Deep Learning for Detection and Tracking of Underwater Pipelines using Multibeam Imaging Sonar

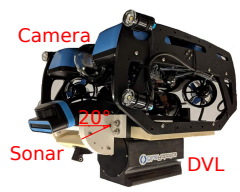
Jinkun Wang, Tixiao Shan, Muthukumaran Chandrasekaran, Timothy Osedach and Brendan Englot

***Abstract*— We propose a methodology for the automated detection and tracking of an underwater pipeline along the seafloor by an autonomous underwater vehicle (AUV) or remotely operated vehicle (ROV) equipped with a forward-looking multibeam imaging sonar. After training on a few hundred representative hand-segmented images, we use U-Net to segment sonar imagery in real-time and detect any portions of pipeline present in the imagery. The geometry of the pipeline is estimated by fitting a parametric curve to a skeletonization of the detected pipe segments, and this is used for repeatedly issuing waypoints to the underwater robot performing the inspection, allowing an AUV or ROV to perform a full-coverage automated flyover of a pipeline that is visible along the seafloor. The proposed framework is tested and validated using a BlueROV2 robot equipped with an Oculus multibeam imaging sonar. The ROV is tasked with performing multiple automated flyovers of a 100-foot pipeline placed in our testing tank, in configurations with varying curvature. The proposed deep learning framework is found to yield accurate estimation of the pipeline's geometry with very few false positives, and its integration with the ROS navigation stack yields fast and efficient pipeline inspections.**

I. Introduction

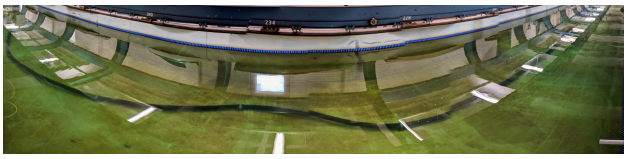
Underwater pipeline inspection is an important task for safe operations in the oil and gas, chemical and power generating industries. This task is often the responsibility of divers and manually piloted, remotely operated vehicles (ROVs), and it is both time and cost consuming. Divers performing underwater inspections risk injury, and can operate over limited ranges and durations. The requirement for intensive human supervision and control in ROV teleoperation is also inefficient, especially if long-duration, pervasive deployments are desired. For these reasons, untethered, autonomous underwater vehicles (AUVs) are a desirable solution for this task, which are increasing in maturity.

Existing robot platforms designed for pipeline tracking rely on a variety of sensors, including cameras, side scan sonar, multibeam echo sounders, sub-bottom profilers, and magnetic sensors. A vision-based system provides straightforward guidance, in which camera images can be processed in a traditional manner, e.g., edge detection followed by the Hough transform to track a straight pipeline [1], [2], or segmentation to allow the detection of rectangular pipeline sections [3]. A nonlinear control algorithm using visual feedback was also proposed to track a straight pipeline [4]. Side scan sonars provide a high-resolution image of the seafloor even in turbid water, but due to a limited field of view, their imagery has been used in aggregate to detect a pipeline prior to the initiation of tracking missions [5], or in offline mapping processes [6]. Multibeam echo sounders can be utilized for pipeline tracking once the pipeline is identified from side scan sonar [5]. Multiple sensing modalities, including cameras and sonars, have been fused to overcome the limitations of one individual sensor [7], [8].



(a) BlueROV2

(b) Onboard camera view



(c) A pipeline placed at the bottom of Stevens' testing tank

Fig. 1: Configuration of our vehicle and experiment setup.

In contrast with existing work, the aim of this work is to propose perception and navigation tools that can enhance the robustness and reliability of AUVs performing close-up pipeline inspections only with a high-resolution, forward-looking multibeam imaging sonar, without requiring extensive prior knowledge of a pipeline's location and geometry.

In this paper, we propose to use a deep neural network to perform pipeline detection using images from a multibeam imaging sonar. More specifically, we adapt an encoder-decoder network architecture and perform real-time pipeline segmentation. We obtain the dataset for training the neural network by hand-segmenting a collection of representative sonar images. A parametric curve fitting approach is then used to estimate the pipeline's geometry. Candidate waypoints are sampled from the curve and sent to the navigation system to achieve pipeline following by a BlueROV2 robot equipped with a RTI SeaPilot Doppler velocity log, a VectorNav VN100 inertial measurement unit, and an Oculus multibeam imaging sonar (Fig. 1). We conduct laboratory experiments in a testing tank as a proof of concept to demonstrate the utility of the proposed approach.

J. Wang, T. Shan and B. Englot are with the Department of Mechanical Engineering, Stevens Institute of Technology, Castle Point on Hudson, Hoboken NJ 07030 USA, {jwang92, tshan3, benglot}@stevens.edu.

M. Chandrasekaran and T. Osedach are with Schlumberger-Doll Research Center, One Hampshire Street, Cambridge MA 02139 USA, {mchandrasekaran, tosedach}@slb.com.

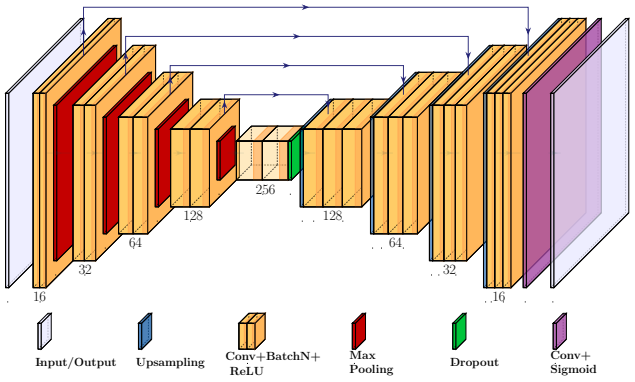


Fig. 2: A simplified U-Net model for pipeline segmentation using sonar images. The network follows an encoder-decoder architecture. Skip-connections are denoted by solid lines with arrows.

II. U-Net for Pipeline Detection

The wide variety of pipeline shapes and appearances in sonar imagery, and the similarity of intensities representing pipelines to those of walls and other structures, presents a great challenge to segmenting pipelines using a traditional approach. Therefore, we adapt a successful segmentation architecture, U-Net [9], to the task of pipeline detection, in which case, similar to U-Net for biomedical image segmentation, we rely heavily on a limited set of annotated training images. To achieve real-time segmentation and pipeline detection, we simplify the original U-Net, reducing feature channels while using the full resolution sonar image.

a) Deconvolution: While the imaging sonar is most sensitive within the beam width, the reflections originating from side lobes are also noticeable. During our laboratory experiments in a testing tank, we found Wiener deconvolution [10] to be of crucial importance for pipeline detection, and for discrimination between a pipeline and a vertical wall. As a result, both training and testing take the sonar image after deconvolution as input. The raw data and processed data after deconvolution are visualized in Fig. 4a.

b) Network architecture: We adapt the U-Net encoder-decoder architecture for this problem setting, shown in Figure 2. The encoder of the network consists of a sequence of convolutional blocks and average pooling layers for down-sampling the feature spatial resolutions while increasing filter banks. On the other hand, the decoder has a reversed structure with transposed convolutions for upsampling the feature spatial resolutions. All convolutions in the convolutional blocks are followed by batch normalization [11] and ReLU [12]. The output layer produces the final high-res range image using a single convolution filter without batch normalization. Note that a dropout layer is added to prevent over-fitting.

c) Data augmentation: The sonar image is represented in a polar coordinate system, where columns denote individual beams and rows denote range samples. As the number of rows changes with respect to the maximum sensing range, and the number of columns is fixed to 512 beams for our imaging sonar, the raw image is resized to $512\,\text{px}$ by $512\,\text{px}$.

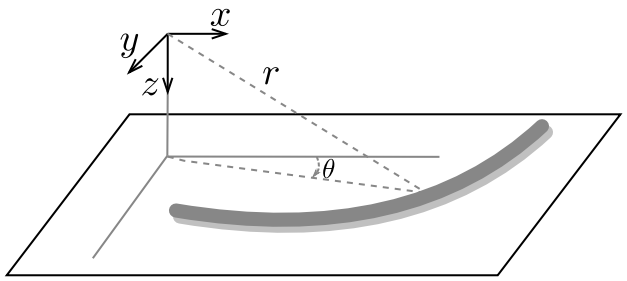


Fig. 3: Points extracted from sonar images are projected onto a flat floor based on altitude measurements from a DVL.

During training, the image is augmented by a series of random operations: horizontal flip ($p = 0.5$), resized crop (scale $\in [0.7, 1.0]$), and affine transformation ($|\text{rotation}| \leq 20°$, $|\text{translation}| \leq 50\,\text{px}$, $|\text{shear}| \leq 20°$).

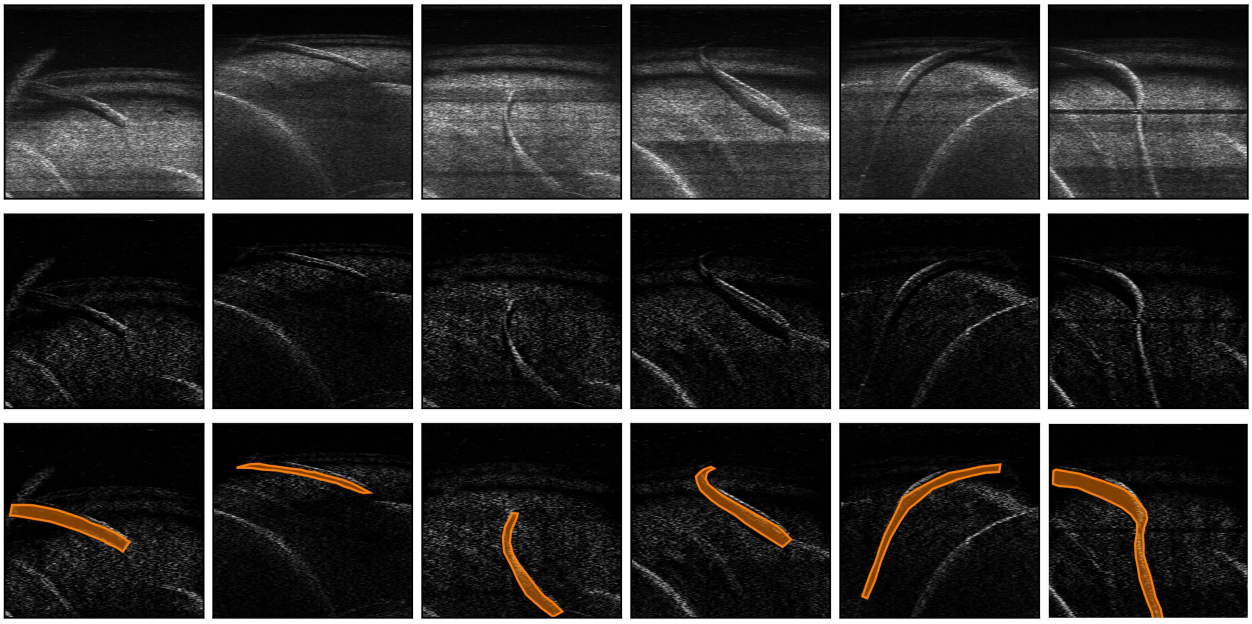
d) Training: The training dataset for the network is obtained by augmenting 114 segmented images that are labeled manually. SGD optimizer [13] is used with a learning rate of 10^{-4}, momentum of 0.99 and decay factor of 10^{-5} after each epoch for network optimization. Dice loss [14] is utilized for penalizing the differences between the network output and ground truth, as it achieves high accuracy, fast convergence and improved stability during training. The filter size for each convolutional block is shown in Figure 2. A kernel size of 3 by 3 is used for all convolutional operations.

III. Pipeline Following

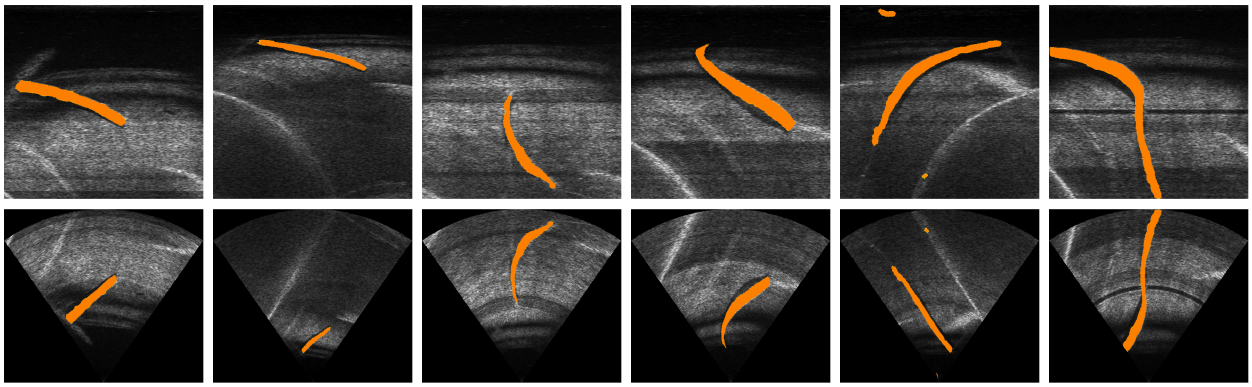
We now describe the procedures that enable us to achieve automated pipeline following. We first perform curve fitting to the output of the neural network. The result will be considered to represent the geometry of the pipeline detected in the sonar image. Then we project the pipeline curve from polar coordinates into Cartesian coordinates. At last, we issue a waypoint, which is sampled from the pipeline curve, to the robot operating system (ROS) [15] navigation stack. The details for each procedure are described below.

a) Curve Fitting: Since our sonar only covers a limited range, we would like to fit a curve to an observed pipeline to allow our robot to plan beyond its immediate field of view. We apply a parametric curve fitting approach to handle a variety of shapes as follows. First, the pipeline area is chosen to maximize the contour area in the output image after thresholding. Next, skeletonization is applied to the pipeline area. We then find a smooth curve that passes through the middle of the data using principal curves [16] to remove artifacts resulting from skeletonization. Considering the pipeline generally has small curvature, the final step is to fit a second-order parametric curve, and the pipeline can be reasonably extrapolated to a desired distance.

b) Curve Projection: The detected pipe is represented in polar coordinates relative to the sonar, and we would like to issue waypoints in Cartesian coordinates. However, the elevation angle is lost during the formation of the sonar image, which poses a problem to recovering the true geometry of the pipeline. As shown in Fig. 3, we obtain



(a) Visualization of samples of training data. **At top**: raw sonar intensity images in polar coordinates. **Middle row**: images after the deconvolution process are shown, which are the actual input to the neural network. **At bottom**: Hand-drawn polygons used to label the training data are denoted by the orange regions shown. Unlabeled, bright returns from our tank walls are visible in the imagery.



(b) Demonstration of automated pipeline detection. Output pixels with probability higher than 0.9 are visualized on top of raw images. Images are transformed to Cartesian coordinates in the bottom row.

Fig. 4: Examples showing the input and output of the neural network depicted in Fig. 2. For illustrative purposes, the network is queried here over images used for training. The pipeline boundaries predicted in (b) are improved compared with the manual labeling in (a).

the range r and bearing θ measurements directly, but the elevation relative to the imaging plane is ambiguous. In this work, we assume the environment has a flat bottom, such that we are able to estimate the 3D location of a given point in the imagery with the knowledge of altitude from the bottom. The altitude measurement can be obtained from a DVL or a pressure sensor given a fixed water depth.

c) Waypoint Following: We then sample waypoints from the pipeline's curve in Cartesian coordinates. For the purpose of smooth motion, we continuously select a waypoint that is a designated distance ahead of the robot's current position along the parametric curve ($1.5\,\mathrm{m}$ in Fig. 5). Thus, the robot always faces the pipeline as it moves forward. The velocity control commands for following the waypoint are provided by the ROS navigation stack. Since we assume the operational environment is flat, we only need three types of velocity commands: two velocities for forward and lateral translation, and a yaw rate command. Note that for non-flat environments, we can implement a controller that adjusts the depth of the robot using the readings from the onboard depth sensor, or which controls altitude using the DVL returns.

IV. Experiments

Two experiments were carried out to demonstrate the performance of our proposed pipeline tracking framework. The first experiment was conducted in the Stevens towing tank (with dimensions $313 \times 16 \times 8\,\mathrm{ft}^3$). We anchored a 100-foot long, 4.5-inch diameter polyethylene drainage pipe to the floor, and our ROV was tasked with performing an autonomous flyover of the pipe (Fig. 1(b)(c)). The experiment

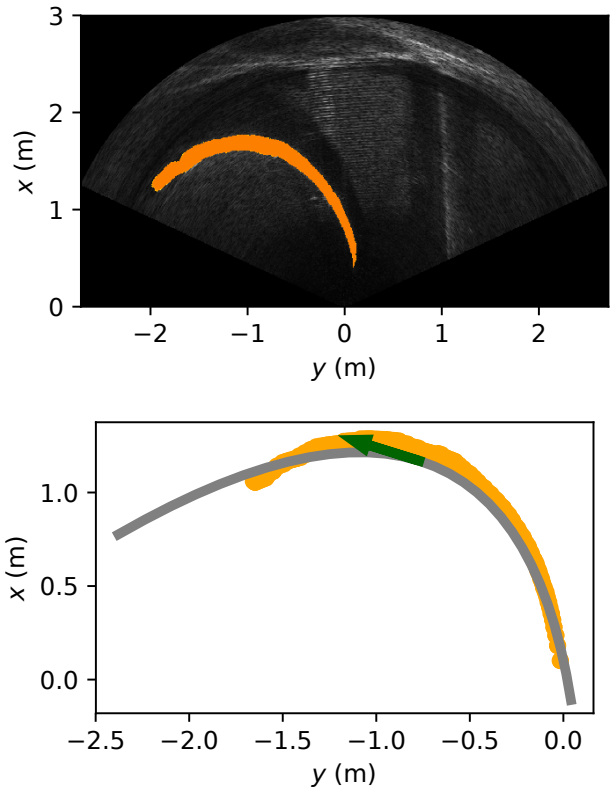


Fig. 5: A representative example of a sharply curved pipeline at the corner of a small tank. The skeletonized points and the curve fitting result are projected to the tank floor based on an altitude of 1.0 m. The corresponding waypoint is shown by the green arrow.

was performed using the BlueROV2, with inertial data from a VectorNav VN-100 IMU, vehicle velocity from an RTI SeaPilot DVL (600 MHz), and depth from a Bar30 pressure sensor. These measurements were used to construct odometry factors for the iSAM2 [17] factor graph used as our basis for state estimation, via the GTSAM library [18]. We used the Oculus M750d multi-beam imaging sonar, operated in 1.2 MHz mode with a field of view of maximum range 3 m, horizontal aperture 70° and vertical aperture 12°. The sonar image updates at 10 Hz and has a range resolution of 0.005 m and an angular resolution of 0.0024 rad. The sonar is mounted pointing 20° downward from the horizontal, to obtain the largest possible coverage. The proposed method was implemented using PyTorch [19] in ROS Kinetic running Ubuntu 16.04 on a laptop with an i7-7700HQ CPU and an NVIDIA GTX 1050 GPU.

The setting for this experiment is depicted in Fig. 1. The ROV was manually driven to collect pipeline images from a variety of perspectives. The parameters of our simplified U-Net model were tuned by splitting a total of 114 annotated images into 91 training images and 23 validation images, achieving a dice score of 0.73 and an intersection of union (IOU) of 0.60. The model was re-trained using the entire dataset afterwards, and we were able to perform pipeline segmentation at 5 Hz using the computer resources described above. We can see from Fig. 4b that the pipeline boundaries obtained from U-Net are largely accurate. Although some outliers are detected from the tank walls, their small area allows them to be filtered easily. We further tested the pipeline following capability in an extreme case as shown at the right of Fig. 6. The pipeline formed a C shape in the tank, and the robot successfully tracked the pipeline with the tank wall present in the sonar's field of view.

Raw sonar images are stitched together using the dead-reckoned trajectory, which is visualized in Fig. 6. The intensity of every location in the mosaic is the average intensity of all measurements interpolated from sonar images at different poses assuming the elevation angle is zero. It is worth noting that due to ambiguous elevation angles, the mosaic does not accurately represent the true pipeline's geometry. The detected pipeline regions, which are extracted from prediction with the largest connected area after thresholding, were accumulated during the tracking mission as shown in the orange regions in Fig. 6. The mosaic image also exhibits a significant amount of drift from dead reckoning. Although the automated pipeline following wasn't influenced by drift, as the robot was navigating in a body frame, it would be desirable in the future to introduce sonar-derived measurement constraints into iSAM2 to produce a globally consistent map of a pipeline. Representative results from successful automated flyovers of straight[1], curved[2], U-shaped[3], and loop-shaped[4] pipe geometries are linked below. All results described above were produced from training data gathered over a single pipe configuration - the layout depicted in Fig. 1(c) and at the left of Fig. 6.

Our second experiment was conducted in a much smaller tank ($20 \times 9 \times 4.5\,\mathrm{ft}^3$) with a different sensor, the higher-resolution Oculus M1200d multi-beam imaging sonar, to evaluate the approach across different sensor arrangements. As the operational conditions (e.g. altitude, properties of the tank floor and walls) and sonar specifications (e.g. vertical aperture, resolution) have changed, the acoustic shadows cast by the pipeline, which are informative for distinguishing them from other objects, changed accordingly. Therefore we re-trained the model using a separate dataset, though in future work, the previous dataset can be expanded to incorporate training data from different environments. Fig. 5 shows a result from this experiment, collected using the Oculus M1200d operated at 1.2 Mhz but with a horizontal aperture of 120° and a vertical aperture of 20°. A large portion of the pipeline is visible in a single image, and a sufficiently accurate curve is recovered from its predicted segmentation.

V. Conclusion

We have proposed a methodology for the automated detection and tracking of an underwater pipeline using a forward-looking multibeam imaging sonar. Our method performs real-time pipeline segmentation from sonar images using a

[1] https://youtu.be/pzIS3_mFft0
[2] https://youtu.be/oL45QrxqbYI
[3] https://youtu.be/SYWY6X3eMZw
[4] https://youtu.be/eMaWhvyYFLg

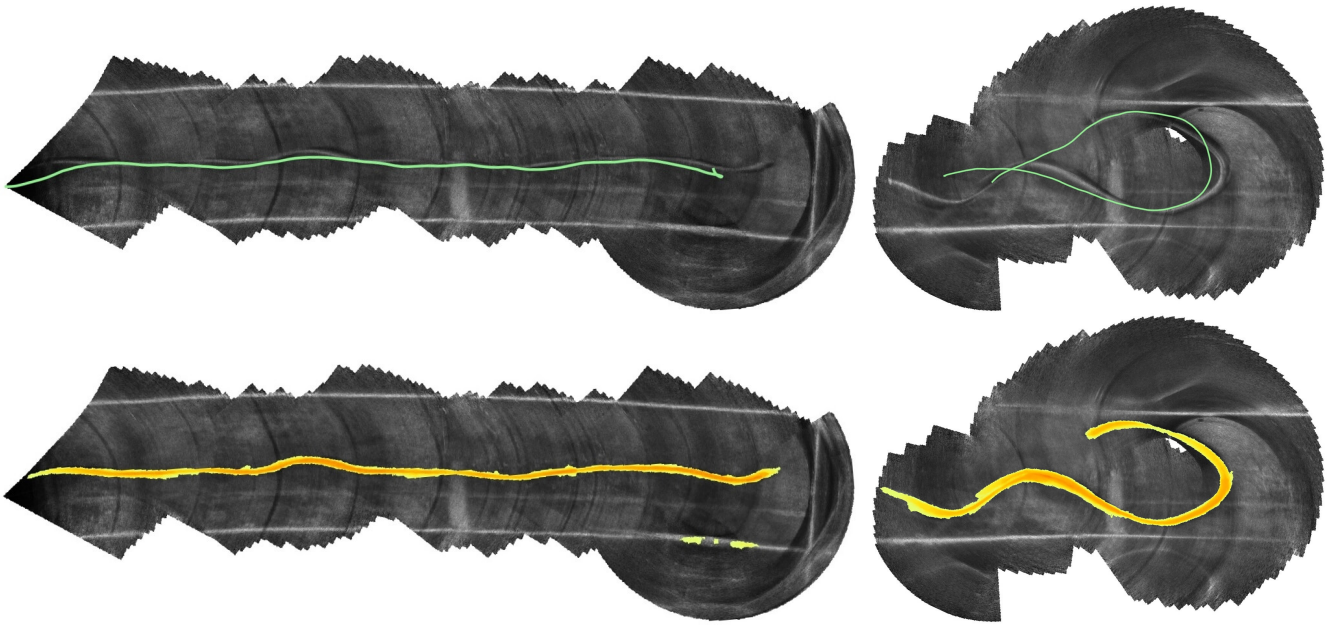


Fig. 6: Dead reckoning-derived mosaics of raw sonar images during two pipeline tracking missions in the 16-foot wide Stevens tank. The estimated trajectory is shown by the green line, and the detected pipeline is shown by the orange region (intensity denotes number of detections). **Left**: tracking of a straight pipeline (100 ft) with some local bends and curves, **right**: Tracking of a sharply curved pipeline.

deep neural network, U-Net, on a laptop. With the predicted pipeline segments, we adapt a parametric curve fitting approach to represent the pipeline as a curve. Then we sample waypoints from the pipeline curve and issue them to the ROS navigation stack, which provides velocity commands for pipeline following. We conducted laboratory experiments to demonstrate the utility of the proposed approach. Future work will involve performing experiments on non-flat terrain, such as pipeline following in true subsea environments, and incorporating sonar-derived measurement constraints into our simultaneous localization and mapping framework.

Acknowledgments

This research was supported by a grant from Schlumberger Technology Corporation.

References

[1] S. Matsumoto and Y. Ito, "Real-time vision-based tracking of submarine-cables for AUV/ROV," *Proceedings of the IEEE/MTS OCEANS Conference*, 1995

[2] M. Narimani, S. Nazem, and M. Loueipour, "Robotics vision-based system for an underwater pipeline and cable tracker," *Proceedings of the IEEE/MTS OCEANS Conference*, 2009.

[3] J.O. Hallset, "Simple vision tracking of pipelines for an autonomous underwater vehicle," *IEEE International Conference on Robotics and Automation,* pp. 2767-2772, 1991.

[4] S. Krupiński, G. Allibert, M.D. Hua, and T. Hamel, "Pipeline tracking for fully-actuated autonomous underwater vehicle using visual servo control," *Proceedings of the American Control Conference*, pp. 6196-6202, 2012.

[5] Y.R. Petillot, S.R. Reed, and J.M. Bell, "Real time AUV pipeline detection and tracking using side scan sonar and multi-beam echo-sounder," *Proceedings of the IEEE/MTS OCEANS Conference*, 2002.

[6] A. Bagnitsky, A. Inzartsev, A. Pavin, S. Melman, and M. Morozov, "Side scan sonar using for underwater cables & pipelines tracking by means of AUV," *IEEE Symposium on Underwater Technology and Workshop on Scientific Use of Submarine Cables and Related Technologies*, 2011.

[7] M. Jacobi and D. Karimanzira, "Underwater pipeline and cable inspection using autonomous underwater vehicles," *Proceedings of the IEEE/MTS OCEANS Conference*, 2013.

[8] M. Jacobi and D. Karimanzira, "Multi sensor underwater pipeline tracking with AUVs," *Proceedings of the IEEE/MTS OCEANS Conference*, 2014.

[9] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional Networks for Biomedical Image Segmentation," *Proceedings of the International Conference on Medical Image Computing and Computer-assisted Intervention*, pp. 234-241, 2015.

[10] P.V. Teixeira, F.S. Hover, J.J. Leonard, and M. Kaess, "Multibeam Data Processing for Underwater Mapping," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1877-1884, 2018.

[11] S. Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *Proceedings of the International Conference on Machine Learning*, pp. 448-456, 2015.

[12] V. Nair and G.E. Hinton, "Rectified linear units improve restricted boltzmann machines," *Proceedings of the International Conference on Machine Learning*, pp. 807-814, 2010.

[13] J. Kiefer, and J. Wolfowitz, "Stochastic estimation of the maximum of a regression function," *The Annals of Mathematical Statistics*, vol. 23(3), pp. 462-466, 1952.

[14] C.H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M.J. Cardoso, "Generalised Dice Overlap as A Deep Learning Loss Function for Highly Unbalanced Segmentations," *Proceedings of the International Workshops on Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 240-248, 2017.

[15] N. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A.Y. Ng. "ROS: an open-source Robot Operating System," *IEEE International Conference on Robotics and Automation, Workshop on Open Source Software*, 2009.

[16] U. Ozertem and D. Erdogmus, "Locally defined principal curves and surfaces," *Journal of Machine Learning Research*, vol. 12, pp. 1249-1286, 2011.

[17] M. Kaess, H. Johannsson, R. Robert, V. Ila, J.J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *The International Journal of Robotics Research*, vol. 31(2), pp. 216-235, 2012.

[18] GTSAM, `https://bitbucket.org/gtborg/gtsam`

[19] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," *NIPS 2017 Autodiff Workshop: The Future of Gradient-based Machine Learning Software and Techniques*, 2017.