2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

A novel shared position control method for robot navigation via low throughput human-machine interfaces

Dmitry A. Sinyukov and Taşkın Padır[1]
Email: {dsinyukov, tpadir}@northeastern.edu

Abstract— **In this paper, we analyze systems with low throughput human-machine interfaces (such as a brain-computer interface, single switch interface) from the controls perspective. We develop some principles for performance improvement in such systems based on the parallelization of inference and robot motion. The proposed principles are used to design a novel shared position control to navigate a circular massless holonomic robot in a known environment. The system is implemented in simulation and integrated with a real robotic wheelchair. Robot experiments demonstrated the viability of the proposed navigation method in various modes of operation.**

I. Introduction

The ultimate goal of robotics is to improve human life. For a wide range of people with limited upper-body mobility, however, interaction with robots remains a challenging problem, since robots (wheelchairs, manipulators) are typically equipped with joystick interfaces often inaccessible for such users.

A number of nonconventional human-machine interfaces (HMIs) have been designed to accommodate this audience [1]. People who maintained some mobility can often operate a single switch interface. It is a button or a sensor panel accompanied with a "scanning interface" that displays options one by one. When the desired option is presented, operator pushes the button. With the Sip-and-Puff system [2], the operator can send four distinctive commands to the wheelchair by blowing air in or out of a tube. Another approach is to use gaze tracking and blink detection[3]. Good results in navigating a wheelchair have been demonstrated with Tongue Drive System [4]. Facial expressions [5] and voice control [6] have also been successfully employed for navigation tasks.

For people in a more severe condition, such as a Locked-in Syndrome (LIS), the only control modality is a brain-computer interface (BCI). LIS patients maintain full cognitive abilities, but are unable to physically interact with the world due to a trauma or a degenerative disease, such as Amyotrophic Lateral Sclerosis (ALS). A successful usage of BCI has been demonstrated in a wide range of applications: a computer mouse cursor control [7], a typing interface [8], quadcopter, humanoid, mobile robot [9], and wheelchair control [10].

The discussed HMIs are commonly known as *low throughput human-machine interfaces* (LTI) [11], where the "throughput" refers to the information transfer rate from the human to the machine. An integration of LTIs with robots is an open research topic [9]. The simplest and historically the first method is *Direct Control*. It maps LTI commands directly to robot velocities or accelerations, but it is usually impractical even for basic navigation due to collisions with obstacles. A more advanced approach is to combine robot and human intelligence in a *Shared Control* unit [12], [13]. For navigation tasks, it can be implemented as *Shared steering control* or *Shared position control* (relevant terms in the BCI community are *process control* and *goal selection* [14]).

Shared steering control interprets the LTI commands as steering directions and combines them with obstacle information and/or intent prediction to improve the navigation process. Various modes of command fusion have been explored: in [15], the commands that lead to collisions are simply discarded, in [16] obstacle avoidance is implemented using predefined zones around the robot, and in [17] it is based on artificial potential fields. Another approach is *opening selection*, when robot identifies valid directions based on the sensor data, and the operator selects one of them. It was implemented on a robotic wheelchair [18] and a UAV [19]. One advantage of shared steering control is that it naturally parallelizes the inference and navigation processes. At the same time, it requires the operator's participation until the goal is reached and makes it nearly impossible to achieve a fine control over the destination position. The shared position control developed in this work does not have these limitations.

Shared position control systems use LTIs only to infer the user's intended destination and perform the navigation autonomously. The simplest approach is to directly map the LTI commands to predefined destinations [20]. If there are too many possible destinations, they can be iteratively grouped into subsets to identify the goal [21], [22], [23]. The disadvantage of these methods is that the robot starts moving only when the destination is inferred which increases the total control time. More intelligent systems incorporate probabilistic predictions of the destination [24]. At the border between the steering and position control, there's a notable work [25], where navigation and inference were modeled as a partially observable Markov decision process (POMDP). Intended trajectory, user commands, and wheelchair actions are taken as POMDP state, observations and actions respectively. Another use of a POMDP model is demonstrated in [26] for a wheelchair navigation using speech control. In their model, POMDP states are the possible destinations,

[1]Department of Electrical and Computer Engineering, Northeastern University, Boston, MA, 02115

observations are taken from the spoken command, while actions can be either asking a question or commanding a wheelchair to move. These POMDP-based approaches enable probabilistic reasoning and certain optimality (in the POMDP sense), but, as opposed to the method developed in this work, are limited to a set of predefined destinations.

Another position control approach that parallelizes navigation and inference is proposed in [27] where several local goals are presented to the operator in a 3D model of the real environment. Operator selects the local goal, wheelchair moves there, and a new set of local goals is presented. In [28], a similar method is implemented, but they also automatically detect potential points of interest. These step-by-step techniques suffer from either a low speed of movement or a low resolution of the goal selection, and can't maximize the information transfer rate. In [29], a quatitative comparison of several previously known wheelchair control paradigms is presented.

As outlined above, in this work, we propose and develop a novel approach to LTI navigation that addresses the limitations of all shared navigation methods known to the authors. It enables navigation to any point on a known map with accuracy limited by the map resolution, it parallelizes the inference and navigation processes, uses probabilistic inference and maximizes the information transfer rate, has configurable policies for waypoint selection, allows for mostly smooth navigation, accounts for map topology, supports any number of discrete commands in an HMI, enables sophisticated probabilistic models of user navigation habits that may include popular areas (not just individual destinations).

We named our approach NoVeLTI as an acronym for Navigation Via a Low Throughput human-machine Interface. This is how it works (refer to the video attachement): 1) we display a navigation map divided into in N regions, 2) the operator selects the region with his intended destination using an LTI, 3) the intent estimate probability distribution is updated 4) a new waypoint location is calculated 5) the robot starts moving towards the new waypoint 6) the process repeats from 1) until the destination is inferred and the robot reaches it.

In Section II we propose a formal definition of control systems with LTIs and analyze them. Based on the defined models, in section III we generate principles of optimal design of *Shared Control* and exemplify them by developing a novel *Shared position control* for a massless holonomic robot navigating in a known 2D environment. Section IV, presents results of comparative simulation experiments. In section V, the designed *Shared position control* is adopted for a robotic wheelchair, the results of real robot experiments are presented to demonstrate the viability of our approach. Section VI discusses the overall results which are further summarized in section VII.

II. Problem Analysis

A. Definition of a control system with a LTI

The adjective "low" in the term *low throughput HMI* is typically understood w.r.t. the information transfer rates

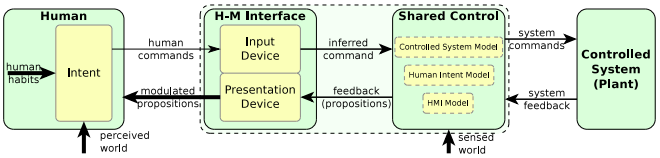


Fig. 1: Human-interface-object model

of conventional HMIs [11]. From the design perspective, however, the critical factor is the relation between the average time T_{exec} of executing a *task* by a given control system and the average time T_{inf} of inferring the task specification via an LTI. When they are of the same order of magnitude ($T_{exec} \sim T_{inf}$), we call such system a *control system with a low throughput HMI*. If $T_{exec} \gg T_{inf}$, the inference delay can be ignored reducing the problem to a standard control problem, whereas if $T_{exec} \ll T_{inf}$, the system inertia can be ignored reducing the problem to an information transfer optimization problem, but when $T_{exec} \sim T_{inf}$, we can parallelize task inference and task execution, effectively creating a new type of a control problem.

Using the state space terminology, T_{exec} is the average time of moving between two states in the state space $\mathbb{X}$ or the output space $\mathbb{Y}$, while T_{inf} is determined by the information transfer rate and the amount of information required to encode a state vector with the desired accuracy.

B. Human-HMI-Object model

Figure 1 demonstrates a general model of human-robot interaction via an HMI. For a joystick-controlled wheelchair, *Input Device* is the joystick itself, *Shared Control* is a motor controller translating the joystick position into the wheelchair velocities, *Presentation Device* is a static image with arrows on the joystick panel. In a more sophisticated example of a BCI-controlled wheelchair, such as in [11, Chapter 6], *Presentation Device* modulates propositions to the users with visual, audio or tactile feedback, the user can then reply through a BCI which constitutes the *Input Device*, the *Shared Control* block maintains a POMDP model generates propositions and sends velocity commands to the wheelchair. We will now discuss our assumptions about each of the component from .

C. Modeling the plant

In general, a dynamic system can be modeled by:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}), \mathbf{x} \in \mathbb{X} \subset \mathbb{R}^n, \mathbf{u} \in \mathbb{U} \subset \mathbb{R}^m \qquad (1)$$

with *state vector* $\mathbf{x}$ and control signals $\mathbf{u}$.

Many concepts discussed in this paper are applicable to the general system (1). We will present an implementation of these concepts for a simple *Proof-of-Concept system* where a massless holonomic circular robot navigates in a known 2D environment. Mathematically: $\mathbf{x} = [x, y]$, $\mathbf{u} = [V_x, V_y]$, $f(\mathbf{u}) = \mathbf{u}$, $\|\mathbf{u}\| = \sqrt{V_x^2 + V_y^2} \leqslant V_{robot}$, and state space constraints are dictated by the environment map modeled as a 2D grid of free and blocked cells. By inflating obstacles we can assume a point-size robot.

To enable optimal pose selection, we assume that, an optimal control problem for the given system is solved. For system (1), the standard definition of the problem problem is formalized as follows: *Devise a control signal* $\mathbf{u}(t)$ *such that it moves the system from its initial state* $\mathbf{x}(t_0) = \mathbf{x}^0$ *to a desired state* $\mathbf{x}(t_1) = \mathbf{x^1}$ *and minimizes a functional:*

$$J\left[\mathbf{x}, \mathbf{u}, \mathbf{x}^0, \mathbf{x}^1\right] = \int_{t_0}^{t_1} F(\mathbf{x}, \mathbf{u}) dt \tag{2}$$

When $F \equiv 1$, it turns into a minimum time problem. A solution to the problem is an optimal control signal $\mathbf{u}^*(t) = \mathbf{u}^*(t, \mathbf{x}^0, \mathbf{x}^1)$ and an optimal trajectory $\mathbf{x}^*(t) = \mathbf{x}^*(t, \mathbf{x}^0, \mathbf{x}^1)$.

We can now construct a cost function that assigns an optimal value of the functional to any $(\mathbf{x}^0, \mathbf{x}^1)$:

$$C(\mathbf{x}^0, \mathbf{x}^1) = J\left[\mathbf{x}^*(t, \mathbf{x}^0, \mathbf{x}^1), \mathbf{u}^*(t, \mathbf{x}^0, \mathbf{x}^1), \mathbf{x}^0, \mathbf{x}^1\right] \tag{3}$$

Clearly, for most real systems, finding $C(\mathbf{x}^0, \mathbf{x}^1)$ is not practical, but approximations of the function can be utilized.

For the *Proof-of-Concept system*, we assume $F \equiv 1$. In this case, $C(\mathbf{x}^0, \mathbf{x}^1) \equiv d_{obst}(A, B)$, where $d_{obst}(A, B)$ is the *obstacle-aware distance*, the length of the shortest path between vertices A and B given the obstacles. A state-of-the-art algorithm for single-source any-angle path planning has been developed [30] to calculate $d_{obst}(A, B)$ in real time.

D. Modeling human intent

With a certain level of generality, we assume that 1) At every moment the user has an intended state (destination) $\mathbf{x}_d$ in mind 2) $\mathbf{x}_d$ changes slow, 3) $\mathbf{x}_d$ is unknown to the *Shared Control* (has to be inferred), 4) Given other constraints, human wants to reach the intended state as soon as possible.

For the *Proof-of-Concept system*, we also assume: 1) $\mathbf{x}_d$ is a vertex on the grid map, 2) Robot velocity at $\mathbf{x}_d$ is 0.

E. Modeling human-machine interface

HMIs can be classified by **temporal properties** [9] into *synchronous* (commands are selected at a constant frequency) and *asynchronous* (commands are initiated by the operator at any moment). Any asynchronous interface can be converted into synchronous by enforcing periodic inquiries to the user.

We assume a discrete LTI that accommodates r options (commands). By selecting a specific option, the user conveys their intent to the machine. For most BCIs, $r \leq 4$. The semantics of the commands is discussed in section III.

BCIs are naturally probabilistic due to measured signal noise. In general, the **accuracy** of an LTI can often be represented with an *interface matrix* that establishes a probabilistic relation between the detected (D) and intended commands (I). This conditional probability matrix $P(D|I)$ has the following structure:

		Detected		
		L_1	...	L_r
Intended	L_1	$P(D_1\|I_1)$	...	$P(D_r\|I_1)$
	...	...	...	...
	L_r	$P(D_1\|I_r)$	...	$P(D_r\|I_r)$

where L_1 ... L_r are permitted commands, $P(D_i|I_j)$ is the probability that command L_i is detected given that command L_j was intended. In this work, we assume: 1) HMI has a constant update period T_{HMI} 2) The alphabet size is r 3) The *interface matrix* is constant (obtained by repetitive experiments).

III. Synthesis

A. Intent estimate representation

Differently from other methods of shared position control, we represent the *intent estimate* with a probability *density* function $p(\mathbf{x})$ over the entire state space. Integrating over a region in the state space, in this case, yields the probability that the user's goal is located in that region. For the *Proof-of-Concept system*, the intent estimate becomes a probability *distribution* function (*PDF*) that assigns each vertex a probability value. The process of learning the *PDF* is discussed in Section III-C. At this point we can assume that *PDF* is known and updated every T_{HMI} s.

B. Intermediate destination state selection

To parallelize the inference and robot motion, *Shared Control* unit can utilize the information which comes with every update of the *intent PDF* by moving the system into an intermediate destination state. Clearly, the next intermediate state must belong to the T_{HMI}*-reachability area* (the set of states that the robot can reach within T_{HMI} s). For the *Proof-of-Concept system*, the *reachability area* (*RA*) is the set of vertices P for which $d_{obst}(C, P) \leqslant V_{robot} \cdot T_{HMI}$, where C is the current position. It is calculated using [30].

Now we can explore various approaches to choosing the intermediate state. In the simplest case (*no_move*-policy), the robot doesn't move until a cumulative probability within a small region exceeds a certain threshold ($\mathbf{x}_d$ inferred). This policy, however, results in the total control time $T_{control} = T_{inf} + T_{exec}$ (no parallelization). In this work, we propose the (*opt*-policy) that moves the robot to the state that will probabilistically minimize $C(\mathbf{x}_{cur}, \mathbf{x}_d)$. Consider function:

$$\widetilde{C}(\mathbf{x}) = \int \cdots \int_{\mathbb{X}} C(\mathbf{x}, \overline{\mathbf{x}}) p(\overline{\mathbf{x}}) d\overline{\mathbf{x}} \tag{4}$$

It measures the *probabilistic cost* of reaching the intended destination from a given state $\mathbf{x}$. Thus, with *opt*-policy, the intermediate state is:

$$\mathbf{x}_{opt} = \underset{\mathbf{x} \in RA}{\mathbf{argmin}} \left(C(\mathbf{x}_{cur}, \mathbf{x}) + \widetilde{C}(\mathbf{x})\right) \tag{5}$$

For the *Proof-of-Concept system*, the *probabilistic cost* of moving from A to the destination reduces to

$$\widetilde{C}(A) = \sum_{i=1}^{n} d_{obst}(A, B_i) p(B_i) \tag{6}$$

and the time-optimal intermediate vertex is

$$P_{opt} = \underset{A \in RA}{\mathbf{argmin}} \, \widetilde{C}(A) \tag{7}$$

To find an optimal robot pose, we need to solve (5), an optimization problem where a) calculation of $\widetilde{C}(.)$ is slow,

b) the gradient of $\widetilde{C}(.)$ cannot be easily calculated, c) the constraints are defined by the grid map. These properties render the direct use of gradient methods and brute-force search impractical.

For the *Proof-of-Concept system*, we find a *local* minimum by starting at a certain vertex P_{start} and iteratively moving to the the vertex in the 4-neighborhood with the lowest value of $\widetilde{C}(.)$ until we reach the minimum. The algorithm we developed in [30] was utilized for fast calculation of $\widetilde{C}(.)$.

It can be shown [31] that $\widetilde{C}(.)$ can have multiple minima with different values. This implies that the algorithm described above may not always find the global minimum. At this stage, however, we will consider the local minimum satisfactory.

One can observe [31] that when probability of a single vertex is close to 1, all of the following three vertices converge to the *intended destination vertex* P_d:

- $P_{maxprob}$ vertex with maximum probability in the *PDF*
- P_{cog} vertex closest to the *PDF* center of gravity (COG) (not always inaccessible);
- P_{near_cog} accessible vertex that is closest to P_{cog} in terms of euclidean distance;

Given that, in addition to *no_move*, we implemented the following intermediate pose selection policies:

- *cog2lopt*: finds local minimum in RA as described above, with $P_{start} = P_{near\_cog}$
- *maxprob_obst*: vertex in RA closest to $P_{maxprob}$ in terms of *obstacle-aware distance*;
- *nearcog_obst*: vertex in RA closest to P_{near_cog} in terms of *obstacle-aware distance*;
- *ra_maxprob*: vertex in RA with maximum probability

In section IV, we compare the performance of these policies.

C. Intent estimate update (Inference)

To learn the intent PDF, we divide the space into r regions every T_{HMI} s. Each region corresponds to an individual HMI command. The operator then selects the region their intended destination state belongs to.

We utilize Bayesian inference to update the *PDF*. Let

$$P(I_i^k) = \int_{\mathbb{X}_i^k} p^k(\mathbf{x}) d\mathbf{x} \tag{8}$$

be an a priori probability of the user choosing i-th region at k-th iteration, and let $P(I^k)$ be a vector of such probabilities. Here $\mathbb{X}_i^k$ is the i-th region at the k-th iteration. Now using an extended Bayes's rule, we can write the intent update rule:

$$P(I^{k+1}) = P(I^k|D^k) = \alpha P(D^k|I^k)P(I^k) \tag{9}$$

where $P(D^k|I^k)$ is the interface matrix. The initial value $p^0(\mathbf{x})$ can be calculated based on the prediction model or initialized with a uniform distribution.

The shape and size of $\mathbb{X}_i^k$ are up to the designer. To maximize the information transfer rate, the total a priori probabilities of the regions should be as close to the *optimal a priori vector* as possible. This vector is determined by the HMI matrix (for example, for a matrix with equal diagonal elements, the elements in the optimal a priori vector should be equal).

It can be shown [31], that even when this criterion is satisfied, various space divisions are possible yielding different performance results. The optimal space division method for a general system is yet to be identified. For the *Proof-of-Concept system*, we implemented several methods of map division and compared their performance in simulation.

D. Map segmentation

Map segmentation in case of a 2D grid is the process of assigning each accessible vertex an integer index between 1 and r. Each index corresponds to an HMI command and can be represented with a color. At every iteration of the HMI-cycle, the colored map divided is presented to the operator who then selects the HMI command that corresponds to the region where their intended destination is located.

As it was discussed in section III-C, the optimal way of dividing a 2D map is yet to be identified, certain requirements, however, are clear:

1) The total a priori probabilities of the regions should be as close to the *optimal a priori vector* as possible.
2) If human commands are detected correctly, the map segmentation process shall guarantee that a probability of a single vertex converges to 1.0

We developed a simple one-pass map segmentation algorithm [31]. By iterating through accessible grid map vertices in any given order, the algorithm can generate a segmented map that satisfies the above requirements. The order of iteration defines the geometric shapes of the regions. Fig. 2 shows four implemented policies: *htile* (left-to-right, top-to-bottom), *vtile* (top-to-bottom, left-to-right), *equidist* (first sorted by distance from a given center vertex C, then by angle to C), *extremal* (first sorted angle to a given center vertex C, then by distance from C), *equidist* and *extremal* are based on the algorithm developed in [30].

Additionally, we implemented two alternating map division methods: *altertile* (alternates *vtile* and *htile*), and *extredist* (alternates *extremal* and *equidist*). In the next section, we compare how different map segmentation policies affect navigation performance.

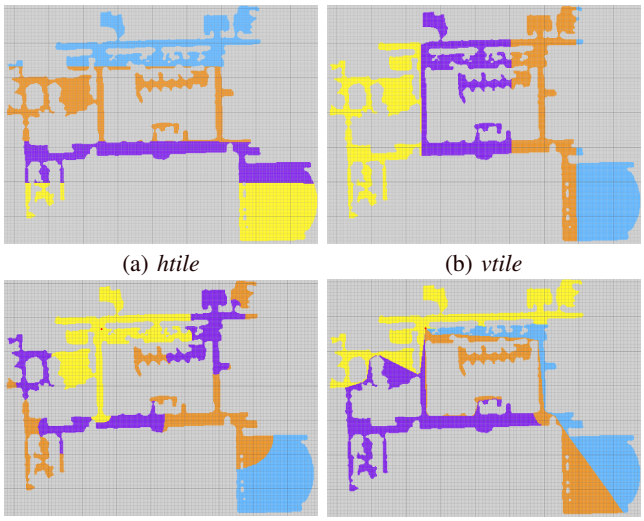
IV. Simulation Experiments

To compare various methods of pose selection and map segmentation developed for the *Proof-of-Concept system*, we designed an automated test framework. For seamless integration with the real robot system, the framework was implemented in Robotic Operating System (ROS).

A. Automated test setup

The system consists of the following ROS nodes (Fig. 3):

- *best_pose_finder*, *inference_unit*, *map_divider* were discussed in Sections III-B, III-C, III-D, respectively.
- *experimentator* publishes the navigation map and randomly pre-generated start and goal poses.
- *human_model* accepts the map divided into four colored regions, finds what color of the goal vertex is and



(a) *htile* (b) *vtile*

(c) *equidist* (d) *extremal*

Fig. 2: Examples of map segmentation policies: each of the 4 colors corresponds to an individual HMI command

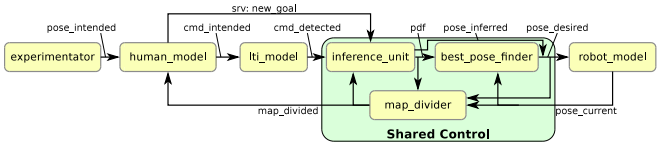


Fig. 3: System for automated testing of simulated robot navigation via LTI

publishes a corresponding *intended command*. In this simulation, the region is always picked correctly.

- *lti_model* simulates an LTI by randomizing *intended commands* based on the configured interface matrix. We tested only symmetric matrices with equal diagonal elements and evenly distributed error elements. *lti_model* also introduced a T_{HMI}=1 s interface delay.
- *robot_model* is a simulation of a holonomic massless point-size robot. Whenever it receives a message with the new desired position, it starts moving towards it with constant speed V_{robot}=3 m s^{-1} along the shortest path, publishing its current pose at the rate of 100 Hz. The high value of V_{robot} was chosen to speed up the simulation. It is compensated by a relatively short T_{HMI}.

The intent PDF was initialized with a uniform distribution. A navigation experiment starts at the moment when the goal is published. When the probability of a single vertex reaches 99%, the goal vertex is considered inferred. When the robot reaches the inferred goal, the experiment ends. We used the same map as shown in Fig. 2 (resolution: 500 × 370, cell size 10 cm).

B. Simulation experiment results

Four parameters have been varied between the experiments: interface matrix, route (three random routes were defined), best pose selection policy, map segmentation policy.

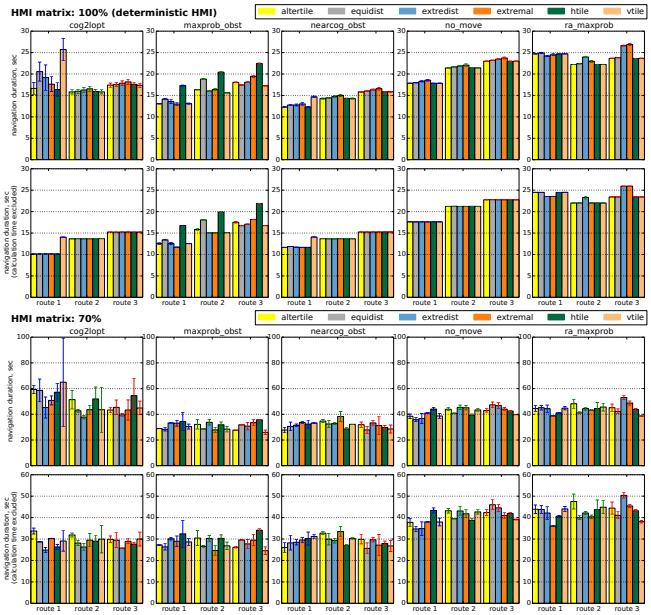


Fig. 4: Results of the automated simulation tests

For each combination, 16 runs were executed. The experiments were run on ThinkPad W520 with Intel(R) Core(TM) i7-2760QM CPU @ 2.40GHz CPU and took several days to complete. All experiments were recorded. Fig. 4 shows results of automated testing.

Despite our attempts to speed up *cog2lopt*, the calculation time still noticeably affected the performance, that is why we show both the actual navigation duration and the navigation duration with calculation time excluded.

Expectedly, for both HMIs, *no_move* and *ra_maxprob* resulted in the worse navigation time compared to other pose selection methods. For the deterministic HMI, *cog2lopt* expectedly resulted in the shortest navigation time (calculation excluded), even though the 20% difference from *nearcog_obst* is only seen on route 1. For the 70% HMI matrix, however, there is no visible difference in the performance. Surprisingly, the map segmentation policy does not seem to affect the navigation time in these experiments. With only three routes tested, however, these observations should not be deemed as ultimate conculsions.

V. Robot Experiments

In this section, we discuss an integration of the *Shared Control* developed for the *Proof-of-Concept system* with a robotic wheelchair and the results of navigation experiments. The purpose of the experiments reported here was not to obtain statistically significant results, but to demonstrate the viability of the proposed solution in various modes of operation, particularly to validate the advantages of the NoVeLTI approach listed in Section I. They involved only one human subject, the designer of the system.

A. Experimental system setup

The semi-autonomous robot (Fig. 7a) was built on top of a commercially available electric wheelchair CTM HS-2800. It

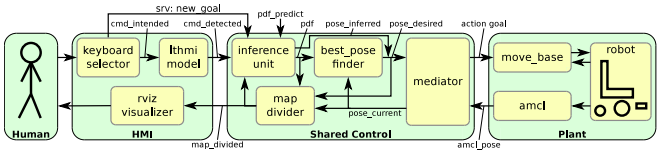


Fig. 5: Test framework architecture

is equipped with two Lidars, wheel-on-wheel encoder modules, and infrared sensors. Using Robotic Operating System (ROS) packages, the robot is capable of SLAM, localization (AMCL) and autonomous navigation (ROS navigation stack).

The wheelchair is a differentially-driven platform with non-zero inertia, and, thus, the *Shared Control* developed in section III will not be optimal. Nevertheless, the robot experiments demonstrated a satisfactory performance. Circular robot footprint was chosen for two reasons: a) for easier integration with *Shared Control* b) *move_base* is more reliable with circular robots.

The overall navigation system consists of the following ROS-nodes (Fig. 5): *inference_unit*, *best_pose_finder*, *map_divider*, and *lti_model* are the same as in section IV. *amcl* is a ROS localization package that constantly publishes robot pose. *move_base* is the standard ROS-package that implements autonomous navigation. To prevent unnecessary rotations, the goal orientation tolerance was set to 2π. *mediator* ensures smooth goal preemption and eliminates issues caused by localization errors, *rviz visualizer* displays two maps to the user (Fig. 6): Intent Estimate PDF with a goal marker and a colored Segmented Map. The *operator* identifies the color of the map region their goal marker belonged to, and presses a corresponding key which is translated by the *keyboard selector* into *intended command*. The goal marker wasn't displayed on the segmented map intentionally, since in real life it doesn't exist. The user had to refer to the obstacle configuration instead. As it is discussed below, that became a major challenge for the operator.

A custom environment ($27\,\mathrm{m} \times 17\,\mathrm{m}$) has been constructed for navigation tests (Fig. 6). The following parameters have been constant in all experiments: shared control grid map cell size: $10\,\mathrm{cm}$, robot maximum linear and angular velocities: $0.5\,\mathrm{m\,s^{-1}}$ and $0.6\,\mathrm{rad\,s^{-1}}$, T_{HMI}=3 s, intermediate pose selection policy: *nearcog_obst* (because it's fast to calculate and yields almost the same performance as *cog2lopt*, map segmentation policy: *nearcog_extremal* (*extremal*-policy with the center at P_{near_cog}). To prevent unwanted stops at every intermediate vertex, V_{robot} was configured to be 30% higher than the real robot velocity.

B. Robot experiment results

More than 250 point-to-point navigation experiments were recorded with various modes of operation [31]. Here we present some of them (refer to Fig. 7 and video attachment):

a) ***94%- vs 70%-HMI matrix****:* demonstrates the effect of the accuracy of HMI. Here we can see that for longer routes (door1→livroom1, office1→bathroom1) (except 1 try), the inference via 94%-HMI finishes before the

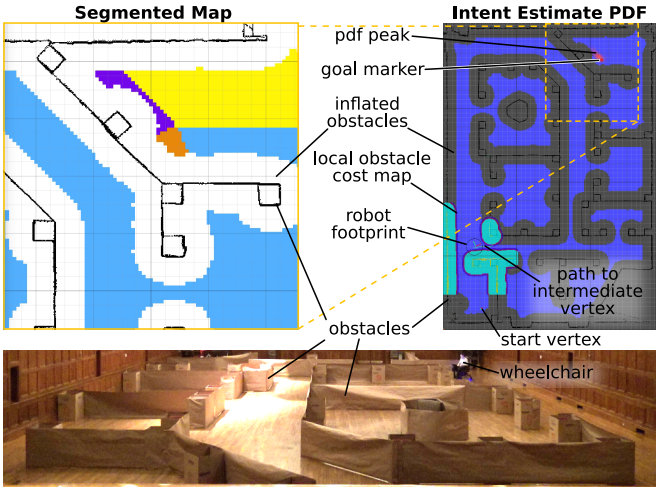


Fig. 6: Robot experiment snapshot: map with *Intent PDF*, robot navigation data, and goal marker (*top right*), zoomed map divided into colored regions as displayed to the operator (goal marker is intentionally not displayed) (*top left*), video camera snapshot (*bottom*).

robot arrives to the destination ($T_{exec} > T_{inf}$). On the other hand, the inference via 70% appears to be 2-4 times slower, and results in $T_{exec} < T_{inf}$. We can also observe from the distance-to-goal plots, that in the 70%-HMI case, when the robot almost reaches the goal, it keeps moving around it for $20\,\mathrm{s}$ to $100\,\mathrm{s}$. Nevertheless, for longer routes and especially for 94%-HMI, the parallelization of the inference and motion makes the robot paths almost indistinguishable from the shortest paths. On a shorter route (music1→bedroom1), however, the parallelization can result in an unnecessary movement of the robot, even more so for the 70%-HMI.

b) ***Effect of POIs****:* here, for 95%-HMI matrix, we compare two cases that differ by the initial value of intent PDF: 1) PDF is initialized with a uniform distribution over accessible vertices 2) PDF is initialized with a sum of gaussians (σ's vary from 0.8 to 1.3, centers are at the points of interest (POIs) shown on the map). The second case demonstrates how prior information about the user preferred locations can speed up the inference process. Indeed, as we can see on the PDF entropy plots, the T_{inf} halved. Interestingly, the total navigation time on routes door1→livroom1, office1→bathroom1 is longer in the POI-scenario. The explanation is that the concentration of POIs in the top right corner of the map was higher, and thus, in the beginning, the robot rotated towards that corner and only after a couple of detected decision the navigation system would move into the right direction. From the operator's perspective, navigation with predefined POIs felt much easier.

c) ***Effect of PDF smoothening****:* Distinguishing individual vertices and identifying the color of the intended destination vertex on a full-size map turned out to be a major challenge for the operator. We attempted to mitigate this problem in 2 steps: 1) we zoomed the segmented map view to display only those vertices whose probability was higher than

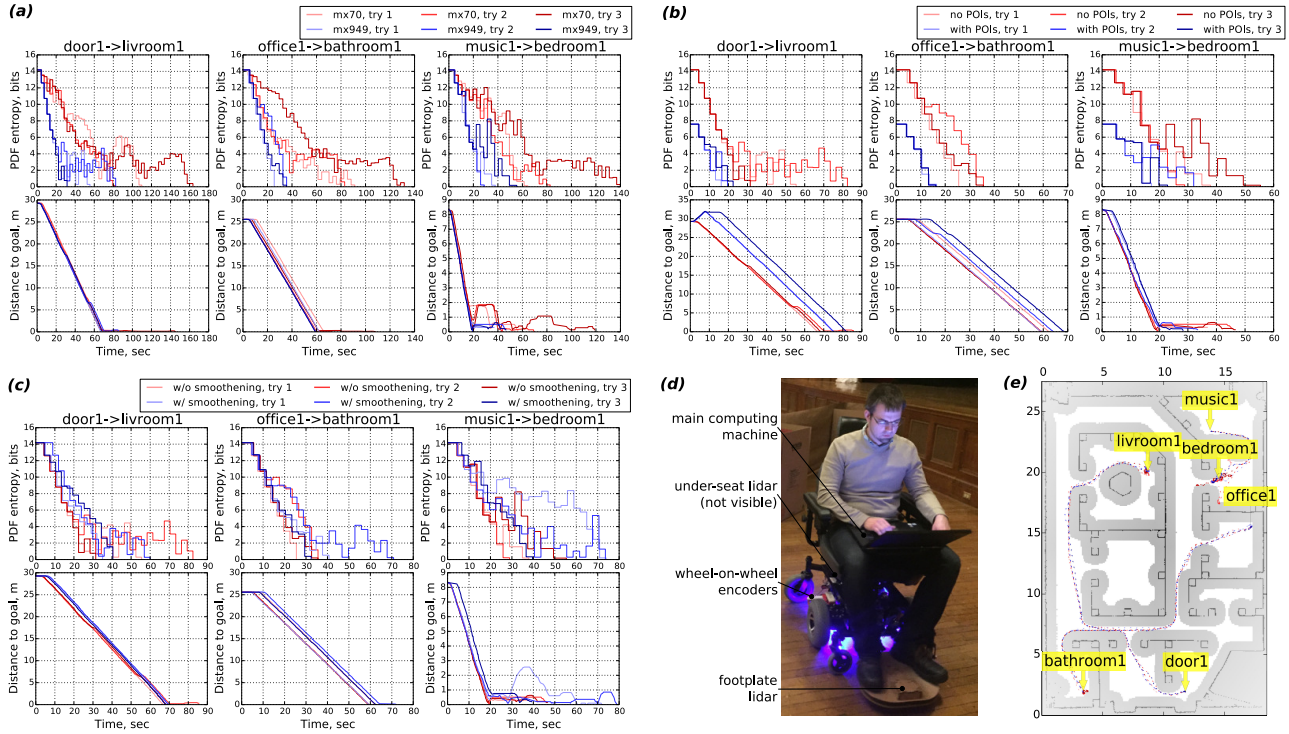


Fig. 7: Robot navigation experimental results and effects of various parameter variations: (a) 94%- vs 70%-HMI matrix; (b) effect of POIs; (c) effect of PDF smoothening (d) robotic wheelchair used for exepriments; (e) navigation map;

a threshold 2) after every PDF update, the probability value of every vertex in the vicinity of any region boundary was set to the average of its r-neighborhood, where r depended on the zoom factor. The higher zoom factor, the smaller r.

The latter removed stiff slopes that used to appear in the PDF at the first steps of inference process, and subjectively made the first choices easier for the operator. The plots show that in most runs it increased the inference time, but improved the accuracy of the goal inference. Note how the reddish curves in the distance-to-goal plot do not reach the 0 at the end. This is a sign that the inferred goal is $10\,\mathrm{cm}$ to $20\,\mathrm{cm}$ off the intended goal.

VI. Discussion and Future Work

As demonstrated by design and robot experiments, the proposed navigation framework NoVeLTI, regardless of what pose selection and map segmentation policy is used, has a set of features, none of the existing shared control methods has. First, it doesn't restrict user navigation to a small set of predefined destinations. It gives the operator the freedom to go to any vertex on the map. Second, it naturally integrates the a priori knowledge of human intent (predicted PDF). The latter is not limited to a set of individual destination, permits arbitrary shapes of areas of interest without performance degradation. It implements probabilistic reasoning and maximizes information transfer rate, can be adapted for specific users by reconfiguring interface matrix, supports any number of commands. An important direction for future work is to automatically calculate the predicted PDF. It can be a static PDF as shown in section V, or, more interestingly, a dynamic 2D function, calculated based on time of the day and external events. In this case, the *inference_unit* will be able to suggest the user when to start moving. The experiments showed that a small distance between start and end points, results in unnecessary robot motion. This, however is not a limitation of the presented framework itself. An intermediate pose selection policy can be modified such that robot starts moving only when it is probabilistically certain that the destination is far. Such policy, however, will not be time-optimal.

In the robot experiments, accurately identifying the colored region with the intended destination was not always easy and took time. That was the main reason to increase the T_{HMI} to $3\,\mathrm{s}$. When an intended destination is far away, it is hard to locate it with $10cm$ accuracy on a map. Such accuracy becomes reasonable only in the immediate vicinity of the robot. Zooming and smoothening parameters also affect the performance. Overall, the subjective perception of navigation experience by the operator should not underestimated. Sometimes, selecting the destination felt like fighting and other times it felt rather natural. The method of visualization may play an important role here. An analysis of subjective perception of various navigation configurations by different users is another direction for future work.

Other directions include: speeding up *opt*-policy calculations by caching and multithreading, simulation performance tests on a larger set of routes, development of a *Shared Control* for a nonholonomic robot, integration with a BCI.

It shall be noted that a performance comparison between various shared control methods is a very challenging problem

Given the number of variable parameters and unpredictable elements such as humans most of the works on shared control compare their method to other methods qualitatively, but not quantitatively.

VII. Conclusion

In this paper, we presented an analysis of robotic systems with low throughput human-machine interfaces from the controls perspective. We developed a novel method for navigation via an LTI that has a unique set of features compared to the existing solutions. A detailed analysis and the synthesis of our method is presented. The method was used to design a *Shared Control* unit to navigate a massless holonomic robot in a known environment. An automated test setup was developed to simulate the navigation process and observe the influence of configuration parameters. The *Shared Control* was adapted for a real robotic wheelchair. Robot experiments demonstrated the viability of the proposed solution in various modes of operation.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. 1135854.

References

[1] C. G. Pinheiro, E. L. Naves, P. Pino, E. Losson, A. O. Andrade, and G. Bourhis, "Alternative communication systems for people with severe motor disabilities: A survey," *BioMedical Engineering OnLine*, vol. 10, p. 31, 2011.

[2] I. Mougharbel, R. El-Hajj, H. Ghamlouch, and E. Monacelli, *Comparative study on different adaptation approaches concerning a sip and puff controller for a powered wheelchair*, Oct 2013, p. 597603.

[3] D. Purwanto, R. Mardiyanto, and K. Arai, "Electric wheelchair control with gaze direction and eye blinking," *Artificial Life and Robotics*, vol. 14, no. 3, p. 397400, Dec 2009.

[4] J. Kim, H. Park, J. Bruce, E. Sutton, D. Rowles, D. Pucci, J. Holbrook, J. Minocha, B. Nardone, D. West, and et al., "The tongue enables computer and wheelchair control for people with spinal cord injury," *Science Translational Medicine*, vol. 5, no. 213, p. 213ra166213ra166, Nov 2013, pMID: 24285485.

[5] D. A. Sinyukov, R. Li, N. W. Otero, R. Gao, and T. Padir, "Augmenting a voice and facial expression control of a robotic wheelchair with assistive navigation," in *2014 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, Oct 2014, p. 10881094.

[6] M. R. Walter, S. Hemachandra, B. Homberg, S. Tellex, and S. Teller, "A framework for learning semantic maps from grounded natural language descriptions," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1167–1190, 2014.

[7] J. R. Wolpaw, D. J. McFarland, G. W. Neat, and C. A. Forneris, "An eeg-based brain-computer interface for cursor control," *Electroencephalography and Clinical Neurophysiology*, vol. 78, no. 3, p. 252259, Mar 1991, 00762.

[8] N. Birbaumer, N. Ghanayim, T. Hinterberger, I. Iversen, B. Kotchoubey, A. Kbler, J. Perelmouter, E. Taub, and H. Flor, "A spelling device for the paralysed," *Nature*, vol. 398, no. 6725, p. 297298, Mar 1999.

[9] L. Bi, X.-a. Fan, and Y. Liu, "Eeg-based brain-controlled mobile robots: A survey," *IEEE Transactions on Human-Machine Systems*, vol. 43, no. 2, p. 161176, Mar 2013.

[10] P. F. Diez, S. M. Torres Mller, V. A. Mut, E. Laciar, E. Avila, T. F. Bastos-Filho, and M. Sarcinelli-Filho, "Commanding a robotic wheelchair with a high-frequency steady-state visual evoked potential based braincomputer interface," *Medical Engineering and Physics*, vol. 35, no. 8, p. 11551164, Aug 2013, 00016.

[11] X. Perrin, "Semi-autonomous navigation of an assistive robot using low throughput interfaces," Ph.D. dissertation, 2009. [Online]. Available: http://dx.doi.org/10.3929/ethz-a-006014036

[12] T. B. Sheridan, *Telerobotics, Automation, and Human Supervisory Control*. MIT Press, Jan 1992.

[13] S. Musić and S. Hirche, "Control sharing in human-robot team interaction," *Annual Reviews in Control*, vol. 44, pp. 342–354, Jan. 2017.

[14] A. S. Royer and B. He, "Goal selection versus process control in a brain–computer interface based on sensorimotor rhythms," *Journal of Neural Engineering*, vol. 6, no. 1, p. 016005, 2009.

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

[16] T. Carlson and J. d. R. Millan, "Brain-Controlled Wheelchairs: A Robotic Architecture," *IEEE Robotics Automation Magazine*, vol. 20, no. 1, pp. 65–73, Mar. 2013.

[17] Y. T. Lin and C. H. Kuo, "Development of SSVEP-based intelligent wheelchair brain computer interface assisted by reactive obstacle avoidance," in *2016 IEEE International Conference on Industrial Technology (ICIT)*, Mar. 2016, pp. 1572–1577.

[18] X. Perrin, R. Chavarriaga, F. Colas, R. Siegwart, and J. d. R. Millán, "Brain-coupled interaction for semi-autonomous navigation of an assistive robot," *Robotics and Autonomous Systems*, vol. 58, no. 12, pp. 1246–1255, Dec. 2010.

[19] T. Shi, H. Wang, and C. Zhang, "Brain Computer Interface system based on indoor semi-autonomous navigation and motor imagery for Unmanned Aerial Vehicle control," *Expert Systems with Applications*, vol. 42, no. 9, pp. 4196–4206, Jun. 2015.

[20] H. Nezamfar, D. Sinyukov, U. Orhan, D. Erdogmus, and T. Padir, "Brain Interface to Control a Tele-Operated Robot," 2013.

[21] R. Zhang, Y. Li, Y. Yan, H. Zhang, S. Wu, T. Yu, and Z. Gu, "Control of a Wheelchair in an Indoor Environment Based on a Brain-Computer Interface and Automated Navigation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 24, no. 1, pp. 128–139, Jan. 2016.

[22] B. Rebsamen, C. Guan, H. Zhang, C. Wang, C. Teo, M. H. Ang, and E. Burdet, "A Brain Controlled Wheelchair to Navigate in Familiar Environments," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, no. 6, pp. 590–598, Dec. 2010.

[23] X. a Fan, L. Bi, T. Teng, H. Ding, and Y. Liu, "A Brain-Computer Interface-Based Vehicle Destination Selection System Using P300 and SSVEP Signals," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 1, pp. 274–283, Feb. 2015.

[24] X. Perrin, F. Colas, C. Pradalier, and R. Siegwart, "Learning to Identify Users and Predict Their Destination in a Robotic Guidance Application," in *Field and Service Robotics*, ser. Springer Tracts in Advanced Robotics, A. Howard, K. Iagnemma, and A. Kelly, Eds. Springer Berlin Heidelberg, 2010, no. 62, pp. 377–387.

[25] E. Demeester, A. Hüntemann, D. Vanhooydonck, G. Vanacker, H. Van Brussel, and M. 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, Dec. 2007.

[26] F. Doshi and N. Roy, "Spoken language interaction with model uncertainty: An adaptive human–robot interaction system," *Connection Science*, vol. 20, no. 4, pp. 299–318, Dec. 2008.

[27] I. Iturrate, J. Antelis, A. Kubler, and J. Minguez, "A Noninvasive Brain-Actuated Wheelchair Based on a P300 Neurophysiological Protocol and Automated Navigation," *IEEE Transactions on Robotics*, vol. 25, no. 3, pp. 614–627, Jun. 2009.

[28] Z. Wei, W. Chen, J. Wang, H. Wang, and K. Li, "Semantic Mapping for Safe and Comfortable Navigation of a Brain-Controlled Wheelchair," in *Intelligent Robotics and Applications*. Springer, Berlin, Heidelberg, Sep. 2013, pp. 307–317.

[29] A. Erdogan and B. D. Argall, "The effect of robotic wheelchair control paradigm and interface on user performance, effort and preference: An experimental assessment," *Robotics and Autonomous Systems*, vol. 94, pp. 282–297, Aug. 2017.

[30] D. A. Sinyukov and T. Padır, "CWave: High-performance single-source any-angle path planning on a grid," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[31] D. A. Sinyukov, "Semi-autonomous robotic wheelchair controlled with low throughput human-machine interfaces," Ph.D. dissertation, Worcester Polytechnic Institute, Apr. 2017.