

# Clothoidal local path template for intention estimation by assistive mobile robots

## Abstract

This work proposes and evaluates various improvements to a circular local path template (LPT) that we have designed in the past to estimate driver intents and to provide navigation assistance in teleoperation or wheelchair applications. This LPT may also be used in mobile robotics for fast collision checking in path planning or obstacle avoidance algorithms. The LPT consists of a fixed set of paths that the mobile robot can follow in its local neighbourhood. Based on an efficient look-up table, all path lengths are adjusted such that they are collision-free. However, experiments have shown that in dense environments insufficient paths were found, which impedes correct intention estimation and thus navigation assistance. In this work, the use of clothoidal paths rather than circular paths is evaluated. This substantially improved the capability to find complex paths in dense areas. Furthermore, we adapted the LPT to deal with dynamic obstacles of random shape using motion estimates of these objects.

## Introduction

The ability to move around to any desired location is critical to all human development, activity and interaction. With the increase in average age in almost all nowadays societies, the loss of mobility caused by reduced physical capabilities frequently leads to a loss in social contact and therefore quality of life. The use of robotic technologies can give back a level of mobility and sense of autonomy to the elderly or disabled. For example, a powered wheelchair enables a person with motion impairment to regain movement control and thus to re-engage in more frequent human interactions. However, this renewed mobility comes with its own constraints. Wheelchairs are relatively large compared to their indoor environment. In order to realise the fine maneuvers to navigate the chair without colliding, a significant degree of dexterity is needed. These tasks are even more demanding due to the non-holonomic characteristics of most wheelchairs. It is therefore often a tiresome and frustrating task for elderly and disabled people to control their wheelchair properly.

Similar to several other research groups, we have developed navigation assistance algorithms in the past to lower the users' workload for maneuvering their wheelchair, see

e.g. (Demeester et al. 2012a). This is performed by estimating the navigation intention of the driver and by modifying the driver's steering signals based on the estimated intention. For this, effective observation of the wheelchair's environment and appropriate interfaces to give feedback to the driver are needed. In order to realise this, we have equipped a commercially available wheelchair with several sensors and a haptic interface as shown in Figure 1 (top). Estimation

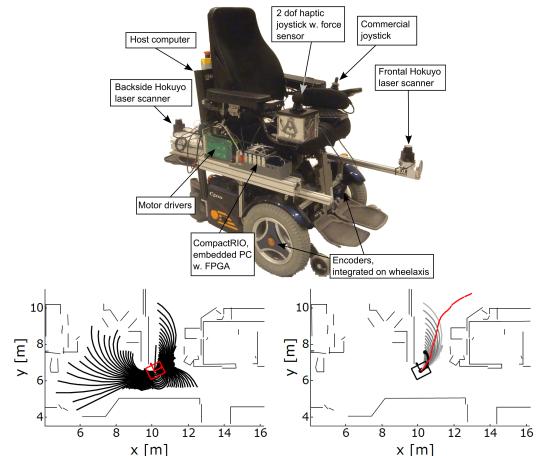


Figure 1: (top) Example of a commercially available wheelchair upgraded with a custom-made haptic interface, laserscanners perceiving the environment and encoders for a more accurate position measurement (Vander Poorten et al. 2012). (bottom, left) Collision-free trajectory generation for a wheelchair in an apartment and plan recognition results (bottom, right) at that location. The rectangle represents the wheelchair, the red path is the user's actual navigation plan; the darker a path, the higher its probability.

of the driver's navigation plan is performed as follows. We model the driver's intention  $i_k$  at time instant  $k$  as a path, a succession of desired robot states from the current state  $\mathbf{x}_{current}$  to a goal state  $\mathbf{x}_{goal}$ :  $i_k = \{\mathbf{x}_{current}, \dots, \mathbf{x}_{goal}\}$ . The robot's state is defined as  $\mathbf{x} = [x \ y \ \theta]^T$  with  $\theta$  the robot's orientation. A probability  $p$  is computed for each local path as shown in Equation 1. Bayes' theorem is adopted to compute the posterior probability  $p_{post}$  on  $i_k$ , based on the given

steering signals ( $\mathbf{u}_k$ ) and a history ( $\mathbf{H}$ ) of driver signals, robot actions, robot poses and sensor readings.  $p_{prior}$  is the prior probability distribution on the set of paths and  $p_{user}$  is a user model, which models how the driver transforms a particular intention  $i_k$  into a certain steering signal  $\mathbf{u}_k$ .  $\eta$  is a scale factor normalizing the probability distribution (Demeester et al. 2012a).

$$\begin{aligned} p_{post}(\mathbf{i}_k \mid \mathbf{u}_k, \mathbf{H}_{0:k}) \\ = p_{user}(\mathbf{u}_k \mid \mathbf{i}_k, \mathbf{H}_{0:k}) \cdot p_{prior}(\mathbf{i}_k \mid \mathbf{H}_{0:k}) \cdot \eta. \end{aligned} \quad (1)$$

In the past, we have adopted a set of circular local paths as an approximation of potential driver navigation plans. This seemed reasonable, as the wheelchair's behaviour can be approximated locally as a circular path given its non-holonomic constraints. Furthermore, it was required to provide haptic feedback at a high frequency (Vander Poorten et al. 2012). In order to realise this, we needed a very fast and accurate collision checking algorithm that could verify potential collisions between all paths and detected obstacles. This was obtained through an obstacle-based look-up table (Demeester et al. 2012b) that uses a fixed set of local paths. However, practical experiments showed that in dense environments, e.g. when entering narrow doors or elevator entrances, too few or even no paths at all were found through these narrow passageways as shown in Figure 2 (left). As a result, no appropriate intent estimation and thus navigation assistance could be provided under these circumstances. This work's goal is to improve the richness of the local paths such that substantially more local paths through narrow passageways can be found. More specifically, we explore whether clothoidal geometries can be adopted for this, see Figure 2 (right). The remainder of the paper is organized

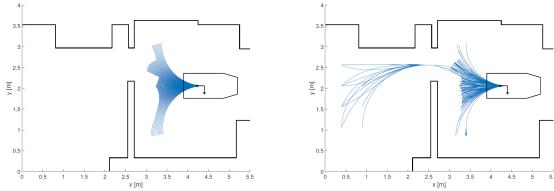


Figure 2: (left) Circular versus clothoidal curves to find paths through narrow doorways, showing that the clothoidal paths have a higher chance to find paths in dense environments.

as follows. First, the clothoidal local path template is described. Then, an extension of this local path template is given that is able to deal with dynamic obstacles. Next, the clothoidal local path template is evaluated and its improved planning capability in dense environments is demonstrated. The paper ends with some conclusions and future work.

### Clothoidal local path template

This section describes the formation of a clothoidal local path template (LPT). The LPT as proposed in (Demeester et al. 2012b) consists of a fixed set (or *template*) of feasible geometric paths starting from the robot's current pose

(hence *local*). A feasible path means that this path respects the robot's kinematic constraints. In (Demeester et al. 2012b), kinematically feasible paths are obtained with the forward generation method, i.e. by integrating achievable linear and angular velocity ( $v, \omega$ ) pairs, over a certain amount of time  $\Delta t$ . (Pivtoraiko and Kelly 2012) propose a more flexible method of creating a set of paths (inverse generation method). They introduce a novel search space, the State Lattice, which represents a discrete set of states connected with feasible paths. These paths are called Motion Primitives (MPs) and connect nearby discretised poses with paths compliant with the robot's motion constraints.

Our clothoidal LPT is created by using a Local State Lattice (LSL) (a local version of the State Lattice of (Pivtoraiko and Kelly 2012)) and an obstacle-based lookup table for collision checking to rapidly obtain a set of collision-free paths. The method can be summarised as follows:

1. Generation of a multi-size grid (fine close to the robot and coarse far from it).
2. Each grid cell in the Region Of Interest (ROI) around the origin is connected with a clothoidal curve. If this path is feasible, it is added to the set of MPs.
3. To ensure a certain degree of flexibility, step 2 is repeated at certain grid cells, called Expansion Positions (EPs). By doing this, certain paths in the LSL will consist of two subsequent curves.
4. The Occupancy Grid (OG) is calculated for each path, by using the geometry of the wheelchair. A look-up table is built for this fixed set of trajectories to quickly assess if a motion is collision-free and where the collision occurs.
5. The LPT algorithm updates each path by adjusting its length in order to be collision-free within the perceived environment.

Steps 1 to 4 are executed offline, whereas step 5 is performed online. The following paragraphs describe each of the steps.

**Step 1 - Generation of a multi-size grid** The LSL constitutes a fixed set of feasible trajectories originating from the robot's current pose to nearby end poses. The first step in the creation of the LSL is to sample the reachable space surrounding the robot by using a multi-size grid (MSG). A clothoidal curve will then be connected to each of these discrete poses, if possible. The MSG is composed of three different sizes, fine, medium and coarse, as shown in Figure 3 (top). The inverse generation method is adopted to create a set of MPs. First, the environment is sampled and a boundary value problem is solved to connect these discrete states with feasible paths using a defined curve geometry. The inverse approach is preferred in this implementation as it makes the state discretization the driver of the design of the LSL. Parameters can then be fine-tuned depending on the application. The main reason to adopt a MSG is to limit the number of paths far away from the origin. It would be inefficient to keep a larger number of paths leading to a pose that is relatively similar with respect to the origin, since the environment and the intention of the driver may change. An

end pose located in the coarse grid represents therefore a large group of poses compared to an end pose in the fine grid. A ROI selects the Candidate End Poses (CEPs) of the grid cells in the immediate surrounding of the origin. This step is illustrated in Figure 3 (top). Depending on the chosen geometry of the MP and the constraints of the robot, a CEP will be reachable or not. If it is reachable, the path leading to the CEP will be added to the set of MPs. Values of the used parameters and a short description can be found in Table 1.

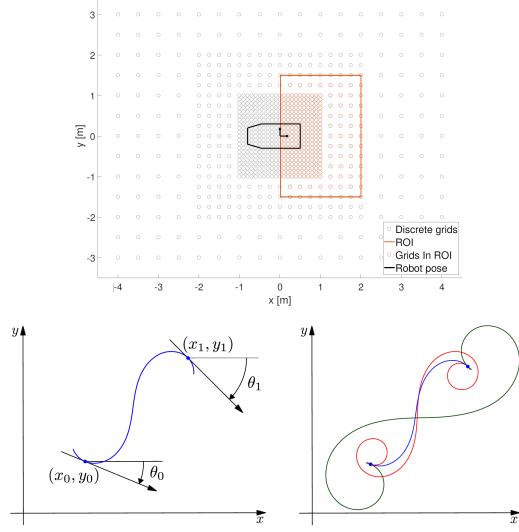


Figure 3: (top) Multi-size grid (MSG) and region of interest (ROI) shown at the origin in the robot's local coordinate frame. Three different grid sizes are used to sample the environment. Some grid cells in the ROI become candidate expansion points (CEPs), which will be connected to the origin by a geometrical curve. (bottom, left) Notation for the G1 Hermite interpolation scheme, and (bottom, right) yielding multiple solutions using clothoids (Bertolazzi and Frego 2015).

**Step 2 - Connection of grid cells in the ROI with a geometrical curve** Once the CEPs around the origin are determined, a geometrical connection has to be established between the origin and these CEPs. In this work, clothoids have been adopted, but other choices are possible. Clothoids (also called Euler or Cornu spirals) are curves whose curvature changes linearly with their arc length, see Equation 2. Clothoids are frequently used for the design of railway tracks (Cope 1993) and highway design (Baass 1982) to connect a tangent to a circular curve, resulting in a continuous curvature profile. Joining a tangent and a circular curve directly would result in a discontinuity, thus leading to an instantaneous change in the centripetal acceleration, causing discomfort to passengers. Clothoids have also found their application in path planning for mobile robots (Fleury et al. 1995; Brezak and Petrović 2011; Kelly and Nagy 2003; Scheuer and Fraichard 1997). An iterative process has to take place in order to connect one pose  $p_0$  to another  $p_1$

Table 1: LSL parameters and corresponding values.

parameter	value	description
$dx_1, dy_1$	0.10 m	Discretization of the fine grid.
$x_{1,max}, y_{1,max}$	1.00 m	Width and height of the fine grid.
$dx_2, dy_2$	0.25 m	Discretization of the medium grid.
$x_{2,max}, y_{2,max}$	2.00 m	Width and height of the medium grid.
$dx_3, dy_3$	0.50 m	Discretization of the coarse grid.
$x_{3,max}$	4.00 m	Width of the coarse grid.
$y_{3,max}$	3.00 m	Height of the coarse grid.
$d\theta$	$\pi/8$ rad	Angular discretization, constant over the whole grid.
$x_{ROI}$	2.00 m	x-distance (+/-) defining the ROI.
$y_{ROI}$	1.50 m	y-distance (+/-) defining the ROI.
$\kappa_{max}$	$1 \text{ m}^{-1}$	Maximum allowed curvature.
$dx_{EP}$	0.50 m	Manhattan Distance ( $\ell_1$ -norm) between EPs
$res_{OG}$	2 cm	Resolution of the OG
$res_{path}$	1 cm	Resolution of the path

using a clothoid, because there is no unique G1 fitting solution, see Figure 3 (b). Extensive research has been done to find stable numerical solutions to calculate the Hermite G1 interpolation with a single clothoid curve, which can be formulated as a system of three nonlinear equations (yielding multiple solutions), see Equations 3 to 5. (Walton and Meek 2009) designed their algorithm to handle three different situations, straight lines, circles and clothoids to then solve only one single nonlinear equation. However, when the solution of a clothoid approaches the shape of a circle ( $\kappa' \approx 0$ ) or a straight line ( $\kappa = \kappa' \approx 0$ ) the root of the nonlinear equation becomes ill-conditioned, resulting in numerical errors. (Bertolazzi and Frego 2015) solve this problem by recasting the problem into a well-conditioned zero of a unique nonlinear equation. Moreover, their algorithm does not treat straight lines, circles and clothoids differently and thus achieves robust results at the transition zones. Their solution has been used without modification to generate the clothoids, see Figure 4 (a).

$$\kappa(s) = \kappa' s + \kappa_0, \kappa', \text{change of curvature, constant} \quad (2)$$

$$\theta'(s) = \kappa(s), \text{where } \theta(0) = \theta_0, \theta(l) = \theta_1 \quad (3)$$

$$x'(s) = \cos \theta(s), \text{where } x(0) = x_0, x(l) = x_1 \quad (4)$$

$$y'(s) = \sin \theta(s), \text{where } y(0) = y_0, y(l) = y_1 \quad (5)$$

By applying G1 interpolation, the smooth transition of curvatures is not guaranteed; this would require G2 Hermite interpolation. G2 fitting would result in a smoother curve but will be computationally more expensive due to additional constraints. As G1 continuity is acceptable for this application, G2 is not applied. If G2 continuity is preferred, e.g. for the generation of local paths for car-like vehicles, the use of piecewise clothoid curves is necessary, which has been developed in (McCrae and Singh 2009).

**Step 3 - Expansion points** Finally, in order to propose a larger variety of paths, the procedure of connecting CEPs to the set of MPs is repeated at certain discrete positions, which we call Expansion Positions (EPs). EPs are defined as reachable end poses directly connected to the origin, with a Manhattan Distance ( $\ell_1$ -norm) equal to a multiple of  $dx_{EP}$ . It should be noted that those discrete poses in the ROI (at that EP) will be connected with the pose at that EP, and not at the origin. There will be significantly less CEPs if the EP

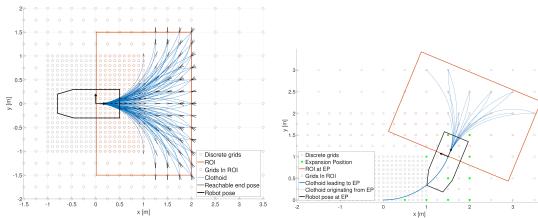


Figure 4: (a) Set of MPs based on clothoids, connecting the origin with feasible states within the ROI. (b) Example of the EP and ROI at  $[1.5, 1, 45^\circ]$ . CEPs are now linked with the pose that the robot would have at that EP and not with the origin.

is further away from the origin, due to the use of the MSG, thereby reducing the number of paths leading far away from the origin. An example of this procedure is shown in Figure 4 (b).

Repeating this procedure for every EP results in a LSL, a set of paths starting from the origin and connecting end poses in the surrounding of the wheelchair with feasible clothoidal trajectories. This is illustrated in Figure 5. Algorithm 1 provides an overview of the complete procedure. The output of this algorithm is the LSL data structure as presented in Table 2, containing all required information to reconstruct the set of paths.

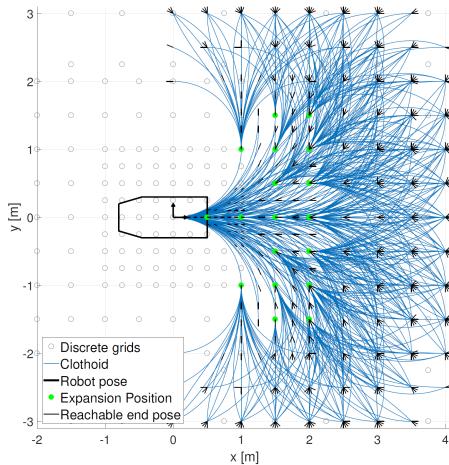


Figure 5: A LSL based on clothoids is obtained after repeating the procedure shown in Figure 4 (b) at every EP. This connects every feasible end pose close to the robot whilst moving forward with one or two clothoids. Backward movements and on-the-spot-turning are also implemented but are not illustrated in the present figure to maintain readability.

**Step 4 - Offline construction of lookup table for collision checking** Once the set of feasible clothoidal trajectories are defined, a lookup table is constructed offline to quickly evaluate online which paths are collision-free and to adjust their length if needed. The first step in the creation of the

lookup table is to calculate the occupancy grid (OG) of each path. This is the space that the wheelchair will occupy when moving along a particular trajectory. The LSL structure defined in Table 2 contains all the information needed to reconstruct the calculated paths. The path index ( $pathIdx$ ) is the row index of  $\mathbf{XY}$ . This matrix contains all the discrete poses the mobile robot will adopt while following the path between the start pose  $p_0$  and the end pose  $p_1$ . The same row index can be used to access other information, such as current path length ( $s$ ) or path curvature ( $\kappa$ ). The resolution for every curve is fixed at  $s(k) - s(k-1) = res_{path} < res_{OG}/2$  to ensure that all cells of the OG are examined (concrete values given in Table 1). The OG of the robot at the ori-

Table 2: REPRESENTATION OF THE LSL DATA STRUCTURE FOR A SINGLE PATH. This structure includes all necessary information to represent each path in the LSL and comprises precomputed data  $(\mathbf{XY}\Theta, s, \kappa)$ . These vectors contain the position and orientation, path length and curvature along each path, discretized in such a way that  $s_{k+1} - s_k < res_{path}$ . The  $blockIdx$  entry (used in the online phase) represents the row index at which a certain path is blocked, meaning that if the row index is increased by one, the path will result in a collision with the environment. Prior to taking obstacles into account, this index is put to  $rowlength(\mathbf{XY}\Theta) + 1$ , meaning that the path should not be shortened. Finally, each path has a unique  $ID$ .

$p_0$	$p_1$	$L_{tot}$	$\kappa_0$	$\kappa'$	$\mathbf{XY}\Theta$	$s$	$\kappa$	$blockIdx$	$ID$
1x3	1x3	0.516	0.78	-0.05	$52 \times 3$	$52 \times 1$	$52 \times 1$	53	14

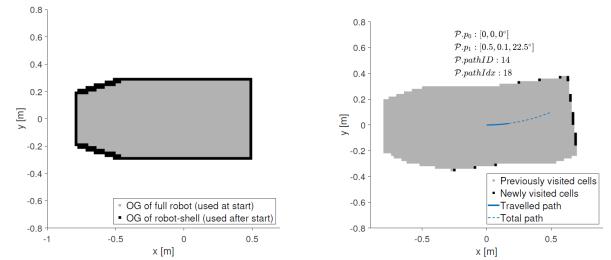


Figure 6: (a) Occupancy grid (OG) of the wheelchair's footprint. (b) The path OG is created by moving the robot along the path. Each time a grid cell is visited for the first time, it is stored along with the path-ID and path-index. This path-index represents the position along the path. The distance between 2 path indices is at most equal to  $res_{path} < res_{OG}/2$  to ensure that no cells in the OG are skipped along the path.

gin is shown in Figure 6. As the mobile robot moves along the path, grid cells that were not visited during the previous steps are stored, along with the current path index and path ID. This procedure is shown in Figure 6 (b). The reason for storing this path index is to be able to efficiently adjust the path length in the presence of obstacles in the online phase (step 5).

**Step 5 - Online computation of collision-free paths** The lookup table is constructed based on the cells the robot occupies when following the clothoidal paths of the LPT. Each entry in this table represents a cell in the grid that, if that cell is occupied, results in a collision with one or more paths from the LSL. The pathID and pathIdx of each trajectory going through this cell are also stored at that entry in the lookup table. Once this table is built, the occupied cells in the grid map representing the environment have to be matched with the cells in the lookup table. If there is a match, each path length is updated by the provided information of that cell (pathID and pathIdx). Note that this only takes place if this reduces the path length. This procedure is displayed in Algorithm 2. An example of the path length adjustment is illustrated in Figure 7. A single entry of the lookup table data structure can be found in Table 3.

Table 3: REPRESENTATION OF A SINGLE CELL OF THE OBSTACLE-BASED LOOKUP TABLE DATA STRUCTURE. The entries  $x$  and  $y$  represent the location of the occupied grid cell by the robot (up to a resolution  $res_{grid} = 2\text{cm}$ ) and contain all the paths (represented by  $pathID$ ) going through this particular cell along with the corresponding path length at which this cell is occupied for the first time by this particular path (represented by  $pathIdx$ , the row index of  $\mathbf{XY}\Theta$ ). This cell is the occupied cell shown in Figure 7 at the top right corner.

$x$	$y$	$pathID$	$pathIdx$
3.90 m	2.96 m	[763, 764, 779]	[202, 192, 226]

**Algorithm 1** OVERVIEW OF THE LSL ALGORITHM computing the structure ( $\mathcal{L}\mathcal{S}\mathcal{L}$ ) shown in table 2. Each new entry (row) in this structure represents a feasible path ( $\mathcal{P}$ ). Paths are added incrementally to the structure in line 9 and 19. The dot-operator (.) is used to access individual fields of each path of  $\mathcal{L}\mathcal{S}\mathcal{L}$ .

```

Input:  $userSettings$  to create matrix  $P_{grid}$  containing all discrete poses  $p_{grid,k}$ .
Output: LSL Structure ( $\mathcal{L}\mathcal{S}\mathcal{L}$ ) containing a set of feasible paths ( $\mathcal{P}$ ).
    % calculate feasible paths at origin
1:  $p_0 \leftarrow [0, 0, 0]$ 
2:  $P_{grid} \leftarrow \text{CREATEMULTISIZEGRID}(userSettings)$ 
3: for all  $p_{grid,i} \in P_{grid}$  do
4:   if  $\text{ISINROI}(p_{grid,i}, p_0)$  then
5:      $p_1 \leftarrow p_{grid,i}$ 
6:      $[x, y, \theta, s, \kappa] \leftarrow \text{GETCLOTHOIDDATA}(p_0, p_1)$ 
7:     if  $\|\kappa\|_\infty \leq \kappa_{max}$  then
8:        $\mathcal{P} \leftarrow [p_0, p_1, x, y, \theta, s, \kappa]$ 
9:        $\mathcal{L}\mathcal{S}\mathcal{L} \leftarrow \text{ADDPATHTOLSLSTRUCT}(\mathcal{L}\mathcal{S}\mathcal{L}, \mathcal{P})$ 
    % calculate feasible paths at Expansion Positions
10: for all  $\mathcal{P} \in \mathcal{L}\mathcal{S}\mathcal{L}$  do
11:   if  $\text{ISEXPANSIONPOSITION}(\mathcal{P}, p_1)$  then
12:      $p_0 \leftarrow \mathcal{P}.p_1$            % end pose path  $\mathcal{P}$  becomes start pose for next paths
13:     for all  $p_{grid,i} \in P_{grid}$  do
14:       if  $\text{ISINROI}(p_{grid,i}, p_0)$  then
15:          $p_1 \leftarrow p_{grid,i}$ 
16:          $[x, y, \theta, s, \kappa] \leftarrow \text{GETCLOTHOIDDATA}(p_0, p_1)$ 
17:         if  $\|\kappa\|_\infty \leq \kappa_{max}$  then
18:            $\mathcal{P} \leftarrow [p_0, p_1, x, y, \theta, s, \kappa]$ 
19:            $\mathcal{L}\mathcal{S}\mathcal{L} \leftarrow \text{ADDPATHTOLSLSTRUCT}(\mathcal{L}\mathcal{S}\mathcal{L}, \mathcal{P})$ 
20:  $\mathcal{L}\mathcal{S}\mathcal{L} \leftarrow \text{CLEANUPLSL}(\mathcal{L}\mathcal{S}\mathcal{L})$            % Remove non-unique paths

```

---

**Algorithm 2** OBSTACLE-BASED LOOKUP TABLE (online phase)

---

```

1: for all  $\mathcal{P} \in \mathcal{L}\mathcal{S}\mathcal{L}$  do
2:    $\mathcal{P}.blockIdx \leftarrow \text{ROWLENGTH}(\mathcal{P}.\mathbf{XY}\Theta) + 1$            % initialize path  $\mathcal{P}$  as free
3: for all  $\mathcal{C} \in \text{GETCELLS OCCUPIED BY LSL}(\mathcal{L}\mathcal{S}\mathcal{L})$  do
4:   if  $\text{GRIDMAPENVIRONMENT}(\mathcal{C}) == \text{occupied}$  then
5:     for all  $\mathcal{P} \in \text{LOOKUPTABLEPATHSINCCELL}(\mathcal{C}, \mathcal{P})$  do
6:       if  $\mathcal{P}.blockIdx > \text{LOOKUPTABLEBLOCKIDX}(\mathcal{C}, \mathcal{P})$  then
7:          $\mathcal{P}.blockIdx \leftarrow \text{LOOKUPTABLEBLOCKIDX}(\mathcal{C}, \mathcal{P})$ 

```

---

## Use of the LPT with dynamic obstacles

This section describes an extension of the clothoidal local path template to deal with dynamic obstacles of which the motion is known. Such a situation is shown in Figure 8. The solution put forward to avoid such collisions relies on the capability of the LPT to perform path adjustments for static obstacles and to calculate an optimal speed profile ( $v(t)$ ) to achieve a collision-free motion avoiding dynamic obstacles. This optimal speed profile is calculated by computing a constraint optimization problem (COP). This method assumes a given motion model for each obstacle and an OG representing their shape. There is however no restriction to the path or the shape (convex or concave) of the obstacles. The next

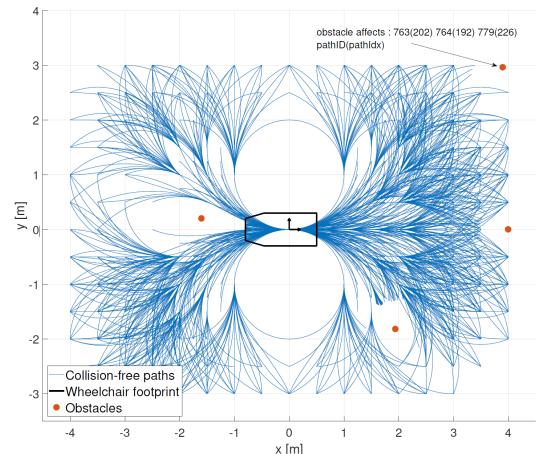


Figure 7: Example of fast online path length adjustment for 3 obstacles points. Each obstacle is exactly one grid cell (although inflated in the figure for readability). The influence on each path of the top right obstacle is shown.

section first explains the creation of a distance-time collision space ( $s, t$  space) where  $s$  is the distance along a fixed path of the LPT. This collision space will be used for a COP, using a dynamic model of the wheelchair along with time-varying separating hyperplanes to provide collision-free motion by finding an optimal speed profile to reach the end of a given path.

## Distance-time collision space

The  $s, t$  space can be seen as a grid, within which all collisions between the robot positioned at a certain distance  $s$  along the fixed path and the time-varying position of the

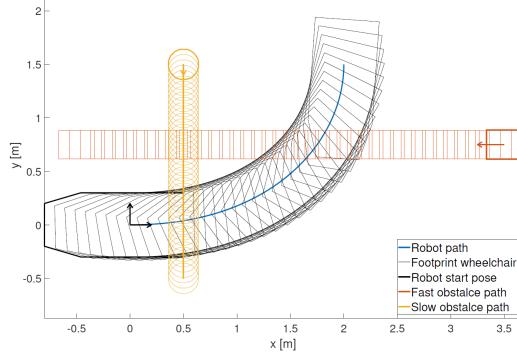


Figure 8: Dynamic obstacles potentially causing a collision with the mobile robot. For clarity, only the shell of the dynamic obstacles' OG are shown.

moving obstacle are calculated. This is computed with the following procedure:

1. The full OG of the robot along the fixed path is used, to determine the first and last impact time of the moving obstacle. This will narrow the search space along the  $t$ -axis (Figure 9 (a)).

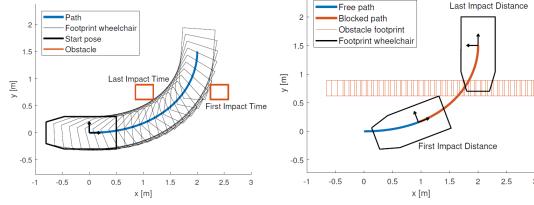


Figure 9: First and last impact time and distance are calculated to reduce the computational time of the individual collision states in the  $s, t$  space.

2. The full OG of the moving obstacle is determined by integrating its velocity. The first and last position on the fixed robot-path colliding with the moving obstacle is calculated. This will narrow the search space along the  $s$ -axis (Figure 9 (b)).
3. In the restricted space determined by  $[t_{first}, s_{first}] - [t_{last}, s_{last}]$ , all discrete  $s, t$  collision states are calculated, determined by the path resolution ( $s$ ) and the time resolution ( $t$ ). An example at a fixed time  $t = 1.25\text{ s}$  for the obstacle is shown in Figure 10 (a). Repeating this for every time instance results in a  $s, t$  space grid, indicating which  $s, t$  pair results in a collision, shown in Figure 10 (b).
4. Further optimisation can be performed on the obtained collision states. As time-varying separating hyperplanes will be used in the COP, it will be more efficient to only keep a convex shape of the obtained collision states. This convex shape can be further simplified when applying

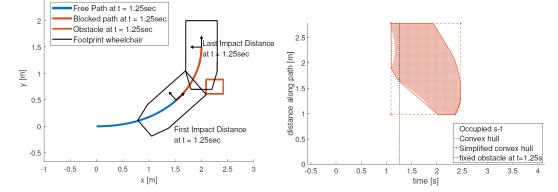


Figure 10: (a) The distance-time collision space is obtained by calculating the discrete distance along the robot-path resulting in a collision at a certain time with the moving obstacle. The slashed vertical line at  $t = 1.25\text{ s}$  corresponds to the situation shown on the left, where distances yielding a collision along the robot-path are marked in red for the position of the obstacle at  $t = 1.25\text{ s}$ . The blocked path corresponds to the occupied  $s, t$  pairs on the distance-time collision space grid map.

the following constraints: (i) assuming that the mobile robot cannot move backwards and (ii) time always moves forward. This results in a nearly rectangular shape determined by  $[t_{first}, s_{first}] - [t_{last}, s_{last}]$ . The resulting shape of the obstacle in  $s, t$  space is not always rectangular, as can be seen in the ( $s, t$  - space) for another dynamic obstacle in Figure 11.

### Optimal speed profile calculation

A COP can be formulated to find an optimal speed profile given a distance-time collision space compliant with kinematic and dynamic constraints of the mobile robot and an objective, which is to arrive within the minimum amount of time. This is shown in Equation 6. This results in finding a path in the  $s, t$  - space, from position  $[0, 0]$  to  $[s(\text{end}), t(\text{end})]$  without colliding with the simplified convex hull of the collision states. This is shown in Figure 11.

$$\begin{aligned}
 & \underset{(\mathbf{x}_{1:N}, u_{1:N}, \mathbf{a}_{1:N}, b_{1:N}, T)}{\text{minimize}} && T \\
 & \text{subject to} \\
 & \mathbf{x}_{k+1} = \mathbf{I}(\mathbf{f}(\mathbf{x}_k, u_k, h), k = 1 \dots N, \\
 & \mathbf{a}_k^T \mathbf{v}_i - b_k \geq 0, i = 1 : N_{\text{vertices}}, \\
 & \mathbf{a}_k^T \mathbf{x}_k - b_k \leq -r_{\text{safe}}, \\
 & \mathbf{x}_1 = [0, v_{\text{start}}], \\
 & \mathbf{x}_{N,1} = s(\text{end}), \\
 & 0 \leq \mathbf{x}_{k,2} \leq v_{\text{max}}, \\
 & u_{\text{min}} \leq u_k \leq u_{\text{max}}, \\
 & T \geq 0, \\
 & \|\mathbf{a}_k\| \leq 1
 \end{aligned} \tag{6}$$

The COP is formulated based on the multiple-shooting approach, resulting in the discretization of the state along the path  $(\mathbf{x}(t) = [s(t), v(t)])$  and force input  $u(t)$  over a finite grid of  $N$  samples. The model representing the wheelchair dynamics ( $\mathbf{f}$ ) is then integrated with an integrator function ( $\mathbf{I}$ ). For this solution, a simple mass-damper model was used. The time ( $T$ ) needed to arrive at the end of the path ( $s(\text{end})$ ) is kept as a decision variable, therefore an extra

variable is defined, the variable time step  $h = T/N$ . Although not explicitly formulated in the COP, there is a one time-varying hyperplane for every moving obstacle, each requiring two parameters  $a(t)$  and  $b(t)$  (Mercy, Loock, and Pipeleers 2016).  $v_i$  are the fixed vertices (positions) defining the simplified convex hull of the  $(s, t)$ -space.  $r_{safe}$  is a safety factor, enforcing a minimum distance in the  $s, t$ -plane between obtained path and the simplified convex hull. Start conditions are enforced on both start distance and speed, but only an end distance is set as a constraint for the end state.

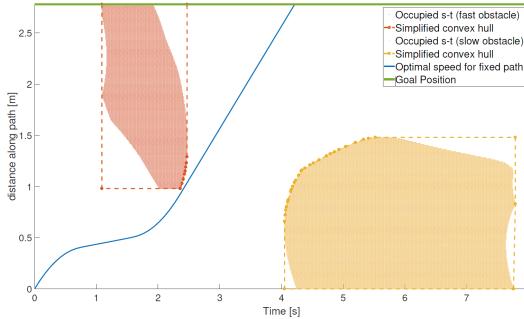


Figure 11: Motion planning among dynamic obstacles is achieved by finding a path in the  $s, t$ -plane without intersecting the occupied cells. This results in an optimal speed profile for the mobile robot, resulting in a collision-free motion.

## Evaluation

This section evaluates the performance of the clothoidal LPT by comparing it with its predecessor, the circular LPT. Two different path planning scenarios will be presented (door and elevator), requiring the planner to plan a trajectory through a narrow opening. In order to objectively compare both LPTs, identical curvature constraints on the path are applied to the circular LPT and the clothoidal LPT. For the circular LPT, paths are generated starting from 500 discrete input velocity pairs  $(v, \omega)$  but paths yielding a curvature  $\kappa > \kappa_{max}$  are discarded. This results in a circular LPT composed of 250 trajectories (both forward and backward), shown in Figure 12. The clothoidal LPT used for this evaluation is the same as shown in Figure 5 and is composed of 1500 trajectories (forward and backward). Turning on the spot rotations were discarded during this evaluation.

### Path planning performance

The first benchmark consists of driving forward through a doorway; for the second benchmark, the wheelchair has to drive backwards into an elevator. Both benchmarks are inspired by real-world environments.

**Forward driving through a doorway** In this situation, the mobile robot has to drive through a doorway to exit a room. Figure 13 (top) shows a successful, collision-free path going through the doorway using both LPTs.

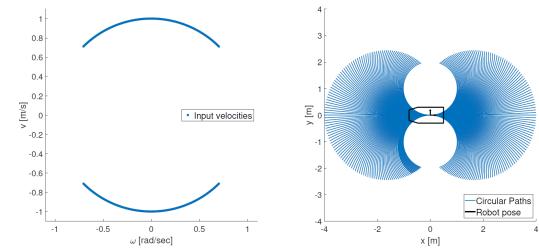


Figure 12: Input velocities (left) integrated over a period of  $t = 4$  s to obtain 250 circular paths complying with the kinematic constraints (right).

A uniform set of start poses are generated to assess the planning performance of both LPTs. This is shown in Figure 13 (bottom, left), where the test region (red polygon) contains the set of uniformly spaced start poses. If a path of the LPT originating from a start pose reaches the goal region (green polygon), that pose is defined as successful. Successful start poses are divided in three cases: (1) only the circular LPT achieved to plan a path reaching the goal area, (2) both LPTs achieved to plan a path reaching the goal area, (3) only the clothoidal LPT achieved to plan a path reaching the goal area.

The final outcome when following this procedure is shown in Figure 13 along with a histogram of the occurrence of the three different cases. The unique successful start poses based on circular, common and clothoidal LPTs are shown respectively in red, black and green. The width of the doorway from the corridor to the robot laboratory is 80 cm while the width of the wheelchair is 60 cm.

The following conclusions can be drawn:

- The majority of the paths from start poses at the lower end of the figure are achieved by the circular LPT. This is because those paths only require a circular arc to enter the doorway. Since the circular LPT is composed of a uniformly spread set of circular trajectories, this LPT is favoured, compared to the clothoidal LPT.
- From the moment the required trajectory is more complex (for example, Figure 13 (top, right) the clothoidal LPT is the only LPT able to plan a path.
- As per the histogram provided in Figure 13, the majority (87%) of the poses with a successful path are generated by the clothoidal LPT, which demonstrates that for this first benchmark, the lack of uniformly spread circular trajectories is not so crucial, as only 13% of the total amount of successful start poses are uniquely found by using the circular LPT.

**Backwards driving in an elevator** In this situation, the mobile robot must drive in reverse from a corridor into an elevator. The width of the elevator is 90 cm while the width of the wheelchair is 60 cm. Figure 14 (top) shows a successful, collision-free path going in reverse through the doorway of the elevator whilst using both LPTs.

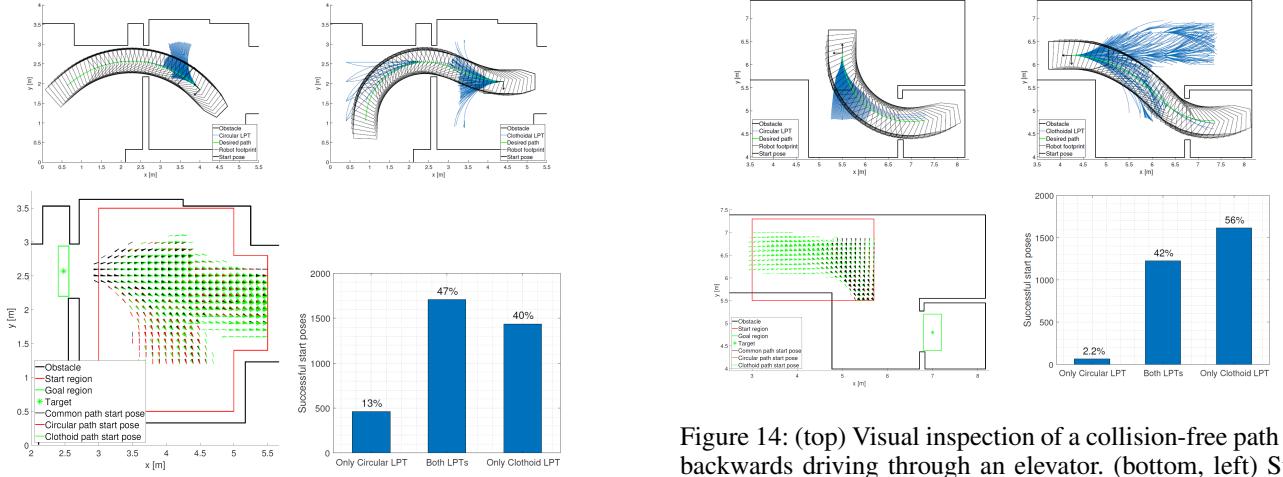


Figure 13: (top) Visual inspection of a collision-free path going through the doorway. (bottom, left) Successful start poses for each LPT finding a path through the doorway. (bottom, right) Histogram showing the outcome of the different cases of driving through the doorway. There are in total 3604 successful start poses, from which 1708 are common to both LPTs, whereas 460 are only from the circular LPT and 1436 from the clothoidal LPT.

The same procedure as provided in Figure 13 is applied for this benchmark, see Figure 14. It can be concluded that from the moment the start pose of the wheelchair is located further away in the corridor, only the clothoidal LPT manages to find a trajectory to the elevator. This is confirmed by the histogram shown in Figure 14 (bottom, right). Nearly all the start poses are successful when using the clothoidal LPT (98%).

### Time performance

The clothoidal LPT has six times more paths than the circular LPT. In order to verify that this does not come at a too high computation cost, the mobile robot is positioned in several poses in the environment as shown in Figure 14 (bottom, left). The presented benchmarks were run on a computer with an Intel Core<sup>TM</sup> i5-4460 quad core 3.20GHz CPU with 16GB of memory. Benchmark scripts were written in MATLAB. Figure 15 (left) shows a histogram of the execution time needed to adapt each path from the LPTs. The median execution time of the circular LPT is 29 ms compared to 114 ms for the clothoidal LPT; the clothoidal LPT is therefore 3.9 times slower. Figure 15 (right) shows the execution time of the path length adjustment over the number of occupied cells.

### Conclusions and future work

This paper proposed various improvements towards a circular local path template (LPT) that we have designed in the past to estimate driver intents and to provide navigation assistance in teleoperation or wheelchair applications. Using two performance criteria, path planning capability and

Figure 14: (top) Visual inspection of a collision-free path for backwards driving through an elevator. (bottom, left) Successful start poses for each LPT planning a path backwards into an elevator. (b) Histogram illustrating the outcome of the different cases for planning a path backwards into an elevator. There are in total 2904 successful start poses, from which 1224 are common to both LPTs. Whereas 64 are only from the circular LPT and 1616 from the clothoidal LPT.

time performance, the clothoidal LPT was compared with the existing circular LPT. The clothoidal LPT allows the use and tuning of more complex paths, resulting in an improved path planning performance. This has been demonstrated in a door and elevator scenario, which regularly require intense manoeuvring by wheelchair drivers. For this, the clothoidal LPT needed nearly four times more computing time to adjust each individual path length as compared to the circular LPT (median value of 114 ms compared to the 29 ms). Fu-

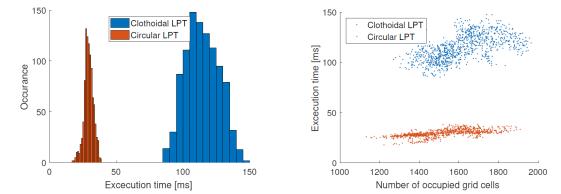


Figure 15: Execution time needed to adapt the path length of the circular and clothoidal LPT for a simulated environment. Different poses are used to obtain a varying number of triggered cells when an obstacle in the surroundings of the mobile robot matches one in the precomputed lookup table. (left) Histogram of the resulting execution time. (right) Computation time as a function of the number of occupied cells.

ture work will include an implementation in C/C++ of these algorithms and an optimisation of the position of the EPs to reduce the number of clothoidal paths. Furthermore, the use of this LPT for fast collision checking in a path planning algorithm will be investigated.

## References

- Baass, K. G. 1982. Use Of Clothoid Templates In Highway Design. *Transportation Association of Canada*.
- Bertolazzi, E., and Frego, M. 2015. G1 fitting with clothoids. *Mathematical Methods in the Applied Sciences* 38(5):881–897.
- Brezak, M., and Petrović, I. 2011. Path Smoothing Using Clothoids for Differential Drive Mobile Robots. *IFAC Proceedings Volumes* 44(1):1133–1138.
- Cope, G. H. 1993. *British Railway Track: Design, Construction and Maintenance*. Permanent Way Institution Loughborough,, UK.
- Demeester, E.; Hüntemann, A.; Vander Poorten, E.; and De Schutter, J. 2012a. ML, MAP and greedy POMDP shared control: Comparison of wheelchair navigation assistance for switch interfaces. In *International Symposium on Robotics*.
- Demeester, E.; Vander Poorten, E.; Philips, J.; and Hüntemann, A. 2012b. Design and evaluation of a lookup-table based collision-checking approach for fixed sets of mobile robot paths. In *International Symposium on Robotics*, 1045–1050.
- Fleury, S.; Soueres, P.; Laumond, J. P.; and Chatila, R. 1995. Primitives for smoothing mobile robot trajectories. *IEEE Transactions on Robotics and Automation* 11(3):441–448.
- Kelly, A., and Nagy, B. 2003. Reactive Nonholonomic Trajectory Generation via Parametric Optimal Control. *The International Journal of Robotics Research* 22(7-8):583–601.
- McCrae, J., and Singh, K. 2009. Sketching piecewise clothoid curves. *Computers & Graphics* 33(4):452–461.
- Mercy, T.; Loock, W. V.; and Pipeleers, G. 2016. Real-time motion planning in the presence of moving obstacles. In *2016 European Control Conference (ECC)*, 1586–1591.
- Pivtoraiko, M., and Kelly, A. 2012. Generating State Lattice Motion Primitives for Differentially Constrained Motion Planning. In *International Conference on Intelligent Robots and Systems*, 101–108.
- Scheuer, A., and Fraichard, T. 1997. Continuous-curvature path planning for car-like vehicles. In , *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems, 1997. IROS '97*, volume 2, 997–1003 vol.2.
- Vander Poorten, E.; Demeester, E.; Reekmans, E.; Philips, J.; Hüntemann, A.; and Schutter, J. D. 2012. Powered wheelchair navigation assistance through kinematically correct environmental haptic feedback. In *2012 IEEE International Conference on Robotics and Automation*, 3706–3712.
- Walton, D. J., and Meek, D. S. 2009. G1 interpolation with a single Cornu spiral segment. *Journal of Computational and Applied Mathematics* 223(1):86–96.