A Deeper Look into Autonomous Corridor Following and Doorway Passing on a Wheelchair

Vishnu Sashank Dorbala
UID 116907569
University of Maryland, College Park

Kulbir Singh Ahluwalia
UID 116836050
University of Maryland, College Park

CONTENTS

I. Abstract

Wheelchair mobility is a challenging task for users with mobility disorders. In several cases, even operating electric wheelchairs can prove difficult for the user. Thus, autonomy in performing certain low level tasks is a very desirable feature. In this report, we examine two tasks of Autonomous Corridor Following and Autonomous Doorway Passing on a wheelchair. The first task of Autonomous Corridor Following uses Vanishing Features of the corridor to perform visual servoing in a corridor. The approach uses a control law that takes in these features as an input to produce a velocity vector for servoing. The second task of Autonomous Doorway Passing makes use of a Lyapunov Based Controller design. Here, the task is modelled using an error function formulated from a geometric feature obtained from doorpost coordinates. The error is driven to zero using an angular velocity estimate obtained satisfying the Lyapunov Equation. We present the results of the original work as well as our simulations, and perform an analysis on both outcomes.

II. Introduction

Wheelchair users often find it hard to navigate through several environments. Even with developments in power/-electric wheelchair technology, maneuvering still requires a certain level of dexterity to move about easily. Users also require a good degree of concentration for them to be able to navigate safely through an environment, which can often be very difficult and mentally demanding, depending on the disability.

In order to tackle these issues, the authors of the paper device an *"Intelligent Wheelchair"* [1] that retrofits an ordinary power wheelchair with a vision sensor and actuator controls. This empowers the wheelchair to perform some key low level tasks autonomously.

Recent advances in robotics have paved the way for a number of opportunities to promote the development of smart devices that improve the lives of people with disabilities. There has been extensive work done in this field, focusing primarily on the issue of raising wheelchair safety rate (and thus reducing the amount of human intervention). For example, the TAO Project [2], NavChair [3], European FP7 Radhar project [4], the SYSIASS project [5] and even the recent SARA [6]. The user gives a high level input (such as a voice command asking to take the person somewhere) while the wheelchair is responsible for the activities of the lower level. There has also been research on using Brain Controlled Interfaces (BCI) for maneuvering wheelchairs [7], [8].

Our inspiration for documenting a report of this work comes from the idea that Computer Vision systems with appropriate control algorithms can have a tremendous impact on these healthcare systems.

The authors of the paper focus on two key low level tasks that make the wheelchair autonomous. Firstly, they describe a method for performing autonomous corridor following using visual inputs as vanishing points of the corridor. These vanishing points are estimated by adding constraints on lines extracted from images. The point of intersection of a large portion of lines in the image will describe the vanishing point, which is a *visual feature* passed through a control law to perform servoing.

The second task of *Autonomous Doorway Passing* is modelled using coordinate outputs from a doorpost detection algorithm. Although a non-trivial task due to several high level details needed (Like is the door closed/open, is it the right door?), it can be achieved at a lower level. The pixel coordinates of the doorposts detected are utilized for this task. These coordinates are used to compute geometric features which are used to formulate a Lyapunov Based Controller for achieving the desired task. The authors use this approach since global asymptotic stability [9] of the system can be demonstrated.

This report is divided into the following parts: Section III discusses how the authors modify their wheelchair, and the modelling aspects in terms of the robot and camera frames chosen. In section IV, we give a brief overview of Visual Servoing, and understand the controller used for performing the first task of Autonomous Corridor Following. Section V gives a general overview and introduction to Lyapunov Functions and their usefulness in finding internal stability of systems. Section VI describes their usage of the Lyapunov function for designing a controller for the autonomous doorway navigation task. Section VII analyses the outcome of the controller described in the previous section through graphs obtained via simulation.

III. Modelling

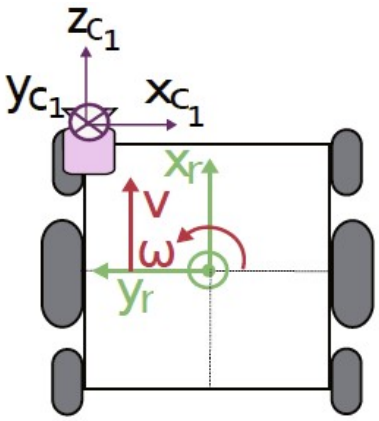

Fig. 1: Top view of the wheelchair Robot with Frame 1

A wheelchair with two differentially actuated wheels along with four passive castor wheels is chosen by the authors. It moves on a horizontal or inclined plane and is modelled as a unicycle robot with non holonomic constraints [10]. We have two control variables which are the translational velocity "v" and the rotational velocity "ω". A constant forward velocity v^* is maintained for the wheelchair.

Three cameras are installed on the wheelchair namely, Camera 1, 2 and 3. The front facing Camera 1 is located on the left handle at a height of h1 while Camera 2 and 3 are located on

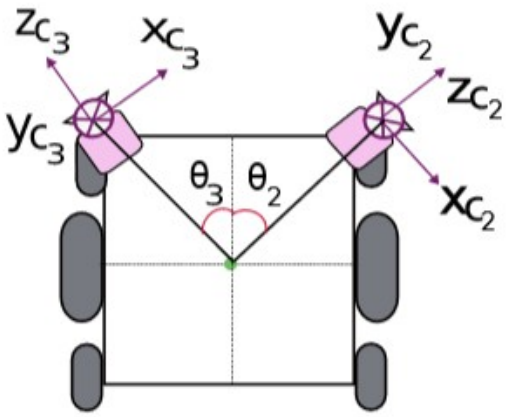

Fig. 2: Top view with Frame 2 and 3

the left and right handles at heights of h2 and h3 respectively. All heights are measured from the floor level as seen in figure 3.

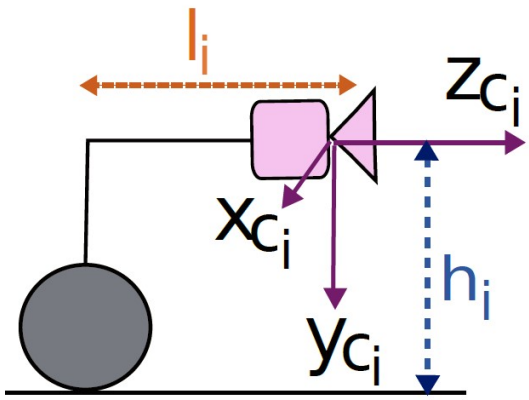

Fig. 3: Side view of camera C_i

Camera 1 is used for corridor following while Cameras 2 and 3 are used for doorway passing for doors on the left and right walls. Cameras 2 and 3 are oriented towards their respective walls to ensure that potential doors are always in the camera's field of view.

In this study, we focus on the right hand side camera 2 as the left hand side camera 3 can be modelled symmetrically. The robot frame is defined as $F_r(P_o, x_r, y_r, z_r)$ with the origin of frame F_r at the mid-point of the line segment joining the two centres of the differential wheels as shown in figure 1.

We define the camera frames as $F_{c_i}(C_i, x_{c_i}, y_{c_i}, z_{c_i})$ where i = 1,2,3 and c_i denotes the optical center of each camera. Also, camera 2 and 3 make angles of θ_2 and θ_3 with x axis of the robot frame as seen in figure 2.

In order to obtain the rotation matrix for rotating from robot frame to the camera frame C_2, we use: -

$$R_r^{c_2} = \begin{bmatrix} x_r.x_{c_2} & y_r.x_{c_2} & z_r.x_{c_2} \\ x_r.y_{c_2} & y_r.y_{c_2} & z_r.y_{c_2} \\ x_r.z_{c_2} & y_r.z_{c_2} & z_r.z_{c_2} \end{bmatrix}$$

Which is equal to: -

$$R_r^{c_2} = \begin{bmatrix} -sin(\theta_i) & -cos(\theta_i) & 0 \\ 0 & 0 & -1 \\ cos(\theta_i) & -sin(\theta_i) & 0 \end{bmatrix}$$

The translation from robot frame to camera frame (with respect to robot frame) represented by $t_{c_1}^r$ is given by $(l_i, w_i, 0)$ which can also be seen in figure 3. For the front facing camera 1, we have $\theta_1 = 0$, $l_1 > 0$ and $w_1 > 0$ whereas camera 2 on the right hand side has $\theta_2 < 0$, $l_2 > 0$ and $w_2 < 0$ which gives us: -

$$R_r^{c_2} = \begin{bmatrix} sin(\theta_i) & -cos(\theta_i) & 0 \\ 0 & 0 & -1 \\ cos(\theta_i) & sin(\theta_i) & 0 \end{bmatrix}$$

Using the notation v^f = (v,ω) = $(v_x^f, v_y^f, v_z^f, \omega_x^f, \omega_y^f, \omega_z^f)$ to represent the x, y and z components of the linear and angular velocities with respect to a frame f, we can write: -

$$v^r = (v, 0, 0, 0, 0, \omega)$$

Using a similarity transform on the Transformation matrix, we obtain the following well studied equation:

$$v^{c_i} = \begin{bmatrix} R_r^{c_i} & [t_r^{c_i}] \times R_r^{c_i} \\ 0 & R_r^{c_i} \end{bmatrix} . v^r$$

Using the above formula for frame 2,

$$v^{c_2} = \begin{bmatrix} R_r^{c_2} & [t_r^{c_2}] \times R_r^{c_2} \\ 0 & R_r^{c_2} \end{bmatrix} . v^r$$

Substituting the value of $R_r^{c_2}$ and $t_{c_1}^r$ we get: -

$$v^r = (v_x^{c_i}, 0, v_z^{c_i}, 0, \omega_y^{c_i}, 0)$$

where

$$v_x^{c_i} = v \sin\theta_i - \omega(l_i \cos\theta_i + w_i \sin\theta_i) \quad (1)$$

$$v_z^{c_i} = v \cos\theta_i + \omega(l_i \sin\theta_i - w_i \cos\theta_i) \quad (2)$$

$$\omega_y^{c_i} = -\omega \quad (3)$$

IV. Controller Design for Autonomous Corridor Following

In this section, we give a brief overview of Visual Servoing and explain the method used by the authors to perform the task of Autonomous Corridor Following.

A. Visual Servoing

Visual Servoing or Vision based control refers to the study of how vision can be used to enable robot control. For example, consider a robot arm in a manufacturing plant with a camera mounted either on it or fixed somewhere else with the arm in view. A visual servoing procedure describes how images obtained from the camera can be utilized to move the arm to perform a particular task, which in this case might be picking up and placing an object on the conveyor belt.

A great tutorial on the fundamentals of visual servoing is

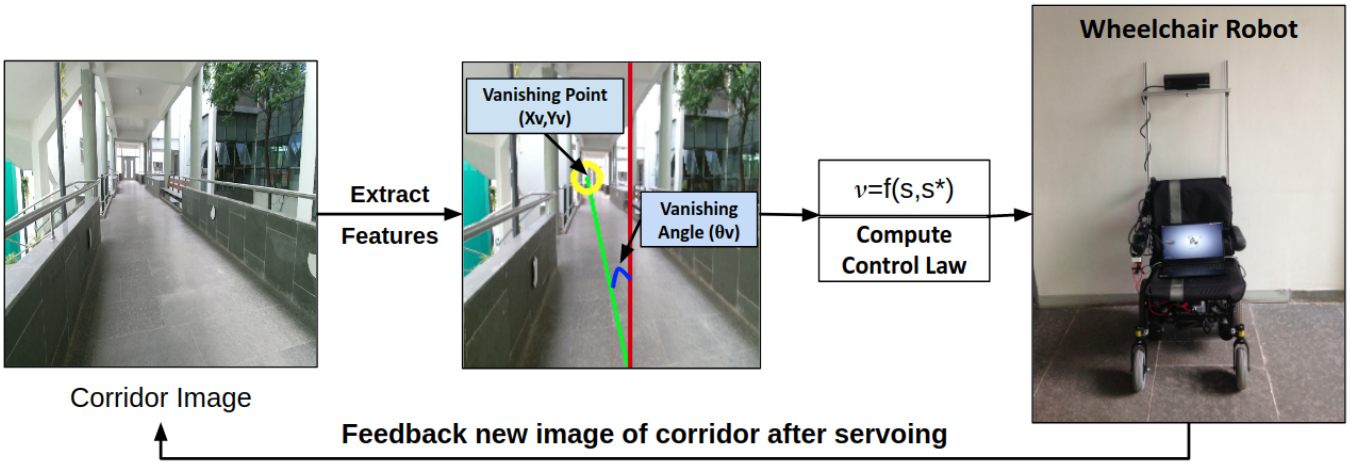

Fig. 4: Overview of the Autonomous Corridor Following Approach proposed by the authors.

provided in [11]. Visual servoing takes place iteratively in the following steps:

In general, Visual servoing can broadly be classified into two types:

- **Image Based Visual Servoing (IBVS)**: In this type servoing, the camera is in motion and is in a eye-in-hand configuration. Desired feature points (say s^*) and current feature points (say s) are chosen in the image frame. The desired features are chosen as the features on the image plane when the robot is at the target position. The current features are features that are extracted from the image frame at the current position of the robot. IBVS essentially tries to find a velocity vector that can minimize the error $s^* - s$.
  A simple example here is an automated landing mechanism in flights. A camera present near the front wheels of the airplane monitors the landing strip in front of it, and accordingly performs IBVS by sending commands to position the plane at the center of the strip.
  A major advantage in this form of servoing is that we do not require any external 3D information about the environment. The servoing takes place implicitly while reducing the pixel error.
- **Pose Based Visual Servoing (PBVS)**: In this configuration, a static camera is placed such that it has the robot which needs to be servoed in its field of view. Then servoing takes place based on the captured image frames of the camera and some environmental constraints that define the 3D geometry of the objects being servoed.
  An example here would be to control a robot arm, given only an image from say a surveillance camera placed somewhere above it. Here, explicit 3D information regarding the position of the arm w.r.t. the camera is necessary to perform servoing.

Apart from IBVS and PBVS, there also exist several *hybrid* methods which try to combine the outcomes of both these controllers, based on the task given. For the autonomous corridor following task, IBVS is performed.

B. Autonomous Corridor Following

The authors perform Autonomous Corridor Following using visual features extracted from a hallway. Figure 4 shows a flowchart of the approach followed.
The first step involves computing Vanishing Point Features. The x_v coordinate of the vanishing point, and the angle θ_v that the vanishing line along the corridor makes with the perpendicular are chosen as the visual servoing inputs. These features are computed using classical techniques in computer vision for extracting lines and points from an image. In this paper, the authors use a Line Segment Detector algorithm [12], [13] to compute line segments from the corridor image captured at each step. They then add constraints to these line segments obtained to get values for x_v and θ_v.

After obtaining x_v and θ_v, a controller is designed to servo the wheelchair along the center of the corridor. As the wheelchair is a 2DOF non-holonomic system, only a linear translational velocity v and an angular velocity ω needs to be obtained. The paper assumes a constant linear velocity v as it is sufficient for the task. This is given by a motion exigency condition as follows:

$$v = v^* = const. > 0 \tag{4}$$

ω applies a non linear feedback to the system based on features that it extracts from the image. This feedback is entirely dependant on the error computed between the desired image features $s^* = (0,0)$ and current features $s = (x_v, \theta_v)$, and is hence non-linear.

The state dynamics of our system can be given by:

$$\dot{s} = J(s)u = J_v(s)v + J_\omega(s)\omega \tag{5}$$

Here, $J(s)$ represents a Jacobian that relates the input image features u to $\dot{s}$. J_v is the linear velocity Jacobian, and J_ω is

the angular velocity Jacobian. Applying the motion exigency condition, this equation becomes.

$$\dot{s} = J(s)u = J_v(s)v^* + J_\omega(s)\omega \tag{6}$$

From this, we can get ω for control as

$$\omega = -J_\omega^+(\lambda e + J_v v^*) \tag{7}$$

This is the equation used for obtaining the non-linear angular velocity of the wheelchair. J_ω^+ is the Moore-Penrose pseudo inverse of the Jacobian of the angular velocity ω. The error e is defined as the difference between the extracted features and the desired feature values.

$$e = (x_v, \theta_v) - (0,0) \tag{4}$$

λ and v^* are gain and translational velocity constants.

V. Lyapunov Functions and Conditions for Stability

The authors use a Lyapunov Based Controller to perform the *Autonomous Doorway Passing* task. In this section, we explore some general information and background about Lyapunov Controllers, and understand how they are used to find if systems are stable or not.

A. Definition of Stability

A system is said to be stable if for a bounded input, it produces a bounded output. Figure 5 is an example of a stable system. At any point in it's graph from $t = 0 \rightarrow \infty$, the outcome $c(t)$ is always bounded between 0 and 1.

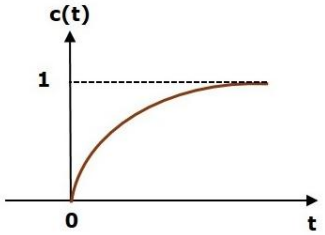

Fig. 5: A stable system

Stability can be categorized into 3 different types:

- **Absolutely Stable System** :
  These systems are stable over the entire range of input component values.
- **Conditionally Stable System** :
  These systems are stable only over a specific range of input values.
- **Marginally Stable System**:
  These systems achieve have a stable output, given a constant input value. The outputs are neither asymptotically stable nor unstable, i.e., they do not blow up, neither do they tend to go to zero.

In this paper, we deal with designing a controller to achieve an absolutely stable system.

B. Lyapunov Functions and Stability Criteria

A *Lyapunov Function* can be thought of as a generalized energy function of a system. The energy represents the error or entropy that the system contains. The lesser the entropy, the more stable the system is as it does not exhibit any random behaviour.

We need to define this function such that it dissipates all the energy as soon as possible to reach an error value of zero. In general, we would prefer to select a Lyapunov Function which shows ***marginal***, ***asymptotic*** and ***exponential*** stability. These terms can be defined as follows.

- **Asymptotic**: A system is called asymptotically stable around its equilibrium point at the origin if it satisfies the following two conditions:
  1) Given any $\varepsilon > 0$, $\exists\ \delta_1 > 0$ such that if $\|x(t_0)\| < \delta_1$, then $\|x(t)\| < \varepsilon, \forall t > t_0$
  2) $\exists\ \delta_2 > 0$ such that if $\|x(t_0)\| < \delta_2$, then $x(t) \rightarrow 0$ as $t \rightarrow \infty$

  This gives us a condition that the output will definitely reach zero at some time in the future.
- **Marginal**: A system is said to be marginally stable, if for a given stable input of a finite magnitude, its output will neither tend towards ∞ nor will it give a zero value. For defining a Lyapunov function, this is important as we would always prefer a function that will not exponentially blow up at certain values of error.
- **Exponential**: We call a system exponentially stable if it follows an exponential curve in its descent. The slope of the exponential function is such that it's descent happens very quickly, which is often a desirable outcome. This is because in most systems, we would want the error to be minimized as quickly as possible.
  However in our case, having a quick descent in the error function might not be preferable due to motion constraints of the wheelchair.

A good Lyapunov function should demonstrate all of these stability criteria. If $x = 0$ is the zero solution of a system, there is a Lyapunov function $V(x)$, then the equilibrium point x = 0 of the system is Lyapunov stable.

The derivative of a Lyapunov Function can determine its stability. Consider a Lyapunov Function $V(x)$ and its derivative $\dot{V(x)}$. The Lyapunov Stability Criterion states that if $V(x)$ satisfies the following criteria, it is stable:

- $V(x) = 0$ if and only if $x = 0$,
- $V(x) = 0$ if and only if $x \neq 0$, and
- $\dot{V}(x) = \frac{d}{dt}V(x) \leq 0,\ \forall\ x \neq 0$

Satisfying these conditions assures stability of the Lyapunov Function.

C. Intuitively Understanding Lyapunov Stability

Given a desired outcome ($\hat{y}$), and an error function the describes the shift of the actual outcome (y) from this, the Lyapunov function essentially gives us a condition that can be used to drive this error to zero.

For example, consider an error function $E = (\hat{y} - y) = x^2$. In order to prove its stability, we look at its differential w.r.t. which is $\dot{y} = 2x$.
Now $\dot{E} \leq 0$ (The condition for Lyuapunov Stability) is true when $x \in (-\infty, 0]$. Hence, this is the range for which the equations $y = x^2$ is stable.

Figure 6 shows y and $\dot{y}$ for reference. Observing the output of the system $y = x^2$ between $(-\infty, 0]$, we can see that the error equation is driven to zero in this range. This is a desirable characteristic for the stability of a Lyapunov function.

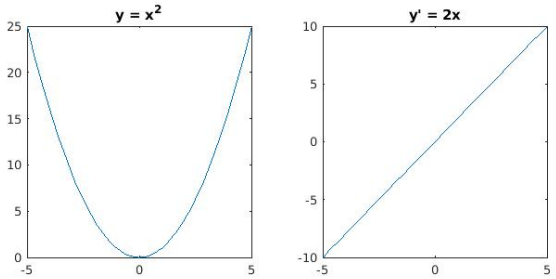

Fig. 6: Observe that $\dot{y} \leq 0$ when $x \in (-\infty, 0]$. For these values, the original error equation y is driven to zero.

D. The Lyapunov Equation

For a linear system $\dot{x} = Ax + Bu$, consider the Lyapunov function as $V(z) = x^T P x$, where x is the input state, and P is a positive semi-definite matrix. P here has to be positive, because we consider $V(z)$ as a quadratic energy equation.
As Lyapunov transforms are used for internal system stability, we consider $u = 0$.

To check for stability, we differentiate this function to get,

$$\dot{V}(z) = (Az)^T Pz + z^T P(Az) \tag{8}$$

For the linear system to drive itself to to zero, we need a negative slope on this function, which gives us,

$$\dot{V}(z) = A^T P + AP = -Q, \tag{9}$$

$$or\ \dot{V}(z) = A^T P + AP + Q = 0 \tag{10}$$

where Q is a representation of the slope of the system.

This equation is called the Lyapunov Equation. where A, P, $Q \in \mathbf{R}^{n \times n}$, and P, Q are symmetric.

VI. Lyapunov Based Controller for Autonomous Doorway Passing

In this section, we state the problem of Autonomous Doorway Passing, and a Lyapunov based control strategy for achieving this task. Figure 7 shows a process flow of the approach used.

A. Defining the Problem

The task of *Autonomous Doorway Passing* involves two parts. The first part involves detecting doorposts using a 2-D edge tracking algorithm [14], [15]. An example outcome of this algorithm is shown on hallway doors being detected in figure 8. The bottom point of the nearest doorpost detected by the algorithm is used as a servoing input to the task.
The second part involves using the detected doorpost to formulate a Lyapunov based controller for achieving the doorway passing task.

B. Defining Controller Parameters

A geometrical representation of the doorway passing task is represented in figure 11. Here, $D = (x_d, h_2, z_d)$ represent the selected doorpost point. In polar coordinates, D can be represented as,

$$r = \sqrt{x_d^2 + z_d^2} \tag{11}$$

$$\phi_d = \arctan \frac{x_d}{z_d} \tag{12}$$

Now, in the image plane, consider the point D in 3D as a point P in 2D represented as (x_p, y_p). By applying some basic 3D geometry, we get the projections of the 3D point on the camera image plane as,

$$x_p = \frac{x_d}{z_d} \tag{13}$$

and

$$y_p = \frac{y_d}{z_d} \tag{14}$$

Substituting this in the equation 12, we get

$$\phi_d = \arctan(x_p) \tag{15}$$

From the figure 11, we can also conclude that,

$$\cos \phi_d = \frac{z_d}{r}. \tag{16}$$

which can be substituted in y_p to finally get,

$$r = \frac{h_2}{y_p . \cos \phi_d} \tag{17}$$

C. The Lyapunov Controller

ϕ_d^* represents the desired angle that the camera frame makes with the doorpost. It keeps varying with time as the wheelchair gets closer to the door. The actual angle that the camera makes with the doorpost is defined as ϕ_d. Our aim is to minimize ϕ_d - ϕ_d^*. In the following section, we look upon the desired trajectory.

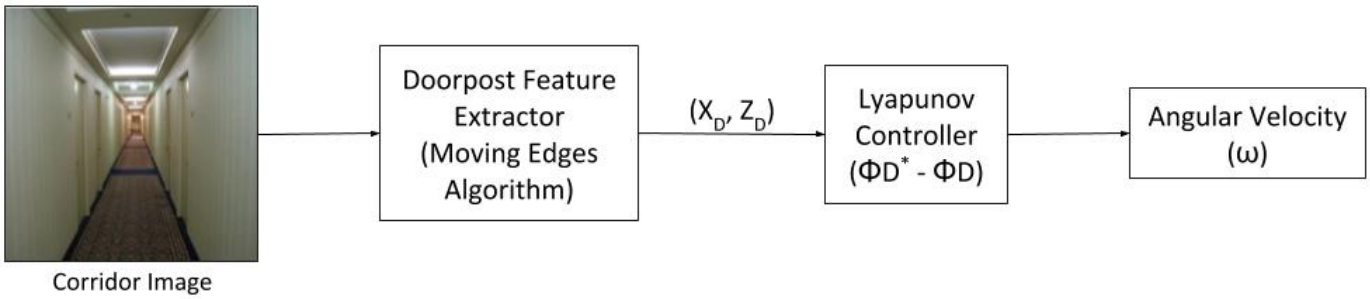

Fig. 7: Flowchart showing the process flow of the corridor following task. Observe that the X_D and the Z_D coordinates of the Doorpost come from the image directly, are not directly dependant on the output ω value. ω however drives the wheelchair to a new position during each iteration, which in turn gives a new image for feature extraction. This feature extraction need not always be reliable, and might lead to unexpected spikes in the trajectory.

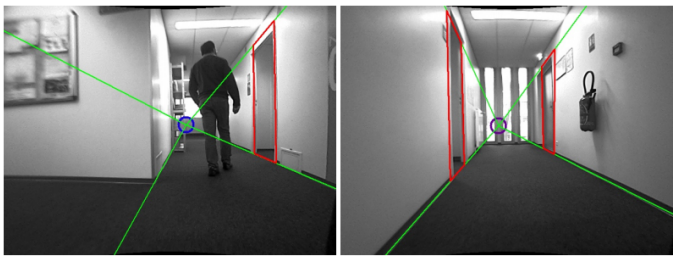

Fig. 8: The Moving Edges algorithm. The red border around each door represent the detected edges. The green lines are vanishing lines and the blue circle represents the vanishing point (VP), which are used to achieve the other task of *Autonomous Corridor Following*.

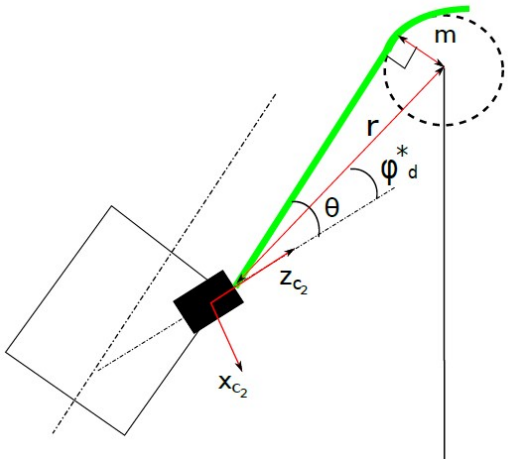

Fig. 9: Desired camera trajectory when r > m

1) Desired trajectory: In order to avoid collisions with the doorpost or the walls, we chose a tolerance "m" which is defined as the radius of an imaginary circle centered at the doorpost as seen in figure 9. When r is greater than m, the wheelchair follows a tangential path towards the imaginary circle. On the other hand, when r is less than or equal to m, it follows the circular trajectory of the circle with radius "m" as seen in figure 10.

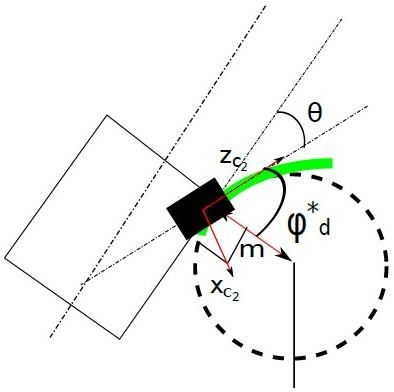

Fig. 10: Desired camera trajectory when r ≤ m

We see that: -

$$\theta - \phi_d = \arcsin\left(\frac{m}{r}\right) \tag{18}$$

Since θ and θ_2 have the same values geometrically but they are defined in the camera frame and robot frame respectively, we have $\theta = -\theta_2$.

Therefore,

$$\phi_d^* = \theta_2 + \arcsin\left(\frac{m}{r}\right) \quad \text{if} \quad r > m \tag{19}$$

When r ≤ m, the wheelchair follows a circular trajectory and thus: -

$$\phi_d^* = \frac{\pi}{2} - \theta \quad \text{if} \quad r \leq m \tag{20}$$

As $\theta = -\theta_2$, we have: -

$$\phi_d^* = \frac{\pi}{2} + \theta_2 \quad \text{if} \quad r \leq m \tag{21}$$

We see that when $r \leq m$: -

$$\dot{\phi_d^*} = 0 \tag{22}$$

This will be used later to find $\dot{V}$ which is the derivative of the Lyapunov function "V".

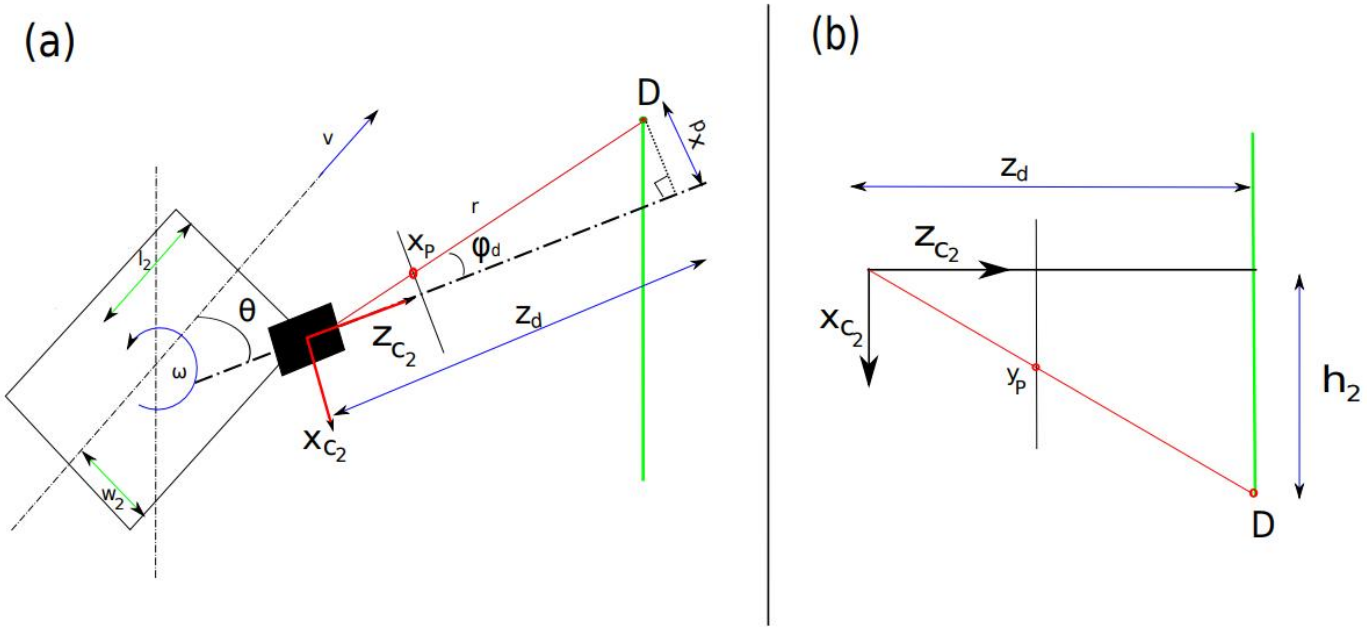

Fig. 11: Top View and Front View of the doorway passing task described by geometry.

2) Control law formulation: In order to design a globally asymptotically stable system, we choose the Lyapunov candidate function as: -

$$V = \frac{1}{2}\left(\phi_d - \phi_d^*\right)^2 \tag{23}$$

Now we require $\dot{V} < 0$ and thus taking the derivative of V we have,

$$\dot{V} = \left(\phi_d - \phi_d^*\right)\left(\dot{\phi}_d - \dot{\phi}_d^*\right) \tag{24}$$

Using equation 12, we have: -

$$\dot{\phi}_d = \frac{z_d \dot{x}_d - x_d \dot{z}_d}{x_d^2 + z_d^2} \tag{25}$$

From the well known kinematics equation $\dot{\mathbf{x}} = -v - [\omega]_\times \mathbf{x}$, we deduce $\dot{x}_d$ and $\dot{z}_d$ as

$$\begin{cases} \dot{x}_d = -v_x^{c_2} - z_d^{c_2}\omega_y \\ \dot{z}_d = -v_z^{c_2} + x_d^{c_2}\omega_y \end{cases} \tag{26}$$

Using equation 1, 2 and 3, we get:-

$$\dot{x}_d = (-\sin\theta_2)v + \omega\left(l_2\cos\theta_2 + w_2\sin\theta_2\right) + z_d\omega \tag{27}$$

$$\dot{z}_d = (-\cos\theta_2)v - \omega\left(l_2\sin\theta_2 - w_2\cos\theta_2\right) - x_d\omega \tag{28}$$

Since $x_d = r\sin\phi_d$ and $z_d = r\cos\phi_d$, we obtain,

$$\dot{\phi}_d = \frac{r\cos\phi_d\left[-\sin\theta_2 v + \omega\left(l_2\cos\theta_2 + w_2\sin\theta_2\right) + r\cos\phi_d\omega\right] - r\sin\phi_d\left[-\cos\theta_2 v - \omega\left(l_2\sin\theta_2 - w_2\cos\theta_2\right) - r\sin\phi_d\omega\right]}{r^2} \tag{29}$$

which can be written as: -

$$\dot{\phi}_d = \frac{1}{r}\left[\left(v\left(-\cos\phi_d\sin\theta_2 + \sin\phi_d\cos\left(\theta_2\right)\right)\right] + \frac{1}{r}\left[\omega\left(l_2\left(\cos\phi_d\cos\theta_2 + \sin\phi_d\sin\theta_2\right) + w_2\left(\cos\phi_d\sin\theta_2 - \sin\phi_d\cos\theta_2\right)\right) + r\omega\right)\right] \tag{30}$$

Using trigonometric properties we get,

$$\dot{\phi}_d = \omega + \frac{1}{r}\left[\sin\left(\phi_d - \theta_2\right)v + \left(l_2\cos\left(\phi_d - \theta_2\right) - w_2\sin\left(\phi_d - \theta_2\right)\right)\omega\right] \tag{31}$$

Using equation 19 in the case of r greater than m we get:-

$$\dot{\phi}_d^* = \frac{d}{dt}\left(\arcsin\left(\frac{m}{r}\right)\right) = \frac{-\dot{r}m}{r\sqrt{r^2-m^2}} \tag{32}$$

Using trigonometric properties we get,

$$\dot{\phi}_d = \omega + \frac{1}{r}\left[\sin\left(\phi_d-\theta_2\right)v + \left(l_2\cos\left(\phi_d-\theta_2\right) - w_2\sin\left(\phi_d-\theta_2\right)\right)\omega\right] \tag{33}$$

Similarly to obtain $\dot{r}$ in order to derive $\dot{\phi}_d^*$ in equation 32, we deduce from equation 11:-

$$\dot{r} = \frac{x_d\dot{x}_d + z_d\dot{z}_d}{\sqrt{x_d^2+z_d^2}} \tag{34}$$

Therefore using equations 27 and 28 we get:-

$$\dot{r} = \frac{r\sin\phi_d\left[-\sin\theta_2 v + \omega\left(l_2\cos\theta_2 + w_2\sin\theta_2\right) + r\cos\phi_d\omega\right] + r\cos\phi_d\left[-\cos\theta_2 v - \omega\left(l_2\sin\theta_2 - w_2\cos\theta_2\right) - r\sin\phi_d\omega\right]}{r} \tag{35}$$

As the "r" cancels out from the numerator and the denominator, we get:-

$$\dot{r} = -v\left(\sin\phi_d\sin\theta_2 + \cos\phi_d\cos\theta_2\right) + \omega\left[l_2\left(\sin\phi_d\cos\theta_2 - \cos\phi_d\sin\theta_2\right) + w_2\left(\sin\phi_d\sin\theta_2 + \cos\phi_d\cos\theta_2\right)\right] \tag{36}$$

Using the trigonometric identities, we get :-

$$\dot{r} = -\cos\left(\phi_d-\theta_2\right)v + \left(l_2\sin\left(\phi_d-\theta_2\right) + w_2\cos\left(\phi_d-\theta_2\right)\right)\omega \tag{37}$$

We can write ϕ_d^* using equation 32 as:-

$$\dot{\phi}_d^* = \frac{m\left[\cos\left(\phi_d-\theta_2\right)v - \left(l_2\sin\left(\phi_d-\theta_2\right) + w_2\cos\left(\phi_d-\theta_2\right)\right)\omega\right]}{r\sqrt{r^2-m^2}} \tag{38}$$

Using the substitutions $\frac{m}{r} = \sin(\theta_d^* - \theta_2)$ and $\frac{\sqrt{r^2-m^2}}{r} = \cos(\phi_d^* - \theta_2)$ in the equation 19,

$$\dot{\phi}_d^* = \frac{\sin\left(\phi_d^*-\theta_2\right)\left[\cos\left(\phi_d-\theta_2\right)v - \left(l_2\sin\left(\phi_d-\theta_2\right) + w_2\cos\left(\phi_d-\theta_2\right)\right)\omega\right]}{r\cos\left(\phi_d^*-\theta_2\right)} \tag{39}$$

Using equations 31 and 39, we can write $\dot{\phi}_d - \dot{\phi}_d^*$ as:-

$$= \omega + \frac{1}{r}\left[\sin\left(\phi_d-\theta_2\right)v + \left(l_2\cos\left(\phi_d-\theta_2\right) - w_2\sin\left(\phi_d-\theta_2\right)\right)\omega\right] - \frac{\sin\left(\phi_d^*-\theta_2\right)\left[\cos\left(\phi_d-\theta_2\right)v - \left(l_2\sin\left(\phi_d-\theta_2\right) + w_2\cos\left(\phi_d-\theta_2\right)\right)\omega\right]}{r\cos\left(\phi_d^*-\theta_2\right)} \tag{40}$$

Finally, we can rewrite the equation as:-

$$\dot{\phi}_d - \dot{\phi}_d^* = v\left[\frac{\sin\left(\phi_d-\phi_d^*\right)}{r\cos\left(\phi_d^*-\theta_2\right)}\right] + \omega\left[1 + \frac{l_2\cos\left(\phi_d-\phi_d^*\right) - w_2\sin\left(\phi_d-\phi_d^*\right)}{r\cos\left(\phi_d^*-\theta_2\right)}\right] \tag{41}$$

Equation 41 is used to find $\dot{V}$. Rewriting equation 24:-

$$\dot{V} = (\phi_d - \phi_d^*)\left(\dot{\phi}_d - \dot{\phi}_d^*\right) \tag{42}$$

Substituting the values of $\dot{\phi}_d$ and $\dot{\phi}_d^*$, we obtain:-

$$\dot{V} = (\phi_d - \phi_d^*)\left(vA(r,\phi_d) + \omega(1 + B(r,\phi_d))\right) \tag{43}$$

where A and B have the values as shown in the following cases. Case I: When r > m,

$$\begin{cases} A(r,\phi_d) = \left(\dfrac{\sin(\phi_d - \theta_2)}{r} + m\dfrac{\cos(\phi_d - \theta_2)}{r\sqrt{r^2 - m^2}}\right) \\ B(r,\phi_d) = \dfrac{l_2\cos(\phi_d - \theta_2) - w_2\sin(\phi_d - \theta_2)}{r} \\ \qquad -m\dfrac{(l_2\sin(\phi_d - \theta_2) + w_2\cos(\phi_d - \theta_2))}{r\sqrt{r^2 - m^2}} \end{cases} \tag{44}$$

Case II: When r $\leq$ m, we have according to equation 22, $\dot{\phi}_d^* = 0$ and hence substituting this value in $\dot{V}$ we get:-

$$\begin{cases} A(r,\phi_d) = \left(\dfrac{\sin(\phi_d - \theta_2)}{r}\right) \\ B(r,\phi_d) = \dfrac{l_2\cos(\phi_d - \theta_2) - w_2\sin(\phi_d - \theta_2)}{r} \end{cases} \tag{45}$$

Thus, if we choose ω equal to:-

$$\omega = \frac{-k\left(\phi_d - \phi_d^*\right) - A(r,\phi_d)v^*}{1 + B(r,\phi_d)} \tag{46}$$

Then we have $\dot{V} < 0$ provided that k is taken to be a positive gain factor. Consequently, the system is guaranteed to be globally asymptotically stable and the visual feature ϕ_d converges asymptotically to the desired value ϕ_d^*. When $r > m$, we can see from the equations that $\omega = 0$ when $\phi_d = \phi_d^*$. When the tangential motion is finished, and the circular motion begins, we can see that $A = \frac{1}{r}$ and $B = -\frac{w2}{r}$, from which we get $\frac{-v^*}{r-w2}$. This corresponds to a circular motion when $r = m$ of radius $m - w2$.
The ω is an angular velocity feedback which is fed into the wheelchair in every loop.

VII. Discussion and Analysis of Simulation Outputs

In this section we discuss the outcomes of the original paper, some of our own simulations, and try to analyze the difference in the outcome.

A. Original Results

The results of the original paper are presented in figure 12. The wheelchair follows a straight line trajectory till $r = m$, and then performs a switch that causes it to change to a circular trajectory.

The graphs clearly indicate that the wheelchair moves gradually towards the doorpost. Once $r = 0.2m$ which is the defined margin (m), a switching motion begins to take place, which can be observed in the ω values in the bottom right graph. The angular velocity changes from 0 to -0.5 at this point, and shows that the wheelchair begins to rotate at one point.

We can also observe that the error $\phi_d^* - \phi_d$ value changes to zero very quickly and remains the same throughout except for a minor aberration during the switch. This shows the robustness of the controller they have designed, as the current and desired trajectories correlate very well.

B. Analysing Our Simulations

As we do not have a wheelchair with a camera mounted on it, we cannot physically realize the system shown in the paper. From the flowchart in figure 7, as we do not have real world corridor image sequences, we do not have real time

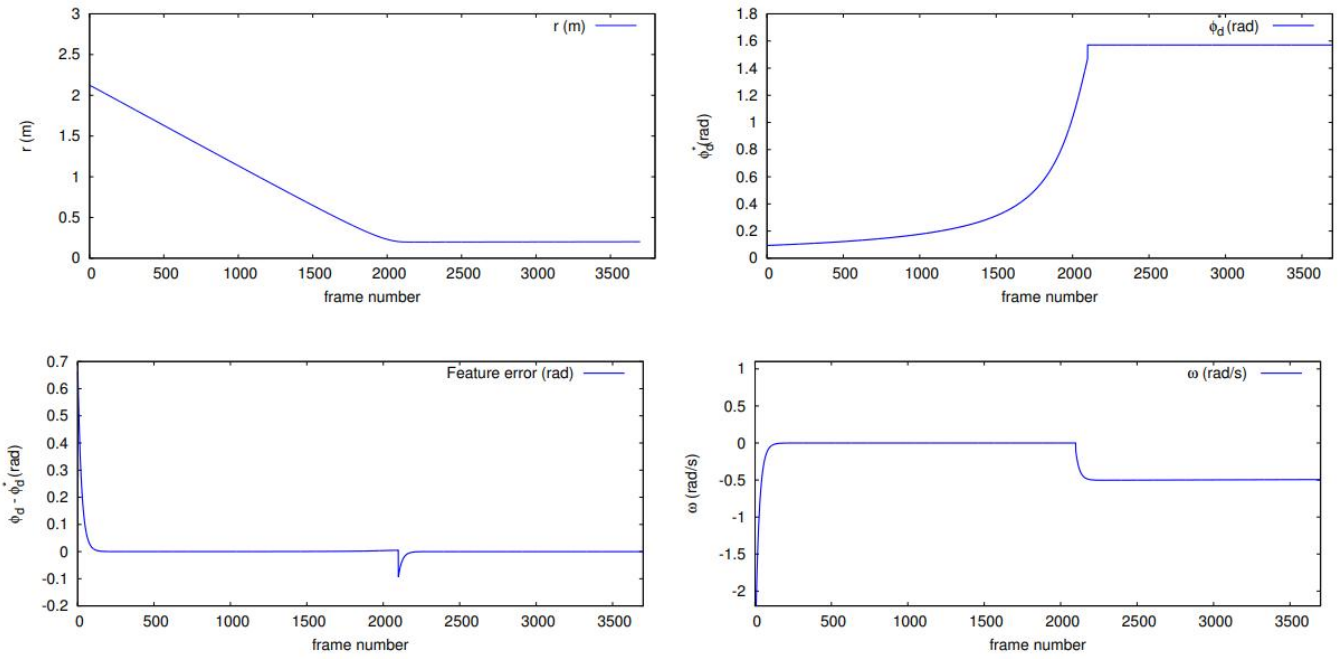

Fig. 12: The results from the original paper. Observe that the ω output remains zero until the switching motion of the Lyapunov function begins to take place (when $r \leq 0$). Once the switching motion takes place, the ϕ_d^* and r remain constant. This is because the wheelchair now performs a circular motion about the center of the doorway, positioning itself in front of the door.

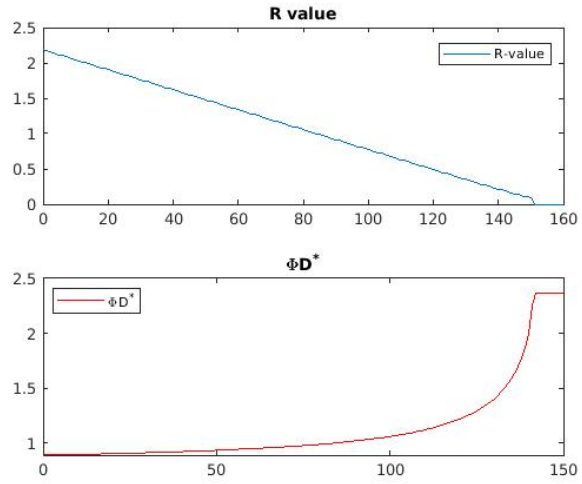

Fig. 13: The output R and desired ϕ values

values of X_D and Z_D. Hence, we try to give a range of values with for X_D and Z_D that try to emulate the pixel coordinates of the doorpost over a sequence of images of the autonomous doorway passing task.

We run our implementation on MATLAB (Code attached below). We run the Lyapunov equations given in the previous section to obtain r. ω, ϕ_d and ϕ_d^* values. We simulate the equations for cases when $r \leq m$ and $r > m$. Each simulation runs in a loop till the condition of $r \leq m$ is reached. This is the point where the switching motion takes place. The wheelchair has a constant angular velocity after this point, while having no linear velocity.

Consider figure 13. Here, we notice a gradual decrease in the r values till $r = m = 0.2$, just like the original outcome. This shows that the wheelchair is getting closer to the door in every loop. Although this is evident from the wheelchair coordinates being hardcoded, it provides us with an initial check that the ranges of X_D and Z_D chosen are correct. From the same figure, we can also observe that the desired trajectory given by the ϕ_d^* value is very similar to the original values in figure 12 (Top right graph).

Both these outcomes act as preliminary checks to verify that our implementation of the Lyapunov Controller algorithm is correct.

Figure 14 now shows the r value, the ω output and the ϕ values for comparison. Notice that the ω stays almost constant till the r value reaches $0.2m$. Once this condition is reached, there is a small spike in values before it stabilizes again and begins to output a constant ω. This spike represents where the switching action of the Lyapunov controller is taking place.

The bottom-most graph in this figure shows a comparison between ϕ_d^* and ϕ_d values. In an idealistic case, the trajectory of ϕ_d should follow that of ϕ_d^*. However, as we do not have accurate estimates of the target pixel coordinates X_D and Z_D, ϕ_d does not entirely follow ϕ_d^*.

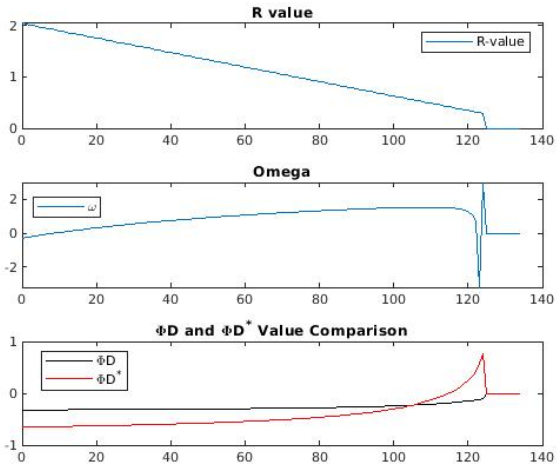

Fig. 14: Fig. 14: Graphs comparing the r value, ω and ϕ_d vs ϕ_d^*. Observe the sudden spike in the w value after the r value reaches the margin m of 0.2m. Because we cannot accurately simulate the doorpost pixel coordinates X_D and Z_D, the trajectory of ϕ_d does not exactly follow ϕ_d^*.

C. Comparison between Simulation and Original Outcome

The original output depends on an image feature estimate for determining the doorpost pixel coordinate. This is done iteratively over several frames of images captured by the wheelchair over the course of the servoing process.

In our simulations however, we do not have these estimates, as a real time model of a wheelchair is difficult to simulate. Due to this, we only input the pixel coordinates of the doorpost in terms of a sequence of values in a loop, which are constantly decreasing towards the goal.

The major difference between our simulation graph and the original graph comes in case of which one follows the desired trajectory determined by ϕ_d^*. In the original case, the current trajectory ϕ_d follows the desired trajectory ϕ_d^* very closely. However in our case, as we only simulate these values using an estimate on increments in arrays, it does not accurately follow the desired path.

Overall, we can observe that our simulated graphs follow a shape similar to that of the graphs in the original paper. However, we do have some abnormalities, which can be attributed to two factors: Firstly, the number of simulation timesteps we consider are significantly smaller, due to the computational costs. This affects the shape of the graph slightly over a large period of time.

Secondly and most importantly, the outcome of the simulations does not have an element of Computer Vision, which is a crucial aspect of the original paper. We do not perform doorpost extraction to get pixel coordinates for servoing, which is an important aspect of the physical task.

VIII. Conclusion

We have provided a report on a paper that describes a Visual Servoing approach for Autonomous Corridor Following and a Lyapunov Based Controller for Autonomous Doorway Passing. Our report presents an analysis of both these approaches (The Doorway Passing task in particular). For the task of Autonomous Corridor Following, a controller is designed using Vanishing Point features extracted from the corridor image. We have discussed the method used and the derivation of the control law.

The second task of Autonomous Doorway Passing is achieved using a Lyapunov based Controller. This controller drives the geometric error between a desired trajectory and a specified trajectory to zero by estimating an appropriate angular velocity. It can be observed that this avoids collision with the wall and guarantees that the wheelchair positions itself in front of the doorway regardless of its initial position.

Finally, we perform a detailed analysis of the original paper against our own simulations. We can conclude from this that the method demonstrated in the paper is robust, and performs both Autonomous Corridor Following and Autonomous Doorway Passing successfully.

References

[1] F. Pasteau, V. K. Narayanan, M. Babel, and F. Chaumette, "A visual servoing approach for autonomous corridor following and doorway passing in a wheelchair," *Robotics and Autonomous Systems*, vol. 75, pp. 28–40, 2016.

[2] T. Gomi and A. Griffith, "Developing intelligent wheelchairs for the handicapped," in *Assistive Technology and Artificial Intelligence: Applications in Robotics, User Interfaces and Natural Language Processing*, V. O. Mittal, H. A. Yanco, J. Aronis, and R. Simpson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 150–178, ISBN: 978-3-540-68678-1. DOI: 10.1007/BFb0055977. [Online]. Available: https://doi.org/10.1007/BFb0055977.

[3] S. P. Levine, D. A. Bell, L. A. Jaros, R. C. Simpson, Y. Koren, and J. Borenstein, "The navchair assistive wheelchair navigation system," *IEEE Transactions on Rehabilitation Engineering*, vol. 7, no. 4, pp. 443–451, 1999, ISSN: 1558-0024. DOI: 10.1109/86.808948.

[4] E. Demeester, E. B. V. Poorten, A. Hüntemann, and J. D. Schutter, "Wheelchair navigation assistance in the fp7 project radhar: Objectives and current state," 2012.

[5] A.-M. Kokosy, T. Floquet, G. Howells, H. Hu, M. G. Pepper, M. Sakel, and C Donze, "Sysiass ? an intelligent powered wheelchair," in *First International Conference on Systems and computer Science*, 2012. [Online]. Available: https://kar.kent.ac.uk/59307/.

[6] V. S. Dorbala, A. H. A. Hafez, and C. V. Jawahar, *A deep learning approach for robust corridor following*, 2019. arXiv: 1911.07896 [cs.RO].

[7] 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, 2010, ISSN: 1558-0210. DOI: 10.1109/TNSRE.2010.2049862.

[8] T. Carlson, R. Leeb, R. Chavarriaga, and J. del R. Millán, "The birth of the brain-controlled wheelchair," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5444–5445. DOI: 10.1109/IROS.2012.6386299.

[9] S. Blažič, "A novel trajectory-tracking control law for wheeled mobile robots," *Robotics and Autonomous Systems*, vol. 59, no. 11, pp. 1001–1007, 2011, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2011.06.005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889011001023.

[10] J. J. Park and B. Kuipers, "A smooth control law for graceful motion of differential wheeled mobile robots in 2d environment," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 4896–4902. DOI: 10.1109/ICRA.2011.5980167.

[11] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," *IEEE Robotics Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006, ISSN: 1558-223X. DOI: 10.1109/MRA.2006.250573.

[12] R. Grompone von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "LSD: a Line Segment Detector," *Image Processing On Line*, vol. 2, pp. 35–55, 2012. DOI: 10.5201/ipol.2012.gjmr-lsd.

[13] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "Lsd: A fast line segment detector with a false detection control," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 4, pp. 722–732, 2010.

[14] P. Bouthemy, "A maximum likelihood framework for determining moving edges," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 5, pp. 499–511, 1989, ISSN: 1939-3539. DOI: 10.1109/34.24782.

[15] S. Boukir, P. Bouthemy, F. Chaumette, and D. Juvin, "A local method for contour matching and its parallel implementation," *Machine Vision and Applications*, vol. 10, no. 5, pp. 321–330, 1998, ISSN: 1432-1769. DOI: 10.1007/s001380050082. [Online]. Available: https://doi.org/10.1007/s001380050082.