pre-print: *The 43rd Intl. Symp. on Robotics (ISR2012)*, Taipei, Taiwan, 29 – 31 August 2012, pp. 1106 – 1111.

ML, MAP and greedy POMDP shared control: comparison of wheelchair navigation assistance for switch interfaces

Eric Demeester, Alexander Hüntemann, Emmanuel Vander Poorten and Joris De Schutter

Department of Mechanical Engineering,
University of Leuven
Heverlee, Belgium
e-mail: eric.demeester@mech.kuleuven.be

***Abstract*—This paper describes and compares three approaches for providing navigation assistance to powered wheelchair users: a Maximum Likelihood (ML) approach, a Maximum A Posteriori (MAP) approach, and a greedy Partially Observable Markov Decision Process (POMDP) approach. The approaches are evaluated by controlling a wheelchair in simulation using a switch interface. The results show that for this experimental setup (1) all three approaches allow the driver to reach any of the specified goal positions with greater accuracy and faster than without assistance, (2) ML produces paths that are jagged, because its decisions are based on the latest user signals only, (3) MAP decisions are much less impulsive than ML, except at the start, (4) greedy POMDP is more cautious in taking actions prematurely because it considers the probability of all driver plans when evaluating the effect of an action.**

Keywords:-robotic wheelchair, plan recognition, intention estimation, switch interface, shared control.

I. INTRODUCTION

Many elderly and physically impaired people experience difficulties when manoeuvring a powered wheelchair. In order to provide improved manoeuvring, powered wheelchairs have been equipped with sensors, additional computing power and intelligence by various research groups. This paper presents a Bayesian approach to robotic assistance for wheelchair driving, which can be adapted to a specific user. The proposed framework explicitly takes the uncertainty on the user's intent into account. Besides during intent estimation, user-specific properties and uncertainty on the user's intent are incorporated when taking assistive actions, such that assistance is tailored to the user's driving skills.

Providing appropriate assistance to a user who is himself in control of a robot requires the robotic system to decide in a proper manner to which degree corrective actions should be taken. This is a key challenge in designing intuitive and safe *shared control*, which we define as an application where control over a system is shared among one or more humans and one or more computerized controllers. Related definitions can be found in [1]. The purpose of shared control is to combine the strengths of human and machine and to reduce their weaknesses.

In order to take acceptable corrective actions, assistive robots should be aware of the plan the user has with the robot. Furthermore, in order for robot control to be easy and intuitive, we believe that users should not be required to explicitly communicate their plan prior to executing the task. For a large part of our user group, explicitly stating which task should be executed constitutes a cognitively and physically challenging or even impossible task. Instead, we consider it the task of the assistance system to infer the user's plan from the user's noisy actions and from environmental perception. This is the problem of *plan recognition* or *intention estimation* [2] (the terms *plan*, *intent* or *intention* will be used interchangeably in this paper). The problem of plan recognition is more formally defined as "taking (...) *as input a sequence of actions performed by an actor and to infer the goal pursued by the actor and also to organize the actions in terms of a plan structure*". Due to the noisy user signals or the limited set of user actions that are possible with certain user interfaces, the user's intent is inherently uncertain.

This paper evaluates three navigation assistance approaches, which were proposed in [4], to assist a wheelchair driver who adopts a switch interface to control a wheelchair in simulation. The difference with [4] is that we now apply these approaches to a robot with non-holonomic kinematics instead of a holonomic robot. We adopt a switch interface firstly because drivers with this interface typically have more navigation difficulties, given the smaller set of interface signals at their disposal. Secondly, our greedy POMDP approach is computationally expensive, and it is easier to verify its performance if fewer user signals are present.

Section II explains our approach to plan recognition in general. It is similar to the approach in [5] but is repeated here for completeness. Section III explains the benchmark test that will be used to compare the shared control approaches. Section IV details some concrete choices for the plan recognition framework for button interfaces. The difference with [5] is that we adopt a user model that has been learned from driver data, rather than a manually tuned user model. Section V then explains the shared control approaches, which are compared in Section VI.

II. BAYESIAN PLAN RECOGNITION FOR BUTTON INTERFACES: GENERAL APPROACH

The Bayesian plan recognition approach that is briefly discussed in Section II.C has been presented in several papers, e.g. [4] and [5]. We have shown that this framework can be adopted for both discrete and continuous interfaces, and that the framework can be adapted to drivers with different abilities.

The purpose of our Bayesian plan recognition framework is threefold. First, we would like to recognize even complex user plans such as parking maneuvers in narrow spaces in an accurate way. Second, the uncertainty on these plan estimates should be determined. Third, the estimation algorithm should take general human characteristics into account such as the maximum possible human control bandwidth, and should also be adaptive to the specific user that is interacting with the robot. In order to realize these requirements, the following representation of driver plans is chosen.

A. Representation of driver plans $\boldsymbol{i}$

Generally speaking, wheelchair drivers want to reach a certain goal pose $\boldsymbol{p_{goal}} = [x_{goal}\ \ y_{goal}\ \ \theta_{goal}]^T$ with a certain goal velocity $\boldsymbol{t_{goal}} = [v_{goal}\ \ \omega_{goal}]^T$, where θ_{goal} represents the robot orientation at the goal position $[x_{goal}\ \ y_{goal}]^T$, and where v_{goal} denotes the desired linear velocity and ω_{goal} the desired rotational velocity at $\boldsymbol{p_{goal}}$. A velocity $\boldsymbol{t}$ and pose $\boldsymbol{p}$ will be represented jointly as the robot state $\boldsymbol{x}$. A user plan or *intention* $\boldsymbol{i_k}$ at time k can then be generically described as a trajectory or sequence of robot states:

$$\boldsymbol{i_k} = \{\boldsymbol{x_{current}}, \cdots, \boldsymbol{x_{goal}}\}, \tag{1}$$

which the user has in mind to achieve the goal state $\boldsymbol{x_{goal}}$ from the current robot state $\boldsymbol{x_{current}}$.

B. Generation of driver plans $\boldsymbol{i_k}$

Hypotheses regarding user plans can be generated in a variety of ways, e.g. using a two-step approach: first, all plausible goal state candidates are generated, and in a second phase all trajectories to these goal states. In the first step, goal state candidates can be learned by remembering at which poses the user stands still for a certain amount of time. Furthermore, a user or physiotherapist can generate additional goal state hypotheses by indicating on an estimated map all possibly interesting places. In a second step, trajectories $\{\boldsymbol{x_{current}}, \cdots, \boldsymbol{x_{goal}}\}$ to the candidate goal states are generated. The trajectories are calculated by a motion planner. In order to be able to recognize also complex user plans such as docking and parking maneuvers, the fine motion planner takes the robot's kinematics, orientation and geometry explicitly into account.

C. Probability distribution over driver plans

A probability distribution over the generated driver plan hypotheses is maintained. Typically, at start-up, the probability function over user plans is modeled to be a uniform distribution, since at that time nothing is known regarding the goal the driver wants to go to. In the future, additional information can be taken into account, such as where the user typically wants to drive to at that time of the day.

Assuming that information from at most m past time steps influences the user plan and user signal at time k, the probability distribution can be updated as follows. Based on the robot state encoded as $\boldsymbol{x_k}$ and the actual interface signals $\boldsymbol{u_k}$ the driver gives, the probability function over $\boldsymbol{i_{k-m:k}}$ becomes:

$$\begin{aligned} & p_k\,(\boldsymbol{i_{k-m:k}} \mid \boldsymbol{u_{k-m:k}}) \\ &= p_{user}\,(\boldsymbol{u_k} \mid \boldsymbol{i_{k-m:k}}, \boldsymbol{u_{k-m:k-1}}) \\ &\quad \cdot p_{process}\,(\boldsymbol{i_k} \mid \boldsymbol{i_{k-m:k-1}}, \boldsymbol{u_{k-m:k-1}}) \\ &\quad \cdot p_{k-1}\,(\boldsymbol{i_{k-m:k-1}} \mid \boldsymbol{u_{k-m:k-1}}) \cdot \eta \\ &\quad \text{(see text, point 6)} \\ &= p_{user}\,(\boldsymbol{u_k} \mid \boldsymbol{i_{k-m+1:k}}, \boldsymbol{u_{k-m+1:k-1}}) \\ &\quad \cdot p_{process}\,(\boldsymbol{i_k} \mid \boldsymbol{i_{k-m:k-1}}, \boldsymbol{u_{k-m:k-1}}) \\ &\quad \cdot p_{k-1}\,(\boldsymbol{i_{k-m:k-1}} \mid \boldsymbol{u_{k-m:k-1}}) \cdot \eta \end{aligned} \tag{2}$$

Since the robot state $\boldsymbol{x_k}$ is encoded in our user plan representation $\boldsymbol{i_k}$, $\boldsymbol{x_k}$ and actions $\boldsymbol{a_k}$ do not explicitly appear in Eq. (2). Furthermore, in this equation:

1) p_{k-1} is the a priori distribution over user intents, given previous user signals $\boldsymbol{u_{k-m:k-1}} = \{\boldsymbol{u_{k-m}}, \cdots, \boldsymbol{u_{k-1}}\}$. It reflects the belief in the different possible user intents prior to having moved and prior to having taken new user signals into account.
2) p_{user} is the *user model*, which expresses the likelihood that the user gives the observed interface signal $\boldsymbol{u_k}$, given that the user has had intent evolution $\boldsymbol{i_{k-m+1:k}}$, and given previous user signals $\boldsymbol{u_{k-m:k-1}}$.
3) $p_{process}$ is the *plan process model*, which determines both the shape and the probability of a user plan $\boldsymbol{i_k}$ at time k, given that the user has had intent evolution $\boldsymbol{i_{k-m:k-1}}$.
4) p_k is the a posteriori distribution over user intents, i.e. the probability of the different user plans after user signals and wheelchair motion have been taken into account.
5) η is a scale factor to normalize the probability distribution.
6) the assumption made in the second equation corresponds to the assumption that only a time window of size m is adopted.

A priori determined parameters such as maps or user model parameters have been left out for notational simplicity. They all appear at the right side of the conditional probability functions. Because of the assumption that only information in the limited time window m is required, marginalizing p_k over $\boldsymbol{i_{k-m}}$ allows to keep the state size fixed:

$$\begin{aligned} & p_k\,(\boldsymbol{i_{k-m+1:k}} \mid \boldsymbol{u_{k-m+1:k}}) \\ &= \sum_{i_{k-m}} p_k\,(\boldsymbol{i_{k-m:k}} \mid \boldsymbol{u_{k-m:k}}) \\ &= \eta \cdot p_{user}\,(\boldsymbol{u_k} \mid \boldsymbol{i_{k-m+1:k}}, \boldsymbol{u_{k-m+1:k-1}}) \\ &\quad \cdot \sum_{i_{k-m}} \Big(p_{process}\,(\boldsymbol{i_k} \mid \boldsymbol{i_{k-m:k-1}}, \boldsymbol{u_{k-m:k-1}}) \\ &\quad \cdot p_{k-1}\,(\boldsymbol{i_{k-m:k-1}} \mid \boldsymbol{u_{k-m:k-1}}) \Big) \end{aligned} \tag{3}$$

This is also formulated and executed as a prediction and correction step. The *prediction* or *time update* step corresponds to the calculation of the sum in Eq. (3). This step predicts the user plans and their probability after a robot action has been executed. The *correction* step takes the user signals into account based on the user model. The

new state becomes $\boldsymbol{i_{k-m+1:k}}$. The shared control approaches presented in Section V only require knowledge of p_k ($\boldsymbol{i_k} \mid \boldsymbol{u_{k-m+1:k}}$), which is obtained by marginalising over $\boldsymbol{i_{k-m+1:k-1}}$:

$$p_k(\boldsymbol{i_k} \mid \boldsymbol{u_{k-m+1:k}}) = \sum_{i_{k-m+1:k-1}} p_k(\boldsymbol{i_{k-m+1:k}} \mid \boldsymbol{u_{k-m+1:k}}) . \quad (4)$$

III. BENCHMARK TEST

This section defines the benchmark test that is used to evaluate the shared control approaches, see Fig. 1. The user starts in the middle of a circle of possible goal locations. Using a switch interface containing 3 buttons (*left*, *right*, *forward*) the driver tries to execute straight-line paths to different goal locations positioned in a circle, where the driver is either in *full control* of the wheelchair, or gets *ML, MAP or POMDP assistance*. In order to get rid of difficulties in wheelchair control caused for example by friction between castor wheels and the ground, and in order to more clearly show to the user which path to follow, the benchmark test has been performed with a simulated wheelchair. This test does not include any obstacles, in order to verify whether the navigation assistance approach is able to assist users even in the absence of obstacles. Most existing approaches are not able to do this as they rely on the detection of obstacles and on the subsequent activation of obstacle avoidance algorithms to assist the driver. Furthermore, the purpose of this setup is to verify to which degree the driver can go to any position (s)he likes, and to verify whether our plan recognition algorithm will estimate this goal position quickly and correctly.

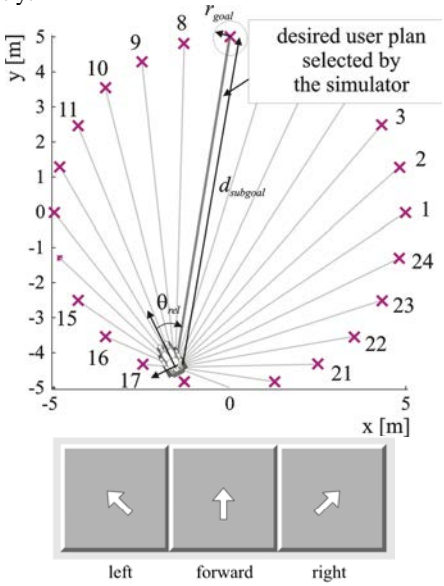

Fig. 1. Benchmark test for comparing approaches to shared control. The driver is asked to drive to one of 24 goal locations positioned on a circle, using a switch interface with 3 buttons (left, right, forward). The figure also defines the variables that may have an impact on the user model p_{user}: the relative angle θ_{rel} of the goal location w.r.t. the wheelchair, and the distance $d_{subgoal}$ to the goal location.

The simulator selects a goal position at random, and indicates this to the user via a line from the robot pose to the goal position, as shown in Fig. 1. This way, ground truth regarding driver plans is known, which can be used to evaluate the plan recognition performance. In order to make navigation intuitive, the simulator always shows a top view fixed to the wheelchair, such that the user has a view that moves together with the wheelchair rather than having a fixed view of the environment.

The commands *left*, *forward*, and *right* respectively correspond to turning $\Delta\alpha$ = 0.5 rad counterclockwise, moving Δd = 1 m forward, and turning $\Delta\alpha$ = 0.5 rad clockwise. The robot is nonholonomic but can turn on the spot. Due to the discrete nature of robot actions with the switch interface, arbitrary goal positions cannot be reached exactly. Therefore, the simulator considers a goal position to be reached if the robot comes within a circle with radius r_{goal} = 0.5 m of the goal position, after which a new goal position is immediately selected. Fig. 2 shows the positions that are reachable using the chosen navigation resolution $\Delta\alpha$ and Δd, as well as the accuracy with which goal locations can be reached (all goal locations can be reached within 10 cm or less).

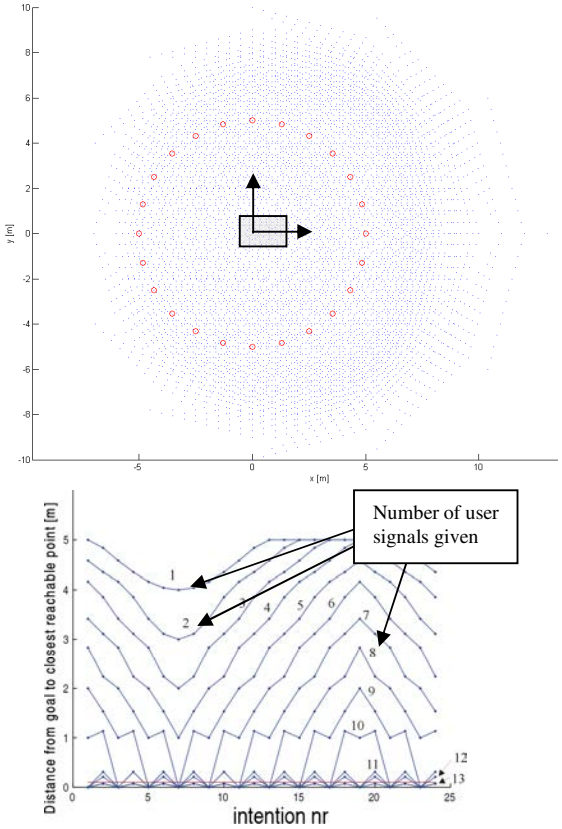

Fig. 2. The top figure shows the positions (blue) that are reachable from the center, using rotations over $\Delta\alpha$ = 0.5 rad and translations over Δd = 1 m, and this with a sequence of at most 13 user signals. The driver starts in the middle with orientation 0 rad, as shown by the coordinate frame. Since the driver first has to turn in order to reach positions to the left, more positions are reachable to the right with a sequence of at most 13 switch pushes. The 24 goal locations are also shown as red circles. The bottom figure shows the minimal distance to each of the 24 goal locations, depending on the given number of user signals. It is shown that with 13 user signals, each goal location can be reached from the centre with an accuracy of 0.1 m or less.

IV. BAYESIAN PLAN RECOGNITION FOR BUTTON INTERFACES: CONCRETE CHOICES

The following sections describe in more detail the concrete choices for the elements in our Bayesian plan recognition framework, specific to the proposed benchmark test and switch interface.

A. User plan representation

Since no obstacles are present, user plans can simply be represented as straight-line paths from the current robot location to the goal locations. The user is assumed to first turn over the shortest angle in the direction of the goal location, and then to move forward. Hence, for this benchmark test, no motion planner is required, which allows to easily reproduce the benchmark test. The set of goal positions of all user plans is assumed to be known a priori. The 24 crosses in Fig. 1 correspond to these global goal positions. The plan recognition algorithm is aware of the robot position and the goal locations.

B. User Model

The user model p_{user} ($\boldsymbol{u_k} \mid \boldsymbol{i_{k-m+1:k}}, \boldsymbol{u_{k-m+1:k-1}}$) receives as input the user's plan, i.e. a straight path from the current pose to a goal position. The goal may be reached with any end orientation. For this benchmark test, the user is modelled to adopt only the current path to the goal location ($m = 1$) to produce a new user signal, hence the user model can be simplified to p_{user} ($\boldsymbol{u_k} \mid \boldsymbol{i_k}$).

The user model is calibrated by asking the user to drive to various goal locations in his neighbourhood using the switch interface. The user is in full control of the wheelchair during this calibration phase. Fig. 3 (left) plots which user signal (left, right or forward) the user gives depending on the relative location (angle and distance) of the goal. From these experimental data, it follows that the user signals mainly seem to be dependent on the relative angle, and less on the distance to the goal. Therefore, we have chosen to make our user model only dependent on the relative angle θ_{rel}. Fig. 3 (right) shows a histogram obtained from the data in Fig. 3 (left). This histogram is used as our likelihood model p_{user} ($\boldsymbol{u_k} \mid \boldsymbol{i_k}$). The obtained result is intuitively plausible: a user gives signal *left* if $\theta_{rel} \in [0, \pi]$, signal *right* if $\theta_{rel} \in [-\pi, 0]$, and signal *forward* if $\theta_{rel} \approx 0$. Around $\theta_{rel} \approx 0$ and $\theta_{rel} \approx \pi$, more uncertainty exists regarding the signal the user will give. As could be expected, the region of increased uncertainty is at the borders of the interval·$[-\Delta\alpha, \Delta\alpha]$.

C. Plan Process Function

Due to the absence of obstacles, the plan process function $p_{process}$ ($\boldsymbol{i_k} \mid \boldsymbol{i_{k-m:k-1}}, \boldsymbol{u_{k-m:k-1}}$) is straightforward as well. Each time the robot moves from pose $\boldsymbol{x_{k-1}}$ to pose $\boldsymbol{x_k}$, the straight path between $\boldsymbol{x_{k-1}}$ and the j-th goal is transformed into the straight path between $\boldsymbol{x_k}$ and the j-th goal. The j-th probability is completely transferred to this new path.

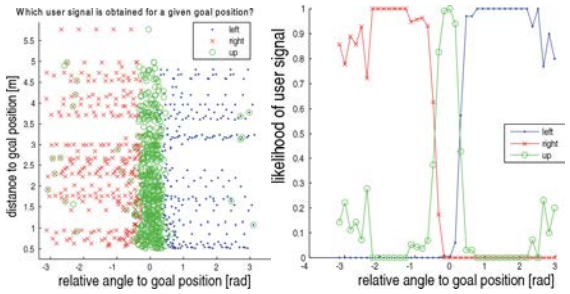

Fig. 3. Figure (left) shows the signals (*left* as blue points, *right* as red crosses, and *forward* as green circles) a user would give depending on the relative position (relative angle θ_{rel} and distance $d_{subgoal}$) of the desired goal location. Figure (right) shows the histogram extracted from the data in the left figure. This histogram is only a function of the relative angle of the goal location, and it represents the user model p_{user} ($\boldsymbol{u_k} \mid \boldsymbol{i_k}$).

V. ML, MAP AND GREEDY POMDP SHARED CONTROL

This section explains three probabilistic approaches to provide navigation assistance to a wheelchair driver, based on the estimated intention. These approaches will be evaluated and compared to user control in Section VI. As shown in Fig. 4, the ML approach only uses the most recent user signal $\boldsymbol{u_k}$ to determine a navigation decision $\boldsymbol{a_k}$. Many existing approaches to wheelchair navigation assistance are of this type. However, many approaches are not expressed in the Bayesian formalism, nor do they represent intentions as paths or trajectories, but rather as "behaviours" such as avoid-collision, avoid-obstacle, drive-through-door, follow-wall, etc. In comparison, the MAP approach additionally adopts all past information to determine a navigation decision $\boldsymbol{a_k}$. The greedy POMDP approach considers all a posteriori probabilities, as well as the effects of possible actions. This approach chooses those actions that maximize a *reward function r*. Some further background can be found in [4]. This section will mainly focus on implementation choices specific for the adopted switch interface.

A. Shared control actions $\boldsymbol{a_k}$ and reward function r

Navigation assistance action $\boldsymbol{a_k}$ corresponds to driving to a specific location in the robot's neighborhood, just as it was in the case where the user was in full control. In the shared control case however, the robot has more actions at its disposal as compared to the three actions for full user control mode, allowing it to move both more precisely and farther. Provided that the robot takes the right decisions, this gives it the ability to offer the desired assistance. For our comparison, we chose for shared control actions that may be three times as accurate, and may result in motions twice as far as compared to the user control actions. Furthermore, shared control actions may exist of a sequence of two control actions, a rotation followed by translation. Therefore, the shared control algorithm can allow the user to move both faster and more accurately. The adopted reward function r ($\boldsymbol{a_k}, \boldsymbol{i_k}, \boldsymbol{u_k}$) only includes a term that determines how well action $\boldsymbol{a_k}$ brings the robot towards the goal position of user plan $\boldsymbol{i_k}$. It is chosen to be the same for the three approaches.

B.Maximum Likelihood and Maximum A Posteriori

The ML and MAP approach presented in this work only differ in the function they maximise to determine the optimal robot action. ML chooses the user plan $\boldsymbol{i_{ML}}$ that maximises the likelihood function or user model:

$$\boldsymbol{i_{ML}} = \arg \max_{i_k} \; p_{user} (\boldsymbol{u_k} \mid \boldsymbol{i_{k-m+1:k}} , \boldsymbol{u_{k-m+1:k-1}}), \quad (5)$$

whereas MAP chooses the user plan $\boldsymbol{i_{MAP}}$ that maximises the posterior probability:

$$\boldsymbol{i_{MAP}} = \arg \max_{i_k} \; p_k (\boldsymbol{i_k} \mid \boldsymbol{u_{k-m+1:k}}), \quad (6)$$

where $p_k (\boldsymbol{i_k} \mid \boldsymbol{u_{k-m+1:k}})$ is obtained from Eq. (4).

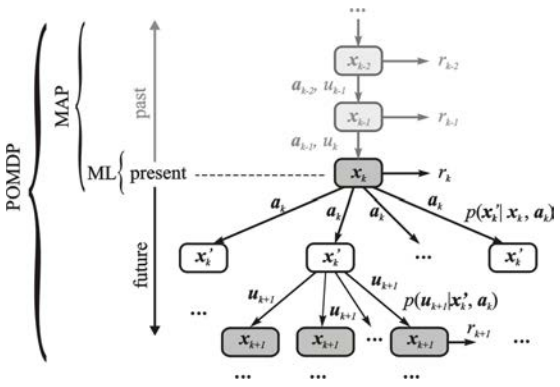

Fig. 4. The displayed decision tree illustrates the difference between Maximum Likelihood (ML), Maximum A Posteriori (MAP) and Partially Observable Markov Decision Processes (POMDP) approaches to shared control. ML takes shared control actions based on the latest sensor and user signals only, MAP additionally takes previously estimated user plans into account to take actions, and POMDP shared control additionally looks into the future and evaluates the effects of actions prior to choosing an action.

After selection of the most likely or most probable user plan, the robot action can be determined. The pseudo-code for the ML and MAP algorithms is shown in Algorithm 1. The algorithm first determines the mental trajectory that is most likely for ML or most probable for MAP (line 4). For all possible robot actions $\boldsymbol{a_j} \in A$, the reward of the action is computed. The action set A denotes the set of applicable actions in robot state $\boldsymbol{x_k}$, thereby eventually dealing with robot geometry, kinematics and dynamics. The optimal action $\boldsymbol{a_{opt}}$ is the action $\boldsymbol{a_j}$ with maximum reward r_j (line 7).

C.Greedy Partially Observable Markov Process

The ML and MAP approaches do not take possible ambiguity over the set of user plans into account. Therefore, they can be expected to yield good results if the probability function over user plans or over user signals is sharply unimodal. In case of multi-modal probability functions, their performance may decrease if user plans corresponding to the multiple modes require different assistive actions. POMDP approaches explicitly deal with this ambiguity, and are therefore expected to perform better in such cases.

The state in the POMDP description will be modelled to comprise the robot state $\boldsymbol{x_k}$ and the user plan $\boldsymbol{i_k}$. For the performed experiments, only uncertainty on the user plans $\boldsymbol{i_k}$ is considered. The robot state will be considered to be fully observable and robot actions will be assumed to be executed deterministically and modelled perfectly. Given the computational complexity of POMDPs, actions are determined only for the *current* belief state in this evaluation, and a finite-horizon measure is adopted.

The pseudo-code for the greedy POMDP algorithm is shown in Algorithm 2. When the assistive robot has one step remaining, all it can do is take a single action. In contrast to the ML and MAP approaches, that action is now chosen that maximises the expected reward, i.e. it takes the belief in the different user plans into account. This is performed in the algorithm in line 7, where reward r_m for user plan $\boldsymbol{i_m}$ is weighted with belief b_k ($\boldsymbol{i_m}$) for that user plan. The set of user plans at time k is denoted as I_k. If the robot only looks one time step ahead, lines 8-14 should be omitted.

With two steps to go, the assistive robot can take an action $\boldsymbol{a_j}$, make an observation $\boldsymbol{u_t}$, and then take another action $\boldsymbol{a_o}$. This is described in lines 8 to 14. In line 9, b_{k+1} is calculated for all user plans in I_{k+1} at time k+1, based on the taken robot action $\boldsymbol{a_j}$ and the possible user signals $\boldsymbol{u_t} \in U_{k+1}$. In lines 10 to 13 the rewards of all second actions $\boldsymbol{a_o}$ are calculated in a similar way as the reward for the first action $\boldsymbol{a_j}$ in lines 5 to 7. In line 14, the reward of the second action $\boldsymbol{a_o}$ with the highest reward is added to the reward of the first action $\boldsymbol{a_j}$, multiplied with a discount factor γ.

ALGORITHM 1: ML OR MAP SHARED CONTROL

1: Input: (a) p = probability function p_k or likelihood function p_{user}
2: (b) user signal $\boldsymbol{u_k}$
3: Output: robot command $\boldsymbol{a_{opt}}$ at time k
4: $\boldsymbol{i_{opt}} \leftarrow \arg \max_{i_k} \; p$
5: **for** j = 1 **to** $|A|$ **do**
6: $r_j \leftarrow r (\boldsymbol{a_j}, \boldsymbol{i_{opt}}, \boldsymbol{u_k})$
7: $\boldsymbol{a_{opt}} \leftarrow \arg \max_{a_j} \; r_j$

ALGORITHM 2: GREEDY POMDP SHARED CONTROL

1: Input: (a) b_k = probability function p_k at time k
2: (b) user signal $\boldsymbol{u_k}$
3: Output: robot command $\boldsymbol{a_{opt}}$ at time k
4: **for** j = 1 **to** $|A|$ **do**
5: $r_j \leftarrow 0$
6: **for** m = 1 **to** $|I_k|$ **do**
7: $r_m \leftarrow b_k (\boldsymbol{i_m}) \cdot r (\boldsymbol{a_j}, \boldsymbol{i_m}, \boldsymbol{u_k})$
8: **for** t = 1 **to** $| U_{k+1} |$ **do**
9: compute $b_{k+1} (\boldsymbol{i_n} | \boldsymbol{u_t}, \boldsymbol{a_j})$ using Bayes' rule
10: **for** o = 1 **to** $|A|$ **do**
11: $r_o \leftarrow 0$
12: **for** n = 1 **to** $|I_{k+1}|$ **do**
13: $r_o \leftarrow r_o + b_{k+1} (\boldsymbol{i_n}) \cdot r (\boldsymbol{a_o}, \boldsymbol{i_n}, \boldsymbol{u_t})$
14: $r_m \leftarrow r_m + \gamma \cdot \max_{a_o} \; r_o$
15: $r_j \leftarrow r_j + r_m$
16: $\boldsymbol{a_{opt}} \leftarrow \arg \max_{a_j} \; r_j$

VI. EXPERIMENTS AND DISCUSSION

Fig. 5 shows some plan recognition results. The algorithm works satisfactorily (it evolves to the correct intentions), but it can be seen that the probability distribution is not always clearly unimodal, especially at the beginning of a maneuver.

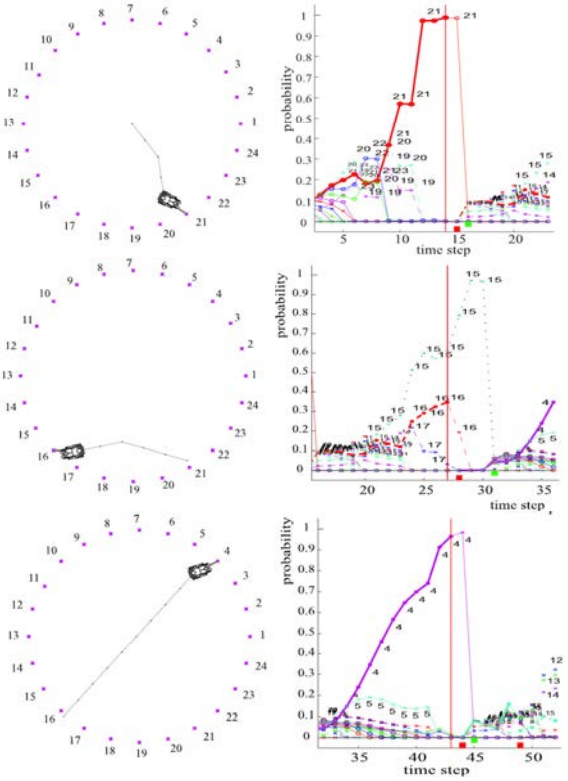

Fig. 5. Shown are several snapshots of trajectories to different goal positions, together with the probability function evolution during the trajectory.

Fig. 6 shows some quantitative results obtained with the three shared control approaches. In this concrete experiment, the user was asked to subsequently go to goal positions 10, 24, 16, 4, and then to repeat this sequence once again. The main conclusions are that (1) all three approaches allow the driver to reach any of the specified goal positions with greater accuracy and faster than without assistance, (2) ML produces paths that are jagged, because its decisions are based on the latest user signals only, (3) MAP decisions are much less impulsive, except at the start or after detection of a driver plan change, (4) greedy POMDP is more cautious in taking actions prematurely because it considers the probability of all driver plans when evaluating the effect of an action. Because various user plans are taken into account simultaneously, the POMDP shared control actions do not always try to align the wheelchair with one of the 24 goal positions, as is the case in ML and MAP shared control. Nevertheless, MAP and POMDP do not seem to perform very differently in this setup. Additional quantitative and qualitative analyses are required to confirm these preliminary results. Furthermore, more advanced POMDP approaches and reward functions will be investigated.

ACKNOWLEDGMENT

This research has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 - Challenge 2 - Cognitive Systems, Interaction, Robotics - under grant agreement No. 248873-RADHAR.

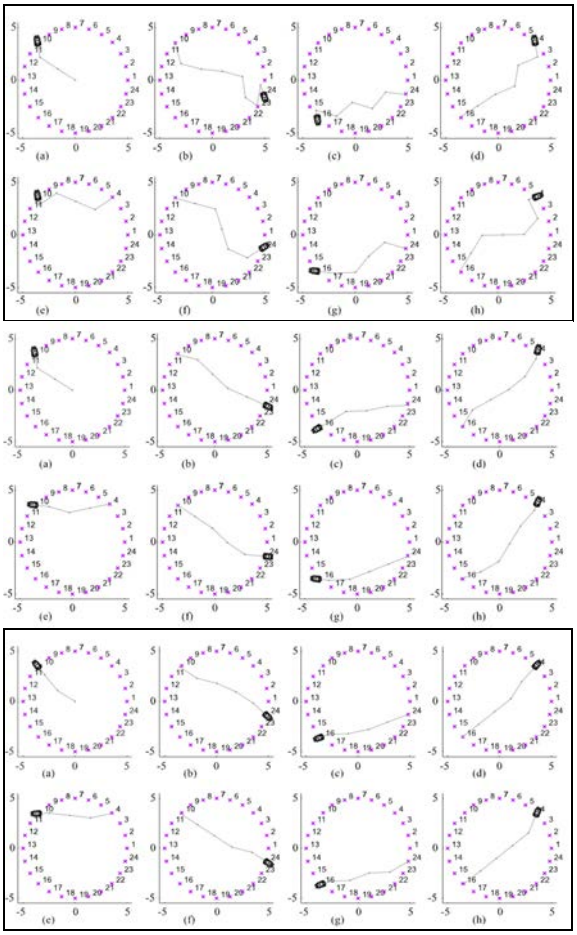

Fig. 6. Maneuvers executed with ML (top 8), MAP (middle 8) and with one step look-ahead greedy POMDP (bottom 8) activated.

REFERENCES

[1] Thomas B. Sheridan, "Telerobotics, Automation, and Human Supervisory Control," *The MIT Press*, 1992.

[2] Sandra Carberry, "Techniques for Plan Recognition," *User Modeling and User-Adapted Interaction*, Vol. 11, No. 1–2, pp. 31–48, 2001.

[3] "Reinventing the Wheelchair - Autonomous Robotic Wheelchair Projects in Europe Improve Mobility and Safety," *IEEE Robotics and Automation Magazine*, Vol. 8, No. 1, March 2001.

[4] 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*, Vol. 24, No. 2, pp. 193 – 211, February 2008.

[5] Eric Demeester, Alexander Hüntemann, José del R. Millán and Hendrik Van Brussel, "Bayesian Plan Recognition for Brain-Computer Interfaces," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, May 2009.