Real-Time Video Inpainting for RGB-D Pipeline Reconstruction

Luyuan Wang[1], Tina Tian[1], Xinzhi Yan[1], Fujun Ruan[1], G. Jaya Aadityaa[1], Howie Choset[1,*] and Lu Li[1,*]

***Abstract*— This paper presents a Video Inpainting algorithm that enables monocular-camera-laser-based pipeline inspection robots to capture both color and 3D information using only one video stream. Conventional monocular-camera-laser inspection methods are limited to capture either 2D color images or 3D point clouds since the laser tends to overexpose the actual color of the scanning area. We propose a real-time Video Inpainting method to solve this problem with minimal hardware needs that can be easily integrated with conventional pipeline profiling robots. The algorithm is accelerated by two components: a lightweight network that directly predicts the complete optical flow and simplifies the algorithm pipeline, and the Polar coordinate transformation, which significantly reduces the image processing compexity. Real-world experiments demonstrate that our online algorithm has comparable or better color estimation accuracy against state-of-the-art offline algorithms, while is capable of running at 23 frames per second (FPS) on a laptop computer with a resolution of 1024×1024 pixels. In addition, we verify that this method can be used for video pre-processing for downstream tasks that require high-quality visual inputs, such as Simultaneously Localization and Mapping (SLAM). To the best of our knowledge, this is the first real-time Video Inpainting algorithm that can be used for in-pipe environments, serving as an important building block for highly compact RGB-D inspection sensors and robots for the pipeline industry.**

I. INTRODUCTION

The U.S. has a large-scale oil and gas transmission network, which has about 1.8 million miles of mainline pipes by the end of the year 2021 [1]. The number will increase drastically if we also count service pipelines. These pipes are suffering from aging, corrosion, rupture, and other types of damage [2], which eventually cause leakage, fire, and even explosion. Such incidents can result in significant property damage, injuries, and loss of life. Thus, it's crucial to monitor and evaluate the pipeline status on a regular basis to prevent these accidents from happening. Currently, the mainstream solution is to send Closed-Circuit Television (CCTV) robots into these pipes, which consist of a remote-control mobile platform and a camera. The streamed-out videos can be used for image-based damage analysis [3].

However, the lack of 3D structural information makes CCTV robots hard to perform fine-grained evaluation tasks. In recent years, an increasing number of robots are equipped with 3D perception systems, such as stereo cameras, Lidar, and laser profilometry [4]. The properties of low cost, compact size, and high scanning accuracy make laser profilometry gains popularity for 3D scanning solutions.

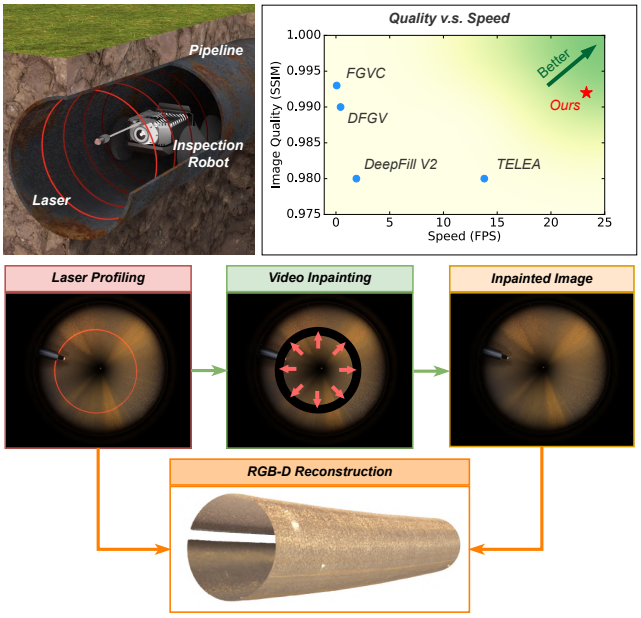

Fig. 1. A laser-camera-based robot is scanning an underground pipe. The onboard camera captures the laser profiling frames, which are fed into the Video Inpainting module. The profiling frames and the output inpainted frames are used for RGB-D pipeline reconstruction. Compared with other methods, our algorithm runs faster with comparable or better image quality.

Conventional laser profiling robots only provide either 2D visual images or monochrome 3D point clouds at a time, as the projected laser over-saturates the actual pipe color behind the laser strip. One solution to colorize the point cloud is to use Infra-Red (IR) laser that does not interfere with the visible spectrum [5]. This approach needs an additional IR sensor, which cannot utilize existing laser-camera-based robot platforms. In our previous work [6], a laser alternating method is used to generate two video streams from one camera. That is, one stream turns on the laser to get the depth information, and the other stream turns off the laser and captures the color information. This approach, however, needs specific hardware to synchronize the camera shutter, the laser projector, and the LEDs and only half of the image frames can be used to retrieve the depth information.

The proposed algorithm allows us to estimate the actual pipe color under the laser strip purely by software, as shown in Fig. 1. This approach can help us design highly compact robots with minimum hardware requirements without wasting half of the frames to get color information. The

This work was supported by the Department of Energy Advanced Research Projects Agency-Energy (ARPA-E) under the Rapid Encapsulation of Pipelines Avoiding Intensive Replacement (REPAIR) program.

[1]The authors are with the Biorobotics Lab, the Robotics Institue at Carnegie Mellon University, Pittsburgh, PA 15213, USA

*Howie Choset and Lu Li are corresponding authors.

Email: choset@andrew.cmu.edu, lilu12@andrew.cmu.edu

laser strip that blocks the actual pipe color can be viewed as a corrupted region of a normal visual image without a laser on it. The task of filling in the corrupted region of each frame in a video sequence is a Video Inpainting problem [7]. Our method can inpaint the laser regions at high throughput and can be used as a plug-and-play pre-processing module for downstream tasks, such as SLAM. When using the algorithm online, it runs fast enough to support robot visual localization; when using it offline, it can pre-process the recorded data and enable full frame rate scanning with relatively low computation resources. Compared with other solutions, for example, projecting an IR laser, the key advantage of our proposed method is that it does not require additional sensing or controlling hardware, thus is easy to upgrade existing robot platforms to enable RGB-D pipeline scanning capabilities at a low cost.

The main contributions can be summarized as follows: (1) We are the first to address the in-pipe color estimation problem with Video Inpainting. (2) A lightweight network fine-tuned with Unsupervised Learning simplifies existing algorithms by directly estimating complete in-pipe optical flow. (3) The Polar coordinate transformation further reduces computation costs. (4) Experiments in real-world natural gas pipes demonstrate our method's efficacy in real-time operation and supporting demanding downstream tasks.

II. Related Work

A. 3D Pipeline Reconstruction

Geometry information is important to evaluate pipeline attributes, such as curvature, dents, or cracks. However, depth sensors that are commonly used in robotic applications may not be suitable for in-pipe environments. Lidar is commonly used for tunnel inspection [8], [9], but is not suitable for pipes because of the insufficient resolution for short-range sensing. Stereo-camera sensors either only achieve centimeter-level scanning accuracy [4], or require feature-rich patterns and perform offline optimization [10]. [11], [12] uses laser profilometry and achieved sub-millimeter accuracy, but can't provide color information. [5] uses an IR laser and IR camera to scan the pipe and associate color information from another regular RGB camera. This approach needs two cameras, which increases the size of the robot so it cannot navigate through narrower pipes. Also, the robot localization is achieved by wheel encoders, which may suffer from drift. One way to reduce drift is to fuse multiple sensors using SLAM. VLI-SLAM [6] is designed for laser-based 3D scanning. It is built on a monocular visual-inertial SLAM algorithm, VINS-Mono [13], but also fuses laser information to achieve a higher localization accuracy. Our testing robot is based on VLI-SLAM, as it has a more strict image quality requirement compared with other downstream tasks.

B. Image and Video Inpainting

In Computer Vision, Image Inpainting is the task of filling the corrupted region in a given image with reasonable contents. Learning-based methods have outperformed traditional methods and become mainstream [14]. The task of inpainting each frame in a video sequence is called Video Inpainting. Flow-based deep learning methods achieve state-of-the-art performances [15], [16], [17]. Generally, these methods first estimate bi-directional optical flows and then propagate pixels in past and future frames to fill the missing region in the current frame. The Computer Vision community mainly focuses on improving the image reconstruction quality, consequently that these methods are not optimized for real-time applications like SLAM, since they require both past and future frames, refine the result in multiple iterations and consume massive memory space. Moreover, the networks are trained with natural video datasets and do not work well with in-pipe images. In this work, we aim to provide a lightweight Video Inpainting solution for pipeline laser profiling robots.

C. Optical Flow Estimation

Optical flow estimation plays an important role in flow-based Video Inpainting algorithms. [15] and [16] use pre-trained networks of FlowNetV2 [18] and RAFT [19] for optical flow estimation, respectively. These optical flow networks are computationally heavy and cannot be used for real-time tasks on portable devices. More importantly, they are trained with synthetic datasets that mimic open-space natural images with ground truth labels [20], [21], but perform poorly on in-pipe images. However, it's hard to obtain ground truth optical flow of in-pipe images for fine-tuning. Unsupervised optical flow estimation methods use image pairs themselves as supervision and do not require labels [22], [23]. Our work is partially based on a lightweight flow estimator, FastFlowNet [24], which is fine-tuned with unsupervised learning on our unlabeled in-pipe dataset.

III. Method

A Video Inpainting algorithm must satisfy the following constraints to be used for online inspection tasks.

- Causality: Only information from previous frames can be used, as future frames are yet to be captured.
- High resolution: Features within pipes can be very small and some tasks, like SLAM, require high-resolution image inputs to effectively track feature points.
- Real-time processing speed: VLI-SLAM requires a minimum of 10 FPS to operate, which needs to be achieved on a laptop computer for field deployment.

This section describes the algorithm design for fulfilling the above-mentioned constraints. Sec. III-A overviews the Video Inpainting algorithm pipeline. Sec. III-B expands upon the training details for the deep flow estimator. Sec. III-C uses VLI-SLAM as an example to explain how the algorithm is integrated with downstream tasks.

A. Algorithm Overview

We design the real-time Video Inpainting algorithm based on a classic Flow-based method, Deep Flow-Guided Video Inpainting (DFGV) [15], which has three major components: a flow estimator, a flow completion network, and an Image Inpainting network. Several changes are made for the real-time requirement: First, the bi-directional flow estimation

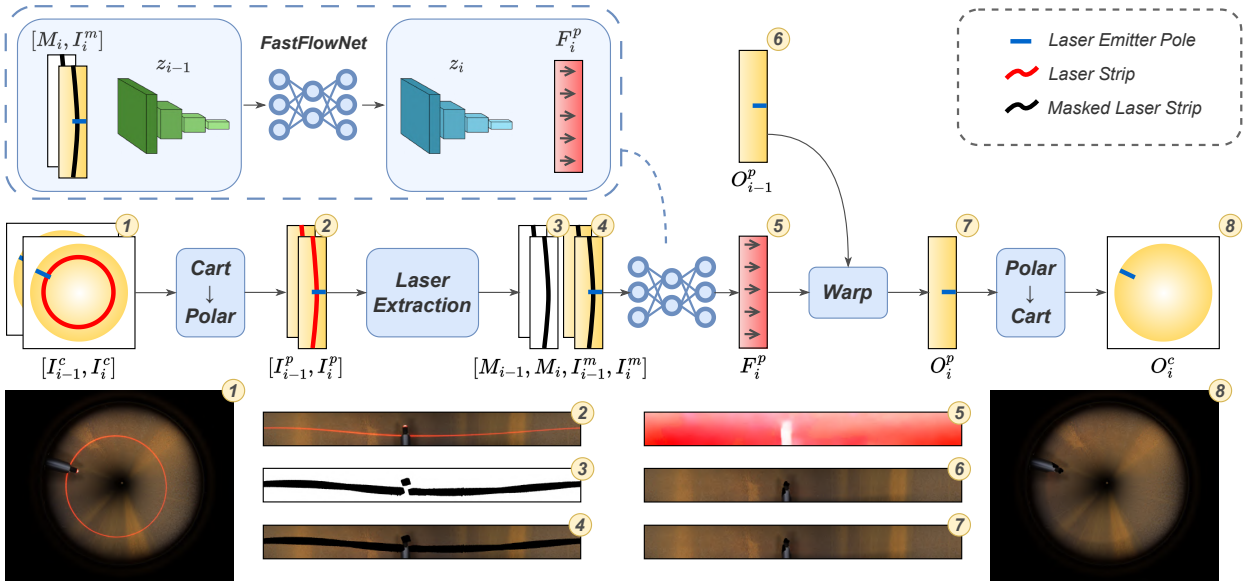

Fig. 2. **Overview of the proposed video inpainting algorithm.** I, M, F, z and O represent the input image, mask, optical flow, latent feature pyramid, and output image, respectively. The subscript i and $i-1$ represent the current (i-th) frame and the previous frame. The superscript c and p represent the Cartesian and the Polar coordinate. The number badges indicate the correspondence between the data symbols and the actual images. The images in the Polar coordinate are rotated by 90 degrees for better visualization (②-⑦). The FastFlowNet is modified to input and output latent feature pyramids to save redundant computes, as visualized in the dashed box.

is replaced with unidirectional, as we can only utilize the past information. Second, the Image Inpainting network is removed. As the robot always moves along the pipe axial direction, all the corrupted pixels should be visible in previous frames, so that single-frame Image Inpainting becomes unnecessary. Last, the multi-iteration refinement is changed to single-pass for inference efficiency.

The proposed algorithm pipeline has the following core operations: coordinate transformation, laser mask extraction, optical flow estimation, and flow warping, as shown in Fig. 2. The input is an image pair of the current frame I_i^c and the previous frame I_{i-1}^c. Our goal is to estimate the color behind the red laser so that the output frame O_i^c should be an image without the laser. Because the raw image from the sensor is black on the sides caused by the fish-eye lens, the input image is center cropped to 1024×1024. Such a high resolution is required for preserving the small visual features. These features are important for some downstream tasks, such as the SLAM feature tracker. However, the region of interest (ROI) is always around the laser ring, which only takes a small portion of the whole image. We perform a Cartesian to Polar coordinate transformation (denoted as $Cart \rightarrow Polar$), which converts the circular ROI to a rectangle for the ease of cropping and reducing the processing region. The coordinate transformation is essentially a pixel-wise mapping from the $\{x, y\}$ domain to the $\{\theta, \rho\}$ domain, as shown in Eq. (1) and Eq. (2), where (C_x, C_y) is the image center, H and W are the height and width of the image, R is the maximum radius, which is set to half of the image size, i.e., 512. We crop the image around the laser line along the horizontal axis, as visualized by $[I_{i-1}^p, I_i^p]$. When cropping, we leave enough context information for optical flow estimation as well as to handle the imperfect pipes, where the laser pixels deviate more from a straight line.

$$\boldsymbol{I} = \begin{bmatrix} x - C_x \\ y - C_y \end{bmatrix} \tag{1}$$

$$\begin{cases} \theta = (H/2\pi) \cdot \angle \boldsymbol{I} \\ \rho = (W/R) \cdot \|\boldsymbol{I}\| \end{cases} \tag{2}$$

The coordinate transformation brings additional overhead that needs to be reduced. Assume the image frames captured by the sensor always have the same size, then the $\{x, y\} \rightarrow \{\theta, \rho\}$ mapping is constant. We can pre-calculate and cache the result, and only perform image warping at inference time. Moreover, image warping can be accelerated on GPU. The caching and GPU acceleration reduce the transformation time to 5.8% of the original OpenCV implementation. We similarly accelerate the Polar to Cartesian coordinate transformation (denoted as $Polar \rightarrow Cart$ in Fig. 2).

After the ROI cropping, the image size is reduced to 26% of the input. Next, a simple color filter is performed to extract the laser mask, which indicates the region to be inpainted. The masked images I_{i-1}^m, I_i^m and the masks M_{i-1}, M_i are stacked together and fed into the FastFlowNet [24] to obtain the optical flow F_i^p, which represents the pixel movement from I_{i-1}^p to I_i^p. The details of the flow estimator are elaborated in Sec. III-B.

The previous inpainted frame O_{i-1}^p is warped with F_i^p to fill the masked-out region and get the inpainted output O_i^p. For the very first frame, we set $O_0^p = I_0^m$. The pixel propagation equation is shown in Eq. (3), where M_i is the

laser mask that contains a collection of laser pixel indices. $\boldsymbol{f} = F_i^p(x, y)$ is the vector of the pixel movement at the given location.

$$O_i([x,y]^\top) = \begin{cases} I_i([x,y]^\top + \boldsymbol{f}), & \text{if } \{x,y\} \in M_i \\ I_i([x,y]^\top), & \text{otherwise} \end{cases} \quad (3)$$

Finally, we replace the ROI in the original Polar-coordinate image with O_i^p and convert it back to the Cartesian coordinate to get the output O_i^c without laser. The result is saved for processing the next input image I_{i+1}^c.

B. Unsupervised Optical Flow Estimation

The optical flow estimator plays an important role in flow-based Video Inpainting algorithms. If the estimated optical flow is inaccurate, the pixel propagation stage will fail and significantly influence the image reconstruction quality.

State-of-the-art optical flow networks are reported to have very low Endpoint Errors (EPEs), however, most of these pre-trained networks perform poorly for in-pipe images because of the domain shift. Public large-scale optical flow datasets, such as KITTI [25], Sintel [26], and Flying Chairs [21] are all designed for natural images. Trained on these datasets, the networks learn to track the movement of cars, humans, or other objects. On the contrary, features in pipelines are mainly made of rust, dust, or merely the texture of the cast iron. These small features are outside the training data distribution and cause the pre-trained networks to predict inaccurate results, so fine-tuning is necessary. One challenge is that, unlike the synthetic dataset, there are no in-pipe optical flow labels. To address this problem, we leverage Unsupervised Learning to fine-tune the optical flow network on our unlabeled in-pipe dataset collected by the robot.

Another challenge is that, since the laser strips are masked-out in the input images, the output optical flow is incomplete in the laser regions. DFGV uses a cascading network to predict the complete optical flow. We propose to predict the complete optical flow directly from the optical flow estimation network for better efficiency.

In this work, the optical flow is estimated by a modified version of FastFlowNet [24]. It is described in three sub-sections: the network architecture, the loss function, and the data augmentation methods.

Network Architecture: To support real-time applications, we need a lightweight optical flow estimation network. FastFlowNet balances well between accuracy and efficiency [24]. It has three major components. The Head Enhanced Pooling Pyramid (HEPP) and the Center Dense Dilated Correlation (CDDC) module heavily compress the spatial size and the number of channels when encoding the feature pyramid. The Shuffle Block Decoder (SBD) utilizes ShuffleNet [27] blocks to reduce the computation when decoding the optical flow.

As our objective is to directly predict the complete optical flow, more information needs to be provided to the network. The mask M_i, which indicates what region is masked out, together with the image I_i^m are stacked and form a 4-channel input tensor, i.e., mask channel and RGB channels. The input for the first convolutional layer is thus increased from 3-channel to 4-channel. When loading the pre-trained network, the weights for the third channel are copied to the new channel. This helps the network converge faster than random initialization during training.

FastFlowNet uses a shared weight feature extractor for the two input images to reduce the number of parameters [24]. However, due to our input frames being sequential, calculating the feature pyramid for the same image twice is redundant. More specifically, at time i, the input to the flow estimator is $\{[M_{i-1}, I_{i-1}^m], [M_i, I_i^m]\}$. At time $i+1$, the input becomes $\{[M_i, I_i^m], [M_{i+1}, I_{i+1}^m]\}$, where the frame $[M_i, I_i^m]$ is processed twice. To remove the redundancy, we modify FastFlowNet to output both the estimated optical flow F_i^p and the latent feature pyramid z_i for $[M_i, I_i^m]$, as shown in Fig. 2. The z_i is cached and used as part of the input at the next timestamp.

Loss Function: An unsupervised loss function is utilized for both improving the network performance on our unlabeled dataset as well as directly predicting the complete optical flow. We adopt the unsupervised photometric loss and the smooth regularization from [23]. The photometric loss is defined in Eq. (4), where Θ is the set of learnable parameters of the estimator f, $\hat{I}_1$ is the first image warped with the network's output. I_1^m and I_2^m are the masked image inputs. I_2 is the second image, which is used as the supervision. ρ is the function of pixel-wise structural similarity (SSIM), and we calculate the summation for each pixel p. This loss function is used to optimize the optical flow such that the warped first image matches the second image. The inputs are corrupted images, but instead of using the same image pair as supervision as in [23], we calculate the photometric loss with the complete image pair, such that the network learns to predict the complete flow directly.

$$L_{ph} = \sum_p \rho(\hat{I}_1(f(I_1^m, I_2^m; \Theta)), I_2) \quad (4)$$

The smooth regulation term is defined in Eq. (5), where F is the estimated optical flow. This is to eliminate the ambiguity of texture-less regions, which is common in in-pipe environments. The richness of textures is measured by the image gradient. For regions with small gradients in the image, we also want small gradients in the optical flow domain to get a smooth flow output.

$$L_{sm} = \sum_{d \in x,y} \sum_p ||\nabla_d F||_1 \; e^{-|\nabla_d I|} \quad (5)$$

The overall loss function is defined as Eq. (6), where $\lambda = 50$ is a hyper-parameter, following the same setup in [23].

$$L = L_{ph} + \lambda L_{sm} \quad (6)$$

Data Augmentation: Data augmentation helps us increase the size of the training set and prevent over-fitting. Two types of data augmentation techniques are used for training. The first one, speed augmentation, trains the network to handle different robot moving speeds. To achieve this, instead of

giving two consecutive frames $\{I_i, I_{i+1}\}$ in the dataset as the input, we skip some frames and use $\{I_i, I_{i+n}\}$ to simulate the robot moving faster. We randomly choose n in the range of $[1, 3]$ during the training phase.

The second data augmentation technique is random laser mask generation. Our testing pipes are all nearly perfect cylinders, such that the laser masks always look similar. However, certain defects may change the pipe's shape, and a pipeline inspection robot shouldn't fail in these cases. We use B-Splines [28] to synthesize laser masks with imperfection. Four knots at the top, right, bottom, and left are first created to form a perfect circle, and then random perturbations are added to make the circle imperfect. During the evaluation phase, we use real laser masks extracted from the images.

C. Downstream Task Integration

Our algorithm can be easily plugged into existing systems since it only requires a video stream, and is not coupled with hardware. We demonstrate the downstream task integration method with VLI-SLAM [6], as it is more complicated and has higher input quality requirements than other tasks, such as wheel encoder-based laser scanning.

Components in VLI-SLAM communicate with each other using the Robot Operating System (ROS) message-passing interface. In the original VLI-SLAM, the robot sensor driver outputs three topics, where two image topics `/profile` and `/visual` are used to obtain depth and color information and perform feature tracking. The inertial topic `/imu` and the tracked features are sent into the backend for online optimization. With the proposed method, the laser projector no longer needs to be repeatedly turned off, and the `/visual` topic from the sensor can be simply removed. The newly added inpainting module takes the `/profile` topic as the input, inpaint the laser area, and outputs the equivalent `/visual` topic. Since the inpainting module is decoupled with the VLI-SLAM and only used for input pre-processing, the other parts of the SLAM system remain unchanged.

IV. Experiments

We designed three different experiments to evaluate the algorithm performance as well as verify it can support downstream tasks. We first study the image reconstruction quality and compare it against several previous Image and Video Inpainting methods. Then, we conduct two other experiments to analyze the scanning and localization accuracy difference between the original laser alternating and the proposed method for VLI-SLAM. We show that the difference is small and the downstream task is negligibly affected.

A. Robot Platform and Experiment Setup

We build a custom pipeline inspection robot for data collection. The robot consists of a modular pipe crawler and a sensor payload, as shown in Fig. 3. The robot is tethered, and all data is transferred to a laptop computer via an Ethernet cable. A 3D-printed calibration board for scanning accuracy analysis is also shown in the figure, which is introduced later in Sec. IV-C.

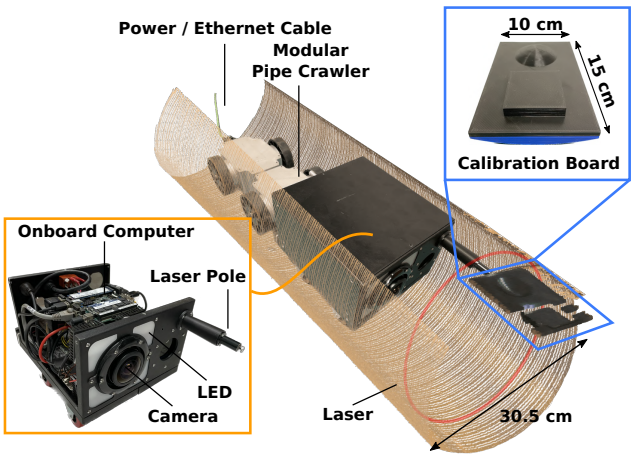

Fig. 3. A photo of the sensor payload with the case open, together with a composed image where the robot is surrounded by the scanned pipeline point cloud to demonstrate the working scenario. The calibration board is attached on the pipe's inner surface.

The image dataset to train the optical flow estimator is collected from a steel 12-inch diameter natural gas pipe using the robot. The training set consists of 825 consecutive images, with a raw resolution of 1232×1028. At the pre-processing stage, all images are cropped and transformed to the Polar coordinate as described in Sec. III-A. The testing set has 500 images collected from another pipe segment, which has different textures from the training pipe. In order to get the ground truth images without laser, the laser projector is turned off when collecting the training and testing set. We fine-tune the pre-trained FastFlowNet using the Adam optimizer with a learning rate of 1e-4, a beta of 0.999, and a momentum of 0.9. The model is trained for 100 epochs, and the batch size is set to 32. More data with different real-world natural gas pipes (provided by Peoples Gas Company, Pittsburgh, PA) are collected to evaluate the influence on the downstream task.

B. Image Reconstruction Quality Evaluation

The image reconstruction quality is studied by measuring the error between the algorithm's output image and the ground truth. Four quantitative metrics are used, including Peak Signal-to-Noise Ratio (PSNR), Mean Squared Error (MSE), Structural Similarity Index Measure (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS). PSNR and MSE are used to measure the pixel-wise error. SSIM measures the structural similarity between the reconstructed image and the ground truth. LPIPS uses a pre-trained VGG [31] network to calculate the perceptual similarity. This metric is closer to human judgments [32].

In Tab. I, we compare our proposed algorithm with four Video or Image Inpainting baseline algorithms: DFGV [15], FGVC [16], DeepFill V2[29], and TELEA [30]. Our algorithm only uses past information, and thus it is an online method. DeepFill V2 and TELEA are single-frame Image Inpainting algorithms, so they are also online. DFGV and FGVC are two flow-based methods and both use bi-directional optical flows to refine the results. These two

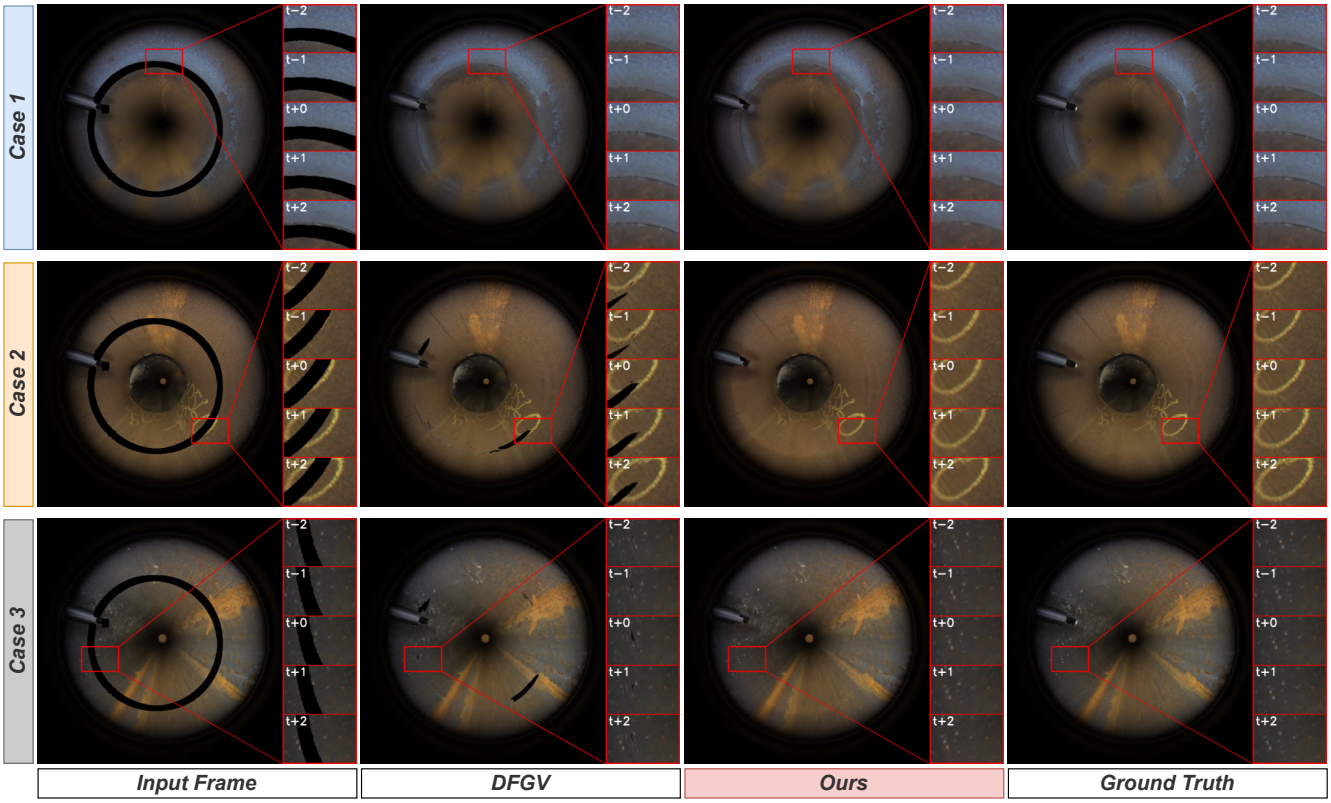

Fig. 4. Visualization of the Video Inpainting results. Our method is able to correctly reconstruct the detailed features inside the corrupted region. DFGV[15] fails for some regions because of the inaccurate optical flow estimation. Some interesting regions are zoomed in and the image sequences are shown.

TABLE I

IMAGE RECONSTRUCTION QUALITY EVALUATION.

Methods	Online	Device	FPS ↑	PSNR ↑	MSE (10^{-5}) ↓	SSIM ↑	LPIPS (10^{-3}) ↓
DFGV [15]	✗	GPU	0.43	43.74	4.24	0.990	6.82
FGVC [16]	✗	GPU	0.068	44.12	3.97	0.993	6.10
DeepFill V2 [29]	✓	GPU	1.9	33.27	47.65	0.980	20.38
TELEA [30]	✓	CPU	13.8	40.06	10.99	0.980	15.07
Ours	✓	GPU	23.3	42.38	5.84	0.992	5.66

↑: higher is better; ↓: lower is better; Red colored metrics are less preferable.

methods cannot be directly used for online applications. A more recent end-to-end offline model, E^{2}FGVI [17], fails to execute on our experiment computer, as it requires too much GPU memory.

The speed tests are performed on a laptop computer with an Intel i7 CPU and an Nvidia RTX 3080 laptop GPU (8GB GPU memory). All deep learning methods are executed on the GPU. We use the OpenCV version of TELEA, which is a CPU implementation. However, we expect it can run faster with GPU acceleration. For comparison fairness, the numbers of iterations for DFGV and FGVC are set to 1, and the deep Image Inpainting modules in these models are removed. All models are implemented in vanilla PyTorch, i.e., no mixed precision, TensorRT, or other acceleration techniques are used. We find our method is significantly faster than all the baseline methods and reached an FPS of 23.3. Such a speed is sufficient for the SLAM real-time requirement. The coordinate transformation component plays a key role to reduce the computation cost: without this component, the FPS drops to 19.9. All experiments are conducted at the image resolution of 1024×1024.

The results in Tab. I show that our method has comparable or better image reconstruction quality with the two offline models. The three Video Inpainting algorithms have notably better quality compared with the two single-frame Image inpainting methods for all four metrics. Our method has slightly worse results on the two pixel-wise metrics, PSNR and MSE, compared with DFGV and FGVC. We suspect this is because our method can only utilize unidirectional optical flow, and thus cannot smooth the pixel intensity like bi-directional methods. Although the pixel intensity is not perfect, the detailed features can be reconstructed. We observe the two structural similarity metrics, SSIM and LPIPS are comparable or even slightly better than DFGV

and FGVC. For our point cloud colorization problem, we care more about the feature structural accuracy on the laser strip. The absolute pixel brightness is less important in our application.

Qualitative results are visualized in Fig. 4. For each case, we visualize the input image, the result from DFGV, the output image from our method, and the ground truth. The main image shows the complete image at the time stamp t. We show a zoom-in view of an interesting region for each case with a time sequence from $t-2$ to $t+2$. The proposed method can reconstruct the corrupted region with high accuracy, even for very detailed features.

C. 3D Scanning Accuracy Comparison

We hypothesize that our algorithm does not induce a major negative influence on the 3D scanning accuracy for VLI-SLAM. This can be verified using a 3D-printed calibration board as the to-be-scanned ground truth object. The calibration board has a flat surface with a square and a spherical protrusion with known dimensions and is attached to the pipe's inner surface. When the robot is traveling through the pipe it scans the calibration board and generates a point cloud, as shown in Fig. 3. The Iterative Closest Point (ICP) [33] algorithm is performed to register the scanned point cloud and the ground truth Computer-Aided Design (CAD) model. The root-mean-square (RMS) error is reported in Tab. II. We observed that both the laser alternating method used in the original VLI-SLAM and the proposed Video Inpainting algorithm have sub-mm level scanning accuracy with only 1.1% difference, which indicates the influence of the proposed method is negligible and supports our hypothesis.

TABLE II
3D SCANNING ERROR

Method	RMS (mm)
Laser Alternating + VLI-SLAM	0.910
Video Inpainting + VLI-SLAM	0.927

D. Localization and RGB-D Reconstruction Analysis

For longer-range pipe scanning, the accuracy of robot localization is important, because if the odometry drifts the scanned pipe will bend in the wrong direction. Fig. 5 shows the Absolute Trajectory Error (ATE) [34] of the original VLI-SLAM and the modified version. The ground truth robot trajectory is measured by using a Leica total station to track a prism mounted on the top of the robot. To test the algorithm's generalization capability, we repeated the experiments on four real-world natural gas pipe segments with different textures. The estimated robot trajectories and the ground truth are first aligned with the Horn's method [35], and the Euclidean distances between each pair of matched points are visualized in the plot with respect to the position along the pipe's axial direction. We find that both methods have relatively small drifts. It's interesting that the proposed method has a slightly better performance compared with the original VLI-SLAM algorithm. This is more significant for the feature-less pipe (pipe B). One potential reason is that after propagating the features following the optical flow, the features in the output image are more easily to be detected and tracked by the SLAM feature tracking module. However, more investigation needs to be conducted in the future to find the root cause. In conclusion, the proposed Video Inpainting algorithm doesn't negatively affect the robot localization accuracy, and the RGB-D pipeline reconstruction is able to reveal the pipeline texture.

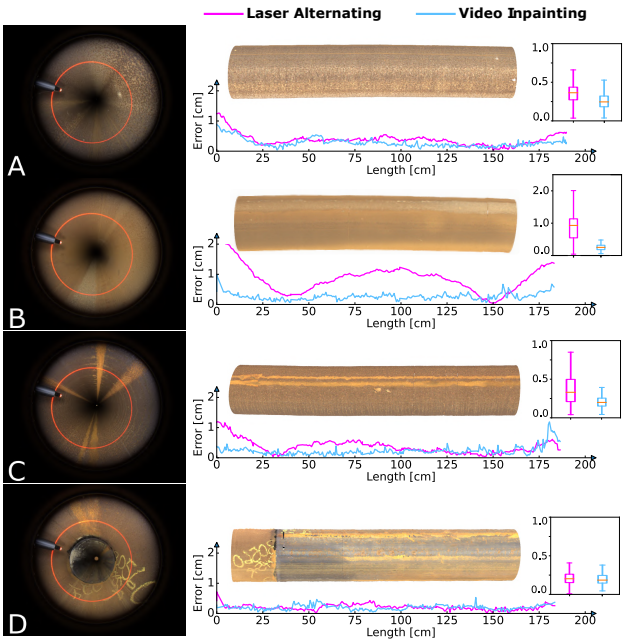

Fig. 5. The localization error and RGB-D reconstruction visualization. The box plots show the mean and spread of the errors. Input data is from the test set, which is not exposed to the model during training.

V. CONCLUSION, LIMITATION, AND FUTURE WORK

In this paper, we present a novel algorithm for RGB-D pipeline reconstruction by framing the color restoration problem as a Video Inpainting task. To our knowledge, this is the first real-time Video Inpainting algorithm applicable to in-pipe environments, making it highly valuable for constructing compact RGB-D inspection systems for the pipeline industry. Real-world experiments showcase that our algorithm surpasses other inpainting methods in terms of accuracy and real-time performance. Moreover, our approach supports demanding downstream tasks, such as SLAM, and can be easily integrated to existing visual-laser-based pipeline inspection systems with minimal hardware changes.

While our method offers several advantages, there are still limitations that can be addressed in future work. The first limitation arises from our reliance on the brightness consistency assumption in the optical flow algorithm. This assumption causes a slight boundary effect around the laser region in the output image, as shown in Fig 4. Although this issue does not significantly affect our primary goal of estimating textures inside the laser region and has no noticeable

impact on SLAM performance, it may have more pronounced effects on downstream tasks that are more sensitive to such artifacts. To mitigate this issue, we plan to explore the use of Image Harmonization [36] techniques to more effectively and efficiently blend the inpainted region into the original image and produce a more seamless result. Another limitation is that due to the lack of accessible pipes, our current dataset does not include images with severe geometric or material defects. To address this limitation, we intend to expand the range of testing pipe samples and damage conditions in the next phase of our research.

REFERENCES

[1] Bureau of Transportation Statistics, "U.s. oil and gas pipeline mileage," https://www.bts.gov/content/us-oil-and-gas-pipeline-mileage. 1

[2] W. Wang, X. Mao, H. Liang, D. Yang, J. Zhang, and S. Liu, "Experimental research on in-pipe leaks detection of acoustic signature in gas pipelines based on the artificial neural network," *Measurement*, vol. 183, p. 109875, 2021. 1

[3] Y. Liu, X. Zhang, Y. Li, G. Liang, Y. Jiang, L. Qiu, H. Tang, F. Xie, W. Yao, Y. Dai, *et al.*, "Videopipe 2022 challenge: Real-world video understanding for urban pipe inspection," in *2022 26th International Conference on Pattern Recognition (ICPR)*. IEEE, 2022, pp. 4967–4973. 1

[4] C. H. Bahnsen, A. S. Johansen, M. P. Philipsen, J. W. Henriksen, K. Nasrollahi, and T. B. Moeslund, "3d sensors for sewer inspection: A quantitative review and analysis," *Sensors*, vol. 21, no. 7, p. 2553, 2021. 1, 2

[5] A. Gunatilake, L. Piyathilaka, A. Tran, V. K. Vishwanathan, K. Thiyagarajan, and S. Kodagoda, "Stereo vision combined with laser profiling for mapping of pipeline internal defects," *IEEE Sensors Journal*, vol. 21, no. 10, pp. 11 926–11 934, 2020. 1, 2

[6] D. Cheng, H. Shi, A. Xu, M. Schwerin, M. Crivella, L. Li, and H. Choset, "Visual-laser-inertial slam using a compact 3d scanner for confined space," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 5699–5705. 1, 2, 5

[7] D. Kim, S. Woo, J.-Y. Lee, and I. S. Kweon, "Deep video inpainting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5792–5801. 2

[8] F. Mascarich, S. Khattak, C. Papachristos, and K. Alexis, "A multi-modal mapping unit for autonomous exploration and mapping of underground tunnels," in *2018 IEEE aerospace conference*. IEEE, 2018, pp. 1–7. 2

[9] R. Fernandes, T. L. Rocha, H. Azpúrua, G. Pessin, A. A. Neto, and G. Freitas, "Investigation of visual reconstruction techniques using mobile robots in confined environments," in *2020 Latin American Robotics Symposium (LARS), 2020 Brazilian Symposium on Robotics (SBR) and 2020 Workshop on Robotics in Education (WRE)*. IEEE, 2020, pp. 1–6. 2

[10] Z. Shang and Z. Shen, "Single-pass inline pipeline 3d reconstruction using depth camera array," *Automation in Construction*, vol. 138, p. 104231, 2022. 2

[11] N. Stanić, M. Lepot, M. Catieau, J. Langeveld, and F. H. Clemens, "A technology for sewer pipe inspection (part 1): Design, calibration, corrections and potential application of a laser profiler," *Automation in Construction*, vol. 75, pp. 91–107, 2017. 2

[12] M. Lepot, N. Stanić, and F. H. Clemens, "A technology for sewer pipe inspection (part 2): Experimental assessment of a new laser profiler for sewer defect detection and quantification," *Automation in Construction*, vol. 73, pp. 1–11, 2017. 2

[13] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018. 2

[14] Z. Qin, Q. Zeng, Y. Zong, and F. Xu, "Image inpainting based on deep learning: A review," *Displays*, vol. 69, p. 102028, 2021. 2

[15] R. Xu, X. Li, B. Zhou, and C. C. Loy, "Deep flow-guided video inpainting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3723–3732. 2, 5, 6

[16] C. Gao, A. Saraf, J.-B. Huang, and J. Kopf, "Flow-edge guided video completion," in *European Conference on Computer Vision*. Springer, 2020, pp. 713–729. 2, 5, 6

[17] Z. Li, C.-Z. Lu, J. Qin, C.-L. Guo, and M.-M. Cheng, "Towards an end-to-end framework for flow-guided video inpainting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 17 562–17 571. 2, 6

[18] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2462–2470. 2

[19] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *European conference on computer vision*. Springer, 2020, pp. 402–419. 2

[20] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4040–4048. 2

[21] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2015. 2, 4

[22] S. Meister, J. Hur, and S. Roth, "Unflow: Unsupervised learning of optical flow with a bidirectional census loss," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018. 2

[23] L. Liu, J. Zhang, R. He, Y. Liu, Y. Wang, Y. Tai, D. Luo, C. Wang, J. Li, and F. Huang, "Learning by analogy: Reliable supervision from transformations for unsupervised optical flow estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6489–6498. 2, 4

[24] L. Kong, C. Shen, and J. Yang, "Fastflownet: A lightweight network for fast optical flow estimation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 10 310–10 316. 2, 3, 4

[25] M. Menze, C. Heipke, and A. Geiger, "Joint 3d estimation of vehicles and scene flow," in *ISPRS Workshop on Image Sequence Analysis (ISA)*, 2015. 4

[26] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *European Conf. on Computer Vision (ECCV)*, ser. Part IV, LNCS 7577, A. Fitzgibbon et al. (Eds.), Ed. Springer-Verlag, Oct. 2012, pp. 611–625. 4

[27] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6848–6856. 4

[28] C. De Boor and C. De Boor, *A practical guide to splines*. springer-verlag New York, 1978, vol. 27. 5

[29] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Free-form image inpainting with gated convolution," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 4471–4480. 5, 6

[30] A. Telea, "An image inpainting technique based on the fast marching method," *Journal of graphics tools*, vol. 9, no. 1, pp. 23–34, 2004. 5, 6

[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. 5

[32] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *CVPR*, 2018. 5

[33] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. Spie, 1992, pp. 586–606. 7

[34] D. Prokhorov, D. Zhukov, O. Barinova, K. Anton, and A. Vorontsova, "Measuring robustness of visual slam," in *2019 16th International Conference on Machine Vision Applications (MVA)*. IEEE, 2019, pp. 1–6. 7

[35] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Josa a*, vol. 4, no. 4, pp. 629–642, 1987. 7

[36] Y.-H. Tsai, X. Shen, Z. Lin, K. Sunkavalli, X. Lu, and M.-H. Yang, "Deep image harmonization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3789–3797. 8