Haptic-Guided Path Generation for Teleoperation of Car-Like Vehicles

Michael Fennel*, Antonio Zea*, Uwe D. Hanebeck*

***Abstract*— Despite significant advances in robot autonomy, manual intervention by a human operator is necessary in many situations. This usually requires qualified staff and some robot-specific input device even for the comparatively simple case of platform locomotion. For this reason, we propose a novel path generation method for the teleoperation of car-like vehicles. With this method, the operator "draws" a desired 2D path by walking in a large-scale haptic interface while a guiding force is exerted, which ensures that the generated path can be accurately followed by a path tracking controller running offline on a robot. We present a local optimization-based path planner, a higher-level path generation algorithm utilizing the aforementioned planner, and a force feedback law. Experiments show an improved feasibility of the generated paths without affecting the operator's ability to make decisions independently.**

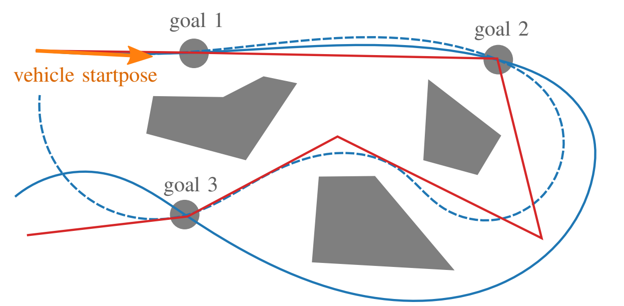


Fig. 1. Example scenario with obstacles. The execution of the red path might lead to collision with obstacles. The blue paths are feasible and can be followed precisely.

I. Introduction

One of the aims of the ROBDEKON [1] research project is the automatic decontamination in inhospitable environments by autonomous construction machines leveraging recent developments in robotics. Despite the application of state-of-the-art technology, most of the developed autonomy functions are still suffering from certain limitations regarding reliability, generalization under varying environmental conditions, and dexterity. For this reason, a logical step in the evolution of the involved robots is the application of teleoperation principles to delegate difficult or unhandled situations to a human operator that can then operate the robot's drive-train or manipulator manually.

Traditionally, a realistic copy of the driver's cabin is created for this purpose. The disadvantage of this method is that the needed replica is costly and tailored to a single machine type. Furthermore, this solution requires a permanent data link to the robot and highly trained personal as the robot is controlled on its lowest level of abstraction. To overcome these issues, the methods of shared autonomy [2] can be applied. As a result, only sub-tasks that cannot be handled by the robot are delegated to the operator.

In this paper, we present such a shared autonomy approach for an intuitive and precise way to move a car-like mobile platform across a challenging planar construction site, as illustrated in Fig. 1. We assume that the environment is unstructured and that the robot is not able to obtain a map that is suitable for path planning. In this case, the overall task is split into the generation of a path performed by the human operator, and the follow-up control of the path executed by the robot. In contrast to a holonomic platform, where a valid path can be generated easily by selecting an arbitrary polyline (red line in Fig. 1), the constrained kinematics of the car-like platform must be taken into account. To achieve this, we propose a path generation based on haptic force feedback, in which the user's hand position resembles the platform position. As long as the user does not violate the kinematic constraints of the platform and hence does not generate an infeasible path, he or she can move around freely. However, if these constraints are violated, a force guiding the user back to a feasible path is applied. Since the resulting path (blue lines in Fig. 1) is feasible, the actual execution of the path can then be performed offline and with high accuracy.

The remainder of this paper is structured as follows: First, we give a brief overview of related work in section II and define the precise problem in section III. In section IV, we introduce the proposed path generation algorithm and associated components. Finally, we present our evaluation in section V and conclude our work in section VI.

II. Related Work

The field of path planning and generation under non-holonomic constraints is a well-researched topic. Common approaches like rapidly exploring random trees or lattice-based path planning are sampling-based [3], [4]. By default, these algorithms need a map of the environment and a pre-defined goal whilst they are unable to interact with the user during the planning phase. Taïx [5] mitigates the latter issue by taking the position of the human operator as a clue during exploration, but the need for a map remains.

On the other hand, existing haptic guidance systems usually assume that the path is already fully known and shall be followed by a human [6], [7]. This can be generalized to surfaces by applying the concept of virtual fixtures [8].

This work was supported by the ROBDEKON project of the German Federal Ministry of Education and Research.

*All authors are with the Intelligent Sensor-Actuator-Systems Laboratory (ISAS), Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology (KIT), Germany. E-mails: michael.fennel@kit.edu, antonio.zea@kit.edu, uwe.hanebeck@kit.edu.

One approach that combines haptic guidance and path planning is made by Ladeveze [9]. The author suggests a system for the teaching of industrial robots that allows the incorporation of the user's cognitive capabilities. However, the system still requires precise information about the environment and non-holonomic movements are not supported. In the work of Kuiper [10], a haptic guidance algorithm for a steering task of a non-holonomic vehicle is presented. The proposed system can work without predefined trajectories by predicting future vehicle poses, but the loop between the platform and the user must be permanently closed and the user input is expected in the less intuitive configuration space. Another approach, that combines haptic-guidance, user interaction, and non-holonomic constraints, is contributed by Rahal [11]. Similar to the previous approach, a closed loop is required. Further drawbacks are the need for rotational haptic feedback and constraints that do not resemble physical kinematics.

To the best of our knowledge, no approach is known yet that combines offline path generation with haptic feedback under kinematic constraints.

III. Problem Statement

Given: According to [3], the kinematics of a forward driving car-like vehicle can be stated as non-linear, time-continuous dynamic system of the form

$$\dot{\underline{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \underline{f}\left(\begin{bmatrix} x \\ y \\ \theta \end{bmatrix}, \begin{bmatrix} v \\ \phi \end{bmatrix} \right) = \begin{bmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \frac{v}{L}\tan(\phi) \end{bmatrix}, \quad (1)$$

where the system state $\underline{x}$ consists of the center position of the rear axle $[x, y]^\mathsf{T}$ and the heading angle θ. The control inputs $\underline{u} = [v, \phi]^\mathsf{T}$ are defined as longitudinal velocity v and steering angle ϕ. L denotes the wheelbase. To keep the vehicle within its feasible operating conditions, the constraints

$$0 \leq v \leq v_{\max}\,, \quad (2)$$

$$-\phi_{\max} \leq \phi \leq \phi_{\max} \quad (3)$$

limit the steering angle and the velocity.

Goal: We aim to capture a path describing the movement of the center position of the vehicle's rear axle in a plane that is approximated by a series of points $\left[\underline{p}_0, ..., \underline{p}_N\right]$, $\underline{p}_i = [x_i, y_i]^\mathsf{T} \in \mathbb{R}^2$. Note that no time information is given since the speed of execution will be determined by the robot. The resulting path shall be able to satisfy the kinematic constraints (1)–(3).

This is achieved by recording the position of the operator's hand in the room-sized ISAS semi-mobile haptic interface (SMHI) [12] and interpreting it as the vehicle position. Simultaneously, an appropriate cartesian force $\underline{F} \in \mathbb{R}^2$ is displayed as guidance to the operator that keeps the input path feasible. For example, the operator will feel a counter-acting force when proceeding on a path that is incompatible with the vehicle kinematics. Due to the nature of the SMHI, torques as well as non-planar forces cannot be displayed and therefore not be used for guiding the operator.

The generated path is then sent to the robot and executed offline by the embedded controller. This implies that the connection between the control station and the robot is not mandatory after the path generation is completed.

IV. Proposed Method

The proposed method for the path generation with haptic guidance consists of three linked modules:

A) A *local path planner* formulated as an optimization problem predicts the vehicle path depending on the user position relative to a vehicle state.
B) A higher-level *path generation algorithm* utilizes the local path planner for creating arbitrarily long paths.
C) The planned path and the user position are used for the generation of guiding *force feedback*.

A. Local Path Planner

Without loss of generality, assume that the vehicle is located at

$$\begin{bmatrix} x(0) & y(0) & \theta(0) \end{bmatrix}^\mathsf{T} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^\mathsf{T} \quad (4)$$

as a starting point. The desired position of the vehicle is supposed to be located at $[x^*, y^*]^\mathsf{T}$ somewhere in the vicinity of that starting point. A goal orientation θ^* is not given.

The objective is then to find a direct state and input trajectory, whose final state at time t^* minimizes the error to the desired position $[x^*, y^*]^\mathsf{T}$ while satisfying the kinematic constraints. This can be formulated as a non-linear, constrained optimization problem:

$$\begin{aligned} &\underset{\underline{x}(t), \underline{u}(t), t^*,}{\operatorname{argmin}} \left\| \begin{bmatrix} x(t^*) \\ y(t^*) \end{bmatrix} - \begin{bmatrix} x^* \\ y^* \end{bmatrix} \right\|^2 \\ &\text{subject to} \quad \text{equations (1)–(3)}\,, \\ &\qquad\qquad\quad \underline{x}(0) = \underline{0}\,, \\ &\qquad\qquad\quad |\theta(t)| < \frac{\pi}{2}\,. \end{aligned} \quad (5)$$

The last constraint ensures that no approaches from the reverse direction are possible, because this would disregard the locality of the planner.

The above problem can yield multiple solutions. One option is to select the solution with the shortest path. This is equivalent to selecting the solution with the smallest t^* due to constraint (2). An alternative way to remove the superfluous solutions is the approximation of the original problem under the following assumptions:

- The vehicle is driving with constant velocity $v = v_0$. This is reasonable since the set of feasible paths for an ideal car-like vehicle does not depend on the velocity.
- Additionally, the steering angle is held constant within the optimization horizon, which guarantees that only one path exists for any given local endpoint. The resulting approximation error is small due to the locality, i.e., short time horizon, of the problem.

According to (1), the movement of a car-like vehicle under these assumptions can be described using circular arcs.

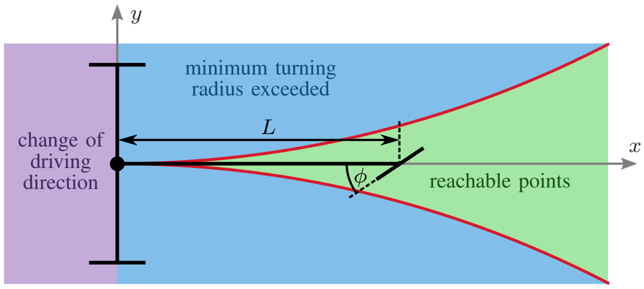


Fig. 2. Partitioning of the set of possible desired positions relative to the vehicle, whose position is defined by the black dot at the center of the rear axle. The red circular arcs indicate the paths with the smallest possible turning radius.

As stated in [3], the relation between turning radius r and steering angle ϕ is then given by

$$r = \frac{L}{\tan \phi}\,, \; \phi \neq 0\,. \tag{6}$$

Now it is possible to derive a solution for the approximation of problem (5) analytically by partitioning the set of feasible endpoints as depicted in Fig. 2.

1) Goal is reachable: If the goal is located within the set of reachable points represented by the green area in Fig. 2, it can be reached exactly by driving along an arc with the specific radius r. Mathematically, the optimization problem then reduces to the equation

$$x^{*2} + \left(r - y^*\right)^2 = r^2,\; y^* \neq 0\,. \tag{7}$$

Consequently, the solution for the desired radius is calculated as

$$r = \frac{x^{*2} + y^{*2}}{2y^*}\,,\; y^* \neq 0\,. \tag{8}$$

Using (6), this can be rearranged to the required steering angle

$$\phi = \arctan \frac{2Ly^*}{x^{*2} + y^{*2}}\,. \tag{9}$$

Note, that this equation also yields correct results for $y^* = 0$ as long as $x^* \neq 0$. The new vehicle heading at the final position follows from trigonometry:

$$\theta\left(t^*\right) = \arcsin \frac{x^*}{r} = \arcsin \frac{2x^*y^*}{x^{*2} + y^{*2}}\,. \tag{10}$$

With this information, the full state trajectory and the final time t^* can be determined by integrating (1).

2) Goal exceeds minimum turning radius: If the goal is located in one of the blue areas of Fig. 2, it cannot be reached exactly since the minimum turning radius

$$r_{\text{min}} = \frac{L}{\tan \phi_{\text{max}}} \tag{11}$$

with the corresponding steering angle ϕ_{max} is exceeded. For now, assume that $y^* > 0$. If $y^* < 0$, the assumption can be restored by reflecting the problem on the x-axis. The curved border of the upper blue area, that describes feasible vehicle end positions, can be parameterized as

$$x = r_{\text{min}} \sin \alpha\,, \tag{12}$$

$$y = r_{\text{min}}(1 - \cos \alpha)\,. \tag{13}$$

Thus, minimizing the objective in problem (5) yields the new objective function

$$J = \left(r_{\text{min}} \sin \alpha - x^*\right)^2 + \left(r_{\text{min}}\left(1 - \cos \alpha\right) - y^*\right)^2\,. \tag{14}$$

By setting $\frac{\partial J}{\partial \alpha}$ to zero, the solution

$$\alpha^* = \arctan \frac{x^*}{r_{\text{min}} - y^*} \tag{15}$$

is obtained. This determines the optimal vehicle position $\left[x\left(t^*\right), y\left(t^*\right)\right]^\mathsf{T}$ after insertion into (12) and (13). The optimal heading is given by $\theta\left(t^*\right) = \alpha^*$. The full state trajectory is calculated analogous to the previous case.

3) Other regions: A desired endpoint in the purple region of Fig. 2 is not reachable with the vehicle described by equations (1)–(3), as reversing would be required. Endpoints that meet

$$\left(\left|y^*\right| \geq r_{\text{min}} \wedge x^* < r_{\text{min}}\right) \vee \left(\left|y^*\right| > x^* \wedge x^* \geq r_{\text{min}}\right) \tag{16}$$

are also not reachable due to the last constraint of optimization problem (5).

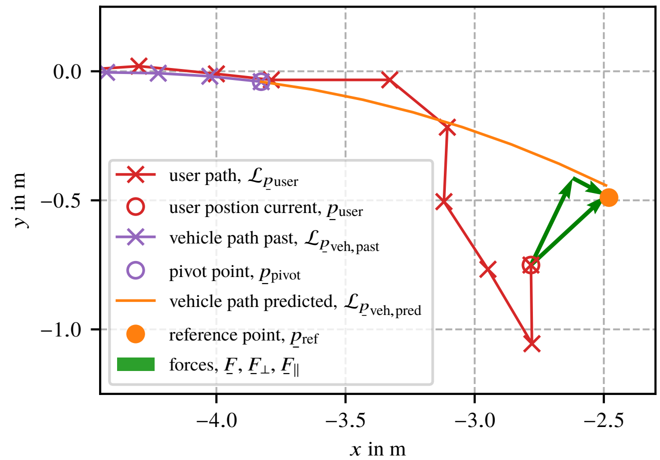


Fig. 3. Illustration of the proposed path generation algorithm.

B. Path Generation Algorithm

The presented path planning algorithm only provides a local prediction of the vehicle path. To overcome this limitation and to generate arbitrarily long paths, algorithm 1 was designed. A new iteration is carried out whenever a new measurement of the user position $\underline{p}_{\text{user}} = \left[x_{\text{user}}, y_{\text{user}}\right]^\mathsf{T}$ is available. Based on Fig. 3, the key concepts of this algorithm are briefly described in the following.

The pivot element $\left(\underline{p}_{\text{pivot}}, \theta_{\text{pivot}}\right)$ (purple circle) describes the pose that is used as the starting point for the local planning algorithm in section IV-A. To meet assumption (4), the current user position $\underline{p}_{\text{user}}$ (red circle) is transformed before the actual optimization problem is solved

Algorithm 1 Path generation under kinematic constraints.

$\mathcal{L}_{\underline{p}_{\mathrm{user}}}.\mathrm{append}(\underline{p}_{\mathrm{user}})$
  ▷ Calculate how much the user moved forward/backward w.r.t. reference pose.
$\underline{t}_{\mathrm{ref}} \leftarrow [\cos\theta_{\mathrm{ref}}, \sin\theta_{\mathrm{ref}}]^{\mathsf{T}}$
$d_{\|\mathrm{ref}} \leftarrow \big(\underline{p}_{\mathrm{user}} - \underline{p}_{\mathrm{ref}}\big)^{\mathsf{T}} \underline{t}_{\mathrm{ref}}$
$b_{\mathrm{behindref}} \leftarrow \big(d_{\|\mathrm{ref}} < 0\big)$
  ▷ Calculate how much the user moved forward/backward w.r.t. pivot pose.
$d_{\|\mathrm{pivot}} \leftarrow \big(\underline{p}_{\mathrm{user}} - \underline{p}_{\mathrm{pivot}}\big)^{\mathsf{T}} [\cos\theta_{\mathrm{pivot}}, \sin\theta_{\mathrm{pivot}}]^{\mathsf{T}}$
$b_{\mathrm{behindpivot}} \leftarrow \big(d_{\|\mathrm{pivot}} < 0\big)$
  ▷ User is behind pivot pose: Problem (5) is infeasible.
if $b_{\mathrm{behindpivot}}$ **then**
  $\underline{n}_{\mathrm{pivot}} \leftarrow [-\sin\theta_{\mathrm{pivot}}, \cos\theta_{\mathrm{pivot}}]^{\mathsf{T}}$
  $d_{\perp\mathrm{pivot}} \leftarrow \big(\underline{p}_{\mathrm{user}} - \underline{p}_{\mathrm{pivot}}\big)^{\mathsf{T}} \underline{n}_{\mathrm{pivot}}$
  return
end if
  ▷ Call the local path planner from section IV-A.
$\big(\underline{p}_{\mathrm{veh,pred}}, \theta_{\mathrm{veh,pred}}, \phi_{\mathrm{pred}}\big) \leftarrow$ localPlanner $\big(\underline{p}_{\mathrm{pivot}}, \theta_{\mathrm{pivot}}, \underline{p}_{\mathrm{user}}\big)$
if $\big(\underline{p}_{\mathrm{veh,pred}}, \theta_{\mathrm{veh,pred}}, \phi_{\mathrm{pred}}\big) = \varnothing$ **then**
  return
end if
$\big(\mathcal{L}_{\underline{p}_{\mathrm{veh,pred}}}, \mathcal{L}_{\theta_{\mathrm{veh,pred}}}\big) \leftarrow$ samplePrediction $\big(\underline{p}_{\mathrm{pivot}}, \theta_{\mathrm{pivot}}, \underline{p}_{\mathrm{veh,pred}}, \phi_{\mathrm{pred}}\big)$
  ▷ Update reference pose.
if $\neg b_{\mathrm{behindref}}$ **then**
  $\big(\underline{p}_{\mathrm{ref}}, \theta_{\mathrm{ref}}\big) \leftarrow \big(\underline{p}_{\mathrm{veh,pred}}, \theta_{\mathrm{veh,pred}}\big)$
end if
  ▷ Update pivot pose.
while $\neg b_{\mathrm{behindref}} \wedge d_{\|\mathrm{pivot}} > d_{\mathrm{th}}$ **do**
  $\mathcal{L}_{\underline{p}_{\mathrm{veh,pred}}}.\mathrm{pop}(), \mathcal{L}_{\theta_{\mathrm{veh,pred}}}.\mathrm{pop}()$
  $\big(\underline{p}_{\mathrm{pivot}}, \theta_{\mathrm{pivot}}\big) \leftarrow \big(\mathcal{L}_{\underline{p}_{\mathrm{veh,pred}}}[0], \mathcal{L}_{\theta_{\mathrm{veh,pred}}}[0]\big)$
  $\mathcal{L}_{\underline{p}_{\mathrm{veh,past}}}.\mathrm{append}\big(\mathcal{L}_{\underline{p}_{\mathrm{veh,pred}}}[0]\big)$
  $d_{\|\mathrm{pivot}} \leftarrow \big(\underline{p}_{\mathrm{user}} - \underline{p}_{\mathrm{pivot}}\big)^{\mathsf{T}} [\cos\theta_{\mathrm{pivot}}, \sin\theta_{\mathrm{pivot}}]^{\mathsf{T}}$
end while

in localPlanner(). The solution of the local planner is then transformed back to match the real pivot pose and yields $\big(\underline{p}_{\mathrm{veh,pred}}, \theta_{\mathrm{veh,pred}}, \phi_{\mathrm{pred}}\big)$. Subsequently, the local prediction is discretized in samplePrediction() by numerical integration of (1). The sampling interval, i.e., the distance between two subsequent sample points on the path, is denoted as d_{sample}. The result is stored in $\big(\mathcal{L}_{\underline{p}_{\mathrm{veh,pred}}}, \mathcal{L}_{\theta_{\mathrm{veh,pred}}}\big)$ afterwards (orange path).

To detect the direction of the user motion, the reference pose $\big(\underline{p}_{\mathrm{ref}}, \theta_{\mathrm{ref}}\big)$ (orange dot) is used. An update of this pose based on the last prediction occurs whenever the user moves "forward" ($b_{\mathrm{behindref}} = 0$).

If the user has not moved backward relative to the reference pose and the forward movement $d_{\|\mathrm{pivot}}$ with respect to the pivot pose exceeds a threshold d_{th}, the pivot element is updated. To accomplish that, the first element of the predicted path is iteratively set as pivot pose until d_{th} is no longer exceeded. In parallel, all superseded pivot poses as well as the current one are stored in the past and immutable vehicle path $\mathcal{L}_{\underline{p}_{\mathrm{veh,past}}}$ (purple line).

In theory, it is sufficient when the parameters of the algorithm satisfy the condition $d_{\mathrm{sample}} \leq d_{\mathrm{th}}$. However, to achieve the best possible performance, it is recommended to choose the parameters according to $d_{\mathrm{sample}} \ll r_{\min}$ and $5\, d_{\mathrm{sample}} \leq d_{\mathrm{th}} \leq \frac{r_{\min}}{2}$. In doing so, the first condition ensures that the errors of the numerical integration are kept small. The latter one makes the algorithm robust towards noisy input positions and prevents the prediction from getting unnecessarily long.

For the initialization, the pivot pose has to be set to a feasible vehicle pose that is behind the initial vehicle position coinciding with the initial user position $\underline{p}_{\mathrm{user},0}$. Moreover, the first reference pose is assigned to the initial vehicle pose. Mathematically, this can be written as

$$\underline{p}_{\mathrm{pivot}} = \underline{p}_{\mathrm{user},0} - \frac{1}{2} d_{\mathrm{th}} \begin{bmatrix} \cos\theta_0 \\ \sin\theta_0 \end{bmatrix} , \tag{17}$$

$$\underline{p}_{\mathrm{ref}} = \underline{p}_{\mathrm{user},0} , \tag{18}$$

$$\theta_{\mathrm{pivot}} = \theta_{\mathrm{ref}} = \theta_{\mathrm{veh},0} , \tag{19}$$

where $\theta_{\mathrm{veh},0}$ is the initial heading of the vehicle.

C. Force Feedback

In order to close the loop and limit possible deviations from a feasible path, appropriate force feedback in each iteration is required. For this reason, two force components as drawn in Fig. 2 (green arrows) were designed:

1) The lateral guidance force

$$\underline{F}_{\perp} = -D_{\perp} \begin{cases} d_{\perp\mathrm{pivot}} \underline{n}_{\mathrm{pivot}} & \text{if } b_{\mathrm{behindpivot}} \\ \underline{p}_{\mathrm{user}} - \underline{p}_{\mathrm{veh,pred}} & \text{otherwise} \end{cases} \tag{20}$$

with scaling factor $D_{\perp}$ ensures that the user does not plan a path that requires the vehicle to move sideways.

2) The longitudinal guidance force

$$\underline{F}_{\|} = \begin{cases} -D_{\|}\, d_{\|\mathrm{ref}}\, \underline{t}_{\mathrm{ref}} & \text{if } b_{\mathrm{behindref}} \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

with scaling factor $D_{\|}$ prevents the user from moving backwards as this would violate constraint (2). This force is inactive if the user moves forward.

Finally, the haptic interface renders the superposition $\underline{F} = \underline{F}_{\perp} + \underline{F}_{\|}$ of both forces for the user.

V. Evaluation

The proposed algorithm was implemented as a node within the Robot Operating System (ROS) framework [13] and an experimental study was conducted. For all trials, the SMHI described in [6], [12], [14] and depicted in Fig. 4 was used. This large-scale haptic interface utilizes admittance control to render planar forces with a magnitude of up to $100\,\mathrm{N}$. In contrast to other haptic interfaces, the SMHI is able to cover a workspace of about $5 \times 5\ \mathrm{m}^2$, which facilitates the natural locomotion of its users.

A total of 18 subjects participated in the experiments. Six of them reported to be familiar with the haptic interface,

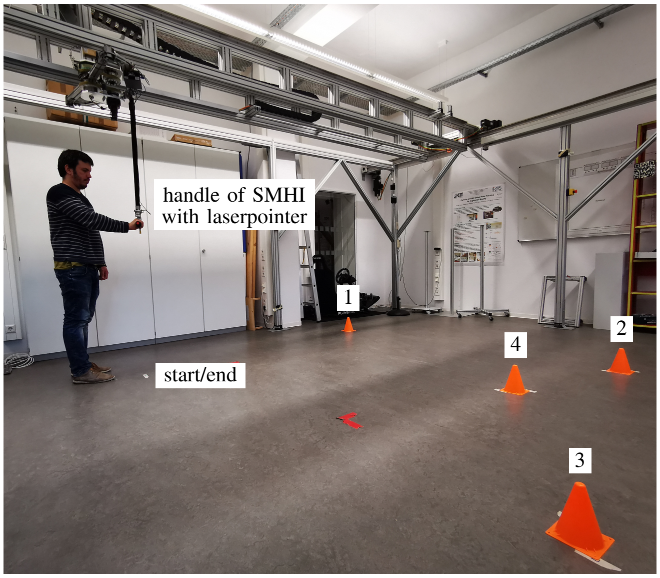


Fig. 4. Experimental setup with SMHI handle, goals, start, and end point.

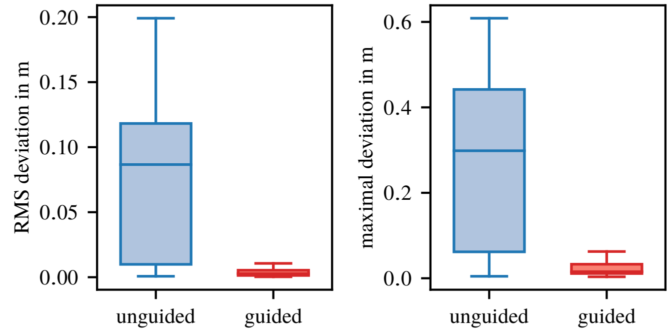
but no candidate has used the proposed algorithm before. Four of the participants had little experience with vehicle operation, eight had average experience, while six were highly experienced.

A. Experimental Setup and Procedure

The experimental setup within the SMHI is depicted in Fig. 4. The task for each subject is to move the handle of the haptic interface along a self-chosen path for a car-like vehicle, that moves through the goals 1–4 in ascending order and then returns to the starting point. Reversing is not allowed. In order to utilize the workspace effectively, the desired scenario including the vehicle is scaled down. To provide the user with a visual cue about the position of the virtual vehicle, the current handle positions is projected onto the floor using a laser pointer.

At the beginning of each experiment, the subjects are made familiar with the basic operation principles of the SMHI. The participants are then verbally informed about the initial heading and the precise minimum turning radius of the assumed vehicle kinematics. After that, the subjects are asked to execute the given task twice by "walking" a path, that satisfies the given kinematic constraints of the vehicle. During the first run, no haptic guidance is active and the SMHI is in a zero force control mode. For the immediately following second run, the proposed guidance algorithm is activated.

The parameters of the car-like kinematics were chosen as $L = 0.5\,\mathrm{m}$ and $\phi_{\max} = 35°$. According to (6), this yields a minimum turning radius of $r_{\min} = 0.71\,\mathrm{m}$. The parameters of the guidance algorithm were determined empirically to $d_{\mathrm{sample}} = 0.02\,\mathrm{m}$, $d_{\mathrm{th}} = 0.1\,\mathrm{m}$ and $D_{\perp} = D_{\parallel} = 500\,\mathrm{N\,m^{-1}}$.



(a) RMS deviation. (b) Maximal deviation.

Fig. 5. Statistical evaluation of the deviation between the desired path and the closest feasible path.

B. Performance Measure

The planned vehicle path $\mathcal{L}_{\underline{p}_{\mathrm{veh,past}}}$ cannot be used directly for the evaluation since there is no equivalent for the unguided run. However, it is possible to compare both runs by evaluating the feasibility of the user path $\mathcal{L}_{\underline{p}_{\mathrm{user}}}$. To get a quantitative measurement of the feasibility, the user path can be used as the setpoint for a path tracking controller and the deviation between the desired and the executed path can be calculated. However, the result highly depends on the type of controller being used [15] and hence, is not objective.

For this reason, the path tracking control is replaced by a global optimization problem, that finds the realizable path with the smallest possible deviation to the desired path. As a preparation for this, the user path $\mathcal{L}_{\underline{p}_{\mathrm{user}}}$ is resampled equidistantly resulting in the desired points $\underline{p}_i^*$, $i \in \{0, 1, \ldots, N\}$. These points are then fed into

$$\begin{aligned} J = \min_{\substack{\underline{u}_0,\ldots,\underline{u}_{N-1}\\ \underline{p}_0,\ldots,\underline{p}_N\\ \theta_0,\ldots,\theta_N}} & \left(\frac{1}{N+1} \sum_{i=0}^{N} \left\| \underline{p}_i - \underline{p}_i^* \right\|^2 \right)^{\frac{1}{2}} \\ \text{subject to} \quad & \text{discretization of equations (1)–(3)}\,, \\ & \underline{p}_0 = \underline{p}_0^*\,, \\ & \theta_0 = \theta_{\mathrm{veh},0}\,, \end{aligned} \tag{22}$$

which is numerically solved by CasADi [16]. Here, the points $\underline{p}_i$, $i \in \{0, \ldots, N\}$ correspond to the optimal, realizable path. J quantifies the RMS deviation between the desired and the closest realizable path. In addition, a measure for the maximum deviation can be obtained using $\max_i \left\| \underline{p}_i - \underline{p}_i^* \right\|$. Due to the global nature of the optimization, these quantities can be interpreted as a lower bound for the control performance of any path tracking controller, that follows a quadratic distance measurement.

C. Results

The RMS and maximal deviation as defined in the previous section were evaluated for both runs. A statistical analysis of the results for all subjects is depicted in Fig. 5. It can be seen that the deviation between the desired and the closest feasible path is reduced significantly when the guidance is active: For the RMS deviation, the median value

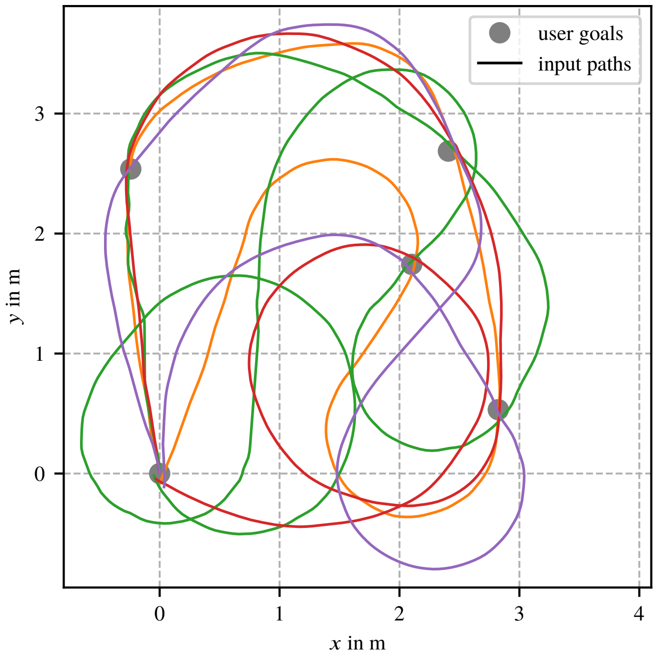


Fig. 6. Different realizations of the given task with activated guidance.

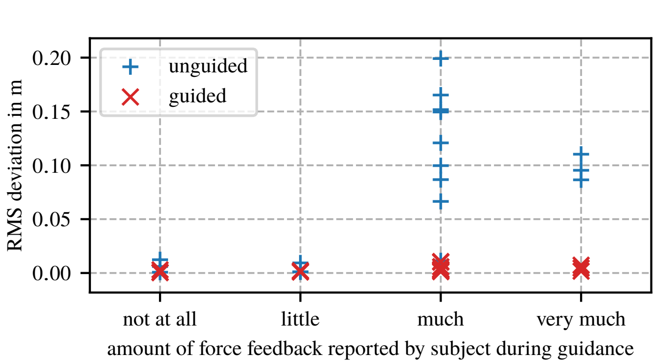
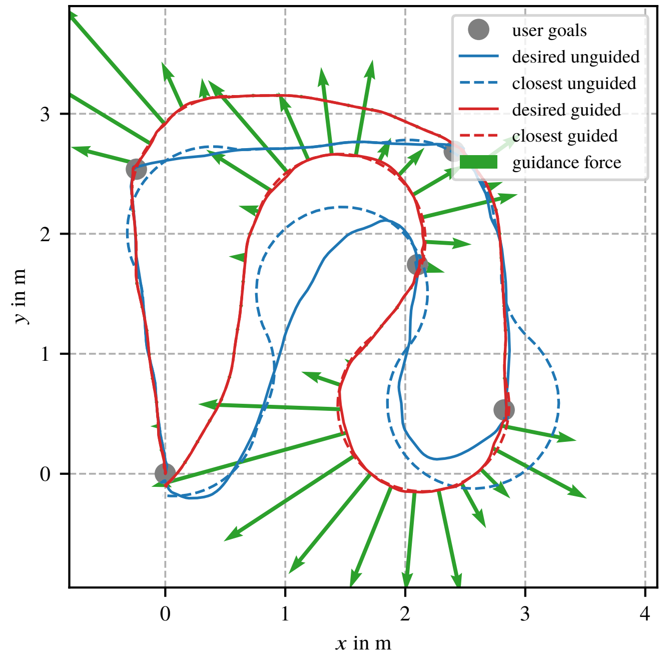


Fig. 7. Correlation between measured driver skill experienced force feedback.

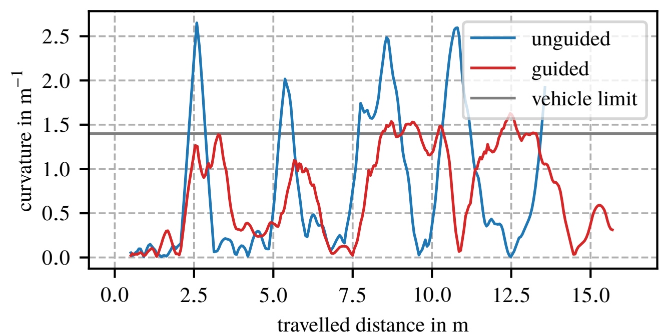
is reduced from $0.087\,\mathrm{m}$ to $0.003\,\mathrm{m}$, while the maximal value is reduced from $0.199\,\mathrm{m}$ to $0.011\,\mathrm{m}$. Similarly, the median and the maximum of the highest achieved deviation from the desired path were decreased from $0.298\,\mathrm{m}$ to $0.015\,\mathrm{m}$ and $0.609\,\mathrm{m}$ to $0.063\,\mathrm{m}$, respectively. This implies that the desired paths become more feasible in case of activated guidance, which in turn allows to use them directly as an input for a path tracking controller.

At the same time, the ability of the user to contribute their situation awareness by selecting goals and paths independently is largely unaffected: The average distance between the demanded goals and the user paths is $0.039\,\mathrm{m}$ with and $0.038\,\mathrm{m}$ without guidance across all subjects. Furthermore, Fig. 6 demonstrates that individual solutions of the task are possible and that the proposed guidance imposes no constraints beyond the vehicle kinematics.

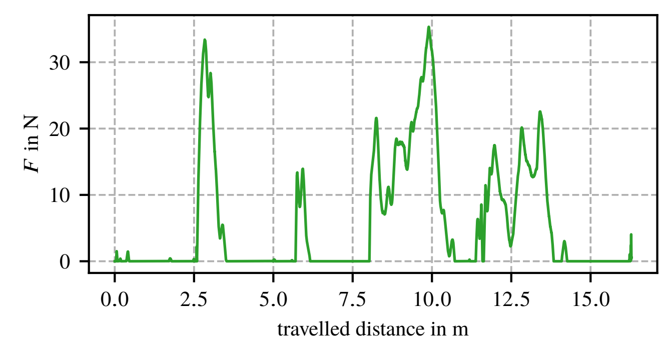
As shown in Fig. 7, there is a correlation between perceived force feedback and driver skill: A driver that already performs well (i.e., low deviation) during the run without



(a) Desired paths and their closest feasible realizations.



(b) Curvatures of the desired paths.



(c) Force exerted by the guidance algorithm during the guided run.

Fig. 8. Detailed comparison between the guided and the unguided run of the same subject.

guidance, tends to report little to no force during the run with guidance. This indicates that the proposed guidance has neither a significant effect on the performance nor on the user experience of a skilled operator.

The comparison in Fig. 8 depicts both runs of a less skilled subject to get a better understanding of how the guidance affects a user. The path without guidance in Fig. 8(a) exhibits sharp turns and large deviations to its closest feasible path, while the path with guidance has very small deviations. This is also reflected in Fig. 8(b), where the curvature of the guided user path is effectively limited to a value close to the vehicle's maximum. Fig. 8(a) and Fig. 8(c) show, that the guiding force is only exerted when the user starts to violate the kinematic constraints of the vehicle. This matches the expected behavior of a guidance algorithm that does not undermine the user's choice.

VI. CONCLUSIONS

In this paper, a new path generation method for the teleoperation of car-like robotic platforms was presented. Haptic feedback was introduced to ensure the feasibility of the path pursued by the user. This was achieved by the combination of an optimization-based local path planner, that predicts a feasible vehicle path from a given pivot pose to the current user position, and a higher level path generation algorithm that allows the creation of arbitrary long paths through iterative updates of this pivot pose. To close the loop between the user and the algorithm, appropriate guidance forces were defined.

Experiments showed the effectiveness of the proposed method: Compared to an unguided path generation, the guidance algorithm significantly increased the feasibility of the path that is pursued by the user, which means that the generated paths can be used in scenarios requiring high precision. Despite the guidance, users were still fully capable of contributing their experience and their understanding of the situation. Furthermore, it was found that skilled operators are only slightly affected by the guidance algorithm, whereas less skilled operators get appropriate force feedback, that can be useful for learning the vehicle characteristics.

Future work will focus on extending the proposed method to additional kinematic structures. In addition, the application to dynamic instead of kinematic vehicle models will be investigated.

REFERENCES

[1] Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IOSB. (2020, Oct.) ROBDEKON: Robot systems for the decontamination in inhospitable environments. [Online]. Available: https://robdekon.de

[2] B. Pitzer, M. Styer, C. Bersch, C. DuHadway, and J. Becker, "Towards perceptual shared autonomy for robotic mobile manipulation," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 6245–6251.

[3] S. M. LaValle, *Planning Algorithms*. Cambridge: Cambridge University Press, 2006.

[4] L. E. Kavraki and S. M. LaValle, "Motion planning," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer, 2008, pp. 109–131.

[5] M. Taïx, D. Flavigné, and E. Ferré, "Human interaction with motion planning algorithm," *Journal of Intelligent & Robotic Systems*, vol. 67, no. 3, pp. 285–306, Sept. 2012.

[6] A. Pérez Arias and U. D. Hanebeck, "Wide-area haptic guidance: Taking the user by the hand," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2010, pp. 5824–5829.

[7] A. Pérez Arias, H. P. Eberhardt, F. Pfaff, and U. D. Hanebeck, "The plenhaptic guidance function for intuitive navigation in extended range telepresence scenarios," in *2011 IEEE World Haptics Conference*, June 2011, pp. 475–480.

[8] P. Marayong, M. Li, A. M. Okamura, and G. D. Hager, "Spatial motion constraints: Theory and demonstrations for robot guidance using virtual fixtures," in *2003 IEEE International Conference on Robotics and Automation*, vol. 2, Sept. 2003, pp. 1954–1959.

[9] N. Ladeveze, J.-Y. Fourquet, and B. Puel, "Interactive path planning for haptic assistance in assembly tasks," *Computer & Graphics*, vol. 34, no. 1, pp. 17–25, Feb. 2010.

[10] R. J. Kuiper, D. J. F. Heck, I. A. Kuling, and D. A. Abbink, "Evaluation of haptic and visual cues for repulsive or attractive guidance in nonholonomic steering tasks," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 5, pp. 672–683, Oct. 2016.

[11] R. Rahal, F. Abi-Farraj, P. Giordano, and C. Pacchierotti, "Haptic shared-control methods for robotic cutting under nonholonomic constraints," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov. 2019, pp. 8151–8157.

[12] A. Pérez Arias and U. D. Hanebeck, "A novel haptic interface for extended range telepresence: Control and evaluation," in *Proceedings of the 6th International Conference on Informatics in Control, Automation and Robotics*, vol. 1, July 2009, pp. 222–227.

[13] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: An open-source robot operating system," in *ICRA workshop on open source software*, 2009.

[14] A. Pérez Arias and U. D. Hanebeck, "Motion control of a semi-mobile haptic interface for extended range telepresence," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept. 2011, pp. 3053–3059.

[15] N. H. Amer, H. Zamzuri, K. Hudha, and Z. A. Kadir, "Modelling and control strategies in path tracking control for autonomous ground vehicles: A review of state of the art and challenges," *Journal of Intelligent & Robotic Systems*, vol. 86, no. 2, pp. 225–254, May 2017.

[16] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, Mar. 2019.