DMB

DATA MANAGEMENT AND BIOMETRICS

APPLYING A MODEL-BASED APPROACH FOR VISUAL RECONSTRUCTION OF A CHALLENGING INTERNAL PIPELINE ENVIRONMENT

Jasper Bovenkerk

MASTER'S THESIS ASSIGNMENT

Committee:
Nicola Strisciuglio
Le Viet Duc

August, 2024

Data Management and Biometrics
EEMathCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

Applying a model-based approach for visual reconstruction of a challenging internal pipeline environment

Jasper Bovenkerk

***Abstract*—Evaluation of water pipeline integrity is essential to prevent loss of water, restricted water access, and environmental damage. In most circumstances, Nondestructive Evaluation (NDE) is the desired approach. Inline inspection is the most practical, since pipelines are mostly located underground. Tools for these inspections can be equipped with many sensors, such as RGB cameras. However, pipeline environments put significant constraints on the tool regarding size.**

Separate cameras present a suboptimal view of the pipeline for human evaluators to work with. A better visual overview is required to ease and speed up the work of the human evaluators. Previous research has provided Simultaneous Localization And Mapping (SLAM)- and Structure-from-Motion (SfM)-based methodologies to perform similar tasks. However, those methods are not easily applicable in water pipeline environments, as the setup and environment differ. The current setup consists of more cameras with less overlap, and the pipeline is filled with water.

This research aims to present a methodology that can create a canvas view in a pipeline environment despite the challenges it brings. This is done by a model-based approach that is as independent from feature points as possible. A model-based approach that can stitch images radially is presented. This approach consists of 1. intrinsic and extrinsic camera calibration, utilizing the constraint of a cylindrical pipeline environment. 2. a model to reconstruct the pipeline radially. 3. a stitcher that stitches together the radial reconstructions.

Results show that the model-based approach works well when the parameters have been estimated correctly, which can be challenging for the setup of cameras with limited overlap. Extrinsic calibration inside the pipeline is good as long as a single location with enough reliable features can be found. Stitching of radial reconstructions is of lower quality at times due to a lack of texture in a pipeline environment. The proposed method is a start, but does not suffice to deal with all challenges of the environment.

***Keywords*—** N ondestructive Evaluation (NDE), fisheye cameras, model-based, reconstruction, multi-camera calibration, Structure-from-Motion (SfM),

I. Introduction

Sudden pipeline failures can be costly and problematic. Depending on the size of the pipeline and the transported substances, the damage can range from waste of material and flushed sediments to severe environmental damage and flooding [1]. In the US and Canada, 11-14 breaks per 100 miles of water mains are reported, and water loss due to failures is estimated to be around 10% [2]. Therefore, preventing such failures by inspecting pipelines for possible weak points is a highly researched topic. Such inspection aids in accurately determining the necessity of replacement, which can prevent high costs.

Inspecting pipelines occurs in two ways, from the outside or the inside. Inspection from the outside can take many forms, from excavating the area to using ground-penetrating sensors to observe the state of the pipeline. Excavating gives a good overview of the state of the pipeline, but is very expensive compared to other methods. Besides, it involves risks of damage to the pipeline or other structures in the area. A generally preferred approach is to perform a Nondestructive Evaluation (NDE), which can happen without taking apart the pipeline. An example of an aboveground NDE method would be ground penetrating radar [3]. The technique is very flexible in its usage, but is limited in accuracy for localizing leakages. [3]

As pipelines are generally located underground, the best option for inspection is from inside the pipeline. A tool to perform this inspection is called a PIG (pipeline inspection gauge). These tools again come in all kinds of forms and can contain various sensors such as lidar, different light spectra, ultrasound, magnetic flux sensors, and more [3], [4], [5].

Objective

Rosenxt[1], a company with a main focus on water pipeline inspection devices, is developing a new inline pipeline inspection tool for water pipelines employing RGB cameras. Currently, Rosenxt is directly providing the output of the cameras for inspecting the pipeline. Although this is functional, more advanced representations of the internal pipeline are desired to make inspection better and easier. The goal of the camera tool is to register the internal surface of the pipeline, such that the obtained imagery can be used to assess the state of the pipeline.

A canvas view or panoramic view, where all images in an area are stitched together would be useful to make evaluating the pipeline easier, faster, and more reliable. This view allows a human evaluator to more easily see installations, pipeline connections, and anomalies in the pipeline.

The goal of this work can be summarized as follows: obtain a combined view from multiple cameras, allowing for easier and more accurate human evaluation of pipeline inspection imagery.

Challenges

The tool developed by Rosenxt has a different composition compared to similar tools. First of all, there are different types

[1]https://rosen-nxt.com/

of cameras: regular RGB cameras, RGB-D cameras (RGB with a depth estimation), and stereo cameras. Next, the positioning and the amount of cameras can vary. Lastly, the water inside the pipeline is also expected to have a significant impact.

Quite often cameras with fisheye lenses are used, as they can capture large areas of interest with their wide Field of View (FoV). This is an advantage over RGB-D cameras, which tend to have a more limited FoV. A fisheye lens is effectively used by Zhang et al. to obtain a 3D reconstruction utilizing SfM (Structure from Motion) [6]. However, Shang et al. show that the use of 4 RGB-D cameras that are all slightly rotated towards the pipeline wall can also be used effectively [7]. They make use of the depth capabilities of the cameras and visual odometry to obtain a 3D reconstruction of the pipeline.

The tool currently under development at Rosenxt, is designed to work with water inside the pipeline. This limits the use of RGB-D cameras, as they will become less accurate and have a significantly limited range in water [8]. Therefore, the tool was developed to work with RGB cameras. These cameras have fisheye lenses to capture a large amount of the pipeline at the same time.

The water in the pipeline also brings additional challenges such as particles that are floating around in the water, which may obstruct the camera view. Also, the lighting available may be limited.

The space in a pipeline is very limited, so the tools are restricted in size. Due to these size restrictions, the number of cameras is limited, which leads to limited overlap between the FoV of the different cameras. This is another reason why the choice of fisheye cameras is relevant, as regular lenses would have provided even less overlap or none at all. Still, the small overlapping area could be a limiting factor for many approaches, as they require common points between the cameras to work.

Additional information about the environment is often used to obtain better results. In the case of operating inside a pipeline, this could be the constraint for a cylindrical shape. This was already applied to visual Simultaneous Localization And Mapping (vSLAM) and SfM successfully multiple times, yet in somewhat different setups. Single front-facing cameras were used in other work as opposed to multiple radially oriented cameras in this work. [9], [10], [11]

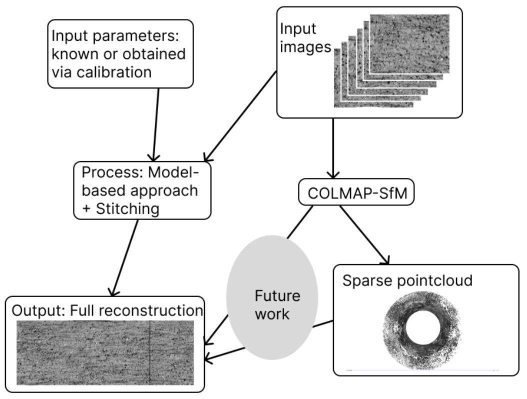
The tool not only has cameras, but also contains Inertial Measurement Units (IMUs) and flippers that measure the offset of the tool with respect to the pipeline walls. These sensors could be used as supporting information for the algorithms. However, relying on the cameras is preferred, as additional sensors require additional calibration.

The challenges can be summarized as follows:

- Limited space, thus a limited number of cameras and limited overlap between the cameras.
- Particles in the water and limited lighting.
- Different setup to what was proven to work previously in the literature.

Contributions

A rough overview of the inputs, processes, and results can be found in Figure 1. The contributions of this thesis can be summarized as follows:



Input parameters: known or obtained via calibration
Input images
Process: Model-based approach + Stitching
COLMAP-SfM
Future work
Sparse pointcloud
Output: Full reconstruction

Fig. 1: A rough overview of inputs, outputs, and processes utilized in this work.

- a model-based method is proposed to obtain a canvas view from separate imagery, which can better deal with the challenging environment of a pipeline than general approaches;
- several camera calibration methods are tested;
- the different methods are qualitatively and quantitatively evaluated on data from the field, data from the workshop, and simulated data;
- a feasibility test was done for applying SfM on this setup.

Structure

The structure of this report will be as follows: in Section II some important background technologies are discussed. In Section III related work will be discussed that tried to tackle similar problems. Based on this information, research questions are presented in Section IV. In Section V the approach is formulated. Then, Section VI discusses the available data. After that, the results are presented in Section VII. After which a discussion will follow in Section VIII, along with ideas for future work. The final section, Section IX, will present the conclusions.

II. Background

A. Camera model

In computer vision, the camera model is relevant to understanding image registration and the influence of camera characteristics on the resulting image. The basic pinhole camera model consists of two matrices of parameters that convert a point in 3D to an image coordinate. This can be written as Equation 1 and in shorter notation as Equation 2. [12]

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} fx & 0 & Cx \\ 0 & fy & Cy \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (1)$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = C_{int} \cdot C_{ext} \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{2}$$

u and v describe the image coordinates, and the additional 1 makes it a homogeneous representation. [12]

C_{int} is called the intrinsic camera matrix. It describes how a point in camera coordinates is mapped to image coordinates. For this purpose, it contains 4 parameters, the focal length in the x and y direction (fx and fy), and the x and y coordinates of the center of projection (Cx and Cy). The focal length is related to zoom and is usually approximately equal in the x and y direction. The center of projection describes where the middle of the FoV ends up on the image. In the ideal case, this is perfectly in the center, but usually varies a little bit. [12]

C_{ext} is the extrinsic camera matrix. It describes how to convert points from world coordinates to camera coordinates. Therefore, it is a representation of the position of the camera. It consists of a translation, expressed as a translation vector (tx, ty, and tz), and a rotation, expressed as a rotation matrix (r_{11}-r_{33}). [12]

The conversion between world coordinates and image coordinates drops information, so inverting the process is not possible. If the inverse equation is calculated, it only returns a point on the line between the pinhole and the actual point. Using this, the light ray that hits the sensitive plate can be determined, but the exact point from where it originates cannot. [12]

This camera model is usually extended by distortions caused by the lens. In this scenario fisheye lenses are used, which have their own set of parameters to model them, k1 up to k4. These are as used by OpenCV [13], which is based on a paper by Kannala et al. [14]. How these parameters work can be seen in the Equations 3 to 7 [13]. These start with x, y, and z which are the coordinates of a point expressed in camera coordinates. The resulting x' and y' represent the undistorted location of the point in camera coordinates. It can be converted to image coordinates by applying the camera intrinsic matrix to the homogeneous version of this point.

$$a = x/z, \quad b = y/z \tag{3}$$

$$r^2 = a^2 + b^2 \tag{4}$$

$$\theta = \arctan r \tag{5}$$

$$\theta_d = \theta(1 + k_1\theta^2 + k_2\theta^4 + k_3\theta^6 + k_4\theta^8) \tag{6}$$

$$x' = (\theta_d/r)a, \quad y' = (\theta_d/r)b \tag{7}$$

B. Bundle Adjustment

Bundle adjustment (BA) is the problem of obtaining jointly optimal viewing parameters (intrinsic, extrinsic, and distortion parameters) and 3D points. In its most simple form, it is about minimizing the reprojection error using least-squares optimization. This reprojection error is the difference between a point on the actual images and the location based on a projection from the 3D point onto the image plane. This reprojection is done using the camera model and the estimated viewing parameters. The smaller this error, the more accurately the obtained parameters model the camera and environment where the image was taken. [15]

The reprojection error is written down in Equation 8. Where $\mathbf{X}_j$ is the j-th 3D point and q_{ij} is the 2D observation of $\mathbf{X}_j$ from the i-th view $\mathbf{P}_i$. $\mathbf{K}_i$ includes the camera parameters (intrinsic, extrinsic, and distortion) belonging to the i-th view and π is a function that projects scene points to the image plane. [9]

BA is commonly used to obtain and refine camera parameters or to reconstruct points in a 3D space. This includes intrinsic and extrinsic camera calibration, SLAM, SfM, Visual Odometry (VO), and more. [15]

$$E_{rep}(\mathbf{X}, \mathbf{P}, \mathbf{K}) = \sum_{i \in \mathbf{P}} \sum_{j \in \mathbf{X}} (\|q_{ij} - \pi(\mathbf{X}_j, \mathbf{P}_i, \mathbf{K}_i)\|) \tag{8}$$

C. Features

Features are of high importance in the field of computer vision. They are commonly used to detect certain things or aspects in an image. These detected features can then be described uniquely so that they can be matched when the same point is described again. However, many different feature descriptors have been developed in the past years. These include, but are not limited to, SURF [16], SIFT [17], BRIEF [18], FAST [19], and ORB [20]. SIFT is one of the older, but most robust detectors that exists. Since it is so robust, it is still commonly used. The main disadvantage is the relatively slow calculation compared to other feature descriptors. For exactly this reason, ORB features are also becoming a commonly chosen feature descriptor. This is mainly in real-time applications, as speed is ORB's biggest advantage. In aspects like accuracy and robustness, it can achieve performance equal to SIFT. [21]

In recent years, another set of features has emerged as state-of-the-art. Neural networks can be trained to detect more and more robust features than classical methods. An example of this is SuperPoint [22], which uses self-supervised learning of a CNN to detect and describe points. Also matching these features has become a task where machine learning approaches outperform classical ones, with algorithms such as SuperGlue [23].

D. Camera calibration

Camera calibration is the process of determining the camera parameters. As this is vital for many applications in computer vision it is a frequently researched topic. In general, to obtain the parameters it is required to have some known points in the images. These points can be obtained by detecting them in the environment, which is in the realm of self-calibration. Self-calibration does not require any additional setup. It is also possible to have a separate calibration phase, with objects placed in the environment that are known and easy to detect. Additionally, it is also possible to use the movement of the constellation to have them within the FoV of each camera. However, this requires quite precise knowledge of the movement, which is difficult to achieve. [12]

Detecting the features in the environment is done using feature descriptors such as SIFT and ORB. These features are matched and then the camera parameters can be optimized using BA. It is also possible to have some patterns placed in the environment to create more features than naturally present. [12]

A common way to perform reliable intrinsic calibration is to present a checkerboard in front of the camera. This pattern can easily be recognized. The fact that the placement of the points in a plane is known, allows to determine how they are processed to end up on the image. An important requirement for this calibration is that the checkerboard should reach as close to the corners as possible to also model the distortions at the edges of the camera frame. [12]

This checkerboard approach can also be used for extrinsic calibration. The checkerboard is detected by two or more cameras and the corners of the checkerboard become the matching points between the images. This is a very robust approach, but it requires the checkerboard to be visible on both cameras. Besides, the bigger part of the view the checkerboard occupies, the more accurately the corner points can be detected. This means that limited overlap between the cameras is a disadvantage to the accuracy of this method. [12]

E. *SfM*

SLAM and SfM are two commonly used techniques in the field of computer vision that aim to navigate and reconstruct unknown environments using data from cameras. Reconstructing the environment can be based, either on feature point matching or on dense methods, where all pixels are used. For sparse methods, the matched features are passed to BA algorithms to obtain the camera parameters and the 3D coordinates of the features. This is not only done on a local scale, but also for the entire environment that is to be reconstructed. It is attempted to find loops in the trajectory and make the multiple passes align with one another. Some of the state-of-the-art algorithms in this area include ORB-SLAM3 [24] and COLMAP [25]. ORB-SLAM3 is a real-time SLAM algorithm that combines IMU data with any possible camera (monocular, stereo, RGB-D, pinhole, fisheye). It has advanced algorithms for loop-closing and relocalization, resulting in a very robust system for general SLAM tasks. COLMAP is an incremental SfM approach, that was designed to be a general-purpose SfM that combines many of the state-of-the-art techniques. COLMAP has a practical advantage, as it has more options for reusing the algorithms easily.

III. Related work

A. *'Simple' stitching*

Image stitching is a well-researched field of image processing, with many techniques available for panorama and mosaic stitching [21]. The general steps for these techniques can be described as follows:

1) Calibration, this step includes obtaining the intrinsic and extrinsic camera parameters.
2) Registration, which is aligning two or more images that are captured from different perspectives.
3) Blending, which is combining the images to form one as seamless as possible image.

These steps are required to solve the problem at hand, but the environment limits the effectiveness of many commonly used approaches. The most effective way to stitch two images is to first extract features, which are most often SIFT features. These features can then be matched between the different images. These matches tell us something about how these images relate to one another, which can be captured in a homography matrix. This matrix is obtained by a combination of RANSAC and BA. [21]

This methodology works well and is generally very robust, but it requires an overlapping area with good features. This is unfortunately not the case. As discussed previously, the tool will have limited overlap due to limitations on tool size. In addition, while some areas of the pipelines inner walls have a decent amount of texture, other parts are completely featureless. Available implementations of this kind of stitching, such as in OpenCV [13], are unable to successfully stitch together the pipeline images.

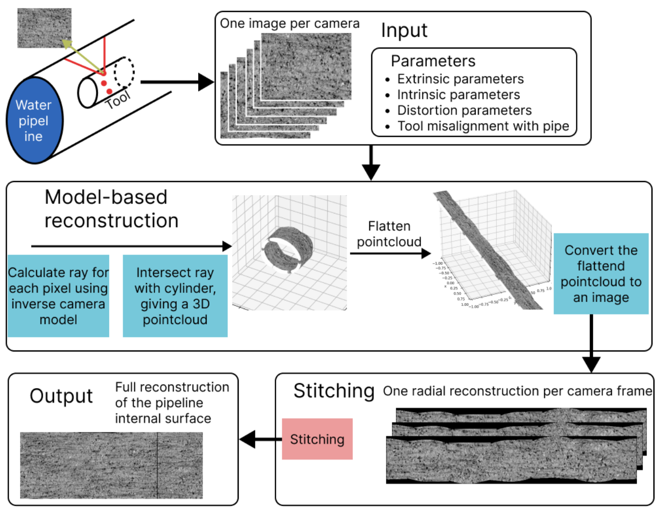
B. *SfM in a pipeline environment*

As discussed before, a pipeline environment is especially challenging since it is quite narrow, with little overlap in FoV, and potentially limited or repetitive features. This has not refrained researchers from finding effective ways to use SfM in a pipeline environment.

Several approaches have been made involving feature-based techniques. Shang et al. use feature-based VO to determine the camera trajectory. They use the depth estimation of the RGB-D cameras to deal with the challenges of working inside a pipeline, such as triangulation which may not work well due to limited texture. They also make a step from sparse to dense reconstruction by combining the camera pose and known depth information for each pixel. [7]

Hansen et al. did something very similar but with a single fisheye camera. They also use feature-matching and VO, but they detect straight sections in the pipeline. With these straight sections, they make a new estimation of the positions of the cameras and reconstruction of the pipeline with the error functions adjusted for the deviation from the cylindrical shape. They again make a dense reconstruction from the sparse one. [10]

Interesting to mention is that Hansen et al. did consider placing the cameras in a radial structure. However, they refrained from doing so because it would require more cameras



Water pipeline
Tool
One image per camera
Input
Parameters
Extrinsic parameters
Intrinsic parameters
Distortion parameters
Tool misalignment with pipe
Model-based reconstruction
Calculate ray for each pixel using inverse camera model
Intersect ray with cylinder, giving a 3D pointcloud
Flatten pointcloud
Convert the flattend pointcloud to an image
Output
Full reconstruction of the pipeline internal surface
Stitching
One radial reconstruction per camera frame
Stitching

Fig. 2: Schematic of the model-based approach. The parameters are considered known at this point and estimating them is not shown in this figure.

to capture the entire pipeline as opposed to the current front-facing camera. They made the consideration that this would most likely require too much space to operate the tool in smaller pipelines. [10]

Zhang et al. did something quite similar to Hansen et al., with sparse SLAM and cylindrical constraints. All three approaches show improved performance compared to less sophisticated algorithms like standard ORB-SLAM. [11]

IV. Research Questions

This research aims to answer the following questions:

RQ1 *How can the extrinsic camera parameters of the tool with radially oriented cameras be determined effectively?*

RQ2 *How effectively can a pipeline be reconstructed using a model-based approach with the assumption that the relative position of the tool to the pipeline is known?*

With the following sub-questions.

RQ2.1 *How effectively can a pipeline be reconstructed using a model-based approach at a single instance, combining one image from each camera?*

RQ2.2 *How effectively can a pipeline reconstructed using a model-based approach be extended to combine multiple radial instances over time into one?*

RQ3 *How feasible is a reconstruction approach based on SfM in a pipeline scenario?*

V. Methodology

A. Model-based approach

The first approach is the model-based approach, which aims to operate as independently from the textures in the images as possible. This is because it is considered likely that the textures will be lacking at some point in a run. It consists of multiple steps, where different approaches are tried. A schematic overview of this approach is shown in Figure 2.

Intrinsic Calibration: The intrinsic parameters were obtained by the classical approach of checkerboard calibration. For this, the OpenCV library [13] was used. This is a commonly used library, with reliable implementations. The implementations rely on some more classical methods, but the resulting parameters are expected to be accurate enough. To verify if the obtained parameters and distortions are approximately correct, the focal length was recalculated based on the calibration results and compared to the known values for the cameras. Also, the images of the checkerboards were reprojected, which should again produce a square checkerboard with perfectly straight lines if the camera parameters and distortions were accurately determined. It was verified that this approach is good enough for this use, but there is room for improvement.

Extrinsic Calibration: The extrinsic camera parameters were obtained by several different methods. First, the extrinsic parameters were obtained from the construction information of the tool. The design of the tool is known and from that, the extrinsic parameters can be determined. This assumes an ideal scenario, in which the tool was assembled perfectly. Therefore it is likely that the exact extrinsic parameters will deviate from this. Nevertheless, it is a good starting point. These extrinsic parameters will be referred to as initial extrinsic parameters.

The second methodology involves detecting and matching feature points in the overlapping parts of adjacent cameras. This feature detection was done based on SIFT, as it proved most reliable in the pipeline environment compared to other classical feature descriptors. The obtained matches could be filtered based on where the matches were located on the images, as it is known which parts of the cameras overlap. This already removed the largest part of outliers. Further outliers were removed by calculating the reprojection error with the initial parameters, the outliers have errors that are of several orders higher than inliers. The found correspondences between the cameras can then be used to refine the camera positions. The initial extrinsic parameters from the geometry of the tool were taken as a starting point. Then, triangulation can be used to estimate the approximate 3D coordinates of the matched points. The initial extrinsic parameters and 3D positions can then be used as input to BA to minimize the reprojection error.

This approach resulted in scale issues. The starting scale would be correct because the scale of the initial values was correct. However, there would be a slight drift during the BA process. To counter this, constraints were placed on the locations of the 3D points. For each point, the distance to the cylinder of known radius was calculated and the square of this distance was added to the loss function of the BA. This is described in Equation 9. $\mathbf{X}_i$ represents the i-th 3D point and r is the known radius of the pipeline. d is a function that calculates the distance to the centerline of the cylinder.

$$E_{cyl}(\mathbf{X}, r) = \sum_{i \in \mathbf{X}} \|r - d(\mathbf{X}_i)\| \tag{9}$$

The full loss function is a combination of the reprojection error as described in Equation 8 and the cylindrical constraint in Equation 9. To balance them a parameter α is used. This can be seen in Equation 10. For α, the value of 10000 was found

Parameter	Value
Method	Trust region reflective algorithm
Loss	Linear
Bounds	Orientation: $\pm 5°$, Translation: ± 1cm

TABLE I: Settings for BA using SciPy least_squares.

to work well in most scenarios, but tuning may be required in certain situations.

$$E_{loss} = E_{rep} + \alpha E_{cyl} \tag{10}$$

The third method utilizes checkerboards in the overlapping FoV of adjacent cameras to obtain matching points. The checkerboard can be detected in the images of both the left and right camera and then all the detected cornerpoints form pairs of matching image coordinates. These can then again be used as input for the BA process. The advantage of this method is that detecting the checkerboards is a quite reliable process, which means that the resulting matches are unlikely to have outliers.

The BA algorithm was based on the least squares solver available in the SciPy package[26]. In Table I, the used settings are shown. Some experiments were done with different loss settings, but Linear performed best. The bounds were added because the model would occasionally find a minimum with impossible camera positions. The bounds have a broad range and the actual value is expected to lie well within this range. The bounds should generally not be reached if the initialization is done correctly.

Reconstruction: The obtained information on the camera's intrinsic parameters, distortions, and extrinsic parameters, can now be used to make a reconstruction. This also requires information about the pipeline diameter and the relative position and orientation of the tool to the pipeline. The pipeline diameter is considered known in this scenario and the tool is assumed to be centered, as during runs the tool is quite stable in the center most of the time. With all this information every pixel of this image can be mapped to its 3D location. Through the inverse of Equation 1, a pixel cannot be mapped to the original point in 3D space directly. The inverse Equation produces a point through which a ray passes from the pinhole. This ray does go through the original point, but also through infinitely many other points. An additional constraint is required, which can be given by the known cylinder from which the point originates. Finding the intersection between the ray and the cylinder then rewards the original point in 3D. Repeating this process for every pixel in every image produces a point cloud of the pipeline.

To obtain a good view of the pipeline, this point cloud was converted to an image. This was done by first unwrapping the cylinder to a flat plane and then by interpolating the points to produce an image. For this, linear interpolation was used as it is fast. Also, more refined interpolation ran into issues with memory requirements.

After the reconstruction of a single instance, these instances could be combined to create a full reconstruction of the pipeline. For this, an estimation of the relative movement between two moments in time is required. This can be complex to estimate, but there is the advantage that the movement is quite constrained. The tool is, in general, very stable in the pipeline; it is very close to the center and does not rotate very much. With this information, we can simplify the movement of the tool. In theory, the tool has 6 degrees of freedom in the pipeline. So, to obtain the optimal reconstruction it is required to determine the movement in each of these directions. This requires a lot of information, by feature points or other methods which are scarce in the pipeline. So there is the possibility to assume that the tool makes a simplified movement, which is easier to estimate. However, it also produces a larger error. So, when the information from the pipeline is limited, it may be necessary to use a simpler model.

Several methods were used to estimate the simplified movement of the camera. First, a perspective transform was estimated by matching feature points between two consecutive images. Since the framerate is high relative to the speed of the tool, there is a large amount of overlap between consecutive images making this approach viable in most situations. The second approach estimated pixel shift in the x and y directions based on the same matched features. This has the advantage that it could in theory already work with one correctly matched point, whereas the perspective transform requires at least 4. As opposed to feature-based methods, also some techniques that look at the entire image were investigated. These include cross-correlation and χ^2-shift. These have the advantage that they use more information, but the disadvantage is that much of the information they use is very similar.

When the shift is found, the images are blended by taking the median pixel value at each location. This is robust to small misalignments if not all images align perfectly. An additional effect is the removal of most of the particles floating around.

B. *SfM feasibility*

In addition to the work on the model-based approach, a small investigation into the feasibility of SfM for this problem was performed. For this, it was chosen to work with COLMAP, as it is a high-quality algorithm with an implementation in Python.

Reconstructions were performed on small sections from the actual runs, the test data, and the simulated data. COLMAP was run in sequential matching mode as the images were passed in the order of occurrence in the pipeline. COLMAP is also designed to be able to estimate the intrinsic camera parameters, but that does require a large amount of data. So, for now, it was given the intrinsic parameters from the checkerboard calibration.

VI. Data and experiments

For evaluation of the methods, Rosenxt provided some of their inspection runs with camera tools. This is useful because it can provide insight into how well the methodologies work on the real data. For evaluation purposes, however, this data is hard to work with. The runs were done at an earlier point in time, so obtaining any additional information from them is no longer possible. The intrinsic camera calibration was already done with the checkerboards, but no further tests are possible.

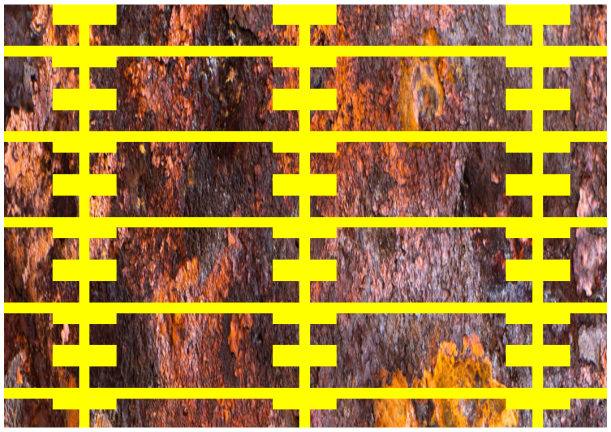


Fig. 3: Example of the evaluation pattern placed in the pipeline.

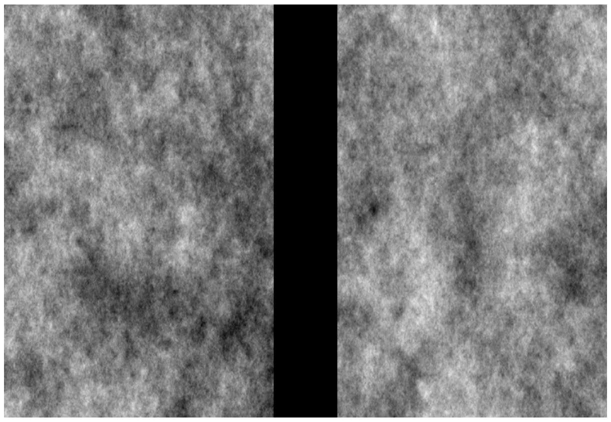


Fig. 4: Example of the random pattern for calibration placed in the pipeline.

In addition, there is no reference to what the pipeline looks like, so there is no ground truth available for evaluation. The data is also not fully practical as it was not recorded with this specific purpose in mind.

A. *Test setup*

To better evaluate the results a test was formulated to be done in a spool in the workshop. Here it is possible to have a good reference of what the pipeline looks like internally. In addition, the test was specifically for this purpose, so there was ample opportunity to do calibration.

The test setup consisted of a 694mm metal blue spool in a controlled environment in air. Also, the calibration for this was done in air consequently. The scenario remains very similar in air, only the FoV of the cameras is slightly different and the camera intrinsic parameters and distortion are different. A constellation of cameras could be placed inside this spool with an accuracy of approximately 1 cm. The tool was also placed level, so rotational misalignment should also be minimal. The setup did not include a full tool for practical reasons. The setup used is representative of a full tool and will be referred to as a tool in the following parts.

Calibration outside the pipeline: For calibration purposes, two recordings were made outside the pipeline. One recording where a checkerboard was placed in front of each camera separately following the general checkerboard calibration steps. And another where a checkerboard was placed in the overlapping area of each pair of cameras.

Recordings made inside the pipeline: Two different recordings were made inside the pipeline. One with the sheets for extrinsic calibration and one with the sheets for evaluating the reconstruction. The sheets for extrinsic calibration were designed such that they contain as many feature points as possible, which could be detected for the purpose of performing extrinsic calibration. It also included a straight bar, which mimics a connection and can be used for evaluating the reconstruction visually. An example can be seen in Figure 4. The sheets for evaluation were designed such that they are very suitable for visual evaluation. For this purpose, they contain many straight lines and distinguishable patterns. An example can be seen in Figure 3.

During these recordings, the tool was stepwise placed at different heights in the pipeline to simulate the tool passing through the pipeline. For this 5 positions were used. For each of these positions, the tool was placed in the center as accurately as possible.

Limitations to the tests: Unfortunately, the test data had its limitations. First, tests with the tool not in the center were impossible due to time limitations. This would have given interesting information about the behavior of the algorithms when the tool is not perfectly in the center at some known misalignment. Second, the sheets were not perfectly attached to the pipeline and might have moved a little during the repositioning of the tool. The effect of this is likely minimal, but it could cause some small misalignments in the reconstruction. Third, the tool was rotated a little between each position, this was not accounted for, but the algorithms should be able to handle this. As a final downside, even though the data was very nice for qualitative visual evaluation, the test data still did not provide a very accurate ground truth, which still made quantitative evaluation very hard. For this reason, a switch to simulated data was made as described in the next section.

B. *Simulation*

A simulation was done to obtain data similar to what is obtained by a tool, but with exact information on what the pipeline looks like. So for a start, a pipeline was created as a point cloud with any texture that may be wanted. In this pipeline, a tool could be placed in any desired position along with the cameras attached to the tool. These cameras would in principle be placed according to the tool specifications, but they could be varied to simulate any imperfections. For each camera, a set of camera parameters and distortion coefficient could be picked. These were chosen to be similar to those observed in a real scenario.

With these values defined, any point in the point cloud can be projected to the image coordinates of each camera. Interpolating these points then produces an image similar to those obtained by the actual cameras.

C. Experiments

With this simulation in place, experiments could be done to quantitatively evaluate the reconstruction algorithm.

Pipelines were created with 3 different textures. These patterns were chosen in such a way that they had different amounts of feature points that could be detected on them. They can be found in Figure 5.

With these different pipelines, multiple runs were simulated. In these runs the camera positions would be randomly chosen within a reasonable range. They were allowed to deviate 2.5° on every axis of rotation and 1 cm on every axis of translation. These deviations are on the high side of what is generally expected in a real inspection run.

The tool was then simulated to move through the pipeline at a fixed speed. The speed and framerate were chosen such that it resembled that of a real inspection run. The intrinsic parameters, camera distortions, offset and orientation of the tool compared to the pipe were kept constant.

The now available data can be used to estimate the extrinsic parameters by feature matching and BA. After this estimation, the internal pipeline can be reconstructed at each single instance. These images can then be stitched together to form a fully reconstructed pipeline. The resulting image should in theory closely resemble the original texture that was placed as a point cloud. In addition to comparing the original texture with a reconstruction based on estimated extrinsic parameters, the original texture was also compared with reconstructions made with the initial extrinsic parameters and with the exact extrinsic parameters that were used for creating the simulation data (referred to as known extrinsic parameters).

D. Evaluation

The resulting reconstructions were evaluated using several metrics. Evaluating how well an image is reconstructed is a complex subject because pixel-wise comparison of the images in the form of MSE or RMSE usually is not too indicative of reconstruction quality. The human eye is usually more focused on structure as opposed to absolute color values, which is why the Structural Similarity Index Measure (SSIM)[27] was developed.

SSIM compares local patterns of pixel intensities normalized for luminance and contrast. The score consists of 3 components, one for comparing luminance, one for comparing contrast, and one for comparing structure after the prior two have been normalized. These three are then combined into one score. This leads to a measure more in line with human perception of image quality than previous work.[27]

An improvement of the SSIM came with the feature similarity (FSIM) index. This index takes lessons from what we learned about the human visual system and is designed to emphasize on the quality of those parts that human perception focuses on most. These important parts on which focus is placed are edges and high contrast areas. [28]

After evaluating the simulation data, another look was given to the gathered testdata and the available rundata. The data was reconstructed as well as possible and by visual evaluation, some conclusions were drawn on the success of the methods.

Error measure	Known	Initial	Calculated
RMSE	0.00368	0.00757	0.00615
FSIM	0.720	0.449	0.566
SSIM	0.986	0.940	0.965

TABLE II: Metrics for reconstructions done on simulated data. The reconstruction includes radial reconstruction and stitching together the radial reconstructions. For stitching together, the same approach was used for each set of extrinsic parameters. More results in Appendix A.

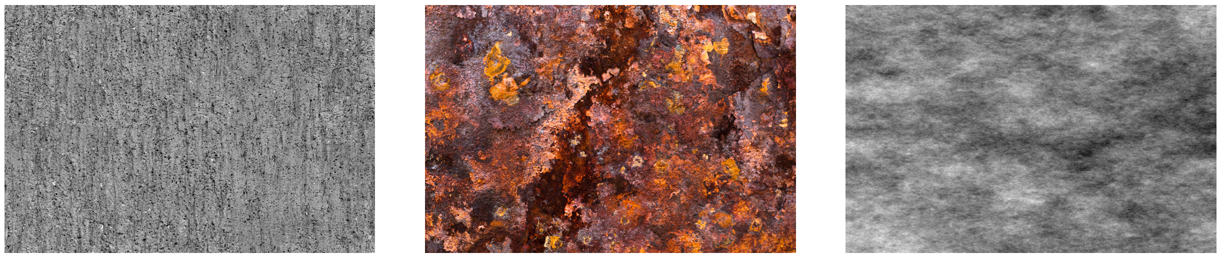
VII. Results

A. Simulation results

Table II shows the scores achieved for a full reconstruction of a section of the simulated pipeline. This table includes the results of three repeated runs with extrinsic parameters randomly chosen within a reasonable range (as described in Section VI-C) for each of the three patterns. Three different reconstruction types are compared, one with the known extrinsic parameters as were used for creating the data, one with the initial extrinsic parameters as assumed by the construction of the tool (which deviate from the actual values in the simulation), and one with calculated extrinsic parameters as reconstructed by feature point matching and BA. As expected, the known extrinsic parameters perform best with the highest scores for the used metrics by quite a margin. After that, it can be observed that the initial extrinsic parameters still do a somewhat reasonable job, but are beaten by the extrinsic parameters estimated by the BA. All metrics show a clear improvement of the scores from ideal to estimated parameters. However, even the estimated parameters are still quite far away from the scores obtained by the known parameters.

The quality ranking created by the scores can be confirmed by visual inspection as seen in Figure 6. The reconstruction with the known parameters is closest to the original, as should be the case, but after that, the reconstruction based on the estimated parameters is visually superior to the initial extrinsic parameters. The reconstruction based on the estimated extrinsic parameters can visually barely be distinguished from the one created with the known parameters, but the values of the extrinsic parameters still differ significantly. This difference is smaller than the difference between the known extrinsic parameters and the initial extrinsic parameters, but it is still quite significant. A deviation in the reconstruction compared to the original image that may cause the difference in the metrics can be seen in Figure 7. The overlapping parts of the different cameras seem to match nicely, but some sinusoidal distortion can still be seen over the full reconstruction. This sinusoidal distortion is not clearly visible in all reconstructions made with estimated extrinsic parameters. Based on observations on repeated simulations with different parameters, the extent of this sinusoidal behavior seems related to the number of features that were available at the BA phase. The presence of fewer features gives a higher probability of this distortion. Most likely the amount of rotation present in the extrinsic parameters also plays a role in the presence of this behavior.

Comparison of performance with different quality textures: When the quality of reconstruction on the different textures is compared, it shows a difference between them. This can be

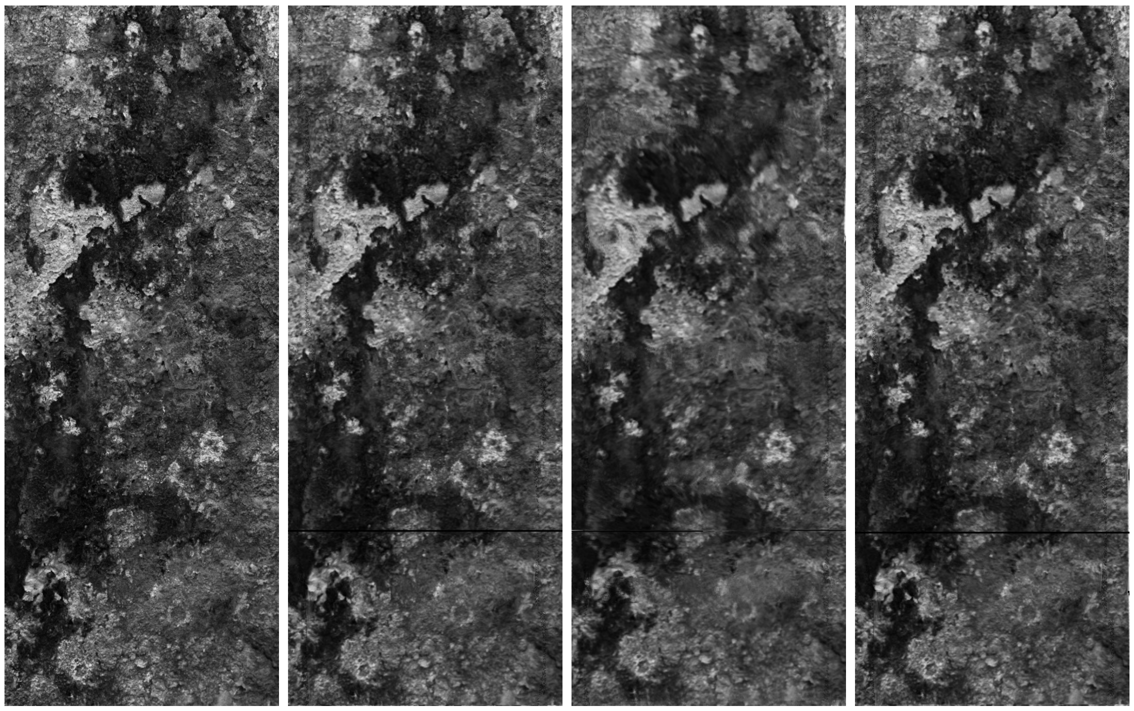


(a) A texture of concrete with many features detected.

(b) A texture of rust with a limited amount of features detected

(c) A texture generated by a random procedure with very few features detected.

Fig. 5: The textures used for the simulated data.



(a) The original pattern.

(b) Known extrinsic parameters.

(c) Initial extrinsic parameters.

(d) Calculated extrinsic parameters.

Fig. 6: A comparison of reconstructions made with different extrinsic parameters. **6a** is the original texture that was placed on the simulated pipeline. **6b** is a reconstruction with the same parameters as the simulation. **6c** is a reconstruction with the initial extrinsic parameters that are known from the construction of the tool, but are not exactly correct. **6d** is a reconstruction with extrinsic parameters calculated based on feature point matching and BA. **Note** that in the reconstructions, a bar is visible at around a quarter of the images from the bottom. This line occurs because the image is reconstructed in a particular position and at the edge certain artifacts are introduced by interpolation. Since this image is shifted to match the original, these edge artifacts end up in the middle of the image. In productive versions, these artifacts could be removed by adjusting the interpolation.

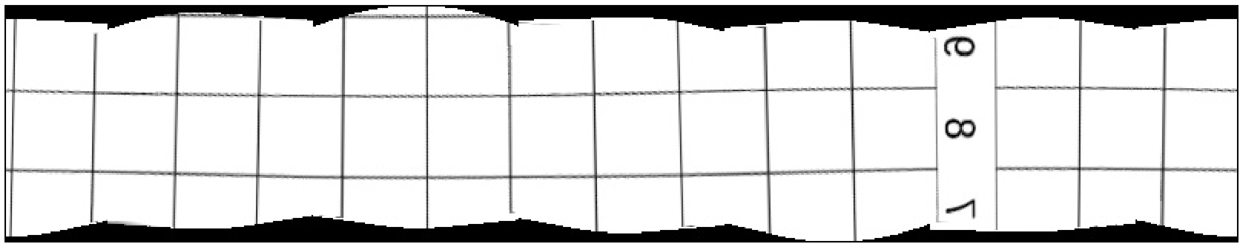


Fig. 7: A reconstruction with extrinsic parameters based on feature point extraction and BA with a different pattern. The reconstruction shows a sinusoidal distortion.

seen in Table III. The most relevant difference is that for the random pattern, the calculated extrinsic parameters only perform slightly better than the initial extrinsic parameters, where the difference is more substantial with the other textures. The probable explanation for this is in the number of feature points per overlapping area that the textures produce. On average 10 feature points of the random texture are just not enough for a high-quality reconstruction, whereas the other textures with around 50 and 100 feature points are more reliable.

Comparison of different stitching methods: The use of the different methods to stitch the different radial reconstructions together turned out to only have a quite small influence. Reconstructions with the four different methodologies on simulation data were visually indistinguishable and only had a small difference in the metrics. There are some cases in which one of the methods did not produce an acceptable result at all. It is hard to determine the exact cause of this, but the pattern on the pipeline and the available feature points do most likely play an important role.

These results are mainly based on the simulated data, but might not be fully representative of the real data. Stitching together the radial reconstructions requires some amount of features to find the right shift between the two camera positions, which is always present in the simulation data. However, on the data from actual runs, this is no guarantee. When this information is not available, the reconstruction can at best be made with information from the last locations where there were good features, but this will result in limited accuracy in all cases and is very hard to compare. It also greatly varies with the data which method will perform best in tricky scenarios.

B. Testdata results

The reconstruction of a single instance from the testdata looks like a correct reconstruction as can be seen in Figure 8. Compared to a reconstruction of the same data with the initial extrinsic parameters there is less ghosting and the alignment is better. This can be seen in Figure 9. However, there is some color disparity visible in both reconstructions. In the reconstructions of the simulation, this was not an issue as all images were simulated with perfectly similar conditions. However, in the workshop lighting may have been different when facing a different direction. This lighting difference increases the importance of good blending, but the linear interpolation used in this methodology is not suited to deal with this.

The images in the testdata have quite a high number of features that can be detected (around 200 per overlapping area) allowing for a more accurate reconstruction of the extrinsic parameters. This is even significantly more than available in the simulation data.

After reconstructing a single instance, combining the separate instances into a full reconstruction produces a result that can be seen in Figure 10. As can be seen, the general structure is well preserved, but the color difference clearly shows that the reconstruction is not perfect yet. Also, in the lower right corner, some ghosting effects can now be seen because two radial reconstructions were not aligned perfectly. This is the case as the separate radial reconstructions do not show any significant ghosting.

Extrinsic calibration with checkerboards: As described in the methods, extrinsic calibration using checkerboards in the overlapping areas of the cameras was done. The performance of this calibration was unfortunately lacking, due to drifting in scale. The cylindrical correction that could be applied with the calibration inside the pipeline is not applicable for calibration outside the pipeline. To make the calibration effective some form of correction would be required, but it fell outside the scope of this project.

C. Rundata results

The reconstructions of the actual runs are good on the parts where enough structure is available, as can be seen in Figure 11. The textures align well, but the connection that is visible does show a weird curve, where it can be expected to be relatively straight. This resembles the sinusoidal pattern observed in Figure 7. Nevertheless, it is an improvement to the reconstruction with simpler initial parameters in Figure 12. It can also be observed that other reconstructions on the rundata vary in quality. This may be correlated to the number of features that can be found. However, on most textures, it is hard to verify the quality of the reconstruction.

When looking at a full reconstruction of the rundata it can be observed that in the more structured areas, it looks plausible to be correct, but in the more featureless areas, it becomes very blurry. This is to be expected as the stitching approach is based on the features in the images. Figure 13 shows a small section of a run that was fully reconstructed. There still is some misalignment in the stitching, but the general structure is clear. Figure 14 shows a larger section of the pipeline that was reconstructed. The bottom part of the reconstruction has a plausible-looking reconstruction, as the available features overlap nicely. The top part however has fewer features and thus the result is very blurry. Because there are no features, it is also hard to judge whether it is reconstructed correctly, but it seems that estimating the movement does not work well.

D. SfM results

Figure 15 shows the results of a simulated pipeline reconstructed by COLMAP. The cylindrical structure can be seen and there are only very slight outliers. Since the pointcloud is only sparse, it is hard to say whether the pattern is properly reconstructed, however, it looks promising.

The result of COLMAP on the testdata can be seen in Figure 16. It shows a clear cylindrical shape again, but it also shows a part where it is slightly disconnected. The parts with the calibration pattern are rich in points, but parts of the pipeline were also featureless.

After this, an attempt was made to reconstruct an actual pipeline with COLMAP, but this was significantly less successful as can be seen in Figure 17. At first, this reconstruction does not look too great, but there are two parts with a somewhat cylindrical shape to be spotted, each of those are the images of two cameras being put together. These two shapes are then incorrectly placed together. The images from

Metric	Concrete pattern			Rust pattern			Random pattern		
	Known	Initial	Calculated	Known	Initial	Calculated	Known	Initial	Calculated
RMSE	0.00447	0.00966	0.00753	0.00415	0.00856	0.00629	0.00295	0.00447	0.00465
FSIM	0.74204	0.47346	0.60249	0.72777	0.44406	0.62075	0.68942	0.42972	0.47430
SSIM	0.98259	0.91426	0.94749	0.98289	0.92492	0.96523	0.99241	0.98209	0.98262

TABLE III: Comparison of the quality of reconstruction with different textures. More results in Appendix A.

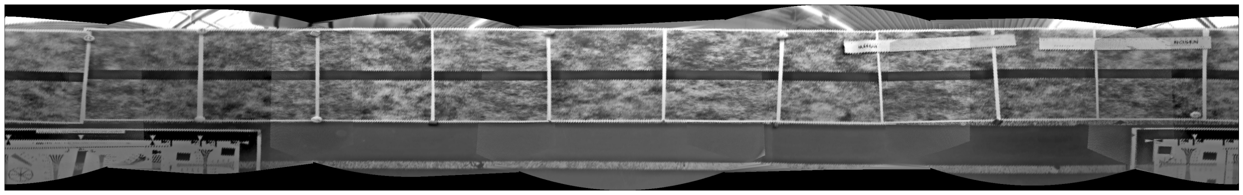


Fig. 8: A radial reconstruction at a single instance of the testdata with calculated extrinsic parameters. The alignment of the separate images in the overlapping parts is good, but there is a brightness difference present between the individual images.

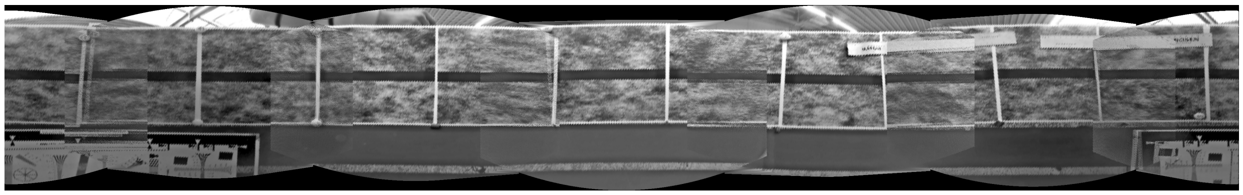


Fig. 9: Radial reconstruction of the testdata with initial extrinsic parameters. This shows the same part of the pipeline compared to Figure 8, but uses different parameters for reconstruction. Some ghosting and misalignment are visible in the area where the different cameras overlap.

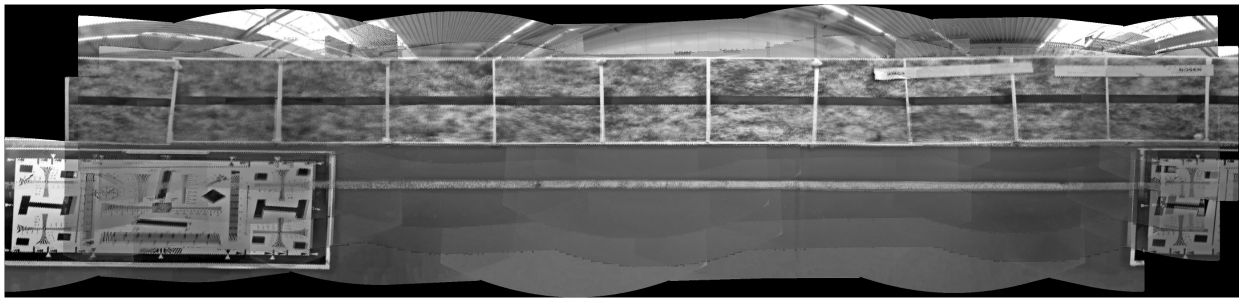


Fig. 10: A reconstruction of the full test dataset with calculated extrinsic parameters. It includes five radial reconstructions that were stitched together. The structure overall is correct, as most pattern continue logically from one side of the image to the other. However, some brightness differences can be seen. Also, some ghosting can be seen, especially in the bottom right corner.

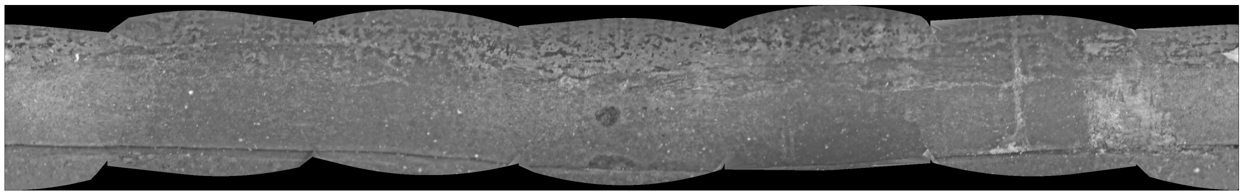


Fig. 11: A reconstruction at a single instance of real data with estimated parameters. A connection between pipeline segments can be seen in the bottom part of the image.

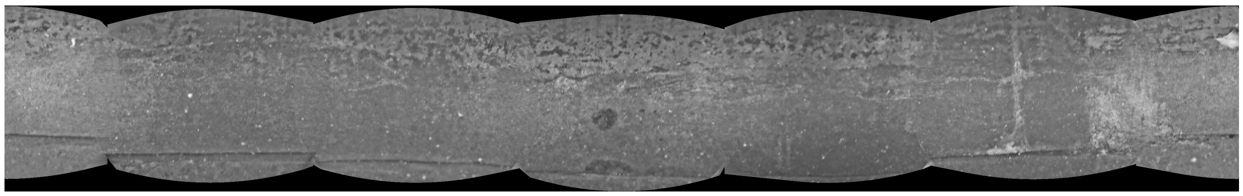


Fig. 12: A reconstruction at a single instance of real data with initial parameters. A connection between pipeline segments can be seen in the bottom part of the image.

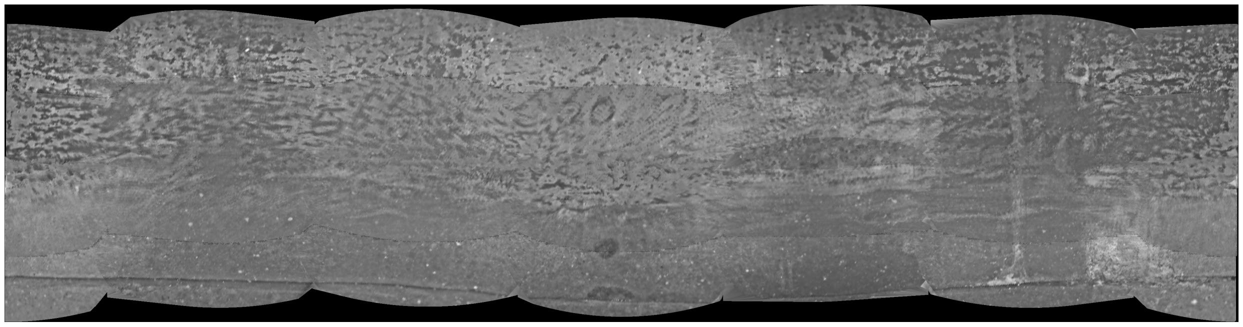


Fig. 13: Reconstruction of a small part of a pipeline inspection of real data. It includes several radial reconstructions that were stitched together. The overall structure seems good, but the patterns are not perfectly aligned by the stitching.

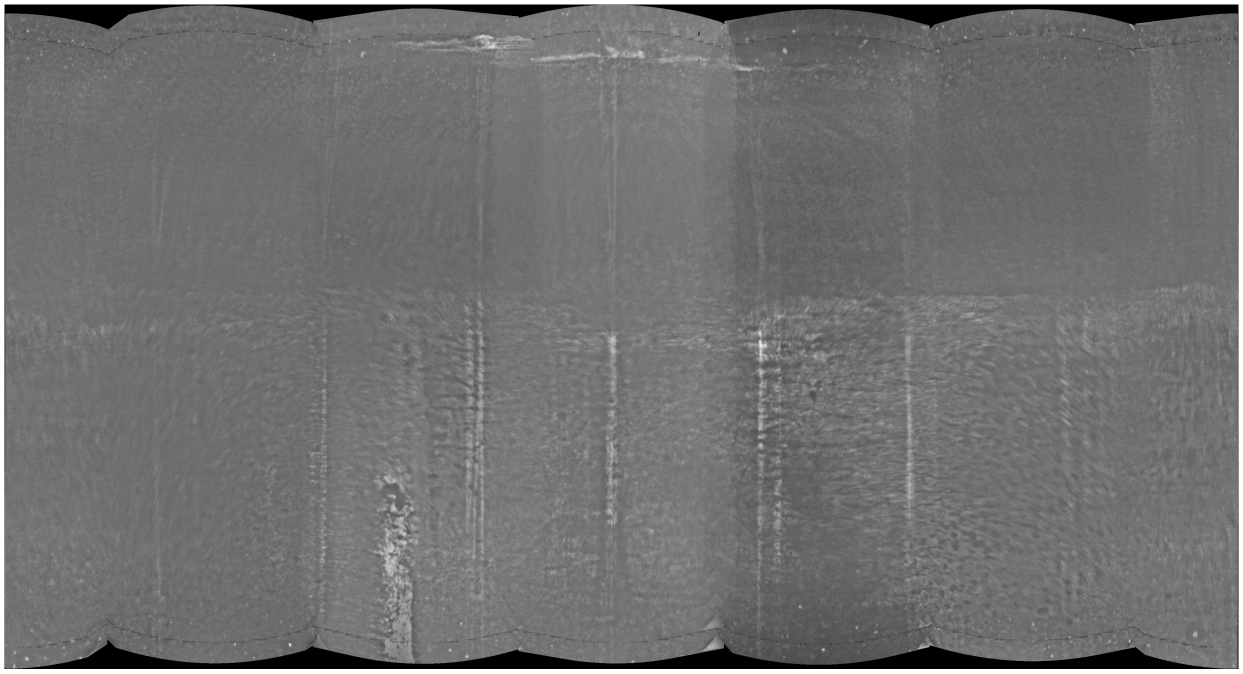


Fig. 14: Reconstruction of a small part of a pipeline inspection run of real data. It includes around 20 radial reconstructions. It can be observed that the bottom parts seem to align well, whereas the top part is significantly more blurry.

the other two cameras could not be connected to the others by the algorithm. This indicates that between some cameras there was enough texture in the overlapping areas and between others, there was not. However, if there was a little more texture it seems plausible that this reconstruction would work. In addition, it can also be seen that the pairs of two cameras curve significantly. At first it was thought that this was due to the plastic parts of the tool being visible on the video. These parts do not move and any feature points detected here may give the false impression of stationarity. However, after masking these out, the behavior remained. It could be that the limited number of matching points causes this behavior, but then it is quite curious that the behavior is too similar for both pairs of cameras. Ultimately, it may be because some parts of the plastic are not perfectly masked out.

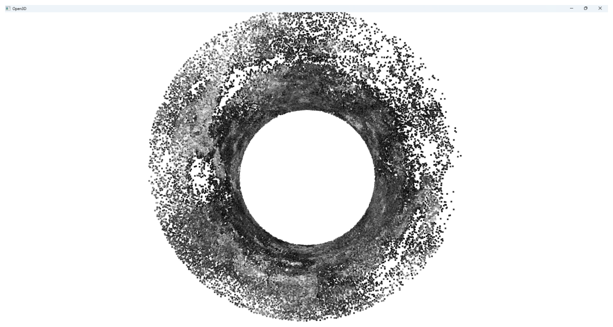


Fig. 15: A pointcloud produced by COLMAP on the simulation data.

VIII. Discussion and Future work

A. Discussion

The results show that when all parameters can be estimated correctly this model-based approach allows for high-quality reconstructions at a single instance. Yet, estimating all parameters with high enough accuracy was quite challenging in

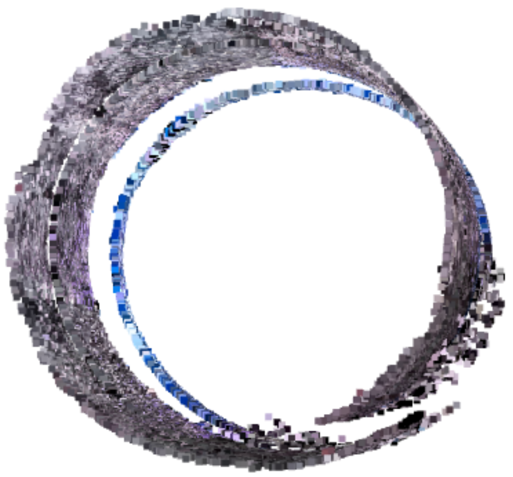


Fig. 16: A pointcloud produced by COLMAP on the testdata.

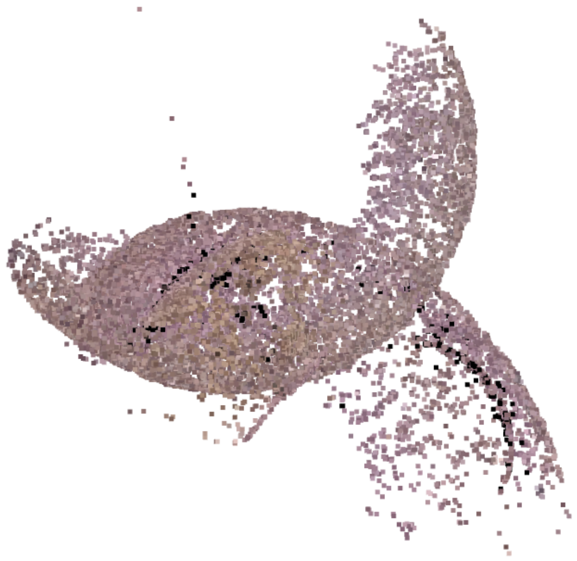


Fig. 17: A pointcloud produced by COLMAP on some data from a run.

this setup. Intrinsic camera parameters and distortions could be estimated with high enough accuracy, but the extrinsic camera parameters remain tricky. Extrinsic calibration inside the pipeline works best with a relatively high number of feature points. With fewer features, more error is present, in the form of misalignment, but also sinusoidal distortion. Calibration outside the pipeline should be possible with improvements to the algorithms. This may include steps for keeping the scale, but also a different calibration setup, with one 3D structure visible by all cameras. The advantage of the model-based approach is that the lack of observed features in parts of the pipelines is not an issue. After the calibration, the extrinsic parameters are known and can be applied anywhere in the pipeline, assuming the camera setup does not change.

The effect of the position and orientation of the tool to the pipeline was assumed to be minimal and with the available data that assumption was reasonable. In the simulation and test data, no misalignments were present by design, but also in the run data, no indications were found that any present misalignment was a big problem. In scenarios where the tool is less stable inside the pipeline and bigger misalignments occur, it may provide issues with radial reconstruction.

The intrinsic calibration was observed to be of sufficient quality in this scenario. The reprojected checkerboards look good, although not perfect. Also, in the reconstructions, no significant errors can be attributed to the failure of intrinsic calibration. In the future, it may be possible to obtain improvements by better intrinsic calibration, but it should not be the highest priority.

Extending the model-based approach to multiple instances does rely more on the textures of the images than the previous step. Approaches work well when there is some minimal level of texture, but they struggle when features are a bit more scarce. This highlights one of the limitations of the current approach, yet improvements in feature detection and matching may benefit this model, as described in the future work section. It may be considered to take the tool speed as determined by other sensors and use that to determine how the images should be stitched, but that does result in a dependence on other sensors, which was aimed to be avoided for now. It might however be an option in scenarios where the textures lack.

The SfM that was attempted showed some promising results on the testdata, but had significantly more trouble with the rundata. This can most likely be attributed to the fact that the conditions on the rundata are more challenging than those of the testdata, with fewer and more similar features.

B. Future work

Multiple approaches for future work are interesting to explore based on this research.

1) Deep learning features: In many steps, both in the model-based approach and in the SfM approach, feature points are very relevant for good reconstruction However, it was discussed that they were not plentiful and reliable in some challenging pipeline environments. It was observed that at some locations there were very few or even no features. Additionally, a significant part of these points would be outliers, when only a few are present.

SIFT features are some of the best classical feature extractors, but recently more advanced feature extractors and descriptors based on deep learning have been developed, such as SuperPoint[22] and SuperGlue[23]. These were shown to outperform classical methods with more robust matches and more features. Combining these features with the architectures discussed in this research may be beneficial for the performance;

2) Calibration setup: The calibration setup outside the pipeline was shown to be imperfect. Extrinsic calibration based on rundata is possible, but requires good texture at some point in the pipeline during the run. It would be beneficial if the reconstruction would not depend on the availability of good texture in the pipeline for calibration. Moreover, if the calibration is well designed, it can produce many more matching features than the pipeline itself, leading to more accurate calibration.

Possibilities for this better calibration setup could take several shapes. For instance, it could be similar to the setup in the testdata with feature-rich patterns in a pipeline of known diameter. It could also include a more advanced setup outside the pipeline with a 3D structure with markers and known dimensions. These solutions would counter the lack of scale in the current calibration setup outside the pipeline.

3) Topic specific SfM: The setup used for SfM in this paper was a general approach, that does not utilize all the information from the constrained environment in which the tool operates. Approximate information about the relative camera positions is already known, in addition, it is clear in which parts of images overlap can be expected, and the environment will be cylindrical. Utilizing this information in a SfM approach can improve the quality, whereas the model in the current approach struggles to find overlapping parts. [9]

Previous research also showed that SfM approaches can benefit from input from IMU sensors [24], which are also present in this tool. In addition, information about the position and orientation of the tool relative to the pipe from dipper sensors may also provide useful information to the model in this challenging environment.

4) Blending: The blending used in the current work was a basic approach. To overcome color differences due to lighting and have more visually appealing stitching, improved blending may be worthwhile to investigate.

C. Practical usage

Some of the fruits of this research include an early prototype of algorithms that can provide canvas view reconstructions of internal pipeline inspection videos. The approach is by no means mature or practically reliable, but it provides a prototype and a solid basis for further development at Rosenxt. One of the issues with the current algorithms is that it is also quite slow. There is room for speeding up the implementations, but in the meantime, it will not be practical to reconstruct entire pipelines. It is suitable to reconstruct areas of interest specifically.

The radial reconstructions are possible at any location in the pipeline, given that a few conditions are met: the extrinsic parameters were calibrated, the misalignment of the tool is minimal, the diameter is known, and the pipeline is straight. This is mainly useful for inspecting connections or other objects in the pipeline that stretch over the frames of the different cameras. Stitching together the radial reconstructions at different times is still limited to the areas with a reasonable amount of texture. This is a disadvantage, but it can be thought that very featureless areas are also not of the highest interest for inspection. Which stitching algorithm to use depends on the situation and no clear best one can be chosen with the current information.

IX. Conclusions

The proposed model-based approach for stitching images in a pipeline environment was shown to be feasible, but leaves room for improvement.

To **RQ1** we can conclude that the extrinsic parameters could best be estimated by feature detection and matching in a pipeline environment. Optimized by BA with constraints on the cylindrical shape. If the inspected pipeline does not have enough features, this may be done in a calibration pipeline with additional textures. The other attempted approach showed lacking performance, but there is opportunity to improve this method.

In **RQ2.1** we can conclude that the model-based reconstruction as proposed was very effective. It does have some constraints on correct parameter estimation, which remains a challenge. The model-based aspect does bring the advantage that this process is independent of the textures after calibration.

The effectiveness of stitching together the separate instances as questioned in **RQ2.2** is somewhat limited. It still depends quite heavily on good features, which are very limited in some areas of the pipelines. Within textured areas of the pipeline, however, it can prove very useful.

With this **RQ2** can be answered. The full reconstruction does show to be successful in some scenarios, but it is not a very robust approach in its current state. It is a good starting point for further work and it requires improvement to make it into a practical application.

Regarding the SfM approach from **RQ3**, it can be concluded that the current setup is only effective in the highly textured areas. With COLMAP the current feasibility of actual pipeline run reconstruction is limited. With the proposed extensions, however, this approach may become successful in the reconstruction of the pipeline internals.

References

[1] E. Kuliczkowska, "AN ANALYSIS OF ROAD PAVEMENT COLLAPSES AND TRAFFIC SAFETY
}HAZARDS RESULTING FROM LEAKY SEWERS," *Baltic Journal of Road and Bridge Engineering*, vol. 11, no. 4, pp. 251–258, 2016.

[2] S. Folkman, "Water Main Break Rates In the USA and Canada: A Comprehensive Study Overall Pipe Breaks Up 27% In Six Years," tech. rep., Utah state university, Logan, 3 2018.

[3] J. Latif, M. Z. Shakir, N. Edwards, M. Jaszczykowski, N. Ramzan, and V. Edwards, "Review on condition monitoring techniques for water pipelines," 4 2022.

[4] Y. Wang, P. Li, and J. Li, "The monitoring approaches and non-destructive testing technologies for sewer pipelines," *Water Science and Technology*, vol. 85, pp. 3107–3121, 5 2022.

[5] Q. Ma, G. Tian, Y. Zeng, R. Li, H. Song, Z. Wang, B. Gao, and K. Zeng, "Pipeline in-line inspection method, instrumentation and data management," 6 2021.

[6] Zhang, Zhao, Hu, Wang, Ai, and Li, "A 3D Reconstruction Pipeline of Urban Drainage Pipes Based on MultiviewImage Matching Using Low-Cost Panoramic Video Cameras," *Water*, vol. 11, p. 2101, 10 2019.

[7] Z. Shang and Z. Shen, "Single-pass inline pipeline 3D reconstruction using depth camera array," *Automation in Construction*, vol. 138, p. 104231, 6 2022.

[8] A. Anwer, F. Meriaudeau, and S. H. Adil, "Customized graphical user interface implementation of Kinect Fusion for underwater application," in *2017 IEEE 7th International Conference on Underwater System Technology: Theory and Applications (USYS)*, vol. 2018-January, pp. 1–6, IEEE, 12 2017.

[9] S. Kagami, H. Taira, N. Miyashita, A. Torii, and M. Okutomi, "3D Pipe Network Reconstruction Based on Structure from Motion with Incremental Conic Shape Detection and Cylindrical Constraint," in *2020 IEEE 29th International Symposium on Industrial Electronics (ISIE)*, pp. 1345–1352, IEEE, 6 2020.

[10] P. Hansen, H. Alismail, P. Rander, and B. Browning, "Visual mapping for natural gas pipe inspection," *The International Journal of Robotics Research*, vol. 34, pp. 532–558, 4 2015.

[11] R. Zhang, M. H. Evans, R. Worley, S. R. Anderson, and L. Mihaylova, *Towards Autonomous Robotic Systems*, vol. 13054 of *Lecture Notes in Computer Science*. Cham: Springer International Publishing, 2021.

[12] Y.-J. Zhang, *3-D Computer Vision*. Singapore: Springer Nature Singapore, 2023.

[13] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[14] J. Kannala and S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 1335–1340, 8 2006.

[15] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle Adjustment — A Modern Synthesis," pp. 298–372, 2000.

[16] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," pp. 404–417, 2006.

[17] D. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pp. 1150–1157, IEEE, 1999.

[18] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary Robust Independent Elementary Features," pp. 778–792, 2010.

[19] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pp. 1508–1515, IEEE, 2005.

[20] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2564–2571, 2011.

[21] E. Adel, M. Elmogy, and H. Elbakry, "Image Stitching based on Feature Extraction Techniques: A Survey," *International Journal of Computer Applications*, vol. 99, pp. 1–8, 8 2014.

[22] D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperPoint: Self-Supervised Interest Point Detection and Description," 12 2017.

[23] P.-E. Sarlin, D. Detone, T. Malisiewicz, A. Rabinovich, and E. Zurich, "SuperGlue: Learning Feature Matching with Graph Neural Networks," tech. rep., 2019.

[24] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM," *IEEE Transactions on Robotics*, vol. 37, pp. 1874–1890, 12 2021.

[25] J. L. Schönberger and J.-M. Frahm, "Structure-from-Motion Revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[26] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, A. Vijaykumar, A. P. Bardelli, A. Rothberg, A. Hilboll, A. Kloeckner, A. Scopatz, A. Lee, A. Rokem, C. N. Woods, C. Fulton, C. Masson, C. Häggström, C. Fitzgerald, D. A. Nicholson, D. R. Hagen, D. V. Pasechnik, E. Olivetti, E. Martin, E. Wieser, F. Silva, F. Lenders, F. Wilhelm, G. Young, G. A. Price, G.-L. Ingold, G. E. Allen, G. R. Lee, H. Audren, I. Probst, J. P. Dietrich, J. Silterra, J. T. Webber, J. Slavič, J. Nothman, J. Buchner, J. Kulick, J. L. Schönberger, J. V. de Miranda Cardoso, J. Reimer, J. Harrington, J. L. C. Rodríguez, J. Nunez-Iglesias, J. Kuczynski, K. Tritz, M. Thoma, M. Newville, M. Kümmerer, M. Bolingbroke, M. Tartre, M. Pak, N. J. Smith, N. Nowaczyk, N. Shebanov, O. Pavlyk, P. A. Brodtkorb, P. Lee, R. T. McGibbon, R. Feldbauer, S. Lewis, S. Tygier, S. Sievert, S. Vigna, S. Peterson, S. More, T. Pudlik, T. Oshima, T. J. Pingel, T. P. Robitaille, T. Spura, T. R. Jones, T. Cera, T. Leslie, T. Zito, T. Krauss, U. Upadhyay, Y. O. Halchenko, and Y. Vázquez-Baeza, "SciPy 1.0: fundamental algorithms for scientific computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 3 2020.

[27] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, pp. 600–612, 4 2004.

[28] L. Zhang, L. Zhang, X. Mou, and D. Zhang, "FSIM: A feature similarity index for image quality assessment," *IEEE Transactions on Image Processing*, vol. 20, pp. 2378–2386, 8 2011.

[29] C. Lanaras, J. Bioucas-Dias, S. Galliani, E. Baltsavias, and K. Schindler, "Super-resolution of Sentinel-2 images: Learning a globally applicable deep neural network," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 146, pp. 305–319, 12 2018.

[30] Zhou Wang and A. Bovik, "A universal image quality index," *IEEE Signal Processing Letters*, vol. 9, pp. 81–84, 3 2002.

Appendix A
Additional metrics

Some additional metrics were obtained from the reconstructed simulated data. These were left out of the main report for brevity. In addition, these metrics show a correlation with the scores of the main metrics. The additional metrics that can be found here are the Peak Signal to Noise Ratio(PSNR), Signal to Reconstruction Error ratio(SRE)[29], and Universal Image Quality(UIQ)[30].

Error measure	Known extrinsic parameters	Initial extrinsic parameters	Calculated extrinsic parameters
RMSE	0.00368	0.00757	0.00615
FSIM	0.720	0.449	0.566
SSIM	0.986	0.940	0.965
PSNR	48.41	42.88	44.39
SRE	55.33	52.70	53.33
UIQ	0.780	0.123	0.423

TABLE IV: Additional metrics for reconstructions done on simulated data.

Metric	Concrete pattern			Rust pattern			Random pattern		
	Known	Initial	Calculated	Known	Initial	Calculated	Known	Initial	Calculated
RMSE	0.00447	0.00966	0.00753	0.00415	0.00856	0.00629	0.00295	0.00447	0.00465
FSIM	0.74204	0.47346	0.60249	0.72777	0.44406	0.62075	0.68942	0.42972	0.47430
SSIM	0.98259	0.91426	0.94749	0.98289	0.92492	0.96523	0.99241	0.98209	0.98262
PSNR	47.002	40.30	42.48	47.64	41.35	44.04	50.60	47.00	46.66
SRE	56.75	53.54	54.50	51.17	48.14	49.36	58.05	56.42	56.12
UIQ	0.82634	0.08742	0.46592	0.74214	0.14973	0.52242	0.77300	0.13220	0.28039

TABLE V: Comparison of the quality of reconstruction with different textures with additional metrics.