Towards a Visual Perception System for Pipe Inspection: Monocular Visual Odometry

Peter Hansen, Hatem Alismail, Peter Rander and Brett Browning

CMU-CS-QTR-104
CMU-RI-TR-10-22

July 22, 2010

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

© Carnegie Mellon University

This report was made possible by the support of an NPRP grant from the Qatar National Research Fund.
The statements made herein are solely the responsibility of the authors.

Keywords: oil and gas, pipe inspection, pipe crawler, visual odometry

Abstract

Liquid Natural Gas (LNG) processing facilities contain large complex networks of pipes of varying diameter and orientation intermixed with control valves, processes and sensors. Regular inspection of these pipes for corrosion, caused by impurities in the gas processing chain, is critical for safety. Popular existing non-destructive technologies that used for corrosion inspection in LNG pipes include Magnetic Flux Leakage (MFL), radiography (X-rays), and ultrasound among others. These methods can be used to obtain measurements of pipe wall thickness, and by monitoring for changes in pipe wall thickness over time the rate of corrosion can be estimated. For LNG pipes, unlike large mainstream gas pipelines, the complex infrastructure means that these sensors are currently employed external to the pipe itself making comprehensive, regular coverage of the pipe network difficult to impossible. As a result, a sampling-based approach is taken where parts of the pipe network are sampled regularly, and the corrosion estimate is extrapolated to the remainder of the pipe using predictive corrosion models derived from metallurgical properties. We argue that a robot crawler that can move a suite of sensors inside the pipe network, can provide a mechanism to achieve more comprehensive and effective coverage. In this technical report, we explore a vision-based system for building 2D registered appearance maps of the pipe surface whilst simultaneously localizing the robot in the pipe. Such a system is essential to provide a localization estimate for overlaying other non-destructive sensors, registering changes over time, and the resulting 2D metric appearance maps may also be useful for corrosion detection. For this work, we restrict ourselves to linear pipe formations.

We explore two distinct classes of algorithms that can be used to estimate this pose are investigated, both visual odometry systems which estimate motion by observing how the appearance of images change between frames. The first is a class of *dense* algorithms that use the greyscale intensity values and their derivatives of all pixels in adjacent images. The second class is a *sparse* algorithm that use the change in position (sparse optical flow) of salient point feature correspondences between adjacent images. Pose estimate results obtained using the dense and sparse algorithms are presented for a number of images sequences captured by different cameras as they moved through two pipes having diameters of 152.40mm (6") and 406.40mm (16"), and lengths 6 and 4 meters respectively. These results show that accurate pose estimates can be obtained which consistently have errors of less than 1 percent for distance traveled down the pipe. Examples of the stitched images are also presented, which highlight the accuracy of these pose estimates.

Contents

1 Introduction

Corrosion of the pipes used in the sour gas processing stages of Liquid Natural Gas (LNG) facilities can lead to failures that result in significant damage to the infrastructure, loss of product, and most importantly fatalities and serious injuries to human operators. Detailed inspection of these pipes to monitor corrosion rates is therefore a high priority. In current industry practice, such inspection is performed using Non-Destructive sensors located external to the pipe, and/or by inserting sacrificial metal samples that corrode over time, but can be easily measured. Example sensors include Magnetic Flux Leakage (MFL), ultrasound, and radiography (e.g. [29]). Typically, these sensors can only be deployed in reachable locations. As a result, predictive models of corrosion rates derived from sampled portions and metallurgical models are used to estimate the status of the pipe wall.

An alternative approach is to use a pipe crawling robot to measure all parts of the pipe surface thereby avoiding the need for extensive, and potentially unreliable, predictive models. This approach has proven highly successful for inspecting gas pipelines, as with a Smart inspection PIG[1], and in downstream distribution networks [35]. Typically, such systems either rely on high quality, expensive Inertial Motion Units (IMUs) for pose estimation or wheel odometry, which is known to be inaccurate. In this paper, we explore visual odometry approaches to achieving high accuracy position estimates with a low cost camera system. Our goal is to develop a low-cost system that can support building high resolution models of the pipe wall which can later be used for corrosion detection and/or augmenting existing sensors. Concretely, our system operates with a monocular camera mounted on a vehicle moving through the pipe network. The output of the system is an estimate of the vehicle pose at each time step. A second output of the system is a 2D metric appearance map of the inner pipe surface. Figure 1 shows an example of a stitched image generated from several hundred individual images taken by a camera as it moved through a 152.4mm (6") diameter carbon steel pipe.

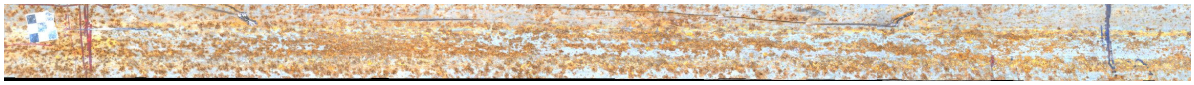


Figure 1: A stitched image of a region inside a 152.4mm (6") diameter carbon steel pipe. The stitched image was generated from several hundred individual images using a visual odometry developed in this work.

We explore two distinct classes of algorithms for estimating camera pose from monocular image sequences, both visual odometry systems which integrate incremental changes in camera pose to find the position of the camera, relative to some reference point, at the time each image was taken [32, 33, 24, 39, 20, 34, 18, 1]. The first of these is a *dense* class of algorithms which uses the greyscale values of all overlapping pixels in adjacent images to estimate the transform between the images, a process known as registration [42, 38, 2, 40, 5], which is then use to estimate the change in pose between the cameras. The dense algorithms estimate the transform between images using different techniques. The *full-search*[2] algorithm uses the difference of greyscale intensity values of overlapping regions to formulate a cost function. The *model-based* algorithm uses the difference of greyscale intensity gradients and a hierarchical (coarse to fine) approach. We also explore a hybrid approach that combines full-search with model-based based techniques for sub-pixel resolution. The second class a *sparse* algorithm that finds corresponding scene points in adjacent images. The measured change in image coordinates of these correspondences (sparse optical flow) is used to estimate a change in camera pose. The sparse algorithm enforces strictly that all scene points are assumed to lie on the surface of a cylindrical pipe, and it uses this assumption when estimating the change

[1] `http://www.geoilandgas.com/businesses/ge_oilandgas/en/prod_serv/serv/pipeline/en/inspection_services/index.htm`

[2] Also referred to commonly as a exhaustive search.

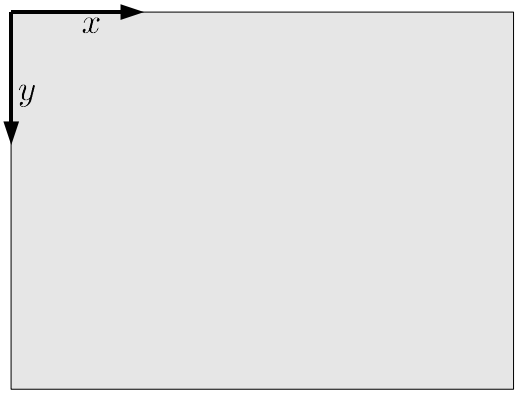


Figure 2: Image coordinates

in pose between camera views. To resolve the monocular visual odometry scale ambiguity [32], the dense algorithms use a reference scale measurement to convert pixel displacement to a metric Euclidean change in pose, while the sparse algorithm uses the precisely measured radius of the pipe.

We conducted experiments to assess the accuracy of our dense and sparse visual odometry algorithms. Our datasets include four image sequences captured by two different cameras as they traversed through different carbon steel pipes. The first pipe was 6 meters long with a diameter of 152.40 millimeters (6”), and the second pipe was 4 meters long with a diameter of 406.40 millimeters (16”). The results indicate that both the dense and sparse visual odometry algorithms were able to estimate the change in camera pose in the direction down the length of pipe with an error consistently less than 1 percent. Examples of the stitched images for each pipe are also presented which illustrates the accuracy of the visual odometry estimates, and form one part of the appearance maps that we wish to build as part of a visual perception system for LNG pipe inspection.

The remainder of this report is structured as follows. In section 2, we outline and describe the coordinate conventions as well notation used throughout this report. In section 3, we describe in detail the dataset collection process, including the methods used to obtaining a ground truth measurement, and the process we use to model and account for the non-uniform lighting distribution from our generated light source in the pipe. The dense methods are described in section 4, including both the full-search and model-based algorithms, which is followed by the description of the sparse algorithm in section 5. In section 6, we present the experimental results and a detailed discussion of the results and various factors which influence the accuracy of the visual odometry estimates obtained using the dense and sparse algorithms. In section 7, we present our conclusions and our outlook for future work.

2 Preliminaries and Notations

The notations and coordinate conventions used throughout this report are described in this section, which includes the parameterization of a cameras pose within a cylindrical pipe.

The coordinate system in image space is defined according to the standard convention, where the x axis is along the width of the image (columns), the y axis is along the height of the image (rows), and the origin of the coordinate system is located at the top left corner of the image, as illustrated in Fig. 2. Each pixel has a coordinate $\mathbf{x}_i = (x_i, y_i)^T$ whose homogeneous representation is $\boldsymbol{\chi}_i = (\mathbf{x}_i, 1)^T$.

As will be discussed in section 3, a small robotic platform equipped with a camera moves through the

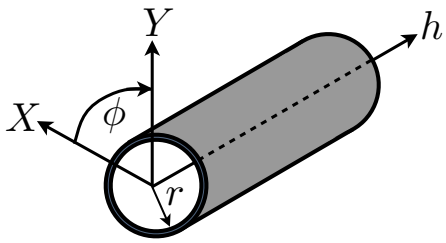


Figure 3: The coordinate system for a straight cylindrical pipe with constant radius r. The position of a point $\mathbf{X}_i$ on the interior surface of the pipe can be parameterized using Cartesian coordinates as $\mathbf{X}_i = (X_i, Y_i, h_i)^T$, where $X_i^2 + Y_i^2 = r^2$, or using cylindrical coordinates as $\mathbf{X}_i = (r\,\cos\phi_i, r\,\sin\phi_i, h_i)^T$.

interior of a straight cylindrical pipe. Referring to Fig. 3, we define $\mathbf{X} = (X, Y, h)^T \in \mathbb{R}^3$ as the position of a world point in the pipe's coordinate frame of reference, and denote its homogeneous representation as $\boldsymbol{\mathcal{X}} = (\mathbf{X}, 1)^T$. If r is the radius of the cylindrical pipe, then the coordinate of a scene point, constrained to lie on the interior surface of the pipe, can be parameterized as:

$$\mathbf{X} = \begin{bmatrix} r\,\cos\phi \\ r\,\sin\phi \\ h \end{bmatrix}, \tag{1}$$

where $\phi = \tan^{-1}(Y/X)$ is an angle about the central axis of the pipe.

The pose P_t of the camera relative to the pipe at time t is written as:

$$P_t = \begin{bmatrix} R & -R\,\mathbf{C} \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{2}$$

where R is a 3×3 rotation matrix, and $\mathbf{C}$ is a 3×1 Euclidean position vector of the camera with respect to the origin of the pipe. The pose P_t defines the transform of the homogeneous coordinate $\boldsymbol{\mathcal{X}}_i$ of a world point $\mathbf{X}_i$ in the pipe coordinate frame of reference to the homogeneous coordinate $\tilde{\boldsymbol{\mathcal{X}}}_i$ of the same scene point in the cameras coordinate frame of reference:

$$\tilde{\boldsymbol{\mathcal{X}}}_i = P_t\,\boldsymbol{\mathcal{X}}_i, \tag{3}$$

and has the inverse relationship:

$$\boldsymbol{\mathcal{X}}_i = P_t^{-1}\,\tilde{\boldsymbol{\mathcal{X}}}_i, \tag{4}$$

$$= \begin{bmatrix} R^{-1} & \mathbf{C} \\ \mathbf{0}^T & 1 \end{bmatrix} \tilde{\boldsymbol{\mathcal{X}}}_i. \tag{5}$$

Our visual odometry algorithms estimate incremental changes in camera pose to find the pose P_t of a camera as it moves through a pipe.

Many visual odometry algorithms estimate the *change* in pose between camera views. If $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{X}}'$ are the coordinates of world points respectively in camera frames 1 and 2, and $\tilde{\mathbf{X}} \leftrightarrow \tilde{\mathbf{X}}'$ is a known set of correspondences, the change in pose $\tilde{Q}_{1,2}$ between the two cameras is the projective transform between the homogeneous coordinates

$$\tilde{\boldsymbol{\mathcal{X}}}' = \tilde{Q}_{1,2}\,\tilde{\boldsymbol{\mathcal{X}}}, \tag{6}$$

where we use the notation $\tilde{}$ to signify that this change in pose is measure in the camera coordinate frame of reference. If $\tilde{P}_1$ is known, then the pose of the second camera is $\tilde{P}_2 = \tilde{Q}_{1,2}\,\tilde{P}_1$. As mentioned, this change in pose $\tilde{Q}_{1,2}$ is often measured. For our application is is necessary to be able to convert this change in pose

Table 1: Summary of datasets. The ground truth measurement is the distance between two marks on the interior surface of each pipe (see section 3.1)

	Pipe 1	Pipe 2
Location	Pittsburgh	Qatar
Pipe Material	Carbon Steel	Carbon Steel
Pipe Dimensions	6 meters long 152.40mm (6") outer diameter 153.32mm inner diameter	4 meters long 406.40mm (16") outer diameter 387.56mm inner diameter
Camera	Point Grey Research dragonfly 1024×768 pixels RGB, 7.5 frames per second	Point Grey Research firefly 640×480 pixels RGB (bayer), 30 frames per second
Lens	70° horizontal angle of view	25° horizontal angle of view
LEDs	4×3.5W (280 lumen) high intensity	4×3.5W (280 lumen) high intensity
ImageSequences	1: forward, 4256 images 2: forward, 4176 images 3: forward & reverse, 3369 images	1: forward & reverse, 2392 images
Ground Truth ($\delta\, h$)	5844.4mm	3391.0mm

into a change $Q_{1,2}$ in the pipe coordinate frame of reference, which is possible if P_1 is known (i.e. P_1 is the current estimate of the camera pose obtained using a visual odometry algorithm). Letting $\tilde{\boldsymbol{X}}' = P_2 \boldsymbol{X}$, then substituting this and Eq. 3 into Eq. 6 gives

$$P_2 \boldsymbol{X} = \tilde{Q}_{1,2}\, P_1 \boldsymbol{X}, \tag{7}$$

from which the pose P_2 is

$$P_2 = \tilde{Q}_{1,2}\, P_1. \tag{8}$$

Since $P_2 = P_1\, Q_{1,2}$, it follows that

$$P_1\, Q_{1,2} = \tilde{Q}_{1,2}\, P_1, \tag{9}$$

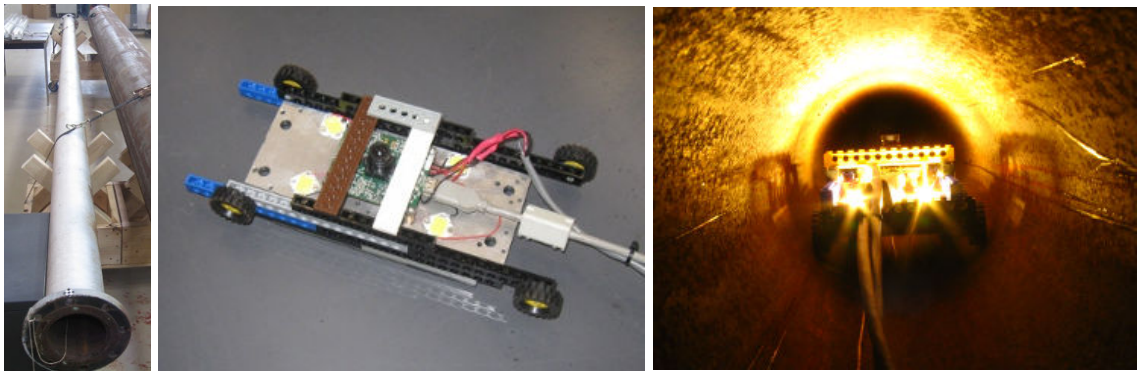
$$Q_{1,2} = P_1^{-1}\, \tilde{Q}_{1,2}\, P_1, \tag{10}$$

which enables the change in pose $Q_{1,2}$ in the pipe coordinate frame of reference to be found.

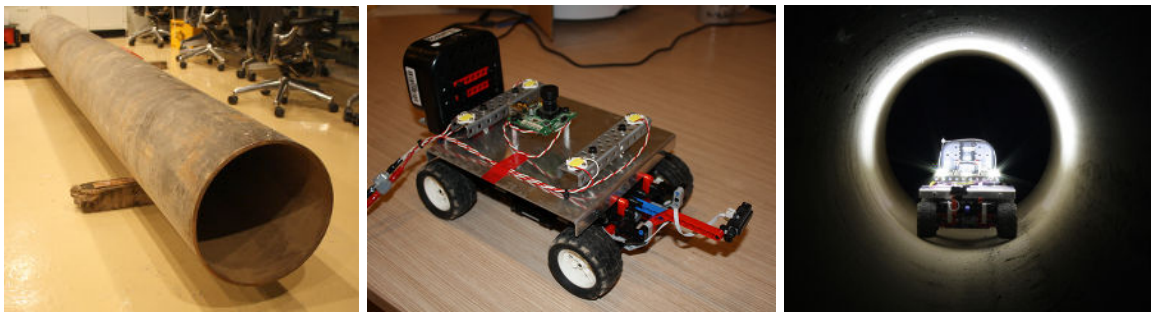
3 Dataset Collection

We have obtained datasets for two different pipes, which we refer to simply as pipe 1 (located in Pittsburgh) and pipe 2 (located in Qatar). For each pipe, we have a small robotic platform retrofitted with a camera and lens assembly, and four high intensity (3.5 Watt, 280 lumen) Light Emitting Diodes (LEDs), as shown in Figure 4. These LEDs provide all the lighting for the cameras. As the robots traverse through their respective pipes (this motion being primarily in the h direction — see Figure 3) the images from the cameras are logged, and are later processed off-line. The details for each pipe, the camera and lenses used, and the images sequences is summarized in table 1.

It should be noted here that the the 406.4mm (16") diameter of pipe 2 is much larger than the 152.4mm (6") diameter of pipe 1, and the camera used with the pipe 2 dataset has a much narrower angle of view than the camera used for pipe 1. As a result, the degree of foreshortening (i.e. the curvature of the pipe seen) in the images is far greater for the pipe 1 datasets than the pipe 2 datasets. An example image for each pipe dataset can be seen in the middle column in Figure 6.



(a) Pipe 1 (Pittsburgh): 6 meter, 152.4mm (6") diameter pipe (left), robot platform (middle), and the robot moving through the pipe (right).



(b) Pipe 2 (Qatar): 4 meter, 406.4mm (16") diameter pipe (left), robot platform (middle), and the robot moving through the pipe (right).

Figure 4: The two pipes and robotic platforms used to obtained datasets. Note that the 406.4mm (16") diameter of pipe 2 is much larger than the 152.4mm (6") diameter of pipe 1.

3.1 Ground Truth Measurement

In order to evaluate the performance of our visual odometry estimates, we manually augment the pipe with a very fine permanent maker. One of the markers for pipe 1 is shown in Figure 5. These reference markers are located at either end of the pipe on the uppermost point of the interior surface. The *ground truth* measurement for each pipe (see table 1) is the measured distance $\delta\, h$ between the two markers in each pipe.

3.2 Gain-mask Correction

The high intensity LEDs provide the only source of light in the pipe. Unfortunately, this lighting is not uniformly distributed within the field of view of the camera. As a result, there are distinctive lighting patterns visible in the images, especially in the images in the pipe 1 datasets. These patterns are evident in the middle column of Figure 6a in which it can be clearly seen that there is significant light intensity drop-off towards the periphery of the image[3]. For the dense visual odometry algorithm in particular, which is detailed in Section 4, these non-uniform lighting artifacts have a significant impact on the accuracy of the visual odometry estimates. Furthermore, these lighting artifacts need to be minimized to ensure that they do not appear in the stitched images generated. We use a gain-mask image to minimize these lighting artifacts; a process we refer to as *gain-mask correction*, which produces *gain-corrected* images.

The gain correction images were obtained as follows. A white piece of card was affixed to the surface of

[3]This intensity drop-off towards the periphery is the result of both vignetting and the non-uniform light distribution provided by the LEDs. We attempt to account for both these effects when we apply gain-mask correction.

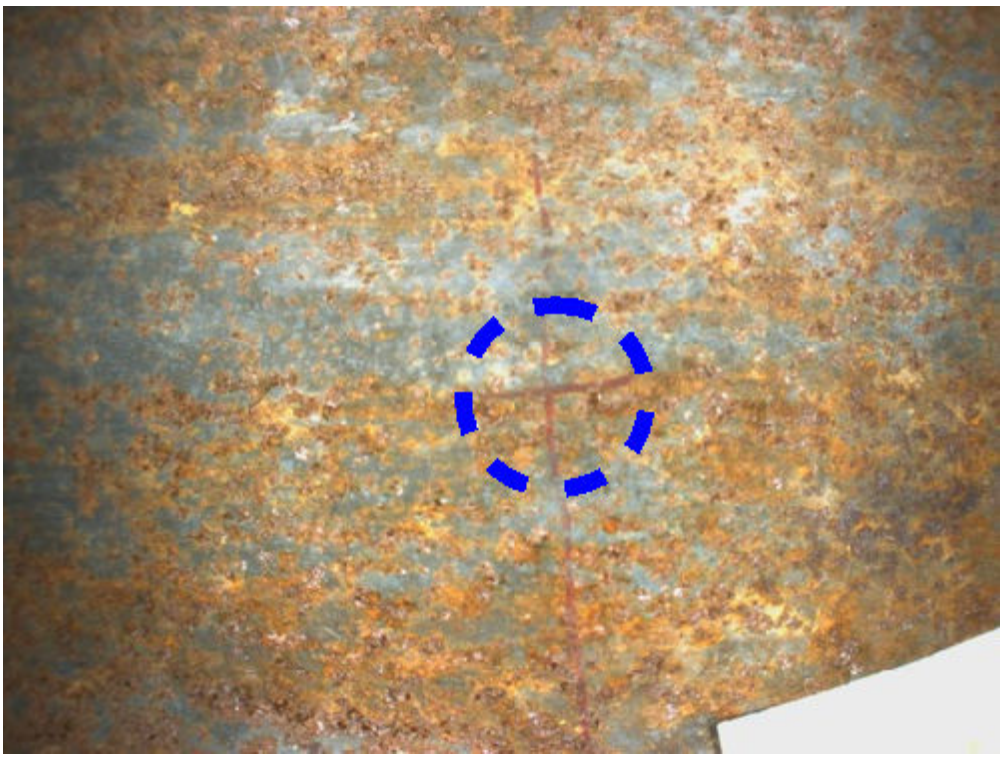
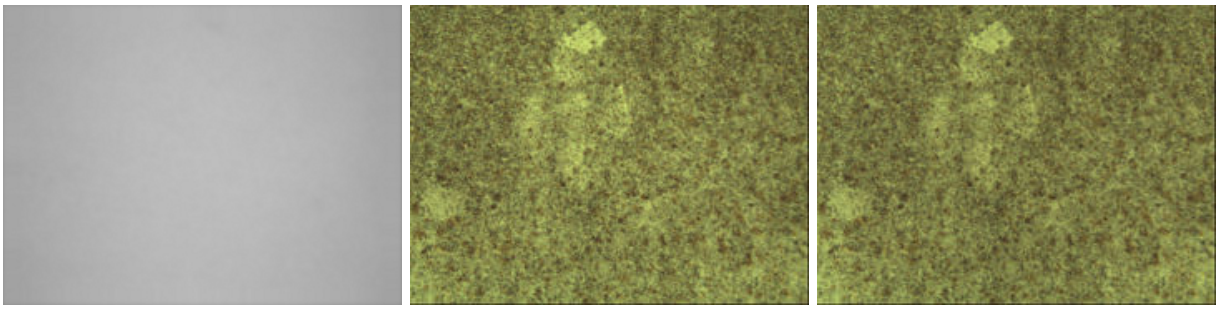


Figure 5: One of the markers (inside blue circle) in pipe 1 which is used to measure ground truth.

(a) Pipe 1.



(b) Pipe 2.

Figure 6: The gray-scale gain mask images (left column), original images from the cameras (middle column), and the gain corrected images (right column) obtained using gain mask correction. Notice that the light is more uniformly distributed in the pipe 2, which has a larger diameter than pipe 1.

each pipe. Each robot was then placed in its respective pipe, with the LEDs turned on, and several gray-scale image of the card obtained. The final gain-mask images were obtained by averaging these images and then smoothing with a Gaussian to reduce high frequency noise. The gain-mask images for each pipe are shown in the leftmost column in Fig. 6. Figure 6 also shows an example of the images for each pipe before (middle column) and after (right column) gain-mask correction. The gain-mask correction is simply the division of the original images with the gain-mask image. If color images are being used, each color channel is divided by the same gray-scale gain-mask image.

It is important to mention here that the gain corrected images in Fig. 6 have been manually rescaled for display purposes. This rescaling was applied as the gain correction process does not preserve the original dynamic range of the images. However, this is not an issue since the gain-corrected images are stored as double precision images, and both the dense and sparse algorithms have been designed to process double precision images with arbitrary dynamic range.

3.3 Reference Measurements: Monocular Scale Ambiguity

The dense and sparse visual odometry algorithms both estimate camera motion from a monocular image sequence. As a result they both need a form of *reference measurement* to resolve the well known scale-ambiguity which exists when estimating motion from a monocular image sequence. This scale-ambiguity is the inability to recover the Euclidean camera position $\mathbf{C}$ with metric units.

The dense algorithms measures, in units of pixels, the change in pose between adjacent images in a sequence. The reference measurement ζ used by the dense algorithms has units of pixels per meter, and is used to convert an x, y pixel shift between adjacent images to a change in Euclidean position with units of meters — this procedure will be outlined in more detail in Section 4. Figure 7 illustrates, for pipe 2, the process used to obtain the reference scale measurement ζ. A target image is fixed to the inner surface of the pipe, and an image of this target is captured. The centroids of the circles are manually measured and converted to coordinates in an undistorted (rectified)[4] perspective image. This un-distorting process is the same as that used by the dense algorithm discussed in Section 4. The metric distance between the centroids of the circles is measured using calipers, which enables the mean pixel per meter reference measurement ζ to b estimated. The reference scale measurements that were obtained for the two pipes are: pipe 1 - $\zeta = 10.4698 \times 10^{-3}$ pixels/meter; pipe 2 - $\zeta = 7.1636 \times 10^{-3}$ pixels/meter.

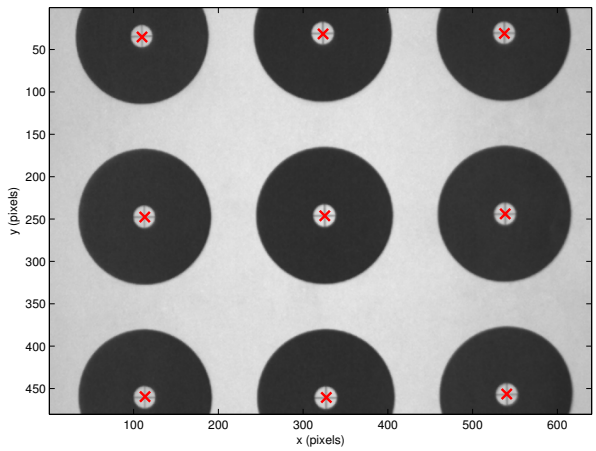
The sparse monocular algorithm enforces that all world points observed in the dataset images lie on the interior surface of the pipe. To enforce this constraint, and to resolve the monocular scale-ambiguity, the precise internal diameters of the pipes were measured[5]. The measured internal diameter was 153.32mm for pipe 1 (outer diameter of 152.4mm, or 6”), and 387.56mm for pipe 2 (outer diameter of 406.4mm, or 16”).

4 Dense Monocular Algorithm

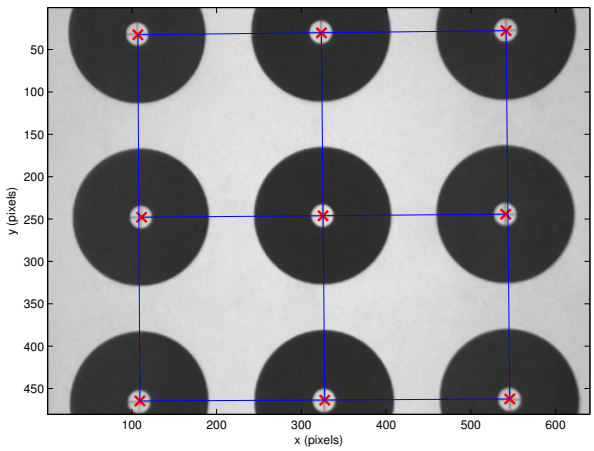
Image registration, or alignment, is the problem of aligning two or more images taken of a scene with some degree of overlap. Applications for image registration include: image mosaicing, video compression [15, 19], image stabilization [26], medical imaging [42], remote sensing [42] and motion estimation [13]. This vast array of applications make image registration one of the most widely studied problems in Computer Vision [38, 2].

[4]The Matlab Calibration Toolbox `http://www.vision.caltech.edu/bouguetj/calib_doc/` is used to calibrate the perspective cameras with both the pipe 1 and pipe 2 datasets.

[5]The diameters of the pipes quoted in table 1 are the values listed by the pipe manufacturers. They are not necessarily the internal diameters of the pipes.



(a) Original reference scale image with the centroids of the circles shown.



(b) Undistorted (rectified) reference scale image with the centroids of the circles shown.

Figure 7: The process used to obtain the reference scale measurement (example shown for pipe 2). The circle centroids are manually selected in the original image, and their positions mapped to the undistorted perspective image. The metric distance between the centroids of the circles is manually measured, which enables the mean pixel per meter reference measurement ζ to be found.

Image registration algorithms can be classified in two categories based on the amount of information they use as: *dense*, or *sparse*. Dense methods, also known as direct, or pixel-based, use all the pixel information in the image to find an optimal alignment under an assumption on the underlaying motion model relating image pairs. Sparse methods, also known as feature-based, rely on a small number of point feature correspondences to find the optimal registration parameters. These features are often referred to as corners, or keypoints, and are typically salient scene points detected in the image.

In order to estimate the ego motion of the camera between two images using dense registration, we need to define a motion model as well as an appropriate error function. Correct choices of the motion model and error function are crucial for accurate estimates. A motion model is characterized by its number of degrees of freedom (DOF). The more the DOF, the more difficult it is to fit the model. Generally, it is preferred to select a motion model with the minimum DOF. Ultimately, however, the choice of motion model is application dependent. Similarly, there are different error functions to assess the accuracy of the motion model fit, ranging from simple and computationally efficient measures, such as Sum of Absolute Differences (SAD) and Sum of Squared Differences (SSD), to more complex and more robust cost functions, such as the Root Mean Squared (RMS) or the Huber [14] cost function from robust statistics.

In this section, we discuss the use of dense 2D image registration to estimate the translational motion of perspective camera inside a straight and cylindrical pipe. First, we formalize the problem of dense registration, and then discuss two methods that are used to estimate camera motion, namely a full search approach and a model-based iterative refinement approach. Finally, we show how to use the two approaches to estimate the ego-motion of a perspective camera translating through a pipe, with the largest magnitude of translation being the the axial (h) direction of the pipe.

4.1 Dense Image Registration

Given two images $I_{\mathbf{t}}$ and I_n with non-zero overlap, where $I_{\mathbf{t}}$ is called a *template* image sampled at pixel locations $\mathbf{x}_i = (x_i, y_i)^T$ and image I_n called the *target* image, we seek a transformation $\mathcal{M} : \mathbf{x}_i \rightarrow \mathbf{x}'_i$ such

that the following criteria is minimized:

$$\sum_i \mathcal{E}\left(I_n[\mathbf{x}'_i], I_{\mathbf{t}}\left[\mathbf{x}_i\right]\right), \tag{11}$$

where $\mathcal{E}(\cdot)$ is an error function used to asses the quality of registration. Next, we present some of the common motion models in image space as well as some common error functions.

4.1.1 Motion models

As mentioned earlier, an accurate estimate of camera ego motion depends on the selected 2D motion model in image space. Motion models in image space can be considered as belonging to one of three classes [5]:

- **Fully parametric:** The local motion between pixels is modeled in a fully parametric form. Some common examples include translational, similarity, and affine motion models. Most relevant to our application is the two degree of freedom translational motion model in image space, described by:

$$\mathcal{M}_T(\boldsymbol{\chi}) = \begin{pmatrix} I_{2\times 2} & \mathbf{t} \end{pmatrix} \boldsymbol{\chi} = \boldsymbol{\chi}', \tag{12}$$

  where, $\mathbf{t} = \begin{pmatrix} t_x & t_y \end{pmatrix}^T$ is a 2×1 vector of translation amount in the x and y directions respectively, $\boldsymbol{\chi} = \begin{pmatrix} \mathbf{x} & 1 \end{pmatrix}^T$ is a homogeneous pixel coordinates, and $I_{2\times 2}$ is the 2×2 identity matrix.

- **Quasi-parametric:** The motion of local groups (neighborhoods) of pixels are described in a parametric form, although the motion of each pixel in the image can be defined independently. For example, the motion of rigid body in 3D space under perspective projection constrain the flow to be along a line, however, amount of pixel motion for groups of pixels may vary.

- **Non-parametric:** These motion models are used in optical flow computations and make use of uniformity of smoothness constraints.

In this work, we focus on parametric models where typically we assume translational only motion corresponding to translation while viewing a flat surface parallel to the image plane. While in practice this is not a correct model, we find that the resulting odometry estimates are more reliable than more complex models and still retain significant accuracy.

4.1.2 Error function

It is necessary to define an *error function*, or *cost metric*, to evaluate how accurately a given motion motion and its parameters describe, in image space, the change in camera pose between views. Some common error

functions include:

$$\mathcal{E}_{SAD} = \sum_i |I_{\mathbf{t}}[\mathbf{x}_i] - I_n[\mathbf{x}'_i]| = \sum_i |\epsilon_i| \tag{13}$$

$$\mathcal{E}_{SSD} = \sum_i \left(I_{\mathbf{t}}[\mathbf{x}_i] - I_n[\mathbf{x}'_i]\right)^2 = \sum_i \epsilon^2 \tag{14}$$

$$\mathcal{E}_{ZNCC} = -\frac{\sum_i \left(I_{\mathbf{t}}[\mathbf{x}_i] - \mu_{\mathbf{t}}\right)\left(I_n[\mathbf{x}'_i] - \mu_n\right)}{\sqrt{\sum_i \left(I_{\mathbf{t}}[\mathbf{x}_i] - \mu_{\mathbf{t}}\right)^2 \left(I_n[\mathbf{x}'_i] - \mu_n\right)^2}} \tag{15}$$

$$\mu_{\mathbf{t}} = \frac{1}{N}\sum_i I_{\mathbf{t}}[\mathbf{x}_i] \tag{16}$$

$$\mu_n = \frac{1}{N}\sum_i I_n[\mathbf{x}'_i], \tag{17}$$

where, N is the number of pixels in the region of image overlap. Sum of Absolute Difference (SAD) and Sum of Squared Difference (SSD) are the most efficient metrics to compute, with SAD being slightly more efficient. From a statistical point of view, SAD and SSD are fundamentally different. However, in the context of dense image registration, results obtained using SAD and SSD are practically identical. On the other hand, Zero-Normalized Cross Correlation (ZNCC) is more computationally expensive than SAD and SSD, but it is less influenced by illumination or color variations as it searches for shape similarity. The ZNCC correlation function in equation 15 is negated so that notation is consistent and we seek a minima in the cost space of all of the functions above.

In their current form, the error functions in equations. 13 through 15 do not take into account variations in illumination or gain, due to images taken at different exposures. The exception of course is ZNCC which accounts for a global bias and global gain, but does not account for any local changes. More details about accounting for bias and gain in the cost function can be found in [3, 38]. For this application, we opt to correct lighting artifacts and image gain beforehand and keep the registration step efficient.

4.2 Motion Estimation in a Pipe

Referring to figure 3, the robot translates in the h direction with minimal change in X, Y position or rotation. Therefore, we assume that there is no rotational or scale change between adjacent images taken by the camera on the robot, and estimate image motion using a fully-parametric dense translational model. Strictly speaking, the portion of the pipe surface observed by the camera is not planar. However, due to the relatively narrow field of view of the camera, depth variations are minimal and the majority of overlapping regions can be related by a translation model. Hence, we assume that the surface being imaged is planar. Further, we assume that the camera is calibrated and any optical distortion artifacts have been corrected.

Once the dense translational model parameters are determined in the image space, we can obtain 2D camera motion estimates inside the pipe in metric units using the reference scale measurement ζ (see section 3.3). We assume that the camera starts at the origin of the pipe coordinate frame of reference, and is looking directly upward whereby that the principal axis of the camera is aligned with the Y axis of the pipe (see figure 3). The initial pose P_0 is set to

$$P_0 = \begin{bmatrix} R^{-1} & \mathbf{C} \\ \mathbf{0}^T & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 0 & 1 & 0 & C_X \\ 0 & 0 & 1 & C_Y \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}, \tag{18}$$

and the change in pose Q between adjacent images, measured in the pipe coordinate frame of reference, is

$$Q = \begin{bmatrix} I_{3\times3} & \delta\,\mathbf{C} \\ \mathbf{0}^T & 1 \end{bmatrix}^{-1}, \quad \delta\,\mathbf{C} = \frac{1}{\zeta}(\delta y, 0, \delta x)^T, \tag{19}$$

where $\delta x, \delta y$ is the pixel shift (translation) between the images, and $I_{3\times3}$ is the 3×3 identity matrix. It is clear from equations.18 and 19 that the initial values for C_X and C_Y are unknown. If desired these offsets could be manually measured. However, they have no influence on the estimate of the distance the robot travels in the h direction down the pipe. Equation 19 also illustrates that a pure translational motion model is used, where an x and y translation in image space corresponds, respectively, to a translation in the Y and h direction in the pipe (see figure 3).

A simple and complete approach to find a translation motion between two images is to conduct a *full search* in the 2D $x, y \in \mathbb{Z}$ parameter space. Given sufficient overlap[6] between the two images of the same illumination, full search yields the correct solution. However, the solution lacks sub-pixel precision. Sub-pixel precision refinement can be obtained via *model-based*, or *iterative* methods. In order to speed up the registration process and reduce effects of noise, we resize images to quarter of their original resolution[7].

In the following sections, we describe the methods that we have used to estimate the parameters of the translational model, in particular full search, iterative model-based, and a hybrid approach that combines the two.

4.2.1 Full search

The translational motion is parameterized by two values, an x axis y axis image shift (see figure 2). In the image space, the search range of these shifts are bounded by image width (number of columns) and image height (number of rows). Referring to equation 18, we can use the reference measurement ζ to convert an x, y image shift to a change in camera position $\delta\,\mathbf{C} = (\delta X, \delta Y, \delta h)^T = (\delta y, 0, \delta x)^T$. Therefore, the search range of the image shifts could be further reduced based on knowledge about the camera's velocity and frame rate. If we know that the camera is traveling slowly down the pipe and mostly in the h direction, then we can search over a small subset of the x space, and even a smaller subset in the y space. The algorithm in pseudo code is shown in Algorithm 1.

Performance of the different error functions shown in equations. 13 through 15 was similar for almost all datasets. For the majority of the work reported here, we use SAD as it is most computationally efficient. However, we have found that ZNCC is superior to SAD in cases that include image blur combined with large image displacements. A more detailed discussion regarding the relative performance of the different cost functions is reserved for section 6.2.1.

4.2.2 Iterative Model-based Methods

Full search is usually conducted using integer increments. This may not be accurate enough and it is desirable to gain more accuracy via iterative refinement to obtain sub-pixel motion estimates. Several approaches exist, Lucas-Kanade [23] being the most cited. Other algorithms can be found in [40] as well as a model-based registration framework [5].

Iterative methods are better suited for estimating small image motions, which may not be the case for a majority of applications. To remedy this, iterative methods are usually implemented in coarse-to-fine hierarchical fashion. Image pyramids are used to subsample the image at different resolutions starting at the

[6]The amount of sufficient overlap depends on the amount of texture in the images, and the choice of the error function.

[7]For the 6" pipe (pipe 1), we use a resolution of 256×192, while for pipe 2 we use 160×120.

Algorithm 1 Full search for 2D dense translation motion between two images

FULL_SEARCH$(I_{\mathbf{t}}, I_n, x_{MAX}, y_{MAX})$

1 **for** $i = 1$ **to** x_{MAX}
2 **do for** $j = 1$ **to** y_{MAX}
3 **do** $E[i,j]$ = abs $\left(\frac{I_{\mathbf{t}}[i,j] - I_n[W(\mathbf{x};\mathbf{p})]}{\text{overlap\_area}}\right)$
▷ W is a warping function that warps image at location $\mathbf{x} = (x, y)$
▷ according to parameters $\mathbf{p} = \begin{pmatrix} t_x & t_y \end{pmatrix}^T$
4
5
6 **return** $(x_{BEST}, y_{BEST}) = \min_{i,j}(E)$

finest level and proceeding to the coarsest level, where each level reduces the resolution of the previous level by half, typically using Gaussian smoothing. The use of a hierarchical approach is not only important to handle large image motions, but also to reduce aliasing artifacts caused by large displacements [5]. Aliasing is a source of local minima in the error function that could be eliminated by searching for a solution across multiple scales.

In this work, we use the hierarchical model-based registration framework proposed by [5]. The basic idea is that motion between two frames can be approximated iteratively using a *motion model*, where the refinement process (typically Gauss-Newton) aims at minimizing the sum of squared differences between the template image I_t and the warped target image I_n based on the current estimation of motion.

The algorithm employs the intensity constancy assumption, common to the majority of optical flow and motion estimation methods:

$$I(\mathbf{x}, t) = I(\mathbf{x} - \mathbf{u}(\mathbf{x}), t - 1), \tag{20}$$

where I is the image, $\mathbf{x}$ is the position in the image, t denotes time, and $\mathbf{u}(\mathbf{x}) = (u(x, y), v(x, y))$ is the image velocity. Motion is then obtained by minimizing the SSD error:

$$\mathcal{E}(\{\mathbf{u}\}) = \sum_{\mathbf{x}} \left[I(\mathbf{x}, t) - I(\mathbf{x} - \mathbf{u}(\mathbf{x}), t - 1)\right]^2, \tag{21}$$

where $\{\mathbf{u}\}$ denotes the entire flow field in the region. The complex patterns of intensity variations between images requires solving a nonlinear least squares problem. Gauss-Newton method can be used to solve for the minima of $\mathcal{E}$. Using Gauss-Newton the incremental motion error is:

$$\mathcal{E}(\{\delta\mathbf{u}\}) = \sum_{\mathbf{x}} \left[\Delta I + \nabla I \cdot \delta\mathbf{u}(\mathbf{x})\right]^2, \tag{22}$$

where $\nabla I = (\partial I/\partial\mathbf{x}, \partial I/\partial\mathbf{y})^T$ is the image spatial gradient evaluated at $\mathbf{x}$, and $\Delta I(\mathbf{x}) = I(\mathbf{x}, t) - I(\mathbf{x} - \mathbf{u}_i(\mathbf{x}), t - 1)$, and $\mathbf{u}_i$ is the current motion estimate.

For a translational motion $\mathbf{u}(\mathbf{x}, \mathbf{t}) = (x + t_x, y + t_y)^T$, which can be written as

$$\mathbf{u}(\mathbf{x}) = \mathbf{U}(\mathbf{x})\mathbf{t}, \tag{23}$$

where

$$\mathbf{U}(\mathbf{x}) = \begin{pmatrix} 1 & x & 0 & 0 \\ 0 & 0 & 1 & y \end{pmatrix} \tag{24}$$

$$\mathbf{t} = \begin{pmatrix} t_x & t_y \end{pmatrix}^T. \tag{25}$$

The incremental motion $\delta\mathbf{t}$ is then:

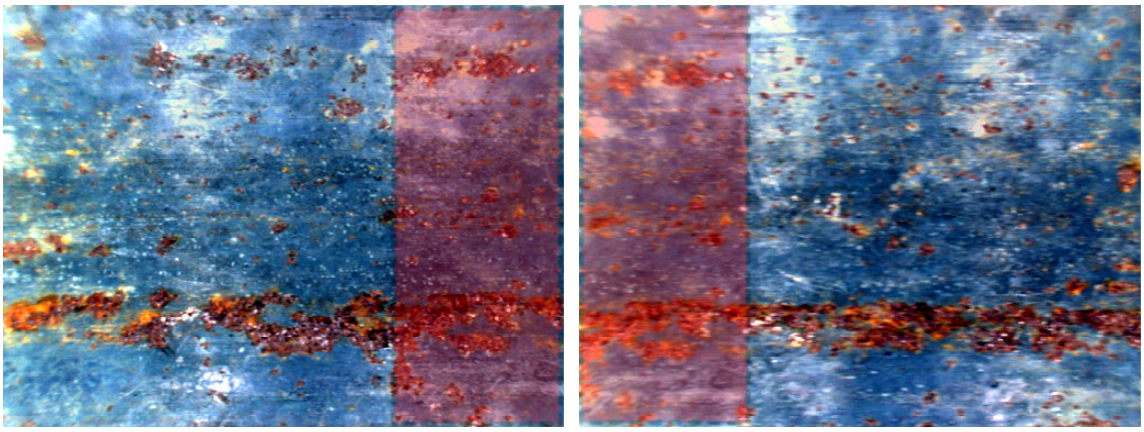
$$\mathcal{E}(\delta\mathbf{t}) = \sum_{\mathbf{x}} \left[\Delta I + (\nabla I)^T \mathbf{U}\delta\mathbf{t}\right]^2 \tag{26}$$

Minimizing with respect to $\delta\mathbf{t}$ results in the expression

$$\left(\sum_{\mathbf{x}} \mathbf{U}^T(\nabla I)(\nabla I)^T\mathbf{U}\right)\delta\mathbf{t} = -\sum_{\mathbf{x}} \mathbf{U}^T(\nabla I)(\Delta I) \tag{27}$$

Estimating the incremental motion $\delta\mathbf{t}$ is repeated iteratively until convergence, or for an empirically chosen constant number of iterations. More details can be found in [5].

Iterative refinement is still prone to local minima even when using image pyramids. This is especially true in the case of large image motions and scenes with a low number of distinctive features. For example, figure 8 shows two consecutive image pairs with a very small area overlap of approximately 30% combined with image distortion artifacts as well as indistinct texture. In such a case, iterative refinement approaches fail to converge to the global optima, even when using a multi-scale approach.



(a) Image 1 **(b) Image 2**

Figure 8: Difficult image pair; large displacement, blur and optical distortion correction artifacts. Overlap region is highlighted in boxes. The image colors have been modified for display purposes.

4.2.3 Full search followed by iterative refinement

In order to address cases when iterative methods fail to find the global minima of the cost function, we use a two step registration process. First, a full search is conducted to compute a set of initial motion parameters in integer increments between each pair of images. Given a careful selection of the error function, this step yields a solution that is within a pixel of the desired global minima. These initial estimates are fed into a model-based iterative refinement algorithm operating on the finest resolution pair of images without resorting to image pyramids.

This approach has the advantage of guaranteed convergence to the global minima in the search space, as well sub-pixel accuracy motion estimates. However, solutions obtained via iterative refinement are not necessarily better in all cases. Due to limited numerical precision, iterative methods will almost always estimate non-zero motion, even if the camera does not move in many frames. These non-zero motion estimates can occur as a result of image noise, and the addition of incorrect non-zero motion estimates has a negative impact on the accuracy of visual odometry estimates. This is the well known Markov random walk effect. This is reflected in our results in section 6.1.

5 Sparse Monocular Algorithm

The sparse monocular algorithm estimates camera ego-motion by observing the change in positions of sparse keypoint correspondences in adjacent images. This change in position is typically referred to as the *sparse optical flow*, and is used as the basis for many visual odometry algorithms [32, 33, 24, 39, 20, 34, 18, 1]. However, most visual odometry algorithms are designed to operate in unstructured environments where the relative Euclidean coordinates of scene points cannot be derived from a single image; they must be triangulated from pairs or sequences of images [12]. For our application the camera mounted on the robot is constrained to lie within a straight cylindrical pipe. If the pose P_a of the camera is known, then the Euclidean coordinates $\mathbf{X}$ of all the keypoints $\mathbf{x}$ points which appear in the image can be derived — this procedure is described in section 5.2. These scene points $\mathbf{X}$ are constrained to lie on the interior surface of the pipe, and the sparse algorithm uses this constraint to obtain accurate visual odometry estimates. Furthermore, the ability to recover the Euclidean coordinates of the scene points enables visual odometry estimates to be obtained with metric units of translation. The overall scale ambiguity inherent with traditional monocular visual odometry algorithms is therefore avoided.

The algorithm consists of a number of steps and procedures which are summarized here and discussed in greater detail in this section:

- **Obtaining scene point correspondences (section 5.1):** given a sequence of images I, find candidate keypoint correspondences $\mathbf{x} \leftrightarrow \mathbf{x}'$ between automatically selected key frames, and use epipolar constraints to remove incorrect correspondences.
- **Scene point constraint (section 5.2):** given the pose P_a of the camera for key-frame a, and the positions of the keypoints $\mathbf{x}$ in the image, find the coordinates $\mathbf{x} \mapsto \mathbf{X}$ of the world points on the interior surface of the pipe.
- **Visual Odometry Estimation (section 5.3):** estimate the change in pose (ego-motion) between adjacent key frames, and integrate the estimates to find the position of the camera at each key-frame time a.

5.1 Obtaining Scene Point Correspondences

The process used to find scene point correspondences between views consists of three primary steps: keypoint detection, matching and tracking. Here, the term keypoint is used to denote a salient scene point in the image.

A region-based Harris corner detector [11] is used to identify keypoints in the original (not undistorted) gain-corrected gray-scale images. Each image is divided into an equally spaced 6×8 grid, and the 20 most salient keypoints in each cell retained after non-maxima suppression of the saliency values using a 7×7 pixel wide window. A region-based scheme is used to ensure that there is a uniform distribution of keypoints throughout the image, as illustrated in figure 9. The saliency of a keypoint is defined by its Harris 'cornerness' score $\mathcal{C}$, which is a function of the eigenvalues λ_1, λ_2 of the gray-scale autocorrelation (second moment) matrix A evaluated at a given pixel[8]:

$$\mathcal{C} = \lambda_1 \lambda_2 + k\,(\lambda_1 + \lambda_2) \tag{28}$$

$$= \det(A) - k\,\mathrm{trace}(A), \tag{29}$$

[8]The autocorrelation function for an image is a measure of the cross-correlation of the image with itself. The autocorrelation matrix A is an approximation of the autocorrelation function. It is derived based on the assumption that the image function $I(x + \Delta x, y + \Delta y)$ at some small shift $\Delta x, \Delta y$ in the x, y directions respectively from $I(x, y)$ can be approximated by a first order Taylor expansion of the image about the point $\mathbf{x} = (x, y)^T$.

where $k = 0.04$ is an empirical constant, and

$$A(x,y) = \begin{bmatrix} \sum_W I_x(x,y)^2 & \sum_W I_x(x,y)\, I_y(x,y) \\ \sum_W I_x(x,y)\, I_y(x,y) & \sum_W I_y(x,y)^2 \end{bmatrix} \tag{30}$$

is the auto-correlation matrix evaluated at a pixel position $\mathbf{x}(x,y)$. Here, W is a square window element (integrating kernel). We use a Gaussian $G(\sigma_W)$ as our window element W with scale $\sigma_W = 3.0$ pixels. The first order derivatives I_x and I_y in the x and y directions respectively are computed using a derivative of Gaussian kernel $G(\sigma_D)$ with standard deviation $\sigma_D = 1.5$ pixels:

$$I_x = \frac{\partial I}{\partial x} = I * \frac{\partial G(\sigma_D)}{\partial x}, \quad I_y = \frac{\partial I}{\partial y} = I * \frac{\partial G(\sigma_D)}{\partial y}, \tag{31}$$

where the notation $*$ indicates a discrete convolution in image space.

Sub-pixel accuracy is obtained using a two-dimensional version of the sub-pixel quadratic interpolation scheme developed by Brown and Lowe in [6] as part of the Scale-Invariant Feature Transform (SIFT) [22]. The cornerness score $\mathcal{C}(x + \Delta x, y + \Delta y)$ at a sub-pixel shift $\Delta\mathbf{x} = (\Delta x, \Delta y)^T$ from $\mathbf{x} = (x,y)^T$ is approximated by the quadratic Taylor expansion

$$\mathcal{C}(\mathbf{x} + \Delta\mathbf{x}) = \mathcal{C}(\mathbf{x}) + \frac{\partial \mathcal{C}}{\partial \mathbf{x}}^T \Delta\mathbf{x} + \frac{1}{2}\Delta\mathbf{x}^T \frac{\partial^2 \mathcal{C}}{\partial \mathbf{x}^2}\Delta\mathbf{x}. \tag{32}$$

The variables $\frac{\partial \mathcal{C}}{\partial \mathbf{x}}$ and $\frac{\partial^2 \mathcal{C}}{\partial \mathbf{x}^2}$ are, respectively, the 2×1 and 2×2 matrices of first and second order partial derivatives of $\mathcal{C}$ evaluated at the pixel $\mathbf{x} = (x,y)^T$. Setting the derivative of equation 32 to zero,

$$0 = \frac{\partial \mathcal{C}}{\partial \Delta\mathbf{x}} = \frac{\partial \mathcal{C}}{\partial \mathbf{x}} + \frac{\partial^2 \mathcal{C}}{\partial \mathbf{x}^2}\Delta\mathbf{x}, \tag{33}$$

the sub-pixel shift $\Delta\mathbf{x}$ is computed, using Gaussian elimination, as

$$\Delta\mathbf{x} = -\frac{\partial^2 \mathcal{C}}{\partial \mathbf{x}^2}^{-1} \frac{\partial \mathcal{C}}{\partial \mathbf{x}}. \tag{34}$$

A 128-dimensional SIFT descriptor [22] is then evaluated for each keypoint from the gray-scale intensity values within a fixed sized region surrounding it. Each keypoint is defined by its sub-pixel position $\mathbf{x} = (x,y)^T$ and SIFT descriptor. Since the cameras used have been calibrated, each pixel position $\mathbf{x}_i$ can be mapped to a spherical coordinate $\boldsymbol{\eta}$, $\mathbf{x}_i \mapsto \boldsymbol{\eta}_i$, which defines a ray in space originating from the camera center. It is worth noting the the Harris corner detector is not suited for detecting the same scene points in different images separated by a large scale change. Although there are many *scale-invariant* algorithms which are capable of doing so [22, 8, 27, 4, 25, 16, 17], there is very minimal scale-change between adjacent images in our datasets, and the Harris corner detector is more efficient to implement than most of these algorithms. Although we do not present any results, we observed the Harris detector (coupled with the SIFT descriptor) to perform on par with the SIFT with respect to the number and reliability of keypoint correspondences obtained between images. The Harris corner detector has also been shown to perform consistently well when compared to other similar (i.e. *fixed scale*) keypoint detectors [36, 41, 37].

Given any two images, corresponding keypoints are found using the ambiguity metric [22] for the SIFT descriptors with a mutual consistency check. However, the correspondence between every adjacent pair of images are not used to estimate motion since many are separated by only a few pixels difference. We use a method similar to that of Mouragnon et al [28] and Tardif et al [39] to automatically select only key frames (images) that are used to obtain the visual odometry estimates. Starting with the first image I_0 in

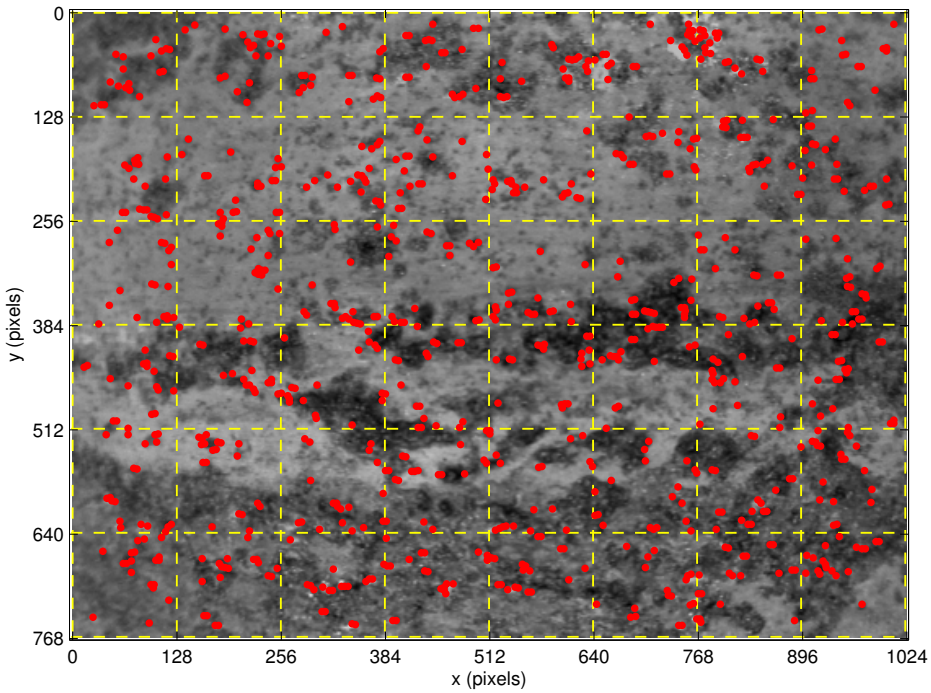


Figure 9: A region based Harris detector is used to extract salient scene points in the gray-scale images. The red dots are the keypoints, and the yellow lines indicate the equally sized 6×8 regions.

the sequence, we keep finding the correspondences between image I_0 and I_n, where n is the n^{th} image in the sequence. When the number of correspondences between image I_0 and I_n falls below a threshold, or the median magnitude of the sparse optical flow is above some threshold, the set of correspondences between images I_0 and I_{n-1} are kept, and images I_0 and I_{n-1} are assigned a camera pose $P_{a=0}$ and $P_{a=1}$ respectively. This process is then repeated starting at image I_{n-1}, and continued for the remainder of the sequence. The output of this process is a set of correspondences between adjacent camera poses P_a. Any outliers which may remain in these sets of correspondences are removed using RANSAC [9] and Nistér's five-point algorithm [32], although the simplified implementation of the five-point algorithm proposed by Li and Hartley could also be used [21].

5.2 Scene Point Constraint

The sparse monocular algorithm enforces that all world points visible in an image must lie on the interior surface of a straight cylindrical pipe with a constant radius r. If both the measured radius r and the pose P_a of the camera for key frame a are known, the world coordinate $\mathbf{X}_i$ defined in equation 1 of a keypoint can be derived from its coordinate $\mathbf{x}_i$ in the image. This is possible as the camera calibration parameters can be used to map an image coordinate $\mathbf{x}_i$ to a spherical coordinate $\boldsymbol{\eta}_i \in \mathbb{S}^2$ on the unit sphere — the ray from the center of projection of the camera to the scene point position $\mathbf{X}_i$ intersects the unit sphere at $\boldsymbol{\eta}_i$.

Recall from equations. 2 and 3 that a homogeneous scene point position $\boldsymbol{\mathcal{X}}_i$ in the pipe coordinate frame is related to the homogeneous scene point coordinate $\tilde{\boldsymbol{\mathcal{X}}}_i$ in the camera coordinate frame by $\boldsymbol{\mathcal{X}}_i = P^{-1}\tilde{\boldsymbol{\mathcal{X}}}_i$, where P is a matrix consisting of the 3×3 rotation $R = (\mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3)$ and 3×1 position $\mathbf{C} = (C_X, C_Y, C_h)^T$ of the camera. If $\boldsymbol{\eta}_i$ is the spherical coordinate of a world point at position $\tilde{\mathbf{X}}_i$, then we can write $\tilde{\mathbf{X}}_i = \kappa_i \boldsymbol{\eta}_i$, where κ_i is a scalar and $\tilde{\mathbf{X}}_i$ is the homogeneous coordinate, in the camera's frame of reference, of the world point on the surface of the pipe. Referring to figure 3, and to equation 1, a world

point with coordinate $\mathbf{X}_i = (X, Y, h)^T$ lies on the interior surface of a cylinder with radius r if

$$r^2 = X^2 + Y^2 \tag{35}$$

$$= (\kappa_i \mathbf{R}_1 \boldsymbol{\eta}_i + C_X)^2 + (\kappa_i \mathbf{R}_2 \boldsymbol{\eta}_i + C_Y)^2 . \tag{36}$$

Expanding (35) produces a quadratic in κ:

$$\kappa_i^2 \left((\mathbf{R}_1 \boldsymbol{\eta}_i)^2 + (\mathbf{R}_2 \boldsymbol{\eta}_i)^2 \right) + \kappa_i \left(2\, C_X\, \mathbf{R}_1 \boldsymbol{\eta}_i + 2\, C_Y\, \mathbf{R}_2 \boldsymbol{\eta}_i \right) + (C_X^2 + C_Y^2 - r^2) = 0. \tag{37}$$

If the camera is inside the pipe, two real solutions for κ_i are obtained from equation 37, one negative and the other positive. The positive solution is correct since $\tilde{\mathbf{X}}_i = \kappa_i \boldsymbol{\eta}_i$ must be a coordinate in front of the perspective camera. Once the solution for κ_i has been obtained, the homogeneous coordinate $\tilde{\boldsymbol{\chi}}_i$ in the camera frame of the world point is

$$\tilde{\boldsymbol{\chi}}_i = \begin{bmatrix} \kappa_i \boldsymbol{\eta}_i \\ 1 \end{bmatrix}, \tag{38}$$

whose position in pipe's coordinate frame is

$$\boldsymbol{\chi}_i = P^{-1} \tilde{\boldsymbol{\chi}}_i. \tag{39}$$

To summarize, given the position $\mathbf{x}_i$ of a keypoint in an image with camera pose P, the position $\mathbf{X}_i$ of the point in the surface of a straight cylindrical pipe with constant radius r can be found.

5.3 Visual Odometry Estimation

Six degree of freedom motion estimates are obtained using a number of steps:

- One degree of freedom motion estimation: Given all camera poses $P_0, P_1, \ldots, P_{a-1}$ up to key-frame a, and the associated world point coordinates $\mathbf{X}$ for all the keypoints in these frames, an estimate for P_a is obtained using a one degree of freedom motion model (camera translates down axis of the pipe).
- Optimization of initial camera position: If $a < N_{X,Y} < N$, where $N_{X,Y} = 10$ is a constant, optimize the initial position $\mathbf{C} = (C_X, C_Y, C_h)^T$ of the first camera pose P_0. The camera rotation is fixed, and only the values C_X and C_Y are optimized.
- Six degree of freedom refinement: If $a = \tau N$, where τ is any integer and $N = 50$ is a constant, refine the previous $2N$ camera poses using a six degree of freedom motion model.

Each of these steps, and the objective functions minimized during the optimization steps, are described in this section.

Before proceeding it is necessary to define a *global* index g for each world point. Assume that the same world point is detected in multiple images, and the keypoint correspondences found between key frames enables us to identify that these keypoints do in fact belong to the same world point. If the pose P is know for each of the images, then the coordinate $\mathbf{X}$ of this point could be derived from any of the images using the procedure described in the previous section. However, there are always some inaccuracies in camera pose estimates, calibration parameters, and the coordinates $\mathbf{x}$ of keypoint positions in the image. This means that the world point coordinate derived in one image will very likely be different to the coordinates derived from the other images. Therefore, we say that the world point has some global index g, and there are n estimates of its coordinate (one for each image); $\mathbf{X}(g)_{j=1}, \mathbf{X}(g)_{j=2}, \ldots, \mathbf{X}(g)_{j=n}$.

5.3.1 One degree of freedom motion estimation

Since the camera motion in a straight cylindrical pipes was constrained primarily to a change in translation δh down the pipe, the initial estimate of the camera pose P_{a+1} was set to

$$P_{a+1} = \begin{bmatrix} \hat{R}_a & R_a\,(\mathbf{C}_a + \delta\,\mathbf{C}) \\ \mathbf{0}^T & 1 \end{bmatrix}^{-1}, \quad \delta\,\mathbf{C} = (0, 0, \delta h)^T, \tag{40}$$

which has only a single degree of freedom $\delta\,h$. The estimate for δh is obtained using all relevant prior information in the sequence, here the coordinates $\mathbf{X}$ of all the world points up to and including key-frame a.

Define S as a matrix which stores information relating to all n estimates of the world point coordinate $\mathbf{X}(g)$ up to and including key-frame a. As discussed, the different estimates $\mathbf{X}(g)_{j=1}, \ldots, \mathbf{X}(g)_{j=n}$ arise from the fact that the same point may have been detected in many images. Referring to equation1 (pg. 3), the coordinates of a world point can be parameterized as $\mathbf{X}(g)_j = (h(g)_j, \rho(g)_j)^T$, where $\rho(g)_j = r\phi(g)_j$ and $\phi \in [0, 2\pi)$[9]. For each global index g, the matrix $S(g)$ contains the following information:

$$S(g) = \begin{bmatrix} n & \sum h(g)_j & \sum h(g)_j^2 & \sum \rho(g)_j & \sum \rho(g)_j^2 \end{bmatrix}, \tag{41}$$

where the summation is taken over all points $j \in \{1, \ldots, n\}$.

For an estimate of the camera matrix P_{a+1}, the procedure outlined in section 5.2 is used to find the coordinate $\hat{\mathbf{X}}(g) = (\hat{h}(g), \hat{\rho}(g))$ of the world points on the interior surface of the pipe. The notation $\hat{\mathbf{X}}(g)$ is used simply to denote that these estimates were obtained in the key-frame $a + 1$, and each point has a global index g. Note also that no two points detected in the same image can have the same global index g. A non-linear optimization (Levenberg-Marquardt) is used to find the estimate of $\delta\,h$ in equation 40 which minimizes the error

$$\epsilon = \sum_g \epsilon_g, \tag{42}$$

where

$$\epsilon_g = \sum_j \Big(\hat{h}(g) - h(g)_j\Big)^2 + (\hat{\rho}(g) - \rho(g)_j)^2 \tag{43}$$

is the cumulative error for each n estimates of the world point with global index g. By expanding equation 43,

$$\epsilon_g = n\,\hat{h}(g)^2 - 2\,\hat{h}(g)\,\sum_j h(g)_j + \sum_j h(g)_j^2 + n\,\hat{\rho}(g)^2 - 2\,\hat{\rho}(g)\,\sum_j \rho(g)_j + \sum_j \rho(g)_j^2, \tag{44}$$

the error term ϵ_g can be computed extremely efficiently since the values n, $\sum_j h(g)_j$, $\sum_j h(g)_j^2$, $\sum_j \rho(g)_j$ and $\sum_j \rho(g)_j^2$ can be retrieved directly from the matrix $S(g)$ in equation 41. The summations in equations. 42 and 44 are again taken over all points $j \in \{1, \ldots, n\}$. Once the optimized estimate for P_{a+1} is obtained, for each global index g the values of $S(g)$ can be updated to include all information up to and including key-frame $a + 1$.

5.3.2 Optimization of initial camera position

A limitation of the sparse algorithm is that an accurate initial pose of the camera needs to be found, and more specifically the initial position $\mathbf{C} = (C_X, C_Y, C_h)^T$ defined in equation2 (the rotation R remains

[9]For the camera configuration used, no scene points have an angle ϕ near $0, 2\pi$. If this did occur, an constant global offset ϕ_{global} could be applied to all scene points to prevent complications during the optimization procedure described in this section.

fixed). The magnitude of C_X and C_Y have a significant influence on the accuracy of the visual odometry estimates as they act like an overall scale factors. Therefore, a batch optimization using the first 10 frames is implemented in an attempt to find the most accurate estimate of the initial position $\mathbf{C}$ possible.

The optimization used does not minimize the errors between the scene point coordinates $\mathbf{X}_g$ on the pipe. The reason is that the coordinates of all of these points change during optimization, and it is not clear how a suitable normalization factor can be selected — moving the camera closer to the surface of the pipe minimizes the dispersion of the scene points and the magnitude of the errors. For this reason, the error minimized is defined in image space with respect to the coordinates of the keypoints detected in the original (not undistorted) images.

The estimate for the initial camera pose P_0 is set to the same initial pose used by the dense algorithms, which was given previously in equation 18 as

$$P_0 = \begin{bmatrix} R_0^{-1} & \mathbf{C}_0 \\ \mathbf{0}^T & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 0 & 1 & 0 & C_X \\ 0 & 0 & 1 & C_Y \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}, \tag{45}$$

where C_X, C_Y are the X, Y offsets of the camera from the pipe's central, and $C_h = 0$. These offsets are measured manually before optimization. This initial orientation has the camera's principal axis aligned with the Y axis of the pipe and the x pixel direction pointing down the axis of the pipe in the h direction. The coordinates $\mathbf{X}(g)$ of the scene points in this frame are then found and each matrix $S(g)$ populated. For the next 9 frames, a one degree of freedom estimate of the camera poses is then found using the procedure described in section 5.3.1. Once complete, the batch optimization of the first ten frames is implemented, where each pose P_a^{-1} is

$$P_a^{-1} = \begin{bmatrix} R_0^{-1} & \mathbf{C}_0 + \delta\,\mathbf{C}_a \\ \mathbf{0}^T & 1 \end{bmatrix}, \quad \delta\,\mathbf{C}_a = (0, 0, \delta\,h_a)^T. \tag{46}$$

As evident in equation 46, the initial offset C_X, C_Y remains the same for each frame. The offsets C_X, C_Y and each $\delta\ h_a$ are optimized using Levenberg-Marquardt.

For each image (key-frame) a, there are a set of keypoints in the image. Each keypoint has a global index g and a position vector $\mathbf{u}(g) = (u(g), v(g))^T$. If we can find m keypoints in the remaining ten images with the same global index g, the coordinate $\mathbf{u}'(g) = (u'(g), v'(g))^T$ in this image for each is found using a two-step mapping via the pipe wall. The error for this global index g, in this key-frame a, is evaluated as

$$\epsilon(g)_a = \sum_m \left[(u(g) - u'(g))^2 + (v(g) - v'(g))^2 \right]. \tag{47}$$

The combined error for all the keypoints in this key-frame is then

$$\epsilon_a = \sum_g \epsilon(g)_a. \tag{48}$$

The optimized estimates for C_X, C_Y and each $\delta\ h_a$ are obtained by minimizing the error

$$\epsilon = \sum_{a=1}^{10} \epsilon_a. \tag{49}$$

5.3.3 Six degree of freedom refinement

The initial estimates of the camera poses P obtained using the one degree of freedom model are reasonably accurate. However, we improve their accuracy by extending them to have six degrees of freedom. We choose to introduce these additional degrees of freedom now, since attempting to do this directly results in poor overall motion estimates (i.e. significant drift).

The six degree of freedom refinement is implemented using a *sliding window*. After each $a = \tau N$ frames, where τ is any integer and $N = 50$ is a constant, the six degree of freedom refinement for the previous $2N$ frames is implemented. This implementation is performed separately for each frame in the window, and no 'batch' optimization of all the frames is used. For camera pose P_k, where $1 \leq k \leq a$, the coordinate $\mathbf{X}(g)$ for each scene point in the image is found. Since these scene points have already contributed values to the matrices $S(g)$, their contributions are now subtracted. Then, for each six degree of freedom estimate of P_k found during optimization, where R is parameterized using quaternions, the new entries for $\mathbf{X}(g)$ in the image are found, and the error ϵ_g in equation 44 evaluated to find the total error ϵ in equation 42. The optimal estimate of P_k is defined as the one which minimizes the error ϵ. The optimization is implemented using Levenberg-Marquardt. Once the new optimized estimate for P_k has been found, the coordinates $\mathbf{X}(g)$ for the keypoints in the image are found, and the entries of S are updated to include these values. This process is implemented sequentially from the first to last image in the window. Since this method limits the degree by which the pose can change, it is run for an empirically selected number of times, which in the following experiments is two.

One advantage of this approach is that it is efficient to implement as no batch optimization of multiple camera pose estimates is performed. It is also suitable as the covariance matrices for all the world points in Euclidean space are similar.

5.3.4 Selection of incremental addition of degrees of freedom

During keypoint detection and matching an estimate of the Essential matrix is obtained. Referring to equation 7 in section 2, the change in pose $\tilde{Q}$ between two camera views (in the camera frame of reference) can be extracted from the essential matrix [12]. Then, making use of equation 10, the change in pose Q between frames in the pipe coordinate frame of reference can be found. This change in pose Q could be used as an initial estimate of the six degree of freedom change in pose between frames, and could be optimized using the objective function in equations. 43 and 44. However, as discussed, attempting to estimate the six degree of freedom motion directly (i.e. not within the sliding window scheme described) without first obtaining a one degree of freedom estimate gives unreliable results. One explanation for this is the difficulty in reliably decoupling rotational and translational motion when using narrow field of view cameras [10], particularly when:

- There are minimal depth discontinuities in the scene [7].
- There are small changes in camera rotation and/or translation [31, 30].
- The focus of expansion or contraction is outside the camera's field of view [10].

The difficulty in decoupling rotational and translational motion is illustrated in figure 10. Assume that there is a camera in the pipe, as shown in figure 10a, which is surrounded by the yellow view sphere. If the camera translates in the direction $-h$ down the pipe, the spherical flow field appears as shown in the left column of figure 10b, and the corresponding sparse optical flow field in a perspective image in the right column of the same figure — the red boundary indicates the angle of view for a typical perspective camera. If the camera were to rotate about the pipes X axis by some rotation R_X, then the resulting spherical

flow field and corresponding sparse optical flow field in a perspective image would appear as shown in figure 10c. Since the sparse optical flow fields for translational and rotational motion appear very similar in the perspective images, it is difficult to obtain an accurate initial estimate of motion from the Essential matrix, particularly when considering that the sparse optical flow values obtained in practice contain some degree of noise as a result of inaccuracies during keypoint localization.

6 Results and Discussion

6.1 Results

Visual odometry results were obtained for the dense (full search – section 4.2.1, and model-based – section 4.2.3) and the sparse algorithm for the datasets described in Section 3. The estimates of the distance traveled down the axis of the pipe versus the ground truth measurements are summarized in tables 2a through 2d. The absolute percentage errors are also recorded in these tables, where

$$\epsilon\,\% = 100\ \times\ abs\left(\frac{\text{ground truth - estimate}}{\text{ground truth}}\right). \tag{50}$$

For the datasets where the camera moves forward down the pipe (tables 2c and 2d), and then back in the opposite direction, the absoluter error is

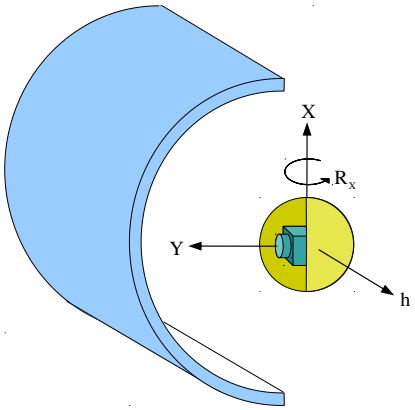
$$\epsilon\,\%\ \text{(forward \& reverse)} = 100\ \times\ abs\left(\frac{\text{forward estimate - reverse estimate}}{2\ \times\ \text{ground truth}}\right), \tag{51}$$

which is, in some respects, a type of loop closure error, although no loop closure detection is used. Figure 11 shows the estimates of the $\mathbf{C} = (C_X, C_Y, C_h)^T$ position estimates using the sparse estimate for pipe 1, dataset 1.

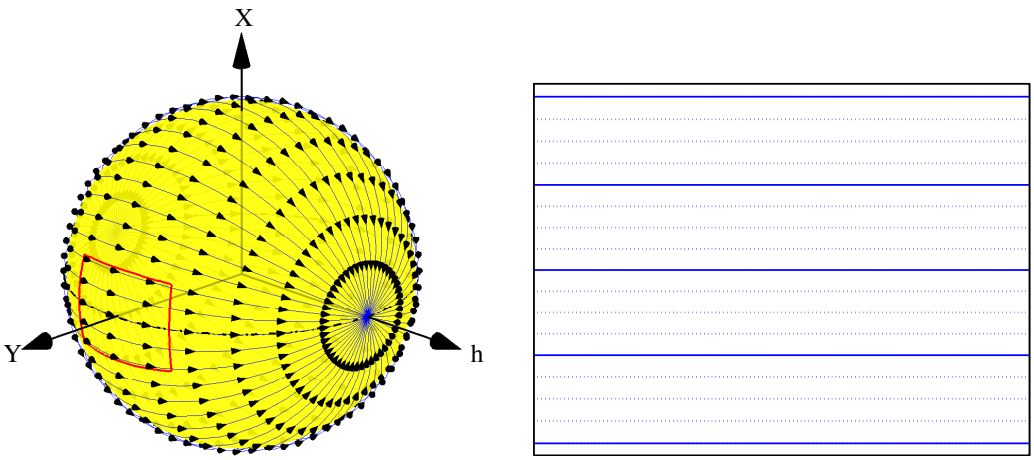
An example of the stitches images for the dense (model-based) and the sparse algorithm are illustrated, respectively, in figures 14 and 15 in appendix A for pipe 1 dataset 3 (forward run), and in figures 16 and 17 in appendix B for the pipe 2 dataset (forward run). The *tiled* versions of the stitched images are shown in these figures for display purposes only. For the dense algorithm, not every image is used to create the stitched images. The adjacent images used are separated by an x translation magnitude of at least 50 pixels. For the sparse results the images are stitched on the surface of the pipe, where a u, v coordinate in the stitched image corresponds to an $h, \rho = r\phi$ coordinate on the surface of the pipe. In both cases a simple stitching algorithm is used, where the value in the stitched image is the mean of all the values of the contributing gain-corrected images. We do not claim that this is an optimal stitching algorithm. It has been used as it makes irregularities in the stitched images, which correspond to inaccurate visual odometry estimates, easily identified.

6.2 Discussion

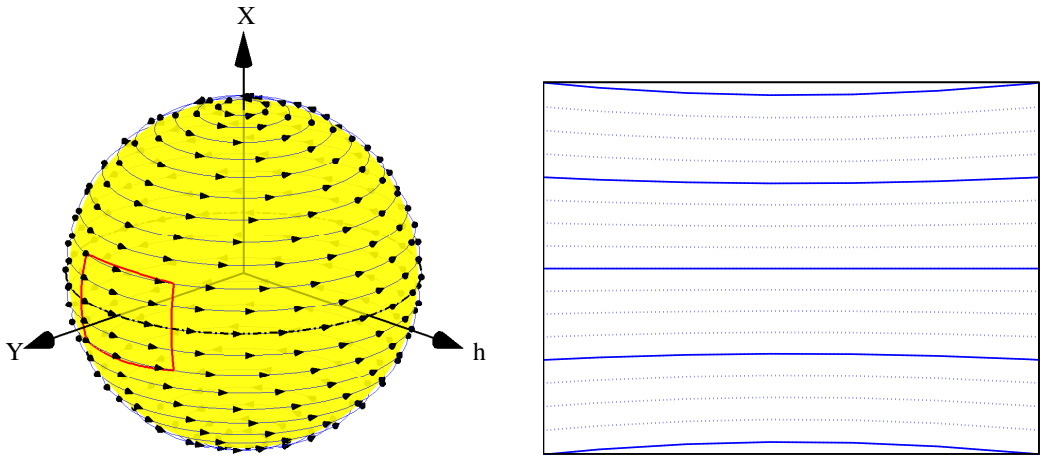
The results in tables 2a through 2d show that the dense and sparse algorithms are capable of finding accurate visual odometry estimates with respect to distance traveled down the pipe, with errors consistently less than 1 percent. The exception is the results obtained using the dense algorithms for the reverse sequence in pipe 2. Overall, the dense model-based approach outperforms the full-search approach, which is expected when considering that the full-search algorithm is limited to integer resolution. The results also indicate that, overall, the sparse algorithm performs marginally better than the dense algorithms. A number of factors which influence the accuracy of the visual odometry estimates are discussed in the following sections.



(a) A camera in a cylindrical pipe surrounded by a spherical view sphere (yellow). The coordinates shown are those of the pipe.



(b) Spherical flow field (left) and sparse optical flow field in a perspective image (right) for a camera translation down the h axis of the pipe.



(c) Spherical flow field (left) and sparse optical flow field in a perspective image (right) for a camera rotation $-R_X$ about the X axis of the pipe.

Figure 10: There is an ambiguity when attempting to decoupling rotational and translation motion using the sparse optical flow in a typical perspective camera. The red outline indicates the camera's field of view on a spherical view sphere. This ambiguity makes it difficult to obtain an accurate six degree of freedom motion (change in pose) estimate between camera views.

Table 2: Visual odometry results for distance traveled down the pipe - (a) pipe 1 dataset 1, (b) pipe 1 dataset 2, (c) pipe 1 dataset 3, and (d) pipe 2.

(a) Pipe 1 (6 meters, 6" diameter): dataset 1.

Metric	Dense		Sparse
	Full Search	Model	
Ground Truth (mm)	5844.4	5844.4	5844.4
Estimate (mm)	5797.7	5853.4	5826.5
Error ϵ (mm)	46.7	-9.0	17.9
Abs Error ϵ (%)	0.799	0.154	0.306

(b) Pipe 1 (6 meters, 6" diameter): dataset 2.

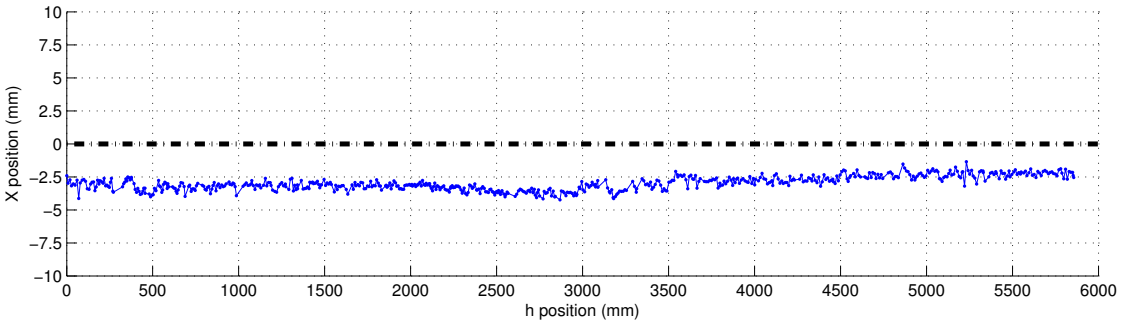
Metric	Dense		Sparse
	Full Search	Model	
Ground Truth (mm)	5844.4	5844.4	5844.4
Estimate (mm)	5792.4	5802.0	5827.5
Error ϵ (mm)	52.0	42.4	16.9
Abs Error ϵ (%)	0.890	0.725	0.289

(c) Pipe 1 (6 meters, 6" diameter): dataset 3.

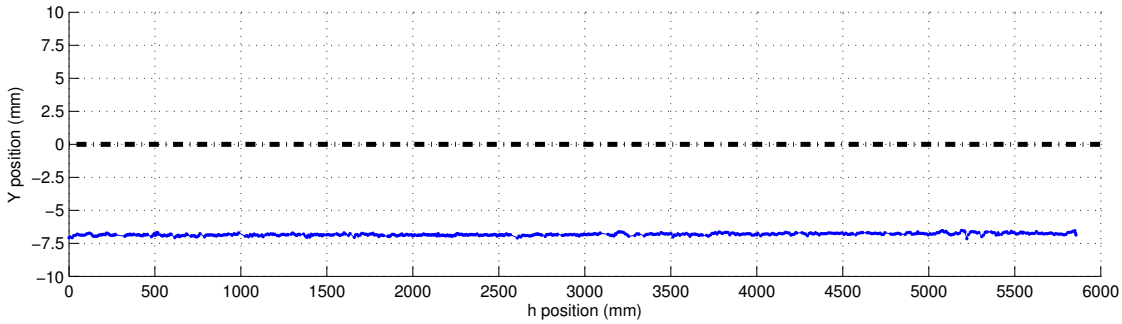
Metric	Dense		Sparse
	Full Search	Model	
Ground Truth (mm)	5844.4	5844.4	5844.4
Forward Estimate (mm)	5858.8	5835.7	5836.9
Forward Error ϵ (mm)	-14.4	8.7	7.5
Forward Abs Error ϵ (%)	0.246	0.149	0.128
Reverse Estimate (mm)	5893.3	5865.5	-5840.9
Reverse Error ϵ (mm)	-48.9	-21.1	3.5
Reverse Abs Error ϵ (%)	0.837	0.361	0.060
Total Estimate (mm)	-34.5	-29.8	3.98
Total Error ϵ (mm)	34.5	29.8	-3.98
Total Abs Error ϵ (%)	0.295	0.255	0.034

(d) Pipe 2 (4 meters, 16" diameter): dataset 1.

Metric	Dense		Sparse
	Full Search	Model	
Ground Truth (mm)	3391.0	3391.0	3391.0
Forward Estimate (mm)	3382.0	3362.6	3383.9
Forward Error ϵ (mm)	9.0	28.4	7.1
Forward Abs Error ϵ (%)	0.27	0.84	0.21
Reverse Estimate (mm)	3516.2	3490.0	3370.3
Reverse Error ϵ (mm)	-125.2	-99	20.7
Reverse Abs Error ϵ (%)	3.69	2.92	0.61
Total Estimate (mm)	-134.3	-127.4	13.6
Total Error ϵ (mm)	134.3	127.4	-13.6
Total Abs Error ϵ (%)	1.980	1.879	0.201



(a) X, h **camera position.**



(b) Y, h **camera position.**

Figure 11: The X, Y, h position estimate for the camera (pipe 1, dataset 1) using the sparse algorithm. The dashed line is the central axis of the pipe. The radius of the pipe is 76.66 millimeters.

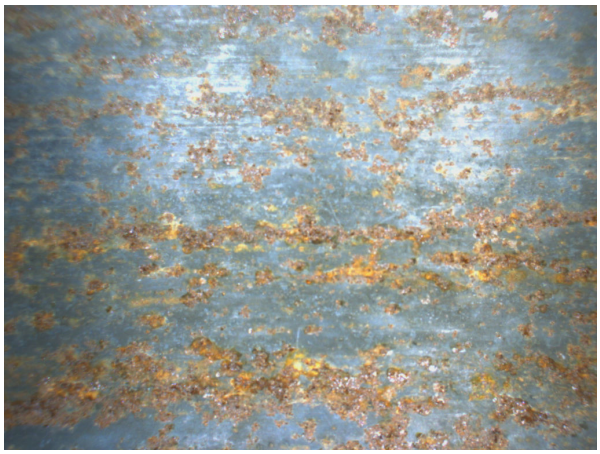
6.2.1 Gain-correction and illumination artifacts

Several considerations need to be addressed for a dense motion estimation to produce accurate estimates. Since dense motion estimation uses pixel intensities to compute the best alignment parameters, it is of critical importance that images have a similar illumination.

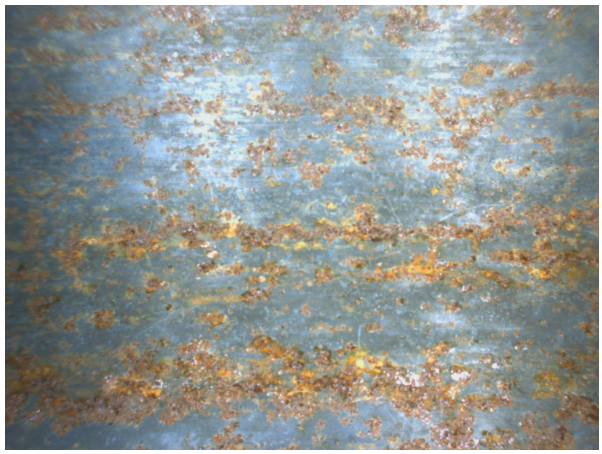
Figure 12 shows two images taken inside the pipe without correcting the lighting artifacts due to both the non diffuse LED lighting and specular reflections. As can be seen in the figure the global minima in the cost space corresponds the zero motion $(0, 0)^T$. However, this is an incorrect estimate of the true motion parameters between the frame pair. This is due to the strength of lighting artifacts that dominate the error surface. The true motion parameters in this case correspond to a local minima in the cost space.

Another important consideration for the dense algorithms is the choice of the error function. SAD is efficient and quick to compute, however, it fails in some cases. Figure 13 shows an example of the cost surfaces generated by each of SAD, SSD and ZNCC error functions. The illustrated cost surfaces correspond to a full search for the best translation parameters between the image pair shown in figure 8. As can be seen, SAD and SSD share the same shape of cost surface, with the exception of scale. ZNCC however, is distinctly different. In this case, ZNCC finds the correct motion parameters, while the global minima in SAD and SSD fails to capture the correct motion parameters.

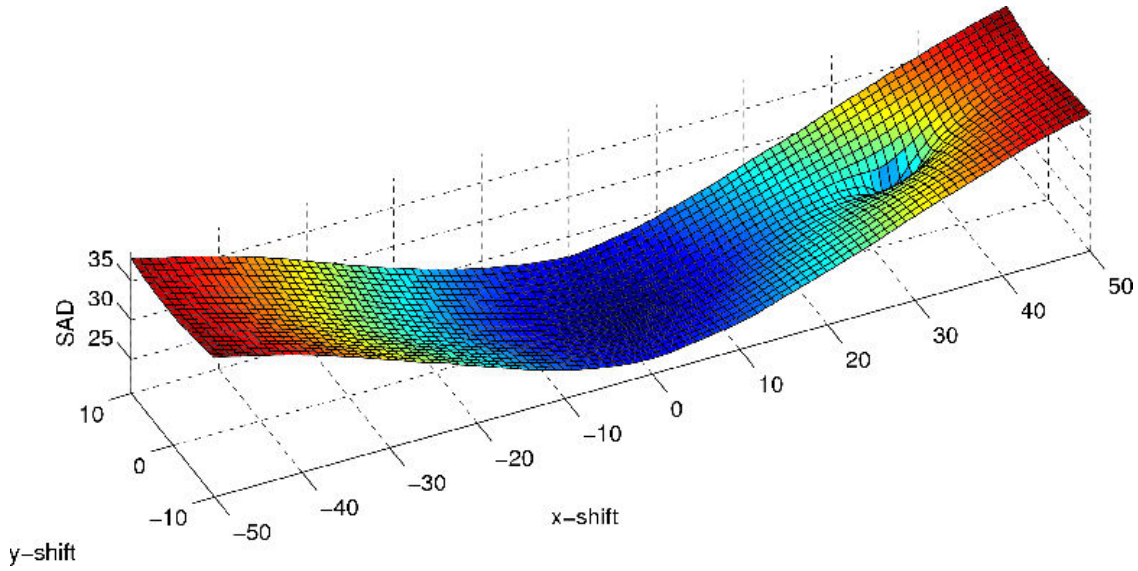
In contrast, the motion estimates obtained using the sparse algorithm are less affected by lighting variations and artifacts between frames. These lighting variations and artifacts have a negative impact during keypoint detection by limiting the number of the same scene points that can be detected and correctly matched in adjacent images. However, the sparse motion estimates are based only on the locations of the matched keypoints and not on the image intensity values. In any case, strong lighting variations may have



(a) Image 1



(b) Image 2



(c) Cost surface

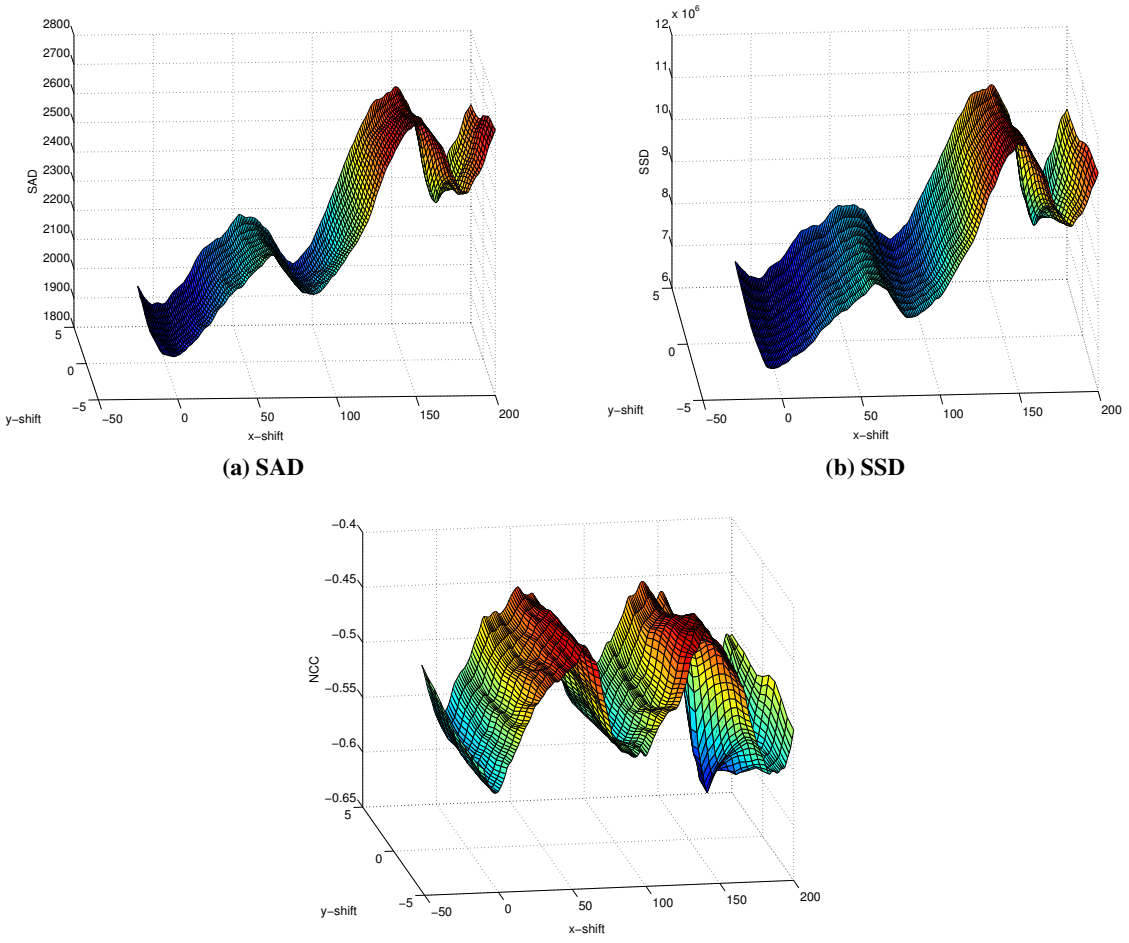
Figure 12: Failure case for dense motion estimation due to strong lighting artifacts causing large differences in illumination between the images. The global minima at $(0, 0)^T$ corresponds to the alignment of the specular reflections in the images. However, a local minima at approx. $(35, 1)^T$ corresponds to the correct value of the translational motion between the images.

some influence on the localization accuracy of the keypoints, which is dependent on the image intensity values, so it is still beneficial to minimize lighting variations when possible using the sparse algorithm.

For both the dense and sparse algorithms, it is also necessary to correct for lighting variations using the gain-correction image to ensure that consistent stitched images are produced.

6.2.2 Inherent motion model assumptions

The accuracy of the dense visual odometry estimates have the potential to be limited by the inherent motion model assumptions made by the dense algorithms. As discussed in section 4, the dense algorithms assume that each image pair is separated by a u, v pixel shift, which corresponds to a two degree of freedom trans-



(a) SAD **(b) SSD**

(c) ZNCC

Figure 13: Comparison between different error functions. The cost surface corresponds to motion between the two images shown in figure 8. We show the negative of ZNCC to keep the problem as a minimization problem

lation model in Euclidean space (X, h motion of the camera in the pipe). However, the motion of the two robot platforms is well described by a one dimensional translation model (along the h axis), which is one of the reason that accurate visual odometry estimates were found using the dense algorithms. If the robot were to rotate about the central axis (i.e. travel up the side of the pipe), the accuracy of the visual odometry estimates would be expected to deteriorate.

Although the sparse algorithm models six degree of freedom motion, it is still limited in some respects by the fact that it first finds a one degree of freedom estimate of motion for a sequence of 100 frames before applying six degree of freedom refinement. If the camera were to rotate significantly about the central axis of the pipe for example, then the optimized six degree of freedom refinement estimates may in fact converge on local minima. As mentioned, no large rotations about the central axis occurred, so it was suitable to assume that for short distances traveled (100 frames, which corresponds to approximately 1 meter traveled down the pipe) the camera motion was well approximated by a one degree of freedom translation model.

6.2.3 Pipe diameter and camera angle of view

Both the diameter of the pipe and the angle of view of the camera can impact the accuracy of the dense and sparse algorithm differently.

For the pipe 1 datasets, the diameter of the pipe is 152.4mm (6"), and the 70° horizontal angle of view of the camera is moderately large. This means that the images obtained for the pipe 1 datasets exhibit a significant degree of foreshortening in the images since the distance of the camera to different regions on the surface of the pipe changes. This is most apparent when viewing the image sequences, where it is easy to identify that regions near the top and bottom of the images appear to move slower than those near the central band of the image. This foreshortening is also the reason why the upper and lower regions in the stitched images for the pipe 1 dataset in figure 14, obtained using the dense model-based algorithm, appear blurred. As discussed in Section 4, the dense algorithms assume that adjacent images are separated by an x, y pixel shift. This foreshortening can therefore have a negative impact on the accuracy of the visual odometry algorithms obtained using the dense algorithms. However, it is encouraging to observe that accurate visual odometry estimates can be obtained using the dense algorithm for the pipe 1 datasets, even when the images contain significant foreshortening.

The diameter of the pipe for used in the pipe 2 dataset is significantly larger at 406.4mm (16"), and the 25° horizontal angle of view of the camera is narrow. As a result, the degree of foreshortening in the pipe 2 dataset images is far less than that in the pipe 1 dataset images. One would therefore expect that the accuracy of the visual odometry estimates found using the dense algorithms would potentially be more accurate than those for the pipe 1 datasets. However, there is no clear evidence in the results that this is the case. One possible explanation for this is the fact that the robot moved slightly up the side of the pipe while capturing the images in the pipe 2 dataset. As mention in the previous section, the dense algorithms do not model this type of motion.

The sparse algorithm uses the known radius of the pipe to obtain camera pose estimates, which means that the curvature of the pipe, and any foreshortening in the images, is accounted for when estimating camera pose. However, a number of factors influence the accuracy of monocular visual odometry estimates such as the distance of scene points to camera, and the angle of view of the camera. As mentioned, for the smaller diameter pipe used in the pipe 1 datasets, the scene points are closed to the camera than those in the pipe 2 dataset. The horizontal angle of view of the camera is also greater. Both these factors can have a positive effect on the accuracy of monocular visual odometry estimates [10, 7, 30]. The resolution of the camera used in the pipe 2 dataset was also greater than that used in the pipe 2 dataset (1024×768 pixels versus 640×480 pixels), which means that the accuracy of keypoint localization for the pipe 1 dataset is likely to be more accurate than that for the pipe 2 dataset. This improved accuracy of keypoint localization can have a positive effect on the accuracy of the visual odometry estimates. The sparse results support these claims. Overall, the percentage accuracy of the sparse visual odometry estimates are better for the pipe 1 datasets than those for the pipe 2 dataset.

7 Conclusions and Future Work

We have investigated two classes of algorithms that can be used to estimate monocular visual odometry in a straight cylindrical pipe as the first step towards a visual perception system for LNG pipes inspection. The first class included a number of *dense* algorithms that use all image information to compute an optimal translational alignment between images in the image space. This alignment is then converted to an estimate of camera ego-motion using a suitable reference scale measurement. The second class was a *sparse* algorithm which extracts a small set of salient keypoints in the images, and uses keypoint correspondences between views to solve the for the 6 degree of freedom ego-motion of the camera.

Advantages of the dense algorithms include the ability to obtain an accurate estimate of motion in the presence of blur or large image displacements. It is also possible to refine motion estimates in the image space to a sub-pixel precision. However, the approach assumes that the camera observes a planar surface, and the motion model is only able to model camera translation. Dense registration using the simple cost metrics discussed in this work require constant illumination between image pairs. We attempt to remove lighting variations using a suitable *gain-mask*. The dense algorithms estimate the motion between every frame pair independently from the rest of the data. This has the strong advantage of not accumulating error over time. However, it could be argued that making use of motion estimates from adjacent frame pairs could enhance accuracy.

On the other hand, the sparse algorithm is more general and incorporates more degrees of freedom which enables it to model the pipe surface more accurately. Further, the algorithm could handle small rotations robustly. However, it requires more overlap between image pairs than the dense algorithms. This should not be of any concern as we can control the frame-rate of the camera as well as the velocity of vehicle traveling inside the pipe. Also, the algorithm could be more affected by noise as the number of degrees of freedom it estimates is larger than the dense algorithms. Since the sparse approach is incremental and error accumulation overtime is inevitable, an optimization scheme is very important to reduce error growth.

The dense and sparse algorithms were evaluated on different datasets taken from different pipes and we have found them to be comparable, repeatable and robust. In the experiments presented, the dense and sparse algorithms were able to estimate camera motion within 1% accuracy of the ground truth, which is a very encouraging result for using visual odometry in constrained environment, such as a pipe.

In the future, we plan to integrate the algorithms developed in this work into a more complete visual perception system for LNG pipe inspection. This will included the generation of appearance maps associated with dense Euclidean structure, and algorithms to register appearance maps of the same pipe produced at different times. By monitoring changes in these appearance maps over time, we anticipate that a system can be designed that is capable of detecting corrosion and measuring its rate of change. Further, we intend to make use of different camera systems, such as stereo cameras to recover 3D information as well as catadioptric and fisheye camera systems to obtain a full 360° view of the interior of the pipe surface using a single camera. Catadioptric and fisheye systems are of particular interest as they allow the creation of very rich appearance maps.

Acknowledgments

The authors would like to thank James Ketterer and the members of the E-lab at the Natoional Robotics Engineering Center (NREC) for their help in lighting, and the members of the education group at NREC for their help in building the test rig prototype in Pittsburgh. Also, thanks to Mohamed Mustafa for his help in building a data collection prototype in Qatar.

A Tiled Stitched Images for Pipe 1

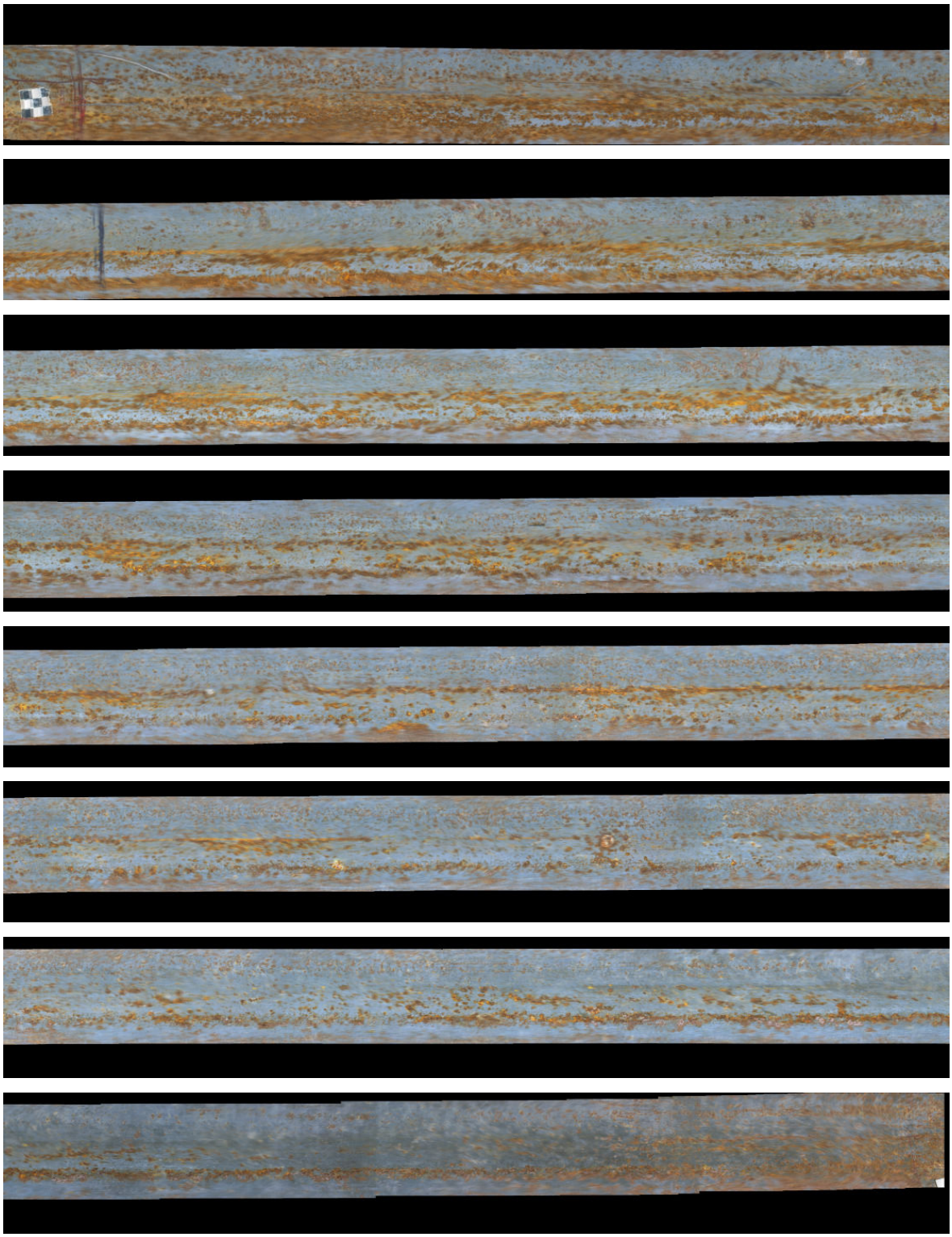


Figure 14: The tiled stitched image for pipe 1 (6 meters long 152.4mm (6") diameter) dataset 3 (forward) obtained using the dense model-based algorithm. The images are stitched in image space.

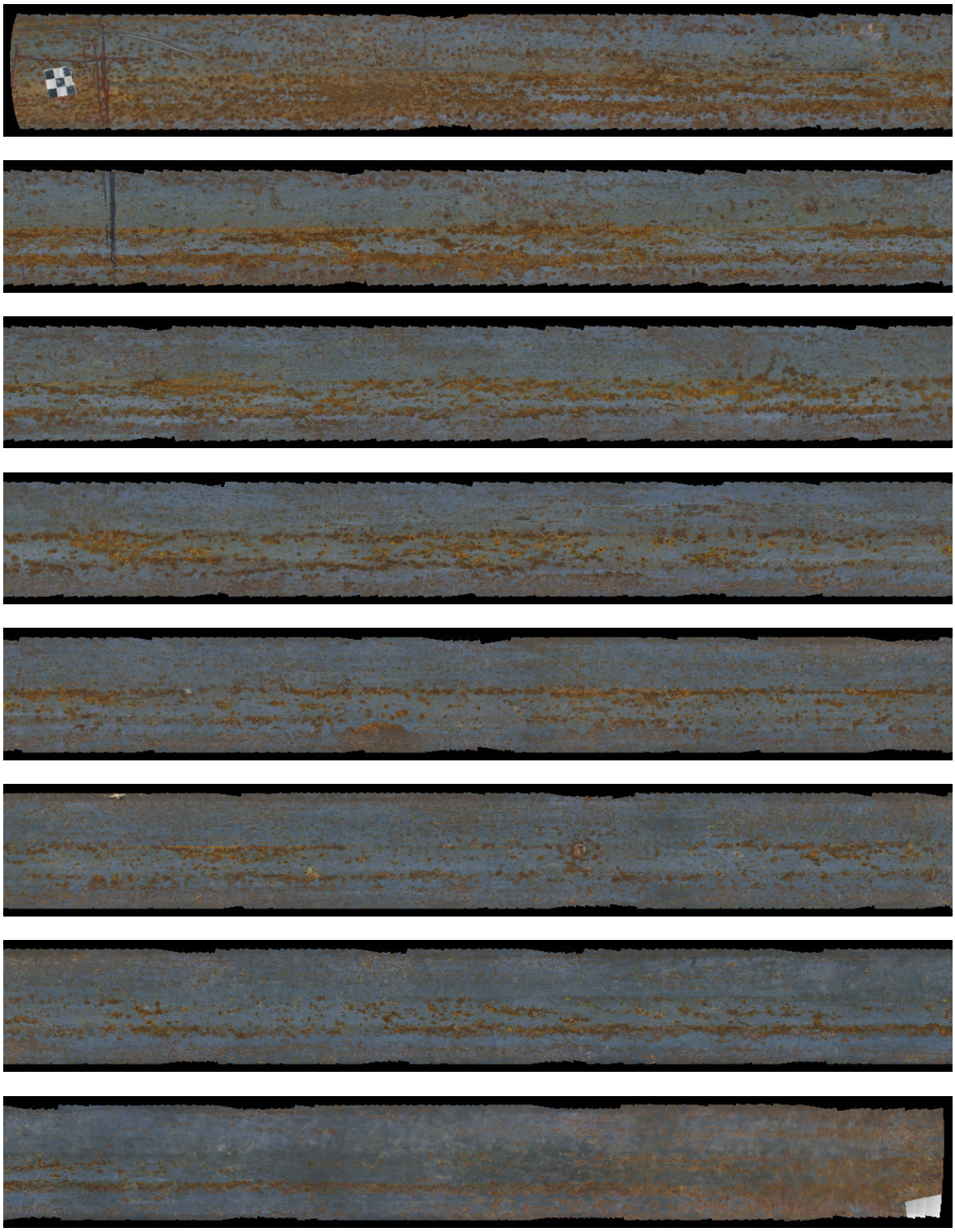


Figure 15: The tiled stitched image for pipe 1 (6 meters long 152.4mm (6") diameter) dataset 3 (forward) obtained using the sparse algorithm. The images are stitched on the surface of the pipe, where a u, v coordinate in the stitched image corresponds to an $h, \rho = r\phi$ coordinate on the surface of the pipe.

B Tiled Stitched Images for Pipe 2

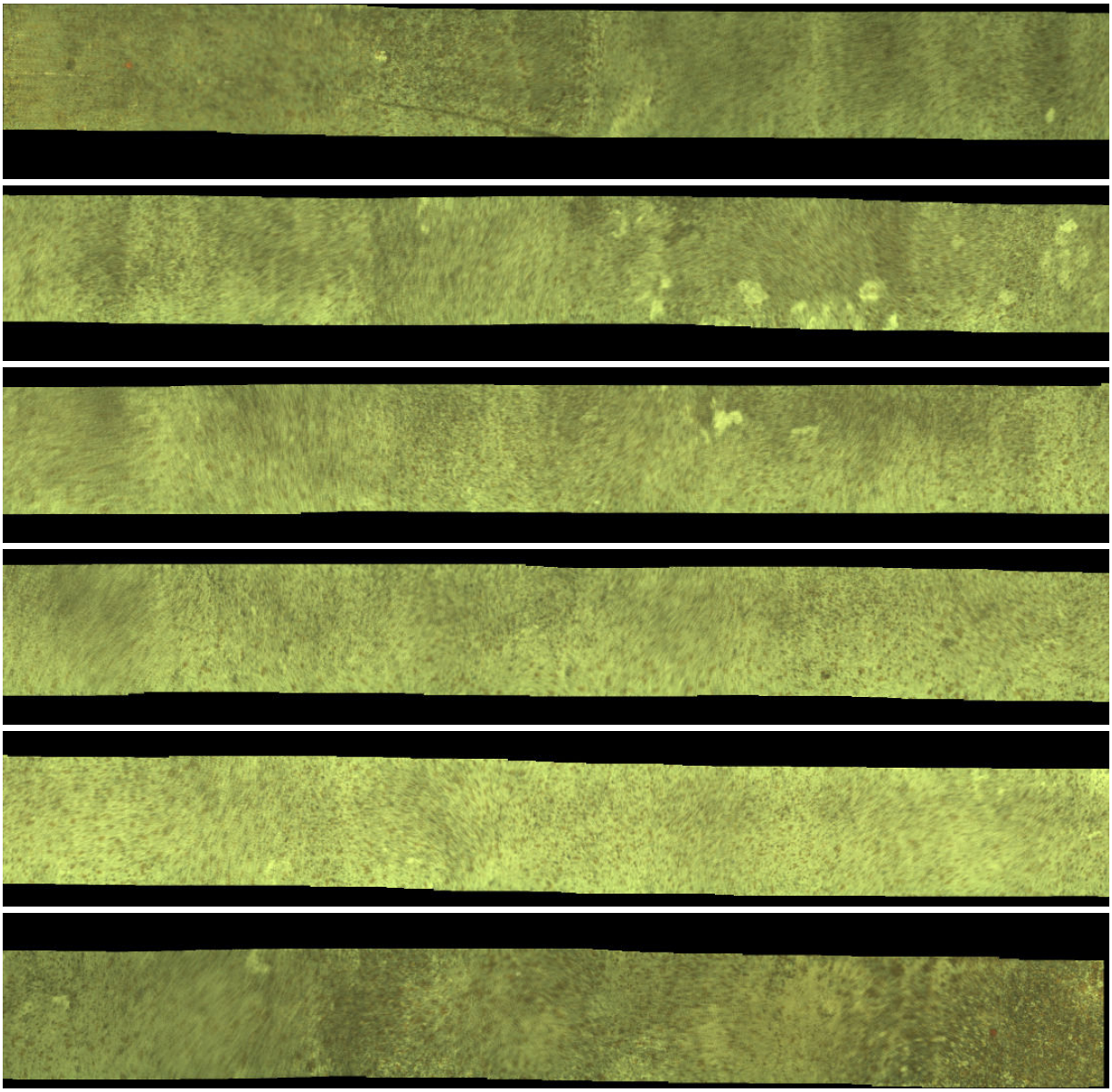


Figure 16: The tiled stitched image for the pipe 2 (4 meters long 406.4mm (16”) diameter) dataset (forward) obtained using the dense model-based algorithm. The images are stitched in image space.

Figure 17: The tiled stitched image for the pipe 2 (4 meters long 406.4mm (16”) diameter) dataset (forward) obtained using the sparse algorithm. The images are stitched on the surface of the pipe, where a u, v coordinate in the stitched image corresponds to an $h, \rho = r\phi$ coordinate on the surface of the pipe.

References

[1] Motilal Agrawal and Kurt Konolige. Rough terrain visual odometry. In *International Conference on Advanced Robotics (ICAR)*, August 2007.

[2] Arthur Ardeshir. *2-D and 3-D Image Registration*. John Wiley & Sons Inc., 2005.

[3] S. Baker, R. Gross, and I. Matthews. Lucas-kanade 20 years on: A unifying framework: Part 3, 2002.

[4] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, June 2008.

[5] James Bergen, P. Anandan, Keith Hanna, and Rajesh Hingorani. Hierarchical model-based motion estimation. In *ECCV'92*, pages 237–252, 1992.

[6] Matthew Brown and David Lowe. Invariant features from interest point groups. In *Proceedings British Machine Vision Conference*, pages 656–665, Cardiff, Wales, September 2002.

[7] Konstantinos Daniilidis and Hans-Hellmut Nagel. The coupling of rotation and translation in motion estimation of planar surfaces. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 188–193, June 1993.

[8] Yves Dufournaud, Cordelia Schmid, and Radu Horaud. Matching images with different resolutions. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 612–618, June 2000.

[9] Martin A Fischler and Robert C Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[10] Joshua Gluckman and Shree Nayar. Ego-motion and omnidirectional cameras. In *Proceedings International Conference on Computer Vision*, pages 999–1005, 1998.

[11] C.G. Harris and M.J. Stephens. A combined corner and edge detector. In *Proceedings Fourth Alvey Vision Conference*, pages 147–151, 1988.

[12] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2 edition, 2003.

[13] Berthold K. P. Horn. *Robot Vision (MIT Electrical Engineering and Computer Science)*. The MIT Press, mit press ed edition, March 1986.

[14] Peter J. Huber. *Robust Statistics*. John Wiley & Sons Inc., 1981.

[15] M. Irani, P. Anandan, and S. Hsu. Mosaic based representations of video sequences and their applications. In *Proceedings of the Fifth International Conference on Computer Vision*, ICCV '95, pages 605–, Washington, DC, USA, 1995. IEEE Computer Society.

[16] Timor Kadir and Michael Brady. Scale, saliency and image description. *International Journal of Computer Vision*, 45(2):83–105, 2001.

[17] Timor Kadir, Andrew Zisserman, and Michael Brady. An affine invariant salient region detector. In *Proceedings of 8th European Conference on Computer Vision*, pages 228–241, Pague, Chech Republic, May 2004.

[18] Kurt Konolige, Motilal Agrawal, and Joan Solà. Large-scale visual odometry for rough terrain. In *International Symposium of Robotics Research (ISRR)*, Hiroshima, Japan, 2007.

[19] Didier Le Gall. Mpeg: a video compression standard for multimedia applications. *Commun. ACM*, 34(4):46–58, April 1991.

[20] Anat Levin and Richard Szeliski. Visual odometry and map correlation. In *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 611–618, June 2004.

[21] Hongdong Li and Richard Hartley. Five-point motion estimation made easy. In *Proceedings of the 18th International Conference on Pattern Recognition*, pages 630–633, 2006.

[22] David Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[23] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI81*, pages 674–679, 1981.

[24] Mark Maimone, Yang Cheng, and Larry Matthies. Two years of visual odometry on the mars exploration rovers. *Journal of Field Robotics*, 24(3):169–186, March 2007.

[25] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22:761–767, 2004.

[26] Y. Matsushita, E. Ofek, Weina Ge, Xiaoou Tang, and Heung-Yeung Shum. Full-frame video stabilization with motion inpainting. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(7):1150–1163, 2006.

[27] Krystian Mikolajczyk and Cordelia Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[28] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3D reconstruction. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006.

[29] JB Nestleroth and TA Bubenik. Magnetic flux leakage (MFL) technology for natural gas pipeline inspection. Technical report, Battelle, Report Number GRI-00/0180 to the Gas Research Institute, 1999.

[30] Jan Neumann, Cornelia Fermüller, and Yiannis Aloimonos. Eyes form eyes: New cameras for structure from motion. In *Proceedings Workshop on Omnidirectional Vision (OMNIVIS)*, 2002.

[31] David Nistér. Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors. In *Proceedings European Conference on Computer Vision (ECCV 2000)*, pages 649–663, 2000.

[32] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004.

[33] David Nistér, Oleg Naroditsky, and James Bergend. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23(1):3–20, January 2006.

[34] Davide Scaramuzza and Roland Siegwart. Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles. *IEEE Transactions on Robotics*, 24(5), October 2008.

[35] Hagen Schempf. Visual and nde inspection of live gas mains using a robotic explorer. *International Journal of Field Robotics*, Winter, 2009.

[36] Cordelia Schmid, Roger Mohr, and Christian Bauckage. Evaluation of interest point detectors. *International Journal of Computer Vision*, 37(2):151–172, 2000.

[37] Cordelia Schmid, Roger Mohr, and Christial Bauckhage. Comparing and evaluating interest points. In *International Conference on Computer Vision*, pages 230–235, 1998.

[38] Richard Szeliski. Image alignment and stitching: a tutorial. *Found. Trends. Comput. Graph. Vis.*, 2(1):1–104, 2006.

[39] Jean-Philippe Tardif, Yanis Pavlidis, and Kostas Daniilidis. Monocular visual odometry in urban environments using an omnidirectional camera. In *IEEE/RSJ International Conference in Intelligent Robots and Systems*, pages 2531–2538, 2008.

[40] Qi Tian and Michael N Huhns. Algorithms for subpixel registration. *Comput. Vision Graph. Image Process.*, 35:220–233, August 1986.

[41] P. Tissainayagam and D. Suter. Assessing the performance of corner detectors for point feature tracking applications. *Image and Vision Computing*, 22:663–679, 2004.

[42] B. Zitova and J. Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21(11):977–1000, October 2003.