JOURNAL OF
EMERGING INVESTIGATORS

Article

Comparing neural networks with a traditional method for identifying the vanishing points of surgical tools

Bhavani Venkatesan[1], Yun-Hsuan Su[2]

[1] International School of Hyderabad, Hyderabad, Telangana, India

[2] Department of Computer Science, Mount Holyoke College, South Hadley, Massachusetts

SUMMARY

Robot-assisted minimally invasive surgery (RMIS) offers numerous advantages, such as higher precision and shorter patient recovery time, leading to more successful outcomes. Force feedback from the surgical tool is essential for the surgeon to control the pressure applied. Recently, computer vision techniques have been explored as an alternative to expensive sensors for force feedback. A crucial step is identifying the vanishing point of the surgical tool, where the two edges of the tool seem to converge, but this becomes challenging when the tool is obstructed by tissue (occlusion). We hypothesized that neural networks could achieve higher accuracy than traditional techniques such as the minimum area enclosing triangle method in finding the vanishing point, particularly with occlusion. To test our hypothesis, we trained a neural network for vanishing point detection using datasets of synthetic images of surgical tools with and without occlusion, and then tested the triangle method on the same datasets. We then evaluated and compared their performances. The results partially supported the hypothesis as the neural network performed much better than the triangle method for unjointed tools, both with and without occlusion. For jointed tools, the triangle method surprisingly performed better than the neural network, although both methods require improvement in accuracy. Overall, the neural network's consistency regardless of the level of occlusion, as compared to the triangle method's poor performance when there is occlusion, demonstrates the neural network's potential to improve computer vision for force feedback in RMIS, leading to better outcomes for patients.

INTRODUCTION

Robot-assisted minimally invasive surgeries (RMIS) are becoming increasingly popular, with applications such as coronary artery bypass and cutting away cancer tissues (1). Over 12 million robot-assisted surgeries have been performed, with advancing technologies having significant advantages, such as increased stability and precision, which leads to shorter recovery time and less medication for patients (2, 3). For these micro-surgical procedures, force feedback is essential to accurately position the instrument and help the surgeons by improving dexterity and tactile sensitivity (4). Without this, there could be unintentional tissue damage if too much force is applied. However, one major drawback is that there are limited technologies to provide haptic (force and tactile) feedback to the surgeon (5). While force sensors can be used, they need to be biocompatible and sterilizable or be disposed of after a single use. This makes these sensors very expensive, costing up to $500 for certain procedures, and therefore, makes them an unviable solution (6-9).

A new approach that has been suggested is to provide force feedback using computer vision (10). Computer vision here refers to semantic segmentation, which is where the image pixels that make up the tool are differentiated from the rest of the image to locate the tool in the image. The deformation of tissues around the tool tip is then used to predict the force being applied (11). Semantic segmentation of the tool is shown to be more effective when the image is transformed to polar coordinates with the tool's vanishing point as the center (12). Therefore, identifying the vanishing point of the surgical tool, where the two edges of the tool seem to converge, is a critical part of the process (**Figure 1**). The main method reported to identify vanishing points in surgical fields is the minimum area enclosing triangle method, but it has some drawbacks and limitations, as discussed below.

This method uses the minimum area triangle enclosing the tool and uses the triangle's vertex that is closest to the image center to estimate the vanishing point (12). This method is expected to work well primarily for unoccluded images, as the algorithm for identifying the minimum-area triangle requires the tool's edges to be clearly identifiable (12). However, if there are multiple tools or tools with joints, the triangle will be drawn around all the tools and joints, which may lead to less accurate results, giving this method specific and limited uses. Currently, there are a few neural networks which have been trained to find the vanishing point in natural scenes (roads, buildings, etc.), but no neural networks have been applied to surgical tools to date. Results from various studies show that neural networks were more effective than traditional methods of identifying the vanishing point in natural scenes (13, 14). During surgeries, the images of tools are often obstructed due to tissue and blood; this is called occlusion. Any method to identify the vanishing point must be able to do so with occluded images. Moreover, it is also unclear whether neural networks would perform better than the triangle method in different cases, such as when the images are jointed and potentially occluded.

However, neural networks seem to be the most promising as they can learn and make sophisticated human-like decisions because of their consecutive, interconnected layers inspired by neuroscientific theories, something which the other method cannot do (15). Therefore, we hypothesized that a neural network could achieve equally high or higher accuracy

https://doi.org/10.59720/24-115

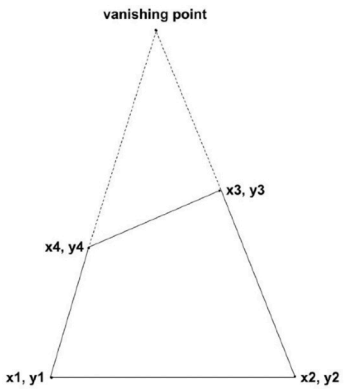


Figure 1: Visual representation of how the vanishing point is determined. The line connecting the points (x1,y1) and (x4,y4) forms the left edge of the tool, and the line connecting the points (x2,y2) and (x3,y3) forms the right edge of the tool. The point where the lines converge is the predicted vanishing point.

than the minimum area enclosing triangle method in finding the vanishing point of a surgical tool, both with and without occlusion. We test this hypothesis by modifying and training a pre-existing model for environmental landscapes on four sets of surgical tool images to see how it adapts to environments within the human body, and comparing its performance with that of the triangle method.

As there were no pre-existing datasets available, artificial datasets were created to model the tools by writing a program to generate thousands of randomized surgical images. To simulate occlusion, random blobs were generated to cover parts of the tool (**Figure 2**).

The NeurVPS Conic Convolution Neural Network for natural scenes was trained on both unoccluded and occluded images of unjointed and jointed surgical tools, and it performed well in identifying the vanishing point even with occlusion (14). The triangle method was implemented based on the method described by Huang et al. (12). The neural network performed better for unjointed tools, however the triangle method performed better for jointed tools. Both methods had high angle errors for jointed tools, however, representing an area for further improvement. Overall, neural networks have the potential to overcome the challenge of occlusion, showing promise for improving force feedback and reducing the cost of robot-assisted surgery, enabling its adoption and leading to better outcomes for patients.

RESULTS

The angle error graphs for each method (**Figures 3a, 4a, 5a, 6a**) show the cumulative distribution function of this error. The percentile rank on the *y* axis shows the percentage of images that have an angle error less than or equal to the corresponding value on the *x* axis. This means that the angle error graph of a well-performing model would have a steep slope at first and flatten out quickly, showing that a high proportion of predictions have a small error.

Neural Network

Testing the untrained model on unjointed unoccluded tool images resulted in a low accuracy (90th percentile of 81°). After training it on the datasets, the accuracy improved by a lot and the model performed very similarly on unoccluded and occluded unjointed tools, with a 90th percentile angle error of 18-18.5°, and the CDF graph reaches 1 around 21° showing high accuracy (**Figure 3a, 4a**). However, the model performed poorly on unoccluded and occluded jointed tools, with a 90th percentile angle error of 62.5-64°, and the CDF reached 1 past 70° showing very low accuracy (**Figure 5a, 6a**). The difference between the model's performance on unjointed and jointed tools is also highlighted by the epoch loss during training (described in Materials and Methods), which was 3.5 for unjointed tools and 4.5-4.6 for jointed tools, showing that the model is better suited to the former. The model's performance is also shown by the sample output images for each dataset (**Figures 3b, 4b, 5b, 6b**).

Minimum Area Enclosing Triangle

For unoccluded unjointed tools, the 90th percentile of the angle error was 40°, which is only a bit lower than that for occluded unjointed tools (44°). Surprisingly, the result for unoccluded jointed tools was better than for unjointed tools, as the 90th percentile was 27°. However, this method still did not work that well on occluded jointed tools (90th percentile: 54°), although it did better than the neural network.

The results of both methods (training neural network and minimum area enclosing triangle) are summarized in **Table 1**.

DISCUSSION

Our data gives a better understanding of the strengths and weaknesses of neural networks, the minimum area triangle method, and their effectiveness in identifying the vanishing points of surgical tools. As previously mentioned, surgeons must receive clear haptic feedback to minimize tissue damage during RMIS. The location of the tool tip is required for this, and accurately identifying the tool's vanishing point is a crucial step in this process, underscoring the need for an effective method.

The neural network performed better than the triangle method for unjointed tools, both with and without occlusion. The 50th percentile of the angle error was similar for both methods, but the 90th percentile for the triangle method was more than double that of the neural network, showing that the neural network is much more reliable. Most importantly, the neural network performed as well on the occluded dataset as it did on the unoccluded dataset, showing its potential to identify the vanishing point in realistic images of surgical tools and the consistency in its results. On the other hand, the triangle method performed worse on the occluded dataset as

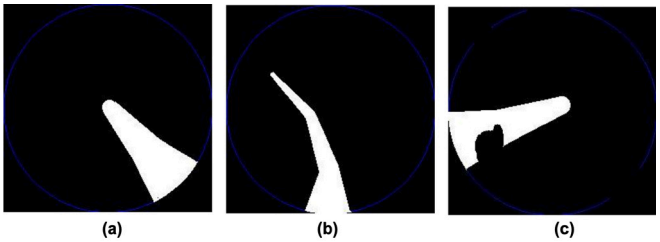


Figure 2: Sample images from three of the four artificial datasets that were created to simulate surgical tools. (a) Tool with a single joint. (b) Tool with two joints where each quadrilateral is smaller than the previous one. (c) Tool with some occlusion.

https://doi.org/10.59720/24-115

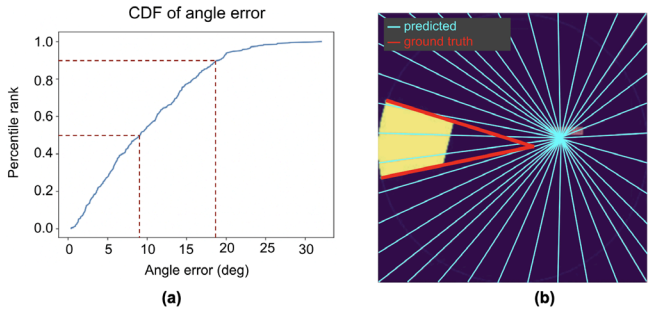


Figure 3: Results for the neural network's performance on unoccluded unjointed tools. (a) Cumulative distribution function (CDF) of the angle error for unoccluded unjointed surgical tools. (b) A sample image of the predicted vanishing point.

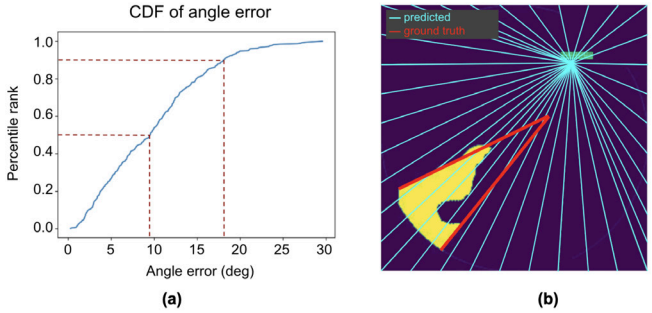


Figure 4: Results for the neural network's performance on occluded unjointed tools. (a) Cumulative distribution function (CDF) of the angle error for occluded unjointed surgical tools. (b) A sample image of the predicted vanishing point.

compared to the unoccluded one, which indicates that neural networks are more useful when occlusion may be involved.

For unoccluded jointed tools, the neural network performed more poorly than it did on unjointed tools with angle errors above 30° for the 90th percentile. The neural network performed very similarly for occluded jointed tools as well, once again confirming that it is not affected much by occlusion. Since the triangle method draws a triangle around the *entire* tool, it was expected that it would perform poorly with jointed tools where each segment of the tool points in a different direction. However, the results were surprising as the triangle method performed better on unoccluded jointed tools than it did on unjointed tools, as the 90th percentile of the angle error was more than 10° lower for unoccluded jointed tools. This was also substantially better than the neural network, with a 90th percentile angle error which was less than half that of the neural network. As expected, the triangle method performed much more poorly on occluded jointed tools, with the 90th percentile of the angle error being twice that of unoccluded jointed tools.

Overall, these results indicate that while the minimum area enclosing triangle method works better for jointed tools, the neural network handles occlusion better. However, both methods still have high angle errors for jointed tools and need further improvement.

Some approaches that were tried to improve the accuracy of the neural network on jointed tools included inverting the image colors, changing the learning rate and hyperparameters, and removing the image circle outline, but when applied, these did not improve the model's performance. However, since some of the images showed good prediction of the vanishing point, the model could potentially be improved by training it on more images and for more epochs.

The main limitation in this experiment is that artificial images were used instead of real-life images taken during surgery, due to the lack of available data. While the images generated by the program are a fair representation of the surgical tool, it is still a relatively simplified version compared to the real images that the model would have to work with. It is also possible that there was not enough variety in the dataset because of the train/test percentages, causing the model to overfit. Finally, the program used for the triangle method provided a more favorable result than it would in real-life due to the assumption that the coordinates of the vanishing point are already known, but this is not necessarily a limitation as the neural network nonetheless performed better than the triangle method for unoccluded images, despite this advantage. Nevertheless, these results show promise for both methods in the future with further improvements.

In conclusion, the neural network performed much better than the minimum area enclosing triangle method in identifying the vanishing point of unjointed tools, both with and without occlusion, partially supporting our hypothesis. However, for both unoccluded and occluded jointed tools, the triangle method worked better than the neural network. One potential suggestion would be to utilize a method that combines neural networks and the triangle method. This may optimize the accuracy of the vanishing point prediction as the triangle method performs better on jointed tools while the neural network is more consistent regardless of the level of occlusion. Overall, neural networks have a lot of potential to enable the application of computer vision to estimate force feedback as their performance is not impaired by occlusion, helping them overcome a significant challenge in the adoption of RMIS. This will be very useful in practical applications as real-life images are often messy and not very clear.

In the future, the neural network could also be trained on a dataset that combines the four datasets (unoccluded and occluded jointed and unjointed tools) to see if this improves the accuracy. More complex scenarios could also be modeled using datasets with multiple tools in each image. The neural network would need to be modified to give multiple vanishing point coordinates as the output, based on the number of tools in the image. A layer of complexity could be added by having tools which overlap each other, so the model would have to learn to differentiate the tools. Further, to simulate the motion of tools in real-life surgeries, a dataset with continuous motion could be created. In each consecutive image, the tool would be displaced by a few pixels in a given direction. When the series of images is viewed side-by-side, it would appear that the tool follows a random path within the image circle. These improvements could help make neural networks more accurate, allowing them to fully replace the minimum triangle method.

MATERIALS AND METHODS

Dataset

A Python program was written to create a dataset of

https://doi.org/10.59720/24-115

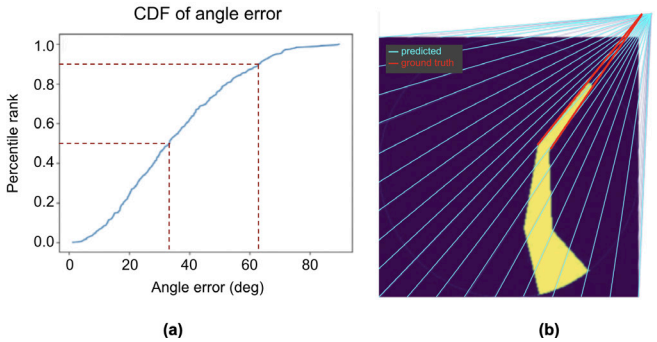


Figure 5: Results for the neural network's performance on unoccluded jointed tools. (a) Cumulative distribution function (CDF) of the angle error for jointed surgical tools. (b) A sample image of the predicted vanishing point.

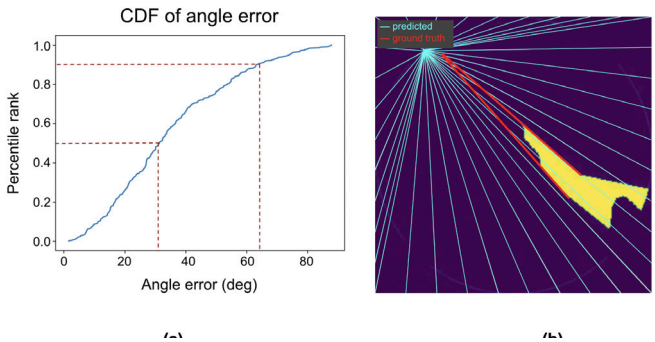


Figure 6: Results for the neural network's performance on occluded jointed tools. (a) Cumulative distribution function (CDF) of the angle error for jointed surgical tools. (b) A sample image of the predicted vanishing point.

10,000 images with randomly generated surgical tool images and ground truth masks with the corresponding vanishing point due to a lack of published datasets including surgical field vanishing points. The dataset was created using the ImageDraw module from Python's 'pillow' library (Python v3.11.4). The content from endoscopic cameras is limited to a circular area since the image sensor is usually larger than the image circle of the endoscope, so the tools in the dataset were confined within a circle (16).

The dataset was divided into four subsets based upon the following categories: unjointed tools without occlusion, unjointed tools with occlusion, jointed tools without occlusion, and jointed tools with occlusion. The datasets with occlusion have black blobs covering parts of the tool to simulate the occlusion caused by tissues and blood in real life. All datasets were split into 96% for training, 4% for testing.

To create the tool shape, the point from which the tool appears was randomly selected along the bottom half of the image circle, as this is generally the case in real life. The second point was chosen randomly to the right of the first point within a range of π/9 radians to π/5 radians. This forms the bottom edge of the first quadrilateral. To create the rest of the tool, two points were randomly chosen to create a top edge. This process was then repeated with the top edge of the first quadrilateral being the bottom edge of the second quadrilateral and so on to create two or three quadrilaterals to form a jointed tool (**Figure 2b**). Finally, a small semi-circle was drawn at the tip of the last quadrilateral to make the tool shape look realistic. To avoid any biases, all parameters were chosen randomly within given ranges. Note that the coordinates are chosen in an anti-clockwise order, with the first coordinate being at the bottom left (**Figure 1**). A GitHub repository has been created with the code for the dataset (17).

To generate the top edge of a quadrilateral, a random point was chosen along the perpendicular bisector (within the range of 45.0 to 85.0 pixels) of the bottom edge. A random length between 40% and 60% of the base length was chosen for the length of the upper edge. A line of this length was drawn parallel to the base, with the point on the perpendicular bisector as its center, forming a trapezium. This line was then rotated by a random angle between -π/5–π/5 radians. This forms the upper edge of the quadrilateral. This method can be easily generalized for any number of quadrilaterals using the top edge of the first quadrilateral as the base for the second one. This is the method that was finally used to generate a dataset of 10,000 images.

In real life, the tool image is often occluded due to tissues covering the tool. The occluded dataset was generated by creating blobs on the image to cover parts of the tool. The code for creating the blobs was generated based on code found on Stack Overflow (18). The 'seedval' and 'threshold' parameters were randomized within a range to vary the amount of occlusion created by the blobs, making the occlusion more realistic (**Figure 2**).

For the ground truth masks, the neural network expected the coordinates of the vanishing point with respect to the image center, not an image with the vanishing point. Therefore, a '.txt' file with the coordinates was given for each image. A third value was also required in the '.txt' file: the focal length. However, this value can only be found based on the camera parameters, and this information is not available, so it was set to a common value of 1.0. The vanishing point is the point of intersection of the lines forming the two sides of the tool's final quadrilateral and its coordinates are calculated as explained below (**Figure 1**).

To find the point of intersection of the two lines, we must first find the equations of the lines by calculating their respective slopes and *y* intercepts.

m_l = slope of the line connecting (x_1, y_1) and (x_4, y_4)

m_r = slope of the line connecting (x_2, y_2) and (x_3, y_3)

$c_l = y$ intercept of the line connecting (x_1, y_1) and (x_4, y_4)

$c_r = y$ intercept of the line connecting (x_2, y_2) and (x_3, y_3)

$$m_l = \frac{y_4 - y_1}{x_4 - x_1} \qquad m_r = \frac{y_3 - y_2}{x_3 - x_2}$$

$$y_1 = m_l \cdot x_1 + c_l \qquad y_2 = m_r \cdot x_2 + c_r$$

At the point where the two lines intersect, we know that $y_1 = y_2$. We can use this to solve for the coordinates of the point of intersection, which is the vanishing point (x_v, y_v):

$$m_l . x_v + c_l = m_r . x_v + c_r$$

$$\therefore \; x_v = \frac{c_l - c_r}{m_r - m_l}$$

$$y_v = m_l . x_v + c_l$$

To find the coordinates of the vanishing point with respect to the center, the coordinates of the image center were sub-

https://doi.org/10.59720/24-115

Dataset	Neural Network			Minimum Area Enclosing Triangle	
	Epoch loss	50th percentile	90th percentile	50th percentile	90th percentile
Unoccluded Unjointed	3.4	9°	18.5°	7°	40°
Occluded Unjointed	3.5	9.5°	18°	9.5°	44°
Unoccluded Jointed	4.5	33°	62.5°	7°	27°
Occluded Jointed	4.6	31°	64°	11°	54°
Pre-trained model on unoccluded unjointed	–	52°	81°	–	–

Table 1: Statistical results for the neural network's and minimum area enclosing triangle method's performance on all four datasets. The 50th and 90th percentiles of angle error for both methods and mean training loss for neural networks when tested on unjointed and jointed tools, both with and without occlusion. The 50th and 90th percentiles for the pre-trained model's angle error are also included for the neural network.

tracted from the coordinates of the vanishing point.

Neural Network

The neural network used in this paper is an "end-to-end trainable deep network", which uses geometry-inspired convolutional operators to detect the vanishing points (14). This uses a novel approach involving conic convolution to extract features like structural lines. This model was chosen because it is expected to be more accurate as it has been created specifically for identifying vanishing points, although in a different context of natural scenes. A summary of the neural network's configuration is shown in the Appendix.

The code used for the neural network requires a CUDA enabled GPU, which was not available on the local device. Therefore, the code was transferred from the PyCharm IDE on a Mac OS laptop to Google Colaboratory, where a T4 GPU was available. The original code uses two GPUs, but only one GPU was used for this paper due to hardware limitations.

The source code for the neural network was written to accept one of three specific datasets: Wireframe dataset, ScanNet dataset, or Tmm17 dataset. The 'datasets.py' file contained three classes which were written specifically to process each of these datasets. Since the dataset of surgical tools has a different format and structure, a new class was written to process this data and change it from JPEG images to the tensor format required for the neural network.

To determine the performance of the neural network, a binary cross entropy loss function was used to determine the performance of the neural network. It tracks incorrect labeling and penalizes the model for confident but incorrect predictions (19). If the log loss value is low, it means that the model's accuracy is high. The formula represents this loss function, where y is the ground truth value and p is the probability that the predicted value for the vanishing point is correct, i.e., in the positive class (19).

$$\text{Binary cross entropy} = -[y * \log(p) + (1 - y) * \log(1 - p)]$$

The NeurVPS Conic Convolution Neural Network, which was trained on the Tmm17 (Transactions on Multimedia) dataset of natural scene images, was first tested on surgical tool images, using a dataset of unjointed surgical tools for evaluation (14). The model was then trained and tested with four different datasets: unjointed unoccluded tool, unjointed occluded tools, jointed unoccluded tools, and jointed occluded tools. For each experiment, the neural network was trained for 2 epochs (i.e, 2 passes through the dataset) on a dataset of 10,000 images, which was split into 96% for training and 4% for testing. The higher proportion for training was chosen to improve the model's accuracy by maximizing training data availability.

During evaluation, the angle error was calculated for the results using both methods of finding the vanishing point. The angle error is the difference between the angle to the ground truth vanishing point and the angle to the predicted vanishing point, with respect to a common reference point.

Minimum Area Enclosing Triangle Method

To obtain the minimum area triangle (12), the vertices of the tool first need to be obtained. The program for this process was created by building on the code from an online source (20) and using the *cv2.findContours* function to get the boundary points of the tool shape. These points were then used as input to the *cv2.minEnclosingTriangle* function, which returns the area and vertices of the minimum area enclosing triangle (**Figure 7**).

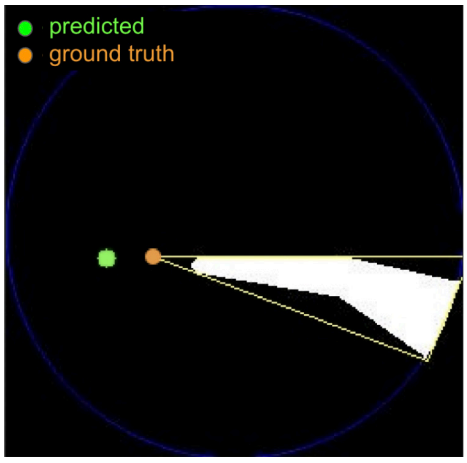


Figure 7: The ground truth vanishing point of a jointed tool and the vanishing point predicted using the minimum area enclosing triangle. The yellow triangle is the minimum area enclosing triangle. The green point shows the ground truth vanishing point and the orange point shows the vanishing point predicted using the triangle method.

https://doi.org/10.59720/24-115

The vanishing point is obtained by choosing the triangle vertex that is closest to the image center if at least one vertex is within the image circle, and in the case where all the vertices are outside the image circle, it suggests choosing the point on the circumference that coincides with the line joining the tool tip and the triangle vertex nearest to it. However, in this paper, the process was slightly simplified for the case where the triangle vertices are outside the image circle. In this case, the vertex closest to the ground truth vanishing point was chosen as the prediction. This could be done as the dataset for training the neural network was created manually, so the ground truth vanishing point is already known.

ACKNOWLEDGMENTS

Bhavani Venkatesan would like to thank the Pioneer Research Program for facilitating this research and giving her the opportunity to work with Professor Su.

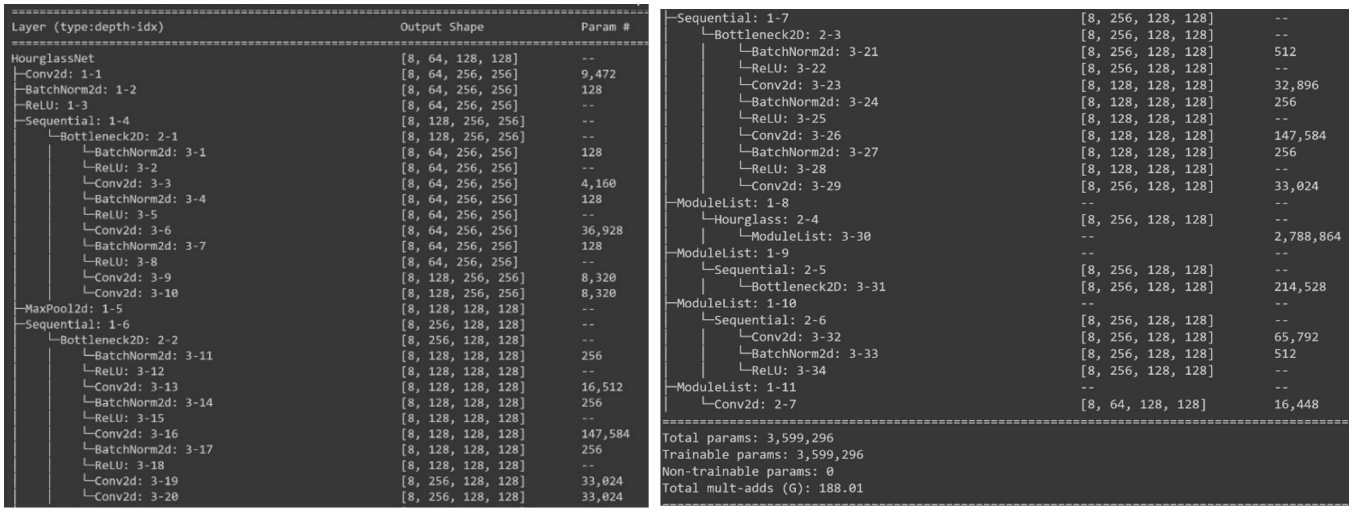
Received: January 22, 2024
Accepted: August 17, 2024
Published: March 28, 2025

REFERENCES

1. "Robotic surgery." *National Library of Medicine, MedlinePlus.* medlineplus.gov/ency/article/007339.htm. Accessed 3 Nov. 2024.
2. "Robotic Surgery Is Here to Stay—and So Are Surgeons." *American College of Surgeons.* www.facs.org/for-medical-professionals/news-publications/news-and-articles/bulletin/2023/may-2023-volume-108-issue-5/robotic-surgery-is-here-to-stay-and-so-are-surgeons/. Accessed 3 Nov. 2024.
3. Palep, Jaydeep H. "Robotic assisted minimally invasive surgery." *Journal of Minimal Access Surgery*, vol. 5, no. 1, Jan-Mar. 2009, pp. 1-7. https://doi.org/10.4103/0972-9941.51313.
4. Wagner, Christopher R., et al. "The role of force feedback in surgery: analysis of blunt dissection." *Proceedings 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems,* 24-25 Mar. 2002, Orlando. https://doi.org/10.1109/HAPTIC.2002.998943.
5. Okamura, Allison M. "Haptic feedback in robot-assisted minimally invasive surgery." *Current Opinion in Urology*, vol.19, no. 1, Jan. 2009, pp. 102-7. https://doi.org/10.1097/MOU.0b013e32831a478c.
6. Park, Cheol Hee, and Sang Jun Song. "Sensor-Assisted Total Knee Arthroplasty: A Narrative Review." *Clinics in Orthopedic Surgery,* vol. 13, no. 1, Feb. 2021, pp. 1-9. https://doi.org/10.4055/cios20034.
7. Enayati, Nima, et al. "Haptics in robot-assisted surgery: Challenges and benefits." *IEEE Rev Biomed. Eng.*, vol. 9, Mar. 2016, pp. 49-65. https://doi.org/10.1109/RBME.2016.2538080.
8. Amirabdollahian, Farshid, et al. "Prevalence of haptic feedback in robot-mediated surgery: a systematic review of literature." *Journal of Robotic Surgery*, vol. 12, no. 1, Mar. 2018, pp. 11-25. https://doi.org/10.1007/s11701-017-0763-4.
9. Muscolo, Giovanni G. and Paolo Fiorini. "Force-Torque sensors for minimally invasive surgery robotic tools: An overview." *IEEE Transactions on Medical Robotics and Bionics*, vol. 5, no. 3, Aug. 2023, pp. 458-71. https://doi.org/10.1109/TMRB.2023.3261102.
10. Huang, Kevin, et al. "Characterizing limits of vision-based force feedback in simulated surgical tool-tissue interaction." *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, 20-24 Jul. 2020, Montreal. https://doi.org/10.1109/EMBC44109.2020.9176658.
11. Ko, Dae-Kwan, et al. "Vision-based interaction force estimation for robot grip motion without tactile / force sensor." *Expert Systems with Applications*, vol. 211, Jan. 2023. https://doi.org/10.1016/j.eswa.2022.118441.
12. Huang, Kevin, et al. "Surgical tool segmentation with pose-informed morphological polar transform of endoscopic images." *Journal of Medical Robotics Research*, vol. 7, no. 2 & 3, Jun. 2022. https://doi.org/10.1142/S2424905X22410033.
13. Borji, Ali. "Vanishing point detection with convolutional neural networks." *arXiv preprint*, Sep. 2016. https://doi.org/10.48550/arXiv.1609.00967.
14. Zhou, Yichao, et al. "Neurvps: Neural vanishing point scanning via conic convolution." *Advances in Neural Information Processing Systems*, vol. 32, Mar. 2021. https://doi.org/10.48550/arXiv.1910.06316.
15. Taherdoost, Hamed. "Deep Learning and Neural Networks: Decision-Making Implications." *Symmetry 15*, vol.15, no. 9, Sep. 2023. https://doi.org/10.3390/sym15091723.
16. Huang, Kevin, et al. "Enhanced U-Net Tool Segmentation using Hybrid Coordinate Representations of Endoscopic Images." *2021 International Symposium on Medical Robotics (ISMR)*, 17-19 Nov. 2021, Atlanta. https://doi.org/10.1109/ISMR48346.2021.9661519.
17. "vpd_data." *GitHub*. github.com/bhavaniv1101/vpd_data. Accessed 20 Nov. 2023.
18. "Is it possible to create a random shape on an image in python?" *Stack Overflow*. stackoverflow.com/questions/71865493/is-it-possible-to-create-a-random-shape-on-an-image-in-python. Accessed 26 Jul. 2023.
19. "Understanding Binary Cross-Entropy and Log Loss for Effective Model Monitoring." *Aporia.* www.aporia.com/learn/understanding-binary-cross-entropy-and-log-loss-for-effective-model-monitoring/. Accessed 25 Jul. 2024.
20. "Find Co-ordinates of Contours Using OpenCV | Python." *GeeksforGeeks*. www.geeksforgeeks.org/find-co-ordinates-of-contours-using-opencv-python/. Accessed 5 Mar. 2024.

https://doi.org/10.59720/24-115

APPENDIX



```
==========================================================================
Layer (type:depth-idx)                             Output Shape              Param #
==========================================================================
HourglassNet                                       [8, 64, 128, 128]         --
├─Conv2d: 1-1                                      [8, 64, 256, 256]         9,472
├─BatchNorm2d: 1-2                                 [8, 64, 256, 256]         128
├─ReLU: 1-3                                        [8, 64, 256, 256]         --
├─Sequential: 1-4                                  [8, 128, 256, 256]        --
│    └─Bottleneck2D: 2-1                           [8, 128, 256, 256]        --
│    │    └─BatchNorm2d: 3-1                       [8, 64, 256, 256]         128
│    │    └─ReLU: 3-2                              [8, 64, 256, 256]         --
│    │    └─Conv2d: 3-3                            [8, 64, 256, 256]         4,160
│    │    └─BatchNorm2d: 3-4                       [8, 64, 256, 256]         128
│    │    └─ReLU: 3-5                              [8, 64, 256, 256]         --
│    │    └─Conv2d: 3-6                            [8, 64, 256, 256]         36,928
│    │    └─BatchNorm2d: 3-7                       [8, 64, 256, 256]         128
│    │    └─ReLU: 3-8                              [8, 64, 256, 256]         --
│    │    └─Conv2d: 3-9                            [8, 128, 256, 256]        8,320
│    │    └─Conv2d: 3-10                           [8, 128, 256, 256]        8,320
├─MaxPool2d: 1-5                                   [8, 128, 128, 128]        --
├─Sequential: 1-6                                  [8, 256, 128, 128]        --
│    └─Bottleneck2D: 2-2                           [8, 256, 128, 128]        --
│    │    └─BatchNorm2d: 3-11                      [8, 128, 128, 128]        256
│    │    └─ReLU: 3-12                             [8, 128, 128, 128]        --
│    │    └─Conv2d: 3-13                           [8, 128, 128, 128]        16,512
│    │    └─BatchNorm2d: 3-14                      [8, 128, 128, 128]        256
│    │    └─ReLU: 3-15                             [8, 128, 128, 128]        --
│    │    └─Conv2d: 3-16                           [8, 128, 128, 128]        147,584
│    │    └─BatchNorm2d: 3-17                      [8, 128, 128, 128]        256
│    │    └─ReLU: 3-18                             [8, 128, 128, 128]        --
│    │    └─Conv2d: 3-19                           [8, 256, 128, 128]        33,024
│    │    └─Conv2d: 3-20                           [8, 256, 128, 128]        33,024
├─Sequential: 1-7                                  [8, 256, 128, 128]        --
│    └─Bottleneck2D: 2-3                           [8, 256, 128, 128]        --
│    │    └─BatchNorm2d: 3-21                      [8, 256, 128, 128]        512
│    │    └─ReLU: 3-22                             [8, 256, 128, 128]        --
│    │    └─Conv2d: 3-23                           [8, 128, 128, 128]        32,896
│    │    └─BatchNorm2d: 3-24                      [8, 128, 128, 128]        256
│    │    └─ReLU: 3-25                             [8, 128, 128, 128]        --
│    │    └─Conv2d: 3-26                           [8, 128, 128, 128]        147,584
│    │    └─BatchNorm2d: 3-27                      [8, 128, 128, 128]        256
│    │    └─ReLU: 3-28                             [8, 128, 128, 128]        --
│    │    └─Conv2d: 3-29                           [8, 256, 128, 128]        33,024
├─ModuleList: 1-8                                  --                        --
│    └─Hourglass: 2-4                              [8, 256, 128, 128]        --
│    │    └─ModuleList: 3-30                       --                        2,788,864
├─ModuleList: 1-9                                  --                        --
│    └─Sequential: 2-5                             [8, 256, 128, 128]        --
│    │    └─Bottleneck2D: 3-31                     [8, 256, 128, 128]        214,528
├─ModuleList: 1-10                                 --                        --
│    └─Sequential: 2-6                             [8, 256, 128, 128]        --
│    │    └─Conv2d: 3-32                           [8, 256, 128, 128]        65,792
│    │    └─BatchNorm2d: 3-33                      [8, 256, 128, 128]        512
│    │    └─ReLU: 3-34                             [8, 256, 128, 128]        --
├─ModuleList: 1-11                                 --                        --
│    └─Conv2d: 2-7                                 [8, 64, 128, 128]         16,448
==========================================================================
Total params: 3,599,296
Trainable params: 3,599,296
Non-trainable params: 0
Total mult-adds (G): 188.01
--------------------------------------------------------------------------
```

Figure A1: Summary snapshot of the neural network configuration. The neural network configuration has been included in the Appendix, as it is not a result of the experiment. Rather, it is a piece of additional information for anyone interested in learning more about the neural network.