \pm 0.25$ rad/s. The reason we chose such slow speeds is to keep a lower rate of optical flow in the visual feedback, which is known to be correlated with cybersickness [38]. A position controller, which continuously received the robot's location, made sure that the robot moved according to the received translation and rotations steps defined. The navigation stack from ROS was used for path planning and the `AMCL` ROS package [39] for robot's localization.

5.3.2. Simulated Robot: For simulated robot conditions (S), a 2D simulated version of the physical robot was constructed using stage simulator package in ROS [40]. Hereby, only the hardware components which are related to navigation were simulated, namely the robot body, the mobile base and the laser scanners.

5.3.3. Collision Avoidance Mode: The collision avoidance behavior is realized with a velocity filter, whereby if the distance from the robot's body to the closest obstacle in the direction of travel (r) gets less than a predefined threshold r_{safe}, the incoming velocity commands undergo a reduction before they get delivered to the low-level controller (i.e. the robot slows down). The actual reduction is determined in proportion to the observed distance. If the distance to the closest obstacle in the direction of travel becomes equal to a second predefined threshold $0 < r_{\text{stop}} < r_{\text{safe}}$, the robot halts and any further velocity commands which might lead the robot closer to this obstacle will be filtered out and blocked. Consequently, this means that the robot always keeps at least r_{stop} distance to the closest obstacle. The default values are set to $r_{\text{stop}} = 0.3$ m and $r_{\text{safe}} = 0.6$ m during all experiments. For a wide range of applications, these values for r_{stop} and r_{safe} seem reasonable, but they can be tuned online as well, if it is required that the robot gets closer to a specific obstacle (or rather a possible target), e.g. to perform some manipulation tasks on objects located on top of a table.

5.3.4. User Interface: The robot avatar was embedded in a remote physical or a simulated environment and received teleoperation commands from the user who conveyed his/her commands to the system with a chosen interface, namely with an SSVEP-based BCI (B) or a keyboard (K). During physical robot operation, subjects received a continuous 3D stereoscopic video stream from the ego-perspective of the robot avatar, which they viewed with the help of a head-mounted display (HMD) weighting 380 gram (Oculus VR, United States). Additionally, the user's head movement was continuously tracked, via the built-in head tracker available in the HMD, and transmitted to the robot side to be mapped into similar movements at the robot's neck. In the simulated environment conditions, the environment was shown on an LCD monitor. In the BCI conditions, visual stimuli were presented overlaid on the received video stream or the simulated environment, as can be seen in Figs. 9 and 10. The temporal resolution of the SSVEP-based interaction depends on the size of the EEG segment used for classifying user's commands and the level of overlap between two consecutive segments as has been pointed out in [36]. The segment size and the temporal resolution were respectively set by default to 2 and 0.25 s. The interaction rate of the BCI (4 Hz) was chosen higher than that of the control loop (e.g. a translational command takes on average 1 s), as it was also important to provide continuous visual feedback about the performance of the interface, so that users could better predict the responsiveness of the interface. This rate mismatch was more pronounced for the keyboard-based interaction, and therefore user commands which arrived while the robot was moving were completely ignored by the interface unless they were meant to stop the robot. The SSVEP detection was based on the supervised CVARS algorithm [36].

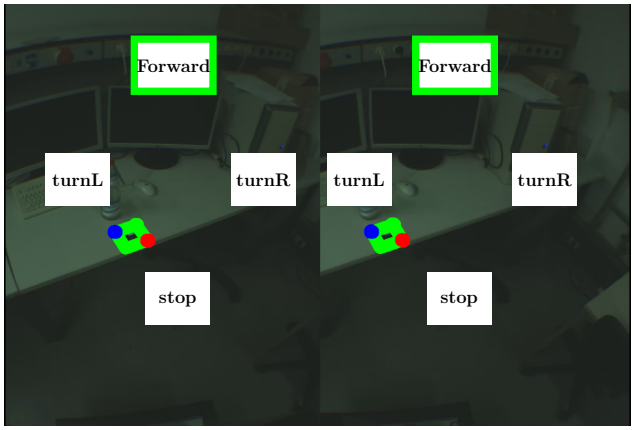
Whereas keyboard interfaces potentially can provide a large set of commands using single buttons or combinations thereof, SSVEP-BCIs can only provide a limited set of commands. This is mainly due to the immersive nature of the application which necessitates that SSVEP stimuli to be shown overlaid on the video-stream received from the remote robot. The more stimuli are shown for display, the less will be the quality of the visual feedback. As a sensible trade-off, we fixed the number of possible commands to 4. By default, the commands move forward, turn left, turn right and stop are available to users. These commands, to which we will refer as *normal mode commands*, were chosen as they allow for human-

like navigation around and in the direction of the symmetry plane. Naturally, backward and sideways movements are the exceptions, not the norm [41]. In situations where the robot receives a command which gets completely blocked as to avoid collisions, the set of user commands gets automatically replaced by a second set that includes move backward, move right, and move left. The set of the new commands is referred to as the *recovery mode commands*. Most if not all the time, the recovery commands will be sufficient to bring the robot to a free space, where the normal mode of operation can be resumed by the user through a dedicated interface element. Adaptations to the interface were accompanied by an auditory feedback that produced the speech of "normal mode commands" or "recovery mode commands", signaling the interface change to subjects.

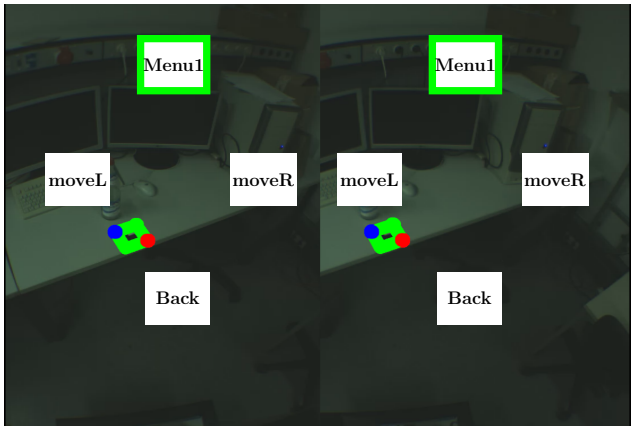
5.3.5. Extraction of Gaze Direction: Typically, humans adjust their gaze by moving both their head (or more precisely the neck) and eyes in order to bring the focus of attention to the spatial regions of interest around them. Specific to our immersive embodiment application, the user's eye movement might not be spontaneous all the time, as it is the case in natural settings. In SSVEP-based interaction in particular, users overtly attend to one of the stimuli distributed at the sides of the display when they decide to issue a control command. Therefore, the user's focus of attention in this work is estimated based on the head orientation only. This simplification is supported by the results in [42], which show that the head orientation contributes 70% to the overall gaze direction on average and that head orientation data alone is sufficient to accurately estimate the focus of attention.

The gaze movements of a user observing a scene can be, for the sake of simplicity, broadly separated into two classes. The first is characterized by sudden and rapid gaze movements known as *saccades*, and the second is characterized by a relative stability of the gaze for typically 200-600 ms [43] and is referred to as *fixations*. Ignoring the contribution of eye movements to gaze direction implies that fixations in this work are approximated by the steady head orientations which last for a certain amount of time, i.e. fixations of the head rather than the gaze.

At the technical level, we define the gaze frame (H) as a moving coordinate system, whose origin is positioned at the midpoint between the robot's eyes, x-axis is aligned with the gaze direction and z-axis parallel to the robot's face plane and pointing upwards, i.e. from the neck to the forehead. In the following, we limit the tracking data to the yaw (Θ) and pitch (Φ) angles of head rotations, as the roll component, i.e. the head rotation around the axis of view, hardly affects the actual gaze direction.



(a) Normal mode commands



(b) Recovery mode commands

Figure 9: SSVEP stimuli overlaid on stereoscopic images coming from the remote physical environment. Recognized user commands are typically highlighted in green. The "Menu1" command in recovery mode resumes the normal mode commands. The scene shows an example target location, where a glass bottle can be observed behind the AR marker (which is highlighted in the figure with green). Part of the robot base is visible at the bottom side of the images. The visible lens barrel distortion (at the capturing side) is counteracted by the HMD which magnifies the images it receives (i.e. pincushion distortion).

In order to extract the gaze information from the raw head tracking data, we adopt a two-stage filter, to which we will refer as the *gaze filter*. In the first stage, fixation points are extracted from the head tracking data as the sample mean of consecutive yaw-pitch pairs, during which the head movements do not exceed a certain velocity threshold for a period of e.g. $t = 200$ ms. This stage is very similar to the I-VT filter algorithm used to classify eye movements [44].

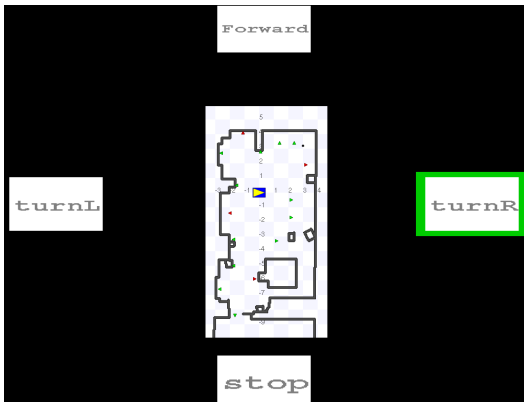


Figure 10: Simulated environment with SSVEP stimuli overlaid at the sides of the display. The stimuli constellation allows for intuitive interaction.

In the second stage, the sample mean of the incoming classified fixations in the yaw-pitch plane is recursively computed to determine the sample gaze mean $[\bar{\Theta}, \bar{\Phi}]$. The newly incoming fixations continuously update the old sample mean as long as their Euclidean distance to the current mean is below a certain threshold denoted by $\mathrm{d_{thresh}}$. Otherwise, i.e. if the distance is larger than $\mathrm{d_{thresh}}$, the sample mean is set to the new fixation, as shown in Fig. 11. Every time the sample mean is set anew, it gets broadcasted as a new gaze point, i.e. $\mathbf{h}_k = [\bar{\Theta}_k, \bar{\Phi}_k]^T$. With the help of this information and the available transformation tree to the system, the transformation from the fixed world frame (W) to the moving gaze frame (H), i.e. ${}_W T^H$, can be easily computed.

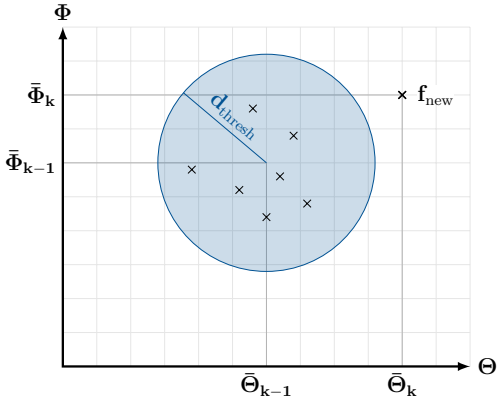


Figure 11: Mean gaze estimation from incoming fixation points (indicated as black crosses). When a new fixation is received, its distance to the old mean is checked against $\mathrm{d_{thresh}}$ and the new mean $[\bar{\Theta}_k, \bar{\Phi}_k]$ is computed accordingly either by updating the old mean $[\bar{\Theta}_{k-1}, \bar{\Phi}_{k-1}]$, or by setting the mean to the new value.

5.3.6. Goal Recognition and Shared Control: The GR module was active in all conditions, but its computed belief vectors were used to modulate the translational and rotational steps only in the conditions $L2$ and $L3$. The default translational and rotational steps were respectively set to $\delta d_{\mathrm{def}} = \pm 0.25$ m/command and $\delta\theta_{\mathrm{def}} = \pm\pi/12$ rad/command. In conditions $L2$ and $L3$, the translational steps were modulated if $s(\mathbf{P}_k) > 0.3$, whereas a more conservative threshold was chosen to modulate the rotational steps, i.e. $s(\mathbf{P}_k) > 0.8$.

The recovery mode and stop commands were excluded from the user input model as they are not typically oriented towards the final goal but rather dictated by the presence of obstacles in the robot's surroundings. In some cases, users might need to recede away from a goal in order to avoid the immediate obstacles. For the user input model of forward commands, the inner and mid opening angles were respectively set to $\pi/4$ rad and $5\pi/12$ rad. The corresponding scores for normal mode commands were chosen as defined in Fig. 4 and the parameter d_{subgoal} was set to 1.5 m.

The user gaze model was defined with the parameters $\epsilon_0 = \delta_0 = 0.1$ rad, $\epsilon_1 = \delta_1 = 0.2$ rad, $a = 1$, $b = 0.9$ and $c = 0.8$. The parameters of the plan evolution model were set as follows: $\alpha = \beta_1 = 0.8$, $\beta_2 = 10$ and $\beta_3 = 0.2$, rendering ${}^m s_k \in [0.2, 1]\ \forall m, k$.

Fig. 12 shows a block diagram of the whole robotic embodiment system for navigation which highlights the interconnections and message passing between the GR and SC blocks with other components.

5.4. Subjects

A total of 22 healthy adults (5 females) aged 27.59 ± 5.66 (range $20-38$) with normal or corrected-to-normal vision served as paid volunteer subjects in this study.

5.5. Task

A 2D occupancy grid of the physical 3D remote environment (a cluttered laboratory space, with many desks, tables and other robots) was built using a laser-based SLAM algorithm available in ROS, i.e. the `gmapping` package [45]. In total, 16 different goals were manually defined on the map, e.g. at the lab doors and other salient spots like desks, and were additionally marked in the physical environment with augmented reality (AR) markers. The occupancy grid of the environment and the set of all goals are shown in Fig. 13. The same occupancy grid was used to build a simulated environment with the stage simulator package in ROS [40].

Participants were instructed to accomplish the same navigational task in all conditions as fast as

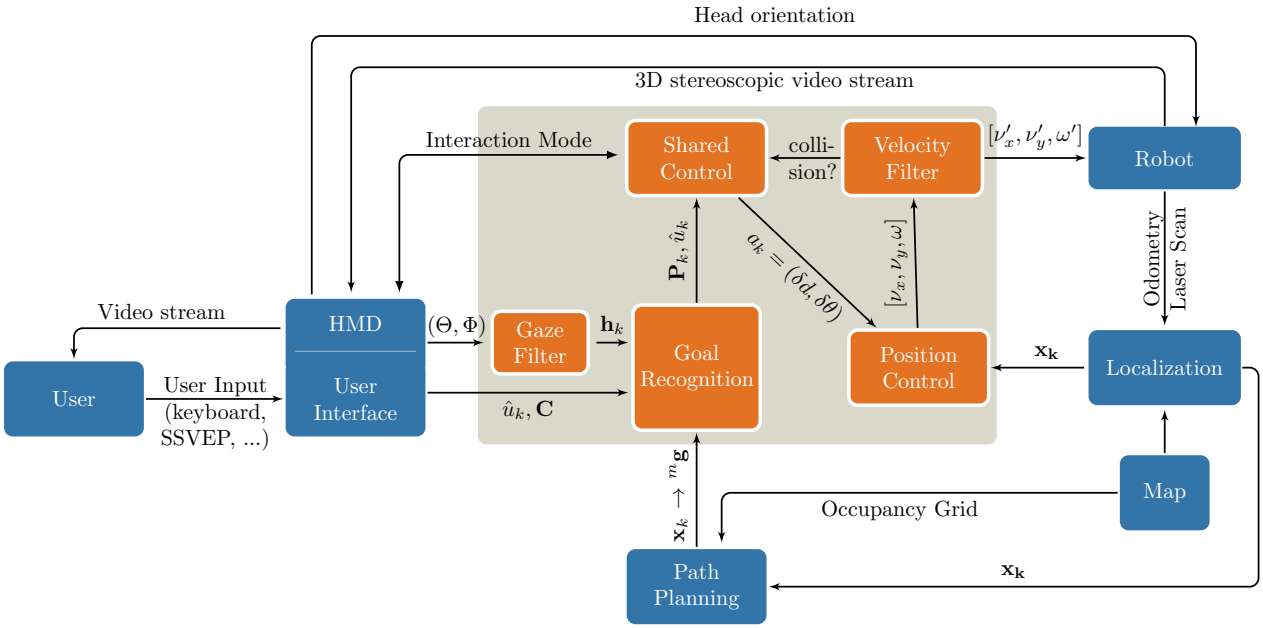


Figure 12: System block diagram. This work focuses on the blocks in the shaded area.

possible. The task was to drive the robot with the chosen interface and visit 4 target goals out of the available 16, where the different subjects were assigned different sets of goals. Participants were free to decide the order, in which they might visit the target goals, but they were instructed to stick to the same order in all yet-to-be-performed sessions. Since the target goals were unfamiliar to the participants, exactly as the non-target ones, glass bottles were placed besides the AR markers of the target goals (on target desks or tables) to allow the participants to recognize them from afar, especially if neighboring goals were very close. A sample screenshot from the ego-perspective of the robot in front of a target goal is shown in Fig. 9. Since doors have less ambiguity, they were only marked with the AR markers. Additionally, in the simulated environment, target goals were colored in red whereas the remaining ones were marked in green. The simulated environment for subject S1 is shown as an example in Fig. 10. A goal was considered reached if its distance to the robot became less than or equal to 0.8 m and its relative orientation with respect to the robot's heading was less than or equal to 0.5 rad. Additionally, the goal had to be positioned within the range$[-0.2, 0.2]$ m with respect to the y-coordinate of the robot's frame. Subjects were instructed simply to bring the robot to face each of the assigned goals. An auditory feedback signal was played back to participants when the robot arrived at any goal location. This way, we guaranteed that participants did not recede from a goal earlier (assuming they arrived) or later (assuming they did not arrive yet) than they should.

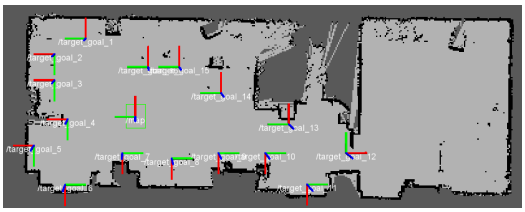


Figure 13: The environmental map was built using laser-based SLAM. The 2D poses of the 16 goals are shown, with respect to the world frame.

5.6. Procedure

Upon their arrival at the laboratory, subjects were provided in written form, all information they needed about the course of the experiment and the different conditions they were going to perform. All participants gave their written informed consent. Participants were additionally asked to fill in a pre-questionnaire to collect some demographical data. They were additionally given a printed floor plan of the remote environment and were asked to decide upon the order in which they wanted to traverse the subset of goals consistently across conditions.

Each participant was assigned 1 session per condition and 10 sessions in total. The order of the experimental conditions was counterbalanced across subjects, where half of the participants were assigned to complete all simulated robot conditions

first, and the other half were assigned the physical robot conditions first. The experimental conditions were not fully randomized as switching back and forth between the simulated and the physical robot conditions involves adding significant time overhead to the already lengthy procedures. If a subject was found to spend relatively long time (i.e. around 3 min) trying to reach the first end goal in the BCI conditions, these sessions were stopped since otherwise experiments might have extended over uncomfortable time lengths to participants, let alone the effect of long recordings on the quality of the EEG setup. Prior to the actual experiments, participants were familiarized with the different system components in the physical and simulated environments, e.g. the head tracking and mapping to the robot's head movement, 3D visual feedback and keyboard interface. In SSVEP-based interaction conditions, electrode placement and setup were done exactly as described in [36], and EEG signals were acquired with a sampling rate of 256 Hz at full DC. Two SSVEP training sessions were additionally collected during experiments with each subject. One training session preceded the simulated environment conditions with stimuli presented against a dark screen on an LCD monitor and viewed binocularly by the participants. The other session was collected immediately before the physical environment conditions. Stimuli in the latter case were presented on the HMD against a static view from the remote environment and were viewed binocularly. Training data were used to learn two linear discriminant analysis (LDA) classifiers for the CVARS scores as described in [36], which were later used in online SSVEP-based interaction conditions, i.e. conditions 3 and 4 for the first classifier and 8, 9 and 10 for the second one. The complete training sessions were used to train the LDA classifiers but the classifier confusion matrix was estimated with a 5-fold cross split of these sessions. To elicit natural behavior, participants were not informed about the existence of the recognizer or the fact that online modulation of the system parameters was active in some conditions. On the other hand, participants were instructed to pay great attention to auditory feedback signaling task completion after each visited goal and signaling the automatic entry to the recovery/normal modes of interaction. Again, when the familiarization phase of an individual subject took relatively long time, experiments were discontinued for them, as the actual experiments were predicted thereby to extend to longer times.

5.7. Performance Metrics

5.7.1. Direct Measures: Recall and precision metrics are adopted as in [46–48] to provide direct evaluation of our GR method. Recall is defined as the fraction of belief updates which has the true user's end goal in the set of N-best predictions. Precision is defined as the ratio of belief updates, in which the recognizer is confident and the true goal is among the N-best predictions, to the total number of belief updates, in which the recognizer is confident. The recognizer's confidence is defined here, as in Appendix A, by the non-uniformness metric of its belief vectors, i.e. when $s(\mathbf{P}_k) > s_{\text{thresh}}$. Formally, let ${}^m\mathbf{P}_{1:k} = \{{}^m\mathbf{P}_1, {}^m\mathbf{P}_2, \dots {}^m\mathbf{P}_k\}$ be the sequence of belief updates during which the true user's hidden goal is ${}^m\mathbf{g}$, recall and precision can be computed as follows

$$\begin{aligned} \text{Recall} &= \frac{\sum_{i=1}^{k} I_N({}^m\mathbf{P}_i)}{k}, \\ \text{Precision}(s_t) &= \frac{\sum_{i=1}^{k} I_N({}^m\mathbf{P}_i) \cdot I_S({}^m\mathbf{P}_i)}{\sum_{i=1}^{k} I_S({}^m\mathbf{P}_i)}, \end{aligned} \tag{13}$$

where the indicator function $I_N({}^m\mathbf{P}_i) = 1$ if the goal m is among the N-best predictions and 0 otherwise and the indicator function $I_S({}^m\mathbf{P}_i) = 1$ if the non-uniformness metric of the belief vector $s({}^m\mathbf{P}_i) > s_t$ and 0 otherwise. Recall and precision were computed on the basis of the four belief sequences, that were obtained per subject and per experimental condition.

5.7.2. Indirect Measures: As described earlier, the performance of the GR module can be indirectly evaluated by considering the effect of SC on the number of commands issued to complete each task. Aiming at generalizing our results with respect to the task space, each subject was assigned a different set of goals, and hence the observed number of commands per condition and subject can be affected by the varying complexity and path length of each sequence of goals. In order to correct for this, a baseline was computed for each sequence, relative to which the observed number of commands can be computed. The baseline is computed as the expected number of discrete normal mode commands needed to visit these goals in the same order which was undertaken by the individual subjects, but assuming free space. Fig. 14 illustrates the computation of the baseline number of commands with a simple example.

6. Results

6.1. Task Completion

Experiments lasted for around 3 hours, including the time for reading the instructions, familiarization, training for SSVEPs, actual task performance and breaks between conditions. Out of the 22 subjects, 12 subjects were able to complete all conditions. 2 subjects completed all conditions except the three PB conditions, i.e. the physical robot with BCI

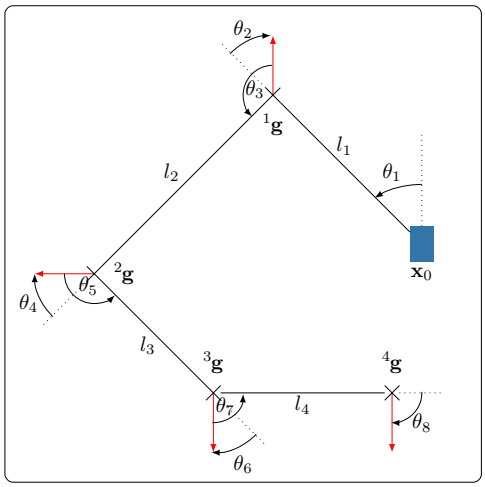


Figure 14: Baseline computation example showing 4 different goal poses visited in the order $^1\mathbf{g}$, $^2\mathbf{g}$, $^3\mathbf{g}$ and $^4\mathbf{g}$ starting from the initial pose $\mathbf{x}_0$. The baseline number of commands is computed as $\sum_{i=1}^{4} l_i + \sum_{j=1}^{j=8} \theta_j$

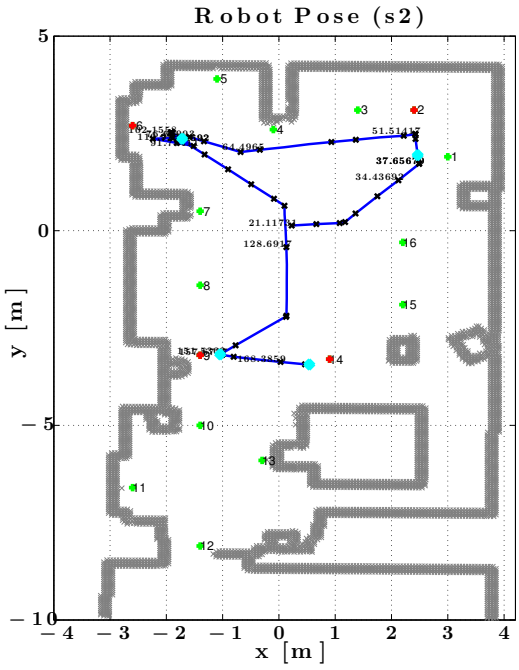


Figure 16: The complete robot's path $\mathbf{x}_k$ for subject S2 in condition $PKL3$. Target goals are marked in red. Additionally, points marked with cyan represent the points (in time and space), at which the user received the auditory signal signaling goal-reaching. All goals in the environment are shown with their identification numbers. The time in seconds is shown for some points along the path, where $t = 0$ represents the time when the session started.

conditions. These two subjects, namely S1 and S3 were the first to use the system, where at this stage of the experiment, we recorded only one training session for SSVEPs using the monitor stimulation. Since the PB conditions require visual stimulation through the HMD for online interaction, it has been observed with these two subjects that the classifier learned for the monitor-based stimulation could not generalize very well to the HMD stimulation, and therefore we decided hereafter to record another training session for the HMD stimulation as described earlier in Sec. 5.5. We decided to keep the incomplete data of these two subjects in our data analysis since other conditions still compare to the procedure used for other subjects. Additionally, 3 other subjects were able to complete all the keyboard conditions, but none of the BCI's, as the accuracies for their SSVEP detection were relatively low. The familiarization step took longer than expected with 4 subjects, and therefore experiments were discontinued with them. 1 subject felt motion sickness during the familiarization stage and decided to drop out.

6.2. Direct Measures

Fig. 15 shows an example of the belief vector $\mathbf{P}_k$ (in condition $PKL3$ for subject S2) as it unfolds over time during the complete session in the form of 16 staircase plots. The rising and falling edges correspond to time instances, at which the belief vector underwent gaze or command-triggered updates. Upon arrival at any goal, the belief vector is reset. Correspondingly, Fig. 16 shows the robot's pose during the complete session. These two figures combined can give a first insight about the performance of the Bayesian GR system.

Fig. 17 summarizes the results from all subjects and all finished sessions with respect to the precision and recall metrics introduced in Sec. 5.7, where precision was evaluated for different values of s_{thresh}. The recall plots indicate that the recognizer was able to correctly estimate the user's hidden end goals as the 1-best prediction around 40% of the time in all experimental conditions. This rate increases with increasing N, e.g. it becomes around 70% in the case of 4-best predictions. On the other hand, the increased precision observed with higher s_{thresh} in Fig. 17(b) indicates the suitability of our proposed non-uniformness metric as a confidence measure for the recognizer.

6.3. Indirect Measures

As previously mentioned, the relative number of commands issued to complete the task is adopted as the comparison criterion (i.e. the dependent variable) using the indirect measures. Since this measure reflects in a way the user's effort needed to accomplish the

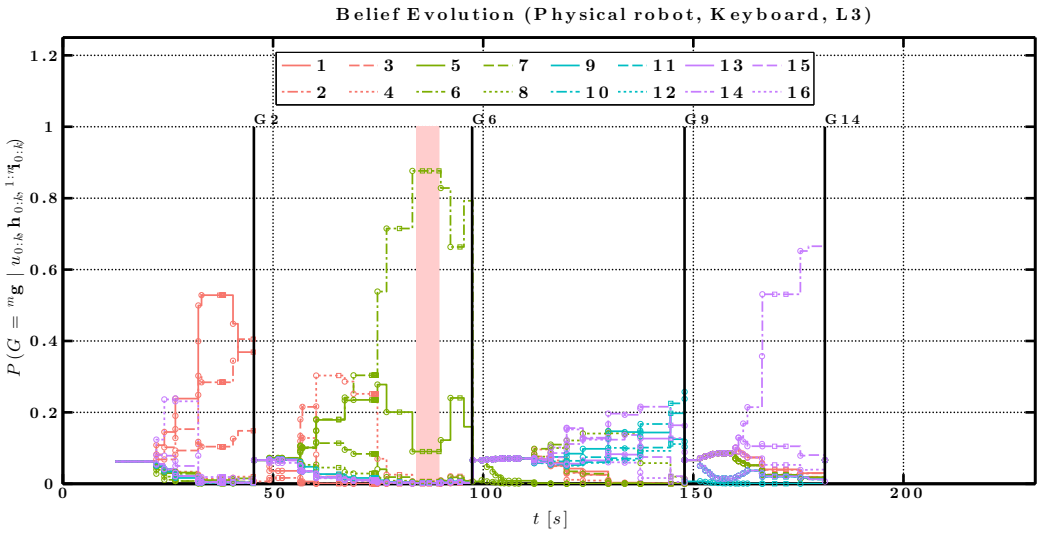
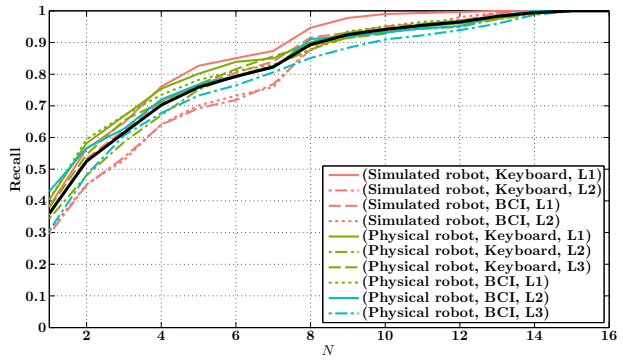
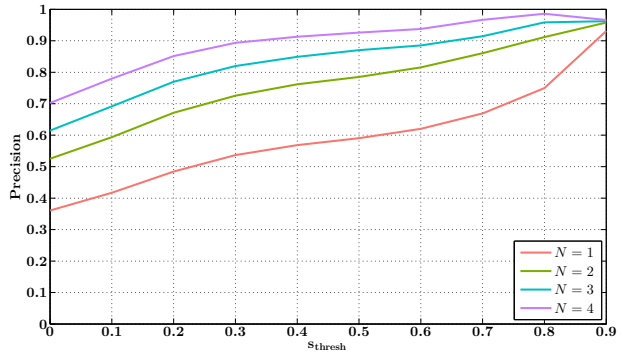


Figure 15: Belief evolution shown for subject S2 in condition $PKL3$. The shaded area shows the time interval(s), during which the recovery mode was active and belief updates were not. The vertical lines correspond to the time instants when the robot arrived at a goal and are annotated accordingly.



(a) Recall for different N-best values



(b) Precision for different N-best values as a function of s_{thresh}

Figure 17: Recall and precision of the GR module. Around 40% of the time, the GR correctly estimated the hidden user intention as the 1-best prediction. From the precision plot, it can be seen the s_{thresh} can be used as a confidence metric.

assigned tasks, we will hereafter refer to it as the user effort.

Our experiment in this study is a partially crossed $2 \times 2 \times 3$ repeated-measures within-subjects factorial design (with missing data). The fact that the incorporation of gaze into the GR system is only feasible in the physical robot conditions renders it legitimate to run the analysis in two steps. Firstly, we consider the fully crossed $2 \times 2 \times 2$ design that excludes the level ($L3$) in the factor $SCTRL$ which leaves us with the levels S/P, K/B and $L1/L2$ of the main factors in the experiment. Secondly, we consider the $L3$ level within the fully crossed 2×3 design corresponding to the K/B and $L1/L2/L3$ levels of the $INTFC$ and $SCTRL$ main factors, respectively.

6.3.1. Statistical Analysis of the $2 \times 2 \times 2$ *Design* ($RBT \times INTFC \times SCTRL$)

The data obtained from all subjects and all sessions (excluding $L3$ sessions) was analyzed using a linear mixed-effects model, where the subject factor was treated as a random factor, and the RBT, $INTFC$ and $SCTRL$ were treated as fixed factors. To this end, the function `lmer` from R `lmerTest` package ‡ was used since it is able to handle unbalanced data, as it is the case here, by approximating the denominator degrees of freedom using either Satterthwaite's or Kenward-Roger's approximations [49]. The three-way ANOVA of type III shows no significance in the three way interaction, $F(1, 97.12) = 0.045, p = 0.83$. However, the two-way interaction terms of $RBT \times INTFC$

‡ The complete analysis can be viewed in this link http://rpubs.com/moh-marwan/rembodiment

and $INTFC \times SCTRL$ are significant, respectively with $F(1, 97.97) = 4.68, p < 0.05$ and $F(1, 97.12) = 4.33, p < 0.05$. The main factors $INTFC$ and $SCTRL$ are found to be significant as well. Follow-up simple effects tests are performed with the help of `multcomb` [50] and `lsmeans` [51] packages. The results are summarized in Figs. 18(a) and 18(b). Hereby, the RBT is found non-significant across all levels of $INTFC$, whereas the levels of $INTFC$ are found to be significantly different across the levels of RBT, as can be seen in Fig. 18(a). Additionally, one can see in Fig. 18(b) that all simple effects of $INTFC \times SCTRL$ are significant, but with different significance levels and therefore we can attribute the significant interaction to these different levels of significance.

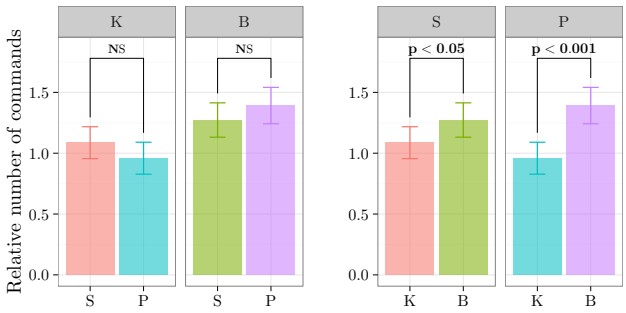
6.3.2. Statistical Analysis of the 2 × 3 Design ($INTFC \times SCTRL$)

The two-way ANOVA of type III tests reveals significance for the two main factors, i.e. $INTFC$ and $SCTRL$, with no significant interaction. Post hoc pairwise comparisons with Bonferroni corrections reveal that the $L2$ required significantly less effort than $L1$. There is also a trend that less user effort is required in case of $L3$, but given our sample size this does not reach significance. Conforming with the three-way ANOVA from the previous subsection, the BCI requires higher effort.

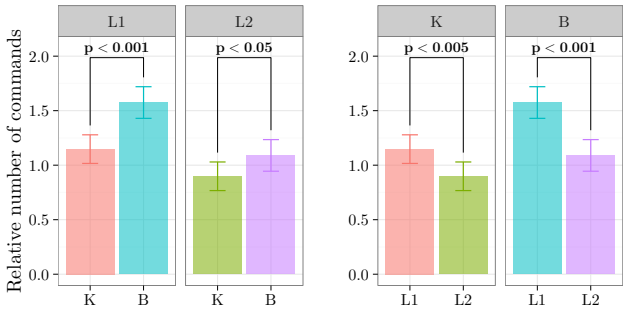
7. Discussion

Only 12 participants out of the 22 were able to complete all the BCI experimental conditions. For those who could not finish these conditions, it is observed that their SSVEP-BCI accuracy was far better than chance level, but lower than or very close to the 70% limit. This value (i.e. 70%) is often used to indicate the threshold, above which communication or BCI control is still possible [52, 53].

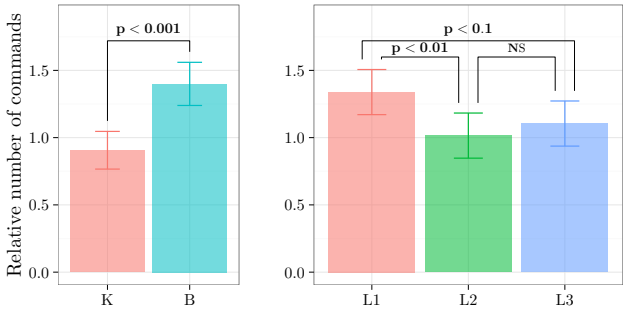
The precision and recall results have shown that the GR system was able to infer the correct target goals, as the best prediction, 40% of the time. Comparing the $L1$ and $L2$ conditions show that a small reduction in recall is observed, despite that the goal recognition was exactly the same in the two conditions. This can be explained by the fact that SC is applied in $L2$, which yields a smaller number of user commands and therefore less evidence about the possible target goals is available to the goal recognition module. The non-uniformness metric, i.e. $s(\mathbf{P}_k)$, was shown additionally to be an adequate measure for the confidence of the goal recognition module, as higher values of $s(\mathbf{P}_k)$ resulted in higher precision of the goal recognition module as can be seen in Fig. 17(b). The Kullback-Leibler divergence is expected to produce



(a) Simple effects for the interaction between the robot type and the interface type



(b) Simple effects for the interaction between the interface type and shared control type



(c) Main factors in the 2 × 3 model

Figure 18: Barplots of the least squares means and 95% confidence interval plotted for (a,b) the 2 × 2 × 3 design and (c) the 2 × 3 design.

similar results as the non-uniformness metric, but we chose the latter for its linearity with respect to the number of goals that share the total probability as shown in Appendix A.

Our statistical analysis, on the other hand, has demonstrated that SC based on GR beliefs resulted in less user effort compared to the case of direct user command, i.e. when SC was disabled (H_1).

Contrary to expectations, the incorporation of gaze information into the GR module did not help to enhance the performance of our GR-SC system (H_2). Several reasons might have contributed to this result.

First, we have assumed a simple model for the user gaze based on the hypothesis that users frequently fixate their vision at the goal of interest to help update their global plans. While this is true in general, the constant time which we assumed in this work (i.e. 200 ms) for head fixations seems not suitable to capture fixation times realistically. Second, users tend to look more often at the immediate vicinity of the robot for better local trajectory planning. Therefore, we predict that considering these details into the user gaze model in future developments will be beneficial.

The results from the three-way ANOVA have shown no significant difference between the simulated (S) and the physical (P) robots/environments (H_3). This is indeed a desirable feature of simulation systems as further developments in GR can be, if feasible, tested first in simulation.

Additionally, it comes as no surprise that the BCI resulted in higher user effort, with average increase of around 30% compared to the keyboard. This is mainly due to the imperfect detection of the SSVEP signals. However, most of the participants were able to use the system and accomplished relatively complex navigation tasks during the different conditions. Subjects, for which the SSVEP detection did not get over 70% accuracy, encountered difficulties to accomplish the task in the BCI conditions, and therefore, BCI sessions were discontinued with them as has been previously reported. With respect to the BCI inclusion in the GR algorithm, Fig. 17(a) clearly shows that this did harm the precision of the GR module. Importantly, the effect of $L2$ shared control has been shown to be more significant in the BCI sessions compared to the keyboard case (H_4), as can be seen in Fig. 18(a). This also should come as no surprise since shared control should have a larger effect for noisy interfaces.

Even when the GR module was highly confident about its prediction, the correct target goal was sometimes confused with other goals in the environment. The belief evolution plots for all subjects and conditions show that such confusion happens most of the time with adjacent goals. Therefore, the performance of the GR module is expected to improve for environments with less goal density. Reciprocally, with a higher goal density, the performance of the system might be negatively affected. The number of goals in the environment per se is not expected to have a high influence on the precision or recall, but will definitely affect the processing speed. This is due to the fact that the computation of the path plans to all goals, which takes place at the end of each movement, constitutes the processing bottleneck of the GR system. In order to efficiently generalize to environments with higher number of goals, and environments with higher goal density, a hierarchical structure of target goals in the environment might be of high benefit. Hereby, target goals can be clustered, with respect to their distance to each other, within different levels. GR can be then applied to the goal hierarchy from the coarse to the refined representation. In other words, recognition is applied first between clusters, and then between the goals within each cluster. Further investigation of this approach is required.

Furthermore, online modulation of the system parameters on the basis of the GR belief resulted in an average reduction by a factor of 30% (for level $L2$) in the total number of commands required to accomplish the tasks. Given the observed precision levels of the GR system shown in Fig. 17, more proactive assistance can also be provided in further developments to the system, e.g. to autonomously maneuver the robot towards a goal, when the system is highly confident about its predictions. This brings us back to the autonomy continuum we discussed in Sec. 2, where the belief vector lends itself as a plausible criterion to automatically change the operational mode of the robot and/or to trigger automatic interface adaptations, whereby e.g. the most probable goals are shown to users for selection.

During the experiments reported here, most subjects had difficulties, which were also reported verbally by some of them, to issue stop commands as quickly as required in the BCI conditions. This is due to the limiting factors of the buffering step in the SSVEP detection method. Therefore, devising a reliable way to stop the robot, e.g. with an EMG channel [54], if feasible, might be necessary to ensure that users can maintain supervisory control over the actions of the robot.

We assumed throughout this work that the map of the remote environment and the possible goal locations are known a priori. If not available, this information can be also learned throughout interaction with users. For instance, states of interest (goals or sub-goals) were extracted from successful task episodes based on the averaged occurrence frequency in [55]. The map itself is easily obtained with the SLAM algorithm, e.g. with `gmapping` ROS package [45] which was used for the purposes of this work.

We assumed additionally in this work that users, at each time, have a specific end goal in mind. The proposed GR module is expected to perform also well in situations when users change the targeted goal during movement. The reason for adopting a *keyhole* approach in this work was primarily to elicit natural behavior of users so that neglecting user cooperation while evaluating the performance of the GR-SC algorithms becomes legitimate. In this regard, *intended* intention recognition approaches might be appropriate for robotic embodiment systems as well.

For instance, should the user become aware of gaze-triggered belief updates about their end goals, users might choose to fixate purposely at these goals for extended periods of time, so that some ambiguity can be resolved at the recognizer side.

Due to the length of our experiments, we were not able to include another factor to test the effects of self-adaptations to the SSVEP-based BCI, i.e. the automatic update of the BCI recovery mode commands. However, we argue that these adaptations have played a major role in making the task easy to complete and in keeping the number of interactions required to complete the assigned tasks comparable with the baseline. For instance, consider the scenario where the robot is facing a desk where the target goal is only some tens of centimeters away from the robot, such that it is located in front and to the left of the robot. With the normal mode commands only, the user needs to convey several commands to arrive at the goal, e.g. perform a half-turn, move forward, perform a 90° turn, move forward, move for another 90° and move forward. This sequence is also not guaranteed to arrive at the goal location in case of BCIs. However, with the adaptive interface, the target goal is only one command away from the user (i.e. using the move left command). This very advantage of the adaptive BCI has been observed often during the experiments.

Throughout this work, we also assumed that the localization module provided perfect estimates about the robot's location. Given the underlying probabilistic nature of the `AMCL` ROS package that implements the adaptive particle filter for localization [39], the estimated robot's pose is a noisy version of the true pose. We expect that the localization noise affected all experimental conditions uniformly, and therefore its effect on the obtained results can be ignored. However, future developments might benefit from taking the uncertainty in the robot's pose into consideration.

There are some technical details that we skipped so far for the sake of not cluttering the board with too many details that obscure the most relevant issues to GR, but these details remain part of the bigger picture of robotic embodiment. For instance, the stereoscopic visual feedback provided to users in the physical condition did not perfectly match direct line-of-sight. Though the robot was equipped with cameras that provide large field of view, the complete FOV of humans was not reached. As a result, the perceived distances through the visual channel did not map one-to-one to real distances. Obviously, users were able to learn the actual mapping as they could navigate freely in the remote environment accordingly. Furthermore, the requirements for real-time video streaming necessitate compression and decompression of captured images at the robot and the user side, respectively. This leads to some compression artifacts that reduce the quality of viewed video stream, which is further reduced by the resolution of the used display, i.e the HMD. Additionally, due to physical constraints, the 3 DoFs neck of the robot did not exactly match the range of human neck movement. As a result, it has been observed during experiments, that such mismatch often affected the user ability to explore the environment. Improvements to such technical details will undoubtedly contribute to improved quality of user experience in robotic embodiment systems.

On a different vein, the length of the conducted experiments did not allow us to additionally gather questionnaire data about the subjective experience of the different users after each experimental condition. This data would otherwise have provided us with more insights on how the several factors affect the level of embodiment. For instance, in spite of the observed non-significant difference between the simulated and the physical conditions with respect to the required user effort, the two conditions have fundamental differences with respect to embodiment. This is due to the fact that the simulated condition (S) requires that the users observe the robot/environment from a third person perspective (3PP), while in the physical condition (P) the user has a first person perspective (1PP). It is established in [56] that the user's visual perspective has a significant impact on the level of body ownership and embodiment. Furthermore, modulating the length of translational and rotational movements may have as well affected the users' sense of agency and embodiment. As such, the possibility that these discrepancies may have affected our results cannot be excluded with absolute certainty.

8. Conclusions

It is argued in this work that adaptive BCIs offer a way out of the bandwidth bottleneck in BCI-based robotic applications. In order for BCI self-adaptations to be effective, it is crucial to reason about the user hidden intention. On this account, we have focused on goal recognition within a specific robotic application, namely navigation tasks in collision avoidance mode. The proposed goal recognition module is based on a recursive Bayesian update rule, for which some simplifications are made by considering intuitive heuristics that model the behavior of the general population in the domain, and therefore can be used in a plug-and-play fashion. The output of the goal recognition system is a belief vector that assigns a probability value to each possible target goal in the environment. A novel metric that reflects the non-uniformness of the belief vector was proposed to estimate the confidence of the goal recognition module

in its predictions.

In order to evaluate the proposed goal recognition system, experiments were conducted with healthy subjects within robotic embodiment settings. These experiments varied along three factors: the type of the robot/environment (simulated and physical), the type of the interface (keyboard or BCI), and the way goal recognition was used to guide a simple shared control driving scheme. Our results have shown that the proposed GR algorithm was able to track and infer the hidden user goals with relatively high precision and recall. Results have shown also that the simple SC scheme could benefit from the output of the goal recognition system and was able to reduce the user effort needed to accomplish the assigned tasks, where user effort was thought to be reflected in the number of issued commands. Additionally, we found that there was no significant difference in user effort between the simulated and real environments, suggesting that the simulated environment might be used in the test phases of further developments to the goal recognition algorithm. Despite the fact that BCI required higher effort compared to the keyboard conditions, most subjects were able to complete the assigned tasks, and the proposed goal recognition system was additionally shown able to handle the uncertainty in user input during SSVEP-based interaction. The SC application of the belief vector has shown that the benefit of the goal recognition module was more pronounced for BCIs than in the case of keyboard interfaces. On the other hand, subjects, who had relatively low SSVEP signal-to-noise ratio encountered considerable difficulties, especially that frequent erroneous interactions give rise to oscillatory behavior of the robot, where e.g. random in-place rotations are very frequent. As a result, corresponding BCI experiments were discontinued with them since otherwise, experiments would have lasted for uncomfortably long periods of time.

The collision avoidance mode was adopted in this work since it allows to leave most of the control in the hands of the user. Setting the collision avoidance as the default operational mode within navigation tasks brings other advantages. This mode allows the goal recognition module to benefit from high rate of user commands, on which inference can be based. In higher autonomy modes of robot operation, where less user input is available, other goal recognition methods are required. Additionally, the obtained belief vector from the goal recognition module can be used for other purposes than the simple shared control application presented in this work. For instance, it can be used as a basis for interface self-adaptations. That is, when the belief reaches high confidence regarding target goals, the most probable goals can be e.g. shown to the user for selection.

In summary, we have devised a recursive Bayesian rule to infer and track the target goal locations pursued by users while navigating in robotic embodiment systems. Being based on different intuitive heuristics and assuming that users typically exhibit similar navigation behavior in structured environments, the proposed method can be used without prior training of system parameters to individual users. The output of the goal recognition module lends itself as a plausible criteria to guide more advanced shared control driving schemes and strategies for interface self-adaptation.

Acknowledgments

This work is supported in part by the VERE project within the 7th Framework Programme of the European Union, FET-Human Computer Confluence Initiative, contract number ICT-2010-257695 and the Institute for Advanced Study, Technical University of Munich (TUM). The authors would like to thank Mahmoud Kassar and Tailin Li who assisted in conducting the experiment.

Appendix A. Belief Confidence

The belief vector is of great importance on its own, but it is also necessary sometimes to have a metric that summarizes the confidence of the GR in such beliefs. To this end, we chose a novel metric (s) that reflects the non-uniformness of the belief vector. The new metric can be computed for any probability mass function (like the belief vector) characterizing a random variable X with n possible values, with the following steps:

(i) The probability masses are ordered in ascending order, and the cumulative distribution function (CDF) is computed.

(ii) The zeroth moment M_0 and the first moment M_1 of the CDF are computed with $M_0 = \sum_m cdf[m]$ and $M_1 = \sum_m m \cdot cdf[m]$.

(iii) The x-coordinate of the centroid of the CDF is evaluated with $c = M_1/M_0$.

(iv) The non-uniformness metric is computed by normalizing c in the range $[0, 1]$ with $s = (c - c_{\min})/(c_{\max} - c_{\min})$, where $c_{\min} = (2 \cdot n + 1)/3$ and $c_{\max} = n$.

The metric s is a function of the belief vector, i.e. $s(\mathbf{P}_k)$ that ranges from 0 (for fully uniform beliefs) to 1 (for unit mass beliefs). The values of s can be considered as to reflect the recognizer's confidence about its belief, since the concentration of probability at one mass point typically reflects the accumulation of enough evidence about a specific goal location. One desired feature of the new metric is its linearity with respect to the number of goals

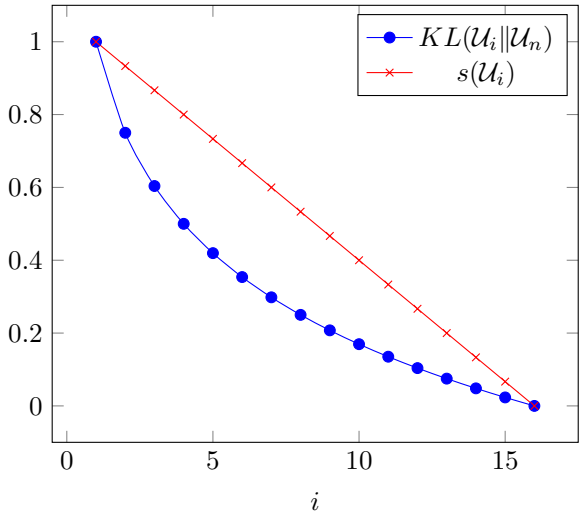


Figure A1: The KL-divergence measure compared to the non-uniformness metric.

that share the total probability. This is shown in Fig. A1 comparing the non-uniformness metric of $\mathcal{U}_i$ to Kullback-Leibler divergence $KL(\mathcal{U}_i\|\mathcal{U}_n)$, where $\mathcal{U}_n$ is the uniform assignment over the n possible values of X, i.e. $\mathcal{U}_n = [1/n, \cdots, 1/n]^T$ and $\mathcal{U}_i$ is the uniform assignment of total probability over a subset $i \leq n$ of the sample space, e.g. $\mathcal{U}_i = [1/2, 1/2, 0, \cdots 0]^T$ for $i = 2$. In Fig. A1, the KL-divergence measure is normalized with respect to the value $KL(\mathcal{U}_1\|\mathcal{U}_n)$.

Appendix B. Optimal Translational Step

In order to compute the optimal translational step when the recognizer is quite confident about its belief, we model the different goals as different point attractors on a 2D plane. Hereby, the different goals (represented by their corresponding subgoals from the user input model in Sec. 4.2.1) exert different forces on the robot's translational movement in proportion to their squared probabilities, i.e. ${}^m{P_k}^2$.

We define δd_{opt} as the minimizer of the energy function

$$f(\delta d) = \frac{1}{2}\sum_m {}^m{P_k}^2 \cdot \|T(\mathbf{x}_k, \delta d) - {}^m\mathbf{g}_k\|_2^2, \qquad \text{(B.1)}$$

where $\delta d \in [0, \delta d_{\max}]$ and $T(\mathbf{x}_k, \delta d)$ defines the new position of the robot after translating δd meters in the direction of travel. The term $\|T(\mathbf{x}_k, \delta d) - {}^m\mathbf{g}_k\|^2$ can be computed with

$$({}^m x_k - x_k - \delta d\cos(\theta_k))^2 + ({}^m y_k - y_k - \delta d\sin(\theta_k))^2, \qquad \text{(B.2)}$$

where $[{}^m x_k, {}^m y_k]^T$ denotes the position of the subgoal m, and $[x_k, y_k]^T$ and θ_k, respectively denote the robot's position and heading at time instant k. Intuitively, minimizing the energy function in (B.1) minimizes the squared distance to all goals jointly, where the importance of each goal is weighted by its squared probability. For instance, when all the probability is concentrated on one goal (e.g. goal 2 in Fig. B1), δd_{opt} will be chosen so that it brings the robot very close to its corresponding subgoal. However, it is rare that IR converges to a point mass probability vector $\mathbf{P}_k$, and therefore the distance to all goals are taken into consideration in the minimization of (B.1). Noteworthy here is that the effectiveness of δd_{opt} is highly dependent on the accuracy of the $\mathbf{P}_k$.

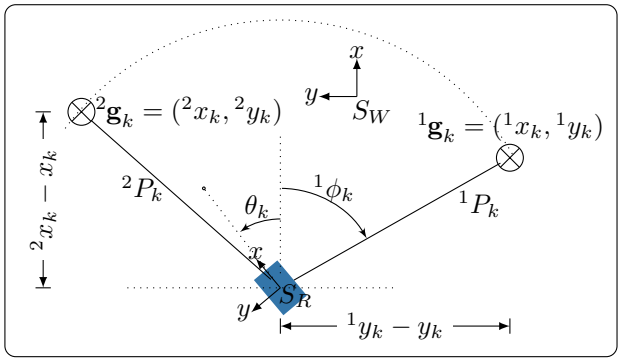


Figure B1: An example illustrating the quantities used in the computation of the optimal translational and rotational steps. Computed subgoals on the path to the different end goals often lie on a circle of radius d_{subgoal}. The optimal steps are computed so as to minimize an energy function determined by attraction forces of the different goals. The angle ${}^1\phi_k = \text{atan2}({}^1y_k - y_k, {}^1x_k - x_k)$

Since the energy function is defined such that it has a single minimum, i.e.

$$\frac{\partial^2}{\partial x^2} f(\delta d) = \sum_{m=1}^{N} {}^m{P_k}^2 > 0 \qquad \forall \delta d. \qquad \text{(B.3)}$$

Solving $\frac{\partial}{\partial x} f(\delta d) = 0$, yields the following value of δd_{opt},

$$\delta d_{\text{opt}} = \frac{\sum_{m=1}^{n} {}^m{P_k}^2 \cdot [({}^m x_k - x_k)\cos(\theta_k) + ({}^m y_k - y_k)\sin(\theta_k)]}{\sum_{m=1}^{n} {}^m{P_k}^2}.$$

Appendix C. Optimal Rotational Step

Similarly, different goals apply different forces on the robot's rotational movements in proportion to their squared probabilities. The optimal rotational step $\delta\theta_{\text{opt}} \in [-\pi, \pi]$ is defined as the global minimizer of

the energy function

$$f(\delta\theta) = \frac{1}{2}\sum_m {}^mP_k{}^2 \cdot (\mathrm{d}(\mathbf{x}_k, {}^m\mathbf{g}_k) - \delta\theta))^2,$$

where $\mathrm{d}(\mathbf{x}_k, {}^m\mathbf{g}_k)$ is defined as

$$\begin{cases} \mathrm{atan2}({}^my_k - y_k, {}^mx_k - x_k) - \theta_k. & \text{if } \|\mathbf{x}_k - {}^m\mathbf{g}_k\| \geq d_g \\ {}^m\theta - \theta_k & \text{if } \|\mathbf{x}_k - {}^m\mathbf{g}_k\| < d_g \end{cases}$$

where d_g is a distance threshold, below which the robot is assumed close to an end goal. Intuitively, this means that when users get very close to specific goals, they are more likely willing to align the robot with the heading of that goal, or otherwise, they rather align the robot to face the corresponding subgoals. The value of d_g is set to 0.5 m. The optimal rotational step can be computed with

$$\delta\theta_{\mathrm{opt}} = \frac{\sum_m {}^mP_k{}^2 \cdot \mathrm{d}(\mathbf{x}_k, {}^m\mathbf{g}_k)}{\sum_m {}^mP_k{}^2}. \qquad \text{(C.1)}$$

References

[1] Jonathan R Wolpaw, Niels Birbaumer, Dennis J McFarland, Gert Pfurtscheller, and Theresa M Vaughan. Brain-computer interfaces for communication and control. *Clinical neurophysiology*, 113(6):767–791, jun 2002.

[2] Mike Chung, Willy Cheung, Reinhold Scherer, and Rajesh P. N. Rao. A hierarchical architecture for adaptive brain-computer interfacing. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence*, pages 1647–1652, 2011.

[3] Matthew Bryan, Joshua Green, Mike Chung, Lillian Chang, Reinhold Scherer, Joshua Smith, and Rajesh P. N. Rao. An adaptive brain-computer interface for humanoid robot control. In *11th IEEE-RAS International Conference on Humanoid Robots*, pages 199–204, Bled, Slovenia, 2011.

[4] Carlos Escolano, Javier Mauricio Antelis, and Javier Minguez. A telepresence mobile robot controlled with a noninvasive brain-computer interface. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics*, 42(3):793–804, jun 2012.

[5] Ori Cohen, Sébastien Druon, Sébastien Lengagne, Avi Mendelsohn, Rafael Malach, Abderrahmane Kheddar, and Doron Friedman. fMRI based robotic embodiment: a pilot study. In *IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pages 314–319, 2012.

[6] M. Alimardani, S. Nishio, and H. Ishiguro. BCI-teleoperated androids; a study of embodiment and its effect on motor imagery learning. In *2015 IEEE 19th International Conference on Intelligent Engineering Systems (INES)*, pages 347–352, 2015.

[7] Emmanuele Tidoni, Pierre Gergondet, Gabriele Fusco, Abderrahmane Kheddar, and Salvatore Aglioti. The role of audio-visual feedback in a thought-based control of a humanoid robot: a BCI study in healthy and spinal cord injured people. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 4320(c):1–1, 2016.

[8] Mel Slater and Maria Sanchez-Vives. Enhancing Our Lives with Immersive Virtual Reality. *Frontiers in Robotics and AI*, 3(December):74, 2016.

[9] Konstantina Kilteni, Raphaela Groten, and Mel Slater. The Sense of Embodiment in Virtual Reality. *Presence: Teleoperators and Virtual Environments*, 21(4):373–387, 2012.

[10] Saul Greenberg, John J. Darragh, David Maulsby, and Ian H. Witten. Predictive interfaces: What will they think of next? In A. D. N Edwards, editor, *Extra-ordinary human-computer interaction: interfaces for users with disabilities*, pages 103–140. Cambridge University Press, 1995.

[11] Richard C. Simpson, Simon P. Levine, David A. Bell, Lincoln A. Jaros, Yoram Koren, and Johann Borenstein. NavChair: an assistive wheelchair navigation system with automatic adaptation. In Vibhu O. Mittal, Holly A. Yanco, John Aronis, and Richard Simpson, editors, *Assistive Technology and Artificial Intelligence: Applications in Robotics, User Interfaces and Natural Language Processing*, pages 235–255. Springer, 1998.

[12] Antonis Argyros, Pantelis Georgiadis, Panos Trahanias, and Dimitris Tsakiris. Semi-autonomous navigation of a robotic wheelchair. *Journal of Intelligent and Robotic Systems*, 34:315–329, 2002.

[13] Sebastian Muszynski, Jörg Stückler, and Sven Behnke. Adjustable autonomy for mobile teleoperation of personal service robots. In *IEEE International Workshop on Robot and Human Interactive Communication*, pages 933–940, Paris, France, 2012.

[14] Thomas B. Sheridan and William L. Verplank. Human and Computer Control of Undersea Teleoperators. Technical report, DTIC Document, 1978.

[15] Mica R. Endsley and David B. Kaber. Level of automation eŒ ects on performance , situation. *Ergonomics*, 42(3):462–492, 1999.

[16] Eric Demeester. *User-adapted plan recognition and shared control for wheelchair driver assistance under uncertainty*. Phd thesis, Katholieke Universiteit Lueven, Belgium, 2007.

[17] Tom Carlson and Yiannis Demiris. Human-wheelchair collaboration through prediction of intention and adaptive assistance. In *IEEE International Conference on Robotics and Automation*, pages 3926–3931, 2008.

[18] Aditya Goil, Matthew Derry, and Brenna D. Argall. Using machine learning to blend human and robot controls for assisted wheelchair navigation. In *IEEE International Conference on Rehabilitation Robotics*, pages 1–6, 2013.

[19] Charles F. Schmidt, N. S. Sridharan, and John L. Goodson. The plan recognition problem: an intersection of psychology and artificial intelligence. *Artificial Intelligence*, 11(1-2):45–83, 1978.

[20] Chris L. Baker, Rebecca Saxe, and Joshua B. Tenenbaum. Action understanding as inverse planning. *Cognition*, 113(3):329–49, dec 2009.

[21] David W Albrecht, Ingrid Zukerman, and Ann E. Nicholson. Bayesian models for keyhole plan recognition in an adventure game. *User modeling and user-adapted interaction*, 8(1-2):5–47, 1998.

[22] Taro Kanno, Keiichi Nakata, and Kazuo Furuta. A method for team intention inference. *International Journal of Human-Computer Studies*, 58(4):393–413, 2003.

[23] Karen E. Lochbaum. An algorithm for plan recognition in collaborative discourse. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics*, pages 33–38, 1991.

[24] Sviatoslav Braynov. Adversarial planning and plan recognition: Two sides of the same coin. In *Secure Knowledge Management Workshop*, 2006.

[25] Eugene Charniak and Robert P Goldman. A Bayesian model of plan recognition. *Artificial Intelligence*, 64(1):53–79, 1993.

[26] Marcelo Gabriel Armentano and Analía Amandi. Plan recognition for interface agents. *Artificial Intelligence Review*, 28(2):131–162, feb 2009.

[27] Xavier Perrin, Francis Colas, Cédric Pradalier, Roland Siegwart, Ricardo Chavarriaga, and José Del R Millán. Learning user habits for semi-autonomous navigation using low throughput interfaces. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 1–6, 2011.

[28] Miquel Ramrez and Hector Geffner. Plan Recognition as Planning. In *Proceedings of the 21st international joint conference on Artifical intelligence.*, pages 1778–1783, 2009.

[29] Miquel Ramirez, Hector Geffner, Miquel Ram, and Hector Geffner. Probabilistic Plan Recognition Using Off-the-Shelf Classical Planners Example : Noisy Walk. *Artificial Intelligence*, pages 1121–1126, 2010.

[30] Kartik Talamadupula, Gordon Briggs, Tathagata Chakraborti, Matthias Scheutz, and Subbarao Kambhampati. Coordination in human-robot teams using mental modeling and plan recognition. *IEEE International Conference on Intelligent Robots and Systems*, (Iros 2014):2957–2962, 2014.

[31] Eric Demeester, Hüntemann Alexander, José del R Millán, and Hendrik Van Brussel. Bayesian Plan Recognition for Brain-Computer Interfaces. In *IEEE International Conference on Robotics and Automation*, pages 653–658, 2009.

[32] Eric Demeester, Alexander Hüntemann, Dirk Vanhooydonck, Gerolf Vanacker, Hendrik Van Brussel, and Marnix Nuttin. User-adapted plan recognition and user-adapted shared control: A Bayesian approach to semi-autonomous wheelchair driving. *Autonomous Robots*, 24:193–211, 2008.

[33] Eric Demeester, Alexander Hüntemann, Dirk Vanhooydonck, Gerolf Vanacker, Alexandra Degeest, Hendrik Van Brussel, and Marnix Nuttin. Bayesian estimation of wheelchair driver intents: Modeling intents as geometric paths tracked by the driver. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5775–5780, 2006.

[34] J B Pelz and R Canosa. Oculomotor behavior and perceptual strategies in complex tasks. *Vision research*, 41(25):3587–96, 2001.

[35] Kathleen A Turano, Duane R Geruschat, and Frank H Baker. Direction of Gaze while Walking a Simple Route : Persons with Normal Vision and Persons with Retinitis Pigmentosa. 78(9):667–675, 2001.

[36] Mohammad Abu-Alqumsan and Angelika Peer. Advancing the detection of steady-state visual evoked potentials in brain-computer interfaces. *Journal of neural engineering*, 13(3):36005, 2016.

[37] Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Ng. ROS: an open-source Robot Operating System. In *ICRA Workshop on Open Source Software*, 2009.

[38] Joseph J. LaViola Jr. A discussion of cybersickness in virtual environments. *ACM SIGCHI Bulletin*, 32(1):47–56, jan 2000.

[39] Dieter Fox. KLD-sampling: Adaptive particle filters. In *Advances in Neural Information Processing Systems 14*, 2001.

[40] Brian P. Gerkey, Richard T. Vaughan, and Andrew Howard. The Player / Stage Project : Tools for Multi-Robot and Distributed Sensor Systems. In *Proceedings of the International Conference on Advanced Robotics (ICAR 2003)*, pages 317–323, 2003.

[41] Zygmunt Pizlo, Yunfeng Li, Tadamasa Sawada, and Robert M Steinman. *Making a machine that sees like us*. Oxford University Press (UK), 2014.

[42] Rainer Stiefelhagen and Jie Zhu. Head orientation and gaze direction in meetings. In *Extended Abstracts on Human Factors in Computing Systems*, pages 858–859, New York, New York, USA, 2002. ACM Press.

[43] Robert J.K. Jacob. Eye movement-based human-computer interaction techniques: Toward non-command interfaces. *Advances in human-computer interaction*, 4:151–190, 1993.

[44] Dario D. Salvucci and Joseph H. Goldberg. Identifying Fixations and Saccades in Eye-Tracking Protocols. In *Proceedings of the Eye Tracking Research and Applications Symposium*, pages 71–78, 2000.

[45] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved Techniques for Grid Mapping. *IEEE Transactions on Robotics*, 23(1):34–46, 2007.

[46] Han The Anh and Luis Moniz Pereira. Corpus-Based Incremental Intention Recognition via Bayesian Network Model Construction. In *Proceedings of the First Workshop on Goal, Activity and Plan Recognition*, pages 1–8, 2011.

[47] Marcelo Gabriel Armentano and Analía Amandi. Goal Recognition with Variable-Order Markov Models. In *International Joint Conference on Artificial Intelligence*, pages 1635–1640, 2009.

[48] Nate Blaylock and James Allen. Recognizing instantiated goals using statistical methods. In *IJCAI Workshop on Modeling Others from Observations*, 2005.

[49] Alexandra Kuznetsova, Rune H. B. Christensen, Cecile Bavay, and Per Bruun Brockhoff. Automated mixed ANOVA modeling of sensory and consumer data. *Food Quality and Preference*, 40:31–38, 2014.

[50] Torsten Hothorn, Frank Bretz, and Peter Westfall. Simultaneous Inference in General Parametric Models. *Biometrical Journal*, 50(3):346–363, 2008.

[51] Russell V Lenth. Least-Squares Means: The R Package lsmeans. *Journal of Statistical Software*, 69(1):1–33, 2016.

[52] Andrea Kübler and Donatella Mattia. Chapter 14 BrainComputer Interface Based Solutions for End-Users with Severe Communication Disorders. In Steven Laureys, Olivia Gosseries, and Guilio Tononi, editors, *The Neurology of Conciousness*, pages 217–240. 2 edition, 2016.

[53] Andrea Kübler, Nicola Neumann, Jochen Kaiser, Boris Kotchoubey, Thilo Hinterberger, and Niels P. Birbaumer. Brain-computer communication: Self-regulation of slow cortical potentials for verbal communication. *Archives of Physical Medicine and Rehabilitation*, 82(11):1533–1539, 2001.

[54] Tom Carlson, Luca Tonin, Serafeim Perdikis, Robert Leeb, and José del R Millán. A hybrid BCI for enhanced control of a telepresence robot. In *35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 3097–100, jan 2013.

[55] Sang Hyoung Lee, Sanghoon Lee, Il Hong Suh, and Wan Kyun Chung. Learning of subgoals for goal-oriented behavior control of mobile robots. In *International Conference on Neural Information Processin*, pages 64–71, 2009.

[56] Antonella Maselli and Mel Slater. The building blocks of the full body ownership illusion. *Frontiers in human neuroscience*, 7(March):83, 2013.