A Deep Learning Approach for Robust Corridor Following

Vishnu Sashank Dorbala[1], A.H. Abdul Hafez[2] and C.V. Jawahar[1]

***Abstract*— For an autonomous corridor following task where the environment is continuously changing, several forms of environmental noise prevent an automated feature extraction procedure from performing reliably. Moreover, in cases where pre-defined features are absent from the captured data, a well defined control signal for performing the servoing task fails to get produced. In order to overcome these drawbacks, we present in this work, using a convolutional neural network (CNN) to directly estimate the required control signal from an image, encompassing feature extraction and control law computation into one single end-to-end framework. In particular, we study the task of autonomous corridor following using a CNN and present clear advantages in cases where a traditional method used for performing the same task fails to give a reliable outcome. We evaluate the performance of our method on this task on a Wheelchair Platform developed at our institute for this purpose.**

I. INTRODUCTION

The task of autonomous corridor following has been well discussed in the past [1]–[5], especially on smart wheelchair platforms. Several classical works achieving this [6]–[11] use vision based algorithms. They extract selected features from a captured image, and pass them to a control law that computes a corrective velocity signal for adjusting the position of the robot on the corridor. There also exist other approaches that use different sensors [12]–[14], and follow a similar procedure. In all of these works, there is an inherent reliance on a robust feature extraction and tracking step to provide reliable features to the control law. As such, the behaviour of the robot becomes undefined when the system fails to provide these features reliably.

In traditional visual servoing (TVS) approaches, when selected features do not appear in the captured image, or when the extracted features are grossly inaccurate, the control law fails to produce a reliable velocity for servoing the robot along the corridor. We can infer from this that for a TVS process to take place reliably, three major factors need to be accounted for to a good degree of accuracy.

1) Quality image features need to be selected for servoing.
2) They need to be available in the environment.
3) A robust algorithm is needed for tracking and extracting these features.

The works presented in [6], [7], [9] are examples of TVS works. In [6], autonomous corridor following is performed on a wheelchair following a TVS approach that uses vanishing point features from a corridor image for devising a control signal. The work in [7] extends this to doorway traversal and presents it comprehensively. A similar approach for mobile robots is also proposed in [8] and extended further in [9]. Steps for feature extraction which are suggested in these works however do not account for images taken in dynamically changing environments, or for various noise types in the captured images. This hinders with the practical capability of the robot, as motion noise and occlusions are a common occurrence in the environment. Moreover, in cases where the required features cannot be estimated from the image, the outcome of the control law is undefined as it may encounter mathematical singularities. In order to overcome all of these challenges, we propose using a convolutional neural network for approximating a velocity vector output for corridor following directly from a camera image. Figure 1 provides an overview of the proposal to solve the problem as a robust alternative to traditional visual servoing.

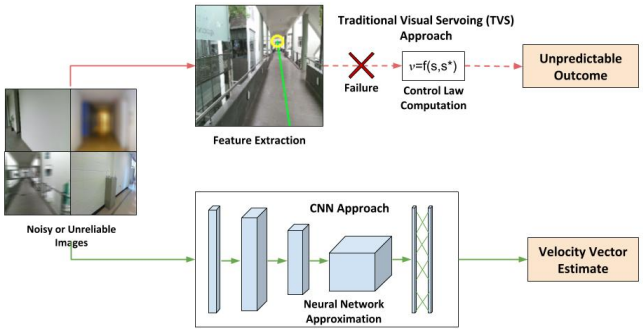


Fig. 1: Overview of the proposed *CNN Approach* compared to the *Traditional Visual Servoing (TVS)* used for autonomous corridor following. While the TVS approach performs well on clean images, it often fails to give a reliable output on noisy or occluded images taken from dynamically changing environments. The dotted line represents an unreliable step in the process. In such cases, a neural network can be used to estimate a desirable output.

Although there are a few works in the literature that use deep neural networks for visual servoing such as [15], [16], our proposal (Also see [17]) differs from them as we combine both the feature extraction and control signal computation in one stage, while they primarily focus on approximating the feature extraction stage. In [15] the authors finetune FlowNet [18] to estimate the relative angular and translational pose differences between the desired image and the current image. Similarly in [16], AlexNet has been used to approximate a relative pose between the current image and a reference image. In both these papers, the approximated relative pose is fed into a control law that computes a velocity vector for servoing. The approach that we present in this work

[1] The authors are associated with the International Institute of Information Technology, Hyderabad, India vdorbala@gmail.com, jawahar@iiit.ac.in

[2] A.H. Abdul Hafez is with Hasan Kalyoncu University, Gaziantep, Turkey abdul.hafez@hku.edu.tr

combines the feature extraction and control law computation stages into one framework to directly predict a velocity vector, given an image.

In addition to this, in [19], a Q learning based approach for visual servoing has been described for performing a target following task using a drone in a simulated environment. They demonstrate the efficacy in using deep features for robust servoing in noisy and occluded environments which further reinforces our usage of deep learning for this visual servoing task.

Our paper makes the following two contributions: i) we introduce a novel CNN based approach for performing an image based visual servoing task of autonomous corridor navigation, and ii) we present a robust comparison showcasing the advantage of our CNN approach against some critical drawbacks of the traditional approach described in [6]. We carry out a rigorous analytical and practical analysis here to make our case for supporting this claim.

The paper has been organized as follows. Section II describes fundamental TVS concepts used for autonomous corridor following, and provides details of our CNN based approach. In Section III, we describe the methods we use for robust analysis of our CNN approach where TVS-based approaches fail to perform well. In IV, we showcase the results of our experimentation both statistically and practically on a Wheelchair Platform developed at our institute. Finally, we present the advantages of our method by evaluating it on fail cases of the traditional approach.

II. CNN-Based Autonomous Corridor following

The basic modelling and control concepts of using TVS in a corridor following task is presented in this section. After that we discuss our CNN-based design along training and data preparation issues.

A. TVS Modelling and Velocity Estimation

The wheelchair is assumed to be a four wheeled robot with two passive castor wheels in front and two actuated wheels in the rear. It thus behaves as a non-holonomic system constrained by two degrees of freedom. In order to servo this system along the corridor, a translational velocity v and an angular velocity ω that describe its motion need to be computed. As our purpose for autonomous corridor following is for assisting the disabled wheelchair users, a constant and slow forward velocity v, along with an ω for adjusting the position of the chair on the corridor is sufficient for completing the task.

In [6], the authors describe a traditional approach for autonomous corridor following that makes use of vanishing point and vanishing line features to perform this task. They use an automated feature extraction mechanism that adds constraints on the lines detected by the LSD algorithm [20], [21] from a captured image to obtain x_v and θ_v, the selected features for servoing. x_v is the x coordinate of the vanishing point, while θ_v is the angle that the vanishing line makes with the corridor plane. (Refer Figure 2)

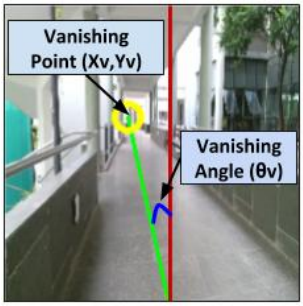


Fig. 2: Vanishing Point (x_v) and Vanishing Angle (θ_v) features that are extracted by the TVS approach in [6]. These features are passed through a control law that computes a corrective velocity for autonomous corridor following. The red line is perpendicular to the corridor plane.

In our approach, we use a human annotator to mark x_v and θ_v features on an image. This ensures that the outcome of these features is reliable, as it is often the case that the automated feature extraction step fails to extract accurate features. This occurs mainly due to environmental noise in the captured image or sub-optimal feature extraction parameters which are difficult to tune.

For autonomous corridor following, the desired motion is achieved when the wheelchair is moving straight and is positioned at the center of the hallway. This occurs when x_v lies at the center of the captured frame i.e the origin, and θ_v is perpendicular to the corridor plane in the image. The corresponding feature values of (0,0) are consequently chosen as the desired feature values.

In [22], a control law for servoing an image based path following system such as ours is formulated as:

$$\omega = -J_w^+(\lambda e + J_v v^*) \tag{1}$$

The ω value here represents the ground truth angular velocity that we require to train our network. The Jacobians J_w and J_v are defined in [6] as follows:

$$J_w = \begin{bmatrix} 1+x_v^2 \\ -\lambda_{\theta_v} lc + \lambda_{\theta_v} w\rho + \rho s \end{bmatrix} \tag{2}$$

$$J_v = \begin{bmatrix} 0 \\ -\lambda_{\theta_v}\rho \end{bmatrix} \tag{3}$$

Here x_v and θ_v are the selected features, $c = \cos\theta_v$, $s = \sin\theta_v$, $\rho = x_v \cos\theta_v + y_v \sin\theta_v$ and $\lambda_{\theta_v} = \cos\theta_m / h$. The h, w and l values represent the position of the camera on the wheelchair, which in our case is set to $h = 0.5$m, $w = 0$m, and $l = 0$m. λ and v^* are gain and translational velocity constants that are tuned to 10^2 and $0.2m/s$ for our task. The error e is defined as the difference between the extracted features and the desired feature values.

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

The vanishing point coordinates are measured in meters and the slope of the vanishing lines are taken in radians.

The ω obtained here is the ground truth value used for training our convolutional neural network model.

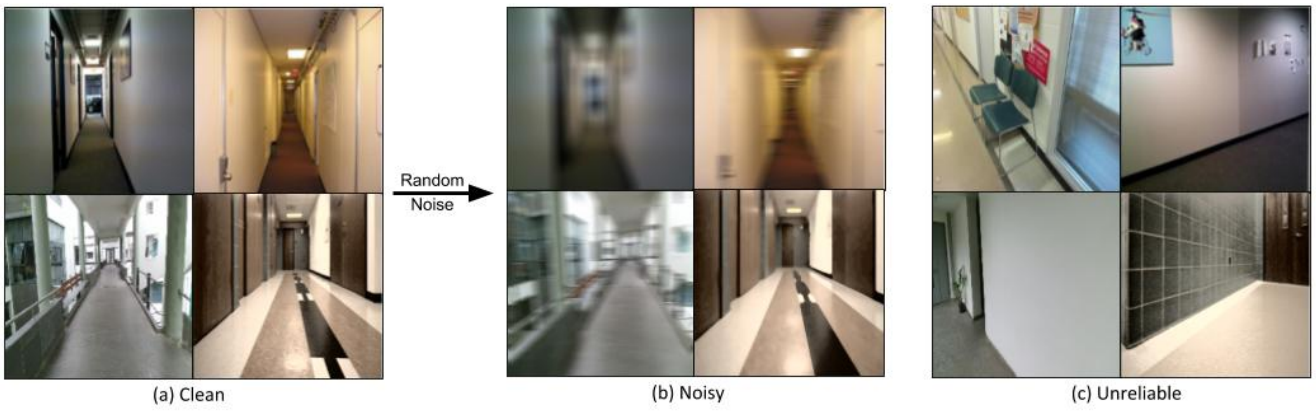


Fig. 3: Samples of images present in the dataset. The diversity in corridor environments can be observed in (a). Four different types of noises are artificially mixed with these images to obtain the noisy images shown in (b). Images where the required vanishing point features for computing the ground truth ω could not be extracted are shown in (c). These have been discarded from training.

B. The Training Dataset

For training our network, we gather suitable corridor images from various open access sources [23]–[27]. We also create our own dataset of corridor images belonging to our institute. The accumulated set consists of 3563 images in total. A small subset of 403 images belonging to this set have been discarded from training as their ground truths could not be estimated. This is because the vanishing point feature x_v in these cases lies outside the frame of the image and cannot be extracted reliably. These images are deemed unreliable and Figure 3(c) shows examples of such cases. The remaining samples compose a clean set of images.

We add 4 different types of artificial noise (Mild and Strong Gaussian Blur, Motion Blur and JPEG Compression) randomly to the entire clean set and obtain a separate noisy set of images. The final dataset is a combination of the clean and noisy sets and contains 6320 images. The ground truth values for samples in the noisy set are identical to their counterparts in the clean set. Fig 3(a) and 3(b) show examples of clean images their noisy counterparts.

Adding noisy images to our dataset serves a dual purpose of increasing the data used for training our model as well as helping the neural network generalize better to noisy data. The train-test split on the final dataset is 90-10% and 10% of the train set generated is used for validation. As the test set is randomly sampled from the final dataset, and contains a mixture of clean and noisy images.

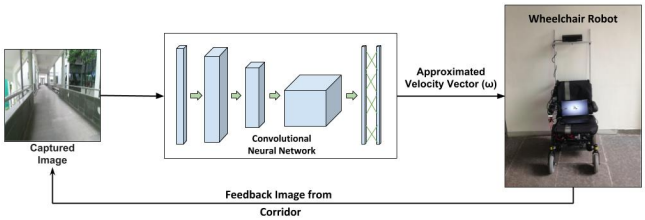


Fig. 4: Overview of our CNN Approach. An image captured from the corridor environment and passed through the CNN. The approximated velocity vector ω is then used to perform servoing on the wheelchair.

C. Designing and Training our CNN

Figure 4 provides an overview of our visual servoing approach. We employ a technique called transfer learning [28], [29] for training our model. In the most literal sense, transfer learning refers to "transferring" knowledge obtained from training one model to another model that performs a task of a different nature. This is especially useful in cases such as ours where the dataset size is small and there is insufficient training material for neural network to converge.

We exploit this technique and fine-tune a ResNet-18 architecture [30] pre-trained on ImageNet for our task. This setup was chosen considering ResNet-18's exceptional performance on ImageNet despite having a comparatively small model size. The model was pre-trained on ImageNet with an input size of 224x224, and an output size of 1000 classes. All images in our dataset have accordingly been re-sized to 224x224 to make sense of the pre-trained weights. We replace the final layer of this pre-trained model with a 1-dimensional output that represents the required angular velocity ω for our servoing task. We perform regression using this setup.

A Mean Squared Error (MSE) loss function determines the gradients for backpropagation for each iteration. In our case, this can be written as,

$$loss = \frac{1}{n}\sum_{i=0}^{n}(\hat{\omega} - \omega)^2 \tag{5}$$

Here, n is the batch size during training which has been set to 8. $\hat{\omega}$ is the predicted angular velocity after a forward pass through the network and ω is the target angular velocity for that sample.

We train the network on an Nvidia 1080 Ti having 12GB of GPU memory and 64GB of RAM. It takes around 30 minutes for running 40 train-validation epochs. We employ a Stochastic Gradient Descent scheme with a weight decay of 0.005 and momentum of 0.9. The learning rate is set to 0.005. 10-fold cross validation was performed to understand the variance in training data and accordingly tune these network hyperparameters.

D. Network Evaluation Metric

The R^2 value or coefficient of determination has been used as an evaluation metric for assessing the performance of the neural network on the task of regression. It can be defined in our case as:

$$R^2(\omega, \hat{\omega}) = 1 - \sum_{i}^{n-1} \frac{(\omega_i - \hat{\omega})^2}{(\omega_i - \bar{\omega})^2} \tag{6}$$

Here, ω represents the true value i.e, the target distribution, $\hat{\omega}$ represents the predicted distribution, and $\bar{\omega}$ is the mean of the target distribution. n here represents the number of samples taken from the distribution, which in our case is the number of images in the test set.

The R^2 value ranges from $-\infty$ to 1. A positive value closer to 0 indicates that the model is unable to explain the variability of the data, while a value closer to 1 shows that the output corresponds well with the target distribution.

III. Robustness Analysis of CNN-based Corridor following Scheme

In this section, we discuss two methods for evaluating the performance of our CNN approach in cases when the TVS-based approaches like the one based on vanishing feature approach fails to perform well.

A. Comparing Deep and Vanishing Features

When the vanishing feature method fails to produce a good ω, it is often due to feature extraction going wrong. Our CNN approach is independent of this explicit feature extraction step and gives an approximation for ω even in these fail cases. In order to better understand the outcome of our CNN approach, we try to estimate the deep feature x_v from the ω obtained by the CNN, and compare its performance against the vanishing point feature x_v obtained by the vanishing point approach on the ground truth.

Recall the control law from [22] presented in section II.

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

Here, the Jacobians J_v, J_w and error e are described in equations 2, 3, and 4 respectively. After substituting these values and other constants mentioned in section II, we obtain:

$$\omega = \begin{bmatrix} -1-x_v^2 & -\rho\sin\theta_v \end{bmatrix} \begin{bmatrix} \lambda x_v \\ \lambda\theta_v \end{bmatrix} + \begin{bmatrix} -1-x_v^2 & -\rho\sin\theta_v \end{bmatrix} \begin{bmatrix} 0 \\ -\rho\cos\theta_v v^*/h \end{bmatrix}$$

This equation can be reduced to the following:

$$\omega = -\lambda x_v - \lambda x_v^3 - \rho\lambda\theta_v\sin\theta_v + \frac{\rho^2}{h}\sin\theta_v\cos\theta_v v^* \tag{8}$$

Now, as we have only one equation with three variables, given ω we cannot find a closed form solution for x_v, y_v and θ_v. We can however limit the range of these values for our specific task to obtain a solution for a deep feature representing the vanishing point coordinate x_v from ω. Note that we do this only for illustrating the performance of our network against the TVS approach.

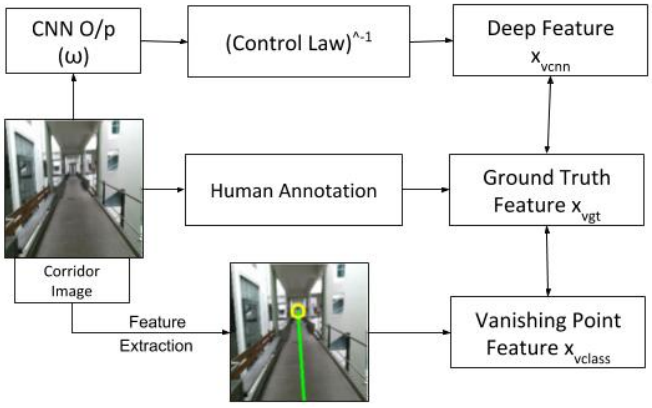


Fig. 5: Flowchart comparing Deep Features with TVS-based Vanishing Features. We perform a comparative analysis to understand why our CNN approach performs better in cases when feature extraction fails traditionally. This is achieved by extracting the x_v feature using both approaches and comparing them against a common human annotated ground truth.

As the x_v and y_v values are represented in metres in the image plane, we can safely assume that their absolute values would be less than $1m$. Thus, the absolute value of ρ i.e., $\sqrt{x_v^2 + y_v^2}$ also becomes less than 1. Using this observation, we can neglect the last term in equation 8 as it is comparatively smaller to the other terms that contain λ, a large constant.

From [6], we can also conclude that $\theta_v \in (-\frac{\pi}{2}, \frac{\pi}{2})$. However, in our experiments, we have observed that for most images in the dataset, $\theta_v \in (-\frac{\pi}{6}, \frac{\pi}{6})$. Due to this reason, we chose to neglect the third term containing θ_v in equation 8 as it does not have a significant effect on the ω value in our case. This can also be experimentally observed by computing ω while changing the θ_v value. We can then solve the following equation for x_v:

$$\omega = -\lambda x_v - \lambda x_v^3 \tag{9}$$

As this is a third degree equation, with the discriminant $\Delta < 0$, its real root x_v can be expressed in terms of ω as:

$$x_v = \sqrt[3]{\frac{-\omega}{2\lambda} + \sqrt{\left(\frac{\omega}{2\lambda}\right)^2 + \frac{1}{27}}} + \sqrt[3]{\frac{-\omega}{2\lambda} - \sqrt{\left(\frac{\omega}{2\lambda}\right)^2 + \frac{1}{27}}}$$

x_v here is a deep feature representing the vanishing point coordinate, that is obtained from the ω predicted by our CNN approach. This along with the traditional x_v obtained from a TVS-based automated feature extraction mechanism is compared with the ground truth. Figure 5 shows a flowchart of our approach.

B. Verifying Approximations for Unreliable Images

While training the model, we accounted for cases where the captured image is noisy, by adding noisy samples to the training data. However, there exist the unreliable images (Figure 3(c)) that were discarded from training the model as their ground truths could not be estimated using the

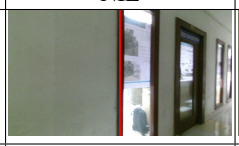
1					
CNN	-1.067	0.289	0.500	-0.622	0.059
TVS-Based	-1.045	0.263	0.438	-0.689	-0.025
Ground Truth	-0.978	0.275	0.437	-0.659	-0.017
2					
CNN	-1.472	-0.450	0.117	-0.346	-0.027
TVS-Based	NIL	-0.446	0.219	-0.389	-0.014
Ground Truth	NIL	-0.434	0.206	-0.359	-0.052
3					
CNN	0.360	0.785	0.148	-0.252	-0.042
TVS-Based	2.687	0.227	0.146	-2.397	0.052
Ground Truth	NIL	1.376	0.109	-0.445	0.03

TABLE I: Practical Results: Image sequences captured from different corridors locations at our institute. The red line represents the ideal vanishing line when the wheelchair is at the center of the corridor and is used as a directional reference. The values are the estimated angular velocities for our CNN approach, TVS-Based vanishing feature approach, and a human annotated Ground Truth (GT). A positive ω value represents clockwise motion, and a negative value represents anti-clockwise motion.

traditional method. Our trained CNN model however can predict an approximation for ω for these images.

Just by looking at these unreliable images, a human can decipher if the wheelchair is meant to turn left or right to initiate the corridor following task. Also once a prediction is made, we know the direction of motion from the sign of the ω value. We leverage upon these two pieces of detail for partially verifying the accuracy of the predicted outcome on unreliable images using human annotation.

For each unreliable image in the dataset, the following steps are taken by a human annotator:

- Pass the image through the trained network and obtain an approximation for ω. Classify it as left or right based on the sign of the value predicted.
- Show the same image to a human annotator equipped with a binary output console corresponding to the left or right direction.
- Compare the human annotated output with the network output and update two score values representing accuracy and false positive accuracy.

Each annotator is shown the entire dataset 3 times, and an average of the scores obtained is chosen for evaluation.

The accuracy score tells us how well the network is able to predict the correct direction of motion from the image. It is the percentage of unreliable images that have had their outcome predicted in the right direction of motion.

$$Accuracy\ Score = \frac{n_i}{n}\% \tag{11}$$

Here, n_i is the number of correctly predicted samples and n is the total number of unreliable images.

The false positive score quantifies the severity of the network's bad performance on unreliable images. It is the average of the absolute ω value on images where the wrong direction has been predicted.

$$False\ Positive\ Score = \frac{\Sigma\omega_j}{n_j} \tag{12}$$

Here, n_j is the total number of false positive samples and ω_j is the angular velocity predicted for these samples.

For autonomous corridor following in unreliable cases, a high accuracy score and low false positive score is desired.

IV. Experiments and Results

A. Evaluation of Neural Network Performance

We evaluate our trained model on 4 noisy test sets in addition to the original test set using the R^2 score described in section II-D. Each noisy set comprises of images having one specific type of artificial noise. Table II shows the percentage R^2 value for each test set. Here, a similar score in all the test sets shows that the performance of the CNN on the noisy sets is as good as its performance on the original, non-noisy set, thereby establishing robustness to noise.

On the unreliable test set, following the human verification method described in the earlier section (III-B), we get an accuracy score of 78.75% on predicting the right direction of motion on 403 unreliable images. We get a false positive score of 0.180 on the 88 images that were predicted wrong. This translates to 5.16% of the highest ω value obtained in our tests which shows that even when the wrong direction is predicted, the magnitude of the velocity vector output remains relatively small. Although this is specific to our case,

Test Set Type	R^2 Value(%)
Original (Clean)	88.321
Motion Blur	88.011
JPEG Compression	88.572
Gaussian Blur	88.340

TABLE II: Comparison of R^2 Values on Test Sets: Here, as the R^2 values are similar across all the test sets, we can safely conclude that the performance of the neural network on noisy images is on par with that of clean images.

it is a significant result as the CNN performs well on these images where the traditional vanishing feature approach would fail entirely.

B. Practical Implementation and Results

We practically evaluate our method on an Intelligent Wheelchair Platform developed at IIIT, Hyderabad[1]. A Kinect v2 has been retrofitted onto this platform as a sensor for capturing images. All processing is done on board on a laptop having an Nvidia 1050 Ti with 4GB GPU memory and 8GB RAM. A Sabertooth motor controller attached to the wheelchair takes serial commands from the laptop and translates them into actuary signals that controls its motion. The entire epoch time from capturing an image to actuation takes ≈ 1.8 seconds or ≈ 0.6Hz on this setup. Although slow for many real time systems, this control frequency is sufficient for our task as our final application is on an assistive wheelchair for the disabled. Our translational velocity is set to 0.2m/s, ensuring that the wheelchair is slow enough for sufficient coherency between subsequent frames.

We conduct autonomous corridor following experiments on different corridors across our institute, including environments that were previously unseen in the training dataset. In each experiment, the wheelchair is made to start at an arbitrary position at the beginning of the corridor making an arbitrary angle between $[0^\circ, 90^\circ]$ with the wall. The corridor following task is then carried out using the proposed CNN method and the images captured are stored along with their corresponding CNN ω values. We then use the traditional vanishing feature approach to estimate an ω on these stored images, and also a ground truth ω using human annotation (Refer II-A).

Table I has samples of image sequences captured during the experiment and their corresponding ω values for the CNN, vanishing feature approach and the ground truth. There is a high correspondence between the values of the CNN and vanishing feature approach in sequence 1. In sequence 2 however, when the wheelchair starts at a sharper angle with the corridor wall, an 'unreliable image' is captured due to which the traditional ω does not get computed. A ground truth does not exist here either, as a human annotator cannot accurately mark features outside the image frame. The CNN approach here predicts a velocity in the anti-clockwise direction, which enables the wheelchair to initiate and follow through the servoing task.

[1]https://youtu.be/aRGVXq8cqDs

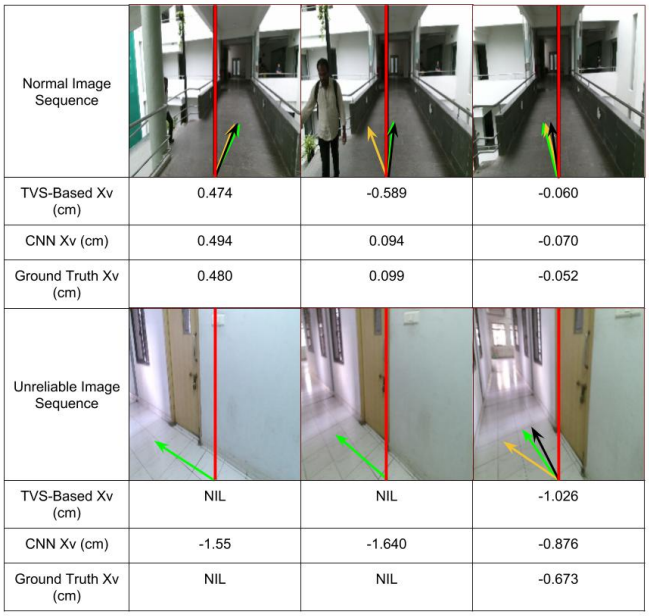


Fig. 6: Comparison of Deep vs Traditional Features. The yellow arrow represents the traditional approach, the green one represents the CNN approach, while the black one represents the ground truth. The red line at the center of the image is for directional reference. The angles that the arrows make with this line are directly proportional to x_v. In the normal sequence, observe the disruptive value of the traditional x_v when a person (environmental noise) enters the frame. In the unreliable sequence, observe the inability of the TVS method to extract the x_v feature in the first two images.

Sequence 3 is taken from an environment outside the training dataset. Observe the erratic ω values that the traditional approach estimates due to bad feature extraction. This is primarily because we do not re-tune the traditional approach parameters for extracting features from this environment. The CNN on the other hand first predicts a small value for servoing on the unreliable image captured in the beginning. Once the corridor is fully visible, it predicts a better approximation closer to the GT value, and successfully completes the servoing task.

C. Advantages of the CNN approach

We use the method described in III-A to extract x_v from the CNN ω, and compare it with the vanishing point outcome (Refer Figure 6).

1) Robustness to Environmental Noise: As our CNN is trained offline on several images of various corridor environments (both noisy and non-noisy) it works well on different environments including ones that are dynamically changing.
   The normal image sequence in Figure 6 illustrates this with an example. In the second image, when a person enters the frame, the traditional approach fails to compute a correct ω, as its feature extraction step that is dependant on a line detector fails. The x_v obtained is thus not representative of the actual vanishing point. Our CNN approach on the other hand predicts a velocity ω in the correct direction, which is backed up by a good deep feature x_v extracted from the image.

2) Approximations for Unreliable Images: As mentioned earlier in Section II, using the traditional method for servoing fails on unreliable images. This is due to the feature extraction step failing by virtue of the required vanishing point feature x_v not lying on the image frame. Here, even if x_v is extracted as an extended coordinate, its value is cannot be verified. A very large x_v can cause the control law parameters to "explode" leading to the calculation of an unstable ω. This holds true especially in cases where the x_v extracted tends to ∞. The control law reaches a mathematical singularity here.
Observe the unreliable image sequence in Figure 6. Here, in the first two images, as a well defined vanishing point x_v does not exist, the traditional method fails. However, our CNN estimates x_v as a deep feature, that enables motion in the correct direction, due to which corridor following becomes feasible.

V. Conclusion and Future Work

We have shown that our approach has an advantage where a velocity outcome is predicted regardless of the input image. This can also be a disadvantage in some cases, where the wheelchair needs to stop in order to complete the task. To overcome this, future work may include training the CNN with 'end of the corridor' and 'object of interest' cases where the wheelchair would have to stop and reconsider its position before moving.

There is also a disadvantage in terms of the time taken for gathering a dataset for the purpose of corridor following, which is not required in traditional schemes. We plan to release the dataset of corridor images along with their human annotated ground truths to alleviate this issue for other researchers.

In conclusion, we have presented an end to end CNN based approach for autonomous corridor navigation on a wheelchair. Our network is trained to predict a velocity signal for the servoing task from a captured image. In doing this, our method overcomes some key limitations of a traditional visual servoing based approach. We demonstrate these by performing a statistical and experimental validation of our approach against the traditional approach.

References

[1] S. R. Bista, P. R. Giordano, and F. Chaumette, "Appearance-based indoor navigation by ibvs using line segments," *IEEE RA-L*, vol. 1, no. 1, pp. 423–430, 2016.

[2] J. M. Toibero, C. M. Soria, F. Roberti, R Carelli, and P Fiorini, "Switching visual servoing approach for stable corridor navigation," in *2009 Int. Conf. on Advanced Robotics*, IEEE, 2009, pp. 1–6.

[3] N. Ohnishi and A. Imiya, "Appearance-based navigation and homing for autonomous mobile robot," *Image and Vision Computing*, vol. 31, no. 6-7, pp. 511–532, 2013.

[4] A. Paolillo, A. Faragasso, G. Oriolo, and M. Vendittelli, "Vision-based maze navigation for humanoid robots," *Autonomous Robots*, vol. 41, no. 2, pp. 293–309, 2017.

[5] R. P. R. Padhy, S. Verma, S. Ahmad, S. K. Choudhury, and P. K. Sa, "Deep neural network for autonomous uav navigation in indoor corridor environments," *Procedia computer science*, vol. 133, pp. 643–650, 2018.

[6] F. Pasteau, M. Babel, and R. Sekkal, "Corridor following wheelchair by visual servoing," in *IEEE/RSJ IROS*, 2013, pp. 590–595.

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

[8] S. H. J. Vassallo R Frizera and J. Santos-Victor, "Visual navigation: Combining visual servoing and appearance based methods," in *Int. Symposium on Intelligent Robotic Systems, SIRS*, vol. 98, 1998.

[9] S. H. J. Vassallo Raquel F and J. Santos-Victor, "Visual servoing and appearance for navigation," *IEEE RAS*, vol. 31, no. 1-2, pp. 87–97, 2000.

[10] M. Meng and A. C. Kak, "Mobile robot navigation using neural networks and nonmetrical environmental models," *IEEE Control Systems Magazine*, vol. 13, no. 5, pp. 30–39, 1993.

[11] N. Winters, J. Gaspar, G. Lacey, and J. Santos-Victor, "Omni-directional vision for robot navigation," in *Proceedings IEEE Workshop on Omnidirectional Vision (Cat. No. PR00704)*, IEEE, 2000, pp. 21–28.

[12] J. Park, T. Kim, and T. Park, "Autonomous navigation system for a mobile robot using a laser scanner in a corridor environment," in *IEEE/SICE International Symposium on System Integration (SII)*, IEEE, 2015, pp. 512–516.

[13] R. Carelli and E. O. Freire, "Corridor navigation and wall-following stable control for sonar-based mobile robots," *IEEE RAS*, vol. 45, no. 3-4, pp. 235–247, 2003.

[14] G. Schouten and J. Steckel, "A biomimetic radar system for autonomous navigation," *IEEE Transactions on Robotics*, pp. 1–10, 2019, ISSN: 1552-3098.

[15] A. Saxena, H. Pandya, G. Kumar, A. Gaud, and K. M. Krishna, "Exploring convolutional networks for end-to-end visual servoing," in *2017 IEEE ICRA*, 2017, pp. 3817–3823.

[16] Q. Bateux, E. Marchand, J. Leitner, F. Chaumette, and P. Corke, "Training deep neural networks for visual servoing," in *2018 IEEE ICRA*, 2018, pp. 1–8.

[17] V. S. Dorbala, A. H. Abdul Hafez, and C. V. Jawahar, "A deep learning approach for robust corridor following from an arbitrary pose," *27th IEEE Signal Processing and Communications Applications Conf. (SIU)*, pp. 1–4, 2019.

[18] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *IEEE ICCV*, 2015.

[19] A. X. Lee, S. Levine, and P. Abbeel, "Learning visual servoing with deep features and fitted q-iteration," *arXiv preprint arXiv:1703.11000*, 2017.

[20] 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.

[21] J. J. Von Gioi Rafael Grompone, J.-M. Morel, and G. Randall, "Lsd: A line segment detector," *Image Processing On Line*, vol. 2, pp. 35–55, 2012.

[22] A. Cherubini, F. Chaumette, and G. Oriolo, "Visual servoing for path reaching with nonholonomic robots," *Robotica*, vol. 29, no. 7, 1037–1048, 2011.

[23] A. Bonarini, W. Burgard, G Fontana, M. Matteucci, D. Sorrenti, and J. Tardos, "Rawseeds: Robotics advancement through web-publishing of sensorial and elaborated extensive data sets," Jan. 2009.

[24] S. Ceriani, G. Fontana, A. Giusti, D. Marzorati, M. Matteucci, D. Migliore, D. Rizzi, D. G. Sorrenti, and P. Taddei, "Rawseeds ground truth collection systems for indoor self-localization and mapping," *Autonomous Robots*, vol. 27, no. 4, p. 353, 2009, ISSN: "1573-7527".

[25] G. Tsai and B. Kuipers, "Dynamic visual understanding of the local environment for an indoor navigating robot," in *2012 IEEE/RSJ IROS*, 2012, pp. 4695–4701.

[26] A. Quattoni and A. Torralba, "Recognizing indoor scenes," in *IEEE CVPR*, 2009, pp. 413–420.

[27] S. Yang, D. Maturana, and S. Scherer, "Real-time 3d scene layout from a single image using convolutional neural networks," in *2016 IEEE ICRA*, May 2016, pp. 2183–2189.

[28] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin, "Exploring strategies for training deep neural networks," *Journal of machine learning research*, vol. 10, no. Jan, pp. 1–40, 2009.

[29] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" In *Advances in neural information processing systems*, 2014, pp. 3320–3328.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE CVPR*, 2016.