2022 IEEE 9th International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA) | 978-1-6654-3445-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/CIVEMSA53371.2022.9853652

3D Text Recognition and Localization From Point Clouds via 2D Projection and Virtual Camera

Adrian Mai, Chelsea Mediavilla, Jane Berk, Mark Bilinski, Raymond Provost
Naval Information Warfare Center Pacific
53560 Hull Street, San Diego, CA 92152-5001
{adrian.mai, chelsea.m.mediavilla, jane.n.berk, mark.bilinski, raymond.c.provost2}.civ@us.navy.mil

***Abstract*—The lack of training datasets and computational complexity of 3 dimensions make text localization and recognition in point cloud environments challenging tasks, resulting in them being relatively undeveloped topics in the research community. In this paper, we introduce a method to adapt 2D text detection and recognition techniques in panoramic images with appropriate 3D mapping. This combined with heuristic methods such as a virtual camera creates a fast and efficient 3D text localization and recognition system. In real world applications, the objects of interest for a computer vision task may not be captured by the sensor from an ideal perspective; instead skewed imagery or partial occlusions are common. In a virtual 3D environment, we have full control of viewing angles and distances from the object. Therefore, by placing a virtual camera in certain positions, we can generate synthetic imagery of the object that is based on real imagery and avoids skewed views or occlusions. We use this synthetic imagery to improve the performance of text recognition when the object of interest is sufficiently close to the scanner, and hence the point density is high enough to generate quality imagery. The simplistic nature of this system is attractive and computationally inexpensive as it uses 2D data processes instead of natively 3D. The system shows promising results with over 85% accuracy for detection and localization tasks and 80% on the recognition task.**

***Index Terms*—Computer Vision, Object Localization, Point Cloud, Virtual Camera, Text Detection, Text Recognition**

I. Introduction

Despite growing bodies of research focusing on 2D text detection and optical character recognition (OCR), these tasks in 3D spaces are still relatively new to the computer vision community due to limited 3D data resources.

Current public 3D datasets often focus on applications such as autonomous driving or interior structure. In our previous paper [1], we provided a new and unique 3D dataset made up of shipboard 3D scans and used it for the purpose of text detection. In this work, we continue this research and rectify some of our previously discovered shortcomings when detecting, localizing, and recognizing the text of bullseyes, text placards that help Sailors navigate ships. We believe this can enable interior shipboard navigation in robotic and augmented reality applications.

In [1], we were challenged by the noisy nature and massive size of 3D point clouds, and the skewing phenomenon of the 2D panoramic images. In this work, we address previous problems, consider multiple bullseyes in one scan, and introduce a virtual camera method to recognize 3D text. We make our architecture more modular so we can test multiple combinations of detection engines and OCRs. Our code [1] and data[2] are available.

II. Background

A. Text Detection and Recognition

OCR has become a common place task in computer vision and is readily available in many off-the-shelf packages. Similarly, the preliminary step to OCR, text detection, is often provided as pre-trained models for easy application in various text filled scenes. When combined, the pipeline is able to detect and locate text in a scene (text detection) and then transcribe the characters into human-readable characters (OCR). We focus on text detection algorithms Paddle [2], Efficient and Accurate Scene Text (EAST) [3], and Character Region Awareness For Text (CRAFT) [4]. For the OCR algorithms, we look at Tesseract [5], Paddle [2], and Attentional Scene Text Recognizer with Flexible Rectification (ASTER) [6].

While out of the box algorithms work for general tasks, current OCR engines suffer when there is image distortion; e.g. skewed angles or out-of-focus text. Since our data was not collected with the intention of OCR, the camera angles are often not aligned with the text (such as in Figure 9), making the task of text recognition more challenging.

To tackle this problem, we introduce heuristic methods such as rectification of images recognized with ASTER [6] and deskewing images by converting from equirectangular to perspective view. When applied to the image itself, the rectification and deskewing methods can help mitigate unwanted effects from the 2D panorama. However, when the text is out of focus, or the angle is too sharp, even these methods are insufficient – Figure 12a for example.

B. 3D object detection

Current state of the art 3D detection methods directly extract the geometric features of objects inside a point cloud structure [7], [8]. Although these methods give decent results, it is very computationally expensive, and labeling in the 3D domain is difficult and tedious. Instead, we leverage the one-to-one mapping between 2D panoramas and 3D point clouds to utilize more mature 2D computer vision models.

[1] https://github.com/quocanh010/3DTextDLR
[2] https://drive.google.com/open?id=1-JmWPIzUmuzz9g-f-XLtgj808bcN0QhE

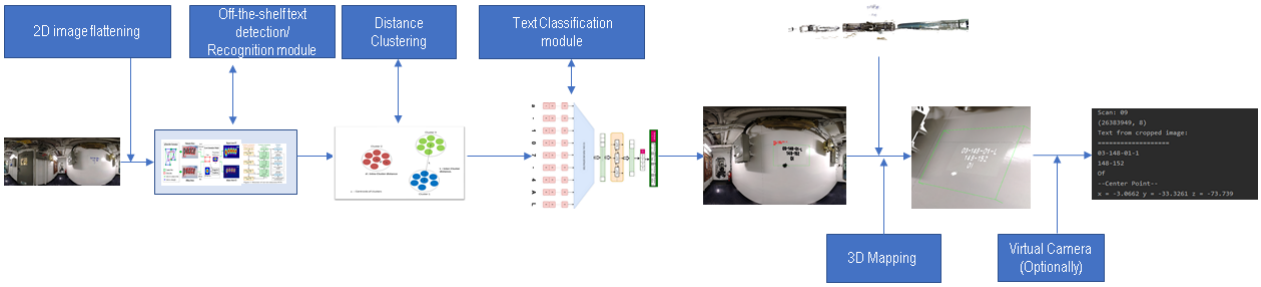


Fig. 1: Algorithm Pipeline

C. Synthetic Data

Training on synthetic data has proven useful as an alternative [9]–[14] to manually labeling the large amounts of data needed to properly train a machine learning model. We utilize synthetic text to train a classifier for our target dataset: bullseye text (discussed further in Section III-B).

Additionally, synthetic imagery presents the opportunity for obtaining new or novel viewing angles of data. This has taken forms including generative adversarial networks [15], [16], domain adaption and style transfer techniques [17], [18], and image generation via gaming engines [19], [20]; all of which work toward the goal of increasing the usability of synthetically developed data.

Novel synthetic angles have been demonstrated in a number of different applications. In [21], the authors combine imagery from real cameras and synthetic images made from virtual cameras to refine a 3D reconstruction of bubbles and droplets. In facial recognition, there is substantial research in the ability to recognize a face from a previously unseen angle [22]–[24]. In this paper, we introduce a kindred but unique method to our text recognition problem by positioning a virtual camera perpendicular to our target in 3D space. This allows us to create a modified, synthetic view with clearer, more readable text samples.

D. Data

We expand (double) our dataset in [1], captured with a FARO scanner [25], and perform analysis on 105 bullseye instances across five ships, A - E. Our ship data containing bullseyes has the following distribution: 14%, 7%, 18%, 13%, and 48% for Ships A - E respectively. There are two styles of bullseyes in the various 3D scans. Ship C contains scans from the USS Midway Museum, which has legacy bullseyes with a yellow background and black stenciled text. The other ships (A, B, D, and E), are modern and have a different bullseye format: white background with blue stenciled text – shown in Figure 2. Selection of images that contain bullseyes and labeling of ground truth for detection and localization were established through manual annotation as in [1]. Additionally, text was manually transcribed and recorded for the recognition task.

E. Metrics

The evaluation of each detection algorithm in the 2D domain is presented as a binary confusion matrix containing true and false, positive and negative detections. These matrices (Figure 3) were created by manually evaluating

Fig. 2: Example of a modern bullseye.

the results from each algorithm in the 2D detection step; this process is further discussed in section III-B. A circle was automatically drawn on each detected bullseye, then the image was visually inspected.

In the 3D localization phase, the Euclidean distance between the ground truth and predicted midpoint, and the angle between the ground truth normal vector and the predicted normal vector are measured in centimeters and degrees, respectively. Each result is binned into a histogram with unequal bins as in [1].

Combinations of algorithms are tested against all three metrics in order to determine the most accurate combination. This combination is then used to generate the 3D centroids and normal vectors for the recognition stage and further analysis with the virtual camera.

We compare the accuracy of the OCR engines' predictions to the text transcribed in the ground truth, both before and after the application of the virtual camera.

III. Methodology and Results

A. Algorithm Combinations

We considered three detection and three OCR algorithms as discussed in Section II. Each possible pairing results in nine algorithm combinations, and the accuracy of each combination is evaluated on all bullseye instances. Further, by designing our overall architecture to be modular, it can readily take advantage of new developments in any of these algorithms in the future.

B. 2D Detection and Recognition

We follow a basic detection-recognition pipeline, but also introduce some heuristic methodologies to help identify the bullseyes; the pipeline is outlined in Figure 1. First, we perform bullseye text detection in 2D on the flattened panoramic images collected in conjunction with the LiDAR point clouds. The normalized binary confusion matrices representing the detection results for each algorithm combination are presented in Figure 3.

Next, we use an OCR engine to identify the characters in the text. Since the bullseyes all observe the same general

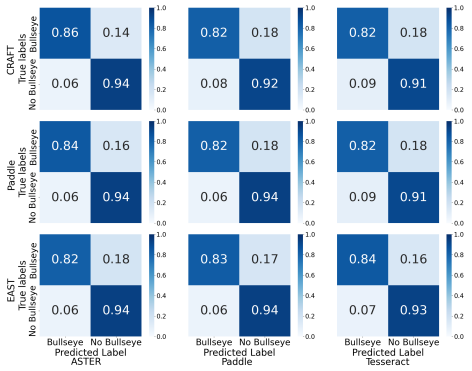


Fig. 3: Normalized binary confusion matrices of the 2D detections for each detector and OCR combination.

format (three lines of alphanumeric text), we use the hierarchical clustering distance method to eliminate false detects or stand alone text that does not meet this known standardized format. Remaining text instances are then fed into a classification model to determine if the detect is a true bullseye or other text in the scene.

However, recognition engines can be prone to error when the viewing angle of the text is distorted; this effect is exacerbated in panoramas. In order to aid the recognition process, we introduce a virtual camera method to consider a synthetic image taken from a more ideal perspective, directly in front of and directly facing the bullseye. This method uses orthographic projection to create a synthetic image that magnifies and flattens the panoramic region at the centroid of each cluster. The flattened region is then passed into the classification model.

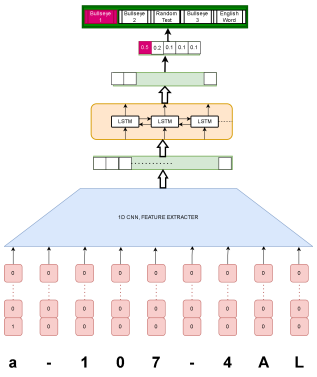


Fig. 4: Architecture of the text classifier.

For the text classifier, we utilize character embeddings instead of word embeddings to encode each character differently – as the text in bullseyes do not form words. To train this network, we created a synthetic dataset containing five categories: random text, English words, first line, second line, and last line of the bullseye. The bullseye dataset is based on standards set by the US Navy [26]. We train a classifier (outlined in Figure 4) to differentiate between bullseyes and unwanted text. Input text first goes through a one-hot-encoding module. Then, the message is embedded into a tensor using our 1D CNN feature extractor to encode the string of characters. The bidirectional LSTM extracts sequence information of the encoded message.

The bullseye's text and 2D center location are also saved and used in the 3D mapping step of Section III-C.

C. 3D Mapping

In this step, the 2D text detections are mapped to the 3D space using a grid mapping algorithm [1]. The grid construction is a one-to-one mapping based on the principle of spherical projection, and allows for the construction of a 2D RGB panoramic image from the 3D point cloud, and vice versa. Since each 2D pixel has a corresponding point in 3D through this grid mapping, we can use the 2D midpoint of a bullseye to find the 3D centroid of its 3D bounding box. The algorithmically determined 3D centroid is compared against the manually annotated ground truth via simple distance metric. The performance of the algorithm combinations using this distance metric are presented in Figure 5.

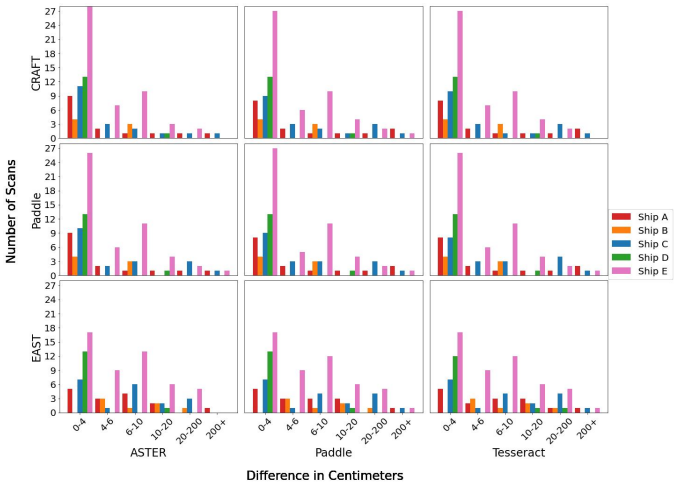


Fig. 5: The 3D distance between prediction and ground truth of bullseye 3D centroids for each of the nine detector/OCR combinations, separated by the five ships.

We crop all points within 0.5m of the centroid as the target point cloud for further processing; this provides a smaller area for the OCR step in Section III-E. This target point cloud's normal vector determines the orientation of the bounding box. Note that we assume bullseyes are flat, as they are on a wall. Plane estimation is performed using Open3D's [27] random sample consensus (RANSAC) algorithm from which the normal vector is then calculated. After plane estimation, we only select points close enough to belong to that plane: within a distance threshold of 2mm. The difference in angle between the algorithmically determined normal vector and the ground truth is calculated for each algorithm combination shown in Figure 6.

In our previous work, we encountered several challenges with the 2D to 3D mapping step which we address in this paper. First, when the 2D bullseye center is mapped to 3D, there is the potential for the mapped point to be empty; e.g. result of a reflective surface, noise, etc. An empty point is automatically assigned by the scanner to have coordinate $(x, y, z) = (0, 0, 0)$, causing our prediction bounding box to move to the origin instead of its correct

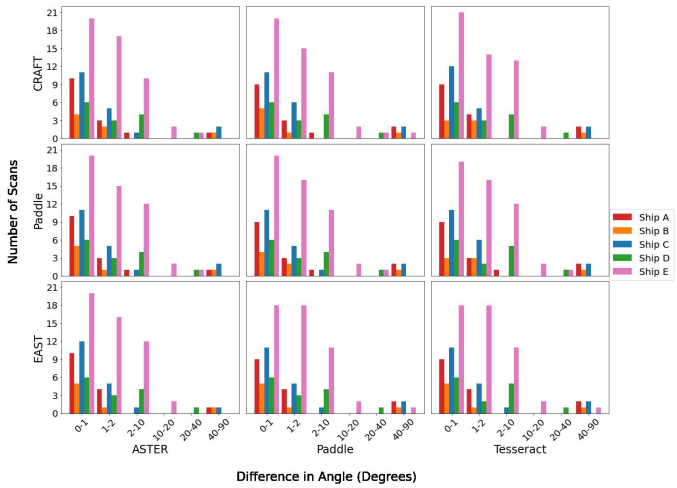


Fig. 6: Angular difference between prediction and ground truth of bullseye normal vectors for each of the nine detector/OCR combinations, separated by the five ships.

location. To solve this issue, all empty points in the grid are first removed, then if the centroid is empty, the next closest non-zero 2D point in the new 3D grid is used instead.

The second challenge is the presence of noisy point clouds; more specifically, noisy data caused by reflective surfaces. This is especially prevalent in modern blue text bullseyes, shown in Figure 7. We expect to find the point cloud of a given section of wall to have a fairly high point density, empirically about $10-30$ points clustered in a 2cm radius; the actual density is dependent on the distance between the LiDAR scanner and the wall. Since noise will typically have a lower density, we are able to determine if the calculated centroid is part of such noise. First, the point cloud is downsampled 100 times; then for each mapping point in 3D, we create a norm ball with a radius of 5cm. If the density of points in that norm ball is less than 10 points, it is assumed to be noise and the process is repeated until the point density exceeds 10.

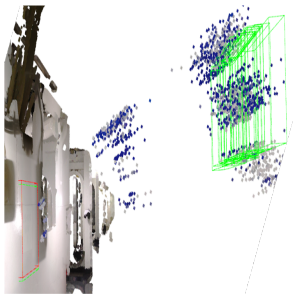


Fig. 7: An illustration of the noise avoidance process. The green boxes capturing the noise will be eliminated. Once the density threshold is met, the correct green box is found near the red ground truth box.

D. Best Combination Selection

The final algorithm is selected by considering each of the three metrics discussed in Section II-E: binary confusion matrix, 3D centroid distance, and normal vector angle difference – Figures 3, 5, and 6 respectively. Although all combinations have strong performance, we determine CRAFT - ASTER to be the strongest across all metrics (see Section IV). Thus it is used to generate the input 3D centroids and normal vectors used in the recognition analysis in Section III-E.

E. Text Recognition and Virtual Camera

Unfortunately, even after the image is flatted and magnified using perspective projection, the text can still be skewed. We address this by using a virtual camera in the 3D domain to create a synthetic image of the area of interest from a new perspective. The point cloud area of interest is first cropped using the centroid, then the virtual camera is placed at a fixed distance from the centroid along the normal vector obtained in Section III-D. Given 3D point world coordinates matrix P_w, rotation matrix R (defined by the target bounding box), and camera location x_C, we can transform the world system to the camera system by $P_c = R*(P_w - x_C)$. The camera now is positioned at the optimal location, looking directly at the bullseye.

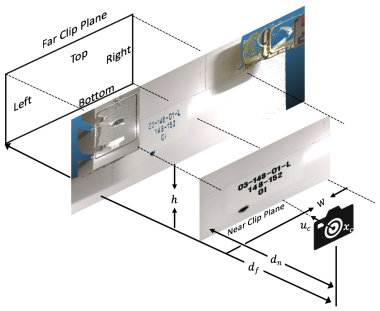


Fig. 8: The virtual camera system has the location of the camera at the origin and the image plane in between the camera and the object of interest.

Finally, we transform the points from the camera frame to the image frame using orthographic projection. Consider the transform matrix T_m shown in Equation 1, the variables d_n and d_f are defined as the distance from the camera to the near and far planes respectively. The resulting coordinates in the image plane are given in Equation 2, where the subscripts i and c denote image frame and camera frame, and variables u and v give the horizontal and vertical grid for each pixel in the synthetic image. Each pixel is then mapped to its associated color pixel, thus generating a flat colorized synthetic image centered on the bullseye. Figure 8 shows the virtual camera system and Figure 9 shows two examples compared raw skewed images and virtual camera produced images.

$$T_m = \begin{bmatrix} 2\dfrac{d_n}{w} & 0 & 0 & 0 \\ 0 & 2\dfrac{d_n}{h} & 0 & 0 \\ 0 & 0 & \dfrac{d_n + d_f}{d_n - d_f} & \dfrac{2d_n d_f}{d_n - d_f} \\ 0 & 0 & -1 & 0 \end{bmatrix} \tag{1}$$

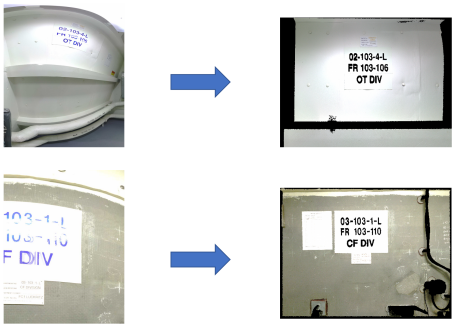


Fig. 9: Distorted images in 2D on the left, synthetic focused images from virtual camera on the right.

$$u = \frac{-x_c}{z_i};\ v = \frac{-y_c}{z_i};\ P_i = T_m P_c$$
$$z_i = \frac{d_n + d_f}{d_n - d_f} z_c + \frac{2 d_n d_f}{d_n - d_f} \quad (2)$$

This image is then sent through the text recognition module to obtain a final text recognition prediction. Let r be the scanner distance from bullseye. We present our results comparing OCR accuracy with and without the virtual camera, when r is smaller or larger than 1.2 meters (Figures 10 and 11 respectively).

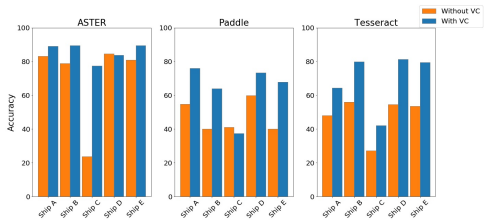


Fig. 10: Accuracy for various OCR engines on original (orange) vs virtual camera (blue) images when $r < 1.2$ m.

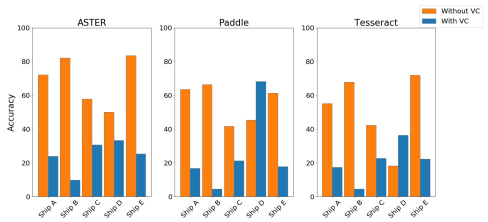


Fig. 11: Accuracy for various OCR engines on original (orange) vs virtual camera (blue) images when $r > 1.2$ m.

IV. Discussion

We present the results of three different comparisons to identify shipboard bullseyes: 2D detection, 3D localization (distance and angle), and 3D text recognition. Figure 3 clearly shows CRAFT-ASTER as the strongest algorithm on 2D detection with both the best true positive and true negative rates of 0.86 and 0.96 respectively. In 3D localization, Figures 5 and 6 qualitatively show this combination as one of the strongest performers; however, the evaluation of multiple ships and bins make the results difficult to distinguish between. In the previous paper [1], we considered a distance < 10cm and < 20cm as *good* and *acceptable*, respectively. Similarly we considered an angular difference of $< 10°$ and $< 20°$ as *good* and *acceptable*, respectively. Combining the distance performance across all ships, CRAFT-ASTER has 93 *good* results which is the highest of any combination. For the angle difference, CRAFT-ASTER had 97 *good* results while the highest of any combination was 99. Performance across the three metrics led us to choose CRAFT-ASTER as the best combination.

We call the method used in this paper 3DTextDLR, and focus the rest of the discussion assuming the choice of the CRAFT-ASTER combination. However, note that 3DTextDLR is modular by design to take advantage of improved state of the art algorthims. 3DTextDLR produces better results over all, compared to our previous paper's method, MP3LTSPC. Particularly in 2D detection, Table I shows that our new method increases accuracy, precision, and recall. This is the foundational step in the process.

TABLE I: Detection – 3DTextDLR vs MP3LTSPC.

Method	Accuracy	Precision	Recall
MP3LTSPC	85.3%	58.4%	68.6%
3DTextDLR	92.7%	76.9%	86.1%

The 3D localization results from the 3D mapping step are shown in Table II. In both evaluation metrics, 3DTextDLR shows tremendous improvement over our previous method. Note that the difference in angle of the normal vector is slightly more accurate than the distance between centroids. We attribute this to the stability of using plane normal estimation for normal vector calculation; i.e., as long as the closest point lives on the plane, the angle measurement is mostly unaffected, unlike the distance metric which uses a single predicted centroid.

TABLE II: 3D Localization (Distance and Angle Difference) – 3DTextDLR vs MP3LTSPC

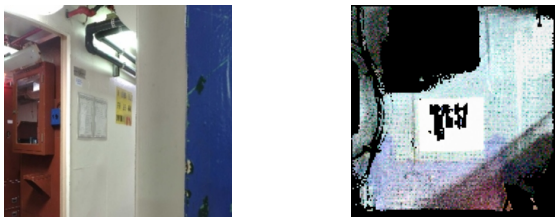
	Distance		Angle	
Metric	MP3LTSPC	3DTextDLR	MP3LTSPC	3DTextDLR
Acceptable	40.6%	94.3%	61.9%	94.3%
Good	26.6%	88.6%	59.0%	92.4%

The text recognition step in the pipeline is itself an improvement over MP3LTSPC; for this, we compare the output of the OCR engine with or without using the virtual camera. The advantage of using the virtual camera is that the object of interest is now the primary focus of the camera system. In addition, the camera specifications can be fully customized for individual scenarios.

After empirically studying the performance of the virtual camera, we noticed a significant drop in performance when the scanner was greater than 1.2 meters away from the bullseye. This was the reason for separately presenting the results for below and above that threshold in Figures 10 and 11, respectively. The former indicates that the virtual camera method often improves performance for cases with short distances, whereas the latter usually results in a

dramatic drop in performance. When the scanner is further from the bullseye, the point cloud is noisier and much less dense, resulting in a more sparse – and hence harder to read – synthetic image. In practice, we suggest using this distance threshold as a heuristic for optionally implementing the virtual camera in the pipeline. As scanners generate increasingly more dense point clouds, it is possible this threshold will increase.

We encountered a few edge cases that were challenging for the 3DTextDLR method and need to be explored further. First, consider the case when a bullseye is highly skewed in a panoramic image, shown in Figure 12a. The location of the bullseye could not accurately be detected due to the pronounced angle and blurred text. Additionally, bullseyes can be very reflective. This can cause significant distortion, making detection and recognition extremely difficult. In some cases, our 2D detection step misclassified various non-bullseye text instances located within the ship, resulting in lower localization and recognition scores. Finally, the effectiveness of the virtual camera is highly dependent on the distance between the scanner and the bullseye: the greater the distance, the more sparse the point cloud, and thus, the more distorted and unrecognizable the text becomes – Figure 12b for example.



(a) The detector could not detect the skewed bullseye.

(b) Virtual camera produces an unrecognizable bullseye when scanner is far.

Fig. 12: Failure Cases

A. Future Work

We intend to continue to improve the 3DTextDLR method by expanding our methodology to address the edge cases mentioned in Section IV. We plan to explore techniques to improve the virtual camera, specifically by merging multiple scans which would result in a point cloud that is more dense, clean, and robust. In addition, we will modify the extrinsic and intrinsic parameters of the virtual camera for specific cases.

We also plan to explore the use of synthetic images to train an end-to-end deep neural network for detecting the 2D locations, rather than the current two step process: text detector followed by a bullseye classifier. This would allow us to have full control over our dataset, while having the added advantage of not needing to manually label each image.

REFERENCES

[1] A. Mai, M. Bilinski, and R. Provost, "Method to perform 3d localization of text in shipboard point cloud data using corresponding 2d image," in *2020 IEEE Eighth International Conference on Communications and Electronics (ICCE)*. IEEE, 2021, pp. 433–438.

[2] Y. Du, C. Li, R. Guo, X. Yin, W. Liu, J. Zhou, Y. Bai, Z. Yu, Y. Yang, Q. Dang *et al.*, "Pp-ocr: A practical ultra lightweight ocr system," *arXiv preprint arXiv:2009.09941*, 2020.

[3] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang, "East: an efficient and accurate scene text detector," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 5551–5560.

[4] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9365–9374.

[5] "Tesseract-ocr," 2021. [Online]. Available: https://github.com/tesseract-ocr/

[6] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai, "Aster: An attentional scene text recognizer with flexible rectification," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 9, pp. 2035–2048, 2018.

[7] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," *IEEE access*, vol. 7, pp. 128 837–128 868, 2019.

[8] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4490–4499.

[9] T. A. Le, A. G. Baydin, R. Zinkov, and F. Wood, "Using synthetic data to train neural networks is model-based reasoning," *arXiv.org*, Mar 2017. [Online]. Available: https://arxiv.org/abs/1703.00868

[10] C. M. Ward, J. Harguess, and C. Hilton, "Ship classification from overhead imagery using synthetic data and domain adaptation," in *OCEANS 2018 MTS/IEEE Charleston*. IEEE, 2018, pp. 1–5.

[11] B. Hurl, K. Czarnecki, and S. Waslander, "Precise synthetic image and lidar (presil) dataset for autonomous vehicle perception," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 2522–2529.

[12] S. Long, "Scene text detection and recognition: The deep learning era," *International Journal of Computer Vision*, 2021.

[13] M. Jalal, J. Spjut, B. Boudaoud, and M. Betke, "Sidod: A synthetic image dataset for 3d object pose recognition with distractors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[14] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Synthetic data and artificial neural networks for natural scene text recognition," *arXiv preprint arXiv:1406.2227*, 2014.

[15] T. R. Shaham, T. Dekel, and T. Michaeli, "Singan: Learning a generative model from a single natural image," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4570–4580.

[16] C. Mediavilla, "GAN-based unpaired image-to-image translation for maritime imagery," in *Geospatial Informatics X*. SPIE, 2020.

[17] L. A. Gatys, "Image style transfer using convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[18] G. Csurka, "Domain adaptation for visual applications: A comprehensive survey," *arXiv preprint arXiv:1702.05374*, 2017.

[19] C. Hilton, J. Berk, and S. Parameswaran, "Maritime LOD balancing: evaluating the effect of level of detail on ship classification," in *Geospatial Informatics X*, P. J. Doucette, J. D. Harguess, K. Palaniappan, and G. Seetharaman, Eds., vol. 11398, International Society for Optics and Photonics. SPIE, 2020, pp. 106 – 115. [Online]. Available: https://doi.org/10.1117/12.2561723

[20] M. Liao, B. Song, S. Long, M. He, C. Yao, and X. Bai, "Synthtext3d: synthesizing scene text images from 3d virtual worlds," *Science China Information Sciences*, vol. 63, no. 2, pp. 1–14, 2020.

[21] A. U. M. Masuk, A. Salibindla, and R. Ni, "A robust virtual-camera 3d shape reconstruction of deforming bubbles/droplets with additional physical constraints," *International Journal of Multiphase Flow*, vol. 120, p. 103088, 2019.

[22] H. Qiu, "Synface: Face recognition with synthetic data," in *Proceedings of the IEEE/CVF International Conf on Computer Vision*, 2021.

[23] R. Gross, S. Baker, I. Matthews, and T. Kanade, "Face recognition across pose and illumination," in *Handbook of face recognition*. Springer, 2005, pp. 193–216.

[24] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[25] "Software download, installation, and release notes for scene," [online], 2019, https://knowledge.faro.com/ [Feb2022].

[26] US Navy, "Compartment damage control marking guide."

[27] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.